**Machine Learning Engineer Nanodegree**

**Capstone Project**

**Prediction of Loan Approval**

Lavanuru aparna

February 5th 2019

**I. Definition**

**1.1Project Overview**

In these days the domain of banking is stepping to get engaged with artificial intelligence and machine learning. This project is based on domain of banking. This gives accurate results regarding the classification of loan approval. It takes the data from the Kaggle machine learning

<https://www.kaggle.com/hafidhfikri/loan-approval-prediction/data>

The aim of this project is to predict whether the loan can be approved for the individual or not.

**1.2Problem Statement:**

The main aim of my project is to predict the loan approval. For doing this I selected the dataset from Kaggle (<https://www.kaggle.com/hafidhfikri/loan-approval-prediction/data>). So my goal is to predict the loan approvals. Here I am using classification models to find the accuracy of each model and select the best model with high accuracy to predict the approvals. Here the input parameters are training data that we took and the output will be whether to approve the loan or not.

The tasks involved in it are:

1. Download the data

2. Cleaning the data and removing the null data points, duplicated data rows.

3. Visualizing the data.

4. Split the data and train and test which classifier performs good on the dataset based on the evaluation metric we have chosen.

5. Choose the best classifier that gives the best accuracy scores.

**1.3 Metrics:**

I want to use accuracy score as an evaluation metric for the prediction of loan approval. In the dataset the class labels (Y,N) are very closely balanced so we can use accuracy score as the evaluation metric. Here I am predicting the accuracy score of the selected models. We will select a model whose accuracy score is greater than all the other models and we treat it as the best.

For finding out the accuracy we have the following formula:

Accuracy Score= TP+TN/TP+FP+FN+TN

True Positives: are the values which are correctly predicted as positives True Negatives: are the values which are correctly classified as negatives.

False Positives: are the values which are wrongly classified as positives. These are also type-1 errors.

False Negatives: are the values which are wrongly classified as negatives. These are also called as type-2 errors.

**II. Analysis**

**2.1 Data Exploration**

This file concerns loan applications. All attribute names and values have been changed to meaningless symbols to protect confidentiality of the data. This dataset is interesting because there is a good mixture of attributes – Categorical, integer and real valued attributes. There are also few missing values. On a total there are 13 attributes.

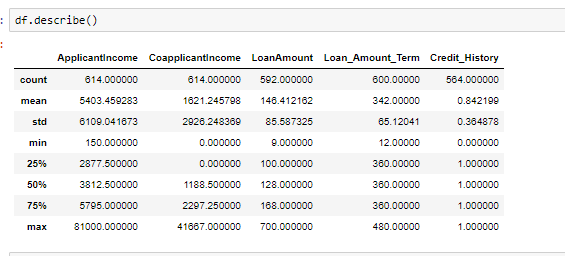
The total number of instances in the datasets is 615. The characteristics of the dataset are multivariate. Here there are mainly two classes: Y(approving credit card) and N(not approving credit card).

There are 422 Y’s and 192 N‘s which are closely balanced. For the best result we will split the data into training set and testing set. On a whole we will assign 70% of the data to training set and 30% of the data to testing set. The information about the data is a follows

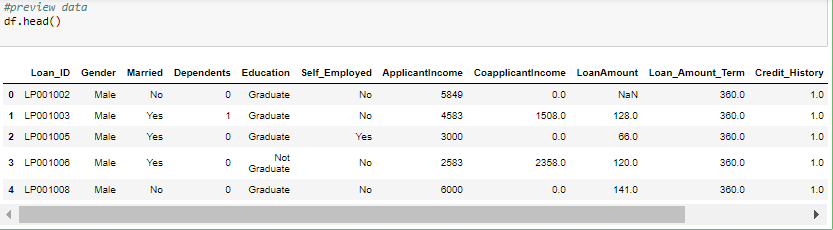
The attributes are:

* **Loan\_ID :-**  Unique Loan ID
* **Gender** :- Male/ Female
* **Married** :- Applicant married (Y/N)
* **Dependents:-**  Number of dependents
* **Education** :- Applicant Education (Graduate/ Under Graduate)
* **Self\_Employed:-**  Self employed (Y/N)
* **ApplicantIncome:-** Applicant income
* **CoapplicantIncome:-**  Coapplicant income
* **LoanAmount:-**  Loan amount in thousands
* **Loan\_Amount\_Term** :- Term of loan in months
* **Credit\_History** :- credit history meets guidelines
* **Property\_Area** :- Urban/ Semi Urban/ Rural
* **Loan\_Status :-**  Loan approved (Y/N)

The descriptive stats of the data are as follows:



A snapshot of the dataset containing first 5 rows is as follow:



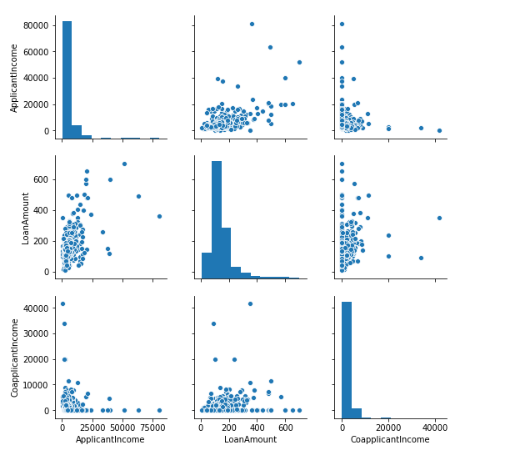
**2.2 Exploratory Visualization**:

**Pair plot**:

Pair plot is used to understand the best set of features to explain a relationship between two variables or to form the most separated clusters. It also helps to form some simple classification models by drawing some simple lines or make linear separation in our dataset.

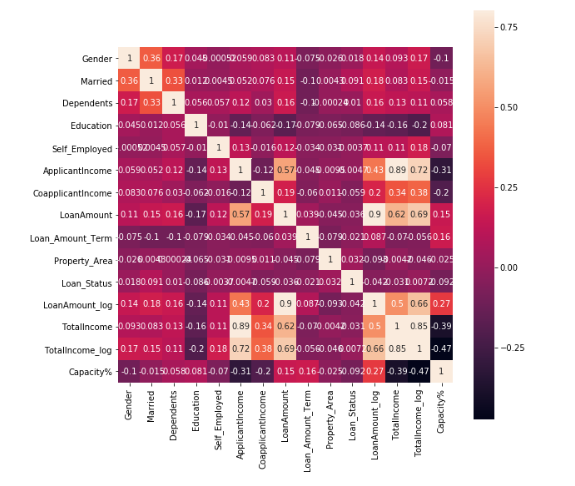
A Simple 2D plots is used to understand the relationship or pattern between two variables or dimensions in our dataset. A 3D plot will be used for three variables or dimensions. However, what would be do if we have more than 3 dimensions or features in our dataset as we humans do have the capability to visualize more than 3 dimensions. One solution to this problem is pair plots. It is one of the most effective starting tools. Here we will understand the relation between the numerical attributes (7,8,9) The pair plot we got by doing so is as follows:

From the below pair plot we cannot clearly visualize the correlation between the numerical attributes so we will go with the heatmap to know clearly about the correlation.

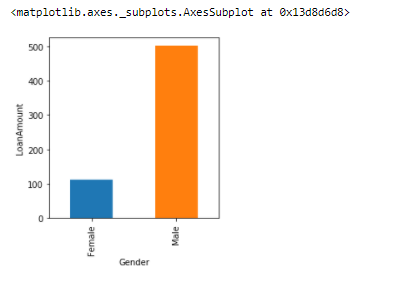


**Heatmap:**

The heatmap is a 2-D representation of data in which values are represented by colours. A simple heatmap provides the immediate visual summary of information. More elaborate heatmaps allow the user to understand complex data.

****

The insights observed from the above plots I analysed that there is a lot of unbalanced data that means some features have missing data. From the above heatmap the data distribution is clearly visualized when compared to pairplot. By using the visualization we can get clear idea of which features are essential to make loan approval predictions and which of them is having the highest mean by having shades in the visuals like darker and lighter shades and it also have max value having 0.8 .We may change the max value to 1 also in order to get clear visualizations.



Also the above plots helps to analyse that the gender features has more number of males applied for loans and less number of females applied for loans .

My conclusions are to decide the missing data is filled with majority people

The model gradient boosting having ensemble family having the potential challenges that can strength the weak learners to strong learners and make the features strong

**2.3 Algorithms and techniques** :

Three supervised learning approaches are selected for this problem. Care is taken that all these approaches are fundamentally different from each other, so that we can cover as wide an umbrella as possible in term of possible approaches. For example- We will not select Random Forest and Gradient Boosting together as they come from the same family of ‘ensemble’ approaches.

The choice of algorithms was influenced from these source:

https://stackoverflow.com/questions/2595176/which-machine-learning-classifier-to-choose-ingeneral

For each algorithm, we will try out different values of a few hyper parameters to arrive at the best possible classifier. For these dataset we apply the different supervised learning algorithms like knn , svm, random forest, decision tree, logistic regression, but Gradient boosting builds the model in a stage-wise fashion like other boosting methods do and it generalizes them by allowing optimization of an arbitrary differentiable loss function and boost the performance even there are more features to train and strength the weak learners and also changing the parameters also doesn’t have much effect and remaining are:

**GradientBoosting:**

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.

**Advantages and Disadvantages:**

* Random forests are much easier to train. Generally they have two tuning parameters mtry and ntrees.
* Mtry  is number of variables chosen randomly from the set of input variables and ntrees is number of trees to grow. Now default value of mtry in R( sqrt of n) generally gives pretty good results. As for ntree, you can increase the value of variable and it will lead to better fit( More number of trees means less variance).
* Now a days with so much computing power, even training a Random Forest with even 5000 trees is very easy. So essentially there is only one parameter to tune mtry which with its default value gives good results
* Gradient Boosted Methods generally have 3 parameters to train shrinkage parameter, depth of tree, number of trees.
* Now each of these parameters should be tuned to get a good fit. And you cannot just take maximum value of ntree in this case as GBM can overfit fit higher number of trees. But if you are able to use correct tuning parameters, they generally give somewhat better results than Random Forests.

So Gradient based methods generally give better results but they are harder to fit than Random Forests.

**Parameters:**

class sklearn.ensemble*.*GradientBoostingClassifier*(*loss=’deviance’*,*learning\_rate=0.1*,*n\_estimators=100*,*subsample=1.0*,*criterion=’friedman\_mse’*,*min\_samples\_split=2*,*min\_samples\_leaf=1*,*min\_weight\_fraction\_leaf=0.0*,*max\_depth=3*,*min\_impurity\_decrease=0.0*,*min\_impurity\_split=None*,*init=None*,*random\_state=None*,*max\_features=None*,*verbose=0*,*max\_leaf\_nodes=None*,*warm\_start=False*,*presort=’auto’*,*validation\_fraction=0.1*,*n\_iter\_no\_change=None*,*tol=0.0001*)*

Procedure involved:

In my project Gradient Boosting gives better accuracy and gives better performance when compared to other models because it is used to strengthen the weak learners and makes them strong while other models don’t have that capacity when compared to gradient boosting.

In my project I tuned many parameters but the performance for gradient boosting doesn’t decreases but random forest and decision trees performance goes on varying. So based on these I conclude GradientBoosting is the best model helps to predict loan approvals

**Random Forest:**

Tree models are known to be high variance, low bias models. In consequence, they are prone to overfit the training data. This is catchy if we recapitulate what a tree model does if we do not prune it or introduce early stopping criteria like a minimum number of instances per leaf node.

Well, it tries to split the data along the features until the instances are pure regarding the value of the target feature, there are no data left, or there are no features left to split the dataset on. If one of the above holds true, we grow a leaf node. The consequence is that the tree model is grown to the maximal depth and there with tries to reshape the training data as precise as possible which can easily lead to over fitting.

Another drawback of classical tree models like the (ID3 or CART) is that they are relatively unstable. This instability can lead to the situation that a small change in the composition of the dataset leads to a completely different tree model.

**Advantages:**

* Reduction in over fitting: by averaging several trees, there is a significantly lower risk of over fitting.
* Less variance: By using multiple trees, you reduce the chance of stumbling across a classifier that doesn’t perform well because of the relationship between the train and test data.

**Disadvantages :**

* It takes more time to train samples.

**Parameters:**

Class sklearn.ensemble.RandomForestClassifier (n\_estimators=’warn’, criterion=’gini’, max\_depth=None, min\_samples\_split=2, min\_samples\_leaf=1, min\_weight\_fraction\_leaf=0.0, max\_features=’auto’, max\_leaf\_nodes=None, min\_impurity\_decrease=0.0, min\_impurity\_split=None, bootstrap=True, oob\_score=False, n\_jobs=None, random\_state=None, verbose=0, warm\_start=False, class\_weight=None

**Decision TreesClassifier:**

The decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions.Though a commonly used tool in data mining for deriving a strategy to reach a particular goal, also widely used in machine learning.

**Advantages:**

Able to handle categorical and numerical data.

Doesn’t require much data pre-processing, and can handle data which hasn’t been normalized, or encoded for Machine Learning Suitability.

Simple to understand, visualize and interpret.

**Disadvantages:**

* Decision-tree learners can create over-complex trees that do not generalize the data well. This is called over fitting.
* Mechanisms such as pruning (not currently supported), setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.
* Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.
* The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree.
* This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement. There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.
* Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

Complex Decision Trees do not generalize well to the data and can result in overfitting.

* Unstable, as small variations in the data can result in a different

decision tree.

**Parameters**:

Class sklearn.tree.DecisionTreeClassifier(criterion=’gini’, splitter=’best’, max\_depth=None, min\_samples\_split=2, min\_samples\_leaf=1, min\_weight\_fraction\_leaf=0.0, max\_features=None, random\_state=None, max\_leaf\_nodes=None, min\_impurity\_decrease=0.0, min\_impurity\_split=None, class\_weight=None, presort=False

**Logistic Regression:**

Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

**Advantages:**

* Because of its efficient and straightforward nature, doesn't require high computation power, easy to implement, easily interpretable, used widely by data analyst and scientist.
* Also, it doesn't require scaling of features. Logistic regression provides a probability score for observations.

**Disadvantages:**

* Logistic regression is not able to handle a large number of categorical features/variables.
* It is vulnerable to over fitting.
* Also, can't solve the nonlinear problem with the logistic regression that is why it requires a transformation of non-linear features.
* Logistic regression will not perform well with independent variables that are not correlated to the target variable and are very similar or correlated to each other.

**Parameters:**

Class sklearn.linear\_model.LogisticRegression (penalty=’l2’, dual=False, tol=0.0001, C=1.0, fit\_intercept=True, intercept\_scaling=1, class\_weight=None, random\_state=None, solver=’warn’, max\_iter=100, multi\_class=’warn’, verbose=0, warm\_start=False, n\_jobs=None)

The application is classification oriented . So, techniques that are used are taken from Classification techniques.

**KNN:**

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry.

To evaluate any technique we generally look at 3 important aspects:

* Ease to interpret output
* Calculation time
* Predictive Power

**Advantages**:

* The cost of the learning process is zero
* No assumptions about the characteristics of the concepts to learn have to be done
* Complex concepts can be learned by local approximation using simple procedures

**Disadvantages**:

* The model cannot be interpreted (there is no description of the learned concepts)
* It is computationally expensive to find the k nearest neighbours when the dataset is very large
* Performance depends on the number of dimensions that we have (curse of dimensionality ) =⇒ Attribute Selection

**Parameters:**

class sklearn.neighbors.KNeighborsClassifier(n\_neighbors=5, weights=’uniform’ , algorithm=’auto’, leaf\_size=30, p=2, metric=’minkowski’, metric\_params=Non e, n\_jobs=None, \*\*kwargs)

**Support vector machine (svm):**

SVM aims to find an optimal hyper plane that separates the data into different classes, using a method called as kernel to project data points belonging to a particular class into different dimensions, so that a hyper plane can easily pass through and maintain the largest possible distance between itself and these data points.

**Advantages:**

Performs well with high dimensional data.SVM’s are very good when we have no idea on the data.

Works well with even unstructured and semi structure data like text, Images.

The kernel trick is strength of SVM.By using the kernel function to solve the complex problem.

The SVM model have generalization in practice, the risk of overfitting is less in SVM.

**Disadvantages:**

Choosing the good kernel function is not easy and it take long training time for large datasets.

Difficult to understand and interpret the final model, variable weights and individual

**Parameters:**

class sklearn.svm.SVC(C=1.0, kernel=’rbf’, degree=3, gamma=’auto\_deprecated ’, coef0=0.0, shrinking=True, probability=False, tol=0.001, cache\_size=200, clas s\_weight=None, verbose=False, max\_iter=- 1, decision\_function\_shape=’ovr’, random\_state=None)

**2.4 Benchmark Model:**

**Benchmark:-**

Standard, or a set of standards, used as a point of reference for evaluating performance or level of quality.

**Importance**:-

This information can then be used to identify gaps in an model in order to achieve a competitive advantage. Here we will consider SVM model as the benchmark model because we will find out the accuracy of this model and try to achieve the better accuracy than this model. It indicates the absolute fit of the model to the data–how close the observed data points are to the model's predicted values.

Benchmark model is a model which we will take as reference and achieve the best result than the benchmark model in our project we will take SVM as a benchmark model. Initially using SVM model we will achieve an accuracy of 68%. After tuning the parameters my model accuracies varies so the benchmark model in my project is lowest of knn of having 65% in order to meet the minimum performance of the model ,so I keep knn as my benchmark model after watching the benchmark video. Now we will try and achieve the better accuracy than the benchmark model

**III. Methodology**

**3.1 Data Pre-processing**:

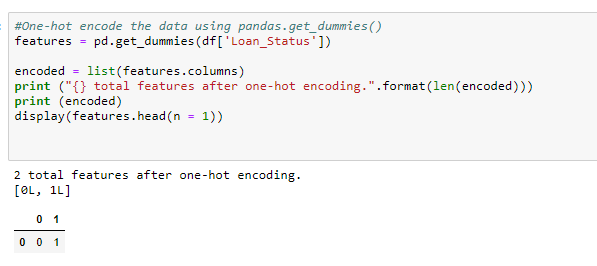
In this step of data pre-processing we will pre-process the data. We will read the data from dataset and replace the null values. We will know the information about all the data types.

We will know the various metrics regarding to the numerical data. To know the correlation between the numerical attributes we will plot the graphs to visualize the data (this context we will use pair plot and heatmap).

We will now remove the true outliers.

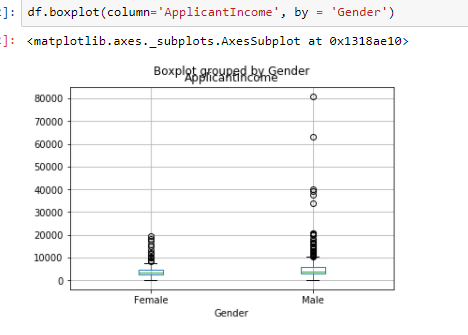
Now we need to perform a One Hot Encoding of the categorical variables to prepare the data for classification. We can do this easily by using from the sklearn.preprocessing module. For simplicity, we will use the later approach. After that the whole dataset is divided into training and testing data using classification model for performance measure by using sklearn.model\_selection.

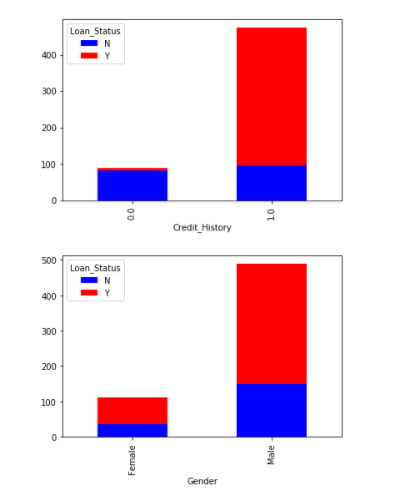
OneHotEncoding is as follows:

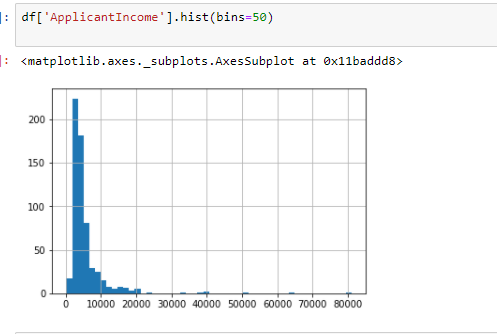


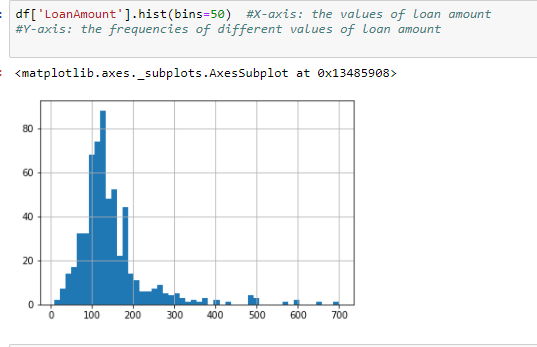
**3.2 Data Visualization:**

After performing the OneHotEncoding the data visualizations is as follows:









**3.3 Implementation:**

Here we will write a function classification\_model which helps us to calculate the accuracy for every classification model. We will find out the training and testing accuracy for the chosen models. We will now choose the best model out of svm, knn, logistic regression, gradient boosting, random forest, decision tree whose test accuracy will be more than that of the benchmark model

Knn : accuracy is 65.96%

Decision tree: accuracy is 69.54%

Gradient Boosting: accuracy is 69.55%

Svm: accuracy is 68.72%

Random forest: accuracy is 69.54%

Logistic Regression: accuracy is 68.72%

Among the above reports Gradient Boosting seems to be performing well. While building the Gradient Boosting Classifier I faced a complication while tuning the parameters. Firstly we should come to a conclusion about the parameters we are going to tune. Then by varying the range of values of the chosen parameter we should be able to get the parameter value which gives us the better accuracy than the untuned value. I changed the parameters and I increased the range of the parameters for all models.

After tuning them I observed that the Gradient Boosting classifier is giving the better performance when compared to other models .

From the above accuracies even random forest and decision tree are giving the better accuracies but the gradient boosting after tuned many parameters the performance for gradient boosting doesn’t decreases while other models performance decreases

The dataset will be split into training and testing set as a 80% of training data and 20% of testing data can be split using train\_test\_split method from sklearn. Random state will be specified as a particular number .

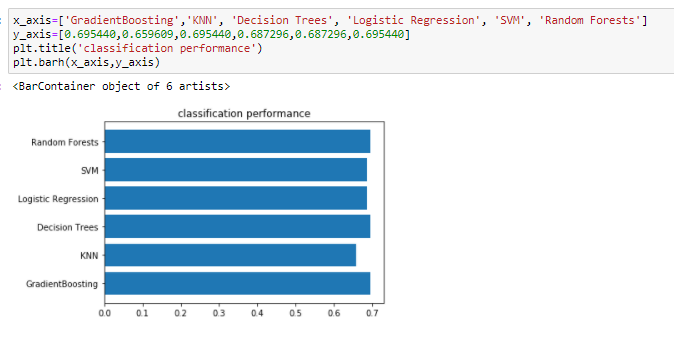
Before applying any supervised learning technique, we will implement a SVM. We will check our accuracy on that predictor. Note that in this case SVM will perform artificially well unlike in real world, a large proportion of patients has around 60% of the people have loan approvals.

Then, a method called as 'classification\_model' is defined that takes as input the following: model,data, data[predictors],data[outcomes]. It returns the accuracy on training and testing set respectively.

The classifiers are sent to the 'train\_predict' method, with 3 ,4 ,5 features training

data respectively so that it can be seen how performance varies with sample size.

Initially, some difficulty was observed in outputting the results in the desired manner. This file has been derived from the 'customer\_segments' project with some changes implemented. The output from this file is as follows:



**3.4 Refinement :**

I found out Gradient Boosting as the best classifier out of the chosen classifiers. Now we will perform tuning of Gradient Boosting classifier in order to achieve the better accuracy. Now we will find out the accuracy.

**IV Result Model** :

**4.1Evaluation**

The final model we have chosen is tuned Gradient Boosting which gave us more accuracy that is 69.55% which is a bit less than the tuned value. Here we can say that the solution is reasonable because we are getting much less accuracy while using other models but random forest and decision tree models are getting closer accuracies while tuning some parameters if we tune with more parameters the accuracies are not consistent as Gradient Boosting .So in this model we use Gradient Boosting inorder to achieve better performance and to predict the loan approvals. The final model that is tuned Gradient Boosting has been tested with various inputs to evaluate whether the model generalizes well.

**4.2 Validation :**

This model is also robust enough for the given problem. We tested the Gradient Boosting for various random states and we can clearly see that there is no big change in the accuracy. By performing hypertuning we have tested for the parameters like

model = GradientBoostingClassifier( n\_estimators=25, min\_samples\_split=25, max\_depth=7, max\_features=1) and we have achieved the mean which is close to the accuracy of 69.55%. So from that we can say that Small changes in the training data will not affect the results greatly. So the results found from this model can be trusted. The features are always randomly permuted at each split. Therefore, the best found split may vary, even with the same training data and max\_features=n\_features, if the improvement of the criterion is identical for several splits enumerated during the search of the best split. To obtain a deterministic behaviour during fitting,random\_state has to be fixed.

Use of Hypertuning

In machine learning, hyper parameter optimization or tuning is the problem of choosing a set of optimal , hyper parameter  for a learning algorithm.

The kind of machine learning model can require different constraints, weights or learning rates to generalize different data patterns.

These measures are called hyper parameters, and have to be tuned so that the model can optimally solve the machine learning problem.

Hyper parameter optimization finds a tuple of hyper parameters that yields an optimal model which minimizes a predefined [loss function](https://en.wikipedia.org/wiki/Loss_function) on given independent data. The objective function takes a tuple of hyper parameters and returns the associated loss. [Cross-validation](https://en.wikipedia.org/wiki/Cross-validation_(statistics)) is often used to estimate this generalization performance

Justification my final model’s solution is better than the benchmark model.

Gradient BoostingAccuracy Benchmark model Accuracy

69.55% 65.96%

From the above we can conclude that the results for the final model are stronger than the benchmark model. Hence we can say that tuned GradientBoosting provides the significant to solve the problem of loan approval.

**4.3 Justification** :

I used accuracy score as evaluation metric for prediction of loan approvals. Here I am predicting the accuracy score Despite the ambiguous results, it was decided to select GradientBoosting Classifier as the final classifier and

Final accuracy score on the testing data: 80%

When we compare Decision tree classifier and random forest classifier has good

accuracy\_score but it will take more training and testing time compare to other

models.

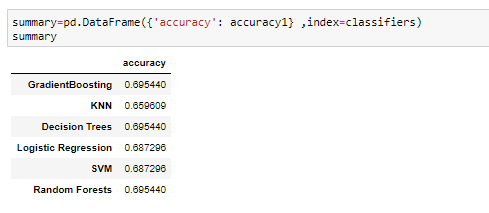
* My model generalizes well to unseen data
* It is not sensitive to small changes to the data or outliers
* It can be trusty because I made changes to the parameters but there is no effect in the model performance.It is high and cant be modified

**V. Conclusion :**

**5.1Free-form Visualization**

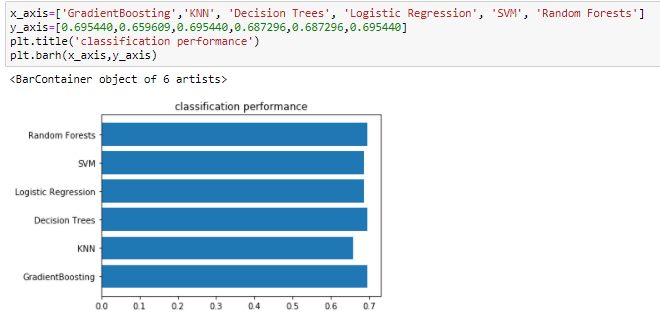
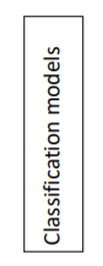
After studying the data and applying all the classification models we can

visualize the following

****

Plotting the above table data in the form of a bar graph taking accuracy score on x-axis and classification model names on y-axis we will get the following bar graph.

**Plot between accuracy and Classifier models**

****

****

From the above bar graph we can visualize that tuned Gradient Boosting will give us the good quality of output than the other. Here accuracy score is the important metric through with we will say that the tuned Gradient Boosting will give the good quality of output.

**5.2Reflection:**

1. I have learnt how to visualize and understand the data.
2. I have learnt that the data cleaning place a very vital role in data analytics.
3. Removing the data features which are not necessary in evaluating model is very important.
4. I got to know how to use the best technique for the data using appropriate ways
5. I got to know how to tune the parameters in order to achieve the best score.
6. On a whole I learnt how to graph a dataset and applying cleaning techniques on it and to fit the best techniques to get best score.

**5.3 Improvement:**

The process which I have followed can be improved to classify not only loan approvals but also the other approvals and also for the approving of credit cards by banks. This model is a part of my researching in banking sector.

This application can be taken to next level with the many more applications. This model also helps us to reduce the human errors while approving the loans to the customers and also it will help in reducing of the loss in banking sector.

Grid search technique is used to tune parameters in order to achieve best performance and suppose if we take parameters in boston\_housing datasets like max\_depth=3 or 4 and criterion(gini or entropy) we can tune and acheive better performance not only with two parameters by taking more parameters we can achieve better performance

It is only applicable when the learning algorithm requires a certain combination of parameters

One way for improving is to conduct a more exhaustive search for a combination of parameters which might give better results using Grid Search