Avito Context Ad Clicks

Final Project Report- Data Analytics

Maanav Mehrotra
Virginia Tech
Blacksburg, Virginia
maanav@vt.edu

Mohak Bheda
Virginia Tech
Blacksburg, Virginia
mohakbheda@vt.edu

Parth Vora
Virginia Tech
Blacksburg, Virginia
parthv@vt.edu

ABSTRACT

This report discusses the project done for Data Analytics course. In this project, we take the Avito's ad click prediction problem hosted on Kaggle and develop models to predict whether a user clicks on the ad, given a set of features. We first start with merging the relational databases into one table of features. Then move onto to the data preparation phase, where a bulk of entries are removed due to noisy information. In the next step, we explore the data and draw interesting insights from the data. We then move on to the model building step, where we explore classification algorithms like Logistic Regression and ensemble classifiers like Random Forest to model the training data. We also use techniques like grid search to tune model hyperparameters and cross-validation to improve model performance. In the final step, we evaluate our models using various metrics and compare the performance. We can achieve a maximum of 0.72 recall with XGBoost model. For the extra credits section, we perform some interesting analytics on Russian textual data and use the genetic algorithm to get the features that most affect the classification.

KEYWORDS

Ensemble Classifier, XGBoost, Ad Click Prediction

1 INTRODUCTION

With the advent of the internet, all services have started moving online. Out of all industries moving online, the one that is booming the most is the e-commerce industry. Online e-commerce platforms like Amazon and Alibaba have millions of customers every year making around billions of transactions. From a packet of milk to even guns, everything can be purchased with a click of a button. The Internet has also enabled individuals to post items for sale online and earn money by sitting at home.

With so many selling entities with similar products online, there is much competition. Advertisements help a brand or an individual stand out from the rest. However, these publications come at a cost. Advertisements have proved to be so effective that organizations have an entire department dedicated to ad campaigns. However, this reach and impact on customers come at a cost. Depending on the size of the ad and placement, prices vary. The paid publication is often expensive and calls for a strategic solution to get the most out of money invested. This is where ad click prediction comes in. Ad click prediction is a systematic way to study the usage patterns and how they interact with ads online to make meaningful predictions and tailor the ads to target even a broader audience. In some cases, the end goal is not to make the user click on the ad, but to make him/her purchase the product as well.

Avito is an online marketplace that operates in Russia. If you want to buy a laptop or a piece of land, the first place to search for it is Avito.ru. A country as big as Russia has millions of users interacting with ads daily. Traditionally, Avito used standard algorithms to show ads to the user. However, to change this standard method which only considered ad statistics, it has collected user interaction data throughout two months and posted a competition online to help them predict whether a user clicks on an ad or not. Avito provides three types of ads to its sellers- regular, highlighted and contextual. Contextual ads are unique because the seller pays per click on the ad. The goal of this project is to predict whether a particular ad is clicked. These predictions help Avito show relevant ads to the consumer and at the same time save costs for the seller.

The rest of the report is organized as follows; section 3 gives the readers an idea about the motivation to select this problem definition. Section 5 introduces the readers to the dataset provided by Avito and provides some statistics about it. Section 6 discusses the data preparation phase, where we combine relational database tables into one table and perform data pre-processing. In section 7, we perform data exploration by plotting data columns using the python seaborn library. Section 8 discusses the model training, tuning and evaluation using various techniques. Results are discussed in section 8, and section 10 discusses the work done for extra credits. While M.Bheda and M.Mehrotra handled data Pre-Processing and Exploration, Model building and Evaluation is handled by P. Vora.

2 RESPONSE TO PROPOSAL COMMENTS

First comment was regarding the evaluation metrics being considered to gauge the model performance. We have focused on increasing the model performance especially concerning recall/sensitivity of the 'Click' class label. In a practical use case scenario, we do not care if the user will not click on an ad, but we do care if they will. Hence, most model tuning revolves around recall, and it will be fine to have false positives also. The second comment was regarding there not being enough features. However, the training data was not given to us right away. We created it by merging the data columns from different tables given in the relational database. We were able to create a training data of about 25 features using all the tables. There can be many more features that could be created, but because of limitations on computational resources, we had to focus our attention only on the important ones.

We used Random Forest Classifier, Adabooster with Decision Tree classifier, and XGBoost ensemble built on the binary logistic classifier. We have explored a couple of techniques to improve the classifier performance. We use GridSearch to find the best set of hyperparameters for the model. We then take this best set and perform cross-validation with the objective of maximizing the recall.

Moreover, for Random Forest Classifiers we improve performance by giving weight to classes that take care of the unbalanced class problem. While for XGBoost we change the classification threshold to address the unbalanced class issue.

3 MOTIVATION

Many machine learning problem definitions have two tables- the training set and testing set. However, this problem has a set of 8 relational databases which inter-connect with each other to create the training set and the testing set. Doing this enables us to explore and employ SQL techniques, to create the training set. Moreover, the sheer size of the relational database and the training set makes it mandatory that we handle the data most efficiently as possible to prevent the computational resources from bottle-necking us. Since this data is collected from the servers of an operating website, it also gives us an idea of how the real-world data looks like and how to handle problems like, missing and improperly logged data. Ads have much textual data as well, so we get to learn and implement ways to process textual data also. Since this data is from a 2015 Kaggle competition, we also get the opportunity to compare our results with modern models with the competition winning solutions.

4 WORKFLOW

In this section, we discuss the overall workflow of this project. The objective of the project is to be able to predict with the highest accuracy, whether a particular ad is clicked on. Figure 1. shows the workflow of this project. In the first step, we call the Kaggle API to collect data and aggregate it into training data using SQL join operations. After getting the training data, we pre-process it to make it easier to model the data. We then move on to exploration stage where we draw meaningful insights from the dataset and use it to tune our model. In the next step, we train and evaluate various models to select the best model for our data.

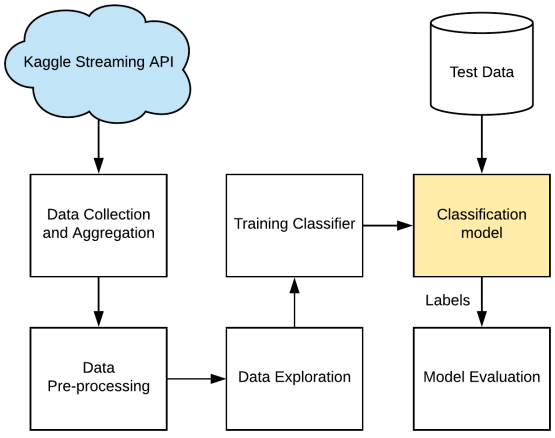


Figure 1: Workflow

5 DATA DESCRIPTION

We obtain data using the Kaggle API. It provides us with 8 relational tables in SQL database format and training dataset. Figure 2 shows the relational schema of the tables provided. Now the train data set contains identifiers to Search and the Ad table. To complete the training dataset, we must join the search and ads table with the training dataset. Search table, on the other hand, contains identifiers to User, Location and Category table. While Ads table contains identifiers to Location and Category tables, so, to aggregate the tables, we first join the Ads, Location and Category tables. Then, join the aggregated Ads table with the training set. Later, we join the Search table with the User, Location and Category tables. Finally, we merge the aggregated Search table with the training dataset to get the complete training dataset. Table 1 shows the fields present and size of individual description. Table 2 gives information about each column present in tables AdsInfo, CategoryInfo, LocationInfo, UserInfo, SearchInfo, and TrainSearchStream respectively. Figure 2 shows the schema of the relational database.

Field	Description
SearchID	identifier for a visitors' search event.
AdID	identifier of the ad.
Position	position of the ad in search result page is first ad on a page starting from the top.
ObjectType	type of the ad shown to user. The options are: 1 - regular free ads added by users; 2 - highlighted regular (owners have to pay fixed price to highlight them and stick to the top for some period of time); 3 - contextual ads (owners have to pay per visitor's click).
HistCTR	some naive history-based estimation of click-through rate for contextual ads
IsClick	1 if there was a click on this ad.

Table 1: Field Description

Figure	No of Values	Size
Ads Info	26893298	5.35 GB
Category	68	752 Bytes
Location	4079	59 KB
User Info	4284823	104.6 GB
SearchInfo	4742881	9.47GB
Train Search stream	15961515	11.02 GB

Table 2: Count of Data Fields

6 DATA PRE-PROCESSING

We perform five steps to pre-process the data and make it more suitable to train the model. The first step is to handle missing values. Since the data is collected from a real-world use case and we perform join operations on the data, there are many rows with missing values. We handle them by removing any row which has even a single Nan value. We can afford to do this because our

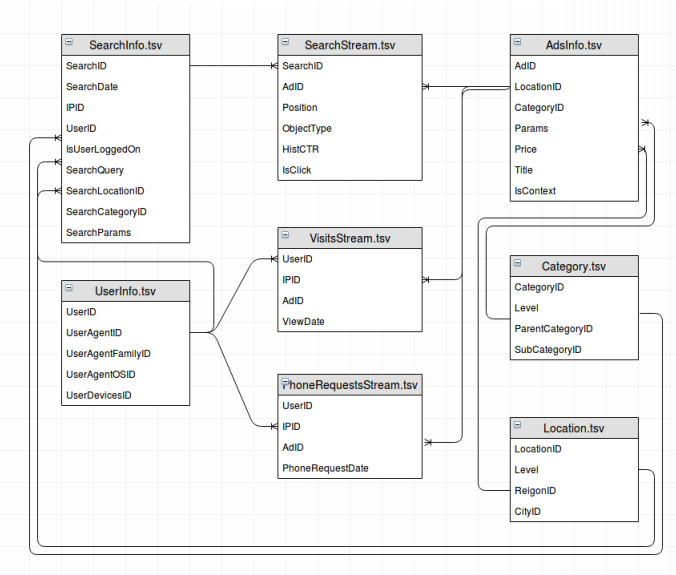


Figure 2: Schema

Category Info	
Category	Identifier of the category
Level	Level of Category for search
ParentCategoryID	identifier of parent category
Subcategory	identifier of subcategory

Table 3: Category Information

Ads Info	
AdId	Ad Identifier
LocationID	Location Identifier
CategoryID	Category Identifier
IsContext	Context Ad or Not

Table 4: Ads Information

Location Info	
LocationId	Identifier of the location
Level	Level of the search
RegionID	Identifier of the search/impression region
CityID	Identifier of the search/impression city

Table 5: Location Information

User Info	
UserId	Identifier of visitor's cookie
UserAgentID	Identifier of user's browser
UserAgentFamilyID	Identifier of user's browser family
UserAgentOSID	Identifier of user's OS
UserDeviceID	Identifier of user device type and model

Table 6: User Information

Search Info	
SearchId	Identifier of search event
SearchDate	Date and time of the search event
IPID	Identifier of visitor's IP
SearchQuery	raw query text
SearchLocationID	Identifier of the search location
SearchCategoryID	category filter of the search
SearchParams	Dictionary like features with optional filters

Table 7: Search Information

training set is large and for a dataset of that magnitude, a couple of hundred rows would not harm the model accuracy.

The second pre-processing step is feature engineering. We have a column which represents the search query entered by the user. To handle this textual data, we replace it with the length of the string. The third step of pre-processing is to scale features. We have two features which need scaling- HistCTR and SearchQueryLength. Both of these features have a bunch of outliers, and we use the Robust Scaler which is immune to outliers. Standard scaler which takes into account the mean and standard deviation would be affected by the outlier values. Robust scaling, on the other hand, considers the quantile ranges.

The fourth step in pre-processing is to binarize categorical data. All columns except the SearchQueryLength and HistCTR column are categorical columns and must be binarized. We perform one hot label encoding on them. The last step in pre-processing is to handle date-time data. We get date and time for a particular search in the training dataset. We split this information into individual

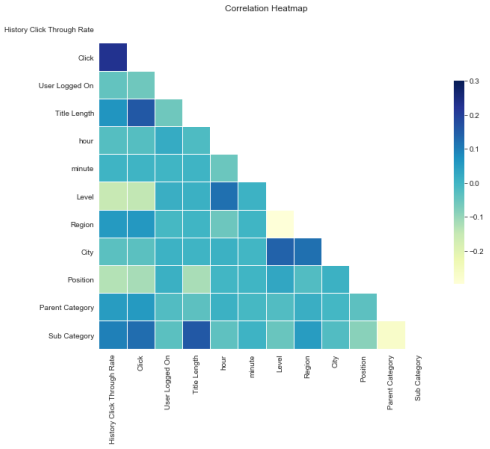


Figure 3: Correlation Matrix

columns. So month becomes one column, day becomes one column and hour of the day becomes another column. We use python's DateTime object to perform this calculation. These steps prepare the final data for data exploration and model training.

7 DATA EXPLORATION

Once the final data is ready, we can perform data exploratory analysis by plotting on data columns. We first start by analyzing the correlation between column values. For this, we create a correlation matrix and make a heat plot in. Fig 3 shows this correlation.

From the plot, we can see that there is a high correlation between historical click rate and a target attribute, between title length and target attribute and between the region and city of ad posting which is quite apparent. Next, we move on to count plots for each category colored on target attribute to see the distribution of values. Discussing the plots in figure 4, we analyze each column, in the Category column we see that more people click on ads in consumer electronic category than in the personal belonging category. In the city column, click rate is more in the cities of Novosibirsk and Yekaterinburg than Krasnodar. For the day column, we see that most ads are posted on Monday, and for ads posted on each day, the ratio of clicked to unclicked ads is almost the same. For the hour column, we observe that as the ratio of clicked ads to unclicked ads decreases after 16:00 hours and most ads are posed between 4:00 hours and 18:00 hours. For the position column, we observe that ads placed at the top of the page have a higher click rate than those placed at the bottom, which is quite intuitive. In the city column, click rate is more in the region of Vladimir Oblast and Kaliningrad Oblast than Bashkortostan. In the Sub Category column, we see that more people click on ads in others category than in clothes category. From the User Logged On column, we observe that the probability of a user clicking on an ad is more if the user is not logged in. For HistCTR and Title length columns, which are continuous variables, we plot their scaled value distribution color coded on target attribute. From the plots, we can conclude that a large number of ads have a low HistCTR, but as the value of HistCTR increases, the probability of click also increases. Title length shows a bimodal normal distribution with higher click rate on ads with longer title lengths. We then perform a KDE and a scatter plot on HistCTR and Title Length columns and observe that the KDE plot shows high density for a certain range of HistCTR and Title Length values. From the scatter plot, we corroborate our previous observation that click rate depends on long Title lengths and high HistCTR values. We also try a violin plot to see the distribution of Title Length on each day, and the results are similar to our previous observations. Next, we draw boxplots on HistCTR values for each city color coded on IsClick and the spread shows the presence of outliers even after scaling.

Now to finally, visualize the enter dataset we use t-Distributed Stochastic Neighbor Embedding to reduce the dimensions to 2-d and then use a scatter plot which is color-coded to class values and the size of the point is dependent on the HistCTR value. Figure 5 shows this plot. The plot appears to be just random points, but it helps us rule out the possibility of using a clustering algorithm for prediction. Secondly, it highlights the presence of outliers and even shows that HistCTR is very highly correlated to positive class label.

8 MODEL BUILDING AND EVALUATION

The primary machine learning task for this data is classification where the goal is to train a classifier on the training data to maximize sensitivity (recall). We focus on maximizing recall because the task is to predict a click, in such a case the other label, i.e., not a click is not significant. We use 4 classification algorithms to model the data. For each algorithm, training involves two steps. In step 1, we perform a grid search with a set of hyperparameters to select the best set of hyperparameters. Next step is to select these set of hyperparameters and train the model over 10 fold cross-validation. For evaluation, we use the precision, recall, f1, accuracy, confusion matrix, and the roc- auc.

The first algorithm we try is Logistic Regression, it is a simple algorithm and gives us an idea of how the data performs. Table 8 shows the set of hyperparameters used and the best set of parameters to maximize recall. Table 9 summarizes all the results. From table 9, we can see that although the classification accuracy is 0.61 and recall is 0.58, the model performs poorly because recall for class label Click is only 0.25. Figure 11 shows the confusion matrix for the same. Next, we move onto Ensemble models for classification. Our choice of using ensemble models depends on the fact that they are simple to use, powerful, flexible and can be interpreted using some visualization. We start with the Adaboosting algorithm, which builds a strong classifier from a collection of weak ones. Table 8 shows the different hyperparameters used and the best performing one. From table 9, we can notice that all the classifications show improvement and especially the recall for class label Click shows a 33 percent improvement. Confusion matrix shown in figure 10 gives a better idea of this change. Next, we use Random Forest Classifier which is a tree-based ensemble classifier. In simple terms, Random Forest Classifier randomly builds multiple trees and merges them to get them out. As seen in table 9, random forest produces better results than previous classifiers with increased accuracy and 10 percent increase in recall value for Click class label.

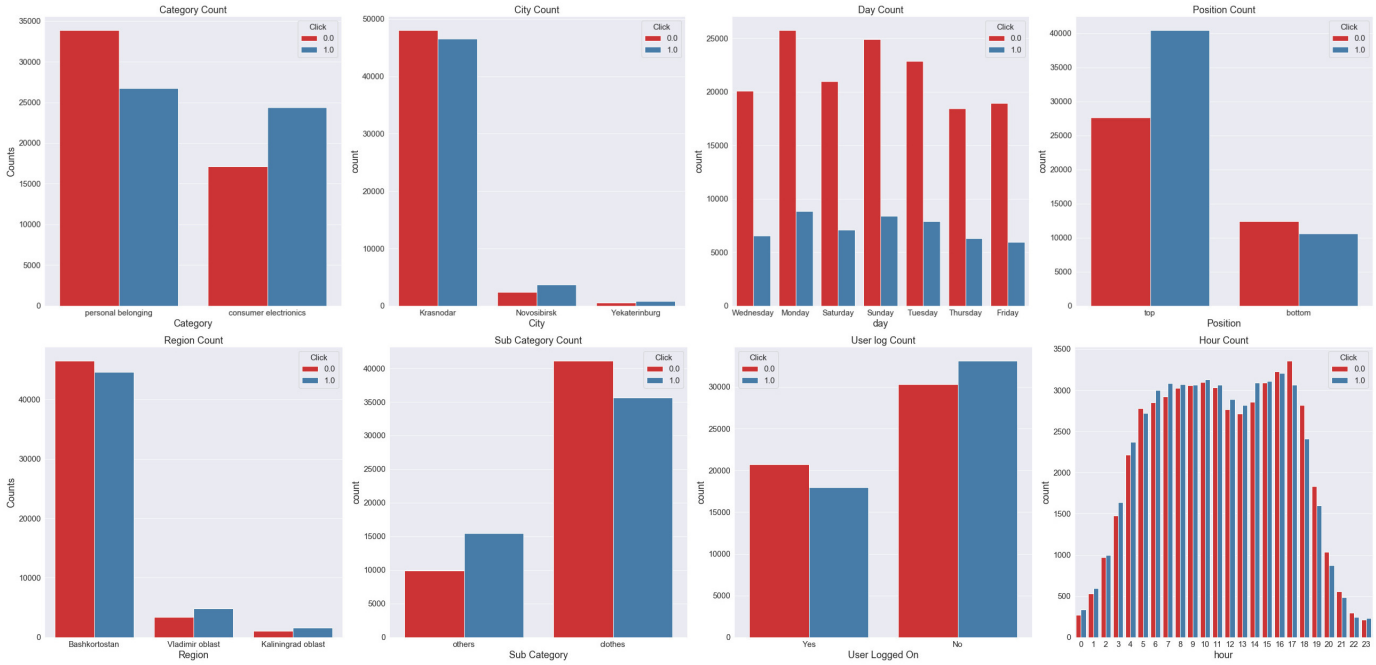


Figure 4: Count Plots for Different fields

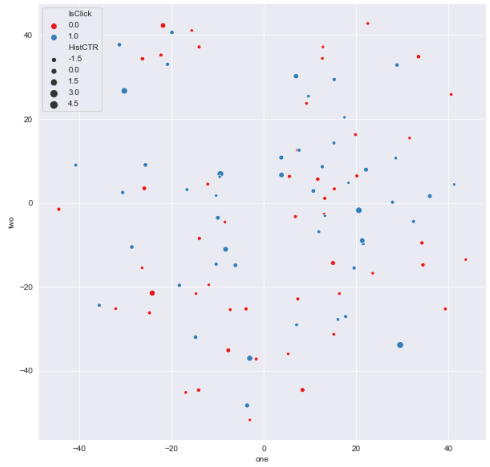


Figure 5: TSNE

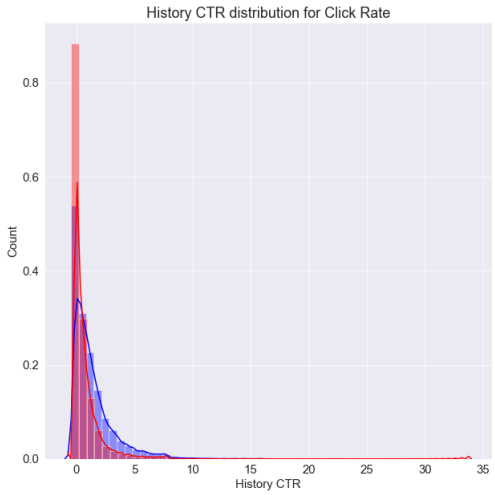


Figure 6: HistCTR for each click

Next, we move onto XGBoost classifier. It was the competition winning algorithm and has proved to be among the top results across all Kaggle competitions. XGBoost is an implementation of gradient boosting trees. To keep the report concise, we skip the technical details. The resultant XGBoost classifier after parameter tuning outperforms all other models. Although it has a lower precision and f1 score for the overall classifier, it improves on recall for Click class by 41 percent and the confusion matrix in fig 11 shows the change in results.

On closer observation of the histogram for prediction probabilities shown in figure 12, we can see that because of more instances for notClick class, the recall for Click class falls. This is a case of unbalanced classes. One solution we found was to change the classification threshold. From the histogram in figure 12, there is a normal distribution at around 0.3, and we change the classification threshold from a default of 0.5 to 0.3 and rerun evaluation metrics. Results of this change are shown in Table 9. One can observe that although the f1 score, overall recall, and accuracy goes down, there is an almost a 100 percent increase in Click class recall. Thus by changing the classification threshold, we improve the previously

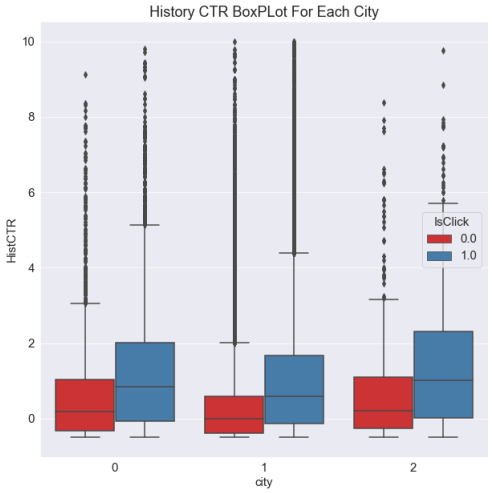


Figure 7: HistCTR for each city

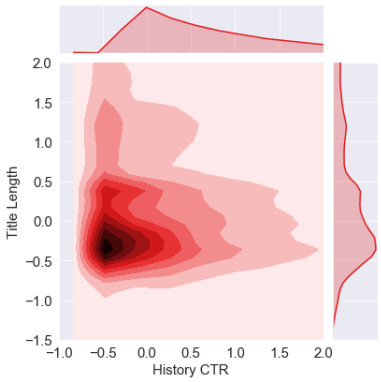


Figure 8: Title vs History

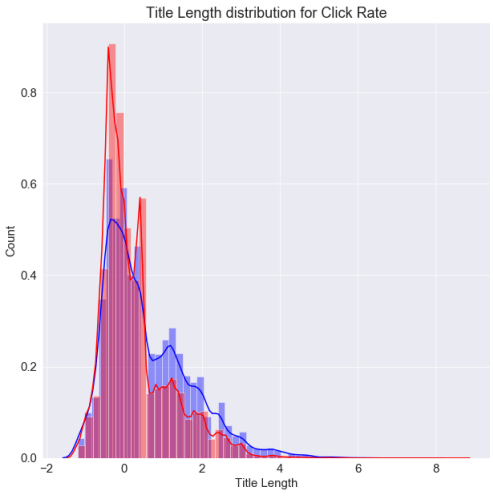


Figure 9: Title length with click

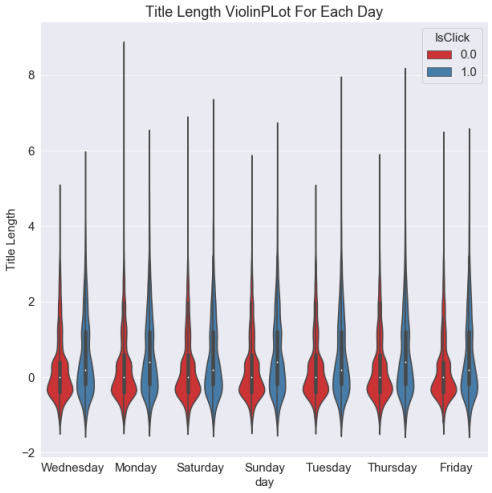


Figure 10: Title length for each day

Classifier	
Logistic Regression	"C":.001, "penalty":"l1" l1 lasso
Adaboost Classifier	AdaBoostClassifier(algorithm= 'SAMME.R', base_estimator=None, learning_rate=1, n_estimators=100, random_state=None)
Random Forest Classifier	{'n_estimators': 5, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 110, 'bootstrap': True}
XGBooost	Learning_rate =0.1, n_estimators= 140, max_depth=4, min_child_weight=6, gamma=0, subsample=0.8, colsample_bytree= 0.8, objective= 'binary:logistic', nthread=4, scale_pos_weight=1, seed=27), param_grid = param_test3, scoring= 'roc_auc',n_jobs=4,iid=False, cv=5

Table 8: Hyperparameters For each Model

overfitted model to be more general. Same is achieved for the Random Forest Classifier by changing the class weights. Hence, we are increasing the recall without giving up on much precision.

9 REAL WORLD INSIGHTS

In a digital world, online marketing is significant to increase business. However, this comes at a cost, featuring ads on top of pages can be very expensive. Our model will help the sellers in targeting specific subsets of users that are more likely to engage with the ads. This increases turn around rate for the sellers. From the buyer's perspective, they will be shown more relevant ads and increase the

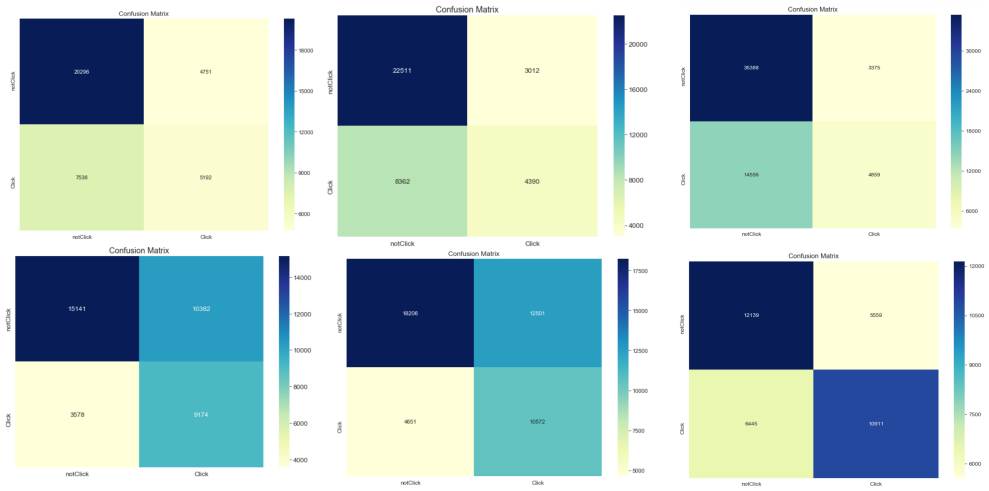


Figure 11: Confusion Matrices for each Model: Models are in the order of that in Table 9, from Top left to bottom right

	Models					
Metrics	Logistic Regression	Adaboost Classifier	Random Forest (Same Class Weight)	Random Forest (Weighted Class)	XGBoost (threshold=0.5)	XGBoost (threshold=0.3)
Recall	0.586	0.605	0.78	0.67	0.7	0.64
f1 score	0.6257	0.6438	0.75	0.66	0.68	0.65
Accuracy	0.6151	0.6451	0.77	0.69	0.7	0.635
roc_auc	0.6857	0.7089	0.57	0.6	0.613	0.656
Recall (Class=Click)	0.25	0.33	0.24	0.41	0.34	0.72
Precision	0.6386	0.6475	0.74	0.66	0.68	0.7

Table 9: Results for each Model

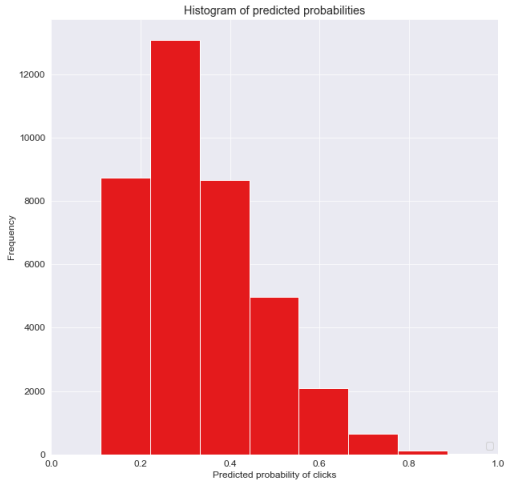


Figure 12: Histogram of Predicted Probabilities

user experience online. It will create a win-win scenario for both the buyer and the seller.

10 EXTRA CREDIT

Our dataset had many columns, and the exploration phase showed only a few columns to have a decent correlation between them. To see which column contributes more to the model training we use genetic algorithms. Genetic algorithms are designed after the theory of evolution where only the fittest survive. There are five phases in this selection process. In the first phase, we start with the entire training dataset. Then define a fitness function, which is like a scorer. Then we randomly select a subset of columns. After selection, we perform cross-over and mutation to get the most important features at the end. Using TPOT library in python, we can generate a genetic algorithm pipeline based on XGBoost Classifier to get the most relevant feature set. We set a limit of 8 to get the most essential 8 features. Figure 14 shows this importance in descending order of importance. From the figure, it is clear that the Historical Click Rate plays a massive role. i.e., ads clicked previously are more likely to be clicked in the future. Next, the hour of the day is also a significant feature. Ad title lengths play a significant role as well. Day of the week affects the Ad click too. Level of the ad, followed by subcategory of the ad, whether the user is logged on, and the position of the ad on the page play smaller roles in model decisions.

The AdsInfo table had a column which specified the title of each Ad. We could not process it because most of the text was in Russian and hence we decided to take the lengths of these titles as a column in our dataset. However, it is evident that Title plays a significant role in attracting a click. To get a more understanding of what these titles were saying. We took a subset of ad titles and created a word cloud out of it. Figure 13 shows this word cloud. One can observe many brand names like Audi, Ford, Zara, Fabia, Acer, Asus, Jeep, and BMW. There is also a presence of product names like PS4, PS3, Note, Pro, Civic, Air and R15. There also some words related to features like LTE, blue, GPS, premium, touch and new. Most of the words were in Russian, and we sought the help of Google Translate to understand them. Most frequent words were stop words and other top frequent words being "summer" given the ads were posted in April and May, "gas," "business," and "wheels," which justifies why many automobile brand names are also present. Thus, text processing reveals interesting facts about the given dataset.

Figure 13: Word Cloud

11 LESSONS LEARNED

The dataset has as a set of relational databases which were being used internally by Avito. This project helped us learn how to take noisy data collected in the backend and aggregate it to create a good training dataset and then perform analytics on it. Secondly, the scale of the data itself coupled with limited computational resources made us learn the importance of feature engineering and selecting the right data from the clutter. We learned the importance of data cleaning as even small pre-processing steps were creating massive changes in the output. We discovered many new ways to visualize the data and get an intuitive feel for the relationship between different data columns. It also helped us understand the importance of selecting the right evaluation metrics to optimize the problem. Although on the surface it looks like a classification task we learned that every classification task has a specific use case and a specific metric that helps judge the model perform. For example, although some models were giving us high accuracy, they turned

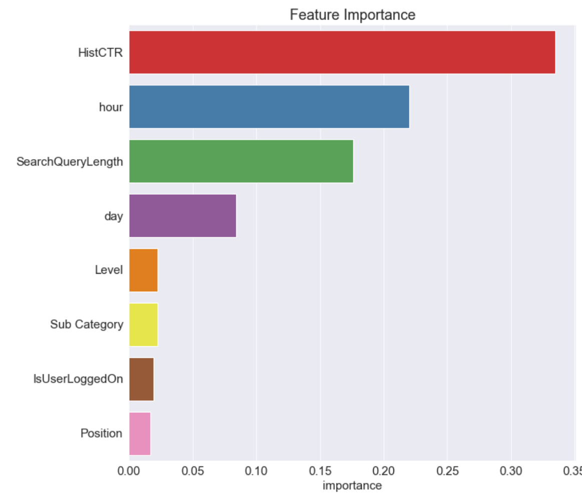


Figure 14: Feature Importance

out to overfit and were not performing that great on other class value. Moreover, lastly, we learned to handle overfit models and various statistical ways one can use to improve model performance for a specific use case.

REFERENCES

[1] McMahan, H. Brendan, et al. "Ad click prediction: a view from the trenches." *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining.*. ACM, 2013.

[2] Zhang, Yuyu, et al. *"Sequential Click Prediction for Sponsored Search with Recurrent Neural Networks." AAAI.*. Vol. 14. 2014.

[3] Cheng, Haibin, and Erick CantÃž-Paz. *"Personalized click prediction in sponsored search." Proceedings of the third ACM international conference on Web search and data mining.*. ACM,2010

[4] Shaparenko, Benyah, ÃŰzgÃijr ÃĞetin, and Rukmini Iyer. "Data-driven text features for sponsored search click prediction. *Proceedings of the Third International Workshop on Data Mining and Audience Intelligence for Advertising.*. ACM,2009.

[5] Dave, Kushal S., and Vasudeva Varma. "Learning the click-through rate for rare/new ads from similar ads. *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval.* . ACM,2010.

[6] Liu, Qiang, et al. "A convolutional click prediction model" *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management.*. ACM,2015.

[7] Richardson, Matthew, Ewa Dominowska, and Robert Ragno."Predicting clicks: estimating the click-through rate for new ads." *Proceedings of the 16th international conference on World Wide Web.*. ACM,2007.

[8] Hillard, Dustin, et al. "Improving ad relevance in sponsored search." *Proceedings of the third ACM international conference on Web search and data mining.*. ACM,2010.

[9] Trofimov, Ilya, Anna Kornetova, and Valery Topinskiy. "Using boosted trees for click-through rate prediction for sponsored search." *Proceedings of the Sixth International Workshop on Data Mining for Online Advertising and Internet Economy.*. ACM,2012.

[10] Sculley, D., et al. "Predicting bounce rates in sponsored search advertisements." *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining.*. ACM,2009.

[11] Chen, Ye, et al. "Large-scale behavioral targeting for advertising over a network." U.S. Patent No. 8,150,723. . 3 Apr. 2012.

[12] Guo, Huifeng, et al. "Deepfm: a factorization-machine based neural network for ctr prediction." *arXiv preprint arXiv:1703.04247.* 2017.

Avito Ad Click Prediction

Group members

Parth Vora

Mohak Bheda

Maanav Mehrotra

Problem Statement

- Avito is the largest online marketplace in Russia connecting sellers and customers.
- Provides three types of ads to increase sales – regular, highlighted, context (pay/click).
- Traditionally, ads were placed on the page using standard statistical formulae.
- To update its system, it has given data to solve the following problem.

"Given the Avito data which includes user and ad data, predict whether a particular ad will be clicked on"

Data Description

- Dataset consists of 8 relational tables (as shown in schema Figure 1.), one training dataset and one test dataset.
- Based on ids in training table, we combine these tables into a final training dataset which will be used to train the model.
- Final data consists of 15961515 rows and 25 columns.

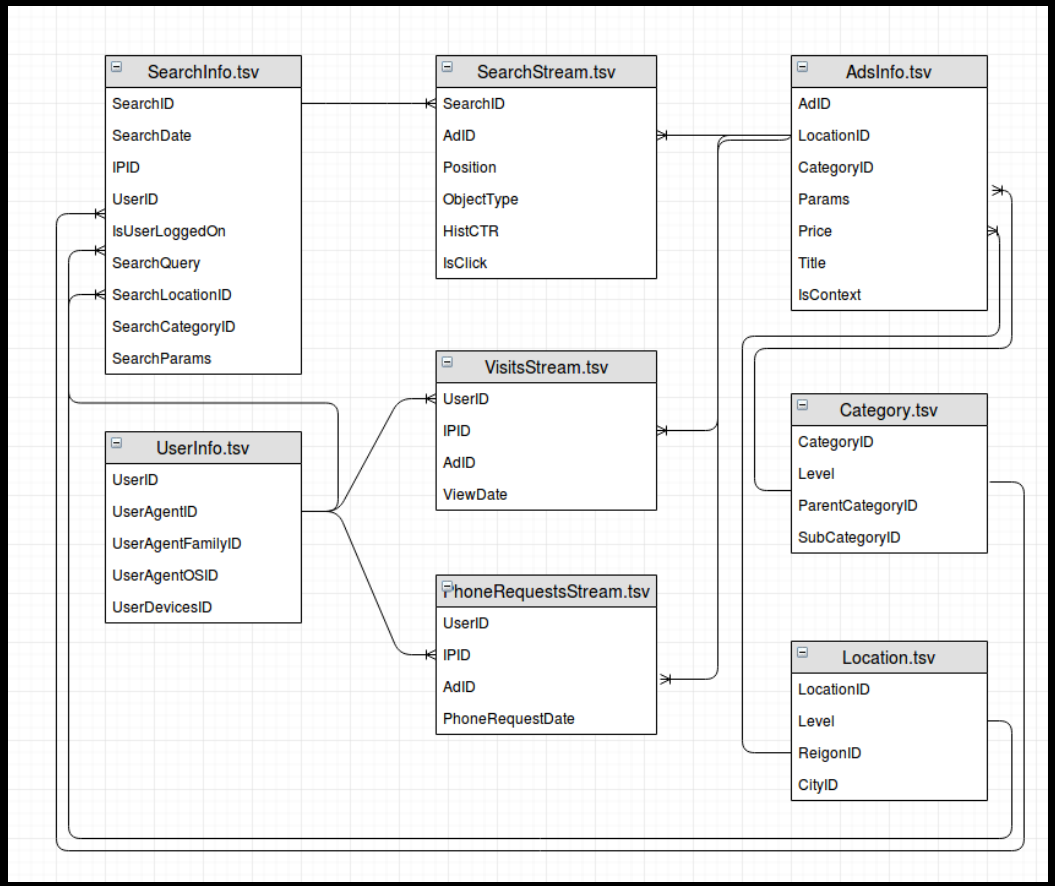


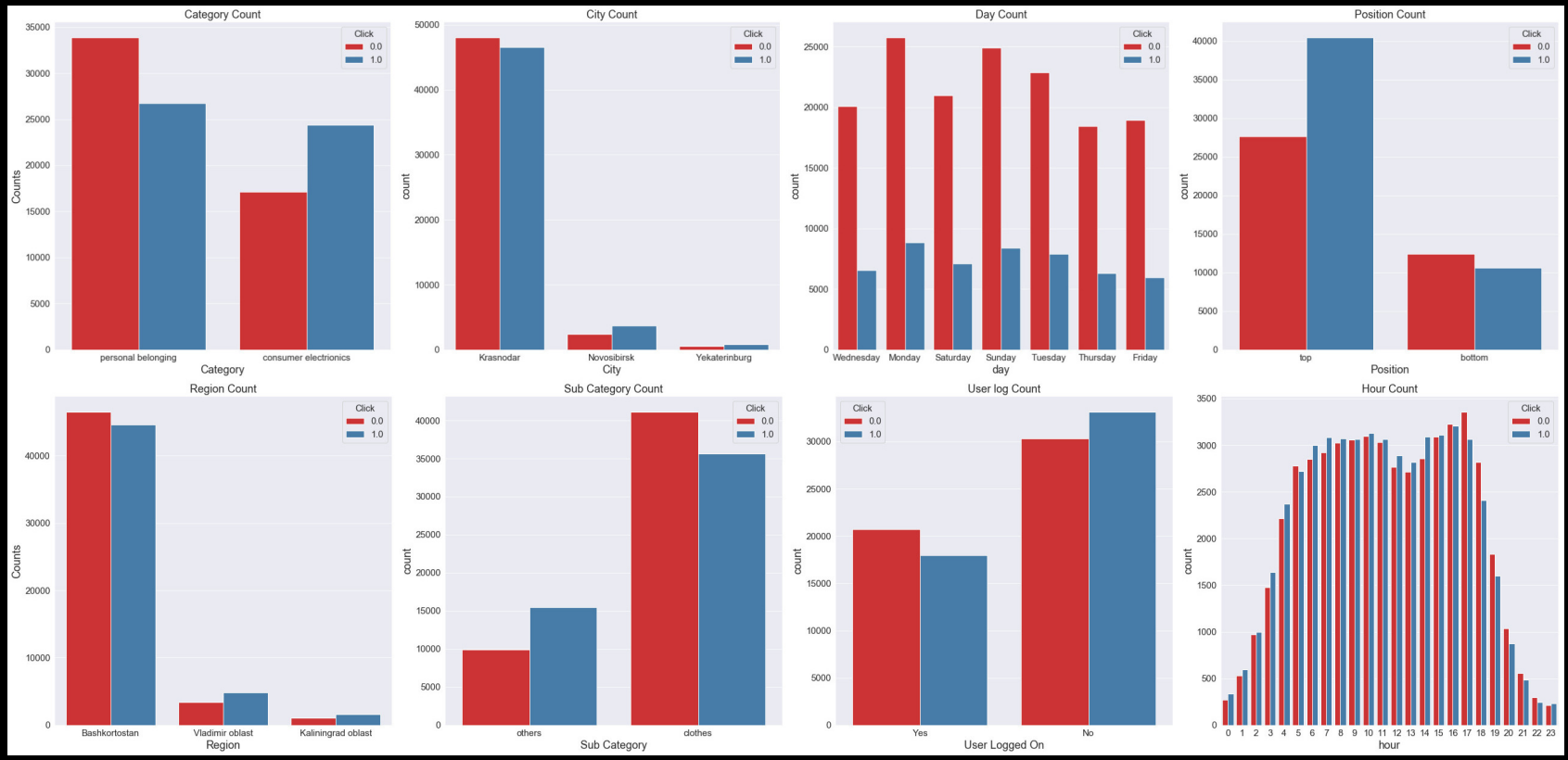
Figure 1. Schema for relational database

Data Preprocessing

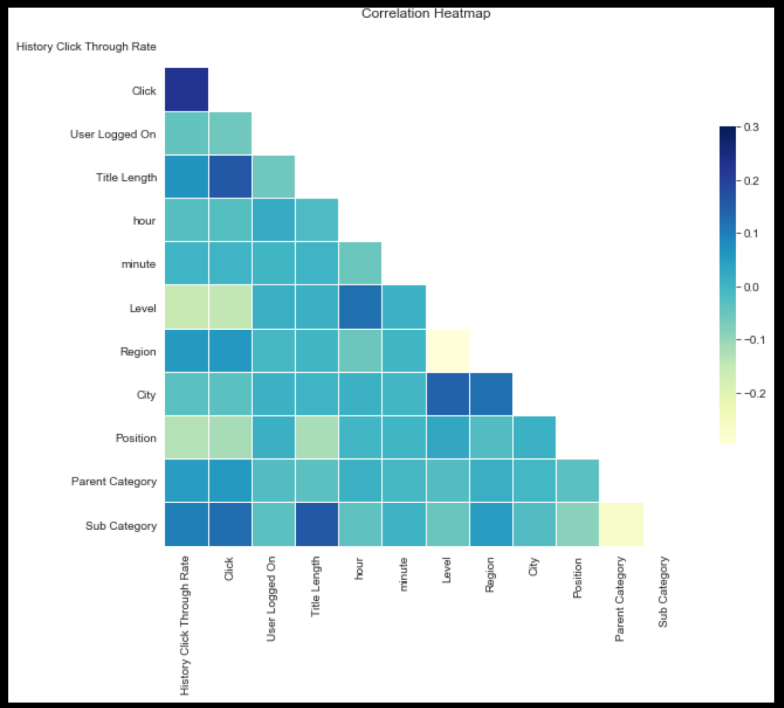
- There are 2 numerical columns and 23 categorical columns.
- Analysis shows presence of outliers in both the numerical columns.
- Perform Robust scaling using quartiles on these columns.
- Split date-time columns into into individual columns.
- One hot encode categorical columns.

Data exploration

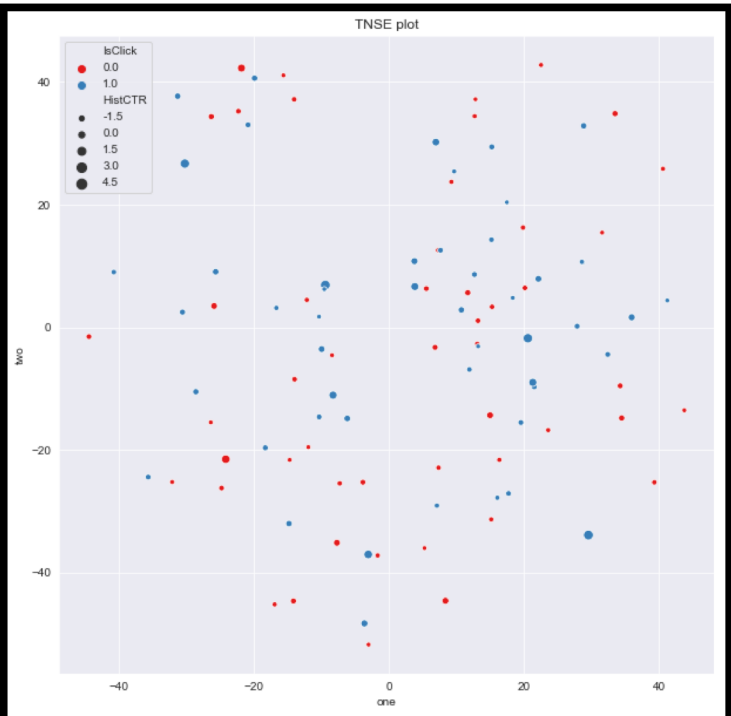
- Count plot on each column attribute color coded to class values.



- Correlation heatmap to visualize attribute dependence.



- t-SNE dimensionality reduction plot color coded on class attribute with point size coded to HistCTR value.



Model Building

- Models used Logistic regression, Adabooster, Random Forest Classifier, XGBoost.
- Steps followed
  - Perform grid search to get the best set of hyper parameters to maximize recall
  - Perform cross validation on the model with best set of hyper parameters.
  - Evalute model.

Table for results

Models	Logistic Regression	Adabooster	Random Forest (same class weights)	Random Forest (weighted classes)	XGBoost (threshold = 0.5)	XGBoost (threshold = 0.3)
precision	0.6386	0.6475	0.74	0.66	0.68	0.7
recall	0.586	0.605	0.78	0.67	0.7	0.64
f1	0.6257	0.6438	0.75	0.66	0.68	0.65
accuracy	0.6151	0.6451	0.77	0.69	0.7	0.635
roc_auc	0.6857	0.7089	0.57	0.6	0.613	0.656
recall (class = Click)	0.25	0.33	0.24	**0.41**	0.34	**0.72**

Real-world insights

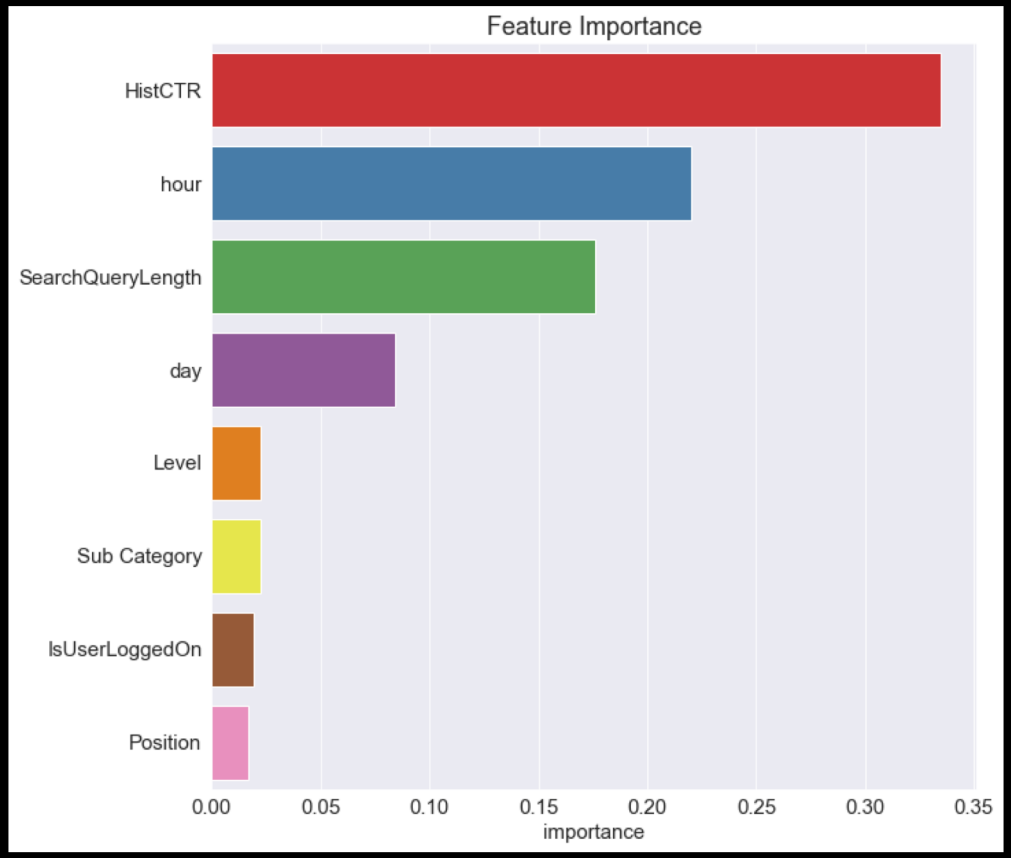
- Advertising can be very expensive.
- Ability to predict and optimize ad click can help sellers put their money in the right place.
- Consumers get to see content which is relevant to them and not spam.
- Helps increase site engagement

Lessons Learned

- Handling big data can be tricky.
- Visualization helps optimize model building and gives an intuitive sense of data.
- Accuracy is not everything.
- Selecting right evaluation metrics and right hyper parameters can significantly improve the model.
- Small statistical hacks can improve model performance like changing the classification threshold.

Extra Credit

- Genetic algorithms to find out most important features.



- Text analysis on words present in title of each ad

Translation

Russian	English
из	of
от	from
бу	Boo
до	before
лето	summer
месяц	month
газ	gas
колеса	wheels
бизнес	business