**Project Name : Loan Application Status Prediction**

**Problem Definition**:

This is a loan application status project where bank have to check the status of the client means bank have to check the client personal status and financial status. After analyse all the details bank will decide they should provide bank loan to the client or not.

**Data Analysis:**

We receive the raw data from the client in csv format and download the data from the github link then convert that csv file into dataframe using **read\_csv () method**. Now we use **shape () & size ()** function to check the total number of rows, columns & elements in dataset.

**EDA Concluding Remarks:**

We use **unique ()** function to check the unique elements of the column. We use **nunique ()** method to check total number of unique elements. After that we **value\_counts ()** function to count the frequency of the unique elements where we find no duplicate elements are containing by any column, no unwanted or Irrelevant column is there in dataset.

We use **info ()** function which will provide us detail information about dataset. Like number of rows, columns, elements, datatype of the data and null value if any column is containing.

We use **isnull (). sum ()** to check total number of null values each column contain. Here we found null values so use some methods to fill the null values.

**fillna (). mode ()** we have a categorical data so through this method we take that value which have a high frequency and fill it.

**SimpleImputer(strategy='most\_frequent')** we also use simpleimputing techniques to fill the null values where it fills the most\_frequent value to the null values.

**Duplicates (). sum ()** through this method we check any duplicate data is containing by dataset.

**Describe ()** this method will tell us short story only about the continuous or descrete column after go through the dataset we assume that some columns have skewed data because mean value is less than and greater than medium.

We also assume outliers is there because max and min value is far from the 1 quadrant and 3 quadrant.

Now we use graphs to check how much our assumption is correct. we use **distplot()** to check how data is distributed means data is normally distributed or not, after plot the **distplot ()** we found data is not normally distributed.

Now we use boxplot () to check the outliers and found some column have outliers so we use **zscore ()** method which first converts the 'normal distribution' into ' standard distribution' then remove the outliers.

Again, we use distplot () to check data is normally distributed where we found data is batter distributed and skewness has been removed.

We remove one feature “**loan id”** from dataset as it will not give any impact on our result.

Now we use encoding techniques to convert string into number because machine learning model understand only numbers. We use **LabelEncoder ()** method to convert string into number.

After transform and structuring the feature data into desirable format now we come onto the target data where we check data is balance or not.

We use **value\_counts ()** method to check the target data where we found data is imbalance means maximum feature data containing information about 1 category that is called majority data point and other is called minority data points. We first we have to balanced the data otherwise model will not perform well it can be underfitted because model will containing maximum information about one category.

We also use **countplot ()** to recheck that data is balanced or not after go through the graph we can say data is not balanced. data is imbalanced now we use sampling technique to balance the imbalance data.

We use **SMOTE (random state=40, k\_neighbors=5, sampling\_strategy=0.70)** sampling technique to increase the data in a minority datapoints to balance the data.

After that we checked the correlation between the features data and also checked the relationship between feature and target data using **corr ()** method. we also use here **heapmap ()** method through which we can analyse the correlation and relationship by the graph with colors.

After using **corr ()** function we found no correlation is there between the feature data.

After that we use **scatterplot ()** tochecked the relationship between feature and target data so that we can remove that feature data which have no significant relation with the target data. After go through the graph we analyse every feature has a relation with target so we cannot remove any feature data on the basis of assumption until we are not sure.

**Pre-processing Pipeline**

As it is binary classification problem so we select those models which can perform well to solve the classification problem like (Logistic regression, Knn classifier, Decision Tree Classifier, Random Forest classifier, AdaBoost classifier, GBDT classifier, Support vector classifier etc.

Now first we separate the target data from the feature data and use data standardized method **StandardScaler ()** which will transform all the feature data into one scale.

Now we use **PCA (Principal Component Analysis) method** It is a dimension reduction technique. It is decomposition technique(Breakdown complex problem into simple once). It is applied only on feature data. It can be applied if you have too many features and their correlation is not significant with the target. It also take care of multicollinearity.

Before apply PCA we have to standard the data then transform the data which convert skewed data into normal data. Then it uses parallel translation which will shift datapoint towards to 0 and fix the mean value at 0 quadrant. Then uses Eighen Vector method which give direction means draw the line where it finds less residual or less error. Then uses Eighen Value which find the distance between 0 quadrant and the closest data points to the eighen vector and save that value in one dimension and so on. It creates dimentions, Principal component or we can say it convert feature data into dimentions now PCA will reduce the dimentions by using **screeplot ().**

We use **screeplot()** plot which will make the curve using PCA () method so on the basis of that curve we will select the dimentions and remaining dimentions will be removed by the **screeplot().**

After select the required dimentions we split the data into two parts train and test. Train data means on the basis of training data we will train the model and on the basis of test data we evaluate the model how much model is performing well or understanding the data.

**Building Machine Learning Models**

As it is binary classification problem so we select those models which can perform well to solve the classification problem like (Logistic regression, Knn classifier, Decision Tree Classifier, Random Forest classifier, AdaBoost classifier, GBDT classifier, Support vector classifier etc.

**KNeighborsClassifier() :** KNN model is used to calculate the distance between test data and all training data points and after calculate the distance it finds those k points (training data point) which are near or closest to test data then on the basis of higher probality it gives their prediction.

Now after trained the model we use metrics to evaluate the KNN model accuracy.

**Confusion metrics** : It is performance evaluation tool which display the number of true positive, false positive, true negative and false negative.

**accuracy metrics :** On the basis of Confusion metrics, it finds the accuracy of the model. It measures true positive and true negative predicted numbers out of all numbers.

**Recall or Sensitivity** : It measures True Positive predicted number out of all positive numbers.

**Precision** : It measures of True Positive predicted number out of all positive predicted number.

**F1\_Score** : It is a combination of Recall and Precision Metrics (some portion from precision and some portion from recall. It is defined as harmonic mean of precision and Recall

**Accuracy :** 0.7283236994219653

**Note : After evaluate the KNN Model we assume that model is performing well but still we will use another model also, may be that model is perform better than KNN.**

**LogisticRegression() :** It is a binary classification algorithm which is used to solve the definite class or data or category not for continues data.

**Accuracy :** 0.7803468208092486

**Note : After evaluate the LogisticRegression Model we assume that model is performing better than KNN but still we will use another model also, may be that model is perform better than LogisticRegression.**

**DecisionTreeClassifier()** : This algorithm is most versatile which can perform both classification and regression analysis. it is very powerful and work great with complex dataset. this algorithem divide the dataset into tree and on the basis situation or scenario it gives their prediction.

**Accuracy :**0.7341040462427746

**Note : After evaluate the DecisionTreeClassifier Model we assume that model is performing better than KNN but not performing better than LogisticRegression() so we will use another model also, may be that model is perform better than DecisionTreeClassifier.**

**RandomForestClassifier()** : It is one of the best model which is used for both problem (class/regression). Mostly used for classification problem. It works on ensemble approach where it uses multiple model and take the decision on the basis of model majority.

**Accuracy :** 0.7976878612716763

**Note : After evaluate the RandomForestClassifier Model we assume that model is performing better than all above models but still will use another model also, may be that model is perform better than RandomForestClassifier.**

**SVC() :** This is another supervised model which is used to solve both classification and regression problem. It tries to divide the target data or label data in such a way that it can be linearly separated and it has many ways to do it but it draw a line in such a way which contain maximum distance or maximum margin between the line and the closest data points of target data and that line is called hyperplane and closest data points is called support vectors.

**Accuracy** : 0.7976878612716763

**Note : After evaluate the RandomForest Model we assume that model is performing well and better than all models except RandomForest Model**

**Concluding Remarks**

Now we take all the models into Dataframe and compare the accuracy of all the models where we found **RandomForest & SVC both provide the same accuracy\_score.**

**ROC & AUC curve metrics :**  Now we use this metric to solve this issue that who is the best model out of all the models, ROC & AUC metric will make curve and select that model which make highest curve means that model understanding the data better than other models.

After use ROC & AUC metric we found Random Forest model make the highest curve means Random Forest model is understanding the data better than other models so Random Forest model is the winner and will be use to solve this problem.

**Hyperparameter Tunning**

Model use default parameter value to build the model so it can be Bios and high variance then how we will check that model is overfitted and underfitted or not.

Then we use Hyperparameter Tunning method where we tune the parameter and check model is overfitted and underfitted or not and we can also increase the accuracy of the model through this method. In Hyperparameter Tunning method we have two techniques which provide the best parameters.

**RandomisedSearchCV** : This techniques will make random combinations means some time it make all the combination, sometime it make one combination so it is just better than default parameter and it is use when client have less time and less budget.

**GridSearchCV**: This techniques is the best technique because it make all the combination of parameter and provide the best parameter to us then we use that parameter with the model. Again, train the model and evaluate the model.

Here we use **Gridseacrhcv ()**

Now we pass the tunable parameter to the Random Forest Model and evaluate the model.

**accuracy\_score** : 0.815028901734104

**Note : After use the hyperparameter tunning method we can say Random Forest Model is not Bios and no high variance is there so this is the best model to solve this business problem.**