Alexander Oberegger, BSc

On the Impact of Weather on the Users' Check-in Behaviour

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Software Development and Business Management

submitted to

Graz University of Technology

Supervisor

Dipl.-Ing. Dr.techn. Christoph Trattner, BSc

Knowledge Technologies Institute (KTI)

Graz, May 2016

The thesis was written using Arkaitz Zubiaga's
skeleton, which can be downloaded from
http://www.zubiaga.org/thesis/
Many thanks Arkaitz for letting me use this
template for my master thesis.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis dissertation.

Date Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch. [1]

Datum Unterschrift

[1] Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

Abstract

Point of interest (POI) recommender systems for location-based social network services, such as Foursquare or Yelp, have gained tremendous popularity in the past few years. Much work has been dedicated into improving recommendation services in such systems by integrating different features that are assumed to have an impact on people's preferences for POIs, such as time and geolocation. Yet, little attention has been paid to the impact of weather on the users' final decision to visit a recommended POI.

This thesis contributes to this area of research by presenting the results of a study that aims to predict the POIs that users will visit based on weather data. First, an empirical study on a dataset of a location based social network was conducted to gain information about the users' mobility behaviours in contrasting weather contexts. This knowledge then serves as a basis for extending the state-of-the-art Rank-GeoFM POI recommender algorithm with weather-related features, such as temperature, cloud cover, humidity or precipitation intensity. The method, named WPOI, not only significantly increases the recommender accuracy in comparison to the original algorithm, but also outperforms its time-based variant Rank-GeoFM-T. Furthermore, the magnitude of impact of each feature on the recommendation quality is shown, revealing the need to study the weather context in more detail in the light of POI recommendation systems.

Kurzfassung

Empfehlungsdienste für „Points of Interest“ (POI) in standortbezogenen sozialen Netzwerken (SSN) wie Foursquare oder Yelp haben in den letzten Jahren an Popularität gewonnen. Viel Forschungsarbeit wurde deshalb in die Entwicklung von Empfehlungsdiensten gesteckt, welche die Bedürfnisse der Benutzer erkennen und dementsprechend ihre Empfehlungen für POIs individuell anpassen. Die Verwirklichung dieser Verbesserungen wurde vor allem mithilfe von kontextueller Information erreicht, von der angenommen wird, dass sie einen Einfluss auf die Entscheidungen der Benutzer, einen Ort zu besuchen oder nicht, hätte. Während noch wenig Aufmerksamkeit auf den Einfluß von Wetter auf diese Entscheidungen gelegt wurde, sind andere Charakteristika wie Zeit, soziale Interaktion und geographische Position von POIs schon erfolgreich in Empfehlungsdiensten integriert worden.

Diese Masterarbeit konzentriert sich auf die Erforschung vom Einfluss des Faktors Wetter auf das Check-in Verhalten von BenutzerInnen in SSNs. Im Zuge dieser Arbeit wird ein Empfehlungsdienst entwickelt, der sich neben der vergangenen Interaktion zwischen BenutzerInnen und POIs auch der Information des Faktors Wetter bedient, um zukünftige Besuche von POIs vorherzusagen. Um das Mobilitätsverhalten von Menschen genauer zu untersuchen, wurde eine empirische Studie am Datensatz eines SSN durchgeführt. Diese Studie diente anschließend als Informationsbasis um den bestehenden Empfehlungsdienst Rank-GeoFM, der sich auf dem letzten Stand der Forschung befindet, zu erweitern und mit der Information von Wetterfaktoren wie Temperatur, Bewölkung, Luftfeuchtigkeit oder Regenintensität zu bereichern. Der daraus resultierende Empfehlungsdienst namens WPOI erhöht die Vorhersagegenauigkeit nicht nur im Vergleich zum Basisalgorithmus Rank-GeoFM signifikant, sondern übertrifft auch

die zeitbasierte Variante Rank-GeoFM-T. Außerdem wird die Kardinalität der einzelnen Wetterfaktoren auf die Vorhersagegenauigkeit geprüft. Diese Arbeit zeigt schlussendlich auch den Bedarf an weiterer Forschung am Einfluss von Wetterfaktoren in POI Empfehlungsdiensten auf.

Danksagung

An dieser Stelle möchte ich mich bei meinem Betreuer Dr. Christoph Trattner für die ausgezeichnete Zusammenarbeit und die unzähligen Ratschläge und Herausforderungen bedanken, die ich von ihm während der Erstellung der Arbeit erhalten habe.

Großer Dank gilt meiner gesamten Familie allen voran meinen Eltern Alois und Renate Oberegger, auf deren Unterstützung ich immer zählen konnte und die mir dieses Studium erst ermöglicht haben. Außerdem möchte ich mich bei meiner Schwester Astrid bedanken, die mich vor allem zu Beginn des Studiums mit dem nötigen Studenten Know-how ausgestattet hat.

Ein großes Dankeschön geht an die Frau an meiner Seite Katrin, auf deren moralische Unterstützung ich immer bauen konnte, die sehr geduldig war, wenn es einmal länger gedauert hat, die aber auch das „Büro“ einmal zu sperrte, damit ich loslassen konnte.

Schlussendlich möchte ich mich auch bei meinen Freunden bedanken, auf die ich mich in zahlreichen Projekten immer verlassen konnte, die aber auch nach der Deadline nicht gleich nach Hause gingen.

Table of Content

1 Introduction

The problem addressed in this thesis is point of interest (POI) recommendation. The question is how accurate the next visitation of a user can be predicted knowing the current context and the historical check-in data of the user. The focus of this thesis lies on the weather context that consists of several weather features.

1.1 Problem Statement

Having the current weather context and a users' historical data, definition 1 explains the problem studied in this thesis.

Definition 1 *Given a user u, her check-in history L^u, i.e., the POIs that she has visited in the past, and the current weather context c the aim is to predict the POIs $\hat{L}^u = \{l_1, \ldots, l_{|L|}\}$ that she will likely visit in the future that are not in L^u.*

According to that definition Figure 1.1 illustrates the data flow of a POI recommender system that uses weather contextual information.

1.2 Motivation

Location-based social networks (LBSN) enable users to check-in and share places and relevant content, such as photos, tips and comments that help other users in exploring novel and interesting places they might not have been before. For instance the US company Foursquare[1] offers an equally named LBSN with millions of subscribers doing millions of check-ins everyday all over the world. This vast amount of check-in data, publicly available through Foursquare's application programming interface (API), has recently inspired many researchers to investigate human mobility patterns and behaviours with the aim of assisting users, by

[1] `https://foursquare.com/`

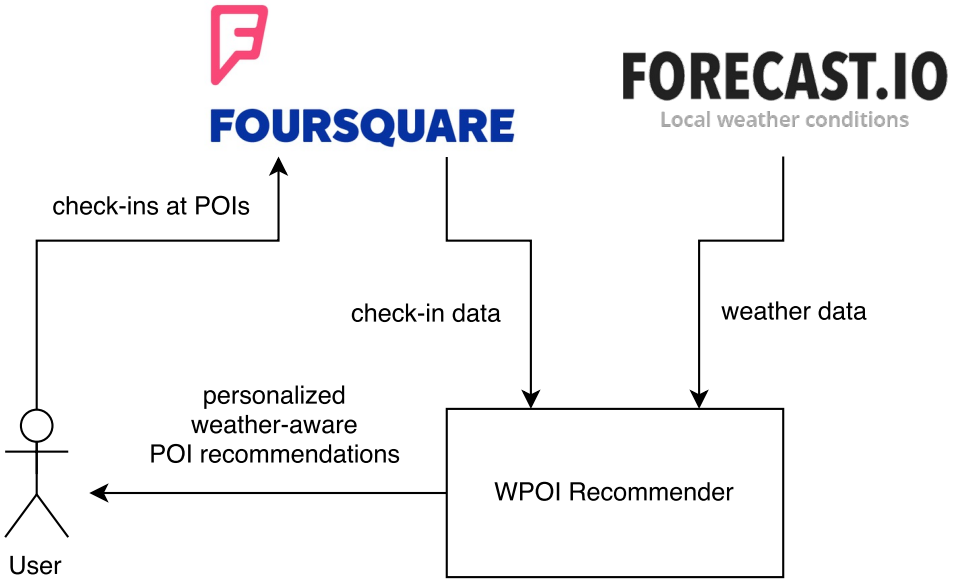


Figure 1.1: Flowchart illustrating the information flow in a POI recommender that uses weather contextual information.

means of personalized recommendation services, in exploring their surroundings more efficiently (see Ye et al. (2011a); Yin et al. (2013)).

Most of the existing approaches on POI recommendation exploit three main factors (or contexts) of the data, namely, social, time and geolocation(see Li et al. (2015); Cheng et al. (2012); Ye et al. (2011a)). While these approaches work reasonably well, little attention has been paid to weather, a factor, already mentioned in Bao et al. (2014), that may potentially have a major impact on users' decision about visiting a POI or not. For example, if it is raining in a certain period of time and place, the recommender should prefer suggesting indoor POIs. On the other hand if the temperature is high the recommender system should prefer venues for warm temperatures, such as "Beach", while recommending venues like "Ski Area" when the temperature is low.

To contribute to this area of research this thesis presents the results of a recommender system that not only takes past preferences of a user into consideration, but also the current weather context, when the recommendation is created. The work in this thesis is based on a given Foursquare dataset (see Yang et al. (2015a,b)) that contains more than 33 million check-ins from 415 cities in 77 countries. However, before dealing with the problem on such a large scale, it was necessary to concentrate the data analysis on a small set of 60 US cities and the recommendation system on selected four cities that could represent the variety of climate in the dataset. Appropriate weather data was then obtained from the

forecast.io Weather API [2]. This API archives weather data for many places up to 60 years in the past. Having check-ins and the according weather contexts the aim was then to investigate the impact of the weather contexts on the users' check-in behaviours using empirical analysis methods.

Using dataset knowledge obtained from the empirical analysis it was possible to enrich a state-of-the-art recommender named "Rank-GeoFM", a factorization based method that learns the factors with Stochastic Gradient Descent (SGD) (see Li et al. (2015)), with weather context information in order to enhance the recommendation performance. Among others available, the eight most important weather features e.g., temperature, windspeed, cloud cover, were chosen to be incorporated into the recommender system.

After including weather context in the algorithm the experiments to measure the enhancement was then made through measuring the recommendation performance on each <*user, venue, weather context*> triple in a subset of the data. This thesis incorporated each of the eight weather features separately into the recommender in order to examine capability of each of them.

1.3 Research Questions

To drive the research of this thesis the following four reserach questions were defined:

Research Question 1

> To what extent are the users' mobility patterns influenced by weather features, namely, cloud cover, visibility, moonphase, precipitation intensity, pressure, temperature, humidity and wind speed? (Results: Chapter 4)

Research Question 2

> How can weather context information be incorporated into existent recommender systems? (Results: Section 5.1)

Research Question 3

> To what extent can we use weather information to increase the current state-of-the art in POI recommender systems? (Results: Section 5.3)

Research Question 4

> Which weather feature provides the highest impact on the recommendation accuracy? (Results: Section 5.3)

[2] `https://developer.forecast.io/`

1.4 Contributions

As far as known this is the first research that investigates the extent to which weather has an impact on users' mobility behaviours and how to use this information in the context of POI recommender systems. The main contributions of this thesis can be summarized as follows:

- Collection of appropriate weather data to a dataset containing user check-ins to POIs.
- An analysis of the dataset with empirical methods in order to find the impact of weather onto the users' decision of checking-in at a POI or not to find coherencies between weather features.
- Incorporation of weather context information into a state-of-the-art recommender system that uses geographical data and time as contextual information and measure whether there is a performance increase or not.

Parts of this master's thesis were submited for publication to the ACM RECSYS 2016 conference.
C. Trattner, A. Oberegger, L. Eberhard, D. Parra and L. Marinho. WPOI: A Weather-Aware POI Recommender System. In Proceedings of the 10th ACM Conference on Recommender Systems (RECSYS 2016), Boston, MA, USA, 2016.

1.5 Thesis Outline

This thesis is structured as follows: Chapter 2 gives an overview of the current work in the area of POI recommender systems. Afterwards Chapter 3 contains a description of the datasets used in this thesis followed by Chapter 4 providing detailed information about the empirical analysis conducted to get an insight into the dataset. The incorporation and the results of the weather context incorporation into a state-of-the-art recommender is illustrated in Chapter 5. A conclusion of the work and a prospect to further research is finally given in Chapter 6.

2

Related Work

This Chapter provides an overview of related and relevant work in the area of POI recommender systems and context aware recommender systems (CARS). Nowadays there are a lot of social applications storing ratings about items. The term item may refer to venues, products, movies and many more. The main challenge for the users is to decide which items to consume out of the huge amount of items provided, making it impossible to find the best item in a proper amount of time. Therefore, the task in this field is to provide the user with an automatical system that supports her in deciding this arbitrations. Such a system can be defined as follows:

"Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user. The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read. "Item" is the general term used to denote what the system recommends to users." (Ricci et al., 2011)

2.1 Location-Based Social Networks (LBSN)

The basis of a POI recommender system is mostly build upon the data of a LBSN. A LBSN is the fusion of a social network and POIs visited by individuals of this social network. An LBSN can be defined as follows:

"A location-based social network (LBSN) does not only mean adding a location to an existing social network so that people in the social structure can share location-embedded information, but also consists of the new social structure made up of individuals con-

nected by the interdependency derived from their locations in the physical world as well as their location-tagged media content, such as photos, video, and text. Here, the physical location consists of the instant location of an individual at a given timestamp and the location history that an individual has accumulated in a certain period. Further, the interdependency includes not only that two persons co-occur in the same physical location or share similar location histories but also the knowledge, e.g., common interests, behaviors, and activities, inferred from an individual's location (history) and location-tagged data." (Zheng and Zhou, 2011)

That means that the venues are not integrated into an existing social structure, but attending common POIs or having similar mobility behaviours, creates a social structure on its own, when perceiving the common visited venues as a social link between users. Furthermore, Figure 2.1 shows the three possibilities of creating a social network graph out of a LBSN (Zheng and Zhou, 2011):

- **Location-location graph.** Nodes are locations and an edge between two nodes indicates a connection between these two venues. This connection results from a common attribute of this two, such as they belong to the same category or having a similar user history.
- **Location-location graph.** Nodes are users and an edge between two users reveals a connection between two users resulting from common visited venues.
- **User-location graph.** A bipartite graph having both, users and venues as nodes and edges between nodes, for instance, indicates that a user visited a particular venue.

Based on these three variants of graphs connections between the two entities user and location of a LBSN can be identified and link analysis based algorithms can be applied to it, as shown in Section 2.2.

2.2 POI Recommendation with Conventional Recommender Systems

In the recent years a lot of work was done in the field of POI recommender systems that provides a variety of methods giving recommendations of POIs to people. In the very beginnings of POI recommendation conventional recommender algorithms such as, collaborative filtering (CF) or Link analysis-based Recommender were taken and POIs were simply considered as items. In other words the special case of POI recommendation was tried to solve with general recommendation approaches.

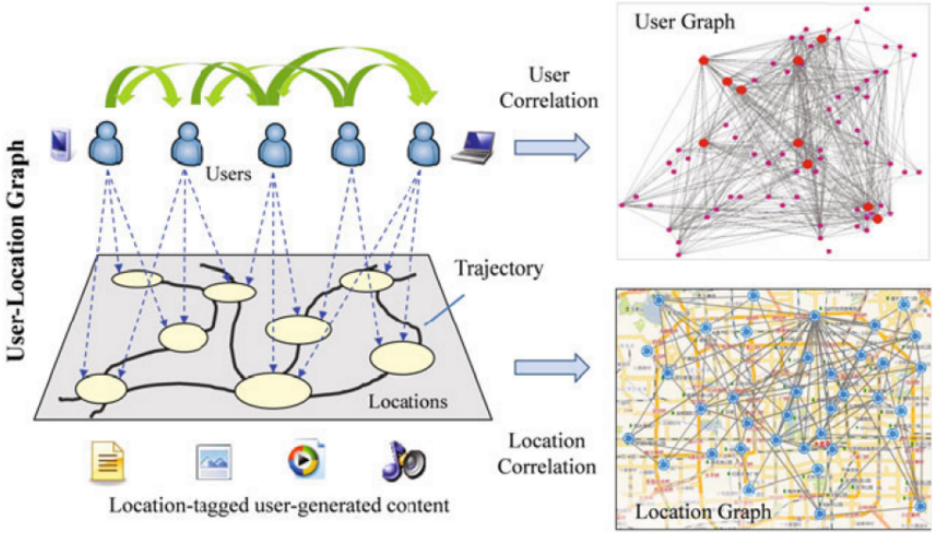


Figure 2.1: Three different ways of illustrating LBSNs as a graph. (Zheng and Zhou, 2011)

Collaborative Filtering (CF)

A basis for recommender systems in common was build by the work of Breese et al. (1998) with their analysis of user based CF approaches where users are described as vectors, containing their check-in history. Based on the similarity of two user vectors, locations are recommended. Additionally, to the user based CF approaches, the work of Sarwar et al. (2001b) and Linden et al. (2003) build the basis for an item to item CF approach calculating similarities over items instead of users. The work of Noulas et al. (2012) then has shown that user based CF works better in LBSNs than the item based version.

However, it has shown that conventional CF approaches are not very suitable for POI recommendations since the CF model suffers from data sparsity and the cold start problem (recommendations for new users and new items). Furthermore, it is very computational intensive with the growing amount of users and items in LBSNs (Bao et al., 2014).

Link Analysis-based Recommendations

As mentioned in Section 2.1, LBSNs can be represented as graph and therefore algorithms based on link analysis like the PageRank (Page et al., 1999) or Hypertext Induced Topic Search (HITS) (Chakrabarti et al., 1998; Kleinberg, 1999) were adjusted to fit the needs of POI recommendations. Zheng et al. (2009) present an adoption of the HITS algorithm in order to be able to recommend POIs. First,

they created a user-location graph as described in Section 2.1 perceiving a visit of a person at a venue as a connection between them. Following the HITS approach they declare users visiting many venues as hubs and venues visited by many users as authorities. Based on this declaration the top-n venues are the n biggest authorities and the top k interesting users are the k biggest hubs. Since the travel experience of each user is regional related, they created geographical regions whereby each user has a different hub size in each of these regions. The resistance against the cold start problem and the incorporation of the users' experience into the algorithm implies a good applicability for POI recommendations but the lack of personalization possibilities connotes a loss of accuracy.

2.3 Purely Context-based POI Recommender Systems

Besides the historical check-in information of a user, most of the time additional context information is available. In order to find the impact of this contextual information on the prediction accuracy some approaches purely rest upon this information and not on the users' historical preferences.

Park et al. (2007) pursue an approach solely based on the data obtained from a user's meta data. Background data, such as age, gender, economical power and education were collected and a Bayesian Network model was created out of that information. Based on this model restaurant recommendations were given.

Balby Marinho et al. (2015) designed an approach based on the assumption that people tend to check-in at places that are close to already visited places. Just this simple approach already showed very strong accuracy compared to the most popular algorithm (MP) and revealed the entropy of geographical information.

Moreover, Ramaswamy et al. (2009) built a concept that tried to examine the amount of influence that user u has on user u' and to use that information as a similarity measure between users. This influence was extracted from call data records where influence is computed on the basis of the call duration. On the other hand social affinity was utilized from the users address book.

However, as mentioned in Bao et al. (2014), content-based recommendations do not consider the users' and communities preference score on the locations. Furthermore, the combination of location data and content data is very computational intense.

2.4 Embedding Contextual Information

Incorporating contextual information to recommender systems enforces an extension of conventional recommender algorithms. Usually such an algorithm is based on a <*user,item,rating*> triple what from the preference relations between

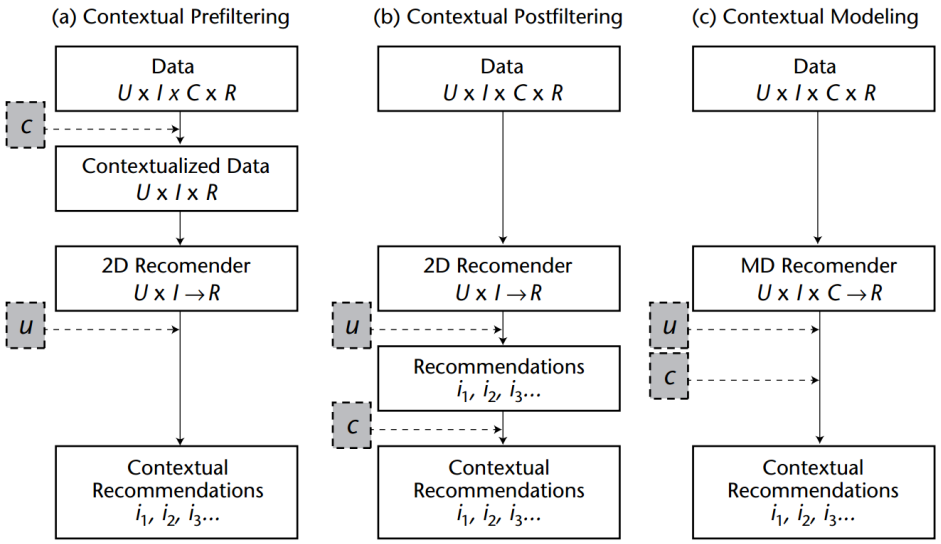


Figure 2.2: Three different ways of incorporating contextual information. (Adomavicius and Tuzhilin, 2011)

users and items are created. Contextual information extends this triple to <*user, item, rating, context*>. The challenge now is to apply this additional information to existing recommender algorithms to take advantage out of it. As shown in Figure 2.2, Adomavicius and Tuzhilin (2011) propose three different prospects to handle this task, namely "Contextual pre-filtering", "Contextual post-filtering" and "Contextual modeling".

Contextual pre-filtering

Contextual pre-filtering applies the contextual information to the dataset before building the recommender model. That approach can be described as a pre-selection or filtering task of the underlying data. After the pre-filtering has been applied the resulting data is reduced to a conventional <*user, item, rating*> triple that can be used in any existing recommender approach. Applying the exact context to the pre-filtering, such as "monday 10 o'clock", leads to a pre-filtering that leaves out all events not been attended at that point in time. That might lead to data sparsity on the one hand (not that many events that have been attended at that time slot) and to an overfitting of the context on the other hand (maybe 11 o'clock is also interesting). Therefore, the context has to be generalized to for example "weekdays in the morning" to bypass this side effects. Finding the right context granularity is the biggest problem when applying pre-filtering whereby

Panniello et al. (2012) stated a decrease of accuracy when increasing granularity too much. Another leverage point is to determine granularity depending on the category that is to recommend:

"A user's current location plays a vital role in generating recommendations in LBSNs due to the following three reasons. First, a user's current location can be represented on different levels of granularity (the hierarchical property of locations). Choosing a proper granularity for the recommendation scenario is important and challenging. For instance, we should use a fine granularity when recommending restaurants to a user, while a relatively coarse granularity (like in a city or state) for local news recommendations." (Bao et al., 2014)

Contextual post-filtering

In contrast to pre-filtering, post-filtering receives a list of recommendations from a conventional recommender system and tries to adjust this list according to the contextual information. There are two possibilities for post-filtering:

- Remove recommendations that are irrelevant in the context
- Resort the recommendations according to the context

If again the current time context would be "monday 10 o'clock" the post-filtering approach would either remove events from the recommendation list that are sparsely attended on weekdays or resort the list preferring events that are preferentially visited on weekdays. Again this approach contains the task of choosing the correct context granularity and has the advantage that it can be applied to any existing recommender algorithm.

Contextual Modeling

Contextual modeling is an approach that leads to an extension of conventional recommender systems that work with the relations between users and items. The contextual information is incorporated directly into the prediction model of the recommender algorithm. In other words the parameters of any traditional recommender system is extended and leads to the following prediction function:

$$Prediction = P(User, Item, Context) \tag{2.1}$$

The recommender system then tries to use the additional context information during the training of the latent parameters of the prediction model. As shown in Section 2.5 most of the existing approaches incorporated contextual information into the model very successfully.

2.5 Contextual Modeling in POI Recommender Systems

As described in Lian et al. (2014) one of the most challenging aspects in this field is that POI check-in dfatasets do not give explicit rating feedback, such as ratings to movies do. Instead feedback is given implicit through a check-in to a venue which is considered as positive feedback. Since most of the people check-in just at a few POIs available and a check-in does not automatically come along with liking the venue, feedback data is very scarce and elusive. To bypass the problem of data scarcity the idea is to rely not only on historical check-in data but also on social-based and in general on contextual data:

"The problem setting that we have examined captures a characteristic that all recommender systems will soon face: the abundance of social-based and contextual data beyond explicit ratings that is available to improve users' recommendations." (Noulas et al., 2012)

The challenge in recommending venues is therefore not only a matter of the interaction between users and venues but also a matter of social and contextual data behind the feedback that is directly accessible from check-in data. Therefore the focus of recent research laid on incorporating this additional data sources into conventional recommender systems. That incorporation is based on technical progress that offers more computational power nowadays. This enhancement then removes the drawback of content-based recommender systems mentioned in Section 2.3. The contextual features described in the following paragraphs were already successfully incorporated into conventional recommender models.

Social Affiliation. Figure 2.3 illustrates the strong affiliation between the social relationships of a user and the choice of her next check-in revealing that friendship leads to a correlated mobility behaviour. Ma et al. (2009) incorporated this social affiliation into a conventional matrix factorization (MF) approach and showed up with highly accurate results. Similar studies of Cheng et al. (2012), have demonstrated similar friendship effects. Macedo et al. (2015) on the other hand probed a social group affiliation model introducing a group frequency model that adds a group entity to the two classical user and item entities. The expectation in this approach is that if a user frequently attends an event belonging to a specific social group it is more likely that the user also will do that in the future.

Time Aware. Macedo et al. (2015) examined the influence of the time context. The assumption of the authors was that one user prefers events occurring at night

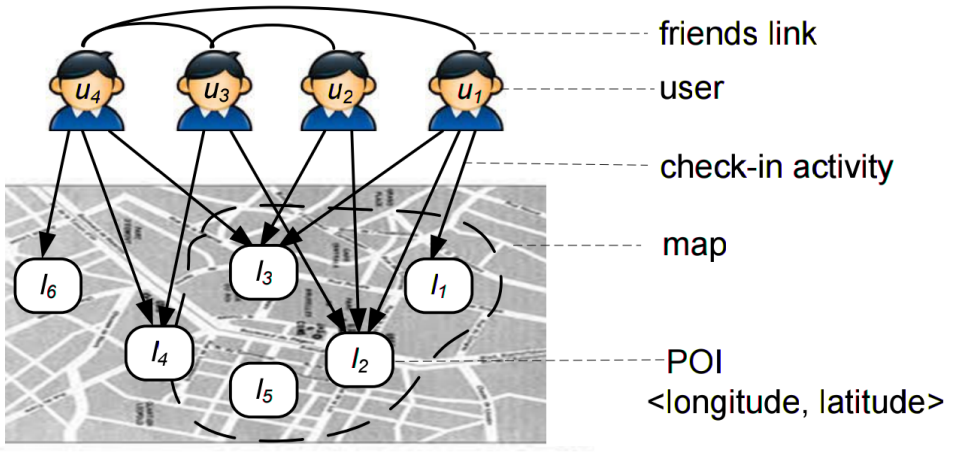
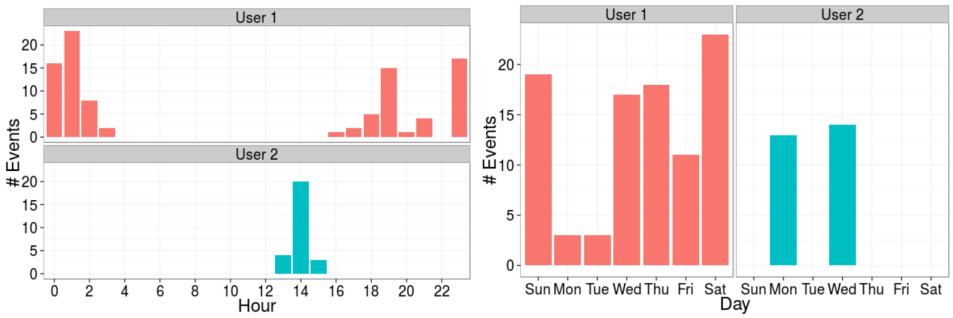


Figure 2.3: Illustration of the strong relationship between social relation and the according locations chosen. (Ye et al., 2011b)



(a) Hourly check-in pattern **(b)** Daily check-in pattern

Figure 2.4: Illustration of the impact of the current Time on the users' check-in pattern. (Macedo et al., 2015)

and a different one prefers to participate at events taking place during the day. This patterns were first examined in an empirical analysis as shown in Figure 2.4 showing the hourly check-in patterns of two users in Figure 2.4a and the daily check-in patterns of two users in Figure 2.4b. Afterwards, this time awareness was incorporated into a recommender system in order to enhance recommendation accuracy.

Geographical Influence. Naturally, the geographical influence factor plays a big role in LBSNs. Tobler (1970) and Macedo et al. (2015) detected a user specific geographical clustering phenomenon. While one type of users tend to attend at events close to their home, others have a widely scattered mobility pattern.

Macedo et al. (2015) utilize this behaviour in their work by introducing a likelihood function that provides the likelihood for each user that she attends a new event by cumulating the likelihood that this event takes place in any areas the user has attended events in the past.

Another approach incorporating the geographical clustering phenomenon was proposed from Lian et al. (2014). Defining geographical influence as a user's activity areas, the area where a user may show up, and the POI's influence areas, the area from which a POI is likely to be visited from, it was possible to incorporate geographical information into a weighted matrix factorization approach (WMF). If both, the user's activity area and the POI's influence area intersect, a recommendation is useful and the higher the intersection the higher the recommendation score.

Fused Matrix Factorization. Finally, the most important step was to fuse several contextual influence factors into one recommender system. Cheng et al. (2012) propose a matrix factorization (MF) method that performs that synthesis. They use a Multi-center Gaussian Model (MGM) to model the concentration of check-ins on several centers, and use the inverse proportionality of the venues' probability being visited to the venues' distance of the user's last check-in. In other words the lower the distance of a venue to the user's last check-in, the higher the check-in probability of that place. Secondly, they introduced probabilistic MF with social regularization (PMFSR) on the basis of friendship and similarity between a user i and his friend f. Finally, they fused the probabilities obtained from the MGM model together with the probabilities obtained from the PMFSR model to achieve a model benefiting both, from the social influence factor and the geographical influence factor.

As already mentioned, similar work was done by Macedo et al. (2015) with their Multi-Contextual Learning to Rank Events approach (MCLRE). MCLRE incorporates the social influence from group memberships, the geographical influence from the user's mobility pattern and temporal influence from the user's temporal check-in preferences into a learning to rank approach using coordinate Ascent (Metzler and Bruce Croft, 2007) for learning the objective function.

3 Datasets

This Chapter describes the data sources used for the empirical analysis in this thesis. A dataset of a LBSN was needed that provides the accurate time of a user's check-in in order to combine the check-in information with weather related features. Yelp[1], for example, is with more than 90 million ratings besides others one of the biggest LBSN's and would therefore match the size constraint. Unfortunately it does not provide accurate time information since a user is able to rate a location before, during or after he or she visited a location. That matter makes it impossible to establish a connection between a weather situation and a venue. In addition to Yelp, Foursquare[2] is with more than 50 million users and more than 8 billion check-ins a settled LBSN that offers accurate time information to the check-ins. Therefore it is in contrast to Yelp possible to associate the check-ins with the prevailing weather situation when the user checked-in at a venue.

3.1 Foursquare Dataset

Foursquare is basically an application that recommends venues also known as POIs close to your current location. A typical use case for this service would be a person that is a tourist in a foreign city and is searching for a good restaurant to take the dinner. As shown in Figure 3.1 the user starts the mobile application and searches for food in the area around Graz and gets a recommendation for "Der Steirer" and "Freiblick By Eckstein" which are restaurants that got good ratings from previous users. Additionally, to the place recommender Foursquare operated as a notifier for locations. Always when a user entered a venue he or she was able to check-in to that location to announce to the social network that

[1] `http://www.yelp.com/about`, last access May 11, 2016

[2] `https://foursquare.com/about`, last access May 11, 2016

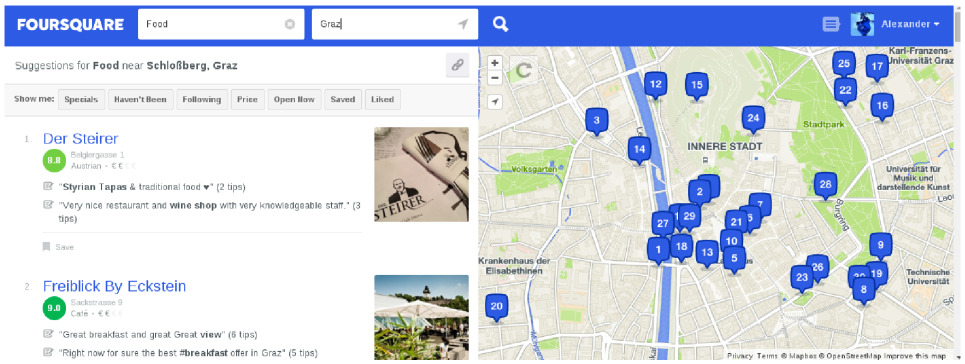


Figure 3.1: A user searches for a dinner place in Graz and finds e.g. "Der Steirer" and "Freiblick By Eckstein"

she is here. The most checked-in person at a specific venue then was the mayor of that location. This simple game mechanic led to a growing community for 4-6 years as founder Dennis Crowley said:

"I think our intent back then was, "hey, these game mechanics will be interesting for 4-6 weeks for people, and then we'll retire them". And it turns out they were interesting for, like, 4-6 years instead of months or weeks,"[3]

After that period, Foursquare was stepping towards a location recommender and therefore the company decided in 2014 to split Foursquare into the recommender application "Foursquare" and the application with the check-in mechanic called "Swarm". As shown in Figure 3.2 the user has the possibility to check-in at the location to let his friends know that she is in the neighborhood. Foursquare's goal with Swarm is to keep check-ins simple while the main application is able to concentrate on location recommendations.

However, it showed that Foursquare check-ins build a terrific base if accurate time check-in data is needed. Therefore the authors of Yang et al. (2015a,b) crawled check-in data from Foursquare before the partitioning of the apps into Swarm and Foursquare. Since check-in information of a user from the Foursquare API is reserved for friends of the user, Yang et al. (2015a,b) decided to crawl Foursquare tagged tweets from the API of twitter[4]. From April 2012 to September 2013 they crawled a global scale dataset with more than 33 million check-ins. Additional to the crawling process they took some noise filtering steps to remove sudden-moves (pairs of check-ins that require travelling speed of more than 1,200 km/h), unresolvable categories and check-ins from inactive users from the dataset. The objective of this thesis is to analyse the impact of weather on the

[3] `http://gu.com/p/3phve/stw`, last access May 11, 2016

[4] `https://dev.twitter.com/streaming/public`, last access May 11, 2016

Figure 3.2: The user is able to check-in at a venue when he or she is there.

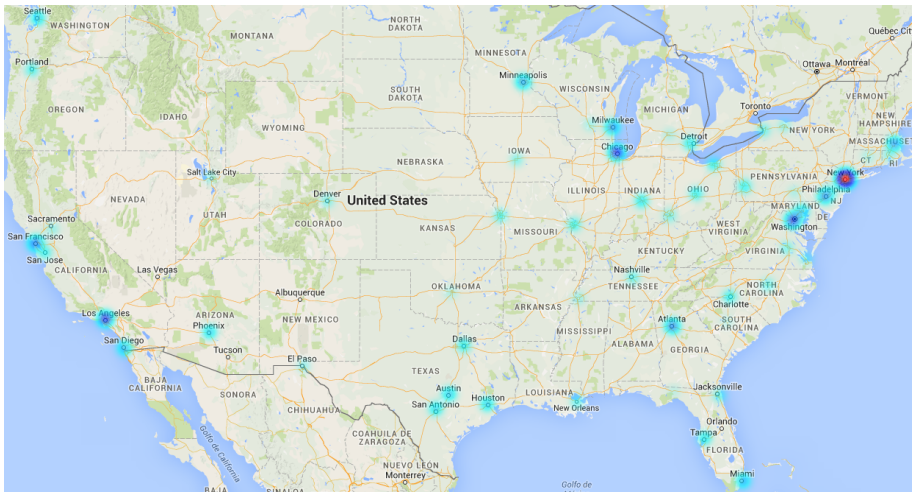


Figure 3.3: A heatmap of the check-ins of the dataset filtered by cities of the USA.

popularity of different venues. Since the described Foursquare dataset was quite huge, it was filtered by cities of the USA. Table 3.1 shows that the filtering process shrank the dataset to a tenth of its original size. Furthermore, it was necessary to concentrate on four cities that represent the variety of climate for the recommender model created in this thesis. Figure 3.3 then reveals the check-ins spread over 60 cities of the USA.

	Original dataset	Filtered dataset
#Venues	3,680,126	501,415
#Cities	415	60
#Countries	77	1
#User	266,909	50,812
#Check-ins	33,278,683	3,545,288

Table 3.1: Summary of the differences between the original and the filtered dataset.

Categories

As Bao et al. (2012) mentioned, the user-location matrix of a LBSN is sparse because the number of check-ins a user can made is limited. To bypass the data sparsity problem for the empirical analysis part in Chapter 4, it was necessary to summarize venues over categories. With dense category data it is easier to deduce weather related trends than with sparse venue data and a good overview of the dataset is provided. Therefore one of the main requirements to the dataset was that it collates categories to the venues. Although the original dataset provided category assignments it showed up that $\sim$25% of the venues were wrong assigned compared to the current response of the foursquare API [5]. The reasons for this incorrect assignments may be the following:

- 4,447 out of 501,900 venues were assigned to the category "General Entertainment". Which more or less means not assigned. After the recrawl process just 3,984 venues are assigned to General Entertainment which means that the other venues got a more appropriate categorization in the meantime.
- Foursquare reviews the category assignments in an evaluation process and reassigns the inappropriate ones (see Figure 3.4).

Considering that the Foursquare API offers an endpoint[6] where users can add venues by themselves, some sort of uncertainty on the correctness of the assignments will always persist, since the accuracy of the assignment always depends on the user that adds the venue.

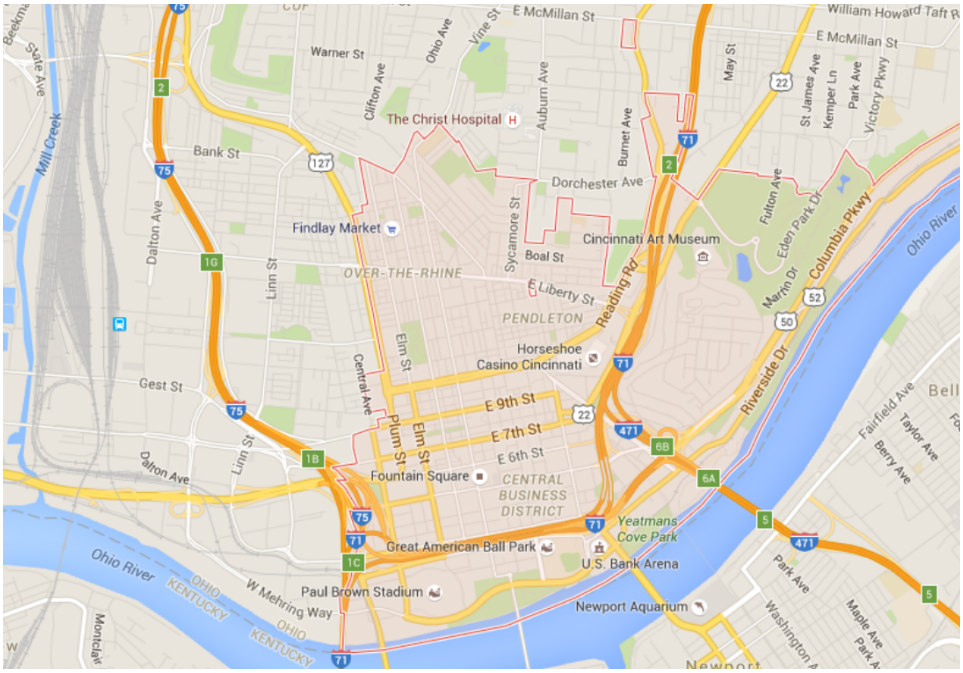
After the validation of the categories with the Foursquare API the dataset ended up with 765 distinct categories. Figure 3.5 shows how the check-ins are distributed over users and categories. It is revealed that the dataset has some very

[5] `https://developer.foursquare.com/docs/explore#req=venues/`, last access May 11, 2016

[6] `https://developer.foursquare.com/docs/venues/add`, last access May 11, 2016

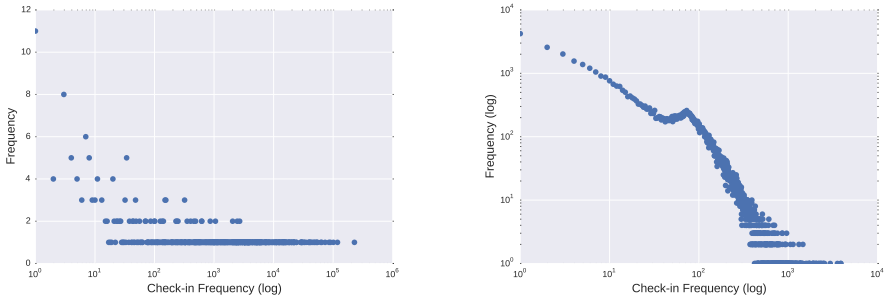
VENUE-ID	LAT	LONG	CAT	COUNTRY
4ff1af57e4b0c63566092ee1	41.450337	31.765511	Home (private)	TR
4ff1af7be4b0b5102f0636e2	29.290319	47.969554	Caf[illegible]	KW
4ff1afafe4b0eddc4b963729	40.177099	28.925818	Factory	TR
4ff1afcee4b00b9a29aac938	40.996200	39.698417	Home (private)	TR
4ff1afd4e4b0dc34772e6862	25.402233	55.506415	Automotive Shop	AE
4ff1afffe4b0cb1358faa5cd	40.719206	29.821742	Dessert Shop	TR
4ff1b02ce4b07e8f74f7c89f	51.374908	-0.092589	Light Rail	GB
4ff1b065e4b000351b3ddf5b	3.585239	98.698572	Home (private)	ID
4ff1b075e4b05cede10400ed	55.807748	37.521331	Bank	RU
4ff1b087e4b0dccbdde0ba58	40.797768	-73.994530	Residential Building	
4ff1b091d86ceeedc57e0114	-23.990017	-46.283898	Office	BR
4ff1b099e4b0d01fb34a135e	39.116737	-84.513809	Ski Trail	US
4ff1b0b1e4b092e4b2df5bc6	47.607006	-122.335537	Coffee Shop	US

(a) Original dataset



(b) Location in google maps.

Figure 3.4: An example why recrawling of the categories was necessary to ensure data accuracy. The original dataset listed the venue as “Ski Trail”, but the current Foursquare API explorer listed the venue as “Neighborhood” and google maps proves that the API is right.



(a) Check-in distribution per Category **(b)** Check-in distribution per User

Figure 3.5: Both user and category check-in distribution show the dataset contains prevailing users and categories.

popular categories with more than 100,000 check-ins compared to 148 categories having less than 100 check-ins. The same applies to the user check-ins. While ∼11,000 user have more than 100 check-ins ∼10,000 users have less than five check-ins. Since it is challenging to display 50,000 users or 765 different categories on one data plot, the data analysis in Section 4 often focuses on the popular categories and the more active users.

Top Level Categories

Besides the above mentioned possibility, to take the top-N categories as representatives for the impact analysis, Foursquare offers a category hierarchy with ten super categories as root [7]. The most conspicuous finding in the check-in distribution over this super categories (see Figure 3.6) is that "Events" has less than 3,000 check-ins whereupon the other categories have much more than 100,000. The reason is that "Events" was not included in the original dataset from Yang et al. (2015b) and came into the dataset during the recrawl process described in section 3.1.

3.2 Weather Dataset

"Weather: The state of the air and atmosphere at a particular time and place : the temperature and other outside conditions, such as rain, cloudiness, etc., at a particular time and place"[8]

[7] `https://developer.foursquare.com/categorytree`, last access May 11, 2016

[8] `http://www.merriam-webster.com/dictionary/weather`, last access May 11, 2016

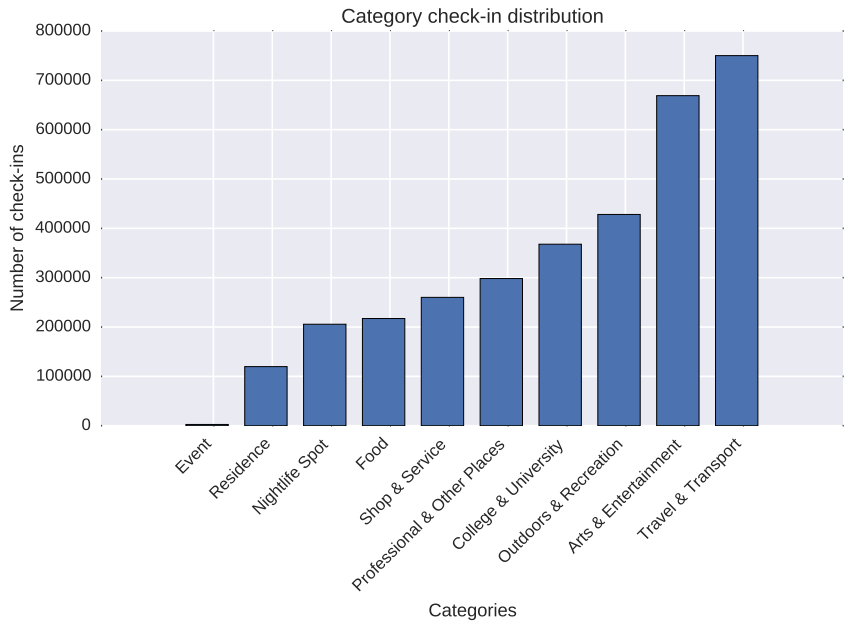


Figure 3.6: Check-in distribution over top level categories in Foursquare. The "Events" category was added during the recrawl process described in Section 3.1 and has less check-ins compared to the others.

According to that definition, building a weather aware recommender assumes three different kinds of information:

- **Place → geographic coordinates (longitude, latitude)**
- **Time → unix timestamp with attendant timezone offset)**
- **Miscellaneous outside conditions**

The dataset already provides place and time information. Therefore for each of those <*time, place*> tuples the appendant outside conditions had to be crawled. For this purpose forecast.io[9] provides an API. With the forecast call

https://api.forecast.io/forecast/APIKEY/LATITUDE,LONGITUDE,TIME

it is possible to retrieve weather information at a given timestamp from the weather archive of forecast.io. Due to the fact that the first 1,000 calls to the API are free and each following request costs 0.0001$, it was important to take as little calls as possible. Because the granularity of the weather API ends at city level one request per <*time, city*> tuple is enough whereby city is represented by the geographical longitude and latitude of the center of the city. To achieve that

[9] `https://developer.forecast.io/docs/v2`, last access May 11, 2016

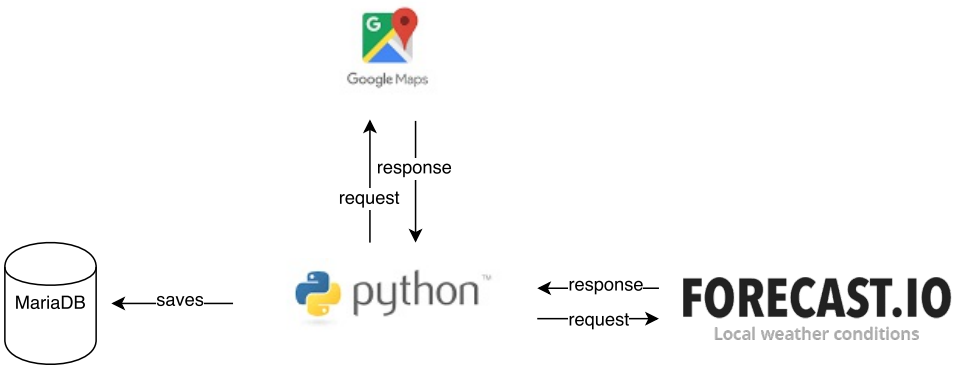


Figure 3.7: Weather data was crawled with a python script. First, it requested the geographical center of the cities. Afterwards it took one request per <day, city> tuple to the weather API and saved the result to a local MariaDB database (see section A.1).

savings it was necessary to first retrieve the geographical position of the center of all 60 cities in the database. Figure 3.7 explains that this was achieved by requesting the google maps geolocation interface[10] and by saving the results into a local MariaDB database[11] (see section A.1). Considering the fact that forecast.io returns an hourly weather report for each daily request it was sufficient to take one call per <*city, day*> tuple to retrieve hourly weather information. Since the check-in dataset consists of 60 cities and 494 distinct days, 29,640 requests were necessary to get all weather information needed. Listing 3.1 shows a sample API request for a check-in on the 4th december 2013 in Minneapolis.

Besides the ease of use and the availability of historical weather data up to 60 years ago, another reason for taking forecast.io as weather source was the number of different weather features. Table 3.2 describes the different weather features and their properties that are used in this thesis.

Listing 3.1: Sample request for a check-in in Minneapolis to the weather API. It returns first the hourly weather report and then the daily summary.

```
1 https://api.forecast.io/forecast/269b283068aaf91a686b93b667bf9e8a/44.924194,-93.307785,1365800400
2
3 {{"latitude":44.924194,"longitude":-93.307785,"timezone":"America/Chicago",
4 ...
5 "hourly":
```

[10] `https://maps.googleapis.com/maps/api/geocode/json?address=`, last access May 11, 2016

[11] `https://mariadb.com/de`, last access May 11, 2016

```
{"summary":"Light snow (under 1 in.) overnight and in the morning.","icon"
    :"snow",
"data":[
{"time":1365742800,"summary":"Light Snow","icon":"snow","precipIntensity":
    0.0068,"precipProbability":0.32,"precipType":"snow","
    precipAccumulation":0.058,"temperature":30.6,"apparentTemperature":22.
    79,"dewPoint":29.15,"humidity":0.94,"windSpeed":8.57,"windBearing":27,
    "visibility":1.71,"cloudCover":1,"pressure":1006.33},
...
{"time":1365825600,"summary":"Overcast","icon":"cloudy","precipIntensity":
    0,"precipProbability":0,"temperature":30.94,"apparentTemperature":25.1
    7,"dewPoint":25.72,"humidity":0.81,"windSpeed":5.76,"windBearing":328,
    "visibility":9.3,"cloudCover":1,"pressure":1011.1}
]},
...
"daily":{
"data":[
{"time":1365742800,"summary":"Light snow in the morning.","icon":"snow","
    sunriseTime":1365766463,"sunsetTime":1365814535,"moonPhase":0.08,"
    precipIntensity":0.0014,"precipIntensityMax":0.0068,"
    precipIntensityMaxTime":1365742800,"precipProbability":0.32,"
    precipType":"snow","precipAccumulation":0.278,"temperatureMin":28.13,"
    temperatureMinTime":1365768000,"temperatureMax":35.82,"
    temperatureMaxTime":1365796800,"apparentTemperatureMin":19.84,"
    apparentTemperatureMinTime":1365757200,"apparentTemperatureMax":29.09,
    "apparentTemperatureMaxTime":1365796800,"dewPoint":26.75,"humidity":0.
    82,"windSpeed":6.7,"windBearing":340,"visibility":6.96,"cloudCover":1,
    "pressure":1007.49}]},
...
```


Weather feature	Properties	Range in dataset
Precipitation intensity	Precipitation intensity measured in milimeters of liquid water/hour.	$0mm/h - 34,29mm/h$
Temperature	Temperature measured in degree Celsius	$-24,48° - 46,58°$
Wind speed	Wind speed measured in meters/second	$0m/s - 19,13m/s$
Cloud cover	Value between 0 and 1 displaying the percentage of the sky covered by clouds.	$0 - 1$
Humidity	Value between 0 and 1 representing the "Percentage relative humidity" is defined as the partial pressure of water vapor in air divided by the vapor pressure of water at the given temperature."(Perry et al., 1997)	$0,02\phi - 1,00\phi$
Pressure	Atmospheric pressure measured in hectopascals.	$957,11hPa - 1046,05hPa$
Visibility	Value representing the average visibility in kilometers capped at 16,09	$0km - 16,09km$
Moonphase	Value from 0 to 1 representing the range between new moon and full moon	$0 - 1$

Table 3.2: The different weather features used and their properties.

4 Empirical Analysis

In order to answer research question one **(RQ1)** this Chapter investigates the data in detail and tries to find out if there is an influence of different weather conditions on the entities of a LBSN, such as people, venues, categories. Furthermore, it was important to see how the different weather features interact with each other to retrieve a maximum of information gain out of them with a minimum of computational power. Additionally, it was a goal to examine the influence of weather on the travel distance of a user and the influence of seasonality on the check-in behaviour. The approach to test this hypothesis is primarily the use of visual methods, such as scatter plots, histograms and line charts. To explore the dataset in detail a handful of statistical tools are used which are described in Chapter 4.1. The analysis was made with the help of *Python2.7*[1], *Matplotlib*[2], *SciPy*[3] and *Seaborn*[4].

4.1 Methodology

To be able to investigate the dataset and answer research question one **(RQ1)** some statistical and graphical methods were used. This methods are described in this Section.

Statistical Methods

In this Section the statistical methods used in the empirical analysis of this thesis are described.

[1] `https://www.python.org/`, last access May 11, 2016

[2] `http://matplotlib.org/`, last access May 11, 2016

[3] `http://www.scipy.org/`, last access May 11, 2016

[4] `http://stanford.edu/~mwaskom/software/seaborn/`, last access May 11, 2016

Statistical Hypothesis testing

*"Hypothesis testing aims at a decision on whether or not a hypothesis on the nature of the population is supported by the sample."*Kuhnt and Taeger (2014)

According to this definition statistical hypothesis testing is based on the construction of a null hypothesis H_0 and an alternative Hypothesis H_1. The objective of this method is to find out if two datasets are statistically related or not. The usage of hypothesis testing in this thesis is, for instance, if there is a statistically significant difference between a check-in distribution of a specific weather type and the distribution over all check-ins. This should lead to the answer if there is an impact of weather to the check-in manner of different users or categories. The result of a hypothesis testing is whether to reject or to keep the hypothesis. A statistical hypothesis test might have the following four outcomes:

- True H_0 not rejected
- False H_0 rejected
- True H_0 rejected (Type I error)
- False H_0 not rejected (Type II error)

The first two cases are those where the hypothesis test was correct and assigned the hypothesis correct as true or false. In the latter cases the test assigned the trueness of the hypothesis wrong. A type I error occurs when the test rejects an hypothesis that is true. Due to the fact that a hypothesis test determines a level of significance α the chance of an error of type I is always α. Therefore a significance level of α=0,05 leads to a chance of 5% having an error of type I. This α represents the smallest significance level that leads to a rejected H_0. Contrary to the error of type I a type II error does not reject a False hypothesis and is determined by the power of the test β. (Lehmann and Romano, 2005)

P-Value. The p-value in an hypothesis testing approach determines the distance between the observed data and the null hypothesis constructed. It depicts the probability of the data under test given the null hypothesis. That means if p-value is beneath a certain significance level α, H_0 can be rejected keeping in mind that errors of type I might occur. Considering the large dataset (see Chapter 3) used in this thesis the effect described in Lin et al. (2013) has to be considered. The authors of this paper illustrate that the standard error between two datasets becomes very low if the size of the sample increases. That means that already very slight differences between H_0 and the dataset can lead to a statistically significant p-value and therefore to a rejection of the hypothesis although the hypothesis may fit. Therefore it is important not to exclusively rely on statistical hypothesis testing but also on using graphical methods as described in Section 4.1.

Chi-Square Test. The chi-square test first introduced in Pearson (1900) is a statistical hypothesis test (see section 4.1) and it can be used to detect if the differences between categorical frequencies of two sets are casual or significant. According to Lane (2016) the chi square value is given by equation 4.1.

$$X^2 = \sum_{i=0}^{k} \frac{(O_i - E_i)^2}{E_i} \tag{4.1}$$

where k is the number of categories, E_i the frequency of category i of the expected distribution and O_i the frequency of category i of the observed distribution. Comparing the test outcome to a chi squared distribution with k-1 degrees of freedom reveals the p-value and thus the significance level.

Cosine Similarity

Cosine similarity is a measurement of the similarity of two vectors $\vec{v_1}$ and $\vec{v_2}$. It computes the angle between the vectors and takes the cosine of the result which leads to a result x where $x \in [0, 1]$. The definition of cosine similarity is shown in equation 4.2.

$$x = cos(\vec{v_1}, \vec{v_2}) = \frac{\vec{v_1} \cdot \vec{v_2}}{\|\vec{v_1}\|\|\vec{v_2}|} \tag{4.2}$$

Pearson correlation coefficient

Calculates the linear correlation between two variables X and Y. The coefficient is defined as the division of the covariance of the two variables by the square root of their variances (see equation 4.3).

$$r_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} * C_{jj}}} \tag{4.3}$$

A coefficient of one means total positive correlation, minus one total negative correlation and zero means no correlation (Pearson and Blakeman, 1906). Most of the times additionally to the correlation coefficient a hypothesis test is made to compute the significance of the calculated correlation. Because the correlation coefficient is an effect size which measures the strength of two distributions it is possible to determine the power of a correlation. Evans (1996) introduced a classification scheme for the absolute value of r as shown in Table 4.1.

Z-Score

A statistical Z-Score is a measurement to reveal the relation of a value in a group to the mean of the group. In this thesis Z-Score is used to relativise the different means of weather features in different regions and to relativise weather features to

Value of r	Classification
0,0 - 0,19	very weak
0,2 - 0,39	weak
0,4 - 0,59	moderate
0,6 - 0,79	strong
0,8 - 1,00	very strong

Table 4.1: The classification scheme for different r values of the correlation calculation.

the specific region and time context. However, the formular is given by equation 4.4.

$$z = \frac{v - avg(city, month)}{\sigma} \tag{4.4}$$

where avg(city, month) gives the average value of a weather feature in a specific city and month, v is the value of a weather feature from a specific check-in and σ the standard deviation. That score reveals the difference between the typical month temperature and the check-in in a specific category.

Standard Error of the Mean

The standard error of the mean (SE) is a metric that determines the precision of the mean of a sample of the population. In other words, it expresses how close the sample mean is to the population mean. Equation 4.5 shows the calculation of the SE where s is the standard deviation of the population and N is the size of the sample. That means that the higher the sample size, the lower the SE and the better the estimate of the mean.

$$SE = \frac{s}{\sqrt{N}} \tag{4.5}$$

Visual Analysis Methods

In order to rely not only on statistical inference methods and its underlying problems discussed in Figure 4.1 also visual tools should be taken into consideration. This tools can be summarized as exploratory data analysis introduced by Tukey (1962).

Q-Q plot

The q-q plot mentioned in M. B. Wilk (1968), is a visual method to compare two distributions with each other. To construct a q-q plot first of all data has to be ordered from smallest to largest values. Furthermore, this ordered data is split up into quantiles. Each quantile has an upper bound which is the representative

value for one quantile. So the common upper bound value of quantile i of the original data is drawn on the y-axis against the upper bound value of quantile i of the theoretical data on the x-axis. The quantiles are usually derived from the percent point function (ppf) which is the inverse function of the cumulative distribution function (cdf). If the two datasets are identically distributed the plot will show a line with slope one.

4.2 Results

This section first examines weather based features to expose if all weather features are equally suitable for the purpose of a context-aware recommender or if some of them are meaningless to check-in actions. Furthermore, the impact of weather on the users' travel distances is analysed to see if it is useful to incorporate weather related travel distance into the recommender model. Afterwards, category check-in patterns under different weather circumstances are analysed to reveal the influence of weather conditions on the popularity of categories. A further interesting question is if venue prediction based on weather features works better in regions were weather variability is higher or lower. The weather distribution and variability analysis for different regions serves as a basis for answering this question in Chapter 5. Finally, the users' check-in behaviours under different weather conditions are investigated to find out if there are users that prefer specific weather conditions respectively if there are weather sensitive users.

4.2.1 General Analysis

This Section provides an analysis on the distributions of the different weather features and how they correlate with each other in order to examine the usefulness of incorporating them into a recommender model.

Distribution Analysis. In order to estimate the cardinality of the particular weather features Figure 4.1 and 4.2 reveal the distributions over the different weather features. Humidity, pressure, temperature and windspeed approximately fit under the Gaussian bell curve what is displayed by the gaussian fitting in the plots and the Q-Q Plot underneath the different distribution plots. A statistical inference approach led to the problems described in Section 4.1 and therefore, a rejection of a normal distributed null hypothesis of the features which are apparently not normal distributed was not possible. For this reason the Q-Q Plots were made to identify cloud cover, moonphase, precipitation intensity and visibility as definitely not normal distributed. Especially moonphase and cloud cover fit to the uniform distribution and therefore Figure 4.1a and 4.1c show the Q-Q Plot

[6] `https://developer.forecast.io/docs/v2`, last access May 11, 2016

against the uniform distribution. This leads to the assumption that moonphase and cloud cover will not have that high impact on the check-in behaviour of the users. Visibility (Figure 4.1b) and precipitation intensity (Figure 4.1d) neither fit a normal distribution nor a uniform distribution. Therefore, the impact of this two features looks very promising. The interesting question concerning the normal distributed features will be to find out if these extreme weather conditions on the right and left side of the curve have a seasonal origin or if it also depends on the check-in behaviour in different categories. For example, does the category "Ski Area" have another temperature distribution than "Ice Cream Shop" and if yes, the question is if the categories have different shaped distributions or if they just shift the mean. The answers to this questions are stated in Section 4.2.3.

Correlation Analysis. Building a recommender is a computation intensive task when operating on a huge dataset. Each weather feature included in the calculations takes additional computation power. Therefore a cost-benefit analysis has to be made to plug a maximum of information into the recommender engine with a minimum of computation costs. According to Hall (1999)

"A good feature subset is one that contains features highly correlated with (predictive of) the class, yet uncorrelated with (not predictive of) each other."

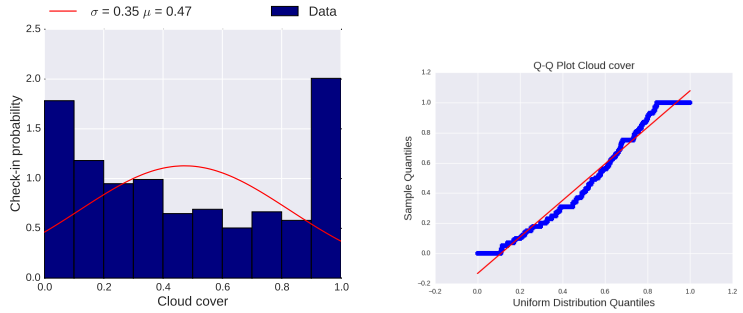
That means that if there are two features that highly correlate with each other it is possible to eliminate one of them because there is no additional information added when keeping both of them. For this selection task the pearson correlation (Section 4.1) coefficient was used to compute correlation between the eight weather features described in Section 3.2.
Figure 4.3 shows some very reasonable correlations between features showing significance stars for the p-value as described in Table 4.2. For instance, there is a

Symbol	Description
*	$P \leq .05$
**	$P \leq .01$
***	$P \leq .001$

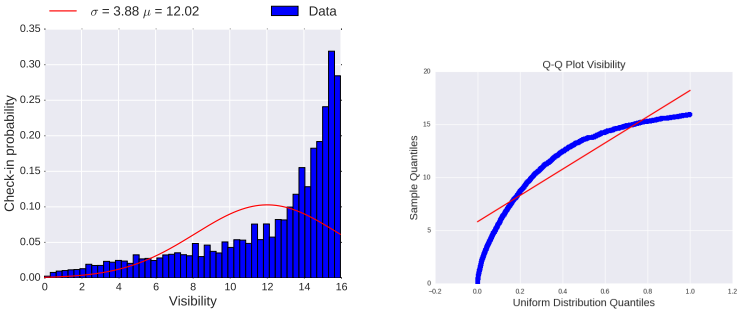
Table 4.2: Significance levels indicated by stars.

quite strong negative correlation between visibility and humidity what describes the natural weather phenomenon that a high humidity blurs visibility also the negative correlation between cloud cover and visibility shows the negative impact of clouds to visibility. Interesting is the negative correlation between relative humidity and temperature which reflects the fact that relative humidity represents the saturation of moisture in the air and cold air does not need that much moisture to be saturated. An interesting high positive correlation is between

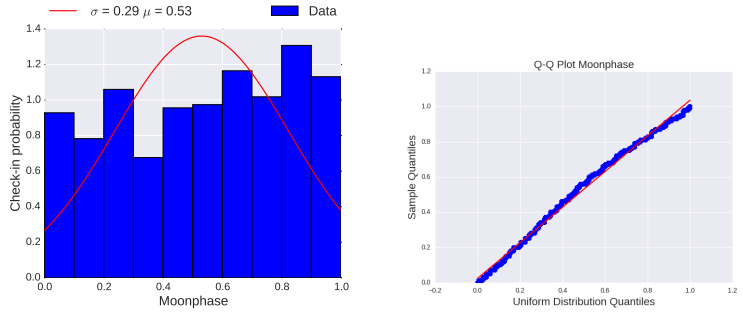
windspeed and temperature which leads to the assumption that the foehn effect took place leading to a high wind speed and high temperatures at the same time. Figure 4.3b instead shows the correlation between the z-scored (see Section 4.1) weather features. It shows whether a feature correlates to another one in terms of deviations from the mean. In other words it describes if a weather features extreme cases correlate with another weather features' extreme cases. Comparing this extreme weather situations it shows up that the correlation between wind speed and temperature disappears since the foehn effect is compensated.



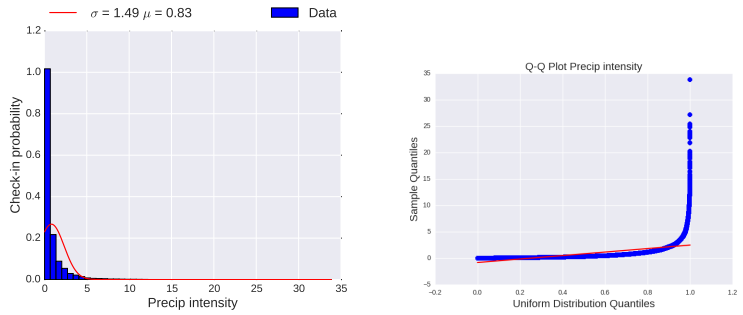
(a) Cloud cover



(b) Visibility

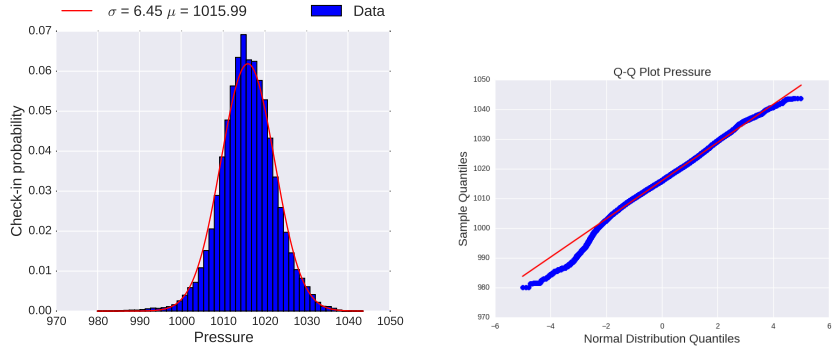


(c) Moonphase

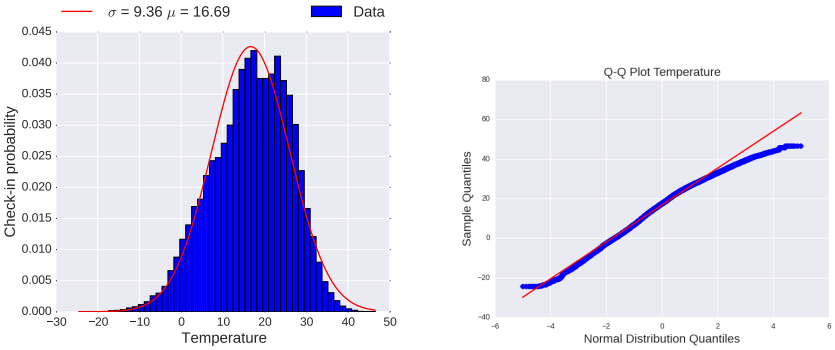


(d) Precipitation intensity

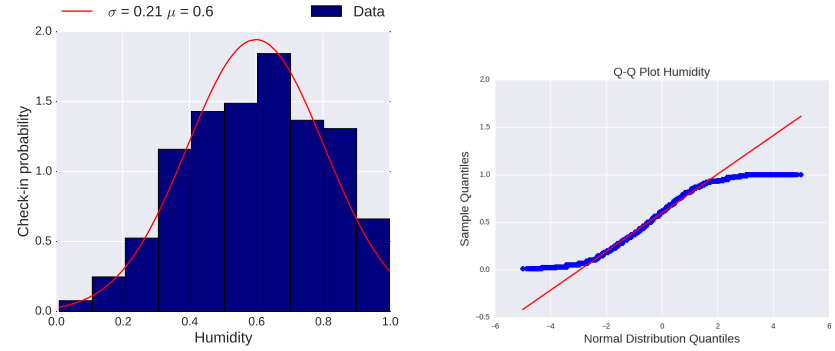
Figure 4.1: The four weather features that are not normal distributed crawled from forecast.io[5]and their appendant check-in frequency distribution. The lower plot of each subfigure shows the Q-Q Plot against the according uniform distribution. It shows that moonphase and cloud cover is uniform distributed. Precipitation intensity and visibility is neither uniform nor normal distributed what promises high impact on the recommender accuracy.



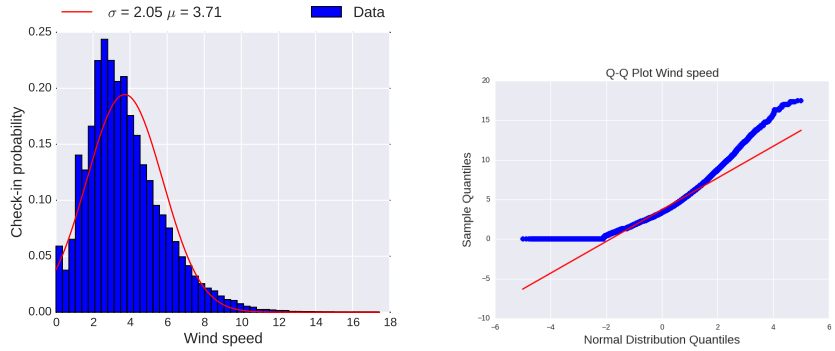
(a) Pressure



(b) Temperature

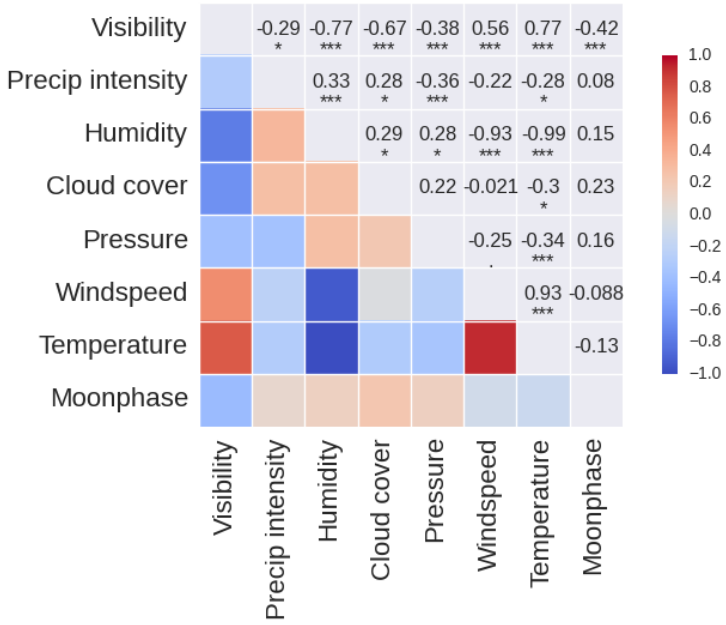


(c) Humidity

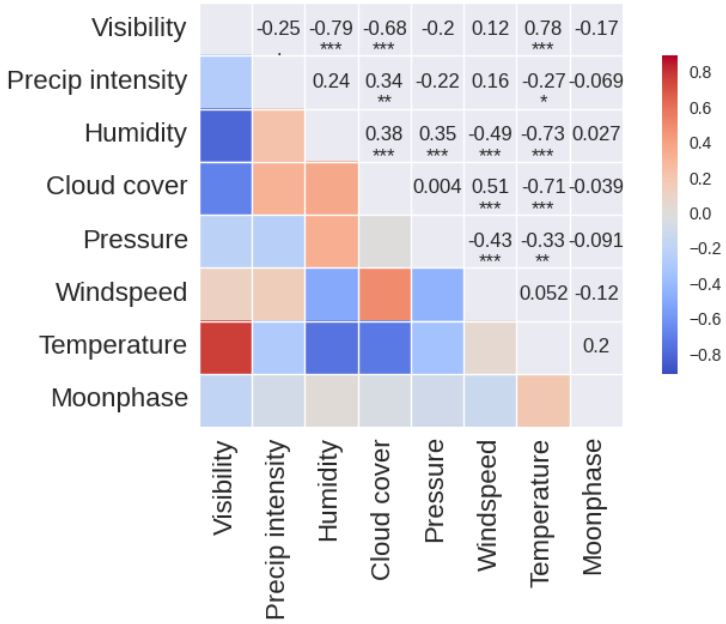


(d) Windspeed

Figure 4.2: Weather features that are normal distributed crawled from forecast.io[6]and their appendant check-in frequency distribution and Q-Q Plot against the normal distribution.



(a) Correlation

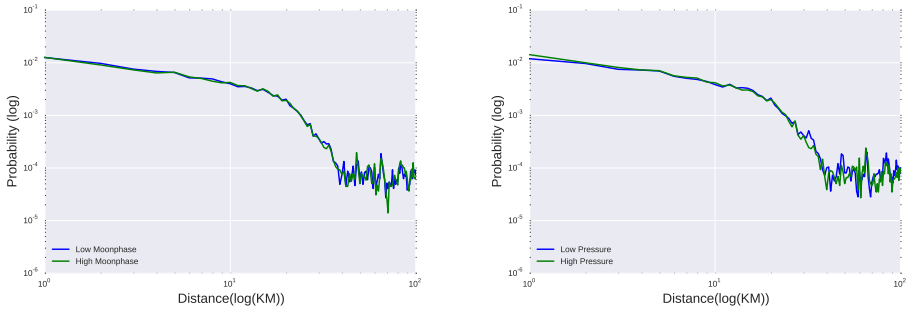


(b) Z-Score Correlation

Figure 4.3: The correlation between the eight weather features. Z-Score computes the correlation between the features with corrected values by the corresponding average value of the city and month.

4.2.2 Distance Analysis

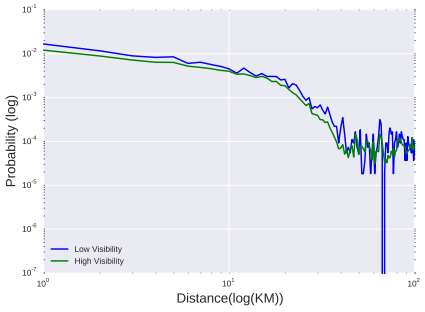
In order to find out if weather has an impact on people's travel distance a function as shown in Balby Marinho et al. (2015), was drawn. The aim was to compare the distance functions in different weather characteristics. For instance, a comparison of the distance function of high temperatures to the distance function low temperatures to answer the question if people tend to travel further when it is hot or if people travel less distances when precipitation intensity is high.
For this reason the distance between two check-ins of a user was calculated and the according values of the weather features of the second check-in were saved. As a result Figure 4.4 shows the distance functions capped at 100km in log-log scale. The results were to a large extent as expected. Moonphase and pressure did not show that much impact on the travel distance. Unforeseen was that precipitation intensity does not show a high impact on the travel distance of the people as Figure 4.4c reveals that there is just little difference between the low and high precipitation intensity function. On the other side visibility, temperature, humidity, wind speed and cloud cover show an effect on the travel distance. Low visibility and temperature cause an increase in short term travels as shown in Figure 4.4d and 4.4e, what implies a low likelihood for long distances. The inverted behaviour is shown by humidity (see Figure 4.4f), wind speed (see Figure 4.4g) and cloud cover (see Figure 4.4h) where high values have the effect of preferring short distances over long ones. However, a p-value (KS-test) smaller than .001 on the hypothesis test smaller than showed significant differences for all eight weather features.



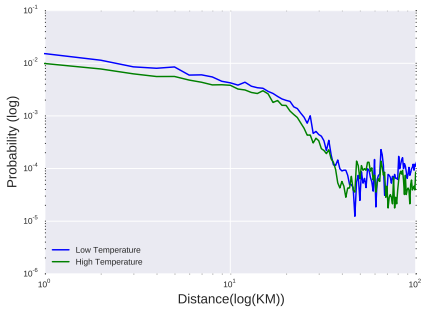
(a) Moonphase

(b) Pressure

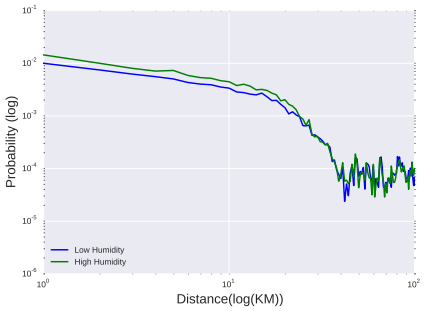
(c) Precipitaton intensity



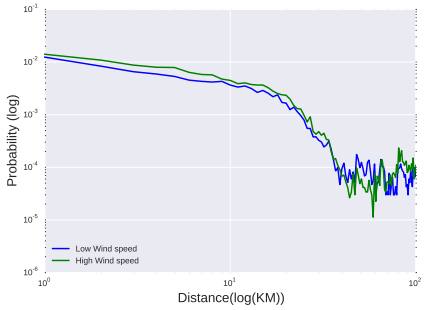
(d) Visibility



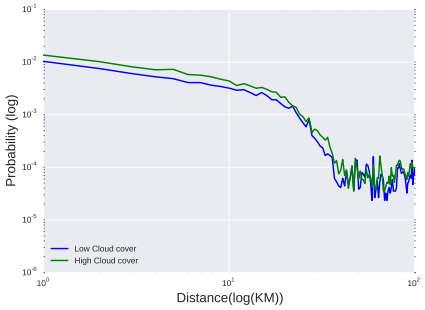
(e) Temperature



(f) Humidity

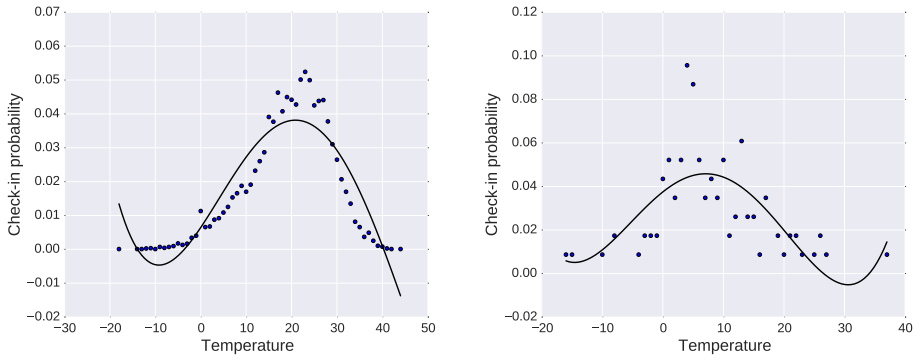


(g) Windspeed



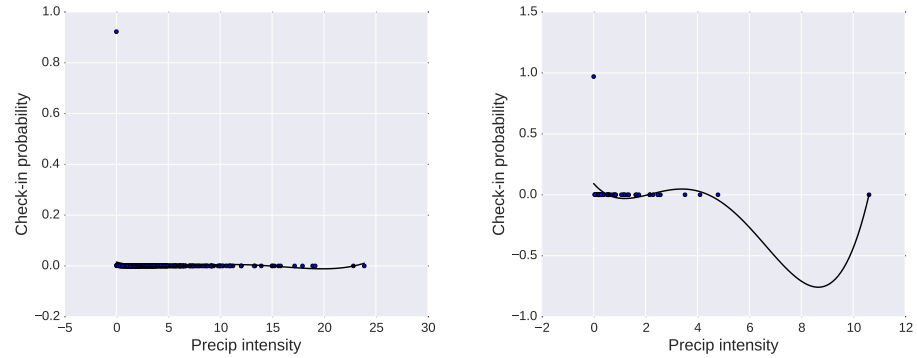
(h) Cloud cover

Figure 4.4: While Pressure and Moonphase do not show a high impact on the travel distance at extreme conditions, the other features do. Unexpected was the little influence of precipitation intensity. Nevertheless, a p-value (KS-test) smaller than .001 for all eight weather features proves a statistical significant difference between the distributions.



(a) "Ice Cream Shop" **(b)** "Ski Area"

Figure 4.5: The difference between "Ski Area" and "Ice Cream Shop" appears in the difference of the temperature distribution of them. While "Ice Cream Shop" has its peak at $\sim 22°C$ and leans to the right "Ski Area" has its peak at $\sim 4°C$ and leans to minus temperatures.



(a) "Bar" **(b)** "Car Wash"

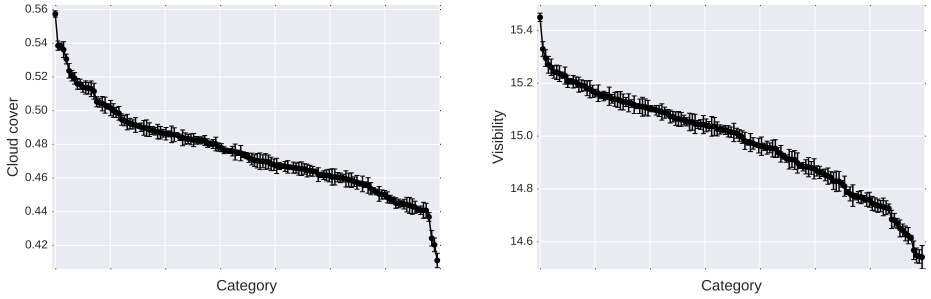
Figure 4.6: A "Car Wash" place is more susceptible to precipitation than a "Bar". While people still like to go to a bar when it is raining (up to 25 mm/h) the probability of washing a car sinks immediately at more than 4 mm/h.

4.2.3 Categorical Analysis

This Section answers the question if different categories are checked-in at different weather conditions. More precisely it was important to find out if it is possible to assign categories to a typical weather condition which is representative for this group or if the weather features are uniform distributed over the categories.

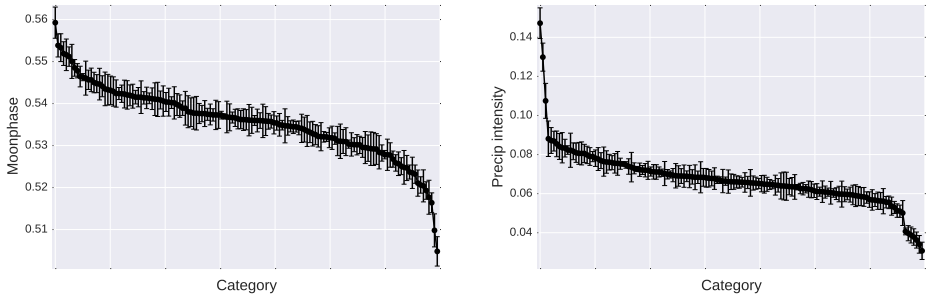
Distribution Analysis. To display the dependence between a weather feature and a category a distribution was drawn having the weather feature on the x-axis and the probability of a check-in at the corresponding value on the y-axis. These diagrams were created on a city level to avoid weakening of significance caused by the different climatic regions. Additionally, the plot contains a fitting line using NumPy's poly1d function[7] with three degrees of freedom (DOF) to manifest the slope of the distribution.
Having a look at Figure 4.5 confirms the suspicion that temperature influences the check-in behaviour of users. Figure 4.5a shows the check-in distribution over temperature in the categoriy "Ice Cream Shop" and it presents a preference of warmer temperatures when consuming ice cream. The distribution peaks at $\sim 22°C$ leading to the assumption that spring could lead people to immediately buy ice cream while in summer people already had enough ice cream or too high temperatures let people stay at home. On the other hand Figure 4.5b reveals a check-in behaviour preferring lower temperatures for ski areas having the peak at $\sim 4°C$. That means that although there is no outdoor skiing area in New York and the category also consists of ski shops, indoor skiing places and ice skating places, people prefer this places at colder temperatures. Taking precipitation intensity into consideration it can be seen in Figure 4.6 that there are categories like "Bar" (Figure 4.6a) that follow the overall precipitation distribution described in Section 4.2.1. On the other side there are categories as "Car Wash" (see Figure 4.6b) showing almost zero tolerance to rainy weather conditions because they have not been been visited more than once when precipitation intensity is higher than five mm/h. However, in general there are no categories favouring for rainy weather conditions.
Another possibility to investigate categories is to create a chart having the categories on the x-axis and the according feature mean on the y-axis. The SE (see Section 4.1) was added to each category sample to state the accuracy of the mean of the category. To enhance visibility and accuracy of the plot all categories having less than 5,000 check-ins were removed. The result in Figure 4.7 states again a very low variety for the moonphase feature. Also precipitation intensity has almost no impact on the most of the categories (but a high on some). On the other side the slope of the distribution of pressure, visibility and cloud cover definitely shows influence of these features on the categories. It is important to emphasise that the lower the overlap of the SE between the categories the stronger the impact of the weather feature. Especially temperature, wind speed and humidity show a powerful effect on the several categories having a steep slope and a little overlap of the standard errors.

[7] `http://docs.scipy.org/doc/numpy-1.10.1/reference/generated/numpy.poly1d.html`



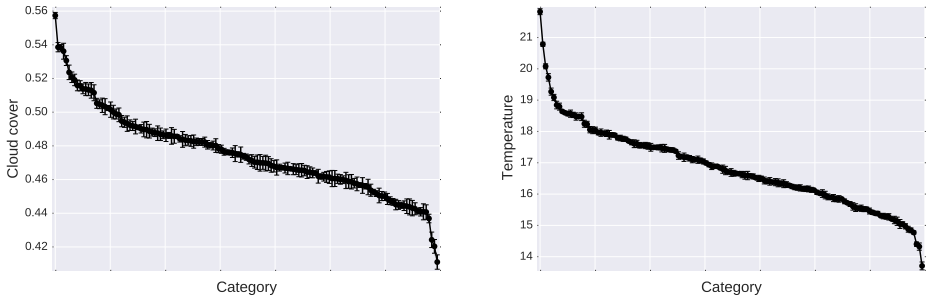
(a) Cloud cover

(b) Visibility



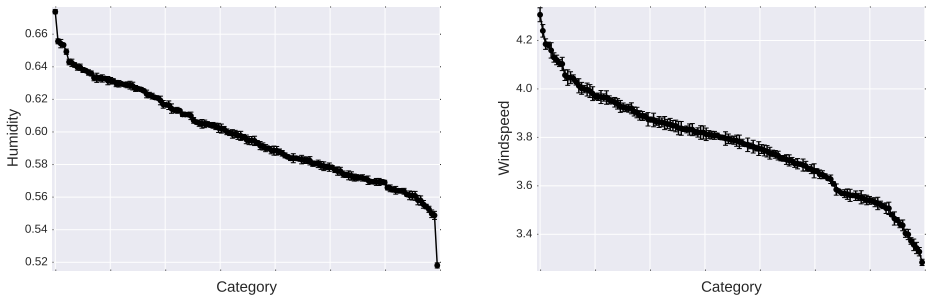
(c) Moonphase

(d) Precipitation intensity



(e) Pressure

(f) Temperature



(g) Humidity

(h) Windspeed

Figure 4.7: Distributions of the eight weather features with the means of the top categories and the standard error of the mean (SE). Especially temperature, wind speed and humidity show a steep slope and little overlap of the SEs what indicates high influence of that features on the popularity of categories.

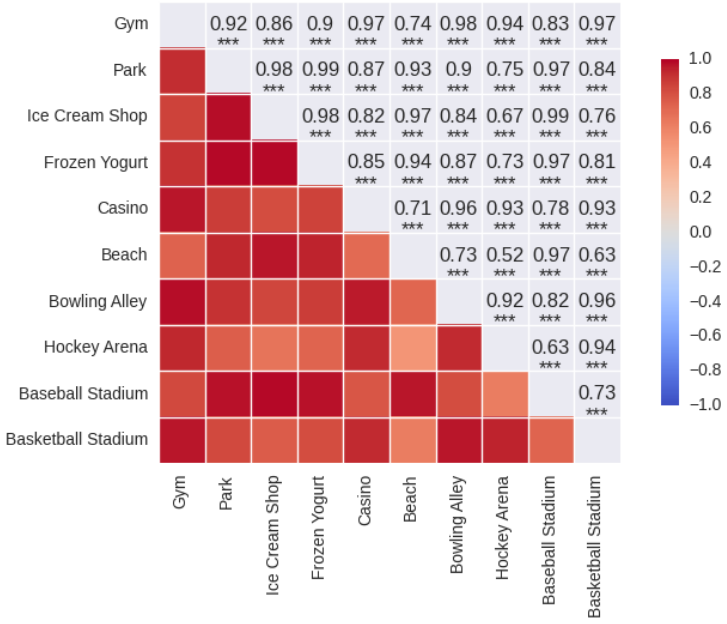
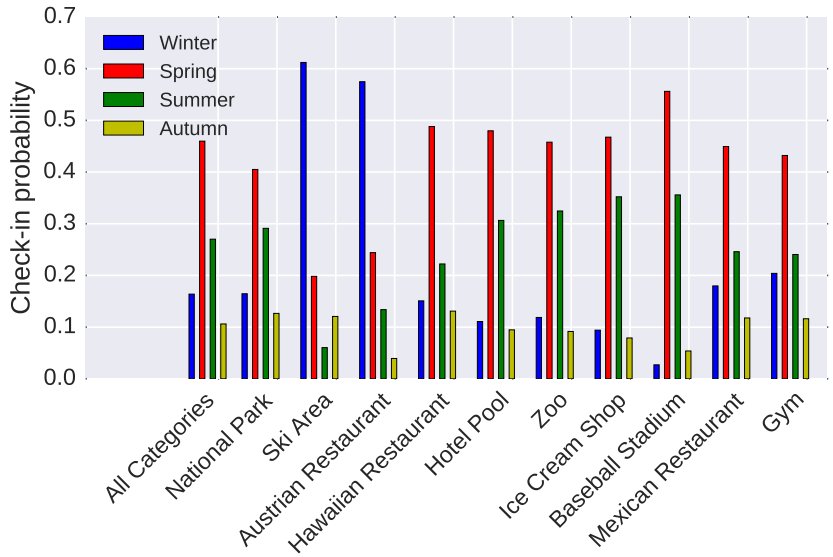


Figure 4.8: A pattern of warm ("Park", "Ice Cream Shop", "Frozen Yogurt", "Beach" and "Baseball Stadium") vs. cold ("Gym", "Casino", "Bowling Alley", "Hockey Arena" and "Basketball Stadium") categories is noticeable. Especially the little correlation of .52 between "Beach" and "Hockey Arena" is outstanding.

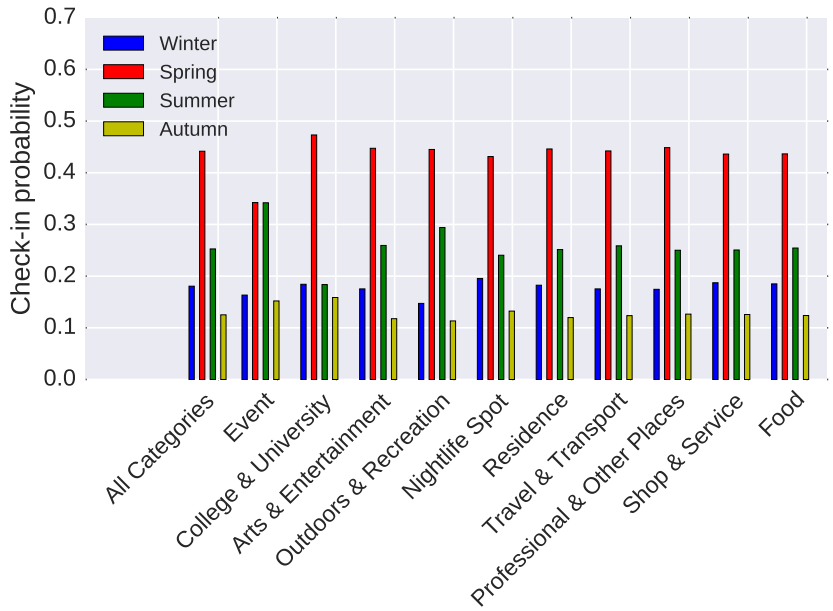
Correlation Analysis. The exploratory of the temperature correlation between different categories results into a natural pattern between warm and cold categories. Calculating the correlation of the temperature frequency of cold categories (as "Gym", "Casino", "Bowling Alley", "Hockey Arena", "Basketball Stadium") with warm categories (as "Park", "Ice Cream Shop", "Frozen Yogurt", "Beach" and "Baseball Stadium") shows that there is a very strong correlation within the groups and just a moderate to strong correlation (according to the classification in Table 4.1) among the groups. That leads again to the assumption that temperature has an impact on the users' check-in behaviours.

Seasonal Analysis. Additional to the eight weather features, also the different seasons might have an impact on the users' check-in behaviours. To answer that question, Figure 4.9 shows the probabilities of a category being checked-in in the four seasons winter, spring, summer and autumn. Figure 4.9a then displays the probabilities of selected popular categories. "All Categories" represents the check-in probabilities of the entire dataset and serves as a ground truth. It should be noted that the dataset crawling was proceeded from April 2012 to September

2013 and therefore there are more check-ins in spring and summer season than in autumn and winter. Nevertheless, it reveals that people prefer winter to autumn and spring to summer for making check-ins. The bar chart also shows that winter is preferred in winter categories as “Ski Area” and “Austrian Restaurant”. On the other side categories as “Hawaiian Restaurant”, “Ice Cream Shop” and “Baseball Stadium” show a slightly higher probability in summer and spring compared to the baseline. Interestingly, “Hotel Pool” does not show a high seasonal impact due to the fact that this category gathers both, indoor and outdoor pools. Having a look at Figure 4.9b reveals that the assignment of different categories to one super category as described in Figure 3.1 disguises the impact of seasons on the sub categories and results in probabilities very close to the baseline. The category events show a different behaviour but that is due to the fact that this group has too little check-ins to balance the probabilities of the sub categories.

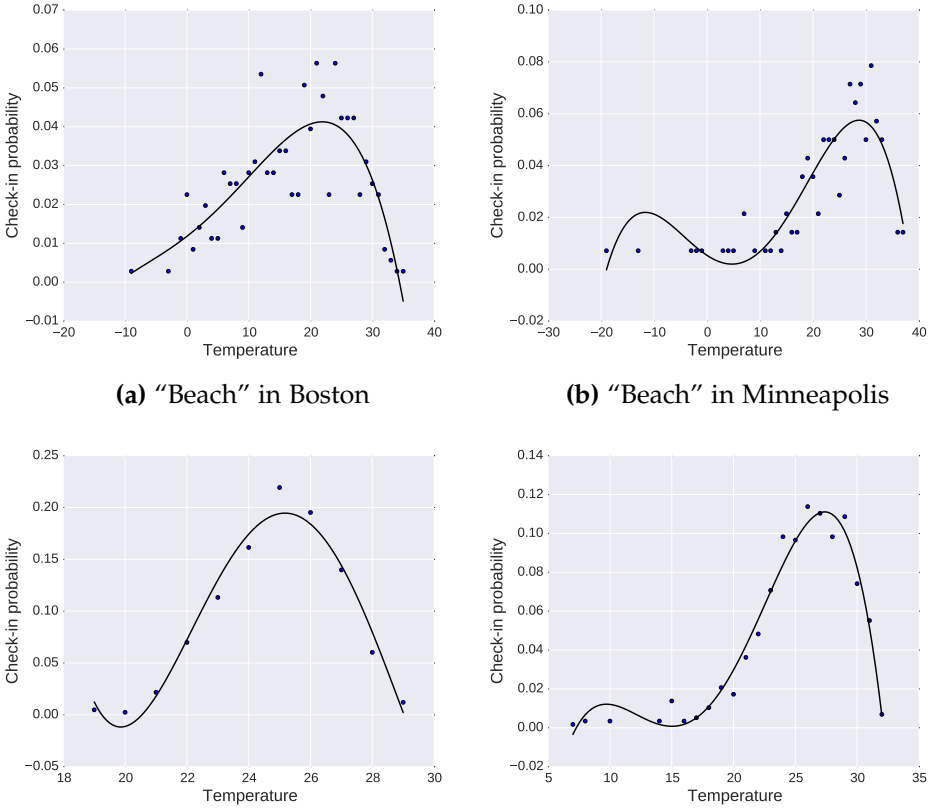


(a) Representative Categories



(b) Super Categories

Figure 4.9: The seasonal check-in probabilities in different categories. "All Categories" is used as a ground truth and displays the probabilities of the whole dataset. It reveals that some categories, such as "Ski Area" and "Austrian Restaurant", burgeon in winter season, while others, such as "Baseball Stadium" and "Ice Cream Shop", have their high season in summer. The assignment of different categories to one super category as described disguises the impact of seasonality and results in probabilities close to the ground truth.



(a) "Beach" in Boston

(b) "Beach" in Minneapolis

(c) "Beach" in Honolulu

(d) "Beach" in Miami

Figure 4.10: The distributions of the category "Beach" shows the diverse behaviour of people in different cities. While people in Boston already go to the beach at lower temperatures as well as when it is hot, people in Minneapolis increasingly go to the beach at higher temperatures.

4.2.4 Regional Analysis

Since weather varies also regional, this section tries to find out in which regions weather has a higher impact and in which regions it has a lower impact. For this reason we chose 4 cities each of them located in one of the cardinal directions to cover the different climatic regions of the USA.

Distribution Analysis. Figure 4.10 shows an interesting comparison of the different check-in distributions in four cities at different temperatures in the category "Beach". Interesting is that although Boston and Minneapolis have comparable temperature variabilities as shown in Figure 4.12f, people in Boston tend to go to the beach at lower temperatures as well as when it is hot, while people

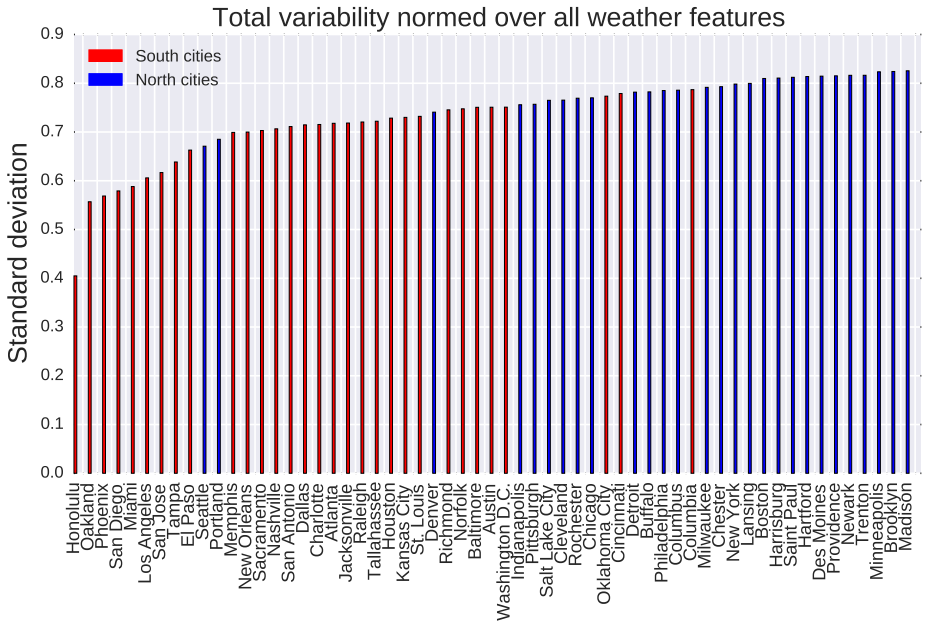


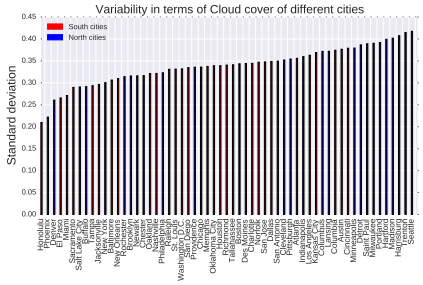
Figure 4.11: The total weather variability normed over all weather features. It shows that in general northern cities are more variable than cities in the south.

in Minneapolis increasingly go to the beach at temperatures higher than $20°C$. The same takes place for the two cities with a low variability. While in Honolulu people almost do not go to the beach at temperatures lower than $22°C$ in Miami the beach is already used at $\sim 6°C$. This analysis shows also the importance of taking regional specifics into account when incorporating weather features into a recommender algorithm.

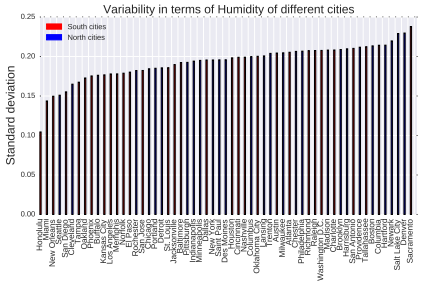
Variability Analysis. Another interesting analysis task is to find out if there are places that have more variable weather than other places. Places with a higher variability might also show a more distinctive weather to check-in behaviour. Variability analysis gives the possibility to identify regions with stable and unstable weather conditions to compare them later in terms of how susceptible they are to different check-in behaviours under contrasting weather conditions. With this information it is possible to ascertain if people living in regions with more weather fluctuation are more truncated to weather conditions or if they react more sensitive. Figure 4.12 reveals the basic phenomenon that weather is on average more variable in regions in the north of the USA. There are some cities having a high weather fluctuation in the south category ($< 39°5'$ latitude) but these are the northern cities of the south category. An interesting exception is precipitation intensity which has a higher inconsistency in the southern cities except the region

around the mojave desert in the south-west (Los Angeles, San Diego, Phoenix, etc.). The total weather variability shown in Figure 4.11 finally states apparently that in general northern Cities have a higher variability in terms of weather than southern ones. Especially the variability of Honolulu demonstrates very stable weather conditions. Total variability v_{total} was computed as stated in equation 4.6 where F is the set of the eight weather features and the max function returns the max variability of the given feature in all cities.

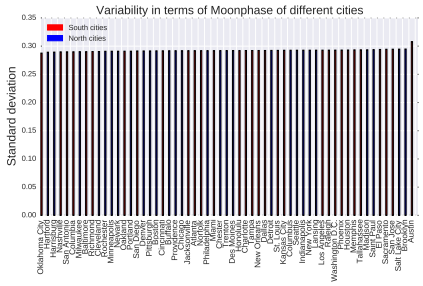
$$v_{total} = \frac{1}{|F|} \sum_{f \in F} \frac{f}{max(f)} \tag{4.6}$$



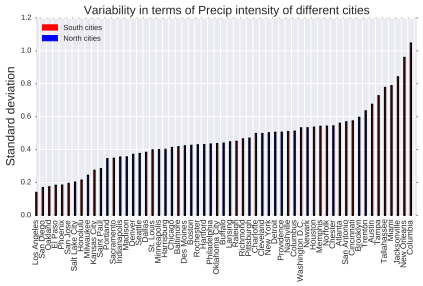
(a) Cloud cover



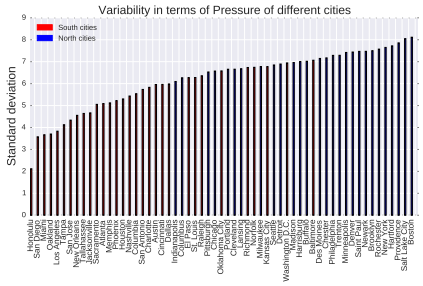
(b) Humidity



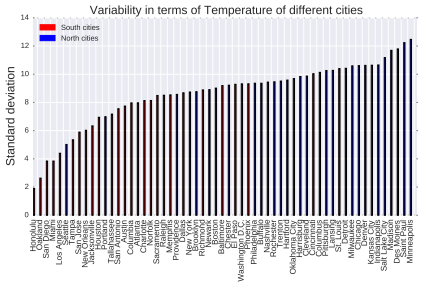
(c) Moonphase



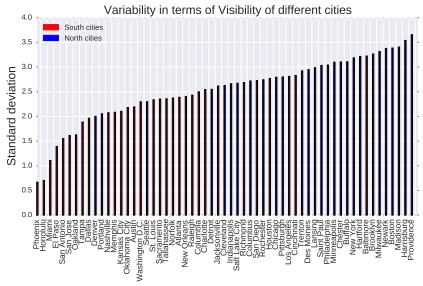
(d) Precipitation intensity



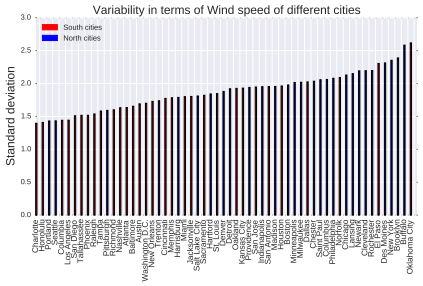
(e) Pressure



(f) Temperature



(g) Visibility



(h) Windspeed

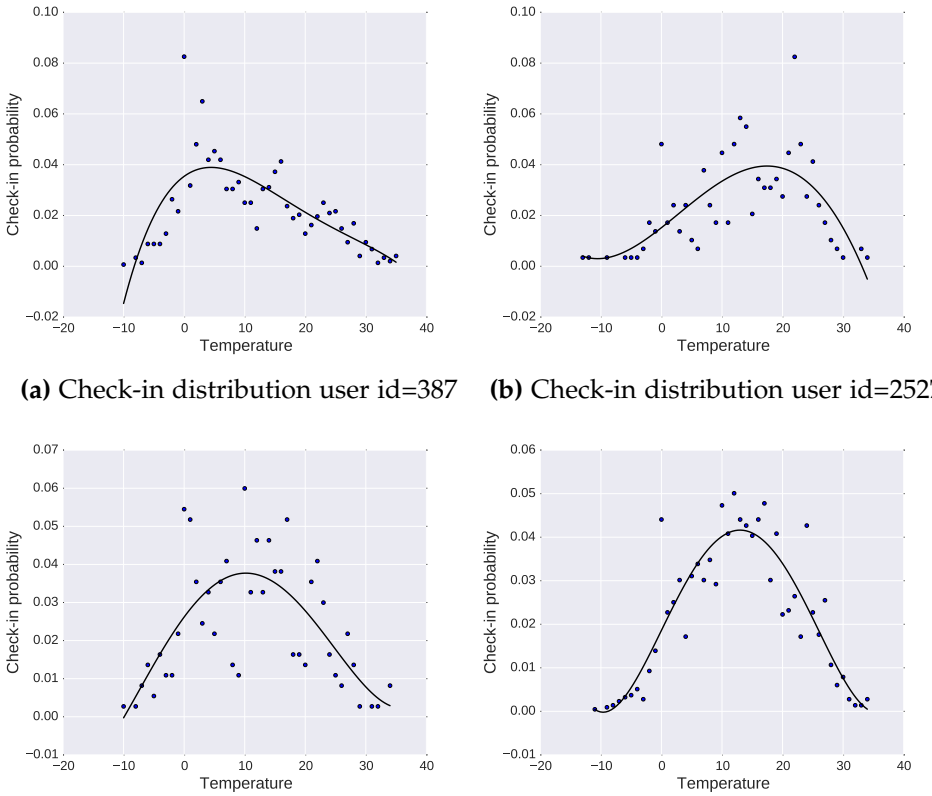
Figure 4.12: In General cities in the north of the USA (>39°5′ latitude) show more variability in weather features. Just precipitation intensity is more fluctuating in cities in the south.

4.2.5 User Analysis

This section tries to reveal if there is an impact of weather on the check-in behaviour of different users. On the one hand the aim is to find out if users have contrasting check-in behaviours among each other. In other words, the question is if there are users preferring some specific weather context or if the check-in distributions over weather features for each particular user are equally to the overall distributions shown in Figure 4.2 and 4.1. After finding weather sensitive users it is also interesting to check if this sensitive users have different category preferences.

Distribution Analysis. This section tries to reveal the differencies in the check-in behaviour of users. The assumption was that there are sensitive users that have preferences for particular values of a weather feature as for example temperature. Figure 4.13 demonstrates the check-in distributions of four users that checked-in in Boston. One can easily see in Figure 4.13a that the user with the id 387 prefers cold temperatures while the user with id 2527 prefers warm temperature as presented in Figure 4.13b. On the other hand the users with id 10400 and id 63 in Figure 4.13c and 4.13d follow a normal distribution as the overall temperature distribution also does (see Figure 4.2b). It can therefore be assumed that there are users that react sensitive to meteorological influences while others do not.

User Sensitivity Analysis. The last analysis tries to find out if users that prefer a specific type of weather also prefer particular categories. For this reason the users were split into groups of sensitive users. On the one side there are users having the preference of a low value of a specific weather feature and on the other side people that favor for high values of a specific weather feature. The goal is to identify if there exists a distinguishing check-in behaviour of that weather sensitive user groups. To discover a dissimilarity among that differing types of people a distribution over the users with the according mean of a weather feature and the appendant SE was plotted to display the different check-in attitudes users have. Additionally, a check-in distribution over categories of the p most sensitive users was made to compare the check-in behaviour of people that mostly check-in at low values versus people that mostly check-in at high values. Using this weather feature sensitive users, the different category preferences between this two groups of users should be pointed out. As an example Figure 4.14 presents the outcome for temperature and $p = 50\%$ splitting users exactly into two groups. The distribution plot definitely reaveals the difference between the users showing a significant difference between the two groups favouring for either high or low temperatures. The function over categories then demonstrates the check-ins in non similar categories between the two user types. The Chi-squared test described in Section 4.1 reveals the rejection of the hypothesis H_0



(a) Check-in distribution user id=387 **(b)** Check-in distribution user id=2527

(c) Check-in distribution user id=10400 **(d)** Check-in distribution user id=63

Figure 4.13: The distributions of different users in Boston. While the user with id 387 prefers cold weather, user with id 2527 prefers warm temperatures. On the other hand the user with id 10400 and with id 63 seem to not have any preferences and follow the general distribution.

with a p-value of < 0.001 that the two distributions are from the same origin. Figure 4.14 shows that users that mainly check-in at low temperatures favour for public transport places (as "Bus Station", "Train Station", "Metro Station") and in general indoor places like ("Donut Shop", "College Academic Building", "Bakery", "Bar", etc.) whereas people favouring for high temperatures often prefer outdoor places or food places that serve food from warmer regions (as "Road", "Mexican Restaurant", "Seafood Restaurant", etc.). User sensitivity analysis concerning other weather features can be seen in the appendix in Section B.1.

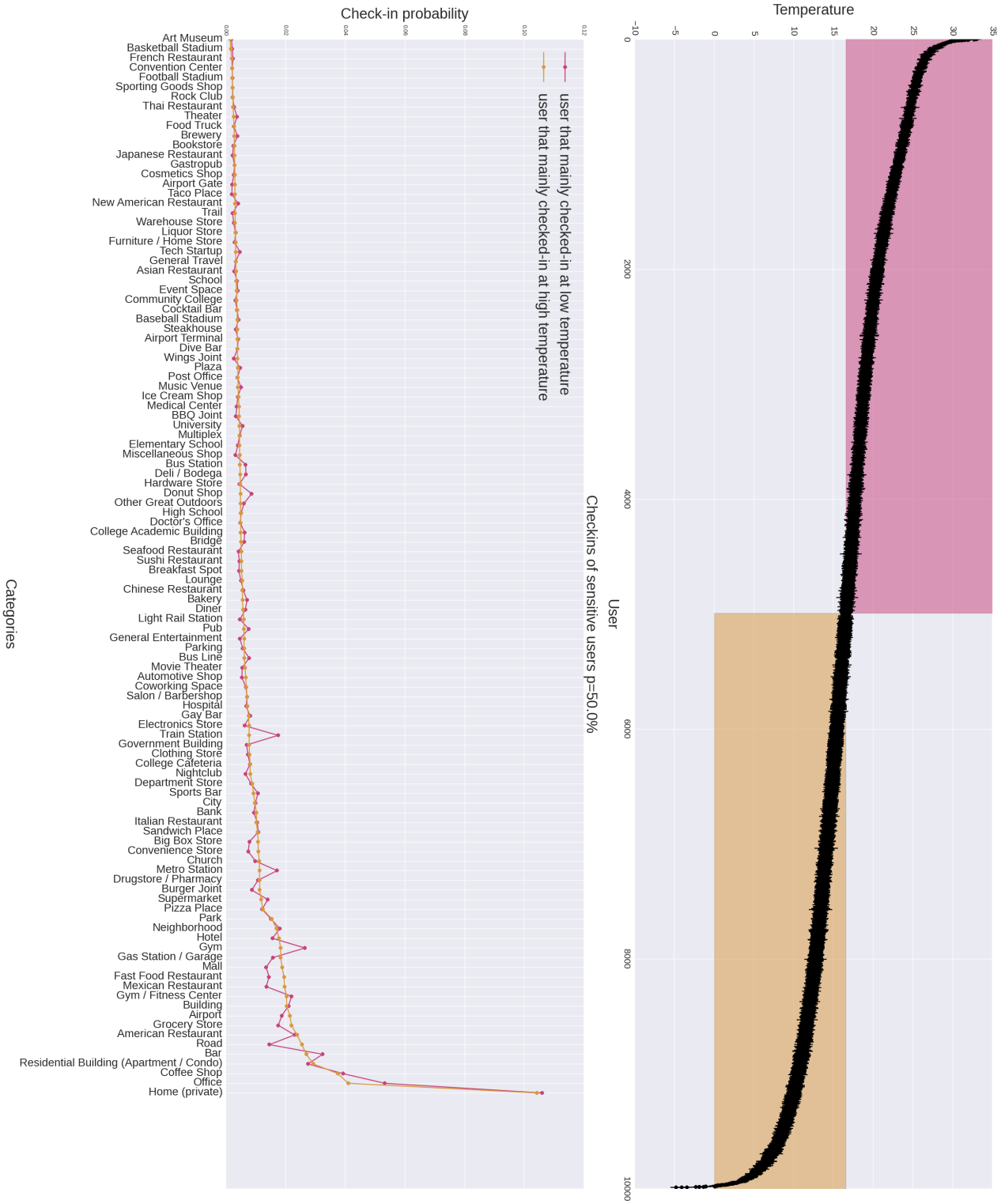


Figure 4.14: Temperature sensitive users and their behaviour. It is revealed that the 50% of users preferring warmer temperatures show a different check-in behaviour than those who prefer lower temperatures.“Gym”, for instance, is highly prefered by people that prefer lower temperatures while “Seafood Restaurant” is preferred by people favouring for higher temperatures.

5

Weather-Aware Recommender Analysis

One of the goals of this thesis is to incorporate weather context information into a recommender system with the aim to improve the predictions of this system. To answer the second research question **(RQ2)** this chapter mentions how weather context was incorporated into an existent recommender system. The improvements obtained by the incorporation are then stated in order to answer research question three **(RQ3)**. Finally, research question four **(RQ4)** is answered through measuring the individual performance increase of each weather feature separately.

5.1 Approach

The base method for the approach in this thesis was introduced by Li et al. (2015) and is called Rank-GeoFM. The approach is based on matrix factorization (see Section 5.2.1) with stochastic gradient descent (SGD) (see Section 5.1), an iterative learning approach to learn the model parameters. Table 5.1 shows the notations that are used in this Section to describe the implemented algorithm. Basically Rank-GeoFM incorporates geographical influence, the users' preference score and the impact of time in their model. Additionally, this thesis extends the algorithm with incorporating the impact of weather on the popularity of venues into the model. As shown in Chapter 4 weather has also an impact on the users' travel distance, the user itself and the popularity of categories what could be focus of future research.

User-Preference-Score. The authors of the Rank-GeoFM algorithm first introduced a method that models the users' preference rankings for POIs into the MF method. To obtain this user preferences an incompatibility function was created to measure incompatibility between the preference score y_{ul} obtained from the

Symbol	Description
U	set of users $u_1, u_2, ..., u_{\|U\|}$
L	set of POIs $l_1, l_2, ..., l_{\|L\|}$
FC_f	set of classes for feature f
X_{ul}	$\|U\|x\|L\|$ matrix containing the check-ins of users at POIs
X_{ulc}	$\|U\|x\|L\|x\|FC_f\|$ matrix containing the check-ins of users at POIs at a specific feature class c
D_1	user-POI pairs: $(u,l)\|x_{ul} > 0$
D_2	user-POI-feature class triples: $(u,l,c)\|x_{ulc} > 0$
$d(l,l')$	geo distance between the latitude and longitude of l and l'
W	geographical probability matrix of size $\|L\|x\|L\|$ where $w_{ll'}$ contains the probability of l' being visited after l has been visited according to their geographical distance
WI	probability that a weather feature class c is influenced by feature class c'
$N_k(l)$	set of k nearest neighbors of POI l
y_{ul}	the recommendation score of user u and POI l
y_{ulc}	the recommendation score of user u, POI l and weather feature class c

Table 5.1: The notations used to describe Rank-GeoFM and the incorporation of the weather context.

predictive model and x_{ul} the frequency of user u visiting location l. The assumption for the user preference score is that POI l is preferred over l' by user u, when the check-in frequency of l is higher than the one of l'. That means that if $x_{ul} > x_{ul'}$ user u favors for POI l. Having a function $I(a)$ returning one if a is true and zero otherwise, equation 5.1 measures the incompatibility between the true frequencies of user u and the predicted rankings of the factorization model. Parameter ϵ is introduced to soften ranking incompatibility.

$$Incomp(y_{ul}, \epsilon) = \sum_{l' \in L} I(x_{ul} > x_{ul'}) I(y_{ul} < y_{ul'} + \epsilon) \tag{5.1}$$

Therefore the incompatibility function counts the amount of locations $l' \in L$ that should have been ranked lower than l but are falsely ranked higher by the model and vice versa. Equation 5.3 shows the objective function over all pairs in D_1 to learn the parameters that minimizes the ranking incompatibility and 5.2 the according error function that converts the incompatibility into a loss. It can be seen that it is a smooth conversion because the weight of each incorrect ranked location is divided by $n+1$. That means, when iterating over the set D_1 or D_2 and having already n incorrect ranked items the $n+1th$ one just contributes $\frac{1}{n+1}$ to the overall loss. On the other hand the authors claim the ability to overcome data sparsity by learning also on the unvisited POIs of user u since $I(x_{ul} > x_{ul'})$ always determines to true on not visited POIs l' (whereby it always determines to false for $(u,l)|x_{ul} = 0$). Therefore, they contribute to the overall loss while they

are often ignored in conventional MF.

$$E(r) = \begin{cases} \sum_{i=1}^{r} \frac{1}{i} & if\, r > 0 \\ 0 & otherwise \end{cases} \tag{5.2}$$

$$\mathcal{O} = \sum_{(u,l) \in D_1} E(Incomp(y_{ul}, \epsilon) \tag{5.3}$$

Geographical Influence Score. The next step of Rank-GeoFM was to use the geographical distance between two locations as an indicator whether a user would go to POI l' when she already visited l. As one can see in Figure 4.4 the check-in distribution over distances follows a power law distribution and therefore one can assume that the probability of visiting POI l' decreases the farer away l' is from previous visited POI l. Similar studies from Ye et al. (2011b) and Balby Marinho et al. (2015), reveal the same behaviour and so the assumption of the authors of Rank Geo-FM was that nearby POIs tend to have a higher probability of being visited. Based on this discovery equation 5.4 shows the calculated probability of l' being visited after l whereby equation 5.5 shows the normalization of each row of the matrix to one to represent probabilities.

$$w_{ll'} = \begin{cases} (0.5 + d(l, l'))^{-1} & if\, l' \in N_k(l) \\ 0 & otherwise \end{cases} \tag{5.4}$$

$$\sum_{l' \in L} w_{ll'} = 1 \tag{5.5}$$

Weather Influence Score. To incorporate weather context into Rank-GeoFM, the weather features' values needed to be discretized. This was done to reduce data sparsity. For example, if one considers temperature as a real number, most of the check-ins concerning specific temperature values would probably be zero. Thus, transforming continuous values of weather features (e.g., temperature) into intervals might alleviate this problem. Hence, a mapping function is introduced (see Equation 5.6) that converts the weather features into interval bins. $|FC_f|$ defines the size of the bin for the current weather feature. Best results were obtained with $|FC_f| = 20$ (validated on held-out data).

$$c_f(value) = \left\lfloor \frac{value - min(f)}{(max(f) - min(f)) * (|FC_f| - 1)} \right\rfloor \tag{5.6}$$

Compute Recommendation Score. In order to create recommendations a function is introduced that calculates the recommendation score. As in conventional MF approaches latent model parameters are needed. These parameters are

proposed as matrices in a K dimensional space. For the purpose of incorporating user-preference-score into the recommendation score the matrices $U^{(1)} \in \mathbb{R}^{|U|xK}$ and $L^{(1)} \in \mathbb{R}^{|L|xK}$ are used to model the preferences of users to specific POIs. Including the geographical-influence-score into the model leads to the creation of the matrix $U^{(2)} \in \mathbb{R}^{|U|xK}$ that constitutes the geographical influence in the interaction between users and locations. As a last step the extension of the algorithm with weather context necessitates three more latent parameters (see Chapter 3.3 in Li et al. (2015)). The first is for incorporating weather-popularity-score that models whether a location is popular in a specific feature class or not and is named $L^{(2)} \in \mathbb{R}^{|L|xK}$. Furthermore, a matrix $L^{(3)} \in \mathbb{R}^{|L|xK}$ is introduced to model the influence between two feature classes. In other words $L^{(3)}$ is to soften the borders between the particular feature classes. The third latent parameter $F \in \mathbb{R}^{|FC_f|xK}$ then is used to parameterize the feature classes of the specific weather feature. Additionally, a Matrix $WI \in \mathbb{R}^{|FC_f|x|FC_f|}$ has been created that stores the probability that a weather feature class c is influenced by feature class c' which is computed as to see in equation 5.7.

$$wi_{cc'} = \frac{\sum_{u \in U} \sum_{l \in L} x_{ulc} x_{ulc'}}{\sqrt{\sum_{u \in U} \sum_{l \in L} x_{ulc}^2} \sqrt{\sum_{u \in U} \sum_{l \in L} x_{ulc'}^2}} \tag{5.7}$$

Having this latent parameters defined recommendation score for user u POI l and feature class c is then computed as shown in equation 5.8.

$$\begin{aligned} y_{ul} &= u_u^{(1)} * l_l^{(1)} + u_u^{(2)} * \sum_{l' \in N_k(l)} w_{ll'} l_{l'}^{(1)} \\ y_{ulc} &= y_{ul} + f_c * l_l^{(2)} + l_l^{(3)} * \sum_{c' \in F} wi_{cc'} f_{c'} \end{aligned} \tag{5.8}$$

One can see that this step converts the algorithm into a context aware recommender as explained in Section 2.4 since the class of the weather feature at the point in time where the recommendation has to be made is taken into consideration. In other words the current weather context has an influence on the recommendation score.

Learning Approach.

Stochastic gradient descent The standard gradient descent (GD) algorithm is an iterative algorithm that updates the latent parameters in each iteration with the objective of minimizing the error function of the training dataset (Bottou, 2010). In other words the algorithm minimizes an objective function J with parameters Θ by updating them in the negative direction of the gradient of J. Equation 5.9 shows the update of the parameters with the learning rate α for

each training example x, y.

$$\Theta = \Theta - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_{\Theta} J(x_i, y_i, \Theta) \tag{5.9}$$

This approach is computational intensive for large samples since the gradient has to be computed over the whole dataset for every update. For this reason SGD approximates the true gradient with the gradient of each training example and performs an update on each training example. Equation 5.10 demonstrates the update of the parameters with the simplified gradient of one example.

$$\Theta = \Theta - \alpha \nabla_{\Theta} J(x_i, y_i, \Theta) \tag{5.10}$$

Although the simplification leads to a noisy representation of the gradient due to the stochastic process of randomly chosen examples at each iteration SGD should behave like GD while having performance advantages on large scale datasets. Asymptotic analysis in Bottou (2010) has shown that the "time to accuracy" ρ decreases from $nlog\frac{1}{\rho}$ to $\frac{1}{\rho}$ and is therefore independent of sample size n. As a result of the high update frequency usually the learning rate has to be chosen very small.

Learning Model Parameters. To learn the latent model parameters defined as $\Theta = \{L^{(1)}, L^{(2)}, L^{(3)}, U^{(1)}, L^{(2)}, F\}$ in the previous paragraph, a learning algorithm has to be proposed in order to find the latent parameters that minimize the objective function $\mathcal{O}$ in equation 5.3 that sums up the loss for each training triple $(u, l, c) \in D_2$. Li et al. (2015) state two issues for optimizing the objective function:

- $E(Incomp(y_{ul}, \epsilon))$ is not differentiable due to its non-continuousness
- Calculating $Incomp(y_{ul}, \epsilon)$ is computational intensive

They first introduce a continuous approximation of $E(Incomp(y_{ul}, \epsilon)$ with approximating the non-continuous indicator function $I(a)$ with a continuous sigmoid function $s(a) = \frac{1}{1+exp(-a)}$ which leads to the computation of the stochastic gradient of E w.r.t Θ in equation 5.12 with $\delta_{ull'}$ defined in equation 5.11.

$$\delta_{ull'} = s(y_{ul'} + \epsilon - y_{ul})(1 - s(y_{ul'} + \epsilon - y_{ul}) \tag{5.11}$$

$$\begin{aligned} &\frac{\partial E(Incomp(y_{ul}, \epsilon))}{\partial \Theta} \\ &\approx E(Incomp(y_{ul}, \epsilon)) \frac{\sum_{l' \in L} I(x_{ul} > x_{ul'}) \frac{\partial s(y_{ul'} - y_{ul})}{\partial \Theta}}{Incomp(y_{ul}, \epsilon)} \\ &= E(Incomp(y_{ul}, \epsilon)) \frac{\sum_{l' \in L} I(x_{ul} > x_{ul'}) \delta_{ull'}}{Incomp(y_{ul}, \epsilon)} \end{aligned} \tag{5.12}$$

Nevertheless, computation of $Incomp(y_{ul}, \epsilon)$ is still very intense. For this reason they invent a fast learning scheme that estimates $Incomp(y_{ulc}, \epsilon)$ with sampling. Since just incorrectly-ranked POIs count to the loss, the idea is to sample POIs and calculate incompatibility until the first POI is incorrectly ranked, with n being the number of sampled POIs at this point. The gradient for one incorrectly ranked POI is then denoted in equation 5.13 which approximates equation 5.12 and disperses the summation.

$$\frac{\partial \overline{E}}{\partial \Theta} = E(Incomp(y_{ul}, \epsilon))\delta_{ull'} \frac{\partial (y_{ul'} + \epsilon - y_{ul})}{\partial \Theta} \tag{5.13}$$

Then n is the number of sampled POIs before the incorrect one and $\frac{1}{Incomp(y_{ul}, \epsilon)}$ the chance of each POI to be the incorrect one. The more incorrect POIs exist the higher n will be on average. Therefore it can be assumed that n follows a geometric distribution dependent on parameter $p = \frac{Incomp(y_{ul}, \epsilon)}{|L|}$. Using the fact that the expectation of a geometrical distribution with parameter $p = \frac{1}{p}$ and $n \approx \left\lfloor \frac{1}{p} \right\rfloor = \left\lfloor \frac{|L|}{Incomp(y_{ul}, \epsilon} \right\rfloor$ the incompatibility function can be estimated with $Incomp(y_{ul}, \epsilon) \approx \left\lfloor \frac{|L|}{n} \right\rfloor$ leading to the computation of the gradient of one incorrectly-ranked POI as shown in equation 5.14.

$$\frac{\partial \overline{E}}{\partial \Theta} \approx E\left(\left\lfloor \frac{|L|}{n} \right\rfloor \right) \delta_{ull'} \frac{\partial (y_{ul'} + \epsilon - y_{ul})}{\partial \Theta} \tag{5.14}$$

This estimation follows the intuition that if for example the third POI was incorrectly-ranked it is expected that $Incomp(y_{ul}, \epsilon) = \frac{|L|}{3}$. Using γ as learning rate the update of the parameters Θ is then calculated as follows:

$$\Theta \leftarrow \Theta - \gamma \frac{\partial \overline{E}}{\partial \Theta} \tag{5.15}$$

Using the estimation of the incompatibility function it turns out that the complexity reduces from $O(K|L|k)$ to $O(Knk)$ whereby n in the beginning will be very small because the parameters are not well fitted to the training data and an incorrect POI will be obtained very fast. It will then grow a bit during the training process. Nevertheless, it is not expected that every item will be ranked correctly so n will be still smaller than $|L|$ when the algorithm converges. Li et al. (2015) have shown that the fast learning scheme was able to iterate 1000 times in 19 hours while equation 5.12 accomplished 5 iterations in the same time.
To avoid overfitting the authors added a constraint to the adjustments of the latent factors at each step of the learning process. They introduced a hyperparameter C that regularizes the magnitude of the latent factors as shown in equation 5.16 to equation 5.21. Additionally, the regularization terms for geographical influence and weather context influence are kept in a smaller value αC and βC to balance

the contributions of this two influence factors.

$$\left\|u_u^{(1)}\right\|_2 \leq C \xrightarrow[regularization]{} u_u^{(1)} \leftarrow C\frac{u_u^{(1)}}{\left\|u_u^{(1)}\right\|_2}, \quad u = 1, 2, ..., |U| \tag{5.16}$$

$$\left\|u_u^{(2)}\right\|_2 \leq \alpha C \xrightarrow[regularization]{} u_u^{(2)} \leftarrow \alpha C\frac{u_u^{(2)}}{\left\|u_u^{(2)}\right\|_2}, \quad u = 1, 2, ..., |U| \tag{5.17}$$

$$\left\|l_l^{(1)}\right\|_2 \leq C \xrightarrow[regularization]{} l_l^{(1)} \leftarrow C\frac{l_l^{(1)}}{\left\|l_l^{(1)}\right\|_2}, \quad l = 1, 2, ..., |L| \tag{5.18}$$

$$\left\|l_l^{(2)}\right\|_2 \leq \beta C \xrightarrow[regularization]{} l_l^{(2)} \leftarrow \beta C\frac{l_l^{(2)}}{\left\|l_l^{(2)}\right\|_2}, \quad l = 1, 2, ..., |L| \tag{5.19}$$

$$\left\|l_l^{(3)}\right\|_2 \leq \beta C \xrightarrow[regularization]{} l_l^{(3)} \leftarrow \beta C\frac{l_l^{(3)}}{\left\|l_l^{(3)}\right\|_2}, \quad l = 1, 2, ..., |L| \tag{5.20}$$

$$\|f_c\|_2 \leq \beta C \xrightarrow[regularization]{} f_c \leftarrow \beta C\frac{f_c}{\|f_c\|_2}, \quad c = 1, 2, ..., |FC_f| \tag{5.21}$$

Weather-Aware POI Recommender System (WPOI). This paragraph describes the incorporation of the weather context into the Rank Geo-FM algorithm. First of all, the hyperparameters have to be initialized. As in Li et al. (2015) a paramater tuning for the hyperparameters $\alpha, k, \beta, andK$ was performed. It turned out that changing K does not change the performance of the algorithm so it was set to 100. The size of the neighborhood k showed best results at 300. Furthermore, tuning the regularization terms α and β for geographical and time influence had the best accuracy at .2. As shown in Algorithm 1, a loop over the set D_1 of check-ins containing each (u, l) tuple is made. Line 3 to line 16 reveal the original Rank Geo-FM algorithm as proposed in Li et al. (2015) iterating over all pairs (u, l) and adjusts the latent parameters accordingly. After that Line 17-30 show the incorporation of the weather context into the base Rank-GeoFM approach. In order to adjust the latent parameters to the according weather context an iteration over all <*user, venue, feature-class*> triples $(u, l, c) \in D_2$ was introduced

Algorithm 1: Pseudo Code for the Weather-Aware POI Recommender System (WPOI), implemented in this thesis.

Input: check-in data D_1 and D_2, geographical influence matrix W, weather influence matrix WI, hyperparameters $\epsilon, C, \alpha, \beta$ and learning rate γ_g and γ_w

Output: parameters of the model $\Theta = \{L^{(1)}, L^{(2)}, L^{(3)}, U^{(1)}, U^{(2)}, F\}$

init: Initialize Θ with $\mathcal{N}(0, 0.01)$; Shuffle D_1, D_2 randomly
repeat
  for $(u, l) \in D_1$ **do**
    Compute y_{ul} as Equation 5.8 and set n = 0
    repeat
      Sample a POI l', Compute $y_{ul'}$ as in equation 5.8 and set n++
    until $I(x_{ul} > x_{ul'})I(y_{ul} < y_{ul'} + \epsilon) = 1$ *or* $n > |L|$
    if $I(x_{ul} > x_{ul'})I(y_{ul} < y_{ul'} + \epsilon) = 1$ **then**
      $\eta = E\left(\left\lfloor \frac{|L|}{n} \right\rfloor\right)\delta_{ull'}$
      $g = \left(\sum_{l^* \in N_k(l')} w_{ll^*} l_{l^*}^{(1)} - \sum_{l^+ \in N_k(l)} w_{ll^+} l_{l^+}^{(1)}\right)$
      $u_u^{(1)} \leftarrow u_u^{(1)} - \gamma_g \eta (l_{l'}^{(1)} - l_l^{(1)})$
      $u_u^{(2)} \leftarrow u_u^{(2)} - \gamma_g \eta g$
      $l_{l'}^{(1)} \leftarrow l_{l'}^{(1)} - \gamma_g \eta u_u^{(1)}$
      $l_l^{(1)} \leftarrow l_l^{(1)} + \gamma_g \eta u_u^{(1)}$
    end
  end
  for $(u, l, c) \in D_2$ **do**
    Compute y_{ulc} as Equation 5.8 and set n = 0
    repeat
      Sample a POI l' and feature class c', Compute $y_{ul'c'}$ as in equation 5.8 and set n++
    until $I(x_{ulc} > x_{ul'c'})I(y_{ulc} < y_{ul'c'} + \epsilon) = 1$ *or* $n > |L|$
    if $I(x_{ulc} > x_{ul'c'})I(y_{ulc} < y_{ul'c'} + \epsilon) = 1$ **then**
      $\eta = E\left(\left\lfloor \frac{|L|}{n} \right\rfloor\right)\delta_{ull'}$
      $g = \left(\sum_{c^* \in FC_f} wi_{cc^*} f_{c*} - \sum_{c^+ \in FC_f} wi_{cc^+} f_{c+}\right)$
      $f_c \leftarrow f_c - \gamma_w \eta (l_{l'}^{(2)} - l_l^{(2)})$
      $l_l^{(3)} \leftarrow l_l^{(3)} - \gamma_w \eta g$
      $l_{l'}^{(2)} \leftarrow l_{l'}^{(2)} - \gamma_w \eta f_c$
      $l_l^{(2)} \leftarrow l_l^{(2)} + \gamma_w \eta f_c$
    end
  end
  Project updated factors to accomplish constraints
until *convergence*
return $\Theta = \{L^{(1)}, L^{(2)}, L^{(3)}, U^{(1)}, U^{(2)}, F\}$

to

to adjust the latent parameters on the incorrect ranked venues according to the specific weather context. That has to be done because the algorithm might rank a triple (u, l, c) correctly with no need for adjustments where on the other hand (u, l, c') might be ranked incorrectly what would involve a parameter adjustment. The modifications of are then done accordingly to the base algorithm. The latent parameters having the best performance on the validation-set are then returned to be validated on the test set. During our studies it turned out that adjusting the latent parameters of the weather context with a learning rate $\gamma = .0001$ as used in Li et al. (2015) was to high and the algorithm did not find the minimum so we decided to introduce learning rate $\gamma_w = .00001$ for the weather context parameters where a stable learning rate was achieved as one can see in Figure 5.1.

5.2 Evaluation Methodology

5.2.1 Baselines

As mentioned in Ricci et al. (2011), it is a common approach to test a new recommender against the state-of-the-art. The procedure is to test the newly developed model on the same dataset as the already existing one and then to compare the accuracy of them. In this thesis some well known recommender systems build the baseline. On the one hand an algorithms from the MF family was chosen since MF is getting more and more popular in the recent years. On the other hand the Most Popular (MP) Approach should build the very baseline to compare with. Finally, User-Based KNN (User-KNN) was taken as a CF approach.

Most Popular

MP is a very basic recommender algorithm that does not personalize recommendations on the users' past behaviour. Instead, solely the popularity of an item serves as weight. Specifically it always returns an item list, ranked by how often the item was consumed in the past. Equation 5.22 shows the calculation of prediction p of item l in the set of check-ins L.

$$p(l) = |L(l)| \tag{5.22}$$

Collaborative Filtering

CF algorithms are based on the assumption that items and user of a system interrelate. The hypothesis is that if two items l_1 and l_2 have a high correlation of users that consumed the items then the probability that a user who consumed l_1, likes also l_2 is higher. There are different algorithms differing in the way how to

compute the correlation and building the correlation basis either on the users or on the items.

Item-Based KNN. The item-based KNN algorithm grounds on the hypothesis that if two items l_1 and l_2 are similar a user that liked l_1 will also like l_2. This raises the question of how to compute this similarity. Wen (2008) define a similarity function based on the adjusted cosine similarity as described in Sarwar et al. (2001a) which also takes the different average ratings of users into account. Having a rating matrix $R \in \mathbb{R}^{|U|x|L|}$ and an average user rating $\overline{R_u}$ the similarity between l_1 and l_2 is then defined as follows:

$$sim(l_1, l_2) = \frac{\sum_{u \in U(l_1) \cap U(l_2)} (R_{l_1 u} - \overline{R_u})(R_{l_2 u} - \overline{R_u})}{\sqrt{\sum_{u \in U(l_1) \cap U(l_2)} (R_{l_1 u} - \overline{R_u})^2 \sum_{u \in U(l_1) \cap U(l_2)} (R_{l_2 u} - \overline{R_u})^2}} \tag{5.23}$$

At this point it has to be said that there exist several other similarity measures, such as the Pearson correlation or the Spearman rank correlation. Using adjusted cosine similarity a prediction of a user and an item can be made. The prediction for user u and item l can be seen in equation 5.24 where N_u^K are the K most similar items to l_1 that are already rated by user u. Therefore it states the average similarity of the new item l_1 to the already rated items of the user.

$$y_{ul_1} = \frac{\sum_{l_2 \in N_u^K(m)} sim(l_1, l_2) R_{l_2 u}}{\sum_{l_2 \in N_u^K(m)} |sim(l_1, l_2)|} \tag{5.24}$$

A sorted list of the predictions of all items gives the top-N recommendations for user u.

User-Based KNN. In contrast to the item-based knn algorithm the user-based knn computes similarities between users and not items. The computation of $sim(l_1, l_2)$ is then computed over the rows instead of columns of the rating matrix. With $\overline{R_l}$ as the average rating of item l similarity between users is computed in the following manner:

$$sim(u, v) = \frac{\sum_{l \in L(u) \cap L(v)} (R_{ul} - \overline{R_l})(R_{vl} - \overline{R_l})}{\sqrt{\sum_{l \in L(u) \cap L(v)} (R_{ul} - \overline{R_l})^2 \sum_{l \in L(u) \cap L(v)} (R_{vl} - \overline{R_l})^2}} \tag{5.25}$$

Using the similarity computation between users, the prediction for item l and user u is defined in equation 5.26 where $N_l^K(u)$ denotes the K most similar users to u that already rated item l.

$$y_{ul} = \frac{\sum_{v \in N_l^K(u)} sim(u, v) R_{vl}}{\sum_{v \in N_l^K(u)} |sim(u, v)|} \tag{5.26}$$

Again having this predictions the top-N ones represent the recommendations for user u.

Matrix Factorization

In recommender systems there is a set of users U and a set of items L. Matrix R of size $|U|x|L|$ shows how user u rated item l. One has to keep in mind, that this ratings do not have to be implicit feedback of the user. Alternatively the rating value can be given by the amount user u used item l. In case of a POI recommender it could be the number of visits of user u in location l. Having 4 items and 5 users the ratings may look like in table 5.2. The goal of a recommender system is to fill the hyphens in the table with values that match the intent of the user given a specific item. In other words the goal is to predict the blanks in the table. The assumption of MF described in Takács et al. (2008) is the existence of

	l1	**l2**	**l3**	**l4**
u1	1	-	-	1
u2	5	2	-	4
u3	4	-	1	3
u4	4	2	-	-
u5	-	2	4	5

Table 5.2: Rating Matrix R shows how user u rated item l. Goal of the recommender system is to fill the hyphens with the correct ratings.

latent parameters storing the features of users and items. The ambition is to learn this latent features from the existing data in order to be able to fill the blanks in the rating Matrix.
Having two matrices $X \in \mathbb{R}^{|U|xK}$ and $Y \in \mathbb{R}^{|L|xK}$, storing the K latent features of users and items, the rating matrix R can be approximated by the product of this latent feature matrices as shown in equation 5.27.

$$R \approx XxY^T = \hat{R} \tag{5.27}$$

Using equation 5.28 the rating for user u and item l is obtained by:

$$\hat{r}_{u,l} = X_l^T Y_j \tag{5.28}$$

The most important step in the MF approach is to train X and Y in a way that optimizes the error between the calculated rating $\hat{r}_{ul}$ and the actual rating obtained from training data T as defined in equation 5.29.

$$e_{u,l} = r_{ul} - \hat{r}_{ul} \quad u,l \in T \tag{5.29}$$

Therefore the optimization task is to find the ideal matrices Q and P that minimize the root-mean-square error defined in equation 5.30.

$$RMSE = \sqrt{\frac{\sum_{u,l \in T} e_{u,l}^2}{|T|}} \tag{5.30}$$

To solve the optimization of the cost function for optimal $X = X^*$ and $Y = Y^*$ determined in equation 5.31 the derivative of the cost function has to be taken for each latent feature $k \in [1, ..., K]$.

$$(X^*, Y^*) = \underset{X,Y}{\operatorname{argmin}} SSE = RMSE \tag{5.31}$$

Equation 5.32 shows the computation of the gradient for training sample u, l with respect to the variables x_{uk} and y_{kl}.

$$\begin{aligned} \frac{\partial e_{ul}^2}{\partial x_{uk}} &= -2e_{ul}y_{kl} \\ \frac{\partial e_{ul}^2}{\partial y_{kl}} &= -2e_{ul}x_{uk} \end{aligned} \tag{5.32}$$

Using the computation of the gradient x'_{uk} and y'_{kl} the values are accordingly updated to the descent of the gradient as shown in equation 5.33. Introducing a learning rate η protects the descent from to high jumps and a resulting miss of the absolute minimum.

$$\begin{aligned} x'_{uk} &= x_{uk} + 2\eta e_{ul}y_{kl} \\ y'_{kl} &= y_{kl} + 2\eta e_{ul}x_{uk} \end{aligned} \tag{5.33}$$

This updates are made iteratively until the error converges to its minimum. In recent years research brought a lot of enhancements to this basic MF approach. Among others regularization terms were introduced to avoid overfitting of the training data and different ways how to handle the input data were proposed. One of this approaches is described in detail in the following paragraph.

Weighted Regularized Matrix Factorization (WRMF). Hu et al. (2008) and Pan et al. (2008) introduced a model using implicit user feedback. They define a set of variables p_{ul} for each user u and each item l. Based on the implicit feedback r_{ul}, p_{ul} is computed as follows:

$$p_{ul} = \begin{cases} 1 & r_{ul} > 0 \\ 0 & r_{ul} < 0 \end{cases} \tag{5.34}$$

The result of this conversion is a binary feedback data with one signalling that the user has a preference for the item and zero signalling no preference. Working with binary data includes a huge amount of uncertainty since the user may have visited a location or watched a movie but did not like it. To avoid this uncertainty the authors introduced a confidence weight for each r_{ul} computed as shown in equation 5.35.

$$c_{ul} = 1 + \alpha r_{ul} \tag{5.35}$$

That means the higher the positive feedback user u gives to item l the higher the confidence weight of u and l. Thus the error function to optimize is defined in

equation 5.36.

$$\min_{x_*, y_*} \sum_{u,l} c_{ul}(p_{ul} - x_u^T y_l)^2 + \lambda(\sum_u \|x_u\|^2 + \sum_l \|y_l\|^2) \tag{5.36}$$

Additional to the basic MF approach described in section 5.2.1 the authors added a regularizing term to avoid overfitting to the training data by regularizing the magnitudes of X and Y. Learning the latent parameters is done by alternating-least-squares optimization process described in Pan et al. (2008). This approach adjusts the user parameters in the first iteration and the item parameters in the second which simplifies the computation of the minimum of the error function. This steps are then taken in an alternating manner until the error function converges to its minimum.

5.2.2 Evaluation Protocol

To bypass the data sparsity problem, users and venues having less than 20 check-ins were removed for measuring the performance of the algorithm. Then the dataset was split into training, test and validation set through adding the first 70% of the check-ins of each user to the training-set, the following 20% to the test-set and the rest to the validation-set. The training-set was then used to train the latent model parameters. During the training phase of the algorithm the validation-set was used to find out when the algorithm converges. As one can see in Figure 5.1 WPOI for precipitation intensity converged at $\approx 5,000$ iterations so the latent parameters at this point were taken to then compute the performance on the test set. Evaluating a context aware algorithm implicates having just one correct item that can be recommended for the current context. Due to this fact evaluation over the test dataset is done for each (u, l, c) triple separately. Therefore we chose NDCG@k (Normalized Discounted Cumulative Gain) with $k = 20$ to measure the performance of the recommender because this measure not only takes into consideration if the correct item was in the top-k result-list but also at which rank it appears in the list. As already mentioned in Section 4.2.4, it is important to bypass the problem of different climatic region when incorporating weather information in a recommender algorithm. Therefore four cities, each of them located in one of the cardinal directions, were taken to cover the different climatic regions of the USA. Table 5.3 shows the statistics of the four cities.

5.2.3 Evaluation Metric

To be able to compare different algorithms with each other different measures are available measuring the quality of a recommender. Typically an evaluation metric sets the relevance of an item (e.g. a location) given by the recommender engine in relation to its actual relevance given by the test dataset. For this issue

City	#Users	#Venues	#Check-ins	Sparsity
Minneapolis	436	797	37,737	89.1%
Boston	637	1141	42,956	94.3%
Miami	410	796	29,222	91.0%
Honolulu	173	410	16,042	77.4%

Table 5.3: Basic statistics of the cities.

the following measure was mainly used in this thesis. Further metrics used in further studies are described in the appendix.

Normalized Discounted Cumulative Gain (NDCG). NDCG is a metric that assigns relevance to items. That means a venue that is more applicable for a user has a higher relevance score than a venue with less importance. The assigned relevance is then summarized. This represents the cumulative gain part of the metric. To penalize high relevant items that appear later in the result list the relevance value is reduced by dividing it by the logarithmic scaled position of the item in the result list (see equation 5.37).

$$DCG = rel_1 + \sum_{i=0}^{p} \frac{rel_i}{log_2(i)} \tag{5.37}$$

Therefore a relevant item getting a low rank from the algorithm is divided by a higher number what leads to a lower cumulative gain and a worse result. Equation 5.38 shows the normalization of DCG by the DCG calculated with the correct ranked list also known as ideal DCG (IDCG).

$$NDCG = \frac{DCG}{IDCG}, NDCG \in [0,1] \tag{5.38}$$

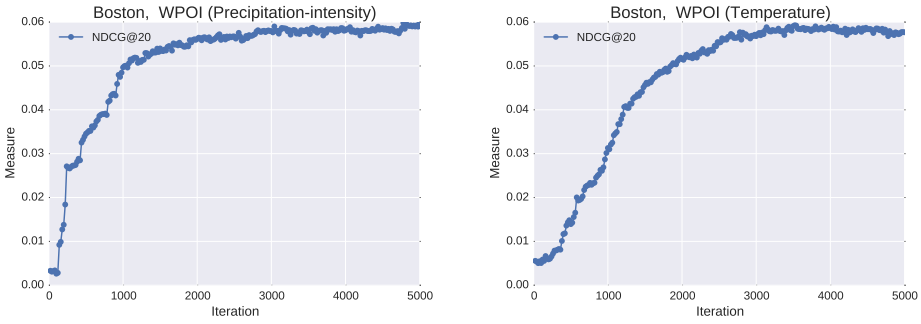
Because of its ranking position sensitivity NDCG was mainly used to compare the performance of the algorithms in this thesis.

5.3 Results

This Chapter provides the results of the performance of the recommender algorithm tested in the offline experiment. First, the learning of the latent parameters is shown. Afterwards the performance of WPOI is compared with the original Rank-GeoFM algorithm followed by the comparison with some well known recommender algorithms.

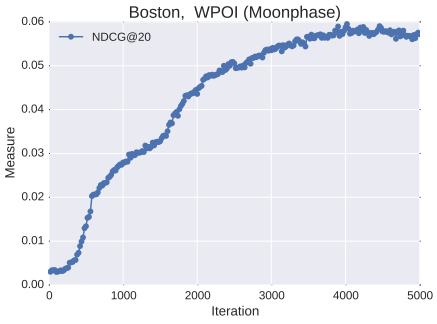
5.3.1 Parameter Learning

Figure 5.1 shows the iterative learning approach of the recommender algorithm. After adjusting the learning rate for the weather context to $\gamma_w = .00001$ the learning rate got stable. Depending on the size of the city convergence of the algorithm was accomplished at $\approx 3,000 - 5,000$ iterations. Learning the parameters for the model in Boston took around 22 hours for the biggest city Boston and 5,000 iterations whereby convergence was often already achieved at $\approx 3,000$ iterations with only little performance increases afterwards. In real word applications one will have to weigh that little performance increases up with the time needed to compute the remaining 2,000 iterations. It also has to be mentioned, that the different weather features show different learning behaviours. On the one hand precipitation intensity in Figure 5.1a gains its major performance increase in the first 1,000 iterations where on the other hand moonphase in Figure 5.1c needs $\approx 2,000$ to reach the same performance level.

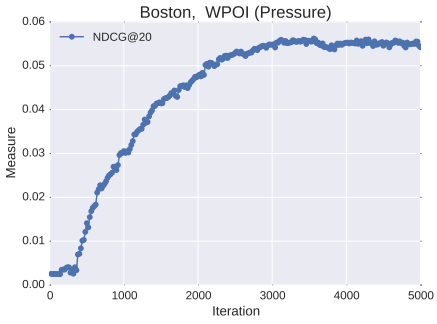


(a) Precipitation intensity

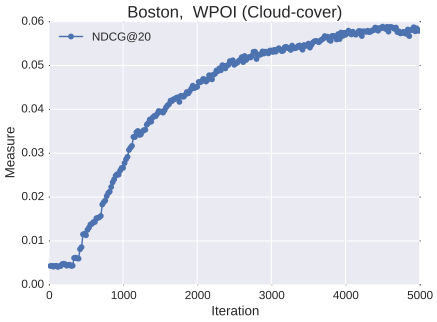
(b) Temperature



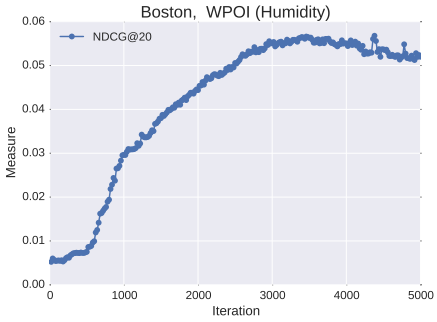
(c) Moonphase



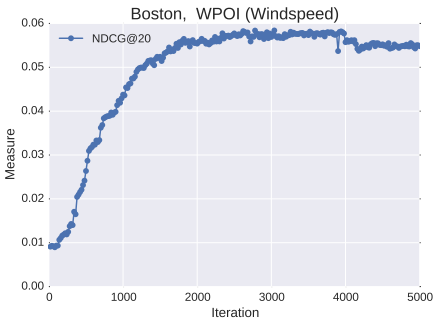
(d) Pressure



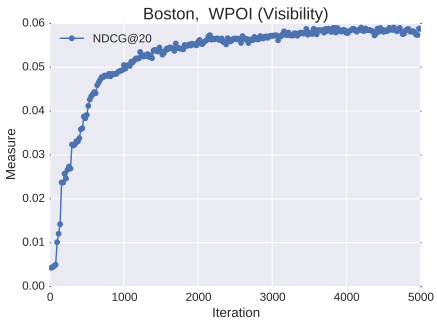
(e) Cloud cover



(f) Humidity



(g) Windspeed



(h) Visibility

Figure 5.1: Learning of the latent parameters of the prediction model of the eight different features in Boston. It shows a stable learning rate and a convergence at $\approx 3,000 - 5,000$ iterations.

5.3.2 Performance of WPOI vs. Rank Geo-FM

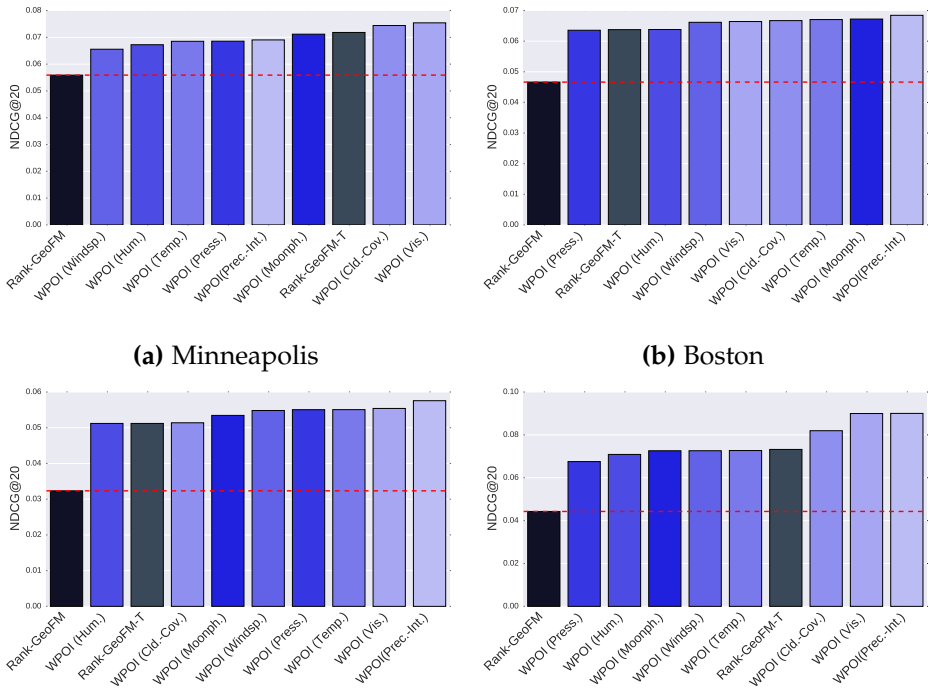
To measure the individual power of each of the eight weather features the approach was to compare the performance against the base algorithms implemented in Li et al. (2015). On the one hand Rank-GeoFM represents the "user-POI setting" with geographical influence and user-preference-score incorporated while on the other hand Rank-GeoFM-T represents the "user-time-POI setting" with additional time information. It's readily to see in Figure 5.2 that the eight weather features significantly outperform the Rank-GeoFM approach. Significance level was tested with a standard t-test showing a p-value always smaller than $p < .001$ in all cities and on all weather features.

What is even more interesting to note is the performance of Rank-GeoFM-T that utilizes the variable of time as contextual factor. As highlighted, in all cases, WPOI with weather features, such as visibility and precipitation intensity, outperforms the time-based variant Rank-GeoFM-T, showing the need to investigate the weather context in more detail in future work.

What also stands out is the fact that certain weather features perform better than others and this pattern seems to be city dependent. For example, in Honolulu (Figure 5.2d) the best performing feature is precipitation intensity, while in Minneapolis (Figure 5.2a) visibility seems to work the best among all investigated weather features. Similar patterns can be observed for other features, such as temperature or cloud cover, changing their relative importance across the four cities. The reason for temperature being not that strong in Honolulu might be shown in Figure 4.12f where Honolulu shows up with a low temperature variability. Furthermore, Figure 4.10c shows the small temperature range on the island. These observations are in line with the results in Figure 4.1, showing a strong tendency of check-ins into POIs under certain weather conditions.

However, one of the most interesting result is the good performance of the moonphase feature, which appeared to be uniformly distributed in general (see Figure 4.1). Hence, it appears, that at the level of locations, there is indeed a strong preference for check-ins in different phases of the moon (however to confirm that, further investigations are needed).

Finally, the most interesting effect of this performance analysis is that cities, such as Honolulu and Miami, where overall weather variability is low, tend to be easier predictable than cities having a high weather variability, such as Minneapolis and Boston. This leads to the assumption that people living in or close to the tropical climate zone tend to be more sensitive in terms of weather changes and the impact on users' decisions about visiting a POI or not is high. On the other side people in the north are used to climatological changes over the year and do not base their mobility behaviours on climatological changes. In other words they are more resistant to weather and therefore harder to predict.



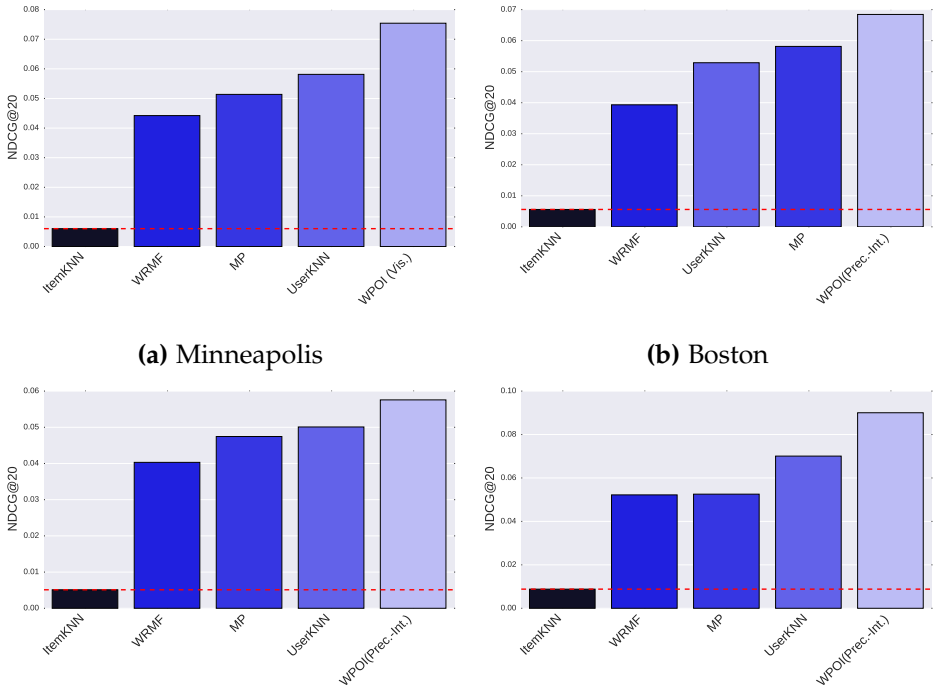
(a) Minneapolis **(b)** Boston

(c) Miami **(d)** Honolulu

Figure 5.2: Comparison of the eight weather features with Rank-GeoFM and Rank-GeoFM-T in the four cities. Rank-GeoFM is outperformed by weather features significantly. Furthermore, this analysis leads to the assumption that weather has more impact on people living closer to the tropical zone (Miami, Honolulu) than on people living closer to the boreal zone (Minneapolis, Boston)

5.3.3 Performance of WPOI vs. Baseline Algorithms

In addition to the tests against the basis algorithm a performance test against some well known recommender algorithms as described in Section 5.2.1 was conducted. As to see in Figure 5.3 MP showed up to be a very strong baseline. Especially in Boston it outperformed the MF and the CF approach. As already shown in Noulas et al. (2012) user based CF builds a very strong baseline in the field of POI recommender what is also shown in the performance tests of this thesis presented in Figure 5.3. Nevertheless, the authors of Noulas et al. (2012) also mentioned the potential of MF approaches that include contextual information available in check-in datasets. This potential is now revealed in the results of the WPOI recommender whose results apparently outperforms conventional MF and CF approaches as shown in Figure 5.3.



(a) Minneapolis **(b)** Boston

(c) Miami **(d)** Honolulu

Figure 5.3: WPOI with the strongest weather feature in the respective city apparently outperforms conventional MF and CF approaches.

6 Conclusions

This Chapter provides a summary of the work done in this thesis. Furthermore, it answers the research questions and gives an overview on expedient feature work. Finally, a section for open science provdes a short technical introduction to the code implemented for the WPOI recommender and how one can use it for further studies.

6.1 Summary of Findings

This thesis presented findings of an empirical analysis on the impact of weather on the users' mobility behaviours and how this information can be used in the context of a POI recommender system.

Empirical Analysis. As the analysis on the Foursquare check-in data in Chapter 4 showed, the weather factors have indeed a significant impact on the people's check-in behavior. In particular the following was obtained:

- The uniform distribution of cloud cover and moonphase lead to the assumption that they will not have a high impact on the accuracy.
- Although sometimes very little, each weather factor has an statistical significant influence on the travel distance of users.
- Analysing the temperature correlation between categories shows a stronger correlation within typical warm or cold categories, but a lower correlation across this two classes.
- Seasonality has an impact on the category popularity.
- The same category in a different climatic region has a diverse popularity for the same prevailing weather condition.

- Northern cities have a higher weather variability than those in the south.
- Not every user has specific weather preferences but some sensitive users prefer some certain weather conditions.

Weather-Aware Recommender System Analysis. A weather-aware POI recommender method named WPOI was introduced that is able to increase the recommender accuracy in comparison to a very strong baseline called Rank-GeoFM by incorporating weather features, such as temperature, visibility or pressure, into the model. The following findings were made in this part:

- Weather context is more useful than the context of time since the time-based method Rank-GeoFM-T was outperformed.
- WPOI works better in regions closer to the tropical zone (Miami, Honolulu) than in regions closer to the boreal zone (Minneapolis, Boston).
- Among the considered weather features, precipitation intensity and visibility are the most significant ones to improve accuracy.
- Although moonphase is uniform distributed, WPOI with that feature incorported also showed performance increases compared to the baseline.
- Weather context information is indeed a very useful contextual information, when incorporated into POI recommender systems.

Conclusively, it can be stated that weather context information is indeed a very useful contextual information when incorporated into POI recommender systems and that weather has an impact on the users' mobility behaviours.

6.2 Answers to Research Questions

The aim of this thesis was to investigate the impact of weather on the users' mobility behaviours and the detection of capabilities to use this impact for improving current state-of-the-art recommender systems. For this reason the research questions defined in Section 1.3 can be answered as follows:

- In order to answer the first research question **(RQ1)** a dataset crawled from Yang et al. (2015a,b) of the LBSN Foursquare was used. According to that check-in data, weather data was crawled from forecast.io. On the basis of this two data sources an empirical analysis was conducted. The objective of this study was to find the impact of weather factors on the users' mobility behaviours. For this reason the dataset was viewed from different perspectives of weather influence, such as travel distance, seasonal, categorical, regional and user. As the analysis revealed, weather has an influence on all of these mentioned factors but the peculiarity differs. On the

one hand weather seems to have little impact on the travel distance where on the other hand weather has a high impact on the popularity of categories. Furthermore, sensitive users were obtained that are very responsive to the prevailing weather conditions and adjust their category preferences according to them. Having a look at Section 4.2.3 reveals the effect of seasonal trends on the different categories. In general, an impact of weather on the users' mobility behaviours can be stated as existent.

- Research question two **(RQ2)** focuses on the feasibility of incorporating weather information gained from the eight weather features introduced in this thesis. The answer to this questions is given in Section 5. The induction of the weather features was achieved by extending a state-of-the-art recommender system called Rank-GeoFM based on matrix factorization and stochastic gradient descent as learning method resulting in an algorithm called WPOI.

- Section 5.3 answers research question three **(RQ3)** by reporting the power of weather factor information in POI recommender systems. The basis Rank-GeoFM approach is outperformed significantly by the weather features and even the factor time shows less impact on the recommender accuracy than most of the weather features. Moreover, the comparison between four cities shows the diverse magnitude of influence of the different weather features in different climatic regions.

- Finally, to answer research question four **(RQ4)**, Section 5.3 has shown that visibility and precipitation intensity have the biggest influence on the accuracy of POI recommender systems.

6.3 Future Work

Currently, WPOI incorporates only one weather feature at a time. Investigating different hybridization methods and other context variables will therefore be a task to conduct in future work. Furthermore, it will help to investigate in more detail, how the algorithm performs on the whole Foursquare dataset, as more interesting patterns across cities may occur.

Additionally, the learning rates of the different cities have shown that learning rate γ_w is dependent on the size of the training set. Therefore, an adaptive learning rate could be introduced to bypass this issue.

Moreover, an analysis of the impact of weather on the users' travel distance has shown a significantly different travelling behaviour in contrasting weather extrema. For this reason, further studies incorporating travel distance probabilities under different weather conditions are reasonable.

Finally, we would like to extend our investigations also at user levels, as the current ones concentrate only on the weather profiles of the POIs.

6.4 Open Science

This Section provides a short technical introduction into the code written for the WPOI algorithm. The basis of the implementation is the work of Gantner et al. (2011) that resulted in the MyMediaLite[1] project. The code from the WPOI project has to be integrated into the MyMediaLite data structure to work properly. The following steps have to be done to get the project to work:

1. Download the sql database dump from `https://github.com/aoberegg/WPOI/tree/master/database/final_database.rar` and create a database out of it.
2. Checkout the MyMediaLite code [2].
3. Checkout the WPOI code [3] and integrate it into the data structure of MyMediaLite (some of the files already exist, please overwrite).
4. Download Xamarin Studio[4] to work with the project.

Structure

The application is separated into two parts, namely the MyMediaLite dll that can be found at

`src/MyMediaLite/bin/Release`

and the test program that is located at

`examples/csharp/TestMediaLite/rating_prediction`

MyMediaLite DLL

To change the MyMediaLite DLL the project storet at

`"src/MyMediaLite/MyMediaLite.sln"`

has to be opened in Xamarin Studio. The WPOI algorithm is then implemented in

[1] `https://github.com/zenogantner/MyMediaLite`, last access May 11, 2016
[2] `https://github.com/zenogantner/MyMediaLite`, last access May 11, 2016
[3] `https://github.com/aoberegg/WPOI/`, last access May 11, 2016
[4] `https://www.xamarin.com/download`, last access May 11, 2016

"src/MyMediatLite/RatingPrediction/
WeatherContextAwareItemRecommender"

Please be aware that the weather data is retrieved from the database stored in the MySql dump.

WPOI Project

The WPOI test project is based on the MyMediaLite DLL and is stored at

"examples/csharp/TestMediaLite/TestMediaLite.sln"

The test programm is implemented in

"examples/csharp/TestMediaLite/rating_prediction.cs"

and provides options for testing the baseline algorithms as well as options for testing WPOI. the resulting executable stored at

"examples/csharp/TestMediaLite/bin/2/Release/TestMediaLite.exe"

can be executed with the following command line options:

```
TestMediaLite.exe "path/to/check-in-data/filename.data"
"SERVER=localhost;DATABASE=<database_name>;UID=<user_id>;
PASSWORD=<password>"
"<algorithm_id>"
"<city_id>"
"<max_iterations>"
"<weather_feature_in_database>"
```

A start of the program with the WPOI algorithm on city 53 with 7000 iterations and the temperature weather feature on Linux OS could look like follows:

```
mono TestMediaLite.exe
"/home/aoberegger/data/evaluation_protocol/city53.data"
"SERVER=localhost;DATABASE=localhost;UID=root;PASSWORD=rootpw"
"4" "53" "7000" "hw.temperature"
```

Table 6.1 shows the command line options for the TestMediaLite executable.

Option	Possible values
"path/to/check-in-data/filename.data"	Path to the check-in data file containing <user_id, item_id, Timestamp> triples.
"SERVER=localhost;-DATABASE=localhost-;UID=root-;PASSWORD=rootpw"	The Database credentials.
city_id	Every city id available in the database but the id has to fit with the datafile containing the check-ins.
max_iterations	The maximum amount of iterations for learning the latent model parameters.
weather_feature_in_database	1 = Beta tuning, 2 = Rank-GeoFM, 3 = Old version of WPOI, 4 = WPOI, 5 = MP, 6 = Item-KNN, 7 = User-KNN, 8 = WRMF, 9 = BPRMF

Table 6.1: Command line options for the TestMediaLite executable.

List of Figures

List of Tables

A Datasets

A.1 Data Storage

To allow accessing and saving, the data retrieved from the crawling process described in Section 3.2 and from Yang et al. (2015b), a database had to be built. The ERM diagram according to Chen (1976) in Figure A.1 describes the six entities needed to represent the data and their connection. The following entities are part of the database model:

- **CHECKIN:** Represents a check-in procedure of a user at a given time and venue
- **VENUE:** Describes the geographical location, city, country and category of a venue that can be checked-in
- **CATEGORY:** Additional to the name of the category this entity represents the hierarchical structure of foursquare categories described in Section 3.1
- **CITY:** Contains the geographical center, country and the name of the cities.
- **DAILY_WEATHER:** Stores a city's average value of a day of the weather features described in Table 3.2.
- **HOURLY_WEATHER:** Further to the DAILY_WEATHER table this table contains the hourly values of the weather features.

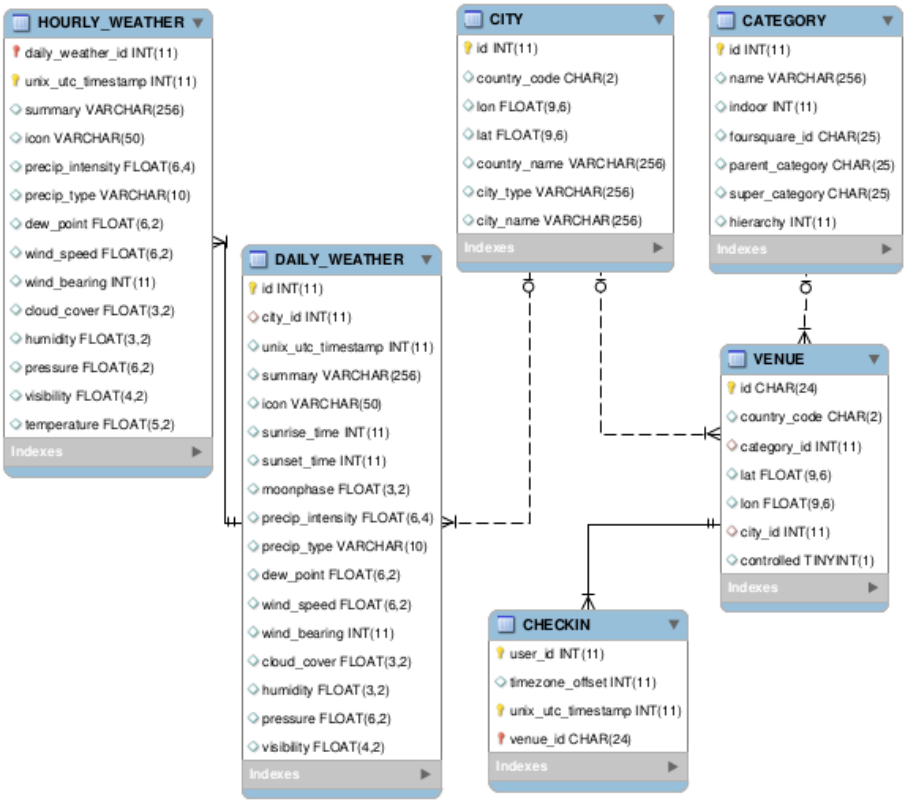


Figure A.1: The ERM diagram used to save the data crawled from the weather API and from the dataset of Yang et al. (2015b).

B

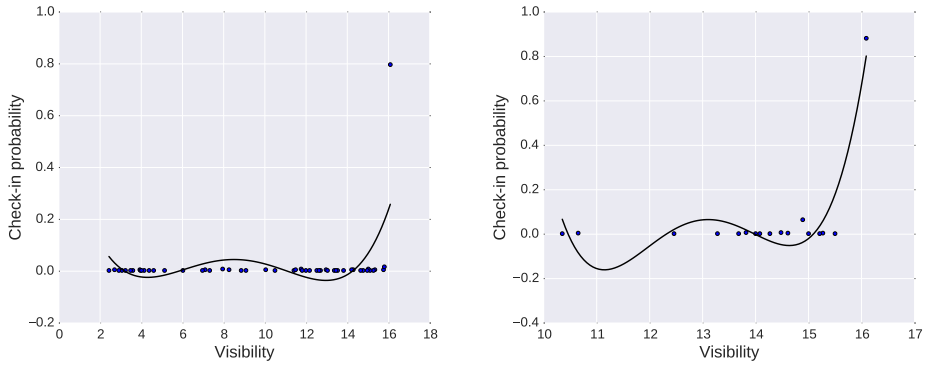
Further Results

This Chapter provides additional results that were not be able to present due to space limitations. The same structure as in the thesis is taken to illustrate their affiliation to the original structure.

B.1 Empirical Analysis

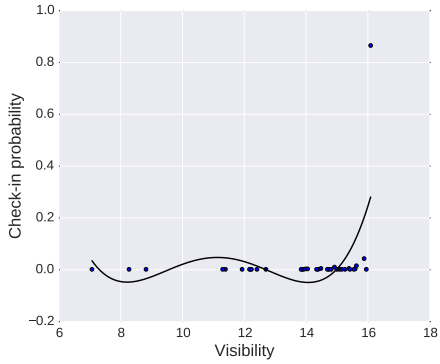
Regional Analysis

Additionaly to Figure 4.10, this Section shows the check-in probabilities of the category “Beach” for five other weather features that showed weather impact.

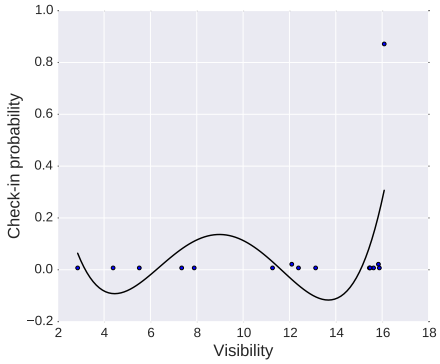


(a) "Beach" in Boston

(b) "Beach" in Honolulu

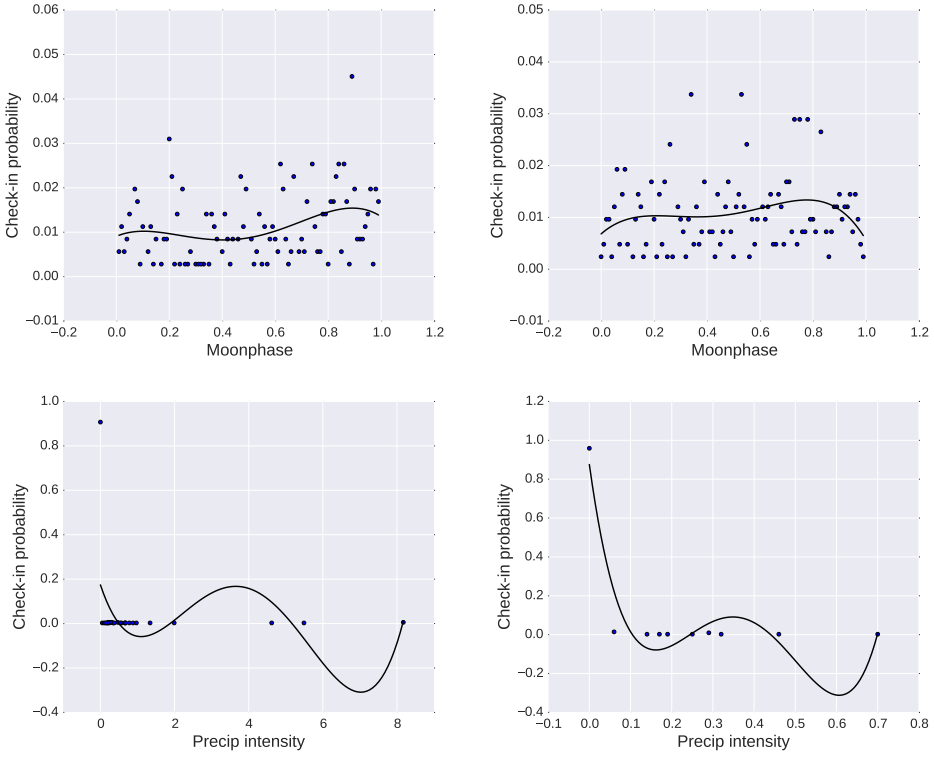


(c) "Beach" in Miami



(d) "Beach" in Minneapolis

Figure B.1: On the one hand people in Boston also go to the Beach when visibility is Bad where on the other hand people in Honolulu prefer better visibility.



(a) "Beach" in Boston

(b) "Beach" in Honolulu

(c) "Beach" in Miami

(d) "Beach" in Minneapolis

Figure B.2: While on the one hand moonphase does not show any impact on the check-in probability of the category "Beach", precipitation has an impact. In Minneapolis probability of going to the beach is almost zero when precipitation intensity is > 0.

User Analysis

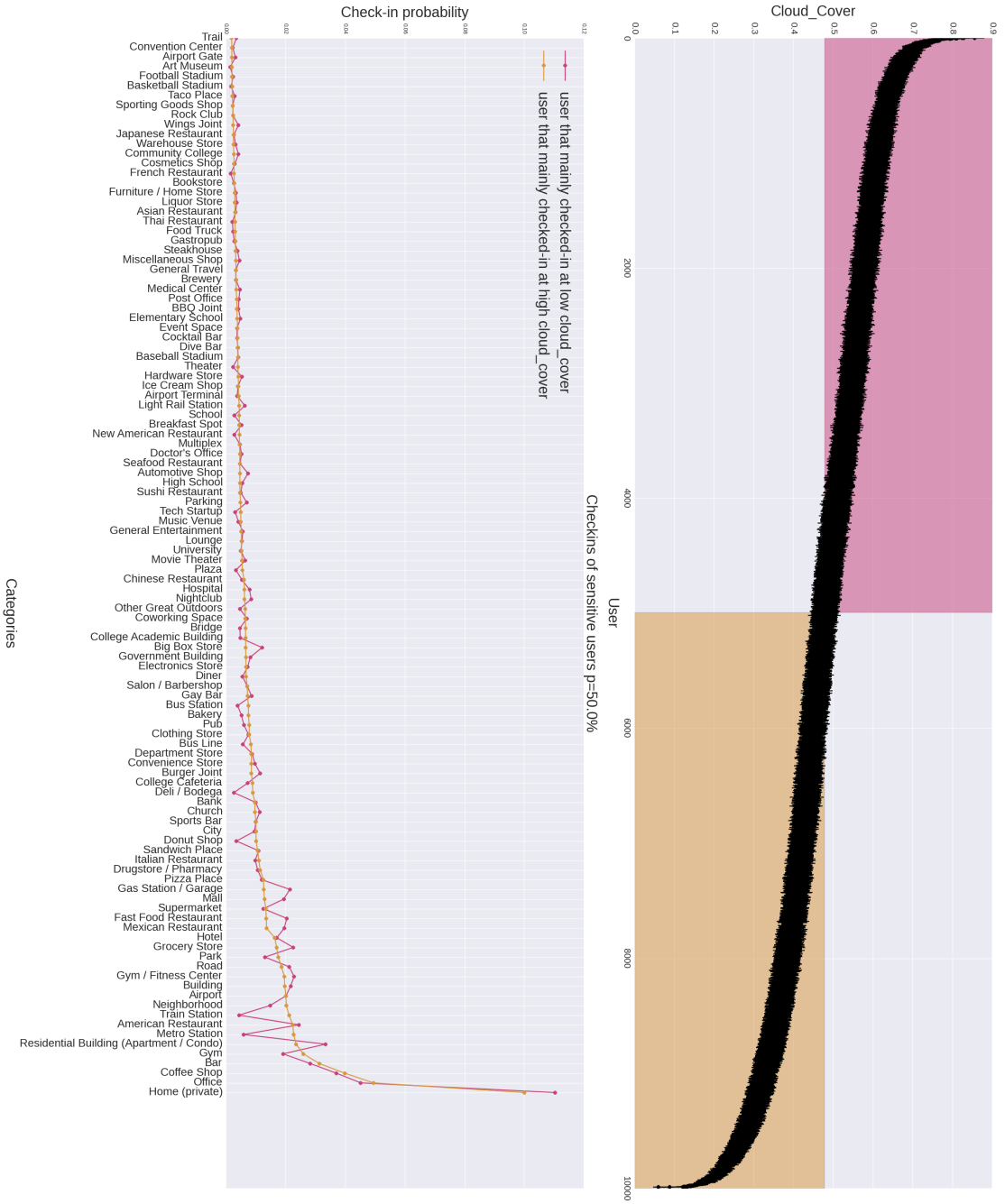


Figure B.3: User sensitivity to cloud fluctuation.

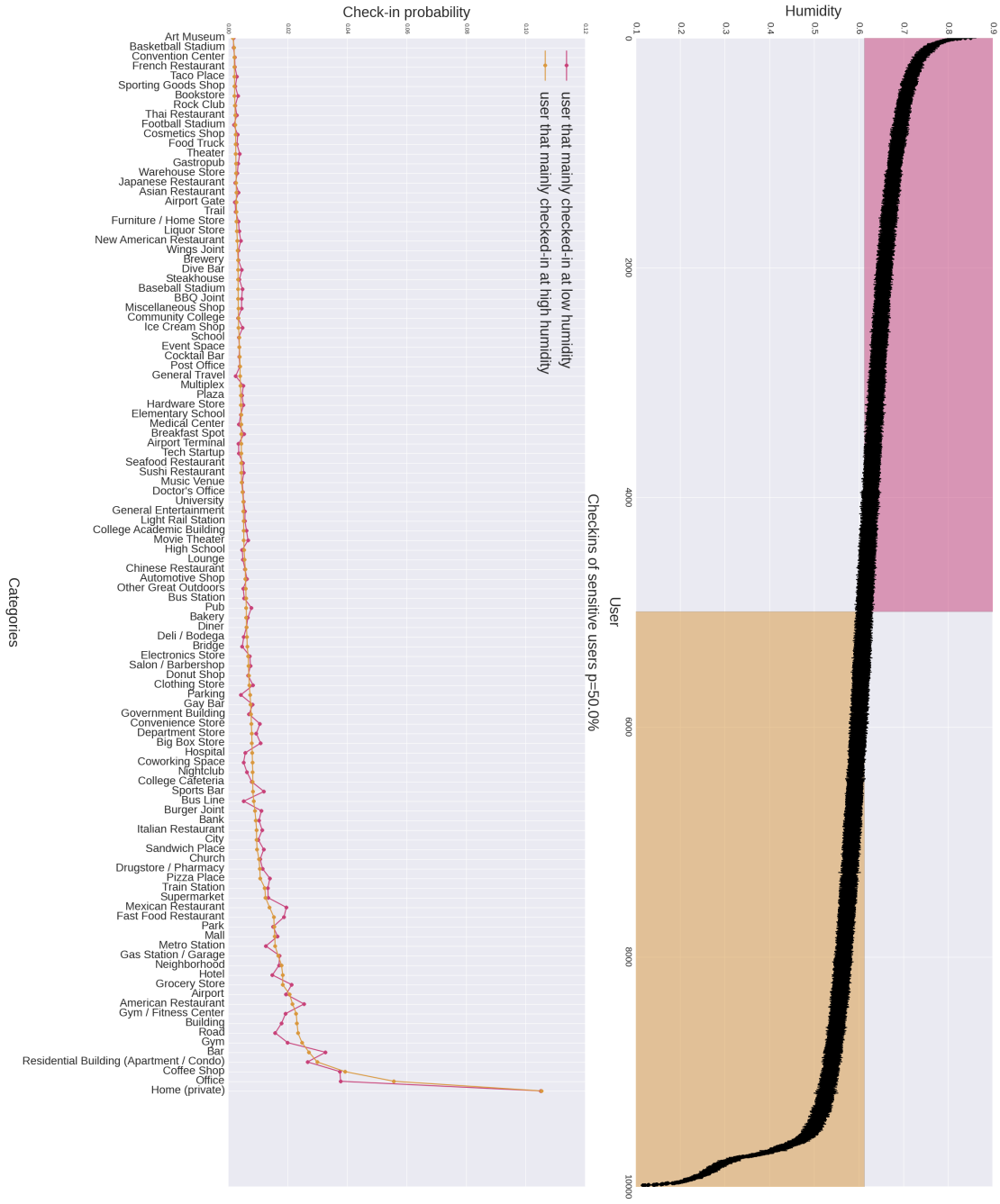


Figure B.4: User sensitivity to humidity fluctuation.

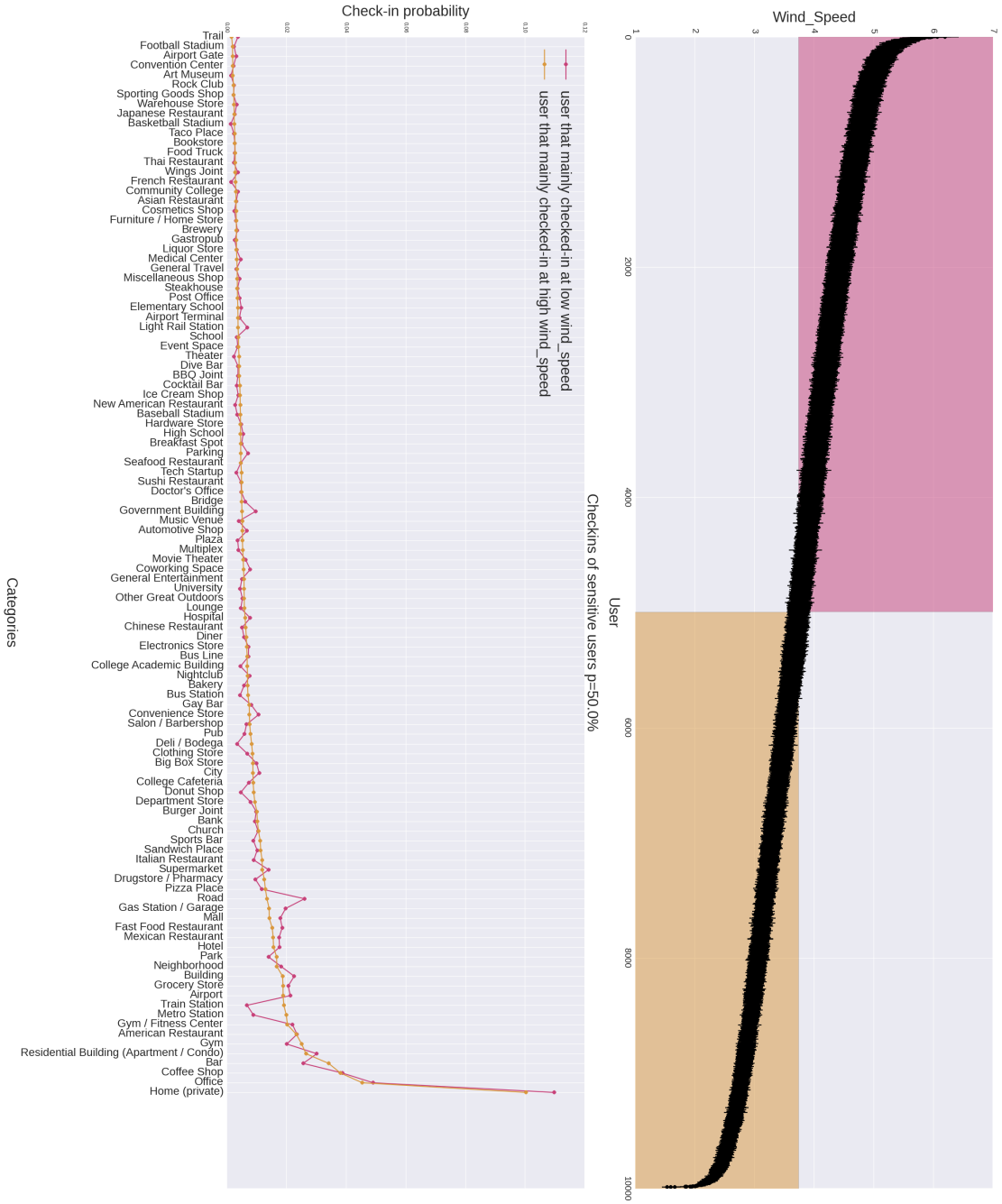


Figure B.5: User sensitivity to windspeed.

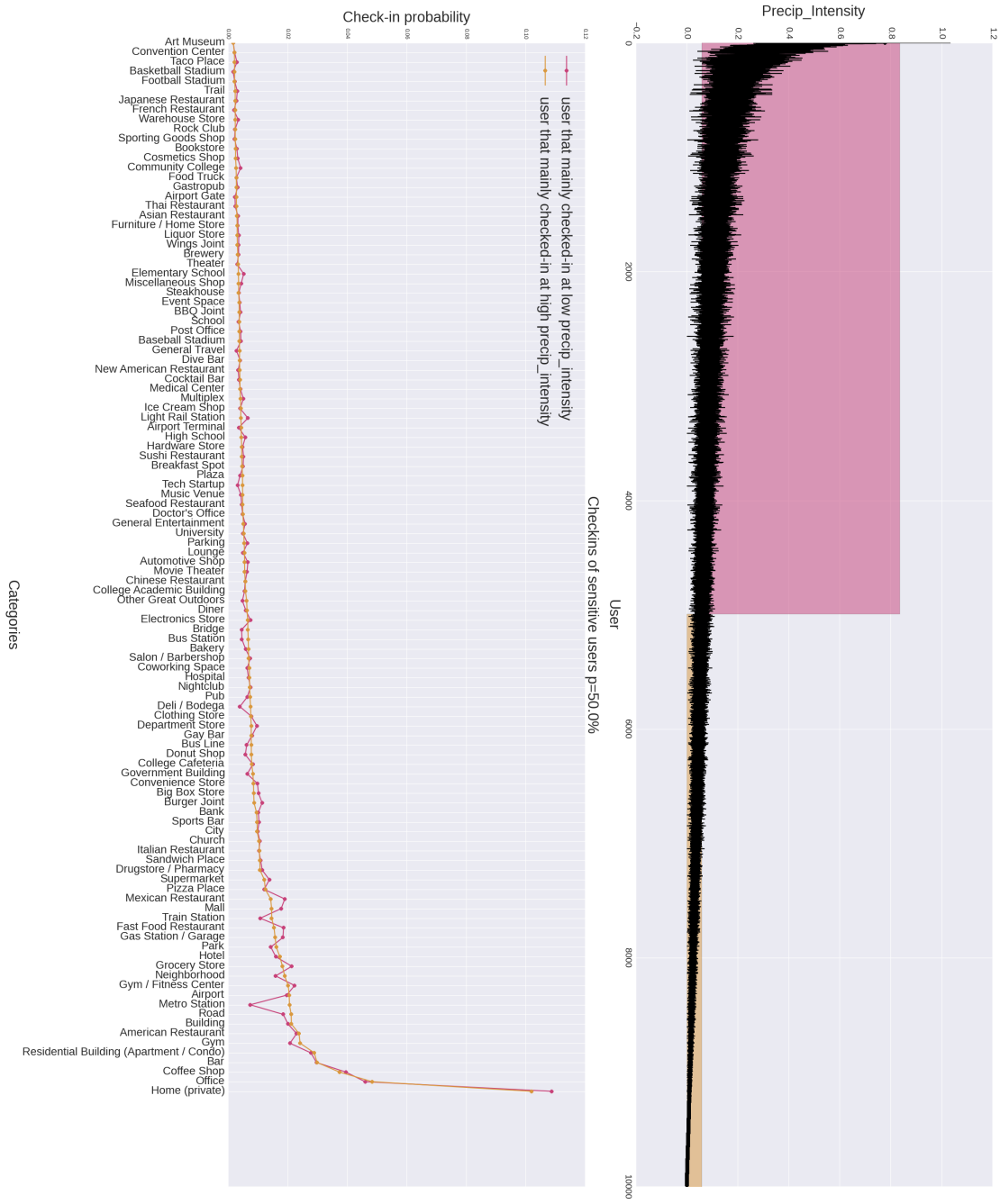


Figure B.6: User sensitivity to precipitation intensity fluctuation.

B.2 Weather-Aware Recommender Analysis

B.2.1 Evaluation Metrics

For further studies the following evaluation metrics have been used:

Recall@N. The recall metric defined in Kent et al. (1955) sets the number of correct ranked items in relation to the number of correct items. The calculation of the recall metric is shown in equation B.1.

$$recall = \frac{|I_{correct} \cap I_{fetched}|}{|I_{correct}|} \tag{B.1}$$

To measure the accuracy of the top-N retrieved items of the recommender recall@N cuts the retrieved items to a list of N. The result of this measure shows the correctness of this top-N list.

Precision@N. Kent et al. (1955) also defined precision metric which sets the number of correct ranked items in relation to the amount of items fetched from the algorithm. Equation B.2 shows the calculation of the recall metric.

$$precision = \frac{|I_{correct} \cap I_{fetched}|}{|I_{fetched}|} \tag{B.2}$$

Equally to the recall metric, precision can be measured at a given cut-off N to get accuracy of the top-N items.

F Score. The F score metric combines the two previous described metrics precision and recall by computing an weighted average of them. The conventional F_1 score is computed with the harmonic mean of precision and recall given by equation B.3.

$$F_1 = 2 * \frac{rec * prec}{rec + prec} \tag{B.3}$$

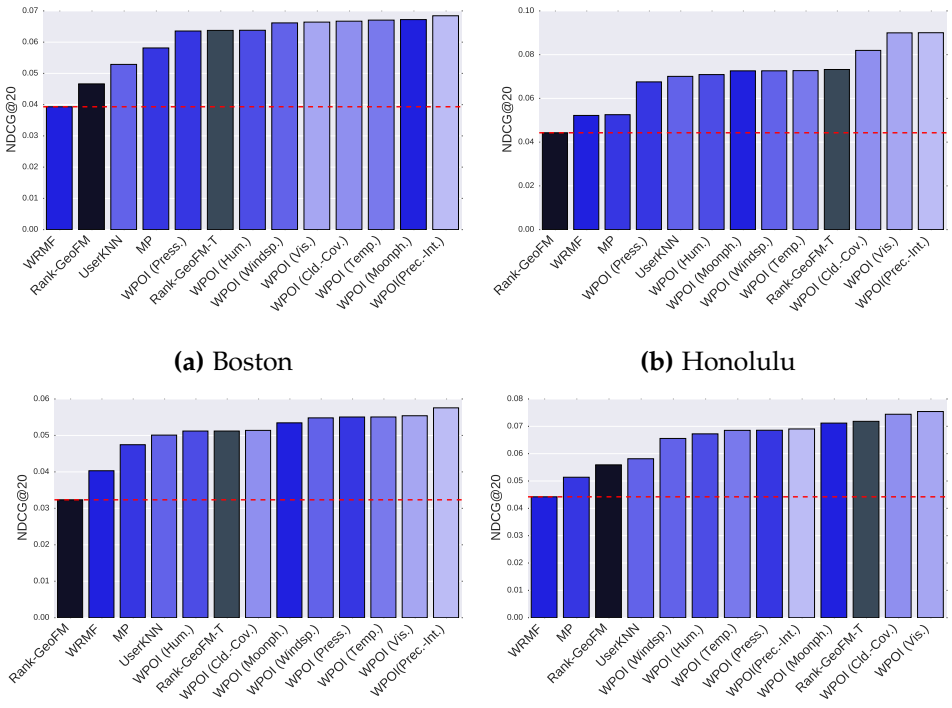
Mean Average Precision (MAP). In contrast to precision and recall the MAP metric does not only rely on whether a correct item is in the recommendation list instead it also takes into account on which position the item is in the ranked list. Average precision (AP) sums up the precision (see Section B.2.1) of each rank k of the recommended list where a relevant item was found and averages it over the amount of relevant items. Usually more than one recommendation query is made which leads to MAP that calculates the mean over the AP's of all Queries. Equation B.4 and B.5 reveal the calculation of MAP having a function rel(k) returning 1 if the item at rank k is a relevant one and 0 otherwise and using Prec@k as described in Section B.2.1.

$$AP = \frac{\sum_{k=1}^{n} Prec@k * rel(k)}{I_{correct}} \tag{B.4}$$

$$MAP = \frac{AP}{\#Queries} \tag{B.5}$$

Performance of WPOI

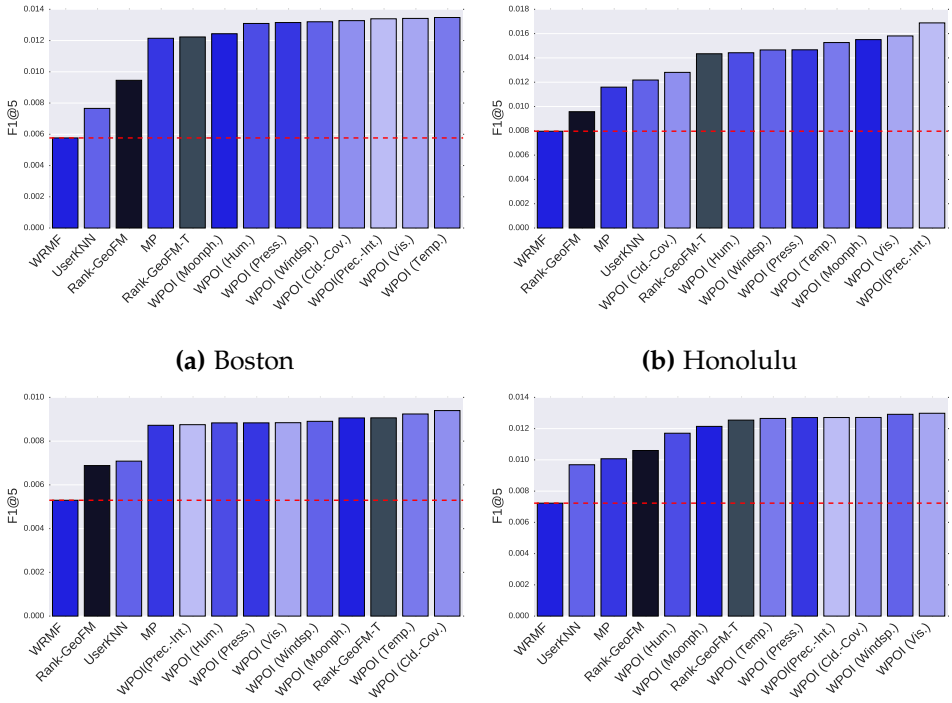
Figure B.7 to B.10 show the comparison of all algorithms used in this thesis with all metrics described in Section B.2.1 and 5.2.3. As a substitute for recall and precision metrics F1 score is taken as it is the harmonic mean of this two metrics. It shows up that Rank-GeoFM performs very poorly in comparison to the baseline algorithms. Weather features in common perform quite good with precipitation intensity and visibility as positive outliers.



(a) Boston **(b)** Honolulu

(c) Miami **(d)** Minneapolis

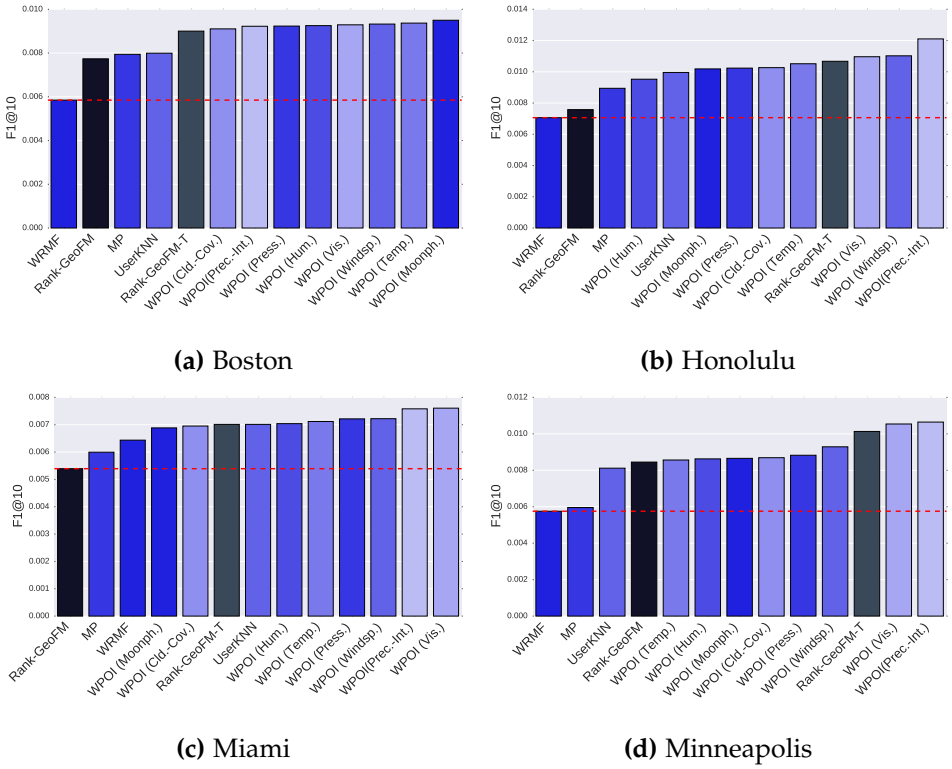
Figure B.7: The performance measure of all algorithms used in this thesis with the NDCG metric.



(a) Boston **(b)** Honolulu

(c) Miami **(d)** Minneapolis

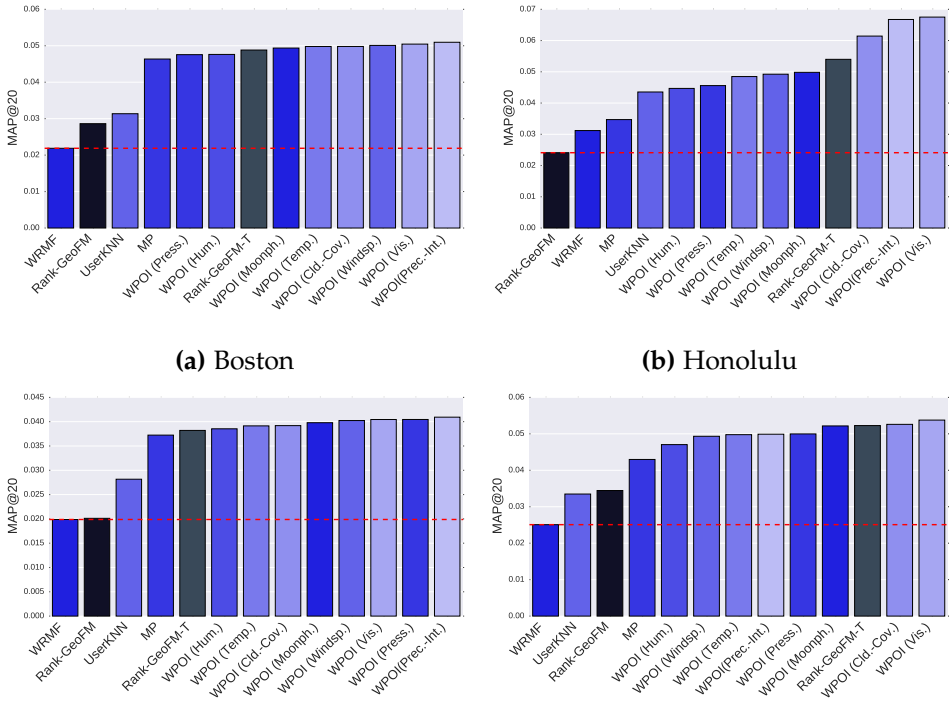
Figure B.8: The performance measure of all algorithms used in this thesis with the F1@5 metric.



(a) Boston **(b)** Honolulu

(c) Miami **(d)** Minneapolis

Figure B.9: The performance measure of all algorithms used in this thesis with the F1@10 metric.



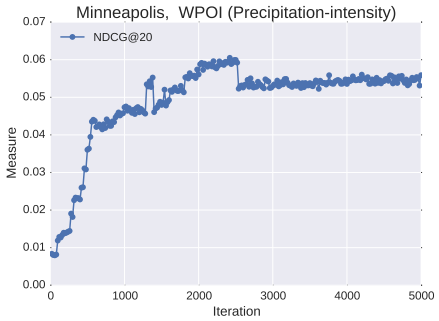
(a) Boston **(b)** Honolulu

(c) Miami **(d)** Minneapolis

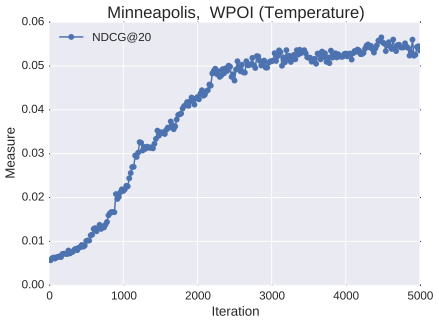
Figure B.10: The performance measure of all algorithms used in this thesis with the MAP metric.

B.3 Parameter Learning

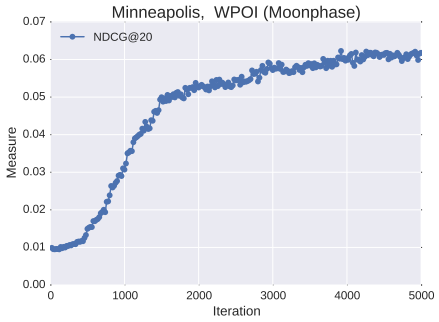
Figure B.11h to B.13 part shows the parameter learning in the three other cities of the investigations.



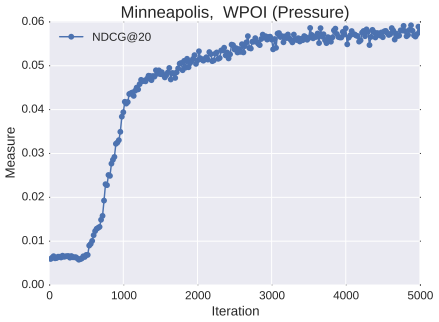
(a) Precipitation intensity



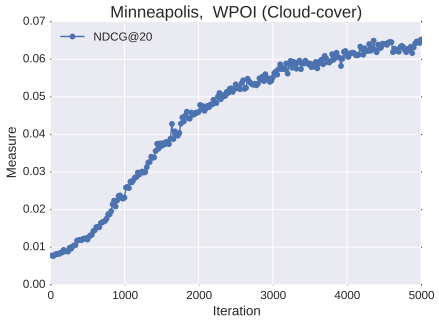
(b) Temperature



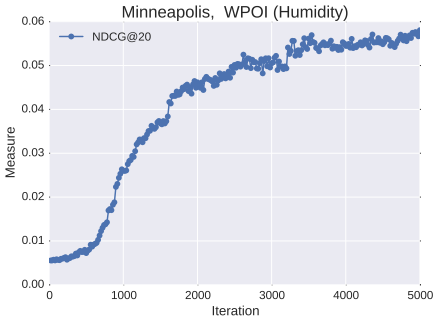
(c) Moonphase



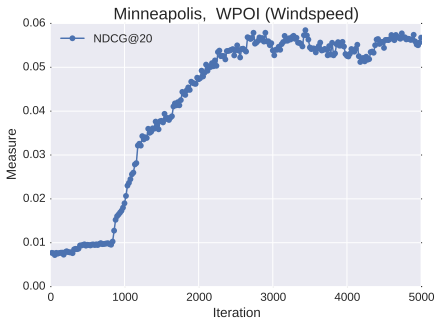
(d) Pressure



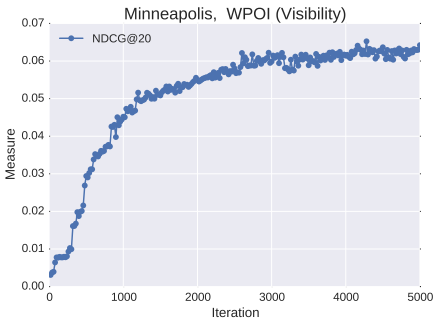
(e) Cloud cover



(f) Humidity

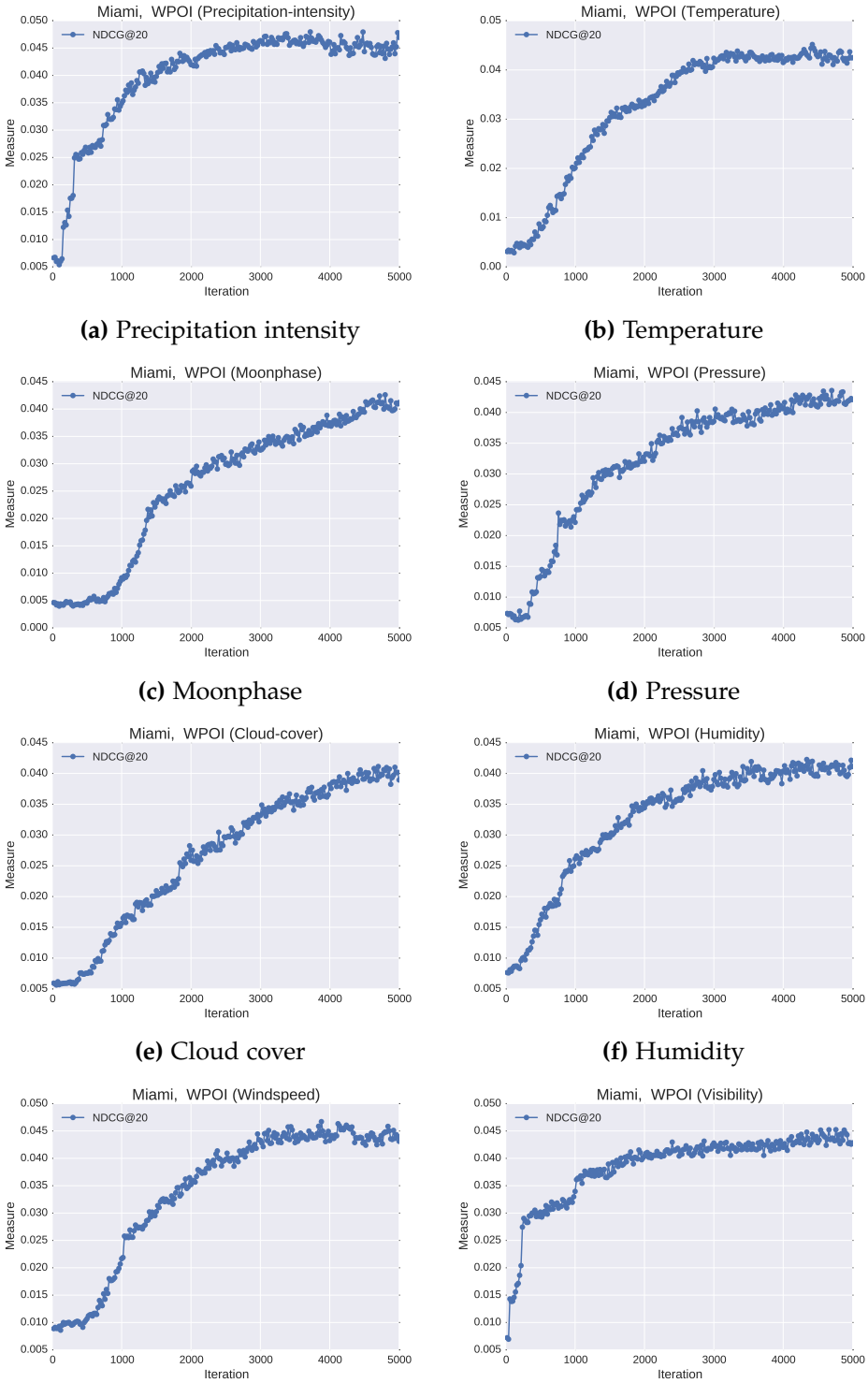


(g) Windspeed



(h) Visibility

Figure B.11: Learning of the latent parameters of the prediction model of the eight different features in Minneapolis. It shows a stable learning rate and a convergence at $\sim 3,000 - 5,000$ iterations.



(a) Precipitation intensity

(b) Temperature

(c) Moonphase

(d) Pressure

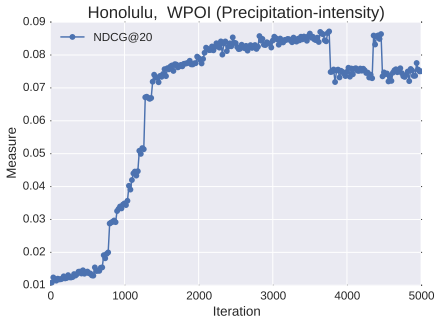
(e) Cloud cover

(f) Humidity

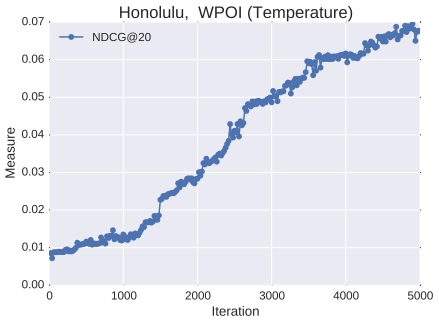
(g) Windspeed

(h) Visibility

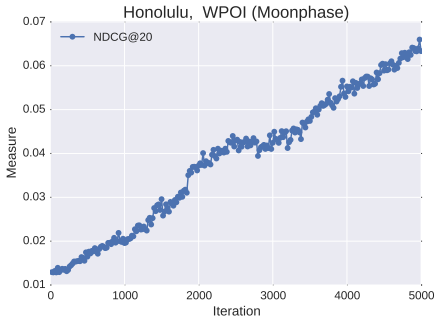
Figure B.12: Learning of the latent parameters of the prediction model of the eight different features in Miami. It shows a stable learning rate and a convergence at $\sim 3,000 - 5,000$ iterations.



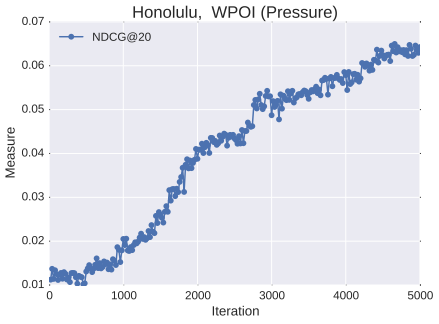
(a) Precipitation intensity



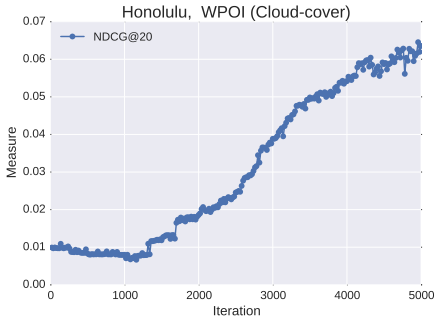
(b) Temperature



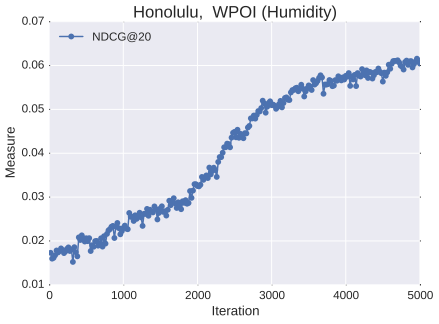
(c) Moonphase



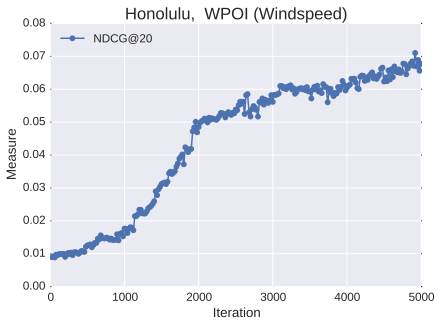
(d) Pressure



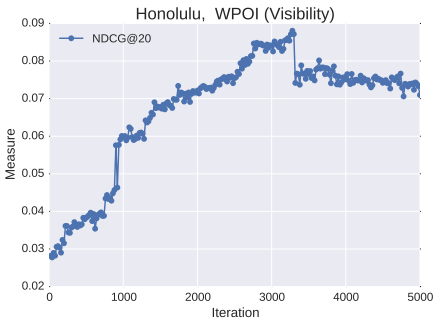
(e) Cloud cover



(f) Humidity



(g) Windspeed



(h) Visibility

Figure B.13: Learning of the latent parameters of the prediction model of the eight different features in Honolulu. It shows a stable learning rate and a convergence at $\sim 3,000 - 5,000$ iterations. Just precipitation intensity and visibility show a deflection downwards at $\sim 3,5000$ iterations.

C

List of Acronyms

This is the list of acronyms used in this thesis:

API Application Programming Interface

CARS Context Aware Recommender Systems

CF Collaborative Filtering

DOF Degrees of Freedom

ERM Entity-Relationship Model

GD Gradient Descent

KNN K-Nearest Neighbours

LBSN Location Based Social Network

MCLRE Multi-Contextual Learning to Rank Events

MF Matrix Factorization

MGM Multi-center Gaussian Model

NDCG Normalized Discounted Cumulative Gain

PMFSR Probabilistic Matrix Factorization with Social Regularization

POI Point of Interest

SE Standard Error of the Mean

SGD Stochastic Gradient Descent

WPOI Weather-Aware POI Recommender System

WRMF Weighted Regularized Matrix Factorization

Bibliography

Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011. 9

Leandro Balby Marinho, Christoph Trattner, and Denis Parra. Are real-world place recommender algorithms useful in virtual world environments? In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pages 245–248, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3692-5. doi: 10.1145/2792838.2799674. URL http://doi.acm.org/10.1145/2792838.2799674. 8, 35, 53

Jie Bao, Yu Zheng, and Mohamed F. Mokbel. Location-based and preference-aware recommendation using sparse geo-social networking data. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '12, pages 199–208, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1691-0. doi: 10.1145/2424321.2424348. URL http://doi.acm.org/10.1145/2424321.2424348. 18

Jie Bao, Yu Zheng, David Wilkie, and Mohamed F. Mokbel. Recommendations in location-based social networks: A survey. *GeoInformatica*, November 2014. URL http://research.microsoft.com/apps/pubs/default.aspx?id=191797. 2, 7, 8, 10

Léon Bottou. *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, chapter Large-Scale Machine Learning with Stochastic Gradient Descent, pages 177–186. Physica-Verlag HD, Heidelberg, 2010. ISBN 978-3-7908-2604-3. doi: 10.1007/978-3-7908-2604-3_16. URL http://dx.doi.org/10.1007/978-3-7908-2604-3_16. 54, 55

John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical Report MSR-TR-98-12, Microsoft Research, May 1998. URL http://research.microsoft.com/apps/pubs/default.aspx?id=69656. 7

Soumen Chakrabarti, Byron Dom, Prabhakar Raghavan, Sridhar Rajagopalan, David Gibson, and Jon Kleinberg. Automatic resource compilation by analyzing hyperlink structure and associated text. In *Proceedings of the Seventh International Conference on World Wide Web 7*, WWW7, pages 65–74, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V. URL http://dl.acm.org/citation.cfm?id=297805.297821. 7

Peter Pin-Shan Chen. The entity-relationship model—toward a unified view of data. *ACM Trans. Database Syst.*, 1(1):9–36, March 1976. ISSN 0362-5915. doi: 10.1145/320434.320440. URL http://doi.acm.org/10.1145/320434.320440. 81

Chen Cheng, Haiqin Yang, Irwin King, and Michael R. Lyu. Fused matrix factorization with geographical and social influence in location-based social networks. In *Proc. of AAAI*, pages 17–23, 2012. 2, 11, 13

James D Evans. *Straightforward statistics for the behavioral sciences*. Brooks/Cole, 1996. 27

Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. MyMediaLite: A free recommender system library. In *5th ACM International Conference on Recommender Systems (RecSys 2011)*, 2011. 74

Mark A Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999. 30

Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. Ieee, 2008. 62

Allen Kent, Madeline M. Berry, Fred U. Luehrs, and J. W. Perry. Machine literature searching viii. operational criteria for designing information retrieval systems. *American Documentation*, 6(2):93–101, 1955. ISSN 1936-6108. doi: 10.1002/asi.5090060209. URL http://dx.doi.org/10.1002/asi.5090060209. 91

Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September 1999. ISSN 0004-5411. doi: 10.1145/324133.324140. URL http://doi.acm.org/10.1145/324133.324140. 7

Sonja Kuhnt and Dirk Taeger. *Statistical Hypothesis Testing with SAS and R*. John Wiley and Sons, 2014. URL http://proquest.techbus.safaribooksonline.de/book/databases/9781118762615/firstchapter#X2ludGVybmFsX0h0bWxWaWV3P3htbGlkPTk3ODExMTg3NjI2MTUlMkZjMDFfbGV2ZWwxXzFfaHRtbCZxdWVyeT0=. 26

David Lane. Introduction to statistics, 2016. URL http://onlinestatbook.com/. 27

E.L. Lehmann and Joseph P. Romano. *Testing statistical hypothesises*. Springer Science+Business Media, New York, 2005. 26

Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. Rank-geofm: A ranking based geographical factorization method for point of interest recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 433–442, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3621-5. doi: 10.1145/2766462.2767722. URL http://doi.acm.org/10.1145/2766462.2767722. 2, 3, 51, 54, 55, 56, 57, 59, 67

Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. Geomf: Joint geographical modeling and matrix factorization for point-of-interest recommendation. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 831–840, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623638. URL http://doi.acm.org/10.1145/2623330.2623638. 11, 13

Mingfeng Lin, Henry C. Lucas, and Galit Shmueli. Too big to fail: Large samples and the p-value problem. *Information Systems Research*, 24(4):906–917, 10 2013. ISSN 1047-7047. doi: 10.1287/isre.2013.0480. 26

G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, Jan 2003. ISSN 1089-7801. doi: 10.1109/MIC.2003.1167344. 7

R. Gnanadesikan M. B. Wilk. Probability plotting methods for the analysis of data. *Biometrika*, 55(1):1–17, 1968. ISSN 00063444. URL http://www.jstor.org/stable/2334448. 28

Hao Ma, Irwin King, and Michael R. Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 203–210, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-483-6. doi: 10.1145/1571941.1571978. URL http://doi.acm.org/10.1145/1571941.1571978. 11

Augusto Q. Macedo, Leandro B. Marinho, and Rodrygo L.T. Santos. Context-aware event recommendation in event-based social networks. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pages 123–130, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3692-5. doi: 10.1145/2792838.2800187. URL http://doi.acm.org/10.1145/2792838.2800187. 11, 12, 13

Donald Metzler and W. Bruce Croft. Linear feature-based models for information retrieval. *Inf. Retr.*, 10(3):257–274, June 2007. ISSN 1386-4564. doi: 10.1007/s10791-006-9019-z. URL http://dx.doi.org/10.1007/s10791-006-9019-z. 13

A. Noulas, S. Scellato, N. Lathia, and C. Mascolo. A random walk around the city: New venue recommendation in location-based social networks. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 144–153, Sept 2012. doi: 10.1109/SocialCom-PASSAT.2012.70. 7, 11, 68

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. URL http://ilpubs.stanford.edu:8090/422/. Previous number = SIDL-WP-1999-0120. 7

Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 502–511. IEEE, 2008. 62, 63

Umberto Panniello, Alexander Tuzhilin, and Michele Gorgoglione. Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction*, 24(1):35–65, 2012. ISSN 1573-1391. doi: 10.1007/s11257-012-9135-y. URL http://dx.doi.org/10.1007/s11257-012-9135-y. 10

Moon-Hee Park, Jin-Hyuk Hong, and Sung-Bae Cho. Location-based recommendation system using bayesian user's preference model in mobile devices. In *Proceedings of the 4th International Conference on Ubiquitous Intelligence and Computing*, UIC'07, pages 1130–1139, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-73548-8, 978-3-540-73548-9. URL http://dl.acm.org/citation.cfm?id=2391319.2391438. 8

K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series*

5, 50(302):157–175, 1900. doi: 10.1080/14786440009463897. URL http://dx.doi.org/10.1080/14786440009463897. 27

K. Pearson and J. Blakeman. *On the Theory of Contingency and Its Relation to Association and Normal Correlation*. Number v. 1-4 in A Mathematical Theory of Random Migration. Dulau and Company, 1906. URL https://archive.org/stream/cu31924003064833#page/n37/mode/2up. 27

Robert H. Perry, Don W. Green, and James O. Maloney. *Perry's chemical engineers' handbook*. McGraw-Hill, New York, 1997. URL https://openlibrary.org/books/OL1011071M/Perry's_chemical_engineers'_handbook. 23

L. Ramaswamy, D. P., R. Polavarapu, K. Gunasekera, D. Garg, K. Visweswariah, and S. Kalyanaraman. Caesar: A context-aware, social recommender system for low-end mobile devices. In *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pages 338–347, May 2009. doi: 10.1109/MDM.2009.66. 8

Francesco Ricci, Lior Rokach, and Bracha Shapira. *Recommender Systems Handbook*. Springer US, Boston, MA, 2011. ISBN 978-0-387-85820-3. doi: 10.1007/978-0-387-85820-3_1. URL http://dx.doi.org/10.1007/978-0-387-85820-3_1. 5, 59

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *PROC. 10TH INTERNATIONAL CONFERENCE ON THE WORLD WIDE WEB*, pages 285–295, 2001a. 60

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001b. ACM. ISBN 1-58113-348-0. doi: 10.1145/371920.372071. URL http://doi.acm.org/10.1145/371920.372071. 7

Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Matrix factorization and neighbor based algorithms for the netflix prize problem. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 267–274, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-093-7. doi: 10.1145/1454008.1454049. URL http://doi.acm.org/10.1145/1454008.1454049. 61

W. R. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography*, 46:234–240, 1970. ISSN 00130095, 19448287. URL http://www.jstor.org/stable/143141. 12

John W. Tukey. The future of data analysis. *Ann. Math. Statist.*, 33(1):1–67, 03 1962. doi: 10.1214/aoms/1177704711. URL http://dx.doi.org/10.1214/aoms/1177704711. 28

Zheng Wen. Recommendation system based on collaborative filtering. Technical report, Citeseer, 2008. 60

Dingqi Yang, Daqing Zhang, Longbiao Chen, and Bingqing Qu. Nationtelescope: Monitoring and visualizing large-scale collective behavior in lbsns. *Journal of Network and Computer Applications*, 55:170–180, 2015a. 2, 16, 72

Dingqi Yang, Daqing Zhang, and Bingqing Qu. Participatory cultural mapping based on collective behavior in location based social networks. *ACM Transactions on Intelligent Systems and Technology*, 2015b. in press. 2, 16, 20, 72, 81, 82

Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik Lun Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In *Proc. of SIGIR'11*, pages 325–334. ACM, 2011a. ISBN 978-1-4503-0757-4. 2

Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 325–334, New York, NY, USA, 2011b. ACM. ISBN 978-1-4503-0757-4. doi: 10.1145/2009916.2009962. URL http://doi.acm.org/10.1145/2009916.2009962. 12, 53

Hongzhi Yin, Yizhou Sun, Bin Cui, Zhiting Hu, and Ling Chen. Lcars: a location-content-aware recommender system. In *Proc. of KDD'13*, pages 221–229. ACM, 2013. ISBN 978-1-4503-2174-7. 2

Yu Zheng and Xiaofang Zhou. *Computing with Spatial Trajectories*. Springer Publishing Company, Incorporated, 1st edition, 2011. ISBN 1461416280, 9781461416289. 6, 7

Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 791–800, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-487-4. doi: 10.1145/1526709.1526816. URL http://doi.acm.org/10.1145/1526709.1526816. 7