**Creating a model for ‘Fraud Loan Prediction’ to check whether to give loan to a person or not**

*By* Chandralekha Raut

In this post, I will go through the whole process of creating machine learning model on the ‘***Fraud Loan Prediction’*** which is used by many peoples all over the world. In this, we need to *predict whether to give loan to a person or not based on the information provided.*

In finance, a loan is the lending of money by one or more individuals, organizations, or other entities to other individuals, organizations etc. The recipient (i.e., the borrower) incurs a debt and is usually liable to pay interest on that debt until it is repaid as well as to repay the principal amount borrowed.

The document evidencing the debt (e.g., a [promissory note](https://en.wikipedia.org/wiki/Promissory_note)) will normally specify, among other things, the principal amount of money borrowed, the interest rate the lender is charging, and the date of repayment. *A loan entails the reallocation of the subject*[*asset*](https://en.wikipedia.org/wiki/Asset)*(s) for a period of time, between the*[*lender*](https://en.wiktionary.org/wiki/lender)*and the*[*borrower*](https://en.wiktionary.org/wiki/borrower)*.*

**We will cover below points in the blog**

*1. Problem Definition*

*2. Data Analysis*

*3. EDA Concluding Remarks*

*4. Pre-processing Pipeline*

*5. Building Machine Learning Models*

*6. Concluding Remarks*

**Source of Dataset**: The dataset for this project is received from below link

[*https://github.com/dsrscientist/DSData/blob/master/loan\_prediction.csv*](https://github.com/dsrscientist/DSData/blob/master/loan_prediction.csv)

**Problem Definition:**

Here, we have project, ‘***Fraud Loan Prediction’***. As the name suggest, this project deals with Loan. In this, we need to predict whether to give loan to a person or not. It clearly states that if a person is capable to pay the loan or his intension to pay the loan.

Some banks have internal scores to see if a person is eligible for the loan. Here are some of the most common things banks look at before approving home loans.

1. *Credit history*
2. *Occupation*
3. *Age*
4. *Distance*
5. *Work experience*
6. *Spouse income source*
7. *Repayment period*
8. *Relationship with the bank*
9. *Purpose of the loan*
10. *Surplus income*

Banks consider all the above points while processing any loan application. It’s used to predict whether the applicant is capable for repayment of loan.

**Below is the example which shows the Loan process.**

***Purpose of the Loan***

First, give the bank a business plan. Show them that your business is solid and you have a strong track record of performance. Convince the bank that you don't really need their money, but if you had it, here's what you could do with it. Banks get queasy about lending to desperate borrowers. Be specific about how much money you need, what you will do with it and how you will pay it back.

The business plan should include:

* *Summary of the business and its products and services*
* *Experience of the management team*
* *Competitive environment*
* *Target market*
* *Financial statements.*

After the banker understands your business, the purpose of the loan and the method of repayment, he will evaluate the bank's risks by using the five Cs:

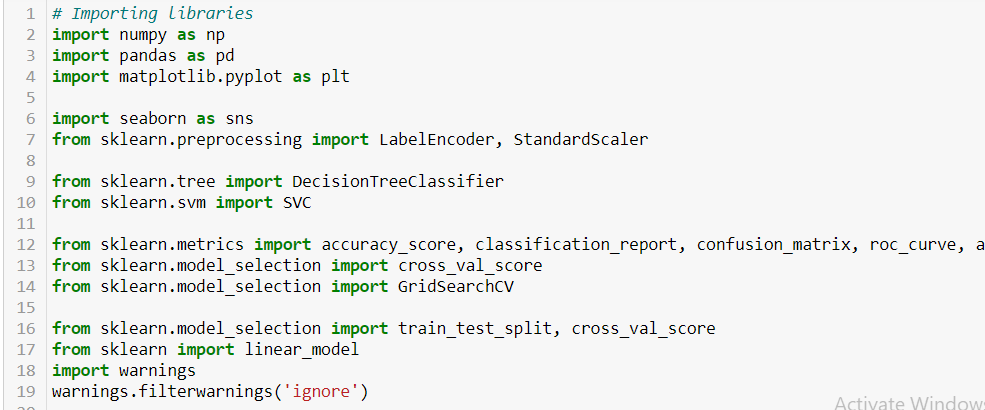
* *Character*
* *Collateral*
* *Capacity*
* *Capital*
* *Conditions.*

*Foremost on the list is character*. If bank doesn't trust you or think you're not an honest person, they will not approve your loan request. It doesn't matter how much collateral you have, it will not be enough to offset a lack of trust.

**Data Analysis:**

For data analysis, we need to get the data and observe it properly.

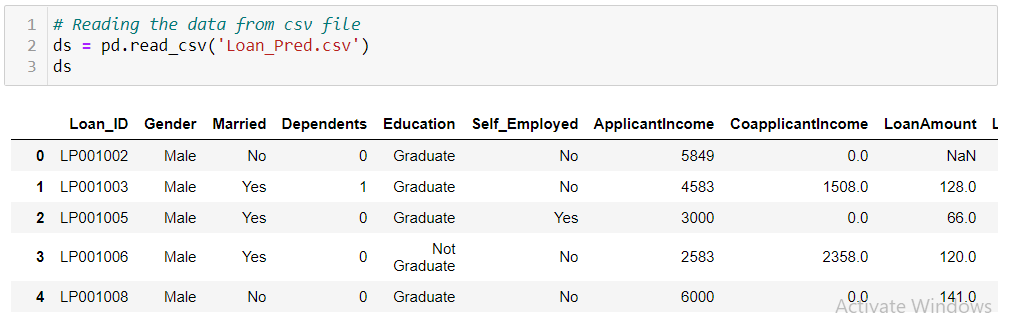
**Importing the libraries**



This is the very first step in model creation. We need to first import all the libraries that will be useful in reading a file, preprocessing, missing values handling, for splitting, plotting graphs, model creation and many more.

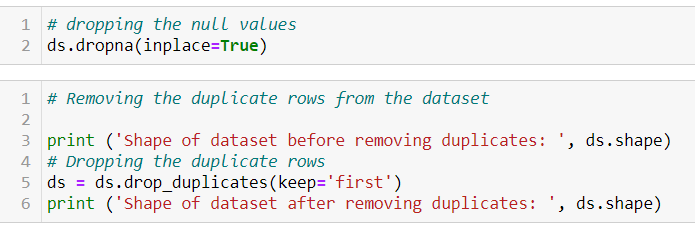
**Reading the csv file**

csv file contains the data, it is dataset, which will need to be used for processing. In short we can say, it’s the soul of all the process.



**EDA/Exploratory Data Analysis**

There is a need to remove the null values and duplicate rows as it will affect the predicted values. Sometimes instead of dropping the null values we can replace them with any appropriate value, we can say it will be mean, median, mode or 0. We can replace it with any value which will be appropriate as per the dataset.



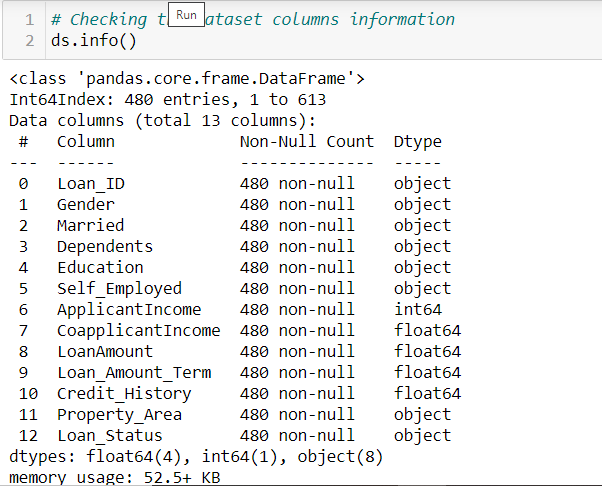
This dataset contains 480 rows, and *13 columns* (including target column).

It has ‘*object’, ‘float64’* and *‘int64’* data types.

***8 ‘object’ data type column***

***1 ‘int64’ data type column***

***4 ‘float64’ data type column***



Below is the short description of features of dataset.

*Loan\_ID* : Id of the loan application

*Gender* : Gender of the applicant

*Married* : Marital status of the applicant

*Dependents* : How many people are dependent on the applicant?

*Education* : Qualification of the applicant

*Self\_Employed* : Check if the applicant is self employed

*ApplicantIncome*  : Applicant’s income

*CoapplicantIncome* : Co applicant’s income

*LoanAmount*  : The amount for which the applicant applied

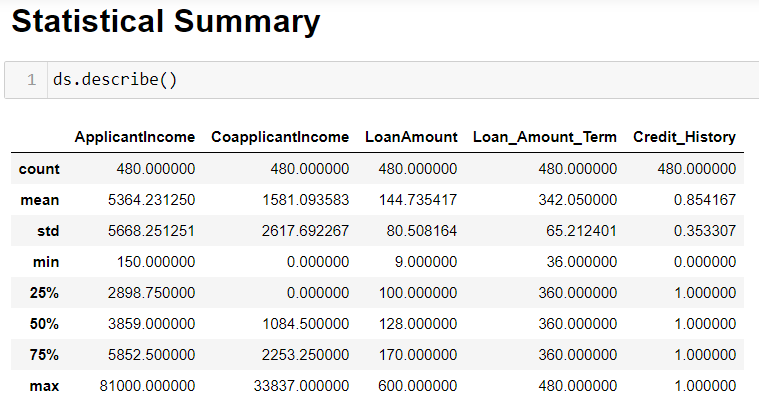
*Loan\_Amount\_Term* : The term of the loan (in number of years we can say)

*Credit\_History* : Credit history is a record of a consumer's ability to repay debts and demonstrat ed responsibility in repaying debts

*Property\_Area* : Property area (It’s also used for loan against property)

*Loan\_Status* : It will check whether to give loan to the applicant or *not*

**Statistical summary of the dataset**

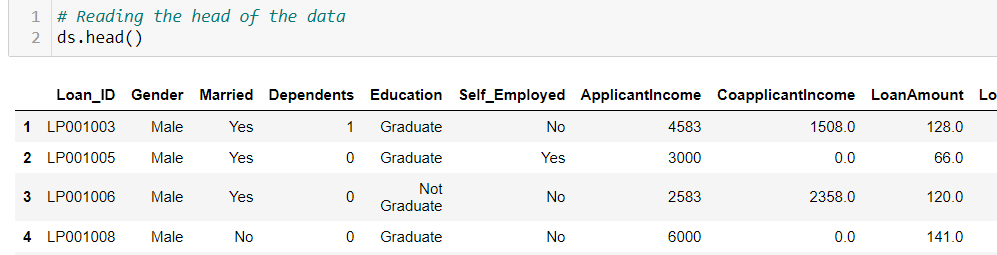
****

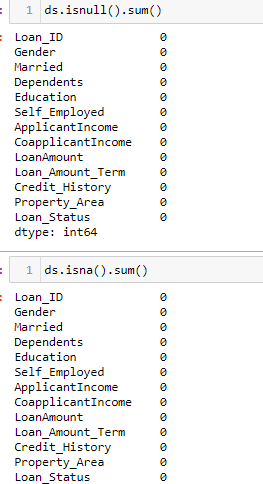
We can see that 5364.231250 is the mean of the applicant’s income, co applicant’s income’s mean is very less as compared to applicant’s income. There are some missing values in the dataset also.

**Viewing Data**

It will show the head of the data. We can see that, there are multiple data types present in the dataset. Some are integer, some are categorical. We need to convert categorical features to numbers as machine understands numbers only. There are some features which have widely different ranges; we need to convert them into some same scale. For that we are going to use- skew(), StandardScalar library also.

Some features have missing values; we need to handle those, as it will affect the target values.

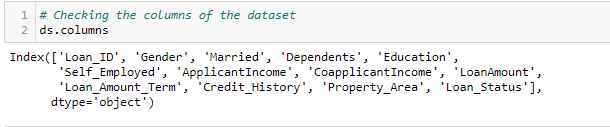
In below image, we can see the **missing values** and **NaN** values. 



* Above screenshot shows, there are no missing values in the dataset.
* If more of the values are missing or NaN then we can drop the columns.

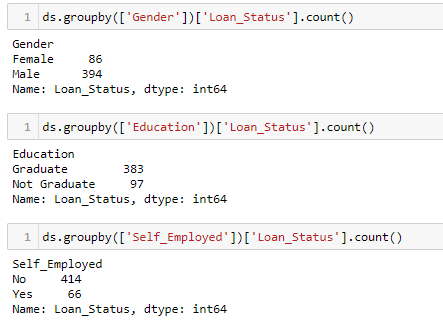
**Columns of dataset**

Now will see the list of columns used in the dataset.



All the above features contribute for prediction of loan application acceptance. It will help in predicting whether to give loan to an applicant or not.

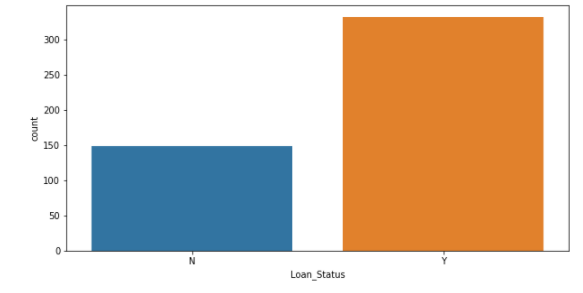
* Now we will see the relationship between various features of the dataset.



We can see some multiple features together as ‘*Gender and Loan\_Status’* , ‘*Education and Loan\_Status’* and ‘*self\_Employed and Loan\_status’*.

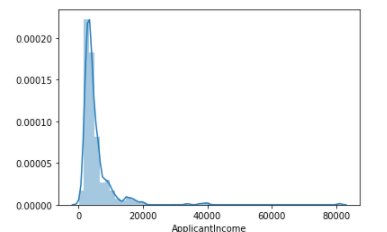
* This shows how many males and females are there which applied for loan.
* How many Graduates and Non-Graduates are there which applied for loan?
* How many people are self-employed and not self-employed which applied for loan.

**How many people are eligible for loan?**

****

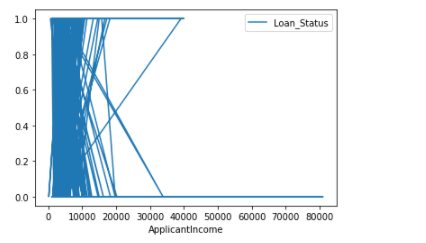
We can see that, above 300 people are eligible for loan and 150 people are not eligible for loan.

**Income of applicant**



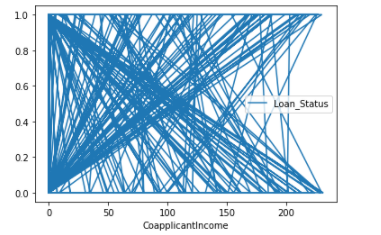
The above graph shows many people who have applied for loan have income between 0 to 20000.

1. **ApplicantIncome and loan status**



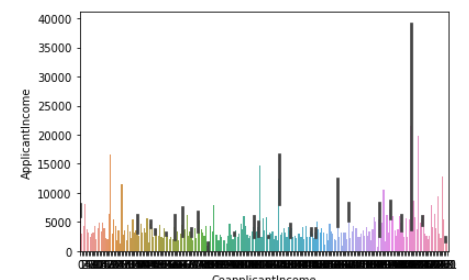
We can see that, more applicants from that category where the income comes under 40000.

1. **CoapplicantIncome and loan\_status**



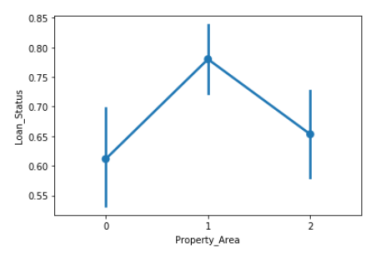
Above plot shows many co applicants are there which are eligible for successful loan application processing.

1. **CoapplicantIncome and ApplicantIncome**



Above graph shows that as the applicant’s income increases, co applicant’s income also increases. This shows the direct proportion of both.

1. **PropertyArea and Loan\_Status**



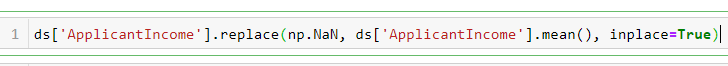
Above plot shows that acceptance of loan application also depends on the property area like ***urban, rural and semi urban***. We can see from which property area more loan application is applied and if they will be accepted or rejected.

**Data Preprocessing**

Firstly, we can drop the columns which are not significant and much useful in model creation. Here in training, we can drop the column ‘*LoanID*’ but it will be useful for test dataset because it will identify which applications are processing successfully and are eligible for loan.

**Missing Data**

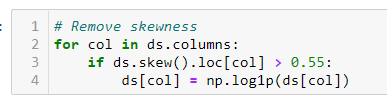
We need to handle the missing values, either we can replace the values with some values or we can drop those values. We can replace the missing values with mean, median, mode or 0 as I have written above.



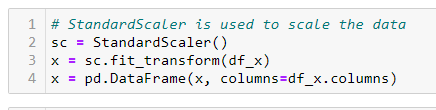
As we can see, where Applicant’s income is not there, we had replaced it with mean of ‘ApplicantsIncome’.

Like this we can do for other features, if any are there.

**Maintaining data in same scale**



* We are using log transformation to scale the widely spread data.

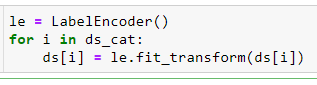


* We are using standard scalar also for the same purpose.

**Converting Features**

* + Convert the categorical features to numbers for further processing.
  + Below are the categorical features which need to be converted into numbers.

*'Loan\_ID', 'Gender', 'Married', 'Dependents', 'Education', 'Self\_Employed', 'Property\_Area', 'Loan\_Status'*



**Building Machine learning models**

Now we will train several Machine Learning models and compare their results. Later on, we will use cross validation.

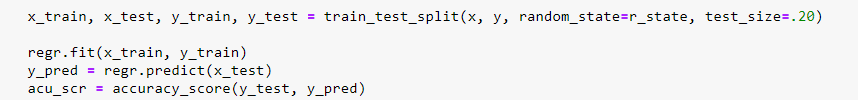
We will *check the maximum accuracy score based on the best random state.*

1. Fits all the base learners on the training dataset
2. At test time, use all base learners to predict test data.

We split the data into train and test datasets. These sets are used to train the model, predict the model, check the score, best parameters, cross validation score, mean and standard variations.

1. *Cross- validation*: In K Fold cross validation, the data is divided into *k subsets*. Now the holdout method is repeated k times, such that each time, one of the k subsets is used as the test set/ validation set and the other k-1 subsets are put together to form a training set. The error estimation is averaged over all k trials to get total effectiveness of our model. As can be seen, every data point gets to be in a validation set exactly once, and gets to be in a training set k-1 times. This significantly *reduces bias as we are using most of the data for fitting, and also significantly reduces variance as most of the data is also being used in validation set.* Interchanging the training and test sets also adds to the effectiveness of this method. **As a general rule and empirical evidence, K = 5 or 10 is generally preferred**, but nothing’s fixed and it can take any value.
2. Using train test split and distinguish between train and test datasets.

Y is the target variable.

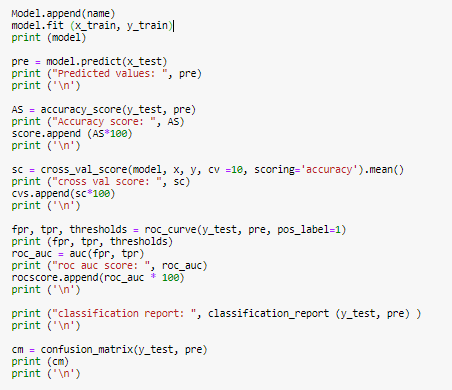


We are going to use the below algorithms to create the model.

1. *KNeighborsClassifier*
2. *SVC*
3. *DecisionTreeClassifier*
4. *RandomForestClassifier*
5. *LogisticRegression*
6. *GaussianNB*

**Model Processing**

*model.fit (x\_train, y\_train):* Used to train the model by using fit method



We have used above code in loop for all algorithms for selecting the best model for ‘*Fraud Loan Prediction*’. Here, we have used some of the methods as mentioned below:

**fit** : There is a fit function in ML, that is used for training of model using data examples. Fit function adjusts weights according to data values so that better accuracy can be achieved. After training, the model can be used for predictions, using **.predict() method call.**

**predict:**  a trained model, predict the label of a new set of data. This method accepts one argument, the new data x\_test (e.g. model. Predict(x\_test) ), and returns the learned label for each object in the array.

**accuracy\_score**: This will gives us the accuracy of how accurate the model is to predict the target.

**Cross\_val\_score**: Cross-validation is a statistical method used to estimate the skill of machine learning models.It is commonly used in applied machine learning to compare and select a model for a given predictive modeling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods.

**roc\_curve**: As the area under an ROC curve is a measure of the usefulness of a test in general, where a greater area means a more useful test, the areas under ROC curves are used to compare the usefulness of tests.

**confusion\_matrix** : A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.

For a binary classification problem, we would have a 2 x 2 matrix as shown below with 4 values:

[](https://cdn.analyticsvidhya.com/wp-content/uploads/2020/04/Basic-Confusion-matrix.png)

Let’s see the code snippet for one of the algorithm, say LogisticRegression

*lr = LogisticRegression()*

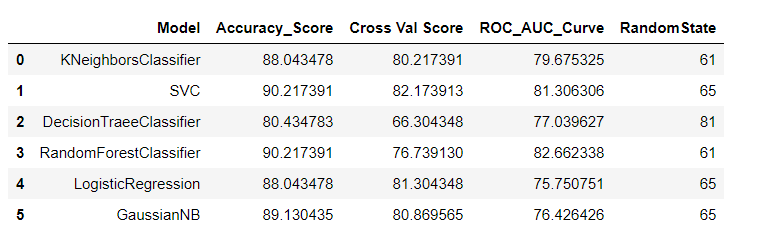
*lr.fit(x\_train, y\_train)*

*y\_pred = lr.predict(x\_test)*

*print (y\_pred)*

*accuracy\_scr = accuracy\_score(y\_test, y\_pred)*

We will use all algorithms like this only. Then we will create a table for all the models with its results. We tried multiple algorithms and found the results accordingly.



We observed that SVC algorithm is working very well with *maximum accuracy score and maximum cross\_val\_score.*

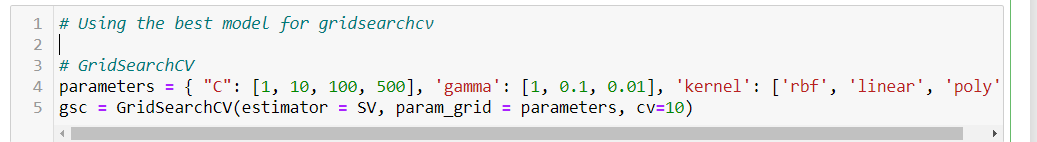
* So, as the final algorithm, we have selected is **SVC**.
* Next, we will use SVC algorithm to crate the model.
* We will use the SVC with best parameters i.e. hyper parameters tuning

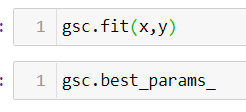
**Hyper parameter tuning:**

In [machine learning](https://en.wikipedia.org/wiki/Machine_learning), hyperparameter optimization or tuning is the problem of choosing a set of optimal [hyperparameters](https://en.wikipedia.org/wiki/Hyperparameter_(machine_learning)" \o "Hyperparameter (machine learning)) for a learning algorithm. A hyperparameter is a [parameter](https://en.wikipedia.org/wiki/Parameter) whose value is used to control the learning process. By contrast, the values of other parameters (typically node weights) are learned.

Hyper parameters can be classified as model hyperparameters, that cannot be inferred while [fitting the machine to the training set](https://en.wikipedia.org/wiki/Model_fitting) because they refer to the [model selection](https://en.wikipedia.org/wiki/Model_selection) task, or algorithm hyperparameters, that in principle have no influence on the performance of the model but affect the speed and quality of the learning process. Examples of algorithm hyper parameters are [learning rate](https://en.wikipedia.org/wiki/Learning_rate) and mini-[batch size](https://en.wikipedia.org/w/index.php?title=Batch_size&action=edit&redlink=1).

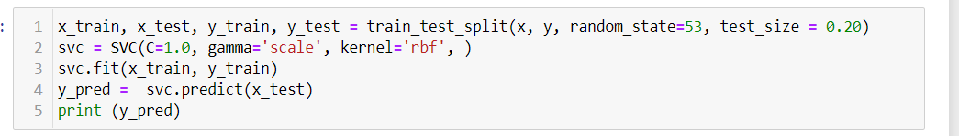
Different model training algorithms require different hyperparameters, some simple algorithms (such as [ordinary least squares](https://en.wikipedia.org/wiki/Ordinary_least_squares) regression) require none. Given these hyperparameters, the training algorithm learns the parameters from the data. For instance, [LASSO](https://en.wikipedia.org/wiki/LASSO) is an algorithm that adds a [regularization](https://en.wikipedia.org/wiki/Regularization_(mathematics)) hyperparameter to [ordinary least squares](https://en.wikipedia.org/wiki/Ordinary_least_squares) regression, which has to be set before estimating the parameters through the training algorithm.

  
We will train the model by using the fit method

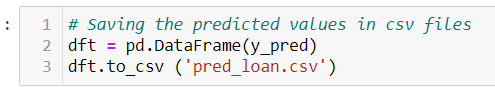


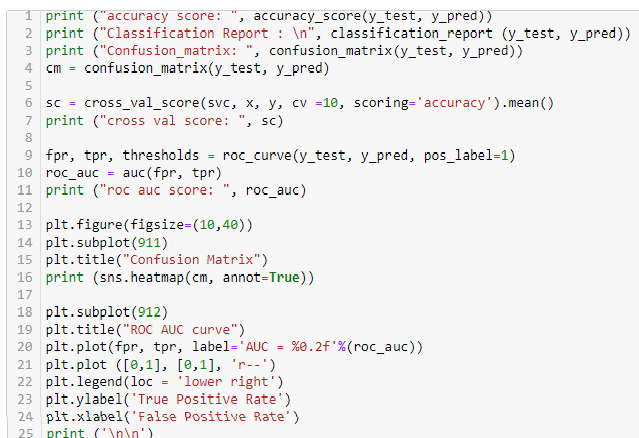
We will choose the best parameters of the SVC model by using the best\_params\_ method.

Now, we will create the instance of the SVC with best parameters received.



Next, with best parameters will predict the target. We will save the predicted values in the csv file.

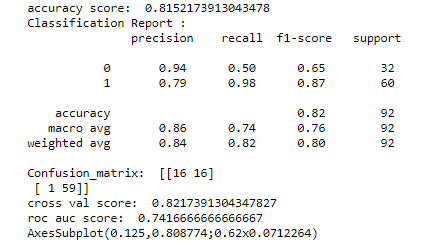


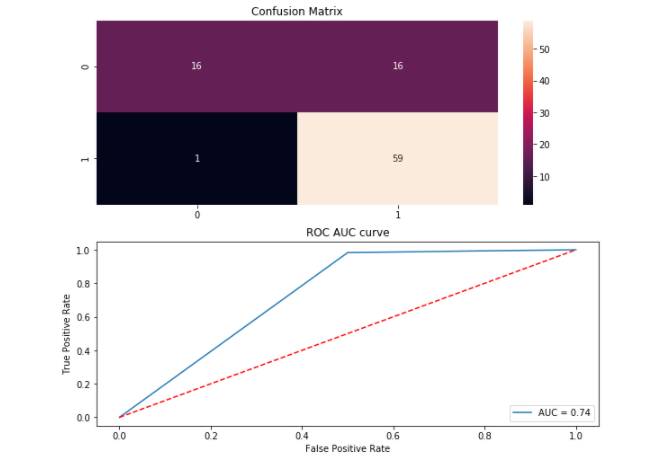


We will compute some values by using functions as mentioned below

1. *accuracy\_score*
2. *classification\_report*
3. *confusion\_matrix*
4. *cross\_val score*
5. *ROC curve*

Below are the scores of computation:





**ROC AUC Score:**

The ROC AUC Score is the corresponding score to the ROC AUC Curve. It is simply computed by measuring the area under the curve, which is called AUC.

A classifiers that is 100% correct, would have a ROC AUC Score of 1 and a completely random classifier would have a score of 0.5.

**Conclusion**

Most classification problems in the real world are imbalanced. Also, almost always data sets have missing values. In this post, we covered strategies to deal with both missing values and imbalanced data sets.

There is no definitive guide of which algorithms to use given any situation. What may work on some data sets may not necessarily work on others. Therefore, always evaluate methods using cross validation to get a reliable estimates.

Firstly, we import the needed libraries for all the processing. Then we read the csv file to get the data. Later did EDA, Exploratory Data Analysis, to check for missing, NaN values, removed the duplicates, replaced or removed the NaN values. We used imported libraries for that. Later we did different plots for the dataset.

Then we changed the categorical data to numbers. We prepared the data to fit in same scale. Then we used the multiple algorithms to find which one is working best for the model. Based on the result, we selected that algorithm to create the final model and used this by predicting the target values. For this, we used the hyper parameters tuning to get the best parameters for the algorithm. Then we computed the values which gives us information about how the model is performing.

In general, how we evaluate the model, how we plan to calculate the target, is all depend on the dataset and how we thought about it. We can predict the target by forming other parameters as groups, some categories, or create some new columns based on some logic.