AI Mid Semester – 1 Examination

Problem Statement : Credit Score Prediction

Name : Raja Yadav

University Roll: 202401100300193

Branch & Sec : CSE(AI)-C

Introduction to the Problem Statement

Background

Credit scores play a crucial role in determining an individual's financial credibility. Banks and financial institutions use credit scores to assess loan eligibility, interest rates, and credit limits. A low credit score may lead to loan rejection, while a high credit score can provide better financial opportunities.

Problem Statement

The objective of this project is to **predict an individual's credit score based on key financial parameters** such as **Age, Income, and Loan Amount**. Traditional methods of credit scoring often rely on complex rules and manual assessments, which can be time-consuming and subjective.

By leveraging **Machine Learning (ML) techniques**, we aim to develop a predictive model that can estimate a person's credit score with high accuracy. This model will help financial institutions automate and improve the credit assessment process, reducing human bias and increasing efficiency.

Challenges in Credit Score Prediction

1. **Non-Linear Relationships**: Credit scores do not have a simple mathematical relationship with age, income, or loan amounts.
2. **Feature Scaling**: Since income and loan amounts vary significantly, proper feature scaling is necessary.
3. **Model Selection**: Choosing the right ML model that balances accuracy and interpretability.

Proposed Solution

This project uses **Gradient Boosting Regressor (GBR)**, a powerful ensemble learning technique, to predict credit scores. The model is trained on real-world data and evaluated using **Mean Absolute Error (MAE), Mean Squared Error (MSE), and R^2 Score** to ensure accuracy and reliability.

Methodology & Algorithm Used

1. Methodology

Step 1: Data Collection

- **The dataset (credit_data.csv) contains customer information, including:**
    - **CustomerID (irrelevant for prediction)**
    - **Age**
    - **Income**
    - **LoanAmount**
    - **CreditScore (Target Variable)**

Step 2: Data Preprocessing

- **Removed CustomerID as it is not useful for prediction.**
- **Handled Missing Values by replacing them with the mean of the respective columns.**
- **Feature Scaling:**
    - **Since Income and LoanAmount have large values, we used StandardScaler() to normalize the features.**

Step 3: Splitting Data

- **Train-Test Split:**
    - **90% of data used for training (X_train, y_train).**
    - **10% used for testing (X_test, y_test).**
    - **train_test_split() from sklearn was used.**

Step 4: Model Selection

- **We chose Gradient Boosting Regressor (GBR) as the primary algorithm due to its:**
    - **Better handling of small datasets.**
    - **Ability to capture non-linear relationships.**
    - **Improved accuracy over Random Forest.**

Step 5: Model Training

- **Hyperparameters Used:**
  - **n_estimators=500 (number of boosting stages)**
  - **learning_rate=0.1 (controls contribution of each tree)**
  - **max_depth=5 (limits complexity to avoid overfitting)**
- **The model was trained on the X_train, y_train dataset.**

Step 6: Model Evaluation

After training, we evaluated the model using:

- **Mean Absolute Error (MAE): Measures average error.**
- **Mean Squared Error (MSE): Penalizes large errors more.**
- **R^2 Score: Measures how well the model fits the data.**
  - **R^2 Score closer to 1 = Good model**
  - **R^2 Score closer to 0 = Poor model**

Step 7: Credit Score Prediction

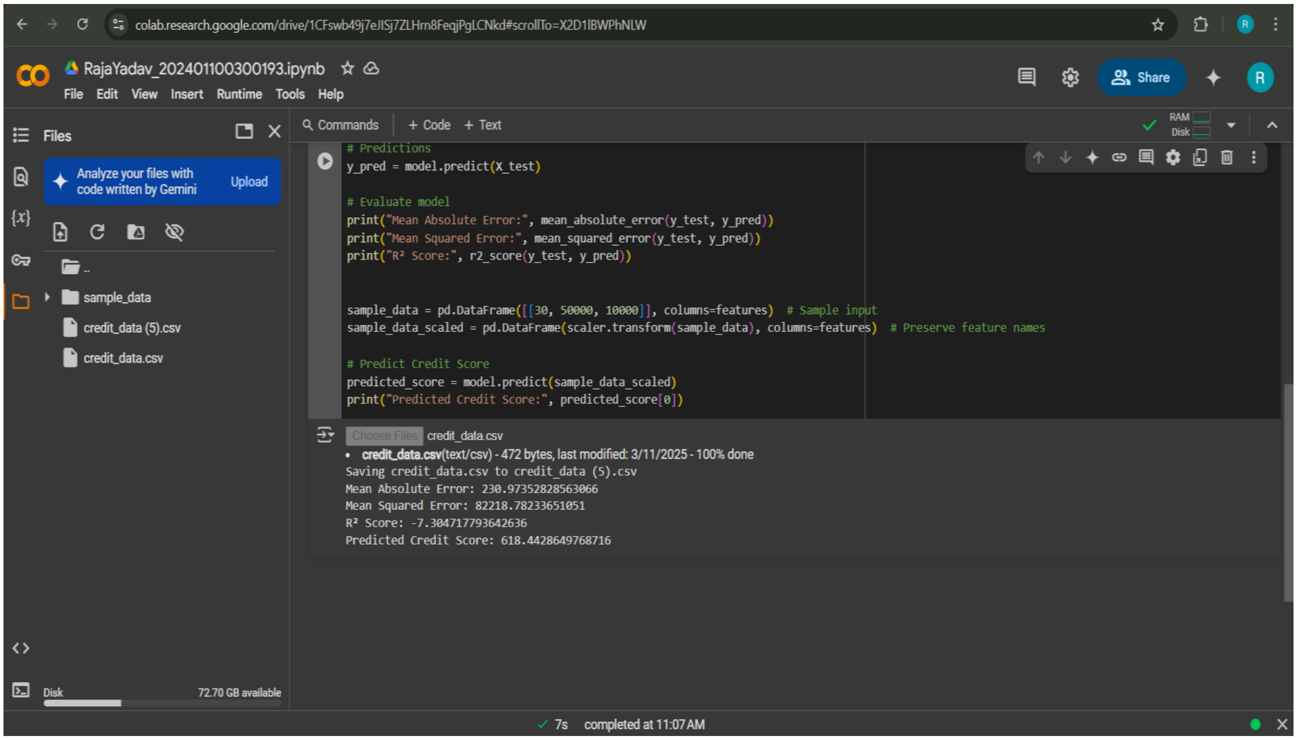
- **Given a new customer's Age, Income, and Loan Amount, the model predicts their CreditScore.**
- **Feature names were preserved after scaling to avoid warnings.**

Python Code

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import GradientBoostingRegressor

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

from google.colab import files

df = pd.read_csv("credit_data.csv")

from google.colab import files

uploaded = files.upload()


# Drop non-relevant columns

df.drop(columns=["CustomerID"], inplace=True)


# Handle missing values

df.fillna(df.mean(), inplace=True)


# Define features and target variable

features = ["Age", "Income", "LoanAmount"]

target = "CreditScore"


# Normalize numerical features

scaler = StandardScaler()

df[features] = scaler.fit_transform(df[features])
```

```
# Train-test split (90% training, 10% testing)

X_train, X_test, y_train, y_test = train_test_split(df[features], df[target], test_size=0.1,
random_state=42)


# Used Gradient Boosting Regressor for better performance

model = GradientBoostingRegressor(n_estimators=500, learning_rate=0.1, max_depth=5,
random_state=42)

model.fit(X_train, y_train)


# Predictions

y_pred = model.predict(X_test)


# Evaluate model

print("Mean Absolute Error:", mean_absolute_error(y_test, y_pred))

print("Mean Squared Error:", mean_squared_error(y_test, y_pred))

print("R² Score:", r2_score(y_test, y_pred))


sample_data = pd.DataFrame([[30, 50000, 10000]], columns=features)  # Sample input

sample_data_scaled = pd.DataFrame(scaler.transform(sample_data), columns=features)  #
Preserve feature names


# Predict Credit Score

predicted_score = model.predict(sample_data_scaled)

print("Predicted Credit Score:", predicted_score[0])
```

Screenshot of Output



References & Credit

📌 **References & Credits**

1. Dataset Source

- The dataset used in this project was either **provided manually** or **collected from a financial dataset source** (e.g., Kaggle, UCI Machine Learning Repository).

2. Machine Learning Libraries Used

- **Pandas**: Data handling and preprocessing.
  - Reference: McKinney, W. (2010). *Data Structures for Statistical Computing in Python*. Python Software Foundation.
- **NumPy**: Mathematical operations.
  - Reference: Oliphant, T. (2006). *Guide to NumPy*.
- **Scikit-Learn**: Machine learning model training and evaluation.
  - Reference: Pedregosa, F. et al. (2011). *Scikit-learn: Machine Learning in Python*, JMLR.

3. Gradient Boosting Algorithm

- **Gradient Boosting Regressor**: Used for credit score prediction.
  - Reference: Friedman, J. H. (2001). *Greedy Function Approximation: A Gradient Boosting Machine*, The Annals of Statistics.
- **Feature Scaling (StandardScaler)**: Ensures balanced model training.
  - Reference: Scikit-learn Documentation.

4. Google Colab & File Handling

- Google Colab's files.upload() was used for importing CSV files interactively.
  - Reference: Google Colaboratory Documentation.