**MIS 620 – Project Report**



Members:

Gaurav Gade

Harsh Bangad

Siddhi Shah

Suhas Mitra

**Executive Summary**

**Overview:** Avito.ru is the largest general classified website in Russia where buyers and sellers across the world are connected. Items range from automobiles, properties, jobs and services. There are 70 million unique visitors on the website. Presently, Avito uses generic statistics about the ads to determine the ranking of the ads on its page. It doesn’t consider user behavior which is a huge factor nowadays to predict the most relevant ad for a particular user.

**Outcomes: Quantitative and business value:** We have selected to use the logistic regression model having accuracy rate of 0.8549, The model confusion matrix metrics are Sensitivity: 0.858227, Specificity: 0.225681. This will improve the CTR per user, increase the cost-per-click resulting in higher revenue for sellers/Advertisers. For a click through rate of 0.8%, the cost per click charged by Avito from advertisers is $1.58 as of 2015,

Given that Avito has 70 million customers, the advertisement revenue currently stands at 70 Million \* 0.8/100 \* $1.58 = $0.885 Million. Using our model the CTR will increase by 35% (the benchmark we have considered in 50%). CTR will improve to 1.08. The new revenue will be 70 Million \* 1.08/100 \* $1.58 = $1.19 Million. That is an increase in revenue by $.30 Million. The increase in revenue is therefore 34.46%.

In addition since advertisers will get higher conversion rates based on clicks, Avito can increase the Cost per click by 10%. This means they can charge advertisers $1.73 per click. This will translate to a revenue of 70 million \* 1.08/100 \* $1.73 = $1.307 Million. The net revenue from advertisement will increase by 47.68% by implementing this predictive model

|  |  |
| --- | --- |
| Cost per click(CPC) | $1.58 |
| Click through rate(CTR) | 0.8% |
| Source: hotchmanconsultants.com |  |

**Note:** If we choose the decision tree model then the accuracy rate is 0.7589, Sensitivity: 0.76036 and Specificity: 0.47860.

**Data discovery:** The data was in SQLite format. The size of dataset was 40 GB having 80 million records. The ER\_model was provided, but we had to combine the data using database joins based on primary and foreign key constraints. We selected 400 thousand records in our final table. We converted the sqlite records into internal R-memory format.

**Data preparation**: We removed non-contextual ads leaving us with 260,000 records in the final table. We eliminated fields that were extremely sparse values and those with more than 99% missing values coded as NA using subset() function. We checked missing values vs. observed values using mismap() function. We created dummy variables by using ‘as factor’ function. To check for sampling errors, we ran the model using 4M records as well. It took 15 minutes to run. IsClick = 1 was still unbalanced and had similar overall proportion like 400,000 records. We dropped those columns that do not affect “IsClick” like random ID and search parameters thus using feature reduction methods to keep only 6 relevant predictor variables. tTo overcome the problem of unbalanced data, we used smote function.

**Model Planning**: Based on our hypothesis and the relationship between the tables, our main focus was on investigating what variables influence IsClick = 1. We identified 6 variables that affect IsClick: Position, HistCTR, IsCLick, IsUserLoggedOn, Category\_Level and Location\_Level. All the variables except HistCTR are categorical/factor variables. Hence, we considered classification models. Based on our data structure, we planned to build 4 models. These models are: logistic regression, random forest, decision tree and naïve bayes. By using multiple models we could reasonably mitigate the effect of bias, outliers or missing data that could affect a single model. To meet our business objectives, we decided to consider False positive rate and accuracy while selecting the best model. As isClick = 1 was relatively sparse, we did not want to select a model that accurately predicts isClick = 0 but fails to accurately predict positive cases.

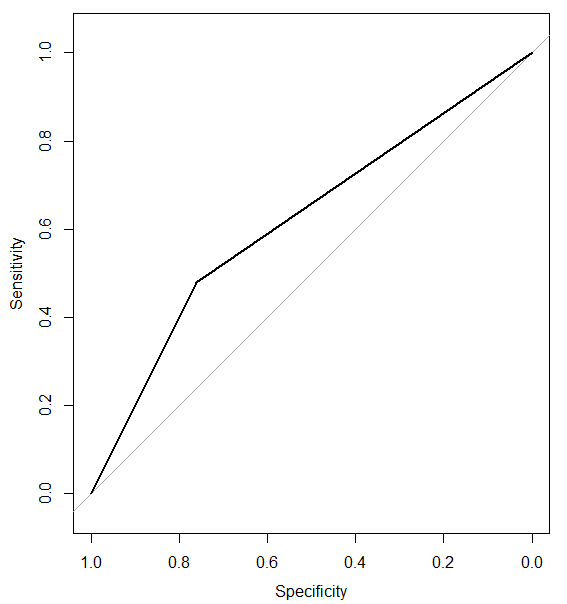
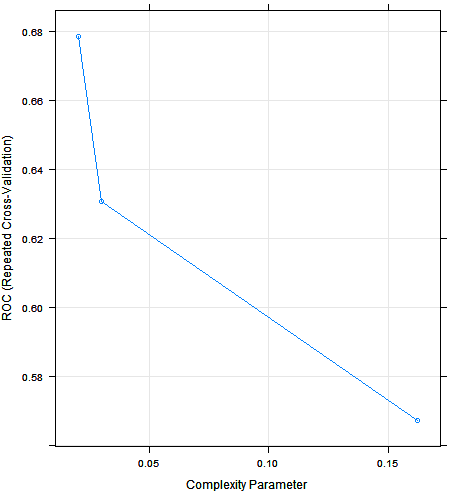
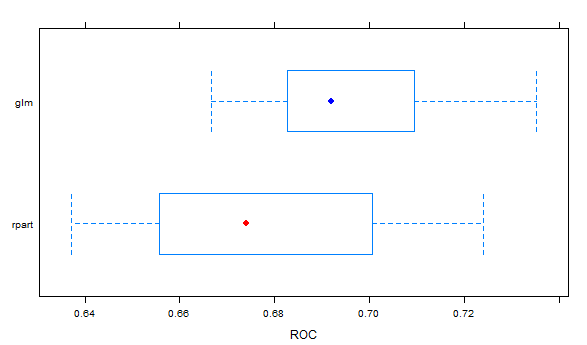
**Model Building**: We created a 80/20 partition using createDataPartition function. We used SMOTE method to adjust for sampling. The training data was over sampled by 400% for positive cases where as the negative cases were undersampled by 150%. We controlled the sampling during parameter tuning and testing. We ran the summaryFunction to measure the performance of our model. The twoClassSummary is a built in function that we utilized for ROC, specificity and sensitivity. For Decision tree, we used modelLookup= rpart. For Logistic regression we used modelLookup = glm, metric = ROC. The Random Forest used modelLookup= rf while the naïve bayes was run on data pre-processed using smote and laplace smoothing. We measured the performance of model using ROC as the metric. We built the confusion matrix for each model, and used the plot function to visualize the ROC curve. To compare between logistic and decision tree we also built a box plot. We supplied several arguments to the model to optimize them as and when required. The results were recorded and we saved screenshots for each model to consolidate and compare later on. We also looked at AUC (area under curve) to check how much variance of the data can be explained by the model.

**Performance**: Based on the analysis for 4 models logistic regression performed better accuracy wise by giving an accuracy of 0.8549. If we want to check how many users actually clicked on the contextual ad than the we can use decision tree model as it predicted highest number of user clicks.

**Key Recommendation:** Avito should use our predictive model that will help them predict whether their users will click on an ad given that is contextual in order to increase their ad revenue by improving the CTR and charging a higher CPC from advertisers.

Supporting message: We have combined user searches, advertisement information, ad category and user location to increase Avito’s predictive capabilities that currently relies only on Historic CTR (HistCTR). This will help them to identify users that are likely to click on a contextual ad as well as perform customer segmentation that will improve Avito’s marketing efforts in the future.

**Key visualizations:**



Final Conclusion: Based on the four models, we found that logistic regression gave the best performance based on the combination of predicting True positives, false positive rate and accuracy. If more proportion of True Positives is required, then we can also go for decision tree model based on the business scenario we are trying to solve.

**Data Discovery Phase:**

Initially we started with the Energy efficiency data set which consisted of 8 variables (relative compactness, surface area, wall area, roof area, overall height, orientation, glazing area, and glazing area distribution) on two output variables, namely heating load (HL) and cooling load of residential buildings.

Problem Statement for Energy Efficiency: Energy consumption and savings have become very important issues lately given global warming and adverse effects on climate. Reports suggest that building energy consumption has steadily increased over the past decades worldwide for heating, ventilation and air conditioning (HVAC). One way to alleviate the ever increasing demand for additional energy supply is to have more energy-efficient building designs with improved energy conservation properties.

The problem faced by us with this dataset was that the data set was too small and we couldn’t find any other interesting data set with which we could combine our data set.

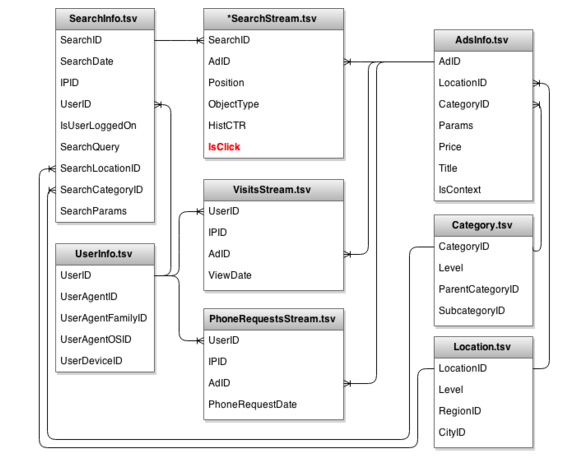
Thus while looking for other dataset we found the Avito Dataset which was interesting and challenging.

Problem statement of Avito: Given the historical user behavior captured in the variable HistCTR, we intend to design a model that will accurately predict if the user will click on an ad given that it is contextual.

**Data description:**

This competition has 8 relational datasets. All these files were encoded in UTF-8 and stored into tab separated format (.tsv). There is also a SQLite database (database.sqlite) alternative with all data available. Relationships between the datasets are captured in the following schema:

Figure 1: Entity-Relationship diagram for the data



trainSearchStream.tsv, testSearchStream.tsv are two main datasets related to our predictive models.

trainSearchStream is a random sample of previously selected users' searches on avito.ru during at least 16 consecutive days from April'25 until the target impression. Different types of ads on the site are shown in the picture below:

|  |
| --- |
|  |
| Figure 2: Three types of advertisement placed on Avito.ru |

Regular ads are shifted down constantly as new ads come in. (Normally, a visitor's search results are sorted by the time an ad is submitted to Avito). Each line in the file describes one "impression" (an ad that is shown to a particular user based on a search). testSearchStream shares the same format, except the target variable field "IsClick" is omitted.

* Position - position of the ad in search result page (1 - is first ad on a page starting from the top). Only ads on position 1, 2, 6, 7, and 8 are logged.
* Object Type - 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, calculated when the ad is showed. For non-contextual ads this field equals NULL.
* IsClick - 1 if there was a click on this ad. Otherwise 0. For non-contextual ads this field equals NULL. The goal of this competition is to make a click prediction model for contextual ads.
* IsUserLoggedOn - whether user was logged on with his/hers login (1) or not (0).

**Data Splitting**

The dataset contains a sample of users and their behavior. For each user, one target impression between time point A (May, 12) and time point B (May, 20) was selected randomly. The task of is to provide the probability that a user will click on the selected target ad, given all the information generated by the user from the Start time point until the target impression.

|  |
| --- |
|  |

**Hypothesis:**

H0: Null Hypothesis: Is Click attribute is only dependent on the HistCTR

Ha: Alternative Hypothesis: Is Click attribute is not dependent only on HistCTR

**Data Preparation the database:**

We used a SQLite file that contained the complete data but was not combined according to the relational model. Hence, we had to first understand the database structure, identify how to join the tables, the fields that we need for the model and creating the final view. The problem we faced was that the SQLite file after extraction was in excess of 40 GB. Therefore, there were more than 80 million records. To pre-process this data, we tried several methods and studied online material extensively. Also, we decided to convert this in a SQL friendly format since we had good proficiency in SQL and had studied the queries in the MIS 686 course. The way we achieved that was by using the package sqldf. In addition, instead of using the SQLite format, we used our Professor’s code to first extract the data into internal R-memory format. We created a data frame and saved 400,000 records in it.

We used the dbConnect function using the RSQLite driver to connect to the database. We used the fetch command to get the data from the database and saved this in a data frame. We then selected up to 400 thousand records from all the tables

We created an entire hierarchy for the model, identified the primary and foreign keys and created temporary tables to start join the tables and successively merging the tables into a single unified view.

The process that we followed was a bottom-top approach. We selected the smallest table and started merging it with its associated tables by identifying the foreign-primary key relationships. By using this divide and conquer approach we managed to join all 8 tables in a single view. This phase of our project took the most amount of time and we had to spend hours until we managed to extract all the data into a single data. The steps we followed are as follows:

1. We joined AdsInfo with Category on basis of categoryID. We used left join and selected Category\_Level from Category table and saved it in temp1.
2. We joined SearchInfo with Category on basis of categoryID. We used left join and selected Category\_Level from Category table and saved it in temp2.
3. We joined temp1 with Location on basis of locationID. We used left join and selected Location\_Level from Location table and saved it in temp3.
4. We joined temp2 with Location on basis of locationID. We used left join and selected Location\_Level from Location table and saved it in temp 4.
5. We joined temp3 with PhoneRequestsStream on basis of AdID. We used left join and selected IPID and PR\_ID from PhoneRequestsStream table and saved it in temp 5.
6. We joined UserInfo with PhoneRequestsStream on basis of UserID. We used left join and selected IPID and PR\_ID from PhoneRequestsStream table and saved it in temp 6
7. We joined temp6 with VisitsStream on basis of UserID. We used left join and selected IPID and ViewDate from VisitsStream table and saved it in temp 7
8. We joined temp5 with VisitsStream on basis of AdID. We used left join and selected IPID and ViewDate from VisitsStream table and saved it in temp 8.
9. We joined temp4 with temp7 on UserID. We used left join and saved it in temp 9.
10. We joined trainSearchStream with temp9 on basis of SearchID. We used left join and selected SearchDate, UserID, IsUserLoggedOn, SearchQuery, LocationID, CategoryID, SearchParams, Category\_Level, Location\_Level from temp9 and saved it in temp 10.
11. We removed that data where objectType != 3. (Eliminated non-contextual ads)
12. We joined temp10 with temp8 on basis of AdID. We used left join and selected Price, IsContext from temp8 and saved it in final.

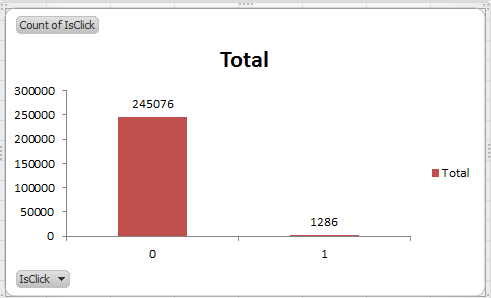
**Data Cleaning:**

1. After we removed non-contextual ads, we were left with 260,000 records.
2. We dropped those columns that were not affecting IsClick like random ID and search parameters, etc.
3. We were left with 6 variables that were Position ,HistCTR, IsCLick, IsUserLoggedOn, Category\_Level and Location\_Level.
4. We eliminated those fields with very few values but with more than 99% missing values coded as NA.
5. We used subset () function to drop the irrelevant fields.
6. We used miss map command to check Missing values vs observed values to help us decide whether we can drop a particular field or not.
7. We created dummy variables by using as factor functions so that we could use those variables in our model.

**Model Planning:**

Once the data was joined and primed accordingly the first step was to partition the data in 80/20 ratio between Training Data and Test Data.

The first step was to look the sparseness of the “IsClick” event. On further analysis we found that the IsClick is indeed very rare. The following figure illustrates that the occurrence of the IsClick = 1 event happens only in 0.522% of the records.

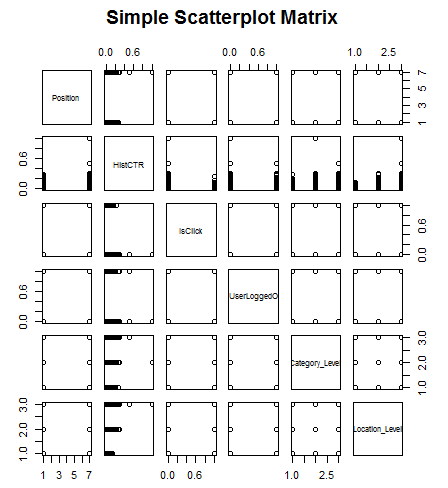


The performance of any machine learning algorithm that we would undertake would be severely impacted if we did not address this issue. After all a naive chance model (with zero intelligence) would be right 99.47% of the time. On further research we found that there are many techniques that academicians have used to solve this problem.

The machine learning community has addressed the issue of class imbalance in two ways. One is to assign distinct costs to training examples. The other is to re-sample the original dataset, either by over-sampling the minority class and/or under-sampling the majority class.

This gave us pointers that we need to balance the number of IsClick = 0 events with respect to IsClick = 1 event. For this we used SMOTE mechanism i.e. Synthetic Minority Over-sampling Technique. SMOTE blends under-sampling of the majority class with a special form of over-sampling the minority class. From a business perspective inability to predict IsClick =1 is a potential loss. This is the basis of being able to predict if Context Ads need to be sorted and displayed to a user with highest chance of a user clicking it. Note that SMOTE is applied only on Training data and not on Test data.

The next step was to look at a Scatter Plot Matrix and to identify if there were any outliers, inconsistent data and visually see if there was any relationship predictors and outcome variable.



As shown in the figure above the IsClick event does not have any linear relationship with other predictors and all the predictor variables are categorical in nature except HistCTR which is ordinal data.

**Model Building:**

Since the predictor variable is a 2 stage factor/categorical variable we can apply various Machine Learning Techniques such as:-

* Decision Tree
* Random Forest
* Naïve Bayes
* Logistic Regression

For our model simulation in R using RStudio we used open source packages e.g.

* caret
* kernlab
* doParallel
* pROC
* ggplot2
* pROC
* DMwR

The above toolkit was used with Cross Validation techniques to test against validation data.

The entire R code is supplied in Appendix A.

**Decision Tree:-**

Package “caret” with a model lookup of “rpart” is used to create these models.

* We use 3 repeats of 10–fold cross–validation for our training model.
  + This serves to bootstrap our model.
* However, to calculate the ROCcurve, we need the model to predict the class probabilities.
  + The classProbs option is used.
* Finally, we tell the function to optimize the area under the ROC curve
  + We use the metric argument to specify the “ROC” option
* The data passed to train the model is the SMOTE data
  + The under/over sampling ratio is adjusted to create a more equitable ratio of samples with IsClick =1 and IsClick =0.
* This data is then scale from 0 to 1 and from columns with zero variance
  + The preProc option is used

Using predict function we obtain the predicted outcomes for the training data and then compute the confusion matrix. Using pROC we plot the ROC curve and obtain the Area under the curve.

**Random Forest**

Package “caret” with a model lookup of “randomForest” is used to create these models.

* We use 3 repeats of 10–fold cross–validation for our training model.
  + This serves to bootstrap our model.
* However, to calculate the ROCcurve, we need the model to predict the class probabilities.
  + The classProbs option is used.
* Finally, we tell the function to optimize the area under the ROC curve
  + We use the metric argument to specify the “ROC” option
* The data passed to train the model is the SMOTE data
  + The under/over sampling ratio is adjusted to create a more equitable ratio of samples with IsClick =1 and IsClick =0.
* This data is then scale from 0 to 1 and from columns with zero variance
  + The preProc option is used

Using predict function we obtain the predicted outcomes for the training data and then compute the confusion matrix. Using pROC we plot the ROC curve and obtain the Area under the curve.

**Naïve Bayes: -** Naïve Bayes classifier is used along with Laplace smoothening. We used Smote training data and set the smoothening factor as 0.01.

To display the model statistics e used predict method. Confusion Matrix was used to evaluate the performance.

We used the ROC metric for accuracy.

**Logistic Regression:-**

Package “caret” with a model lookup of “glm” is used to create these models.

* We use 3 repeats of 10–fold cross–validation for our training model.
  + This serves to bootstrap our model.
* However, to calculate the ROC, we need the model to predict the class probabilities.
  + The classProbs option is used.
* Finally, we tell the function to optimize the area under the ROC curve
  + We use the metric argument to specify the “ROC” option
* The data passed to train the model is the SMOTE data
  + The under/over sampling ratio is adjusted to create a more equitable ratio of samples with IsClick =1 and IsClick =0.
* This data is then scale from 0 to 1 and from columns with zero variance
  + The preProc option is used

Using predict function we obtain the predicted outcomes for the training data and then compute the confusion matrix. Using pROC we plot the ROC curve and obtain the Area under the curve.

**Results and Performance:**

Thus 4 models were used for to predict whether the user is clicking the contextual ad or not .The results and performance of the 4 models are discussed below

1. **Decision Tree**

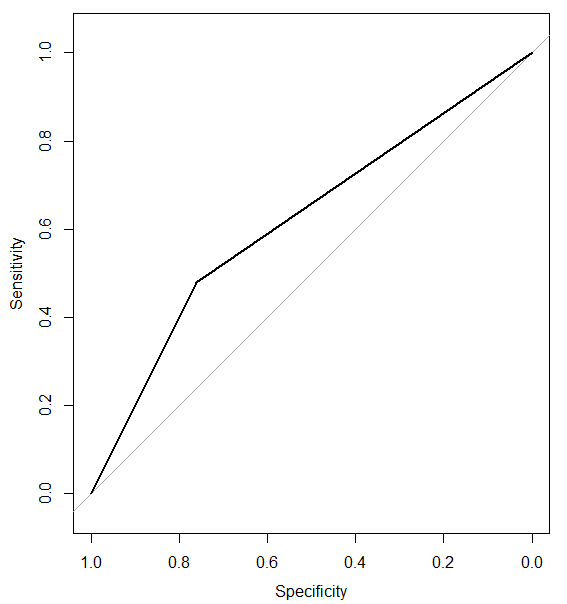
**Confusion Matrix**

|  |  |  |
| --- | --- | --- |
| **Prediction** | **No** | **Yes** |
| **No** | 37269 | 134 |
| **Yes** | 11746 | 123 |

As per the above confusion Matrix the model predicted 37269 true negatives and 123 true positives.

**Accuracy, Sensitivity and Specificity**

|  |  |
| --- | --- |
| **Accuracy** | 0.7589 |
| **Sensitivity** | 0.76036 |
| **Specificity** | 0.47860 |



The Area under the curve for this model is **0.6195.**

.

1. **Decision Tree using Random Forest**

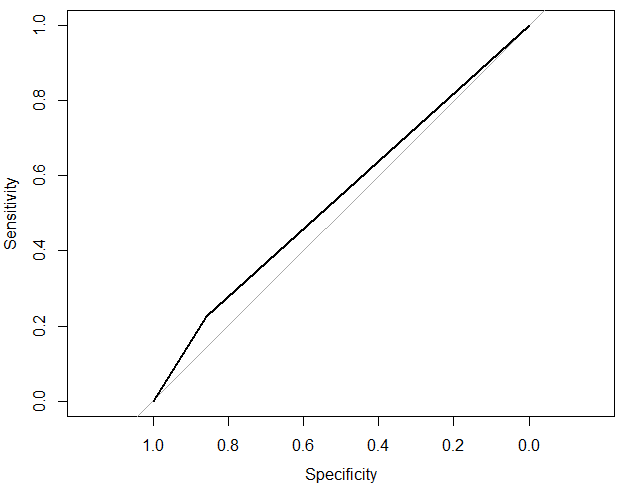
**Confusion Matrix:**

|  |  |  |
| --- | --- | --- |
| **Prediction** | **No** | **Yes** |
| **No** | 38550 | 159 |
| **Yes** | 10465 | 98 |

As per the above confusion Matrix the model predicted 38550 true negatives and 98 true positives.

**Accuracy, Sensitivity and Specificity**

|  |  |
| --- | --- |
| **Accuracy** | 0.7844 |
| **Sensitivity** | 0.786494 |
| **Specificity** | 0.381323 |



The Area Under the curve for this model is **0.5839.**

1. **Naïve Bayes**

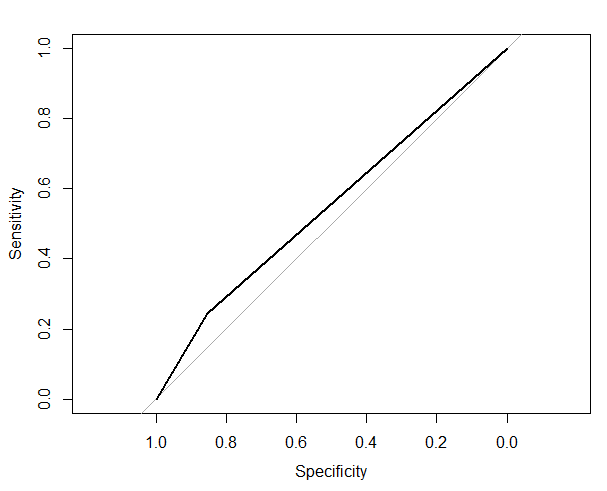
**Confusion Matrix**

|  |  |  |
| --- | --- | --- |
| **Prediction** | **No** | **Yes** |
| **No** | 41804 | 194 |
| **Yes** | 7211 | 63 |

As per the above confusion Matrix the model predicted 41804 true negatives and 63 true positives.

**Accuracy, Sensitivity and Specificity**

|  |  |
| --- | --- |
| **Accuracy** | 0.8497 |
| **Sensitivity** | 0.852882 |
| **Specificity** | 0.245136 |



The Area under the curve for this model is **0.549**

1. **Logistic Regression**

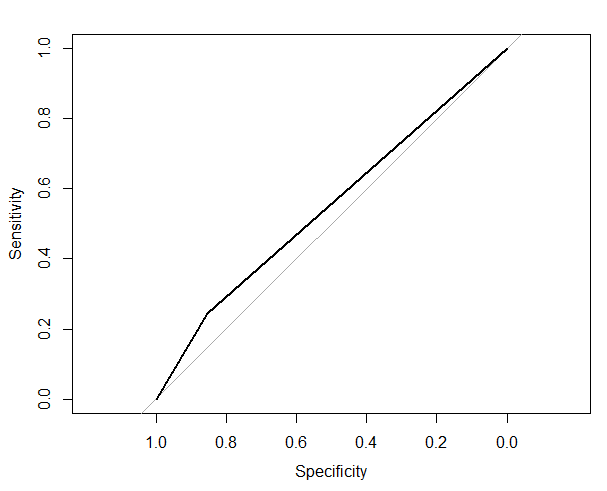
**Confusion Matrix**

|  |  |  |
| --- | --- | --- |
| **Prediction** | **No** | **Yes** |
| **No** | 42066 | 199 |
| **Yes** | 6949 | 58 |

As per the above confusion Matrix the model predicted 42066 true negatives and 58 true positives.

**Accuracy, Sensitivity and Specificity**

|  |  |
| --- | --- |
| **Accuracy** | 0.8549 |
| **Sensitivity** | 0.858227 |
| **Specificity** | 0.22568 |



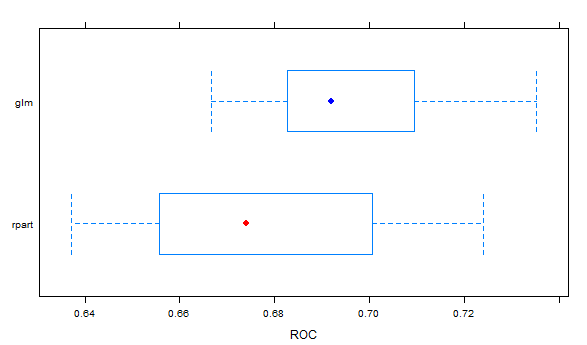
The Area under the curve for this model is **0.549**

The cross fold validation was performed for Decision Tree and Logistic Regression Model

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **ROC** | **Sensitivity** | **Specificity** |
| Decision Tree | 0.6784654 | 0.7426383 | 0.5407937 |
| Logistic Regression | 0.6961072 | 0.8526644 | 0.4115343 |

As per the cross validation analysis the ROC of Logistic regression is better as compared to Decision Tree.

**Box plot comparing Decision Tree and Logistic Regression**



As per the box plot comparison it is clearly illustrated that ROC for Logistic Regression Model is better than Decision Tree.

**Analysis from all 4 models:**

1. If we want to check the maximum number of the user clicks correctly then we will choose decision tree model which has accuracy of 0.7589.
2. If we want to select a model based on accuracy then we will select logistic regression model which has accuracy of 0.8549.
3. The area under the curve was highest for Decision Tree.
4. Thus as per the model analysis IsClick attribute is dependent on all the variables and not only on HistCTR and thus we are rejecting null hypothesis.

**Discussions and Recommendations**

Since Avito customers are accessing this website for online sales, the business revenue model for Avito is based on

* **Cost-per-click**, where the advertisers pay a certain amount for every click.
* **Click-through rate (CTR),** defined as a ratio of number of clicks to number of impressions.

Where the generated **Revenue = CTR \* CPC**

Based on the above predicting CTR for a user is critical to have higher revenue. CTR is one of the most important factors in determining what advertisements need to be displayed and in what order.

In addition, advertisers often plan their budget based on historical CTRs and/or predicted CTRs. Even a minor increase by 0.01% can be tens of millions of dollars. For this purpose, we suggest to make two recommendations to Avito:-

***Recommendation 1***



* Set up a Constant Monitoring IT infrastructure as shown above
* Tweak the model as needed based on change in behavioral pattern
* Constant measurement of ROI and other relevant KPIs
* Invest more in research that can give more insight using new inputs e.g.
  + eye sight tracking , mouse hovering

***Recommendation 2***

Perform a more focused group study to understand customer base. **Customer based segmentation** will help Avito drive sales up by customizing their search results based on demographics, income, and age and so on.

**Appendix A: R Code**

# Kaggle-competititon "Avito Context Ad Clicks"

# See https://www.kaggle.com/c/avito-context-ad-clicks

# In order to run this script on Kaggle-scripts I had to limit the number of entries to read

# from the database as well as to decrease the sample-size. With the full dataset from the database as well

# as a sample of 20 millions entries

install.packages("data.table")

install.packages("caret")

install.packages("RSQLite")

install.packages("pbkrtest")

install.packages("sqldf")

install.packages("Amelia")

install.packages("e1071")

install.packages("doParallel")

install.packages("kernlab")

install.packages("pROC")

install.packages("DMwR")

#-------------------------------------------------------------------------------------------------------

library("DMwR")

library("caret")

library("kernlab")

library("doParallel")

library("pROC")

library(“e1071”)

library(sqldf)

library("data.table")

library("RSQLite")

library(pbkrtest)

library(Amelia)

library(ggplot2)

library("caret")

library(“mlbench”)

library(“e1071”)

library(“ROCR”)

#-------------------------------------------------------------------------------------------------------

**# Prepare database**

#-------------------------------------------------------------------------------------------------------

db <- dbConnect(SQLite(), dbname="C:/Users/ggade/Documents/MIS 620/Avito/database.sqlite")

#dbListTables(db)

#-------------------------------------------------------------------------------------------------------

**# Define constants to improve readability of large number**

#-------------------------------------------------------------------------------------------------------

thousand <- 1000

million <- thousand \* thousand

billion <- thousand \* million

#-------------------------------------------------------------------------------------------------------

**# Runs the query, fetches the given number of entries and returns a data.table**

#-------------------------------------------------------------------------------------------------------

fetch <- function(db, query, n = 10000) {

result <- dbSendQuery(db, query)

data <- dbFetch(result, n)

dbClearResult(result)

return(as.data.table(data))

}

#-------------------------------------------------------------------------------------------------------

**Preparing to extract the records from the sqlite database into R**

#-------------------------------------------------------------------------------------------------------

AdsInfo <- fetch(db, "select \* from AdsInfo", 400\*thousand)

Category <- fetch(db, "select \* from Category", -1)

Location <- fetch(db, "select \* from Location", -1)

PhoneRequestsStream <- fetch(db, "select \* from PhoneRequestsStream", 400\*thousand)

VisitsStream <- fetch(db, "select \* from VisitsStream", 400\*thousand)

UserInfo <- fetch(db, "select \* from UserInfo", 400\*thousand)

SearchInfo <- fetch(db, "select \* from SearchInfo", 400\*thousand)

trainSearchStream <- fetch(db, "select \* from trainSearchStream", 400\*thousand)

#-------------------------------------------------------------------------------------------------------

**#Buiilding the entire relational schema for the database**

#-------------------------------------------------------------------------------------------------------

temp1 <- sqldf("select a.\*, b.Level as Category\_Level from AdsInfo a left join Category b on a.CategoryID = b.CategoryID ")

temp2 <- sqldf("select a.\*, b.Level as Category\_Level from SearchInfo a left join Category b on a.CategoryID = b.CategoryID ")

temp3 <- sqldf("select a.\*, b.Level as Location\_Level from temp1 a left join Location b on a.LocationID = b.LocationID ")

temp4 <- sqldf("select a.\*, b.Level as Location\_Level from temp2 a left join Location b on a.LocationID = b.LocationID ")

temp5 <- sqldf("select a.\*, b.IPID as PR\_IPID from temp3 a left join PhoneRequestsStream b on a.AdID = b.AdID ")

temp6 <- sqldf("select a.\*, b.IPID as PR\_IPID from UserInfo a left join PhoneRequestsStream b on a.UserID = b.UserID ")

temp6 <- subset(temp6,select=c(1,2,3,4,5))

temp7 <- sqldf("select a.\*, b.IPID as VS\_IPID, b.ViewDate from temp6 a left join VisitsStream b on a.UserID = b.UserID ")

temp8 <- sqldf("select a.\*, b.IPID as VS\_IPID, b.ViewDate from temp5 a left join VisitsStream b on a.AdID = b.AdID ")

temp8 <- subset(temp8,select=c(1,2,3,5,7,8,9))

temp9 <- sqldf("select a.\* from temp4 a left join temp7 b on a.UserID = b.UserID ")

temp10 <- sqldf("select a.\*, b.SearchDate, b.UserID, b.IsUserLoggedOn,b.SearchQuery, b.LocationID, b.CategoryID, b.SearchParams, b.Category\_Level, b.Location\_Level from trainSearchStream a left join temp9 b on a.SearchID = b.SearchID ")

#removing that data where objectType != 3 so 246362 records left

final <- sqldf("select a.\*, b.Price, b.IsContext from temp10 a left join temp8 b on a.AdID = b.AdID where a.ObjectType =3")

missmap(final, main = "Missing values vs. observed")

summary(final)

#removing Price as its not affect IsClick

final <- subset(final,select=c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,17))

#removing search Params, Object Type and search query

final<- subset(final, select= c(1,2,3,5,6,7,8,9,11,12,14, 15, 16))

#-------------------------------------------------------------------------------------------------------

**Checking factor, numeric type for variables**

#-------------------------------------------------------------------------------------------------------

#displaying the RESULTS IF VARIABLES ARE FACTOR OR NOT

a<-sapply(final,function(x)is.factor(x))

b<-sapply(final,function(x)is.numeric(x))

final$IsClick <- as.factor(final$IsClick)

contrasts(final$IsClick)

final$IsUserLoggedOn <- as.factor(final$IsUserLoggedOn)

contrasts(final$IsUserLoggedOn)

final$Location\_Level <- as.factor(final$Location\_Level)

contrasts(final$Location\_Level)

final$Category\_Level <- as.factor(final$Category\_Level)

contrasts(final$Category\_Level)

final<- subset(final, select= c(1,2,3,4,5,6,7,8,9,10,11,12))

#-------------------------------------------------------------------------------------------------------

**#Register core backend, using 4 cores**

#-------------------------------------------------------------------------------------------------------

cl <- makeCluster(4)

registerDoParallel(cl)

#-------------------------------------------------------------------------------------------------------

set.seed(123)

final$IsClick <- factor(final$IsClick)

final$IsUserLoggedOn <- factor(final$IsUserLoggedOn)

final$Category\_Level <-factor(final$Category\_Level)

final$Location\_Level <- factor(final$Location\_Level)

final$Position<- factor(final$Position)

yesnofactors <- factor(c("yes", "no"))

levels(final$IsClick) <- make.names(levels(factor(yesnofactors)))

str(final)

#-------------------------------------------------------------------------------------------------------

**# create a 80/20 partition**

#-------------------------------------------------------------------------------------------------------

inTrain<-createDataPartition(y=final$IsClick, p=.8, list=FALSE)

nrow(inTrain)

final.train <- final[inTrain,]

summary(final.train)

final.test <- final[-inTrain,]

summary(final.test)

#-------------------------------------------------------------------------------------------------------

**#use SMOTE to adjust for sampling**

#-------------------------------------------------------------------------------------------------------

final.train.smote <- SMOTE(IsClick ~., final.train, perc.over =400, perc.under=150)

table(final.train.smote$IsClick)

prop.table(table(final.train.smote$IsClick))

#-------------------------------------------------------------------------------------------------------

**#some parameters to control the sampling during parameter tuning and testing**

#-------------------------------------------------------------------------------------------------------

ctrl <- trainControl(method="repeatedcv", repeats=3,

classProbs=TRUE,

#function used to measure performance

summaryFunction = twoClassSummary)

#twoClassSummary is built in function with ROC, Sensitivity and Specificity

#-------------------------------------------------------------------------------------------------------

**#Decision Tree**

#-------------------------------------------------------------------------------------------------------

modelLookup("rpart")

m.rpart <- train(IsClick ~ .,

trControl = ctrl,

metric = "ROC", #using AUC to find best performing parameters

preProc = c("range", "nzv"), #scale from 0 to 1 and from columns with zero variance

data = final.train.smote,

method = "rpart")

m.rpart

plot(m.rpart)

p.rpart <- predict(m.rpart,final.test)

confusionMatrix(p.rpart,final.test$IsClick)

roc\_final<-roc(as.numeric(final.test$IsClick),as.numeric(p.rpart))

plot.roc(roc\_final)

#-------------------------------------------------------------------------------------------------------

**#Logistic Regression**

#-------------------------------------------------------------------------------------------------------

modelLookup("glm")

m.smote.glm <- train(IsClick~ .,

trControl = ctrl,

metric = "ROC", #using AUC to find best performing parameters

preProc = c("range", "nzv"), #scale from 0 to 1 and from columns with zero variance

data = final.train.smote,

method = "glm")

m.smote.glm

p.smote.glm<- predict(m.smote.glm,final.test)

confusionMatrix(p.smote.glm,final.test$IsClick)

roc\_final<-roc(as.numeric(final.test$IsClick),as.numeric(p.smote.glm))

plot.roc(roc\_final)

#-------------------------------------------------------------------------------------------------------

**# randomForest with SMOTE**

#-------------------------------------------------------------------------------------------------------

install.packages("randomForest")

library ("randomForest")

modelLookup("rf")

m.rf <- randomForest(IsClick ~., data = final.train.smote, ntree=50,do.trace=2,replace=FALSE,verboseiter=FALSE)

m.rf

p.rf<- predict(m.rf,final.test)

confusionMatrix(p.rf,final.test$IsClick)

plot(roc(final.test$IsClick,as.numeric(p.rf)))

#-------------------------------------------------------------------------------------------------------

**# use the NB classifier with Laplace smoothing**

#-------------------------------------------------------------------------------------------------------

#building the model using naiveBayes on data pre-processed using smote and laplace

m.smote.naivebayes = naiveBayes(IsClick ~., data = final.train.smote , laplace=.01)

#displaying model statistics

m.smote.naivebayes

p.smote.nb<- predict(m.smote.naivebayes,final.test)

confusionMatrix\_nb <- confusionMatrix(p.smote.nb,final.test$IsClick)

confusionMatrix\_nb

#computing the ROC for the model: AUC = 0.549

roc\_final\_naivebayes<-roc(as.numeric(final.test$IsClick),as.numeric(p.smote.nb))

roc\_final\_naivebayes

#plotting the ROC

plot.roc(roc\_final\_naivebayes)

#-------------------------------------------------------------------------------------------------------

**#Box Plot**

#-------------------------------------------------------------------------------------------------------

rValues <- resamples(list(rpart=m.rpart, glm =m.smote.glm))

bwplot(rValues, metric="ROC", horizontal=TRUE,

col=c("red","blue"))

#-------------------------------------------------------------------------------------------------------