Credit Scoring Model using Logistic Regression and Random Forest

Credit Scoring Model (Code Only)

```python
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score, roc_curve


data = {

    'income': [40000, 30000, 60000, 25000, 80000, 120000, 22000, 35000, 45000, 70000],

    'debt': [5000, 10000, 7000, 2000, 15000, 10000, 3000, 4500, 5000, 9000],

    'payment_history_score': [0.9, 0.4, 0.8, 0.6, 0.2, 0.95, 0.7, 0.5, 0.65, 0.85],

    'creditworthy': [1, 0, 1, 1, 0, 1, 1, 0, 1, 1]

}


df = pd.DataFrame(data)


X = df[['income', 'debt', 'payment_history_score']]

y = df['creditworthy']


scaler = StandardScaler()
```

Credit Scoring Model using Logistic Regression and Random Forest

```python
X_scaled = scaler.fit_transform(X)


X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)


log_model = LogisticRegression()

log_model.fit(X_train, y_train)

y_pred_log = log_model.predict(X_test)

y_prob_log = log_model.predict_proba(X_test)[:, 1]


rf_model = RandomForestClassifier()

rf_model.fit(X_train, y_train)

y_pred_rf = rf_model.predict(X_test)

y_prob_rf = rf_model.predict_proba(X_test)[:, 1]


print("==== Logistic Regression ====")

print(confusion_matrix(y_test, y_pred_log))

print(classification_report(y_test, y_pred_log))

print("ROC-AUC:", roc_auc_score(y_test, y_prob_log))


print("\n==== Random Forest Classifier ====")

print(confusion_matrix(y_test, y_pred_rf))

print(classification_report(y_test, y_pred_rf))

print("ROC-AUC:", roc_auc_score(y_test, y_prob_rf))


fpr_log, tpr_log, _ = roc_curve(y_test, y_prob_log)
```

Credit Scoring Model using Logistic Regression and Random Forest

```python
fpr_rf, tpr_rf, _ = roc_curve(y_test, y_prob_rf)


plt.figure(figsize=(8, 6))

plt.plot(fpr_log, tpr_log, label='Logistic Regression')

plt.plot(fpr_rf, tpr_rf, label='Random Forest')

plt.plot([0, 1], [0, 1], 'k--')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC Curve Comparison')

plt.legend()

plt.grid(True)

plt.show()
```