การสร้างรายการเพลงโดยใช้การกรองร่วมแบบเซสชั่นที่เพิ่มขึ้นด้วยกลไกการลืม และการวิเคราะห์สถิติเชิงมุม

สุเมธ ดาราพิสุทธิ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์
คณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา
กรกฎาคม พ.ศ. 2559

SONG LIST GENERATING USING INCREMENTAL SESSION BASED COLLABORATIVE FILTERING WITH FORGETTING MECHANISM AND CIRCULAR STATISTICS

SUMET DARAPISUT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT
FOR THE MASTER DEGREE OF SCIENCE IN COMPUTER SCIENCE
FACULTY OF INFORMATICS BURAPHA UNIVERSITY
2016.

คณะกรรมการควบคุมวิทยานิพนธ์และคณะกรรมการสอบวิทยานิพนธ์ได้พิจารณาวิทยานิพนธ์ของ สุเมธ ดาราพิสุทธิ์ ฉบับนี้แล้ว เห็นสมควรรับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ ของมหาวิทยาลัยบูรพาได้

คณะกรรมการควบคุมวิทยานิพนธ์

ผู้ช่วยศาสตราจารย์ ดร. จักริน สุขสวัสดิ์ชน อาจารย์ที่ปรึกษา

ผู้ช่วยศาสตราจารย์ ดร. อุรีรัฐ สุขสวัสดิ์ชน อาจารย์ที่ปรึกษาร่วม

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(ศาสตราจารย์ ดร. ชิดชนก เหลือสินทรัพย์)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. จักริน สุขสวัสดิ์ชน)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. อุรีรัฐ สุขสวัสดิ์ชน)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. โกเมศ อัมพวัน)

คณะวิทยาการสารสนเทศ อนุมัติให้รับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ ของมหาวิทยาลัยบูรพา

..... คณบดีคณะวิทยาการสารสนเทศ
(ผู้ช่วยศาสตราจารย์ ดร. กฤษณะ ชินสาร)
วันที่ 9 เดือน กรกฎาคม พ.ศ. 2559

ประกาศคุณูปการ

วิทยานิพนธ์ฉบับนี้เสร็จสมบูรณ์ได้ด้วยดีด้วยการสนับสนุนจากคณาจารย์ คณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพาทุกท่านที่ช่วยให้ความรู้แก่ผู้วิจัย โดยเฉพาะอย่างยิ่ง ผู้ช่วยศาสตราจารย์ ดร.จักริน สุขสวัสดิ์ชน อาจารย์ที่ปรึกษาหลัก และ ผู้ช่วยศาสตราจารย์ ดร.อุรีรัฐ สุขสวัสดิ์ชน อาจารย์ที่ปรึกษาร่วมที่ช่วยให้คำปรึกษาและแนวทางในการดำเนินงานวิทยานิพนธ์เล่มนี้อีกทั้งยังช่วยแก้ไขข้อบกพร่องและแนะนำในงานเขียนวิทยานิพนธ์ด้วยดีเสมอมา และขอขอบคุณ อาจารย์ เหมรัศมิ์ วชิรหัตถพงศ์ สำหรับความอนุเคราะห์ช่วยเหลือในเรื่องเซิฟเวอร์และเครื่องมือต่าง ๆ ในการดำเนินงานวิทยานิพนธ์

ขอขอบคุณ คุณพ่อ คุณแม่ พี่ๆน้องๆ และเพื่อนๆ ที่ช่วยเหลือในการให้คำแนะนำและให้กำลังใจในการเรียนและการดำเนินงานวิทยานิพนธ์ตลอดสามปีที่ผ่านมาทำให้ผู้วิจัยประสบความสำเร็จในการศึกษาด้วยดี

วิทยานิพนธ์นี้ได้รับทุนอุดหนุนวิทยานิพนธ์จากมหาวิทยาลัยบูรพา ปีงบประมาณ ๒๕๕๘

สุเมธ ดาราพิสุทธิ์

56910117: สาขาวิชา: วิทยาการคอมพิวเตอร์; วท.ม. (วิทยาการคอมพิวเตอร์)
คำสำคัญ: วิธีการแนะนำเพลง/การกรองข้อมูลร่วม/เซสชั่น/ช่วงเวลาการฟังเพลง/ระบบแนะนำออนไลน์

สุเมธ ดาราพิสุทธิ์: การสร้างรายการเพลงโดยใช้การกรองร่วมแบบเซสชั่นที่เพิ่มขึ้นด้วยกลไกการลืมและการวิเคราะห์สถิติเชิงมุม คณะกรรมการควบคุมวิทยานิพนธ์: จักริน สุขสวัสดิ์ชน, Ph.D., อุรีรัฐ สุขสวัสดิ์ชน, Ph.D., 109 หน้า. ปี พ.ศ. 2559

งานวิจัยส่วนมากในระบบแนะนำเพลงใช้วิธีการกรองข้อมูลร่วมสำหรับสร้างรายการเพลงแนะนำเฉพาะบุคคล โดยพิจารณาจากประวัติการให้คะแนนกับเพลงหรือประวัติการฟังเพลงในอดีต ซึ่งไม่ได้พิจารณาพฤติกรรมในการฟังเพลงและความชอบการฟังเพลงในปัจจุบันของผู้ฟังอย่างแท้จริง เนื่องจากพฤติกรรมการฟังเพลงมักจะมีการฟังที่ต่อเนื่องและมีการฟังซ้ำ โดยเฉพาะการฟังเพลงล่าสุดของผู้ฟังแสดงถึงความชอบของเพลงในปัจจุบัน นอกจากนี้การสร้างรายการเพลงแนะนำด้วยวิธีการกรองร่วมแบบเดิมยังประสบปัญหาการประมวลผลข้อมูลที่มีปริมาณมากเพราะใช้ประวัติการฟังเพลงทั้งหมดในการสร้างรายการเพลงแนะนำ ส่งผลให้ใช้เวลาในการคำนวณนานและพื้นที่ในการประมวลผลที่มากซึ่งเพิ่มขึ้นตามจำนวนเพลงและผู้ฟังในระบบ

งานวิทยานิพนธ์นี้ได้นำเสนอวิธีการสร้างรายการเพลงแนะนำอัตโนมัติที่เรียกว่า “การกรองร่วมแบบเซสชั่นที่เพิ่มขึ้นด้วยกลไกการลืมและการวิเคราะห์สถิติเชิงมุม” หรือ “ไอเอสเอสซีเอฟ” โดยปรับปรับปรุงจากวิธีการสร้างรายการเพลงแนะนำที่ใช้การกรองร่วมแบบเซสชั่น (เอสเอสซีเอฟ) ซึ่งวิธีการสร้างรายการเพลงที่นำเสนอในงานวิทยานิพนธ์นี้ใช้การสร้างรายการเพลงจาก 2 วิธีการร่วมกัน วิธีที่ 1 การสร้างรายการเพลงจะพิจารณาการฟังเพลงในเซสชั่นปัจจุบันที่คล้ายกับเซสชั่นในอดีตของผู้ฟัง เนื่องจากผู้ฟังมักจะมีการฟังเพลงที่ต่อเนื่องและฟังซ้ำในเพลงที่ชอบร่วมกับการใช้วิธีกลไกการลืมประกอบด้วยวิธีการหน้าต่างแบบเลื่อน และตัวแปรลดเลือน เพื่อลดเวลาการคำนวณและลดหน่วยความจำที่ใช้ในการประมวลผล ส่วนวิธีที่ 2 สร้างรายการเพลงแนะนำโดยพิจารณาช่วงเวลาเฉพาะในการฟังเพลงซึ่งแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติในรอบวันของผู้ฟังโดยใช้การวิเคราะห์สถิติเชิงมุม จากผลการทดลองในฐานข้อมูลเพลง 30Music และ Last.fm แสดงให้เห็นถึงวิธีการที่นำเสนอไอเอสเอสซีเอฟ สามารถให้ความถูกต้องดีกว่าวิธีการเอสเอสซีเอฟ โดยใช้วิธีการวัดประสิทธิภาพ HitRatio และ Precision นอกจากนี้วิธีการไอเอสเอสซีเอฟสามารถใช้เวลาในการคำนวณที่มีประสิทธิภาพซึ่งมีความเหมาะสมในการแนะนำเพลงแบบออนไลน์

56910117: MAJOR: COMPUTER SCIENCE; M.Sc. (COMPUTER SCIENCE)
KEYWORD: RECOMMENDATION APPROACH/COLLOBORATIVE FILTERING/ SESSION/TIME INTERVAL/ONLINE RECOMMENDER SYSTEMS

SUMET DARAPISUT: SONG LIST GENERATING USING INCREMENTAL SESSION BASED COLLABORATIVE FILTERING WITH FORGETTING MECHANISM AND CIRCULAR STATISTICS THESIS ADVISSOR: JAKKARIN SUKSAWATCHON, Ph.D., UREERAT SUKSAWATCHON, Ph.D, 109 P. 2016.

Most of research papers in music recommendation systems use Collaborative Filtering (CF) for generating personalized recommendations based on user's previous song ratings or static usage history data. But those researches adapting CF do not consider behavior of listening to songs and are not able to maintain the systems to sensitive to recent user's preferences. Behavior of music listening is continuous and repetitive process, especially, the latest song listening can infer to the favorite song at that moment. Besides, using traditional CF, it faces with the scalability problem. It uses the fully static listening history of users to perform recommendation and requires very expensive computational time and space with the growth of the number of users and music in a database.

In order to overcome this problem, we propose an algorithm for automatically generating song list by using Incremental Session based Collaborative Filtering with forgetting mechanism and Circular Statistics or ISSCF by adapting Session-based Collaborative Filtering (SSCF). Our proposed method generates recommended list from the two approaches. The first approach considers listening preferred song in the active session for creating recommended list that similar with the past sessions of a user since the user always listens song repetitively and continuously. In order to avoid unnecessary memory usage and processing time, we use forgetting mechanism: sliding windows and fading factors incorporating with SSCF. The second approach generates music recommended list that regards time interval of music listening by using circular statistics analyzed. This approach takes into account to be specific in the repetition of listening music in daily that difference from other time intervals statistics significantly. From experimental results in the 30 Music and Last.fm music dataset that ISSCF outperforms SSCF in term of accuracy by using the HitRatio and the precision measuring. Furthermore, the proposed method an efficient computational time that suitable for online music recommended.

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญภาพ

สารบัญภาพ (ต่อ)

สารบัญภาพ (ต่อ)

สารบัญภาพ (ต่อ)

บทที่1

บทนำ

ที่มาและความสำคัญปัญหาวิทยานิพนธ์

ระบบแนะนำเพลง (Music Recommender System) เข้ามามีบทบาทสำคัญอย่างมากในวงการเพลงดิจิตอลที่ปัจจุบันกำลังเติบโตขึ้นอย่างรวดเร็วทั้งในด้าน 1) ผู้ให้บริการการฟังเพลงบนอินเทอร์เน็ต อาทิเช่น Spotify[1], Last.fm[2] และ Pandora[3] เป็นต้น 2) จำนวนเพลงดิจิตอล และ 3) ผู้ใช้บริการหรือผู้ฟัง ระบบแนะนำเพลงจะช่วยกรองและคัดเลือกเพลงที่มีอยู่มากมายมหาศาลในระบบให้กับผู้ฟังอย่างอัตโนมัติ เพื่อให้ผู้ฟังเข้าถึงและได้ฟังเพลงที่ตนเองชื่นชอบได้ง่ายขึ้น ซึ่งเป็นการเพิ่มผลกำไรและกลุ่มผู้ฟังในตลาดการค้าเพลงอิเล็กทรอนิกส์ให้กับผู้ให้บริการอีกด้วย

วิธีการที่ใช้สร้างระบบแนะนำเพลงมีด้วยกันหลายวิธี แต่วิธีการที่เป็นที่นิยม ได้แก่ วิธีการกรองข้อมูลร่วม (Collaborative filtering) โดยวิธีการนี้จะพิจารณาจากพฤติกรรมของผู้ฟังเพลงที่เกิดขึ้นในอดีต เช่น การให้คะแนนความชอบ, การคลิก, ประวัติการเลือกซื้อเพลง, ประวัติการฟังเพลง เป็นต้น โดยนำข้อมูลพฤติกรรมเหล่านี้ร่วมกับข้อมูลพฤติกรรมของคนฟังคนอื่นที่มีลักษณะความคล้ายคลึงกันมาคำนวณค่าคะแนนความชอบในแต่ละเพลง เพื่อนำไปสร้างเป็นรายการเพลงที่จะแนะนำให้กับผู้ฟังแต่ละคน

วิธีการสร้างรายการเพลงแนะนำให้กับผู้ฟัง ส่วนมากออกแบบและพัฒนาขั้นตอนวิธีต่าง ๆ โดยใช้ข้อมูลการฟังเพลงของผู้ฟังที่เคยเกิดขึ้นในอดีตทั้งหมด (Baltrunas, L., &Amatriain, X., 2009) (Park, S. E., et al, 2011) (Dias, R. & Fonseca, M.J., 2013) ซึ่งหากระบบแนะนำมีเพลงใหม่เข้ามาเก็บในระบบ ก็จะต้องนำข้อมูลการฟังของผู้ฟังคนนั้นในอดีตทั้งหมดมาคำนวณและวิเคราะห์เพื่อสร้างเป็นรายการเพลงที่จะแนะนำ ลักษณะการทำงานเช่นนี้เรียกว่า **การสร้างรายการเพลงแบบออฟไลน์** ซึ่งเป็นการสร้างรายการเพลงที่ให้ความสำคัญของข้อมูลการฟังเพลงเท่า ๆ กัน ส่งผลให้เพลงเก่าที่ไม่ได้ฟังแล้วหรือไม่ได้รับความนิยมอาจถูกแนะนำให้ผู้ฟังอีกครั้ง แต่พฤติกรรมหรือลักษณะการฟังเพลงจะแตกต่างจากพฤติกรรมการอ่านหนังสือ การฟังเพลงมักจะปรับเปลี่ยนไปตามบริบท หรือเปลี่ยนไปตามช่วงเวลา หรือความชอบในการฟังเพลงในขณะนั้นของผู้ฟังแต่ละคน ซึ่งสอดคล้องกับงานวิจัยของ (Baltrunas, L., & Amatriain, X., 2009) (Cebrián, T. et al., 2010) (Park, C. H. & Kahng, M., 2010) ที่ศึกษาพฤติกรรมการฟังเพลง พบว่าผู้ฟังมีการฟังเพลงหรือชอบแนวเพลงที่เปลี่ยนไปตามเวลา โดยมักจะฟังเพลงซ้ำหรือฟังบ่อย ๆ ในเพลงที่ชอบเฉพาะช่วงเวลาและจะทำพฤติกรรมเช่นนี้ติดต่อกันหลายวันหรือหลายสัปดาห์ และยังพบว่าส่วนมากพฤติกรรมการฟังเพลงมักจะฟังเพลงที่ต่อเนื่องเป็นช่วงหรือเซสชั่น (Session) งานวิจัยของ Sung Eun Park และคณะ (2011) ได้เสนอขั้นตอนวิธีที่ชื่อ “Session-based Collaborative Filtering (SSCF)”

---

[1] http://www.spotify.com/

[2] http://www.last.fm/

[3] http://www.pandora.com/

(Park, S. E., et al, 2011) ซึ่งเป็นขั้นตอนวิธีคัดเลือกและสร้างรายการเพลงให้กับผู้ฟังที่กำลังฟังเพลง โดยพิจารณาคัดเลือกเพลงจากเซสชั่นการฟังเพลงในอดีตที่มีลักษณะคล้ายกับการฟังเพลงจากเซสชั่นปัจจุบัน ต่อมา Dias, R. และ Fonseca, M.J. (2013) ได้เสนอขั้นตอนวิธีที่ชื่อ “Temporal Session-based Collaborative Filtering (TSSCF)” (Dias, R. & Fonseca, M. J., 2013) ซึ่งได้ปรับปรุงขั้นตอนวิธีการของ SSCF โดยพิจารณาช่วงเวลาฟังเพลงของผู้ฟังตามช่วงวัน, สัปดาห์ และเดือนในรอบปี และสร้างรายการเพลงโดยใช้วิธีการของ SSCF ที่สอดคล้องกับช่วงเวลา จากผลการทดลองแสดงให้เห็นว่าวิธีการ TSSCF มีความแม่นยำมากกว่าวิธี SSCF แต่อย่างไรก็ตามการสร้างรายการเพลงทั้งวิธีการ SSCF และ TSSCF ยังเป็นการสร้างรายการเพลงแบบออฟไลน์ที่พิจารณาข้อมูลในอดีตทั้งหมดที่ให้ความสำคัญของเซสชั่นเก่าและใหม่เท่ากัน แต่เนื่องจากตลาดเพลงออนไลน์มีเพลงใหม่เพิ่มขึ้นตลอดเวลาจึงทำให้วิธีการ SSCF และ TSSCF เกิด**ปัญหาเกี่ยวกับการประมวลผลข้อมูลที่มีปริมาณมาก** ซึ่งอาจทำให้วิธีการสร้างรายการแบบออฟไลน์มีประสิทธิภาพในการแนะนำลดลง การใช้เวลาในการคำนวณและวิเคราะห์ที่นาน และ ยังใช้พื้นที่ของหน่วยความจำที่มาก เนื่องจากปริมาณข้อมูลเพลงในระบบมีมากมายมหาศาลเกิดขึ้นตลอดเวลา และที่สำคัญข้อมูลเซสชั่นการฟังเพลงของผู้ฟังทั้งหมดจะต้องถูกนำเข้ามาเก็บไว้ในหน่วยความจำตลอดเวลา

ดังนั้นวิทยานิพนธ์นี้จึงได้นำเสนอวิธีการสร้างรายการเพลงแบบใหม่ที่ชื่อ “**Incremental Session-based Collaborative Filtering using Forgetting mechanism and Circular statistics (ISSCF)**” โดยปรับปรุงขั้นตอนวิธี SSCF เดิมให้สามารถสร้างรายการเพลงที่จะแนะนำแบบออนไลน์ เพื่อแก้ปัญหาที่เกิดจากวิธีการสร้างรายการเพลงแบบออฟไลน์ และปัญหาเกี่ยวกับการประมวลผลข้อมูลที่มีปริมาณมาก ซึ่งจะเหมาะกับระบบแนะนำเพลงที่ให้บริการเพลงแบบออนไลน์ โดยวิธีการ ISSCF ที่นำเสนอนี้จะนำข้อมูลการฟังเพลงของผู้ฟังแต่ละคนแบ่งเป็นเซสชั่นที่เรียกกว่า session profile จากนั้นนำ session profile เฉพาะบุคคลไปสร้างเป็นรายการเพลงที่จะแนะนำสำหรับผู้ฟังคนนั้น โดยวิธีการสร้างรายการเพลงของวิธี ISSCF มีด้วยกัน 2 วิธีคือ การสร้างรายการเพลงที่พิจารณาจากความชอบของผู้ฟังจากเซสชั่นในขณะที่กำลังฟังเพลง กับการสร้างรายการเพลงที่พิจารณาจากความชอบของผู้ฟังเฉพาะช่วงเวลาว่ามักจะชอบฟังเพลงช่วงเวลาใดและชอบฟังเพลงอะไรโดยใช้การวิเคราะห์สถิติเชิงมุม (Circular Statistics) และเมื่อผู้ฟังหยุดฟังเพลงเมื่อไรก็จะเกิดเซสชั่นการฟังเพลงเกิดขึ้นใหม่ซึ่งถือว่าเป็นข้อมูลย้อนกลับ (Feed back) ที่จะถูกนำไปปรับปรุง session profile ของผู้ฟัง โดยการปรับปรุงจะใช้วิธีการ Sliding windows เพื่อที่จะลดความสำคัญของเซสชั่นการฟังเพลงในอดีต ซึ่งจะช่วยให้การสร้างรายการเพลงที่จะแนะนำมีความแม่นยำได้เพลงที่ตรงกับความต้องการของผู้ฟังมากยิ่งขึ้น และยังช่วยลดเวลาในการคำนวณและลดพื้นที่ของหน่วยความจำ เนื่องจากข้อมูลการฟังเพลงไม่จำเป็นต้องนำมาคำนวณและเก็บไว้ในหน่วยความจำทั้งหมด จากผลการทดลองในวิทยานิพนธ์นี้แสดงให้เห็นถึงวิธีการ ISSCF สามารถสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังแต่ละคนได้อย่างถูกต้องและใช้เวลาในการคำนวณที่รวดเร็วในขั้นตอนการคำนวณความคล้ายของเซสชั่นและขั้นตอนการทำนายค่าคะแนนความชอบกว่าวิธีการ SSCF แบบเดิม

วัตถุประสงค์ของวิทยานิพนธ์

1. เพื่อศึกษาและวิเคราะห์ปัญหาที่เกิดขึ้นในการสร้างรายการเพลงแนะนำแบบออฟไลน์

2. เพื่อปรับปรุงและพัฒนาขั้นตอนวิธีการสร้างรายเพลงแนะนำให้มีประสิทธิภาพเพิ่มขึ้นทั้งทางด้านความเร็วและความถูกต้องในการสร้างรายการเพลง

3. เพื่อปรับปรุงและพัฒนาขั้นตอนวิธีการแนะนำเพลงให้เหมาะสมกับพฤติกรรมการฟังเพลงของผู้ฟัง

4. เพื่อปรับปรุงและพัฒนาขั้นตอนวิธีการสร้างรายเพลงแนะนำให้มีประสิทธิภาพในการประมวลผลข้อมูลที่มีปริมาณมาก

ขอบเขตของวิทยานิพนธ์

1. วิทยานิพนธ์นี้ได้ดำเนินการทดลองในฐานข้อมูลเพลง 2 ฐานข้อมูล ดังนี้

1.1 ฐานข้อมูลเพลง Last.fm[4] จัดเก็บประวัติการฟังเพลงของผู้ฟัง ประกอบด้วย ผู้ฟัง, เวลาในการฟังเพลง, รหัสศิลปิน, ชื่อศิลปิน, รหัสเพลงและชื่อเพลง ตั้งแต่ปี 2005 ถึงปี 2009 โดยพิจารณาผู้ฟังที่มีการฟังเพลงมากกว่า 400 เซสชั่น ประกอบด้วย ผู้ฟังจำนวน 300 คน, จำนวนเพลง 373,946 เพลงและจำนวนครั้งของการฟังเพลงรวมทุกผู้ฟัง 3,130,850 ครั้ง

1.2 ฐานข้อมูลเพลง 30Music (Turrin, R. et al., 2015) จัดเก็บประวัติการฟังเพลงของผู้ฟังจากเว็บไซต์ Last.fm ระหว่างปี 2014 ถึงปี 2015 โดยพิจารณาผู้ฟังที่มีการฟังเพลงมากกว่า 400 เซสชั่น ประกอบด้วย ผู้ฟังจำนวน 279 คน, จำนวนเพลง 517,451 เพลงและจำนวนครั้งของการฟังเพลงรวมทุกผู้ฟังที่ 1,839,928 ครั้ง

2. วิธีการวัดประสิทธิภาพวิธีการแนะนำเพลงคือ HitRatio (HR@n) ซึ่งเป็นการวัดความถูกต้องของรายการเพลงที่แนะนำกับเพลงที่ผู้ฟังเลือกฟังในเพลงถัดไป และวิธีการ Precision สำหรับการวัดความถูกต้องของรายการเพลงแนะนำกับรายการเพลงแนะนำทั้งหมด

แนวทางในการพัฒนาวิทยานิพนธ์

1. ศึกษาและรวบรวมข้อมูล

- ศึกษาขั้นตอนวิธีการพื้นฐานและทฤษฎีที่ใช้ในการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง
- ศึกษางานวิจัยที่เกี่ยวข้องกับการสร้างรายการเพลงที่จะแนะนำโดยพิจารณาช่วงเวลาของผู้ฟังเพลงและขั้นตอนวิธีการแบบออนไลน์เป็นหลัก
- ศึกษาและรวบรวมข้อมูลจากฐานข้อมูลเพลง Last.fm และ 30Music

[4] http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/

2. การตั้งสมมติฐานวิทยานิพนธ์

วิทยานิพนธ์นี้ได้ศึกษาและพัฒนาขั้นตอนวิธีการสร้างรายการเพลงที่จะแนะนำสำหรับผู้ฟังแต่ละคน โดยมีสมมติฐานดังต่อไปนี้

สมมติฐานที่ 1 : เพลงที่ถูกเลือกฟังล่าสุดน่าจะแสดงถึงความชอบของผู้ฟังเพลง

สมมติฐานที่ 2 : ผู้ฟังน่าจะมีการเลือกฟังเพลงหรือมีลำดับในการฟังเพลงที่คล้ายกับเซสชั่นในอดีต

สมมติฐานที่ 3 : ผู้ฟังน่าจะมีการเลือกฟังเพลงที่ชอบซ้ำในบางช่วงเวลา

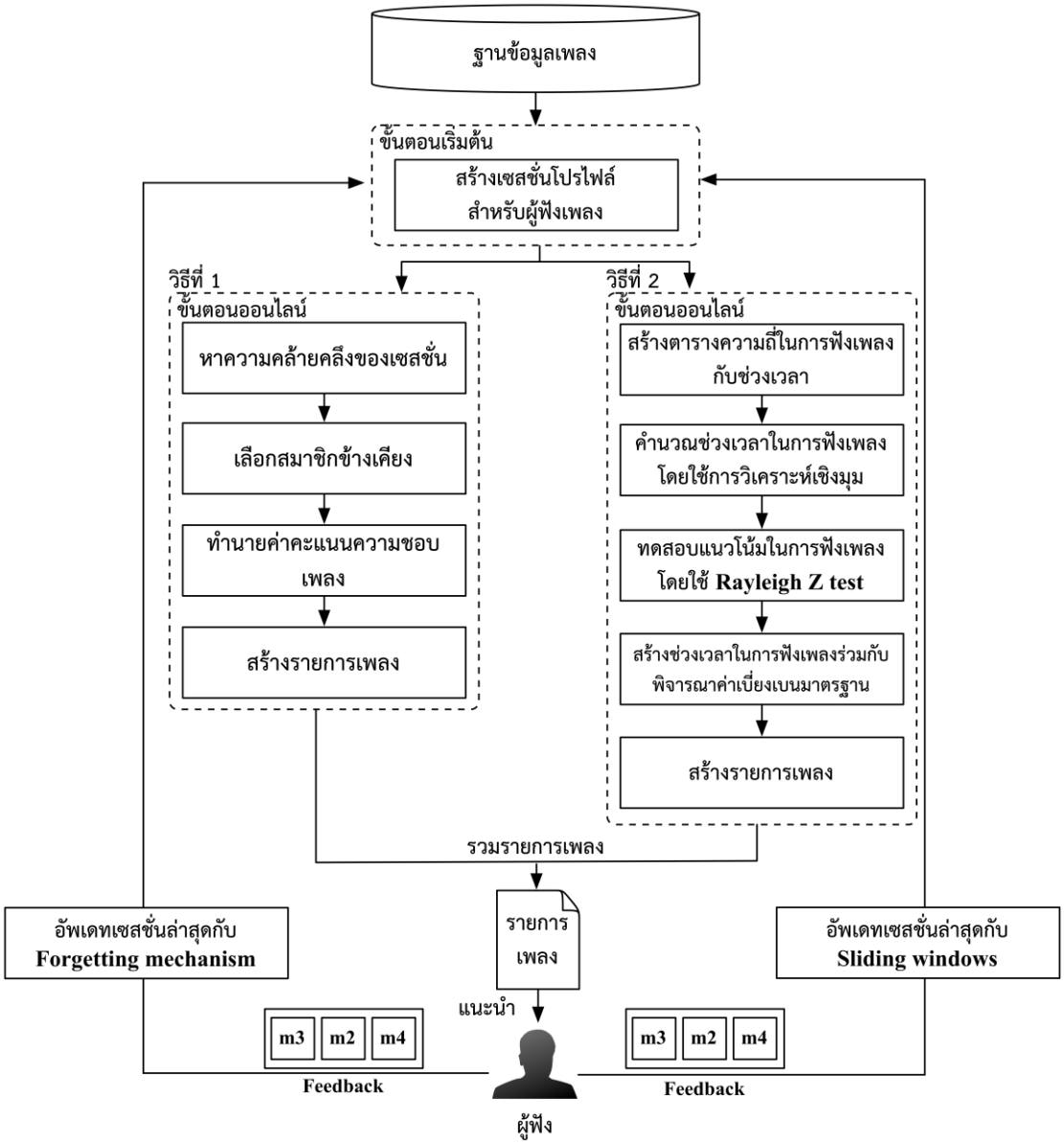
3. การแก้ปัญหาวิทยานิพนธ์

วิทยานิพนธ์นี้ได้นำเสนอวิธีการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังแต่ละคน เพื่อเพิ่มประสิทธิภาพความถูกต้องและการจัดการกับข้อมูลที่มีขนาดใหญ่ ซึ่งมีความเหมาะสมสำหรับระบบแนะนำเพลงแบบออนไลน์โดยนำเสนอวิธีการที่เรียกว่า **Incremental Session Based Collaborative Filtering using Forgetting mechanism and Circular statistics (ISSCF)** ซึ่งเป็นการรวมวิธีการสร้างรายการเพลงแนะนำจาก 2 ขั้นตอน

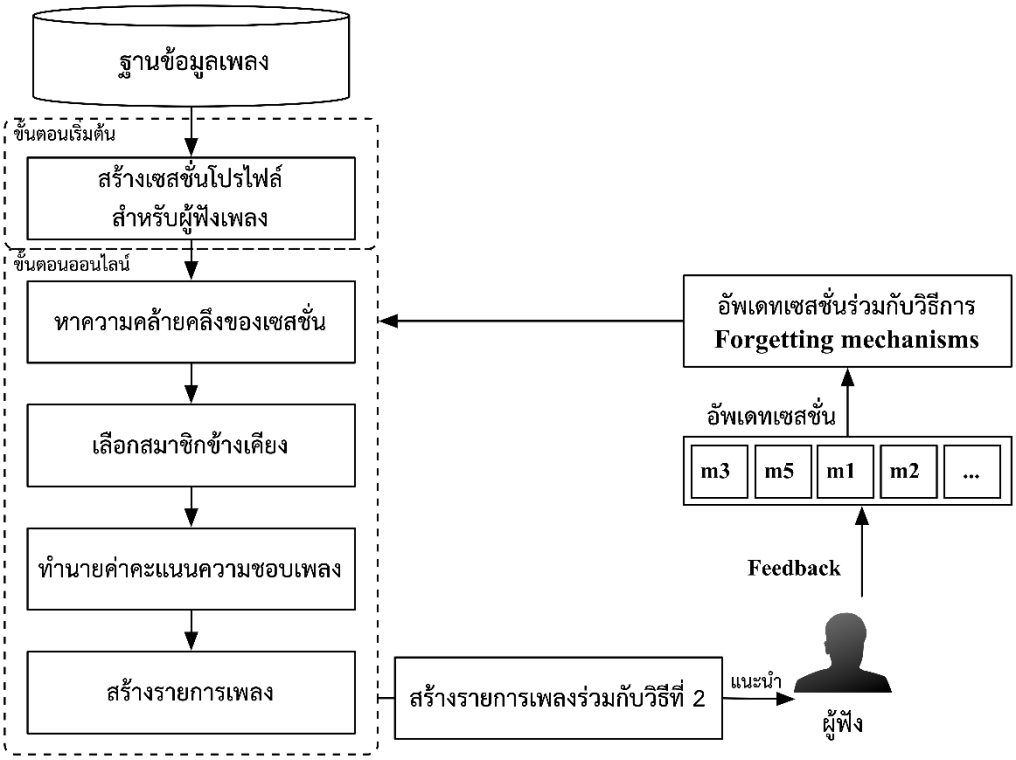
ขั้นตอนวิธีที่ 1 สร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังซึ่งคาดว่าผู้ฟังอาจจะชอบในการเลือกฟังเพลงถัดไป โดยการพิจารณาจากลักษณะการเลือกฟังเพลงในปัจจุบันของผู้ฟังแต่ละคนที่มีความคล้ายกับเซสชั่นโปรไฟล์ในอดีต เนื่องจากผู้ฟังมักจะมีการฟังซ้ำในเพลงเดิม

ขั้นตอนวิธีที่ 2 สร้างรายการเพลงที่ผู้ฟังชื่นชอบให้กับผู้ฟังในช่วงเวลาเฉพาะ โดยการวิเคราะห์สถิติเชิงมุมเป็นการพิจารณาช่วงเวลาในการฟังเพลงกับเพลงที่ผู้ฟังชื่นชอบ ซึ่งผู้ฟังมักจะมีการฟังซ้ำในทุกวันหรือมีการฟังที่ต่อเนื่องในช่วงวันหรือสัปดาห์ซึ่งมีความแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติ

จากทั้ง 2 ขั้นตอนวิธีการ ISSCF ได้อัพเดทเซสชั่นของการฟังเพลงโดยใช้วิธีการ Forgetting mechanism และ Sliding windows ซึ่งให้ประสิทธิภาพในการคำนวณที่รวดเร็วและยังคงความชอบในปัจจุบันของผู้ฟังเพลง แสดงรายละเอียดภาพรวมวิธีการทำงาน ISSCF ดังภาพที่ 1-1

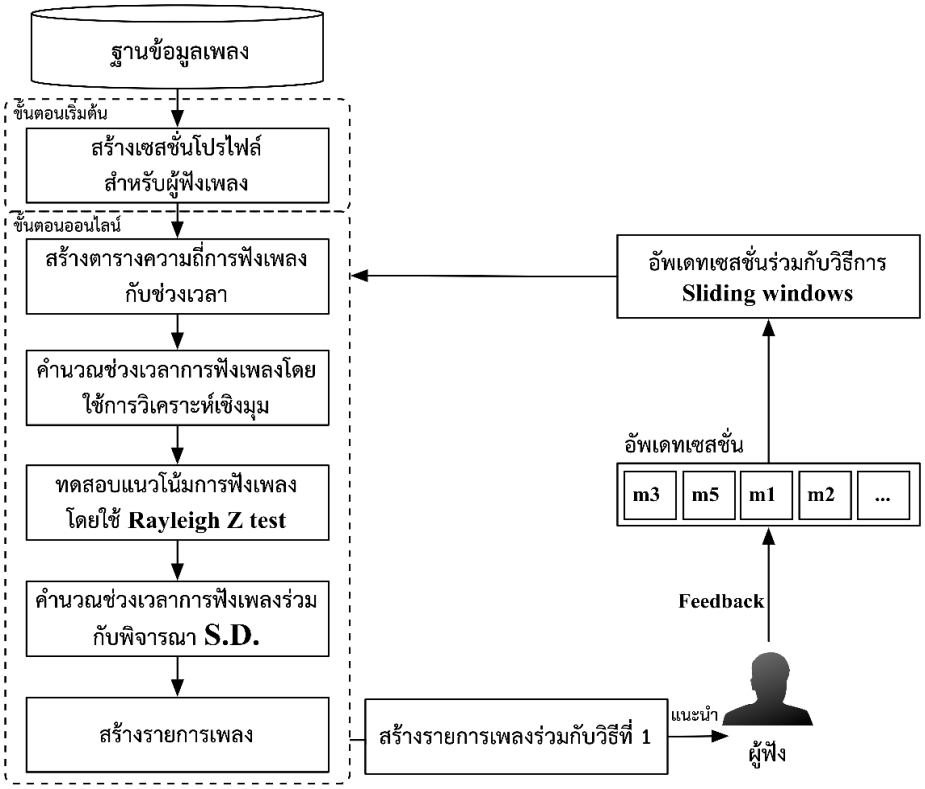


ภาพที่ 1-1 รายละเอียดภาพรวมวิธีการทำงาน ISSCF



ภาพที่ 1-2 รายละเอียดขั้นตอนการสร้างรายการเพลงแนะนำของวิธีที่ 1

รายละเอียดของขั้นตอนวิธีที่ 1 วิธีการนี้ได้ประยุกต์ใช้วิธีการกรองข้อมูลร่วมแบบเดิม โดยแทนที่การพิจารณาความคล้ายจากข้อมูลผู้ฟังคนอื่น ในการสร้างรายการเพลงแนะนำด้วยการหาความคล้ายของเซสชั่นในอดีตกับเซสชั่นปัจจุบันของผู้ฟังแล้วทำนายค่าคะแนนความชอบของแต่ละเพลงเพื่อสร้างรายการเพลงที่จะแนะนำ เนื่องจากผู้ฟังมักจะมีการฟังซ้ำในเพลงหรือแนวเพลงที่ชื่นชอบซึ่งมีความต่อเนื่องกันในลักษณะของเซสชั่น เมื่อผู้ฟังหยุดฟังเพลงในช่วงระยะเวลาหนึ่ง เซสชั่นในปัจจุบันของผู้ฟังจะถูกอัพเดทเพื่อปรับปรุงเซสชั่นโปรไฟล์ของแต่ละผู้ฟังด้วย 2 วิธีการคือ 1) Sliding windows และ 2) Forgetting mechanisms ประกอบด้วยวิธีการ Sliding windows ในการตัดเซสชั่นในอดีตของผู้ฟังร่วมกับวิธีการ Fading factor ในการให้ความสำคัญกับเซสชั่นล่าสุดของผู้ฟังเพลง รายละเอียดขั้นตอนการสร้างรายการเพลงแนะนำของวิธีที่ 1 ดังภาพที่ 1-2



ภาพที่ 1-3 รายละเอียดขั้นตอนการสร้างรายการเพลงแนะนำของวิธีที่ 2

รายละเอียดของขั้นตอนวิธีที่ 2 วิธีการนี้ได้สร้างรายการเพลงที่จะแนะนำในช่วงเวลาเฉพาะ โดยการวิเคราะห์สถิติเชิงมุมเพื่อพิจารณาช่วงเวลาของการฟังเพลงจากความชอบของผู้ฟังเฉพาะช่วงเวลา ว่ามักจะชอบฟังเพลงอะไรในช่วงเวลานั้น จากผลลัพธ์ที่ได้จากการวิเคราะห์สถิติเชิงมุมคือชั่วโมงในการฟังเพลงนั้นเฉพาะผู้ฟังที่มีการฟังแตกต่างจากช่วงเวลาอื่น โดยใช้การทดสอบนัยสำคัญทางสถิติคือ Rayleigh Z-test แล้วสร้างรายการเพลงเมื่อผู้ฟังเข้าสู่ระบบในช่วงเวลาที่ตรงกับช่วงเวลาที่ได้จากการวิเคราะห์นี้ เมื่อผู้ฟังหยุดฟังเพลงในช่วงเวลาหนึ่งเซสชั่นปัจจุบันของผู้ฟังจะถูกอัพเดทเข้าสู่ระบบเพื่อพิจารณาความถี่ในการฟังเพลงกับช่วงเวลาโดยใช้วิธีการ Sliding windows ซึ่งยังคงความชอบในปัจจุบันของผู้ฟังเพลง รายละเอียดขั้นตอนการสร้างรายการเพลงแนะนำของวิธีที่ 2 ดังภาพที่ 1-3

4. การทดลองและประเมินผล

การทดลองและการประเมินผลในวิทยานิพนธ์นี้ได้เปรียบเทียบวิธีการ ISSCF กับวิธีการ SSCF สำหรับการออกแบบการทดลองได้จำลองการฟังเพลงแบบออนไลน์ของผู้ฟังโดยนำข้อมูลเพลงของแต่ละผู้ฟังเข้ามาเป็นข้อมูลทดสอบแบบเป็นลำดับ ซึ่งแบ่งเป็นขั้นตอนเริ่มต้น (ขั้นตอนแบบออฟไลน์) ในการจัดเก็บเซสชั่นโปรไฟล์เริ่มต้นของแต่ละผู้ฟังแล้วขั้นตอนออนไลน์จะพิจารณาแต่ละเพลงที่ถูกฟังอย่างเป็นลำดับ เมื่อผู้ฟังมีการฟังเพลงเสร็จสิ้น เพลงนั้นจะถูกเป็นข้อมูลย้อนกลับซึ่งรวมเป็นข้อมูลที่ใช้ในการสร้างรายการเพลงที่จะแนะนำและเพลงถัดไปจะถูกพิจารณาเป็นข้อมูลสำหรับ

ทดสอบ โดยวิทยานิพนธ์นี้ได้ใช้วิธีการวัดประสิทธิภาพความถูกต้องคือ HitRatio และวิธีการ Precision รวมทั้งทดสอบประสิทธิภาพเชิงเวลาในขั้นตอนการหาความคล้ายของเซสชั่นและขั้นตอนการทำนายค่าคะแนนความชอบของผู้ฟังเพลง

5. ทำรายงานสรุปผล

เผยแพร่ผลงานวิทยานิพนธ์และจัดทำเป็นผลรายงานการทดลอง

แผนการดำเนินโครงงาน

ตารางที่ 1-1 แผนการดำเนินโครงงาน

การดำเนินงาน	2557	2558	2559
	ม.ค. – ธ.ค.	ม.ค. – ธ.ค.	ม.ค. – มิ.ย.
1. รวบรวมข้อมูลและศึกษาปัญหางานวิจัย	←		
2. ศึกษาขั้นตอนและวิเคราะห์ข้อมูล	←		
3. เขียนโปรแกรมและทดสอบ	←		
3. เผยแพร่งานวิจัยที่ 1		←	
4. ศึกษาปรับปรุงขั้นตอนวิธีการ		←	
5. เผยแพร่งานวิจัยที่ 2		←	
6. จัดทำวิทยานิพนธ์		←	→

ประโยชน์ที่คาดว่าจะได้รับ

1. ได้ขั้นตอนวิธีการสร้างรายการเพลงแนะนำซึ่งมีความเหมาะสมกับพฤติกรรมในการฟังเพลงของผู้ฟัง

2. ได้ขั้นตอนวิธีการสร้างรายการเพลงแนะนำซึ่งทำงานได้อย่างเหมาะสมในการประมวลผลข้อมูลที่มีปริมาณมาก

3. ได้ขั้นตอนวิธีการสร้างรายการเพลงแนะนำซึ่งมีประสิทธิภาพเพิ่มขึ้นทั้งทางด้านความเร็วในการประมวลผลและความถูกต้องในการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง

4. เพื่อเป็นแนวทางพื้นฐานในการพัฒนาต่อยอดงานวิจัยต่อไป

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้องที่นำมาใช้ในการพัฒนาขั้นตอนวิธีที่นำเสนอในวิทยานิพนธ์นี้ โดยจะแบ่งเป็นส่วนต่าง ๆ ดังนี้

2.1 วิธีการสร้างรายการแนะนำ

2.2 งานวิจัยที่เกี่ยวข้อง ประกอบด้วย

- 2.2.1 งานวิจัยที่เกี่ยวข้องกับระบบแนะนำเพลง
- 2.2.2 งานวิจัยที่พิจารณาเซสชั่นในการฟังเพลง
- 2.2.3 งานวิจัยที่พิจารณาช่วงเวลาในการฟังเพลง
- 2.2.4 งานวิจัยที่สร้างวิธีการแนะนำแบบออนไลน์

2.3 วิธีการ Session Based Collaborative Filtering

2.1 วิธีการสร้างรายการแนะนำ

การเติบโตของระบบพาณิชย์อิเล็กทรอนิกส์ในปัจจุบันมีการเพิ่มขึ้นอย่างรวดเร็ว ส่งผลให้ฐานข้อมูลของสินค้าและบริการมีจำนวนมาก การที่ผู้ใช้ต้องการสืบค้นสินค้าหรือบริการเป็นเรื่องที่ยุ่งยาก ดังนั้นระบบแนะนำ (Recommender System) จึงเข้ามามีส่วนช่วยในการแนะนำสินค้าหรือบริการให้ตรงกับความต้องการของผู้ใช้จากฐานข้อมูลที่มีขนาดใหญ่ (Information overload) เพื่อช่วยเพิ่มยอดขายให้กับระบบพาณิชย์อิเล็กทรอนิกส์ โดยสร้างรายการสินค้าแนะนำที่คาดว่าผู้ใช้จะพึงพอใจ ดังตัวอย่างเว็บไซต์ที่ประสบความสำเร็จซึ่งมีการใช้วิธีการแนะนำเข้ามาช่วยผู้ใช้ในการเลือกหรือซื้อสินค้า เช่น Amazon[5] ซึ่งเป็นร้านขายหนังสือและสินค้าต่าง ๆ บนอินเตอร์เน็ตโดยการสร้างรายการแนะนำด้วยวิธีการกรองข้อมูลร่วมโดยพิจารณาสินค้าที่ถูกซื้อร่วมกันเป็นหลัก (Linden, G. et al., 2003), Youtube[6] เป็นเว็บไซต์สำหรับรับชมวีดีโอออนไลน์โดยการสร้างรายการวีดีโอแนะนำด้วยวิธีการกรองข้อมูลร่วมซึ่งพิจารณาประวัติในการรับชม, วิดีโอที่ผู้ใช้ชอบและวิดีโอที่ถูกรับชมร่วมกัน (Davidson J. et al., 2010) และ Last.fm[7] เป็นเว็บไซต์ฟังเพลงออนไลน์โดยใช้วิธีการกรองข้อมูลร่วมโดยพิจารณาความสัมพันธ์ทั้งผู้ฟังและเพลงร่วมกันในการสร้างรายการเพลงที่จะแนะนำและศิลปินที่คาดว่าผู้ฟังชอบ (Calandrino J. A., et al., 2011)

เทคนิคที่นิยมใช้ในการสร้างรายการแนะนำสำหรับระบบแนะนำแบ่งออกเป็น 3 วิธีการ ได้แก่ 1) วิธีการกรองข้อมูลร่วม (Collaborative filtering: CF) (Linden, G. et al., 2003) (Candillier, L., et al., 2009) (Kaminskas, M. & Ricci, F., 2012) (Song, Y., et al., 2012) ซึ่งเป็นวิธีการที่นิยมใช้กันอย่างมากในระบบแนะนำและประสบความสำเร็จมากที่สุด โดยใช้การพิจารณาความชอบที่

---

[5] http://www.amazon.com

[6] http://www.youtube.com

[7] http://www.lastfm.com

มีต่อไอเทมทั้งที่เป็นความชอบโดยตรง เช่น คะแนนความนิยม (Rating) ที่ผู้ใช้ให้กับ ไอเทม (Explicit feedback) เป็นต้น หรือความชอบโดยนัย (Implicit feedback) เช่น ประวัติในการซื้อไอเทม หรือประวัติในการฟังเพลง เป็นต้น ซึ่งนำไปพิจารณากับไอเทมที่ผู้ใช้ยังไม่ให้ความนิยมไว้แต่มีความคล้ายกับไอเทมที่ผู้ใช้เคยให้คะแนนไว้แล้ว แบ่งออกเป็น 2 ประเภทคือ

- การกรองข้อมูลร่วมโดยพิจารณาจากผู้ใช้ (User based CF)
- การกรองข้อมูลร่วมโดยพิจารณาจากรายการสินค้า (Item based CF)

ตารางที่ 2-1 นิยามค่าตัวแปร

สัญลักษณ์	**ความหมาย**
$U = \{u_1, u_2, \dots, u_i, \dots, u_{\lvert U\rvert}\}$	เซตของผู้ใช้
$M = \{m_1, m_2, \dots, m_j \dots, m_{\lvert M\rvert}\}$	เซตของไอเทม
R	เมทริกซ์ค่าคะแนนความชอบ $\lvert M\rvert \times \lvert U\rvert$
r_{u_v,m_j}	คะแนนไอเทม m_j ที่ถูกให้คะแนนโดยผู้ใช้ u_v
r_{u_i,m_j}	คะแนนไอเทม m_j ที่ถูกให้คะแนนโดยผู้ใช้ u_i
r_{u_i,m_k}	คะแนนไอเทม m_k ที่ถูกให้คะแนนโดยผู้ใช้ u_i
$r_{u_i,\cdot} = \{r_{u_i,m_i} \in R \lvert m_j \in M\}$	เซตค่าคะแนนความชอบของไอเทมของผู้ใช้ u_i
$r_{\cdot,m_i} = \{r_{u_i,m_i} \in R \lvert u_i \in U\}$	เซตค่าคะแนนความชอบที่ผู้ฟังให้กับไอเทม m_j
$\bar{r}_{u_i,\cdot}$	ค่าคะแนนเฉลี่ยของผู้ใช้ u_i
$\bar{r}_{u_v,\cdot}$	ค่าคะแนนเฉลี่ยของผู้ใช้ u_v
$\bar{r}_{\cdot,m_j}$	ค่าคะแนนเฉลี่ยของไอเทม m_j

ในวิทยานิพนธ์นี้ได้สนใจวิธีการกรองข้อมูลร่วมโดยพิจารณาจากผู้ใช้เป็นหลัก ซึ่งสร้างรายการแนะนำโดยพิจารณาเพียงประวัติในการฟังเพลงเท่านั้น ไม่จำเป็นต้องพิจารณาข้อมูลผู้ใช้หรือเพลง

ขั้นแรกของวิธีการนี้ต้องพิจารณาหาความคล้ายของผู้ใช้เป้าหมายกับผู้ใช้คนอื่นที่อยู่ในฐานข้อมูล โดยตัวอย่างวิธีการที่ใช้ในการหาความคล้าย (Candillier, L., et al., 2009) (Kaminskas, M. & Ricci, F., 2012) ดังสมการที่ (2.1), (2.2) และ (2.3)

วิธีการหาค่าความคล้ายคลึงแบบโคไซน์ (Cosine-based Similarity)

$$sim(u_i, u_v) = \frac{\sum_j (r_{u_i,m_j}, r_{u_v,m_j})}{\sqrt{\sum_j (r_{u_i,m_j})^2}\sqrt{\sum_j (r_{u_v,m_j})^2}} \tag{2.1}$$

วิธีการหาค่าความคล้ายคลึงแบบโคไซน์แบบปรับแก้ (Adjusted Cosine Similarity)

$$sim(u_i, u_v) = \frac{\sum_j \left(r_{u_i,m_j} - \bar{r}_{\cdot,m_j}\right)\left(r_{u_v,m_j} - \bar{r}_{\cdot,m_j}\right)}{\sqrt{\sum_j \left(r_{u_i,m_j} - \bar{r}_{\cdot,m_j}\right)^2}\sqrt{\sum_j \left(r_{u_v,m_j} - \bar{r}_{\cdot,m_j}\right)^2}} \quad (2.2)$$

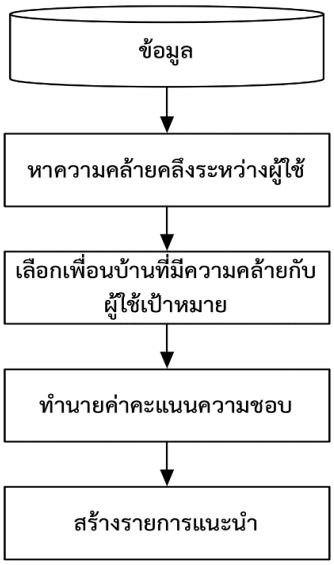
วิธีการหาค่าความคล้ายคลึงแบบค่าสัมประสิทธ์สหสัมพันธ์ของเพียร์สัน (Pearson correlation coefficient)

$$sim(u_i, u_v) = \frac{\sum_{j \in M_u \cap M_v} \left(r_{u_i,m_j} - \bar{r}_{u_i,\cdot}\right)\left(r_{u_v,m_j} - \bar{r}_{u_v,\cdot}\right)}{\sqrt{\sum_{j \in M_u \cap M_v} \left(r_{u_i,m_j} - \bar{r}_{u_i,\cdot}\right)^2}\sqrt{\sum_{j \in M_u \cap M_v} \left(r_{u_v,m_j} - \bar{r}_{u_v,\cdot}\right)^2}} \quad (2.3)$$

ขั้นตอนถัดไปคือการเลือกเพื่อนบ้านที่มีความคล้ายกับผู้ใช้เป้าหมายมากที่สุด (k-Nearest Neighbor) เพื่อนำไปคำนวณค่าคะแนนความชอบของผู้ใช้เป้าหมายโดยใช้สมการ (2.4)

$$p_{u_i,m_j} = \bar{r}_{u_i,\cdot} + \frac{\sum_{u_v \in U} sim(u_i,u_v)\left[r_{u_v,m_j} - \bar{r}_{u_v,\cdot}\right]}{\sum_{u_v \in U} |sim(u_i,u_v)|} \quad (2.4)$$

ข้อด้อยของวิธีการกรองข้อมูลร่วมคือ ข้อมูลไอเทมที่มีอยู่เป็นจำนวนมากเมื่อเทียบกับจำนวนผู้ใช้งานทำให้เกิดปัญหาความเบาบางของข้อมูล (Sparsity problem) และไอเทมใหม่ไม่ถูกแนะนำไปยังผู้ใช้หรือผู้ใช้ที่เข้าสู่ระบบครั้งแรกจะไม่ได้รับการแนะนำเรียกปัญหานี้ว่า Cold start problem ภาพรวมของวิธีการกรองข้อมูลร่วมโดยพิจารณาผู้ฟังเป็นหลักแสดงดังภาพที่ 2-1



ภาพที่ 2-1 ภาพรวมของวิธีการกรองข้อมูลร่วมโดยพิจารณาผู้ฟังเป็นหลัก

2) วิธีการกรองเนื้อหา (Content-based filtering) ใช้วิธีการแนะนำไอเทมโดยพิจารณาจากคุณสมบัติของไอเทมนั้น ๆ ข้อดีของวิธีการกรองเนื้อหาคือสามารถแนะนำไอเทมได้ตรงตามความชอบของผู้ใช้ ส่วนข้อด้อยของวิธีการกรองเนื้อหาคือจะได้รายการไอเทมที่ไม่หลากหลาย (Overspecialization) เช่น รายการเพลงที่มีแนวเพลงเดียวกัน หรือภาพยนตร์ที่มีดารานำเพียงคนเดียว เป็นต้น และต้องการคุณสมบัติของไอเทมจำนวนมากที่เพียงพอในการสร้างระบบแนะนำและความยากในการสกัดคุณสมบัติของไอเทม เช่น แนวเพลงหรือข้อมูลเครื่องดนตรี ตัวอย่างของระบบแนะนำที่ใช้วิธีการกรองเนื้อหา คือ Pandora[8] ซึ่งเป็นระบบแนะนำเพลง โดยระบบจะใช้คุณสมบัติของเพลงคือ คีย์ในเสียงดนตรี ท่อนซ้ำของกีต้าร์ไฟฟ้าและเสียงประสานของเครื่องดนตรี

3) วิธีการกรองแบบผสมผสาน (Hybrid filtering) ใช้เทคนิคข้อดีของทั้งวิธีการกรองร่วมมาผสมกับข้อดีของวิธีการกรองเนื้อหาเพื่อสร้างเป็นระบบแนะนำ โดยข้อด้อยของวิธีการนี้คือต้องการข้อมูลจำนวนมากในการสร้างรายการแนะนำซึ่งไม่เหมาะสมสำหรับบางระบบพาณิชย์อิเล็กทรอนิกส์ที่มีข้อมูลจำนวนน้อย

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 งานวิจัยที่เกี่ยวข้องกับระบบแนะนำเพลง

Jae Sik Lee และ Jin Chun Lee (2007) นำเสนองานวิจัยเรื่อง Context awareness by case-based reasoning in a music recommendation system (Lee J. S. & Lee J. C., 2007) นำเสนอระบบแนะนำเพลงโดยใช้บริบททางด้านข้อมูลสภาพอากาศมาช่วยในการแนะนำ โดยงานวิจัยนี้นำเข้าข้อมูลโดยการใช้ RFID (Radio Frequency Identification) ซึ่งใช้เก็บข้อมูลของผู้ใช้และข้อมูลสภาพอากาศจากเว็บเซอร์วิส และใช้วิธีการ Case-based reasoning ในการแนะนำเพลง ผลการทดลองปรากฏว่าการใช้บริบททางด้านสภาพอากาศเข้ามาทำให้ความถูกต้องโดยใช้ตัววัดประสิทธิภาพคือค่า Precision ดีกว่าวิธีการ Case-based reasoning แบบดั้งเดิมถึง 8 % แต่จุดด้อยของวิธีการนี้คือมีการเก็บข้อมูลผู้ใช้ที่ยุ่งยากและวิธีการทำระบบแนะนำของงานวิจัยนี้คือต้องมีการเก็บองค์ความรู้มาเป็นกรณีเปรียบเทียบเพื่อแนะนำซึ่งมีความยุ่งยาก และเมื่อกรณีที่ผู้ใช้นำเข้ามาไม่ตรงกับองค์ความรู้ที่มีอยู่อาจทำให้การแนะนำไม่มีประสิทธิภาพ

Chan Ho Park และ Minsuk Kahng (2010) นำเสนองานวิจัยเรื่อง Temporal dynamics in music listening behavior : a case study of online music service (Park, C. H. & Kahng, M., 2010) ศึกษาถึงมิติทางด้านเวลาและความชอบของผู้ฟังที่เปลี่ยนไป โดยพิจารณาพฤติกรรมของผู้ฟังเพลงจากประวัติในการฟังเพลงของผู้ใช้ ผลสรุปของงานวิจัยนี้แสดงให้เห็นว่าเมื่อเวลาเปลี่ยนไปผู้ฟังจะมีพฤติกรรมในการฟังเพลงที่เปลี่ยนไป เช่น เพลงแนวแดนซ์และแนวป๊อปถูกฟังมากในช่วงตอนกลางคืนซึ่งตรงกันข้ามกับแนวเพลงอื่น เพลงแนวบัลลาดถูกฟังมากกว่าแนวแดนซ์ป๊อปในช่วงฤดูหนาวแต่ในฤดูร้อนเพลงแนวแดนซ์ป๊อปจะถูกฟังมากกว่า และการฟังเพลงของผู้ใช้ลดลงในวันหยุดและถูกฟังมากที่สุดในเวลา 5 โมงเย็น ส่วนตัวเพลงที่ได้รับความนิยมจะใช้เวลาประมาณ 2 สัปดาห์ถึงจุดที่ได้รับความนิยมสูงสุดหลังจากนั้นก็จะลดลงเรื่อย ๆ ซึ่งจะเห็นได้ว่าในการฟังเพลงมีความแตกต่าง

[8] http://www.pandora.com

จากการดูภาพยนตร์, อ่านข่าว, ท่องเว็บไซต์และเนื้อหาอื่นเพราะผู้ฟังเพลงจะมีการฟังซ้ำในหลายรอบ ซึ่งแสดงถึงเพลงที่ชื่นชอบและการฟังเพลงนั้นสามารถทำร่วมกับกิจกรรมอื่นได้ซึ่งแตกต่างจากเนื้อหาอื่น

Yading Song และคณะ (2012) นำเสนองานวิจัยเรื่อง A Survey of Music Recommendation Systems and Future Perspectives (Song, Y. et al., 2012) โดยศึกษาวิธีการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังโดยเฉพาะซึ่งแบ่งได้เป็น 6 วิธีการ ได้แก่ 1. วิธีการพิจารณาข้อมูลเพลง (Metadata Information Retrieval) เป็นวิธีการที่ง่ายในการค้นหาเพลงโดยใช้ชื่อเพลง, แนวเพลง, ศิลปินหรือเนื้อร้องเป็นต้น 2. วิธีการกรองข้อมูลร่วม โดยพิจารณาความคล้ายของผู้ฟังหรือเพลงสำหรับแนะนำเพลงที่คาดว่าผู้ฟังชื่นชอบ 3. วิธีการวิเคราะห์จากเสียง (Signal-based Music Information Retrieval) โดยพิจารณาเสียงร้องของเพลงที่มีความคล้ายกันในการสร้างรายการแนะนำเพลง เช่น การพิจารณาจังหวะของเพลงหรือลักษณะของเสียงร้องหรือเครื่องดนตรี เป็นต้น 4. วิธีการวิเคราะห์จากอารมณ์ของเพลง (Emotion-based Model) เป็นวิธีการสร้างรายการเพลงจากอารมณ์ของเพลงโดยพิจารณารูปแบบของเสียงร้อง, จังหวะเพลง, แนวเพลงหรือการประสานเสียงในการรู้จำอารมณ์ของเพลง 5. วิธีการพิจารณาบริบทของข้อมูล (Context-based Information Retrieval) วิธีการนี้จะสร้างรายการแนะนำโดยพิจารณาข้อมูลอื่น ๆ เช่น ความคิดเห็น, การวิจารณ์, ความสัมพันธ์ของเพื่อน (Friendship networks) และ Tag นอกเหนือจากประวัติในการฟังเพลงหรือข้อมูลของเพลงและผู้ฟัง และ 6. การผสมวิธีการต่าง ๆ (Hybrid Model Information Retrieval) เป็นวิธีการแนะนำเพลงโดยผสมหลายวิธีการที่กล่าวมาข้างต้นเข้าด้วยกันเพื่อให้ประสิทธิภาพการแนะนำมีความถูกต้องเพิ่มมากขึ้น

Jen-Yu Liu และ Yi-Hsuan Yang (2012). นำเสนองานวิจัยเรื่อง Inferring personal traits from music listening history (Liu, J. Y. & Yang Y. H., 2012) โดยงานวิจัยนี้ได้ศึกษาถึงคุณลักษณะนิสัย (Demographical information) ในการฟังเพลงโดยพิจารณาอายุและเพศของผู้ฟังเพลง โดยวิเคราะห์ 1) เวลาในการฟังเพลง ได้แก่ ชั่วโมง, วันต่อสัปดาห์และเดือนต่อปี 2) เพลงและศิลปิน 3) คุณลักษณะของเสียงเพลง ได้แก่ คีย์, จังหวะ, ระดับเสียง, ทำนอง, โหมด (mode) และความดัง สำหรับวิธีการในการวิเคราะห์คือ Support Vector Machine (SVM) โดยใช้ฟังก์ชัน Redial Basis Function ผลปรากฏว่าการพิจารณาถึงอายุกับวันต่อสัปดาห์ให้ผลดีที่สุดที่ 61.4% คือในวันทำงานผู้ใหญ่ (อายุมากกว่า 24 ปี) มีการฟังเพลงมากกว่าวัยรุ่น (อายุน้อยกว่า 24 ปี) ในวันทำงาน ส่วนวันหยุดวัยรุ่นมีการฟังเพลงมากกว่า ส่วนการพิจารณาถึงเพศของผู้ฟังผลปรากฏว่าการฟังเพลงในชั่วโมงต่อวันให้ผลลัพธ์ที่ดีที่สุดที่ 57.1% โดยผู้หญิงจะฟังเพลงมากสุดที่ช่วง 19 นาฬิกาถึง 22 นาฬิกา ส่วนผู้ชายฟังเพลงมากที่สุดช่วง 6 นาฬิกาถึง 12 นาฬิกา สำหรับเพลงและศิลปิน ในการฟังเพลงโดยพิจารณาอายุกับการฟังศิลปินได้ผลลัพธ์ที่ 71.1% การฟังเพลงโดยพิจารณาเพศกับการฟังเพลงได้ผลลัพธ์ที่ 66.1% ซึ่งจุดด้อยของงานวิจัยนี้คือการที่ใช้ข้อมูลอายุและเพศในฐานข้อมูลนี้อาจเกิดความผิดพลาดได้มากเพราะในการสมัครสมาชิกของผู้ใช้สามารถเว้นว่างไว้ได้สำหรับเพศและอายุ ซึ่งบางครั้งอาจเกิดการที่ผู้ใช้กรอกข้อมูลที่ไม่เป็นความจริงให้กับระบบ

Marius Kaminskas และ Francesco Ricci (2012) นำเสนองานวิจัยเรื่อง Contextual music information retrieval and recommendation: state of the art and challenges

(Kaminskas, M. & Ricci, F., 2012) โดยงานวิจัยนี้ได้นำเสนอถึงระบบแนะนำเพลงซึ่งแสดงให้เห็นถึงบริบทของผู้ใช้มีความสำคัญมากในการเลือกฟังเพลง โดยงานวิจัยนี้ได้แบ่งข้อมูลบริบทของระบบแนะนำเพลงออกเป็น 3 ประเภทคือ 1. บริบทที่เกี่ยวข้องกับสิ่งแวดล้อม ได้แก่ ข้อมูลเกี่ยวกับสถานที่, ข้อมูลเกี่ยวกับเวลาและข้อมูลเกี่ยวกับสภาพอากาศ 2. บริบทที่เกี่ยวกับผู้ใช้ ได้แก่ กิจกรรมของผู้ใช้, ข้อมูลส่วนตัวของผู้ใช้ และอารมณ์ของผู้ใช้ 3. บริบททางด้านมัลติมีเดีย ได้แก่ ข้อความและรูปภาพ โดยทั่วไปมีการรวมข้อมูลบริบทระหว่างบริบทที่เกี่ยวข้องกับสิ่งแวดล้อมกับบริบทที่เกี่ยวกับผู้ใช้ เช่น กิจกรรมในแต่ละวันหรือการใช้ข้อมูลส่วนตัวของผู้ใช้มาผสมกับข้อมูลบริบทเวลา เป็นต้น

Shuiguang Deng และคณะ (2015) นำเสนองานวิจัยเรื่อง Exploring user emotion in microblogs for music recommendation (Deng, S. et al., 2015) งานวิจัยนี้นำเสนอวิธีการแนะนำเพลงโดยพิจารณาอารมณ์ของผู้ใช้จาก micro blogs คือเว็บไซต์สังคมออนไลน์ Sina Weibo ซึ่งจะทำการวิเคราะห์อารมณ์จากสถานะส่วนตัวร่วมกับพิจารณาช่วงเวลา (Time window) ของแต่ละผู้ฟังโดยใช้วิธีการวิเคราะห์ข้อความซึ่งแบ่งระดับของอารมณ์ออกเป็น 2 แง่อารมณ์ (อารมณ์แง่บวก , อารมณ์แง่ลบ), 7 แง่อารมณ์และ 21 แง่อารมณ์ สำหรับวิธีการทำนายค่าคะแนนความชอบโดยประยุกต์ใช้วิธีการกรองร่วมโดยพิจารณาผู้ใช้และเพลงเป็นหลักโดยใช้วิธีการหาความคล้ายแบบโคไซน์ในการหาความคล้ายของอารมณ์ของผู้ใช้ จากผลการทดลองแสดงให้เห็นว่าอารมณ์ของผู้ใช้มีอิทธิผลในการเลือกฟังเพลงและวิธีการกรองร่วมโดยพิจารณาผู้ใช้เป็นหลักให้ความถูกต้องดีที่สุดในตัววัดประสิทธิภาพ Hit rate

2.2.2 งานวิจัยที่พิจารณาเซสชั่นในการฟังเพลง

Sung Eun Park และคณะ (2011) นำเสนองานวิจัยเรื่อง Session-Based Collaborative Filtering for Predicting the Next Song (Park, S. E., et al, 2011) ซึ่งนำเสนอวิธีการแนะนำเพลงที่เรียกว่า Session-Based Collaborative Filtering (SSCF) ซึ่งพิจารณาพฤติกรรมของการฟังเพลงของผู้ฟังที่มีการฟังซ้ำและมีการเล่นเพลงที่เป็นลำดับต่อเนื่องหรือเซสชั่น (Session) โดยในวิธีการแนะนำเพลงได้ประยุกต์ใช้วิธีการกรองข้อมูลร่วมโดยพิจารณาเพลงที่ถูกฟังในแต่ละเซสชั่นแทนที่ของการพิจารณาความชอบของผู้ฟังคนอื่น จากผลการทดลองในฐานข้อมูลเพลง Bugs music โดยใช้วิธีการวัดประสิทธิภาพคือ Hit ratio (HR@n) ซึ่งเป็นการวัดประสิทธิภาพความถูกต้องของรายการแนะนำกับเพลงที่ผู้ฟังเลือกฟังถัดไป ผลการทดลองแสดงให้เห็นถึงวิธีการ SSCF ให้ความถูกต้องที่ดีกว่าวิธีการกรองร่วมแบบเดิม (Collaborative filtering) แต่วิธีการนี้ไม่ได้คำนึงถึงเพลงที่ถูกฟังในปัจจุบันซึ่งแสดงถึงความชอบของผู้ฟังขณะนั้นและเมื่อข้อมูลของเพลงและผู้ฟังมากขึ้น (Scalability problem) ทำให้เกิดปัญหาในด้านประสิทธิภาพในการคำนวณและความถูกต้องของการสร้างรายการเพลงแนะนำให้กับผู้ฟัง

Dias, R. และ Fonseca, M.J. (2013) นำเสนองานวิจัยเรื่อง Improving music recommendation in session-based collaborative filtering by using temporal context (Dias, R. & Fonseca, M.J., 2013) ซึ่งนำเสนอวิธีการแนะนำเพลงที่เรียกว่า Temporal Session-based Collaborative Filtering (TSSCF) โดยจัดกลุ่มเซสชั่นในการฟังเพลงของผู้ฟังโดยใช้ Gaussian Mixture Model using Expectation Maximization algorithm (GMM) ซึ่งพิจารณา

ช่วงเวลาของผู้ฟังเพลงคือ เวลาของวัน, วันของสัปดาห์, วันของเดือน, เดือนและค่า Session diversity ในการคำนวณอัตราส่วนของการฟังซ้ำ โดยใช้วิธีการแนะนำเพลงคือวิธีการกรองร่วมโดยพิจารณาเซสชั่นของการฟังเพลงแทนที่การพิจารณาความชอบของคนอื่น โดยใช้การคำนวณความคล้ายของเซสชั่นที่อยู่ในกลุ่มเดียวกัน จากผลการทดลองแสดงให้เห็นว่าวิธีการ TSSCF มีความถูกต้องมากกว่า SSCF โดยใช้วิธีการวัดประสิทธิภาพ Hit ratio สำหรับการวัดประสิทธิภาพความถูกต้องด้วย Mean Reciprocal Rank (MRR) วิธีการ SSCF มีความถูกต้องมากกว่า TSSCF อย่างไรก็ตามวิธีการนี้เมื่อประยุกต์ใช้สำหรับการแนะนำเพลงแบบออนไลน์อาจมีความไม่เหมาะสม เช่น การกำหนดจำนวนกลุ่มของผู้ฟังตามเวลาที่ต้องเปลี่ยนไปตามความชอบของผู้ฟังหรือเพลงของบริบทในปัจจุบันของผู้ฟังและการกำหนดจำนวนค่าพารามิเตอร์ต่าง ๆ ของวิธีการ GMM

Oscar Carlsson (2014) นำเสนองานวิจัยเรื่อง Cluster User Music Sessions (Carlsson, O., 2014) ได้ศึกษาพฤติกรรมของการฟังเพลงของผู้ฟังเพื่อหาองค์ความรู้ใหม่หรือหาผลสรุปของข้อมูล โดยจัดกลุ่มเซสชั่นของเพลงที่ถูกฟังโดยผู้ฟังจากฐานข้อมูล Spotify มีการใช้วิธีการจัดกลุ่มคือ เคมีนส์ (K-means algorithm), เคมีนส์ดอย (K-medoids algorithm) และการจัดกลุ่มแบบลำดับชั้นคือ Hierarchical Agglomerative จากนั้นวิธีการหาความคล้ายแบบโคไซน์สำหรับโมเดลในวิธีการจัดกลุ่ม จากผลสรุปของข้อมูลแสดงให้เห็นถึงกลุ่มของแนวเพลงหรือกลุ่มของอารมณ์เพลงที่ผู้ฟังชื่นชอบในการฟังร่วมกัน เช่น กลุ่มของแนวเพลง comedy ประกอบด้วย แนวเพลง Bubblegum, Comedy, Swedish Pop และ Brithpop รวมถึงกลุ่มของเซสชั่นแสดงให้เห็นถึงบริบทของผู้ฟัง เช่น เพลงที่เกี่ยวกับเทศกาล Christmas จะถูกฟังเฉพาะช่วงเดือนธันวาคมแสดงให้เห็นถึงบริบทที่เกี่ยวข้องกับเวลา จากงานวิจัยนี้แสดงให้เห็นถึงกลุ่มของเพลงที่เกิดขึ้นร่วมกันในการฟังเพลงซึ่งสามารถนำไปประยุกต์ใช้สำหรับการสร้างระบบแนะนำหรือช่วยตัดสินใจในด้านการตลาดสำหรับเพิ่มยอดขายในร้านค้าเพลงออนไลน์

Ke Ji และคณะ (2015) นำเสนองานวิจัยเรื่อง Next-song recommendation with temporal dynamics (Ji, K., et al, 2015) ได้สร้างรายการเพลงแนะนำให้กับผู้ฟัง โดยการพิจารณาความชอบของผู้ฟังในระยะสั้น (Short-term) ด้วยวิธี Time-based Markov Embedding (TME) ซึ่งวิเคราะห์ความน่าจะเป็นของเพลงที่น่าจะถูกเลือกฟังถัดไป โดยพิจารณาเพลงกับเพลงที่ถูกฟังต่อเนื่องกัน, ความสนใจของผู้ฟังในระยะยาว (Long-term) ด้วยวิธี TME ซึ่งวิเคราะห์ความน่าจะเป็นของผู้ฟังกับเพลงและความสนใจในเซสชั่น (Session-term) ด้วยวิธี TME ซึ่งวิเคราะห์ความน่าจะเป็นของเซสชั่นกับเพลง กำหนดให้การหยุดฟังเพลงนานกว่า 1 ชั่วโมงพิจารณาเป็น 1 เซสชั่น แล้วพิจารณาถึงความชอบที่เปลี่ยนไปตามเวลา (temporal dynamics) ซึ่งคำนวณความน่าจะเป็นของเพลงที่ถูกเลือกฟังถัดไปในแต่ละช่วงเวลา จากผลการทดลองแสดงให้เห็นถึงวิธีการที่นำเสนอสามารถให้ความถูกต้องในการแนะนำเพลงดีกว่าวิธี Bigram, Personalized Bigram, Logistic Markov Embedding, Personalized Markov Embedding โดยใช้วิธีการวัดประสิทธิภาพ Recall และ Precision ในฐานข้อมูลเพลง Last.fm อย่างไรก็ตามงานวิจัยนี้มีการแนะนำเพลงแบบออฟไลน์และไม่พิจารณาช่วงเวลาในการฟังเพลงซึ่งส่งผลต่อการเลือกฟังเพลงของผู้ฟัง

2.2.3 งานวิจัยที่พิจารณาช่วงเวลาในการฟังเพลง

Linas Baltrunas และ Xavier Amatriain (2009) นำเสนองานวิจัยเรื่อง Towards time-dependant recommendation based on implicit feedback (Baltrunas, L., & Amatriain, X., 2009) โดยงานวิจัยนี้มีการพิจารณาช่วงเวลาในการฟังเพลงของผู้ฟังเข้ามาช่วยในการสร้างระบบแนะนำเพลงโดยใช้วิธี Factorization based collaborative filtering algorithm ในการทำนายค่าคะแนนความชอบของผู้ฟังเพลง โดยแบ่งช่วงเวลาในการฟังเพลงของผู้ฟังออกเป็น 3 ชุดข้อมูล ได้แก่ 1. ช่วงเวลาตอนเช้าและตอนกลางคืน 2. วันหยุดและวันทำงาน 3. ฤดูร้อนและฤดูหนาว จากงานวิจัยนี้แสดงให้เห็นว่าข้อมูลบริบททางด้านเวลาช่วยทำให้ระบบแนะนำมีประสิทธิภาพความถูกต้องดีกว่าวิธีการกรองร่วมแบบเดิมโดยใช้วิธีการวัดประสิทธิภาพ Mean Absolute Error (MAE) แต่จุดด้อยของวิธีการนี้คือไม่สามารถแนะนำเพลงในเวลาที่คาบเกี่ยวกันได้ เช่น การแนะนำศิลปินในเวลาเช้าในวันทำงานได้หรือแนะนำศิลปินในวันหยุดของตอนเย็นได้ เป็นต้น

Toni Cebrián และคณะ (2010) นำเสนองานวิจัยเรื่อง Music Recommendations with Temporal Context Awareness (Cebrián, T. et al., 2010) นำเสนอระบบแนะนำเพลงโดยพิจารณาข้อมูลบริบททางด้านเวลา ซึ่งได้แบ่งข้อมูลผู้ฟังออกเป็นโปรไฟล์ย่อย ๆ ตามเวลาจากข้อมูลเพลง แบ่งเป็น 2 ชุดข้อมูลได้แก่ ชุดที่ 1 คือ ช่วงเวลาเช้า, ช่วงเวลากลางวันและช่วงเวลาเย็น และชุดที่ 2 คือ วันทำงานและวันหยุด โดยใช้ผลคูณคาร์ทีเซียนโปรดักต์ (Cartesian product) ในการสร้างช่วงเวลาที่คาบเกี่ยวกัน เช่น ตอนเช้าในวันหยุดหรือตอนเย็นในวันทำงาน สำหรับวิธีการสร้างรายการเพลงแนะนำที่ผู้ใช้ชื่นชอบได้ใช้วิธีการกรองข้อมูลร่วมโดยพิจารณาผู้ฟังเป็นหลักของแต่ละช่วงเวลา แต่จุดด้อยของงานวิจัยนี้คือสำหรับผู้ฟังบางคนอาจจะชอบฟังเพลงในหลายช่วงเวลาโดยช่วงเวลาของวันไม่มีอิทธิพลในการเลือกฟังเพลงของผู้ฟัง

Perfecto Herrera และคณะ (2010) ได้นำเสนองานวิจัยเรื่อง Rocking around the clock eight days a week: an exploration of temporal patterns of music listening (Herrera, P. el al., 2010) โดยศึกษาความชอบของผู้ใช้ ซึ่งพิจารณาผู้ใช้กับศิลปินและผู้ใช้กับแนวเพลงเมื่อเวลาในการฟังเพลงของผู้ใช้เปลี่ยนไป จากสมมติฐานของงานวิจัยนี้คือในช่วงระยะเวลาหนึ่งของวันหรือสัปดาห์ ผู้ใช้จะมีการรับฟังเพลงซ้ำในเพลงที่ชื่นชอบ (Temporal pattern) ซึ่งงานนี้ได้ใช้ข้อมูลเพลงและศิลปินจาก Last.fm โดยการวิเคราะห์สถิติเชิงมุม (Circular statistics) ซึ่งคำนวณช่วงเวลาให้อยู่ในรูปเชิงมุม (Angular scale) แล้วหาค่าโน้มเอียงเข้าสู่ศูนย์กลางของแนวเพลงหรือศิลปิน (Central tendency measure) และค่าเฉลี่ยของความยาวเวกเตอร์เชิงมุม (Mean resultant vector length) เพื่อพิจารณาแนวโน้มเวลาในการฟังศิลปินหรือแนวเพลงของผู้ใช้ จากผลการทดลองพบว่าผู้ฟังมีการฟังศิลปินหรือแนวเพลงที่ชอบซ้ำในบางช่วงของเวลา โดยการฟังเพลงในบางศิลปินของผู้ใช้ในทุกสัปดาห์แสดงให้เห็นถึงรูปแบบการฟังเพลงที่ชัดเจน ส่วนการฟังเพลงโดยพิจารณาถึงแนวเพลงพบว่าในหลายผู้ใช้มีการฟังเพลงตามแนวเพลงไม่บ่อยนัก ส่วนบางคนมีการฟังเพลงตามแนวเพลงอย่างมาก

Markus Schedl และคณะ (2014) นำเสนองานวิจัยเรื่อง Mobile Music Genius: Reggae at the Beach, Metal on a Friday Night? (Schedl, M. et al, 2014) ได้นำเสนอวิธีการสร้างระบบแนะนำเพลงโดยพิจารณาบริบท (Context) ของผู้ฟังจากข้อมูลเซนเซอร์และข้อมูลเวลาจากตัว

อุปกรณ์โทรศัพท์เคลื่อนที่แทนที่การใช้ความคิดเห็นของผู้ฟังโดยตรง (User feedback) เช่น คะแนนความชอบ (Rating) เนื่องจากงานวิจัยส่วนมากไม่ได้พิจารณาบริบทของผู้ฟังในการสร้างรายการเพลงแนะนำและจำเป็นต้องใช้ความคิดเห็นของผู้ฟัง ดังนั้นในงานวิจัยนี้ได้มีการพิจารณาบริบทของผู้ฟัง คือ ช่วงเวลาของผู้ฟัง, สถานที่, สภาพอากาศ, กิจกรรมต่าง ๆ ร่วมกับวิธีการทำนายค่าความชอบโดยใช้โมเดล C4.5 จากผลการทดลองในการเปรียบเทียบกับวิธีการจำแนกประเภทแบบต่าง ๆ วิธีการ C4.5 ให้ความถูกต้องที่ดีที่สุด

Yajie Hu และ Mitsunori Ogihara (2011) นำเสนองานวิจัยเรื่อง NEXTONE PLAYER: A MUSIC RECOMMENDATION SYSTEM BASED ON USER BEHAVIOR (Hu, Y. & Oguhara, M., 2011) ซึ่งทำการสร้างวิธีการแนะนำเพลงโดยคำนึงถึงความชอบของผู้ฟัง, รูปแบบในการฟังเพลง, เพลงใหม่ โดยใช้วิธีการวิเคราะห์ช่วงเวลา (Time series) คือ Autoregressive Integrated Moving Average (ARIMA) ในการพิจารณาลำดับของการฟังแนวเพลงซึ่งจะทำนายแนวเพลงที่ผู้ฟังจะฟังถัดไปรวมถึง Recording year และวิธีการนี้มีการคำนึงถึงความใหม่ของเพลงโดยการใช้วิธีการ Forgetting Curve นอกจากนี้ในงานวิจัยนี้มีการพิจารณาช่วงเวลาในการฟังเพลงซึ่งผู้ฟังมีความชอบที่เปลี่ยนไปตามเวลาโดยการใช้วิธีการ Gaussian Mixture Model ในการประมาณค่าความน่าจะเป็นที่เพลงจะถูกเลือกฟังเฉพาะเวลา จากผลการทดลองแสดงถึงวิธีการที่นำเสนอมีความถูกต้องมากกว่ารายการเพลงที่แนะนำแบบสุ่ม

2.2.4 งานวิจัยที่สร้างวิธีการแนะนำแบบออนไลน์

Manos Papagelis และคณะ (2005) นำเสนองานวิจัยเรื่อง Incremental Collaborative Filtering for Highly Scalable Recommendation Algorithms (Papagelis, M. et al., 2005) นำเสนอวิธีการแนะนำคือ Incremental Collaborative Filtering (ICF) โดยแก้ปัญหาการประมวลผลข้อมูลที่มีปริมาณมาก (Scalability) ซึ่งมีการอัพเดทวิธีการสร้างรายการแนะนำเมื่อมีข้อมูลชุดใหม่เข้ามา (Incremental update) แทนที่การคำนวณในแบบออฟไลน์คือใช้เวลาในการหาความคล้ายของผู้ใช้เป็น $O(m^2n)$ และการทำนายค่าคะแนนความชอบเป็น $O(n)$ หรือสร้างรายการแนะนำแบบออนไลน์เป็น $O(mn) + O(n)$ ซึ่งวิธีการที่นำเสนอนี้ใช้เวลาในการคำนวณที่เร็วกว่าทั้งสองวิธีการที่กล่าวคือใช้เวลาในการหาความคล้ายเป็น $O(mn)$ เมื่อมีผู้ใช้ให้ค่าคะแนนความชอบกับสินค้าใหม่หรือสินค้าที่มีอยู่เดิมและใช้เวลาในการทำนายค่าเป็น $O(n)$ โดยที่ m คือจำนวนผู้ใช้และ n คือจำนวนเพลงจากผลการทดลองแสดงให้เห็นว่าวิธีการ ICF ใช้เวลาในการคำนวณที่รวดเร็วมากกว่าวิธีการกรองร่วมแบบเดิมและมีความเหมาะสมกับระบบแนะนำในแบบออนไลน์

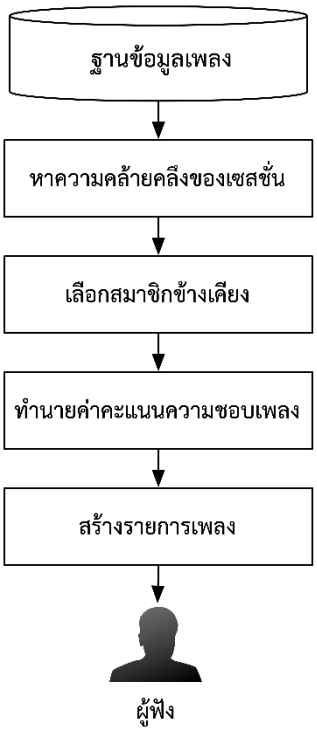
Catarina Miranda และ Alipio M. Jorge (2008) นำเสนองานวิจัยเรื่อง Incremental collaborative filtering for binary ratings. (Miranda, C. & Jorge, A. M., 2008) ได้นำเสนอวิธีการ Incremental item-based CF ซึ่งแก้ปัญหาการประมวลผลข้อมูลที่มีปริมาณมากโดยการคำนวณหาความคล้ายของสินค้าแทนที่ผู้ใช้เพื่อหาความคล้ายเมื่อผู้ใช้ให้คะแนนสินค้าใหม่หรือผู้ใช้อัพเดทคะแนนสินค้าเดิมที่มีอยู่ในระบบ (Incremental update) จากผลการทดลองเปรียบเทียบกับวิธีการกรองร่วมแบบเดิมแสดงให้เห็นว่าวิธีการที่นำเสนอมีประสิทธิภาพดีกว่าทั้งในด้านความเร็วในการคำนวณและความถูกต้องของรายการแนะนำ

Xiao Yang และคณะ (2012) นำเสนองานวิจัยเรื่อง Scalable Collaborative Filtering Using Incremental Update and Local Link Prediction (Yang, X. et al., 2012) ได้นำเสนอวิธีการ Scalable Incremental Collaborative Filtering (SICF) ซึ่งแก้ปัญหาการประมวลผลข้อมูลที่มีปริมาณมากโดยใช้วิธีการกรองร่วมร่วมกับการอัพเดทข้อมูลทีละชุดข้อมูลเมื่อมีข้อมูลใหม่เข้ามา ซึ่งพิจารณาความคล้ายของภาพยนตร์ในฐานข้อมูล MovieLens และ Netflix สำหรับการแก้ปัญหาความเบาบางของข้อมูล (Sparsity problem) ได้ใช้วิธีการ Link prediction โดยจำลองการหาความคล้ายในรูปแบบของกราฟเพื่อใช้คุณสมบัติการถ่ายทอด (Transitive) ของกราฟในการหาความสัมพันธ์ของภาพยนตร์ จากผลการทดลองในการวัดประสิทธิภาพการทำนายค่าความถูกต้องด้วย MAE และ RMSE สามารถให้ความถูกต้องที่ดีกว่าวิธีการกรองร่วมแบบเดิมรวมถึงใช้เวลาในการคำนวณที่รวดเร็วกว่าวิธีการกรองร่วมแบบเดิม

Joao Vinagre และ Alipio M. Jorge (2012) นำเสนองานวิจัยเรื่อง Forgetting mechanisms for scalable collaborative filtering (Vinagre, J. & Jorge, A. M., 2012) นำเสนอวิธีการ Forgetting mechanisms ร่วมกับวิธีการกรองร่วมแบบเดิมเพื่อแก้ปัญหาขนาดข้อมูล (Scalability) และเพิ่มความถูกต้องให้กับวิธีการแนะนำ โดยวิธีการ Forgetting mechanisms ซึ่งเป็นวิธีการสำหรับลดความสำคัญของข้อมูลเก่าเพื่อแสดงถึงความชอบในปัจจุบันของผู้ใช้แบ่งออกเป็น 2 วิธีการคือ 1. Sliding windows คือการพิจารณาขนาดของข้อมูลล่าสุดแบบคงที่ (Fix-size window) และ 2. Fading factors คือการให้ความสำคัญกับข้อมูลใหม่โดยการให้ค่าน้ำหนักกับข้อมูล จากผลการทดลองแสดงให้เห็นว่าวิธีการ Forgetting mechanisms สามารถลดเวลาที่ใช้ในการคำนวณและหน่วยความจำที่ใช้ซึ่งช่วยปรับปรุงปัญหาการประมวลผลข้อมูลที่มีปริมาณมากแต่ยังคงซึ่งความถูกต้องในการแนะนำเพลง นอกจากนี้วิธีการกรองข้อมูลร่วมโดยใช้วิธี Fading factors มีความเหมาะสมสำหรับการอัพเดทข้อมูล ในแบบออนไลน์มากกว่าวิธีการ Sliding windows ซึ่งสอดคล้องกับงานวิจัย Joao Gama และคณะ (2014) นำเสนองานวิจัยเรื่อง A survey on concept drift adaptation (Gama, J. et al, 2014) ได้แบ่งวิธีการ Forgetting mechanisms ออกเป็น 2 วิธีการ คือ 1. Abrupt Forgetting คือการลดความสำคัญของข้อมูลแบบทันทีโดยใช้ข้อมูลเพียงบางส่วนในการพิจารณา (Partial memory) แบ่งออกเป็น 1.1 Sequence based คือการกำหนดขนาดของข้อมูลแบบกำหนดขนาดคงที่ 1.2 timestamp based คือการกำหนดขนาดของข้อมูลตามวันเวลาของระบบ 2. Gradual Forgetting คือการลดความสำคัญของข้อมูลเก่าหรือการให้ความสำคัญของข้อมูลใหม่โดยใช้ข้อมูลในการพิจารณาทั้งหมดซึ่งแตกต่างจาก Abrupt Forgetting โดยมีการให้ค่าน้ำหนักตามความเก่าใหม่ของข้อมูล เช่น การใช้วิธีการ Linear decay ในการลดความสำคัญของข้อมูลโดยพิจารณาวันเวลาที่ถูกนำเข้าร่วมกับ Exponential function

2.3 วิธีการ Session Based Collaborative Filtering

วิธีการ Session based Collaborative Filtering (SSCF) ถูกนำเสนอโดย Sung Eun Park และคณะ (Park, S. E., et al, 2011) โดยนำเสนอวิธีการสำหรับสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง ซึ่งพิจารณาเซสชั่นโปรไฟล์ของผู้ฟังเพลงคือเพลงที่ถูกฟังอย่างต่อเนื่อง เนื่องจากผู้ฟังมักมีการฟังเพลงซ้ำที่คล้ายกับเซสชั่นในอดีต เมื่อผู้ฟังไม่มีการฟังเพลงเกินกว่า 30 นาทีจะพิจารณาเป็น 1 เซสชั่น ซึ่งจะเป็นการพิจารณาการฟังซ้ำและลำดับในการฟังเพลงแล้วนำเข้าไปยังวิธีการกรองข้อมูลร่วมโดยใช้เซสชั่นโปรไฟล์แทนที่ของโปรไฟล์ของผู้ใช้คนอื่น แสดงภาพรวมการทำงานของวิธีการ SSCF ดังภาพที่ 2-2



ภาพที่ 2-2 ภาพรวมการทำงานของวิธีการ Session based Collaborative Filtering

ขั้นตอนแรกของวิธีการสร้างรายการเพลงแนะนำคือการหาค่าความคล้ายของเซสชั่นปัจจุบัน (ss_{as}) กับเซสชั่นในอดีตของผู้ฟัง (ss_v) โดยใช้วิธีการหาความคล้ายแบบโคไซน์ซึ่งเป็นการพิจารณาความคล้ายของเพลงที่ถูกฟังร่วมกันในแต่ละเซสชั่น ดังสมการที่ (2.5)

$$sim(ss_{as}, ss_v) = \frac{\sum_j (r_{ss_{as},m_j}, r_{ss_v,m_j})}{\sqrt{\sum_j (r_{ss_{as},m_j})^2} \sqrt{\sum_j (r_{ss_v,m_j})^2}} \quad (2.5)$$

โดยที่ $sim(ss_{as}, ss_v)$ คือ ค่าความคล้ายของเซสชั่นปัจจุบัน (ss_{as}) กับเซสชั่น (ss_v)

$ss_{ss_{as}}$ คือ เซสชั่นปัจจุบันของผู้ฟังเป้าหมาย

ss_v คือ เซสชั่น v ของผู้ฟัง

r_{ss_{as},m_j} คือ ความถี่ของเพลง m_j ในเซสชั่นปัจจุบัน

r_{ss_v,m_j} คือ ความถี่ของเพลง m_j ในเซสชั่น ss_v

จากนั้นพิจารณาเซสชั่นที่มีความคล้ายกับเซสชั่นปัจจุบันมากที่สุดโดยใช้การพิจารณาเซสชั่นข้างเคียงซึ่งมีความคล้ายกับเซสชั่นเป้าหมายมากที่สุดด้วยวิธี k-Nearest Neighbor (k-NN) ขั้นตอนถัดไปของการสร้างรายการเพลงแนะนำคือการทำนายค่าคะแนนความชอบของผู้ฟังเพลง โดยประยุกต์ใช้วิธีการทำนายค่าของวิธีการกรองร่วมโดยพิจารณาผู้ฟังเป็นหลัก ดังสมการ (2.6)

$$p_{ss_{as},m_j} = \bar{r}_{ss_{as}} + \frac{\sum_{ss_v \in S} sim(ss_{as}, ss_v)\left[r_{ss_v,m_j} - \bar{r}_{ss_v}\right]}{\sum_{ss_v \in S} |sim(ss_{as}, ss_v)|} \quad (2.6)$$

โดยที่ p_{ss_{as},m_j} คือ ค่าการทำนายของเพลง m_j ที่เซสชั่นปัจจุบัน (ss_{as})

r_{ss_v,m_j} คือ ความถี่ของเพลง m_j ในเซสชั่น ss_v

$\bar{r}_{ss_{as}}$ คือ ค่าเฉลี่ยของการฟังเพลงที่เซสชั่นปัจจุบัน (ss_{as})

$\bar{r}_{ss_v}$ คือ ค่าเฉลี่ยของการฟังเพลงที่เซสชั่น ss_v

ขั้นตอนสุดท้ายคือการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง โดยพิจารณาค่าคะแนนความชอบของแต่ละเพลงเรียงลำดับจากมากไปหาน้อย N เพลง

ตารางที่ 2-2 งานวิจัยที่เกี่ยวข้องกับระบบแนะนำเพลง

งานวิจัยที่เกี่ยวข้องกับระบบแนะนำเพลง	วัตถุประสงค์	วิธีการ	ผลสรุป	ฐานข้อมูล
Lee J. S. & Lee J. C., 2007	สร้างระบบแนะนำเพลงโดยพิจารณาบริบททางด้านข้อมูลสภาพอากาศ	Case-based reasoning ร่วมกับพิจารณาข้อมูลสภาพอากาศ	สามารถเพิ่มความถูกต้องให้กับรายการแนะนำที่ดีกว่าวิธีการCase-based reasoningแบบดั้งเดิม	จัดเก็บข้อมูลเอง
Park, C. H. & Kahng, M., 2010	ศึกษาพฤติกรรมของผู้ฟังเพลง	เปรียบเทียบหลาย ๆ แนวเพลงหรือศิลปินในช่วงเวลาต่าง ๆ	บางแนวเพลงหรือศิลปินมีการฟังที่แตกต่างจากช่วงเวลาอื่น	Bugs music
Liu, J. Y. & Yang Y. H., 2012	ศึกษาคุณลักษณะนิสัยในการฟังเพลงของผู้ฟัง	วิเคราะห์และเปรียบเทียบบริบทของผู้ฟังคือเวลาในการฟังเพลง, เพลงและศิลปิน, คุณลักษณะของการฟังเพลงโดยใช้วิธี SVM	ผู้ใช้ที่มีอายุมากกว่า 24 ปีมีการฟังเพลงในวันทำงานมากกว่าผู้ใช้ที่มีอายุน้อยกว่า	Last.fm
Deng, S. et al., 2015	เพิ่มความถูกต้องโดยพิจารณาอารมณ์ของผู้ฟังเพลง	เปรียบเทียบวิธีการ User based CF และ Item based CF	วิธีการ User based CF ให้ผลความถูกต้องที่ดีที่สุดและอารมณ์ของผู้ฟังเพลงส่งผลต่อการเลือกฟังเพลงของผู้ฟัง	Sina Weibo

ตารางที่ 2-3 งานวิจัยที่เกี่ยวข้องที่ใช้เซสชั่นในการฟังเพลง

งานวิจัยที่ใช้เซสชั่นในการฟังเพลง	วัตถุประสงค์	วิธีการ	ผลสรุป	ฐานข้อมูล
Park, S. E., et al, 2011	ปรับปรุงความถูกต้องของรายการเพลงแนะนำโดยพิจารณาเซสชั่นของผู้ฟังเพลง	Session Based Collaborative Filtering (SSCF)	สามารถเพิ่มความถูกต้องได้ดีกว่าวิธีการกรองร่วมแบบเดิม (CF) โดยใช้ตัววัด HitRatio	Bugs music
Dias, R. & Fonseca, M.J., 2013	ปรับปรุงความถูกต้องของการแนะนำเพลงโดยพิจารณาเซสชั่นของผู้ฟังเพลงร่วมกับช่วงเวลา	GMM ในการจัดกลุ่มช่วงเวลาของผู้ฟังร่วมกับวิธีการ SSCF เรียกว่า Temporal Session Based Collaborative Filtering (TSSCF)	สามารถเพิ่มความถูกต้องดีกว่า SSCF โดยใช้ตัววัด HitRatio	Last.fm
Oscar, C., 2014	ศึกษาความสัมพันธ์ของเซสชั่นการฟังเพลง	เปรียบเทียบวิธีการจัดกลุ่มเซสชั่นโดยใช้ K-mean, K-medoids และ Hierarchical agglomerative	กลุ่มของแนวเพลงที่มีความคล้ายกัน เช่นกลุ่มแนวเพลง comedy จะเกิดขึ้นในเซสชั่นเดียวกัน	Spotify

ตารางที่ 2-4 งานวิจัยที่เกี่ยวข้องที่พิจารณาช่วงเวลาในการฟังเพลง

งานวิจัยที่พิจารณาช่วงเวลาในการฟังเพลง	วัตถุประสงค์	วิธีการ	ผลสรุป	ฐานข้อมูล
Baltrunas, L., & Amatriain, X., 2009	เพิ่มความถูกต้องของรายการเพลงโดยพิจารณาช่วงเวลาในการฟังเพลง	MF ร่วมกับช่วงเวลาของผู้ฟังแต่ละช่วงเวลา (micro-profiles) ได้แก่ ช่วงเช้า, ช่วงเย็น, วันหยุดและวันทำงาน	สามารถเพิ่มความถูกต้องดีกว่า CF แบบเดิม	Last.fm
Cebrián, T. et al., 2010	เพิ่มความถูกต้องของรายการเพลงร่วมกับแก้ไขปัญหาช่วงเวลาที่คาบเกี่ยวกันของ micro-profiles	ใช้วิธีคาทีเซียนโปรดักซ์กับช่วงเวลาต่าง ๆ ร่วมกับวิธีการกรองข้อมูลร่วม	สามารถแนะนำเพลงในช่วงเวลาที่คาบเกี่ยวกันได้ เช่น ช่วงเช้าของวันหยุด	จัดเก็บข้อมูลเอง
Herrera, P. el al., 2010	ศึกษาพฤติกรรมในการฟังศิลปินหรือแนวเพลงที่มีการรับฟังแตกต่างจากช่วงเวลาอื่น	Circular Statistics	บางแนวเพลงมีการฟังเด่นชัดในบางช่วงเวลาซึ่งแตกต่างจากช่วงเวลาอื่น	Last.fm
Schedl, M. et al, 2014	สร้างระบบแนะนำเพลงโดยพิจารณาบริบทของผู้ฟังบนอุปกรณ์โทรศัพท์เคลื่อนที่	ใช้โมเดลในการจำแนกประเภทร่วมกับพิจาณาบริบทของผู้ฟังเพลง	วิธีการจำแนกประเภทโดยใช้ C4.5 ให้ผลลัพธ์ที่ดีที่สุด	จัดเก็บข้อมูลเอง
Hu, Y. & Oguhara, M., 2011	สร้างระบบแนะนำเพลงโดยพิจารณาพฤติกรรมของผู้ฟัง	วิธีการ ARIMA วิเคราะห์เพลงที่ถูกฟังในแต่ละช่วงเวลา, วิธี GMM ทำนายเพลงที่จะถูกเลือกฟัง และใช้วิธี Forgetting curve สำหรับแนะนำเพลงใหม่	วิธีการที่นำเสนอให้ความถูกต้องดีกว่าการแนะนำเพลงแบบสุ่ม	จัดเก็บข้อมูลเอง

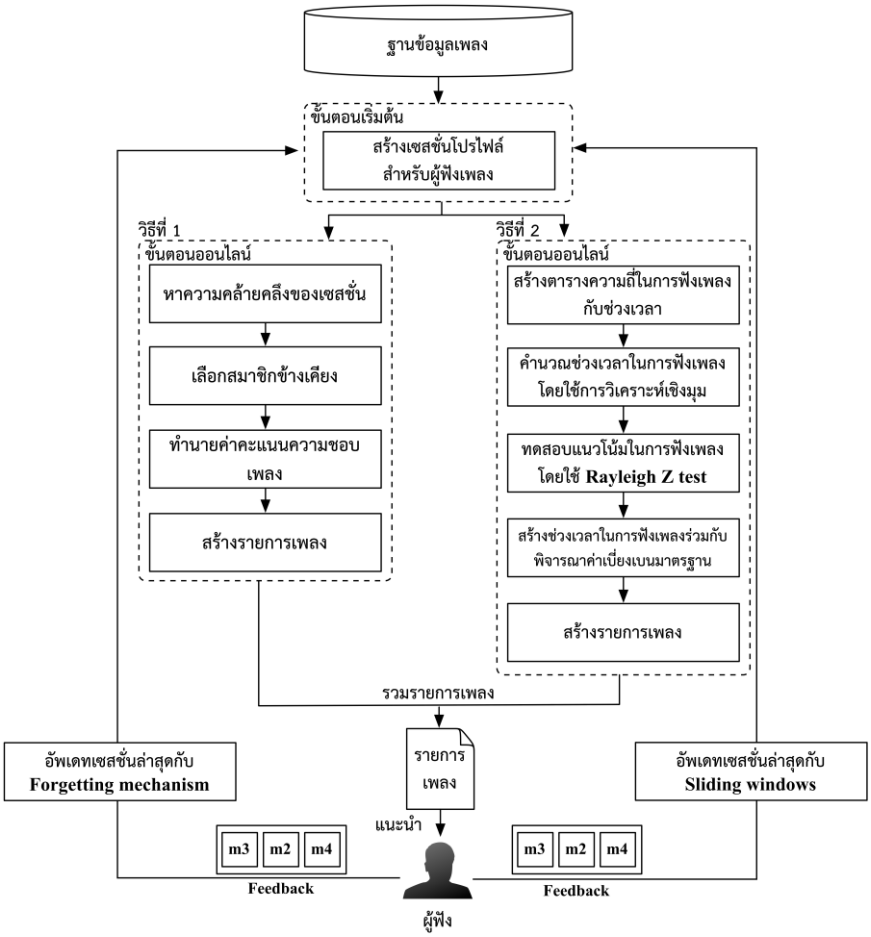
ตารางที่ 2-5 งานวิจัยที่เกี่ยวข้องกับระบบแนะนำแบบออนไลน์

งานวิจัยระบบแนะนำแบบออนไลน์	วัตถุประสงค์	วิธีการ	ผลสรุป	ฐานข้อมูล
Papagelis, M. et al., 2005	แก้ไขปัญหาการประมวลผลข้อมูลขนาดใหญ่ (Scalability problem)	Incremental Collaborative Filtering (ICF)	วิธีการ ICF สามารถคำนวณได้รวดเร็วกว่า CF แบบเดิม	จัดเก็บข้อมูลเอง
Miranda, C. & Jorge, A. M., 2008	แก้ไขปัญหาการประมวลผลข้อมูลขนาดใหญ่	Incremental Item based Collaborative Filtering	สามารถเพิ่มความถูกต้องได้ดีกว่าวิธี ICF และสามารถคำนวณได้รวดเร็ว	Economics และ E-learning
Yang, X. et al., 2012	แก้ไขปัญหา Sparsity และการประมวลผลข้อมูลขนาดใหญ่	วิธีการ link prediction ร่วมกับวิธี ICF	สามารถเพิ่มความถูกต้องและคำนวณได้รวดเร็วกว่าวิธี CF แบบเดิม	Movielens, Netflix
Vinagre, J. & Jorge, A. M., 2012	เพิ่มความถูกต้องของรายการเพลงและแก้ไขปัญหาการประมวลผลข้อมูลขนาดใหญ่	วิธีการ CF ร่วมกับ Forgetting mechanism	สามารถเพิ่มความถูกต้องและคำนวณได้รวดเร็วกว่าวิธี CF แบบเดิม	จัดเก็บข้อมูลเอง

บทที่ 3

วิธีการที่นำเสนอ

วิทยานิพนธ์นี้เป็นการต่อยอดวิธีการสร้างรายการเพลง Session-based Collaborative Filtering, SSCF (Park, S. E., et al, 2011) เพื่อปรับปรุงให้วิธีการสร้างรายการเพลงของวิธี SSCF ให้มีความถูกต้องและได้เพลงที่ตรงใจกับผู้ฟังเฉพาะบุคคลมากยิ่งขึ้น รวมทั้งแก้ไขปัญหาการประมวลผลข้อมูลที่มีปริมาณมาก (Scalability problem) ซึ่งสามารถประยุกต์ใช้กับการสร้างรายการเพลงแนะนำแบบออนไลน์ วิทยานิพนธ์นี้ได้นำเสนอวิธีการสร้างรายการเพลงที่ชื่อว่า “**Incremental Session-based Collaborative Filtering using Forgetting mechanism and Circular statistics**” โดยเรียกสั้นๆว่า “ISSCF” ซึ่งภาพรวมของขั้นตอนวิธีการที่นำเสนอ ISSCF แสดงดังภาพที่ 3-1



ภาพที่ 3-1 ภาพรวมวิธีการ ISSCF

จากภาพที่ 3-1 ขั้นตอนวิธีการ ISSCF ได้แบ่งเป็น 3 ขั้นตอน ได้แก่ 1. ขั้นตอนเริ่มต้น เป็นขั้นตอนการสร้างเซสชั่นโปรไฟล์สำหรับผู้ฟังเพลงแต่ละคน 2. ขั้นตอนการสร้างรายการเพลงออนไลน์ของทั้ง 2 วิธี และ 3. การรวมรายการเพลงที่จะแนะนำให้กับผู้ฟัง ดังนี้

3.1 ขั้นตอนเริ่มต้น

3.1.1 ฐานข้อมูลเพลง

- **ฐานข้อมูลเพลง Last.fm**

ฐานข้อมูลเพลงนี้ได้จัดเก็บประวัติการฟังเพลงจากเว็บไซต์ Last.fm (Oscar C., 2010) ซึ่งถูกรวบรวมตั้งแต่ปี 2005 ถึงปี 2009 ประกอบด้วยประวัติการฟังเพลงของผู้ฟังแต่ละคน ได้แก่ รหัสผู้ฟัง, เวลาที่ผู้ฟังฟังเพลง, รหัสศิลปิน, ชื่อศิลปิน, รหัสเพลงและชื่อเพลง ตามลำดับ ดังภาพที่ 3-2

รหัสผู้ฟัง	เวลา	รหัสศิลปิน	ศิลปิน	รหัสเพลง	เพลง
ผู้ฟัง 1	2009-01-01T09:00:10Z	art1	Don Moy	m1	Moy Or Less
ผู้ฟัง 1	2009-01-01T09:04:40Z	art1	Don Moy	m2	Contest Moy
ผู้ฟัง 1	2009-01-01T09:07:47Z	art3	Pogo	m3	Alice
...	...	...	...	...	...

ภาพที่ 3-2 ตัวอย่างฐานข้อมูลเพลง Last.fm

- **ฐานข้อมูลเพลง 30Music**

ฐานข้อมูลเพลง 30Music (Turrin, R. et al., 2015) จัดเก็บประวัติการฟังเพลงของผู้ฟังจากเว็บไซต์ Last.fm ระหว่างปี 2014 ถึงปี 2015 รวมถึงรายการเพลงของผู้ฟังและเซสชั่นการฟังเพลงของผู้ฟัง วิทยานิพนธ์นี้ได้สนใจถึงข้อมูลเซสชั่นการฟังเพลงประกอบด้วย รหัสผู้ฟัง, เวลาการฟังเพลง, รหัสเพลง, เวลาเริ่มเล่นและเวลาเล่นทั้งหมด ตามลำดับ แสดงดังภาพที่ 3-3

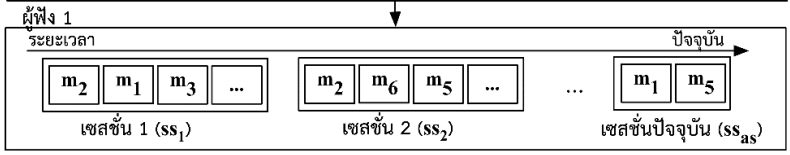
รหัสผู้ฟัง	เวลา	รหัสเพลง	เวลาเริ่มเล่น (s)	เวลาเล่นทั้งหมด (s)
44361	1390231051	4698874	0	121
44361	139023111	4698256	121	258
44361	1390231224	3298174	258	490
...	...	...	...	...

ภาพที่ 3-3 ตัวอย่างฐานข้อมูลเพลง 30Music

3.1.2 ขั้นตอนการสร้างเซสชั่นโปรไฟล์สำหรับผู้ฟังเพลง

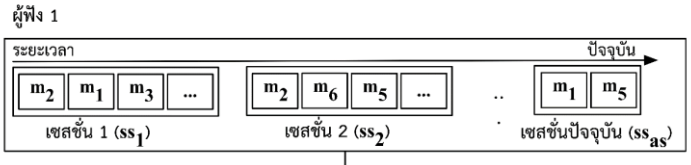
เนื่องจากวิทยานิพนธ์นี้ได้สนใจพฤติกรรมในการเลือกฟังเพลงของผู้ฟังที่มักจะมีการฟังที่ชื่นชอบซ้ำ และจากที่กล่าวไปในขั้นต้น พฤติกรรมของผู้ฟังเพลงมักมีการฟังอย่างต่อเนื่องและเป็นลำดับ ซึ่งเรียกลักษณะการฟังเพลงนี้ว่าเซสชั่น (Park, S. E., et al, 2011) (Dias, R. & Fonseca, M. J., 2013) ดังนั้นขั้นตอนนี้จึงได้พิจารณาเซสชั่นของผู้ฟังโดยการสร้างเซสชั่นโปรไฟล์ของแต่ละคน (session profiles) เมื่อผู้ฟังมีการฟังเพลงที่ต่อเนื่องกันแล้วหยุดการฟังเพลงในระยะเวลาหนึ่งจะถูกพิจารณาเป็น 1 เซสชั่น ตัวอย่างการสร้างเซสชั่นโปรไฟล์แสดงดังภาพที่ 3-4

รหัสผู้ฟัง	เวลา	รหัสศิลปิน	ศิลปิน	รหัสเพลง	เพลง
ผู้ฟัง 1	2009-01-01T09:00:10Z	art1	Don Moy	m_1	Moy Or Less
ผู้ฟัง 1	2009-01-01T09:04:40Z	art1	Don Moy	m_2	Contest Moy
ผู้ฟัง 1	2009-01-01T09:07:47Z	art3	Pogo	m_3	Alice
...	...	...	...	...	...



ภาพที่ 3-4 ตัวอย่างการสร้างเซสชั่นโปรไฟล์ของผู้ฟังในฐานข้อมูลเพลง Last.fm

ขั้นตอนถัดไปจะสร้างเมทริกซ์ความถี่ในการฟังเพลงจากเซสชั่นโปรไฟล์สำหรับแต่ละผู้ฟัง ที่ประกอบด้วย แถวคือเซสชั่นของผู้ฟังเพลง คอลัมน์คือเพลงที่ผู้ฟังได้ฟังในแต่ละเซสชั่น และข้อมูลภายในเซลล์คือความถี่ในการฟังเพลงแต่ละเซสชั่น แสดงดังภาพที่ 3-5



	m_1	m_2	m_3	m_4	m_5	m_6	...
ss_1	1	2	2	1	3	4	...
ss_2	2	2	1	-	3	3	...
ss_3	-	-	-	2	2	5	...
...	...	...	...	...	...	...	...
ss_{100}	1	2	3	3	2	-	...

ภาพที่ 3-5 ตัวอย่างการสร้างเมทริกซ์จากเซสชั่นโปรไฟล์ของผู้ฟัง 1

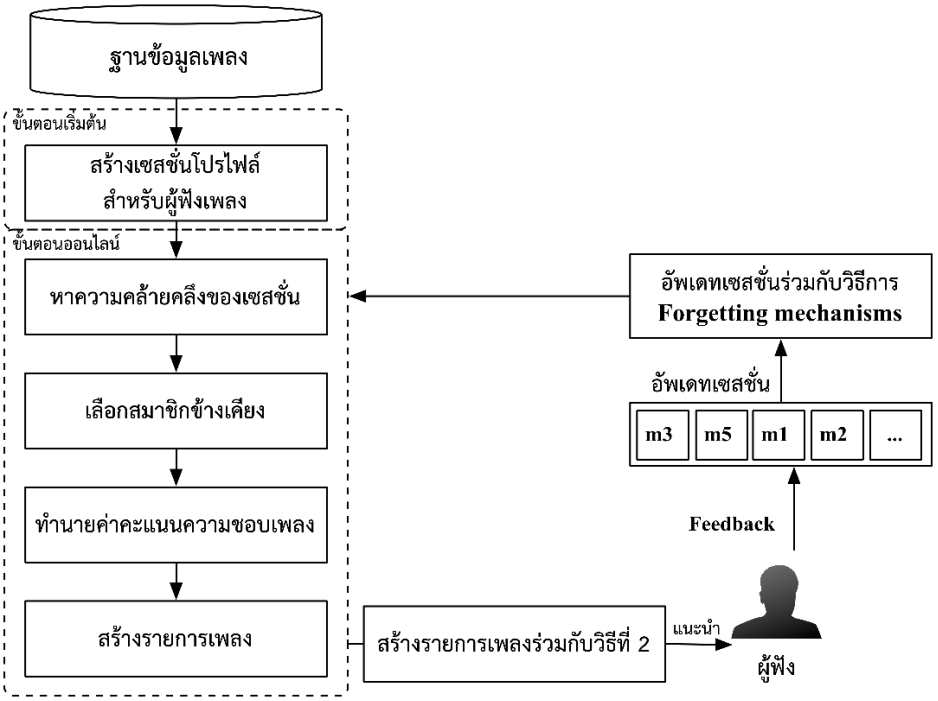
3.2 ขั้นตอนการสร้างรายการเพลงออนไลน์

- **วิธีที่ 1** จากสมมติฐานที่ว่าผู้ฟังน่าจะมีการเลือกฟังเพลงที่คล้ายกับเซสชั่นในอดีตเนื่องจากผู้ฟังเพลงมักจะมีพฤติกรรมในการฟังเพลงซ้ำอย่างต่อเนื่องเป็นช่วงหรือเซสชั่น และการเลือกฟังเพลงล่าสุดแสดงถึงความชอบของผู้ฟังขณะนั้น ดังนั้นวิธีการสร้างรายการเพลงจึงได้ประยุกต์ใช้วิธีการกรองข้อมูลร่วม ซึ่งพิจารณาความชอบการเลือกฟังเพลงในเซสชั่นปัจจุบันที่มีความคล้ายกับเซสชั่นในอดีต โดยให้ความสำคัญกับเซสชั่นล่าสุดของผู้ฟังเพลง
- **วิธีที่ 2** จากสมมติฐานที่ว่าเพลงบางเพลงที่ผู้ฟังชื่นชอบมักจะมีการฟังซ้ำเฉพาะช่วงเวลา ดังนั้นวิธีนี้จึงใช้วิธีการวิเคราะห์สถิติเชิงมุม (Circular Statistics) ซึ่งพิจารณาความชอบของผู้ฟังเฉพาะช่วงเวลาว่ามักจะชอบฟังเพลงช่วงเวลาใดและชอบฟังเพลงอะไรที่มีความแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติ

เมื่อผู้ฟังหยุดการฟังเพลงสักระยะเวลาหนึ่ง วิธีการทั้ง 2 จะนำเซสชั่นล่าสุดของผู้ฟังเป็นข้อมูลย้อนกลับ (feedback) ไปยังระบบเพื่อปรับปรุงเซสชั่นโปรไฟล์ให้ยังคงความชอบในปัจจุบันของผู้ฟังเพลงซึ่งน่าจะตรงกับความต้องการของผู้ฟังมากยิ่งขึ้น

3.2.1 การสร้างรายการเพลงแนะนำวิธีที่ 1

สำหรับขั้นตอนออนไลน์วิธี ISSCF วิธีที่ 1 ประกอบด้วยการหาความคล้ายของเซสชั่น ขั้นตอนการเลือกสมาชิกข้างเคียง การทำนายค่าคะแนนความชอบ การสร้างรายการเพลง จากนั้นจะรวมรายการเพลงเข้ากับรายการเพลงแนะนำในวิธีที่ 2 แสดงดังภาพที่ 3-6



ภาพที่ 3-6 ภาพรวมขั้นตอนวิธี ISSCF วิธีที่ 1

3.2.1.1 ขั้นตอนการหาความคล้ายคลึงของเซสชั่น

ขั้นตอนแรกของการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง คือการคำนวณความคล้ายคลึงของเซสชั่นปัจจุบันของผู้ฟังเพลง (SS_{as}) กับเซสชั่นในอดีตที่เก็บรวบรวมไว้ในขั้นตอนเริ่มต้น แสดงดังภาพที่ 3-7

ภาพที่ 3-7 ตัวอย่างการหาความคล้ายของเซสชั่นปัจจุบัน (SS_{as}) กับเซสชั่นในอดีตของผู้ฟัง

วิธีการ ISSCF ได้เลือกใช้วิธีการหาความคล้ายแบบโคไซน์ (Cosine similarity) สำหรับการพิจารณาความคล้ายของเซสชั่นในปัจจุบันกับเซสชั่นในอดีตเพื่อสร้างรายการเพลงแนะนำให้กับผู้ฟัง (Park, S. E., et al, 2011) แสดงดังสมการที่ (3.1) โดยที่ค่าความคล้ายที่เข้าใกล้ 1 แสดงถึงทั้งสองเซสชั่นมีความคล้ายกันมาก ถ้าค่าความคล้ายที่เข้าใกล้ 0 แสดงถึงทั้งสองเซสชั่นไม่มีความคล้ายกัน แบ่งได้เป็น 2 กรณี คือ

- **กรณีเมื่อผู้ฟังเข้าสู่ระบบ** สำหรับกรณีนี้จะยังไม่มีเซสชั่นการฟังเพลงเกิดขึ้น ทำให้เซสชั่นครั้งล่าสุดของผู้ฟังจะถูกพิจารณาเป็นเซสชั่นปัจจุบันของผู้ฟังเพลง (SS_{as}) ซึ่งจะนำไปหาความคล้ายกับเซสชั่นในอดีตแล้วสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังแต่ละบุคคล โดยในกรณีนี้จะถูกคำนวณไว้ในขั้นตอนออฟไลน์
- **กรณีเมื่อมีการฟังเพลงเกิดขึ้น** สำหรับกรณีนี้เมื่อผู้ฟังมีการเลือกฟังเพลงเกิดขึ้น เพลงที่ผู้ฟังเลือกฟังจะถูกพิจารณาเป็นเซสชั่นปัจจุบัน (SS_{as}) ของผู้ฟังซึ่งนำไปพิจารณาหาค่าความคล้ายแล้วสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังต่อไป โดยทั้ง 2 กรณีถ้าผู้ฟังมีการเลือกฟังเพลงที่ตรงกับรายการเพลง แสดงถึงผู้ฟังยังคงชื่นชอบรายการเพลงนั้น ดังนั้นรายการเพลงจะไม่ถูกสร้างใหม่จนกว่าผู้ฟังมีการเลือกฟังเพลงที่ไม่ตรงกับรายการเพลง ซึ่งสามารถช่วยลดการคำนวณของระบบได้

$$sim(ss_{as}, ss_v) = \frac{\sum_j (r_{ss_{as},m_j}, r_{ss_v,m_j})}{\sqrt{\sum_j (r_{ss_{as},m_j})^2}\sqrt{\sum_j (r_{ss_v,m_j})^2}} \quad (3.1)$$

โดยที่ $sim(ss_{as}, ss_v)$ คือ ค่าความคล้ายของเซสชั่นปัจจุบัน (ss_{as}) กับเซสชั่น (ss_v)

ss_{as} คือ เซสชั่นปัจจุบันของแต่ละผู้ฟัง

ss_v คือ เซสชั่น v ของผู้ฟัง

r_{ss_{as},m_j} คือ ความถี่ของเพลง m_j ในเซสชั่นปัจจุบัน

r_{ss_v,m_j} คือ ความถี่ของเพลง m_j ในเซสชั่น ss_v

จากตัวอย่างดังภาพที่ 3-8 แสดงให้เห็นว่าเซสชั่นปัจจุบันมีการฟังเพลงที่คล้ายกับเซสชั่นที่ 2 มากกว่าเซสชั่นอื่น

	SS_1	SS_2	SS_3		SS_{100}
ss_{as}	0.48	0.68	0.25	...	0.41

ภาพที่ 3-8 ตัวอย่างค่าความคล้ายของเซสชั่นปัจจุบันกับเซสชั่นในอดีตของผู้ฟัง

3.2.1.2 ขั้นตอนการเลือกสมาชิกข้างเคียง

สำหรับขั้นตอนการเลือกสมาชิกข้างเคียงของเซสชั่นที่มีความคล้ายกับเซสชั่นในปัจจุบันมากที่สุดนั้น สามารถแบ่งได้เป็น 2 วิธีการ คือ 1. การหาจำนวนเพื่อนบ้านที่ดีที่สุด (k-NN) โดยเลือกเซสชั่นที่มีความคล้ายกับเซสชั่นปัจจุบันมากที่สุด k เซสชั่นและ 2. การกำหนดเกณฑ์ความคล้าย (th) ซึ่งเลือกเซสชั่นที่มีความคล้ายกับเซสชั่นปัจจุบันมากที่สุดที่มีค่าความคล้ายมากกว่าค่าเกณฑ์ที่กำหนดไว้ ซึ่งมีความยืดหยุ่นมากกว่าวิธีการ k-NN เนื่องจากจำนวนของเซสชั่นใกล้เคียงมีความแตกต่างกันไปในแต่ละเพลงที่ผู้ฟังเลือกฟัง ดังภาพที่ 3-9

th=0.1	SS_1	SS_2	SS_{100}	SS_3	
ss_{as}	0.68	0.48	0.41	0.25	...

ภาพที่ 3-9 ตัวอย่างการพิจารณาเซสชั่นที่มีความคล้ายกับเซสชั่นปัจจุบันมากที่สุดที่ th เท่ากับ 0.1

3.2.1.3 ขั้นตอนการทำนายค่าคะแนนความชอบของเพลง

สำหรับขั้นตอนการทำนายค่าคะแนนความชอบของเพลงแต่ละเพลง วิธีการ ISSCF ได้พิจารณาจากความชอบในการเลือกฟังเพลงในปัจจุบัน ซึ่งมีคล้ายกับเซสชั่นในอดีตของผู้ฟัง จากนั้นทำนายค่าคะแนนความชอบของเพลงแต่ละเพลงที่คาดว่าผู้ฟังน่าจะเลือกฟังเป็นเพลงถัดไป ดังนั้นเพลงที่ผู้ฟังยังไม่ได้ฟังในเซสชั่นปัจจุบันสามารถที่จะถูกแนะนำได้ ซึ่งสามารถคำนวณได้จากสมการที่ (3.2) แสดงตัวอย่างค่าคะแนนความชอบของเพลงแต่ละเพลง ดังภาพที่ 3-10

$$p_{ss_{as},m_j} = \bar{r}_{ss_{as}} + \frac{\sum_{ss_v \in S} sim(ss_{as},ss_v)\left[r_{ss_v,m_j} - \bar{r}_{ss_v}\right]}{\sum_{ss_v \in S}|sim(ss_{as},ss_v)|} \quad (3.2)$$

โดยที่ p_{ss_{as},m_j} คือ ค่าการทำนายของเพลง m_j ที่เซสชั่นปัจจุบัน (ss_{as})
r_{ss_v,m_j} คือ ความถี่ของเพลง m_j ในเซสชั่น ss_v
$\bar{r}_{ss_{as}}$ คือ ค่าเฉลี่ยของการฟังเพลงที่เซสชั่นปัจจุบัน (ss_{as})
$\bar{r}_{ss_v}$ คือ ค่าเฉลี่ยของการฟังเพลงที่เซสชั่น ss_v

	m_1	m_2	m_3	m_4	m_5	m_6	
ss_{as}	0.35	0.84	0.69	0.74	1.34	2.1	...

ภาพที่ 3-10 ตัวอย่างค่าคะแนนความชอบของเพลงแต่ละเพลง

3.2.1.4 ขั้นตอนการสร้างรายการเพลงแนะนำ

สำหรับขั้นตอนการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง เมื่อคำนวณค่าคะแนนความชอบของเพลงแต่ละเพลงครบทุกเพลงแล้ว ขั้นตอนนี้จะเลือกเพลงที่คาดว่าผู้ฟังจะเลือกรับฟังเป็นเพลงถัดไปโดยการพิจารณาค่าคะแนนความชอบที่มีค่ามากที่สุด N เพลง (Top-N recommendation) โดยเรียงลำดับจากค่าการทำนายที่มากที่สุดจนถึงค่าการทำนายที่ต่ำที่สุด N ลำดับ แสดงตัวอย่างรายการเพลงแนะนำ 5 อันดับ (Top-5 recommendation) ที่ถูกแนะนำให้กับผู้ฟัง ดังภาพที่ 3-11

รายการแนะนำ 5 อันดับ	m_6	m_5	m_2	m_4	m_3
ss_{as}	2.1	1.34	0.84	0.74	0.69

ภาพที่ 3-11 ตัวอย่าง 5 อันดับรายการเพลงแนะนำ

3.2.1.5 ขั้นตอนการอัพเดทเซสชั่นของผู้ฟังเพลง

วิทยานิพนธ์นี้ได้จำลองลักษณะของการฟังเพลงของผู้ฟังในแบบออนไลน์ ซึ่งจะนำข้อมูลเพลงเข้ามาใหม่อยู่เสมอเพื่อให้สามารถประยุกต์ใช้ได้กับปัญหาจริง (Real world problem) ซึ่งแตกต่างจากงานวิจัยส่วนมากซึ่งมีลักษณะการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังแบบออฟไลน์ (Static setting) คือการใช้ข้อมูลการฟังเพลงของผู้ฟังทั้งหมดในการสร้างรายการเพลงเพียงรายการเดียว

เมื่อผู้ฟังหยุดฟังเพลงในระยะเวลาหนึ่ง สำหรับฐานข้อมูล Last.fm คือ 30 นาที (Park, S. E., et al, 2011) และฐานข้อมูล 30Music คือ 13 นาที (Turrin, R. et al., 2015) ดังนั้นวิธีการ ISSCF จะอัพเดทเซสชั่นล่าสุดของผู้ฟังเข้าสู่ระบบ เพื่อปรับปรุงเซสชั่นโปรไฟล์ของผู้ฟังเพลงซึ่งจะนำไปพิจารณาสร้างรายการเพลงแนะนำในช่วงเวลาถัดไป

ก่อนที่จะอัพเดทเซสชั่นล่าสุดเข้าสู่ระบบ วิธีการ ISSCF ได้ให้ความสำคัญกับเซสชั่นของผู้ฟังเพราะแสดงถึงความชอบของผู้ฟังในปัจจุบันโดยแบ่งเป็น 2 วิธีการ คือ

1) วิธีการ Sliding windows ซึ่งเป็นการลดความสำคัญของข้อมูลเก่าทันที (Abrupt Forgetting) โดยตัดเซสชั่นของผู้ฟังเพลงแบบเป็นลำดับในลักษณะโครงสร้างแบบ first-in-first-out (FIFO) ซึ่งกำหนดขนาดข้อมูล (sw) ที่จัดเก็บเซสชั่นของผู้ฟังเพลงล่าสุดก่อนสร้างเมทริกซ์ เมื่อเซสชั่นใหม่ของผู้ฟังถูกอัพเดทเข้าสู่ sw แล้วเซสชั่นที่เก่าที่สุดจะถูกตัดทิ้ง ดังภาพที่ 3-12 ข้อดีของวิธีการนี้คือข้อมูลการฟังเพลงของผู้ฟังจะไม่มีขนาดที่เพิ่มขึ้นตามเวลาทำให้มีการคำนวณที่รวดเร็วในวิธีการหาความคล้ายของเซสชั่นและวิธีการทำนายค่าคะแนนความชอบซึ่งช่วยแก้ปัญหาการประมวลผลข้อมูลที่มีปริมาณมาก

		m_1	m_2	m_3	m_4	m_5	m_6	...
ลบเซสชั่นเก่า →	ss_1	1	2	2	1	3	4	...
	...	...	...	...	...	...	...	...
	ss_t	-	4	-	2	2	5	...
	...	...	...	...	...	...	...	...
	ss_{100}	1	-	3	3	2	-	...
เพิ่มเซสชั่นใหม่ →	New ss	1	-	-	-	1	-	...

ภาพที่ 3-12 ตัวอย่างการอัพเดทเซสชั่นของวิธีการ Sliding windows

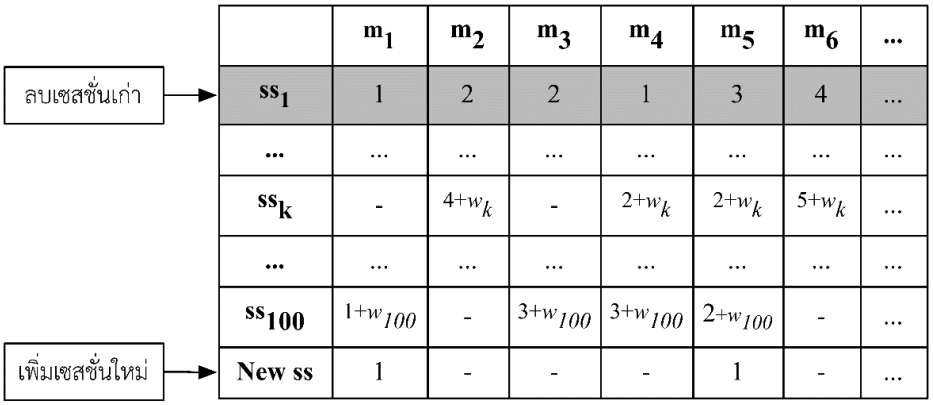
2) วิธีการ Forgetting Mechanisms ประกอบด้วยวิธีการ Sliding windows ร่วมกับวิธีการ Fading factors โดยให้ค่าน้ำหนักตามลำดับของเซสชั่น ซึ่งเซสชั่นล่าสุดของผู้ฟังเพลงจะมีค่าน้ำหนักที่มากกว่าเซสชั่นในอดีต วิทยานิพนธ์นี้ได้เลือกใช้วิธีการ Exponential time decay function สำหรับการให้ความสำคัญของข้อมูล สามารถคำนวณโดยใช้สมการที่ (3.3) แสดงตัวอย่างการอัพเดทเซสชั่นร่วมกับวิธีการ Forgetting Mechanisms ดังภาพที่ 3-13

$$w_t = e^{-\alpha t} \tag{3.3}$$

โดยที่ w_t คือ ค่าน้ำหนักของเซสชั่นที่ t

α คือ ค่าพารามิเตอร์ในการควบคุมความเร็วในการลดความสำคัญ

t คือ ลำดับของเซสชั่นที่ถูกฟัง

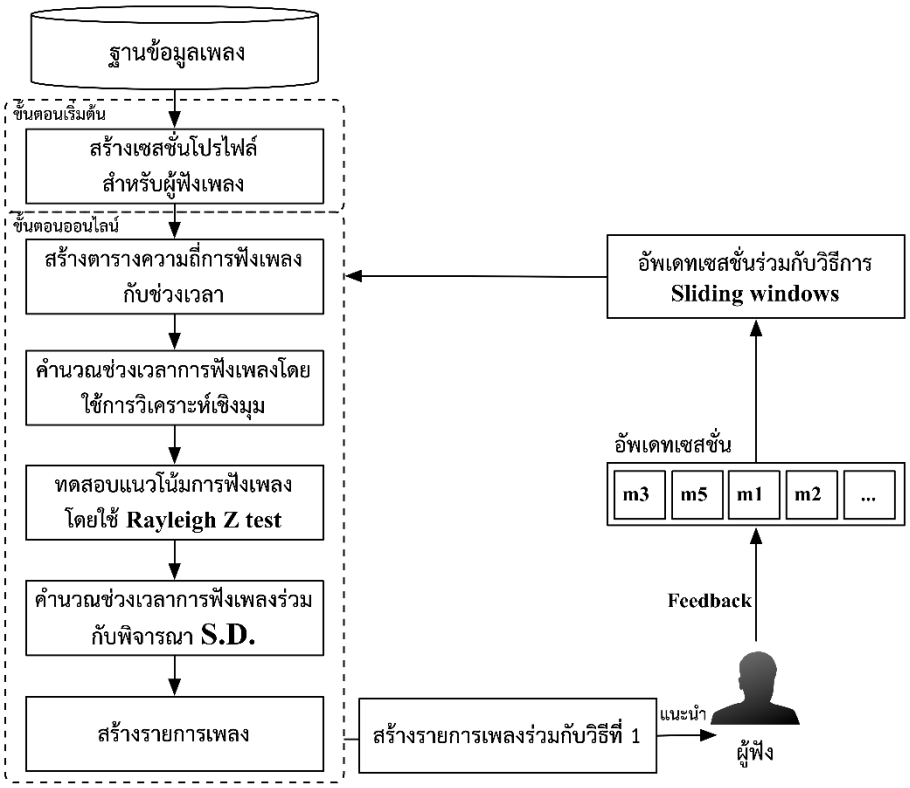


	m_1	m_2	m_3	m_4	m_5	m_6	...
ss_1	1	2	2	1	3	4	...
...	...	...	...	...	...	...	...
ss_k	-	$4+w_k$	-	$2+w_k$	$2+w_k$	$5+w_k$	...
...	...	...	...	...	...	...	...
ss_{100}	$1+w_{100}$	-	$3+w_{100}$	$3+w_{100}$	$2+w_{100}$	-	...
New ss	1	-	-	-	1	-	...

ภาพที่ 3-13 ตัวอย่างการอัพเดทเซสชั่นของวิธีการ Forgetting Mechanisms

3.2.2 การสร้างรายการเพลงแนะนำวิธีที่ 2

เนื่องจากการสร้างรายการเพลงแนะนำวิธีที่ 1 ได้พิจารณาความชอบจากประวัติการฟังเพลงจากเซสชั่นโปรไฟล์ของผู้ฟังแต่ละคน แต่ไม่ได้พิจารณาถึงความชอบในการเลือกฟังเพลงตามช่วงเวลาของผู้ฟังซึ่งผู้ฟังน่าจะมีการเลือกฟังเพลงที่ชื่นชอบซ้ำในบางช่วงเวลา (Herrera, P. et al., 2010) (Park, C. H. & Kahng, M., 2010) (Dias, R. & Fonseca, M.J., 2013) ดังนั้นวิธีการ ISSCF วิธีที่ 2 จึงได้นำเสนอวิธีการสร้างรายการเพลงแนะนำ โดยการวิเคราะห์ว่าผู้ฟังชอบฟังเพลงใดเฉพาะช่วงเวลาซึ่งมีความแตกต่างจากช่วงเวลาอื่น ซึ่งใช้การวิเคราะห์สถิติเชิงมุมโดยพิจารณาเพลงที่มักจะถูกฟังซ้ำอย่างต่อเนื่องเฉพาะช่วงเวลาซึ่งมีความแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติ แสดงภาพรวมวิธีการ ISSCF วิธีที่ 2 ดังภาพที่ 3-14



ภาพที่ 3-14 ภาพรวมของวิธีการ ISSCF วิธีที่ 2

3.2.2.1 ขั้นตอนการสร้างตารางความถี่ในการฟังเพลงกับช่วงเวลา

ขั้นตอนนี้เป็นการนำเซสชั่นโปรไฟล์แต่ละคนมาสร้างเป็นตารางสำหรับเก็บความถี่ของเพลงกับช่วงเวลา ซึ่งเพลงที่จะพิจารณาในตารางความถี่จะต้องมีการฟังมากกว่า 6 ครั้ง ซึ่งเป็นเงื่อนไขของ Reyleigh Z-test ดังตารางที่ 3-1 เป็นตารางความถี่ในการฟังเพลงกับช่วงเวลา เช่น ผู้ฟังที่ 1 ฟังเพลง m_1 ในช่วงเวลา 10 นาฬิกา 1 ครั้ง เวลา 11 นาฬิกา 3 ครั้ง เวลา 12 นาฬิกา 2 ครั้ง เวลา 13 นาฬิกา 1 ครั้งและเวลา 14 นาฬิกา 1 ครั้ง

ตารางที่ 3-1 ตัวอย่างตารางความถี่ในการฟังเพลงกับช่วงเวลาของผู้ฟังที่ 1

เพลง	เวลาที่ฟังเพลง (นาฬิกา)
m_1	10, 11, 11, 11, 12, 12, 13, 14
m_2	9, 9, 10, 11, 9, 10, 9, 10
m_3	23, 1, 4, 11, 18, 19
m_4	13, 12, 15, 14, 11, 11, 18, 18, 14
m_5	12, 13, 14, 11, 10, 11, 11, 8

3.2.2.2 ขั้นตอนการคำนวณช่วงเวลาในการฟังเพลงโดยใช้การวิเคราะห์สถิติเชิงมุม

เมื่อสร้างตารางความถี่ในการฟังเพลงเสร็จสิ้น ขั้นตอนนี้จะพิจารณารูปแบบหรือพฤติกรรมในการฟังเพลงเฉพาะช่วงเวลา โดยใช้การวิเคราะห์สถิติเชิงมุม (Circular Statistics) ซึ่งมีการคำนวณข้อมูลการฟังเพลงของผู้ฟังในรูปแบบเชิงมุม เริ่มต้นขั้นตอนนี้จะแปลงข้อมูลเวลาเป็นข้อมูลเชิงมุมโดยวิธีการ ISSCF ได้พิจารณาช่วงเวลาในรอบวัน (24 ชม.) ซึ่งสามารถคำนวณด้วยสมการที่ (3.4) จากตารางความถี่ในการฟังเพลงกับช่วงเวลาที่ 3-1 จะถูกแปลงเป็นข้อมูลเชิงมุม ดังตารางที่ 3-2

$$\alpha^\circ = \frac{(360^\circ)(X)}{24} \tag{3.4}$$

โดยที่ α° คือ มุมในเชิงดีกรี (degree)

X คือ เวลา (นาฬิกา) ของเพลงแต่ละเพลงที่ถูกฟัง

ตารางที่ 3-2 ตัวอย่างตารางความถี่ข้อมูลเวลาเชิงดีกรี

เพลง	เวลาที่ฟังเพลง (ดีกรี)
m_1	150°, 165°, 165°, 165°, 180°, 180°, 195°, 210°
m_2	135°, 135°, 150°, 165°, 135°, 150°, 135°, 150°
m_3	345°, 15°, 60°, 165°, 270°, 285°
m_4	195°, 180°, 225°, 210°, 165°, 165°, 270°, 270°, 210°
m_5	180°, 195°, 210°, 165°, 150°, 165°, 165°, 120°

จากตารางที่ 3-2 จะถูกนำมาคำนวณค่าเฉลี่ยของเวกเตอร์ (r) ในเชิงเรเดียน (radians) ดังสมการที่ (3.7) สำหรับการฟังเพลงแต่ละเพลง ซึ่งเป็นการวัดค่าการกระจายเชิงมุมของข้อมูล คล้ายกับการหาค่าเบี่ยงเบนมาตรฐาน แต่ค่าเฉลี่ยเวกเตอร์มีค่าในช่วง 0-1 โดยค่า 0 แสดงถึงผู้ฟังมีการฟังสม่ำเสมอในทุกช่วงเวลาและค่า 1 แสดงถึงข้อมูลการฟังเพลงมีแนวโน้มการฟังเพลงในช่วงเวลาเดียว โดยพิจารณาผลรวมค่าเฉลี่ย sin ดังสมการที่ (3.5) และผลรวมค่าเฉลี่ย cos ดังสมการที่ (3.6) แสดงตัวอย่างค่าเฉลี่ยเวกเตอร์ของแต่ละเพลง ดังตารางที่ 3-3

$$Y = \frac{\sum_{i=1}^{n} sin_{\alpha_i^c}}{n} \tag{3.5}$$

$$X = \frac{\sum_{i=1}^{n} cos_{\alpha_i^c}}{n} \tag{3.6}$$

$$r = \sqrt{X^2 + Y^2} \tag{3.7}$$

โดยที่ Y คือ ค่าเฉลี่ยเชิงมุม sin
X คือ ค่าเฉลี่ยเชิงมุม cos
n คือ จำนวนครั้งของการฟังเพลง
r คือ ค่าเฉลี่ยเวกเตอร์

ตารางที่ 3-3 ตัวอย่างค่าเฉลี่ยเวกเตอร์ของแต่ละเพลง

เพลง	ค่าเฉลี่ยเวกเตอร์ (r)
m_1	0.95
m_2	0.98
m_3	0.32
m_4	0.80
m_5	0.90

จากตารางที่ 3-3 เพลงแต่ละเพลงของผู้ฟังคนที่ 1 จะถูกคำนวณค่าแนวโน้มเฉลี่ยในการฟังเพลงของผู้ฟัง (θ_r) ซึ่งพิจารณาจากค่าเฉลี่ยเวกเตอร์ร่วมกับค่าเฉลี่ยเชิงมุม sin และ cos สามารถคำนวณได้จากสมการที่ (3.8)

$$\theta_r = tan^{-1}\left(\frac{sin\,\bar{\alpha}}{\cos\bar{\alpha}}\right) \tag{3.8}$$

โดยที่ $sin\,\bar{\alpha}$ คือ $\frac{Y}{r}$ และ $\cos\bar{\alpha}$ คือ $\frac{X}{r}$
จากนั้นพิจารณาควอแรนท์ (quadrant) ของมุมในเชิงดีกรี ดังนี้

$$\theta_r = \begin{cases} 180 - \theta_r \ ; sin+ \ , cos - \\ 180 + \theta_r \ ; sin- \ , cos - \\ 360 - \theta_r \ ; sin- \ , cos + \end{cases}$$

จากขั้นตอนนี้จะได้แนวโน้มการฟังเพลงของผู้ฟังแต่ละเพลง แสดงตัวอย่างแนวโน้มการฟังเพลงของผู้ฟังเชิงดีกรี ดังตารางที่ 3-4

ตารางที่ 3-4 ตัวอย่างแนวโน้มการฟังเพลงของผู้ฟังเชิงดีกรี

เพลง	แนวโน้มในการฟังเพลง (ดีกรี)
m_1	177.76°
m_2	144.34°
m_3	334°
m_4	208.55°
m_5	168.97°

จากตารางที่ 3-4 ข้อมูลแนวโน้มการฟังเพลงของผู้ฟังที่ 1 จะถูกแปลงกลับเป็นข้อมูลเชิงเวลา ซึ่งสามารถคำนวณได้จากสมการที่ 3.4 แสดงตัวอย่างแนวโน้มการฟังเพลงของผู้ฟัง ดังตารางที่ 3-5

ตารางที่ 3-5 ตัวอย่างแนวโน้มการฟังเพลงของผู้ฟัง

เพลง	แนวโน้มในการฟังเพลง (นาฬิกา)
m_1	11:00
m_2	9:00
m_3	22:00
m_4	13:00
m_5	11:00

3.2.2.3 ขั้นตอนการทดสอบแนวโน้มในการฟังเพลงโดยใช้ Rayleigh Z-test

ขั้นตอนนี้จะทดสอบแนวโน้มในการฟังเพลงเพื่อพิจารณาช่วงเวลาที่ผู้ฟังชอบฟังเพลงใด ๆ ซึ่งมีความแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติโดยใช้วิธีการ Rayleigh Z-test โดยการพิจารณาถึงค่าเฉลี่ยเวกเตอร์ (r) และจำนวนการฟังเพลงของผู้ฟัง (n) แสดงดังสมการที่ (3.9)

$$Z = nr^2 \tag{3.9}$$

เมื่อทดสอบแนวโน้มในการฟังเพลงโดยใช้ Rayleigh Z-test จะได้ค่า Z ของเพลงแต่ละเพลง ดังตารางที่ 3-6

ตารางที่ 3-6 แสดงค่า Z ที่ผ่านการทดสอบ Rayleigh Z-test

เพลง	ค่า Z
m_1	6.29
m_2	7.74
m_3	0.61
m_4	5.78
m_5	6.53

สำหรับการทดสอบนัยสำคัญทางสถิติได้พิจารณาสมมติฐานทางสถิติ ดังนี้

H_0 : ช่วงเวลาในการฟังเพลงของผู้ฟังไม่มีความแตกต่างจากช่วงเวลาอื่น

H_1 : ช่วงเวลาในการฟังเพลงของผู้ฟังมีความแตกต่างจากช่วงเวลาอื่น

ถ้าผู้ฟังมีการเลือกฟังเพลงเดิมเฉพาะช่วงเวลาบ่อยครั้งซึ่งแสดงถึงเป็นเพลงที่ผู้ฟังชื่นชอบ ค่า Z จะมีค่ามาก ถ้าผู้ฟังมีการฟังเพลงเดิมบ่อยครั้งแต่ฟังในหลายช่วงเวลา ค่า Z จะมีค่าที่น้อยซึ่งจะส่งผลให้เพลงนั้นไม่ถูกแนะนำ โดยการพิจารณาจะเปรียบเทียบกับตารางทางสถิติ Rayleigh Z-test (Zar, J.H., 1984) แสดงตัวอย่างการคำนวณ ดังนี้

ตัวอย่างเพลง m_1 มีค่าเฉลี่ยเวกเตอร์ (r) คือ 0.95 และมีการฟังเพลง m_1 ทั้งหมด 8 ครั้ง จะได้ $Z = 8(0.95)^2 = 7.25$ ที่ $p - value_{0.05} = 2.865$

ดังนั้น $7.25 > 2.865$ จึงปฏิเสธ H_0 ยอมรับ H_1 แสดงว่าแนวโน้มการฟังเพลง m_1 ในการฟังช่วงเวลา 11 นาฬิกามีความแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติ

ตัวอย่างเพลง m_3 มีค่าเฉลี่ยเวกเตอร์ (r) คือ 0.32 และมีการฟังเพลงทั้งหมด 6 ครั้ง จะได้ $Z = 6(0.32)^2 = 0.61$ ที่ $p - value_{0.05} = 2.865$

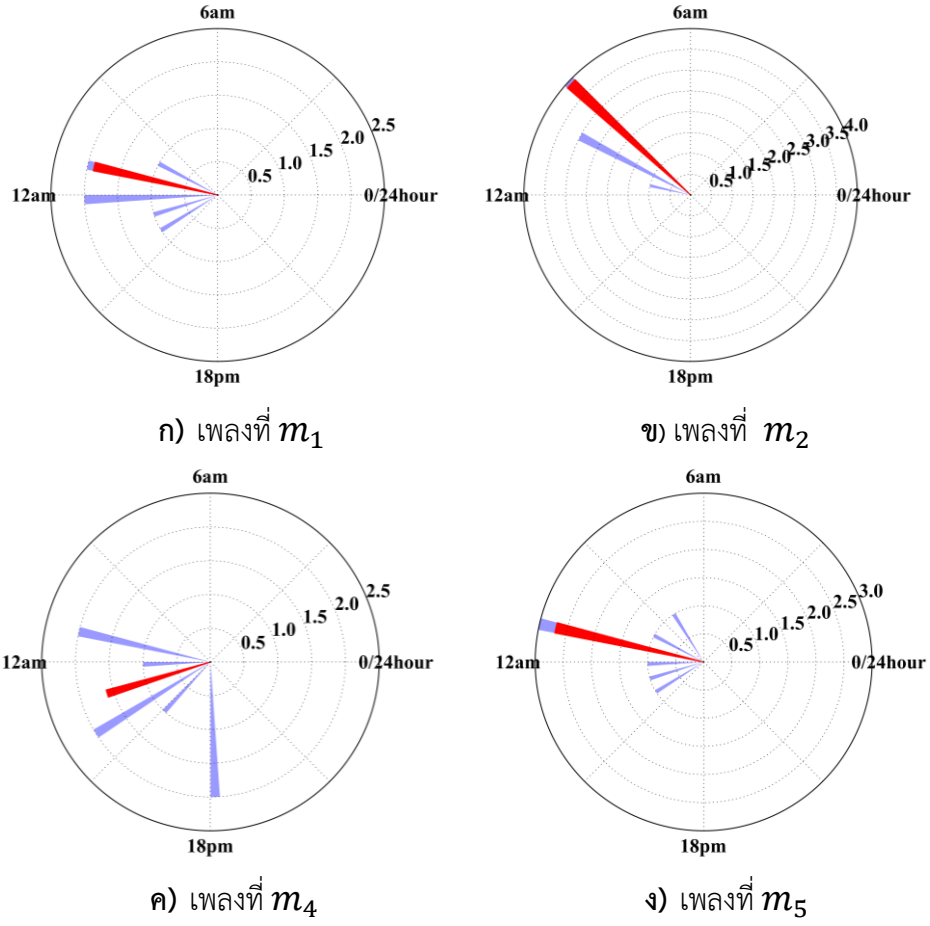
ดังนั้น $0.61 < 2.865$ จึงยอมรับ H_0 แสดงว่าแนวโน้มการฟังเพลง m_3 ไม่มีความแตกต่างจากช่วงเวลาอื่น ซึ่งทำให้เพลง m_3 ไม่ถูกแนะนำให้กับผู้ฟังในช่วงเวลานั้น

จากตารางที่ 3-5 เมื่อผ่านการพิจารณาแนวโน้มการฟังเพลงโดยใช้ Rayleigh Z-test เพลงที่ไม่ผ่านการทดสอบจะถูกตัดทิ้ง แสดงแนวโน้มการฟังเพลงหลังผ่านวิธีการทดสอบ Rayleigh Z-test ดังตารางที่ 3-7

ตารางที่ 3-7 ตัวอย่างแนวโน้มการฟังเพลงหลังผ่านวิธีการทดสอบแนวโน้มโดยใช้ Rayleigh Z-test

เพลง	แนวโน้มในการฟังเพลง (นาฬิกา)
m_1	11:00
m_2	9:00
m_4	13:00
m_5	11:00

ภาพที่ 3-15 แสดงให้เห็นถึงผู้ฟังมีการเลือกฟังเพลงเดิมในช่วงเวลาเฉพาะ โดยเส้นสีดำแสดงถึงแนวโน้มการฟังเพลงของผู้ฟังต่อเพลงนั้น ถ้าผู้ฟังมีการฟังเพลงในช่วงเวลานั้นมากส่งผลให้ค่า Z มีค่ามาก จากการตั้งสมมติฐานทางสถิติโดยใช้การทดสอบ Rayleigh Z-test จึงปฏิเสธ H_0 ยอมรับ H_1 เพลงนั้นจึงถูกแนะนำไปยังผู้ฟังคนที่ 1 เฉพาะช่วงเวลา สำหรับเส้นสีเทาแสดงถึงเพลงที่ผู้ฟังเลือกฟังตามช่วงเวลาต่าง ๆ

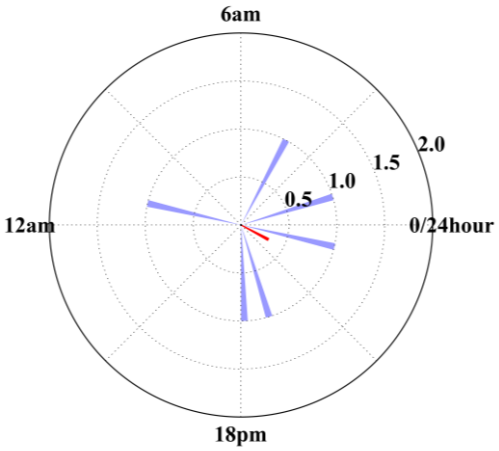


ก) เพลงที่ m_1 ข) เพลงที่ m_2

ค) เพลงที่ m_4 ง) เพลงที่ m_5

ภาพที่ 3-15 ตัวอย่างแนวโน้มในการฟังเพลงแต่ละเพลงของผู้ฟังที่ 1 ซึ่งแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติ

จากภาพที่ 3-16 แสดงตัวอย่างการฟังเพลง m_3 ของผู้ฟังที่ 1 ซึ่งไม่มีความแตกต่างจากช่วงเวลาอื่น เนื่องจากผู้ฟังมีการเลือกฟังเพลงในหลายช่วงเวลาทำให้แนวโน้มในการฟังเพลงมีค่าที่น้อยหรือมีการฟังเพลงที่ไม่มีทิศทางจึงไม่สามารถพิจารณาช่วงเวลาเฉพาะของการฟังเพลงได้



ภาพที่ 3-16 ตัวอย่างแนวโน้มการฟังเพลง m_3 ของผู้ฟังคนที่ 1

3.2.2.4 สร้างตารางช่วงเวลาในการฟังเพลงร่วมกับพิจารณา S.D.

จากรายการเพลงที่ได้จากขั้นตอนการทดสอบแนวโน้มช่วงเวลาในการฟังเพลง แสดงให้เห็นถึงผลลัพธ์ที่ได้คือชั่วโมงการฟังเพลงสำหรับหนึ่งเพลง แต่การฟังเพลงของผู้ฟังมักจะมีการเลือกฟังในช่วงเวลาที่ใกล้เคียงจากแนวโน้วการฟังเพลง ดังนั้นในขั้นตอนการสร้างรายการเพลงจึงคำนวณค่าเบี่ยงเบนมาตรฐาน ($S.D.$) ดังสมการที่ (3.10) ซึ่งจะแนะนำเพลงโดยพิจารณาจากค่า $S.D.$ โดยคำนวณชั่วโมงก่อนหน้าและหลังจากชั่วโมงแนวโน้มในการฟังเพลงของผู้ฟัง เช่น เมื่อคำนวณจากการวิเคราะห์สถิติเชิงมุมแล้วทราบว่าผู้ฟังชอบฟังเพลง m_1 เมื่อเวลา 11.00 นาฬิกาแล้วค่า $S.D.$ คือ 1 ดังนั้นวิธีการนี้จะแนะนำเพลงในเวลา 10.00 และ 12.00 นาฬิกาด้วย เป็นต้น แสดงดังตารางที่ 3-8 ซึ่งจะถูกเพิ่มไปยังตารางช่วงเวลาการฟังเพลงเพื่อสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังต่อไป

$$S.D._{m_j} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{3.10}$$

โดยที่ $S.D._{m_j}$ คือ ค่าเบี่ยงเบนมาตรฐานของเพลง m_j

x_i คือ เวลาการฟังเพลงของผู้ฟังในเพลง m_j

$\bar{x}$ คือ ค่าเฉลี่ยในการฟังเพลงของผู้ฟังในเพลง m_j

n คือ จำนวนชั่วโมงทั้งหมดในการฟังเพลง m_j

ตารางที่ 3-8 ตารางเวลาของเพลงที่ผู้ฟังชื่นชอบสำหรับผู้ฟังแต่ละบุคคล

เวลา	เพลง
9.00	m_2
10.00	m_1, m_5
11.00	m_1, m_4, m_5
12.00	m_1, m_4, m_5
13.00	$m_4,$
14.00	m_4
15.00	m_4

3.2.2.5 ขั้นตอนการสร้างรายการเพลงแนะนำ

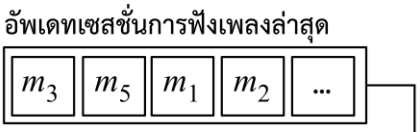
สำหรับการสร้างรายการเพลงสำหรับแต่ละผู้ฟังที่มีพฤติกรรมในการฟังเฉพาะช่วงเวลา เมื่อผู้ฟังเข้าสู่ระบบในช่วงเวลาที่ตรงกับช่วงเวลาที่ได้จากการวิเคราะห์สถิติเชิงมุม ดังนั้นขั้นตอนนี้จะสร้างรายการเพลงโดยพิจารณาอันดับของเพลงจากค่า Z ที่ได้จากขั้นตอนการทดสอบแนวโน้มในการฟังเพลงโดยใช้ Rayleigh Z-test แสดงตัวอย่างรายการเพลงแนะนำ 3 อันดับแรกที่เวลา 12 นาฬิกา ดังตารางที่ 3-9

ตารางที่ 3-9 ตัวอย่างรายการเพลง 3 อันดับแรกที่เวลา 12 นาฬิกา

อันดับ	รายการเพลงแนะนำในช่วงเวลา 12 นาฬิกา
1	m_5
2	m_1
3	m_4

3.2.2.6 ขั้นตอนการอัพเดทเซสชั่นร่วมกับวิธีการ Sliding windows

วิธีการ ISSCF จะอัพเดทเซสชั่นล่าสุดของผู้ฟังเพื่อปรับปรุงตารางความถี่ของการฟังเพลงโดยใช้วิธีการ Sliding windows โดยวิธีการนี้จะพิจารณาเพลงที่ถูกฟังในรอบ N วันล่าสุดร่วมกับเพลงจะต้องมีการฟังที่มากกว่า k วันจาก N วัน เพื่อตัดการพิจารณาเพลงที่ผู้ฟังไม่ได้ชื่นชอบ ซึ่งมีการฟังเพียงแค่ครั้งเดียวหรือไม่มีการฟังที่ต่อเนื่อง ตัวอย่างการอัพเดทเซสชั่นล่าสุดเข้าสู่ตารางความถี่ของผู้ฟังเพลง แสดงดังภาพที่ 3-19



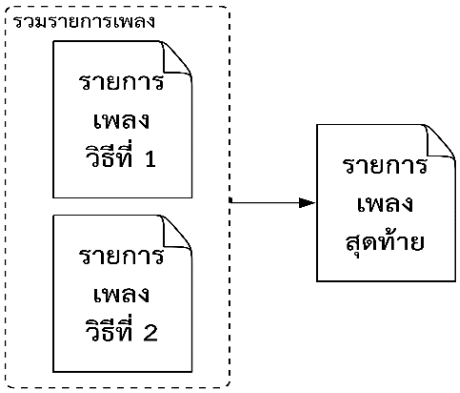
N=14 วัน, k=7 วัน

เพลง	เวลาที่ฟังเพลง (นาฬิกา)
m_1	10, 11, 11, 11, 12, 12, 13, 14
m_2	9, 9, 10, 11, 9, 10, 9, 10
m_3	23, 1, 4, 11, 18, 19
m_4	13, 12, 15, 14, 11, 11, 18, 18, 14
m_5	12, 13, 14, 11, 10, 11, 11, 8

ภาพที่ 3-19 ตัวอย่างการอัพเดทเซสชั่นล่าสุดเข้าสู่ตารางความถี่ของผู้ฟังเพลงที่เพลงนั้นมีการฟังมากกว่า 7 วันจาก 14 วัน

3.3 ขั้นตอนการรวมรายการเพลงที่จะแนะนำให้กับผู้ฟัง

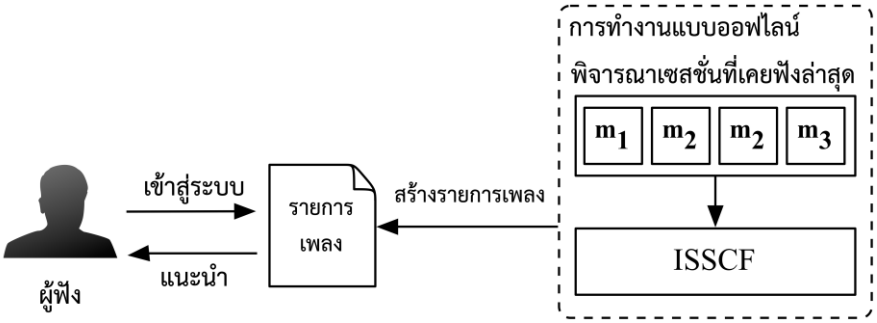
จากวิธีการสร้างรายการเพลงแนะนำ ISSCF ทั้ง 2 วิธีการ ในวิธีที่ 1 จะเลือกเพลงซึ่งมีค่าคะแนนความชอบมากที่สุด k เพลงและวิธีที่ 2 จะเลือกเพลงที่มีค่า Z มากที่สุดจากขั้นตอนการทดสอบแนวโน้มในการฟังเพลงโดยใช้ Rayleigh Z-test k เพลง แล้วรวมรายการเพลงที่จะแนะนำให้กับผู้ฟังทั้งสองขั้นตอนวิธี (Cascade) เป็นรายการเพลงสุดท้ายเพื่อแนะนำให้กับผู้ฟัง ดังภาพที่ 3-20 โดยการแนะนำเพลงแบ่งได้เป็น 2 กรณี ดังนี้



ภาพที่ 3-20 ตัวอย่างการรวมรายการเพลงที่จะแนะนำให้กับผู้ฟัง

3.3.1 เมื่อผู้ฟังเข้าสู่ระบบ

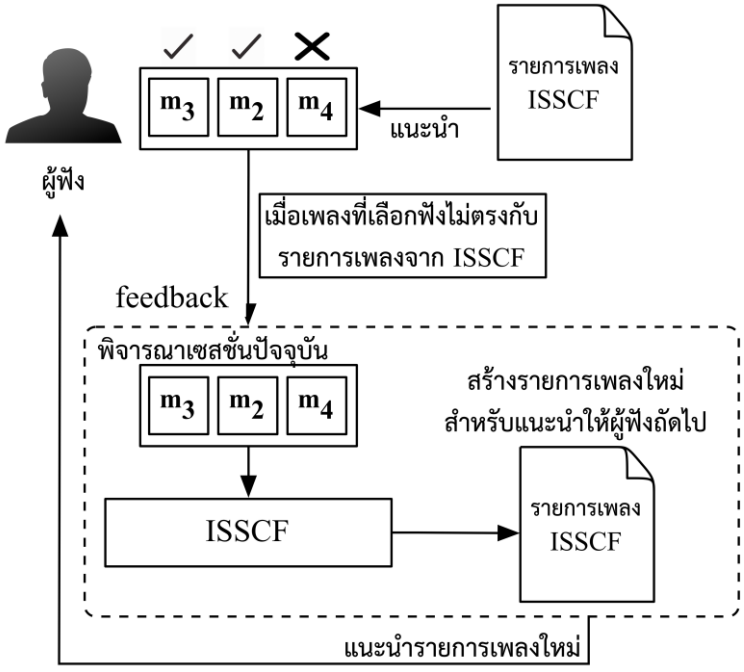
กรณีนี้รายการเพลงแนะนำจะถูกสร้างจากเซสชั่นล่าสุดของผู้ฟังแล้วสร้างรายการเพลงในแบบออฟไลน์ เมื่อผู้ฟังเข้าสู่ระบบจะใช้รายการเพลงที่เตรียมไว้แนะนำให้กับผู้ฟังในครั้งแรก แสดงดังภาพที่ 3-21



ภาพที่ 3-21 การสร้างรายการเพลงแนะนำเมื่อผู้ฟังเข้าสู่ระบบ

3.3.2 เมื่อผู้ฟังเลือกฟังเพลง

เมื่อผู้ฟังมีการฟังเพลงถัดไปที่ตรงกับรายการเพลงจะไม่มีการสร้างรายการเพลงใหม่เพราะผู้ฟังยังคงความชอบของรายการเพลงนั้นอยู่ กรณีที่ผู้ฟังเลือกฟังเพลงถัดไปไม่ตรงกับรายการเพลงจะสร้างรายการเพลงใหม่ โดยพิจารณาจากเซสชั่นปัจจุบัน (เพลงแรกเมื่อเริ่มเซสชั่นจนถึงเพลงปัจจุบันของผู้ฟังเพลง) เทียบกับเซสชั่นในอดีตของผู้ฟังเพลง แสดงดังภาพที่ 3-22



ภาพที่ 3-22 การสร้างรายการเพลงแนะนำเมื่อผู้ฟังเลือกฟังเพลง

บทที่ 4

ผลการดำเนินงาน

ในบทนี้จะกล่าวถึงผลการดำเนินงานโดยการวัดประสิทธิภาพความถูกต้องและเวลาที่ใช้ในการสร้างรายการเพลงแนะนำของวิธีการที่นำเสนอ Incremental Session based Collaborative Filtering (ISSCF) เปรียบเทียบกับวิธีการ Session Based Collaborative Filtering (SSCF) (Park, S. E., et al, 2011) ใน 2 ฐานข้อมูลเพลง คือ ฐานข้อมูลเพลง Last.fm และฐานข้อมูลเพลง 30Music โดยใช้วิธีการวัดประสิทธิภาพความถูกต้อง HitRatio (HR@n) สำหรับพิจารณาความถูกต้องของรายการเพลงกับเพลงที่ผู้ฟังเลือกฟังถัดไปและวิธีการวัดประสิทธิภาพ Precision สำหรับพิจารณาความถูกต้องของรายการเพลงแนะนำ โดยรายละเอียดของผลการดำเนินงานแสดงดังต่อไปนี้

4.1) ข้อมูลที่ใช้ในการทดลอง

4.2) การออกแบบการทดลองและวิธีการวัดประสิทธิภาพความถูกต้อง

4.3) ผลการทดลองจากฐานข้อมูลเพลง 30Music

4.4) ผลการทดลองจากฐานข้อมูลเพลง Last.fm

4.1 ข้อมูลที่ใช้ในการทดลอง

วิทยานิพนธ์นี้ได้เปรียบเทียบผลการทดลองบนฐานข้อมูลเพลงจาก 2 ฐานข้อมูล คือ ฐานข้อมูลเพลง Last.fm และฐานข้อมูลเพลง 30Music สำหรับฐานข้อมูลเพลง Last.fm จัดเก็บประวัติการฟังเพลงจากเว็บไซต์ Last.fm (Celma, O., 2010) ซึ่งเป็นเว็บไซต์เพลงออนไลน์และมีระบบแนะนำเพลงสำหรับผู้ฟัง ซึ่งถูกรวบรวมตั้งแต่ปี 2005 ถึงปี 2009 โดยได้คัดเลือกผู้ฟังที่มีการฟังมากกว่า 400 เซสชั่นจากผู้ฟังทั้งหมดเพื่อศึกษาพฤติกรรมในการฟังเพลงในระยะยาว กำหนดให้เมื่อผู้ฟังหยุดฟังเพลงนานกว่า 30 นาทีจะถูกพิจารณาเป็น 1 เซสชั่นและต้องประกอบด้วยเพลงที่มีการฟังมากกว่า 2 เพลง (Park, S. E., et al, 2011) รายละเอียดฐานข้อมูลเพลง Last.fm แสดงดังตารางที่ 4-1

ตารางที่ 4-1 รายละเอียดฐานข้อมูลเพลง Last.fm

รายละเอียด	จำนวน
ผู้ฟัง	300
เพลง	373,946
ประวัติการฟังเพลง	3,130,850
เซสชั่น	120,000
จำนวนเพลงเฉลี่ยต่อ 1 เซสชั่น	26

ฐานข้อมูลที่ 2 คือ ฐานข้อมูลเพลง 30Music (Turrin, R. et al., 2015) จัดเก็บประวัติการฟังเพลงของผู้ฟังจากเว็บไซต์ Last.fm ระหว่างปี 2014 ถึงปี 2015 โดยพิจารณาผู้ฟังที่มีการฟังเพลงมากกว่า 400 เซสชั่นและระยะห่างระหว่างเซสชั่นมากกว่า 13 นาที รายละเอียดฐานข้อมูลเพลง 30Music แสดงดังตารางที่ 4-2

ตารางที่ 4-2 รายละเอียดฐานข้อมูลเพลง 30Music

รายละเอียด	**จำนวน**
ผู้ฟัง	279
เพลง	517,451
ประวัติการฟังเพลง	1,839,928
เซสชั่น	111,600
จำนวนเพลงเฉลี่ยต่อ 1 เซสชั่น	16.5

4.2 การออกแบบการทดลองและวิธีการวัดประสิทธิภาพความถูกต้อง

สำหรับการออกแบบวิธีการทดลองของวิธีการ ISSCF และวิธีการ SSCF ได้แบ่งข้อมูลการฟังเพลงออกเป็น 2 ขั้นตอน ดังนี้

4.2.1 ขั้นตอนเริ่มต้น สำหรับขั้นตอนเริ่มต้นของวิธี ISSCF นั้นต้องการข้อมูลเซสชั่นเบื้องต้นของผู้ฟัง เนื่องจากวิธีที่ 1 ขั้นตอนการหาความคล้ายเซสชั่นของผู้ฟังจำเป็นต้องใช้ข้อมูลเซสชั่นเบื้องต้นในการสร้างรายการเพลงแนะนำและวิธีที่ 2 การสร้างตารางความถี่ของการฟังเพลงต้องใช้ข้อมูลเซสชั่นของผู้ฟังสำหรับการวิเคราะห์เชิงมุมของการฟังเพลง ดังนั้นเซสชั่นเบื้องต้นที่เหมาะสมจึงต้องมีเพลงที่ถูกฟังซ้ำที่เพียงพอในการสร้างรายการเพลงแนะนำ

วิธีการ ISSCF จึงได้พิจารณาจำนวนของเซสชั่นเริ่มต้นที่แตกต่างกัน โดยกำหนดจำนวนเซสชั่นที่ 25, 50, 75 และ 100 เซสชั่นเริ่มต้น โดยใช้การวัดค่า Song diversity ดังสมการที่ (4.1) สำหรับการวัดความหลากหลายของการฟังเพลงค่าที่เข้าใกล้ 1 แสดงถึงผู้ฟังมีการฟังเพลงที่ไม่ซ้ำกัน ถ้าเพลงที่เข้าใกล้ 0 แสดงถึงเพลงในเซสชั่นมีการฟังซ้ำมาก แสดงดังตารางที่ 4-3 และตารางที่ 4-4

$$Song\ diversity = \frac{\text{จำนวนเพลงที่ไม่ซ้ำ}}{\text{จำนวนเพลงทั้งหมด}} \tag{4.1}$$

ตารางที่ 4-3 ค่า Song diversity ของฐานข้อมูลเพลง Last.fm

เซสชั่นเริ่มต้น	**ค่า Song diversity**
25	0.74
50	0.67
75	0.62
100	0.59

ตารางที่ 4-4 ค่า Song diversity ของฐานข้อมูลเพลง 30Music

เซสชั่นเริ่มต้น	**ค่า Song diversity**
25	0.8
50	0.74
75	0.71
100	0.69

สำหรับการพิจารณาการฟังซ้ำของเพลงคือเพลงที่ถูกฟังมากกว่า 1 ครั้ง แสดงผลลัพธ์จาก 2 ฐานข้อมูลดังตารางที่ 4-5 และตารางที่ 4-6

ตารางที่ 4-5 แสดงการฟังซ้ำของฐานข้อมูลเพลง Last.fm

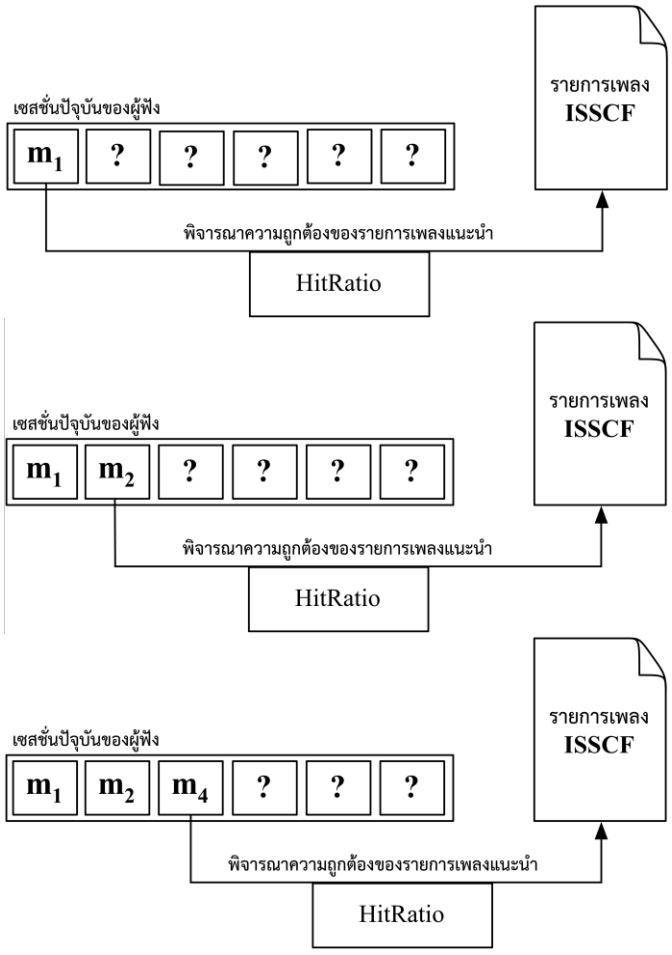
เซสชั่นเริ่มต้น	**จำนวนฟังซ้ำ (%)**
25	29.58
50	35.96
75	40.15
100	43.52

ตารางที่ 4-6 แสดงการฟังซ้ำของฐานข้อมูลเพลง 30Music

เซสชั่นเริ่มต้น	**จำนวนฟังซ้ำ (%)**
25	22.78
50	27.56
75	29.81
100	31.71

จากการประเมินผลโดยใช้ตัววัดประสิทธิภาพ Song diversity และการฟังซ้ำของผู้ฟัง แสดงให้เห็นว่าเซสชั่นเริ่มต้นที่ 100 เซสชั่น ซึ่งพิจารณาเป็น 25% จากข้อมูลทั้งหมด ส่งผลให้การหาความคล้ายของเซสชั่นในวิธีที่ 1 ของ ISSCF และการพิจารณาช่วงเวลาในการฟังเพลงที่ชื่นชอบของวิธีที่ 2 ของ ISSCF ได้ผลความถูกต้องที่ดีเมื่อเปรียบเทียบกับเซสชั่นเริ่มต้นที่ 25, 50 และ 75 เซสชั่น ดังนั้นวิธีการ ISSCF จึงใช้เซสชั่นเริ่มต้นที่ 100 เซสชั่นทั้งฐานข้อมูลเพลง last.fm และ 30Music

4.2.2 ขั้นตอนออนไลน์ วิทยานิพนธ์นี้ได้ใช้การออกแบบการทดลอง Prequential evaluation ซึ่งมีความเหมาะสมในการทดลองแบบ Incremental update หรือแบบ Data stream (J. Gama, 2014) โดยในส่วนของขั้นตอนแบบออนไลน์ได้จำลองการฟังเพลงของผู้ฟังเป้าหมาย (ผู้ฟังที่กำลังฟังเพลงแบบออนไลน์) เข้ามาทดสอบแบบเป็นลำดับ โดยเพลงถัดไปที่ผู้ฟังเลือกฟังจะถูกพิจารณาเป็นข้อมูลทดสอบกับรายการเพลงแนะนำปัจจุบัน โดยใช้วิธีการวัดประสิทธิภาพ HitRatio เมื่อวัดผลความถูกต้องของรายการเพลงเสร็จสิ้น ข้อมูลทดสอบจะถูกพิจารณาเป็นเซสชั่นปัจจุบันในการสร้างรายการเพลงแนะนำใหม่เมื่อผู้ฟังเลือกฟังเพลงไม่ตรงกับรายการเพลงแนะนำเดิม แสดงดังภาพที่ 4-1



ภาพที่ 4-1 ตัวอย่างการสร้างรายการเพลงจากเซสชั่นปัจจุบันของผู้ฟังแล้วประเมินผลโดยพิจารณาเพลงถัดไปที่เลือกฟัง

ข้อดีของวิธีการออกแบบการทดลอง Prequential evaluation คือจะไม่มีการแบ่งข้อมูลออกเป็นข้อมูลฝึกฝนและข้อมูลทดสอบทำให้ไม่เกิดปัญหาความเบาบางของข้อมูลและข้อมูลทดสอบจะถูกนำมาเป็นข้อมูลฝึกฝนต่อไปซึ่งเหมาะสมในการแนะนำเพลงตามความชอบในปัจจุบันของผู้ฟัง โดยการทดลองในขั้นตอนนี้จะใช้เซสชั่นสำหรับเป็นข้อมูลทดสอบในการแนะนำเพลงออนไลน์ที่ 300 เซสชั่นพิจารณาเป็น 75% ของข้อมูลทั้งหมด

สำหรับขั้นตอนการวัดประสิทธิภาพความถูกต้องได้เลือกใช้วิธีการวัดประสิทธิภาพคือ $HitRatio$ ซึ่งพิจารณาความถูกต้องของรายการเพลงที่จะแนะนำให้กับผู้ฟังกับเพลงที่ผู้ฟังเลือกฟังถัดไปแสดงดังสมการที่ (4.2) ซึ่งเป็นการประเมินผลความถูกต้องแบบออนไลน์ โดยการซ่อนเพลงที่ผู้ฟังเลือกฟังถัดไปเป็นข้อมูลทดสอบแบบเป็นลำดับ

$$HitRatio = \frac{\sum_{i=1}^{N} HR@n}{N} \tag{4.2}$$

สำหรับค่าเฉลี่ยของวิธีการวัดประสิทธิภาพ $HitRatio$ ของผู้ฟังทั้งหมด สามารถคำนวณได้จากสมการที่ (4.3)

$$AverageHitRatio = \frac{\sum_{i=1}^{|User|} HitRatio}{|User|} \tag{4.3}$$

โดยที่ $HR@n$ คือ ถ้าเพลงถัดไปที่ผู้ฟังเลือกฟังถัดไปตรงกับรายการเพลงแนะนำจะมีค่าเท่ากับ 1 ถ้าไม่ตรงจะมีค่าเป็น 0

n คือ รายการเพลงแนะนำ n เพลง
N คือ จำนวนข้อมูลทดสอบทั้งหมด
$|User|$ คือ จำนวนผู้ฟังทั้งหมดในระบบ

การวัดความถูกต้องของรายการเพลงแนะนำ โดยวิธีการ $Precision$ ซึ่งพิจารณาสัดส่วนของเพลงที่ผู้ฟังเลือกฟังกับรายการเพลงที่จะแนะนำทั้งหมด แสดงดังสมการที่ (4.4)

$$precision@n = \frac{\#\text{hit}}{\#n} \tag{4.4}$$

การวัดความถูกต้องทุกรายการเพลงแนะนำสำหรับแต่ละผู้ฟัง แสดงดังสมการที่ (4.5)

$$Precision = \frac{\sum_{i=1}^{N} precision@n}{N} \tag{4.5}$$

สำหรับค่าเฉลี่ยของ $Precision$ ของผู้ฟังทั้งหมด สามารถคำนวณได้จากสมการที่ (4.6)

$$AveragePrecision = \frac{\sum_{i=1}^{|User|} Precision}{|User|} \quad (4.6)$$

โดยที่ n คือ รายการเพลงแนะนำ n เพลง

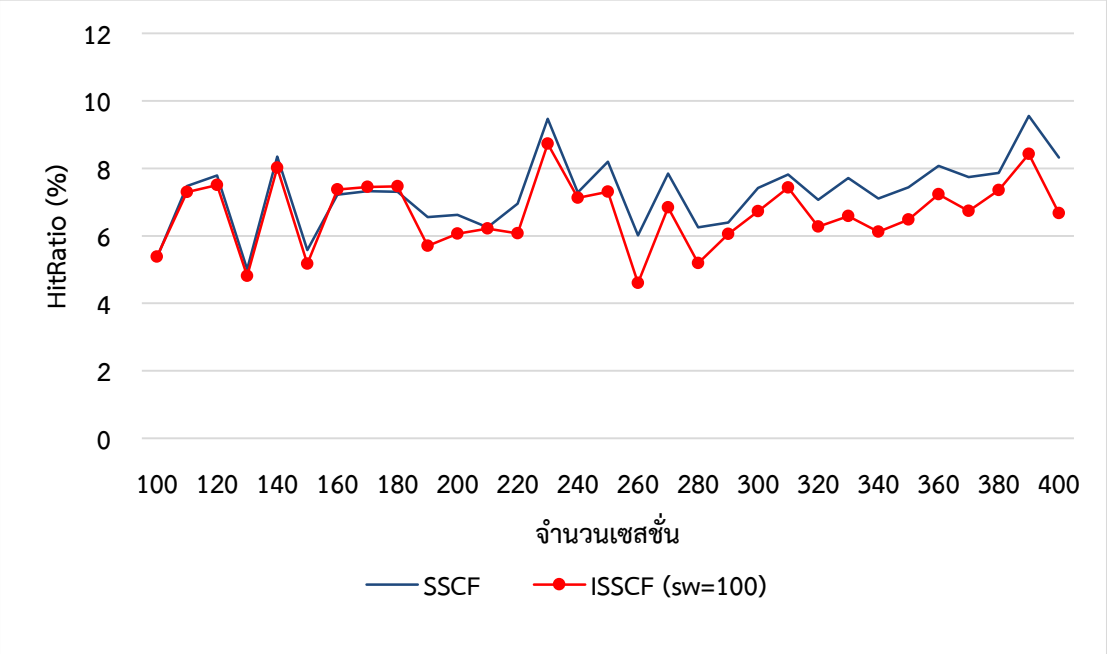
hit คือ ถ้าเพลงถัดไปที่ผู้ฟังเลือกฟังถัดไปตรงกับรายการเพลง ค่า hit เท่ากับ 1 ถ้าไม่ตรงจะมีค่าเป็น 0

N คือ จำนวนรายการเพลงทั้งหมด

$|User|$ คือ จำนวนผู้ฟังทั้งหมดในระบบ

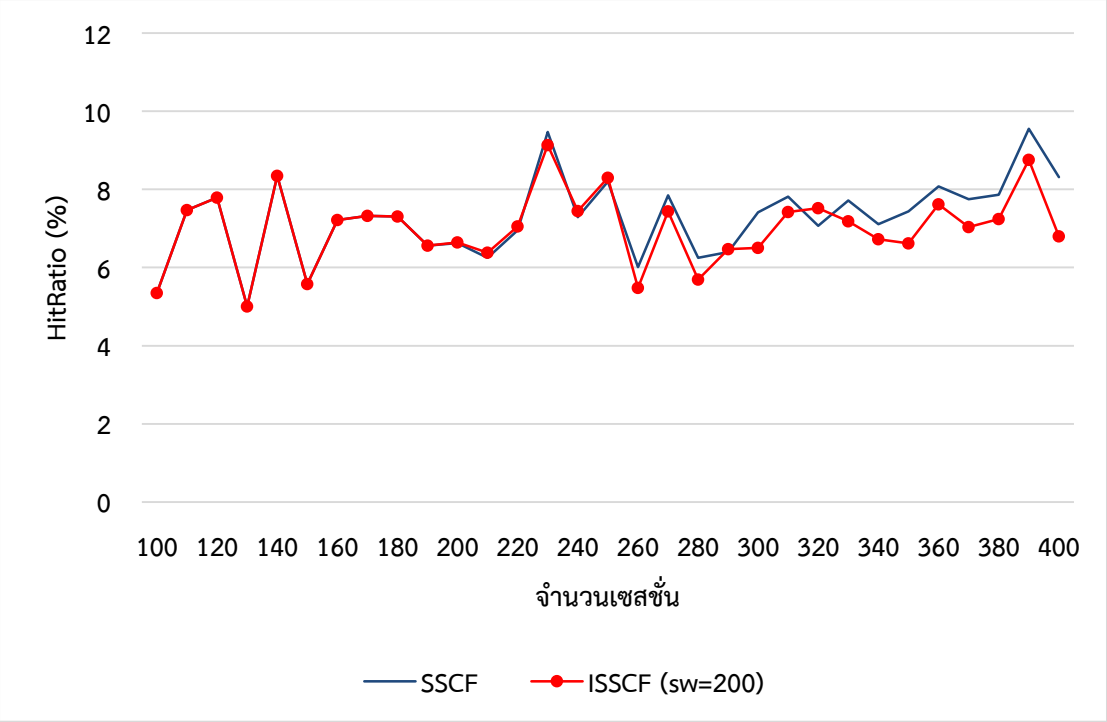
4.3 ผลการทดลองฐานข้อมูลเพลง 30Music

การทดลองแรกในวิทยานิพนธ์นี้ได้เปรียบเทียบวิธีการ Session based Collaborative Filtering (SSCF) กับวิธีการ Incremental Session based Collaborative Filtering (ISSCF) ร่วมกับการอัพเดทข้อมูลโดยใช้ Sliding windows โดยเปรียบเทียบขนาดของข้อมูล (sw) ที่ 100, 200 และ 300 เซสชั่นล่าสุด จากผลการทดลองโดยใช้วิธีการวัดประสิทธิภาพความถูกต้อง HitRatio ที่ 10 รายการเพลง ($HR@10$) แสดงผลการทดลองดังภาพที่ 4-2, 4-3 และ 4-4



ภาพที่ 4-2 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง ($HR@10$) โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 100 เซสชั่นล่าสุด (sw=100)

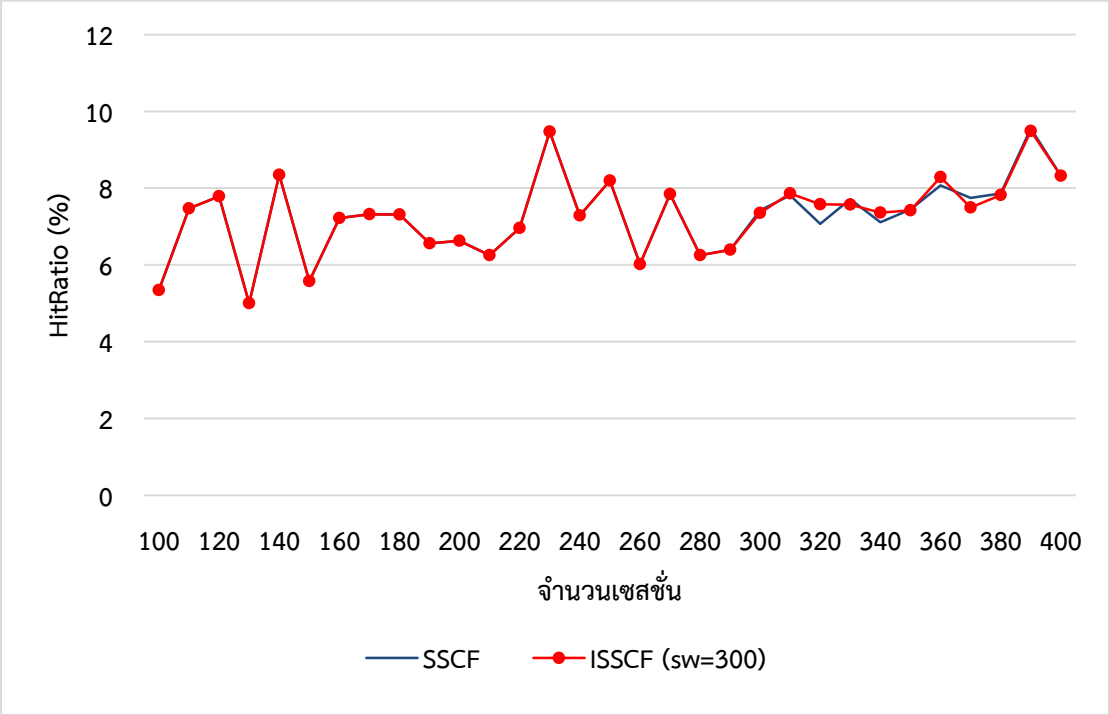
จากภาพที่ 4-2 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงแนะนำเฉลี่ยทุกผู้ฟังในแต่ละเซสชั่น โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 100 เซสชั่นล่าสุด (sw=100) จากผลการทดลองแสดงให้เห็นถึงทั้งสองวิธีการให้ความถูกต้องที่ใกล้เคียงกันแต่วิธีการ SSCF ยังคงให้ความถูกต้องมากกว่าวิธีการ ISSCF โดยวิธีการ SSCF ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 7.16 % และวิธีการ ISSCF (sw=100) ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 6.48 % จากผลการทดลองดังกล่าวแสดงถึงการอัพเดทเซสชั่นที่ 100 เซสชั่นล่าสุดเป็นการเปลี่ยนแปลงความชอบการฟังเพลงของผู้ฟังที่เร็วเกินไป ซึ่งข้อมูลในอดีตของผู้ฟังยังคงมีความสำคัญในการเลือกฟังเพลงของผู้ฟัง อย่างไรก็ตามวิธีการ ISSCF สามารถให้ประสิทธิภาพในเชิงเวลาที่รวดเร็วกว่าวิธีการ SSCF ซึ่งสามารถช่วยแก้ปัญหาขนาดของข้อมูลได้



ภาพที่ 4-3 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง ($HR@10$) โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 200 เซสชั่นล่าสุด (sw=200)

จากภาพที่ 4-3 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงเฉลี่ยทุกผู้ฟังในแต่ละเซสชั่น โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 200 เซสชั่นล่าสุด (sw=200) จากผลการทดลองแสดงให้เห็นถึงทั้งสองวิธีการให้ความถูกต้องที่ใกล้เคียงกัน โดยวิธีการ SSCF ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 7.16 % และวิธีการ ISSCF (sw=200) ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 6.98 % แต่วิธีการสร้างรายการเพลง ISSCF

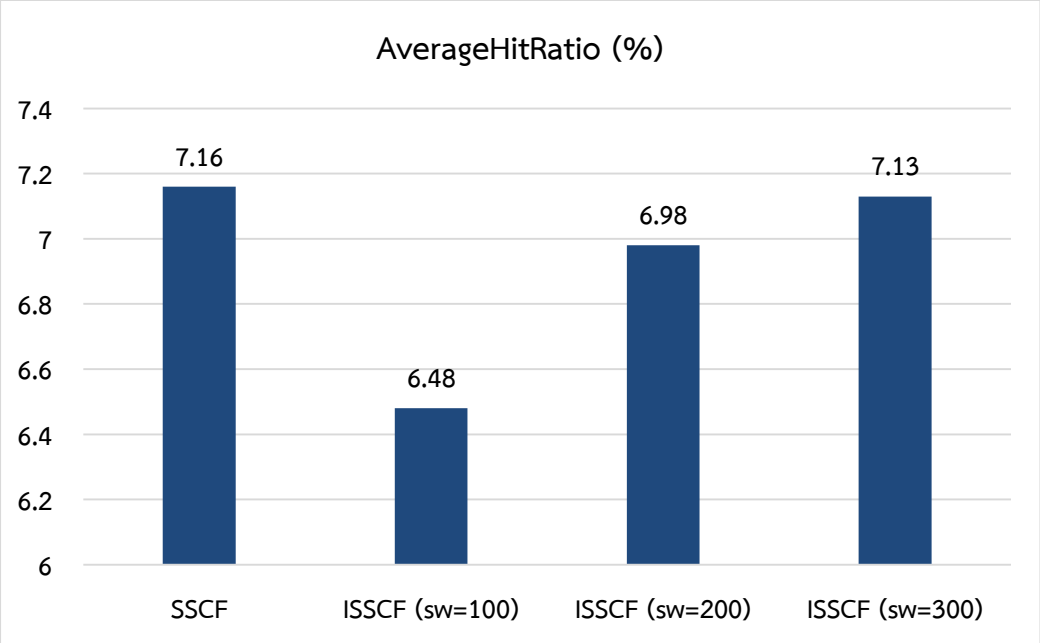
(sw=200) ยังคงให้ความเร็วในการประมวลผลที่รวดเร็วกว่าวิธีการ SSCF ซึ่งข้อมูลเพลงจะไม่เพิ่มขึ้นตามเวลาซึ่งจะอธิบายเพิ่มเติมในส่วนถัดไป



ภาพที่ 4-4 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง ($HR@10$) โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 300 เซสชั่นล่าสุด (sw=300)

จากภาพที่ 4-4 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงเฉลี่ยทุกผู้ฟังในแต่ละเซสชั่น โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 300 เซสชั่นล่าสุด (sw=300) ในแต่ละเซสชั่นจากผลการทดลองแสดงให้เห็นถึงวิธีการ ISSCF (sw=300) ให้ผลการทดลองที่ดีกว่าการอัพเดทเซสชั่นที่ 100 และ 200 เซสชั่นล่าสุด โดยวิธีการ SSCF ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 7.16 % และวิธีการ ISSCF (sw=300) ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 7.13 %

จากผลความถูกต้องเฉลี่ยรวมทุกเซสชั่นโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงที่จะแนะนำให้กับผู้ฟังของวิธีการ SSCF และวิธีการ ISSCF ร่วมกับอัพเดทเซสชั่นที่ขนาดข้อมูล (sw) 100, 200 และ 300 ล่าสุด แสดงดังภาพที่ 4-5 โดยทั้งสองวิธีการมีความถูกต้องที่ใกล้เคียงกันมากที่สุด ดังนั้นในวิทยานิพนธ์นี้จึงได้มีการทดสอบความแตกต่างของผลการทดลองอย่างมีนัยสำคัญทางสถิติ ซึ่งจะเลือกขนาดของ Sliding windows ที่มีความเหมาะสมสำหรับข้อมูลชุดนี้



ภาพที่ 4-5 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ในแต่ละขนาดของ Sliding windows เฉลี่ยรวมทุกเซสชั่น

4.3.1 การทดสอบความแตกต่างของผลความถูกต้องอย่างมีนัยสำคัญทางสถิติ

เนื่องจากความถูกต้องของวิธีการ ISSCF และวิธีการ SSCF มีความถูกต้องที่ใกล้เคียงกัน ดังนั้นวิทยานิพนธ์นี้จึงได้ทดสอบความแตกต่างความถูกต้องที่ได้จาก 2 วิธีการ โดยวัดความแตกต่างระหว่างความผิดพลาดของทั้ง 2 วิธีการอย่างมีนัยสำคัญทางสถิติ (Dietterich, T. G., 1998) เปรียบเทียบวิธีการสร้างรายการเพลงของวิธีการ A และ B ซึ่งสามารถคำนวณสัดส่วนของความผิดพลาดค่า Z ได้จากสมการที่ (4.5) และ (4.6) ดังนี้

$$p_A = \frac{n_{00}+n_{01}}{n} \tag{4.5}$$

$$p_B = \frac{n_{00}+n_{10}}{n} \tag{4.6}$$

โดยที่ p_A คือ สัดส่วนความผิดพลาดของวิธีการสร้างรายการเพลง A

p_B คือ สัดส่วนความผิดพลาดของวิธีการสร้างรายการเพลง B

n_{00} คือ จำนวนความผิดพลาดของวิธีการสร้างรายการเพลง A และ B

n_{01} คือ จำนวนความผิดพลาดของวิธีการสร้างรายการเพลง A แต่ถูกต้องในวิธีการสร้างรายการเพลง B

n_{10} คือ จำนวนความถูกต้องของการแนะนำโดยวิธีการสร้างรายการเพลง A แต่ผิดพลาดในวิธีการสร้างรายการเพลง B

n คือ จำนวนข้อมูลทดสอบ

จากนั้นพิจารณาค่าความผิดพลาดมาตรฐาน Standard error (se) ดังสมการที่ 4.7

$$se = \sqrt{\frac{2p(1-p)}{n}} \tag{4.7}$$

โดยที่ $p = \frac{(p_A + p_B)}{2}$ คือ ค่าเฉลี่ยของความผิดพลาดทั้งวิธีการสร้างรายการเพลง A และ B จากการวิเคราะห์นี้สามารถหาค่านัยสำคัญทางสถิติโดยใช้ *Z-test* (z) ได้ ดังสมการที่ 4.8

$$\mathrm{z} = \frac{p_A + p_B}{se} \tag{4.8}$$

กำหนดให้ H_0 : ทั้งวิธีการ SSCF และ ISSCF ร่วมกับ Sliding windows (sw) มีความถูกต้องในการแนะนำเพลงที่ไม่แตกต่างกัน

H_0 : $z_{0.1} = z$

H_1 : ทั้งวิธีการ SSCF และ ISSCF ร่วมกับ Sliding windows (sw) มีความถูกต้องในการแนะนำเพลงที่แตกต่างกันอย่างมีนัยสำคัญทางสถิติ

H_1 : $z_{0.1} \neq z$

จากผลการทดลองในวิทยานิพนธ์นี้ วิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่น Sliding windows (sw) ที่ 100, 200 และ 300 เซสชั่นล่าสุดเปรียบเทียบกับวิธีการ SSCF โดยการทดสอบความแตกต่างของผลการทดลองอย่างมีนัยสำคัญทางสถิติ ที่ค่าความผิดพลาด ***p-value*** เท่ากับ 0.1 และเป็นการทดสอบแบบสองหาง ($z_{0.1/2}$) ได้ค่า $z_{0.05}$ เท่ากับ 1.64 แสดงผลการทดลอง ดังนี้

1. วิธีการ SSCF กับวิธีการ ISSCF (sw=100) ได้ z = 24.67 ดังนั้น $z > z_{0.05}$ (24.67 > 1.64) แสดงว่าปฏิเสธ H_0 ยอมรับ H_1

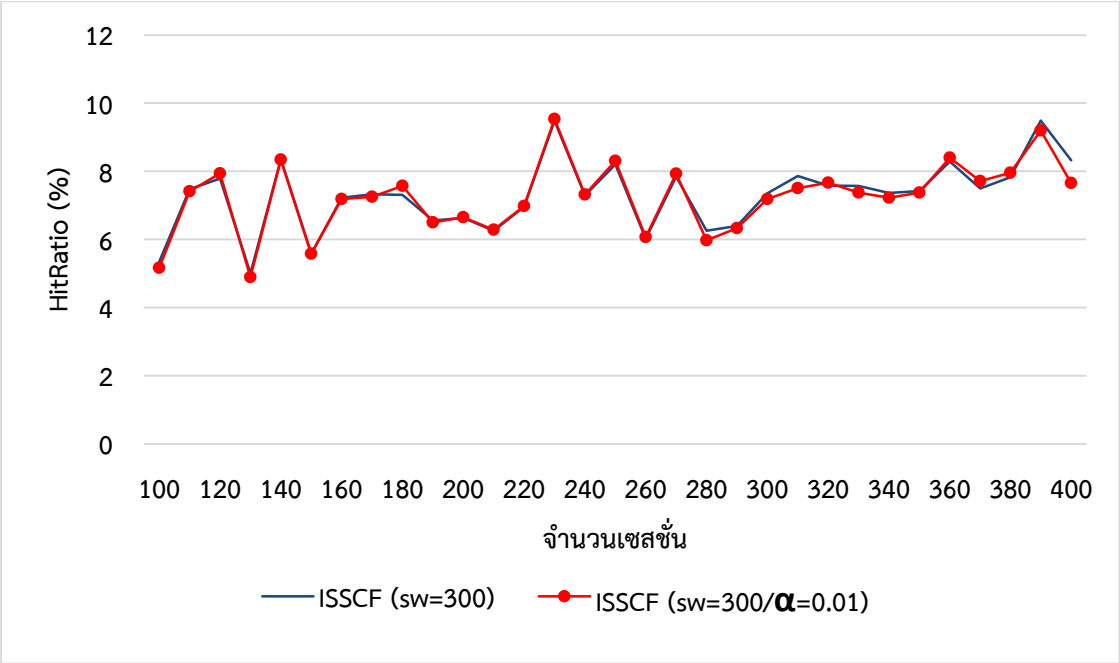
   จากผลสรุปแสดงถึงวิธีการ SSCF ให้ผลความถูกต้องดีกว่าวิธีการ ISSCF (sw=100) อย่างมีนัยสำคัญทางสถิติ
2. วิธีการ SSCF กับวิธีการ ISSCF (sw=200) ได้ z = 5.91 ดังนั้น $z > z_{0.05}$ (5.91 > 1.64) แสดงว่าปฏิเสธ H_0 ยอมรับ H_1

   จากผลสรุปแสดงถึงวิธีการ SSCF ให้ผลความถูกต้องดีกว่าวิธีการ ISSCF (sw=200) อย่างมีนัยสำคัญทางสถิติ
3. วิธีการ SSCF กับวิธีการ ISSCF (sw=300) ได้ z= 0.77 ดังนั้น $z < z_{0.05}$ (0.77 < 1.64) แสดงว่ายอมรับ H_0

จากผลสรุปแสดงถึงวิธีการ SSCF ให้ผลความถูกต้องดีกว่าวิธีการ ISSCF ขนาดข้อมูล (sw) ที่ 300 เซสชั่นล่าสุดอย่างไม่มีนัยสำคัญทางสถิติ

จากการทดสอบความแตกต่างของผลการทดลอง แสดงให้เห็นถึงวิธีการ ISSCF ขนาดข้อมูล (sw) ที่ 300 เซสชั่นล่าสุด ซึ่งมีประสิทธิภาพความถูกต้องที่ไม่แตกต่างกับวิธีการ SSCF แบบเดิม แต่ให้ความรวดเร็วในการคำนวณซึ่งช่วยแก้ปัญหาการประมวลผลข้อมูลที่มีปริมาณมากได้ ซึ่งวิธีการ Sliding windows นั้นใช้เวลาในการคำนวณที่คงที่ตามขนาดของข้อมูลซึ่งไม่มีการพิจารณาข้อมูลการฟังเพลงที่ล้าสมัยทำให้ใช้เวลาในการคำนวณที่รวดเร็วกว่า SSCF

การทดลองที่ 2 ได้เปรียบเทียบประสิทธิภาพความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ของวิธีการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง โดยใช้วิธีการ ISSCF ร่วมกับ Sliding windows (sw) ที่ 300 เซสชั่นล่าสุด เนื่องจากวิธีการนี้ให้ความถูกต้องที่เทียบเท่ากับวิธีการ SSCF แบบเดิมแต่สามารถประมวลผลได้รวดเร็วในฐานข้อมูลเพลงขนาดใหญ่เปรียบเทียบกับวิธีการ ISSCF โดยการอัพเดทข้อมูลด้วยวิธี Forgetting mechanisms ประกอบด้วยวิธีการ Sliding windows ที่ 300 เซสชั่นล่าสุด (sw=300) ร่วมกับวิธีการ Fading factors เพื่อให้ความสำคัญกับการเลือกฟังเพลงในปัจจุบัน เนื่องจากแสดงถึงความชอบของผู้ฟังเพลงขณะนั้น โดยเปรียบเทียบค่าพารามิเตอร์ในการเพิ่มความสำคัญของข้อมูล (α) ที่ 0.01, 0.1 และ 0.5 ตามลำดับ โดยผลการทดลองแสดงดังภาพที่ 4-6, 4-7 และ 4-8



ภาพที่ 4-6 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง โดยเปรียบเทียบวิธีการ ISSCF (sw=300) กับ ISSCF ที่ (sw=300) ร่วมกับค่า α ที่ 0.01

จากภาพที่ 4-6 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงของวิธีการ ISSCF ขนาดข้อมูล (sw) ที่ 300 เซสชั่นล่าสุดร่วมกับค่าน้ำหนักในการเพิ่มความสำคัญ (α) ที่ 0.01 จากฟังก์ชัน Exponential time decay function (sw =300/α=0.01) เปรียบเทียบกับวิธีการ ISSCF ร่วมกับ Sliding windows ที่ 300 เซสชั่นล่าสุด(sw=300)

จากผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio แสดงให้เห็นว่าวิธีการ ISSCF (sw =300/α=0.01) เฉลี่ยรวมทุกเซสชั่นมีความถูกต้องที่ 7.13 % ซึ่งมีความถูกต้องที่เทียบเท่ากับ ISSCF (sw=300) ที่ 7.13 % โดยให้ผลความถูกต้องที่ไม่แตกต่างกับการไม่ใส่ค่าน้ำหนัก

จากการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของทั้งสองวิธีการ กำหนดให้

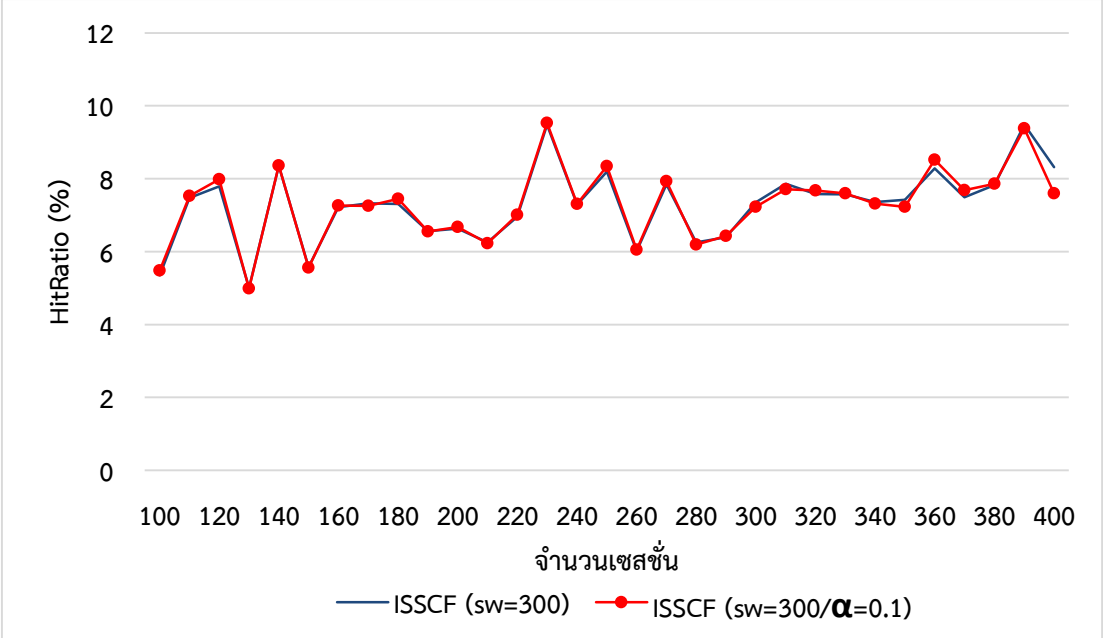
H_0 : ทั้งวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.01) มีความถูกต้องในการแนะนำเพลงที่ไม่แตกต่างกัน

$$H_0 : z_{0.1} = z$$

H_1 : ทั้งวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.01) มีความถูกต้องในการแนะนำเพลงที่แตกต่างกันอย่างมีนัยสำคัญทางสถิติ

$$H_1 : z_{0.1} \neq z$$

การพิจารณาวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.01) ได้ z = 0.33 ดังนั้น $z < z_{0.05}$ (0.33 < 1.64) แสดงว่ายอมรับ H_0 นั่นคือ ISSCF (sw =300/α=0.01) ไม่มีความแตกต่างกับวิธี ISSCF (sw =300) ในทางสถิติ

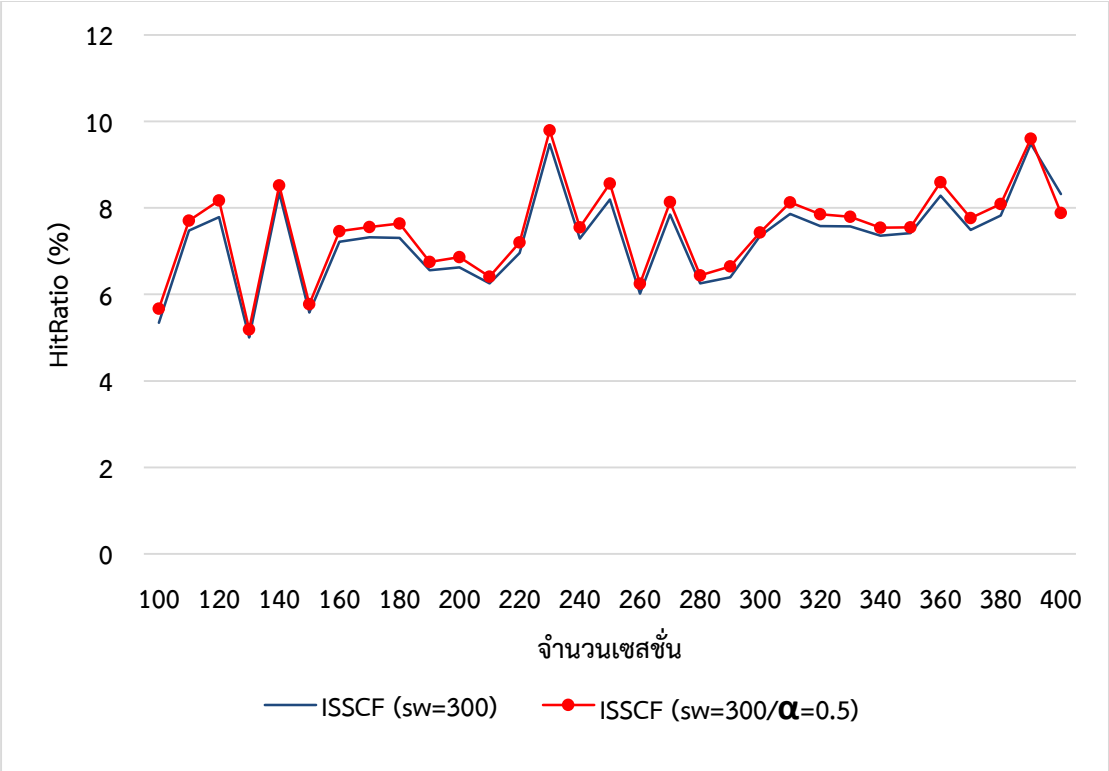


ภาพที่ 4-7 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง โดยเปรียบเทียบวิธีการ ISSCF (sw=300) กับ ISSCF ที่ (sw=300) ร่วมกับค่า α ที่ 0.1

จากภาพที่ 4-7 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงของวิธีการ ISSCF ขนาดข้อมูล (sw) ที่ 300 เซสชั่นล่าสุดร่วมกับค่าน้ำหนัก (α) ที่ 0.1 (sw =300/α=0.1) เปรียบเทียบกับวิธีการ ISSCF ร่วมกับ Sliding windows ที่ 300 เซสชั่นล่าสุด (sw =300)

จากผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio แสดงให้เห็นว่าวิธีการ ISSCF (sw =300/α=0.1) มีความถูกต้องที่ 7.18 % ซึ่งมีความถูกต้องที่ดีกว่าวิธีการ ISSCF (sw =300) คือ 7.13 % โดยวิธีการ ISSCF (sw =300/α=0.1) ให้ผลความถูกต้องที่ดีกว่าวิธีการ ISSCF (sw =300)

จากการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.1) ได้ค่า z = 1.33 ดังนั้น $z < z_{0.05}$ (1.33 < 1.64) แสดงว่ายอมรับ H_0 นั่นคือ ISSCF (sw =300/α=0.1) ไม่มีความแตกต่างของความถูกต้องในการสร้างรายการเพลงกับวิธี ISSCF (sw =300) ในทางสถิติ

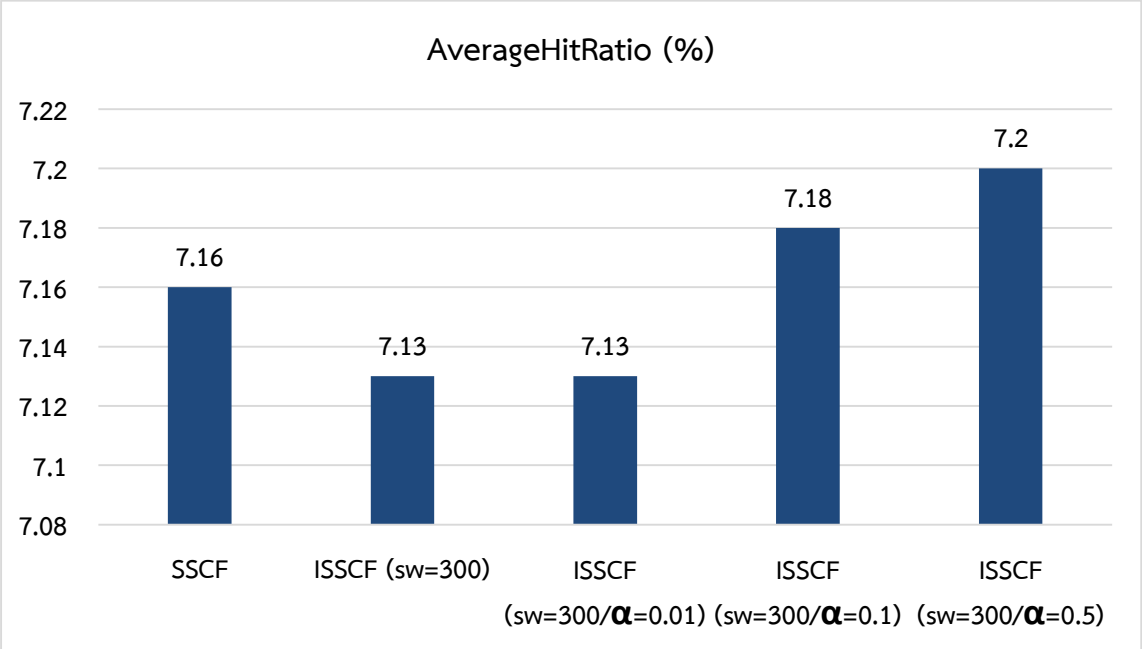


ภาพที่ 4-8 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง โดเปรียบเทียบวิธีการ ISSCF (sw=300) กับ ISSCF (sw=300) ร่วมกับค่า α ที่ 0.5

จากภาพที่ 4-8 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงของวิธีการ ISSCF ขนาดข้อมูล (sw) ที่ 300 เซสชั่นล่าสุดร่วมกับค่าน้ำหนัก (α) ที่ 0.5 เปรียบเทียบกับวิธีการ ISSCF ร่วมกับ Sliding windows (sw) ที่ 300 เซสชั่นล่าสุด

จากผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio แสดงให้เห็นว่าวิธีการ ISSCF (sw =300/α=0.5) มีความถูกต้องที่ 7.20 % และวิธีการ ISSCF (sw =300) มีความถูกต้องที่ 7.13 % จากผลสรุปการทดลอง วิธีการ ISSCF (sw =300/α=0.5) ให้ผลความถูกต้องที่ดีกว่าวิธีการ ISSCF (sw =300)

จากการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.5) ได้ค่า z = 1.67 ดังนั้น $z > z_{0.05}$ (1.67 > 1.64) แสดงว่าปฏิเสธ H_0 ยอมรับ H_1 นั่นคือ **วิธีการ ISSCF (sw =300/α=0.5) ให้ผลความถูกต้องที่ดีกว่าวิธีการ ISSCF (sw=300) อย่างมีนัยสำคัญทางสถิติ**

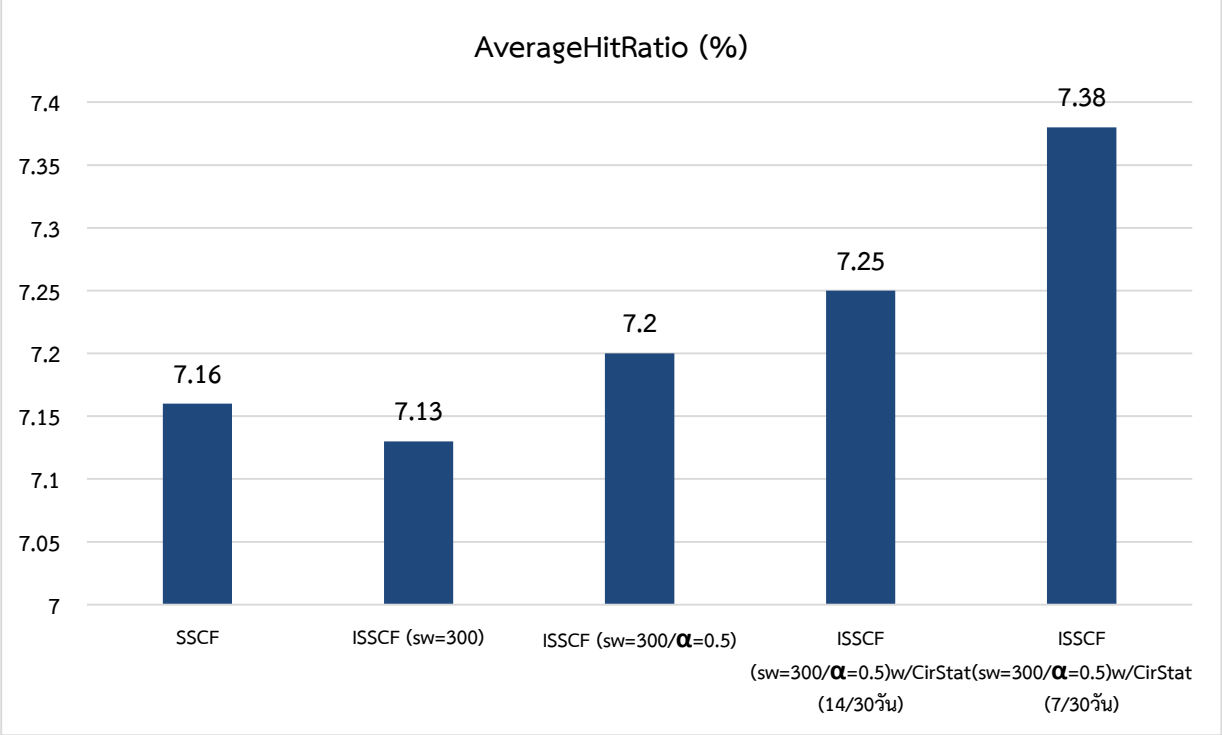


ภาพที่ 4-9 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ AverageHitRatio โดยการเปรียบเทียบวิธีการ SSCF, ISSCF ที่ 300 เซสชั่นล่าสุด (sw=300) และวิธีการ ISSCF โดยใช้วิธี Forgetting mechanism ที่ (sw=300) ค่าน้ำหนักโดยใช้ α ที่ 0.01, 0.1 และ 0.5

จากภาพที่ 4-9 แสดงผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ AverageHitRatio ที่ 10 รายการเพลง โดยการเปรียบเทียบวิธีการ SSCF, ISSCF ที่ 300 เซสชั่นล่าสุด (sw=300) และวิธีการ ISSCF โดยใช้วิธี Forgetting mechanism ที่ (sw=300) ค่าน้ำหนักโดยใช้ α ที่ 0.01, 0.1 และ 0.5 ซึ่งค่า AverageHitRatio ของวิธีการ ISSCF โดยการอัพเดทข้อมูลด้วย Forgetting mechanism ที่ค่าน้ำหนักในการเพิ่มความสำคัญที่ 0.5 (sw =300/α=0.5) ให้ผลลัพธ์ที่ดีที่สุด

วิธีการ ISSCF วิธีที่ 2 คือการสร้างรายการเพลงแนะนำโดยใช้วิธีการวิเคราะห์สถิติเชิงมุม

(Circular Statistics) ซึ่งพิจารณาช่วงเวลาเฉพาะในการฟังเพลงที่มีความแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติ โดยใช้วิธีการทดสอบทางสถิติ Rayleigh Z-test ที่ p-value เท่ากับ 0.05 ซึ่งจะสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังก็ต่อเมื่อผู้ฟังเข้าสู่ระบบที่ตรงกับช่วงเวลาที่ได้จากการวิเคราะห์สถิติเชิงมุม สำหรับการอัพเดทเซสชั่นล่าสุดของวิธีการนี้ได้ใช้วิธีการ Sliding windows โดยการพิจารณาการฟังเพลงล่าสุด N วันและเพลงที่มีการฟังมากกว่า k วัน ทำให้วิธีการนี้สร้างรายการเพลงซึ่งยังคงความชอบของผู้ฟังในปัจจุบัน

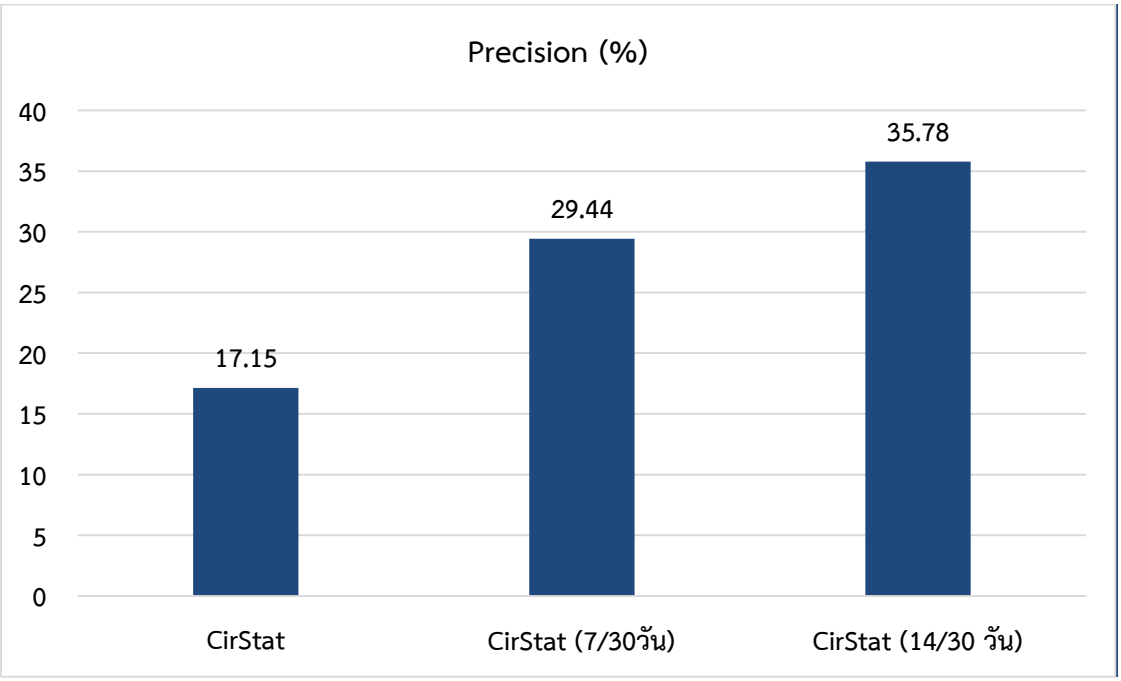


ภาพที่ 4-10 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ของวิธีการ SSCF, ISSCF วิธีที่ 1 และการรวมรายการเพลงกับการวิเคราะห์สถิติเชิงมุม (CirStat) วิธีที่ 2

จากภาพที่ 4-10 เปรียบเทียบความถูกต้องของรายการเพลงแนะนำโดยใช้วิธีการ SSCF, ISSCF วิธีที่ 1 และการรวมรายการเพลงกับการวิเคราะห์สถิติเชิงมุม (CirStat) วิธีที่ 2 โดยพิจารณาในรอบ 1 เดือนล่าสุด (N=30) โดยเพลงถูกฟังมากกว่า 14 วัน (k=14) และ 7 วัน (k =7)

ผลสรุปการทดลองโดยใช้วิธีการวัดประสิทธิภาพ HitRatio เฉลี่ยทุกผู้ฟังแสดงให้เห็นถึงวิธีการ ISSCF ร่วมกับการวิเคราะห์สถิติเชิงมุม (14/30 วัน) มีความถูกต้องที่ 7.25 % และวิธีการ ISSCF ร่วมกับการวิเคราะห์สถิติเชิงมุม (7/30 วัน) ได้ค่าความถูกต้องที่ 7.38 % ซึ่งมีความถูกต้องที่ดีกว่าการพิจารณาเพียง ISSCF วิธีการที่ 1 วิธีการเดียว โดยความถูกต้องของวิธีการ ISSCF ร่วมกับกับการวิเคราะห์สถิติเชิงมุม (7/30 วัน) ให้ความถูกต้องที่ดีที่สุด เพราะการพิจารณาเพลงที่ฟังมากกว่า 7 วัน แสดงถึงผู้ฟังมีความชื่นชอบในเพลงนั้นและสามารถสร้างรายการเพลงได้มากกว่าเพลงที่ฟังมากกว่า 14 วัน อย่างไรก็ตามการพิจารณาความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio อาจ

มีความไม่เหมาะสมกับการวิเคราะห์สถิติเชิงมุม เนื่องจากรายการเพลงที่สร้างจากวิธีการนี้ไม่ได้เปลี่ยนไปตามข้อมูลทดสอบแต่จะมีการสร้างรายการเพลงต่อเมื่อช่วงเวลาที่ได้จากวิธีการวิเคราะห์สถิติเชิงมุมตรงกับช่วงเวลาที่ผู้ฟังเข้าสู่ระบบ ดังนั้นวิทยานิพนธ์นี้จึงได้เลือกใช้วิธีการวัดประสิทธิภาพ $\boldsymbol{precision}$ (Shani, G. & Gunawardana, A., 2009) สำหรับพิจารณาความถูกต้องของรายการเพลงแนะนำแสดงดังภาพที่ 4-11



ภาพที่ 4-11 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ Precision เปรียบเทียบวิธีการวิเคราะห์สถิติเชิงมุม (CirStat) และวิธีการวิเคราะห์สถิติเชิงมุมในรอบ 30 วันล่าสุดที่เพลงถูกฟังมากกว่า 7 วันและ 14 วัน

จากภาพที่ 4-11 แสดงผลความถูกต้องเฉลี่ยของการวิเคราะห์สถิติเชิงมุม (CirStat) ซึ่งมีความถูกต้อง คือ 17.15 % และการวิเคราะห์สถิติเชิงมุมในรอบ 30 วันล่าสุดที่เพลงถูกฟังมากกว่า 7 วันได้ค่าความถูกต้อง คือ 29.44 % และเพลงถูกฟังมากกว่า 14 วันได้ค่าความถูกต้องที่ 35.78 % จากผลสรุปการทดลองแสดงถึงวิธีการวิเคราะห์สถิติเชิงมุมสามารถให้ความถูกต้องในการแนะนำเพลงได้อย่างดีและการอัพเดทข้อมูลการฟังเพลงของผู้ฟังด้วยวิธี Sliding windows สามารถเพิ่มความถูกต้องให้กับรายการเพลงที่จะแนะนำให้กับผู้ฟังได้

จากตารางที่ 4-7 แสดงผลความถูกต้องที่ดีที่สุดของวิธีการสร้างรายการเพลงแนะนำ SSCF, ISSCF วิธีที่ 1 และการรวมรายการเพลงแนะนำ ISSCF ของวิธีที่ 1 ร่วมกับการวิเคราะห์สถิติเชิงมุม (CirStat) วิธีที่ 2 โดยใช้ตัววัดประสิทธิภาพ HitRatio ในฐานข้อมูลเพลง 30Music

ตารางที่ 4-7 ผลความถูกต้องของวิธีการสร้างรายการเพลงแนะนำโดยใช้ตัววัดประสิทธิภาพ HitRatio

วิธีการ	HitRatio (%)
SSCF	7.16
ISSCF (sw=300)	7.13
ISSCF (sw=300/α=0.5)	7.2
ISSCF (sw=300/α=0.5) ร่วมกับ CirStat (7/30วัน)	7.38

จากตารางที่ 4-8 แสดงผลความถูกต้องของวิธีการสร้างรายการเพลงแนะนำ ISSCF วิธีที่ 2 คือการวิเคราะห์สถิติเชิงมุม (CirStat) โดยใช้ตัววัดประสิทธิภาพ Precision ในฐานข้อมูลเพลง 30Music

ตารางที่ 4-8 ผลความถูกต้องของวิธีการสร้างรายการเพลงแนะนำ ISSCF วิธีที่ 2 (การวิเคราะห์เชิงมุม ,CirStat) โดยใช้ตัววัดประสิทธิภาพ Precision

วิธีการ	Precision (%)
CirStat	17.15
CirStat (7/30วัน)	29.44
CirStat (14/30 วัน)	35.78

4.3.2 การวัดประสิทธิภาพเชิงเวลาของฐานข้อมูลเพลง 30Music

สำหรับการวัดประสิทธิภาพเชิงเวลาของวิธีการที่นำเสนอ Incremental Session based Collaborative Filtering (ISSCF) ได้เปรียบเทียบกับวิธีการ Session based Collaborative Filtering (SSCF) (Park, S. E., et al, 2011) ซึ่งสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังแบบออฟไลน์ที่ใช้ข้อมูลเซสชั่นในอดีตทั้งหมดมาพิจารณาโดยให้ความสำคัญของเซสชั่นเก่าและใหม่เท่ากัน ทำให้ความซับซ้อนเชิงเวลาในขั้นตอนการหาความคล้ายเป็น $O(m^2n)$ ซึ่งเป็นการหาความคล้ายของทุกเซสชั่นและเวลาในการทำนายค่าคะแนนความชอบของเพลงเพื่อสร้างรายการเพลงสำหรับแต่ละผู้ฟังเป็น $O(n)$ ซึ่งแตกต่างจากวิธีการ ISSCF เป็นการสร้างรายการเพลงแบบออนไลน์ โดยหาความคล้ายของเพลงที่ผู้ฟังเลือกฟัง (เฉพาะเซสชั่นปัจจุบันของผู้ฟังเพลง) กับเซสชั่นในอดีตซึ่งมีความซับซ้อนเชิงเวลาเป็น $O(mn)$ เนื่องจากเป็นการพิจารณาความคล้ายเฉพาะเซสชั่นปัจจุบันของผู้ฟังเพลงกับเซสชั่นในอดีตและใช้เวลาในการทำนายค่าคะแนนความชอบของเพลงเพื่อสร้างรายการเพลงสำหรับแต่ละผู้ฟังเป็น $O(n)$ โดยที่ m คือ จำนวนเซสชั่นของผู้ฟัง และ n คือ จำนวนเพลง แสดงความซับซ้อนเชิงเวลาของวิธีการ SSCF เปรียบเทียบกับวิธีการ ISSCF ดังตารางที่ 4-9

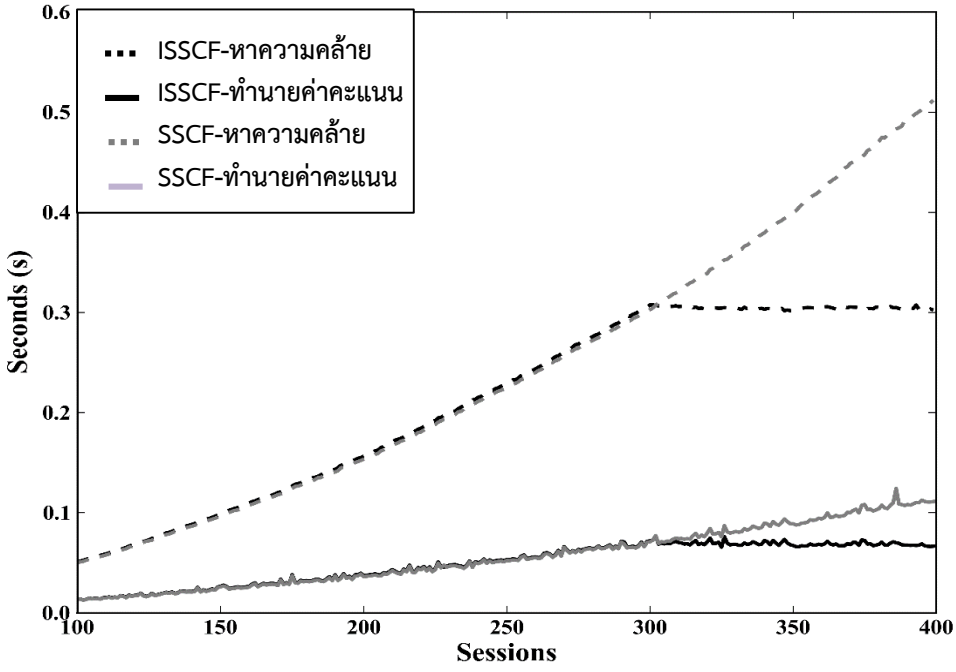
ตารางที่ 4-9 ความซับซ้อนเชิงเวลาของวิธีการ SSCF และ ISSCF

วิธีการ	ขั้นตอนหาความคล้าย	ขั้นตอนการทำนายค่า
Offline SSCF	$O(m^2n)$	$O(n)$
ISSCF	$O(mn)$	$O(n)$

จากตารางที่ 4-10 แสดงเวลาที่ใช้ในการคำนวณของขั้นตอนการหาความคล้ายและการทำนายค่าคะแนนความชอบของวิธีการ SSCF เปรียบเทียบกับวิธีการ ISSCF ที่จำนวนข้อมูล 200 เซสชั่นเฉลี่ย 20 ผู้ฟังเพลง แสดงให้เห็นถึงวิธีการ ISSCF สามารถช่วยลดการคำนวณในขั้นตอนการหาความคล้ายได้ดีกว่าวิธีการ SSCF แบบดั้งเดิม (Offline SSCF)

ตารางที่ 4-10 เวลาที่ใช้ประมวลผลของวิธีการ SSCF และ ISSCF ฐานข้อมูล 30Music

วิธีการ	ขั้นตอนหาความคล้าย	ขั้นตอนการทำนายค่า
Offline SSCF	13.19 วินาที	0.023 วินาที
ISSCF	**0.12 วินาที**	**0.023 วินาที**

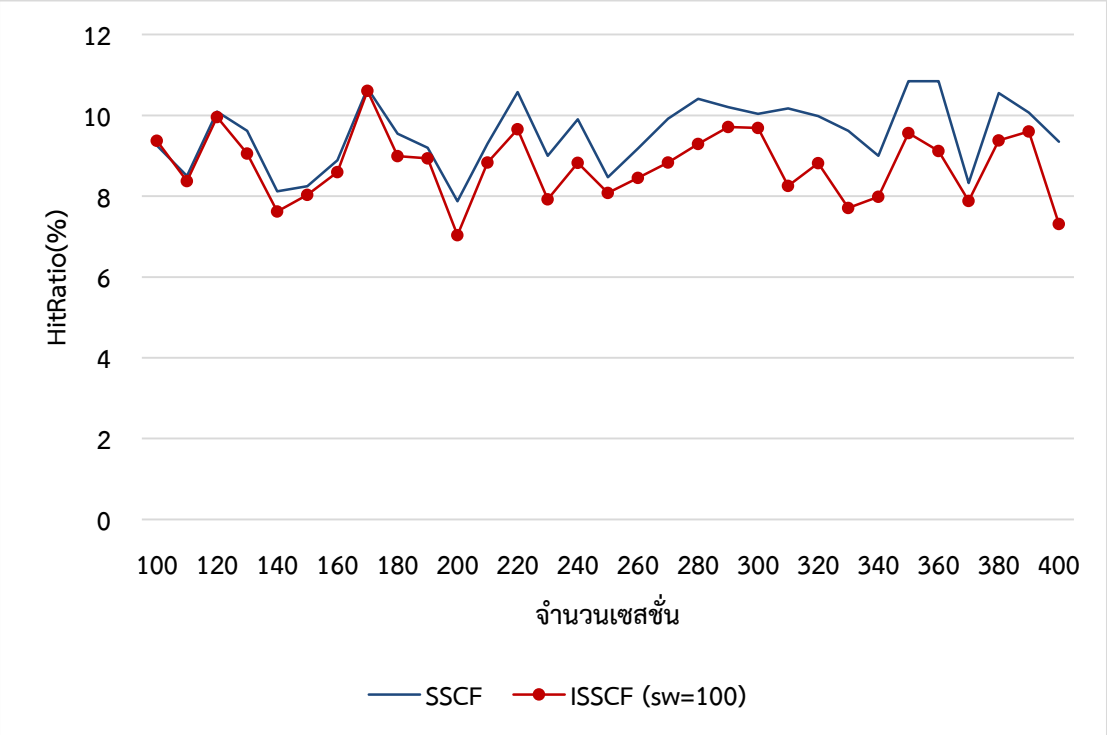


ภาพที่ 4-12 เวลาที่ใช้ในการหาความคล้ายของเซสชั่นและการทำนายค่าคะแนนความชอบของผู้ฟังเพลงของวิธีการ SSCF และ ISSCF (sw=300)

จากภาพที่ 4-12 แสดงเวลาที่ใช้ในการหาความคล้ายของเซสชั่นและการทำนายค่าคะแนนความชอบของผู้ฟังเพลงของวิธีการ SSCF และ ISSCF (sw=300) ซึ่งวิธีการ ISSCF (sw=300) นั้นใช้เวลาในการคำนวณที่คงที่ตามขนาดข้อมูลและยังคงความชอบในปัจจุบันของผู้ฟังเพลง โดยช่วยลดเวลาในการประมวลผลข้อมูลที่มีขนาดใหญ่ได้ซึ่งแตกต่างกับวิธีการ SSCF ที่ข้อมูลการฟังเพลงของผู้ฟังจะเพิ่มขึ้นตามเวลา ส่งผลให้ใช้เวลาในการคำนวณที่นานตามข้อมูลที่เพิ่มมากขึ้น โดยประมวลผลวิธีการทดลองบนเครื่อง CPU Intel Core 2 Duo 3.0 GHz และหน่วยความจำ (RAM) 8 GB ซึ่งเขียนโปรแกรมโดยใช้ภาษาไพธอน (Python)

4.4 ผลการทดลองฐานข้อมูลเพลง Last.fm

สำหรับฐานข้อมูลเพลง Last.fm วิทยานิพนธ์นี้ได้เปรียบเทียบวิธีการ Session based Collaborative Filtering (SSCF) กับวิธีการ Incremental Session based Collaborative Filtering (ISSCF) ร่วมกับการอัพเดทข้อมูลโดยใช้ Sliding windows โดยเปรียบเทียบขนาดของข้อมูล (sw) ที่ 100, 200 และ 300 เซสชั่นล่าสุด จากผลการทดลองโดยใช้วิธีการวัดประสิทธิภาพความถูกต้อง HitRatio ที่ 10 รายการเพลง ($HR@10$) แสดงผลการทดลองดังภาพที่ 4-13, 4-14 และ 4-15



ภาพที่ 4-13 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง ($HR@10$) โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 100 เซสชั่นล่าสุด (sw=100)

จากภาพที่ 4-13 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงที่จะแนะนำให้กับผู้ฟังเฉลี่ยทุกผู้ฟังในแต่ละเซสชั่น โดยเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 100 เซสชั่นล่าสุด (sw=100)

จากผลการทดลองแสดงให้เห็นถึงทั้งสองวิธีการให้ความถูกต้องที่ใกล้เคียงกันแต่วิธีการ SSCF ยังคงให้ความถูกต้องมากกว่าวิธีการ ISSCF โดยวิธีการ SSCF ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 8.74 % และวิธีการ ISSCF (sw=100) ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 8.06 %

จากผลสรุปการทดลองแสดงการพิจารณาช่วงข้อมูลที่ 100 เซสชั่นล่าสุด (sw=100) นั้นยังไม่สามารถสร้างรายการเพลงที่ดีเทียบเท่ากับวิธีการ SSCF ได้ เนื่องจากการหาความคล้ายของผู้ฟังเพลงนั้น ขนาดข้อมูลที่ 100 เซสชั่นล่าสุดเป็นการให้ความสำคัญกับข้อมูลที่ไม่ดีพอ เนื่องจากผู้ฟังยังมีความชอบหรือความคล้ายในอดีตที่สำคัญอยู่

สำหรับการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของทั้งสองวิธีการ กำหนดให้

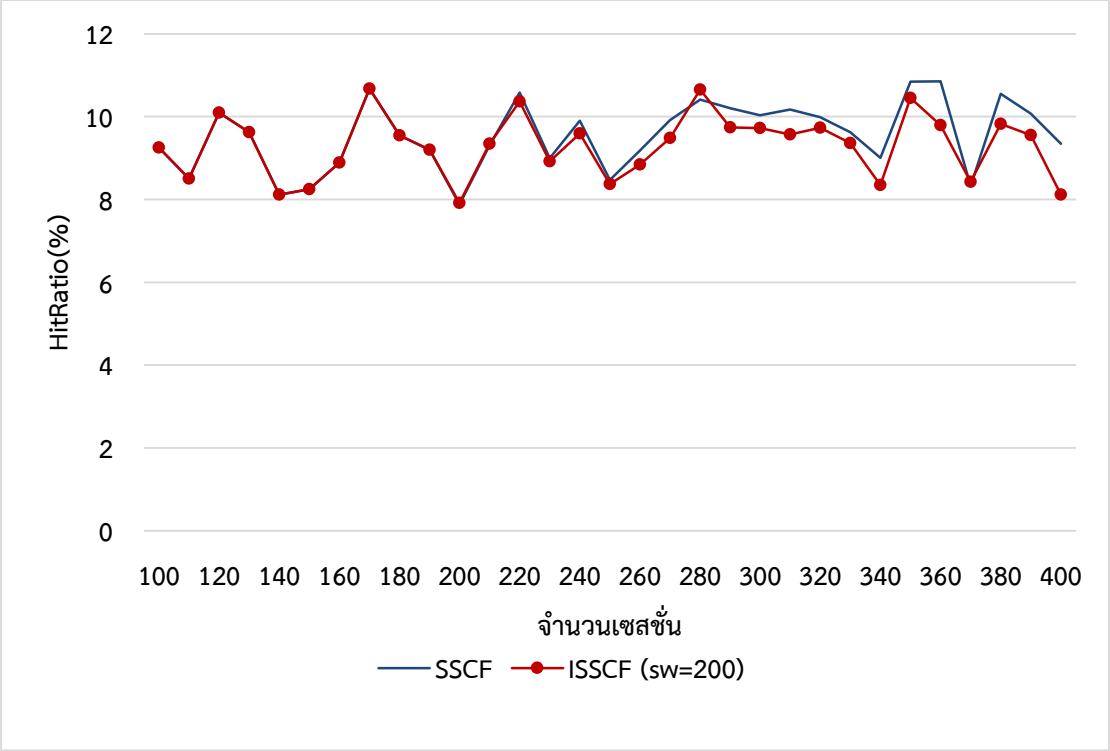
H_0 : ทั้งวิธีการ SSCF และ ISSCF (sw=100) มีความถูกต้องในการแนะนำเพลงที่ไม่แตกต่างกัน

$$H_0 : z_{0.1} = z$$

H_1 : ทั้งวิธีการ SSCF และ ISSCF (sw=100) มีความถูกต้องในการแนะนำเพลงที่แตกต่างกันอย่างมีนัยสำคัญทางสถิติ

$$H_1 : z_{0.1} \neq z$$

การพิจารณาวิธีการ SSCF และ ISSCF (sw=100) ได้z = 22 ดังนั้น $z > z_{0.05}$ (22 > 1.64) แสดงว่าปฏิเสธ H_0 ยอมรับ H_1 นั่นคือ วิธีการ SSCF ให้ผลความถูกต้องที่ดีกว่าวิธีการ ISSCF (sw=100) อย่างมีนัยสำคัญทางสถิติ

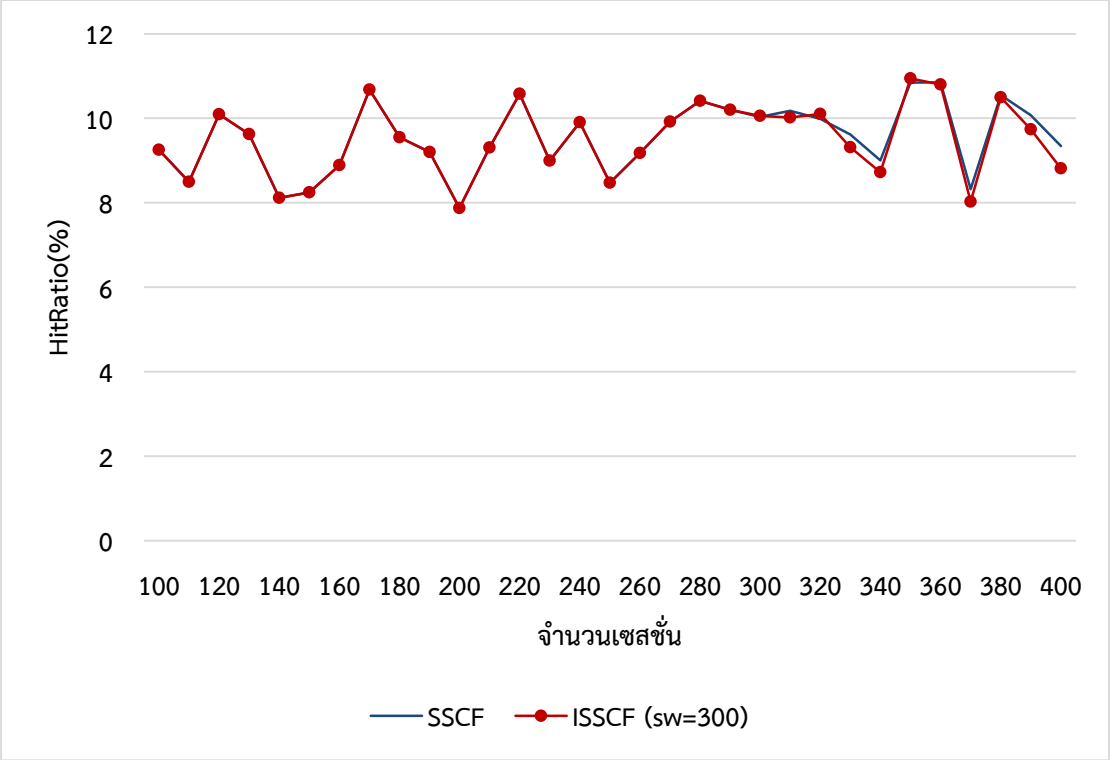


ภาพที่ 4-14 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง ($HR@10$) โดยเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 200 เซสชั่นล่าสุด (sw=200)

จากภาพที่ 4-14 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงเฉลี่ยทุกผู้ฟังในแต่ละเซสชั่น โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 200 เซสชั่นล่าสุด (sw=200)

จากผลการทดลองแสดงให้เห็นถึงทั้งสองวิธีการให้ความถูกต้องที่ใกล้เคียงกันแต่วิธีการ SSCF ยังคงให้ความถูกต้องมากกว่าวิธีการ ISSCF โดยวิธีการ SSCF ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 8.74 % และวิธีการ ISSCF (sw=200) ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 8.56 %

สำหรับการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของการพิจารณาวิธีการ SSCF และ ISSCF (sw=200) ได้ค่า z = 6 ดังนั้น $z > z_{0.05}$ (6 > 1.64) แสดงว่าปฏิเสธ H_0 ยอมรับ H_1 นั่นคือ วิธีการ SSCF ให้ผลความถูกต้องที่ดีกว่าวิธีการ ISSCF (sw=200) อย่างมีนัยสำคัญทางสถิติ



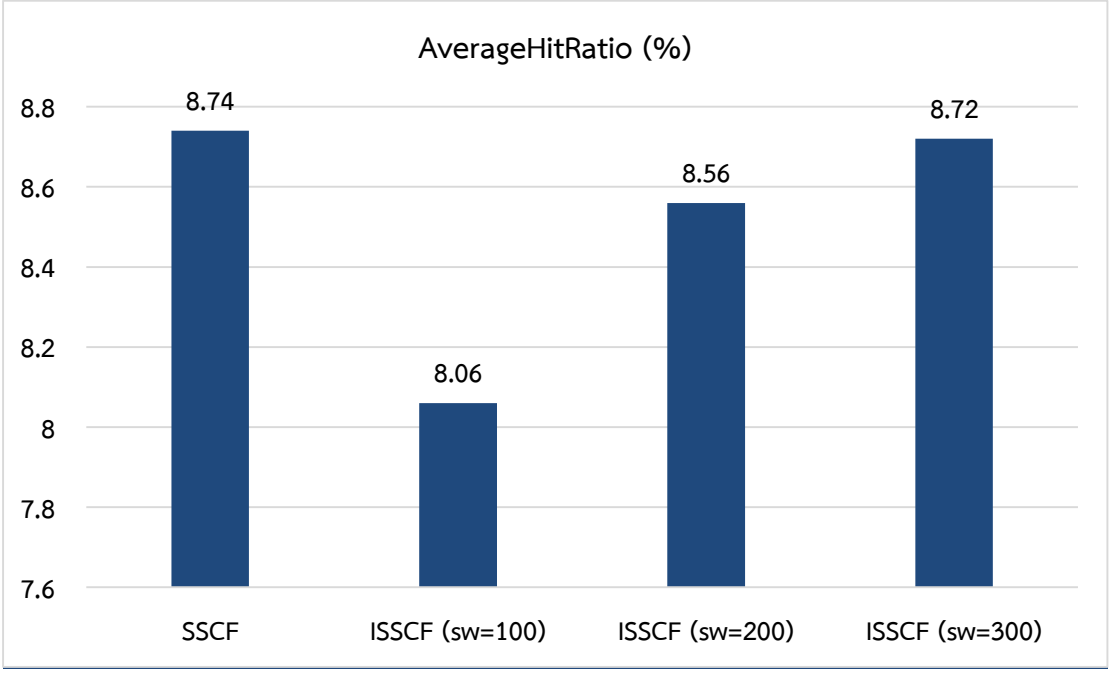
ภาพที่ 4-15 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง ($HR@10$) โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 300 เซสชั่นล่าสุด (sw=300)

ภาพที่ 4-15 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงเฉลี่ยทุกผู้ฟังในแต่ละเซสชั่น โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ร่วมกับการอัพเดทเซสชั่นที่ 300 เซสชั่นล่าสุด (sw =300) จากผลการทดลองแสดงให้เห็นถึงทั้งสองวิธีการให้ความถูกต้องที่ใกล้เคียงกันแต่วิธีการ SSCF ยังคงให้ความถูกต้องมากกว่าวิธีการ ISSCF โดยวิธีการ SSCF ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 8.74 % และวิธีการ ISSCF (sw =300) ได้ค่า HitRatio เฉลี่ยทุกเซสชั่นที่ 8.72 %

จากผลสรุปการทดลองแสดงให้เห็นถึงทั้งสองวิธีการมีความถูกต้องที่ใกล้เคียงกัน สำหรับการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของการพิจารณาวิธีการ SSCF และ ISSCF (sw=300) ได้ $Z = 0.67$ ดังนั้น $Z < Z_{0.05}$ (0.67 < 1.64) แสดงว่ายอมรับ H_0 นั่นคือ วิธีการ ISSCF (sw=300) ให้ผลความถูกต้องที่ไม่แตกต่างกับวิธีการ SSCF

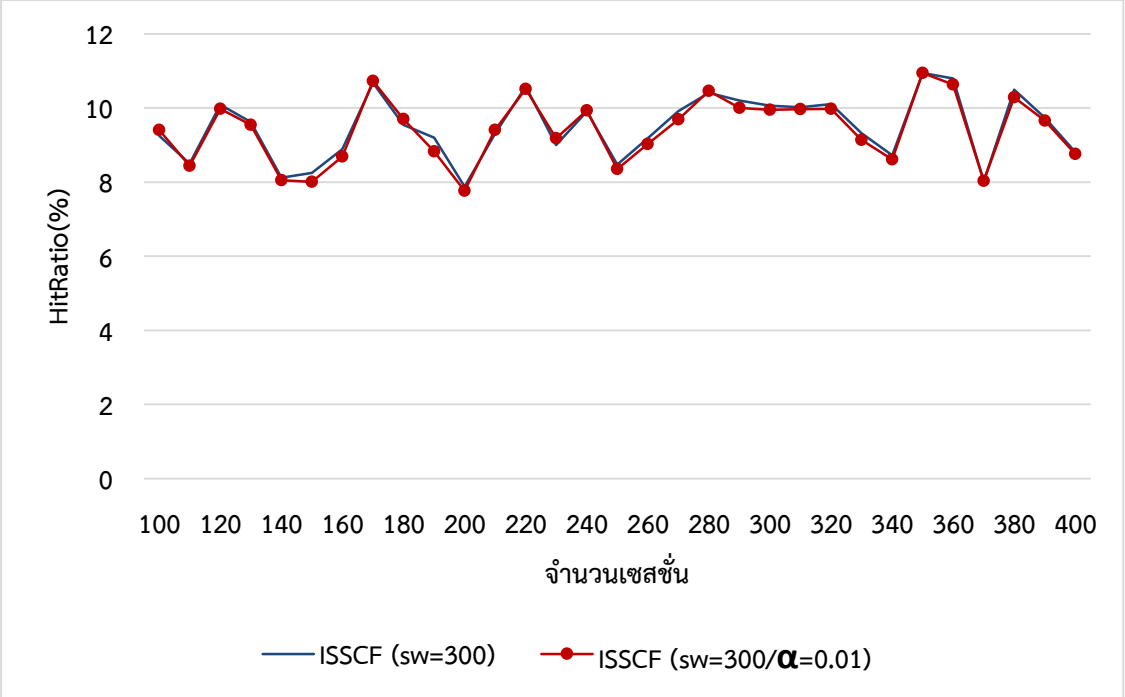
ดังนั้นในการทดลองถัดไปจึงมีการพิจารณาขนาดของข้อมูลที่ 300 เซสชั่นล่าสุด (sw=300) เนื่องจากมีความถูกต้องที่เทียบเท่ากับวิธีการ SSCF แต่สามารถให้ประสิทธิภาพในการประมวลผลข้อมูลที่ดีกว่าซึ่งมีความเหมาะสมกับระบบแนะนำเพลงแบบออนไลน์ โดยการประมวลผลข้อมูลการฟังเพลงของผู้ฟังจะไม่เพิ่มขึ้นตามเวลา แสดงผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ

AverageHitRatio ที่ 10 รายการเพลงแนะนำ โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ในแต่ละขนาดของ Sliding windows เฉลี่ยรวมทุกเซสชั่น ดังภาพที่ 4-16



ภาพที่ 4-16 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง โดยการเปรียบเทียบวิธีการ SSCF กับวิธีการ ISSCF ในแต่ละขนาดของ Sliding windows เฉลี่ยรวมทุกเซสชั่น

การทดลองที่ 2 ของฐานข้อมูลเพลง Last.fm ได้เปรียบเทียบวิธีการ ISSCF โดยการอัพเดทข้อมูลด้วยวิธี Sliding windows (sw=300) กับ ISSCF โดยการอัพเดทข้อมูลด้วยวิธี Forgetting mechanisms (Sliding windows ร่วมกับวิธีการ Fading factors) เพื่อให้ความสำคัญกับความชอบในการเลือกฟังเพลงในปัจจุบัน เนื่องจากแสดงถึงความชอบของผู้ฟังเพลงขณะนั้น โดยเปรียบเทียบค่าน้ำหนัก (α) ที่ 0.01, 0.1 และ 0.5 ตามลำดับ โดยผลการทดลองแสดงดังภาพที่ 4-17, 4-18 และ 4-19



ภาพที่ 4-17 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง โดยการเปรียบเทียบวิธีการ ISSCF ที่ sw=300 กับ ISSCF ที่ sw=300 ร่วมกับค่าน้ำหนัก (α) ที่ 0.01

จากภาพที่ 4-17 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงที่จะแนะนำให้กับผู้ฟังของวิธีการ ISSCF วิธีที่ 1 โดยพิจารณาขนาดข้อมูล (sw) ที่ 300 เซสชั่นล่าสุดร่วมกับความเร็วในการเพิ่มความสำคัญ (α) ที่ 0.01 (sw =300/α=0.01) เปรียบเทียบกับวิธีการ ISSCF ร่วมกับ Sliding windows (sw =300)

จากผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio แสดงให้เห็นว่าวิธีการ ISSCF (sw =300/α=0.01) เฉลี่ยรวมทุกเซสชั่นมีความถูกต้องที่ 8.67% ซึ่งมีความถูกต้องที่ใกล้เคียงกับ ISSCF (sw =300) ที่ 8.72 %

จากการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของทั้งสองวิธีการ กำหนดให้

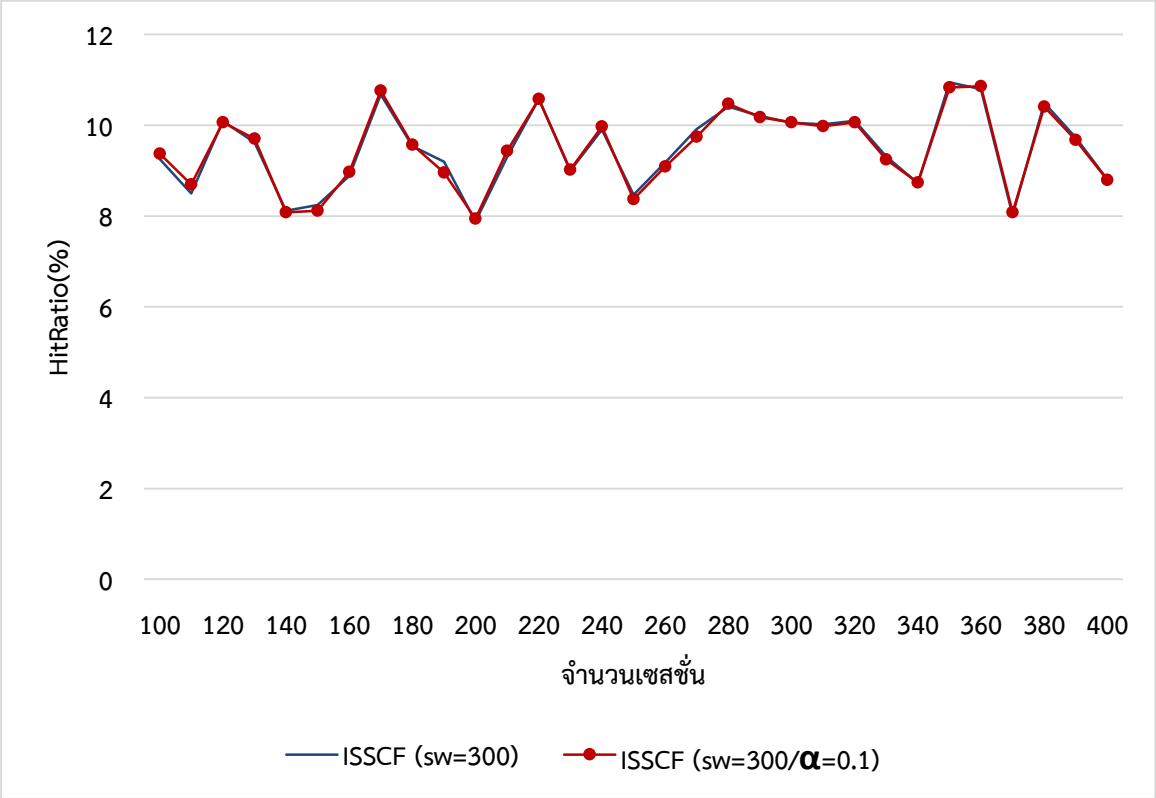
H_0 : ทั้งวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.01) มีความถูกต้องในการแนะนำเพลงที่ไม่แตกต่างกัน

$$H_0 : z_{0.1} = z$$

H_1 : ทั้งวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.01) มีความถูกต้องในการแนะนำเพลงที่แตกต่างกันอย่างมีนัยสำคัญทางสถิติ

$$H_1 : z_{0.1} \neq z$$

การพิจารณาวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.01) ได้ z = 2.67 ดังนั้น $z > z_{0.05}$ (2.67 > 1.64) แสดงว่าปฏิเสธ H_0 ยอมรับ H_1 นั่นคือวิธีการ ISSCF (sw=300) ให้ผลความถูกต้องดีกว่าวิธีการ (sw =300/α=0.01) อย่างมีนัยสำคัญทางสถิติ



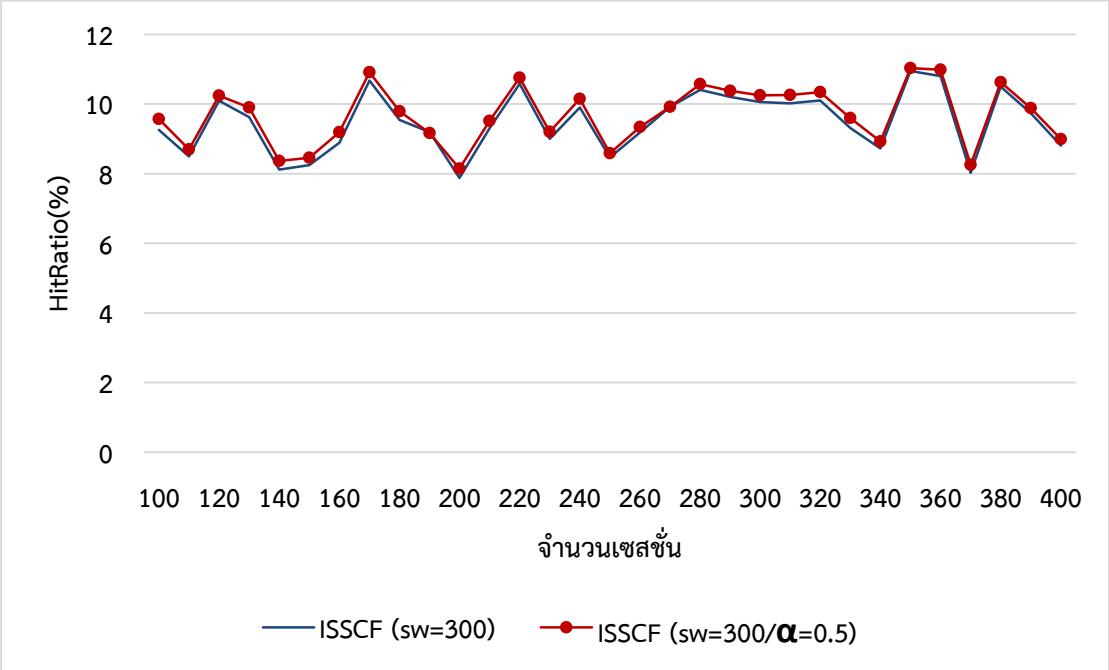
ภาพที่ 4-18 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง โดยการเปรียบเทียบวิธีการ ISSCF ที่ sw=300 กับ ISSCF ที่ sw=300 ร่วมกับพิจารณาความเร็วในการเพิ่มความสำคัญ (α) ที่ 0.1

จากภาพที่ 4-18 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงที่จะแนะนำให้กับผู้ฟังของวิธีการ ISSCF ขนาดข้อมูล (sw) ที่ 300 เซสชั่นล่าสุดร่วมกับความเร็วในการเพิ่มความสำคัญ (α) ที่ 0.1 (sw =300/α=0.1) เปรียบเทียบกับวิธีการ ISSCF ร่วมกับ Sliding windows (sw =300)

จากผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio แสดงให้เห็นว่าวิธีการ ISSCF (sw =300/α=0.1) เฉลี่ยรวมทุกเซสชั่นมีความถูกต้องที่ 8.74 % ซึ่งมีความถูกต้องที่ใกล้เคียงกับ ISSCF (sw =300) ที่ 8.72 %

จากการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.1) ได้ z = 0.33

ดังนั้น $z < z_{0.05}$ (0.33 < 1.64) แสดงว่ายอมรับ H_0 นั่นคือการสร้างรายการเพลงแนะนำด้วยวิธี ISSCF (sw =300/α=0.1) ไม่มีความแตกต่างกับวิธี ISSCF (sw =300) ในทางสถิติ



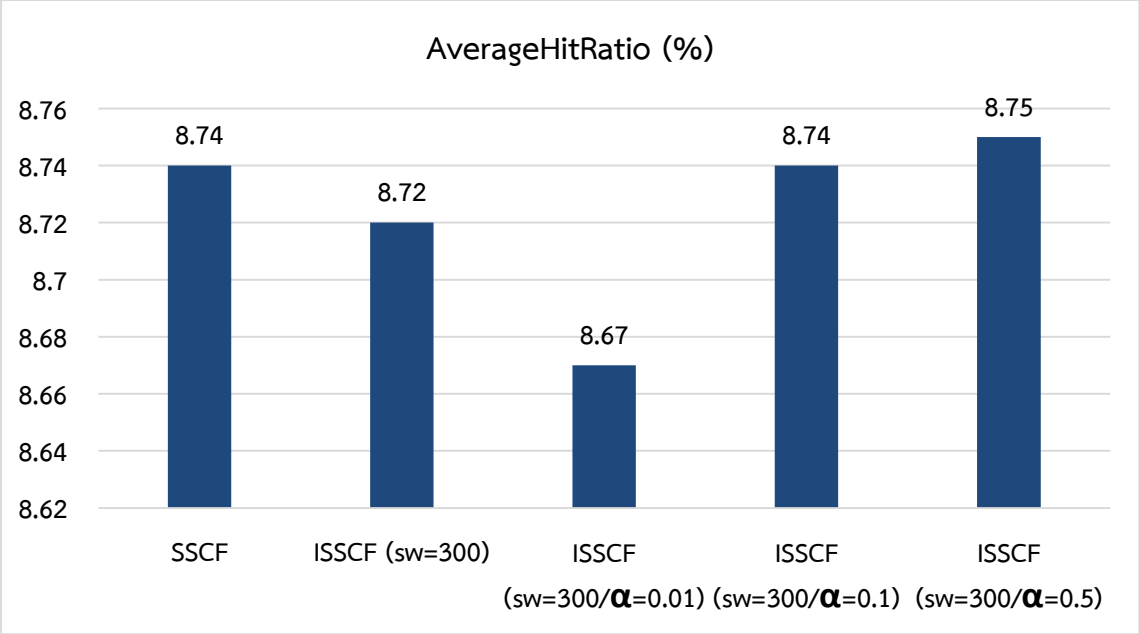
ภาพที่ 4-19 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง โดยการเปรียบเทียบวิธีการ ISSCF ที่ sw=300 กับ ISSCF ที่ sw=300 ร่วมกับพิจารณาความเร็วในการเพิ่มความสำคัญ (α) ที่ 0.5

จากภาพที่ 4-19 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงของวิธีการ ISSCF ขนาดข้อมูล (sw) ที่ 300 เซสชั่นล่าสุดร่วมกับความเร็วในการเพิ่มความสำคัญ (α) ที่ 0.5 (sw =300/α=0.5) เปรียบเทียบกับวิธีการ ISSCF ร่วมกับ Sliding windows (sw =300)

จากผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio แสดงให้เห็นว่าวิธีการ ISSCF (sw =300/α=0.5) เฉลี่ยรวมทุกเซสชั่นมีความถูกต้องที่ 8.75% ซึ่งมีความถูกต้องที่ดีกว่าวิธีการ ISSCF (sw =300) ที่ 8.72 %

ในการทดสอบความแตกต่างอย่างมีนัยสำคัญทางสถิติของวิธีการ ISSCF (sw=300) และ ISSCF (sw=300/α=0.5) ได้ z = 0.67

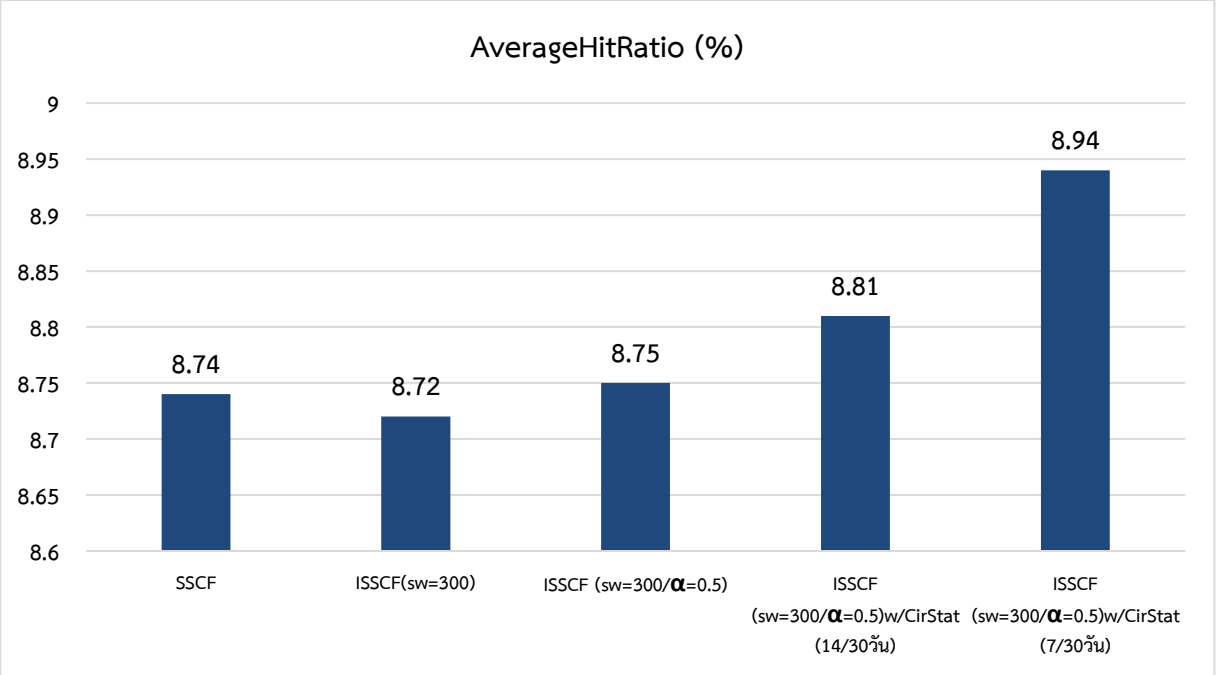
ดังนั้น $z < z_{0.05}$ (0.67 < 1.64) แสดงว่ายอมรับ H_0 นั่นคือ ISSCF (sw =300/α=0.5) ไม่มีความแตกต่างในการแนะนำเพลงกับวิธี ISSCF (sw =300) ในทางสถิติ



ภาพที่ 4-20 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ AverageHitRatio ที่ 10 รายการเพลง โดยการเปรียบเทียบวิธีการ SSCF ,ISSCF ที่ 300 เซสชั่นล่าสุด (sw =300) และวิธีการ ISSCF โดยใช้วิธี Forgetting mechanism ที่ค่าน้ำหนักในการเพิ่มความสำคัญ (α) ที่ 0.01, 0.1 และ 0.5

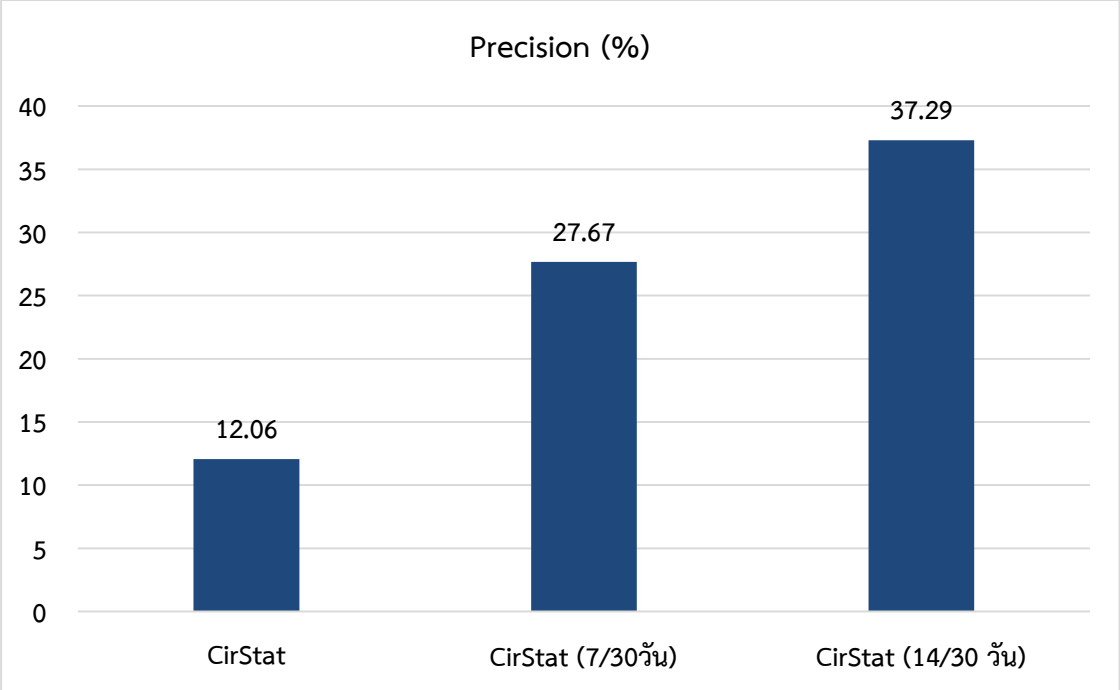
จากภาพที่ 4-20 แสดงผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ AverageHitRatio ที่ 10 รายการเพลง โดยการเปรียบเทียบวิธีการ SSCF ,ISSCF ที่ 300 เซสชั่นล่าสุด (sw=300) กับวิธีการ ISSCF โดยใช้วิธี Forgetting mechanism ที่ค่าน้ำหนักในการเพิ่มความสำคัญ (α) ที่ 0.01, 0.1 และ 0.5 ซึ่งค่า AverageHitRatio เฉลี่ยของวิธีการ ISSCF โดยใช้วิธี Forgetting mechanism ที่ค่าน้ำหนักในการเพิ่มความสำคัญที่ 0.5 ให้ผลลัพธ์ที่ดีกว่าทุกวิธีการ ดังนั้นวิธีการ ISSCF วิธีที่ 1 จึงได้เลือกใช้วิธีการ ISSCF (sw =300/α=0.5)

สำหรับวิธีการ ISSCF วิธีการที่ 2 คือการสร้างรายการเพลงโดยใช้การวิเคราะห์สถิติเชิงมุม (Circular Statistics) ซึ่งพิจารณาช่วงเวลาเฉพาะในการฟังเพลงที่มีความแตกต่างจากช่วงเวลาอื่นอย่างมีนัยสำคัญทางสถิติ โดยใช้วิธีการทดสอบทางสถิติคือ Rayleigh Z-test ที่ p-value เท่ากับ 0.05 สำหรับการอัพเดทเซสชั่นล่าสุดของวิธีการนี้ได้ใช้วิธีการ Sliding windows โดยการพิจารณาการฟังเพลงล่าสุด N วันโดยเพลงที่พิจารณาต้องถูกฟังมากกว่า k วันทำให้วิธีการนี้สร้างรายการเพลงแนะนำซึ่งยังคงความชอบของผู้ฟังในปัจจุบัน



ภาพที่ 4-21 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ของวิธีการ SSCF, ISSCF วิธีที่ 1 และการรวมรายการเพลงกับการวิเคราะห์สถิติเชิงมุม (CirStat) วิธีที่ 2

จากภาพที่ 4-21 แสดงผลความถูกต้องเปรียบเทียบระหว่างวิธีการ SSCF ,ISSCF วิธีที่ 1 ได้ค่าความถูกต้องที่ 8.75 % สำหรับวิธีการ ISSCF เมื่อพิจารณาการวิเคราะห์สถิติเชิงมุมร่วมกับวิธีการ Sliding windows ในการอัพเดทข้อมูลที่ 30 วันล่าสุดร่วมกับเพลงที่ฟังมากกว่า 14 วันนั้นได้ค่าความถูกต้องที่ 8.81 % และการพิจารณาวิธีการ Sliding windows ที่ 30 วันล่าสุดร่วมกับเพลงที่ฟังมากกว่า 7 วันนั้นได้ค่าความถูกต้องที่ 8.94 % จากผลสรุปการทดลองแสดงให้เห็นว่าวิธีการ ISSCF สำหรับวิธีที่ 1 (sw =300/α=0.5) ร่วมกับการวิเคราะห์สถิติเชิงมุมโดยใช้การอัพเดทข้อมูล 30 วันล่าสุดร่วมกับมีการฟังมากกว่า 7 วัน ได้ผลลัพธ์ที่ดีที่สุด เนื่องจากการพิจารณาเพลงที่ถูกฟังมากกว่า 7 วันสามารถสร้างรายการเพลงแนะนำให้กับผู้ฟังได้มากกว่าการพิจารณาเพลงที่ถูกฟังมากกว่า 14 วัน



ภาพที่ 4-22 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ Precision เปรียบเทียบวิธีการวิเคราะห์สถิติเชิงมุม (CirStat) และวิธีการวิเคราะห์สถิติเชิงมุมในรอบ 30 วันล่าสุดที่มีการฟังในช่วง 7 วันและ 14 วัน

ภาพที่ 4-22 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ Precision ซึ่งเป็นการพิจารณาเพลงที่ผู้ฟังเลือกฟังกับรายการเพลงที่ถูกสร้างขึ้นจากการวิเคราะห์สถิติเชิงมุม (CirStat) ซึ่งมีความถูกต้องคือ 12.06 % และวิธีการวิเคราะห์สถิติเชิงมุมในรอบ 30 วันล่าสุดที่ฟังเพลงมากกว่า 7 วันได้ค่าความถูกต้องคือ 27.67 % และการฟังเพลงมากกว่า 14 วันได้ค่าความถูกต้องที่ 37.29 % แสดงถึงการวิเคราะห์สถิติเชิงมุมสามารถให้ความถูกต้องในการสร้างรายการเพลงได้อย่างดีและการอัพเดทข้อมูลการฟังเพลงของผู้ฟังด้วยวิธี Sliding windows สามารถเพิ่มความถูกต้องให้กับรายการเพลงได้ ซึ่งการพิจารณาเพลงที่ฟังมากกว่า 14 วันจาก 30 วันล่าสุดสามารถปรับปรุงความถูกต้องได้ดีกว่าการไม่พิจารณาวิธีการ Sliding windows ถึง 25.23 %

จากตารางที่ 4-11 แสดงผลความถูกต้องที่ดีที่สุดของวิธีการสร้างรายการเพลงแนะนำ SSCF, ISSCF วิธีที่ 1 และการรวมรายการเพลงแนะนำ ISSCF ของวิธีที่ 1 ร่วมกับการวิเคราะห์สถิติเชิงมุม (CirStat) วิธีที่ 2 โดยใช้ตัววัดประสิทธิภาพ HitRatio ในฐานข้อมูลเพลง Last.fm

ตารางที่ 4-11 ผลความถูกต้องของวิธีการสร้างรายการเพลงแนะนำโดยใช้ตัววัดประสิทธิภาพ HitRatio

วิธีการ	HitRatio (%)
SSCF	8.74
ISSCF (sw=300)	8.72
ISSCF (sw=300/α=0.5)	8.75
ISSCF (sw=300/α=0.5) ร่วมกับ CirStat (7/30วัน)	8.94

จากตารางที่ 4-12 แสดงผลความถูกต้องของวิธีการสร้างรายการเพลงแนะนำ ISSCF วิธีที่ 2 คือการวิเคราะห์สถิติเชิงมุม (CirStat) โดยใช้ตัววัดประสิทธิภาพ Precision ในฐานข้อมูลเพลง Last.fm

ตารางที่ 4-12 ผลความถูกต้องของวิธีการสร้างรายการเพลงแนะนำ ISSCF วิธีที่ 2 (การวิเคราะห์สถิติเชิงมุม ,CirStat) โดยใช้ตัววัดประสิทธิภาพ Precision

วิธีการ	Precision (%)
CirStat	12.06
CirStat (7/30วัน)	27.67
CirStat (14/30 วัน)	37.29

4.4.1 การวัดประสิทธิภาพเชิงเวลาของฐานข้อมูลเพลง Last.fm

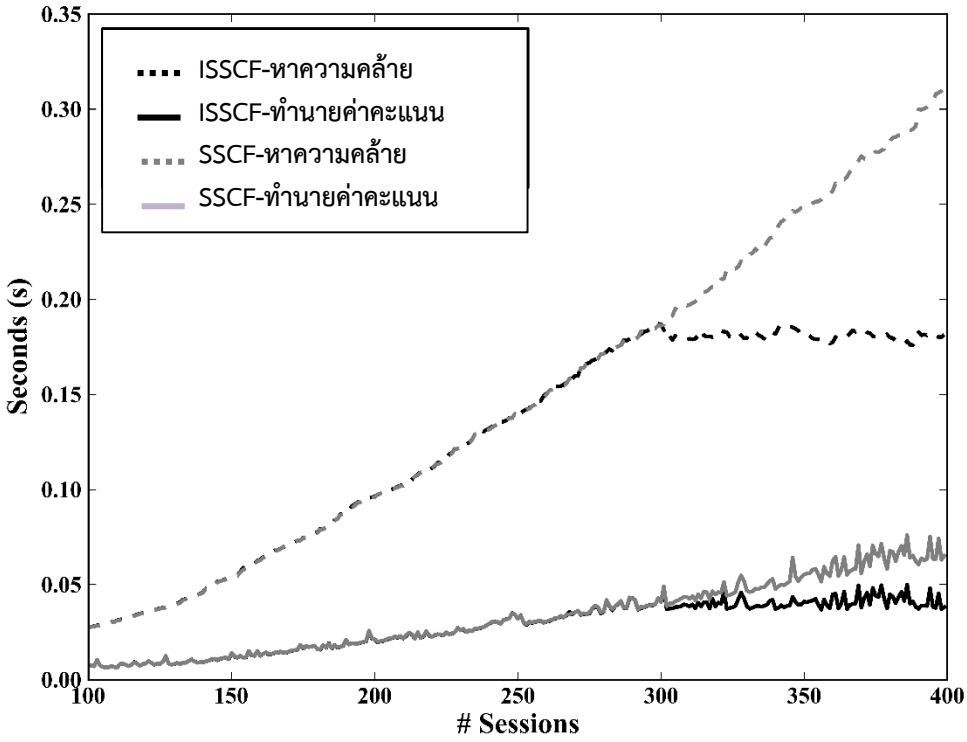
สำหรับการวัดประสิทธิภาพเชิงเวลาของวิธีการ Incremental Session based Collaborative Filtering (ISSCF) เปรียบเทียบกับวิธีการ Session based Collaborative Filtering (SSCF) แสดงเวลาที่ใช้ในการคำนวณของขั้นตอนหาความคล้ายของเซสชั่นใกล้เคียงและขั้นตอนการทำนายค่าที่เซสชั่นขนาด 200 เซสชั่น ดังตารางที่ 4-13

ตารางที่ 4-13 เวลาที่ใช้ประมวลผลของวิธีการ SSCF และ ISSCF ฐานข้อมูล Last.fm

วิธีการ	ขั้นตอนหาความคล้าย	ขั้นตอนการทำนายค่า
Offline SSCF	14.90 วินาที	0.027 วินาที
ISSCF	**0.141 วินาที**	**0.027 วินาที**

จากภาพที่ 4-23 แสดงเวลาที่ใช้ในการคำนวณของวิธีการ SSCF เปรียบเทียบกับวิธีการ ISSCF โดยพิจารณาที่ 100 เซสชั่นถึง 400 เซสชั่นซึ่งใช้สำหรับเป็นข้อมูลทดสอบ โดยพิจารณาเวลาที่

ใช้คำนวณเฉลี่ย 20 ผู้ฟัง จากภาพแสดงถึงวิธีการ ISSCF โดยใช้ขนาดของข้อมูล (sw=300) ซึ่งมีการตัดเซสชั่นที่เก่าที่สุดเมื่อเซสชั่นใหม่ถูกอัพเดททำให้มีการใช้เวลาในการคำนวณขั้นตอนการหาความคล้ายและขั้นตอนการทำนายค่าคะแนนความชอบที่คงที่เปรียบเทียบกับวิธีการ SSCF ซึ่งไม่มีการตัดเซสชั่นเก่าโดยใช้เซสชั่นของผู้ฟังทั้งหมดในการพิจารณาการคำนวณขั้นตอนการหาความคล้ายและขั้นตอนการทำนายค่าคะแนนความชอบมีเวลาที่เพิ่มขึ้นตามจำนวนเซสชั่นของผู้ฟังเพลง



ภาพที่ 4-23 เวลาที่ใช้ในการหาความคล้ายของเซสชั่นและการทำนายค่าคะแนนความชอบของผู้ฟังเพลงของวิธีการ SSCF และ ISSCF

บทที่ 5

สรุปผลการดำเนินงานและข้อเสนอแนะ

สำหรับวิทยานิพนธ์นี้ ผู้จัดทำวิทยานิพนธ์จะกล่าวถึงการสรุปและวิจารณ์ผลการดำเนินโครงงาน ได้แก่ การสรุปผลการทดลอง และการวิจารณ์เกี่ยวกับข้อดี ข้อบกพร่องและรวมถึงข้อเสนอแนะของงานวิทยานิพนธ์ ดังนี้

สรุปผลการดำเนินงาน

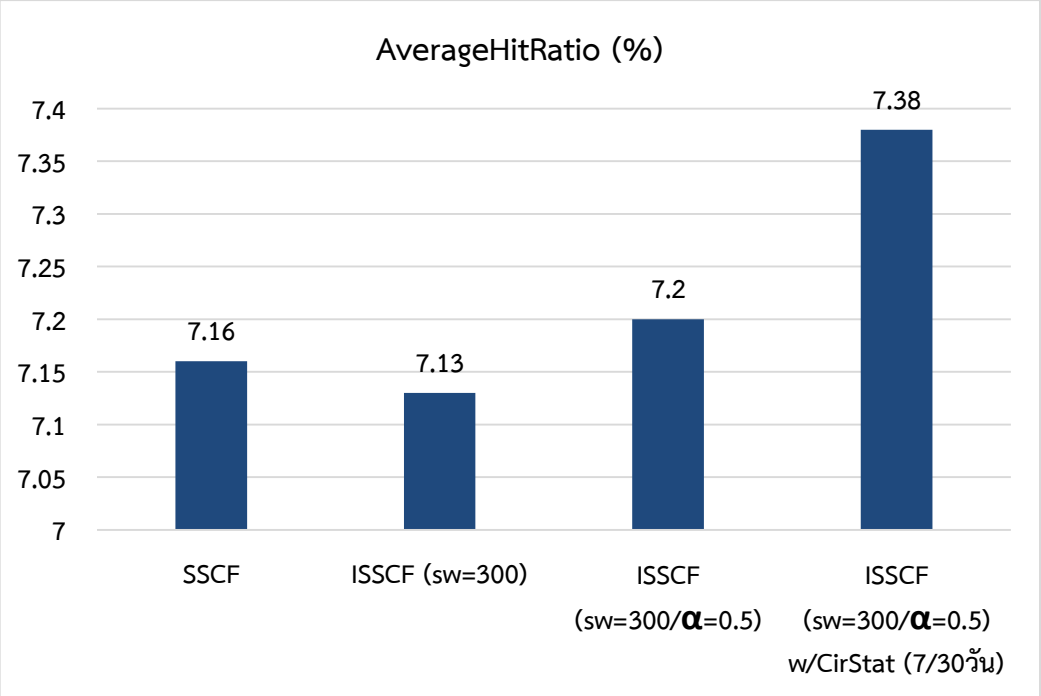
วิทยานิพนธ์นี้ได้นำเสนอวิธีการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง โดยใช้วิธีการ **Incremental Session-based Collaborative Filtering (ISSCF)** โดยสร้างรายการเพลงจาก 2 วิธีการ คือ วิธีที่ 1 ได้พิจารณาความชอบของผู้ฟังเพลงจากเซสชั่นปัจจุบันในขณะกำลังฟังเพลงซึ่งมีความคล้ายกับเซสชั่นในอดีต เนื่องจากพฤติกรรมการฟังเพลงของผู้ฟังมักจะมีการฟังเพลงที่ต่อเนื่องและมีการฟังซ้ำในเพลงที่ชื่นชอบ โดยขั้นตอนวิธีการที่นำเสนอได้ปรับปรุงจากขั้นตอนวิธีการ SSCF (Park, S. E., et al, 2011) ซึ่งปรับวิธีการแนะนำเพลงแบบออฟไลน์เป็นการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังแบบออนไลน์โดยการพิจารณาวิธีการ Sliding windows (sw) และวิธีการ Forgetting mechanism ซึ่งประกอบด้วยการใช้ Sliding windows ร่วมกับ Fading factors ในการตัดเซสชั่นเก่าและให้ค่าน้ำหนักกับเซสชั่นใหม่ และวิธีที่ 2 สำหรับการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังคือการพิจารณาความชอบของผู้ฟังเฉพาะช่วงเวลาว่ามักจะชอบฟังเพลงช่วงเวลาใดและชอบฟังเพลงอะไรโดยใช้การวิเคราะห์สถิติเชิงมุม (Circular Statistics) ร่วมกับพิจารณาข้อมูลการฟังเพลงของผู้ฟังล่าสุดโดยใช้วิธีการ Sliding windows

จากผลการทดลองที่นำเสนอในวิทยานิพนธ์นี้จากฐานข้อมูลเพลง 30Music โดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง ได้ผลสรุปการดำเนินงานดังนี้ วิธีการ SSCF ได้ผลความถูกต้องคือ 7.16 % วิธีการ ISSCF วิธีที่ 1 คือ การใช้วิธีการ ISSCF ร่วมกับ Sliding window ด้วยขนาดข้อมูลที่ 300 เซสชั่น (sw=300) ได้ผลความถูกต้องคือ 7.13 % การใช้วิธีการ ISSCF ร่วมกับ Forgetting mechanism ได้ผลความถูกต้องคือ 7.20 % และวิธีการ ISSCF ซึ่งรวมทั้ง 2 วิธีการได้ผลความถูกต้องคือ 7.38 % แสดงให้เห็นว่าวิธีการสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟังที่นำเสนอในวิทยานิพนธ์นี้ได้ผลความถูกต้องที่ดีที่สุด แสดงดังภาพที่ 5-1

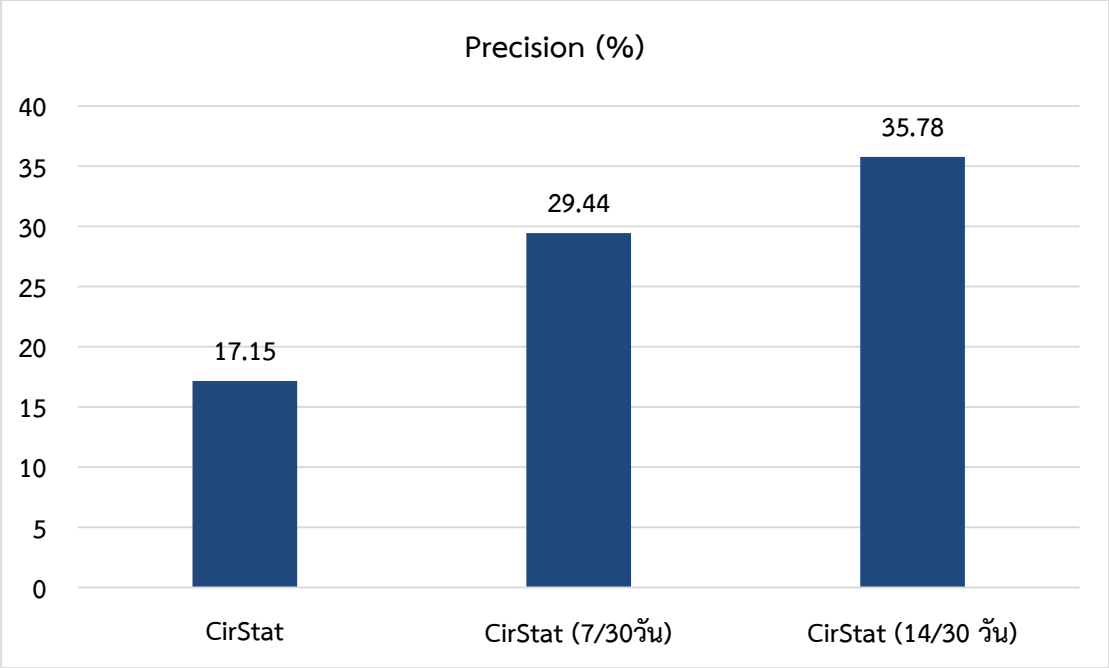
สำหรับการวัดประสิทธิภาพค่า Precision ที่ 10 รายการเพลงซึ่งมีความเหมาะสมในการวัดความถูกต้องของวิธีการวิเคราะห์สถิติเชิงมุม (Circular statistics) มีความถูกต้อง คือ 17.15 % และการวิเคราะห์สถิติเชิงมุมในรอบ 30 วันล่าสุดที่เพลงถูกฟังมากกว่า 7 วันได้ค่าความถูกต้อง คือ 29.44 % และเพลงถูกฟังมากกว่า 14 วันได้ค่าความถูกต้องที่ 35.78 % จากผลสรุปการทดลองแสดงถึงวิธีการวิเคราะห์สถิติเชิงมุมสามารถให้ความถูกต้องในการแนะนำเพลงได้อย่างดีและการอัพเดทข้อมูลการฟัง

เพลงของผู้ฟังด้วยวิธี Sliding windows สามารถเพิ่มความถูกต้องให้กับรายการเพลงที่จะแนะนำให้กับผู้ฟังได้ แสดงดังภาพที่ 5-2

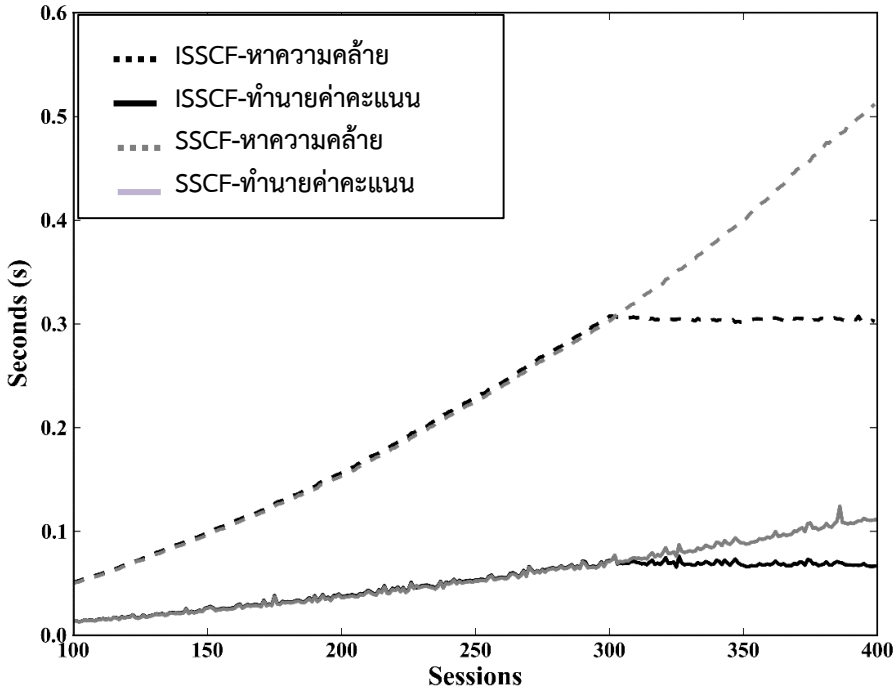
นอกจากนี้วิธีการ ISSCF มีความเหมาะสมกับการสร้างรายการเพลงแบบออนไลน์ โดยการคำนวณเวลาที่ใช้ในการหาความคล้ายของเซสชั่นและการทำนายค่าคะแนนความชอบของผู้ฟังเพลงของวิธีการ SSCF และ ISSCF ดังภาพที่ 5-3 ซึ่งวิธีการ ISSCF มีการพิจารณาข้อมูลล่าสุดโดยการตัดเซสชั่นเก่าของผู้ฟังเพลง ส่งผลให้การคำนวณรายการเพลงมีการใช้เวลาประมวลผลที่คงที่ไม่เพิ่มขึ้นตามจำนวนเซสชั่นของผู้ฟังเพลงซึ่งช่วยแก้ปัญหาการประมวลผลข้อมูลที่มีปริมาณมากได้ (Scalability problem) โดยใช้เวลาในการคำนวณที่รวดเร็วกว่าวิธีการ SSCF



ภาพที่ 5-1 ผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงจากฐานข้อมูลเพลง 30Music



ภาพที่ 5-2 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ Precision เปรียบเทียบวิธีการวิเคราะห์สถิติเชิงมุม (CirStat) และวิธีการวิเคราะห์สถิติเชิงมุมในรอบ 30 วันล่าสุดที่มีการฟังในช่วง 7 วันและ 14 วัน

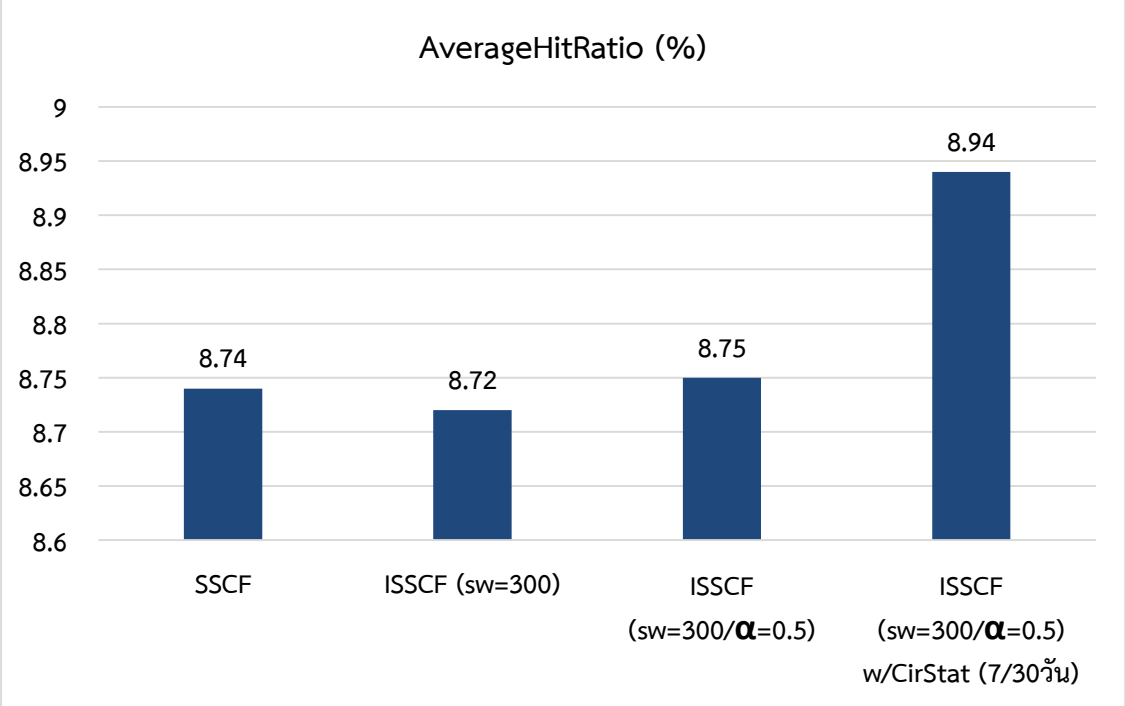


ภาพที่ 5-3 เวลาที่ใช้ในการหาความคล้ายของเซสชั่นและการทำนายค่าคะแนนความชอบของผู้ฟังเพลงของวิธีการ SSCF และ ISSCF จากฐานข้อมูลเพลง 30Music

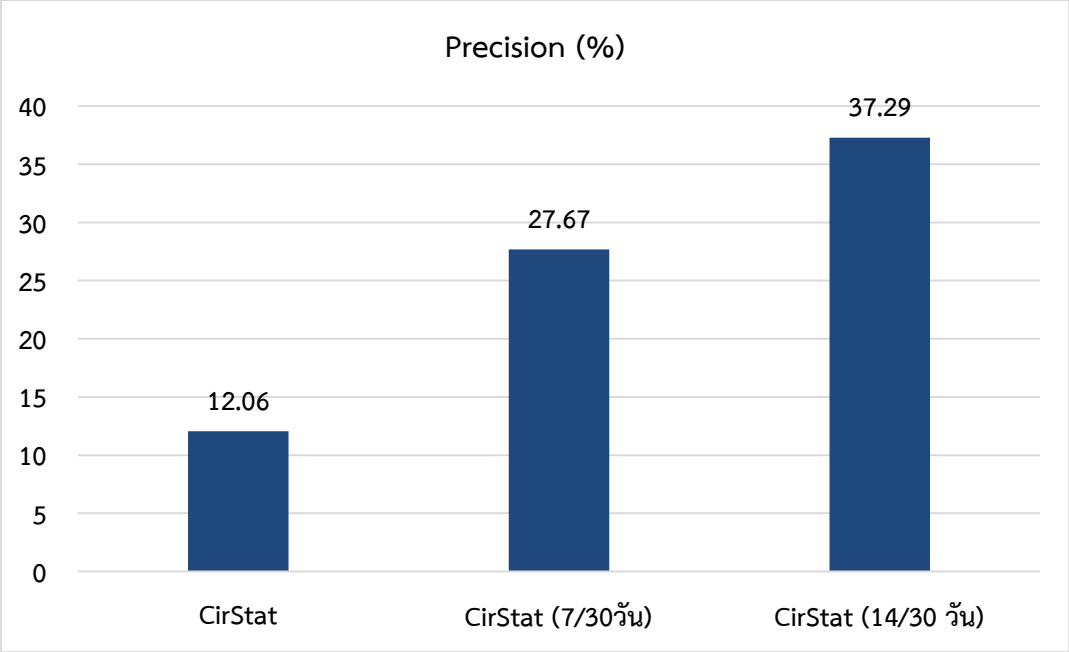
สำหรับฐานข้อมูลเพลง Last.fm โดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลง ได้ผลสรุปการดำเนินงานดังนี้ วิธีการ SSCF ได้ผลความถูกต้องคือ 8.74 % วิธีการ ISSCF วิธีที่ 1 คือ การใช้วิธีการ ISSCF ร่วมกับ Sliding window ด้วยขนาดข้อมูลที่ 300 เซสชั่น (sw=300) ได้ผลความถูกต้องคือ 8.72 % การใช้วิธีการ ISSCF ร่วมกับ Forgetting mechanism ได้ผลความถูกต้องคือ 8.75 % และวิธีการ ISSCF ซึ่งรวมทั้ง 2 วิธีการได้ผลความถูกต้องคือ 8.94 % แสดงดังภาพที่ 5-4 ซึ่งวิธีการ ISSCF ได้ประสิทธิภาพความถูกต้องที่ดีที่สุด

ในการวัดประสิทธิภาพค่า Precision ที่ 10 รายการเพลงซึ่งมีความเหมาะสมในการวัดความถูกต้องของวิธีการวิเคราะห์สถิติเชิงมุม (Circular statistics) ซึ่งมีความถูกต้องคือ 12.06 % และการวิเคราะห์สถิติเชิงมุมในรอบ 30 วันล่าสุดที่เพลงถูกฟังมากกว่า 7 วันได้ค่าความถูกต้องคือ 27.67 % และเพลงที่ถูกฟังมากกว่า 14 วันได้ค่าความถูกต้องที่ 37.29 % แสดงถึงการวิเคราะห์สถิติเชิงมุมสามารถให้ความถูกต้องในการสร้างรายการเพลงได้อย่างดีและการอัพเดทข้อมูลการฟังเพลงของผู้ฟังด้วยวิธี Sliding windows สามารถเพิ่มความถูกต้องให้กับรายการเพลงได้ แสดงดังภาพที่ 5-5

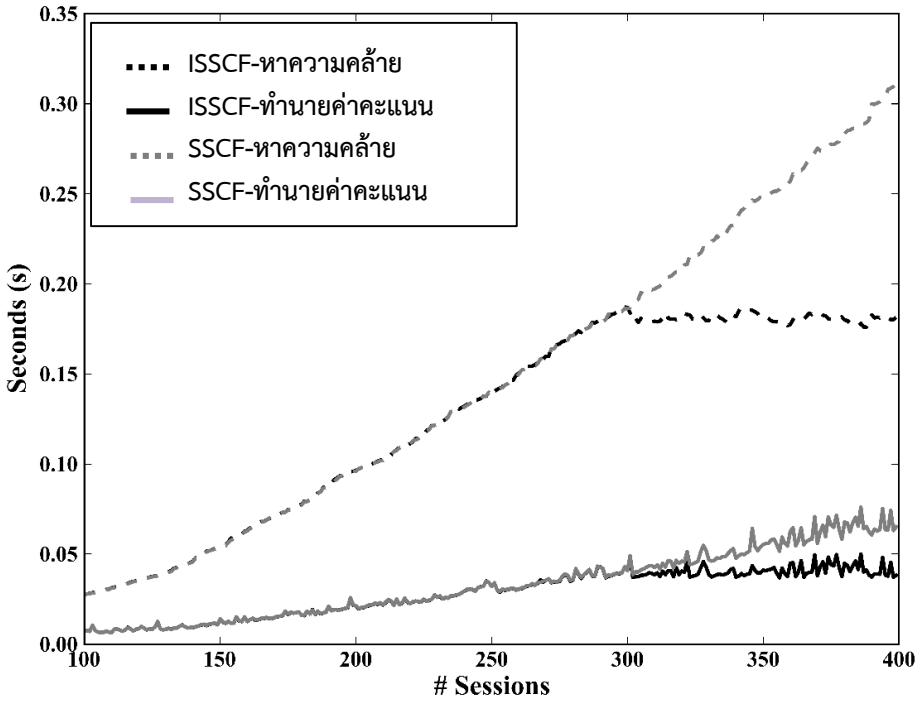
สำหรับการพิจารณาเวลาที่ใช้ในการคำนวณที่มีประสิทธิภาพ แสดงเวลาที่ใช้ในการหาความคล้ายของเซสชั่นและการทำนายค่าคะแนนความชอบเพื่อสร้างรายการเพลงที่จะแนะนำให้กับผู้ฟัง ดังภาพที่ 5-6



ภาพที่ 5-4 แสดงผลความถูกต้องโดยใช้วิธีการวัดประสิทธิภาพ HitRatio ที่ 10 รายการเพลงจากฐานข้อมูลเพลง Last.fm



ภาพที่ 5-5 ผลความถูกต้องเฉลี่ยโดยใช้วิธีการวัดประสิทธิภาพ Precision เปรียบเทียบวิธีการวิเคราะห์สถิติเชิงมุม (CirStat) และวิธีการวิเคราะห์สถิติเชิงมุมในรอบ 30 วันล่าสุดที่มีการฟังในช่วง 7 วันและ 14 วัน



ภาพที่ 5-6 เวลาที่ใช้ในการหาความคล้ายของเซสชั่นและการทำนายค่าคะแนนความชอบของผู้ฟังเพลงของวิธีการ SSCF และ ISSCF

วิจารณ์ผลการดำเนินงาน

ข้อดีของวิทยานิพนธ์

1. วิธีการ ISSCF ได้พิจารณาเซสชั่นในการสร้างรายการเพลงซึ่งเหมาะสมกับพฤติกรรมของผู้ฟัง
2. การวิเคราะห์สถิติเชิงมุม (Circular Statistics) สามารถแนะนำเพลงที่ตรงตามความชอบของผู้ฟังตามช่วงเวลาเฉพาะที่ผู้ฟังชื่นชอบได้
3. การอัพเดทข้อมูลผู้ฟังโดยใช้วิธีการ Sliding windows สามารถให้ความรวดเร็วในการคำนวณและยังคงความชอบของผู้ฟังในปัจจุบัน
4. การพิจารณาความสำคัญของเพลงที่ถูกเลือกฟังล่าสุดโดยใช้ Fading factor ทำให้เพลงที่แนะนำให้กับผู้ฟังตรงกับความชอบของผู้ฟังในขณะฟังเพลงได้

ข้อบกพร่องของวิทยานิพนธ์

1. การพิจารณาขนาดข้อมูลที่คงที่ (sw) กับทุกผู้ฟังนั้นอาจไม่เหมาะสมกับการสร้างรายการเพลงที่ตรงกับความชอบของผู้ฟัง
2. วิธีการ ISSCF มีการสร้างรายการเพลงแนะนำโดยพิจารณาประวัติการฟังเพลงในอดีตทำให้ผู้ฟังไม่ได้รับการแนะนำเพลงใหม่ที่ผู้ฟังไม่เคยฟังมาก่อน

ข้อจำกัดของวิทยานิพนธ์

1. วิทยานิพนธ์นี้ได้ใช้การทดลองโดยใช้ข้อมูลทั้งหมด 400 เซสชั่น สำหรับทุกผู้ฟัง
2. ฐานข้อมูลเพลง 30Music พิจารณาช่วงระยะห่างของเซสชั่นที่ 800 วินาทีและฐานข้อมูลเพลง Last.fm พิจารณาช่วงของเซสชั่นที่ 30 นาที
3. ใช้การพิจารณาความคล้ายของเซสชั่นที่ค่าเกณฑ์ความคล้าย (th) เท่ากับ 0.1

ข้อเสนอแนะของวิทยานิพนธ์

1. การสร้างรายการเพลงควรมีการพิจารณาเพลงใหม่ที่ผู้ฟังอาจจะชื่นชอบแต่ไม่เคยได้รับฟังโดยการพิจารณาความชอบในการเลือกฟังเพลงจากผู้ฟังคนอื่นในระบบที่มีพฤติกรรมการฟังเพลงที่คล้ายกับผู้ฟังเป้าหมาย
2. ขนาดของข้อมูล (sw) ที่ใช้ในการสร้างรายการเพลงควรมีการพิจารณาขนาดข้อมูลที่เหมาะสมกับผู้ฟังแต่ละบุคคล

บรรณานุกรม

Linden, G., Smith, B., & York, J. (2003). Amazon.com Recommendations Item-to-Item Collaborative Filtering. *Internet Computing*, IEEE, 7(1), 76-80.

Davidson, J., Liebald, B., Liu, J., Nandy, P., & Vleet, T. (2010) The YouTube Video Recommendation System. *RecSys2010*, 293-296.

Calandrino J. A., Kilzer A., Narayanan A., Felten E. W., & Shmatikov V. (2011) "You Might Also Like: " Privacy Risks of Collaborative Filtering. *In Proceedings of the 2011 IEEE Symposium on Security and Privacy (SP '11)*, IEEE Computer Society, Washington, DC, USA, 231-246.

Lee, J. S. & Lee, J. C. (2007) Context awareness by case-based reasoning in a music recommendation system. *In Proceedings of the 4th international conference on Ubiquitous computing systems (UCS'07),* 45-58.

Park, C. H. & Kahng, M. (2010) Temporal dynamics in music listening behavior : a case study of online music service. *Computer and Information Science (ICIS)*, IEEE, 573-578.

Song, Y., Dixon S., Pearce, M. (2012) A Survey of Music Recommendation Systems and Future Perspectives. *In Proc. 9th Int. Symp. Computer Music Modelling and Retrieval (CMMR),* 395–410.

Kaminskas, M. & Ricci, F. (2012) Contextual music information retrieval and recommendation: state of the art and challenges. *Computer Science Review*, 6:89--119, 2012.

Deng, S., Wang, D., Li, X. & Xu, G. (2015) Exploring user emotion in microblogs for music recommendation. *Expert Syst. Appl*, 9284-9293.

Park, S. E., Lee, S. & Lee, S. (2011) Session-Based Collaborative Filtering for Predicting the Next Song. *Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference on*, 353-358.

Dias, R. & Fonseca, M.J. (2013) Improving music recommendation in session-based collaborative filtering by using temporal context. *Tools with Artificial Intelligence (ICTAI), 2013 IEEE 25th International Conference on*, 783-788

Carlsson, O. (2014) Cluster User Music Sessions. (Master of Science Thesis in Computer Science: Algorithms. Languages and Logic, Department of Computer Science & Engineering Chalmers University of Technology Gothenburg)

Baltrunas, L., & Amatriain, X. (2009) Towards time-dependant recommendation based on implicit feedback. *In Workshop on context-aware recommender systems (CARS'09).*

Cebrián, T., Planagumà, M., Villegas, P. & Amatriain, X. (2010) Music recommendations with temporal context awareness. *Proceedings of the Fourth ACM Conference on Recommender Systems (RecSys'2010)*. ACM, New York, NY, USA, 349-352.

Herrera, P., Resa, Z. & Sordo, M. (2010) Rocking around the clock eight days a week: an exploration of temporal patterns of music listening. *1st Workshop On Music Recommendation And Discovery (WOMRAD), ACM RecSys, 2010, Barcelona, Spain*

Schedl, M., Breitschopf, G. & Ionescu, B. (2014) Mobile Music Genius: Reggae at the Beach, Metal on a Friday Night?. *Proceedings of International Conference on Multimedia Retrieval,* Glasgow, UK.

Hu, Y. & Ogihara, M. (2011) NEXTONE PLAYER: A MUSIC RECOMMENDATION SYSTEM BASED ON USER BEHAVIOR. *12th International Society for Music Information Retrieval Conference (ISMIR 2011),* 103-108

Papagelis, M., Rousidis, I., Plexousakis, D. & Theoharopoulos, E. (2005) Incremental Collaborative Filtering for Highly Scalable Recommendation Algorithms. *In Proceedings of the 15th international conference on Foundations of Intelligent Systems (ISMIS'05)*, Springer-Verlag, Berlin, Heidelberg, 553-561.

Miranda, C. & Jorge, A. M. (2008) Incremental collaborative filtering for binary ratings. *Web Intelligence and Intelligent Agent Technology, 2008.* 389-392.

Yang, X., Zhang, Z., & Wang, K. (2012) Scalable Collaborative Filtering Using Incremental Update and Local Link Prediction. *In Proceedings of the 21st ACM international conference on Information and knowledge management (CIKM '12).* ACM, New York, NY, USA, 2371-2374.

Vinagre, J. & Jorge, A. M. (2012) Forgetting mechanisms for scalable collaborative filtering. *Journal of the Brazilian Computer Society*. Volume 18, Issue 4, 271-282.

Dietterich, T. G. (1998) Approximate Statistical Tests for Comparing Supervised Classication Learning Algorithms. *Neural Computation* 1895-1923

Turrin, R., Quadrana, M., Condorelli, A., Pagano, R. & Cremones, P. (2015) 30Music listening and playlists dataset. *RecSys 2015* Poster Proceedings, September 16-20, 2015, Austria, Vienna

Candillier, L., Jack, K., Fessant, F. & Meyer, F., State-of-the-Art Recommender Systems , IGI Global, 2009.

Oscar, C., Music Recommendation and Discovery in the Long Tail, Springer, 2010.
Zar, J.H. (1984) Biostatistical Analysis. Eaglewood Cliffs, NJ; Prentice-Hall, Inc., 719pp.
Ke, J., Sun, R., Su, W. & Li, X. (2015) Next-song recommendation with temporal dynamics, Knowledge-Based Systems. Volume 88 Issue C, November 2015

ภาคผนวก

ภาคผนวก ก

การเผยแพร่ผลงานวิจัย

ขั้นตอนวิธีการทำนายค่าคะแนนความชอบที่มีประสิทธิภาพสำหรับระบบแนะนำเพลง โดยใช้วิธีการเทนเดนซีเบสร่วมกับข้อมูลไมโครโพรไฟล์

The Efficient Rating Prediction Algorithm for Music Recommender System by Using Tendencies Based Algorithm with Micro Profiles

สุเมธ ดาราพิสุทธิ์ (Sumet Darapisut)[1] และ จักริน สุขสวัสดิ์ชน (Jakkarin Suksawatchon)[2]

คณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา

[1]dearsumet@gmail.com, [2]jakkaman@gmail.com

บทคัดย่อ

งานวิจัยนี้นำเสนอขั้นตอนวิธีการทำนายค่าคะแนนความชอบที่มีประสิทธิภาพสำหรับระบบแนะนำเพลงชื่อว่า "Tendencies Based Algorithm with Micro Profiles& Sparsity Problem (TBMS)" ซึ่งเป็นการประยุกต์ใช้วิธีการ Tendencies Based Algorithm ร่วมกับบริบททางด้านเวลามาแนะนำเพลง โดยการแบ่งข้อมูลการฟังเพลงของผู้ใช้ออกเป็นข้อมูลการฟังเพลงในแต่ละช่วงเวลาที่เรียกว่า "Micro Profiles" ทั้งหมด 5 ชุดด้วยกันรวมถึงงานวิจัยนี้ยังพิจารณาการแก้ไขปัญหาความเบาบางของข้อมูลโดยการเติมค่าข้อมูลก่อนการทำนายเพื่อเพิ่มประสิทธิภาพของการทำนายที่ดีขึ้นกว่าขั้นตอนการทำนายแบบเดิมที่นิยมใช้วิธีการกรองร่วม ส่วนข้อมูลที่ใช้ในงานวิจัยนี้เป็นข้อมูลการฟังเพลงจากฐานข้อมูลของ last.fm ที่ประกอบด้วยประวัติการฟังเพลงของผู้ใช้แต่ละคนซึ่งจากผลการทดลองแสดงให้เห็นว่าวิธีการที่นำเสนอ TBMS นี้มีประสิทธิภาพ โดยให้ผลการทำนายความชอบที่แม่นยำและมีความรวดเร็วในการประมวลผลมากกว่าวิธีการกรองร่วมแบบดั้งเดิม

คำสำคัญ: ระบบแนะนำเพลง เวลา ไมโครโพรไฟล์ ปัญหาความเบาบางของข้อมูล

Abstract

This research presents the efficient rating prediction algorithm for music recommender system named "Tendencies Based Algorithm with Micro Profiles & Sparsity Problem (TBMS)", which adopts Tendencies Based Algorithm incorporated with time contextual. The user profile is splited into several sub-profile based on time contextual called "Micro Profiles"; totally 5 sets, each representing the user in particular context. In addition, the research also considers to solve the sparsity problem by imputation the missing values to enhance the performance of prediction process better than the traditional algorithm -- collaborative filtering technique. For all experiments, this research used last.fm data set which is composed of the music listening of each users. From our experiments, we have found that the proposed method TBMS is effective. It is give more accuracy and much more efficient computionally than tradition collaborative filtering.

Keyword: music recommender system, time, Micro Profiles, sparsity problem.

1. บทนำ

ระบบแนะนำเพลงหรือที่เรียกว่าโปรแกรมแนะนำเพลงเป็นโปรแกรมที่ช่วยในการค้นหาและแนะนำเพลงที่ผู้ใช้ต้องการฟัง โดยจะวิเคราะห์จากประวัติการฟังเพลงของผู้ใช้ว่าชอบฟังเพลงแนวใด และโปรแกรมก็จะสร้างรายการแนะนำเพลงสำหรับผู้ใช้คนนั้น เช่นแนะนำตามความชอบ แนะนำตามแนวเพลง แนะนำตามศิลปิน หรือแนะนำตามช่วงเวลาที่ฟังเพลง โปรแกรมแนะนำเพลงที่เป็นที่นิยมในปัจจุบันได้แก่ Pandora, Songza, Last.fm, iTunes Genius เป็นต้น

ในปัจจุบันเทคนิคที่ใช้สร้างระบบแนะนำเพลงที่นิยมใช้ได้แก่ วิธีการกรองร่วม (Collaborative filtering) [1,2,3] การแนะนำจะพิจารณาแนะนำเพลงให้กับผู้ใช้จากเพลงที่บุคคลอื่นมีความชอบคล้ายกับผู้ใช้ โดยที่ไม่จำเป็นต้องรู้จักคนเหล่านั้นแล้วนำการแนะนำของบุคคลเหล่านั้นมาช่วยตัดสินใจเลือกเพลงแต่วิธีการนี้ใช้ค่าความชอบหรือคะแนน (Rating) ที่ผู้ใช้มีต่อเพลงแต่ละเพลงซึ่งต้องจัดเก็บค่าไว้ให้กับระบบแนะนำเสียก่อน และหลายครั้งทำให้วิธีการกรองร่วมมีปัญหาที่เกิดขึ้นสำหรับการแนะนำเช่น ปัญหาความเบาบางของข้อมูล (Sparsity Problem) และการแนะนำเพลงที่ไม่ค่อยเป็นที่รู้จัก ซึ่งเกิดจากสาเหตุที่ระบบแนะนำเพลงโดยทั่วไปมีการใช้ปัจจัยเพียงแค่ 2 ปัจจัยสำหรับสร้างระบบแนะนำคือ เพลงและผู้ใช้ ซึ่งไม่เพียงพอต่อการสร้างระบบแนะนำให้มีประสิทธิภาพ แต่ถ้านำข้อมูลบริบทของผู้ใช้ (Contextual information) มาร่วมในการพิจารณาเช่น แนวเพลงที่ผู้ใช้ชอบฟัง รวมไปถึงช่วงเวลาที่ต่างกันผู้ใช้ก็อาจจะชอบฟังแนวเพลงที่ไม่เหมือนกัน ก็จะสามารถพิจารณาแนะนำเพลงได้ตรงกับความต้องการของผู้ใช้มากขึ้น

งานวิจัย [4] ได้ศึกษาและวิเคราะห์พฤติกรรมของผู้ฟังเพลงซึ่งพบว่าบริบททางด้านเวลาของการฟังเพลงมีความแตกต่างจากบริบทอื่นๆ เพราะบางแนวเพลงมีการฟังเฉพาะบางช่วงเวลาของวัน เช่นเพลงแนวแดนซ์ป๊อป, บัลลาดและร็อคถูกฟังมากในตอนกลางวันและฟังน้อยลงในตอนเย็น ส่วนงานวิจัย [5,6,7] มีการนำเสนอระบบแนะนำเพลงโดยพิจารณามิติทางด้านเวลา โดยทำการแบ่งข้อมูลของผู้ใช้ตามช่วงเวลา ที่เรียกว่า “Micro Profiles” เช่น ช่วงเวลาของวัน วันหยุดหรือวันทำงาน เป็นต้น และใช้ข้อมูล Micro Profilesร่วมกับเทคนิคการกรองร่วม ซึ่งผลการวิจัยพบว่าการใช้บริบทของเวลาร่วมกับเทคนิคการกรองร่วมจะให้ความถูกต้องในการแนะนำสูงกว่าการใช้การกรองร่วมเพียงอย่างเดียว แต่อย่างไรก็ตามงานวิจัยดังกล่าวก็ยังพบปัญหาที่เกิดจากความเบาบางของข้อมูล และดำเนินการกับข้อมูลขนาดใหญ่ ซึ่งทำให้การแนะนำเพลงมีประสิทธิภาพการทำงานลดลง

งานวิจัย [8]ได้นำเสนอวิธีการสำหรับระบบแนะนำข้อมูลสำหรับการกรองร่วมที่ชื่อว่า “Tendencies Based Algorithm” ซึ่งจะดูแนวโน้มของความชอบผู้ใช้และสินค้า ข้อดีของวิธีการนี้คือ เวลาในการคำนวณที่น้อยลงมากเมื่อเทียบกับระบบแนะนำแบบการกรองร่วมและยังสามารถใช้กับข้อมูลที่มีปริมาณมากได้ อย่างไรก็ตามปัญหาความเบาบางของข้อมูลก็ยังเป็นจุดด้อยวิธีการนี้ และในปัจจุบันยังไม่มีงานวิจัยที่เกี่ยวกับระบบแนะนำเพลงแบบกรองร่วมโดยนำวิธีการ Tendencies Based มาใช้เพื่อทดสอบประสิทธิภาพการแนะนำเพลงโดยพิจารณาร่วมกับบริบทของเวลา

ดังนั้นงานวิจัยนี้จะนำเสนอวิธีการทำนายค่าคะแนนที่มีประสิทธิภาพสำหรับระบบแนะนำเพลงชื่อว่า “Tendencies Based Algorithm with Micro Profiles& Sparsity Problem” ซึ่งจะประยุกต์ใช้วิธีการ Tendencies Based Algorithm ร่วมกับการใช้บริบททางด้านเวลามาแนะนำเพลง รวมถึงการแก้ไขปัญหาความเบาบางของข้อมูลโดยการเติมค่าข้อมูลก่อนการทำนายซึ่งวิธีการที่นำเสนอนี้จะช่วยลดเวลาที่ใช้ในการคำนวณสำหรับระบบแนะนำที่ใช้วิธีการกรองร่วมแบบเดิม และเพิ่มความถูกต้องให้กับระบบแนะนำเพลงตามช่วงเวลาให้กับผู้ใช้

2. ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ระบบแนะนำ (Recommender System)

เทคนิคที่นิยมในการพัฒนาระบบแนะนำคือวิธีการกรองร่วม ซึ่งแบ่งออกเป็น 2 ลักษณะ กล่าวคือพิจารณาจากตัวผู้ใช้เป็นสำคัญ (User based) และพิจารณาจากไอเท็มเป็นสำคัญ (Item based) [9]แต่ในงานวิจัยนี้เน้นวิธีการกรองร่วมที่พิจารณาจากผู้ใช้เป็นสำคัญ ดังนั้นจึงขอกล่าวเฉพาะวิธีการนี้เท่านั้น โดยขั้นตอนวิธีการกรองร่วมที่พิจารณาจากผู้ใช้เป็นสำคัญนั้นเริ่มต้นจากการหาความคล้ายคลึงระหว่างผู้ใช้ u กับผู้ใช้ v โดยวิธีการหาค่าความคล้ายคลึงที่นิยม [3]ได้แก่

วิธีการหาค่าความคล้ายคลึงแบบโคไซน์ (Cosine-based Similarity)

$$sim(u,v) = \frac{\sum_i r_{u,i} r_{v,i}}{\sqrt{\sum_i r_{u,i}^2}\sqrt{\sum_i r_{v,i}^2}} \quad (1)$$

วิธีการหาค่าความคล้ายคลึงแบบโคไซน์แบบปรับแก้ (Adjusted Cosine Similarity)

$$sim(u,v) = \frac{\sum_i (r_{u,i} - \bar{r}_i)(r_{v,i} - \bar{r}_i)}{\sqrt{\sum_i (r_{u,i} - \bar{r}_i)^2}\sqrt{\sum_i (r_{v,i} - \bar{r}_i)^2}} \quad (2)$$

การหาค่าความคล้ายคลึงแบบค่าสัมประสิทธ์สหสัมพันธ์ของเพียร์สัน (Pearson correlation coefficient)

$$sim(u,v) = \frac{\sum_{i \epsilon I_u \cap I_v}(r_{u,i}-\bar{r}_u)(r_{v,i}-\bar{r}_v)}{\sqrt{\sum_{i \epsilon I_u \cap I_v}(r_{u,i}-\bar{r}_u)^2}\sqrt{\sum_{i \epsilon I_u \cap I_v}(r_{v,i}-\bar{r}_v)^2}} \quad (3)$$

และเมื่อหาค่าความคล้ายโดยสมการ(1)-(3) แล้วก็จะเข้าสู่ขั้นตอนการทำนายว่าผู้ใช้แต่ละคนจะมีค่าคะแนนความชอบกับแต่ละไอเท็มเท่าไร โดยใช้สมการ (4)

$$p_{u,i} = \bar{r}_u + \frac{\sum_{v \in N} sim(u,v)\left[r_{v,i} - \bar{r}_v\right]}{\sum_{v \in N}|sim(u,v)|} \quad (4)$$

โดย $r_{u,i}$ คือคะแนนความชอบที่ผู้ใช้ u ให้กับไอเท็ม i

$\bar{r}_u$ คือค่าเฉลี่ยของผู้ใช้ u

$\bar{r}_v$ คือค่าเฉลี่ยของผู้ใช้ v

$i \epsilon I_u \cap I_v$ คือไอเท็มที่ผู้ใช้ทั้งสองให้คะแนนร่วมกัน

$\bar{r}_i$ คือค่าเฉลี่ยของไอเท็มที่ i

N คือจำนวนเพื่อนบ้านของผู้ใช้ u

2.2 Tendencies Based Algorithm [8]

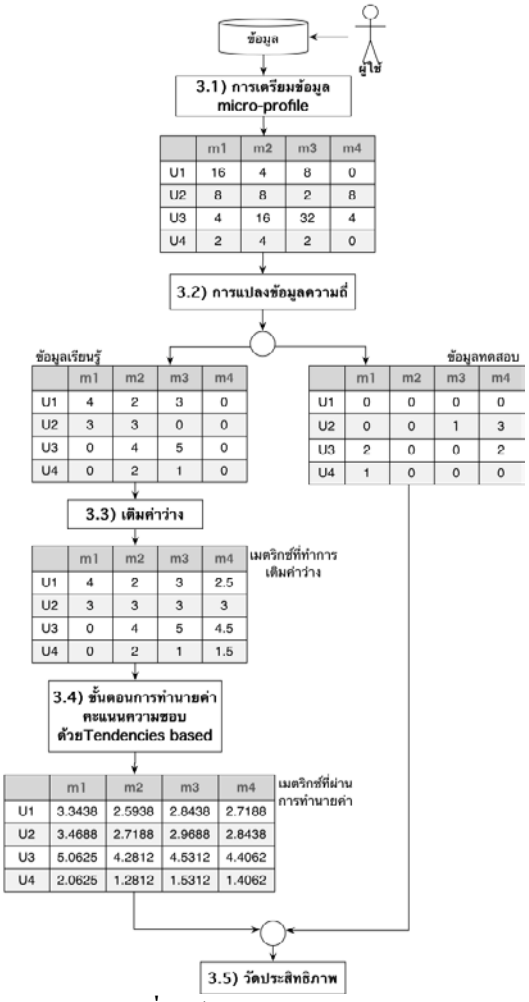
วิธีการนี้เป็นวิธีการกรองร่วมอีกเทคนิคหนึ่งที่ทำนายค่าความชอบของผู้ใช้ที่มีต่อไอเท็มจากค่าแนวโน้มความชอบของผู้ใช้กับไอเท็มนั้น โดยจะพิจารณาจากความแตกต่างระหว่างคะแนนของไอเท็มและค่าเฉลี่ยของไอเท็มหรือค่าเฉลี่ยของผู้ใช้ ซึ่งทำให้ใช้เวลาการคำนวณที่รวดเร็วและทำงานได้ดีกับข้อมูลที่มีความหนาแน่นต่ำ

3. วิธีการที่นำเสนอ

งานวิจัยนี้ได้นำเสนอวิธีการใหม่ที่เรียกว่า *"Tendencies Based Algorithm with Micro Profiles&Sparsity Problem(TBMS)"* ซึ่งมีภาพรวมการทำงานดังภาพที่ 1

3.1 การเตรียมข้อมูล Micro Profiles

ในงานวิจัยนี้ใช้ข้อมูลการฟังเพลงจากฐานข้อมูลของ last.fm[1] [10]ประกอบด้วยประวัติการฟังเพลงของผู้ใช้แต่ละคน ซึ่งจะต้องนำมาแปลงข้อมูลให้อยู่ในรูปแบบของความถี่ในการ



ภาพที่ 1: ขั้นตอนการทำงานของวิธีการที่นำเสนอ

ฟังเพลงแต่ละเพลงของผู้ใช้แต่ละคนจากนั้นนำข้อมูลความถี่ทั้งหมดแบ่งออกเป็น1) ข้อมูลทั้งหมดที่ไม่ได้แบ่งตามช่วงเวลา และ 2) ข้อมูลทั้งหมดที่แบ่งออกเป็น Micro Profiles ตามช่วงเวลา จำนวน 5ชุดด้วยกันคือช่วงเช้า, ช่วงกลางวัน, ช่วงเย็น วันหยุดและวันธรรมดา ซึ่งจะมีข้อมูลที่ใช้ในการทดลองทั้งหมด 6 ชุดดังแสดงในตารางที่ 1

ตารางที่ 1: ข้อมูลที่ใช้ในการทดลอง

ข้อมูล	จำนวนผู้ใช้	จำนวนเพลง
ข้อมูลทั้งหมด	222	183,322
ตอนเช้า (00:00-11:59 น.)	182	130,519
ตอนกลางวัน(12:00-17:59 น.)	184	79,231
ตอนเย็น (18:00-23:59 น.)	187	103,393
วันหยุด (เสาร์-อาทิตย์)	212	98,387
วันธรรมดา(วันจันทร์-วันศุกร์)	190	162,442

[1] http://www.lastfm.com

3.2 การแปลงข้อมูลความถี่

เนื่องจากข้อมูลความถี่ที่ได้มาจากขั้นตอนที่ 3.1 มีความแตกต่างของข้อมูลค่อนข้างมาก คือบางเพลงถูกฟังเพียงครั้งเดียว แต่บางเพลงถูกฟังหลายพันครั้ง ดังนั้นในงานวิจัยนี้จึงได้ทำการแปลงความถี่ของการฟังเพลงออกเป็นคะแนนความนิยม โดยใช้วิธีการ log scale[11] ดังสมการ (5)

$$\widehat{r_{u_i,m_j}} = \log_2(r_{u_i,m_j} + 1) \quad (5)$$

โดยที่ r_{u_i,m_j} คือความถี่ในการฟังเพลงที่ m_j ของผู้ใช้ u_i

3.3 การเติมค่าว่าง

เนื่องจากผู้ใช้ไม่ได้ฟังเพลงทุกเพลงที่มีในฐานข้อมูล ดังนั้นจึงทำให้เพลงที่ผู้ใช้คนนั้นไม่ได้ฟัง จึงไม่มีข้อมูลหรือเกิดค่าว่างเกิดขึ้น ซึ่งทำให้ข้อมูลมีความหนาแน่นน้อยและส่งผลต่อความถูกต้องของระบบแนะนำเพลง และเกิดปัญหาที่เรียกว่า *ปัญหาความเบาบางของข้อมูล* เราจะใช้วิธีการเติมว่างด้วยการเติมคะแนนความชอบเฉลี่ยของแนวเพลงนั้น

ตารางที่ 2: ตัวอย่างเมทริกซ์ที่เก็บค่าคะแนนความชอบที่ผู้ใช้แต่ละคนฟังเพลง โดยที่ ? คือเพลงที่ผู้ใช้ไม่เคยฟัง

	เพลงที่ 1 jazz	เพลงที่ 2 pop	เพลงที่ 3 pop	เพลงที่ 4 pop
ผู้ใช้ u_1	4	2	3	?
ผู้ใช้ u_2	3	3	?	?
ผู้ใช้ u_3	?	4	5	?
ผู้ใช้ u_4	?	2	1	?

ตัวอย่างในตารางที่ 2 แสดงข้อมูลค่าคะแนนความชอบที่ผู้ใช้แต่ละคนชอบฟังเพลงโดยที่ ? คือเพลงที่ผู้ใช้ไม่เคยฟังถ้าต้องการเติมค่าว่าง (?)ของผู้ใช้ u_1ในเพลงที่ 4 ซึ่งเป็นเพลงแนวป๊อบ วิธีการการคำนวณค่าคะแนนความชอบเฉลี่ยจะนำค่าคะแนนความชอบเพลงที่ 2 และ เพลงที่ 3 ซึ่งเป็นเพลงแนวป๊อบเช่นเดียวกันกับเพลงที่ 4 มาคำนวณจะได้ (2+3)/2 = 2.5 ดังแสดงในตารางที่ 3

ตารางที่ 3:ตัวอย่างเมทริกซ์ที่เก็บค่าคะแนนความชอบที่มีการเติมค่าว่าง

	เพลงที่ 1 jazz	เพลงที่ 2 pop	เพลงที่ 3 pop	เพลงที่ 4 pop
ผู้ใช้ u_1	4	2	3	2.5
ผู้ใช้ u_2	3	3	3	3
ผู้ใช้ u_3	?	4	5	4.5
ผู้ใช้ u_4	?	2	1	1.5

3.4 ขั้นตอนการทำนายค่าคะแนนความชอบด้วยวิธีการ Tendencies Based Algorithm

ขั้นตอนนี้เป็นการคำนวณค่าคะแนนความชอบของผู้ใช้แต่ละคน โดยจะพิจารณาจากแนวโน้มในการชอบฟังเพลงของผู้ใช้ ซึ่งประยุกต์ใช้วิธีการ Tendencies Based Algorithm มาใช้ในการทำนาย โดยกำหนดค่าตัวแปรดังตารางที่ 4

ตารางที่ 4: นิยามค่าตัวแปร

สัญลักษณ์	ความหมาย
$U = \{u_1, u_2, \dots, u_i, \dots u_m\}$	เซตของผู้ใช้
$M = \{m_1, m_2, \dots, m_j, \dots m_n\}$	เซตของเพลง
V	เมทริกซ์ของค่าคะแนนความชอบขนาด $m \times n$
v_{u_i,m_j}	คะแนนความชอบของเพลง m_j ที่ผู้ใช้ u_i ให้ไว้
$V_{u_i,\cdot} = \{v_{u_i,m_j} \in V \mid m_j \in M\}$	เซตของค่าคะแนนความชอบของเพลงที่ผู้ใช้ u_i
$V_{\cdot,m_j} = \{v_{u_i,m_j} \in V \mid u_i \in U\}$	เซตค่าคะแนนความชอบที่ผู้ใช้ให้กับเพลง m_j
$\bar{V}_{u_i,\cdot}$	ค่าเฉลี่ยของผู้ใช้ u_i
$\bar{V}_{\cdot,m_j}$	ค่าเฉลี่ยของเพลง m_j

ขั้นตอนวิธีการการทำนายค่าคะแนนความชอบดังนี้

ขั้นที่ 1 หาค่าแนวโน้มความชอบของผู้ใช้ ub_{u_i}และเพลง ib_{m_j}ตามสมการ (6)และสมการ (7)

$$ub_{u_i} = \frac{\sum_{m_j \in M_u} \left(v_{u_i,m_j} - \bar{v}_{\cdot,m_j}\right)}{|M_u|} \quad (6)$$

$$ib_{m_j} = \frac{\sum_{u_i \in U_i} \left(v_{u_i,m_j} - \bar{v}_{u_i,\cdot}\right)}{|U_i|} \quad (7)$$

โดยที่M_u แทนจำนวนเพลงที่ผู้ใช้ u_i ฟัง

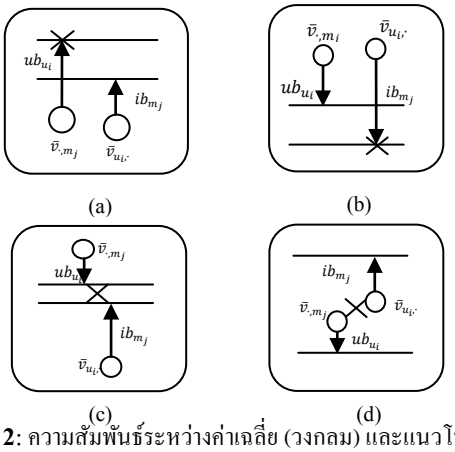
U_i แทนจำนวนผู้ใช้ที่ฟังเพลง m_j

ขั้นที่ 2ทำนายค่าคะแนนความพึงพอใจของผู้ใช้ u_i กับเพลง m_j จะแบ่งการพิจารณาออกเป็น 4 กรณีตามหลักการของ Tendencies Based ดังภาพที่ 2 โดยแบ่งการทำนายออกเป็น 4 กรณีดังนี้

กรณีที่ 1ภาพที่ 2(a) ทำนายจากแนวโน้มความชอบที่เป็นบวกทั้งผู้ใช้และเพลงหมายความว่าผู้ใช้มีแนวโน้มที่ให้คะแนน

เพลงที่ฟังมีค่าสูงในหลายเพลง โดยถ้าค่าแนวโน้มของผู้ใช้มากกว่าค่าเฉลี่ยของเพลงและค่าแนวโน้มของเพลงมากกว่าค่าเฉลี่ยของผู้ใช้จะใช้วิธีการทำนายค่าคะแนนดังสมการ (8)

$$p_{u_i,m_j} = \max(\bar{v}_{u_i,\cdot} + ib_{m_j}, \bar{v}_{\cdot,m_j} + ub_{u_i}) \quad (8)$$



ภาพที่ 2: ความสัมพันธ์ระหว่างค่าเฉลี่ย (วงกลม) และแนวโน้มของผู้ใช้ (ลูกศร)

<u>กรณีที่ 2</u>ภาพที่ 2(b) ทำนายจากแนวโน้มความชอบที่เป็นลบทั้งของผู้ใช้และเพลงดังนั้นถ้าค่าแนวโน้มของผู้ใช้น้อยกว่าค่าเฉลี่ยของเพลงและค่าแนวโน้มของเพลงน้อยกว่าค่าเฉลี่ยของผู้ใช้ จะใช้วิธีการทำนายค่าคะแนนดังสมการ(9)

$$p_{u_i,m_j} = \min(\bar{v}_{u_i,\cdot} + ib_{m_j}, \bar{v}_{\cdot,m_j} + ub_{u_i}) \quad (9)$$

<u>กรณีที่ 3</u>ภาพที่ 2(c) ทำนายจากแนวโน้มความชอบของเพลงที่เป็นบวกและผู้ใช้ที่มีแนวโน้มความชอบเป็นลบ โดยที่ถ้าค่าแนวโน้มของผู้ใช้น้อยกว่าค่าเฉลี่ยของเพลงและค่าแนวโน้มของเพลงมากกว่าค่าเฉลี่ยของผู้ใช้และค่าเฉลี่ยของเพลงมากกว่าค่าเฉลี่ยของผู้ใช้จะใช้วิธีการทำนายค่าคะแนนดังสมการ(10)

$$p_{u_i,m_j} = \min(max(\bar{v}_{u_i,\cdot}, (\bar{v}_{\cdot,m_j} + ub_{u_i}) \propto + \left(\bar{v}_{u_i,\cdot} + ib_{m_j}\right)(1-\propto)), \bar{v}_{\cdot,m_j}) \quad (10)$$

โดยที่ $\propto$ แสดงถึงค่าความเชื่อมั่นในการทำนายของค่าเฉลี่ยของผู้ใช้และเพลง

<u>กรณีที่ 4</u> ภาพที่ 2(d)มีแนวโน้มความชอบที่แตกต่างกันอย่างชัดเจนซึ่งแนวโน้มความชอบของทั้งผู้ใช้และสินค้ามีแนวโน้มที่ตรงกันข้ามกัน โดยที่ถ้าค่าแนวโน้มของเพลงมากกว่าค่าเฉลี่ยของผู้ใช้และค่าแนวโน้มของผู้ใช้น้อยกว่าค่าเฉลี่ยของเพลงและค่าเฉลี่ยของผู้ใช้มากกว่าค่าเฉลี่ยของเพลงจะใช้วิธีการทำนายค่าคะแนนดังสมการ(11)

$$p_{u_i,m_j} = \bar{v}_{\cdot,m_j} \propto + \bar{v}_{u_i,\cdot}(1-\propto) \quad (11)$$

3.5 การวัดประสิทธิภาพในการทำนาย

การวัดประสิทธิภาพของขั้นตอนวิธีการทำนายค่าคะแนนงานวิจัยนี้ใช้ตัววัดความแม่นยำทางสถิติที่เรียกว่าMean Absolute Error (MAE)[12]ซึ่งเป็นการวัดค่าคะแนนความชอบที่ทำนายได้จากระบบมีความแตกต่างจากค่าคะแนนที่ผู้ใช้เคยให้ไว้เท่าไร โดยสูตรการคำนวณ MAE เป็นไปตามสมการที่ (12) และ (13)

$$MAE_{u_i} = \frac{\sum_{j=1}^{n} |v_{u_i,m_j} - pv_{u_i,m_j}|}{n} \quad (12)$$

$$MAE = \frac{\sum_{i=1}^{N} MAE_{u_i}}{N} \quad (13)$$

โดยที่v_{u_i,m_j}คือคะแนนความชอบของเพลง m_j ที่ผู้ใช้ u_i ให้ไว้

pv_{u_i,m_j} คือคะแนนความชอบของเพลง m_j ที่ผู้ใช้ u_iที่ได้จากการทำนาย

n คือ จำนวนเพลงที่ผู้ใช้ให้คะแนนไว้

N คือ จำนวนผู้ใช้

4. ผลการทดลอง

การวัดประสิทธิภาพในงานวิจัยนี้เป็นการเปรียบเทียบประสิทธิภาพระหว่างวิธีการที่นำเสนอ TBMS กับวิธีการกรองร่วมแบบดั้งเดิมที่พิจารณาผู้ใช้เป็นหลัก โดยใช้ตัววัดความคล้ายคลึงหลายตัววัดแบบต่างๆ โดยผลการทดลองเปรียบเทียบประสิทธิภาพมีรายละเอียดดังภาพที่ 3 ซึ่งพบว่าวิธีการที่นำเสนอ TBMS ให้ค่า MAE เฉลี่ยจากทุกชุด Micro Profiles คือ 0.5638 และวิธีการกรองร่วมที่พิจารณาผู้ใช้เป็นหลักในแต่ละความคล้ายคลึงต่างๆ และพิจารณาจำนวนเพื่อนบ้านเท่ากับ 10% ของจำนวนผู้ใช้ทั้งหมด พบว่าวิธีการกรองร่วมที่ใช้การวัดความคล้ายคลึงแบบโคไซน์ให้ค่า MAE คือ0.6201 ใช้การวัดความคล้ายคลึงแบบสัมประสิทธ์สหสัมพันธ์ของเพียร์สันให้ค่า MAE คือ 0.6292 และใช้การวัดความคล้ายคลึงแบบโคไซน์ปรับแก้ให้ค่า MAE คือ0.6166 ซึ่งสามารถสรุปได้ว่าวิธีการที่นำเสนอ TBMS มีประสิทธิภาพมากกว่าวิธีการกรองร่วมที่พิจารณาจากผู้ใช้เป็นหลัก

นอกจากการวัดความประสิทธิภาพของความแม่นยำในการทำนายแล้ว งานวิจัยนี้ยังวัดประสิทธิภาพในเชิงเวลาที่ใช้ในการ

ประมวลผลของแต่ละวิธีการอีกด้วย โดยทดลองบนเครื่องคอมพิวเตอร์ที่ใช้ซีพียูความเร็ว3.20 GHz มีหน่วยความจำหลัก 8 GB ฮาร์ดดิสก์ 2 TBและภาษาโปรแกรมไพธอน (Python) ในการพัฒนาโปรแกรมดังแสดงผลในตารางที่ 5 ถึง 9

ภาพที่ 3:การเปรียบเทียบค่าMAEระหว่างวิธีการที่นำเสนอ TBMSและวิธีการกรองร่วมที่พิจารณาจากผู้ใช้เป็นหลัก

ตารางที่ 5: เวลาที่ใช้ในการประมวลผลสำหรับสร้าง Similarity Matrix

วิธีการ	เวลาที่ใช้ประมวลผล (วินาที)
Cosine Similarity (สมการที่ 1)	8326
Adjust Cosine Similarity (สมการที่ 2)	6027
Pearson correlation coefficient (สมการที่ 3)	6133
TBMS (ขั้นตอน 3.1+3.2+3.3)	171

ตารางที่ 6: เวลาที่ใช้ในการประมวลผลสำหรับขั้นตอนการทำนาย

วิธีการ	การทำนาย (วินาที)
Cosine Similarity (สมการที่ 4)	541
Adjust Cosine Similarity (สมการที่ 4)	587
Pearson correlation coefficient (สมการที่ 4)	583
TBMS (ขั้นตอน 3.4)	**42**

จากตารางที่ 5 วิธีการกรองร่วมแบบดั้งเดิมที่ใช้วิธีการสร้าง Similarity Matrix แบบต่างๆจะใช้เวลาในการประมวลผลสูงมากกว่าวิธีการ TBMS ที่นำเสนอในงานวิจัยนี้หลายเท่า สอดคล้องกับผลการทดลองในตารางที่ 6 ที่แสดงเวลาที่ใช้ในการทำนาย พบว่าวิธีการ TBMS มีเวลาในการประมวลผลที่เร็วกว่าทุกวิธีการเช่นกัน และตารางที่ 7 แสดงผลการทดลองของเวลาที่ใช้ในการทำนายของวิธีการ TBMS ซึ่งจากผลลัพธ์ที่แสดงในตารางพบว่าเวลาในการทำนายจะลดลงเมื่อแบ่งกลุ่มของข้อมูลในการฟังเพลงของผู้ใช้ออกเป็น Micro Profiles เสียก่อน และผลจากแบ่งข้อมูลเป็น Micro Profiles นี้ทำให้มีค่า MAE เพียง 0.5638 และต่ำกว่าทั้งสามวิธีการที่นำมาเปรียบเทียบ (ภาพที่ 3)

ตารางที่ 7: เวลาในการทำนายของวิธีการTBMS(ขั้นตอนที่ 3.4)โดยใช้ข้อมูลนำเข้าที่แบ่งเป็น Micro Profiles

วิธีการ TBMS		
ข้อมูลที่ใช้ในการทดลอง		**เวลา (วินาที)**
ไม่แบ่ง Micro Profiles	**ข้อมูลทั้งหมด**	**42**
แบ่ง Micro Profiles	ตอนเช้า	25
	ตอนกลางวัน	15
	ตอนเย็น	20
	วันหยุด	21
	วันธรรมดา	32

ตารางที่ 8: ประสิทธิภาพเชิงเวลาสำหรับสร้าง Similarity Matrix เปรียบเทียบกับวิธีการ TBMSโดยที่ m คือจำนวนผู้ใช้และ n คือจำนวนเพลง

วิธีการ	ประสิทธิภาพ
Cosine Similarity(สมการที่ 1)	$O(m^2n)$
Adjust Cosine Similarity(สมการที่ 2)	$O(m^2n)$
Pearson Similarity (สมการที่ 3)	$O(m^2n)$
TBMS (ขั้นตอน 3.1+3.2+3.3)	O(mn)

ตารางที่ 9: ประสิทธิภาพเชิงเวลาสำหรับการทำนายโดยที่ m คือจำนวนผู้ใช้และ n คือจำนวนเพลง

วิธีการ	การทำนาย
Cosine Similarity (สมการที่ 4)	O(mn)
Adjust Cosine Similarity(สมการที่ 4)	O(mn)
Pearson Similarity(สมการที่ 4)	O(mn)
TBMS (ขั้นตอน 3.4)	O(1)

จากตารางที่ 8 พบว่าประสิทธิภาพเชิงเวลาสำหรับสร้าง Similarity Matrix แบบต่างๆ ตามสมการที่ (1), (2) และ (3) เป็น $O(m^2n)$ ส่วนวิธีการที่นำเสนอ TBMS มีประสิทธิภาพเชิงเวลาในขั้นตอนที่ 3.1 เป็น O(mn) ขั้นตอนที่ 3.2เป็น O(mn) และ

ขั้นตอนที่ 3.3เป็น O(mn) ดังนั้นประสิทธิภาพโดยรวมทั้ง 3 ขั้นตอนจึงเป็น O(mn) ซึ่งมีประสิทธิภาพดีกว่าทั้งสามวิธีการที่ใช้เปรียบเทียบ และจากตารางที่ 9 แสดงประสิทธิภาพเชิงเวลาของการทำนายพบว่า วิธีการที่เสนอ TBMS ซึ่งประยุกต์ใช้หลักการของวิธีการ “Tendencies Based Algorithm” ในการทำนาย โดยในงานวิจัย [8] ได้ทำการทดลอง และหาค่าประสิทธิภาพในการทำนายของวิธีการ “Tendencies Based Algorithm” เป็น O(1) ดังนั้นวิธีการ TBMS ที่นำเสนอนี้ จึงมีประสิทธิภาพเชิงเวลาของการทำนายเป็น O(1) เช่นเดียวกัน ซึ่งเป็นประสิทธิภาพที่ดีกว่าทั้งสามวิธีการที่เปรียบเทียบอย่างมาก

5. สรุปผลการทดลอง

งานวิจัยนี้นำเสนอขั้นตอนวิธีสำหรับระบบแนะนำข้อมูลเพลงชื่อว่า *“Tendencies Based Algorithm with Micro Profiles & Sparsity Problem (TBMS)”* ซึ่งประยุกต์ใช้วิธีการของ Tendencies Based Algorithm มาใช้ และปรับปรุงขั้นตอนการทำนายค่าคะแนนความชอบในระบบแนะนำเพลงให้มีความแม่นยำในการทำนายมากขึ้น โดยการใช้ Micro Profiles อีกทั้งยังแก้ปัญหาที่เป็นค่าคะแนนที่ว่าง โดยการเติมค่าว่างก่อนทำนาย ซึ่งจากผลการทดลองแสดงให้เห็นว่าวิธีการ TBMS นี้ให้ผลการทำนายความชอบที่แม่นยำกว่าวิธีการกรองร่วมแบบดั้งเดิม นอกจากนี้ประสิทธิภาพในเชิงเวลาของวิธีการ TBMS ก็ยังมีความรวดเร็วมากกว่าวิธีการกรองร่วมแบบดั้งเดิมอีกด้วย

แต่อย่างไรก็ตามงานวิจัยนี้ยังมีแนวทางในการปรับปรุงให้วิธีการทำนายค่าคะแนนให้มีความแม่นยำมากยิ่งขึ้น โดยการปรับปรุงขั้นตอนวิธีในการเติมค่าคะแนนที่ว่างที่พิจารณาจากปัจจัยอื่นที่เกี่ยวข้อง นอกเหนือจากพิจารณาเพียงแค่แนวเพลงเพียงอย่างเดียว รวมทั้งการแบ่งช่วงเวลา Micro Profiles แบบช่วงเวลาของผู้ใช้เฉพาะบุคคลมากขึ้น

เอกสารอ้างอิง

[1] K.Marius and F. Ricci, "Contextual music information retrieval and recommendation:state of the art and challenges,"*Computer Science Review*, Vol.6, pp. 89–119, 2012.

[2] J.Bobadilla, F.Ortega, A.Hernando, A.Gutierrez, "Recommender systems survey" ,*Knowledge-Based Systems*, Vol.46, pp.109-132, July, 2013.

[3] C.Laurent, J.Kris, F.Francoise, M.Frank, *State-of-the-Art Recommender Systems* , IGI Global, 2009.

[4] C. H. Park and M. Kahng, "Temporal Dynamics in Music Listening Behavior:A Case Study of Online Music Service ",*The 9th International Conference on Computer and Information Science (ACIS 2010),* pp.573-578, 18-20 August,2010.

[5] B. Linas and A. Xavier, "Towards time-dependant recommendation based on implicit feedback," *CARS-2009*, New York, USA, October 25, 2009.

[6] S. Antonietta, A. Albert, D. Helenca, E. Gregorio, and V. Paulo, "My Personal Media Entertainer: Context-Adaptive Content Recommendation and Delivery ", *ACM Workshop on Social, Adaptive and Personalized Multimedia Interaction and Access*, pp.33-36, 2010.

[7] T. Cebrián, M. Planagumà, P. Villegas, and X. Amatriain, "Music Recommendations with Temporal Context Awareness," *Recommender Systems (RecSys2010)*, pp.349-352, 2010.

[8] C. Fidel, C. Victor, F. Diego and F. Vreixo, "Comparison of Collaborative Filtering Algorithms." *ACM Transaction on the Web*, Vol. 5, No.1, Febuary, 2011.

[9] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," *Proceedings of the 10th international conference on World Wide Web(www2001),* Hong Kong, May 1-5, 2001.

[10] C.Oscar. Last fm Dataset. [online]. http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/lastfm-1K.html (วันที่ค้นข้อมูล : 1 กันยายน 2556)

[11] B. Xu, J. Bu, C. Chen, and D. Cai, "An Exploration of Improving Collaborative Recommender Systems via User-Item Subgroups "*WWW2012*, Lyon, France, Apr 16-20, 2012.

[12] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *CLIMATE RESEARCH*, Vol. 30, pp.79-82, Dec 19, 2005.

2014 11th International Joint Conference on Computer Science and Software Engineering (JCSSE)

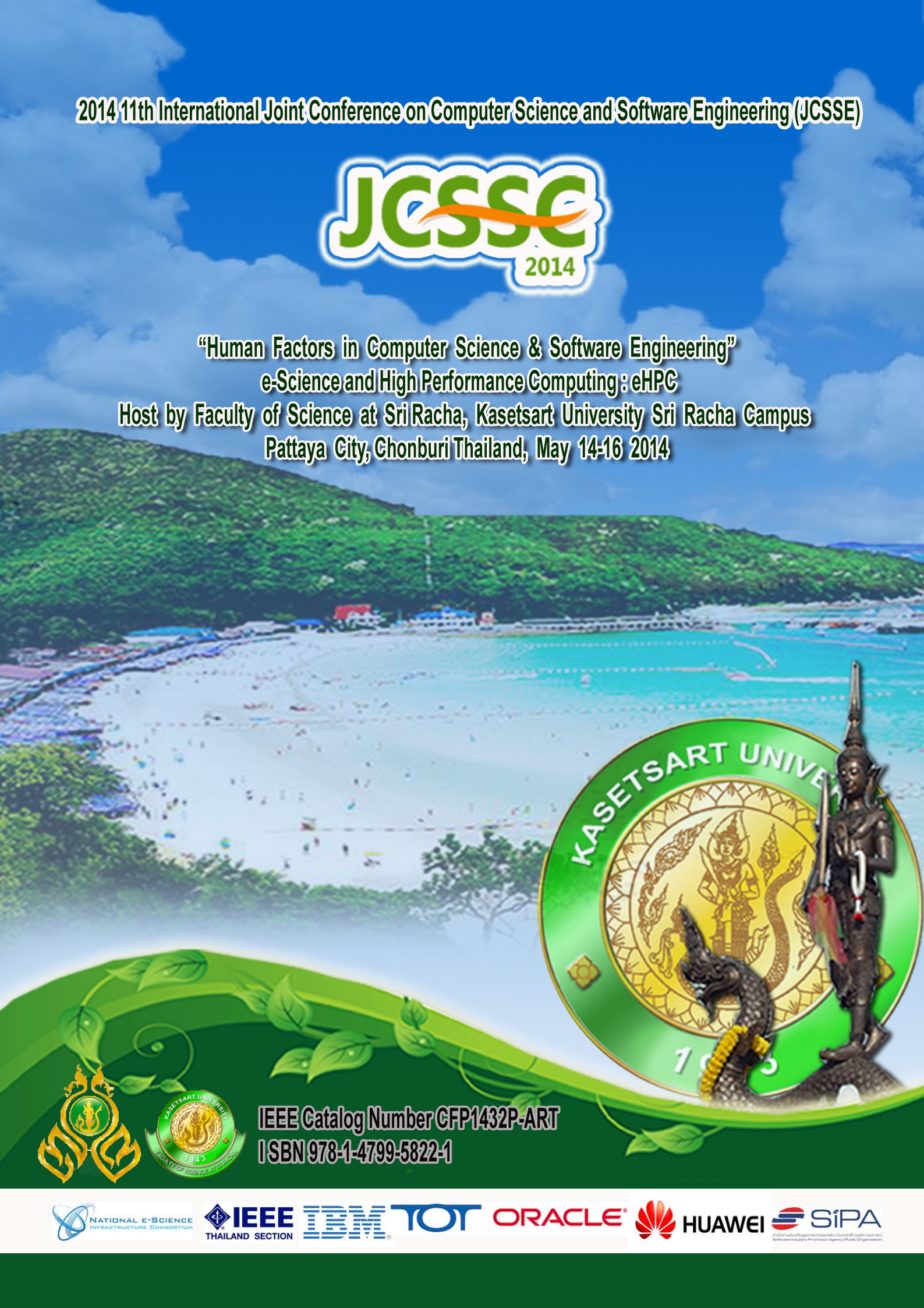
JCSSE 2014

"Human Factors in Computer Science & Software Engineering"

e-Science and High Performance Computing : eHPC

Host by Faculty of Science at Sri Racha, Kasetsart University Sri Racha Campus

Pattaya City, Chonburi Thailand, May 14-16 2014



IEEE Catalog Number CFP1432P-ART

ISBN 978-1-4799-5822-1

National e-Science Infrastructure Consortium | IEEE Thailand Section | IBM | TOT | ORACLE | HUAWEI | SIPA

IT and Applications

Machine Learning and Knowledge Management

Ontology and Knowledge Management

Comparison of the Constant Prediction Time of Collaborative Filtering Algorithms by Using Time Contexts

Sumet Darapisut and Jakkarin Suksawatchon
Faculty of Informatics, Burapha University
Chonburi, 20131 Thailand
dearsumet@gmail.com, jakkaman@gmail.com

***Abstract*—This research presents the comparison of collaborative filtering techniques which are Tendencies Based Algorithm, Item mean algorithm, and Simple mean based algorithm. All these algorithms use the constant time in prediction process. To evaluate our proposed model, we use last.fm dataset including music listening history of each user. Each user's profile is split into several sub-profiles based on specified time ranges called "Time Contexts". Thus the prediction is done using these Time Contexts instead of a single user profile. From our experiments, we have found that Tendencies Based Algorithm with Time Contexts is effective. It is given more accuracy and much more efficient computationally than tradition collaborative filtering algorithms.**

Keywords— music recommender system; collaborative filtering; time contexts

I. INTRODUCTION

Nowadays, the amount of digital music is rapidly increasing in each year. These affect to users to search digital music that they want with difficulty, which is why the music recommender system is developed to help and discover music in information overloading condition.

The popularity of music recommender approaches can be categorized into three categories [1,2,3]. The first one is content-based that uses the property of music such as tonality, key, or tempo of songs and profile of the user's preference for analysis and recommend the type of music this user likes. This means that this algorithm tries to recommend music that is similar to those that a user liked in the past. Second, collaborative filtering is the most successful and popular approach. This approach is based on considering and analyzing the information on users' behaviors or user's preference, and then predicts what the user will like based on his/her similarity to other users. In spite of collaborative filtering technique is a famous technique, but it often suffers from many problems: sparsity, cold start, and scalability. Third, hybrid approach is combining content-based filtering and collaborative filtering to overcome the problems of two methods. Music recommendation systems exploit these three approaches such as Pandora Radio, Songza, Last.fm, iTunes Genius and Songbird 2.0.

Some music recommender systems often recommend in a few popular music in such that users do not prefer or cannot always meet the users' needs. Because such applications do not consider the users' current situations and users may have different preferences in different contexts. For example, users listen to one type of music while working and another type of music before going sleep [5]. There are some researchers studied context-awareness recommendation systems. For example, C. H. Park and M. Kahng [4] presented behavior's listener that founded time dimension in music listening is different other contextual dimensions. Because some genres of music are preferred at specific time of day such as ballad, dance, pop, and rock genres has a peak at noon and decrease rapidly in the evening. The ballad song has been is listened more than pop dance genre in winter but, in summer pop dance genres listened more than ballad song. The researchers in [5][6][7] presented time-aware music recommender system by split user's profile into sub-profiles (micro-profiling) that consider timing context with collaborative filtering approach. The results showed that this approach is better than traditional recommendation approaches. However, this approach still based on collaborative filtering technique, so it also suffers from sparsity and scalability problems that will decrease the performance. In addition, collaborative filtering technique spends time very much in training and prediction processes.

F. Vreixo and et al. [8] presented a series of collaborative filtering algorithm: Tendencies Based Algorithm; Item mean algorithm, Simple mean based algorithm. Especially, Tendencies Based algorithm is easy to compute, and can be accurately calculated using much fewer data with very low computational time. This algorithm captures the tendency of users and items. However, all these algorithms use only users' profile to find tendencies of users and items; it does not use other contexts to find tendencies of users and items.

In recent years, numerous algorithms based on different ideas and concepts have been developed to address this problem. Unfortunately, works that compare these techniques are scarce, making it difficult to select the best algorithm in a

978-1-4799-5822-1/14/$31.00 ©2014 IEEE

given situation. In this work we compare different techniques of collaborative filtering especially the constant prediction time of such algorithm: Tendencies Based Algorithm; Item mean algorithm, Simple mean based algorithm [8]; to see the efficiency of each algorithm. Besides, this work adapts a contextual called "Time Contexts" which split each user's profile into several sub-profiles based on time range. This means that each single user has several sub-profiles representing users in the particular context. This idea will improve the prediction accuracy dealing with sparsity problem. After we compare the performance among those algorithms, we have found that Tendencies Based algorithm gives the best results. This means that we will combine Tendencies Based algorithm and Time Contexts which can improve accurately predict user's preference, as well.

The outline of the rest of this paper is as follows. In Section II, the background knowledge is introduced, and some notation is defined. In Section III, our experiments are proposed. Section IV illustrates the performance study on real data sets. Section V concludes the paper.

II. BACKGROUND

A. Notation

First, we introduce all notations used in this paper for further understand as shown in Table I.

TABLE I. DEFINITION OF ITEM MEAN ALGORITHM, SIMPLE MEAN ALGORITHM AND TBT ALGORITHM

Notation	Description
$U=\{u_1,u_2,\ldots,u_i,\ldots,u_m\}$	Set of users
$M=\{m_1,m_2,\ldots,m_j,\ldots m_n\}$	Set of music (songs)
R	Matrix of rating $m \times n$
r_{u_v,m_j}	Rating of music (song) m_j listened by user u_v
r_{u_i,m_j}	Rating of music (song) $m_j$listened by user u_i
r_{u_i,m_k}	Rating of music (song) m_k listened by user u_i
$r_{u_i,\bullet}=\{r_{u_i,m_j}\in \mathrm{R}\,\vert\, m_j\in M\}$	Set of rating of music (song) $m_j$listened by user u_i
$r_{\bullet,\mathrm{m}_j}=\{r_{u_i,m_j}\in \mathrm{R}\,\vert\, u_i\in U\}$	Set of rating of music (song) listened by users
$\overline{r_{u_i,\bullet}}$	Mean of user u_i
$\overline{r_{u_v,\bullet}}$	Mean of user u_v
$\overline{r_{\bullet,\mathrm{m}_j}}$	Mean of music (song) m_j

B. Recommender System

The most common approach of a recommender system is collaborative filtering (CF) technique. This algorithm will recommend item to target user depended on collecting and analyzing a large amount of information on user's behaviors or preference and predicting what the user will like based on their similarity to other users. The collaborative filtering is divided two types: user based and item based. The basic technique in collaborative filtering starts with similarity measures between users or items [9]. In user based the popular similarity measurements are cosine-based similarity, Pearson correlation-based similarity, and adjusted cosine-based similarity as shown in Eq. (1), Eq. (2), and Eq. (3).

Cosine-based similarity:

$$sim(u_i,u_v)=\frac{\sum_j(r_{u_i,m_j},r_{u_v,m_j})}{\sqrt{\sum_j(r_{u_i,m_j})^2}\sqrt{\sum_j(r_{u_v,m_j})^2}} \quad (1)$$

Pearson correlation-based similarity:

$$sim(u_i,u_v)=\frac{\sum_{j\in M_{u\cap}M_v}\left(r_{u_i,m_j}-\bar{r}_{u_i,\cdot}\right)\left(r_{u_v,m_j}-\bar{r}_{u_v,\cdot}\right)}{\sqrt{\sum_{j\in M_{u\cap}M_v}\left(r_{u_i,m_j}-\bar{r}_{u_i,\cdot}\right)^2}\sqrt{\sum_{j\in M_{u\cap}M_v}\left(r_{u_v,m_j}-\bar{r}_{u_v,\cdot}\right)^2}} \quad (2)$$

Adjust cosine-based similarity:

$$sim(u_i,u_v)=\frac{\sum_j\left(r_{u_i,m_j}-\bar{r}_{\cdot,m_j}\right)\left(r_{u_v,m_j}-\bar{r}_{\cdot,m_j}\right)}{\sqrt{\sum_j\left(r_{u_i,m_j}-\bar{r}_{\cdot,m_j}\right)^2}\sqrt{\sum_j\left(r_{u_v,m_j}-\bar{r}_{\cdot,m_j}\right)^2}} \quad (3)$$

Next, we select the number of users who are the most similar to the target user to be the best neighbors. The popular techniques are best k-nearest neighbors (k-nn). After that, we calculate the user's preference score for each item based on the best neighbors' preferences by using the following equation.

$$p_{u_i,m_j}=\bar{r}_{u_i,\cdot}+\frac{\sum_{u_v\in U}sim(u_i,u_v)\left[r_{u_v,m_j}-\bar{r}_{u_v,\cdot}\right]}{\sum_{u_v\in U}|sim(u_i,u_v)|} \quad (4)$$

III. EFFICIENT COLLABORATIVE FILTERING ALGORITHMS

The overall steps of this research is shown in Fig. 1.

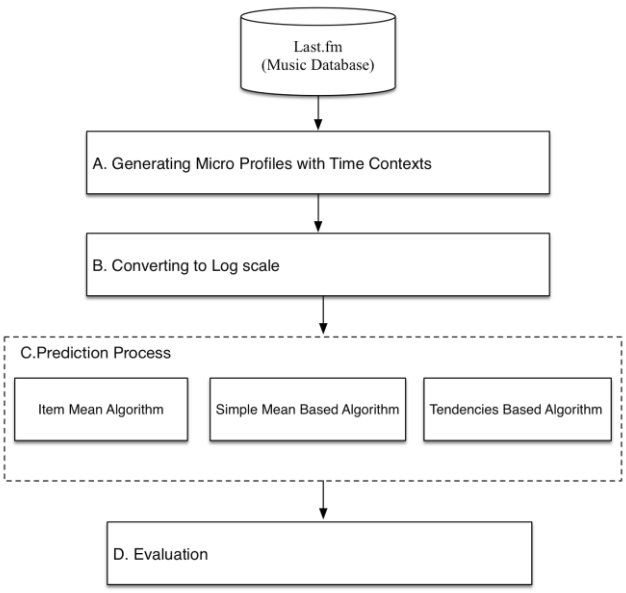


Fig. 1. Our experiment design steps

A. Generating Time Contexts

In this work, we evaluate each method with Last.fm database [10], which is consisted of the history of music listening of individual users together with a time stamp. After that, we must convert to those data in the form of frequency of music listening for each user in a given time. Our proposed method represents a single user by many sub-profiles, again called "Time Contexts". Thus, this research will test the performance with two datasets. First dataset is all data which are not segmented into the specified time range. The another one is Time Contexts that all data of each user are split into pre-define five time segmentation called Morning (00:00-11:59 AM), Afternoon (12:00-17:59 PM), Evening-Night (18:00-23:59 PM), Weekend (Saturday, Sunday), and Weekday (Monday-Friday).

TABLE II. TIME CONTEXTS USED IN OUR RESEARCH.

Data	The number of users	The number of music
Morning	182	130,519
Afternoon	184	79,231
Evening-Night	187	103,393
Weekend	212	98,387
Weekday	190	162,442

B. Converting to Log scale

Since the frequency of music listening of each user obtained the previous step may be different scales. For example, some songs were listened only a few users but some songs were very popular listened many times. So, we convert the number of times the user listened to music as an approximation of rating score by using log scale [11] as shown in Eq. (5). We will call an approximation of rating score in short as rating.

$$\widehat{r_{u_i,m_j}} = \log_2(r_{u_i,m_j} + 1) \tag{5}$$

C. Prediction Process using Tendencies Based Algorithm

The step is used to generate the prediction score of user's music taste given the current time based on Tendencies Based algorithm [8]. This algorithm captures to tendency users and items with observe means of users and items. For example [8], a user that only listens songs that he/she has liked will have a high mean. But if many other users also liked such songs, each song may actually have a higher mean rating that the rating given by the users. Thus the user tends to listen songs negatively, even despite their high mean. In an initial step of this process, we determine the tendency of user by considering mean of music or songs and mean of songs rated by users, and the tendency of song that considering average of user and music rated by user.

Tendency of user (ub_{u_i})

$$ub_{u_i} = \frac{\sum_{m_j \in M_u} (r_{u_i,m_j} - \overline{r_{\bullet,m_j}})}{|M_u|} \tag{6}$$

And, tendency of music or song (ib_{m_j})

$$ib_{m_j} = \frac{\sum_{u_i \in U_i} (r_{u_i,m_j} - \overline{r_{u_i,\bullet}})}{|U_i|} \tag{7}$$

where M_u : the number of songs rated by user u_i

U_i : the number of users listening music m_j

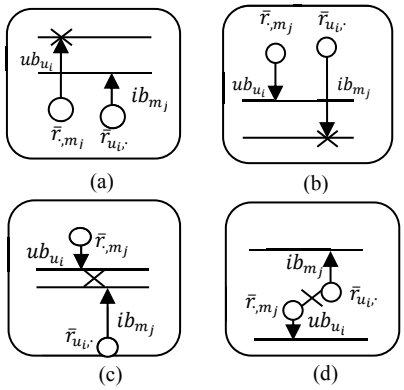


Fig. 2. Relation between mean (circle) of user and music and the tendency of user and music (arrow).

Then this algorithm calculates the prediction score by considering in 4 cases as the following.

The first case, Fig. 2(a), we observed that there are positive tendency both of user and music. So if tendency of user (ub_{u_i}) is more than mean of music ($\overline{r_{\bullet,m_j}}$) and tendency of music (ib_{m_j}) is more than mean of user ($\overline{r_{u_i,\bullet}}$), then prediction score will calculate as:

$$p_{u_i,m_j} = \max(\bar{r}_{u_i,\cdot} + ib_{m_j}, \bar{r}_{\cdot,m_j} + ub_{u_i}) \tag{8}$$

The second case, Fig. 2(b), we noticed that there is negative tendency of both user and music. This means that if tendency of user (ub_{u_i}) is less than mean of music ($\overline{r_{\bullet,m_j}}$) and tendency of music (ib_{m_j}) is less than mean of user ($\overline{r_{u_i,\bullet}}$), then prediction score will calculate as:

$$p_{u_i,m_j} = \min(\bar{r}_{u_i,\cdot} + ib_{m_j}, \bar{r}_{\cdot,m_j} + ub_{u_i}) \tag{9}$$

The third case, Fig. 2(c), we observed that positive item and negative user which observe from tendency of items is more than mean of user, but tendency of the user is less than mean of item. So if tendency of user (ub_{u_i}) is less than mean of music ($\overline{r_{\bullet,m_j}}$) and tendency of music (ib_{m_j}) is more than mean of user ($\overline{r_{u_i,\bullet}}$) and mean of music ($\overline{r_{\bullet,m_j}}$) is more than mean of user ($\overline{r_{u_i,\bullet}}$), then prediction score will calculate as:

$$p_{u_i,m_j} = \min[max(\bar{r}_{u_i,\cdot}, (\bar{r}_{\cdot,m_j} + ub_{u_i}) \propto + \left(\bar{r}_{u_i,\cdot} + ib_{m_j}\right)(1-\propto)), \bar{r}_{\cdot,m_j})] \quad (10)$$

where, alpha is the greater confidence value of mean of user and music.

The last case, Fig. 2(d), tendency have distinct difference tendency. This means that tendency of user and item is opposite. So if tendency of music (ib_{m_j}) is more than mean of user ($\overline{r_{u_i,\cdot}}$) and tendency of user (ub_{u_i}) is less than mean of music ($\overline{r_{\cdot,m_j}}$) and mean of user ($\overline{r_{u_i,\cdot}}$) is more than mean of music ($\overline{r_{\cdot,m_j}}$), then prediction score will calculate as:

$$p_{u_i,m_j} = \bar{r}_{\cdot,m_j} \propto + \bar{r}_{u_i,\cdot}(1-\propto) \quad (11)$$

D. Prediction Process using Item Mean Algorithm

Item Mean algorithm [8] is the other algorithms used to predict users' preference or tendency score. This is a simple algorithm just considering items of users' evaluation. We use mean of item in prediction by not take into account variation of user.

$$p_{u_i,m_j} = \overline{r_{\cdot,m_j}} \quad (12)$$

E. Prediction Process using Simple Mean Based Algorithm

This algorithm is used to predict users' preference [8] considering mean of items and mean of users as shown in the following equation.

$$p_{u_i,m_j} = \bar{r}_{\cdot,m_j} + \frac{\sum_{k \in M_u}(r_{u_i,m_k} - \bar{r}_{\cdot,m_k})}{|M_u|} \quad (13)$$

By M_u number of item at user u_i evaluation

F. Evaluation

This work uses the statistics measurement such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) [12] to evaluate our proposed model. The followings are the equation of two measurements.

$$MAE_{u_i} = \frac{\sum_{j=1}^{n} |r_{u_i,m_j} - pr_{u_i,m_j}|}{n} \quad (14)$$

$$MAE = \frac{\sum_{i=1}^{n} MAE_{u_i}}{N} \quad (15)$$

And, Root Mean Square Error (RMSE)

$$RMSE_{u_i} = \frac{\sum_{j=1}^{n} \sqrt{(r_{u_i,m_j} - pr_{u_i,m_j})^2}}{n} \quad (16)$$

where pr_{u_i,m_j} : the predicted score of music m_j and user u_i

n : the number of music the user u_i has evaluated

N : the number of users

IV. Experimental Results

The evaluation process in this work is a comparison among, Simple Mean Based algorithm, Item Mean algorithm, and Tendencies Based algorithm for all data which are not segmented into the specified time range. Besides, we will compare Tendencies Based algorithm, Simple Mean algorithm and Item Mean algorithm incorporated with Time Contexts data sets because these three algorithms have the same computational time, which show details later. This comparison is illustrated in Table III. All experiments, we run on the CPU speed at 3.20 GHz with 8 GB main memory, ad use Python programming language to develop program.

Fig. 3 shows MAE and RMSE calculated from all dataset, which are not segmented into specified time range, of Simple Mean algorithm, Item Mean algorithm and Tendencies Based algorithm. This graph shows that Tendencies Based algorithm gives the lowest errors both of MAE and RMSE. This means that we can combine Tendencies Based algorithm and Time Contexts to improve accurately predict user's preference.

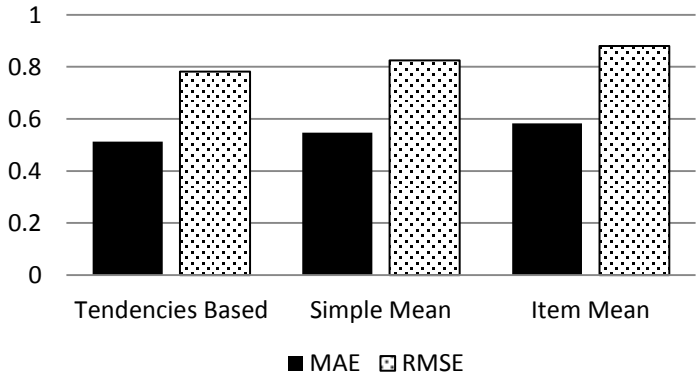


Fig. 3. MAE and RMSE of Tendencies Based algorithm, Simple Mean algorithm, Item Mean algorithm and user-based collaborative filtering.

TABLE III. Comparison among Tendecies Based algorithm, Simple Mean algorithm, and Item Mean algorithm with Time Contexts data sets.

Data	Item Mean		Simple Mean		Tendencies Based	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
Morning	0.4113	0.6208	0.3973	0.5521	0.3656	0.5315
Afternoon	0.2799	0.4598	0.2866	0.4047	0.2545	0.3769
Night	0.3387	0.5272	0.3331	0.4681	0.3067	0.4498
Weekend	0.3267	0.5149	0.3273	0.4606	0.2884	0.4297
Weekday	0.4904	0.7162	0.4588	0.6328	0.428	0.6137

The results displayed in Table III shows that when considering with algorithms having the same as computational complexity, we can see that Tendencies Based Algorithm with

Time Contexts still gives the lowest errors in both of MAE and RMSE with all of Time Contexts data sets.

V. CONCLUSION AND FUTURE WORK

In this paper, we compare different techniques of collaborative filtering especially the constant prediction time of such algorithm: Tendencies Based Algorithm; Item mean algorithm, Simple mean based algorithm. To see the performance of these algorithm with contextual information, each single user's profile is divided into many Time Contexts based on pre-defined time segmentation. These Time Contexts can improve the prediction accuracy and can meet the users' needs Moreover; the adaptation of Tendencies Based Algorithm with Time Contexts can overcome traditional collaborative filtering problems especially high accuracy and more efficiency.

In future work, we will alleviate sparsity problem and cold start problem at new songs added in system that user has not been recommended. This problem suffers in collaborative filtering and tendencies based algorithms. Besides, this work uses only time contextual that we should take into account genres or artist for increase accuracy in prediction.

REFERENCES

[1] K.Marius and F. Ricci, "Contextual music information retrieval and recommendation:state of the art and challenges,"Computer Science Review, Vol.6, pp. 89–119, 2012.

[2] J.Bobadilla, F.Ortega, A.Hernando, A.Gutierrez, "Recommender systems survey" ,Knowledge-Based Systems, Vol.46, pp.109-132, July, 2013.

[3] C.Laurent, J.Kris, F.Francoise, M.Frank, State-of-the-Art Recommender Systems , IGI Global, 2009.

[4] C. H. Park and M. Kahng, "Temporal Dynamics in Music Listening Behavior:A Case Study of Online Music Service ",The 9th International Conference on Computer and Information Science (ACIS 2010), pp.573-578, 18-20 August,2010.

[5] B. Linas and A. Xavier, "Towards time-dependant recommendation based on implicit feedback," CARS-2009, New York, USA, October 25, 2009.

[6] S. Antonietta, A. Albert, D. Helenca, E. Gregorio, and V. Paulo, "My Personal Media Entertainer: Context-Adaptive Content Recommendation and Delivery ", ACM Workshop on Social, Adaptive and Personalized Multimedia Interaction and Access, pp.33-36, 2010.

[7] T. Cebrián, M. Planagumà, P. Villegas, and X. Amatriain, "Music Recommendations with Temporal Context Awareness," Recommender Systems (RecSys2010), pp.349-352, 2010.

[8] F. Vreixo, C. Fidel and C. Victor, "Algorithms for Efficient Collaborative Filtering",2008

[9] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," Proceedings of the 10th international conference on World Wide Web(www2001), Hong Kong, May 1-5, 2001.

[10] C.Oscar. Last fm Dataset. [online]. http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/lastfm-1K.html.

[11] B. Xu, J. Bu, C. Chen, and D. Cai, "An Exploration of Improving Collaborative Recommender Systems via User-Item Subgroups "WWW2012, Lyon, France, Apr 16-20, 2012.

[12] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," CLIMATE RESEARCH, Vol. 30, pp.79-82, Dec 19, 2005.

JCSSE 2015

2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE)

"Shaping the Future with Convergence"

22 – 24 July 2015

Prince of Songkla University, Hat Yai, Songkhla, Thailand

Workshop on e-Science and High Performance Computing (eHPC 2015)

IEEE Catalog Number : CFP1532P-USB

ISBN : 978-1-4799-1965-9

SIPA

IEEE

IEEE THAILAND SECTION

Technical Co-Sponsor ECTI Association

SIFT

Table of Contents

Computational Intelligence

The Constant Time of Predictive Algorithm for Music Recommendation with Time Context

Sumet Darapisut, Ureerat Suksawatchon, and Jakkarin Suksawatchon
Faculty of Informatics, Burapha University
Chonburi, 20131 Thailand
dearsumet@gmail.com, ureerat.w@gmail.com, jakkaman@gmail.com

Abstract— **In this research, we propose using time context to improve predictive accuracy and quality of collaborative filtering for music recommendation. We use time contextual information called *micro-profiling*. Thus, each user has multiple micro profiles, in particular, six-time slots instead of a single profile. The recommendation is performed depended on these micro-profiling. Our method takes into account time intervals in listening music represent user preference in up-to-date. In rating prediction approach, we adopt *tendencies-based* algorithm, one of the collaborative filtering algorithms, which is very low computational time and memory requirements. From the experimental results, it has shown that the performance of our approach gives more accuracy and low computational time than traditional CF and CF with matrix factorization.**

Keywords- music recommender system; time contexts; micro profiling; tendencies based algorithm; predictive model

I. INTRODUCTION

As the amount of digital music increases tremendously nowadays, it has become more difficult for users to choose related digital music. The music recommender system has been developed to help and discover digital music among information overloading condition. The successful and well-known music recommendation is collaborative filtering (CF) approach especially user-based CF. This approach predicts a user's preferences (ratings) on target item by aggregating the preferences that a few similar users have previously given to that item. Similar users are determined using a similarity metric such as the Pearson correlation or the cosine similarity and the K-nearest users with the highest similarities to the given user are selected. Then the predicted rating on that item is generated based considering and analyzing the information of K nearest users [1, 2, 3, 4]. However this approach presents serious problems. First, the computation of similarities between all pairs of users is expensive in quadratic time complexity. Second, the recommendation accuracy depends on the adopted similarity measure [2]. Furthermore, most of recommendation system encounters with sparsity problem because each user rates or gives his/her preference only a small number of the available items. In this case, finding the similarity among different users is challenging task [3].

Most of music recommender systems often recommend in a few popular music in such that users do not prefer or cannot always meet the users' need. Because these systems do not consider the users' current situations, and users may have different preferences in different contexts. For example, users listen to one type of music while working and another type of music before going sleep [5, 6, 7]. Therefore, the additional information beyond users' rating plays an important role in determining users' preference for building predictive models to enhance the recommendation performance. For example, C. H. Park and M. Kahng [5] have presented behavior's user that founded time dimension in music listening is different other contextual dimensions. Because some genres of music are preferred at specific time of day such as ballad, dance, pop, and rock genres has a peak at noon and decrease rapidly in the evening. The ballad song has been listened more than pop dance genre in winter but, in summer pop dance genres listened to more than ballad song [6]. L. Baltrunas and X. Amatriain [7] proposed a new time-aware recommendation approach called micro profiling. This approach assumes that the users' preference changes over time but has a temporal repetition. Thus, it splits each single user profile into sub-profiles that best represent the user in a particular time slot. Pre-defined time segmentation is done for the day (morning, evening/night), the week (weekend, weekday), and the year (cold season, hot season) temporal repetition. Then, the popular factorization based on CF algorithm is performed depended on these micro-profiles. This means that they use only the profiles of the relevant segment. For example, they use only the user micro-profile of the morning to predict the music preference for the morning. Although, this approach considers additional information for improving the music recommendation, it still faces serious problems caused by using CF algorithms that we mentioned such problems before. Besides, this approach cannot predict user's rating on overlapping time partitions such as when predicting a rating for the morning on a weekend.

F. Cacheda and et al. [8] presented a variety of CF named *Tendencies-Based CF* algorithm to overcome those drawbacks. This idea based on the tendencies or difference between users and items instead of on their similarities. Tendencies-Based CF is easy to compute, and can be accurately calculated using much fewer data with very low computational time and memory requirements. This approach is as accurate as most modern methods, and its computational efficiency is much better based on two popular datasets, MovieLens and Netflix. Unfortunately, this approach may be not suitable for other recommendations like music recommendation. Besides, the addition information has not been used in their research that only the ratings are considered. Especially, time context conditions influence to music listening behavior.

978-1-4799-1966-6/15/$31.00 ©2015 IEEE

In this work, we propose a constant time of CF used in music recommendation. Our approach adopts Tendencies-Based CF algorithm based on time contextual information that is implicit user feedback. Our approach assumes that the music listening behavior changes over time but a temporal repetition. For example, users listen to one type of music while working and another type of music before going to sleep. We use time contextual information called micro-profiling that is similar to paper reference no. [7]. A single user has multiple micro profiles in particular time contexts instead of a single profile. Unlike [7], we concern the overlapping time slots. Thus, we decide on six different micro profiling generated by the Cartesian product of set of the day (morning, afternoon, evening/night) and a set of the week (weekend, weekday). The six-time slots will be the same for all users. Our experiments present performance of our method that gives more accuracy and low computational time than traditional CF and CF with matrix factorization.

The outline of the rest of this paper is as follows. In Section II, notation is defined. The background knowledge is introduced in Section III. Our approach is proposed in section IV. Section V illustrates the experimental results of our approach. Section VI is the conclusion of this paper.

II. NOTATIONS

First, we introduce all notations used in this paper as shown in Table I.

TABLE I. NOTATIONS USED IN THIS WORK

Notation	Description
$U = \{u_1, u_2, \dots, u_i, \dots, u_N\}$	Set of N users
$M = \{m_1, m_2, \dots, m_j, \dots, m_K\}$	Set of K songs
R	User-Music rating Matrix $\lvert U\rvert \times \lvert M\rvert$
r_{u_v,m_j}	Rating of song m_j listened by user u_v
r_{u_i,m_j}	Rating of song m_j listened by user u_i
r_{u_i,m_k}	Rating of song m_k listened by user u_i
$r_{u_i,\cdot} = \{r_{u_i,m_i} \in R \lvert m_j \in M\}$	Set of rating of songs listened by user u_i
$r_{\cdot,m_i} = \{r_{u_i,m_i} \in R \lvert u_i \in U\}$	Set of rating of music m_j listened by users
$\bar{r}_{u_i,\cdot}$	Rating mean of user u_i
$\bar{r}_{u_v,\cdot}$	Rating mean of user u_v
$\bar{r}_{\cdot,m_j}$	Rating mean of music m_j
p_{u_i,m_j}	Rating prediction for user u_i listening music m_j

III. BACKGROUND KNOWLEDGE

A. Collaborative Filtering Algorithm (CF)

In user-based CF, the system allows users to give a set of user ratings on items (videos, songs, films, etc.) in such that when enough information is stored in the system, we can make recommendations for each user based on information provided by users who have the most in common preference [1]. So in music recommendation, this approach predicts a user's preferences (ratings) on target music by aggregating the preferences that a few similar users previously given to that music. The similarity measure between users is determined. For now, let $sim(u_i, u_v)$ be the similarity between users *i* and *v*. The popular similarity metrics are cosine similarity, Pearson correlation similarity, and adjusted cosine similarity as shown in (1), (2), and (3) respectively. Then the *K*- nearest neighbors (*K*-nn) is applied to find other users who are the most similarity to the target user. After that, user's preference for each music is computer based on the best neighbors' preferences by using (4) [4].

$$sim(u_i, u_v) = \frac{\sum_j (r_{u_i,m_j}, r_{u_v,m_j})}{\sqrt{\sum_j (r_{u_i,m_j})^2}\sqrt{\sum_j (r_{u_v,m_j})^2}} \quad (1)$$

$$sim(u_i, u_v) = \frac{\sum_{j \in M_{ui} \cap M_v} \left(r_{u_i,m_j} - \bar{r}_{u_i,\cdot}\right)\left(r_{u_v,m_j} - \bar{r}_{u_v,\cdot}\right)}{\sqrt{\sum_{j \in M_{ui} \cap M_v} \left(r_{u_i,m_j} - \bar{r}_{u_i,\cdot}\right)^2}\sqrt{\sum_{j \in M_{ui} \cap M_v} \left(r_{u_v,m_j} - \bar{r}_{u_v,\cdot}\right)^2}} \quad (2)$$

$$sim(u_i, u_v) = \frac{\sum_j \left(r_{u_i,m_j} - \bar{r}_{\cdot,m_j}\right)\left(r_{u_v,m_j} - \bar{r}_{\cdot,m_j}\right)}{\sqrt{\sum_j \left(r_{u_i,m_j} - \bar{r}_{\cdot,m_j}\right)^2}\sqrt{\sum_j \left(r_{u_v,m_j} - \bar{r}_{\cdot,m_j}\right)^2}} \quad (3)$$

$$p_{u_i,m_j} = \bar{r}_{u_i,\cdot} + \frac{\sum_{u_v \in U} sim(u_i,u_v)\left[r_{u_v,m_j} - \bar{r}_{u_v,\cdot}\right]}{\sum_{u_v \in U} \lvert sim(u_i,u_v)\rvert} \quad (4)$$

B. Matrix Factorization

Matrix factorization technique (MF) [9] is a decomposition matrix approach that factorizes the original matrix into two or more matrices. When multiplying them back, it will get the original matrix as shown in (5). In recommendation system, matrix factorization can be used to discover some latent features. If we can find these latent features, we can predict rating with respect to a certain user and a certain item [8].

The mathematics of matrix factorization is described as follows. Let R be the matrix of size $\lvert U\rvert \times \lvert M\rvert$ that contains all the ratings that the users have assigned to the music. If we would like to find K latent features, two metrics user (*i.e.,* $P = \lvert U\rvert \times K$) and music (*i.e.,* $Q = \lvert M\rvert \times K$) from rating pattern are estimated such that their product approximated R:

$$R \approx P \times Q^T = \hat{R} \quad (5)$$

Where $\hat{R}$: approximation matrix $\lvert M\rvert \times \lvert U\rvert$

IV. OUR APPROACH

The framework of our system is described in Fig. 1.

A. Last.fm Data Information

In this work, we use last.fm data [7] set collected during 2005 to May 2009 containing 357 European users who used last.fm services. Each user listened to songs and was stored in the uses' profile together with an appropriate time stamp [7]. This service provides only implicit user feedback as shown in Table II. We also cleaned the data by removing songs that were listened less than five users.

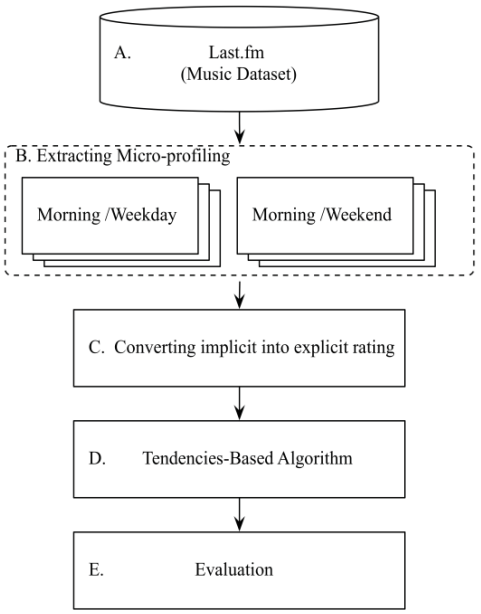


Figure 1. Overall our approach

TABLE II. THE EXAMPLE OF IMPLICIT USER FEEDBACK OF LAST.FM DATA SET

User id	Timestamp	Artist id	Artist name	Music id	Music name
u1	2009-04-08 T01:17:04Z	a1	Something	m1	Aliean
u1	2009-04-08 T23:51:45Z	a2	Simple	m2	Space
u1	2009-04-08 T23:51:45Z	a2	Simple	m2	Space
…	…	…	…	…	…

B. *Extracting Micro-profiling*

Our proposed method represents a single user by multiple micro profiles in particular time segmentation. We defined time segmentation in forms of the day (morning, afternoon, evening/night) and the week (weekend, weekday). Morning is defined as 00:00-11:59 am. The afternoon is defined as 12:00-17:59 pm and evening/night is defined as 18:00-23:59 pm. The weekend is covered only Saturday and Sunday), and Weekday is for Monday to Friday. After that, six different micro profiles are generated by using the Cartesian product between a set of the day and set of the week as shown in Fig 2. Thus, six-time slots will be used for all users. To make recommendations, we will use our pre-defined micro profiles instead of a single profile for each user in the particular time slot. After that, we create User-Music frequency matrix (implicit information) for each micro profile as described by Fig. 3. Table 3 is shown the summary of each micro profile.

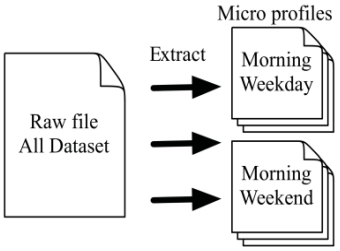


Figure 2. Extracting all data into micro profiles

User_id	Timestamp	Art_id	Artist	M_id	Music
u1	2009-04-08 T01:17:04Z	a1	Some Thing	m1	Aliean
u1	2009-04-08 T01:47:40Z	a2	Simple	m2	Space
u1	2009-04-08 T01:57:47Z	a2	Simple	m2	Space
...	...	...	...	...	...

Create User-Music Frequency Matrix

	m1	m2	m3	m4	...
u1	120	80	71	68	...
u2	62	2	0	20	...
u3	2	18	100	32	...
...	...	...	...	...	...

Figure 3. One of micro profile was converted to User-Music frequency matrix.

TABLE III. SUMMARY OF EACH MICRO PROFILE

Micro Profiling	# of Users	# of Tracks	# of Frequency
Morning/Weekday (MWday)	354	20,928	197,269
Afternoon/Weekday (AWday)	354	27,315	272,429
Night/Weekday (NWday)	351	23,971	233,864
Morning/Weekend (MWend)	340	8,505	67,704
Afternoon/Weekend (AWend)	345	12,322	106,936
Night/Weekend (NWend)	337	10,290	87,195

C. *Converting implicit into explicit rating*

Most CF algorithms use explicit feedback in form of ratings that are collected directly from users. In this work, we focus on the music domain, and there is no explicit user feedback. Thus, we need to map implicit information into the explicit ratings. This work uses mapping method as same as used in the research no. [7]. Firstly, we compute the complementary cumulative distribution of song plays in the micro profile. Songs located in the top 80-100% of the distribution are assigned a score of 5, while songs in the 60-80% range assign a score of 4 as shown in Table IV. Table V shows the example of mapping to rating score for a single user

who plays overall songs 1,631 times from whole dataset. Thus, songs at 100-80% of distribution get rating 5 and song at 79-60% of distribution get rating 4. Table VI shows the rating distribution of the data set. This is a specific property of the music data sets since a single user listens to a lot of unique songs [7]. Figure 4 shows the results of mapping User-Music frequency matrix to User-Music rating matrix for further used in next step.

TABLE IV. MAPPING PERCENTAGE OF CUMULATIVE DISTRIBUTION OF PLAYS INTO RATING SCALE

Percentage	Rating Mapping
100-80%	5
79-60%	4
59-40 %	3
39-20 %	2
19-0 %	1

TABLE V. THE EXAMPLE OF MAPPING IMPLICIT INTO EXPLICIT RATING FOR A SINGLE USER

Frequency (x)	120	80	71	68	…
P(X=x)	120/61	80/161	71/1631	68/1631	…
Cumulative F(x)	1-0.074	1-0.123	1-0.167	1-0.042	…
Percentage	92.6	87.7	83.3	71	…
Rating	5	5	5	4	…

TABLE VI. NUMBER OF RATINGS IN THIS WORK

Rating	1	2	3	4	5
#Ratings	692,015	11,328	4,017	2,073	993
Percentage	97.41	1.6	0.57	0.29	0.14

	m1	m2	m3	m4	...
u1	120	80	71	68	...
u2	62	2	0	20	...
u3	2	18	100	32	...
...	...	...	...	...	...

Mapping to User-Music Rating Matrix

	m1	m2	m3	m4	...
u1	5	5	5	4	...
u2	4	1	0	2	...
u3	1	2	5	3	...
...	...	...	...	...	...

Figure 4. Mapping User-Music frequency matrix to User-Music rating matrix.

D. Tendencies-based algorithm

This work uses an alternative approach of CF named *tendencies-based* algorithm [8], to predict user's music preference in term of rating, given the current time. Unlike CF, the tendencies-based algorithm is not based on the similarities between users or items. This approach captures tendencies of users and songs by finding a relation between them. Instead of using similarity calculations, tendencies are an easy way to compute, and they can be accurately computed using much fewer data [8]. This is a very interesting feature of the tendencies-based algorithm because it will allow the development of accurate method with very low computation time and memory requirement.

The idea of the tendencies-based algorithm is to predict user's music preference by considering tendency of users and songs. This refers to whether a user tends to rate songs positively or negatively. For example [8], a user that only listen songs that he/she has liked will have a high mean, but if many other users also liked such songs, each song may actually have a higher mean rating that the rating given by the users. Thus, the user tends to listen to songs negatively, even despite their high mean.

In the first step, we have to determine the tendency of user u_i (ub_{u_i}) as the average difference between user's ratings and the song mean as shown in (6).

$$ub_{u_i} = \frac{\sum_{m_j \in M_u}\left(r_{u_i,m_j} - \bar{r}_{\cdot,m_j}\right)}{|M_u|} \quad (6)$$

For tendency of songs, we define the tendency of song m_j (ib_{m_j}) as the average difference between user's ratings and the user mean as shown in (7).

$$ib_{m_j} = \frac{\sum_{u_i \in U_i}\left(r_{u_i,m_j} - \bar{r}_{u_i,\cdot}\right)}{|U_i|} \quad (7)$$

Next steps, we calculate the predicted rating by considering in 4 cases.

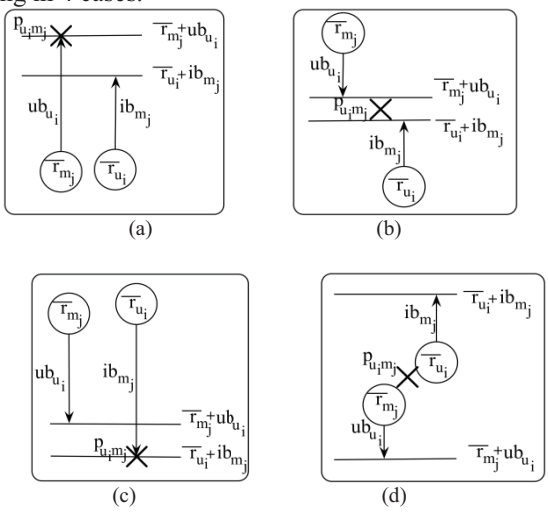


Figure 5. The relation between mean (circle) of user and song, the tendency of user and song (arrow) and rating prediction (cross).

Case I, Fig. 5(a), both user and song have positive tendency. This means user tends to give ratings that are above the song mean rating and the song tends to be rated above the user mean [8]. Then, if tendency of user u_i (ub_{u_i}) is more than average of song m_j ($\bar{r}_{\cdot,m_j}$) and tendency of song m_j (ib_{m_j}) is more than mean of user u_i ($\bar{r}_{u_i,\cdot}$), then we use formula (8) to predict rating for user u_i that will listen song m_j.

$$p_{u_i,m_j} = \max\left(\bar{r}_{u_i,\cdot} + ib_{m_j}, \bar{r}_{\cdot,m_j} + ub_{u_i}\right) \quad (8)$$

Case II, Fig. 5(b), occurs when both user and song have negative tendency. This means the user rates songs below their mean and the song tends to be rated below the user mean [6]. Thus, if tendency of user u_i (ub_{u_i}) is less than

average of song m_j ($\bar{r}_{.,m_j}$) and tendency of song m_j (ib_{m_j}) is less than average of user u_i ($\bar{r}_{u_{i,.}}$), then rating prediction for user u_i listening song m_j, are calculated using formula (9).

$$p_{u_i,m_j} = \min(\bar{r}_{u_{i,.}} + ib_{m_j}, \bar{r}_{.,m_j} + ub_{u_i}) \quad (9)$$

Case III, Fig. 5(c), occurs when the user tendency is to rate songs below their mean, and a good song is to be rated above the user mean. Thus, if tendency of user u_i (ub_{u_i}) is less than mean of song m_j ($\bar{r}_{.,m_j}$) and tendency of song m_j (ib_{m_j}) is more than mean of user u_i ($\bar{r}_{u_{i,.}}$). For corroborate their tendencies, we notice mean of song is more than mean of user or vice versa, then rating score for user u_i listening song m_j, can be predicted using formula (10)

$$p_{u_i,m_j} = \min[\max\left(\bar{r}_{u_{i,.}}, \left(\bar{r}_{.,m_j} + ub_{u_i}\right) \propto + \left(\bar{r}_{u_{i,.}} + ib_{m_j}\right)(1-\propto)\right), \bar{r}_{.,m_j}] \quad (10)$$

where, $\propto$ is the greater confidence value of the average of user and song.

Case IV, Fig. 5(d), their tendency are not corroborate. This means that tendency of user and song is opposite. So if tendency of song m_j (ib_{m_j}) is more than mean of user u_i ($\bar{r}_{u_{i,.}}$) and tendency of user u_i (ub_{u_i}) is less than mean of song m_j ($\bar{r}_{.,m_j}$) but mean of user u_i ($\bar{r}_{u_{i,.}}$) is more than mean of song m_j ($\bar{r}_{.,m_j}$), then predicting score for user u_i listening song m_j, can be calculated using formula (11):

$$p_{u_i,m_j} = \bar{r}_{.,m_j} \propto + \bar{r}_{u_{i,.}}(1-\propto) \quad (11)$$

E. Evaluation

The objective of recommendation system is to recommend only good songs. The traditional prediction accuracy metric, such as mean absolute error (MAE) [10], is not suitable and not the best evaluation metric for the find good songs task. Because MAE metric calculates the average error committed on the whole songs. In this work, we used GIM (Good Item MAE) as same as used in [8] that compute the MAE in the prediction of good songs. In this work, we set a rating threshold for considering good songs those rated with more than 1 point. GIM metric is shown in (12) and (13).

$$GIM_{u_i} = \frac{\sum_{j=1}^{n} |r_{u_i,m_j} - p_{u_i,m_j}|}{n} \quad (12)$$

$$GIM = \frac{\sum_{i=1}^{N} GIM_{u_i}}{N} \quad (13)$$

In addition, this work also evaluated top-N recommendation quality by precision, recall, and F1 measure as shown in equation (14), (15) and (16) respectively. Table VII presents the details of these equations.

$$Precision = \frac{\#\text{tp}}{\#\text{tp}+\#\text{fp}} \quad (14)$$

$$Recall = \frac{\#\text{tp}}{\#\text{tp}+\#\text{fn}} \quad (15)$$

$$F1\ measure = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision}+\text{recall}} \quad (16)$$

TABLE VII. CONDITION OF RECOMMENDATION EVALUATIONS

	Recommended	**Not Recommended**
Preference	tp	fn
Not Preference	fp	tn

V. EXPERIMENTAL RESULTS

This work evaluated using last.fm data set collected during 2005 to May 2009 containing 357 European users who used last.fm services described in section IV. The proposed approach was compared with the following algorithms.

- The classical user-based collaborative filtering (CF). We used cosine based similarity and 10% of users as best nearest neighbors.
- The collaborative filtering with matrix factorization (MF) [9]. In MF, we set the latent factor to 80, iteration to 100, learning rate to 0.005 and beta to 0.02.

Both of CF and MF were tested for the full data set and each micro profile. Those data sets were transformed into explicit ratings as described in section IV (C).

Fig. 6 shows the results evaluated with GIM metric. Testing with full data set, GIM values of the tendencies-based algorithm, CF, and MF are 3.2193, 3.2253, and 3.2195 respectively. Evaluating with each micro profile, most of the results obtained from these three algorithms got GIM accuracy with similar results.

For recommendation quality evaluation, we evaluated with Precision, Recall and F1 measure in top-10 recommendation. The results are shown in Figs. 7, 8, and 9 respectively. From experimental results illustrate that tendencies-based using micro profiling has given higher quality than CF and MF for all data sets. This means that *tendencies-based with micro profiling* (our approach) could be informative enough to determine the users' behavior accordance with time context. Thus, our approach can improve the accuracy and quality of the recommendations.

This work also compared with computational efficient of tendencies-based algorithm, CF, and MF. In offline experiment, CF needs more time to compute similarity between all users with a complexity of $O(m^2n)$ and time to compute a single prediction is $O(mn)$. For MF carries out expensive computational because matrix factorization technique is based on SVD with a complexity of $O(mnk + m^2n)$. Our approach obtains the best results with a complexity of $O(mn)$ in computing equations (6) and (7), and with a complexity of $O(1)$ in prediction process, where m is a number of users, n is a number of music, and k is a number of factors. Table VIII shows the execution time used in those three algorithms.

Table VIII shows the execution time used in those three algorithms. Time was measured on a PC with CPU speed at 3.20 GHz, 8 GB RAM, and all algorithms were implemented in Python.

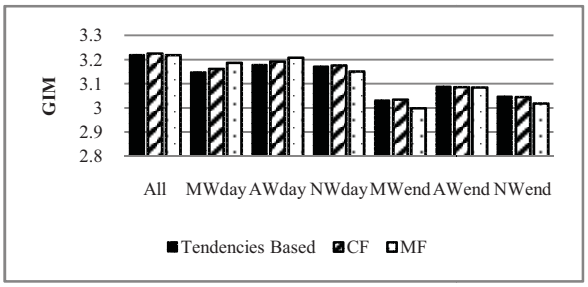


Figure 6. Comparison with GIM among Tendencies-based, CF and MF.

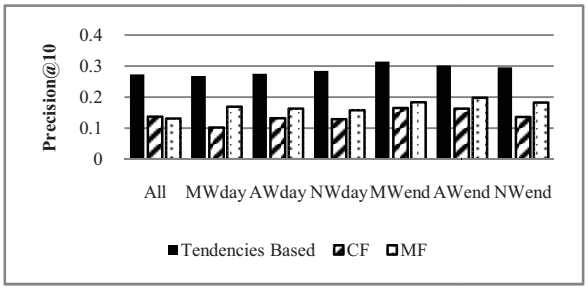


Figure 7. Comparison with Precision value in top-10 recommendations.

TABLE VIII. EXECUTION TIME OF THE DIFFERENT THREE ALGORITHM STUDIED WITH MICRO PROFILING

Approach	Model Construction Time	Prediction time
CF	2,102.34 sec.	456.18 sec.
MF	86,361.95 sec.	1,028.34 sec.
Our approach	**11.81 sec.**	**9.39 sec.**

VI. CONCLUSION

In this work, we exploit time context to improve predictive accuracy and quality of collaborative filtering for music recommendation. Because most of the music recommender systems often recommend songs in a few popular music in such that users do not prefer and do not consider the users' current situations. Especially, the users' preference may change over time but has a temporal repetition. Therefore, this research introduces time contextual information called *micro profiles*. The overlapping micro profiles are generated by using Cartesian product between a set of the day and set of the week. Thus, a single user has multiple micro profiles in particular time contexts instead of a single profile. In rating prediction approach, this work adopts *tendencies-based* algorithm, one of the collaborative filtering algorithms, which is a constant time in this process and very low memory requirements. Thus, applying tendencies-based algorithm is done using those micro profiles in particular time slots. In this work, we compare our approach (tendencies-based algorithm with micro profiles) with CF and MF (CF with matrix factorization). Both CF and MF are also done using those micro profiles. From the experimental results, it has shown

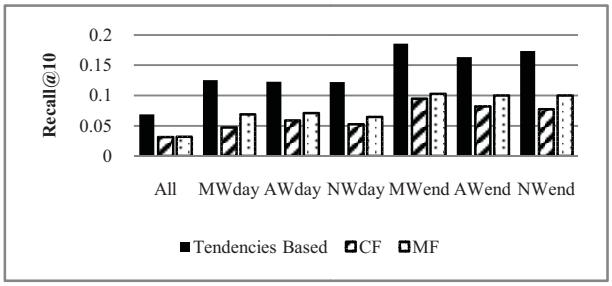


Figure 8. Comparison with Recall value in top-10 recommendations.

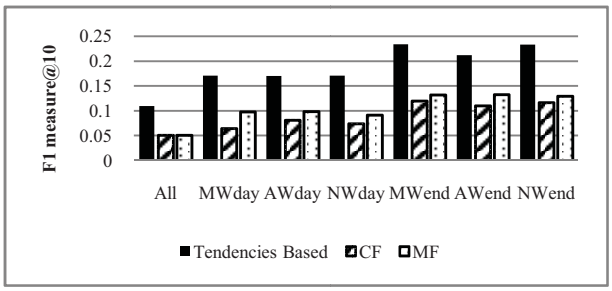


Figure 9. Comparison with F1 measure value in top-10 recommendations

that the performance of our approach gives more accuracy and low computational time than traditional CF and MF.

REFERENCES

[1] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the User-Item matrix: a survey of the state of the art and future challenges", ACM Computing Surveys, Vol. 47, No. 1, Article 3, pp. 3:1-3:45, April, 2014.

[2] K. Marius and F. Ricci, "Contextual music information retrieval and recommendation:state of the art and challenges", Computer Science Review, Vol.6, pp. 89–119, 2012.

[3] J.Bobadilla, F.Ortega, A.Hernando and A.Gutierrez, "Recommender systems survey", Knowledge-Based Systems, Vol.46, pp.109-132, July, 2013.

[4] L. Candillier, K. Jack, F. Fessant and F. Meyer, State-of-the-Art Recommender Systems , IGI Global, 2009.

[5] C. H. Park and M. Kahng, "Temporal Dynamics in Music Listening Behavior:A Case Study of Online Music Service", The 9th International Conference on Computer and Information Science (ACIS 2010), pp.573-578, 18-20 August, 2010.

[6] S. Darapisut, J. Suksawatchon, "Comparison of the Constant Prediction Time of Collaborative Filtering Algorithms by Using Time Contexts", The 11th International Joint Conference on Computer Science and Software Engineering (JCSSE 2014), pp.302-306, May, 2014.

[7] L. Baltrunas and X. Amatriain, "Towards time-dependant recommendation based on implicit feedback", CARS-2009, New York, USA, October 25, 2009.

[8] F.Cacheda., V. Carneiro, D. Fernandez, and V. Formoso, Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. Vol. 5, No. 2, February, 2011.

[9] G. Takács, I. Pilaszy, B. Nemeth, D. Tikk, "Matrix factorization and neighbor based algorithms for the Netflix prize problem", Proceedings of the 2008 ACM Conference on Recommender Systems, Lausanne, Switzerland, October 23 - 25, 2008.

[10] G. Shani and A. Gunawardana, Evaluating Recommendation Systems. Technical report, No. MSR-TR- 2009-159, 2009.

IEEE

ICSEC 2015 The 19th International Computer Science and Engineering Conference (ICSEC)

ICSEC 2015

ICSEC 2015

Chiang Mai, Thailand

The 19th International Computer Science and Engineering Conference (ICSEC)

Hybrid Cloud Computing: A New Approach for Big Data Era

IEEE
THAILAND SECTION

Duangtawan Hotel,
Chiang Mai, Thailand

23 - 26 November 2015

IEEE THAILAND SECTION
IEEE Xplore Digital Library
ARCCIT

NETBRIGHT
Hewlett Packard Enterprise
TOT
CISCO
ECTI Association
IBM

Program at a glance

Day 2 : 24 November 2015

		Real-Time Sensor- and Camera-based Logging of Sleep Postures		ระบบสร้างออนโทโลยีจากฐานข้อมูลเชิงสัมพันธ์แบบอัตโนมัติ	การพยากรณ์ข้อมูลเกี่ยวกับอันตรายของแผ่นดินไหวในเหมืองถ่านหินด้วยตัวแบบเรียนรู้แบบเอ็กซ์ตรีมถ่วงน้ำหนัก	Performance Analysis of a Multi-Tier Video on Demand Services on Virtualization Platforms	
14:40 – 15:00	Coffee Break						
15:00 – 17:00	**Technical Sessions**						
		Pattern Recognition & Analysis 2 Room: Chiang Roong	Data analysis Room : Chiang Dao 1	Thai Track (Network) 2 Room : Chiang Dao 2	Thai Track (Algorithm) 2 Room : Chiang Dao 3	Cloud & Big Data 2 Room : Duangtawa Grand Ballroom	HACKATHON* Workshop Day Room : Chiang Kham
	15:00 – 15:20	1570180689 Identification of Transcription Start Sites via Distribution of A/T-singletons	1570188625 Using Backpropagation Neural Networks for Flood Forecasting in PhraNakhon Si Ayutthaya,Thailand	1570184115 การวิเคราะห์ผลกระทบต่อกรณีทดสอบจากการเปลี่ยนแปลงส่วนต่อประสานกับผู้ใช้สำหรับโปรแกรมประยุกต์บนโทรศัพท์เคลื่อนที่แอนดรอยด์	1570213151 การแปลภาษาไทย-อีสานโดยใช้ฐานกฎ	(Invited Talk) Bring your own dev	
	15:20 – 15:40	1570206085 Analysis of Amino Acid Pairs Relationships Bas on Protein-Protein Interactions	1570213389 Predicting SET50 Stock Prices Using CARIMA (Cross Correlation ARIMA)	1570207439 ระบบยืนยันตัวตนแบบอัตโนมัติสำหรับควบคุมการเข้าถึงเครือข่าย	1570178883 การระบุผู้ต้องสงสัยจากรูปแบบในประวัติการโทร	1570166881 NMRexSeer: Metad Extraction and Sear for Large Scale Nuclear Magnetic Resource (NMR) Experimental Data	
	15:40 – 16:00	1570207357 Using Random Forest Based on Codon Usage for Predicting Human Leukocyte Antigen Gene		1570184407 การสร้างกรณีทดสอบจากแผนภาพบีพีเอ็มเอ็นร่วมกับแผนภาพบีเพล		1570208725 Performance Measurement of SimpleDB APIs for Different Data Consistency Models	
17:00 – 18:00	TPC Meeting: Operation Lounge 1						
18:00 – 20:00	ICSEC2015 Banquet @ Duangtawan Grand Ballroom (The 3th Floor) : All participants are welcome to join Banquet theme: Lanna Authentic Cocktail Dress code : Smart Casual (Please wear ICSEC 2015 Official Polo shirt, Nametag will be inspected)						

Note: Hackathon operate 24 hours, Room Chiang Kham will be reserved for 24 hours after technical sessions end.
Each keynote lecture has 45 minutes including Q&A Session.

Incremental Session Based Collaborative Filtering with Forgetting Mechanisms

Ureerat Suksawatchon*, Sumet Darapisut † and Jakkarin Suksawatchon‡
Faculty of Informatics, Burapha University
Chonburi, 20131 Thailand
*ureerat.w@gmail.com, †dearsumet@gmail.com, ‡jakkarin@informatics.buu.ac.th

***Abstract*—Most of research works in music recommendation systems use Collaborative Filtering (CF) for generating personalized recommendations based on user's previous song ratings or static usage history data. But those researches adapting CF do not consider behavior of listening to songs and are not able to maintain the systems to sensitive to recent user's preferences. Behavior of music listening is continuous and repetitive process, especially, the latest song listening can infer to the favorite song at that moment. In this work, we present *Incremental Session based Collaborative Filtering with forgetting mechanism* or *ISSCF* by adapting Session-based Collaborative Filtering (SSCF), which considers music listened continuously and maintains the recent session. In order to avoid unnecessary memory usage and processing time, we use forgetting mechanism: sliding windows and fading factors incorporating with SSCF. We evaluate our purposed framework by measuring the $HitRatio$. From experimental results, it shows that performance of our purposed approach increases the accuracy of recommendation and low computational time and space when comparing with than SSCF.**

***Keywords*—music recommendation; forgetting mechanism; session; collaborative filtering**

I. Introduction

Music Recommender System (MRS) has an important role to help the users to find songs that they really want from a large amount of songs. In provider aspect, MRS is able to filter and to choose appropriated music for the users. Most of MRS uses Collaborative Filtering algorithm (CF) for generating personalized recommended songs by considering users behaviors (such as rating, clicks, history and purchase, etc.) [1], [2], [3], [4]. However, CF requires many users and many ratings and is unable to recommend songs that have a few ratings. This means that users have to well provide their taste if they need effective recommendation.

Characteristic of music domain has different from other domains like movies, books and news. Listening to music is continuous and repetitive process. Especially, users tend to prefer to listen to preference songs repetitively in *session* rather than isolated [4], [5]. Accordance with S. E. Park and et al.[5] tries to capture order and repetitiveness in the playing songs. They proposed Session-based Collaborative Filtering approach (SSCF) for next song prediction with the currently played songs in Bugs Music dataset. SSCF adapt collaborative filtering based on user by taking into account relation of *session profiles* instead of user profiles. The experiments show that SSCF outperforms than traditional collaborative filtering in term of accuracy.

R. Dias and M.Fonseca [6] presented improving music recommendation approach named Temporal Session based Collaborative Filtering approach (TSSCF). This work extracts temporal context including time of day, weekday, a day of month, a month from session profiles, and takes into account the song diversity played in the session. After that, the TSSCF groups sessions according to different of temporal context by using Gaussian Mixture Model via Expectation Maximization algorithm. Finally, the part of recommendation approach is applied with SSCF. Comparing with the traditional session-based CF, the TSSCF can achieve better accuracy values.

However, SSCF and TSSCF approaches are still based on traditional user-based CF. It uses the fully static listening history of users to perform recommendation and requires very expensive computational time and space with the growth of the number of users and music in a database. Thus, both SSCF and TSSCF approaches are not appropriate for on-line manner because the on-line music service always increases new users and new songs. These two algorithms are faced with the scalability problem. This causes the system to become less predictive ability. In order to overcome this problem, we introduce *Incremental Session based Collaborative Filtering with forgetting mechanisms* or *ISSCF* ,in short; by modifications in SSCF [5]. Our approach is capable to accurately recommend the next songs for the active session by considering past sessions in on-line manner. Because of increasing new users and new songs, our approach uses forgetting mechanisms to handle old and obsolete data, and maintain the MRS concerning to recent data. It is possible to reduce memory usage and processing time as well. In this paper, we evaluate the efficiency of two *forgetting mechanisms* – sliding windows and fading factors. In our experiments, we evaluate the accuracy of our purposed algorithm with the $HitRatio$ (HR@n)[5], [6], and also evaluate the time-consuming of our model by comparing with SSCF. The results are shown that our purposed algorithm outperforms in terms of accuracy and computational time.

II. Background Knowledge

A. Collaborative Filtering Algorithm (CF)

Collaborative filtering (CF) is the well-known personalized recommendation technique that widely used in recommender system. The basic idea of CF is to help users to find the

978-1-4673-7825-3/15/$31.00 © 2015 IEEE

items they would like to purchase based on rating of those items by other users with similar taste. CF produces a prediction score or top-N recommendation list of items for an active user. More formally, there are a set of users $U = \{u_1, u_2, \ldots, u_i, \ldots, u_N\}$ and a set of items $M = \{m_1, m_2, \ldots, m_j, \ldots, m_K\}$. Each user has rated a subset of items such as movies, music, book and etc. All available ratings (r_{u_i,m_j}) are collected in user-item rating matrix denoted R matrix as illustrated in Fig 1. In the first step, it finds similarity between active users u_i and users u_v having co-rated as $M_i \cap M_v$. The popular similarity measures are cosine similarity and Pearson correlation similarity as given in (1) and (2). Next, the k most similar users are selected as the k-nearest neighbors of active user. Then, CF calculates the prediction rating (p_{u_i,m_j}) that active user (u_i) would probably prefer item (m_j) based on his/her neighbors using (3). Finally, the top-N recommendation list is generated based on highest prediction scores [2], [7].

	$\mathbf{m_1}$	...	$\mathbf{m_j}$	...	$\mathbf{m_K}$
$\mathbf{u_1}$	2		3		5
...					
$\mathbf{u_i}$	$\mathbf{r_{u_i,m_1}}$		$\mathbf{r_{u_i,m_j}}$		$\mathbf{r_{u_i,m_K}}$
...					
$\mathbf{u_N}$	5		1		2

Fig. 1. The example of user-song rating matrix

Cosine similarity :

$$sim\left(u_i, u_v\right) = \frac{\sum\limits_{j \in M_i \cap M_v} \left(r_{u_i,m_j}, r_{u_v,m_j}\right)}{\sqrt{\sum\limits_{j \in M_i \cap M_v} \left(r_{u_i,m_j}\right)^2} \sqrt{\sum\limits_{j \in M_i \cap M_v} \left(r_{u_v,m_j}\right)^2}} \tag{1}$$

Pearson correlation similarity :

$$sim\left(u_i, u_v\right) = \frac{\sum\limits_{j \in M_i \cap M_v} \left(r_{u_i,m_j} - \overline{r}_{u_i,\cdot}\right)\left(r_{u_v,m_j} - \overline{r}_{u_v,\cdot}\right)}{\sqrt{\sum\limits_{j \in M_i \cap M_v} \left(r_{u_i,m_j} - \overline{r}_{u_i,\cdot}\right)^2} \sqrt{\sum\limits_{j \in M_i \cap M_v} \left(r_{u_v,m_j} - \overline{r}_{u_v,\cdot}\right)^2}} \tag{2}$$

$$p_{u_i,m_j} = \overline{r}_{u_i,\cdot} + \frac{\sum_{u_v \in U} sim\left(u_i, u_v\right)\left[r_{u_v,m_j} - \overline{r}_{u_v,\cdot}\right]}{\sum_{u_v \in U} \left|sim\left(u_i, u_v\right)\right|} \tag{3}$$

B. Incremental Collaborative Filtering Algorithms

M. Papagelis and et al. [8] presented Incremental Collaborative Filtering (ICF) for handling scalability problem. ICF is based on incremental updates of the user-user similarities. When active user submits a new rating or updates existing rating then similarity between active user and the rest of need to be recalculated in relation to the old similarity values. ICF approach illustrates that it can reduce computation complexity from polynomial time to linear time that gives higher potential than classic CF. C. Miranda and A. M. Jorge [9] proposed the incremental version of item-based CF for binary ratings that regards recommendation approach based on item instead of user. This approach shows that it uses less computational cost and gives predictive accuracy more than user-based CF. In addition, X. Yang and et al. [10] developed scalable item-based collaborative filtering. To deal with scalability problem, incremental update of item-to-item similarity is proposed. In rating prediction process, local link prediction in item similarity graph is used to find implicit neighbor candidates. The experimental results validate that this approach can increase the efficiency in recommendation. Besides, CF should be able to efficiently process data online in order to keep the system up-to-date [11]. J. Vinagre and A. M. Jorge [11] proposed incremental collaborative filtering with forgetting mechanisms approach that maintains recent preference of user. It decreases older importance information with sliding windows and fading factors approaches. The experimental results show that this approach is able to reduce processing time and memory while not significant reducing predictive potentiality of the algorithm.

Although all of these algorithms mentioned before are designed to handle the scalability problem, actually it cannot improve the accuracy of the recommendation system. Because these algorithms consider only user-rating matrix or item-rating matrix and do not take into account behavior of listening to songs or characteristics of songs as the additional information. Moreover, listening to music is continuous *session* and repetitive process [4], [5], [6]. All of those algorithms do not concern about listening behavior. This will lead to increase the accuracy of the recommendation systems.

III. OUR APPROACH

The framework of ISSCF system (our proposed system) is described in Fig. 2.

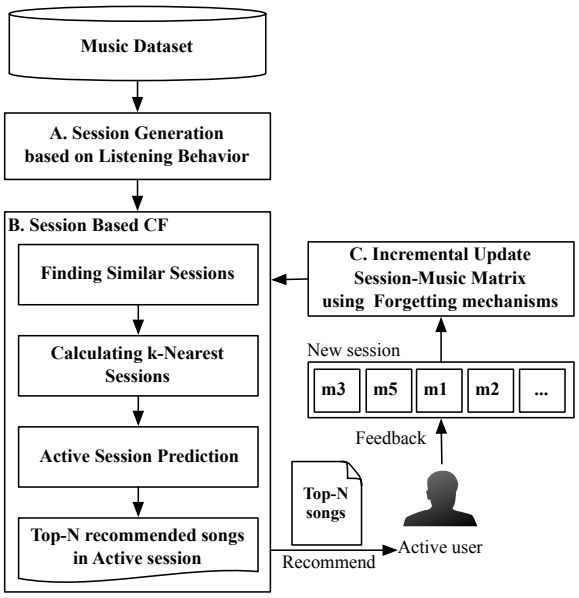


Fig. 2. Overall our approach

A. Session Generation based on Listening Behavior

In this work, we use listening dataset from last.fm[1] database as shown in Fig. 3 that collected during 2005 to May 2009. When a user listens to song, one log in database is generated. Last.fm dataset contains 6,741,330 listening logs obtained from 582 users and 613,117 songs. Each user is analyzed the song diversity by using diversity measuring as shown in (4). This song diversity measures the ratio of different songs played in a session and the total songs played [6]. If song diversity value is closed to 1, it can inform that a user played different songs. Otherwise, this value can inform that a user played repeatedly the same songs. In this dataset, the average of song diversity is 0.21. This means that most of users listened to music repeatedly and continuously.

User id	Timestamp	Artist id	Artist	Music id	Music id
user1	2009-01-01 T09:00:10Z	art1	Don Moy	m1	Moy Or Less
user1	2009-01-01 T09:04:40Z	art1	Don Moy	m2	Contest a Moy
user1	2009-01-01 T09:07:47Z	art3	Pogo	m3	Alice
...	...	...	...	...	...

Fig. 3. The example of listening log data obtained from Last.fm

$$Song\ diversity = \frac{\#Different\ songs}{\#All\ songs} \tag{4}$$

Since our framework is a modification of SSCF (Algorithm 1) [5] designed for on-line handling of on-line music services. First, we define a *session* as the group of songs listened by a user from the moment he/she starts playing songs to the moment they stop it [5]. This work uses continuous time gap of stopping playing songs more than 30 minutes to define as a session. In initial process, we create 100 sessions for a single user to avoid cold start problem as depicted in Fig. 4. Sessions containing less than 2 songs are removed [5], [6].

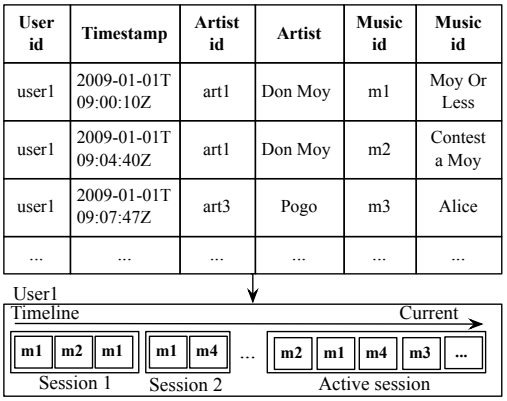


Fig. 4. The example of session generation

[1]http://www.dtic.upf.edu/ ocelma/MusicRecommendationDataset/

Algorithm 1 Traditional SSCF

1: **Input :** D, N, k
2: **Output :** Top N recommended songs
3: Create session-music matrix Met with set of sessions ss ϵ music datasets D
4: Calculate similarity matrix with Met by using cosine similarity
5: ***For*** *each session* ss_u *:*
6: ***For*** *each session* ss_v *:*
7: ***For*** *each song* m_j *by* $j = M_{ss_u} \cap M_{ss_v}$ *:*
8: $sim\,(ss_u, ss_v) = \frac{\sum_j (r_{ss_u,m_j}, r_{ss_v,m_j})}{\sqrt{\sum_j (r_{ss_u,m_j})^2}\sqrt{\sum_j (r_{ss_v,m_j})^2}}$
9: Finding k similar session (S^k)
10: Prediction score of song m_j
11: ***For*** *each song* m_j *with active session as :*
12: $p_{as,m_j} = \overline{r}_{as} + \frac{\sum_{ss_v \in S^k} sim(as, ss_v)\left[r_{ss_v,m_j} - \overline{r}_{ss_v}\right]}{\sum_{ss_v \in S^k} |sim(as, ss_v)|}$
13: Recommend the top-N songs (N_{rec}) based on highest prediction scores

B. Session Based CF

Since this work considers session profiles instead of user profiles for generating recommendation, Session-Music matrix is generated for each user as illustrated in Fig. 5. Based on this data matrix rows represent session profiles, and the columns represent songs. Each cell contains the frequency of songs played in that session. In this process, we concentrate on the prediction of next appropriated songs that user requests in the active session (current session). Thus, our approach ranks all candidate songs and recommend top-n songs that are likely to come after songs that are listening to.

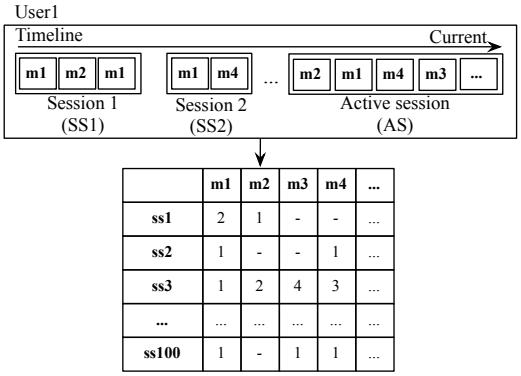


Fig. 5. The example of Session-Music Matrix for a single user

The first step of this process, we find similar sessions by measuring the similarity between an active session (as) and other sessions (ss_v). To measure the similarity between active session (as) and other sessions (ss_v) denoted as $sim\,(as, ss_v)$, we use cosine similarity [5] as shown in (5), where rating (r_{as,m_j}) is the frequency of listening to song (m_j) in active session (as) and rating (r_{ss_v,m_j}) is the frequency of listen to song (m_j) in the session ss_v.

$$sim\,(as, ss_v) = \frac{\sum_j (r_{as,m_j}, r_{ss_v,m_j})}{\sqrt{\sum_j (r_{as,m_j})^2}\sqrt{\sum_j (r_{ss_v,m_j})^2}} \tag{5}$$

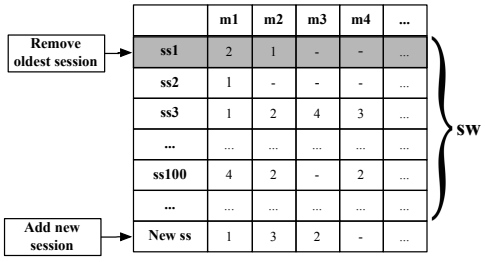
Next, we calculate the k-nearest neighbors is applied to find other sessions which are k most similarity to the active session. Then, the prediction score of song m_j in active session as, defined as (p_{as,m_j}), is determined with k session neighbors (S^k) as shown in (6). Finally, all candidate songs are ranked and recommend top-n songs playing in the active session.

$$p_{as,m_j} = \overline{r}_{as} + \frac{\sum_{ss_v \in S^k} sim\,(as, ss_v)\left[r_{ss_v,m_j} - \overline{r}_{ss_v}\right]}{\sum_{ss_v \in S^k} |sim\,(as, ss_v)|} \quad (6)$$

C. Incremental Update Session using Forgetting Mechanisms

Whenever a user listened to songs from the moment he/she starts playing songs to the moment he/she stops it, and the stopping time of playing songs is more than 30 minutes, a new session emerges as to be feedback from a user. Incremental process has been presented, where our approach incrementally updates Session-Music matrix every time new session is available. In order to maintain recent preference of user, it should be decreased older and obsolete sessions. This work uses forgetting mechanisms to re-update Session-Music matrix. Forgetting mechanisms used in this work are sliding windows and fading factors approaches.

1) Sliding windows: The sliding windows approach is performed using a sequence-based sliding window of size sw that holds information about sw most recent sessions in type of first-in-first-out (FIFO) data structure[11]. In this approach, we process by fixed size of most recent sessions. When an incoming new session is added and sw is reached to fixed window size, then oldest session of the user is discarded as shown in Fig. 6. Then, the similarity value corresponding to songs in new session are updated, while other values are kept. Algorithm ISSCF with sliding windows is shown in Algorithm 2.



	m1	m2	m3	m4	...
ss1	2	1	-	-	...
ss2	1	-	-	-	...
ss3	1	2	4	3	...
...	...	...	...	...	...
ss100	4	2	-	2	...
...	...	...	...	...	...
New ss	1	3	2	-	...

Fig. 6. Sliding windows approach.

2) Fading factors: Since sliding windows provide an effective but abrupt way to forget older data. However, in many cases previous data may contain valuable information and is not necessarily discarded [11]. Fading factors provide another way to gradually forget past data. The fading factors approach is gradual forgetting or full memory approach [12] that uses all sessions of matrix, but decreases importance of old session with small weight as shown in Fig. 7. This approach can be implemented by multiplying the elements of matrix for each session by a factor w using formula (7). Algorithm ISSCF with fading factors is shown in Algorithm 3.

$$w = e^{-\alpha t} \quad (7)$$

Algorithm 2 ISSCF with sliding windows

Input : D, N, k, sw
Output : Top N recommended songs
Initialize session-music matrix Met with set of session S=$\{ss_1, ..., ss_{100}\}$ by $S \in$ music datasets D
For *each new active session* $as \in D$:
    Hidden last song (m_{hide}) of as as test data
    Update matrix Met with as
    If length(Met) $>$ size of window sw ***then***
        Remove oldest session (Met_{ss_1})
    Update session and music (row/column) of Met
    (Re)calculate similarity matrix with Met by using cosine similarity
    For *each session* ss_v *with* as *in* Met *:*
        For *each song* m_j *by* $j = M_{as} \cap M_{ss_v}$ *:*
            Calculate equation (5)
    Finding k similar session (S^k)
    Prediction score of song m_j
    For *each song* m_j *with active session* as *:*
            Calculate equation (6)
    Recommend the top-N songs (N_{rec}) based on highest prediction scores

	m1	m2	m3	m4	...
ss1	2*w*	1*w*	-	-	...
ss2	1*w*	-	-	-	...
ss3	1*w*	2*w*	4*w*	3*w*	...
...	...	...	...	...	...
ss100	4*w*	2*w*	-	2*w*	...
New ss	1	3	2	-	...

Fig. 7. Fading factors approach

where, w : weight value
α : controlling factor to define how fast the weights decrease
t : session order

IV. Experimental Evaluation

A. Experimental setup

In the experimental process, we would like to know that whether our framework could predict the next song to be played in the current active session based on what user has previously listened in that session. This work removes the last songs to be played in the queried current active session as testing datum [6] as shown in Fig 8. One important test is how to deal with a large amount of data and perform in on-line manner. We consider user sessions as a data stream. From the analysis of last.fm data, there are 564 sessions. These sessions are divided into 100 sessions to be used as initial sessions and 464 sessions to be used as the queried active sessions (testing data) that will continuously feed into system. Then, testing data is executed by our framework (ISSCF) and SSCF algorithm [5] to obtain the top-10 recommendations.

The following parameters are set to conduct the tests. For sliding windows, we set window size (sw) in form of the number of sessions at 200 and 400 latest sessions. For fading factors, we set the weight values (alpha) of exponential time

Algorithm 3 ISSCF with fading factors

Input : D, N, k, α
Output : Top N recommended songs
Initialize session-music matrix Met with set of session S=$\{ss_1, ..., ss_{100}\}$ by $S \epsilon$ music datasets D
For *each new active session as* ϵ D
    Hidden last song (m_{hide}) of as as test data
    Update matrix Met with as
    Assigns weight (w) with factor α, session order (t) :
        Calculate equation (7)
    (Re)calculate similarity matrix with Met by using cosine similarity
    For *each session* ss_v *with as in* Met *:*
        For *each song* m_j *by* $j = M_{as} \cap M_{ss_v}$ *:*
            Calculate equation (5)
    Finding k similar session (S^k)
    Prediction score of song m_j
    For *each song* m_j *with active session as :*
        Calculate equation (6)
    Recommend the top-N songs (N_{rec}) based on highest prediction scores

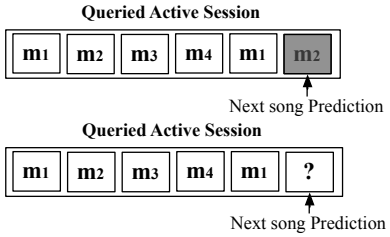


Fig. 8. Evaluation approach in the next song prediction

decay function as controlling how fast the weights decrease at 0.1 and 0.01. The both ISSCF and SSCF approaches use cosine similarity and 30 best neighbor sessions as suggested in [6]. In addition, this work has tested the computational time of ISSCF and SSCF approaches in finding similarity process and prediction process at 10 runs and 200 of the consecutive queried active sessions.

B. Evaluation Metric

In evaluation process, we measure the accuracy of our framework and other methods using $HitRatio$ ($HR@n$) [5]. $HR@n$ indicates that whether the desired songs appear on the top-n recommendation lists for a single user in the queried active session, and how many times they appear [6]. $HR@n$ metric is described in formula (8).

$$HR@n = \frac{\#hit}{k} \tag{8}$$

The average $HR@n$ for one session in all users can be calculated as shown in (9).

$$HitRatio = \frac{\sum_{i=1}^{N} HR@n}{N} \tag{9}$$

Where, hit : if music is listened in the next song appear on recommendation list, so hit is 1 otherwise is 0.

k : the number of hidden songs
N : the number of users

C. Results and Discussions

The result as depicted in Fig 9 shows the $HitRatio$ of ISSCF with sliding window at $sw = 200$ and $sw = 400$ and compared with SSCF for each queried active session of all users. From experimental results, ISSCF with sliding window at $sw = 200$ got the average $HitRatio$ at 0.085 and at $sw = 400$ got the average $HitRatio$ at 0.089. For SSCF approach, the average $HitRatio$ is 0.088. Fig 10 shows the $HitRatio$ ISSCF with fading factors in each queried active session compared with SSCF. The average of $HitRatio$ at $\alpha = 0.01$ is 0.11 and $\alpha = 0.1$ is 0.1146. We have conducted Wilcoxon Tests at $p - value = 0.1$ between ISSCF and SSCF for 50 queried active sessions. We conclude that there was no statistically significant difference between ISSCF with sliding windows and SSCF. Although there was no difference between ISSCF and SSCF, but ISSCF can reduce the computational time that will discuss next. Comparing between ISSCF with fading factors and SSCF, there was a statistically significant. It is clear that ISSCF with fading factors shows a good accuracy than SSCF.

Fig. 11 shows the best result in each algorithms. ISSCF with fading factors at $\alpha = 0.1$ still gives the best average $HitRatio$ than ISSCF with sliding windows and SSCF. We can conclude that ISSCF with fading factor is capable to increase significantly the accuracy of the recommendations.

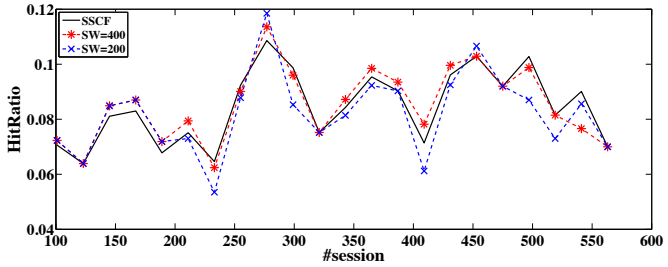


Fig. 9. The $HitRatio$ of ISSCF with sliding windows compared with SSCF

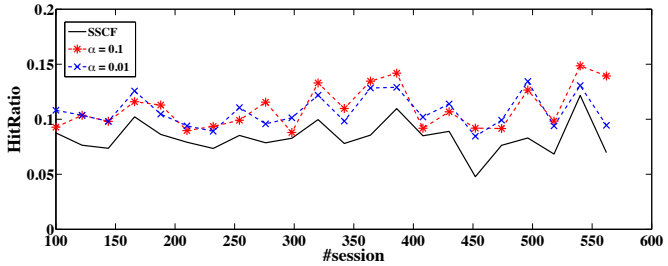


Fig. 10. The $HitRatio$ of ISSCF with fading factors compare with SSCF

This work also evaluated the computational efficient of ISSCF with SSCF. Since SSCF is the algorithm based CF, so it uses the quadratic time, especially, for finding similarity between whole sessions and songs in offline process and uses more time for prediction process. Table I shows the execution time used in those three algorithms at 200 sessions. Time was measured on a PC with CPU speed at 3.20 GHz, 8 GB RAM,

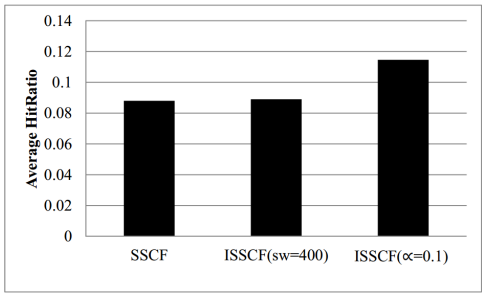


Fig. 11. The average of $HitRatio$ value of SSCF, ISSCF with $sw = 400$, and ISSCF with $\alpha = 0.1$ for all users in all sessions

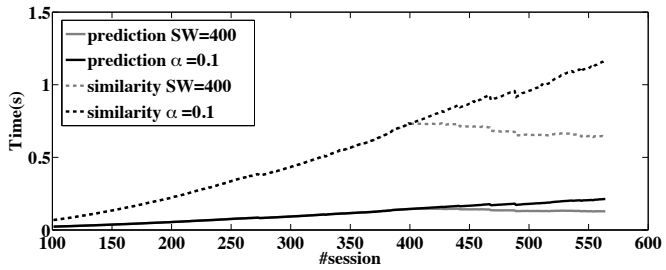


Fig. 12. The execution time of finding session similarity and active session prediction processes of ISSCF

and all algorithms were implemented in Python. The traditional SSCF approach uses $O(m^2n)$ to find similarity (Algorithm 1 lines 5-8) and if we know the session of active user, so time of prediction is $O(n)$ (Algorithm 1 lines 11-12).

For ISSCF approach, it has to find similarity sessions when latest session of active user is updated. This process uses computational time that is $O(mn)$ (Algorithm 2 lines 11-13 and Algorithm 3 lines 10-12), where m is the number of sessions and n is the number of songs in that session. In process of active session prediction, it uses $O(n)$ (Algorithm 2 lines 16-17 and Algorithm 3 lines 15-16). That is appropriate for recommendation systems and helps computational reduction. According to Fig. 12 shows the growth of execution times of finding session similarity process and active session prediction process of ISSCF. The ISSCF with sliding windows presents very efficiency over ISSCF with fading factors approach, hence ISSCF with sliding windows uses the fixed size of windows (sessions). On the other hand, ISSCF with fading factors approach does not discard data. It will affect to computational time that grows with number of sessions in linear time.

TABLE I. EXECUTION TIME OF TWO APPROACHES

Approach	Similarity	Prediciton
Offline SSCF	10.16 sec.	0.02 sec.
ISSCF	**0.09 sec.**	**0.02 sec.**

V. CONCLUSION

Since Session-based Collaborative Filtering (SSCF) requires expensive computations that grow polynomially because of increasing in the number of users and songs, and SSCF is not capable to process data in on-line manner in order to maintain the system up-to-date. Therefore, this paper proposes Incremental Session based Collaborative Filtering with Forgetting Mechanism called ISSCF. It is a new framework that modified in tradition SSCF incorporated with forgetting mechanism. Our approach is suitable for the music domain by taking into account the users listen to songs continuously and repetition in a session. Thus, our purpose is to improve the accurately recommendation for the next songs in active session and to overcome the scalability problem in on-line manner. For supporting on-line process, we adapt incremental algorithm to SSCF by using forgetting mechanisms — sliding windows and fading factors — to deal with old and obsolete data. In our experiments, we evaluate the accuracy of our purposed algorithm compared with SSCF by using $HitRatio$ (HR@n) metric [5], [6], and also evaluate the time-consuming of our model comparing with SSCF. The results are shown that our purposed framework outperforms in terms of accuracy and computational time.

ACKNOWLEDGEMENTS

This research is supported by Faculty of Informatics, Burapha University.

REFERENCES

[1] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *ACM Comput. Surv.*, vol. 47, no. 1, pp. 3:1–3:45, May 2014.

[2] M. Kaminskas and F. Ricci, "Contextual music information retrieval and recommendation: State of the art and challenges," *Computer Science Review*, vol. 6, no. 23, pp. 89 – 119, 2012.

[3] Y. Song, S. Dixon, and M. Pearce, "A survey of music recommendation systems and future perspectives," pp. 395–410, Jul. 2012.

[4] C. H. Park and M. Kahng, "Temporal dynamics in music listening behavior: A case study of online music service," in *Computer and Information Science (ICIS), 2010 IEEE/ACIS 9th International Conference on*, Aug 2010, pp. 573–578.

[5] S. E. Park, S. Lee, and S. goo Lee, "Session-based collaborative filtering for predicting the next song," in *Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference on*, 2011, pp. 353–358.

[6] R. Dias and M. Fonseca, "Improving music recommendation in session-based collaborative filtering by using temporal context," in *Tools with Artificial Intelligence (ICTAI), 2013 IEEE 25th International Conference on*, Nov 2013, pp. 783–788.

[7] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, *Recommender Systems Handbook*, 1st ed., New York, NY, USA, 2010.

[8] M. Papagelis, I. Rousidis, D. Plexousakis, and E. Theoharopoulos, "Incremental collaborative filtering for highly-scalable recommendation algorithms," in *Proceedings of the 15th International Conference on Foundations of Intelligent Systems*, ser. ISMIS'05, 2005, pp. 553–561.

[9] C. Miranda and A. Jorge, "Incremental collaborative filtering for binary ratings," in *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on*, vol. 1, Dec 2008, pp. 389–392.

[10] X. Yang, Z. Zhang, and K. Wang, "Scalable collaborative filtering using incremental update and local link prediction," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, ser. CIKM '12, 2012, pp. 2371–2374.

[11] J. Vinagre and A. Jorge, "Forgetting mechanisms for scalable collaborative filtering," *Journal of the Brazilian Computer Society*, vol. 18, no. 4, pp. 271–282, 2012.

[12] J. a. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 44:1–44:37, 2014.

ประวัติย่อผู้วิจัย

ชื่อ-สกุล สุเมธ ดาราพิสุทธิ์
วันเดือนปีเกิด 9 พฤศจิกายน 2533
สถานที่เกิด จังหวัดชลบุรี
ที่อยู่ปัจจุบัน 111/15 หมู่ 4 ต.สุรศักดิ์ อ.ศรีราชา จ.ชลบุรี
ประวัติการศึกษา

พ.ศ. 2545 ประถมศึกษาปีที่ 1-6 โรงเรียนดวงมณี จังหวัดชลบุรี
พ.ศ. 2551 มัธยมศึกษาปีที่ 1-6 โรงเรียนศรีราชา จังหวัดชลบุรี
พ.ศ. 2555 วิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
คณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา จังหวัดชลบุรี

ผลงานตีพิมพ์

2015

U. Suksawatchon, S. Darapisut and J. Suksawatchon, "Incremental session based collaborative filtering with forgetting mechanisms," 2015 International Computer Science and Engineering Conference (ICSEC), Chiang Mai, 2015, pp. 1-6.

S. Darapisut, U. Suksawatchon and J. Suksawatchon, "The constant time of predictive algorithm for music recommendation with time context," *12th International Joint Conference on Computer Science and Software Engineering (JCSSE),* Songkhla, 2015, pp. 63-68.

2014

S. Darapisut and J. Suksawatchon, "Comparison of the constant prediction time of collaborative filtering algorithms by using time contexts," 11th International Joint Conference on Computer Science and Software Engineering (JCSSE), Chon Buri, 2014, pp. 302-306

สุเมธ ดาราพิสุทธิ์ และ จักริน สุขสวัสดิ์ชน, “ขั้นตอนวิธีการทำนายค่าคะแนนความชอบที่มีประสิทธิภาพสำหรับระบบแนะนำเพลงโดยใช้วิธีการเทนเดนซีเบสร่วมกับข้อมูลไมโครโพรไฟล์” The Tenth National Conference on Computing and Information Technology (NCCIT), ภูเก็ต 2014, หน้า 755-761