**IST 707 – M002**

**People’s Analytics**

**-**

**Dhwani Gandhi**

**Karan Ashar**

**Contents:**

1. **Introduction**
2. **Describing the Data**
3. **Data Visualizations**
4. **Data Pre-Processing**
5. **Feature Selection and Important Features**
6. **Predictive Modelling**
7. **Conclusion**

**Introduction**

The company offers training to individuals in the field of Data. The problem they noticed is that after training the individuals many of them leave the company and take up a job somewhere else. This is a loss to the company. The managers want us to develop a machine learning model which could predict if that individual will take up a job offer at the company or not. Predicting this would help make the selection process easier and could help train only those individuals which are going to be profitable to the company in the future. By doing so time, effort and money, all three get saved.

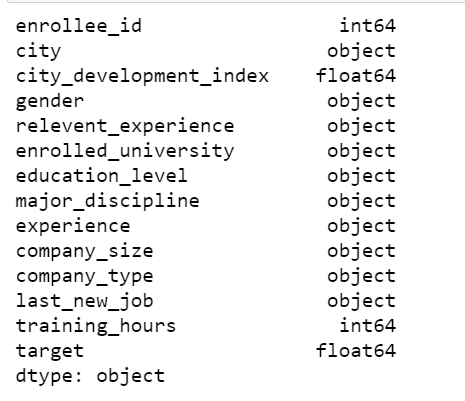
The problem defined is a classification task. We will be using various classification algorithms to help predict if that individual is going to take up a job at the company or not. We will also be creating an ensemble model by combining results of all the individual models.

The classification algorithms we will be using are:  
1. Logistic Regression  
2. Random Forest  
3. Gradient Boosting Tree   
4. Support Vector Machine

**Describing the Data**

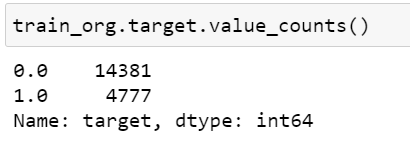
The data consists of 14 columns which includes the target variable.   
The target variable:  
0 - Person will take up the job at the company   
1- Person will not take up the job at the company

The image below shows the columns present in the dataset



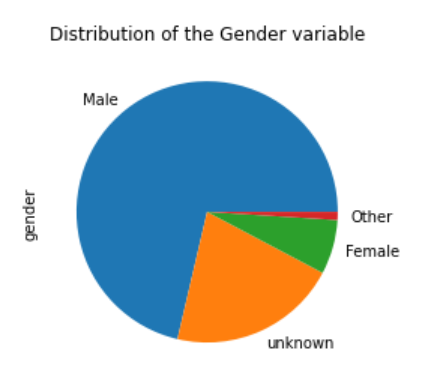
enrollee\_Id is the primary key. City is a demographic variable. ‘city\_development\_index’ is the development index of the city the candidate belongs to. The other variables describe about the candidate based on education and experience. The ‘training\_hours’ indicates the number of hours the candidate has taken up training for.

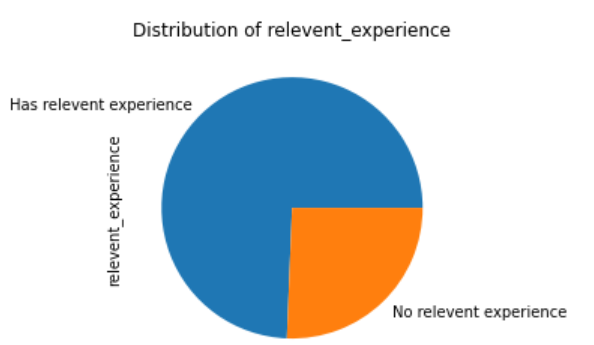
Below is the distribution of the target variable. As we can see the distribution is not even. So, we need to make sure we use the stratify technique while splitting our data.

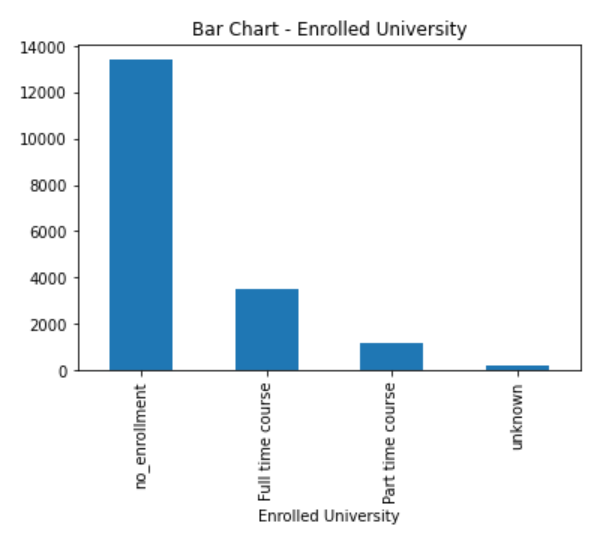


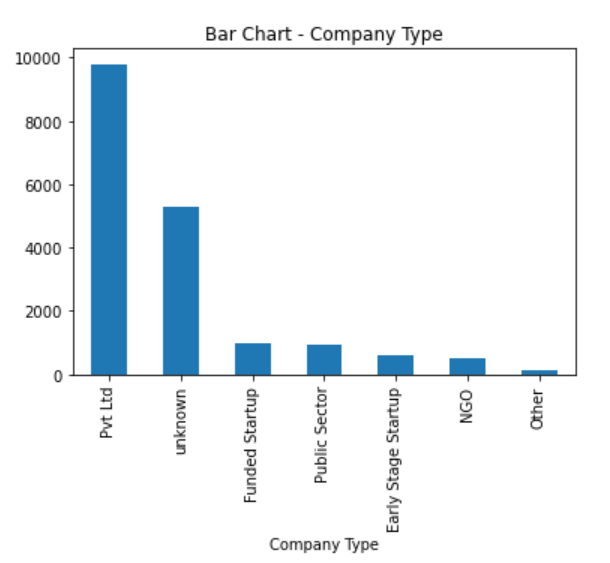
**Data Visualizations**

**Categorical Variables**

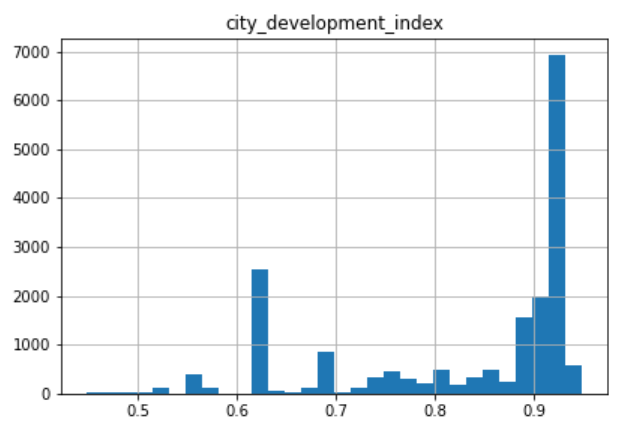




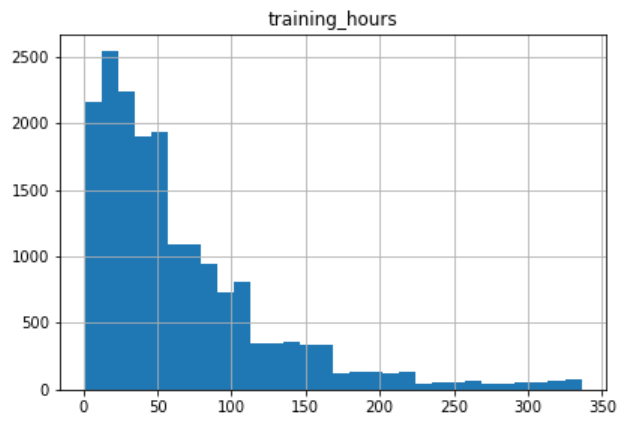




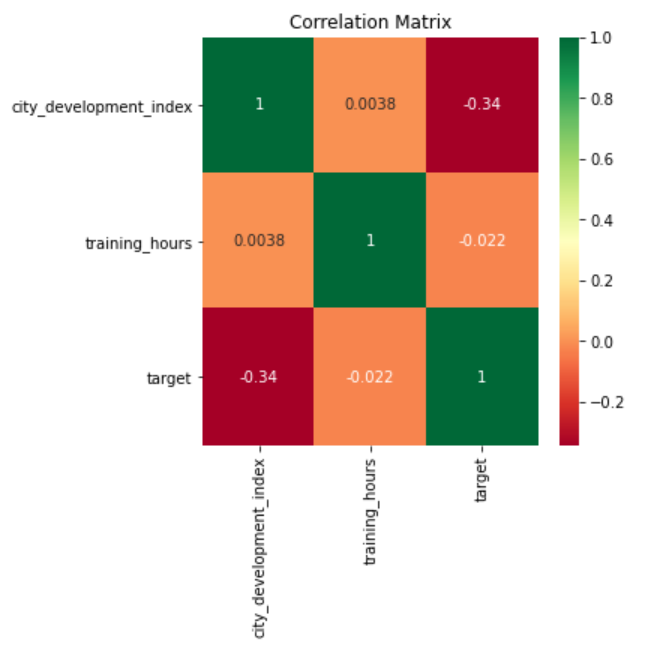
**Numerical Variables**



Histogram – city\_development\_index

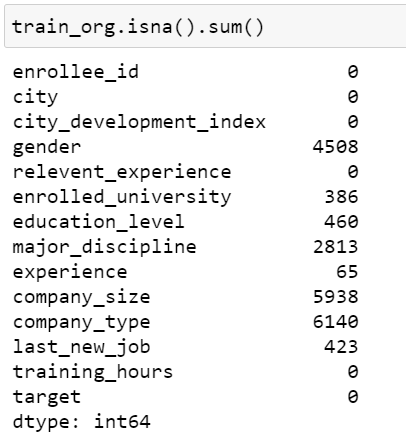


Histogram – training\_hours



**Data Pre-processing**

**Missing Data**  
There is a lot of missing data. All the missing data is in the categorical columns. Below is the number of missing data in each column



We first dropped all the rows which consisted of 3 or more than 3 missing values. This removed all those rows which barely carried any information.

After that we filled all the missing values with ‘unknown’

**Data Encoding**

All categorical features were converted to dummy variables. This was carried out by using the ‘get\_dummies’ method. This opened up all our categorical columns and assigned a ‘1’ if that variable was present and ‘0’ if not present.

**High Cardinality – City**

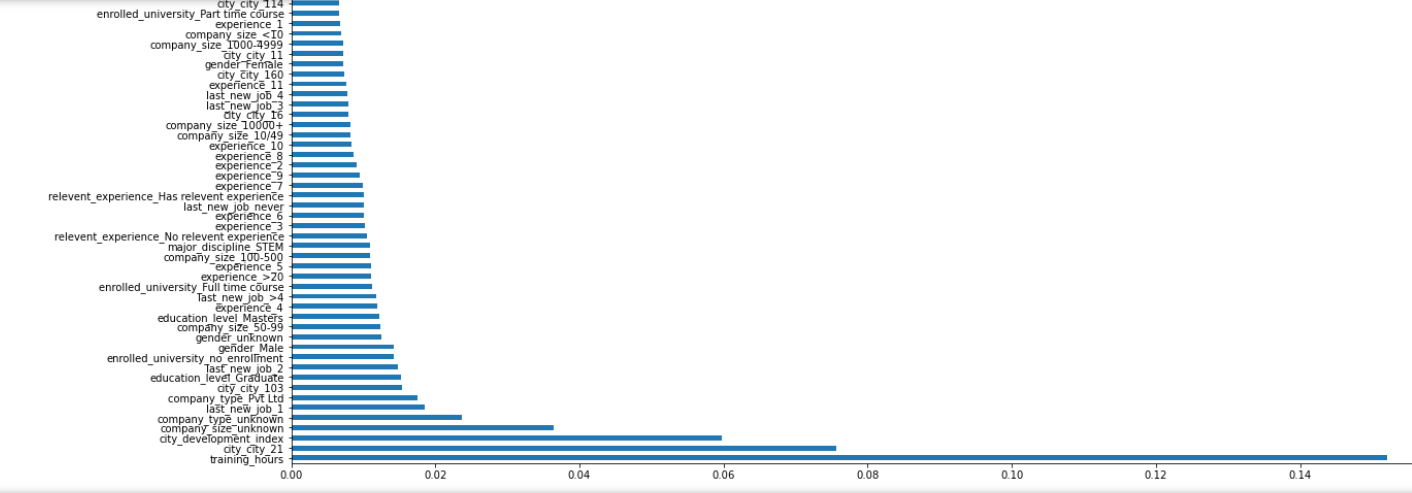
As we did the encoding, we realized that the ‘city’ variable had a very high cardinality. This meant that there were many unique values in the ‘city’ column. There were 126 unique variables. This is not good for a categorical variable.  
To tackle this problem we used Principal Component Analysis – PCA. We reduced the dimensionality of the features from 126 to only 33. In this process we could preserve 85% of the variance explained by these columns. This resolved our high cardinality problem.

**Feature Scaling**

The next step was to scale the numeric features. For this we used the MinMax scaler to scale the two numeric features and the 33 Principal Components.   
So now all our data is between 0 and 1.

**Feature Selection & Important Features**

To identify important features, we used the feature\_importance method of the Extra Trees Classifier. It does this by fitting the ExtraTrees model on the data and then identifying the important features.  
We carried this method twice. Once on the data before PCA was carried out. Second on the data after PCA.   
As we can see ‘training\_hours’ is the most important variable in both the cases.  
  
Feature Importance Before PCA



Feature Importance after PCA

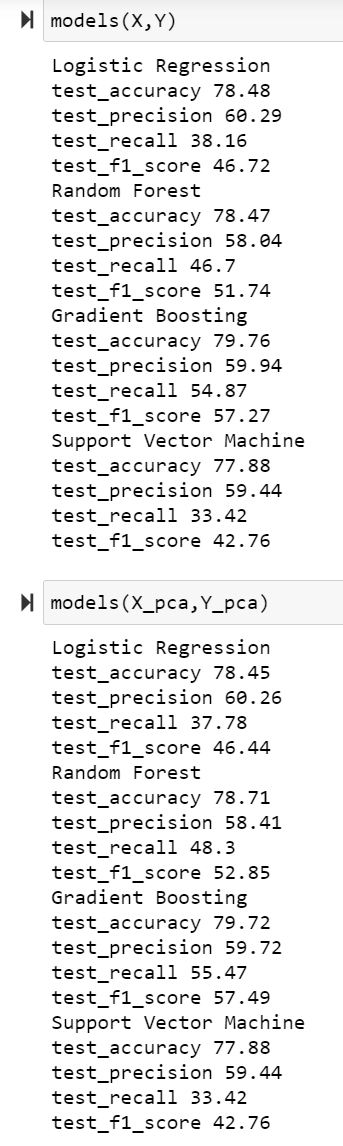
Graphical user interface

Description automatically generated with low confidence

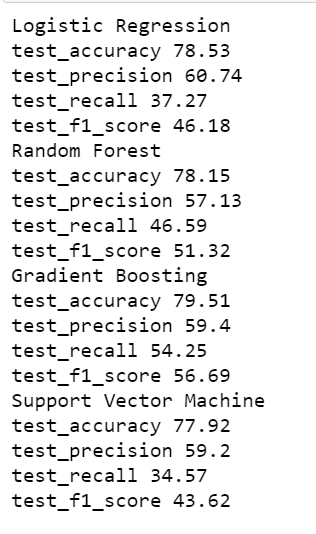
**Predictive Modelling**

The first step was to train default models on the data and record the scores. We trained the models on both the datasets, with PCA and without PCA.  
We used StratifiedKFold cross-validation with K = 5  


Below are the outputs from the models.

  
There does not seem to be much of a difference between both the datasets.

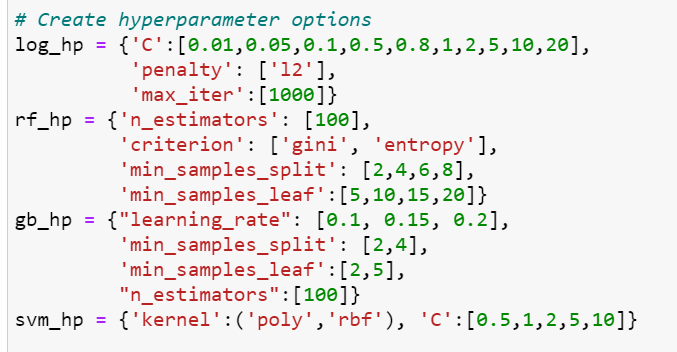
We also carry out the same procedure on the dataset with only the top 50 variables as decided by the ExtraTrees Classifier model.



There is not much change in the output. So, we decided that we will include all the features so that we will not leave out any kind of information. We will be using the dataset after PCA.

**Hyperparameter Tuning**

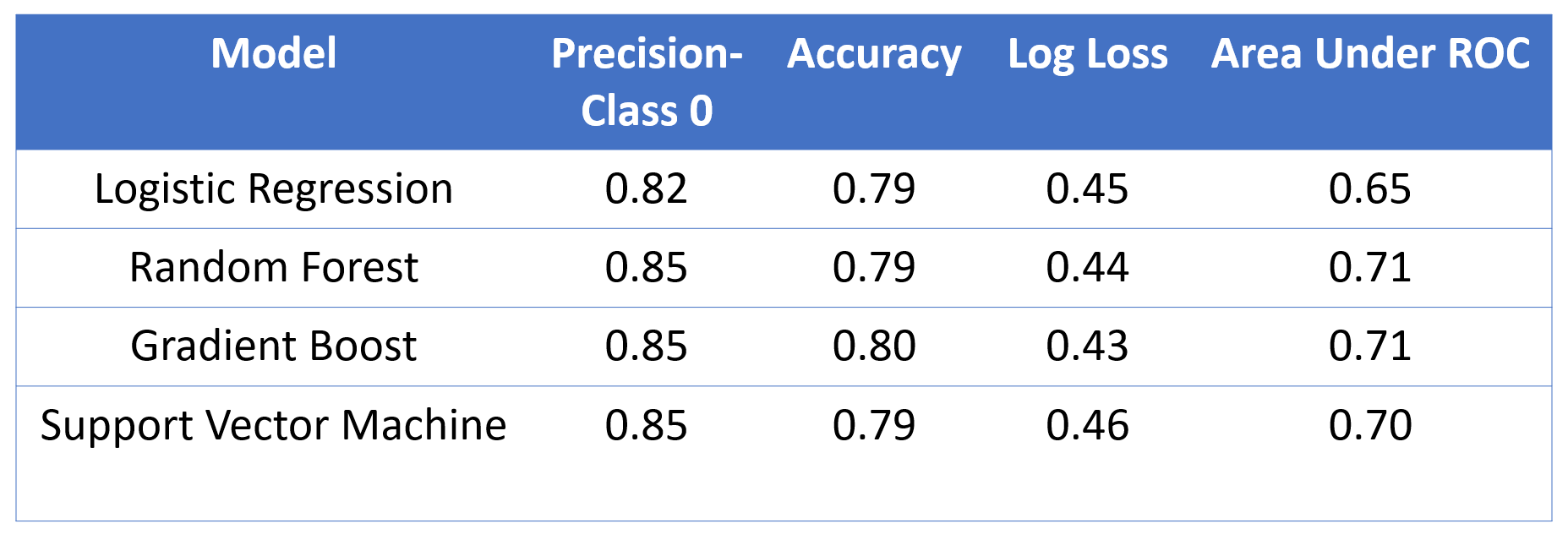
Below are the different hyperparameters we are going to try for each of the model.



After this process we get the best hyperparameters which we will then use to train our final model with.

Next, we do a stratified train-test split with a test ratio of 20%.

Below is a table which helps us compare the performance on the test results.



**Ensemble Model**

The next step is to try to improve the performance of the model. For this we create an Ensemble model. In this process we get the output probability of ‘0’ from all the models and then average them out to generate one output.  
This process helps us to reduce variance in the output.

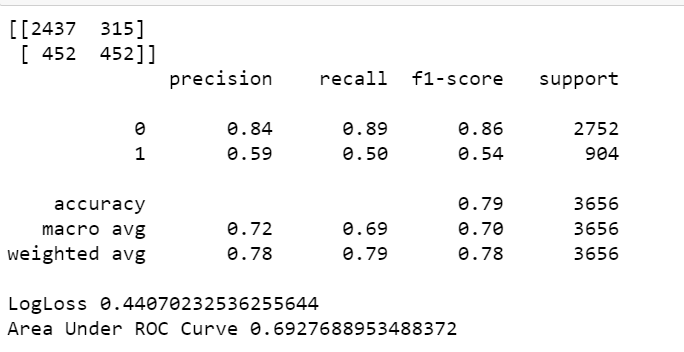
Below is the dataframe generated which contains all the results.   
The avg columns contains the final output probability for the person taking up the job. This is the probability for ‘0’

Table

Description automatically generated

The output variable will be ‘0’ if the ‘avg’ is greater than 0.5 and ‘1’ otherwise.

Below is the classification report of the on the test set with a default (0.5) threshold.



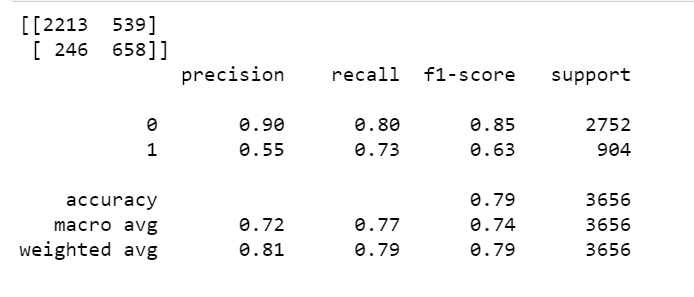
**Finding Optimal Threshold**

The next step to improve the overall performance is to find the optimal threshold.  
So for this we find the Area under the ROC curve for all possible thresholds.  
Chart, line chart

Description automatically generated

The best threshold we get is 0.76 and the corresponding Area Under the ROC Curve we get is 0.76.

Below is the Confusion Matrix and Classification Report for the ensemble model with the optimum threshold of 0.76



We have increased the precision for class 0 to 90% keeping the recall significantly high.

**Conclusion**

Creating the Ensemble model reduces the variance of the models. We identify the precision for Class-0 as the most important metric in our case. This is because we want to be very sure of who we are training. Training a person who will not join our company can be treated as a loss to the company. The number of training hours is identified as a very important variable in decision making.