Star Prediction Based on Yelp Business Data And Application in Physics

Yuyi Tan
PID: A53220110
Electrical and Computer Engineering
University of California, San Diego
La Jolla, California 92093

Zeshi Du
PID: A53220026
Electrical and Computer Engineering
University of California, San Diego
La Jolla, California 92093

Zhiyu Hou
PID: A53209630
Electrical and Computer Engineering
University of California, San Diego
La Jolla, California 92093

Cheng Qian
PID: A53209561
Electrical and Computer Engineering
University of California, San Diego
La Jolla, California 92093

Wenjun Zhang
PID: A53218995
Electrical and Computer Engineering
University of California, San Diego
La Jolla, California 92093

***Abstract*—Rating prediction is one of the most important application of machine learning. In this project, we are looking for appropriate machine learning algorithms to predict the star a user will assign to a business, and investigate the application of these algorithms in physics. The four different models we implement are Linear Regression and Logistic Regression, which extract features from the data to train regression models for prediction, Latent Factor Model(LFM), which cares about the features of users and businesses themselves to train parameters as weights to make prediction, and Feature Vectors From Text method, which utilizes the text feature from the business reviews dataset, such as unigrams and bigrams to train a weighted linear regression. The accuracy of these models were graded by use of the mean squared error(MSE), and the comparison were made between these four methods. Besides rating prediction, these machine learning methods can be widely applied in physics, such as precipitation predictionm components/material usage lifetime prediction and text mining for weather prediction.**

***Keywords*—linear regression, logistic regression, latent factor, text mining, applications in physics.**

I. Introduction

A. Identify Dataset

We identify a business dataset from Kaggle which were Yelp business reviews along with some data about the users and business. The data contains 11,537 businesses, 8,282 check-in sites, 43,873 users, and 229,907 reviews from the Phoenix, AZ metropolitan area. In the reviews data, it contains the business id, user id, stars, review text, date, etc. The star varies from 1 to 5, and its distribution is shown as Figure.1, where the numbers of star from 1 to 5 are 17516, 20957, 35363, 79878 and 76193. So, from the whole dataset, we can see that people prefer to give high star (star above or equal to 4).

The year of the date varies from 2005 to 2013, and the plot of average star in each year is shown as in Figure.2. We can see that the average star basically stays stable in each year.

The number review words vary from 0 to 1006. We take a bin width of 150, the average stars in each bin are shown as in Figure.3. We can see that the average star tends to decrease with the increasing number of review words.

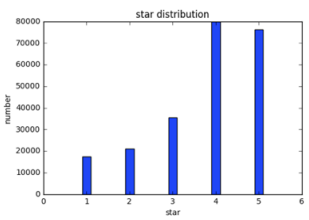

Fig. 1. Star Distribution

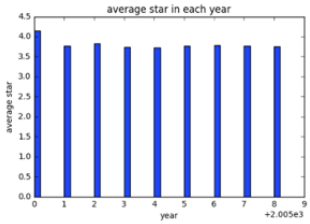

Fig. 2. Average Star in Each Year

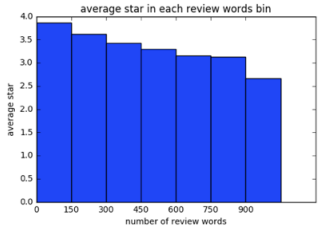

Fig. 3. Average Star in Each Review Words Number Bin

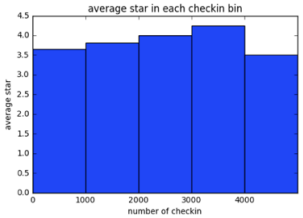

Fig. 4. Average Star in Each Check-in Bin

Then we take a look at the check-in data. The check-in data contains the business id and the check-in info, which are numbers of check-ins from during each hour on all Mondays to all Sundays. The total number of check-in varies from 3 to 22,977. However, there is only one massive check-in 22,977, most of the number of check-in are below 5000. So, we define 5 bins of check-in, which are 0 to 1000, 1001 to 2000, 2001 to 3000, 3001 to 4000, and above 4000. Their average stars are shown as in Figure.4. We can see that when the number of check-in is below 4000, the average star tend to increase with a larger check-in number. And when check-ins are larger than 4000, there are only 2 instances in the dataset, so it may be misleading.

In the user data, it contains the user id, the first name of the user, the review count that the user gave, and the average star of the user, etc. The average star of the user varies from 0 to 5 star. The review count varies from 1 to 5807. We take a bin width of 1000, the average stars in each bin are shown as in Figure.5. The blank from 3000 to 5000 is that there is no data in this range. When user review count is lower than 3000, the average star tend to decrease with a larger number of user review count. And similarly, the instances of review count above 3000 are limited in the dataset, so it may be misleading.

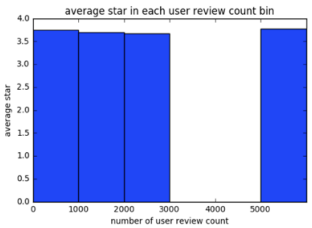

Fig. 5. Average Star in Each User Review Count Bin

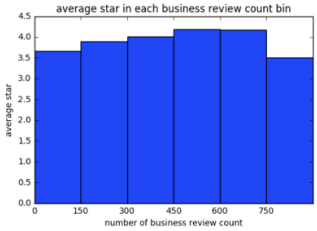

Fig. 6. Average Star in Each Business Review Count Bin

In the business data, it contains the business id, the name of the business, its location information, the stars, the review count it received, etc. Since the business all located in Phoenix, AZ metropolitan area, their city and state properties are the same, and their latitude and longitude are very close. The star of the business varies from 1 to 5 star. The review count varies from 3 to 862. We take a bin width of 150, the average stars in each bin are shown as in Figure.6. When business review count is lower than 600, the average star tends to increase with a larger number of business review count. For large number of business review count, the instances are limited in the dataset, so the statistic may not be strong enough.

From the exploratory analysis above, we can see that the star a user gives a to certain business can be influenced by many factors, such as the number of words of the review, the total number of check-in of the business, the review count the user gave, the review count the business received, or the average stars of the user and the business.

B. Identify Predictive Task

From what have been discussed in 1.A, we already know that the star a user gave to a business can be influenced by many factors. So, we identify a prediction task to predict the star on this dataset.

The review dataset contains 229,907 reviews. We divide the dataset and define the first 100,000 reviews to be the training data, the 100,001 to 200,000 reviews to be the validation data, and the rest 29,907 reviews to be the test data. And since the review data contains the user id and business id, we can use these 2 IDs to obtain some other features like review counts and check-in number from the user dataset, business dataset and check-in dataset.

From the materials in 1.A, we can see that the average star basically stays stable in each year. So, year may not be a very relevant feature in this prediction task. Then, the following features seem to have linear relationships with the stars:

Number of check-in: increases followed by star increasing

User review count: increases followed by star decreasing

Business review count: increases followed by star increasing

Number of review words: increases followed by star decreasing

So, we may assume that the following theories: If a business has more customers check-in, it may mean that its reasonably good and people tend to give higher star.

If a user likes to give reviews, maybe he/she is pickier and tends to give lower star.

If a business received more reviews, maybe its good and people like to commend and give higher star.

If a review contains more words, maybe its because the writer doesnt enjoy the experience and wants write more to accuse it and tend to give lower star.

In addition, some other features like average stars of the users and the business also seem reasonable to influence the star a user will give to a business.

Therefore, we can implement a star prediction task based on different features in this dataset.To do this, many machine learning methods are available, such as Linear Regression, Logistic Regression, Latent-factor model and Bags-of-words.

To assess the validity of the models predictions, we use the measure of Mean Squared Error (MSE) to evaluate the performance of the model.

II. Framework

A. Baseline

A simple model relevant baseline for the star prediction is to just use the average star to predict all the star that a user will give to a business. The trivial predictor is as below:

$$star(user, business) = \alpha \tag{1}$$

B. Linear Regression

From the exploratory analysis, we see that many features seem to have linear relationships with the star. So, we can implement a prediction model based on linear regression.

There were plenty of features that can be reflected by number in the dataset. Among all the features we felt the votes based on three choices, useful, funny and cool would help predict stars. Some other features like number of reviews for a particular user or for a particular user, open information of businesses etc. were also considered to be utilized in this model.

To optimize the linear prediction model, we started with basic features in review. The basic linear regression based on three values of votes had a Mean Square Error of 1.411 on train set and 1.439 on valid set. Afterwards we introduced more features gradually and kept checking MSEs values. When we added one of all features including open, review count, average star for a business, average star for a user, votes, longitude and latitude each time, the MSE decreased continuously and it dropped to 1.069 on test set.

To avoid overfitting issue we introduced complexity and set to various values to execute regularization process. The equation was shown below

$$\arg\min_{\theta} = \frac{1}{N}||y - X\theta||_2^2 + \lambda||\theta||_2^2 \tag{2}$$

When we selected larger than 1, we noticed that the MSE on validation set declined and MSE on test set raised. While we set less than 1 we noticed that the MSE on validation set increased and MSE on test set dropped. Validation set was supposed to help optimize parameters for the model thus we set to 1 and obtained the prediction vector .

There were two features that were not much useful during the optimization of the model: longitude and latitude. Because all the reviews were taken from businesses in a certain area which was Phoenix, AZ metropolitan area. This limited area led to very similar values of longitude and latitude and prediction based on those two features did not work well.

C. Logistical Regression

Also, since the star a user give varies from 1 to 5, we can define: when star is greater than 3.5, we consider it to be good, and when star is equal or lower than 3.5, we consider it to be not good. Then we can design a model based on Logistic Regression for this task, which is to predict the star based on the confidence of the star being good.

We define the labels y_i to be:

$$y_i = \begin{cases} 1, & if \quad star \geq 3.5 \\ 0, & if \quad star < 3.5 \end{cases} \tag{3}$$

For the features of the model, we choose four features in the whole dataset, and the feature vector will be:

X_i=[1, checkin, review words, user's average star, business's average star]

A sigmoid function is defined as below:

$$\sigma(t) = \frac{1}{1 + e^{-t}} \tag{4}$$

Using the method in Logistic Regression, $X_i \cdot \theta$should be maximized when y_i is positive and minimized when y_i is negative. Also, to avoid overfitting, we introduce in the model and set it to 1. Combining the regularizer, we define:

$$l_\lambda(y|x) = \sum_i -log(1 + e^{X_i\theta}) + \sum_{y_i=0} -X_i\theta - \lambda||\theta||_2^2 \tag{5}$$

Then the derivative will be:

$$\frac{dl}{d\theta_k} = \sum_i X_{ik}(1 - \sigma(X_i\theta)) + \sum_{y_i=0} -X_{ik} - \lambda 2\theta_k \tag{6}$$

Using the formula above by setting $\lambda = 1$, we solve using gradient ascent until θ finally converge.

Extracting the validation or test feature vectors X, then we can use the equation below to obtain the $y_prediction$ data.

$$y_{prediction} = X\theta \tag{7}$$

$y_{prediction}$ is the confidence we get to say if the star is good. We first find the maximum and minimum confidence in the $y_{prediction}$ data. And call them y_{max} and y_{min}. Clearly $y_{max} > 0$ and $y_{min} < 0$. Next, since the star varies from 1 to 5, we can predict the star using the formula below:

$$star = \begin{cases} 3.5 + 1.5 * \frac{y_{prediction}}{y_{max}}, & if \quad y_{prediction} \geq 0 \\ 3.5 - 2.5 * \frac{y_{prediction}}{y_{min}}, & if \quad y_{prediction} < 0 \end{cases} \tag{8}$$

In the user data, some user profiles are omitted from the data because they have elected not to have public profiles. Their reviews may still be in the data set if they are still visible on Yelp. And in the check-in data, if there are no check-ins for a business, the entire record will be omitted. In these cases, we will just use the average star of all users or the average check-ins of all business to represent the features.

In the attempt of optimizing the model, there are two features that we thought will be useful but actually are irrelevant, the number of reviews for a particular user or for a particular user. By adding these two features the MSE will actually increase in both the validation set and test set.

D. Latent Factor Model

In this data set, only caring about the features of the users or items themselves cannot give an accurate model. Because the evaluation and classification standards between individuals are different. For example, peoples evaluations towards one business Taco Bell may differs. Some people treat it as a fast food chain and thought it junk food, while others treat it as a traditional Mexican restaurant and found it healthy. Apparently, we can't rely on a single person's subjective idea to build a classification standard to the entire platform user preferences. Latent factor model, which only care about the interaction between user and item, can well model the potential preferences and outperform on the problem of rating predictions.

Partition data into training set, validation set and test set. As we have 229,907 records of data, the most situation we want is to train the whole data. However, in that case, we lost the validation set and cannot decide the optimum value of . Basically, we use the first 100,000 records for training and then 100,000 for validation, leaving the rest for test.

Set initial value to deal with cold start. By statistics, we found there are 5359 records in test set cannot find their corresponding bias. These are the new users or items that have never appeared in our training set, which is the cold start situation. So, we set the median of two bias as the default value of themselves. This operation slightly mitigates the cold start problem but does not make big improvement of model performance.

The basic implementation of LFM is as follow:

$$rating(user, item) \cong \alpha + \beta_{user} + \beta_{item} \tag{9}$$

β_{user} and β_{item} are two vectors that respectively stored how much this user tend to rate things above the mean and how much this item tends to receive higher rating than others. All values are set to be default zero before iterative step.

In real implementation, we set up two 2-D dictionary to store the rating in different structure to make it convenient for checking the ratings in the iterative steps.

After adding the regularizer, the loss function is computed as:

$$\arg\min_{\alpha,\beta}(\sum_{u,i}(\alpha+\beta_u+\beta_i-R_{u,i})^2+\lambda[\sum_u(\beta_u)^2+\sum_i(\beta_i)^2]) \tag{10}$$

Then we need to optimize the jointly convex by iteratively removing the mean and solving for beta. Set the λ =1. And use the first half data for training.

Iterative procedure – repeat the following updates until convergence:

$$\alpha = \frac{\sum_{u,i\in\text{train}}(R_{u,i}-(\beta_u+\beta_i))}{N_{\text{train}}}$$

$$\beta_u = \frac{\sum_{i\in I_u} R_{u,i}-(\alpha+\beta_i)}{\lambda+|I_u|}$$

$$\beta_i = \frac{\sum_{u\in U_i} R_{u,i}-(\alpha+\beta_u)}{\lambda+|U_i|}$$

Fig. 7. Iterative Procedure

By adding the user preference vector and item property vector, we can get a more specific model.

R_{ui}	business1	business2	business3	business4
user1	R11	R12	R13	R14
user2	R21	R22	R23	R24
user3	R31	R32	R33	R34

γ_u	class1	class2	class3
user1	P11	P12	P13
user2	P21	P22	P23
user3	P31	P32	P33

×

γ_i	b1	b2	b3	b4
class1	Q11	Q12	Q13	Q14
class2	Q21	Q22	Q23	Q24
class3	Q31	Q32	Q33	Q34

Fig. 8. Matrix Multiplication of LFM

$$R_{ui} = \sum_{k=1}^{K}\gamma_{uk}\cdot\gamma_{uk} = \gamma_u\cdot\gamma_i \tag{11}$$

$$f(u,i) \cong \alpha+\beta_u+\beta_i+\gamma_u\cdot\gamma_i \tag{12}$$

The two more iterative equations are:

$$\gamma_{uk} = \gamma_{uk}+\alpha[(R_{u,i}-\sum_{k=1}^{K}\gamma_{uk}\cdot\gamma_{ik})\gamma_{ik}-\lambda\gamma_{uk}] \tag{13}$$

and

$$\gamma_{ik} = \gamma_{ik}+\alpha[(R_{u,i}-\sum_{k=1}^{K}\gamma_{uk}\cdot\gamma_{ik})\gamma_{uk}-\lambda\gamma_{ki}] \tag{14}$$

Iteratively repeat the above steps until the difference of between last time and current time is less than 1e-10. After iteration, calculate the MSE on validation set. Next, we alternate the value of λ from 1 to 20 with step size of 0.1, checking the MSE on validation set to decide the λ that have the lowest MSE.

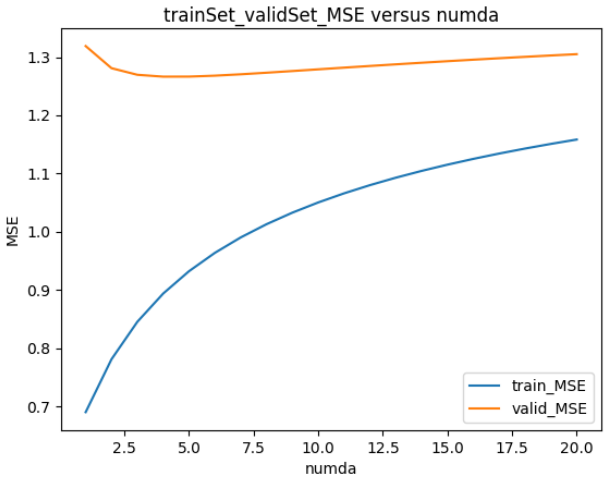

Fig. 9. MSE versus lumda

Finally, we get the best lambda is around 4.4. Then we round the final output to fit with actual number of stars. Some of the final output go beyond 5 which is apparently contradict with the real 5-star limit. So we set the range for final output with upper bound of 5 and lower bound of 1. The mean square error on test set is 1.28357.

E. Feature vectors from text

The model for feature vectors from text is using 1000 unigrams and 1000 bigrams as features to train a weighted linear regression as below.

$$equationstar = \alpha+\sum_{(w\in text)}tfidf(t,d,D)\theta w \tag{15}$$

Those 1000 unigrams and 1000 bigrams are carefully selected by most common unigrams and bigrams in the reviews, respectively. Also, in this model, the corresponding tf-idf representation of original 1000 unigrams and 1000 bigrams are used instead of the original features to try optimize the model.

$$equationtfidf(t,d,D) = tf(t,d)\times idf(t,d,D) \tag{16}$$

where tf(t,d) = number of times the term t appears in document d and idf(t,d,D)=log N/(d"" D:t"" d). When tf-idf is high, it means this word appears much more frequently in this document compared to other documents. When tf-idf is low, it means this word appears infrequently in this document or it appears in many documents.

The text feature has some congenital advantage for predicting stars rating. For each user and business, the most direct way to predict their stars rating is to read their reviews. For machine learning, it is also true. As we tried this model to feat the linear regression, there are two methods, using it original features and using tf-idf representation. To optimize the model, we use both 1000 unigrams and 1000 bigrams as its Xtrain. It turns out to be that the original feature has a way better MSE performance. We also tried the both method with only 1000 unigrams or bigrams. The optimum solution, however, still use both bigrams features and unigrams features.

The scalability issue in this method is the lack of review text. Normally, when you do not have the stars rating of a restaurant, the review text may not be available too. In that case, other method is needed, like linear regression and latent factor method.

The overfitting for this problem is obviously too many features. I used 1000 bigrams and unigrams in my model, but I tried different numbers with it. When the features increased to 2000 bigrams and unigrams, the MSE score on my validation set dropped. This is because when you include too many not so common bigrams and unigrams in the model, other people may not mention at all in their reviews. All these feature leads the complexity to be too high and therefore the MSE dropped.

Since the overfitting issue in the last part shows that more feature may not be beneficial, I tried to use PCA to lower the dimension of the feature vectors. However, after a few attempts on the PCA method, I found that even if I delete only one dimension, the overall MSE dropped.

F. Related literature

In terms of linear regression there is another method Local Weight Regression(LWG) which can adjust parameters to train the predictor and avoid issues about under fitting and overfitting. According to [3] when we predicted a specific point we preferred to select points that were close to the specific point rather than using all points. The smaller the distance between one point and the specific point was the more weight that point wound have. A higher weight meant more contribution to the regression coefficients.

In [4] an article mentioned cross validation test which is often known as k-fold cross validation test. In this test method, the entire dataset was randomly divided into k parts and in each test, we used (k-1) parts of them as train set to obtain a model and test on the rest one part of data which worked as test set. After all k parts were used for training various models we select the one with best performance.

Nave Bayes recommender was used in [5] and it is a feasible way to tackle with cod start issues. In addition, a separate naive Bayes classifier was trained so that no collaborative information was used.

SVM used in [8] might prevent the overfitting problem and makes its solution global optimum.

In [10] N-grams of words were used with unigram and bigrams, but in the occasion where N-grams appeared sparsely the predictor will be unstable. CRR-BoO (Constrained Ridge Regression for Bag-of-Opinions) and CLO (cumulative linear offset model) were two means used in [10] to make rating predictions based on the text of review.

[11] combined the intuitive appeal of the multinomial mixture and aspect models. For online applications, it was not convenient to train the model whenever a rating was given by a user, at that time aspect models did not work well and URP (User Rating Profiles) outperforms the other methods by a significant margin.

Based on observation, [13] modeled a business with two latent factors one for its intrinsic characteristics and the other for its extrinsic characteristics (or its influence to its geographical neighbors). By cooperating geographical neighborhood influences, the new model performed much better than the state-of-the-art models including Biased MF, SVD++, and Social MF. The prediction error is further decreased by introducing influences from business category and review text. The incorporation of geographical neighborhood influence can help tackle cold start issues to some extent, since predicted rating for new businesses based on both their geographical neighbors and business categories would be reasonable and much likely precise.

III. Results and Conclusion

The MSEs of the models discussed in Section 2 on test set are shown in Table.1. Compared to the baseline, which is a trivial model always predicting star to be the global average, the models we discuss all have some improvements.

TABLE I. System Parameters

Model	MSE on test set
Baseline	1.49216246635
Linear Regression	1.069472
Logistic Regression Noise	1.05464099259
Latent Factor Model	1.28357
Feature Vectors From Review Text	0.821051820175

Linear regression model is easy to implement and can be executed rapidly even on a giant dataset. In addition, this model can explore the latent relation between diverse features when introducing several features to this prediction. However, this model can only work well when there is a linear relationship between the predicted goal and features used to generate the predictor. Even when this linear relationship exists the model may be interfered by some particular outliers therefore we need to preprocess to eliminate those outliers before we train our prediction model.

In a broad sense, logistical regression model is still a linear model, so it basically shares the strengths and weakness of the Linear Regression. The model is easy to implement and optimize, but it can only well when linear relationships exist between the predicted goal the and features used in the model. In addition, this model treats the features as they are independent.

The worst model to fit this data is latent factor model. Latent factor model is a learning method. It is good at model the personalized customer preference. In marketing strategy, we call this long tail theory. It aims to cover the increasing number of unpopular but individual requirements.

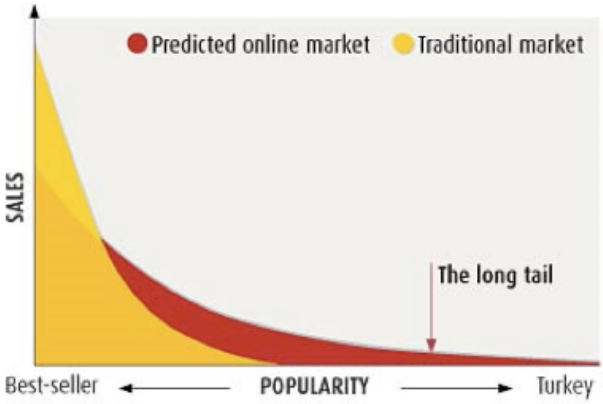

Fig. 10. Long Tail Theory

However, recalling our data set analysis in task 1. The user in this data set tend to give common high ratings. The variance of the stars is low. In this situation, we cannot take advantage of latent factor model. On the contrary, by only caring the rating that users give to business, LFM lost the information of the features that linear or logical regression use. Therefore, the model underperforms the regression model. Moreover, cold start is a more vital problem in LFM. 5359 among 29907 reviews are either from new users or toward new businesses. Features are not useful if we have many observations about users and business, but are useful for which was never observed before. Affected by large amount of cold-start issues, latent-factor model is next-to-useless.

The strength of the Feature vectors from text is that with various features in training set, we could get the lowest MSE among the four models. By including 1000 unigrams and bigrams, the most informative words, such as outstanding, 5 stars, worst and horrible, could clearly demonstrate the idea people made and therefore the stars rating they would state. The weakness of this model is due to its scalability. The model will work only if we have a large amount of review texting data. It not only means that we have to have the review text that is available, but also the length of the review text matters. For example, if we have a full review like Not Bad, it would be difficult to determine the true idea and stars rating of this review.

To sum up, we conclude the model that used feature vectors from text best fit this dataset. Review text becomes the best indicator of the star a customer would assign to a business.

IV. Applications in Physics

These four methods, Linear regression, Logic regression, Latent factor model, and Feature vectors from text can all be applied in physics. For the first three methods, they all use the digitizing data, for instance users ages, average stars. But for Feature vectors from text method, it utilizes the text data in the dataset, which shows both text information and digital information that can be made use of in our method to apply in physics.

A. Precipitation forecast

Similarly, Linear regression, logistic regression, Latent factor model all utilize an inputted variable, which is normally a vector that has multiple features with each one of these features have different values. We based on some of data, use Linear regression or logistic regression method to train a weight, and use the trained weight multiplied by the input vector, which will output a predicted value. In General, the datasets we use and the data we collected, have lots of features, and we choose a result as the predict target, for instance, in precipitation forecast, choose the precipitation as our purpose predicted value. Based on those data and their corresponding actual amount of precipitation, a weight can be trained through any one of these three methods. And then, the trained weight will be applied to make predictions of the future precipitation when the corresponding inputted data relative to the precipitation is given, as shown in Figure.11.

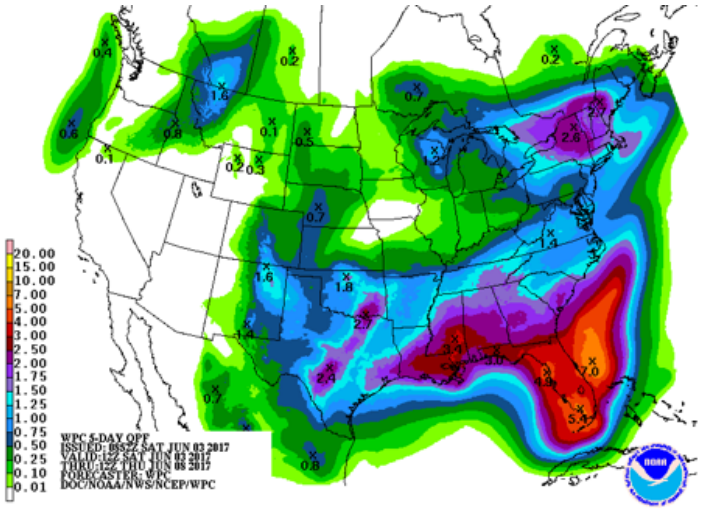

Fig. 11. Precipitation Prediction

B. Prediction of usage lifetime

The above three methods can also be applied in predicting the usage lifetime of materials or components. Take the working environments as the features of the inputted data, for example, loading conditions, surrounding environment, and choose the usage lifetime as our purpose predicted value, trained the weight, and the trained weight will be applied to make predictions of lifetime of the material or the component when the corresponding inputted data relative to the precipitation is given.

C. Prediction of Weather Based on Text

Not only the digitizing data can be utilized, the text information from dataset also very useful and easy to collected. For the Features vectors from text, when we apply this method in Weather prediction, we can select unigrams and bigrams

by the frequency they appear in the comments (as shown in Figure.11) people made about the weather in the social networks, such as Facebook, Instagram, respectively.

Fig. 12. Weather Prediction

REFERENCES

[1] Zhao Y F, Gao H, Lv Y S, et al. Latent factor model for traffic signal control[C]// IEEE International Conference on Service Operations and Logistics, and Informatics. IEEE, 2014:227-232.

[2] Zhang W, Wang J, Feng W. Combining latent factor model with location features for event-based group recommendation[C]// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2013:910-918.

[3] Machine Learning-Local Weight Linear Regression, CSDN.NET. http://blog.csdn.net/herosofearth/article/details/51969517. N.p., n.d. Web. 12 Mar. 2017.

[4] " Cross Validation" - linkin1005s blog Blog Channel - CSDN.NET. http://blog.csdn.net/linkin1005/article/details/42869331. N.p., n.d. Web. 13 Mar. 2017.

[5] Schein, Andrew I., Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. "Methods and metrics for cold-start recommendations." Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '02 (2002): n. pag. Web.

[6] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering. In Proceedings of the 2001 SIGIR Workshop on Recommender Systems, 2001.

[7] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In Proceedings of the Fifth ACM Conference on Digital Libraries, pages 195204, 2000.

[8] Lee, Young-Chan. "Application of support vector machines to corporate credit rating prediction." Expert Systems with Applications 33.1 (2007): 67-74. Web.

[9] Hsu, C.-W., Chang, C.-C., Lin, C.-J. (2004). A practical guide to support vector classification. Technical Report, Department of Computer Science and Information Engineering, National Taiwan University. Available from http://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf.

[10] Lizhen Qu, Georgiana Ifrim Gerhard Weikum. The Bag-of-Opinions Method for Review Rating Prediction from Sparse Text Patterns. COLING '10 Proceedings of the 23rd International Conference on Computational Linguistics Pages 913-921

[11] Marlin, Benjamin M. "Modeling User Rating Profiles for Collaborative Filtering."NIPS. 2003.

[12] Huang, Zan, et al. "Credit rating analysis with support vector machines and neural networks: a market comparative study."Decision support systems37.4 (2004): 543-558.

[13] Hu, Longke, Aixin Sun, and Yong Liu. "Your neighbors affect your ratings: on geographical neighborhood influence to rating prediction." Proceedings of the 37th international ACM SIGIR conference on Research development in information retrieval. ACM, 2014.

[14] National Weather Service, http://www.wpc.ncep.noaa.gov/qpf/day1-7.shtml