Demand Supply Gap Predictor Using Data Mining and Machine Learning Techniques

by

Umair Ahmad

Haider Imam Kazmi

Advised by

Mr. Tahir Rasheed

COMSATS University Islamabad

Vehari Campus

Demand Supply Gap Predictor Using Data Mining and Machine Learning Techniques

by

Umair Ahmad

Haider Imam Kazmi

Advised by

Mr. Tahir Rasheed

COMSATS University Islamabad

Vehari Campus

Demand Supply Gap Predictor Using Data Mining and Machine Learning Techniques

An undergraduate thesis submitted to the Department of Computer Science as partial fulfillment of the requirements for the award of Degree of Bachelor of Science in Computer Science

Name	Registration Number
UMAIR AHMAD	CIIT/FA14-BCS-122/VHR
HAIDER IMAM KAZMI	CIIT/FA14-BCS-044/VHR

Supervisor

Mr. Tahir Rasheed

Department of Computer Science

Signature of Supervisor: ___________________ Date: ________

COMSATS University Islamabad

Vehari Campus

Final Approval

This Thesis Titled

Demand Supply Gap Predictor Using Data Mining and Machine Learning Techniques

By

Umair Ahmad

CIIT/FA14-BCS-122/VHR

Haider Imam Kazmi

CIIT/FA14-BCS-044/VHR

External Examiner: ____

Supervisor: ____

Tahir Rasheed A.P , Department of Computer Science COMSATS Vehari

Internal Examiner: ____

Dr./Mr. _

Designation:_ _

Department of Computer Science COMSATS University Vehari

Convener Project Committee:

____

Dr./Mr. _

Designation:_ _

Department of Computer Science COMSATS University Vehari

DECLARATION

It is certified that the research work presented in this thesis is to the best of my knowledge my own. All sources used and any help received in the preparation of this dissertation have been acknowledged. Hereby, it is declared that this material is not submitted, either in whole or in part, for any other degree at this or any other institution.

Date: _______________ Signature of the Student: ________________________________

Umair Ahmad
CIIT/FA14-BCS-122/VHR

Signature of the Student: ________________________________

Haider Imam Kazmi
CIIT/FA14-BCS-044/VHR

CERTIFICATE

It is certified that Mr. Umair Ahmad(CIIT/FA14-BCS-122/VHR) and Mr. Haider Imam Kazmi(CIIT/FA14-BCS-044/VHR) have carried out all the work related to this thesis under my supervision at the Department of Computer Science COMSATS University Islamabad , Vehari campus. And the work fulfills the requirements for award of BS degree.

Date: _______________ Supervisor: _______________________________________

Tahir Rasheed
Assistant Professor dept. Computer Science

Acknowledgements

In the name of ALLAH, the kindest and most merciful

I would like to thank to my friends and parents who help us directly or indirectly who kept backing me up in all the times, both financially and morally…

I would also like to thank to my Supervisor Mr. Tahir Rasheed (A.P) for his guidance and encouraging us to work hard and smart. I have found him very helpful while discussing the optimization issues in this dissertation work. His critical comments on my work have certainly made me think of new ideas and techniques in the fields of optimization and software simulation.

We are grateful to the ALLAH Almighty who provides all the resources of every kind to us, so that we make their proper use for the benefit of mankind. May He keep providing us with all the resources, and the guidance to keep helping the humanity.

DEDICATION

To Almighty ALLAH and the Holy Prophet Muhammad (P.B.U.H)

&

Our Loving Family

Abstract

The online car-hailing service has gained great popularity all over the world. As more passengers and more drivers use the service, it becomes increasingly more important for the the car-hailing service providers to effectively schedule the drivers to minimize the waiting time of passengers and maximize the driver utilization, thus to improve the overall user experience. we study the problem of predicting the real-time car-hailing supply-demand, which is one of the most important component of an effective scheduling system. Our objective is to predict the gap between the car-hailing supply and demand in a certain area in the next few minutes. Based on the prediction, we can balance the supply-demands by scheduling the drivers in advance. We utilize the Scikit-Learn a popular library used in Data mining and machine learning , to map the high dimensional features into a smaller subspace by providing the Algorithms like linear regression, decision tree regression/ Random Forest and polynomial regression . In the experiment, we show that the regression method enhances the prediction accuracy significantly. Furthermore, with regression, our model also automatically discovers the similarities among the supply-demand patterns of different areas and timeslots.. Moreover, our framework is highly flexible and extendable. Based on our framework, it is very easy to utilize multiple data sources (e.g., car-hailing orders, weather and traffic data) to achieve a high accuracy. We conduct extensive experimental evaluations, which show that our framework provides more accurate prediction results than the existing methods.

Table of Contents

1.Introduction

1.1. Background

As less than 10% of worlds citizens own automobiles, the frequency at which citizens commute on taxis, buses, trains, and planes is very high. Uber, the dominant ride-hailing company, processes over 11 million trips, plans over 9 billion routes and collects over 50TB of data per day. To meet needs of riders, Uber must continually innovate to improve cloud computing and big data technologies and algorithms in order to process this massive amount of data and uphold service reliability. Supply-demand forecasting is critical to enabling Uber to maximize utilization of drivers and ensure that riders can always get a car whenever and wherever they may need a ride. Supply-demand forecasting helps to predict the volume of drivers and riders at a certain time period in a specific geographic area. For instance, demand tends to surge in residential areas in the mornings and in business districts in the evenings. Supply-demand forecasting allows Didi to predict demand surges and guide drivers to those areas. The end result is higher earnings for drivers and no surge pricing for riders!

1.2. Objective:

The goal of this project is basically try to do to gather information from ride giving companies e.g. Uber, Careem and predict the future gap between the demand and supply. Our project will mainly focus on the following objectives:

1. Develop a system that automatically and regularly collects and correlates data for ride giving companies e.g. Uber, Careem.
2. Utilize data mining and machine learning techniques to predict the Gap between Demand and Supply similar to weather forecaster prediction.
3. Try to Minimize the gap as much as possible by giving the Accuracy up to 85 Percent.

1.2.1. Scope:

- This project facilitates the User and ride giving companies.
- Make the companies System more perfect and more useful by future prediction.
- Using this module companies knows the number of demanded rides from specific areas.
- Companies find the Gap according factors temperature, time, days, area and environment to minimize the Gap.
- The companies and user both facilitated by this module, users by saving his time and coma pines by saving fuel.
- satisfy the customer’s approximate requirements.
- Predict the Future of Cab rides and save the cost of both ends.

- A person is only relay on that service but on the meantime, he did not get ride because of shortage of cabs or the cabs are far away from his location so the customer is not satisfied this drawback is solved by advance prediction.
- On office days companies know people want a ride from house to office in morning and from office to house in evening. The driver not know more than this information. If they already know the expected request of rides from specific area on specific time so they go and wait that place before request and get their customer's ride and satisfy and their customers.

1.3. Area of Research

Regression models and Scikit-Learn is the new emerging area of research. We utilize the Scikit-Learn a popular library used in Data mining and machine learning , to map the high dimensional features into a smaller subspace by providing the Algorithms like linear regression, decision tree regression/Random Forest and polynomial regression . In the experiment, we show that the regression method enhances the prediction accuracy significantly. Furthermore, with regression, our model also automatically discovers the similarities among the supply-demand patterns of different areas and timeslots. Moreover, our framework is highly flexible and extendable. Based on our framework, it is very easy to utilize multiple data sources (e.g., car-hailing orders, weather and traffic data) to achieve a high accuracy. We conduct extensive experimental evaluations, which show that our framework provides more accurate prediction results than the existing methods.

1.4. Motivation:

Motivation of this project to facilitate the User and ride giving companies because this is a daily life need so the cab system is introduced. Using this module companies knows the number of demanded rides from specific area according to temperature, time, days, area and environment. The companies and user both facilitated by this module, users by saving his time and coma pines by saving fuel and satisfy the customer's approximate requirements. We will try to maximize the Accuracy of Predicting Gap between Demand and supply.

1.5. Problem Statement:

Now a day most of the people in big cities use rides to reach on their destination but the lack of cabs it is very time consuming sometimes the cab is far away from the user so it takes time to reach the source location which is a drawback of that companies. This is a serious issue to be solved because time wasting ride is not preferred. As the driver only knows some specific areas and time from where they are expected ride like a person is only relay on that service but on the meantime, he did not get ride because of shortage of cabs or the cabs are far away from his location so the customer is not satisfied or in office days companies know people want a ride from house to office in morning and from office to house in evening. The driver not know more than this information. If they already know the expected request of rides from specific area on specific time so they go and wait that place before request and get their customer's ride and satisfy and their customers.

Suppose the current date is the d-th day and the current time slot is t. Given the past order data and the past environment data, our goal is to predict the supply-demand gap gap d,t a for every area a, i.e., the supply-demand gap in the next 10 minutes.

1.6. Project description:

Uber has changed the face of taxi ridership, making it more convenient and comfortable for riders. But, there are times when customers are left unsatisfied because of shortage of vehicles which ultimately led to Uber adopting surge pricing. It's a very difficult task to forecast number of riders at different locations in a city at different points in time. This gets more complicated with changes in weather. In this paper we attempt to estimate the number of trips per borough on a daily basis in New York City. We add an exogenous factor, weather to this analysis to see how it impacts the changes in number of trips. We fetched worth data (approximately 4 million records) of Uber rides from Kaggle. We also gathered weather data (from Weather Underground) for the same period . we attempted to analyze Uber , time and weather data together to estimate the change in the number of trips per borough due to changing weather conditions[2].

Uber has recently been introducing novel practices in urban taxi transport. Journey prices can change dynamically in almost real time and also vary geographically from one area to another in a city, a strategy known as surge pricing. In recent years, data mining research has focused primarily on the mining of spatial trajectories for the development of routing, navigation and mapping applications. While taxi spatial trajectory data has also been exploited heavily in this context [7], there is only little work on the mining of taxi mobility data in the light of other layers of data and in particular those that can provide valuable information on the economic costs of taxi journeys

The purpose of this project is to analyze the impact of weather on Uber Ridership. We chose this topic in particular due to the raising concern among customers about Uber adopting surge pricing to deal with the growing demand. We hypothesized that change in weather will affect the number of Uber rides and through our analysis, we found out that the number of rides increased on an average basis in case of any weather event than a normal day. The dataset compiled for this project serves as a foundation for additional research. Analyzing at least a year worth of data will bring further insights. Demand can be more accurately predicted if the actual number of rides requested information is available along with the number of live rides.

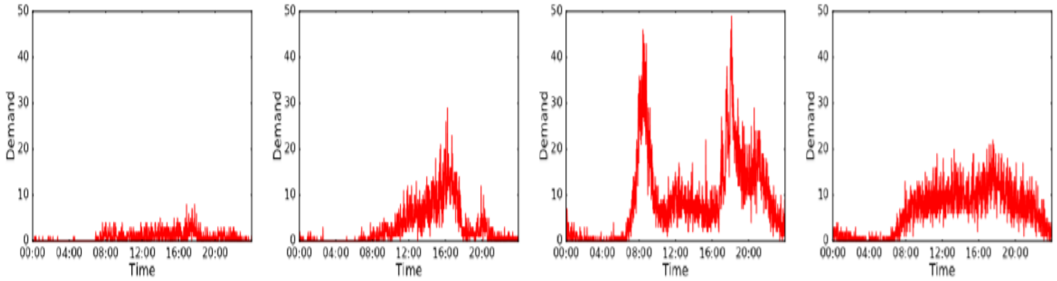
The online car-hailing service has gained great popularity all over the world. As more passengers and more drivers use the service, it becomes increasingly more important for the the car-hailing service providers to effectively schedule the drivers to minimize the waiting time of passengers and maximize the driver utilization, thus to improve the overall user experience.

we study the problem of predicting the real-time car-hailing supply-demand, which is one of the most important component of an effective scheduling system. Our objective is to predict the gap between the car-hailing supply and demand in a certain area. Based on the prediction, we can balance the supply-demands by scheduling the drivers in advance. We present an end-to-end framework called Regression Models. Our approach can automatically discover complicated supply-demand patterns from the car-hailing service data while only requires a minimal amount hand-crafted features. Moreover, our framework is highly flexible and extendable. Based on our framework, it is very easy to utilize multiple data sources (e.g., car-hailing orders,time, weather and traffic data) to achieve a high accuracy. We conduct extensive experimental evaluations, which show that our framework provides more accurate prediction results than the existing methods.

Online car-hailing apps/platforms have emerged as a novel and popular means to provide on-demand transportation service via mobile apps. To hire a vehicle, a passenger simply types in her/his desired pick up location and destination in the app and sends the request to the service provider, who either forwards the request to some drivers close to the pick up location, or directly schedule a close-by driver to take the order. Comparing with the traditional transportation such as the subways and buses, the online car-hailing service is much more convenient and flexible for the passengers. Furthermore, by incentivizing private cars owners to provide car-hailing services, it promotes the sharing economy and enlarges the transportation capacities of the cities. Several car-hailing mobile apps have gained great popularities all over the world, such as Uber, Didi, and Lyft. Large number of passengers are served and volume of carhailing orders are generated routinely every day. For example, Didi, the largest online car-hailing service provider in China, handles around 11 million orders per day all over China. [1]

As a large number of drivers and passengers use the service, several issues arise: Sometimes, some drivers experience a hard time to get any request since few people nearby call the rides ; At the same time, it is very difficult for some passengers to get the ride, in bad weather or rush hours, because the demand in the surrounding areas significantly exceeds the supply. Hence, it is an very important yet challenging task for the service providers to schedule the drivers in order to minimize the waiting time of passengers and maximize the driver utilization. One of the most important ingredient of an effective driver scheduler is the supply-demand prediction. If one could predict/estimate how many passengers need the ride service in a certain area in some future time slot and how many close-by drivers are available, it is possible to balance the supply-demands in advance by dispatching the cars, dynamically adjusting the price, or recommending popular pick-up locations to some drivers. we study the problem of predicting the car-hailing supply-demand. More concretely, our goal is to predict the gap between the car-hailing supply and demand (i.e., max(0,demand−supply)) for a certain area in the next few minutes. Our research is conducted based on the online car-hailing order data of Uber. To motivate our approach, we first present some challenges of the problem and discuss the drawback of the current standard practice for such problem.

- The car-hailing supply-demand varies dynamically due to different geographic locations and time intervals. For example, in the morning the demand tends to surge in the residential areas whereas in the evening the demand usually tends to surge in the business areas. Furthermore, the supply-demand patterns under different days of a week can be extremely different. Prior work usually distinguishes different geographic locations, time intervals or days of week and build several sub-slots of 5 mints. Treating the order data separately and creating many sub slots , and may not suffer from the lack of training data since each sub-model is trained over a small part of data.
- The order data contains multiple attributes such as the timestamp, passenger ID, start location, destination etc, as well as several “environment” factors, such as the traffic condition, weather condition etc. These attributes together provide a wealth of information for supply-demand prediction. However, it is nontrivial how to use all the attributes in a unified model. Currently, the most standard approach is to come up with many “hand-crafted” features (i.e., feature engineering), and fit them into an off-the-shelf learning algorithm such as logistic regression or random forest. However, feature engineering typically requires substantial human efforts (it is not unusual to see data science/ machine learning practitioners creating hundreds different features in order to achieve a competitive performance) and there is little general principle how this should be done. Some prior work only keeps a subset of attributes for training, such as the timestamp, start location and drops other attributes [2]– [6]. While this makes the training easier, discarding the attributes leads to the information loss and reduces the prediction accuracy. To provide some intuitions for the readers and to illustrate the challenges, we provide an example in Fig.1.6.



(a) First area on March 9th (b) First area on March 13th (c) Second area on March 9th (d) Second area on March 13th

Car-hailing demands under four different situations.

[2]. Example 1: Fig. 1.6. shows the demand curves for two areas on March 9th (Wednesday) and March 13th (Sunday). From the figure, we can see very different patterns under different timeslots for the two areas. For the first area, few people require the car-hailing services on Wednesday. However, the demand increased sharply on Sunday. Such pattern usually occurs in the entertainment area. For the second area, we observe a heavy demand on Wednesday, especially during two peak hours around 8

o'clock and 19 o'clock (which are the commute times for most people during the weekdays). On Sunday, the demand of car-hailing services on this area reduced significantly. Moreover, the supply-demand patterns change from day to day. There are many other complicated factors that can affect the pattern, and it is impossible to list them exhaustively. Hence, simply using the average value of historic data or empirical supply-demand patterns can lead to quite inaccurate prediction results, which we show in our Mythology section.

- We proposed an end-to-end framework based on a meachine learning approach. Our approach can automatically learn the patterns across different spatio-temporal attributes (e.g. geographic locations, time intervals and days of week), which allows us to process all the data in a unified model, instead of separating it into the sub-models manually. Comparing with other off-the-shelf methods (e.g., gradient boosting, random forest), our model requires a minimal amount feature-engineering (i.e.,Regression-crafted features), but produces more accurate prediction results.
- We devise Regression models i.e Linear regression, decision tree regression/Random Forest and polynomial regression , which is inspired by the deep residual network (ResNet) proposed very recently by He et al. [3] for image classification. The new network structure allows one to incorporate the "environment factor" data such as the weather and traffic data very easily into our model. On the other hand, we can easily utilize the multiple attributes contained in the order data without much information loss
- We utilize the Scikit-Learn[4], a popular library used in Data mining and machine learning , to map the high dimensional features into a smaller subspace by providing the Algorithms like linear regression, decision tree/Random Forest regression and polynomial regression . In the experiment, we show that the regression method enhances the prediction accuracy significantly. Furthermore, with regression, our model also automatically discovers the similarities among the supply-demand patterns of different areas and timeslots.
- We further study the extendability of our model. In real applications, it is very common to incorporate new extra attributes or data sources into the already trained model. Typically we have to re-train the model from the scratch. However, the machine learning component of our model can utilize these already trained parameters.
- Finally, we conduct extensive experiments on a large scale real dataset of car-hailing orders from uber. The experimental results show that our algorithm outperforms the existing method significantly. The prediction error of our algorithm is. 20% lower than the best existing method.

1.7. Feasibility Study

Our research is feasible one example is, a large organization cannot read all data of customer and not predict by themselves , and they can use gap prediction by the machine to make graphs for decision making.

1.8. Risks Involved:

The main challenge in the Demand supply gap predictor system is the accuracy and the quality of the output.

1.9. Solution Application Areas

we are going to predict demand supply gap which are very helpful in decision making.it is itself a solution Application which solve the real life problem.

1.10. Tools/Technology

1.10.1. Tools

- Anaconda
- Spyder

1.10.2. Language

- iPython 2.7

1.10.3. Toolkit & OS

- Jupiter notebook
- Windows 10

2. Literature Review

A literature review publication is a scholarly paper which contains a summarization of all the current published knowledge about a specific topic. Literature reviews are very important in organizing the current knowledge about a specific topic, and help define future studies. Specifically, by reading a literature review, scholars are able to draw new and original insights from previously published literature. They may be able to find a fresh and original research questions, identify a gap in the literature or make surprising new connections. The basis of a literature review is from secondary sources as it does not report new or original experimental work.

2.0. Related Real Time Application

Data Mining (DM) methods are being increasingly used in prediction with time series data, in addition to traditional statistical approaches. This paper presents a literature review of the use of DM with time series data, focusing on short- time stocks prediction. This is an area that has been attracting a great deal of attention from researchers in the field. The main contribution of this paper is to provide an outline of the use of DM with time series data, using mainly examples related with short-term stocks prediction. This is important to a better understanding of the field. Some of the main trends and open issues will also be introduced.

2.0.1. Data Mining with Time Series

Data Since the seminal paper of Fayyad in 1996 [10], the Data Mining (DM) area has attracted a great deal of interest and can nowadays be considered as an established field. DM applications can be found in a diversified range of application domains. One important application domain is that of time series data. “A time-series data set consists of sequences of numeric values obtained over repeated measurements of time. The values are typically measured at equal time intervals (e.g., every minute, hour, or day)”. [10]. The referred measures can be taken over one variable or several variables—univariate or multivariate time series.

2.0.2. Data Mining with Time Series Data Applications

DM with time series data is popular and many applications can be found in the literature, for instance, for earthquake forecasting [12], characterization of ozone behavior [13], or flood prediction [14]. Other application example is that of financial decision making. A decision support tool for financial forecasting, named as EDDIE, is presented in [15]. In [16], a new architecture that implements a binary neural network, AURA, to produce discrete probability distribution as forecasts, using high frequency data sets, is presented. The use of support vector machines and back propagation neural networks to predict credit ratings is presented in [17]. One important application concerns short-term stocks prediction, which is the main focus of this paper. In [18], an approach to the paradox of obtaining better results with long-horizon forecasts than with short-horizon fore- casts is presented, and it is claimed that the paradox is solved, since the proposed model obtains promising results. Nevertheless, there is a great deal of interest from investors in short-horizon forecasts, thus the authors con- sider that research focusing on this issue is important, namely in using data mining with time series for short- term stocks prediction.

2.0.3. Data Mining Techniques Used with Time Series

Data for Short-Term Stocks Prediction Several DM techniques are used with time series data in order to obtain short-term stocks prediction. An interesting approach to portfolio management, using the Gaussian temporal factor analysis technique, is introduced in [19]. Neural networks are one of the most popular techniques for stocks prediction. [19] are some examples. In [15] rough sets and classification trees are used, as well. Rough sets are also used in [15]. Support Vector Machines are used in [15]. There were not yet been given strong evidences of some technique being better than other, but nonlinear models are more popular.

2.1. Techniques of Predictor Regressors:

Predictive modeling is a name given to a collection of mathematical techniques having in common the goal of finding a mathematical relationship between a target, response, or "dependent" variable and various predictor or "independent" variables with the goal in mind of measuring future values of those predictors and inserting them into the mathematical relationship to predict future values of the target variable. Because these relationships are never perfect in practice, it is desirable to give some measure of uncertainty for the predictions, typically a prediction interval that has some assigned level of confidence like 95%[8].

Regression analysis establishes a relationship between a dependent or outcome variable and a set of predictors. Regression, as a data mining technique, is supervised learning. Supervised learning partitions the database into training and validation data. The techniques used in this research were simple linear regression and multiple linear regression. Some distinctions between the use of regression in statistics verses data mining are: in statistics The data is a sample from a population , but in Data Mining The data is taken from a large database (e.g. 10 million records)[8].

Also in statistics The regression model is constructed from a sample, but in Data Mining the regression model is constructed from a portion of the data (training data). Predictive analytics encompasses a variety of techniques from statistics, data mining and game theory that analyze current and historical facts to make predictions about future events. The variety of techniques is usually divided in three categories: predictive models, descriptive models and decision models. Predictive models look for certain relationships and patterns that usually lead to a certain behavior, point to fraud, predict system failures, assess credit worthiness, and so forth. By determining the explanatory variables, you can predict outcomes in the dependent variables. there are decision models that use optimization techniques to predict results of decisions. This branch of predictive analytics leans particularly heavily on operations research, including areas such as resource optimization, route planning and so forth.

There are three techniques of Predictor Regressors which are machine learning approach of linear regression approach polynomial regression approach and decision tree/Random Forest regression approach[9].

2.1.1. DATA MINING

It is no surprise that data mining, as a truly interdisciplinary subject, can be defined in many different ways. To refer to the mining of gold from rocks or sand, we say gold mining instead of rock or sand mining. Analogously, data mining should have been more appropriately named "knowledge mining from data," which is unfortunately somewhat long. However, the shorter term, knowledge mining may not reflect the emphasis on mining from large amounts of data. Nevertheless, mining is a vivid term characterizing the process that finds a small set of precious nuggets from a great deal of raw material (Figure 2.1.1)[8]. Thus, such a misnomer carrying both "data" and "mining" became a popular choice. In addition, many other terms have a similar meaning to data mining—for example, knowledge mining from data, knowledge extraction, data/pattern analysis, data archaeology, and data dredging. Many people treat data mining as a synonym for another popularly used term, knowledge discovery from data.

The knowledge discovery process is shown in Figure 2.1.1[8] as an iterative sequence of the following steps:

- Data cleaning (to remove noise and inconsistent data)
- Data integration (where multiple data sources may be combined)
- Data selection (where data relevant to the analysis task are retrieved from the Files)
- Data transformation (where data are transformed and consolidated into forms appropriate for mining by performing summary or aggregation operations)4
- Data mining (an essential process where intelligent methods are applied to extract data patterns)
- Pattern evaluation (to identify the truly interesting patterns representing knowledge based on interestingness measures)
- Knowledge presentation (where visualization and knowledge representation techniques are used to present mined knowledge to users).

Steps 1 through 4 are different forms of data preprocessing, where data are prepared for mining. The data mining step may interact with the user or a knowledge base. The interesting patterns are presented to the user and may be stored as new knowledge in the knowledge base. The preceding view shows data mining as one step in the knowledge discovery process, albeit an essential one because it uncovers hidden patterns for evaluation. However, in industry, in media, and in the research milieu, the term data mining is often used to refer to the entire knowledge discovery process (perhaps because the term is shorter than knowledge discovery from data). Therefore, we adopt a broad view of data mining functionality: Data mining is the process of discovering. interesting patterns and knowledge from large amounts of data. The data sources can include databases, data warehouses, the Web, other information repositories, or data that are streamed into the system dynamically. The development of Information Technology has generated large amount of databases and huge data in various areas. The research in databases and information technology has given rise to an approach to store and manipulate this precious data for further decision making. Data mining is a process of extraction of useful information and patterns from huge data. It is also called as knowledge discovery process, knowledge mining from data, knowledge extraction or data/pattern analysis.

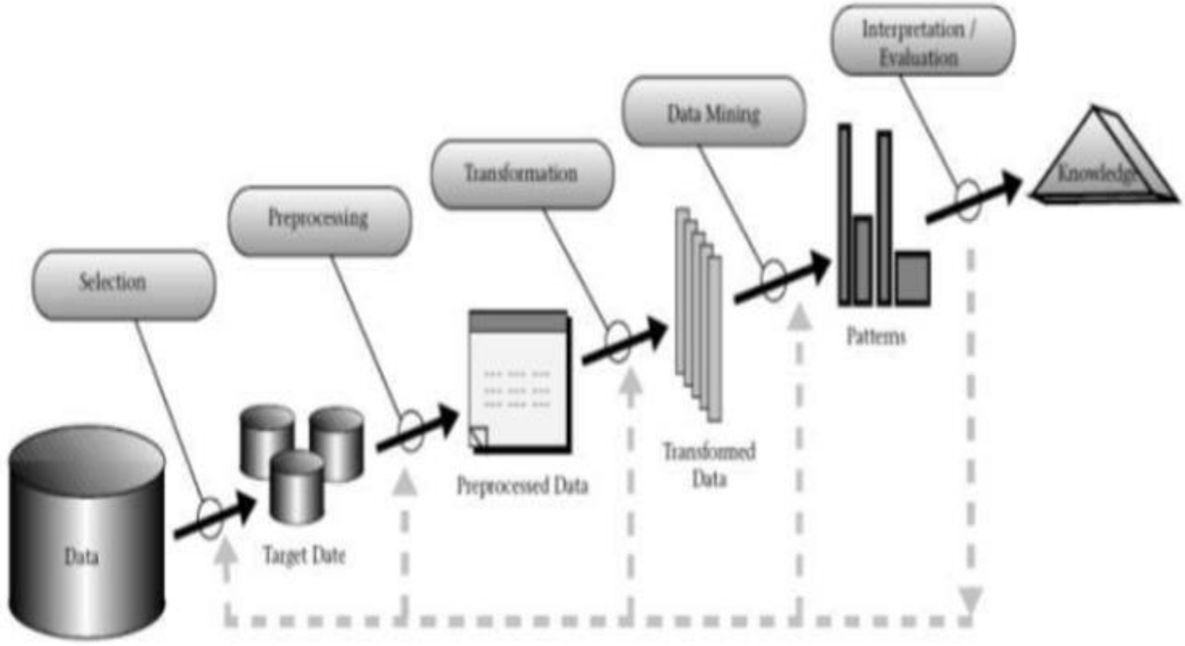


fig 2.1.1 *Process of knowledge discovery process*

[2]

2.2. DATA MINING ALGORITHMS AND TECHNIQUES

Various algorithms and techniques like Classification, Clustering, Regression, Artificial Intelligence, Neural Networks, Association Rules, Decision Trees, Genetic Algorithm, Nearest Neighbor method etc., are used for knowledge discovery from databases.

2.2.1. CLUSTERING

Clustering can be said as identification of similar classes of objects. By using clustering techniques we can further identify dense and sparse regions in object space and can discover overall distribution pattern and correlations among data attributes. Classification approach can also be used for effective means of distinguishing groups or classes of object but it becomes costly so clustering can be used as preprocessing approach for attribute subset selection and classification. For example, to form group of customers based on purchasing patterns, to categories genes with similar functionality.

2.2.2. PREDICATION

Regression technique can be adapted for predication. Regression analysis can be used to model the relationship between one or more independent variables and dependent variables. In data mining independent variables are attributes already known and response variables are what we want to predict. Unfortunately, many real-world problems are not simply prediction. For instance, sales volumes, stock prices, and product failure rates are all very difficult to predict because they may depend on complex interactions of multiple predictor variables. Therefore, more complex techniques (e.g., logistic regression, decision trees, or neural nets) may be necessary to forecast future values. The same model types can often be used for both regression and classification. For example, the CART (Classification and Regression Trees) decision tree algorithm can be used to build both classification trees (to classify categorical response variables) and regression trees (to forecast continuous response variables). Neural networks too can create both classification and regression models. Types of regression methods[8]:

- Linear Regression
- Multivariate Linear Regression
- Nonlinear Regression
- Multivariate Nonlinear Regression
- Decision Tree Regression/Random Forest
- Polynomial Regression

2.3. Regression models using Scikit Learn

scikit-learn is an increasingly popular machine learning library. Written in Python, it is designed to be simple and efficient, accessible to non-experts, and reusable in various contexts. In this paper, we present and discuss our design choices for the application programming interface (API) of the project. In particular, we describe the simple and elegant interface shared by all learning and processing units in the library and then discuss its advantages in terms of composition and reusability. The paper also comments on implementation details specific to the Python ecosystem and analyzes obstacles faced by users and developers of the library. The ambition of the project is to provide efficient and well-established machine learning tools within a programming environment that is accessible to non-machine learning experts and reusable in various scientific areas. The project is not a novel domain-specific language, but a library that provides machine learning idioms to a generalpurpose high-level language. Among other things, it includes classical learning algorithms, model evaluation and selection tools, as well as preprocessing procedures. The library is distributed under the simplified BSD license, encouraging its use in both academic and commercial settings[20].

2.3.1. Core API

All objects within scikit-learn share a uniform common basic API consisting of three complementary interfaces: an estimator interface for building and fitting models, a predictor interface for making predictions and a transformer interface for converting data. In this section, we describe these three interfaces, after reviewing our general principles and data representation choices[20].

2.3.1.1. Predictors

The predictor interface extends the notion of an estimator by adding a predict method that takes an array X test and produces predictions for X test, based on the learned parameters of the estimator (we call the input to predict “X test” in order to emphasize that predict generalizes to new data). In the case of supervised learning estimators, this method typically returns the predicted labels or values computed by the model. Continuing with the previous example, predicted labels for X test can be obtained using the following snippet:

```
y_pred = clf.predict(X_test)
```

Some unsupervised learning estimators may also implement the predict interface. The code in the snippet below fits a k-means model with k = 10 on training data X train, and then uses the predict method to obtain cluster labels (integer indices) for unseen data X test.

```
from sklearn.cluster import KMeans

km = KMeans(n_clusters=10)

km.fit(X_train)

 clust_pred = km.predict(X_test)
```

Apart from predict, predictors may also implement methods that quantify the confidence of predictions. In the case of linear models, the decision function method returns the distance of samples to the separating hyperplane. Some predictors also provide a predict proba method which returns class probabilities. Finally, predictors must provide a score function to assess their performance on a batch of input data. In supervised estimators, this method takes as input arrays X test and y test and typically computes the coefficient of determination between y test and predict(X test) in regression, or the accuracy in classification. The only requirement is that the score method return a value that quantifies the quality of its predictions (the higher, the better). An unsupervised estimator may also expose a score function to compute, for instance, the likelihood of the given data under its model[20].

2.3.2. Transformer

Since it is common to modify or filter data before feeding it to a learning algorithm, some estimators in the library implement a transformer interface which defines a transform method. It takes as input some new data X test and yields as output a transformed version of X test. Preprocessing, feature selection, feature extraction and dimensionality reduction algorithms are all provided as transformers within the library. In our example, to standardize the input X train to zero mean and unit variance before fitting the logistic regression estimator

Furthermore, every transformer allows fit (X train).transform(X train) to be written as fit transform(X train). The combined fit transform method prevents repeated computations. Depending on the transformer, it may skip only an input validation step, or in fact use a more efficient algorithm for the transformation. In the same spirit, clustering algorithms provide a fit predict method that is equivalent to fit followed by predict, returning cluster labels assigned to the training samples [20].

3. Project Methodology

3.1. Introduction

This chapter will cover the details explanation of methodology that is being used to make this project complete and working well. Many methodology or findings from this field mainly generated into journal for others to take advantages and improve as upcoming studies. The method is use to achieve the objective of the project that will accomplish a perfect result. In order to evaluate this project, the methodology based on System Development Life Cycle (SDLC), generally three major step, which is planning, implementing and analysis. [21]

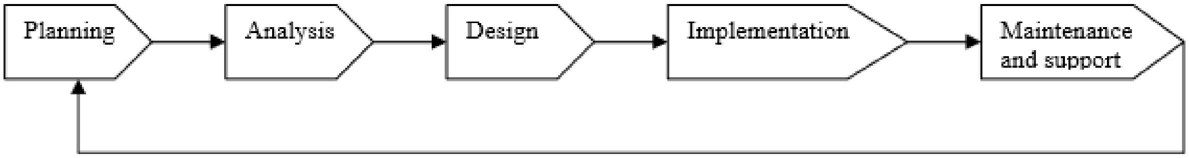


Figure 3.1: SLDC Phase

This final year project used three major steps to implement project starting from planning, implementing and testing. All the methods used for finding and analyzing data regarding the project related

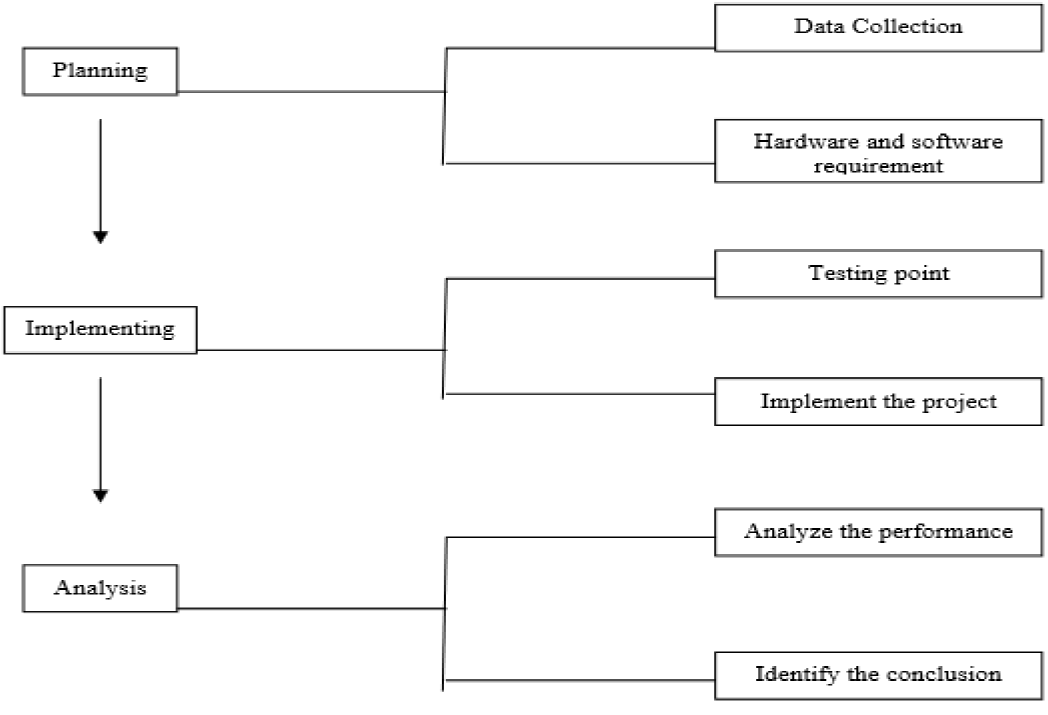


Figure 3.2: Steps of Methodology

3.2 Planning

To identify all the information and requirement such as hardware and software, planning must be done in the proper manner. The planning phase have two main elements namely **data collection** and the **requirements** of hardware and software.

3.2.1. Data collection

Data collection is a stage in any area of study. At this stage we planned about the projects resources and requirements, literature studies and schedule to get more information in this study. All the materials are collected from journal, texts book and research papers gathered from libraries and Internet. Meanwhile the dataset is collected from Kaggle.Several car-hailing mobile apps have gained great popularities all over the world, such as Uber, Didi, and Lyft. Large number of passengers are served and volume of carhailing orders are generated routinely every day. For example, Didi, the largest online car-hailing service provider in China, handles around 11 million orders per day all over China. [1]

3.2.1.1. Data Format

The training set contains three consecutive weeks of data for City M in 2016, and you need to forecast the supply-demand gap for a certain period in the fourth and fifth weeks of City M. The test set contains the data of half an hour before the predicted time slot. The specific time slots where you need to predict the supply-demand gap are shown in the explanation document in the test set.
The Order Info Table, Weather Info Table and POI Info Table are available in the database, while the District Definition
Table and Traffic Jam Info Table are derived from other tables in the database. All sensitive data has been anonymized.

3.2.1.1.1. Order Info Table

Table1:OrderInfo

Field	**Type**	**Meaning**	**Example**
order id	string	orderID	70fc7c2bd2caf386bb50f8fd5dfef0cf
driver-id	string	driverID	56018323b921dd2c5444f98b45509de
passenger id	string	userID	238de35f44bbe8a67bdea86a5b0f4719
start district hash	string	departure	d4ec2125aff74eded207d2d915ef682f
dest-district hash	string	destination	929ec6c160e6f52c20a4217c7978f681
Price	double	Price	37.5
Time	string	Timestampoftheorder	2016-01-1500:35:11

The Order Info Table shows the basic information of an order, including the passenger and the driver (if driver id =NULL, it means the order was not answered by any driver), place of origin, destination, price and time. The fields order id, driver id, passenger id, start hash, and dest hash are made not sensitive.

3.2.1.1.2. District Info Table

The District Info Table shows the information about the districts to be evaluated in the contest. You need to do the prediction given the districts from the District Definition Table. In the submission of the results, you need to map the district hash value to district mapped ID.

Table2:DistrictInfo

Field	**Type**	**Meaning**	**Example**
district hash	string	Districthash	90c5a34f06ac86aee0fd70e2adce7d8a
district id	string	DistrictID	1

3.2.1.1.3. Weather Info Table

The Weather Info Table shows the weather info every 10 minutes each city. The weather field gives the weather conditions such as sunny, rainy, and snowy etc; all sensitive information has been removed. The unit of temperature is Celsius degree, and PM2.5 is the level of air pollutions.

Table3:WeatherInfo

Field	**Type**	**Meaning**	**Example**
Time	string	Timestamp	2016-01-1500:35:11
Weather	int	Weather	7
temperature	double	Temperature	-9
PM2.5	double	pm25	66

3.2.1.1.4. Test Data

All the tables in test data are same except the order table it has the following fields :

Table 4: Order Info (Test)

Field	**Type**	**Meaning**	**Example**
order id	string	order ID	70fc7c2bd2caf386bb50f8fd5dfef0cf
passenger id	string	user ID	238de35f44bbe8a67bdea86a5b0f4719
start district hash	string	departure	d4ec2125aff74eded207d2d915ef682f
dest district hash	string	destination	929ec6c160e6f52c20a4217c7978f681
Time	string	Timestamp of the order	2016-01-15 00:35:11

3.2.1.2. Definition and Evaluation Criteria

3.2.1.2.1. Definition

A passenger calls a ride(request)by entering the place of origin and destination and clicking "Request Pickup" on the Didi app. A driver answers the request (answer) by taking the order. Didi divides a city into *n* non-overlapping square districts $D = d_1, d_2 ..., d_n$ and divides one day uniformly into 144 time slots $t_1, t_2, ..., t_{144}$, each 10 minutes long. In district d_i, and time slot t_j , the number of passengers' requests is denoted as r_{ij}, and drivers' answers as a_{ij}. In district d_i and time slot t_j the demand is denoted as $demand_{ij} = r_{ij}$ and the supply as $supply_{ij} = a_{ij}$, and the demand supply gap is:gap_{ij} : $gap_{ij} = r_{ij} - a_{ij}$. Given the data of every district d_i and time slot t_j, you need to predict $gap_{ij}, \forall d_i \in D$.

3.2.1.2.2. Evaluation Metrics

Given *i* districts and *j* time slots, for district d_i in time slot t_j, suppose that the real supply-demand gap is gap_{ij} , and predicted supply-demand gap is s_{ij}, then:

$$MAE = \frac{1}{n} \sum_{d_i} \left(\frac{1}{q} \sum_{t_i} |gap_{ij} - s_{ij}| \right)$$

The lowest MAE will be the best.

The detailed description of each field is as follows:

Table5:Description

Dataname	**Datatype**	**Example**
DistrictID	string	1,2,3,4(thesameasdistrictmappingID)
Timeslot	string	2016-01-23-1T(hefirsttimeslotonJan.23rd, 2016)
Predictionvalue	double	6.0

3.2.2. Requirements

3.2.2.1 Software requirements

Tools	***Description***
Anaconda	IDE
OS	*Windows 10 / Linux*
Python	Language
JupterNotebook	Interface

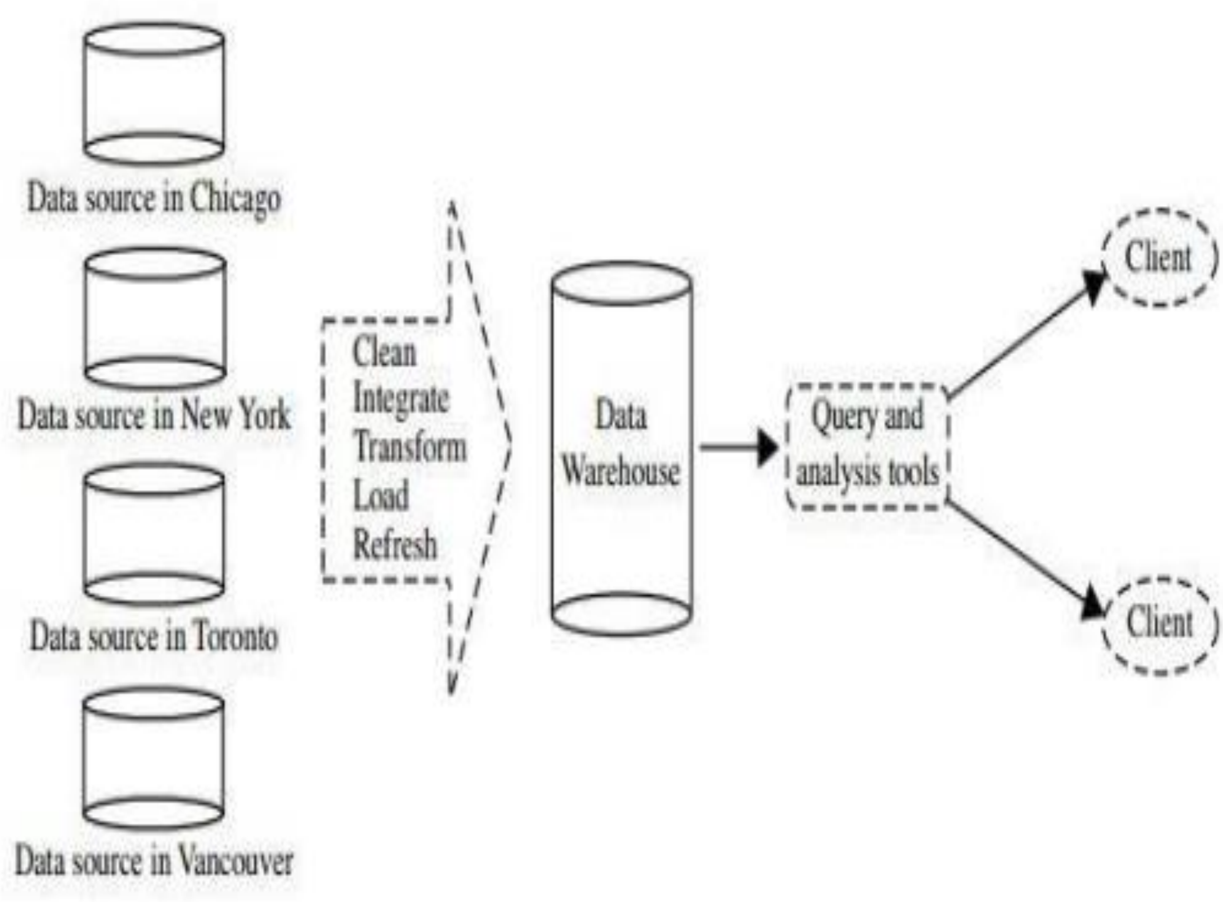
3.2.2.1 Hardware requirements

- It depends on the task and the type of architecture. we want to train, obviously the bigger data , the more memory and processing power we will need. Additionally, most machine learning libraries and toolkits come with GPU capabilities.
- Quad core Intel Core i7 Skylake or higher (Dual core is not the best for this kind of work, but manageable)
- 16GB of RAM (8GB is okay but not for the performance you may want and or expect)

3.3. Implementation

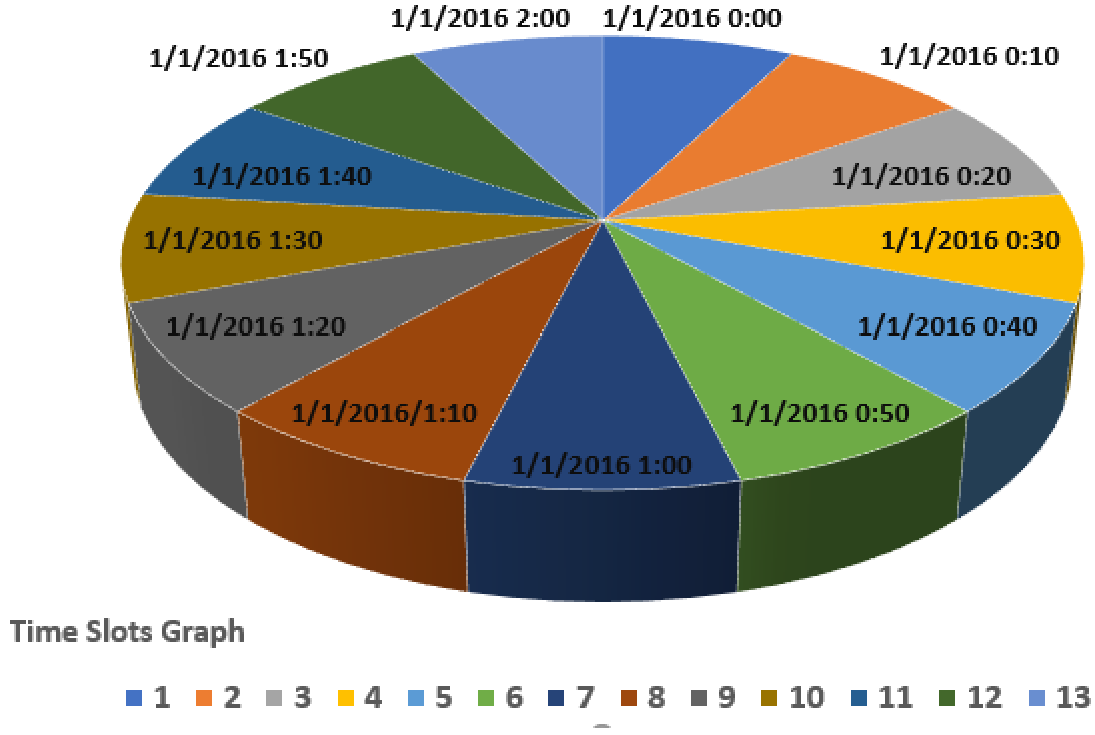
3.3.1. DATA WAREHOUSES

Uber is a successful international company with branches around the world. Each branch has its own set of databases. Suppose the president of Uber has asked you to provide an analysis of the company's Gap between the demand and supply by analyzing its previous data. This process we followed are listed below in the fig .[2]



3.3.2. Data Pre-engineering

In our experiment, we use the public dataset released by uber in the Kaggle supply-demand prediction competition. The order dataset contains the car-hailing orders from uber over more than 21 days of 67 cluster regions. The order dataset consists of 14,0000000 orders app. The gaps in our dataset is approximately power-law distributed. The largest gap is as large as 1434. On the other hand, around 48% of test items are supply-demand balanced, i.e., gap = 0. Auxiliary information include weather conditions (weather type, temperature, PM 2.5) and traffic conditions (total amount of road segments under different congestion levels in each area). The training data is from ._order_data_2016-01-02 to ._order_data_2016-01-21 (21 days in total).To construct the training set ,for each area in each training day in 67 cluster(regions), we generate one training item every 10 minutes from 0:20 to 24:00. Thus, we have 67(areas)×21(days)×500000(items appx) = 703,500,000 training items in total by data engineering 10 mints time slot divions each day hava 60 sub records so 703,500,000 x 60=42,210,000,000 training items in total The test data is from order_data_2016-01-23 to only odd days order_data_2016-01-31 (5 days in total). During the test days, the first time slot is 7:30 and the last time slot is 23:30. We select one time slot t every 2 hours from the first time slot unit the last time slot, i.e., t = 7:30, 9:30, 11:30, ..., 23:30. For each time slot t, we generate one test item. We use T to denote the set of test items.

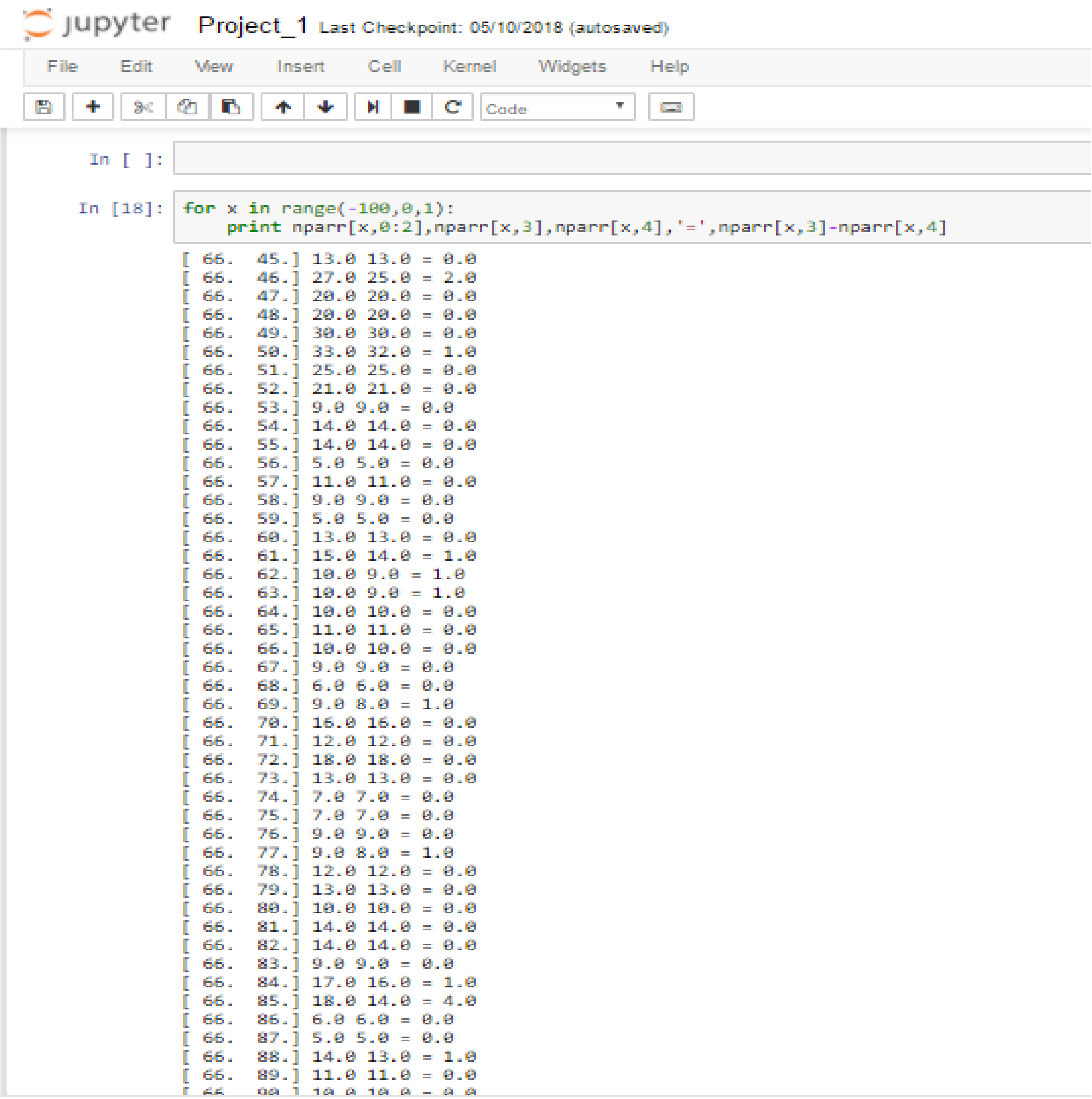


Error Metrics: We evaluate the predicted results using the mean absolute error (MAE) and the root mean squared error (RMSE). Formally, we use predd,t a to denote the predicted value of gapd,t a . Then, the mean absolute error and the root mean squared error can be computed as follows:[2]

$$\mathsf{MAE} = \frac{1}{|T|} \sum_{(a,d,t)\in T} \left| \text{gap}_a^{d,t} - \text{pred}_a^{d,t} \right|$$

$$\mathsf{RMSE} = \sqrt{\frac{1}{|T|} \sum_{(a,d,t)\in T} \left(\text{gap}_a^{d,t} - \text{pred}_a^{d,t} \right)^2}.$$

We calculated it and see the gap are produced in these records.We just show the result of some entries listed below of region 66 we see the gap between demand and supply.



```
for x in range(-100,0,1):
    print nparr[x,0:2],nparr[x,3],nparr[x,4],'=',nparr[x,3]-nparr[x,4]
```

```
[ 66.  45.] 13.0 13.0 = 0.0
[ 66.  46.] 27.0 25.0 = 2.0
[ 66.  47.] 20.0 20.0 = 0.0
[ 66.  48.] 20.0 20.0 = 0.0
[ 66.  49.] 30.0 30.0 = 0.0
[ 66.  50.] 33.0 32.0 = 1.0
[ 66.  51.] 25.0 25.0 = 0.0
[ 66.  52.] 21.0 21.0 = 0.0
[ 66.  53.] 9.0 9.0 = 0.0
[ 66.  54.] 14.0 14.0 = 0.0
[ 66.  55.] 14.0 14.0 = 0.0
[ 66.  56.] 5.0 5.0 = 0.0
[ 66.  57.] 11.0 11.0 = 0.0
[ 66.  58.] 9.0 9.0 = 0.0
[ 66.  59.] 5.0 5.0 = 0.0
[ 66.  60.] 13.0 13.0 = 0.0
[ 66.  61.] 15.0 14.0 = 1.0
[ 66.  62.] 10.0 9.0 = 1.0
[ 66.  63.] 10.0 9.0 = 1.0
[ 66.  64.] 10.0 10.0 = 0.0
[ 66.  65.] 11.0 11.0 = 0.0
[ 66.  66.] 10.0 10.0 = 0.0
[ 66.  67.] 9.0 9.0 = 0.0
[ 66.  68.] 6.0 6.0 = 0.0
[ 66.  69.] 9.0 8.0 = 1.0
[ 66.  70.] 16.0 16.0 = 0.0
[ 66.  71.] 12.0 12.0 = 0.0
[ 66.  72.] 18.0 18.0 = 0.0
[ 66.  73.] 13.0 13.0 = 0.0
[ 66.  74.] 7.0 7.0 = 0.0
[ 66.  75.] 7.0 7.0 = 0.0
[ 66.  76.] 9.0 9.0 = 0.0
[ 66.  77.] 9.0 8.0 = 1.0
[ 66.  78.] 12.0 12.0 = 0.0
[ 66.  79.] 13.0 13.0 = 0.0
[ 66.  80.] 10.0 10.0 = 0.0
[ 66.  81.] 14.0 14.0 = 0.0
[ 66.  82.] 14.0 14.0 = 0.0
[ 66.  83.] 9.0 9.0 = 0.0
[ 66.  84.] 17.0 16.0 = 1.0
[ 66.  85.] 18.0 14.0 = 4.0
[ 66.  86.] 6.0 6.0 = 0.0
[ 66.  87.] 5.0 5.0 = 0.0
[ 66.  88.] 14.0 13.0 = 1.0
[ 66.  89.] 11.0 11.0 = 0.0
[ 66.  90.] 10.0 10.0 = 0.0
```

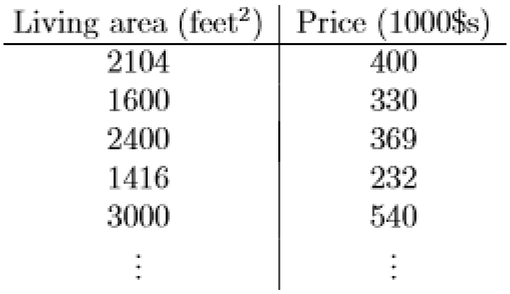
3.3.3. Algorithm implementation:

Regression models predict the value of a dependent numeric variable from the values of independent variables, also referred to as predictors (in statistics, predictors are also referred to as regressors). The regression task is the problem of inducing or learning a regression model from a table of measured values of the dependent and independent variables. The simplest approach to the regression task is linear regression, where the dependent variable is modeled as a linear combination of the predictors. More advanced regression approaches and models include regression and model trees.

3.3.3.1. Regression model Methods

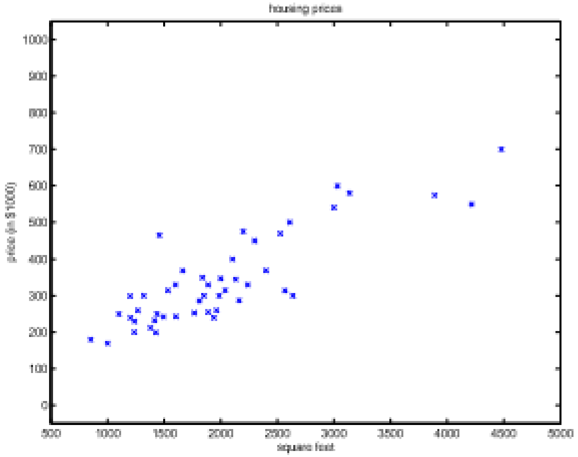
Supervised learning

Let’s start by talking about a few examples of supervised learning problems. Suppose we have a dataset giving the living areas and prices of 47 houses from Portland, Oregon[24]:



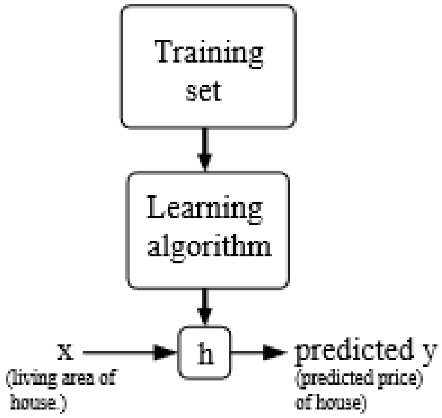
Living area (feet^2)	Price (1000$s)
2104	400
1600	330
2400	369
1416	232
3000	540
⋮	⋮

We can plot this data[24]:



Given data like this, how can we learn to predict the prices of other houses in Portland, as a function of the size of their living areas?

To establish notation for future use, we’ll use x(i) to denote the “input” variables (living area in this example), also called input features, and y(i) to denote the “output” or target variable that we are trying to predict (price). A pair (x(i),y(i)) is called a training example, and the dataset that we’ll be using to learn—a list of m training examples {(x(i),y(i));i = 1,...,m}—is called a training set. Note that the superscript “(i)” in the notation is simply an index into the training set, and has nothing to do with exponentiation. We will also use X denote the space of input values, and Y the space of output values. In this example, X = Y = R. To describe the supervised learning problem slightly more formally, our goal is, given a training set, to learn a function h : X 7→Y so that h(x) is a “good” predictor for the corresponding value of y. For historical reasons, this function h is called a hypothesis. Seen pictorially, the process is therefore like this[24]:



When the target variable that we’re trying to predict is continuous, such as in our housing example, we call the learning problem a regression problem. When y can take on only a small number of discrete values (such as if, given the living area, we wanted to predict if a dwelling is a house or an apartment, say), we call it a classification problem.

A regression is a statistical analysis [3] assessing the association between two variables. It is used to find the relationship between two variables. et.al [16] Neelamadhab Padhy ,and Rasmita defined that is a one kind of predictive model which provides the prediction about the unknown data values by using the known data. There are so many techniques are available like Classification, Regression, Time series analysis, Prediction etc. If a set of random data (x1, y1) T, (x2, y2) T, (x n, y n) T for two numerical variables X and Y, where X is a cause of Y. In this linear regression analysis, the distribution of the random data appears as a straight line in X, Y space when X and Y are perfectly related linearly. This captures a relationship between two variables. This line function can be given as [22].

$\hat{y} = ax + b$

Here, the linear regression model is used to extract the texture features from the correlation in the frequency channel pairs. The energy values of two frequency channels of one of the channel pair in the top ten list are taken from the channel energy matrix M and consider these energy values as the random data (x1, y1)T,(x2,y2)T,………..(x n, y n)T for two variables X and Y represent a straight line in X,Y space. The technique regression is basically used in the case of prediction. In Regression analysis the prediction variables are the continuous variables .So many techniques are available Neural Network, SVM(Support Vector Machine, Linear Regression Decision Tree Regression Polynomial Regression etc .In this project we have used the linear regression Decision Tree Regression Polynomial Regression analysis.

From history of data we predict the Gap Between Demand and supply using Artificial intelligence techniques like datamining and machine learning. If we have millions of records so how we determine the result itself we required some Artificial Intelligence Techniques so solve the Problem. This is a regression problem so we use some Models which we have to train on data There are following Models

- Linear Regression Model
- Decision Tree Regression Model/ Random Forest
- Polynomial Regression Model

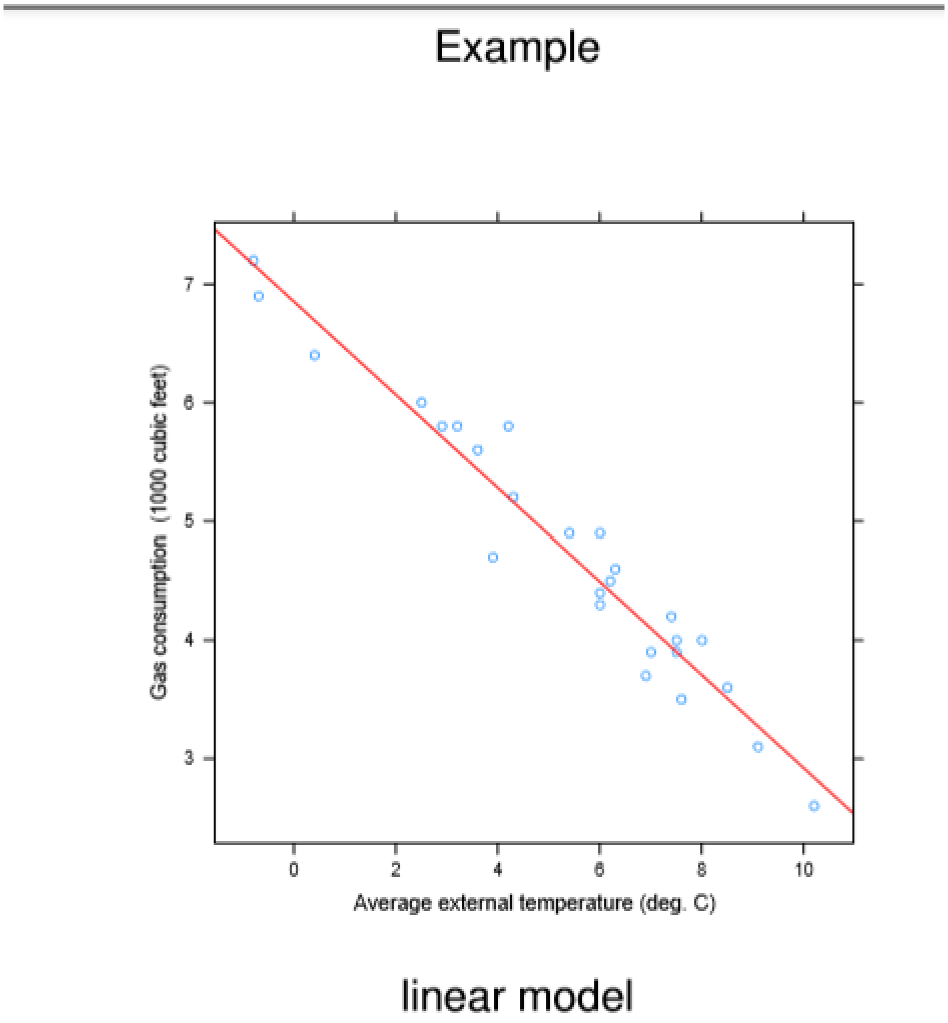
Train these models on data and get the accuracy from each one. we use that model which gives more accuracy by cross checking these model's results. By this regression models we train the machine and predict the Gap between demand and supply.

3.3.3.1.1. Linear Regression Model

Predictive modeling is a name given to a collection of mathematical techniques having in common the goal of finding a mathematical relationship between a target, response, or "dependent" variable and various predictor or "independent" variables with the goal in mind of measuring future values of those predictors and inserting them into the mathematical relationship to predict future values of the target variable. Because these relationships are never perfect in practice, it is desirable to give some measure of uncertainty for the predictions, typically a prediction interval that has some assigned level of confidence like 95%[8].

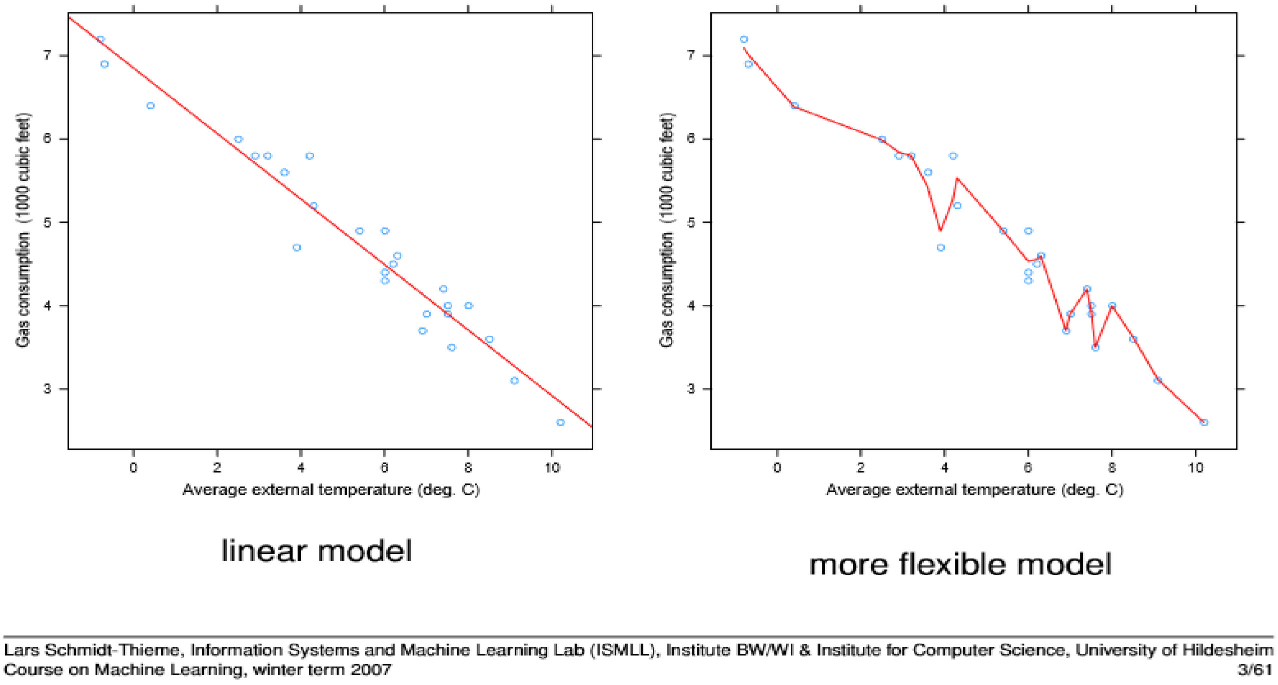
Regression analysis establishes a relationship between a dependent or outcome variable and a set of predictors. Regression, as a data mining technique, is supervised learning. Supervised learning partitions the database into training and validation data. The techniques used in this research were simple linear regression and multiple linear regression. Some distinctions between the use of regression in statistics verses data mining are: in statistics The data is a sample from a population , but in Data Mining The data is taken from a large database (e.g. 10 million records)[8].

Also in statistics The regression model is constructed from a sample, but in Data Mining the regression model is constructed from a portion of the data (training data). Predictive analytics encompasses a variety of techniques from statistics, data mining and game theory that analyze current and historical facts to make predictions about future events. The variety of techniques is usually divided in three categories: predictive models, descriptive models and decision models. Predictive models look for certain relationships and patterns that usually lead to a certain behavior, point to fraud, predict system failures, assess credit worthiness, and so forth. By determining the explanatory variables, you can predict outcomes in the dependent variables. there are decision models that use optimization techniques to predict results of decisions. This branch of predictive analytics leans particularly heavily on operations research, including areas such as resource optimization, route planning and so forth. So one example of linear regression[25]


Example

linear model

But the purpose to move on next approach is shown in next example.

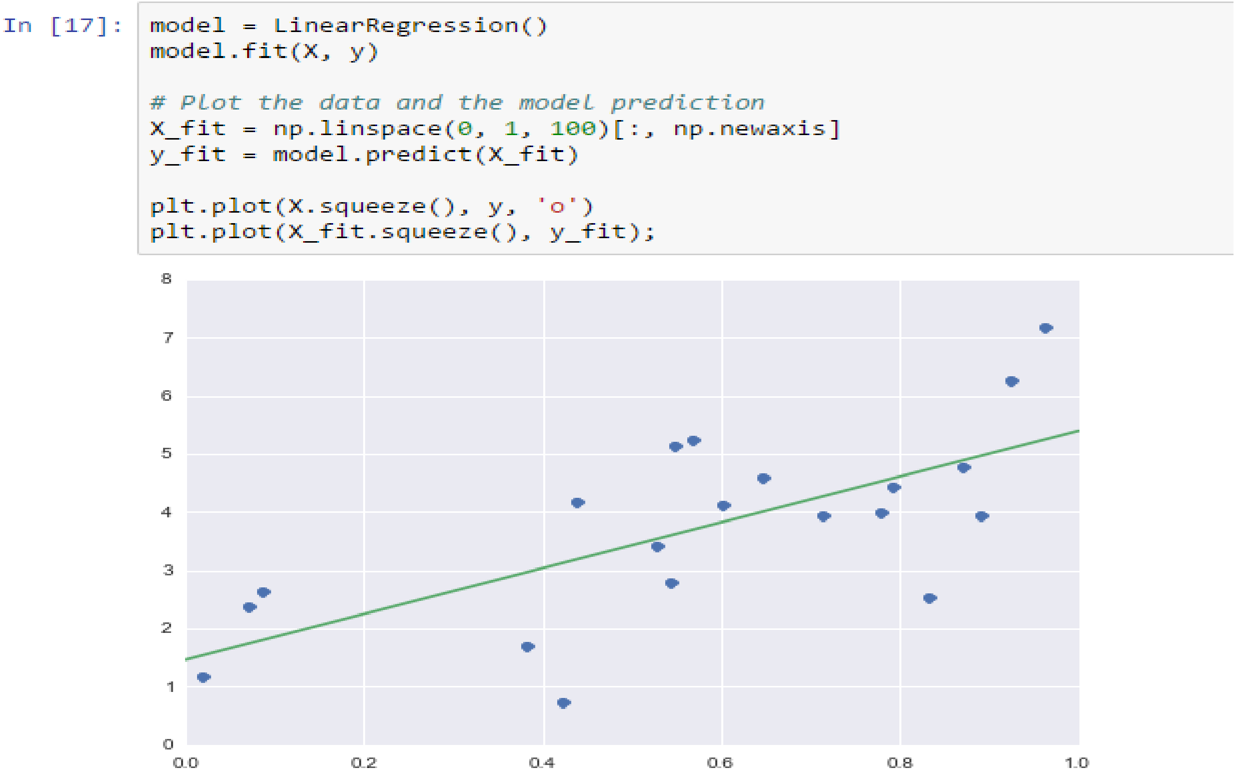


linear model

more flexible model

Lars Schmidt-Thieme, Information Systems and Machine Learning Lab (ISMLL), Institute BW/WI & Institute for Computer Science, University of Hildesheim
Course on Machine Learning, winter term 2007 3/61

We Implement this model on Sickit Learn by these functions.



```
In [17]: model = LinearRegression()
         model.fit(X, y)

         # Plot the data and the model prediction
         X_fit = np.linspace(0, 1, 100)[:, np.newaxis]
         y_fit = model.predict(X_fit)

         plt.plot(X.squeeze(), y, 'o')
         plt.plot(X_fit.squeeze(), y_fit);
```

[26]

If it produced the results on decision tree or polynomial regression then it is more flexible.

3.3.3.1.2. Random Forest

linear regression as a way of making quantitative predictions. In simple linear regression, a real-valued dependent variable Y is modeled as a linear function of a real-valued independent variable X plus noise:

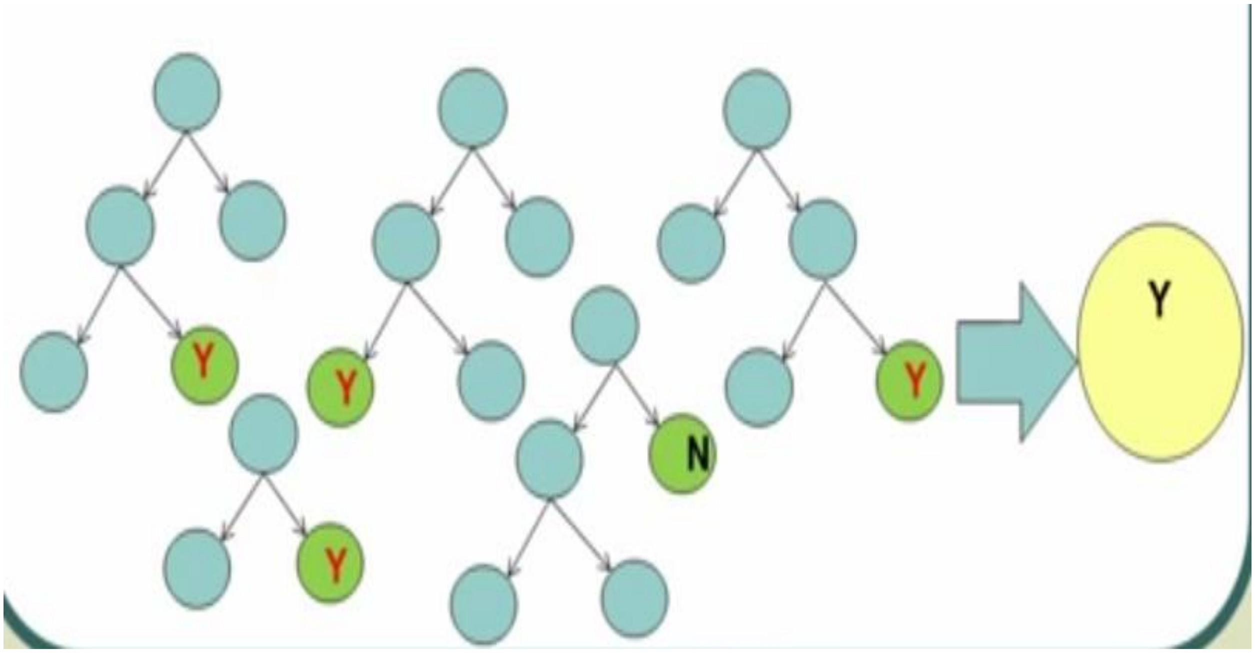
$Y = \beta 0 + \beta 1X + e$

In multiple regression, we let there be multiple independent variables X1,X2,...Xp ≡ X,

Linear regression is a global model, where there is a single predictive formula holding over the entire data-space. When the data has lots of features which interact in complicated, nonlinear ways, assembling a single global model can be very difficult, and hopelessly confusing when you do succeed. An alternative approach to nonlinear regression is to sub-divide, or partition, the space into smaller regions, where the interactions are more manageable. We then partition the sub-divisions again — this is called recursive partitioning — until finally we get to chunks of the space which are so tame that we can fit simple models to them. The global model thus has two parts: one is just the recursive partition, the other is a simple model for each cell of the partition. Prediction trees use the tree to represent the recursive partition. Each of the terminal nodes, or leaves, of the tree represents a cell of the partition, and has attached to it a simple model which applies in that cell only. A point x belongs to a leaf if x falls in the corresponding cell of the partition. To figure out which cell we are in, we start at the root node of the tree, and ask a sequence of questions about the features. The interior nodes are labeled with questions, and the edges or branches between them labeled by the answers.

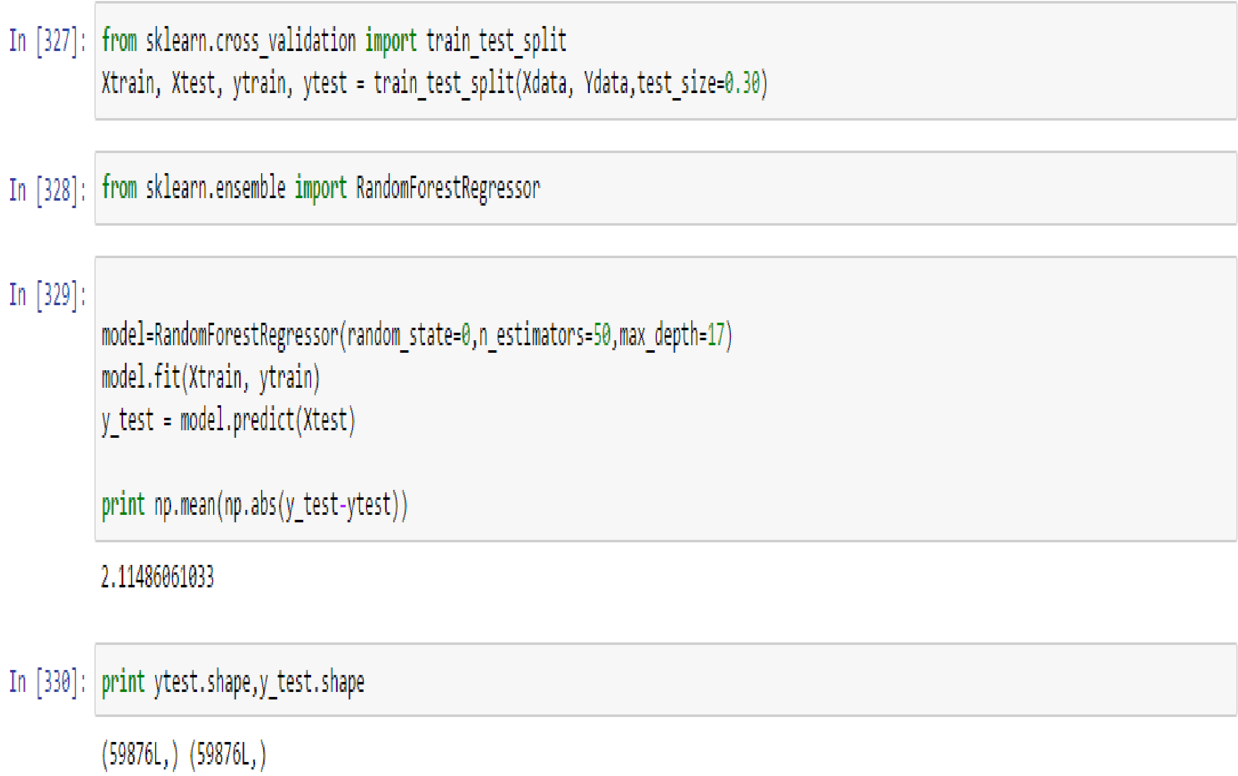
For classic regression trees, the model in each cell is just a constant estimate of Y. That is, suppose the points (xi,yi),(x2,y2),...(xc,yc) are all the samples belonging to the leaf-node l.

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. For instance , The deeper the tree, the more complex the decision rules and the fitter the model.



[36]

We Implement this model on Sickit Learn by these functions.



```
In [327]: from sklearn.cross_validation import train_test_split
          Xtrain, Xtest, ytrain, ytest = train_test_split(Xdata, Ydata,test_size=0.30)

In [328]: from sklearn.ensemble import RandomForestRegressor

In [329]:
          model=RandomForestRegressor(random_state=0,n_estimators=50,max_depth=17)
          model.fit(Xtrain, ytrain)
          y_test = model.predict(Xtest)

          print np.mean(np.abs(y_test-ytest))

          2.11486061033

In [330]: print ytest.shape,y_test.shape

          (59876L,) (59876L,)
```

[26]

3.3.3.1.3. Polynomial Regression Model

One common pattern within machine learning is to use linear models trained on nonlinear functions of the data. This approach maintains the generally fast performance of linear methods, while allowing them to fit a much wider range of data.

For example, a simple linear regression can be extended by constructing polynomial features from the coefficients. In the standard linear regression case, you might have a model that looks like this for two-dimensional data:

ˆ y(w,x) = w0 + w1x1 + w2x2

If we want to fit a paraboloid to the data instead of a plane, we can combine the features in second-order polynomials, so that the model looks like this:

ˆ y(w,x) = w0 + w1x1 + w2x2 + w3x1x2 + w4x2 1 + w5x2 2

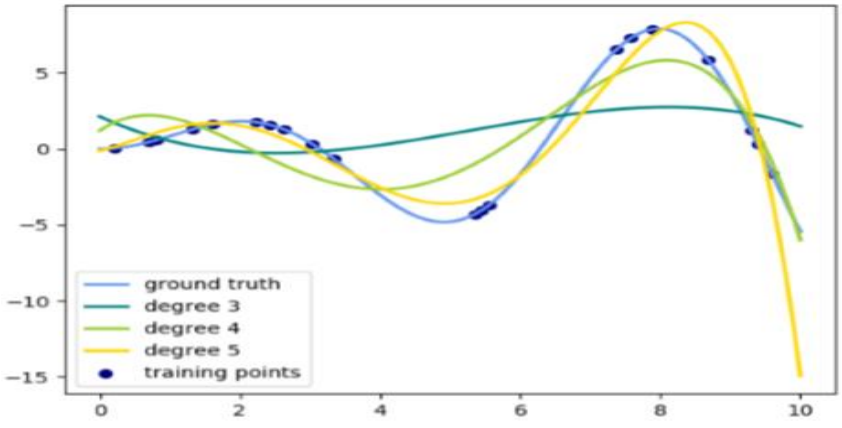
The (sometimes surprising) observation is that this is still a linear model: to see this, imagine creating a new variable

z = [x1,x2,x1x2,x2 1,x2 2]

With this re-labeling of the data, our problem can be written

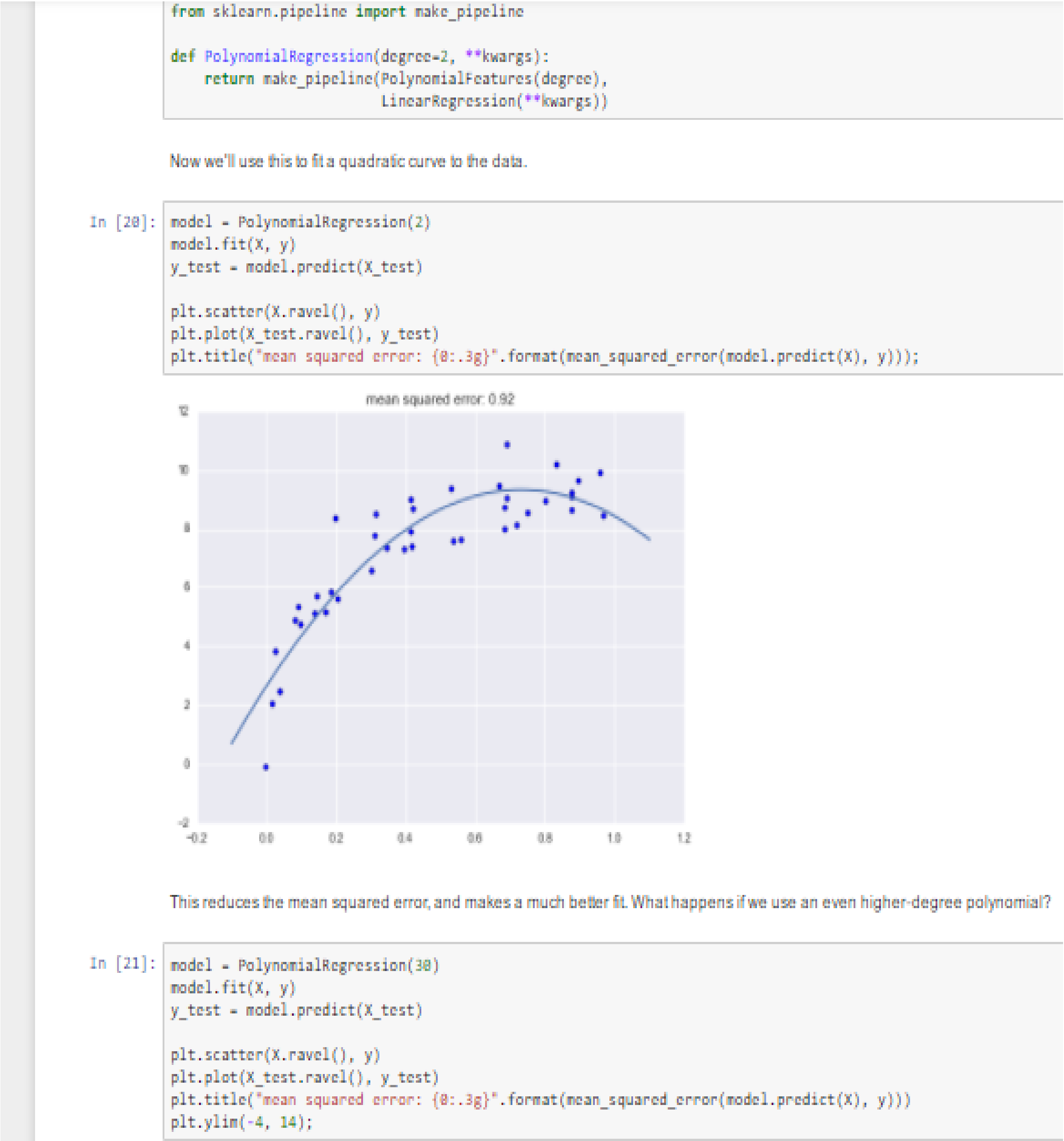
ˆ y(w,x) = w0 + w1z1 + w2z2 + w3z3 + w4z4 + w5z5

We see that the resulting polynomial regression is in the same class of linear models we'd considered above (i.e. the model is linear in w) and can be solved by the same techniques. By considering linear fits within a higher-dimensional space built with these basis functions, the model has the flexibility to fit a much broader range of data. Here is an example of applying this idea to one-dimensional data, using polynomial features of varying degrees:



[26]

We Implement this model on Sickit Learn by these functions.



```
from sklearn.pipeline import make_pipeline

def PolynomialRegression(degree=2, **kwargs):
    return make_pipeline(PolynomialFeatures(degree),
                         LinearRegression(**kwargs))
```

Now we'll use this to fit a quadratic curve to the data.

```
In [20]: model = PolynomialRegression(2)
         model.fit(X, y)
         y_test = model.predict(X_test)

         plt.scatter(X.ravel(), y)
         plt.plot(X_test.ravel(), y_test)
         plt.title("mean squared error: {0:.3g}".format(mean_squared_error(model.predict(X), y)));
```

This reduces the mean squared error, and makes a much better fit. What happens if we use an even higher-degree polynomial?

```
In [21]: model = PolynomialRegression(30)
         model.fit(X, y)
         y_test = model.predict(X_test)

         plt.scatter(X.ravel(), y)
         plt.plot(X_test.ravel(), y_test)
         plt.title("mean squared error: {0:.3g}".format(mean_squared_error(model.predict(X), y)))
         plt.ylim(-4, 14);
```

[26]

4. Results

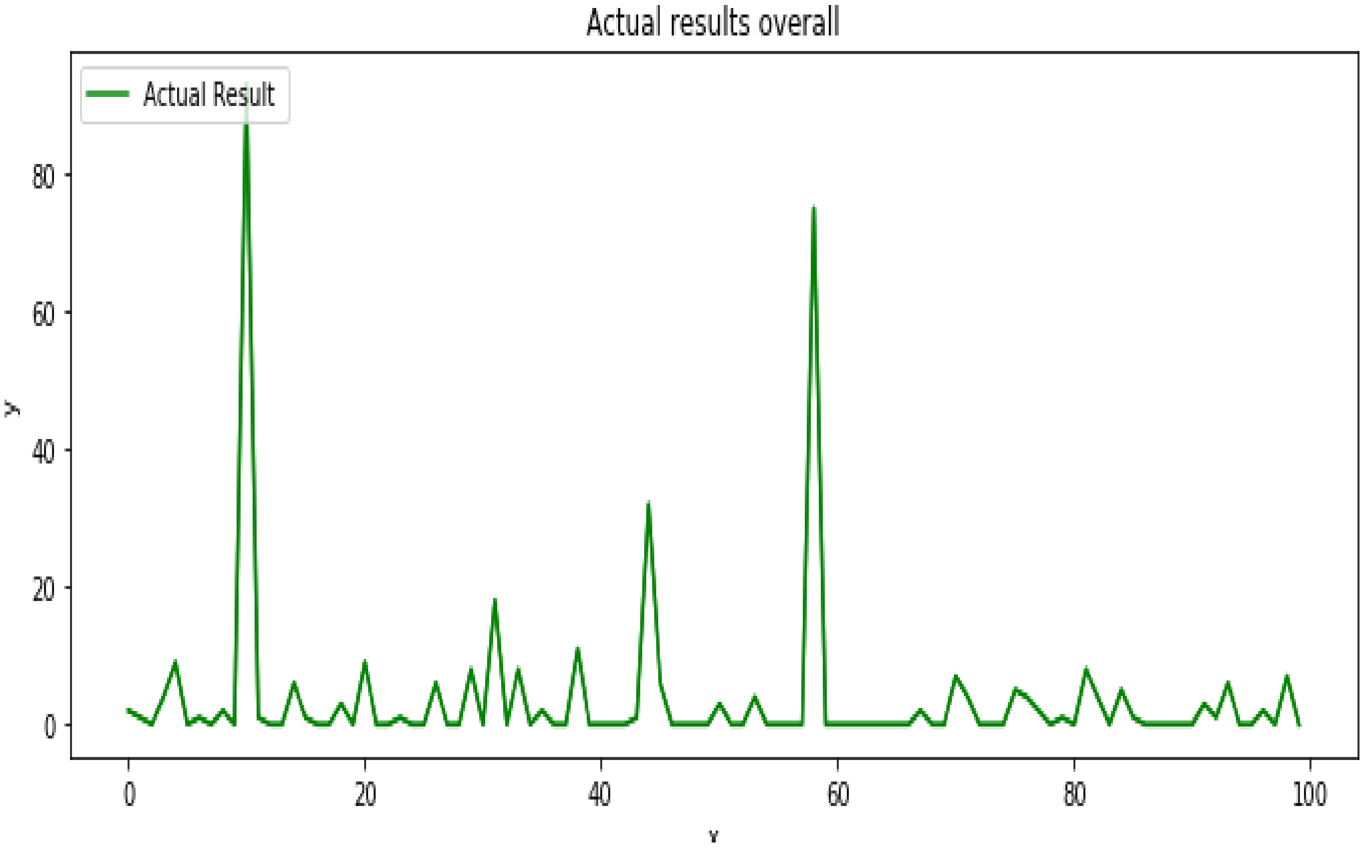
4.1. Error Rates

4.1.1. Error rates of Actual Data:

Actual data error rate is zero because it is ideal condition which is real time gap produced by the demand and supply.

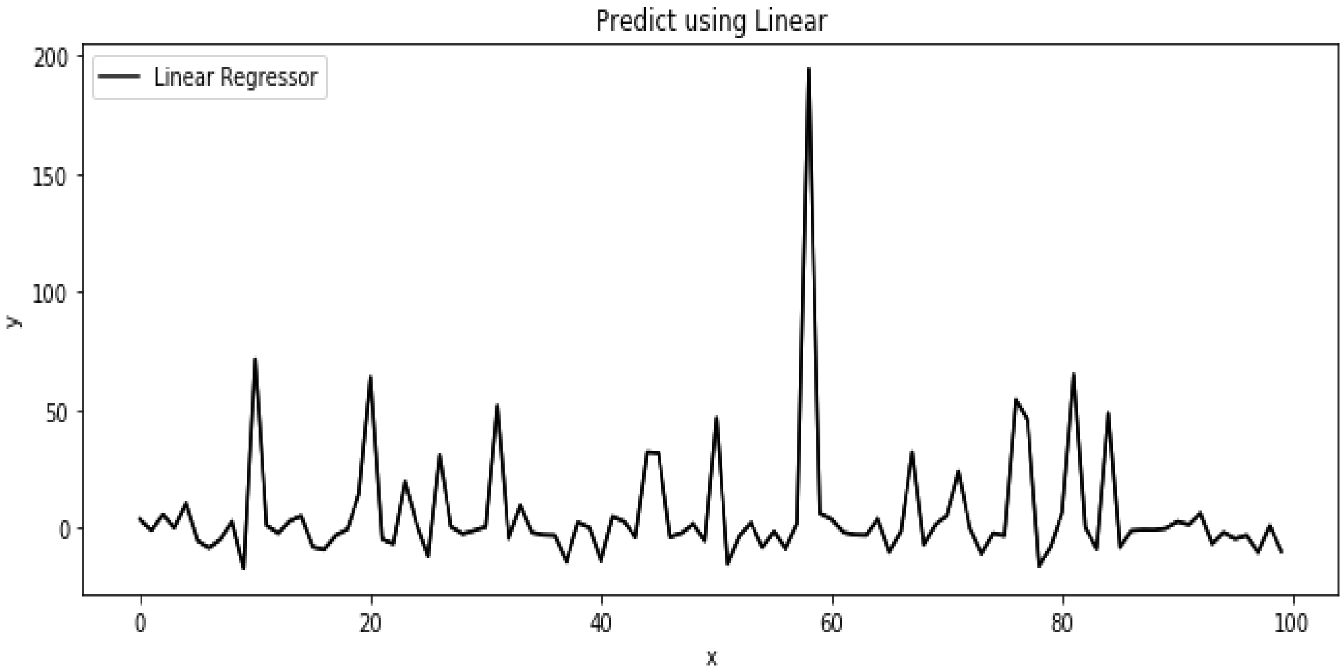
MAE=0

Graph:

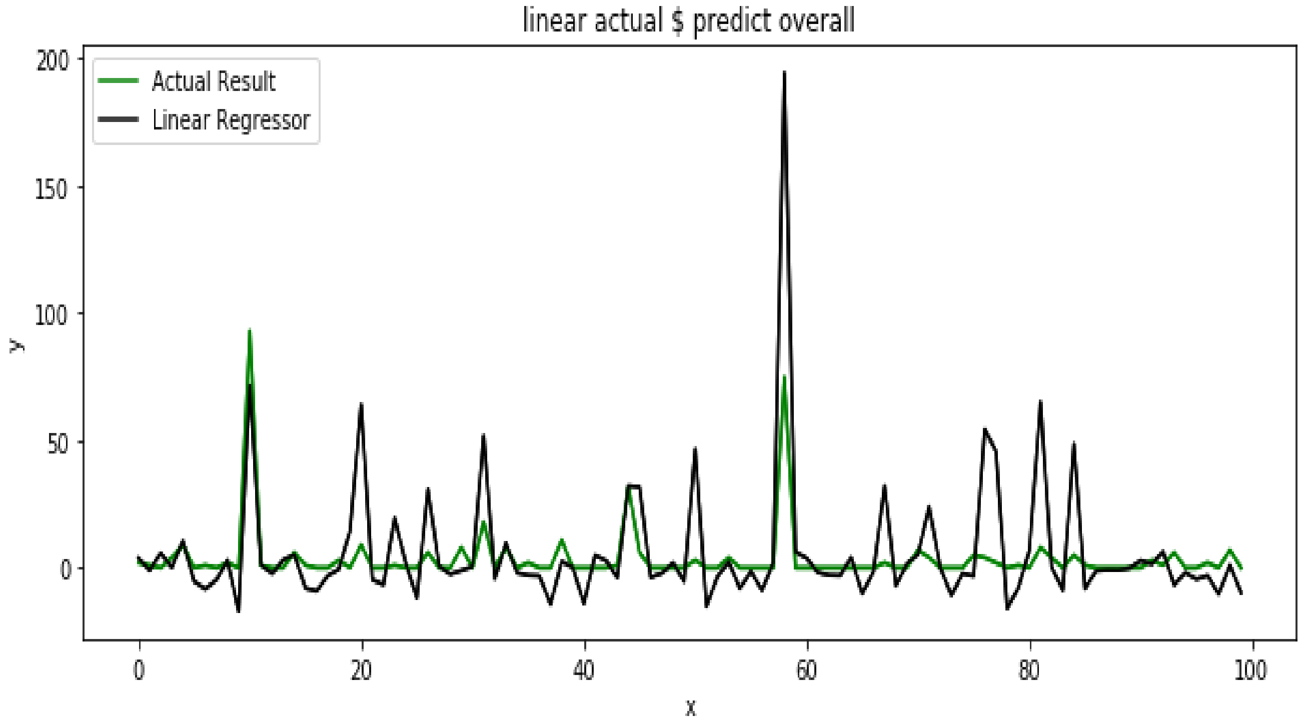


4.1.2. Error rates of Linear Regressor:

Error rates by apply the model of linear regressor is high as shown in the graph

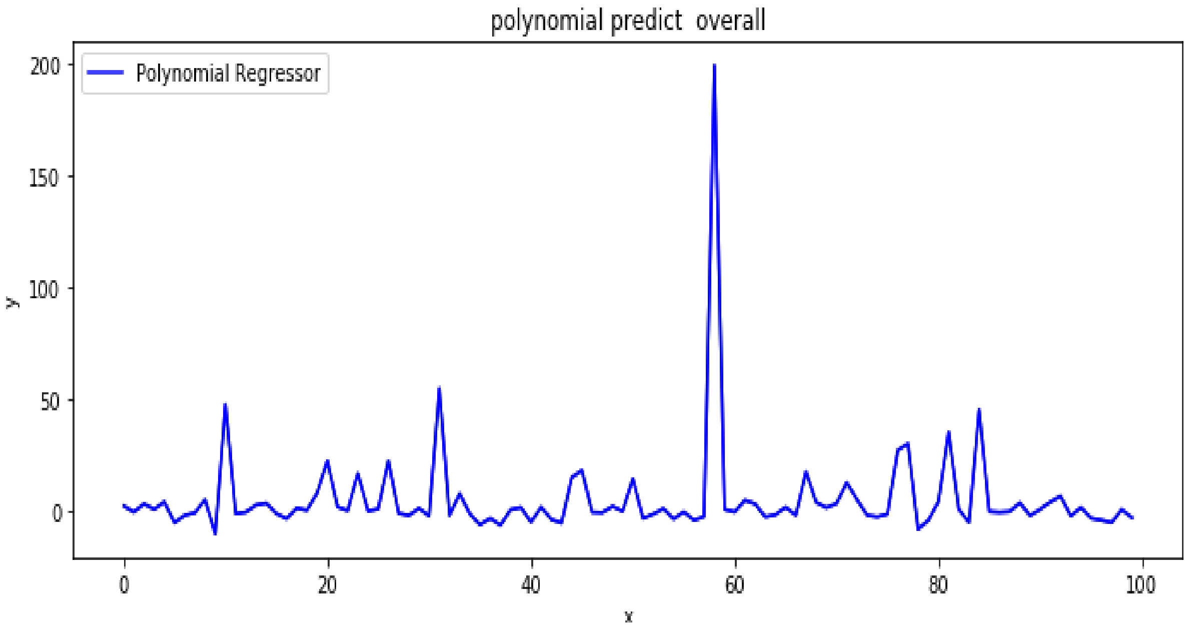


By comparing it with actual results it will look like it

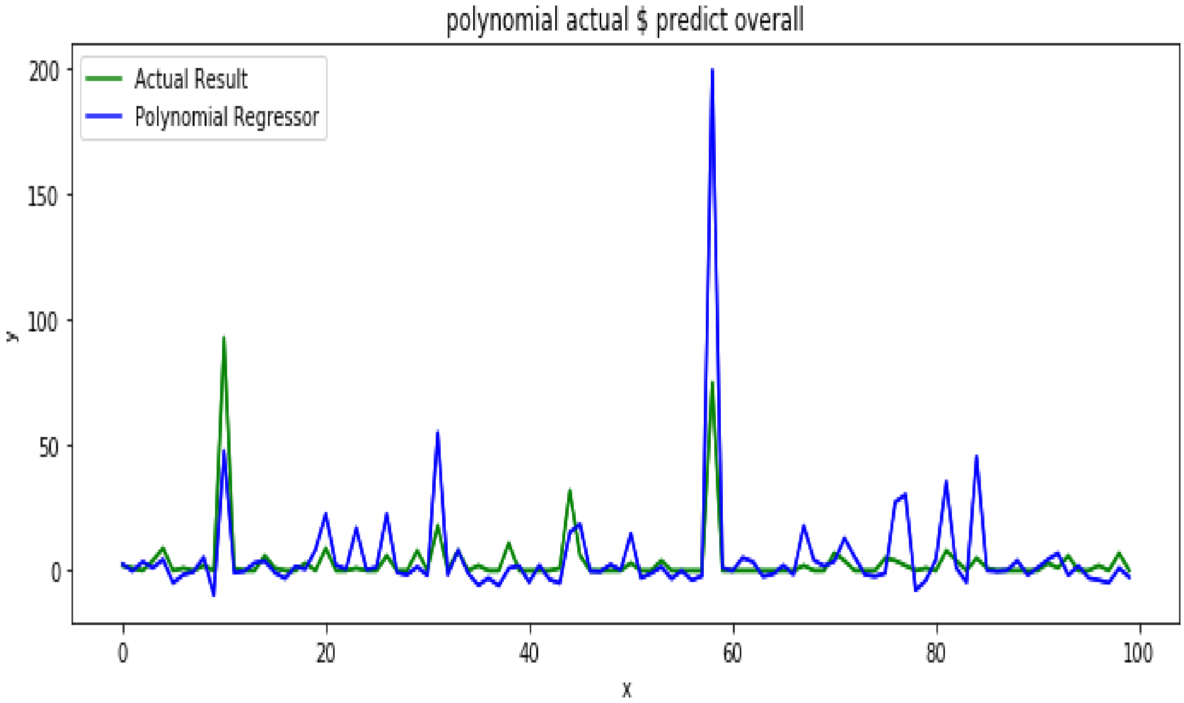


4.1.2. Error rates of polynomial Regressor:

Error rates by apply the model of polynomial regressor is low as compare to Linear regressor. It shows more accuracy than linear it is closer to Actual Results as shown in the graphs.

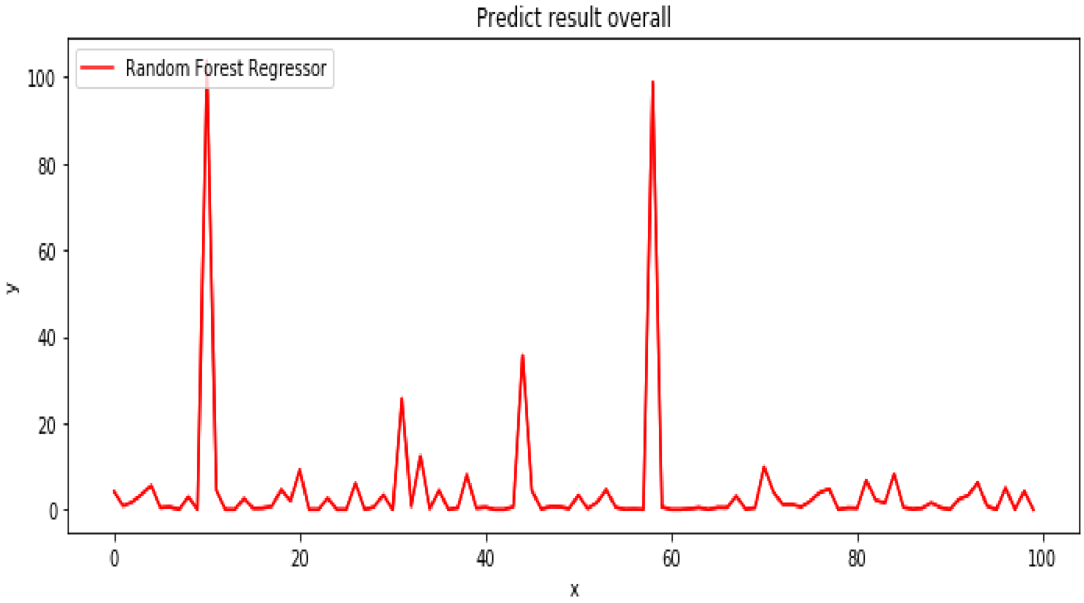


Now with actual results

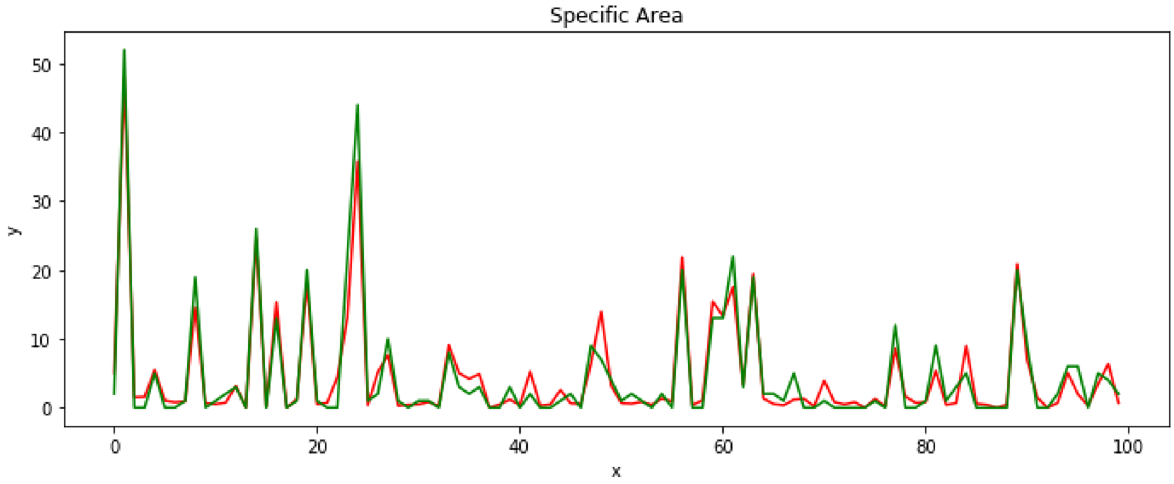


4.1.3. Error rates of Random Forest Regressor:

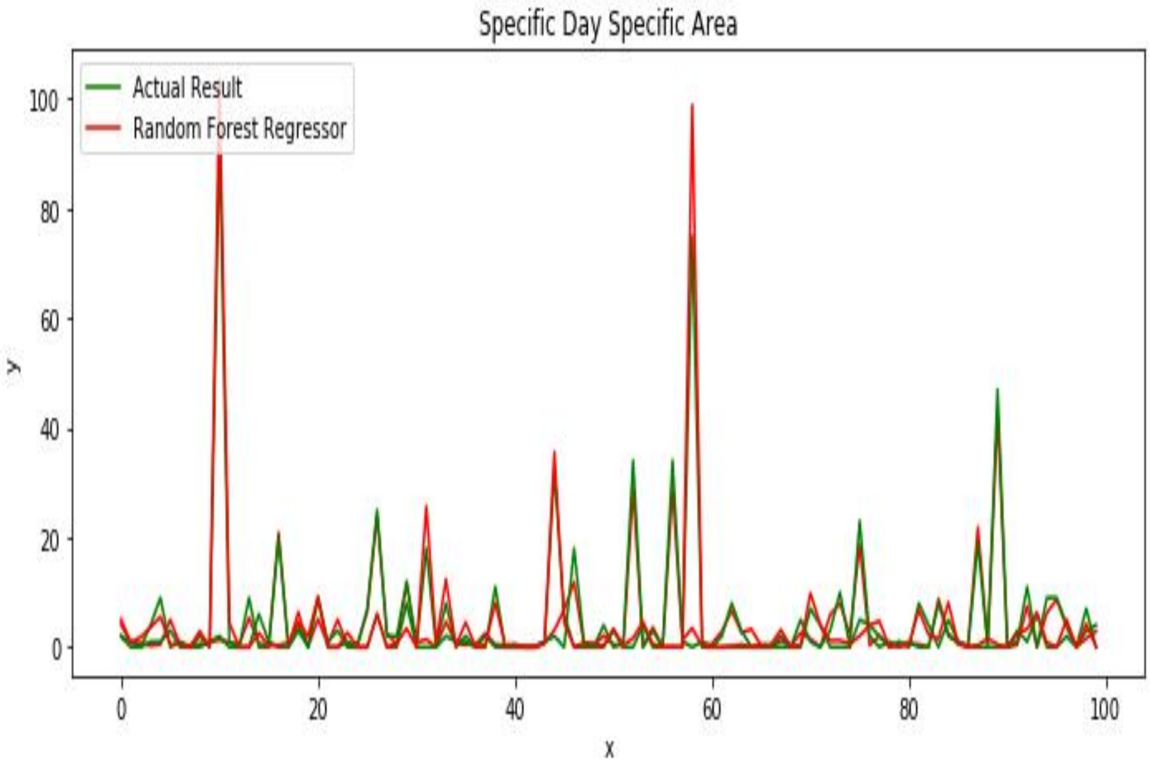
Error rates by apply the model of Random Forest regressor is low as compare to Linear regressor and polynomial regressor. It shows more accuracy than linear and polynomial it is closer to Actual Results as shown in the graphs.



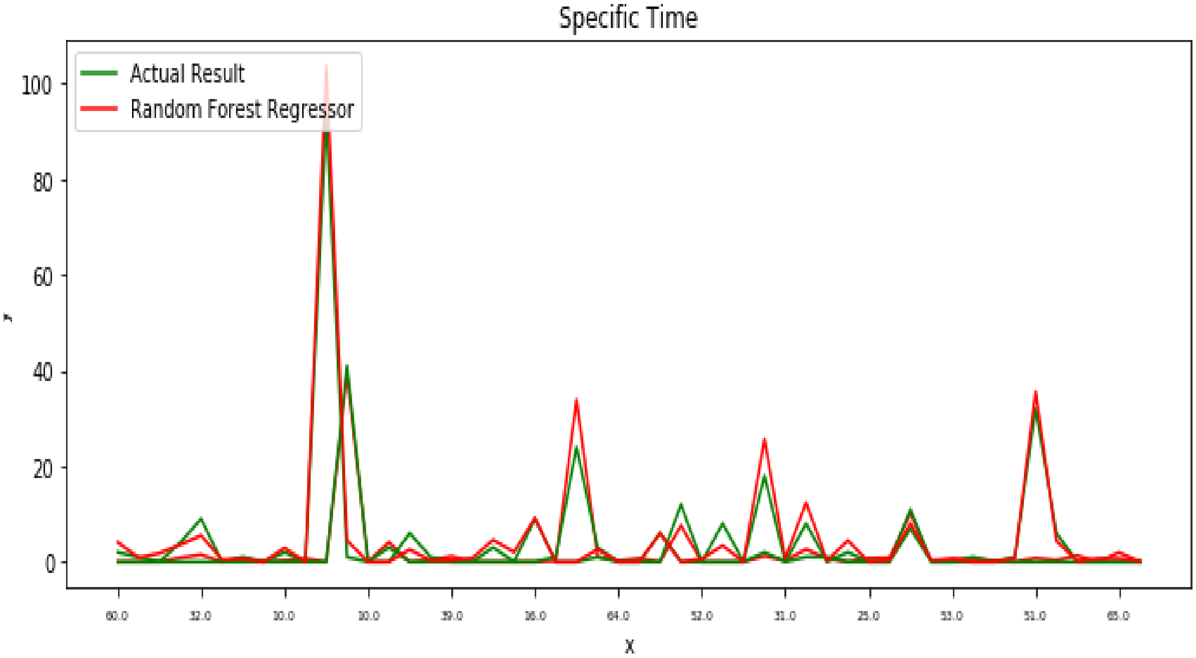
Compare the results of specific Area with Actual Results



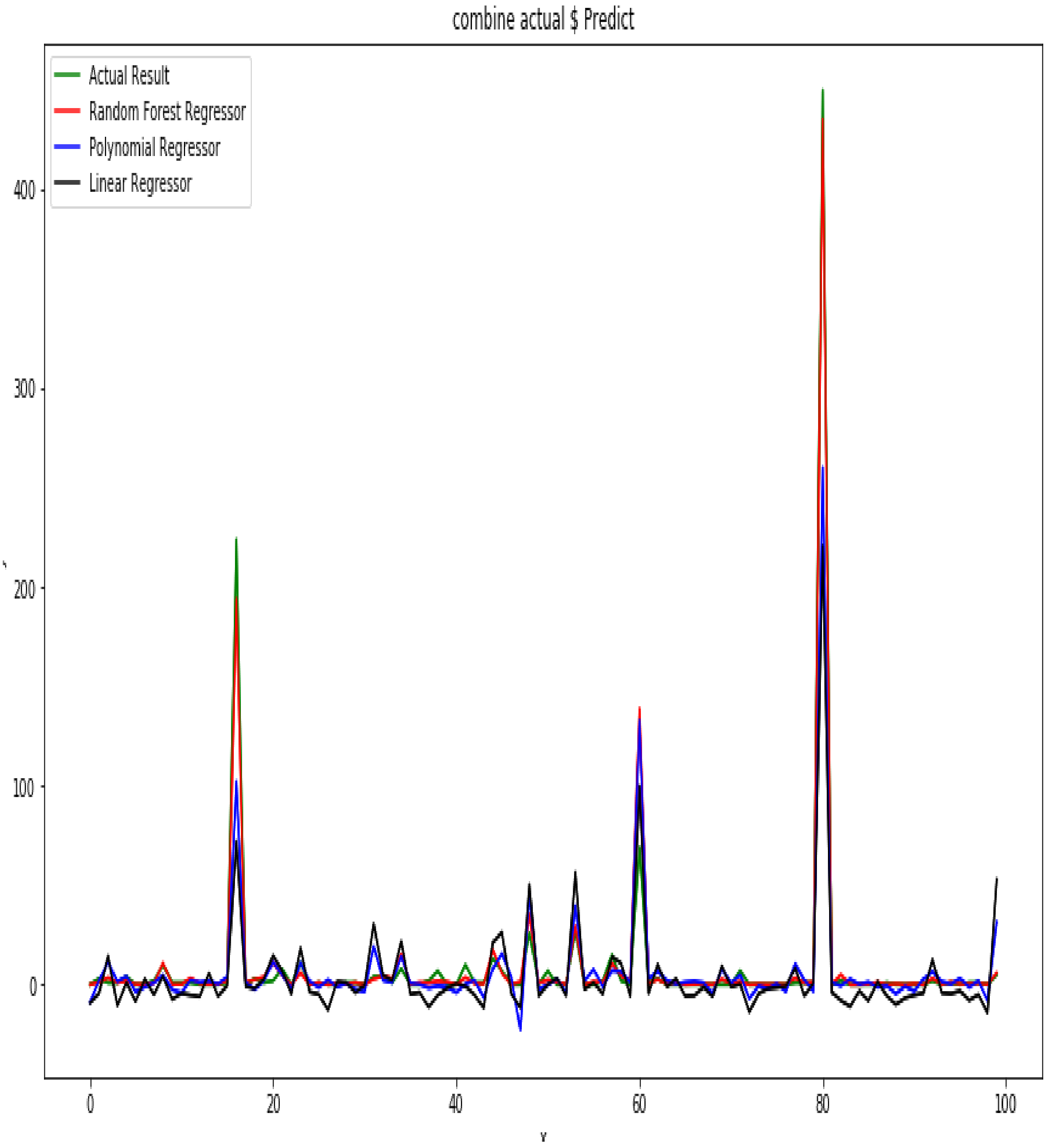
Compare the results of specific Area and specific day with Actual Results



Compare the results of specific time with Actual Results



4.1.4. Compression of all regressor with Actual results:



5. Reference

[1] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, “T-finder: A recommender system for finding passengers and vacant taxis,” IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 10, pp. 2390–2403, 2013.

[2] https://caow13.github.io/deepSD.pdf [accessed jan 2 2018].

DeepSD: Supply-Demand Prediction for Online Car-hailing Services using Deep Neural Networks

Dong Wang1,2 Wei Cao1,2 Jian Li1 Jieping Ye2 1 Institute for Interdisciplinary Information Sciences, Tsinghua University 2 Bigdata Research Lab, Didi Chuxing {cao-w13@mails, wang-dong12@mails, lijian83@mail}.tsinghua.edu.cn yejieping@didichuxing.com

[3] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” Computer Science, 2015.

[4] http://scikit-learn.org/stable/tutorial/index.html [accessed jan 3 2018].

[5] Uber, 2015. www.uber.com. [accessed May 05 2018].

[6] Nicholas Diakopoulos. How uber surge pricing really works, April. http://www.washingtonpost.com/blogs/wonkblog/wp/2015/04/17/ how-uber-surge-pricing-really-works/.

[7] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. T-drive: driving directions based on taxi trajectories. In Proceedings of the 18th SIGSPATIAL International conference on advances in geographic information systems, pages 99– 108. ACM, 2010.

[8] IJCSMC, Vol. 5, Issue. 8, August 2016, pg.207 – 215 Predictive Modeling: Data Mining Regression Technique Applied in a Prototype 1Festim Halili, 2Avni Rustemi 1,2Department of Informatics State University of Tetovo, SUT Tetovo, Macedonia

[9] https://dzone.com/articles/3-machine-learning-algorithms-you-need-to-know[accessed Jun 10 2018].

[10] https://www.researchgate.net/publication/235678507_Using_Data_Mining_with_Time_Series_Data_in_Short-Term_Stocks_Prediction_A_Literature_Review [accessed Jun 10 2018].

[11] T. O. Hill, M. Connor and W. Remus, “Neural Network Models for Time Series Forecasts,” Management Science, Vol. 42, No. 7, 1996, pp. 1082-1092.

[12] S. Fong and Z. Nannan, "Towards an Adaptive Fore- casting of Earthquake Time Series from Decomposable and Salient Characteritics," Proceendings of the 3rd In-ternational Conference on Pervasive Patterns and Appli-cations, Rome, 25 September 2011, pp. 53-60.

[13] K. J. Walsh, M. Milligan, M. Woodman and J. Sherwell, "Data Mining to Characterize Ozone Behavior in Bal- timore and Washington DC," Journal of Atmospheric En-vironment, Vol. 42, No. 18, 2008, pp. 4280-4292. doi:10.1016/j.atmosenv.2008.01.012

[14] C. Damle and A. Yalcin, "Flood Prediction Using Time Series Data Mining," Journal of Hidrology, Vol. 333, No. 2-4, 2007, pp. 305-316. doi:10.1016/j.jhydrol.2006.09.001

[15] E. Tsang, P. Yung and J. Li, "EDDIE-Automation, a De-cision Support Tool for Financial Forecasting," Decision Support Systems, Vol. 37, No. 4, 2004, pp. 559-565. doi:10.1016/S0167-9236(03)00087-3

[16] A. Pasley and J. Austin, "Distribution Forecasting of High Frequency Time Series," Decision Support Systems, Vol. 37, No. 4, 2004, pp. 501-513. doi:10.1016/S0167-9236(03)00083-6

[17] Z. Huang, H. Chen, C. J. Hsu, W. H. Chen and S. Wu, "Credit Ratings Analysis with Support Vector Machines and Neural Networks. A Market Comparative Study," Decision Support Systems, Vol. 37, No. 4, 2004, pp. 542-558. doi:10.1016/S0167-9236(03)00086-1

[18] H. M. Krolzig and J. Toro, "Multiperiod Forecasting in Stock Market: A Paradox Solved," Decision Support Sys-tems, Vol. 37, No. 4, 2004, pp. 531-542. doi:10.1016/S0167-9236(03)00085-X

[19] K. C. Chiu and L. Xu, "Arbitrage Pricing Theory-Based Gaussian Temporal Factor Analysis for Adaptive Port- folio Management," Decision Support Systems, Vol. 37, No. 4, 2004, pp. 485-500.

[20] API design for machine learning software: experiences from the scikit-learn project

Lars Buitinck1, Gilles Louppe2, Mathieu Blondel3, Fabian Pedregosa4, Andreas C. Mu¨ller5, Olivier Grisel6, Vlad Niculae7, Peter Prettenhofer8, Alexandre Gramfort4,9, Jaques Grobler4, Robert Layton10, Jake Vanderplas11, Arnaud Joly2, Brian Holt12, and Ga¨el Varoquaux4

1 ILPS, Informatics Institute, University of Amsterdam 2 University of Li`ege 3 Kobe University 4 Parietal, INRIA Saclay 5 University of Bonn 6 Independent consultant 7 University of Bucharest 8 Ciuvo GmbH 9 Institut Mines-Telecom, Telecom ParisTech, CNRS LTCI 10 University of Ballarat 11 University of Washington 12 Samsung Electronics Research Institute

[21] http://dspace.unimap.edu.my/dspace/bitstream/123456789/3057/4/Methodology.pdf

[03-05-2018][07:00 PM]

[22] A MACHINE LEARNING APPROACH TO POLYNOMIAL REGRESSION Aleksandar Peˇcko

[23] Data Mining: A prediction Technique for the workers in the PR Department of Orissa (Block and Panchayat) Neelamadhab Padhy1 and Rasmita Panigrahi2

1 Assistant. Professor, Gandhi Institute of Engineering and Technology, GIET, Gunupur nmp.phdcmj2010@gmail.com, neela.mbamtech@gmail.com Research Scholar, Department of Computer Science CMJ University, Meghalaya (Shilong) 2 Lecturer, Gandhi Institute of Engineering and Technology, GIET, Gunupur rasmi.mcamtech@gmail.com

[24] CS229 Lecture notes Andrew Ng

[25] Machine Learning 1. Linear Regression Lars Schmidt-Thieme Information Systems and Machine Learning Lab (ISMLL) Institute for Business Economics and Information Systems & Institute for Computer Science University of Hildesheim

[26] scikit-learn user guide Release 0.20.dev0 scikit-learn developers

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[28] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," arXiv preprint arXiv:1603.02754, 2016.

[29] R. Wang, C.-Y. Chow, Y. Lyu, V. Lee, S. Kwong, Y. Li, and J. Zeng, "Taxirec: recommending road clusters to taxi drivers using ranking-based extreme learning machines," in Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM, 2015, p. 53.

[30] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. Pazzani, "An energy-efficient mobile recommender system," in Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2010, pp. 899–908.

[31] A. Grover, A. Kapoor, and E. Horvitz, "A deep hybrid model for weather forecasting," in Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2015, pp. 379–386.

[32] S. Wu, W. Ren, C. Yu, G. Chen, D. Zhang, and J. Zhu, "Personal recommendation using deep recurrent neural networks in netease," in 32nd IEEE International Conference on Data Engineering, ICDE 2016, Helsinki, Finland, May 16-20, 2016, 2016, pp. 1218–1229.

[33] X. Ding, Y. Zhang, T. Liu, and J. Duan, “Deep learning for eventdriven stock prediction,” in Proceedings of the 24th International Joint Conference on Artificial Intelligence (ICJAI15), 2015,

pp. 2327–2333.

[34] E. Mocanu, P. H. Nguyen, M. Gibescu, E. M. Larsen, and P. Pinson, “Demand forecasting at low aggregation levels using factored conditional restricted boltzmann machine,” in 2016 Power Systems Computation Conference (PSCC), June 2016, pp. 1–7.

[35] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, “Traffic flow prediction with big data: a deep learning approach,” IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 2, pp. 865–873, 2015. [36] Z. Junbo, Z. Yu, Q. Dekang, L. Ruiyuan, and Y. Xiuwen, “Dnn-based prediction model for spatio-temporal data.” ACM SIGSPATIAL 2016, October 2016.

[36] https://www.youtube.com/watch?v=LIPtRVDmj1M [05-06-2018][4:35 PM]