LOAN APPLICATION STATUS PREDICTION

PROBLEM STATEMENT:

There is the company named Dream Housing Finance that deals in all home loans. They have presence across all urban, semi-urban and rural areas. Customer first apply for home loan after that company validates the customer eligibility for loan. However, doing this manually takes a lot of time. Hence it wants to automate the loan eligibility process (real time) based on customer information.

So, the final thing is to identify the factors/customer segments that are eligible for taking loan. How will the company benefit if we give the customer segments is the immediate question that arises. The solution is Banks would give loans to only those customers that are eligible so that they can be assured of getting the money back. Hence the more accurate we are in predicting the eligible customers the more beneficial it would be for the Dream Housing Finance Company.

So, I need to classify whether the **Loan_Status** is yes or no. So this can be solved by any of the classification techniques like

- Logistic Regression
- SVC
- KNeighbors classifier
- Decision tree classifier
- Random forest classifier
- Ada boost classifier
- Bagging classifier
- Gradient boost classifier

DATA ANALYSIS:

Let dig into code. First thing first imported the basic and necessary packages like pandas, numpy, seaborn etc. So , I can further run my project.

- Now, I'm going upload or read the file/dataset using pandas for this we use read_csv
- Then I got the dataset on my notebook which don’t have columns names.
- Then, I mentioned the columns titles.

Description for data columns:

-It’s very useful and important to know about the data columns before getting into the actual problem for avoiding confusion at a later state. Now let us understand the data columns that has been already given by the company itself, first so that we will get the peak.

There are all together 13 columns in our dataset. in which Loan_Status is the response variable and rest all are the variable that decide the approval of the loan or not.

Now, let us look into the each one the variable and can make some of the assumptions. There is no harm to assume the some of the statements.

- LOAN ID: As the name suggests each person should have a unique loan ID.
- GENDER: It is men or female. Not including the other gender.
- MARRIED: Applicant who is married is represented by Y and not married is represented as N. The information regarding whether the applicant who is married is divorced or not has not been provided. So, we don't need to worry regarding all these.
- DEPENDENTS: The number of person dependent on the applicant who has taken loan has been provided.
- EDUCATION: It is either non-graduate or graduate. The assumption I can make is the probability of clearing the loan amount would be higher if the applicant is a graduate.
- SELF-EMPLOYED: The name suggests self-employed means. He/she is employed for himself/herself only. Freelancer or having a own business might come in this category. An applicant who is self-

employed is represented by Y and the one who is not is represented as N.

- APPLICANT INCOME: Applicant income indicates the income by applicant. So, the basic assumption that I can make would be “one who earns more have a high probability or chance of clearing loan amount and would be highly eligible for loan”.
- CO-APPLICANT INCOME: It indicates the income of co-applicant I can also think that ‘if co-applicant income is higher, the probability of being eligible would be higher’.
- LOAN AMOUNT: The amount indicates the loan amount in thousands. One assumption is ‘if loan amount is higher, the probability of repaying would be lesser.
- LOAN_AMOUNT_TERM: Its represents the number of months required to repay the loan.
- CREDIT_HISTORY: it is a record of a borrower’s responsible repayment of debts.
- PROPERT_AREA: A area where they belongs to is my general assumption as nothing more is to be told. It can be three types urban, semi-urban and rural.
- LOAN_STATUS: If the applicant is eligible for loan its yes represented by Y otherwise its no represented by N.

Now, I get the top 5 vales for dataset. So I used the Head function df.head().

Then, I checked the shape of the data. I used df.shape function to checked. And checked the columns using df.columns.

Then, I used df.describe() function to check statistical summary of data frame which is used for getting the sense of data distribution and central tendency.

I did check the count of missing values in each columns in data frame. So basically it is the step of cleaning the data and preprocessing. I used df.isnull().sum().

So next I did checked the all variable datatype using df.info().

After that, I checked the unique value of some columns like loan_amount_term, loan amount, dependents using df[‘loanAmount’].nunique. Then I put them into columns.

Then I used the df[col].fillna(df[col].mode()[0],inplace=true) to fill the missing values in each column in the data frame.

Again I checked the null values in data frame using df.isnull().sum().

EXPLORATORY DATA ANALYSIS(EDA):

It is critical step of data analysis process. It includes to summarizing the main characteristics of dataset. Main goal of EDA is to understands the data ,detect pattern, spot, test hypotheses, and check assumptions with the help of summary statistics and graphical representation like:

- The one whose salary is ore can have a greater chance of loan approval.

- One who is graduate has a better chance of loan approval.
- Married people would have a upper hand than unmarried people for loan approval.
- Applicant who has less number of dependents have a high probability for loan approval.
- Lesser the loan amount the higher the chance for getting loan.

Let import the library for visualization

Import matplotlib.pyplot as plt

- First i checked barplot for applicant income with loan status.

Conclusion: Applicant income is not deciding the loan status, whether customer gets the loan or not.

- Second coapplicant income with loan status.

Conclusion: There are chances not to be approved doesn’t matterbut if coapplicant income is higher might be that's depends.

- Third one is , Propert_Area with Loan Status.

Conclusion: People who lived in urban area have 50/50 chances of approval. For rural area there is no chance increasing for approval. Semi-urban area have high chances for approval.

- Forth is, Credit-history with loan status.

Conclusion: So, most of the people who have credit history they are getting the approval while the logical figure the most people who don’t have credit history they are not getting the approvals.

- Fifth one is ,loan amount term with loan status.

Conclusion: People those are taking the loan for 480 months, they are getting the loan. But people who are taking 360 months are more half are getting the approval.

- Sixth one, Self-employed with loan status.

Conclusion: In self-employed, we see around 50% of difference between of approved and not approved people.

- Seventh one with Education with loan status.

Conclusion: In graduated category around 350 approved and around 150 not approved those who were not graduated.

- Eighth one, dependents with loan status.

Conclusion: I noticed people those are not dependant are getting the chance of loan approval.

CONVERTING CATEGORICAL INTO NUMERICAL DATA:

First, I imported the library. From sklearn.preprocesing import LabelEncoder

It basically encodes categorical features into numeric values. It is useful for converting non-numeric value into format so machine learning model can easily work on.

SKEWNESS HANDLING:

It measures the asymmetry of probability distribution of real valued variable. It also helps to understand the direction and the extent of skew in the dataset.

So, in this dataset, I checked the skewness more than +/-0.5 will be treated but not treating the object and target column. Most of the columns in dataset are skewed and most of them are categorical feature and imbalance that why its showing skewness. Now will deal with numerical column.

- Coapplicant Income (will be treated).
- Used Power Transform for skewness removal.

OUTLIER HANDLING:

- There are only some of the columns seems having the outlier after skewness handling.
- And for outlier removal, I used **Zscore**
- Them I imported the library [from scipy.stats import zscore]
- It removed 12 rows from data as outliers.
- **IQR** (interquartile range) it is used to measure the statical dispersion, which is spread in the data. It also calculated the difference between the first quartile (Q1) and third quartile(Q3), also describe the middle 50% of value when ordered from lowest to highest.
- I imported the library [from scipy.stats import IQR]
- Huge data loss in IQR.
- After all data cleaning and data structuring are done, now I'm going for next step which is model building.

DIVIDING 'X' AND 'Y':

Divided the data into two-part features (x) and target (y) for model.

SCALING 'X' VALUE:

Scaled the whole data using MinMaxScaler in min0 to max 1 so model won't be biased for any number.

SPLITTING THE TRAIN AND TEST DATA:

- Splitting the dataset inti training and test sets isa difficult step in building and evaluating machine learning models. Training set is used to train the model, while test set is used to evaluate the model's performance.
- I imported the library[from sklearn.model_selection import train_test_split.
- I put the test size=.27 with random_state-42.
- Then i got my X(train,test) shape and y(train,test)shape.

BUILDING MACHINE LEARNING MODELS:

It involves the several steps, from data processing o model selection, training, evaluation and tunning of the data. machine learning model using python and popular libraries like scikit-learn.

Imported all the libraries which I used in this project for model building.

There are many sampling techniques like random sampling, stratified sampling etc. The major purpose is to improve the accuracy which can be obtained by hiding the some of the portion of train data and running the model so that on an average the one that gives higher accuracy can be taken for test data.

- From sklearn.linear_model import LogisticRegression
- From sklearn.svm import SVM
- From sklearn.tree import DecisionTreeClaasifier
- From sklearn.neighbors import KneigborsClassifier
- From sklearn.model.model_selection import train_test_split
- From skleran.ensemble import RnadomForestClassifier
- From sklearn.ensumble import AdaBoostClassifier
- From sklearn.ensemble import BaggingClassifier

- From sklearn.ensemble importGradientBoostingClassifier
- From sklearn.model_selcetion import cross_val_score

These all models been used. Every model gave the cross-validation score, f1 score, precision, recall, training and testing accuracies. Like

LOGISTIC REGRESSION:

Logistic regression is a statistical method which is used for binary classification that model the probability of a binary outcome based on or more predictor variables.

- Cross-validation score:0.8075184619302265
- Training accuracy:0.816554
- Testing accuracy:0.789156626

DECISION TREE CLASSIFIER:

A Decision Tree Classifier is a supervised machine learning algorithms used for classification takes. It works by splitting the data into subsets based on the values of inputs features, using tree-like models if decisions.

- Cross-validation score: 0.7048114087698
- Training accuracy:1.0
- Testing accuracy: 0.710843373

KNEIGHBORS CLASSIFIER:

The KNeighbors classifier (KNN) is a type of instance learning or non-generalizing learning. It is one of the simplest machine learning algorithms used for classification and regression. KNN is to classify a new data point on its similarity to the data points in the training set.

- Cross-validation score:0.77169132
- Training accuracy:0.807606626398
- Testing accuracy:0.7831325301204

RANDOM FOREST CLASSIFIER:

A random forest classifier is an ensemble learning method or technique that combines the multiple decision trees to improve the overall result of the model. It consists of large number of individual decision trees to operate as an ensemble.

- Cross-validation score:0.7765693722
- Training accuracy:1.0
- Testing accuracy:0.777108433734

ADA BOOST CLASSIFIER:

Ada boost classifier is an ensemble learning technique that combine the predictions of multiple weal classifier to create a strong classifier

- Cross-validation score:0.7737916446324
- Training accuracy:0.8702460850111
- Testing accuracy:0.716874698795

BAGGING CLASSIFIER:

It is ensemble learning technique designed to improve the stability and the accuracy of machine learning algorithms. It reduces variance and helps prevent overfitting. Bagging works particularly well with high-variance, low variance, low-bias models.

- Cross-validation score:0.73947646
- Training accuracy:0.9955525727
- Testing accuracy:0.7590361445

GRADIENT BOOSTING CLASSIFIER:

Gradient boost classifier is an ensemble machine learning technique that builds a model from an ensemble of weak learner in a sequential manner. It combines the concepts of boosting and gradient descent to improve model performance.

- Cross-validation score: 0.7460007403490
- Training accuracy:0.9127516778

- Testing accuracy: 0.75301204819

I have tried many techniques like random forest, Ada boost, decision tree etc. And came to conclusion that the above code gave maximum accuracy. So, there is still a lot of options to enhance accuracy which I must figure it out still.

CONCLUSION:

Key finding and conclusions of study:

- So, there are high chance of approval for graduate people comparing to not graduate.
- There are high chances for loan approval when you are taking loan for less persistence.
- People who are from Semi-urban area are having high chance to get their loan approved comparing people from other area.
- There is high chances of loan approved when you are not having credit history. People those are not having any credit history mostly not getting approved.