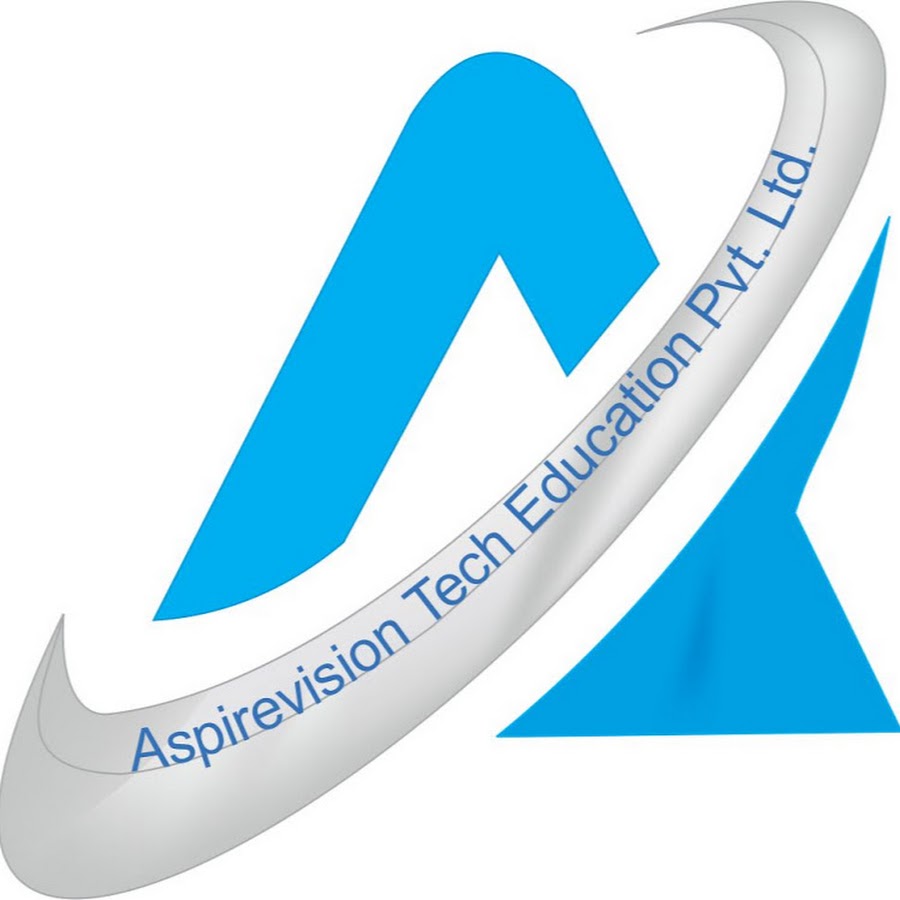
**Aspirevision Tech Education Pvt. Ltd.**



**Project Documentation for:**

**Predicting the likelihood of E-signing based on financial history**

**Submitted by:**

ARNAV JYOTI NATH

**Under the Guidance of:**

*Md. Farmanul Haque*

**1. Acknowledgement:**

I am very happy to compile this project, along with my talented group members. However, it would not have been possible without aid from multiple individuals. I am highly indebted to AspireVision Tech for training me and providing me with all the necessary knowledge. I would like to express my gratitude to Md. Farmanul Haque, whose advice and guidance has proven invaluable in bringing about the end product. It is through his constant encouragement that I have been able to see this project to its completion. I also have immense appreciation for my group members for their constant cooperation and help.

**2. Abstract:**

The objective of this project is to utilize the data of financial history of a group of people who had applied for loan in a bank and analyse their financial status and activities . Through this project, we will attempt to efficiently predict whether or not the user will be e-signing or not. This will help the company engage with the customer who is likely to e-signed and not going for loan. Identifying the behavioral pattern of the customer acts as a catalyst in identifying the reasons behind the disengagement of the customers. There are approximately 20+ attributes being considered while making predictions about e-signing, such as- age, income, has debt, and whether user owns a house or rents one, and so on. This project seeks to utilize a variety of different Machine Learning algorithms to train using the provided data and make the predictions. To conclude, the purpose of the project is to single out all the possible users who may choose to cancel their subscriptions.

**3. Contents**

**1. Acknowledgement………………………………………………………………...2**

**2. Abstract……………………………………………………………………………3**

**3. Contents……………………………………………………………………………4**

**4. Introduction……………………………………………………………………….5**

4.1 Problem Statement…………………………………………………………...4

4.2 Project Description…………………………………………………………...4

4.3 System Requirements………………………………………………………...4

**5. Libraries and Models Used……………………………………………………….5**

5.1 Libraries Used………………………………………………………………...5

5.2 Functions Used………………………………………………………………..6

5.3 Models Used…………………………………………………………………11

**6. Algorithm of the Code..………………………………………………………….14**

6.1 Exploratory Data Analysis…………………………………………………..14

6.1.1 Importing Necessary Libraries and Data Set………………………….14

6.1.2 Exploring the Data Set………………………………………………..15

6.1.3 Early Data Cleaning…………………………………………………..15

6.1.4 Representing Frequency of Data Through Histogram…………………15,16

6.1.5 Representing Correlation…………...17

6.2 Model Fitting and Prediction………………………………………………...18

6.2.1 Data Importing, Cleaning Features, and Splitting…………………….18

6.2.2 Model Fitting and Optimization...…………………………………….18

6.2.3 Confusion Matrix……………………………………………………..19,20

**7. Conclusion………………………………………………………………………..21**

**8. Future Scope……………………………………………………………………..22**

**9. Bibliography……………………………………………………………………..23**

**4.Introduction :**

**4.1 Problem Statement:** E-Companies are usually heavily reliant on paid subscriptions as a primary source of income, however there is always the issue of a certain customer losing interest in the service and eventually churning. This is a very common problem in the industry and can be remedied possibly by re-engaging the user before they churn.

**4.2 Project Description:** The project is the implementation of Machine Learning classification algorithm(s) on a set of data giving some financial and behavioral details of the users of some subscription based service. The objective is to efficiently and accurately predict which customers may churn (cancel subscription), so that the e-company offering the subscription may engage with these customers and prevent future loss.

**4.3 Objectives:** The objectives of the project will be discussed in this particular section. They are as follows:

-Analyze the financial data and behaviour of a set of users.

-Find the frequency of the variants of data and represent them visually.

-Find correlation of each attribute given.

-Clean the data to ensure it can be fit to any classification model.

-Finding accuracy of a variety of classification models to find the best classifier.

-Optimizing the best classification model to obtain further accuracy.

-Displaying the possibility of e-signing on a new set of independent variables.

***System Requirements:***

Operating System: Windows XP SP3 and above Processor: 2 x 64-bit 2.8 GHz 8.00 GT/s CPUs

RAM Requirement: 32 GB Editing

Tools: Spyder or Jupyter Page

**5 5. Libraries and Models Used**

**5.1 Libraries Used:** The following is a list of libraries used throughout the entirety of coding along with their basic purpose:

**Pandas** : For data retrieval, manipulation, storage and analysis.

**numpy** : Adds support for large, multidimensional arrays and matrices, along with a large collection of high-level mathematical functions.

**Matplotlib:** Provides an object-oriented API for embedding plots into applications.

**Seaborn:** Data visualization library based on matplotlib, used for statistical data.

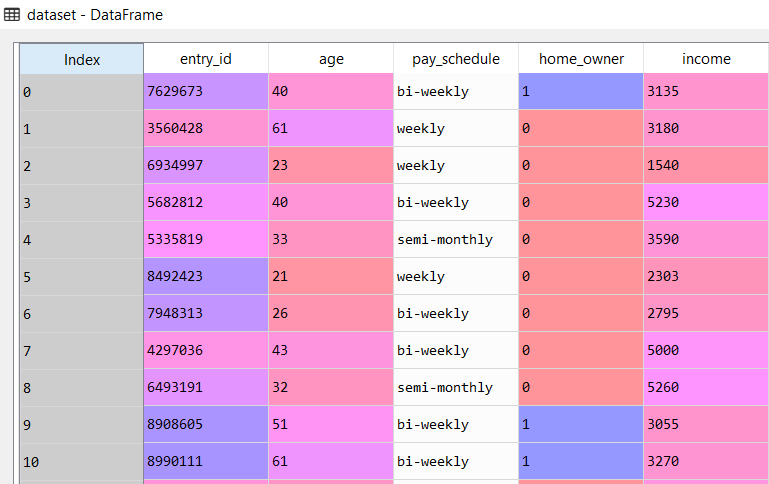
**Time** : Provides various time-related functions

**Random** : pick random elements within a uniform range

**Sklearn** : the library that holds the algorithm of ,machine learning

**5.2 Functions Used:** The following is a list of functions used throughout the entirety of coding along with their basic purpose and their parent library:

**read\_csv( )** - Read a comma-separated values (.csv) file into a data frame. See below for how the data frame is represented in our editor:

*Data set obtained after reading a .csv file*

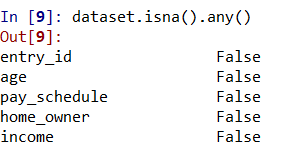
**to\_csv( )** - Write object to a comma-separated values (.csv) file.

**head( )** - Displays first 5 rows of data, unless a number is specified in parameters, in which case, that many number of rows from start is displayed.

**columns** - Displays all the column heads of a given data set.

**describe( )** - is used to view some basic statistical details like percentile, mean, std etc. of a data frame. **isna(** ) - Returns a Boolean value true if missing values are present, otherwise returns false.

**any( )** - Accepts iterable (list, tuple, dictionary etc.) as an argument and return true if any of the element in iterable is true.

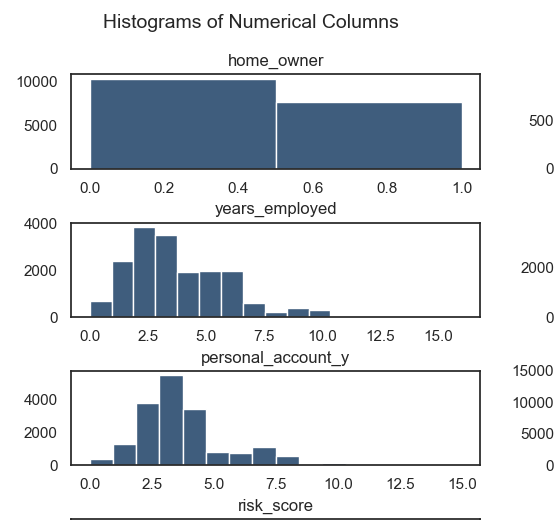


*Output of any( ) function*

**drop(columns = [ ])** - Deletes the column specified by column header name within the [ ] brackets.

**figure ( )** - Create a new figure.

**suptitle( ) -** Add a centered title name to the figure.



Title of histogram obtained

**subplot( )** - Add subplot to the current figure.

**gca( )** - Stands for Get Column Access, iterates through all the values in a column.

**set\_title( )** - Sets a string value as the title of some data visualization instance.

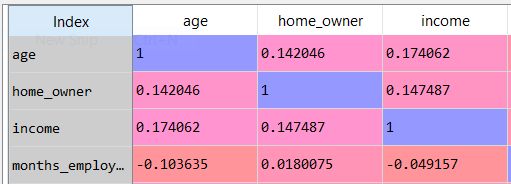
**iloc**[ ] - Stands for index location. Used to select some rows of data from a column.

**tight\_layout( )** - Automatically adjust subplot parameters to give specified padding.

**corrwith( )** - Used to compute pairwise correlation between rows or columns of two data frame objects.

**set( )** - used to convert any of the iterable to the distinct element and sorted sequence of iterable elements, commonly called Set.

**corr( )** - used to find the pairwise correlation of all columns in the data frame.



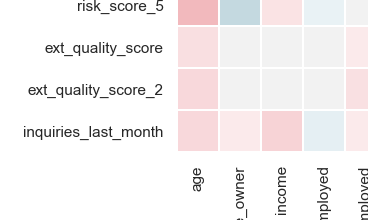
*Snippet of correlation matrix*

**zeros\_like( )** - Return an array of zeros with the same shape and type as a given array.

**triu\_indices\_from( )** - Return the indices for the upper-triangle of two-dimensional array.

**diverging\_palette( )** - Make a diverging palette between two colors.

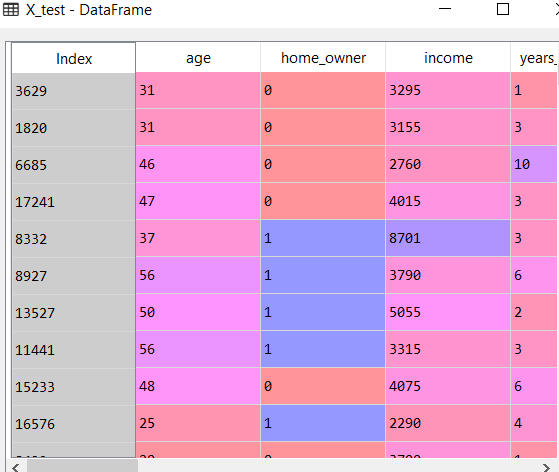
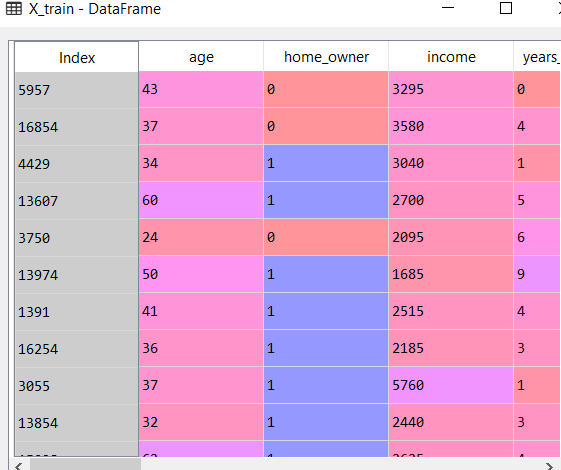
**heatmap( )** - Plot rectangular data as a color-encoded matrix.



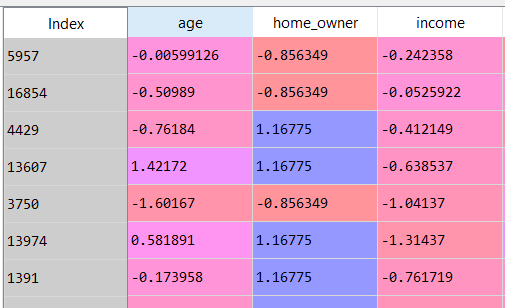
*Snippet of heat map*

**random.seed(100)** - to replicate the results that are done before a random seed is a starting point in generating random numbers. A random seed specifies the start point when a computer generates a random number sequence.

**test\_train\_split( )** - Split arrays or matrices into random train and test subsets. Test data and train data



**StandardScaler( )** - Standardize features by removing the mean and scaling to unit variance.



*Snippet of changed values after scaling*

**DataFrame( )** - Creates a two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns).

**fit( )** - Sets given data according to a specified model.

**predict( )** - Given a trained model, predict the label of a new set of data.

**accuracy\_score( )** - In multilabel classification, this function computes subset accuracy.

accuracy.PNG

**precision\_score( )** - Compute the precision. The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives.

precc.PNG

**recall\_score( )** - Compute the recall. The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives.

recall.PNG

**f1\_score( )** - Compute the F1 score, also known as balanced F-score or F-measure. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The formula for the F1 score is: F1 = 2 \* (precision \* recall) / (precision + recall)

f1.PNG

**append( )** - Adds its argument as a single element to the end of a list.

**cross\_val\_score( )** - Evaluate a score by cross-validation.

**GridSearchCV( )** - Exhaustive search over specified parameter values for an estimator.

**confusion\_matrix( )** - Compute confusion matrix to evaluate the accuracy of a classification. Confusion matrix obtained from data

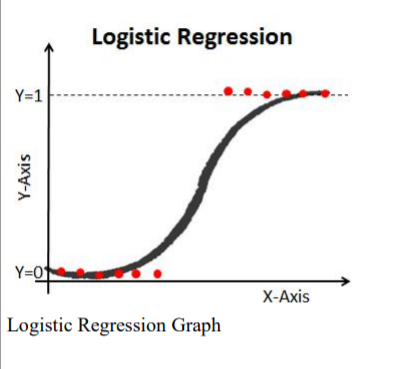
**time( )** - Returns the number of seconds passed since epoch.

**5.3 Models Used:**

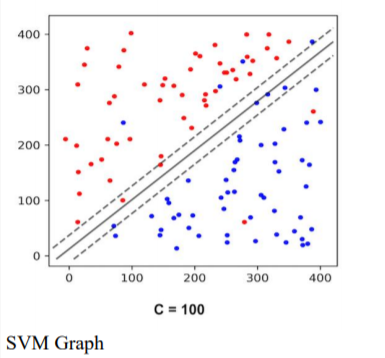
The following Machine Learning models have been used to predict the data:

**Logistic Regression** - It is a statistical method for analysing a data set in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes). The goal of logistic regression is to find the best fitting model to describe the relationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent (predictor or explanatory) variables.

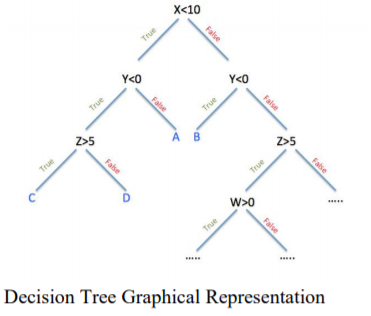
Equation: y = log[ p / (1 - p) ] where p is probability of success Logistic Regression Graph



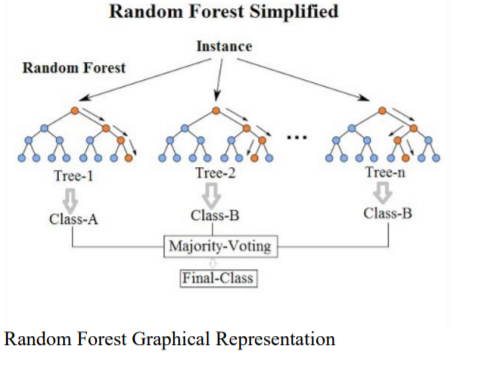
**SVM** - It is a supervised machine learning algorithm capable of performing classification, regression and even outlying entity detection. The linear SVM classifier works by drawing a straight line known as a support vector to divide a plane representation into two hyper planes. All the data points that fall on one side of the vector will be labeled as one class and all the points that fall on the other side will be labeled as the second.



**Decision Trees** - Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches and a leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.



**Random Forest -** Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees’ habit of over fitting to their training set.



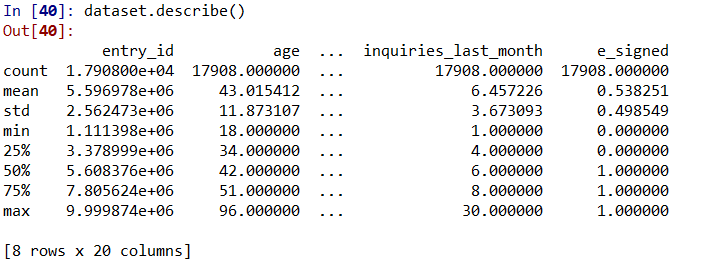
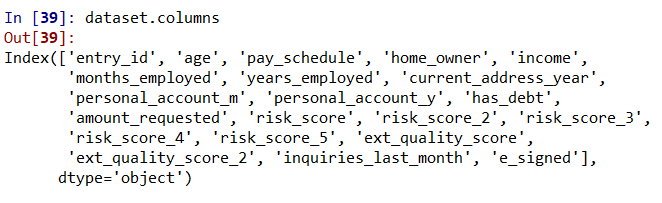
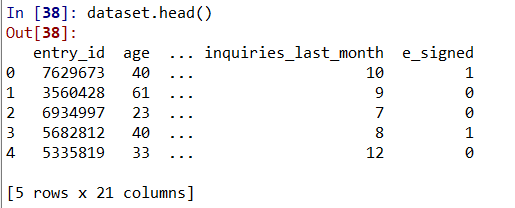
**6. Algorithm of Code**

The following is the algorithm of the code and a list of all the operations performed on the data set to obtain the final prediction and best model:

**6.1 Exploratory Data Analysis (EDA.py)**

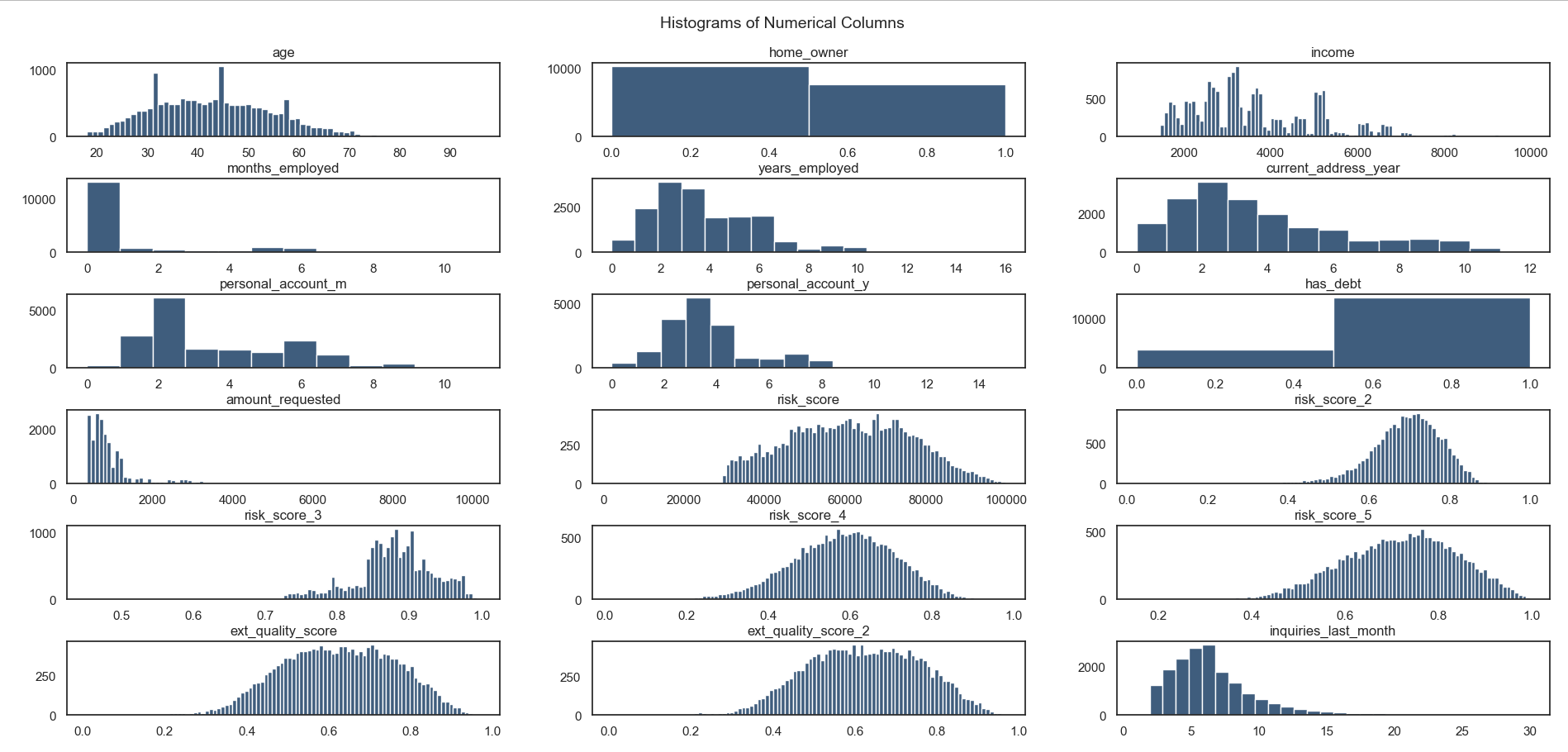
**6.1.1 Importing Necessary Libraries and Data Set**: The aforementioned libraries of *numpy, matplotlib, pandas* are imported for mathematical operations on matrices, graphical representation of data and data retrieval and manipulation respectively. The pandas library is then used to import the given data set through read\_csv( ) function.

**6.1.2 Exploring the Data**: The attributes present in the data set, the first 5 entities as well as a summary of some key values is presented to further get an overview of the data set.



**6.1.3 Early Data Cleaning**: we use the drop( ) function to remove those columns from the data set which are important but not required in the training set.

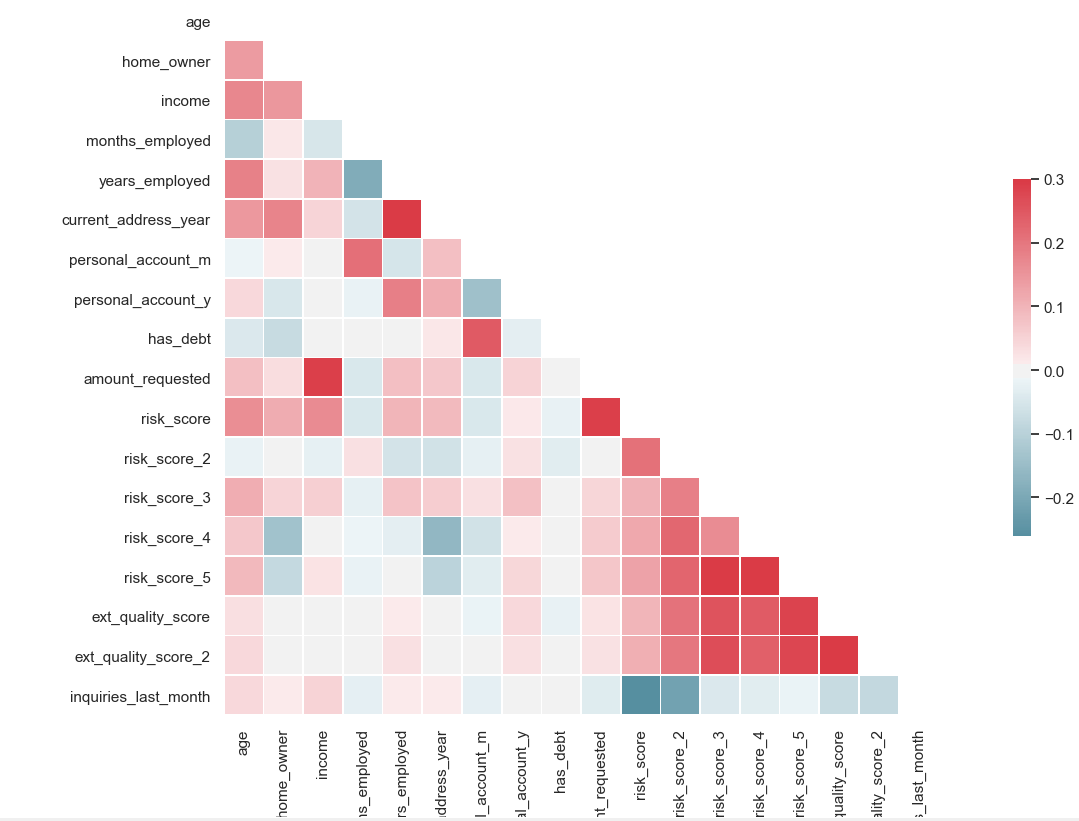
**6.1.4 Representing Frequency of Data through Histograms:** We produce a copy of the data set containing all the classifying attributes and we visually represent the frequency of occurrence of each classification type in our data set through percentages in Histograms. There are 18 attributes including but not limited to *home-owner, personal account is monthly or yearly based,whether inquires\_last mont* or not, and so on. An individual bar graph is made for each of the attributes.



*Each bar graph represents a classifying feature and the distribution of each type of classification between all the entities*

**6.1.5 Representing Correlation:**

A heat map is a two-dimensional representation of information with the help of colors. Heat maps can help the user visualize simple or complex information. Finally, a heat map is produced of all the features, so that the correlation of each feature to every other feature through the entire data set is represented in a single visual representation. We mask the upper half of the heat map as it is redundant due to being a mirror of the lower half. We do not include entry\_ID and whether or not user has e-signed or not, as user ID is not an affecting feature and the *e-signed* is the dependant variable.

Both the axes have all the attributes set along them. A darker shade of red on a cell represents a stronger positive correlation between the two attributes intersecting on that cell, and a darker shade of blue on a cell represents a stronger negative correlation.

**6.2 Model Fitting and Prediction (Model.py)**

**6.2.1 Data Importing, Cleaning Features and Data Splitting:** We import the modified data set through read\_csv( ) function, and we store the entry ID, e\_signed code in a separate array. Accordingly we remove the entryID, e\_signed feature (as we will use it later to identify the users and it has no purpose now). We obtain dummy variables for pay\_scheduled due to its being more than 2 classification groups, and we also delete the columns storing whether or not the features have missing or not applicable values to avoid the dummy variable trap. We divide our data set on 8:2 ratio (80% train, 20% test) where our X (independent variable) constitutes of a feature matrix of all features (excepte-signed status), and Y (dependent variable) will have e-signed status only in a vector. Feature scaling is performed on the training and test sets of X, and we use Data Frame to ensure that our automatically imposed index column is stored safely so we can link the data.

**6.2.2 Model Fitting and Optimization:** To find the best model for our data set, we will fit it to multiple different classification models and check the accuracy, precision, recall and f1 score for each of those models until we find the best model (using the appropriate score functions for each of them). The following models were used in this project:

1. Logistic Regression

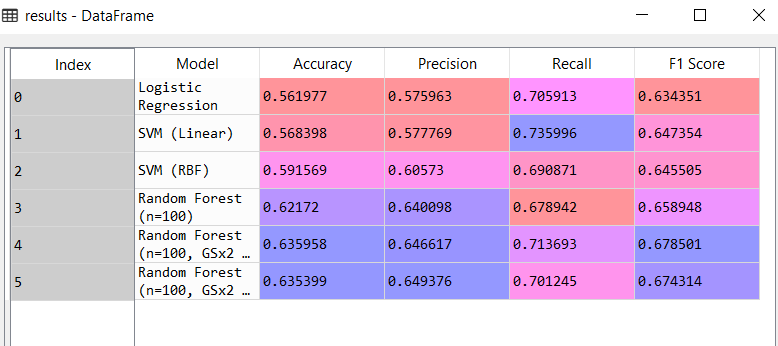
2. SVC (Linear Kernel)

3. SVC (RBF Kernel)

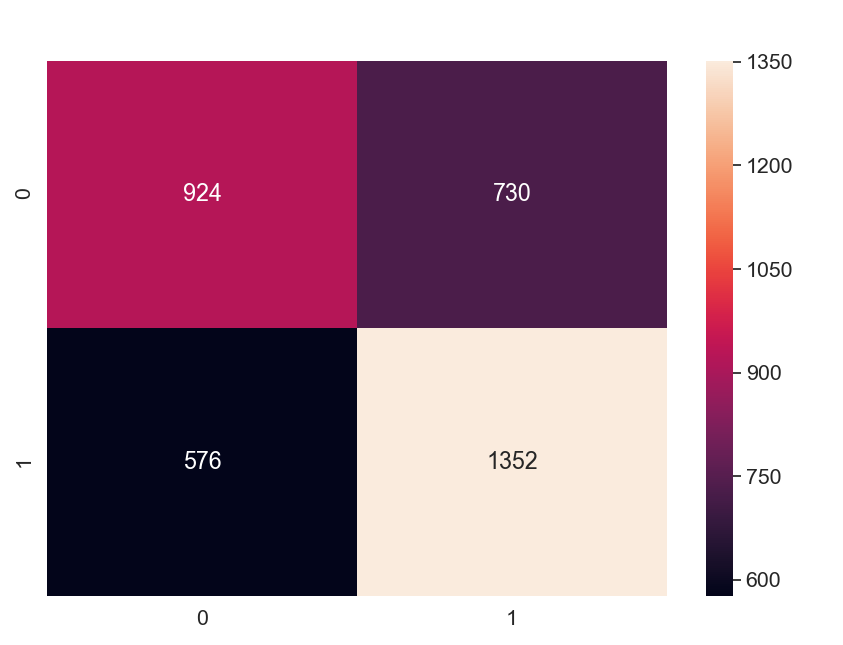
4. Decision Tree

5. Random Forest

We have found that Random Forest gives the best accuracy, so we use K-fold cross validation and 2 rounds of Grid Search (entropy criterion) to further optimize our predictions from Random Forest. K-Folds Cross Validation is a method of training a model by dividing the data set into n number of folds, or divisions. Now, the training of the data is done on (n-1) of those divisions with the n th division being used for testing. This enhances the accuracy, Grid Search is an exhaustive search through each and every value for a given estimator. This also enhances the accuracy of the model. We store the scores of each fit to obtain the final output**:**

****

**6.2.3 Confusion Matrix:** Now that we have finally obtained our best model and optimized it as well, we can represent the exact predictions and their correctness through a Confusion Matrix. Confusion Matrix is a 2x2 matrix where each cell represents a prediction. The first cell [0,0] represents the number of positive predictions that we are correct (true positive), the second cell [0,1] represents number of positive predictions that are actually negative (false positive), the third cell [1,0] represents number of negative predictions that are actually positive (false negative), the fourth cell [1,1] represents number of correctly predicted false cases. Usually, the incorrect predictions of the third cell are fatal errors and must be minimized as much as possible.



This is the output confusion matrix we have obtained from our predictions, where we have 924 true positive predictions, 730 false positive predictions, 576 false negative predictions, 1352 true negative predictions. Darker color represents a lower value of number of predictions in that category.

**7. Conclusion**

After testing accuracy of multiple models, namely - Logistic Regression, SVM, Decision Tree and Random Forest and calculating their prediction accuracy, we have found that Random Forest provides the best accuracy (around 62%) which we have further optimized through Grid Search to obtain a final accuracy of 64.00%(approx), which shows our algorithm is quite accurate, especially considering we are working with 17,000+ entities. It has provided us with an indication of which of these 17,000 users are likely to e-sign. We have purposefully left some attributes by dropping them as the aim is to distinctly single out those customers who are likely to e-sign so the company may engage with them again and rekindle their interest in the service. In conclusion, we have obtained an accurate model (Random Forest) and predicted possible customers’ e-signed out at 64.00% accuracy so that churn-rate can be effectively minimized by the company.

**8. Future Scope**

With a final prediction accuracy of approximately 64.00% on a data set of 17,000 entities, the selected model has proven to be considerably effective, especially so after optimization. Therefore it can be used as the basis or framework for any such bank to predict customer signing and preemptively interact with them to reduce the overall signed rate. While the reduction of sign-rate is a long term plan, on a more immediate time frame, the algorithm can also be used to test the success of any kind of programs launched, since the prediction made by our project, that is- whether or not an individual is likely to apply for a loan or not is essentially a measure of prolonged success of a service. The financial limitations of the users can also be studied so that the interest range can accordingly be increased or decreased to increase the number of customers granting the loan . Also worth noting is that since we test the accuracy of predictions by fitting on multiple models, we can accommodate for a large range of scales of data and use the most appropriate model. Therefore, this project provides has a very wide scope for usage in any bank looking to reduce signed-rate.

**9. Bibliography**

**References Used**:

1. Basics of Python: https://www.w3schools.com/python/python\_reference.aspϖ

2. Python Libraries: https://numpy.org/ϖ https://pandas.pydata.org/ϖ https://matplotlib.org/3.1.1/tutorials/introductory/pyplot.htmlϖ

3. Sci-Kit Learn: https://scikit-learn.org/stable/getting\_started.htmlϖ

4. Seaborn: https://seaborn.pydata.org/introduction.htmlϖ