Credit Scoring Model

Mohammad Jaafar Ali Farhat

1. Step 1:

Getting the dataset, it includes the following features:

Age

Income

Credit History (in years)

Loan Amount

Number of Credit Cards

Number of Late Payments

Credit Score

Target Variable: Creditworthy (1) or Not Creditworthy (0).

(the dataset is uploaded on the repo)

1. Step 2: Data Preprocessing:

Loading the dataset and preprocessing it. I'll handle missing values, encode categorical variables (if any), and normalize numerical features.

The code is here:

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

# Load dataset

data = pd.read\_csv('/mnt/data/credit\_scoring\_data.csv')

# Split the dataset into features and target variable

X = data.drop(columns=['Creditworthy'])

y = data['Creditworthy']

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Standardize numerical features

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Convert back to DataFrame for convenience

X\_train\_scaled = pd.DataFrame(X\_train\_scaled, columns=X.columns)

X\_test\_scaled = pd.DataFrame(X\_test\_scaled, columns=X.columns)

X\_train\_scaled.to\_csv('/mnt/data/X\_train\_scaled.csv', index=False)

X\_test\_scaled.to\_csv('/mnt/data/X\_test\_scaled.csv', index=False)

y\_train.to\_csv('/mnt/data/y\_train.csv', index=False)

y\_test.to\_csv('/mnt/data/y\_test.csv', index=False)

The preprocessed data is uploaded on the repo, it is divided into: X\_train, X\_test, Y\_train, Y\_test.

1. Step 3: Model Training and Evaluation:

I’ll use Logistic Regression, Random Forest, and Gradient Boosting as our models.

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, roc\_auc\_score

# Load preprocessed data

X\_train\_scaled = pd.read\_csv('/mnt/data/X\_train\_scaled.csv')

X\_test\_scaled = pd.read\_csv('/mnt/data/X\_test\_scaled.csv')

y\_train = pd.read\_csv('/mnt/data/y\_train.csv')

y\_test = pd.read\_csv('/mnt/data/y\_test.csv')

# Initialize models

log\_reg = LogisticRegression(random\_state=42)

rf = RandomForestClassifier(random\_state=42)

gb = GradientBoostingClassifier(random\_state=42)

# Train models

log\_reg.fit(X\_train\_scaled, y\_train)

rf.fit(X\_train\_scaled, y\_train)

gb.fit(X\_train\_scaled, y\_train)

# Make predictions

log\_reg\_pred = log\_reg.predict(X\_test\_scaled)

rf\_pred = rf.predict(X\_test\_scaled)

gb\_pred = gb.predict(X\_test\_scaled)

# Evaluate models

models = {'Logistic Regression': log\_reg, 'Random Forest': rf, 'Gradient Boosting': gb}

predictions = {'Logistic Regression': log\_reg\_pred, 'Random Forest': rf\_pred, 'Gradient Boosting': gb\_pred}

results = {}

for model\_name, pred in predictions.items():

accuracy = accuracy\_score(y\_test, pred)

precision = precision\_score(y\_test, pred)

recall = recall\_score(y\_test, pred)

f1 = f1\_score(y\_test, pred)

roc\_auc = roc\_auc\_score(y\_test, pred)

results[model\_name] = {

'Accuracy': accuracy,

'Precision': precision,

'Recall': recall,

'F1 Score': f1,

'ROC AUC': roc\_auc

}

results\_df = pd.DataFrame(results).T

results\_df.to\_csv('/mnt/data/model\_evaluation\_results.csv')

results\_df

and here are the results:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | Accuracy | Precision | Recall | F1 Score | ROC AUC |
| Logistic Regression | 0.955 | 0.777778 | 0.500000 | 0.608696 | 0.744624 |
| Random Forest | 0.980 | 1.000000 | 0.714286 | 0.833333 | 0.857143 |
| Gradient Boosting | 0.990 | 1.000000 | 0.857143 | 0.923077 | 0.928571 |

Thank you!