```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, classification_r
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Load the dataset
df = pd.read_csv('synthetic_ml_dataset.csv')
```

Data Cleaning & Preprocessing

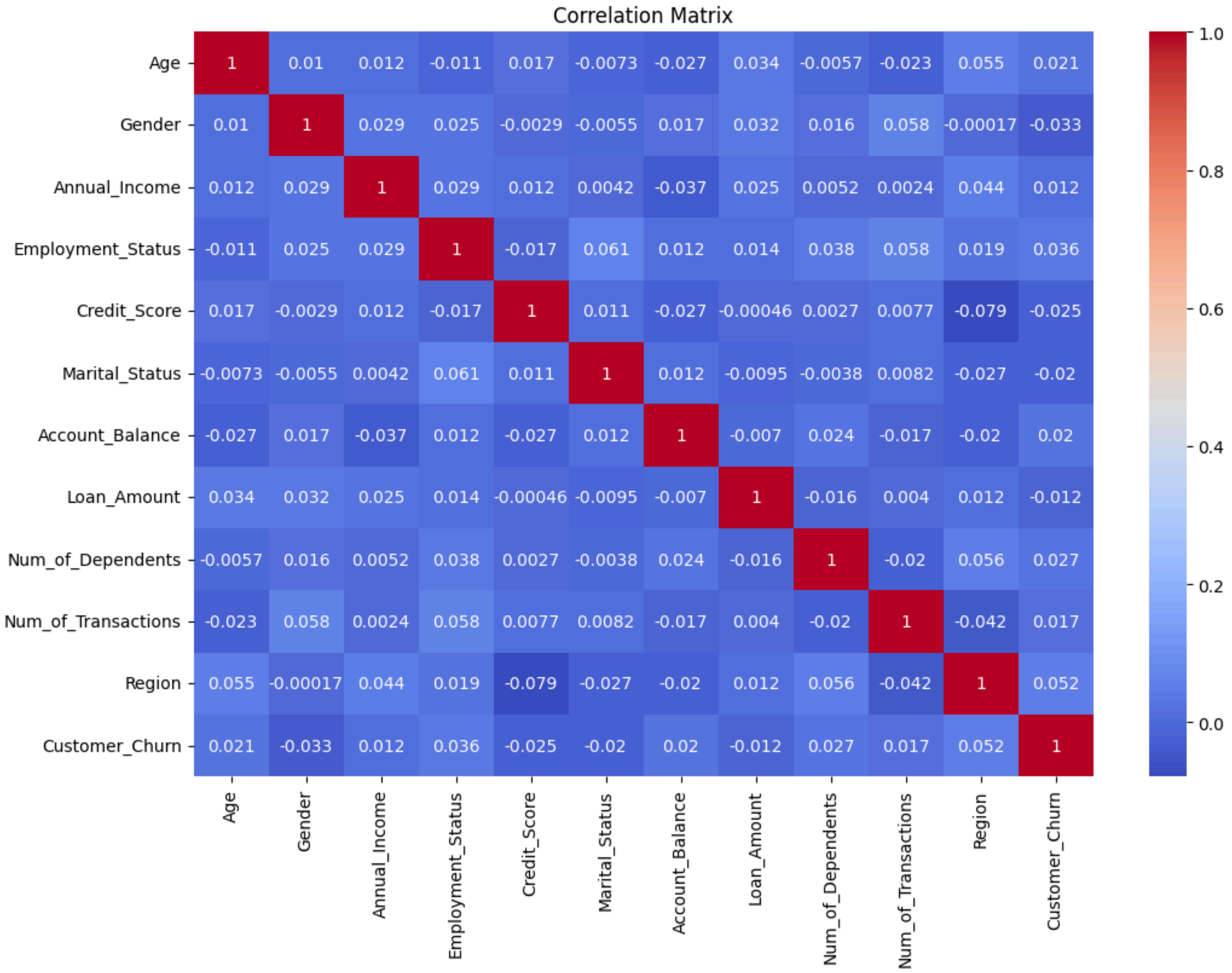
```
# Drop unnecessary columns
df = df.drop(['Customer_ID', 'Joining_Date', 'Last_Activity_Date'], axis=1)
```

```
# Encode categorical variables
label_encoders = {}
for column in ['Gender', 'Employment_Status', 'Marital_Status', 'Region', 'Customer_Churn']:
    le = LabelEncoder()
    df[column] = le.fit_transform(df[column])
    label_encoders[column] = le
```

```
# Feature scaling
scaler = StandardScaler()
df[['Annual_Income', 'Credit_Score', 'Account_Balance', 'Loan_Amount']] = scaler.fit_transform(
    df[['Annual_Income', 'Credit_Score', 'Account_Balance', 'Loan_Amount']]
)
```

Feature Selection

```
# Correlation matrix
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```



Correlation Matrix

	Age	Gender	Annual_Income	Employment_Status	Credit_Score	Marital_Status	Account_Balance	Loan_Amount	Num_of_Dependents	Num_of_Transactions	Region	Customer_Churn
Age	1	0.01	0.012	-0.011	0.017	-0.0073	-0.027	0.034	-0.0057	-0.023	0.055	0.021
Gender	0.01	1	0.029	0.025	-0.0029	-0.0055	0.017	0.032	0.016	0.058	-0.00017	-0.033
Annual_Income	0.012	0.029	1	0.029	0.012	0.0042	-0.037	0.025	0.0052	0.0024	0.044	0.012
Employment_Status	-0.011	0.025	0.029	1	-0.017	0.061	0.012	0.014	0.038	0.058	0.019	0.036
Credit_Score	0.017	-0.0029	0.012	-0.017	1	0.011	-0.027	-0.00046	0.0027	0.0077	-0.079	-0.025
Marital_Status	-0.0073	-0.0055	0.0042	0.061	0.011	1	0.012	-0.0095	-0.0038	0.0082	-0.027	-0.02
Account_Balance	-0.027	0.017	-0.037	0.012	-0.027	0.012	1	-0.007	0.024	-0.017	-0.02	0.02
Loan_Amount	0.034	0.032	0.025	0.014	-0.00046	-0.0095	-0.007	1	-0.016	0.004	0.012	-0.012
Num_of_Dependents	-0.0057	0.016	0.0052	0.038	0.0027	-0.0038	0.024	-0.016	1	-0.02	0.056	0.027
Num_of_Transactions	-0.023	0.058	0.0024	0.058	0.0077	0.0082	-0.017	0.004	-0.02	1	-0.042	0.017
Region	0.055	-0.00017	0.044	0.019	-0.079	-0.027	-0.02	0.012	0.056	-0.042	1	0.052
Customer_Churn	0.021	-0.033	0.012	0.036	-0.025	-0.02	0.02	-0.012	0.027	0.017	0.052	1

Model Building

```
# Split the data
X = df.drop('Customer_Churn', axis=1)
y = df['Customer_Churn']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Train a Random Forest Classifier
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
# Predictions
y_pred = rf.predict(X_test)
```

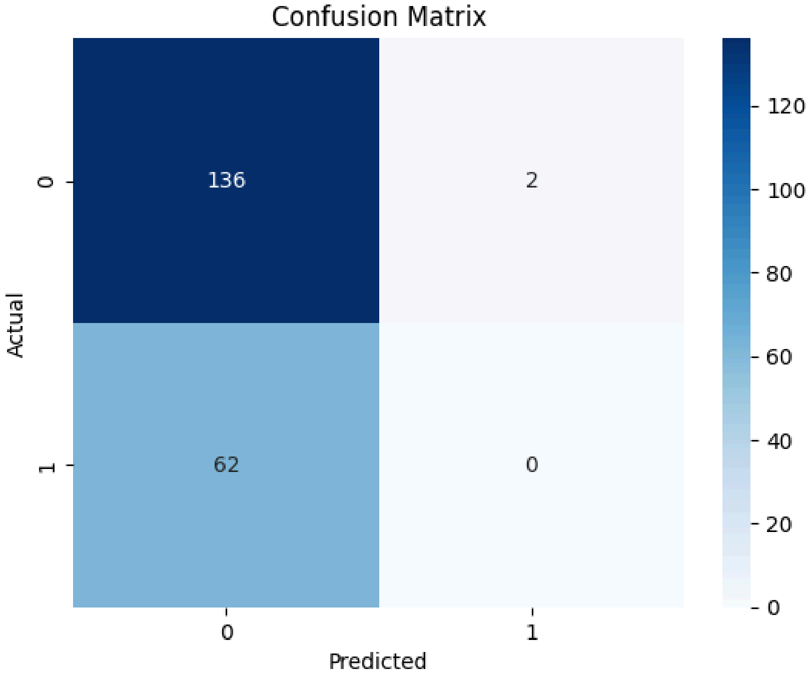
Model Evaluation

```
# Evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred
recall = recall_score(y_test, y_pred)


# Print metrics
print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"F1 Score: {f1:.2f}")
```

```
Accuracy: 0.68
Precision: 0.00
Recall: 0.00
F1 Score: 0.00
```

```
# Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```



```
# Classification Report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.69      0.99      0.81       138
           1       0.00      0.00      0.00        62

    accuracy                           0.68       200
   macro avg       0.34      0.49      0.40       200
```