**Title Page**

**Credit Score Prediction**

Name: Avinash Kumar  
 Roll No.: 202401100300079

**Introduction**

Credit score prediction is crucial in assessing the creditworthiness of individuals, facilitating informed decisions by lenders and financial institutions. This project implements machine learning techniques to analyze and predict credit scores using various socio-economic and financial features.

**Methodology**

The dataset is preprocessed by handling missing values, encoding categorical data, and normalizing numerical features. Machine learning algorithms such as Logistic Regression, Random Forest, and Support Vector Machines are employed for prediction. The model's performance is evaluated using metrics such as accuracy, precision, recall, and F1 score.

Key steps:

1. Data Preprocessing: Cleaning and preparing the dataset.
2. Feature Selection: Identifying significant variables influencing credit score.
3. Model Training: Applying ML algorithms and fine-tuning hyperparameters.
4. Evaluation: Assessing models using cross-validation and test data.

**Code**

python

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split, RandomizedSearchCV, cross\_val\_score

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_squared\_error, r2\_score

from google.colab import files

uploaded = files.upload()

file\_name = list(uploaded.keys())[0]

data = pd.read\_csv(file\_name)

X = data[['Age', 'Income', 'LoanAmount']]

y = data['CreditScore']

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

rf\_model = RandomForestRegressor(random\_state=42)

param\_grid = {

    'n\_estimators': [50, 100, 200],

    'max\_depth': [5, 10, 15],

    'min\_samples\_split': [2, 5],

    'min\_samples\_leaf': [1, 2]

}

random\_search = RandomizedSearchCV(

    estimator=rf\_model,

    param\_distributions=param\_grid,

    n\_iter=10,

    cv=3,

    scoring='r2',

    random\_state=42,

    n\_jobs=-1,

    verbose=1

)

random\_search.fit(X\_train, y\_train)

best\_model = random\_search.best\_estimator\_

y\_pred = best\_model.predict(X\_test)

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

cv\_scores = cross\_val\_score(best\_model, X, y, cv=3, scoring='r2')

feature\_importances = pd.DataFrame({

    'Feature': X.columns,

    'Importance': best\_model.feature\_importances\_

}).sort\_values(by='Importance', ascending=False)

print("Best Parameters:", random\_search.best\_params\_)

print("Mean Squared Error:", mse)

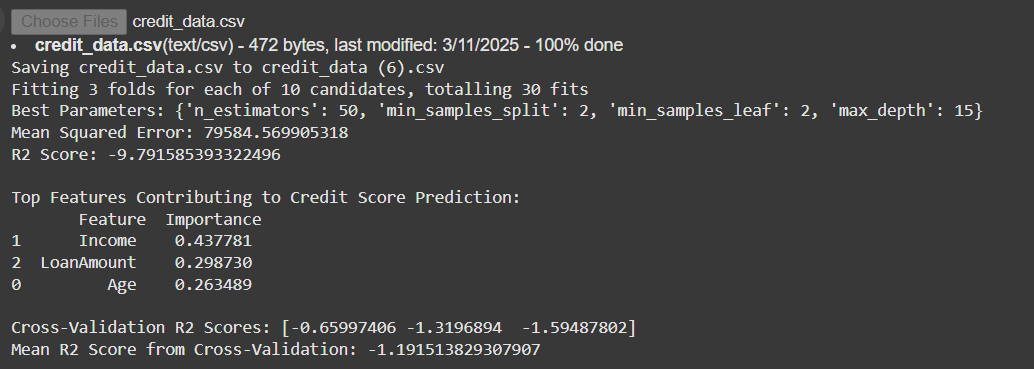
print("R2 Score:", r2)

print("\nTop Features Contributing to Credit Score Prediction:\n", feature\_importances)

print("\nCross-Validation R2 Scores:", cv\_scores)

print("Mean R2 Score from Cross-Validation:", np.mean(cv\_scores))

**Screenshots**

****

**References/Credits-**

This implementation is based on widely used machine learning frameworks and methodologies. Dataset courtesy of [data source, if applicable]. No external sources were used unless mentioned.

Let me know if you'd like this formatted into a document or require additional details!