Project Report

GitHub URL

<https://github.com/SaurabhPatel02/UCDPA_SaurabhPatel>

Abstract

This project is all about the loan approval status prediction. The dataset corresponds to a classification problem on which we need to make predictions on the basis of whether a person will get loan approved on the basis of given the features in the dataset.

Introduction

Data set is simple and easy to understand for beginner to start data science journey. With the help of this simple data set with limited features, we feel more comfortable to perform the data analysis and implement data science and data analytic feature and capabilities using python.

I found this solid basis to grow in machine learning area to predict the feature.

Here, multiple real time datasets got utilized from different data sources like API, CSV to implement different use cases by using python and it’s ML capabilities.

Dataset

For project specific, “Loan Prediction Problem” data set was used considering simple and featured rich data set while to perform other tasks several other data sets were used. The details mentioned as follows.

Data Set for Project (Machine Learning & All) :

Dataset used is named as ‘[Loan Prediction Dataset](https://www.kaggle.com/altruistdelhite04/loan-prediction-problem-dataset)’ sourced from Kaggle. The dataset contains a set of 613 records under 13 attributes which help us to understand the importance and weight of particular feature to predict whether loan will get approved or not and which feature will play most important role.

Data source path URL : <https://www.kaggle.com/altruistdelhite04/loan-prediction-problem-dataset>

Target value column : Loan\_Status

|  |  |  |
| --- | --- | --- |
| 1. No. | Column Name | Description |
| 1 | Loan\_ID | Unique loan ID |
| 2 | Gender | Male or Female |
| 3 | Married | Married (Yes/No) |
| 4 | Dependents | No of dependents on main applicant |
| 5 | Education | Educated (Graduate/Not Graduate) |
| 6 | Self\_Employed | Self employed - Yes/No |
| 7 | Applicant\_Income | Primary Applicant Income |
| 8 | Coapplicant\_Income | Secondary applicant income |
| 9 | Loan\_Amount | Loan amount in thousands |
| 10 | Loan\_Amount\_Term | Loan duration in months |
| 11 | Credit\_History | Credit history |
| 12 | Property\_Area | Urban/Semiurban/Rural |
| 13 | Loan\_Status | Target Value - Loan approved (Yes/No) |

Note : referred in 7\_ML\_SL\_HyperParamTune code file

Data Set for importing data, handle missing values and duplicates :

Data source path URL : <https://www.kaggle.com/shivamb/netflix-shows>

Netflix is one of the most popular media and video streaming platforms. They have over 8000 movies or tv shows available on their platform, as of mid-2021, they have over 200M Subscribers globally. This tabular dataset consists of listings of all the movies and tv shows available on Netflix, along with details such as - cast, directors, ratings, release year, duration, etc.

After analyzing the data, I found this perfect source to perform data cleaning steps as this data having needed missing values and duplicates.

Note : referred in 2\_CSV\_As\_DataSource\_Data\_Cleaning code file

Data Set for finding spam email sender details using Reg Ex :

Data source path URL : <https://www.kaggle.com/rtatman/fraudulent-email-corpus>

This dataset is a collection of more than 2,500 "Nigerian" Fraud Letters, dating from 1998 to 2007.

These emails are in a single text file. Each e-mail has a header which includes the following information:

1. Return-Path: address the email was sent from
2. X-Sieve: the X-Sieve host (always cmu-sieve 2.0)
3. Message-Id: a unique identifier for each message
4. From: the message sender (sometimes blank)
5. Reply-To: the email address to which replies will be sent
6. To: the email address to which the e-mail was originally set (some are truncated for anonymity)
7. Date: Date e-mail was sent
8. Subject: Subject line of e-mail
9. X-Mailer: The platform the e-mail was sent from
10. MIME-Version: The Multipurpose Internet Mail Extension version
11. Content-Type: type of content & character encoding
12. Content-Transfer-Encoding: encoding in bits
13. X-MIME-Autoconverted: the type of autoconversion done
14. Status: r (read) and o (opened)

**Acknowledgement** : Radev, D. (2008), CLAIR collection of fraud email, ACL Data and Code Repository, ADCR2008T001, <http://aclweb.org/aclwiki>

Note : referred in 3\_Pattern\_By\_RegEx code file

Data Set to extract data from API as data source :

Data source path URL :

1. Open Notify
   1. ISS location - <http://api.open-notify.org/iss-now.json>
   2. Number of person in space - <http://api.open-notify.org/astros.json>
2. Alpha Vantage

<https://www.alphavantage.co/documentation/>

Alpha Vantage APIs are grouped into five categories and out of them, I have utilized Time Series and Economic Indicators and Fundamental Data groups of APIs by generating and using person key to implement the use case to extract data from API as source.

2.1 Time Series Intraday - <https://www.alphavantage.co/query?function=TIME_SERIES_INTRADAY&symbol=IBM&interval=5min&apikey=demo>

2.2 Company Overview - <https://www.alphavantage.co/query?function=OVERVIEW&symbol=IBM&apikey=demo>

2.3 Inflation - <https://www.alphavantage.co/query?function=INFLATION&apikey=demo>

2.4 Inflation Expectation - <https://www.alphavantage.co/query?function=INFLATION_EXPECTATION&apikey=demo>

Note : referred in 4\_API\_As\_DataSource code file

Data Set to merge you tube statistics for different countries :

Data source path URL : <https://www.kaggle.com/datasnaek/youtube-new>

This dataset includes several months (and counting) of data on daily trending YouTube videos. Data is included for the CA (Canada) and GB (Great Britain) with up to 200 listed trending videos per day.

Note : referred in 5\_Merge\_Data\_Frames code file

**Acknowledgement :** This dataset was collected using the YouTube API.

Data Set to visualize statistics using advanced and rich UI charts :

Data source path URL :

GBvideos - <https://www.kaggle.com/datasnaek/youtube-new>

Loan Prediction - <https://www.kaggle.com/altruistdelhite04/loan-prediction-problem-dataset>

Note : referred in 6\_Visualization\_By\_Charts code file

Implementation Process

Project got implemented in 5 steps as follows :

1. Input Data Collection
2. Input Data Preparation/Cleaning
3. Feature engineering and visualizing the data to generate insights
4. Execute machine learning algorithms by splitting train and test data
5. Hyperparamter tuning

Note :

please refer 7\_ML\_SL\_HyperParamTune.py code file for detailed coding steps

Please refer Input Data directory for data sets used as input data.

Please refer Output Data directory for parsed data generated as output data.

1. Input Data Collection :

Extract data from CSV by using pandas data frame and verify data by checking shape, head, info and statistics by describe.

1. Input Data Preparation/Cleaning :

Perform data cleaning steps as -

* Drop unnecessary column
* Check columns having missing values
* fill the missing value with most frequent value for each column
* verify that data is still have missing values
* extract cleaned data for verification to Output Data directory

1. Feature engineering and visualizing the data

* split data to categorical and numerical data
* converting categorical values to numbers
* add total income column as rich feature
* checking manipulated dataset for validation
* extract engineered data for verification
* Plot matplotlib histogram chart for each numerical column
* Plot Seaborn Countplot - Loan status distribution by each category
* plot seaborn chart - catplot with categorical variables
* plot seaborn chart - pairplot with categorical variables
* plotting the correlation matrix

Conclusion - We can clearly see that Credit\_History has the highest correlation with Loan\_Status a positive  
# correlation 0.540556. Therefore, our target value is highly dependent on the column 'Credit History'.

1. Execute machine learning algorithm by splitting train and test data

Divide the dataset into two variables X as the features and y as the Loan\_Status the target value we want to predict.

Models we will use to predict the target value : Random Forest, Decision Tree, XGBoost, Logistic Regression

* Create arrays for the features and the response variable
* split train and test data
* Create ML algorithm score function
* call algo score function and execute types of ML and find accuracy score
* print all ML algorithm's accuracy result
* cross validation

1. Hyperparameter tuning

Perform model selection using Grid Search for the optimal hyperparameters of the model.

* Hyperparameter tuning with GridSearchCV
* Hyperparameter tuning with RandomizedSearchCV

Note : please refer below attached pdf file to see console output printed by executing all the implementation steps.



Note : Parsed data (output data) is also available in Output Data directory. 7\_ML\_LoanAmount\_CleanedData.csv

7\_ML\_LoanAmount\_EngineeredData.csv

7\_ML\_Decision\_Tree.csv

7\_ML\_Logistic\_Regression.csv

7\_ML\_Random\_Forest.csv

7\_ML\_XGBoost.csv

**Note : Apart from main Task 7 for project, here are details for the tasks which got performed to complete all the tasks mentioned in assessment.**

Task for importing data, handle missing values and duplicates :

Please refer code file named as 2\_CSV\_As\_DataSource\_Data\_Cleaning.py where netflix data got cleaned and cleaned version is available in output data folder named as 2\_Netflix\_Cleaned\_Data.csv

Task for finding spam email sender details using Reg Ex :

Please refer code file named as 3\_Pattern\_By\_RegEx.py where find pattern for spam email sender/receiver email and name, subject, date on which mail was sent and email body.

Parsed data is available in output data folder named as 3\_Parsed\_Fradulent\_Emails.text

Task to access data from API as data source :

Please refer code file named as 4\_API\_As\_DataSource.py where multiple API got implemented using key and without key.

Parsed data is available in output data folder named as 4\_Parsed\_API\_Data.csv

Task to merge data frames :

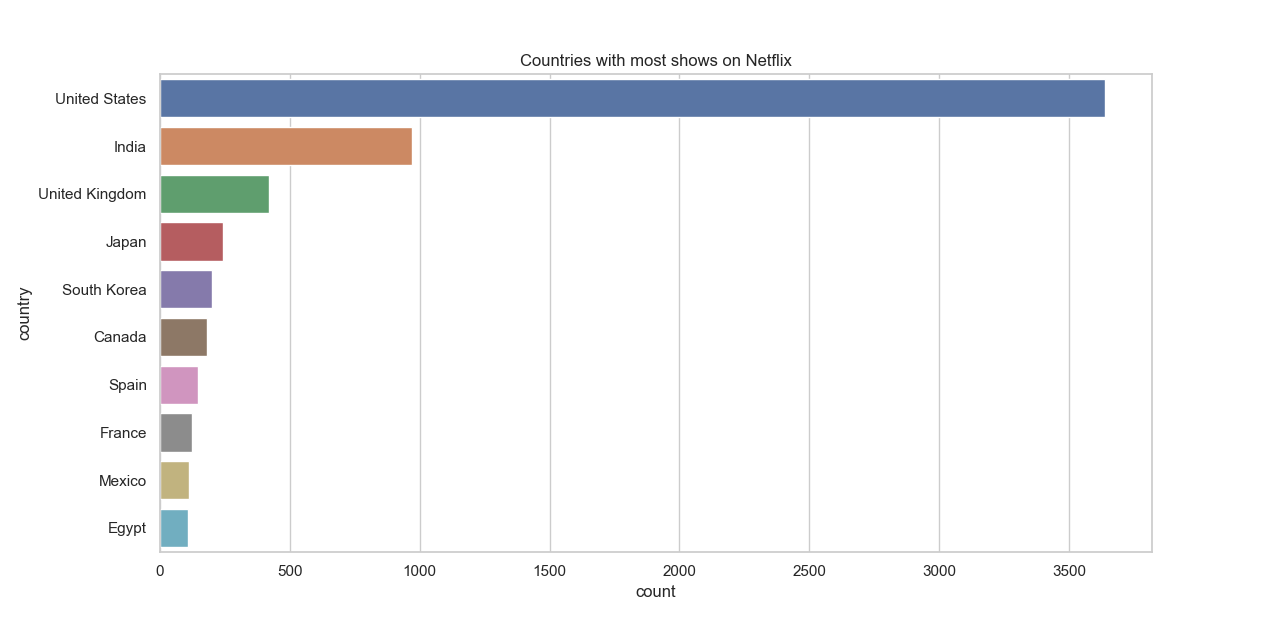
Please refer code file named as 5\_Merge\_Data\_Frames.py where canada and great britain country you tube data got merged by using multiple techniques as concat, join and merge.

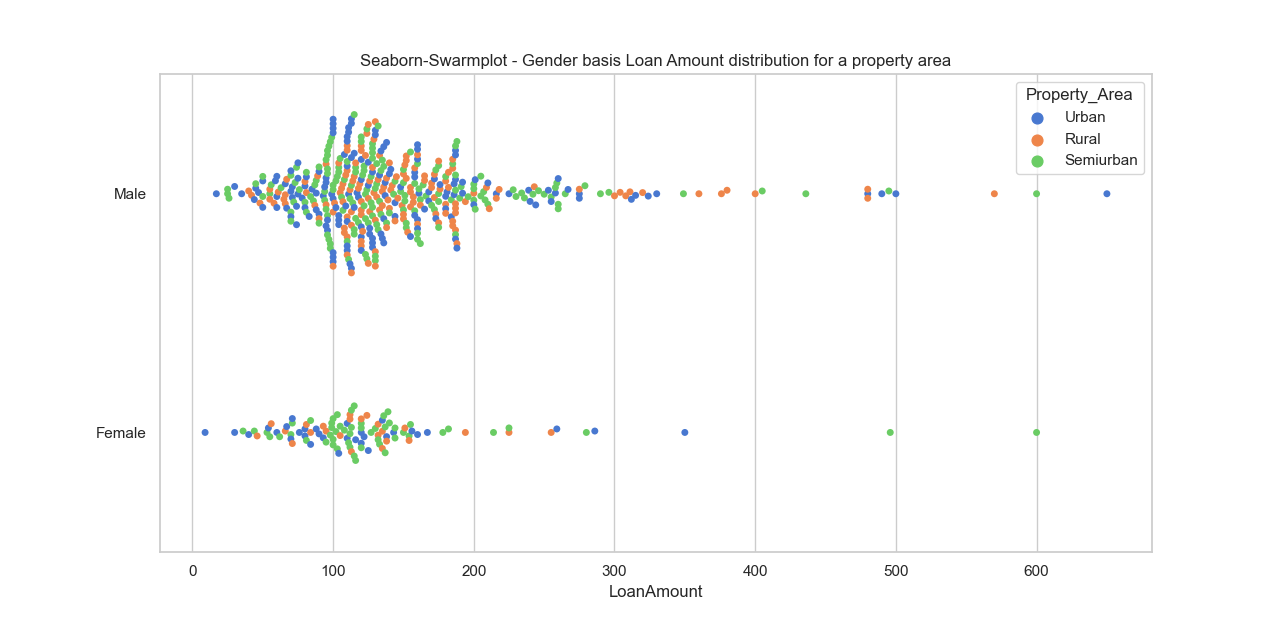
Parsed data is available in output data folder named as 5\_Merged\_CAGBvideos\_Data.csv

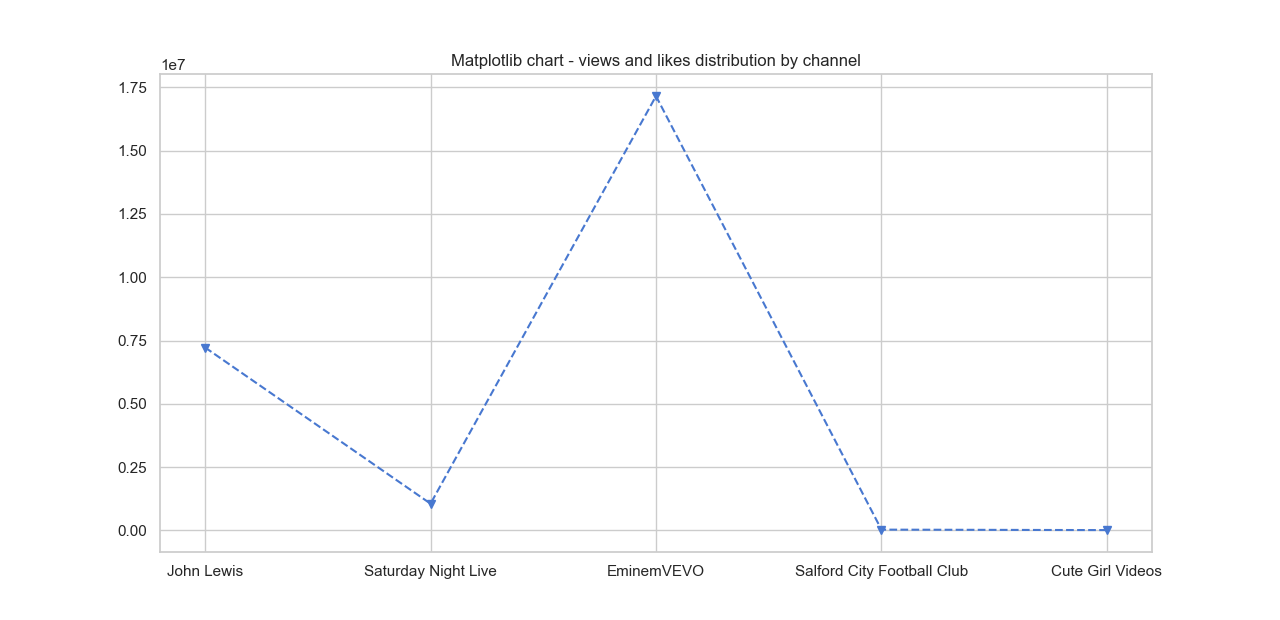
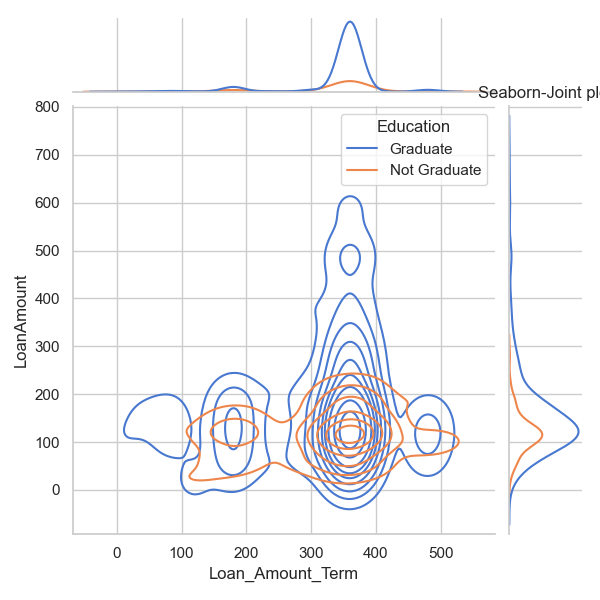
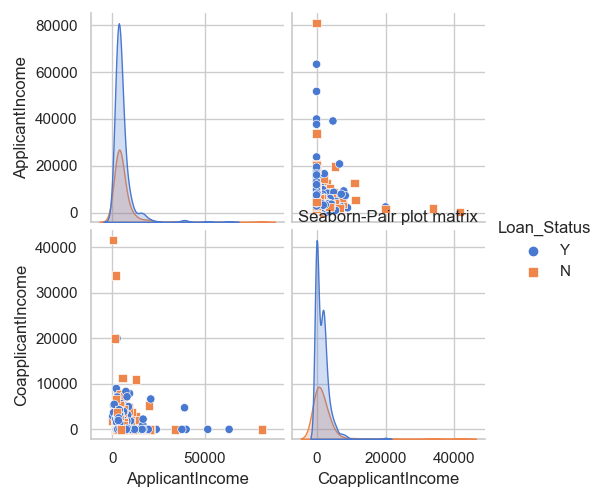
Task to visualize the data by advance charts :

Please refer code file named as 6\_Visualization\_By\_Charts.py and 3\_Pattern\_By\_RegEx.py where data got visualized by multiple charts using matplot and seaborn library.

Few samples are as follows :



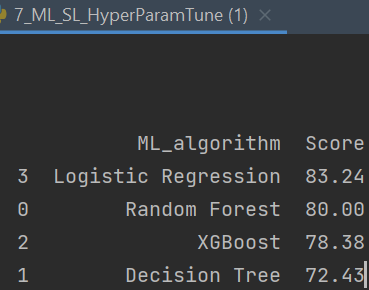




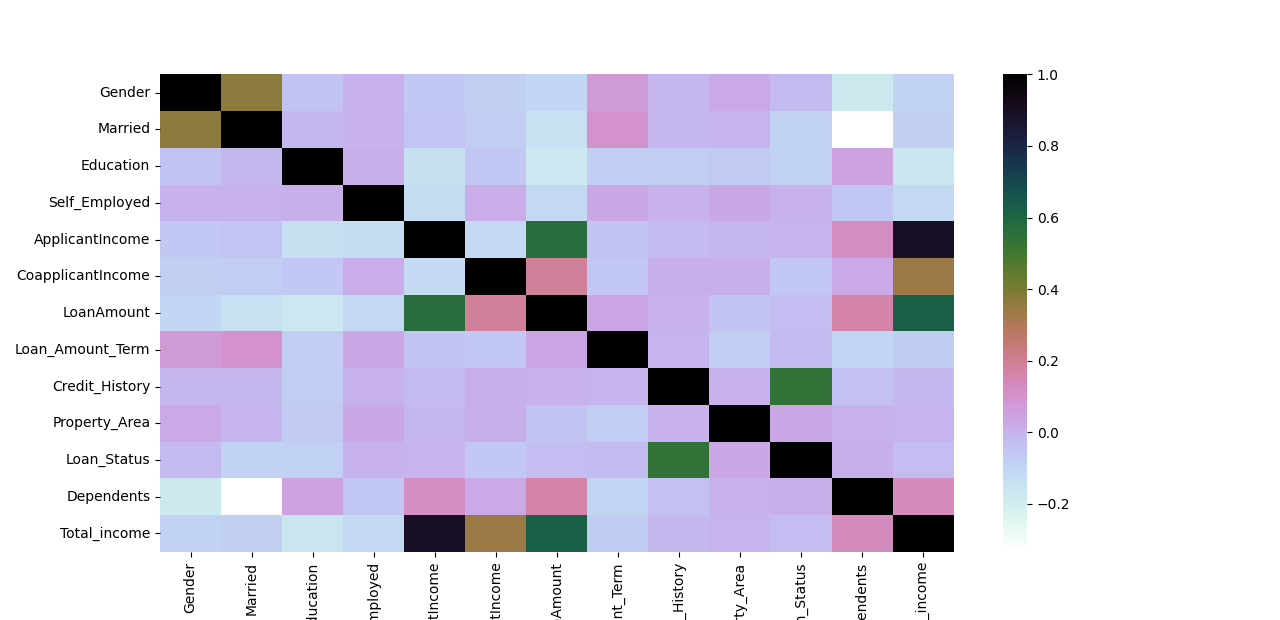
Results

We can clearly observe following output after executing all the implementation steps.

1. Most accurate algorithm is 'The Logistic Regression' with approximately 83%.



1. Correlation Matrix for all features



Correlation matrix shows the relation between all features with each other.

Here, value equal to

-1 indicates a perfectly negative linear correlation between two variables,

And equal to 0 indicates no linear correlation between two variables,

And 1 indicates a perfectly positive linear correlation between two variables.

By looking above correlation matrix, this could be easily observed that Credit\_History has green box which means the positive and highest correlation with Loan\_Status **a positive correlation of** **0.540556**. Therefore, **target value is highly dependent on the column 'Credit History'**.

1. Loan Status distribution using seaborn catplot by Gender and Property Area.

Catplot chart is showing the loan status distirbution across three buckets of property area with respect to Genders.

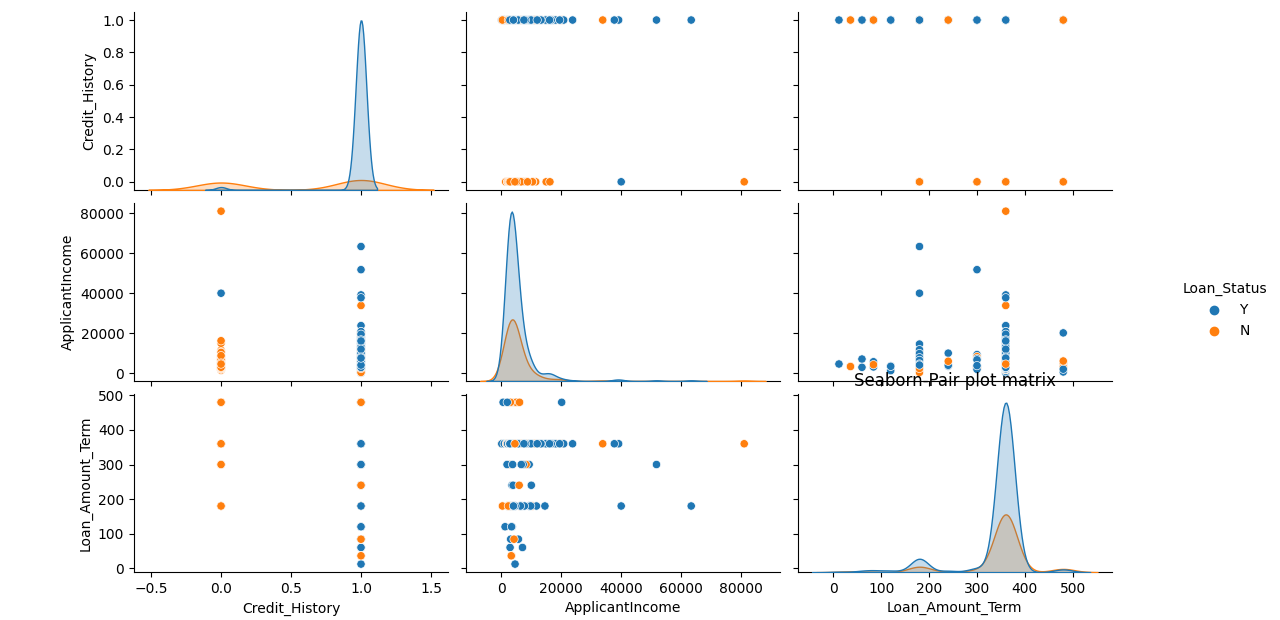
This is an advance chart which clearly depict that Loan got approved majorly in semi urban area and then urban and at last in rural areas.

It also distinguishes the distribution between genders which says that major win with Male applicants.



1. Loan Status Distribution using seanborn pairplot matrix by Loan Amount, Credit History and Applicant Income

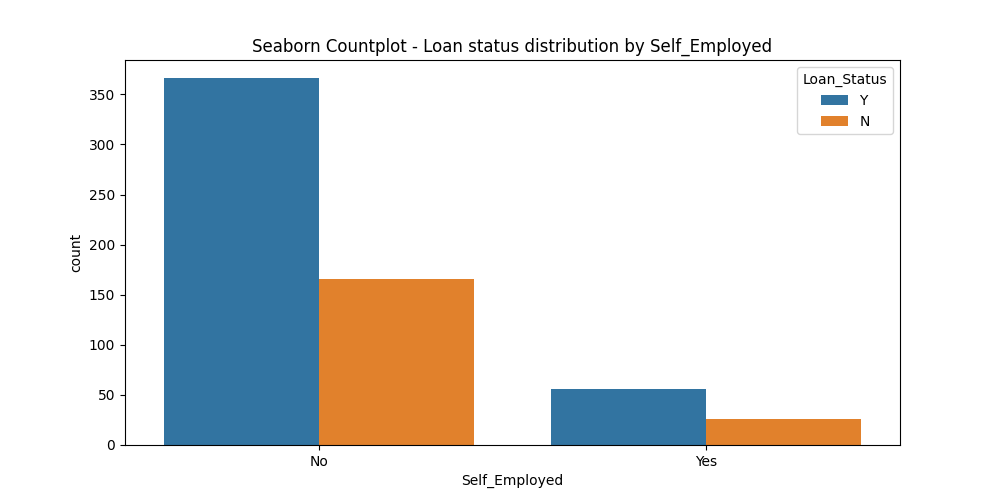
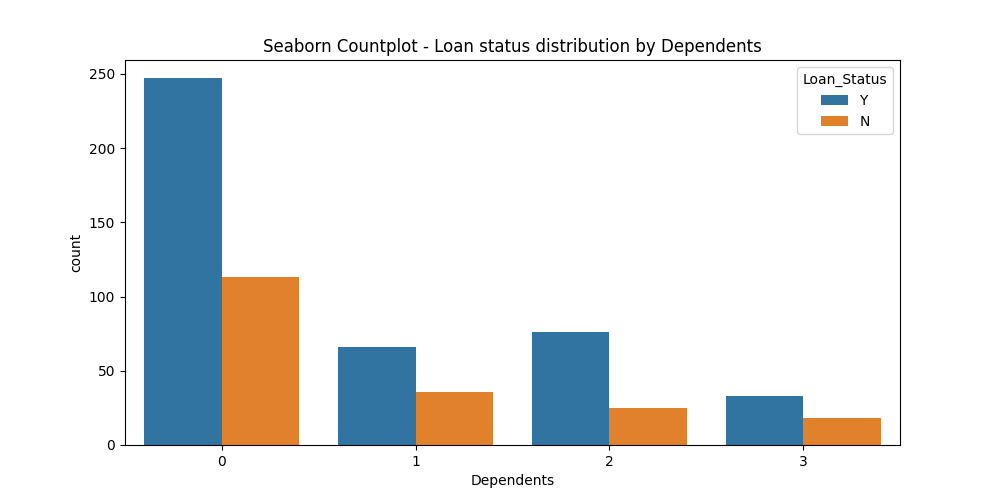
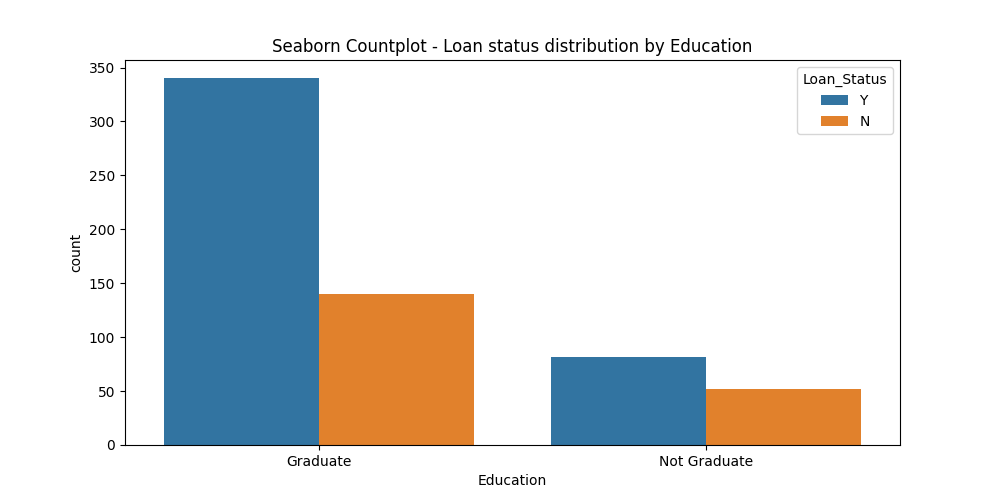
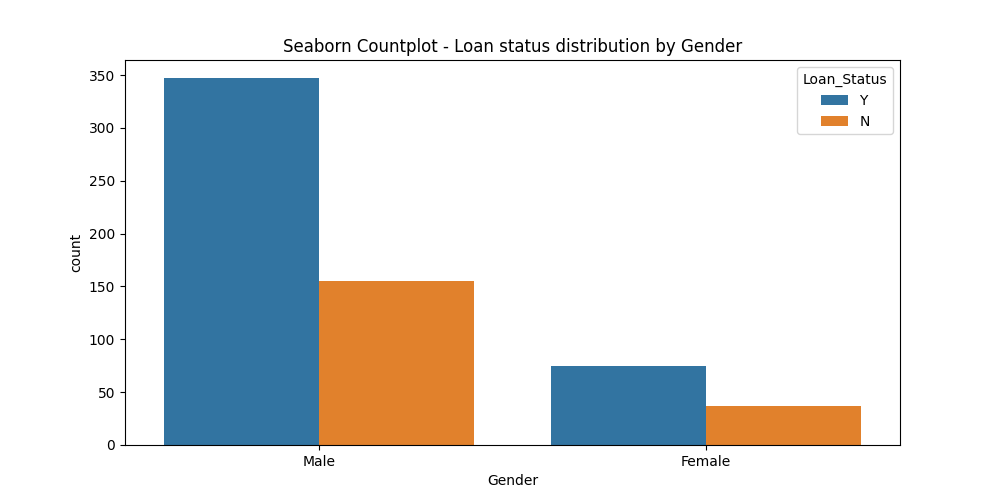
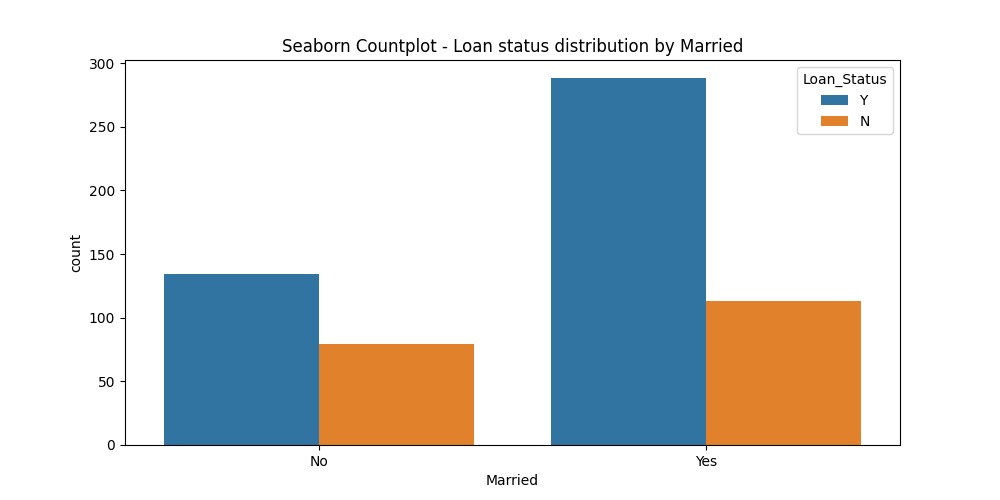
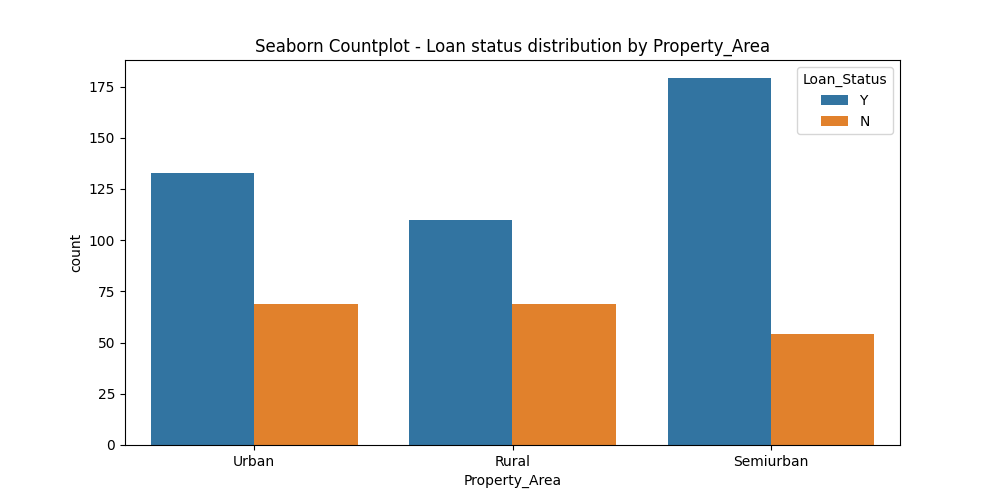
Pairplot chart is best chart to show cross dependency against each other. Here, we can see that loan approval status is majorly with Applicant whose income is more than 20000 approx. and loan amount term is more than 300 approx. with credit history 1.



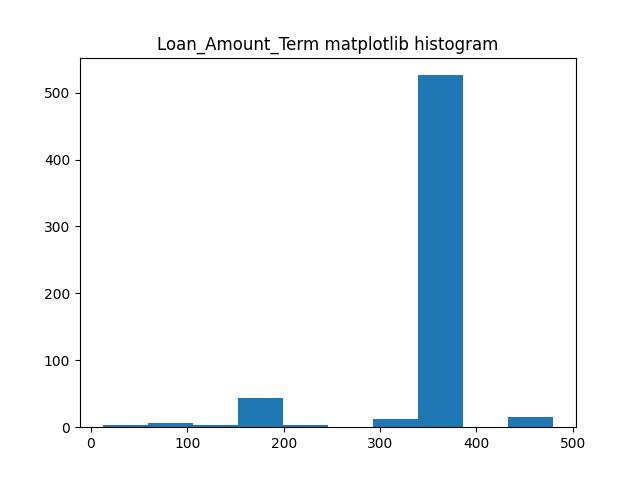
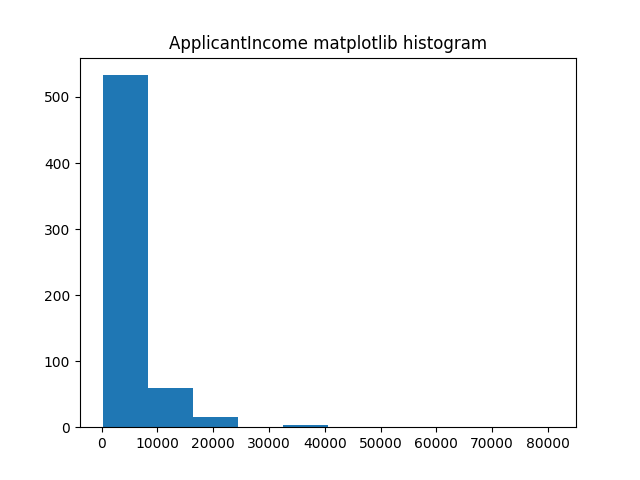
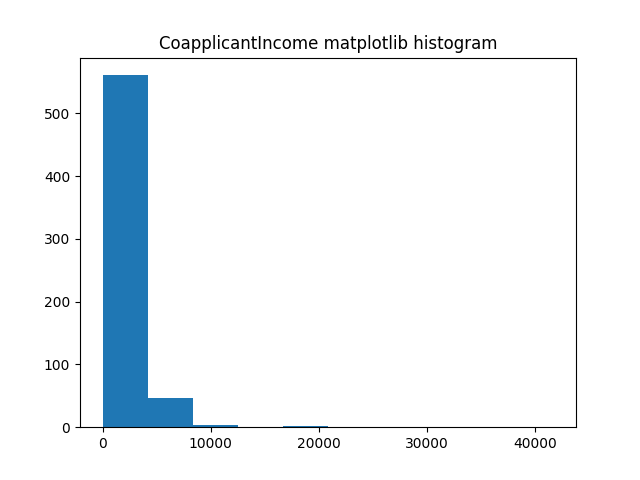
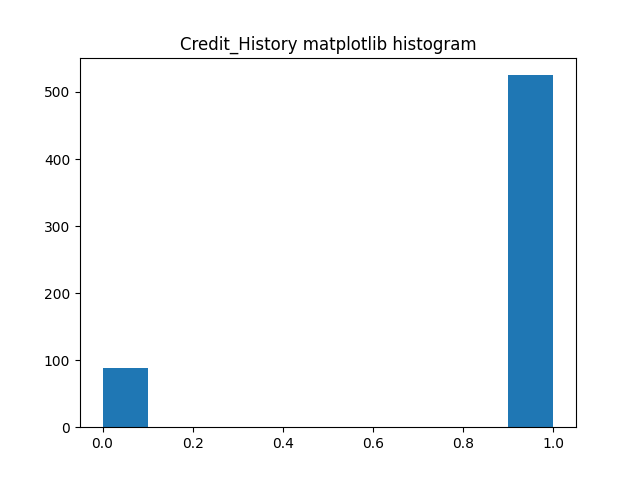
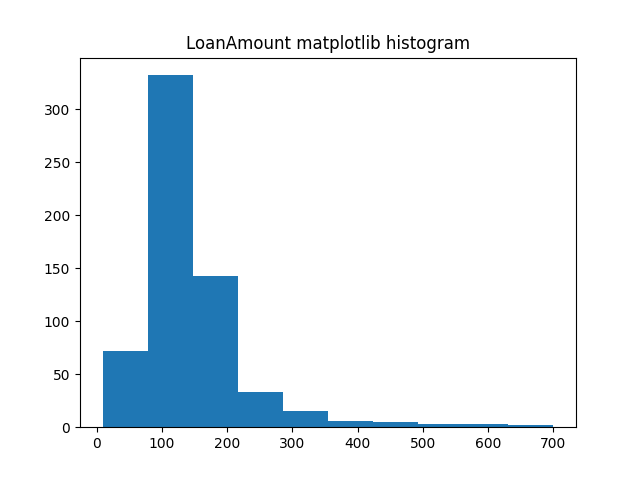
1. Loan Status distribution countplot using seaborn lib by all categorical features

Insights from following charts -

* Loan got approved is majorly in Semiurban area and then in urban.
* Loan got approved for applicants who are married and that too Male.
* Educated applicants who are graduated got approval more than not graduated applicants.
* Applicants who are having no dependents got approval in comparison to having 1 or more dependents.
* Self employed applicants got more approval for loan.



1. Histogram Charts using matplot lib for numerical features



Insights

1. Credit\_History with value 0.540556 as is most important variable due to its high correlation with Loan\_Status.
2. Most accurate algorithm is 'The Logistic Regression' with approximately 83%.
3. Loan got approved majorly in semi urban area and then urban and at last in rural areas.
4. For loan approval, major win is for Male applicants
5. Loan got approved for applicants who are married and that too Male.
6. Educated applicants who are graduated got approval more than not graduated applicants.
7. Applicants who are having no dependents got approval in comparison to having 1 or more dependents.
8. Self employed applicants got more approval for loan.
9. Major loan amount term is more than 300 for whom loan got approved.
10. Applicant whose income is more than approx. 20000/- are having higher chances to get loan approved.

Other insights from nextflix data :

USA is leading country with most shows on netflix with more than 3500 shows and then India with approx. 1000 shows (refer chart generated by running code at file 3\_Pattern\_By\_RegEx.py)

References

# Chart reference :

matplotlib gallery - <https://matplotlib.org/stable/gallery/index.html>

seaborn gallery - <https://seaborn.pydata.org/examples/index.html>

Scatter plot - <https://seaborn.pydata.org/examples/scatterplot_categorical.html>

Stripplot -<https://seaborn.pydata.org/generated/seaborn.stripplot.html?highlight=stripplot#seaborn.stripplot>

Joine Kde - <https://seaborn.pydata.org/examples/joint_kde.html>

Scatterplot matrix - <https://seaborn.pydata.org/examples/scatterplot_matrix.html>

color palette - <https://matplotlib.org/stable/gallery/color/named_colors.html>

line styles - <https://matplotlib.org/stable/gallery/lines_bars_and_markers/linestyles.html>

markers - <https://matplotlib.org/stable/api/markers_api.html>