CSCI 5525 - Machine Learning

FALL 2015

Project Report

Predict the click-through rate of ads given the query and user information

Group ID

G9

Group Members

Charandeep Parisineti

Ravali Kandur

Varun Pandey

Predicting Click-Through-Rate of Ads.

1. ABSTRACT

In this paper, we report our approach to address the KDD Cup 2012 challenge to predict the Click-Through-Rate (CTR) of advertisements. Since most of the tech giants like Google actually earn revenue based on the number of Ads clicked by the user rather than the number of Ads shown to the user, accurately predicting the CTR of an ad is very important. To address this problem, we used/implemented three individual models, Linear Regression, Support Vector Regression (SVR) and Bayesian Online Probit Regression (BOPR) and also adopted multiple ensemble strategies (Uniform Average, Harmonic Mean and Weighted Average) on top of the CTRs reported by the individual models. Considering AUC as an evaluation metric, we observed that among all individual methods, BOPR consistently outperformed others and none of the ensemble models improved the performance, rather they have performed slightly worse than BOPR.

2. INTRODUCTION

Search engine advertising has become a significant element of the web browsing experience. Choosing the right ad at the right time is crucial to ensure a healthy click through rate (CTR) on the advertisement, which in turn has a strong impact on the revenue the search engine generates from the ads. Hence the goal of our project is to come up with better machine learning models to understand the relationship between user queries and the CTR on ads. We are using the dataset provided by 2012 KDD Cup competition [1], which is explained in detail in the next section.

A wide range of methods have been proposed for predicting click-through rates in recent years. They can be broadly categorized as: statistical approaches including variants of regression [2, 3, 7, 16, 19], Bayesian models [4, 6, 18], online learning models [2, 4, 13], collaborative filtering models [19, 24, 25], spatio-temporal models [10]; machine learning approaches [15, 22] including SVM [6], boosting [5], neural networks [7]. In addition, there are a few approaches focusing on a combination of such models [6, 7, 8, 12, 20].

As part of the project, we planned on evaluating the performance of some individual algorithms (discussed in section 4.3) and also some ensemble strategies on top of these individual models (discussed in section 4.4) using Area Under Curve (AUC) as an evaluation metric (discussed in section 4.5).

3. DATA

The dataset has approximately 150million training samples and 20million test samples, which are derived from search session logs of the Tencent proprietary search engine, soso.com. Data is broadly classified into the following categories: user (gender, age), ad (advertiser id, keywords from title, description, url, number of ads shown in a session, relative position of an ad, number of impressions, clicks) and query issued by the user. Also, since the data set is huge, only aggregate instances with the same user id, ad id, query, and setting are provided, therefore we do not have any timestamp information associated with them. All the words are only given as hash values for privacy reasons.

Although the KDD competition provides a separate test data set, we are unable to use that data for evaluation since the true CTR features of test data are hidden and also the submission link of the KDD challenge is closed. Hence, there is no clear way to evaluate the CTR values predicted by our models. Therefore, we created a separate set from unused training data to be used as the test data. Since the dataset is huge, we only used 0.5million samples to train and test our models. Beyond this, we faced issues due to limited memory, compute power on the machines available to us. Trying to solve these issues was leading us into solving an entirely different domain of problems. Therefore, we confined the scope due to the time constraint, as per the course guidelines. More details on data in Appendix A.

4. APPROACH

To improve the model's ability to predict the CTR better, our approach was as follows:

(1) Data preprocessing – to make the existing data usable; and adding some interesting aspects which are otherwise hidden in the raw form.
(2) Feature engineering to make sure that all the features used are actually contributing towards the overall performance achieved by the individual algorithms.
(3) Consider an ensemble of the individual models to reduce the variance in results if any.

4.1 DATA PREPROCESSING

Categorical features like age, gender, etc., are converted into binary features based on categorical values (i.e., each feature value will be 1 if the user belongs to that category, 0 otherwise). For example, gender is categorized as Male, Female and Others. Therefore we need three binary features to represent each of these categories. If the user for a given sample is male, then the values of these binary features would be [1, 0, 0].

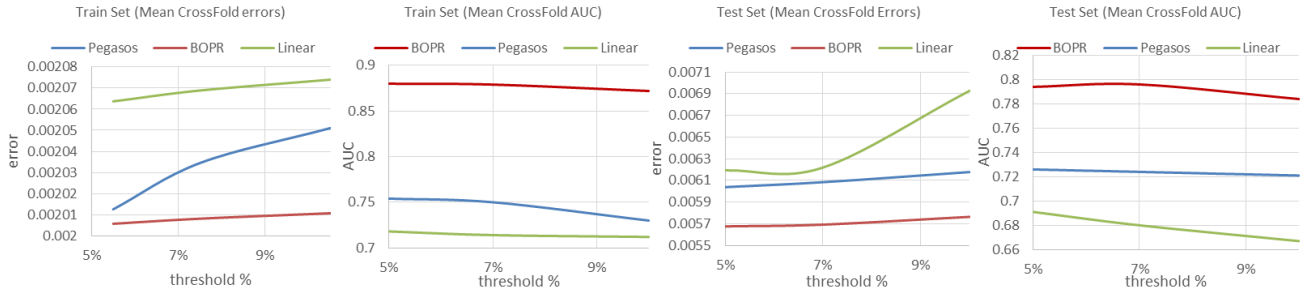


Figure 1: Mean Error/AUC (from 10 fold cross validation) with respect to the value of p.

We expanded query, title, description, keyword word tokens into binary features i.e., first compute frequency of all tokens from the entire data, some of these features become popular features if the frequency exceeds certain threshold, and new binary features are added for each of these popular features. Threshold is decided as p% of the top most frequency, for example, if the top most frequency is 100 and the value of p is 10, then the threshold will be 10. Each of these popular feature value will be 1 if a given sample contains that token, 0 otherwise. Based on our experiments of analyzing the impact of varying p on the performance of the algorithms (*figure 1*), p = 7 was believed to be the optimal choice as it was giving the best tradeoff between the number of popular features (scalability) vs AUC. Increasing p further although leads to adding less number of features, the Accuracy/AUC are lower when compared to p = 7. On the other hand, decreasing p although improved the Accuracy/AUC slightly, the amount of overhead caused due to more number of features was very high leading to very long run times (*figure 2*).

For each categorical features, like AdId, AdvertiserId, etc., we computed average CTR as an additional feature. For example, for each AdId, we compute the average CTR for all instances with same AdId and use this as single feature. However, since some categories come with few or no instances at all, computing average CTR directly in those cases could lead to inaccurate estimations because of insufficient statistics. Thus, we added smoothened out values using (#click + (a*b))/(#impression+b) where a=0.05, b= 0.75 as suggested in [7] as another set of features.

Since we believe that correlation between raw features like query, title, description, and keyword could be useful in determining the CTR, we computed cosine similarity between their corresponding TF-IDF vectors [26], treating them as documents with a set of words.

We added some numerical features like relative position, length features like #tokens in each query, ad etc., frequency features like frequency of AdId, AdvertiserId, queryId_AdvertiserId, userId_queryId, userId_keyword, etc., Features like QueryId, AdId are eliminated after generating new features using the supplemental data (like query, title and description words, keywords, etc) as mentioned below.

4.2 FEATURE ENGINEERING

As we added approximately 200 new features based on the above mentioned ideas, we were curious to see which of these sets of features have contributed (if at all) more towards the overall AUC. We ran the three algorithms in three different scenarios, (1) baseline - considering all the features (2) excluding TF-IDF based cosine similarity features and (3) excluding popular features. From the results (*figure 3*) we see that, both these sets of features provide a significant contribution towards the final AUC for each of these algorithms. Although we wanted to extend this analysis for all the sets of features that we added, due to time constraint, we could only include those which we thought were the crucial ones.

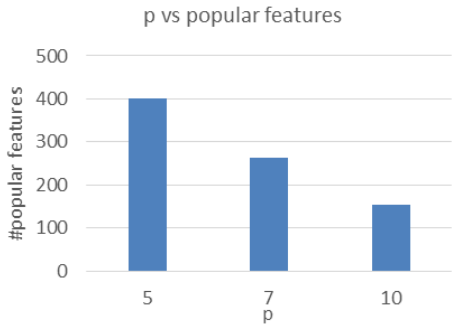


Figure2: Number of popular features with varying p.

4.3 INDIVIDUAL MODELS

Initially we were planning on trying out models from several areas to solve our problem, such as Regression models, Bayesian models, Neural Networks based models and Mixture models. However, as per Professor Arindam's suggestion, we decided to narrow down the scope of the project to only focus on the Regression models mentioned below and try an ensemble of these models if time permits.

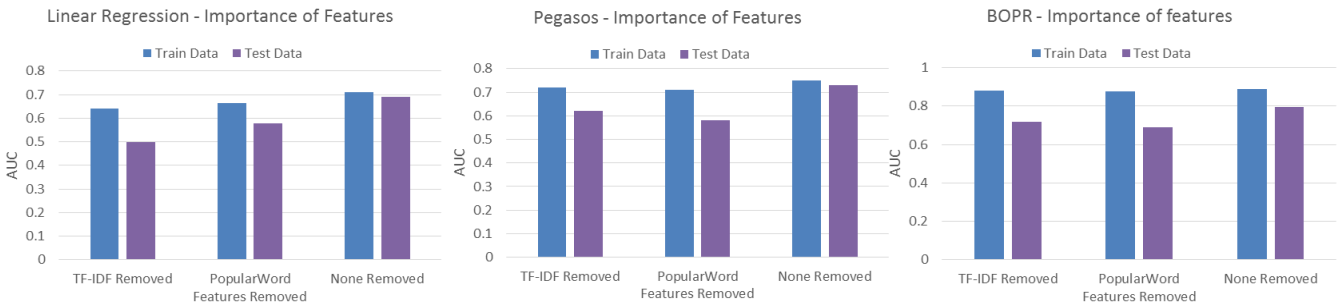


Figure 3: Change in AUC when some of the features are removed.

The state of art work talks about BOPR algorithm when applied to the Online learning setting to predict the CTR of Ads. Since the problem we are trying to solve is quite different from the online learning due to the nature of the data we have, we wanted to see how the BOPR algorithm behaves in our current setting. Because of which, we have chosen one of the regression models as the BOPR algorithm. Also, we wanted to see how BOPR behaves against the Regression models which were earlier used and proven to be useful (*Linear Regression* and *Support Vector Regression*) for predicting the CTR of Ads in the internet industry. Hence the three individual models we decided to use/implement are *Linear Regression*, *Support Vector Regression* and *Bayesian Online Probit Regression*, and then try out different ensemble strategies on top of these individual models.

4.3.1 LINEAR REGRESSION

We used scikit API for this algorithm, which is based on the ordinary least squares Linear Regression method. For computing the AUC and plotting the ROC curve, we used the methods provided by scikit. Our experiments show that the AUC achieved by using this method alone was 0.69 which is the worst of all (*figure 4*, *5*). The reason for this could be because of the existence of the outliers in the data considered.

4.3.2 SUPPORT VECTOR REGRESSION

Support Vector Regression (SVR) is proposed by Vapnik [28], which is an extension from support vector classification [29]. SVR solves the below mentioned optimization problem where C > 0 is the regularization parameter, and ξ is the squared epsilon insensitive loss function given below. We initially used the scikit API for this algorithm, but since the algorithm does not scale well on larger datasets, we later used the Pegasos approach as discussed in [30] for the same. For computing AUC and plotting the ROC curve, we used the scikit API methods.

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{N}\xi_\epsilon(\boldsymbol{w};\boldsymbol{x}_i,y_i)$$

$$\xi_\epsilon(\boldsymbol{w};\boldsymbol{x}_i,y_i) = \max(|\boldsymbol{w}^T\phi(\boldsymbol{x}_i) - y_i| - \epsilon, 0)^2$$

Our experiments show that the AUC achieved by using this method alone was 0.73 which is better than Linear Regression (*figure 4*, *5*), the reason could be that this method is more robust to the outliers. However, we have expected SVR to perform close to 0.8 AUC as per the state of art work done. We are still unsure of the exact reason, but we suspect that this is because of some features that we haven't considered and could be influential in boosting up the AUC.

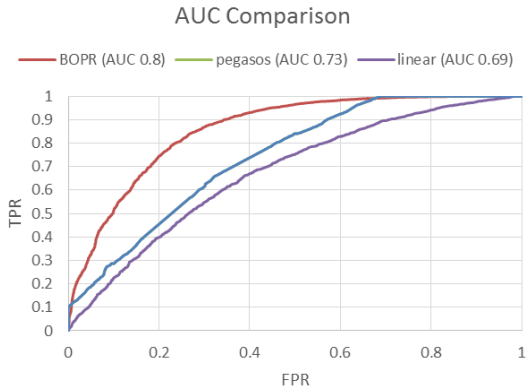


Figure 4: Performance of the individual algorithms (mean of AUC for 10 folds).

4.3.3 BAYESIAN ONLINE PROBIT REGRESSION

The Bayesian Online Probit Regression (BOPR) was introduced by the researchers of Microsoft [4]. The model is based on a probit regression model that maps discrete or real-valued input features to probabilities. It maintains Gaussian beliefs over the weight of the model and performs Gaussian updates derived from approximate message passing. We are only providing the update equations used to build the model and the equations used for prediction in the report due to the space constraints, more information about the algorithm can be found in [4]. For computing AUC and plotting the ROC curve, we have used the scikit API methods. The update for the posterior parameters is given by

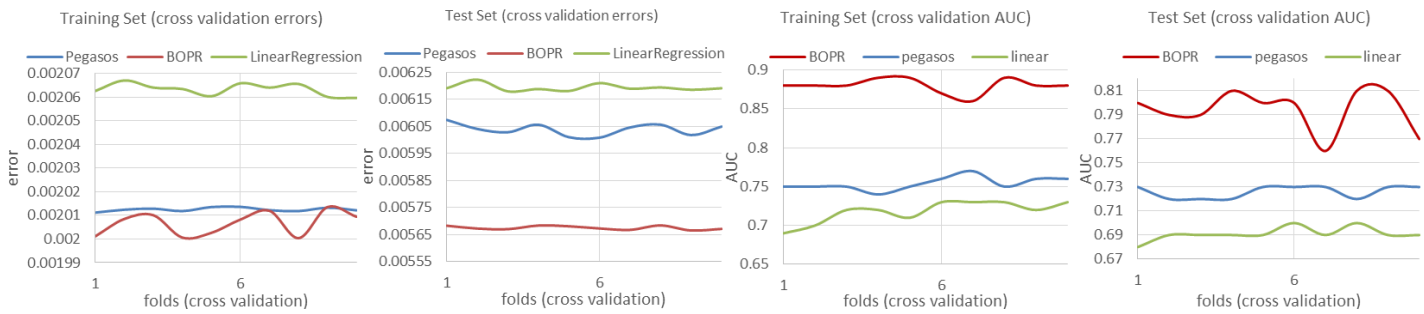


Figure 5: Behavior of the individual algorithms per each fold (Error and AUC).

$$\tilde{\mu}_{i,j} = \mu_{i,j} + yx_{i,j} \cdot \frac{\sigma_{i,j}^2}{\Sigma} \cdot v\left(\frac{y \cdot \boldsymbol{x}^T\boldsymbol{\mu}}{\Sigma}\right) \qquad \tilde{\sigma}_{i,j}^2 \leftarrow \sigma_{i,j}^2 \cdot \left[1 - x_{i,j} \cdot \frac{\sigma_{i,j}^2}{\Sigma^2} \cdot w\left(\frac{y \cdot \boldsymbol{x}^T\boldsymbol{\mu}}{\Sigma}\right)\right]$$

$$v(t) \coloneqq \frac{\mathcal{N}(t;0,1)}{\Phi(t;0,1)}, w(t) \coloneqq v(t) \cdot [v(t) + t].$$

The predictive distribution can be given as shown below.

$$p(y|\boldsymbol{x}) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} p(y|\boldsymbol{x}, \boldsymbol{w}) \cdot p(\boldsymbol{w})\, d\boldsymbol{w}. \quad \text{====>} \quad p(y|\boldsymbol{x}) = \Phi\left(\frac{y \cdot \boldsymbol{x}^T\boldsymbol{\mu}}{\Sigma}\right)$$

We have implemented this algorithm by ourselves since there was no API which was available for the same. Our experiments show that the AUC achieved by using this method alone was 0.8 which is the best of all the other methods considered (*figure 4*, *5*).

4.4 ENSEMBLE STRATEGIES

In order to improve test set performance further, we made an ensemble of the individual models discussed above using the strategies shown below.

(1) Uniform Average method - computes uniform average of the CTR's predicted by the individual models.
(2) Harmonic Mean method - computes harmonic mean of the CTR's predicted by the individual models.
(3) Weighted Average method - computes weighted average of the CTR's predicted by the individual models. Weight of the individual models are proportional to the AUC achieved by the respective models.

Ensemble Strategy	***AUC***
Uniform Average	0.640
Harmonic Mean	0.695
Weighted Average	0.78

Table 1: AUC values averaged out for 10 folds for each of the ensemble strategies.

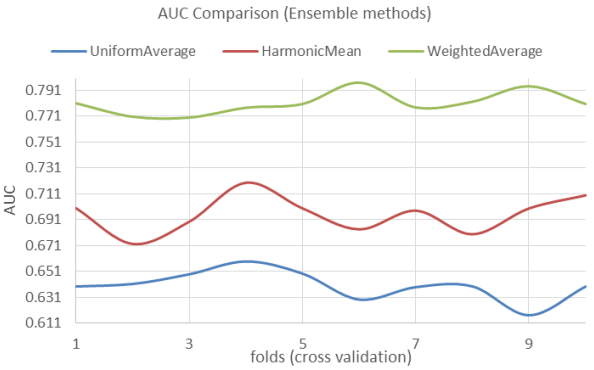


Figure 6: Behavior of the ensemble strategies per fold.

Although we expected the ensemble strategies to perform better than the individual models, our experiments (*Table 1*, *figure 6*) suggest that the ensemble methods are in fact doing slightly worse than the best individual model BOPR.

We think that the reason for this could be because of the underlying individual models not performing as much as we expected. Due to which, even though some of our models do a good job, the degraded performance of the other algorithms is pulling the overall performance of the ensemble model down. This behavior can be clearly seen with the Uniform Average ensemble method. Even though the Harmonic Mean method is doing slightly better than the Uniform Average method, it is still affected due to the reason mentioned above.

On the other hand, Weighted Average is slightly boosted when compared to the other two methods because the algorithm weighs the decision taken by the better performing algorithms more than the decision taken by the other algorithms. However, this method is also still affected by the low performing models because of which the overall performance might have still got affected.

Therefore even this model struggles to cross the performance benchmark set by the best performing individual model BOPR.

4.5 EVALUATION

Since the data set is skewed, i.e., number of samples which have non-zero clicks are very less (about one tenth) when compared to the number of samples with zero clicks, Accuracy will not be a reliable metric for evaluation because, a simple classifier which classifies all samples as zero clicks could also give a higher accuracy, but it doesn't serve our purpose. On the other hand, ROC evaluates the classifier based on both TPR and FPR unlike Accuracy, which considers only the number of True Positives. Since AUC is the Area under the ROC curve, AUC acts as a good general measure of measuring performance. Therefore, we use AUC [8] as the evaluation metric, which is also the metric suggested by the KDD Competition.

5. EXPERIMENTAL SETUP

We used our local machines for running the experiments as none of the CSE lab machines allowed us to store the train data due to the per user memory limit. We have run experiments using K-fold cross validation where K was 10.

Since the dataset is huge, we only used 0.5million samples to train and test our models. Beyond this, we faced issues due to limited memory, compute power on the machines available to us. Since maintaining the original distribution of the dataset plays a crucial role in the behavior of the algorithms on the test set, we have evaluated the overall distribution of the data samples and the small dataset with 0.5 million samples is generated in such a way that it adheres to the original distribution.

6. RESULTS

We have used AUC to compare the performance of the individual models. Each of the individual models are trained and tested in a similar fashion, where the number of the train and test data samples considered is about 0.5 million. As we can see from (*figure 4*), BOPR has the highest AUC (~0.8) when compared to the other Algorithms and consistently out performs the other algorithms whereas Linear Regression performs the worst of all.

As stated in the earlier sections, the reason for this behavior of Linear Regression could be because of the existence of outliers in the data considered. This is however overcome by SVR because of which its performance is slightly boosted when compared to Logistic Regression.

From the state of art work, although we were guessing BOPR to perform at least as close as SVR, we were pleasantly surprised by the extent to which BOPR outperforms the others. We are not entirely sure of the exact reason for why BOPR bumps up the AUC. Had there been some more time, we are certain that would have investigated more to understand this behavior.

Although we expected the ensemble strategies to perform better than the individual models, we were surprised to see that they are in fact doing slightly worse than the best individual model BOPR. As mentioned earlier, we think that the reason for this could be because of the underlying individual models not performing as much as we expected. Due to which, even though some of our models do a good job, the degraded performance of the other algorithms is pulling the overall performance of the ensemble model down.

7. CONCLUSION

In this report, we discuss our approach to address the challenge posted in KDD Cup 2012 as a part of the Machine Learning class project. We model users' behavior on click-through rate by using three different individual models based on Regression. We also discuss our results using three ensemble strategies designed to combine the results of individual models and boost the performance.

8. FUTURE WORK

We had to limit out evaluation to a subset of the data, since our algorithms did not scale well. While this is alright for academic purposes, a more practical approach would be to develop scalable algorithms so that larger datasets can be considered using the strategies mentioned in [2, 5].

Further, we can explore ideas to improve the AUC by using new models, improving the existing ones, additional feature engineering, better ensemble strategies, etc., Since the nature of the dataset blocked us from using online learning strategies, there could be some work done to collect a new data set with time stamp information and pose the same problem as an online learning problem and explore algorithms and strategies in that domain.

9. POINTS TO NOTE

Due to the space constraint in the main section of the report, all the graphs are shrunk. Clear version of the graphs are given in Appendix B.

9. REFERENCES

[1] 2012 KDD Cup Challenge.
[2] Ad Click Prediction: a View from the Trenches
[3] Predicting Clicks: Estimating the Click-Through Rate for New Ads
[4] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-Scale Bayesian ClickThrough rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine. In International Conference on Machine Learning, pages 13–20, 2010.
[5] Practical Lessons from Predicting Clicks on Ads at Facebook
[6] Click-Through Prediction for Sponsored Search Advertising with Hybrid Models
[7] A Two-Stage Ensemble of Diverse Models for Advertisement Ranking in KDD Cup 2012
[8] ROC Graphs: Notes and Practical Considerations for Researchers
[9] Juan Ramos. Using TF-IDF to Determine Word Relevance in Document Queries.
[10] D. Agarwal, B.-C. Chen, and P. Elango. Spatio-temporal models for estimating click-through rate. In Proceedings of the 18th international conference on World Wide Web, pages 21–30. ACM, 2009.
[11] O. Chapelle. Simple and scalable response prediction for display advertising
[12] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine learning, 40(2):139–157, 2000.
[13] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. In COLT, 2010.
[14] H.-Y. Lo, K.-W. Chang, S.-T. Chen, T.-H. Chiang, C.-S. Ferng, C.-J. Hsieh, Y.-K. Ko, T.-T. Kuo, H.-C. Lai, K.-Y. Lin, C.-H. Wang, H.-F. Yu, C.-J. Lin, H.-T. Lin, and S.-D. Lin. An ensemble of three classifiers for KDD Cup 2009: Expanded linear model, heterogeneous boosting, and selective naive bayes.
[15] D. Hillard, S. Schroedl, E. Manavoglu, H. Raghavan, and C. Leggetter. Improving ad relevance in sponsored search. In Proceedings of the third ACM international conference on Web search and data mining, WSDM '10, pages 361–370, 2010.
[16] D. W. Hosmer and S. Lemeshow. Applied logistic regression. Wiley-Interscience Publication, 2000.
[17] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In Proceedings of 16th international conference on WorldWideWeb, pages 521–530. ACM, 2007.
[18] Z. A. Zhu, W. Chen, T. Minka, C. Zhu, and Z. Chen. A novel click model and its applications to online advertising. In Proceedings of the third ACM international conference on Web search and data mining, pages 321–330. ACM, 2010.
[19] K. Bartz, V. Murthi, S. Sebastian, "Logistic Regression and Collaborative Filtering for Sponsored Search Term Recommendation", In Proceedings of the Second Workshop on Sponsored Search Auctions, 2006.
[20] A. Mohan, Z. Chen, and K. Q. Weinberger. Web-search ranking with initialized gradient boosted regression trees. In JMLR W&CP, volume 14, 2011.
[21] B. Jansen and M. Resnick, "Examining Searcher Perceptions of and Interactions with Sponsored Results," In Proceedings of the Workshop on Sponsored Search Auctions, 2005.
[22] M. Regelson and D. Fain, "Predicting click-through rate using keyword clusters," In Proceedings of the Second Workshop on Sponsored Search Auctions, 2006.
[23] Broder, A., & Josifovski, V. (2009). Lecture Introduction to Computational Advertising. Stanford University, Computer Science. Online Lecture Notes.
[24] T. Anastasakos, D. Hillard, S. Kshetramade, and H. Raghavan. A collaborative filtering approach to ad recommendation using the query-ad click graph. In International Conference on Information and Knowledge Management, pages 1927–1930, 2009.
[25] S. Shen, B. Hu, W. Chen, and Q. Yang. Personalized Click Model through Collaborative Filtering. In Web Search and Data Mining, 2012.
[26] G. Salton, A. Wong, and C. S. Yang. Readings in information retrieval. Chapter A vector space model for automatic indexing. Morgan Kaufmann, 1997.
[27] A Tutorial on Support Vector Regression
[28] V. Vapnik. The Nature of Statistical Learning Theory. Springer-Verlag, 1995
[29] B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In COLT, 1992.
[30] Pegasos: Primal Estimated sub-GrAdient SOlver for SVM.

APPENDIX A

A.1 TERMINOLOGY

Feature	*Description*
session	interaction between a user and the search engine
depth	number of ads impressed in a session
position	order of an ad in the impression list
title, description	Ad, when impressed, would be displayed as a short text known as 'title', followed by a slightly longer text known as the 'description'
display URL	URL usually shortened to save screen space
CTR (Click Through Rate)	#clicks/#impressions

A.2 ORIGINAL TRAINING DATA

Each session is divided into multiple instances, where each instance describes an impressed ad under a certain setting (i.e., with certain depth and position values). Therefore, schematically, each instance contains at least the following information:

Feature	*Description*
Session	interaction between a user and the search engine
impression	number of search sessions in which AdId was impressed by UserID who issued the Query(QueryId)
depth	number of ads impressed in a session
display URL	Hash value of the shortened landing page URL
AdId	Id associated with an Ad
Advertiser Id	Advertisers Id
depth	number of impressions in a session
position	order of an ad in the impression list
QueryId	zero based integer value which points to the index in the file which gives set of words in the query
keyword Id	zero based integer value wshich points to the index in the file which gives set of words in the keywords
title Id	zero based integer value which points to the index in the file which gives set of words in the title
description Id	zero based integer value which points to the index in the file which gives set of words in the description of the Ad
user id	zero based integer value which points to the index in the file which gives set of attributes associated with the user. The attributes are gender and age where, gender: 1 -> male, 2 -> female, 0 -> unknown age: 1 -> (0,12], 2 -> (12 18], 3 -> (18, 24], 4 -> (24, 30], 5 -> (30, 40], 6 -> > 40

A.3 TRANSFORMED DATA

Feature	*Description*
user age and gender	binarized features from user age and gender categories
click	number of times ad was clicked by the user
impression	number of times ad was shown to the user
depth	number of impressions in a session
position	order of an ad in the impression list
user id	Id which is used to denote a specific user
ctr	#clicks/#impressions
frequency features	binary features added using the top most words seen in keyword, description, title, query words
similarity features	4C2 = 6 features which measures pairwise similarity between the words from keyword, description, title, query words
frequency features	frequency of occurrence of the feature values like (AdId, AdvertiserId, userId_queryId, userId_keyword, queryId_AdvertiserId)
smoothened average ctr features	smoothened values of average ctr measured for each of (AdId, AdvertiserId, depth, position, relative position)
average ctr features	average ctr measured for each of (AdId, AdvertiserId, depth, position, relative position, age, gender, userId)
count features	#words in query, ad title, description, keywords

APPENDIX B

Figure 1

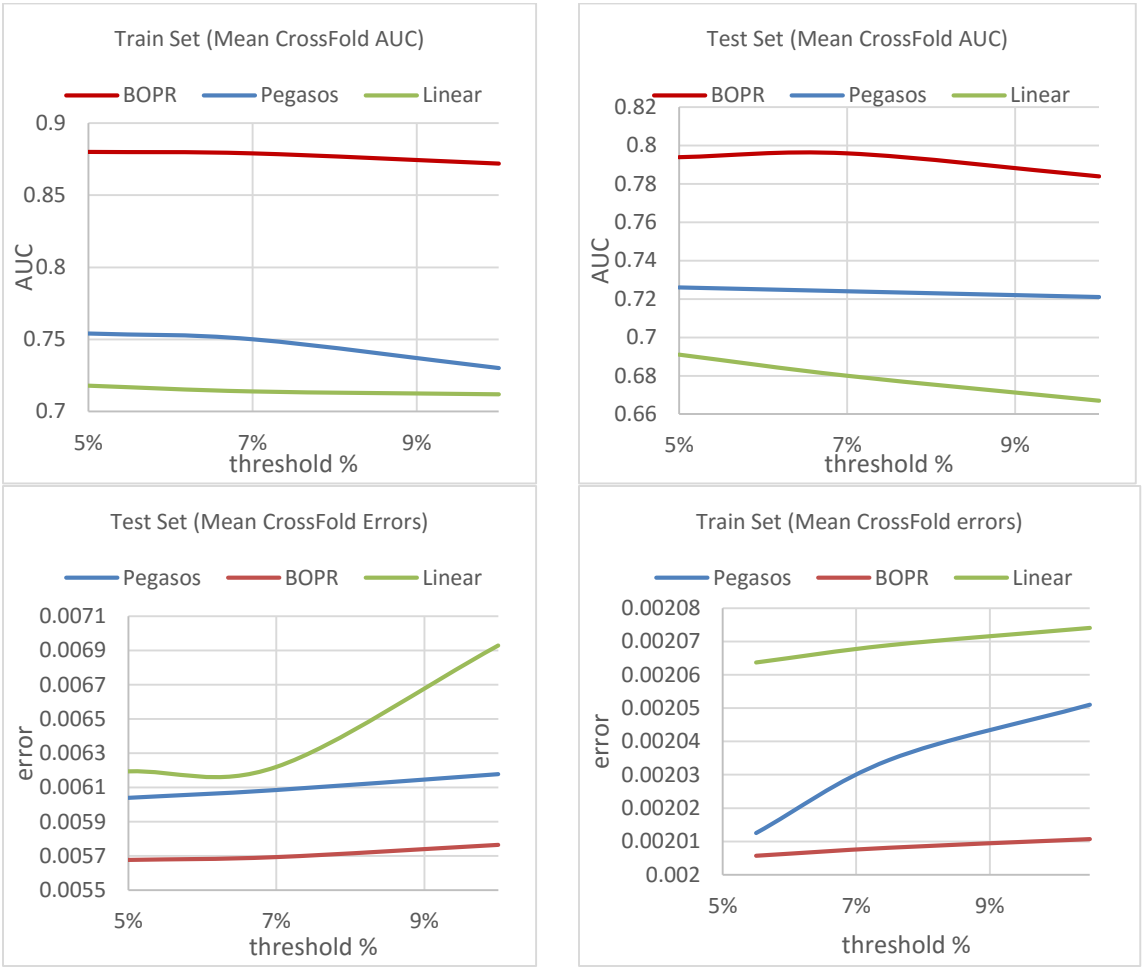


Figure 2

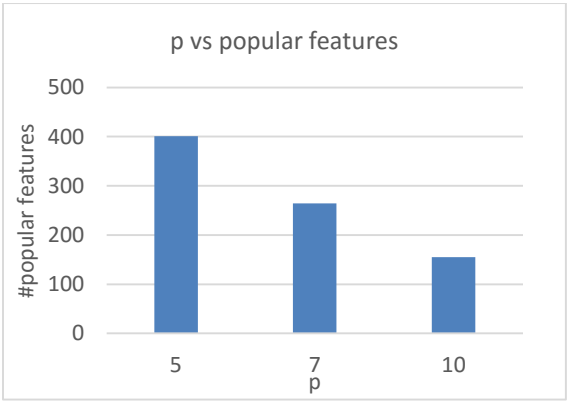
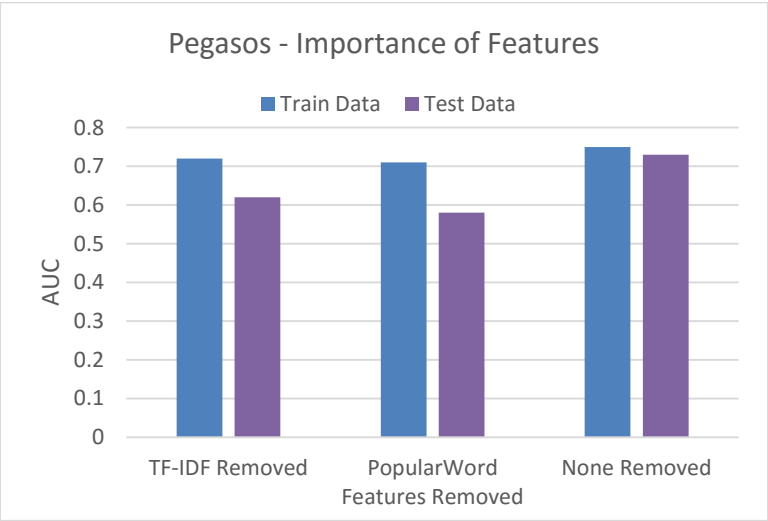
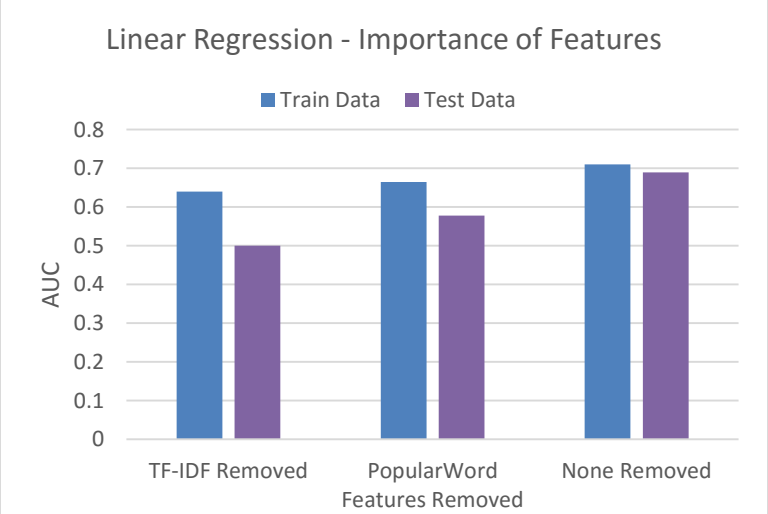


Figure 3


Pegasos - Importance of Features
Train Data
Test Data
AUC
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
TF-IDF Removed
PopularWord Features Removed
None Removed


Linear Regression - Importance of Features
Train Data
Test Data
AUC
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
TF-IDF Removed
PopularWord Features Removed
None Removed

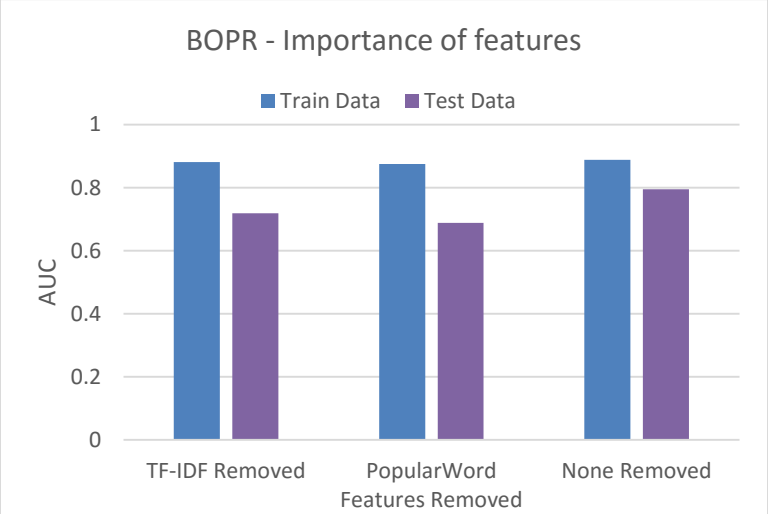

BOPR - Importance of features
Train Data
Test Data
AUC
1
0.8
0.6
0.4
0.2
0
TF-IDF Removed
PopularWord Features Removed
None Removed

Figure 4

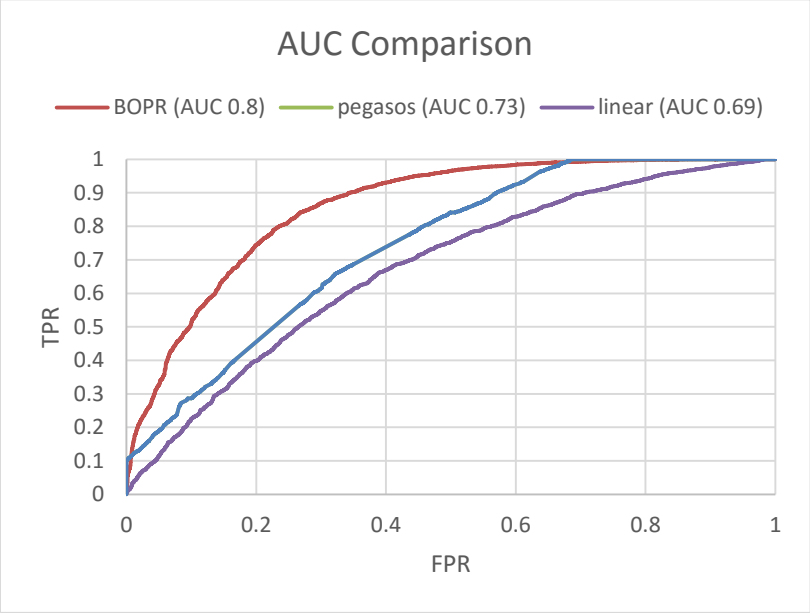
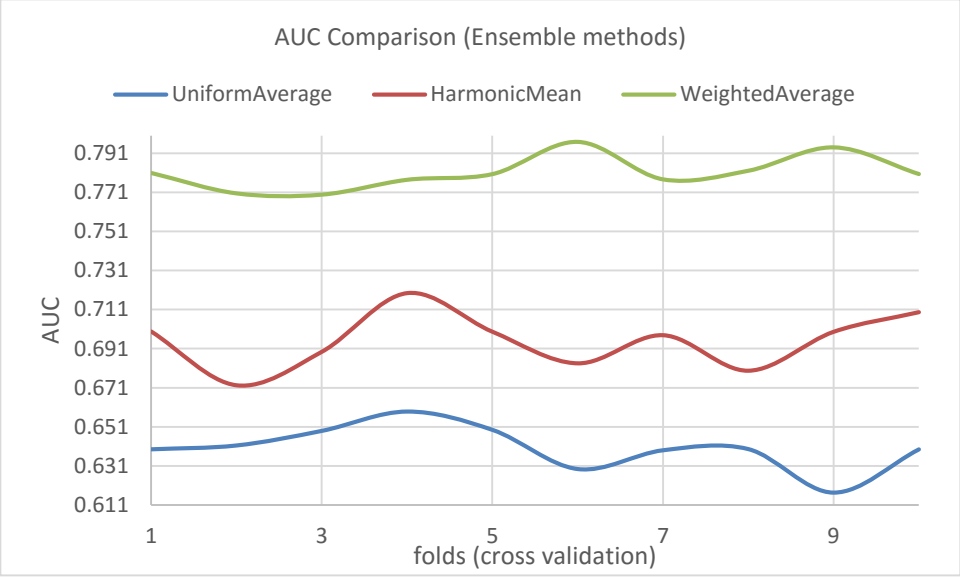


Figure 5

Figure 6


AUC Comparison (Ensemble methods)
UniformAverage
HarmonicMean
WeightedAverage
AUC
0.791
0.771
0.751
0.731
0.711
0.691
0.671
0.651
0.631
0.611
1
3
5
7
9
folds (cross validation)

CONTRIBUTIONS

Our code base which was used for the project has been tracked in Github.

CHARANDEEP PARISINETI

- Worked on researching about the application of IF-IDF computation to compute the similarity between different components in the raw dataset.
- Implemented the code to add the TF-IDF features as a part of data preprocessing.
- Implemented the algorithms Linear Regression and SVM using scikit API. However, SVM was disregarded later as we were considering this as a regression problem rather than a classification problem.
- Implemented the code to compute the AUC and plot the ROC curve using scikit API.
- Worked on the documentation of the work done as a part of Project Proposal, Progress and Final Report.
- Worked on creating a presentation and presenting the work done to the class.

RAVALI KANDUR

- Worked on researching about the different individual models that can be considered to solve the problem.
- Worked on writing code to clean up unnecessary data from the raw training data to filter only a subset of data for our consideration which roughly has the same distribution when compared to the entire dataset.
- Made following contributions to the preprocessing framework (1) compute and add popular word features (2) Binarize categorical features. (3) Added JAVA version to add frequency related features, average CTR features to the dataset for scale up.
- Implemented SVR using Pegasos and Bayesian Online Probit Regression.
- Worked on the documentation of the work done as a part of Project Proposal, Progress and Final Report.
- Worked on creating a presentation and presenting the work done to the class.

VARUN PANDEY

- Worked on researching about the application of different ensemble strategies over the individual models.
- Implemented the code to do an ensemble of the individual models using Uniform Average, Weighted Uniform Average and Harmonic Mean aggregations on top of the CTRs reported by individual models.
- Made following contributions to the preprocessing framework using MATLAB
    - Add frequency of occurrence of certain tuples of features.
    - Add average CTR features.
- Worked on the documentation of the work done as a part of Project Proposal, Progress and Final Report.
- Worked on creating a presentation and presenting the work done to the class.