April 13, 2025

**Loan Approval Prediction using Applied Machine learning techniques.**

**Submitted by:**

Pratyush Acharya(G24AIT2027)

&

Gourav Ray (G24AIT2111)

**Abstract:**

In today’s financial landscape, predicting whether a loan will be approved plays a vital role in helping lenders evaluate the risk involved in giving credit. A reliable prediction system allows financial institutions to make smarter lending decisions and reduce the chances of defaults.

This project explores how machine learning techniques can be applied to forecast loan approvals. To begin with, we gathered and cleaned historical loan data containing key applicant details such as income, job status, previous loan outcomes etc. The dataset was split into two parts—one for training the models and another for testing. The primary goal was to use the training data to build models that could accurately determine loan approval status in the test set.

We experimented with several machine learning algorithms, including logistic regression, random forests, gradient boosting and XGBoost. After evaluating their performance, we found that most models provided reasonable accuracy, with the Gradient Boosting algorithm outperforming the others with a great F1 score of 0.99 and training time of 36 seconds on the present training dataset.

1. **Objective:**

The main objective of predicting loan approval using machine learning algorithms is to accurately assess the risk of extending credit to potential borrowers. This can help lenders make informed decisions about which borrowers to approve for loans and minimize the risk of default.

The aims of this exercise are:

* To improve the efficiency of the loan approval process by automation of the reviewing process of loan prediction files. There have been 10 critical financial parameters used for this model building.
* To reduce the risk of human bias in loan approval decisions because loan prediction would be in objective data using ML algorithms rather than being biased on subjective judgments.
* To identify patterns and trends in the data.
* To improve customer experience by providing more accurate predictions, lenders can improve the experience of potential borrowers and increase customer satisfaction.
* To provide explainable decision making through feature importance analysis.

We have used python programming and libraries such as sci-kit-learn, XGBoost etc. for the entire exercise.

1. **Data Collection and Preprocessing:**
   1. **Dataset composition:**

* 4269 historical loan application data from Indian banks.
* 11 independent features considered for input features.

**2.2** **Preprocessing:**

1. We removed unnecessary leading/trailing white spaces from all the text columns.
2. Target column(loan\_status) had values as Approved and Rejected which we converted to binary format (1 and 0).
3. **Exploratory data analysis:**

**Exploratory data analysis helps in understanding the data distribution, spotting anomalies, and identifying patterns**, which are essential for selecting the right features and models. **It reveals relationships and correlations between variables,** guiding effective feature engineering and model interpretation. **By visualizing and summarizing the data,** EDA allows early detection of data quality issues like missing values and outliers, helping improve model performance.

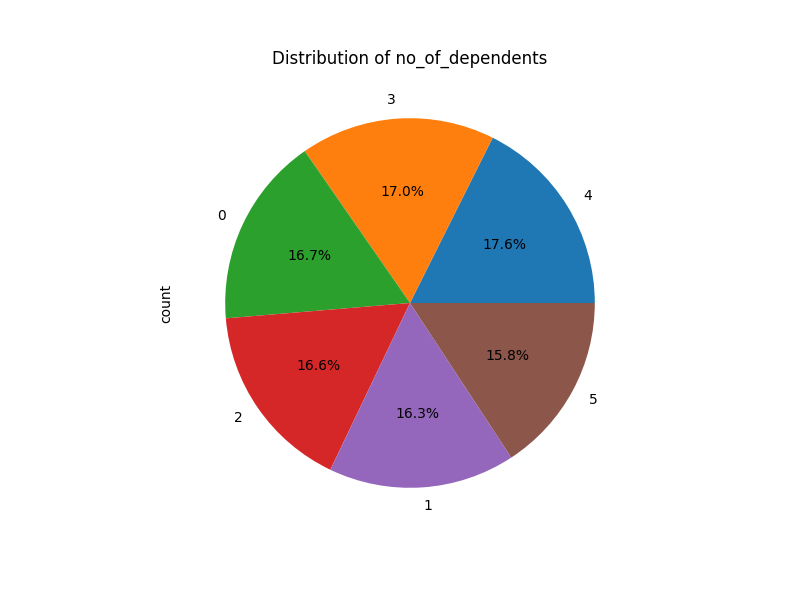
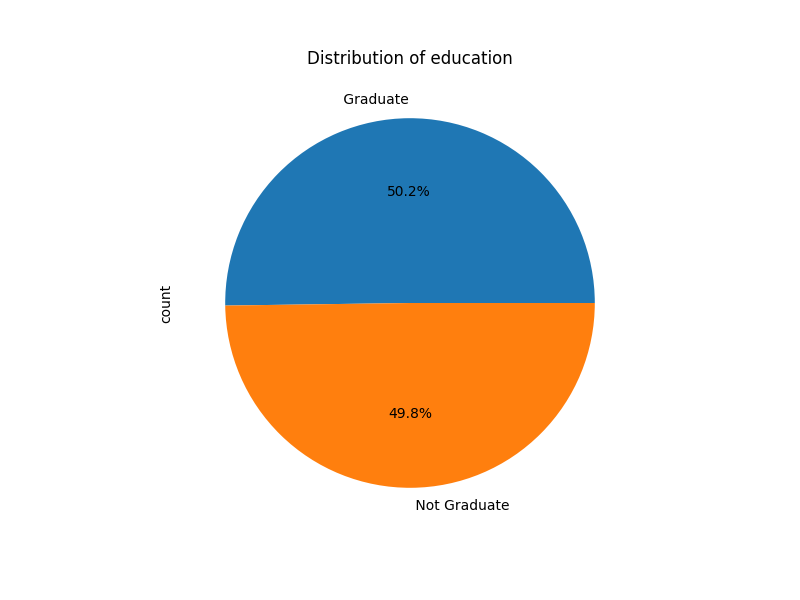
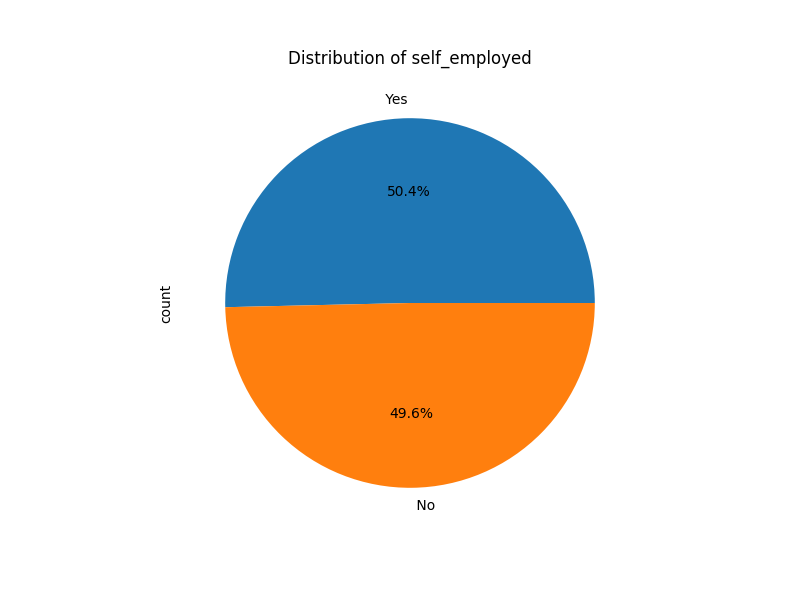
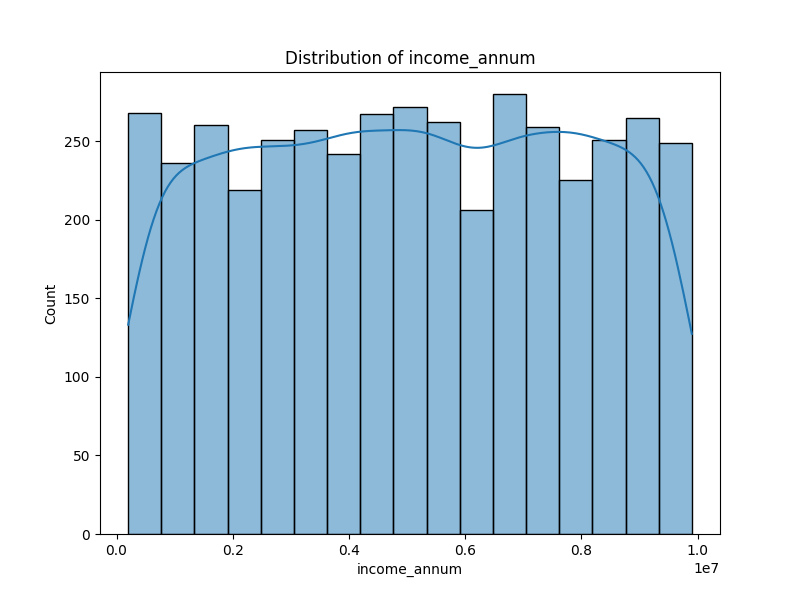
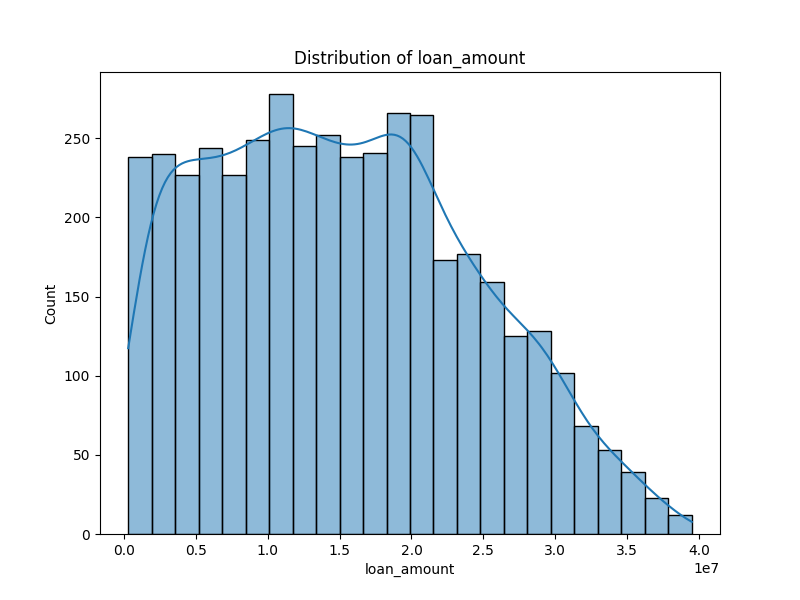
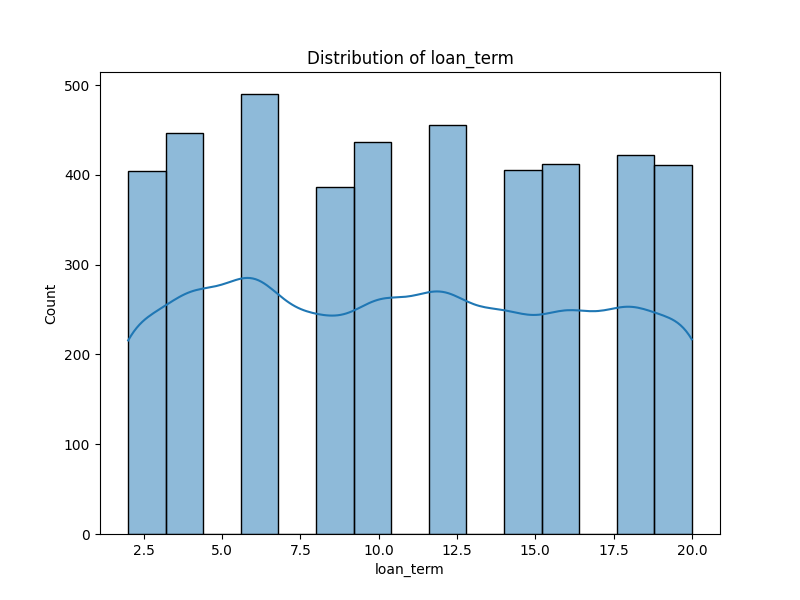
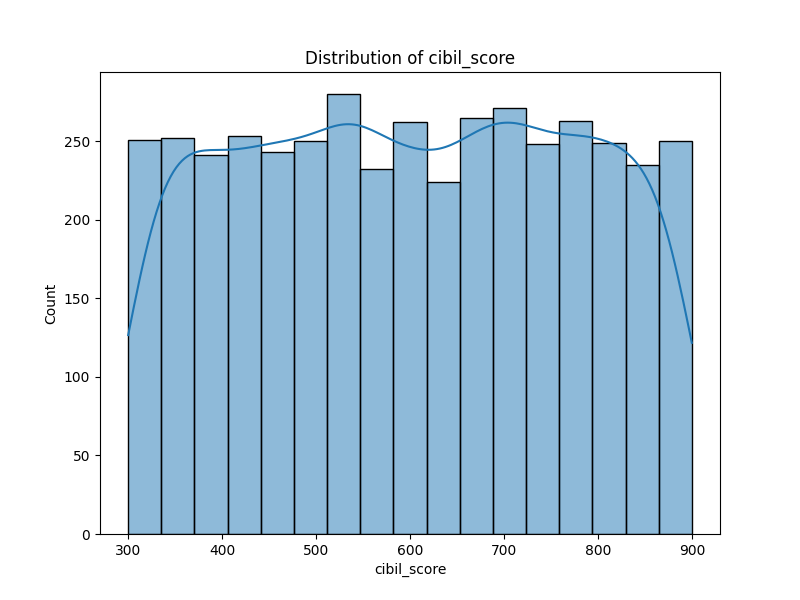
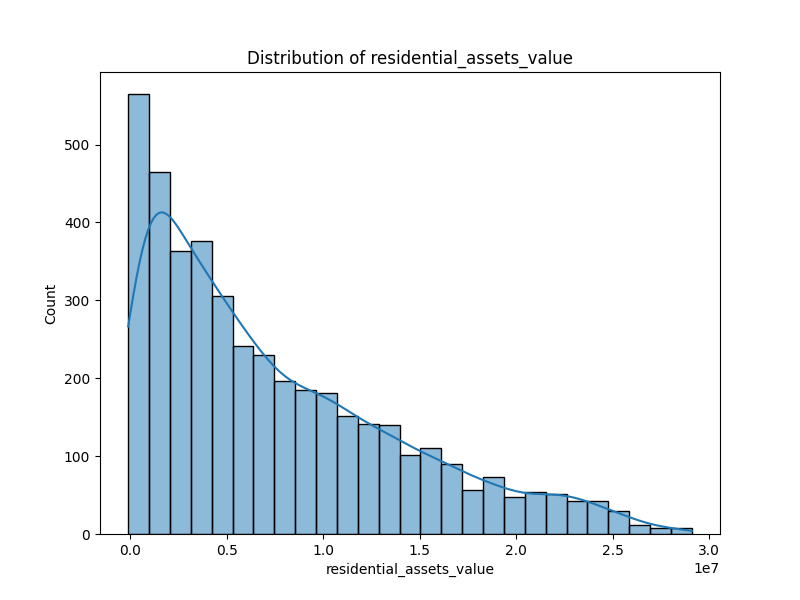
We systematically analysed the loan approval dataset through the following structured approach:

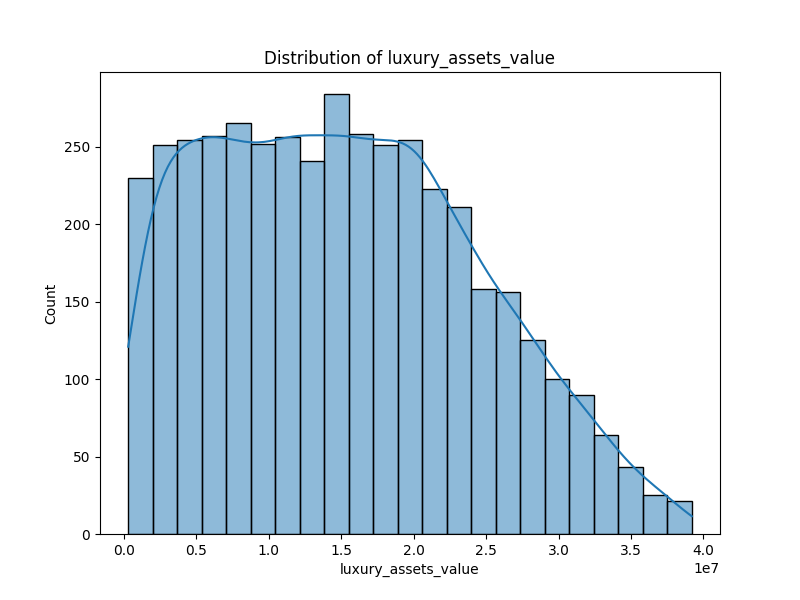
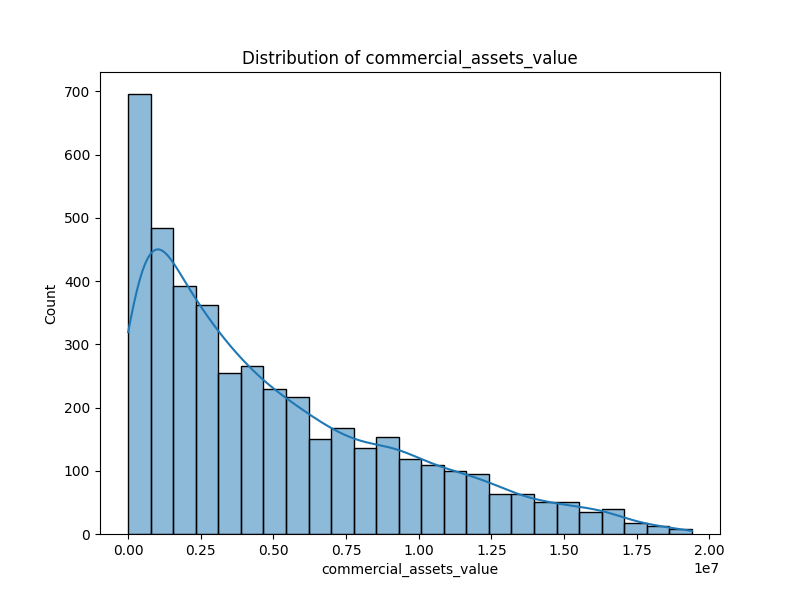
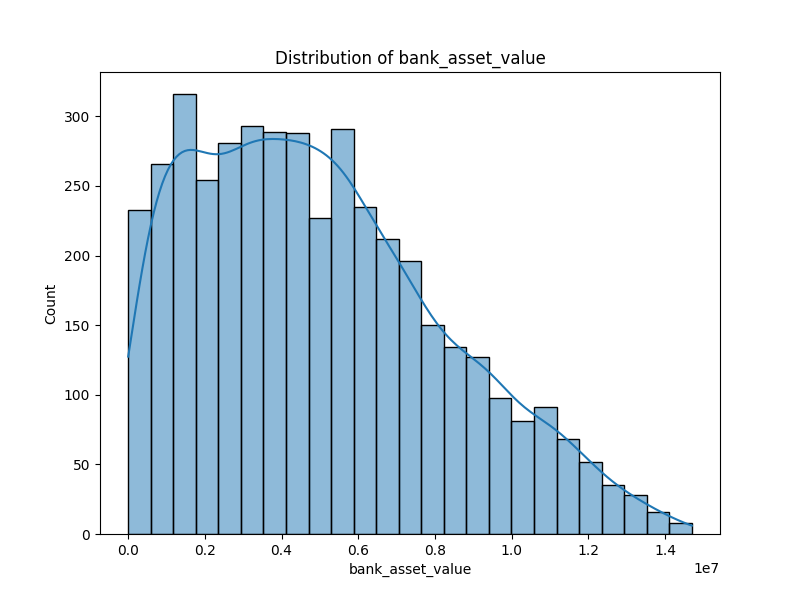
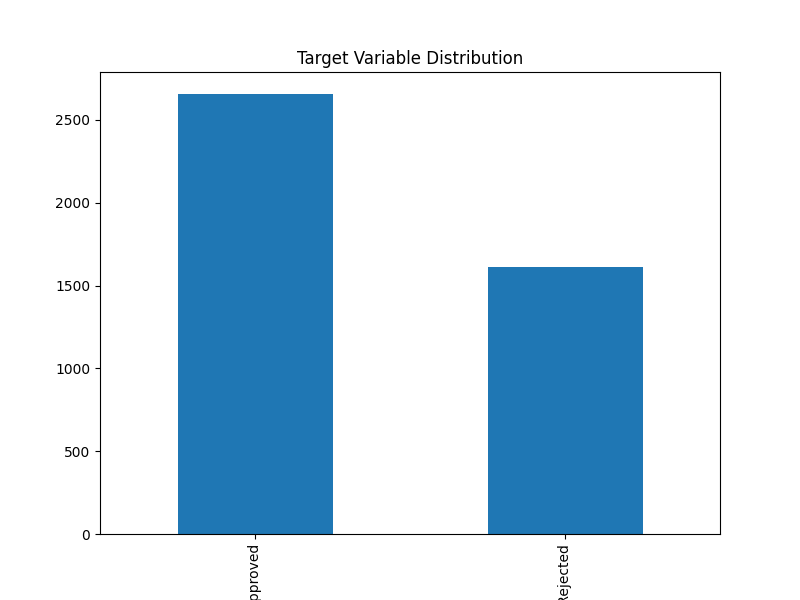
* 1. **Descriptive Statistics:**

Below table shows the statistical parameters like central tendency measures(**mean, median), dispersion metrics(standard deviation, 25th percentile, 75th percentile) and extreme values(minimum and maximum)** values for each feature. Along with that, for categorical features, we have shown no. of unique values along with top values and their frequency.



We have also attached charts below, where each feature has been explored in detail to understand the pattern formed.



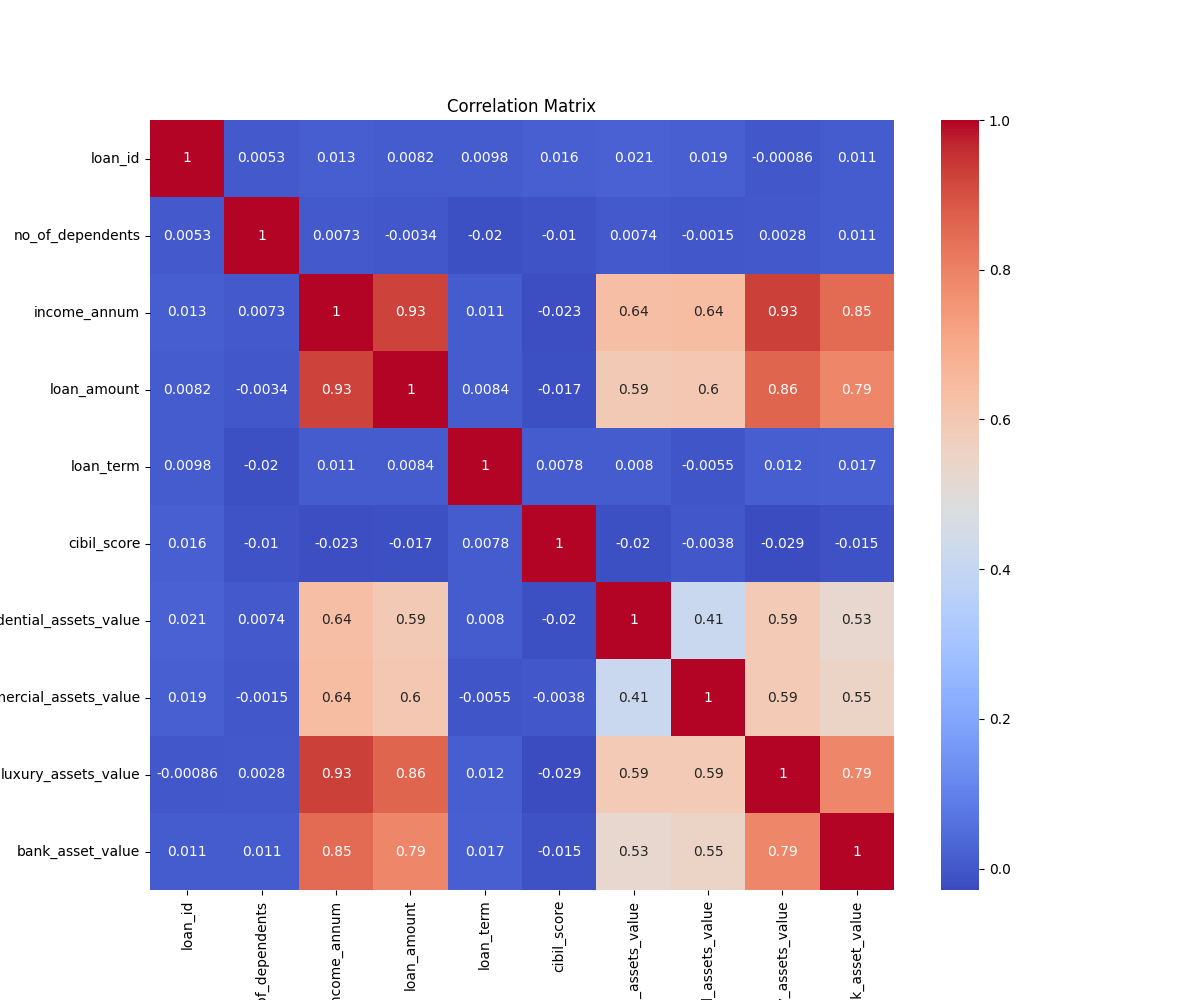
Below table captures the no. of missing values in the dataset.

|  |  |  |
| --- | --- | --- |
| **Feature Name** | **Missing count** | **Missing percentage** |
| **loan\_id** | 0 | 0 |
| **no\_of\_dependents** | 0 | 0 |
| **education** | 0 | 0 |
| **self\_employed** | 0 | 0 |
| **income\_annum** | 0 | 0 |
| **loan\_amount** | 0 | 0 |
| **loan\_term** | 0 | 0 |
| **cibil\_score** | 0 | 0 |
| **residential\_assets\_value** | 0 | 0 |
| **commercial\_assets\_value** | 0 | 0 |
| **luxury\_assets\_value** | 0 | 0 |
| **bank\_asset\_value** | 0 | 0 |
| **loan\_status** | 0 | 0 |

We have captured the distribution of cardinal values in the categorical features along with the target variable in the below tables.



* 1. **Correlation analysis:**

Correlation analysis helps us identify the strength and direction of relationships between variables, aiding in effective feature selection. It is crucial for detecting multicollinearity, which can impact model performance and interpretability, especially in linear models. We have used **Pearson Correlation Coefficient** for understanding the correlation between the independent variables. Below is the heatmap which shows the correlation effect among the variables.

As observed from the heatmap, **loan\_amount and income\_annum exhibit a strong positive correlation,** indicating a potential overlap in the information they convey. Similarly, **luxury\_asset\_value also shows a high correlation with loan\_amount**. In such cases, it is prudent to eliminate one of the highly correlated variables during model development by applying a correlation threshold. **This helps in reducing multicollinearity without compromising model performance,** as the essential information is already captured by the remaining variables.

For our exercise, we have not removed any feature from the feature set.

1. **Feature Engineering:**

We meticulously designed the data transformation pipeline to maintain the integrity of financial data while addressing common challenges in loan application datasets. The approach ensures robustness against data leakage while preserving critical information through the following structured workflow:

**4.1** **Data Segregation Protocol:**  
Prior to any transformations, the dataset underwent an initial split into training (80%) and testing (20%) subsets using stratified sampling. This crucial first step guarantees that:

* Statistical parameters for preprocessing derive exclusively from training data only.
* Test set remains completely isolated during model development.
* Class distribution (approved/rejected loans) remains consistent across splits.

**4.2 Pipeline Architecture:**  
We have established dual processing streams to handle distinct data types effectively:

*Numerical Feature Processing:*

* **Missing Value Resolution**: We have applied median imputation which replaces null values in financial metrics (loan amount, income), preserving robustness against outlier distortion.
* **Normalization**: Z-score standardization ensures equal feature contribution

Formula:    
Where

Z = Z-score

X = the data point

μ = mean of the data

σ = standard deviation of the data

* *μ* (mean) and *σ* (std. deviation) were calculated from training data only

*Categorical Feature Handling*

* **Null Value Treatment**: We have implemented most frequent category imputation to maintain the distribution pattern.
* **Dimension Expansion**: We enforced One-hot encoding for categorical features which converts discrete categories into binary vectors with handle\_unknown='ignore' to manage unseen test categories.

**4.3 Unified Transformation Framework**:

The ColumnTransformer orchestrates parallel processing of numerical and categorical streams, ensuring:

* Seamless integration of heterogeneous data types
* Atomic execution of all transformations
* Reproducible pipeline deployment

**4.4 Feature Traceability System:**  
post-transformation, we ensured the tracking mechanisms which preserve the interpretability:

* We retained original names for numerical features (e.g., cibil\_score, income\_annum)
* Encoded categorical features generated descriptive labels (e.g., education\_Graduate, self\_employed\_Yes)
* Comprehensive name mapping was stored for model diagnostics.

**4.5 Leakage Prevention Measures:**  
The refined implementation we implemented addressed initial data contamination risks through:

* Strict separation of fit/transform operations
* Pipeline fitting restricted to training data.
* Test set transformed using training-derived parameters.

1. **Model Training:**

We have developed the predictive model through a rigorous comparative analysis of 9 machine learning algorithms, implementing best practices in model development.

**5.1 Algorithm Portfolio:**

|  |  |
| --- | --- |
| Model Type | **Algorithms Tested** |
| Tree-Based | GBDT, CatBoost, XGBoost, LightGBM, Random Forest, Decision Tree |
| Linear | Logistic Regression |
| Probabilistic | Naive Bayes |
| Kernel-Based | SVM |

**5.2 Training Protocol:**

* **Hyperparameter Tuning**: Grid Search with 5-fold cross-validation.
* **Evaluation Metric**: F1-Score optimization (handling class imbalance)
* **Validation Strategy**:
  + Train/Test Split (80/20)
  + Stratified sampling for class distribution preservation
  + Comprehensive metrics tracking for both sets.

**5.3 Key Performance Monitoring:**



|  |  |
| --- | --- |
| **Algorithm** | **best\_params** |
| Gradient Boosting | {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} |
| CatBoost | {'depth': 5, 'iterations': 200, 'learning\_rate': 0.1} |
| XGBoost | {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} |
| Decision Tree | {'max\_depth': None, 'min\_samples\_split': 2} |
| LightGBM | {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} |
| Random Forest | {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 100} |
| Naive Bayes | {'var\_smoothing': 1e-09} |
| SVM | {'C': 10, 'kernel': 'rbf'} |
| Logistic Regression | {'C': 0.1, 'solver': 'saga'} |

**5.4 Performance Observations:**

* **Tree Dominance**: Gradient Boosting outperformed all models (99.16% F1 score) with optimal depth (5), learning rate (0.1) and 200 unique decision trees.
* **Speed vs Accuracy**: LightGBM achieved 98.31% F1 in 12s vs CatBoost's 98.69% in 20s
* **Overfitting Signs**:
  + All tree-based models showed perfect training scores (F1=1)
  + Test F1 gaps: 0.84% (Gradient Boosting) to 1.92% (Random Forest)
* **Baseline Comparison**:
  + Logistic Regression: 93.32% F1 (Lowest among tested)
  + Naive Bayes: 95.59% F1 (Surprisingly competitive)

**5.5 Feature Importance Analysis:**

Feature importance helps identify which input variables have the most influence on the model’s predictions. It supports better model interpretation and can guide effective feature selection and dimensionality reduction. Below table and chart explains the feature importance of all the features used in the model building given by the best model (**Gradient Boosting**).

|  |  |
| --- | --- |
| **Feature** | **Importance** |
| cibil\_score | 0.82 |
| loan\_term | 0.09 |
| loan\_amount | 0.05 |
| income\_annum | 0.03 |
| luxury\_assets\_value | 0.01 |
| commercial\_assets\_value | 0.00 |
| residential\_assets\_value | 0.00 |
| bank\_asset\_value | 0.00 |
| no\_of\_dependents\_5 | 0.00 |
| no\_of\_dependents\_2 | 0.00 |
| no\_of\_dependents\_3 | 0.00 |
| no\_of\_dependents\_0 | 0.00 |
| no\_of\_dependents\_1 | 0.00 |
| self\_employed\_Yes | 0.00 |
| no\_of\_dependents\_4 | 0.00 |
| self\_employed\_No | 0.00 |
| education\_Not Graduate | 0.00 |
| education\_Graduate | 0.00 |

**5.5.1 Critical Decision Factors**:

* **CIBIL Score** having **82.13%** feature importance shows high relative importance in probability calculation.
* **Loan Terms** having **8.91%** feature importance explains loan duration plays a critical role for risk assessment.
* Financial Capacitylike **Loan Amount** and **Annual income** contributing **4.6%** and **2.61%** respectively towards feature importance.
* Categorical featureslike **Self-Employment** and **Education** have very minimal impact on decision making.

**5.6 Model metrics:**

**5.6.1 Confusion Matrix:**

|  |  |  |
| --- | --- | --- |
|  | **Predicted Reject** | **Predicted Approve** |
| Actual Reject | TN=316 | FP=7 |
| Actual Approve | FN=2 | TP=529 |

**Key Ratios**:

* **Test Accuracy:** 0.989
* **Test Precision:** 0.987
* **Test Recall:** 0.996
* **Test F1 Score:** 0.992
* **False Positive Rate**: 1.32% (7/316+7)
* **False Negative Rate**: 0.38% (2/529+2)
* **Approval Accuracy**: 99.23% (529/534)

1. **Model Deployment:**

Model deployment is the process of integrating a trained machine learning model into a production environment where it can make real-time or batch predictions. A successful deployment ensures that the model is accessible via APIs or services and can handle incoming data efficiently and reliably.

Monitoring deployed models, it is essential to detect performance drift, data anomalies, or system issues over time. Hence monitoring the deployed models is an essential final step in the entire model development life cycle.

We have created a streamlit application which has been deployed in streamlit public cloud.

URL: <https://smart-loan-application-reviewer-v01.streamlit.app/#smart-loan-application-reviewer>

Below are few of the snapshots captured from the application.

**Landing page:**

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**Forecast page when loan is approved:**

A screenshot of a video game

Description automatically generated

**Forecast page when loan is approved:**

A screenshot of a video game

Description automatically generated

1. **References:**
   1. **Scikit-learn (ML algorithms and preprocessing)**  
      🔗 https://scikit-learn.org/stable/
   2. **Kaggle (datasets, notebooks, and community solutions)**  
      🔗 <https://www.kaggle.com/competitions/loan-prediction-problem>
   3. **UCI Machine Learning Repository (datasets)**  
      🔗 <https://archive.ics.uci.edu/ml/datasets.php>
   4. **Towards Data Science (practical guides and explanations)**  
      🔗 <https://towardsdatascience.com/>
   5. **Analytics Vidhya (domain-specific ML applications)**  
      🔗 https://www.analyticsvidhya.com/blog/2021/06/machine-learning-project-loan-prediction