Employee Future Prediction

**Introduction**

It is common knowledge that one of a company's objectives is to retain its staff. Employees are one of a company's most significant assets. Having a steady fluctuation of staff is expensive for businesses. After an employee leaves, there are additional costs associated with the recruitment process, which is accentuated if the prior employee was a valuable member of the organization. Even after hiring a new employee for the open position, it will take time for the new employee to provide value to the company and get familiar with the processes and duties.

It is vital to understand why employees quit companies and why they are dissatisfied with their working conditions, so that the company may take steps to address these issues in the future.

One of the most difficult issues many businesses confront is retaining qualified and diligent personnel. As a result, we can greatly lessen this problem by enhancing employee happiness and offering a desirable working environment.

**About the dataset**

**The features we are working with are the following:**

1.Education - Education of the employee - {Bachelors, Masters, Phd}.  
  
2.JoiningYear - Joining Year of the employee - {2012,2013,2014,2015,2016,2017,2018}.  
  
3.City - Working location of the employee - {Bangalore, Pune, New Delhi}.  
  
4.PaymentTier - PAYMENT TIER: 1: HIGHEST; 2: MID LEVEL; 3: LOWEST.   
  
5.Age - Age of the employee - {22-41}.  
  
6.Gender - Gender of the employee - {Male, Female}.  
  
7.EverBenched - Whether the employee has ever been KEPT OUT OF PROJECTS FOR 1 MONTH OR MORE - {'Yes', 'No'}.  
  
8.ExperienceInCurrentDomain - How much experience an employee has in current domain? – Years: {0-7)

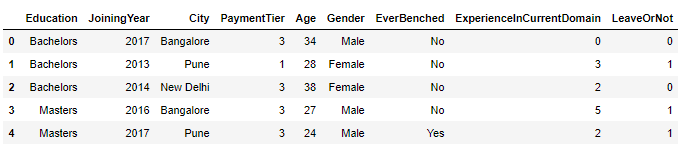
**What we want to accurately predict is whether or not people will leave the company judging by the before presented features:**

LeaveOrNot - Which employee will leave the organization? - {0,1}

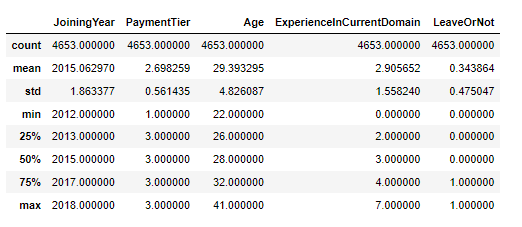
Shape of the data - (4653,9)

Our dataset contains both categorical and numeric features. We will talk about feature encoding in a later section.

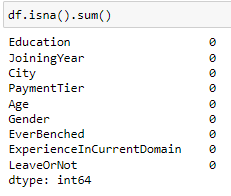
Below is a sample of how the dataset looks:



We have computed some basic statistics on the dataset:



We then checked for missing values to know how can we approach the dataset:



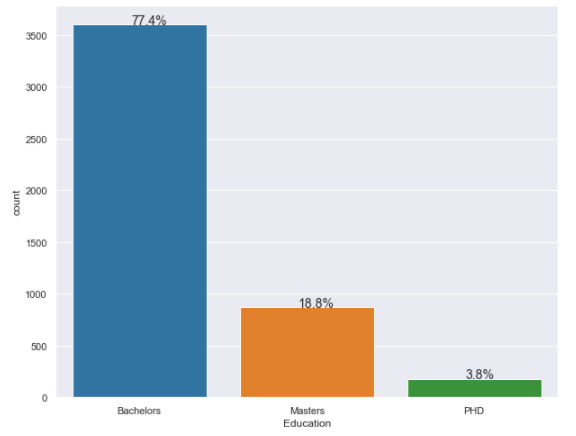
From the above picture we can deduct that we do not have any missing values in the dataset so we can work with it as it is.

**Univariate Analysis**

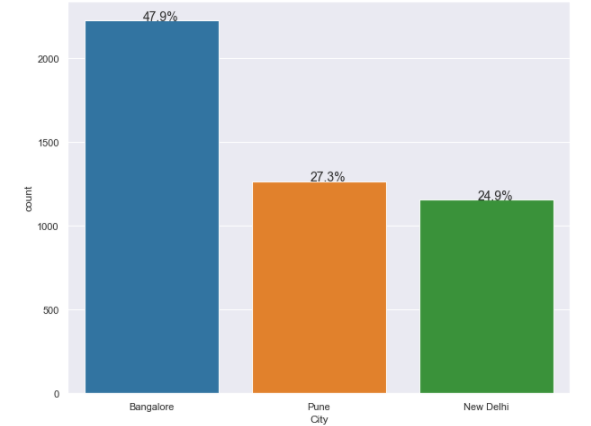
We have computed many plots to see how our features are distributed in the data set:

**Categorical features:**

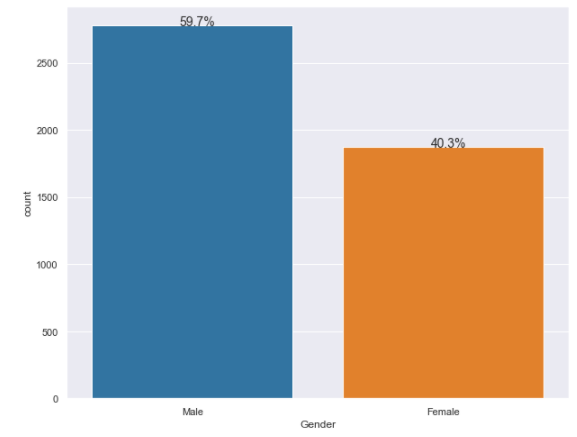
1. Education



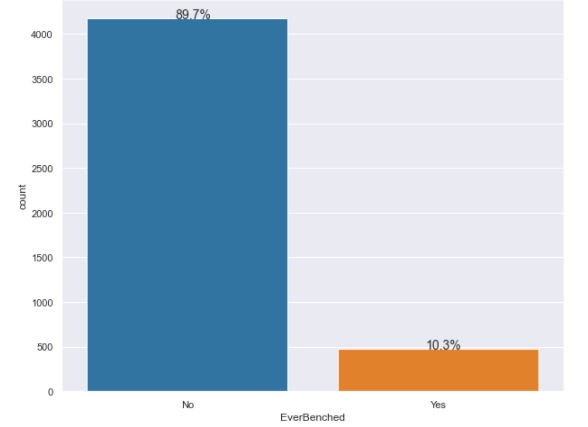
1. City



1. Gender

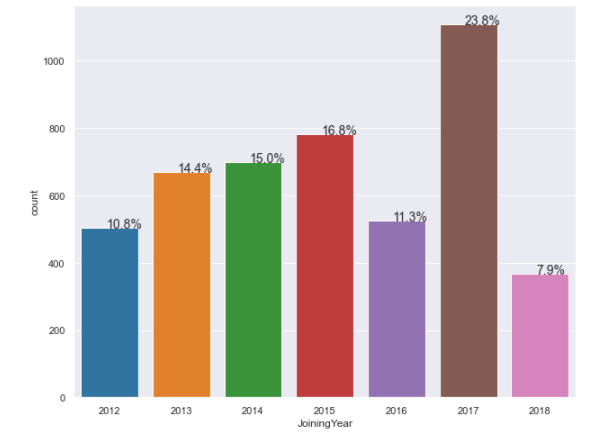


1. Ever Benched

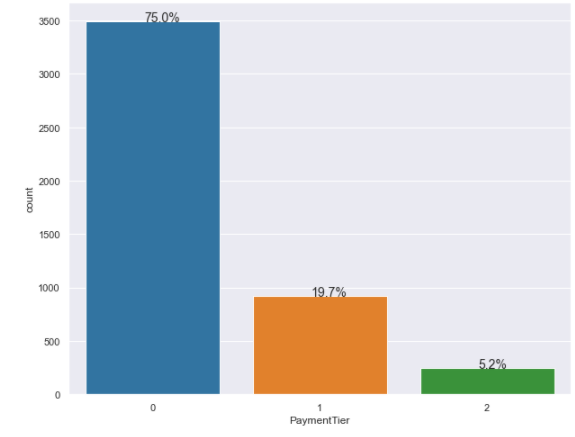


**Numeric features**

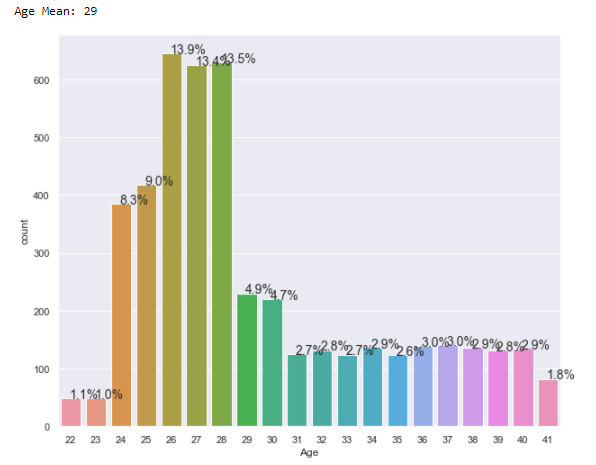
1. Joining Year



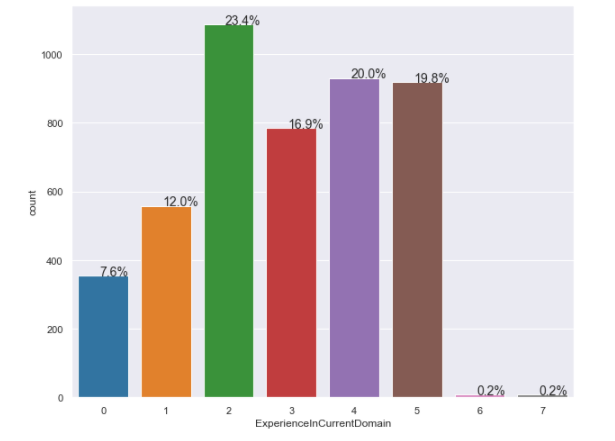
1. Payment Tier



1. Age

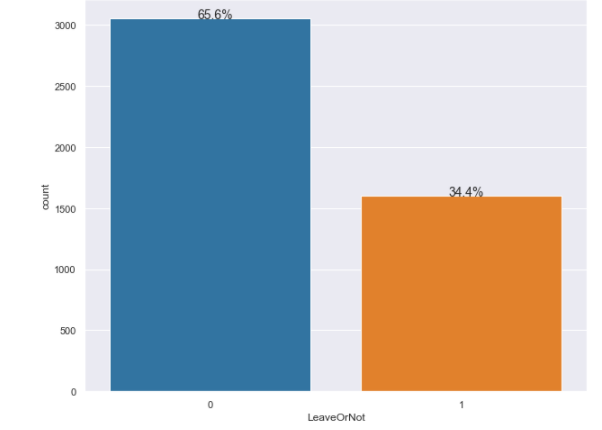


1. Experience in Current Domain



**Prediction values (numerical)**

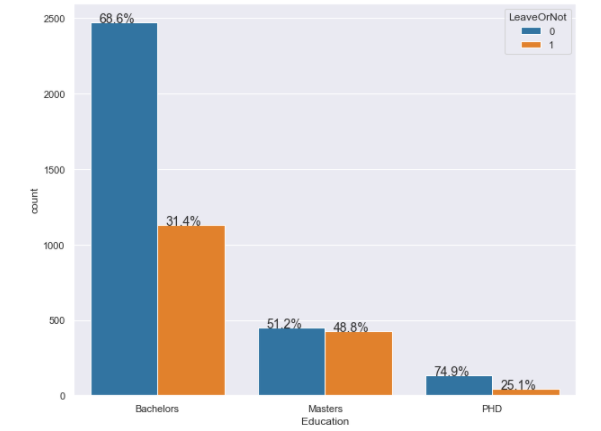
Leave or Not



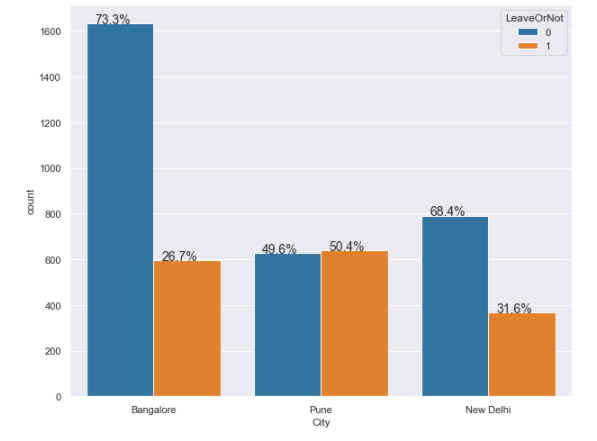
Bivariate Analysis

We have the computed plots for bivariate analysis to see the correlation between features in the dataset and prediction values:

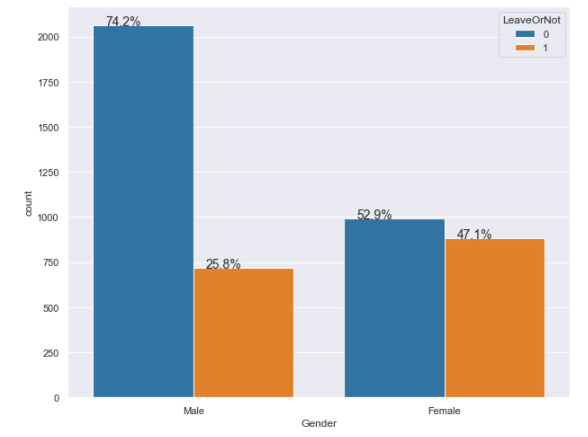
1. Education vs Leave or Not



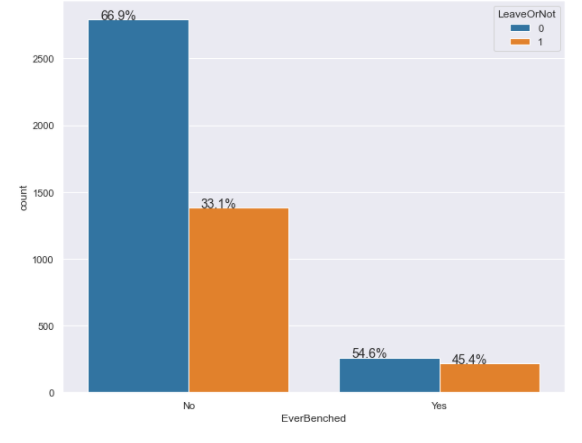
1. City vs Leave or Not



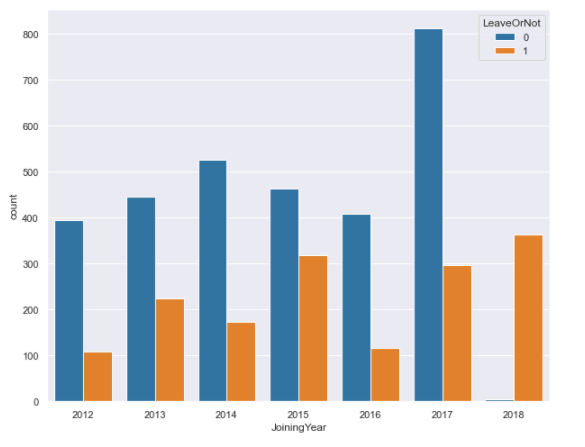
1. Gender vs Leave or Not



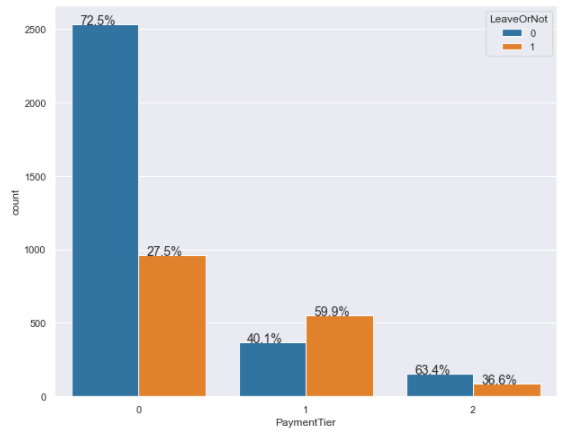
1. Ever Benched vs Leave or Not



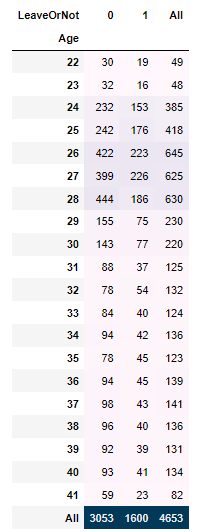
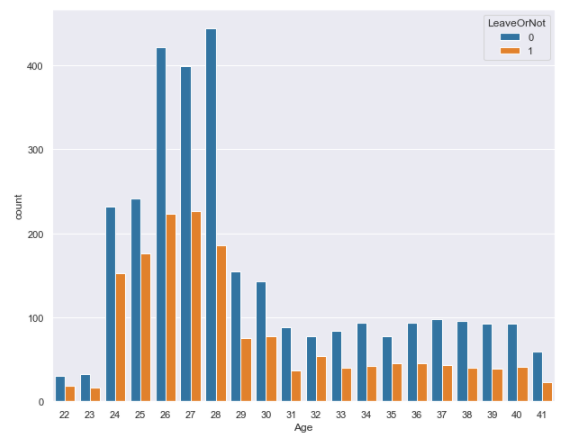
1. Joining Year vs Leave or Not

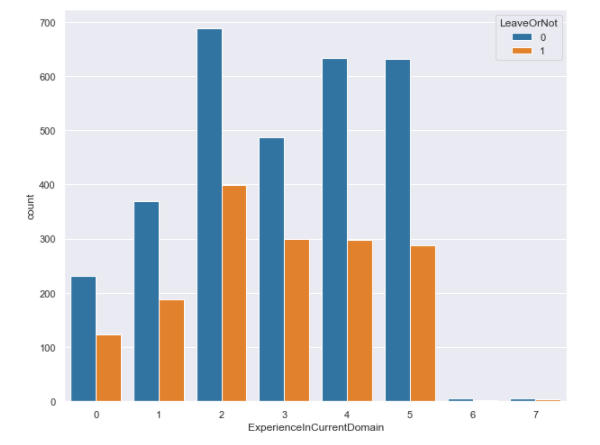


1. Payment Tier vs Leave or Not



1. Age vs Leave or Not

1. ExperiencevsLeaveorNot

Further, we have computed the correlation between features :



As it can be seen from this heatmap there are no meaningful correlations between features, as the values are very small.

**Feature Encoding**

To further use the features in predictive models we had to begin with feature encoding of categorical features.

We have used pandas.get\_dummies to convert some of the categorical variables (City, Gender and EverBenched) into dummy variables.

For Education we have converted the categorical values into ordered variables:

* {“Bachelor” : 0 , “Master”: 1, “PhD”: 2}

We have considered that only education variables should have ordered values.

We have tried to add some other features such as:

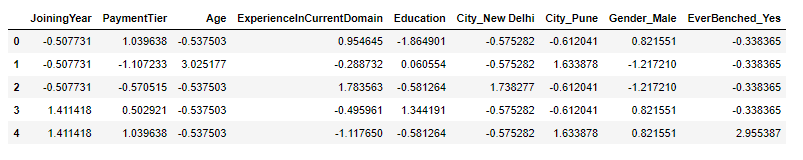
* Senority : to check if the experience in the current domain is more than 3 years
* IsRecentEmployee : to check if the employee came into the company before or after 2017 ( as in 2017 a large number of people got hired)

After testing the models both with and without the additional features we have concluded that they don’t bring any meaningful information and the models even performed worse.

Feature Scaling

We have used StandardScaler() from sklearn to normalize the range of independent features of the dataset.

Now our data look like this:



Feature Selection and Data Splitting

We have used train-test-splitting to evaluate performance of the models with test\_size = 0.2.

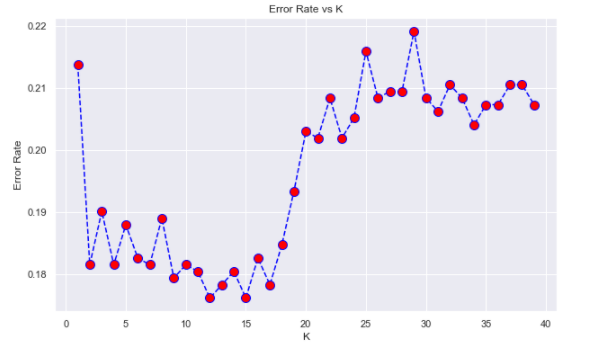
Model selection

For our Leave or Not classification we used the next models:

1. K-nearest neighbors
2. Decision Tree Classifie
3. Logistic Regression
4. Random Forest Classifier
5. Ada Boost Classifier
6. Bagging Classifier
7. Support Vector Classifier

For each model we have computed classification report ( precision, recall, f1-score, support) as well as confusion matrix, to check which model performs better on our dataset.

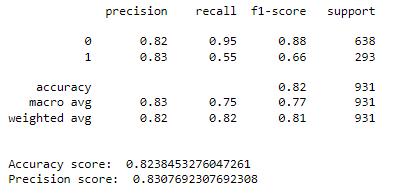
1. K-nearest neighbors

For KNN we have plotted the error rate by K, with K being in a range of 1 to 40.

* 1. Classification report

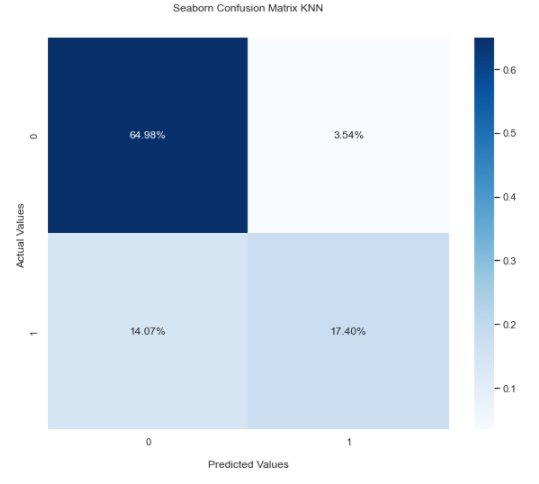
As we can see from the above plot with the error rate, the best option is to choose K= 12 or K=15. We have chose K=12.

1.1 Classification report

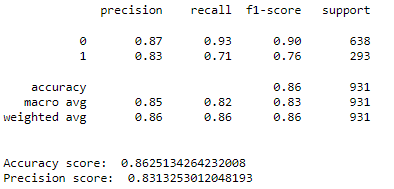




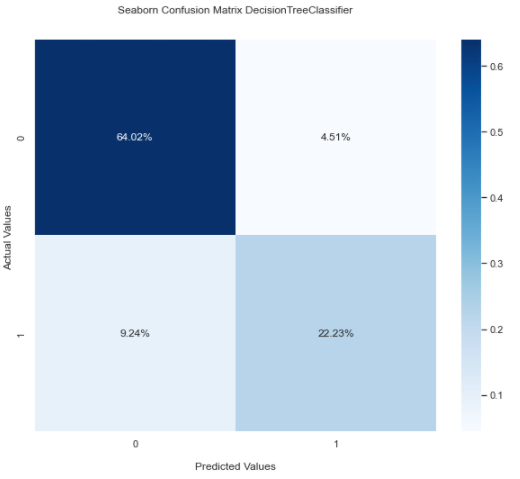
* 1. Confusion matrix



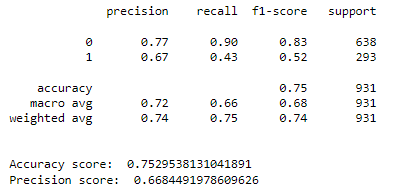
1. Decision Tree Classifier
   1. Classification Report



* 1. Confusion Matrix

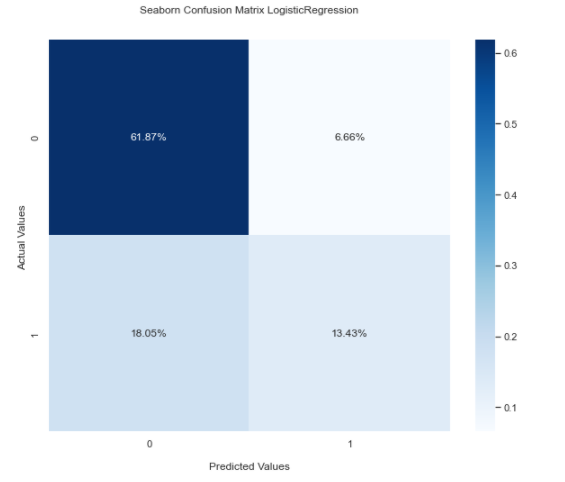


1. Logistic Regression

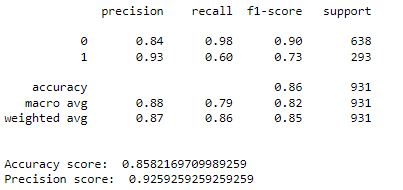
 3.1 Classification Report



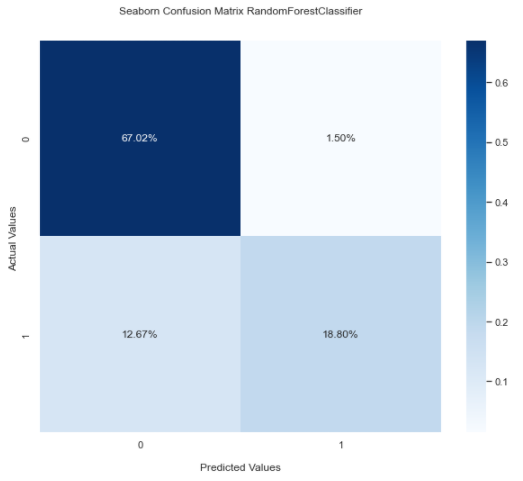
3.2 Confusion Matrix



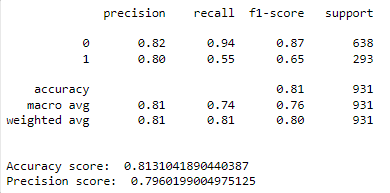
1. Random Forest Classifier
   1. Classification Report



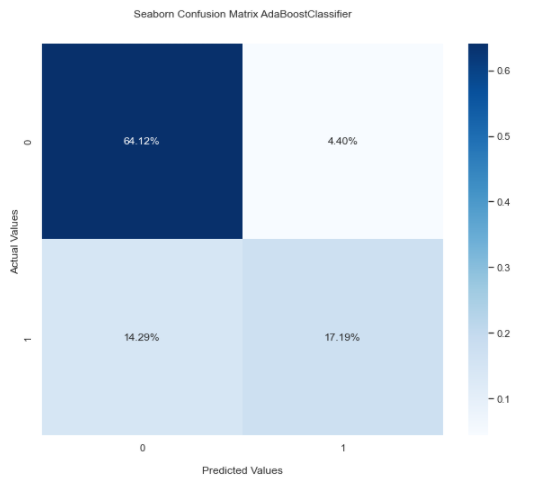
* 1. Confusion Matrix



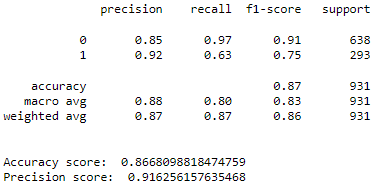
1. Ada Boost Classifier
   1. Classification Report



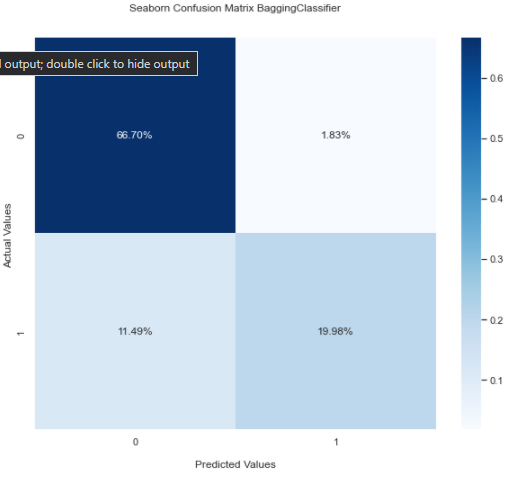
* 1. Confusion Matrix



1. Bagging Classifier
   1. Classification Report

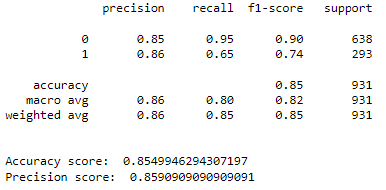


* 1. Confusion Matrix

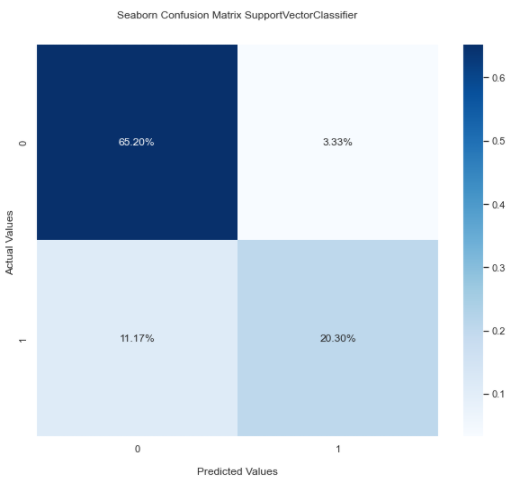


1. Support Vector Classifier

7.1 Classification Report

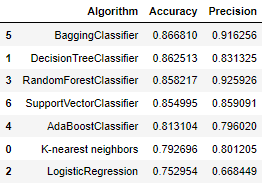


* 1. Confusion Matrix



**Accuracy and Precision Comparison**

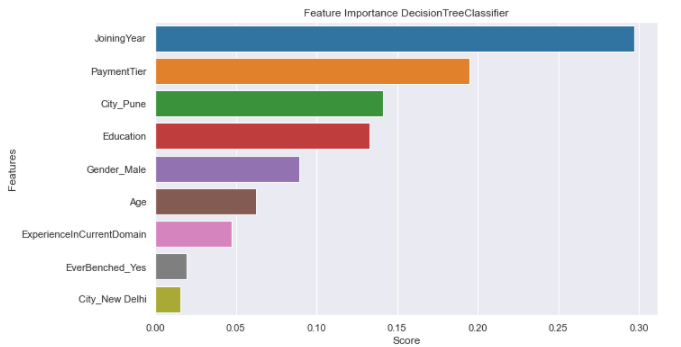
We have sorted all the models by Accuracy score and obtained the following:

****

As we can see, the model with the best Accuracy is Bagging Classifier, followed by Decision Tree Classifier. Models with the best Precision are Bagging Classifier, followed by Random Forest Classifier.

As an average of both quite good Accuracy and Precision, Bagging Classifier performed good.

In the end we wanted to also see the feature importance for some of our models:



Looks like Joining Year had an important role for all our model classifications.