Javiy Wang
STAT 724
12/16/20

Machine Learning Techniques In Hospital Readmissions

Introduction

It is perceived that the management of hyperglycemia in a diabetic patient has a critical effect on the result of whether they will be admitted to the hospital, as far as morbidity. The current analysis of a large clinical database was implemented to analyze past examples of diabetes care in patients with diabetes admitted to a US emergency clinic and to enlighten future bearings that may prompt advancements in a patient's safety. Specifically, we inspected the utilization of HbA1c as a marker of regard for diabetes care in countless people that have been previously diagnosed with other afflictions related to poor health. The HbA1c test, otherwise called the hemoglobin A1c test, is a significant blood test that gives a decent sign of how well your diabetes is being overseen. This test is additionally one of the principal manners by which type 2 diabetes is analyzed.

The purpose of this paper aims to see which variables are most important in predicting whether a patient will be readmitted to the hospital. Machine learning methods that will be used for this paper are Bagging, Decision Tree, Random Forest, and Gradient Boosting Machines. To evaluate the performance of each of my models, I will primarily be using 2 criteria to assess my models which are accuracy rate, and area under the curve (AUC). By comparing the accuracy rate and AUC of each of the models to one another, I will be able to obtain the best model in predicting readmission to the hospital.

Description of The Data

This study utilized the Health Facts information database, a public information warehouse that gathers complete clinical records across emergency clinics throughout the United States. Health Facts is a deliberate program offered to associations that utilize the Cerner Electronic Health Record System. The data set likewise utilizes the accompanying illustrative factors alongside all the missing information related to it.

Feature name	Type	Description and values	% missing
Encounter ID	Numeric	Unique identifier of an encounter	0%
Patient number	Numeric	Unique identifier of a patient	0%
Race	Nominal	Values: Caucasian, Asian, African American, Hispanic, and other	2%
Gender	Nominal	Values: male, female, and unknown/invalid	0%
Age	Nominal	Grouped in 10-year intervals: [0, 10), [10, 20), …, [90, 100)	0%
Weight	Numeric	Weight in pounds.	97%
Admission type	Nominal	Integer identifier corresponding to 9 distinct values, for example, emergency, urgent, elective, newborn, and not available	0%
Discharge disposition	Nominal	Integer identifier corresponding to 29 distinct values, for example, discharged to home, expired, and not available	0%
Admission source	Nominal	Integer identifier corresponding to 21 distinct values, for example, physician referral, emergency room, and transfer from a hospital	0%
Time in hospital	Numeric	Integer number of days between admission and discharge	0%
Payer code	Nominal	Integer identifier corresponding to 23 distinct values, for example, Blue Cross/Blue Shield, Medicare, and self-pay	52%
Medical specialty	Nominal	Integer identifier of a specialty of the admitting physician, corresponding to 84 distinct values, for example, cardiology, internal medicine, family/general practice, and surgeon	53%
A1c test result	Nominal	Indicates the range of the result or if the test was not taken. Values: “>8” if the result was greater than 8%, “>7” if the result was greater than 7% but less than 8%, “normal” if the result was less than 7%, and “none” if not measured.	0%
Change of medications	Nominal	Indicates if there was a change in diabetic medications (either dosage or generic name). Values: “change” and “no change”	0%
Diabetes medications	Nominal	Indicates if there was any diabetic medication prescribed. Values: “yes” and “no”	0%
24 features for medications	Nominal	For the generic names: metformin, repaglinide, nateglinide, chlorpropamide, glimepiride, acetohexamide, glipizide, glyburide, tolbutamide, pioglitazone, rosiglitazone, acarbose, miglitol, troglitazone, tolazamide, examide, sitagliptin, insulin, glyburide-metformin, glipizide-metformin, glimepiride-pioglitazone, metformin-rosiglitazone, and metformin-pioglitazone, the feature indicates whether the drug was prescribed or there was a change in the dosage. Values: “up” if the dosage was increased during the encounter, “down” if the dosage was decreased, “steady” if the dosage did not change, and “no” if the drug was not prescribed	0%
Readmitted	Nominal	Days to inpatient readmission. Values: “<30” if the patient was readmitted in less than 30 days, “>30” if the patient was readmitted in more than 30 days, and “No” for no record of readmission.	0%

Evidently there are a lot more variables I will just not mention until they are relevant.

Data Cleaning

The original database contains incomplete, redundant, and noisy information as expected in any real-world data. Before I ran any exploratory analysis I wanted to do some data cleaning. The “readmission” variable is a multinomial response variable that takes into account if they

were admitted within 30 days, after 30 days, or never readmitted. Since I only cared about whether a person was readmitted to the hospital or not; I recoded the instances of >30 and <30 to be YES=1, regardless of when an individual was readmitted, NO was changed to 0, and lastly if the observation said none it would be changed to -1. Then I also changed many of the other variables such as change, gender, diabetesMed, A1Cresult, and max_glu_serum into 1's and 0's.

When reading over the recorded data it was shown that weight, payer code, and medical specialty had 97%, 54%, and 53% missing data respectively. I decided there was not enough data for those columns to get any use out of it, thus I removed all 3 columns from the data frame. Next, I had to remove certain observations. For instance, if gender was never answered which only happened 3 times it was removed. I also removed any row of data where dsicharge_desposition_id = 11 since that meant the patient died during that visit to the hospital. Lastly, I got rid of any '?' observations in the diag_1 column; I ignored the values of diag_2/diag_3 since they were not the primary diagnosis.

The data also contains 3 different variables: number of inpatient, number of outpatient, and emergency visits. These variables essentially count how many hospital services an individual would receive in a year. Therefore, I believe merging all 3 columns into a single variable called "total visits" would reduce redundancy, thus, giving better results..

Another change I made to the data was counting the number of medication changes made for each individual. The data frame had already separated the 24 different medications into 23 dummy variables to indicate if the patient ever used certain medications or switched their medications during their stay in the hospital. Within the medical journals I have read associated with this dataset, the paper indicated that any changes in medication for a diabetic patient upon admission is negatively correlated to readmission rates. As a result, I decided to count the

number of medication changes made in total for each patient. The reason why I did this is because I wanted to simplify the model by having a single numeric description for 24 categorical variables, which inherently simplifies the model and it might discover correlations between the readmitted variable. I also noticed that for two columns named citoglipton and examide all records showed that they had the same value of "none" so eventually these columns are dropped.

Much like the 24 different medications a diabetic person can be on there was one variable that has over 900 unique ICD codes associated with it, and this was diag_1. Understanding the diagnosis is incredibly important, however, with the sheer number of unique codes it would be incredibly difficult to include all of them in the model and have it also be meaningful. Therefore, I collapsed all 900 different codes to be categorized within 9 different categories: *Circulatory, Respiratory, Digestive, Diabetes, Injury, Musculoskeletal, Genitourinary, Neoplasms, and Others*.

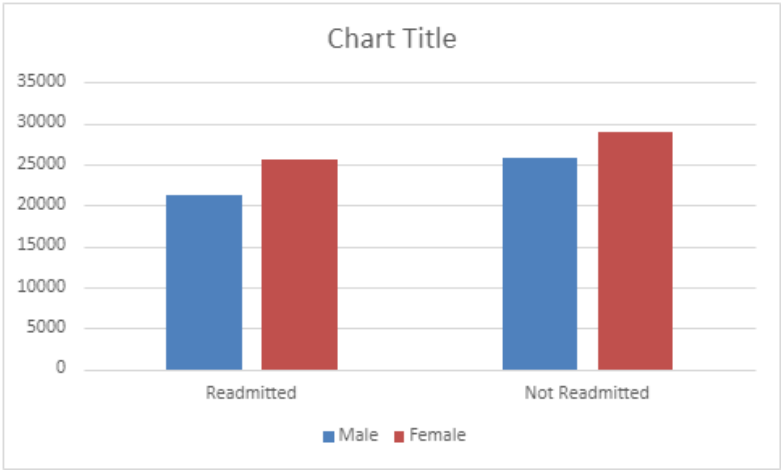
Group name	icd9 codes
Circulatory	390–459, 785
Respiratory	460–519, 786
Digestive	520–579, 787
Diabetes	250.xx
Injury	800–999
Musculoskeletal	710–739
Genitourinary	580–629, 788

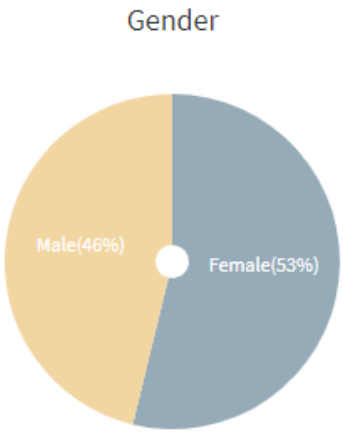
Neoplasms	140–239
Other (17.3%)	780, 781, 784, 790–799
	240–279, without 250
	680–709, 782
	001–139
	290–319
	E–V
	280–289
	320–359
	630–679
	360–389
	740–759

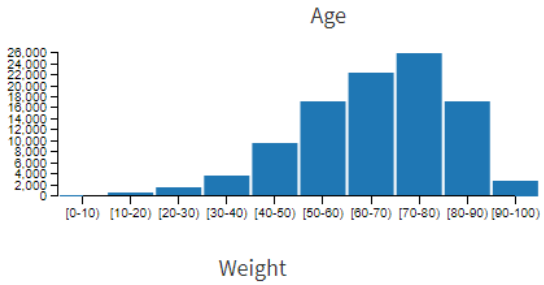
Exploratory Analysis

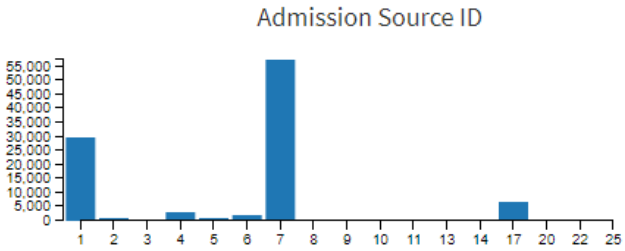
First I wanted to explore the data visually to see if there are any patterns with it and the response variable.

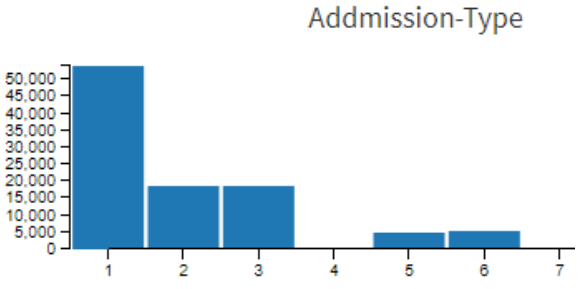
Gender vs Readmitted

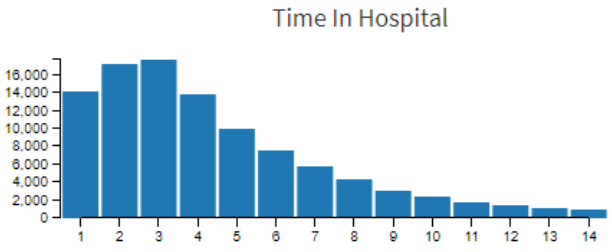












Most of the more important variables I wanted to analyze were already provided from the medical journal, and if they were not already provided I made them with Excel. I hypothesized that gender may play a large role in determining whether a person had diabetes since most of the

people I knew that had diabetes were male. However, as you can probably tell from the first two graphs gender was not significant in determining if a person was going to be readmitted. This was mostly due to a higher percentage of females being admitted to the hospital. As a result, it led to more females being readmitted as well as more females not being readmitted.

The next variable I wanted to look at was age. The histogram shows us that a major percentage of people being admitted to the hospital were older rather than younger. It is probably no surprise that as you age more health disparities come up compared to that of youths. The reason why the histogram dips after its peak from 70-80 is probably because after the age of 80 most individuals are deceased which makes it impossible to be admitted into the hospital in the first place.

The last 3 variables I wanted to look at were admission source id, admission type, and time spent in the hospital. I thought that even before making my models these variables would show patterns that would prove useful when I got to modeling the data. When I came to days spent in the hospital, the graph came out to be positively skewed. This meant that most people were released within the first few days of being in the hospital, which led me to the conclusion that most people admitted to the hospital did not have very serious medical complications. However, from the bar graph for admission type you can see that the first type was the most common reason why people were admitted, which was circulatory complications. After doing some research, many medical journals stated most diseases that lead to circulatory issues are highly treatable, since those kinds of problems would indicate the disease is in a progressive state, aka non-life threatening.

Models: Logistic, Bagging, Decision Trees, Random Forest, GBM

Method	Accuracy	Precision	CV-5 Fold	AUC
Random Forest	0.66	0.66	0.65	0.6
Decision Tree	0.63	0.54	0.62	0.6
Bagging	0.67	0.61	0.66	0.64
Boosting	0.67	0.62	0.7	0.72

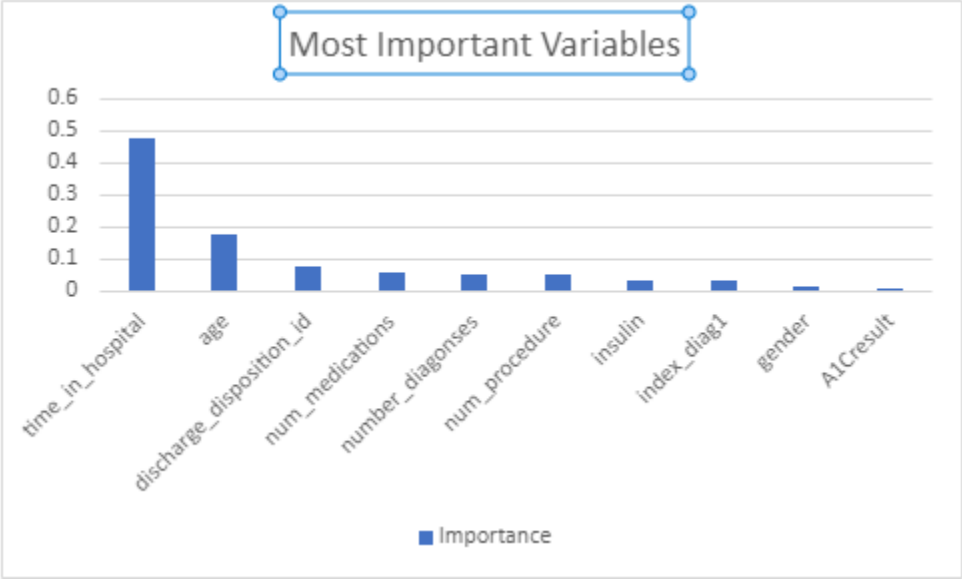
Logistic Regression

I began modeling using a logistic regression model to predict whether a person would be readmitted back to the hospital given the explanatory variables: time_in_hospital, age, discharge_disposition_id, num_medications, number_diagnoses, num_procedures, insulin, index_diag1, and A1C result. There were many reasons why I did not include this method on my final table nor is the code on my jupyter notebook. First, the main disadvantage is that this method cannot properly handle the problems of nonlinearity in the predictors or interactive effects of explanatory variables. Secondly, the logistic regression model's accuracy, precision, recall, and AUC were all worse than my other four models. Nevertheless, this model set the foundation for my other models and it did provide useful information in regards to variable importance.

Tree based methods - Decision Tree and Random Forest

Next I used 2 different tree-based methods to analyze the data set. Decision trees are a usually utilized method for classification issues, which is exactly why I used this method. The great thing about using this method is that it attempts all kinds of interactions between my explanatory variables. When I obtained all my scores from this method it was a slight improvement in all metrics compared to logistic regression. Much like logistic regression it also thought all the explanatory variables from logistic regression were important in addition to the gender variable. From the graph below you can probably tell that time spent in the hospital had

the highest relative importance compared to the other 45 possible variables. These numbers were calculated by comparing the performance of the model with and without a given variable. As a result, a model with the variable age removed from the model would result in us losing predictive power.



Random forests are made up of randomly grown trees. Contrary to a regular decision tree, this method relies on trying many different trees with randomly assigned subsets of explanatory variables, rather than relying on an individual decision tree. My final prediction is then calculated by voting across predictions made by all the trees in the forest. After applying random forest I got an accuracy score of 0.66 regardless of the criterion being gini or entropy.

The last models I tried were bagging and boosting. Bagging and Boosting decrease the variance of a single estimate as they combine several estimates from the different models they create. So the resulting model tends to have higher stability. As you can see from my metrics chart from above the boosting method was a better model and for reasons beyond just the better metrics provided from the table. The problem with bagging here is that my model's performance was already lacking in accuracy and precision, and if my model was already suffering from low

performance, then bagging seldom gets me a better bias let alone a better model. At the end of the day it was not surprising that the boosting model outperformed the bagging one since boosting tries to reduce bias and it also tends to overfit the model.

Conclusion

In this paper, by using various machine learning methods such as logistic regression, decision tree, random forest, bagging, and boosting, the prediction of readmission is performed and assessed. The boosting method with all explanatory variables in the model achieved the best results in Accuracy, Cross Validation, and AUC. Due to the time constraints for this paper, I was not able to fully clean the data and perform more data manipulation in order to have cleaner and more interpretable data to fit the models. Further analysis with new parameters by transforming, interaction, or normalization could have probably resulted in better results for all of the models of each of the given methods I used above. All things considered, I believe that if given the chance to work in a group, my group mates could hopefully provide more insight and possibly provide better techniques to manipulate the data frame or provide a better combination of explanatory variables that could make my model's accuracy and precision metrics better than they currently are.

I was fairly upset at the AUC and accuracy scores across all my models. I genuinely believed that I had prepped my data set enough so that the performance across all my models would be greater than .75, however this was not the case. My best model was the boosting model and it only achieved an accuracy score of 0.67 and an AUC of 0.72. In conclusion, further investigation is warranted if I want to build a better prediction model.

Graphs And Scores For All The Models

Random Forest

In [84]:
```
from sklearn.ensemble import RandomForestClassifier
```

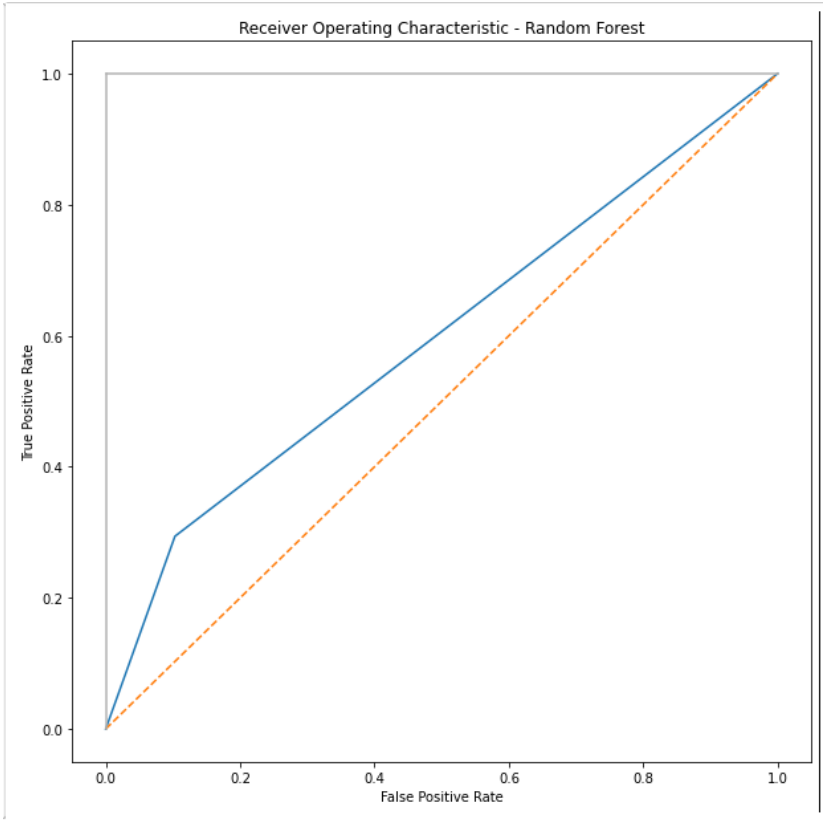
In [86]:
```
clf = RandomForestClassifier(n_estimators=500, max_depth=25, min_samples_leaf=1, random_state=0)
clf = clf.fit(X_train, Y_train)
```

In [87]:
```
print("Cross Validation score: {}".format(np.mean(cross_val_score(clf, X_train, Y_train, cv=5))))
```

```
Cross Validation score: 0.6482155229031312
```

In [88]:
```
Y_test_predict = clf.predict(X_test)
pd.crosstab(pd.Series(Y_test, name = 'Actual'), pd.Series(Y_test_predict, name = 'Predict'), margins =
print("Accuracy is {}".format(accuracy_score(Y_test, Y_test_predict)))

print("Precision is {}".format(precision_score(Y_test, Y_test_predict)))

print("AUC is {}".format(roc_auc_score(Y_test, Y_test_predict)))
```

```
Accuracy is 0.6559948889046638
Precision is 0.6551860649247823
AUC is 0.5954846595177653
```



Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, roc_auc_score,roc_curve
dte = DecisionTreeClassifier(max_depth=25, min_samples_split=100)
dte.fit(X_train, Y_train)
```

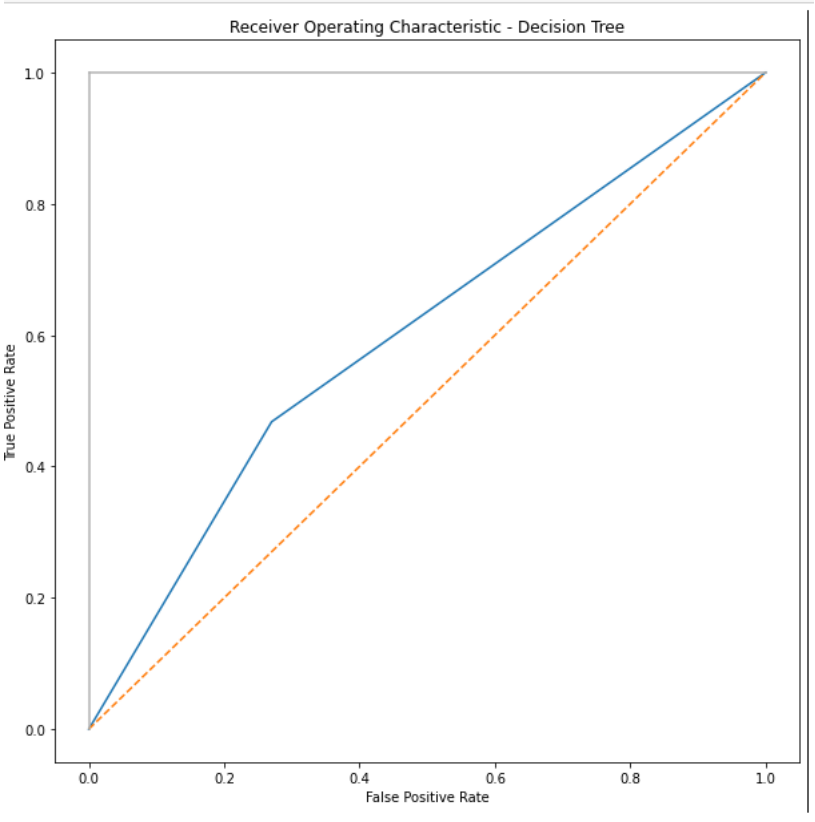
```
DecisionTreeClassifier(max_depth=25, min_samples_split=100)
```

```
from sklearn.model_selection import cross_val_score
print("Cross Validation score: {}".format(np.mean(cross_val_score(dte, X_train, Y_train, cv=5))))
```

```
Cross Validation score: 0.621169027318294
```

```
Y_test_predict = dte.predict(X_test)
pd.crosstab(pd.Series(Y_test, name = 'Actual'), pd.Series(Y_test_predict, name = 'Predict'), margins =
print("Accuracy is {}".format(accuracy_score(Y_test, Y_test_predict)))

print("Precision is {}".format(precision_score(Y_test, Y_test_predict)))

print("AUC is {}".format(roc_auc_score(Y_test, Y_test_predict)))
```

```
Accuracy is 0.6252573294526869
Precision is 0.5356779833299451
AUC is 0.5989781447930824
```



Bagging Classification

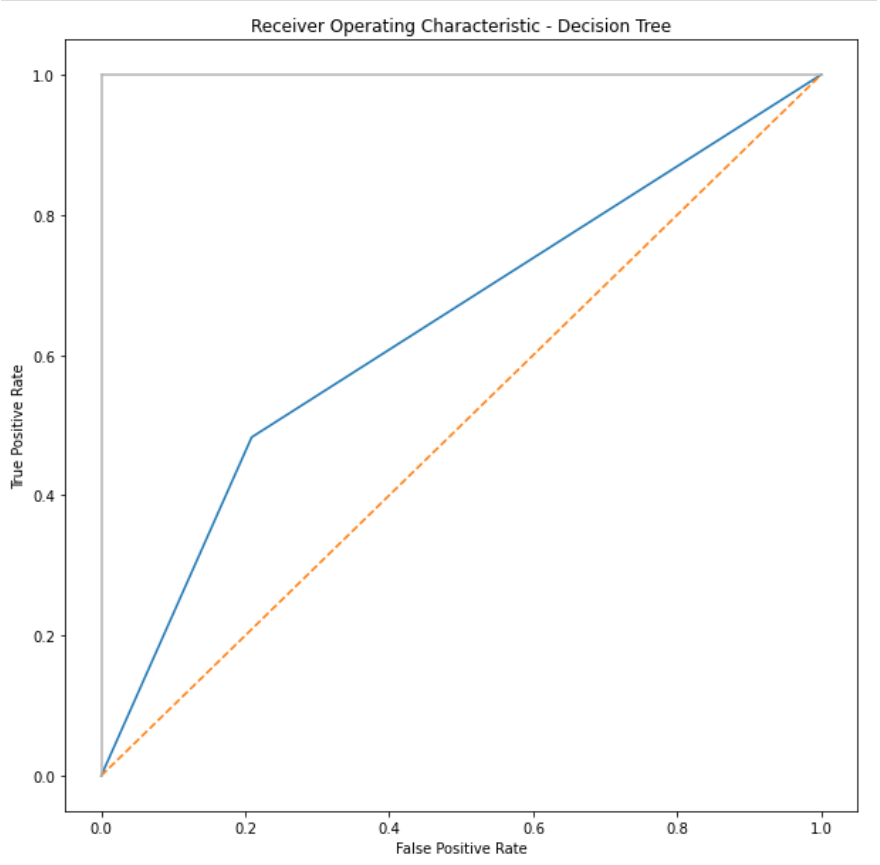
```
from sklearn.ensemble import BaggingClassifier
clf = BaggingClassifier(n_estimators= 500, random_state=0).fit(X_train, Y_train)
```

```
#Cross validation 5 folds
from sklearn.model_selection import cross_val_score
print("Cross Validation score: {}".format(np.mean(cross_val_score(clf, X_train, Y_train, cv=5))))
```

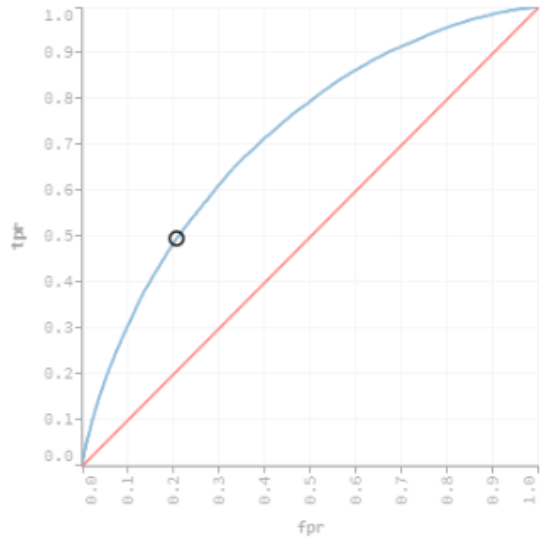
```
Cross Validation score: 0.6562195949768518
```

```
Y_test_predict = clf.predict(X_test)
print("Accuracy is {}".format(accuracy_score(Y_test, Y_test_predict)))

print("Precision is {}".format(precision_score(Y_test, Y_test_predict)))

print("AUC is {}".format(roc_auc_score(Y_test, Y_test_predict)))
```

```
Accuracy is 0.6678497905870661
Precision is 0.6060200668896321
AUC is 0.6369157650729602
```



ROC CURVE - TRAINING METRICS , AUC = 0.718017



Threshold: Choose...

Criterion: max accuracy

Selected mark(s):

threshold	0.4756
f1	0.5511
f2	0.5162
f0point5	0.5911
accuracy	0.6731
precision	0.6211
recall	0.4952
specificity	0.7943
absolute_mcc	0.3040
min_per_class_accuracy	0.4952
mean_per_class_accuracy	0.6448
tns	33281
fns	14403
fps	8619
tps	14131
tnr	0.7943
fnr	0.5048
fpr	0.2057
tpr	0.4952
idx	161

CM

Actual/Predicted	0	1	Error	Rate
0	33281	8619	0.2057	8619 / 41900
1	14403	14131	0.5048	14403 / 28534
Total	47684	22750	0.3269	23022 / 70434

ROC CURVE - CROSS VALIDATION METRICS , AUC = 0.699488