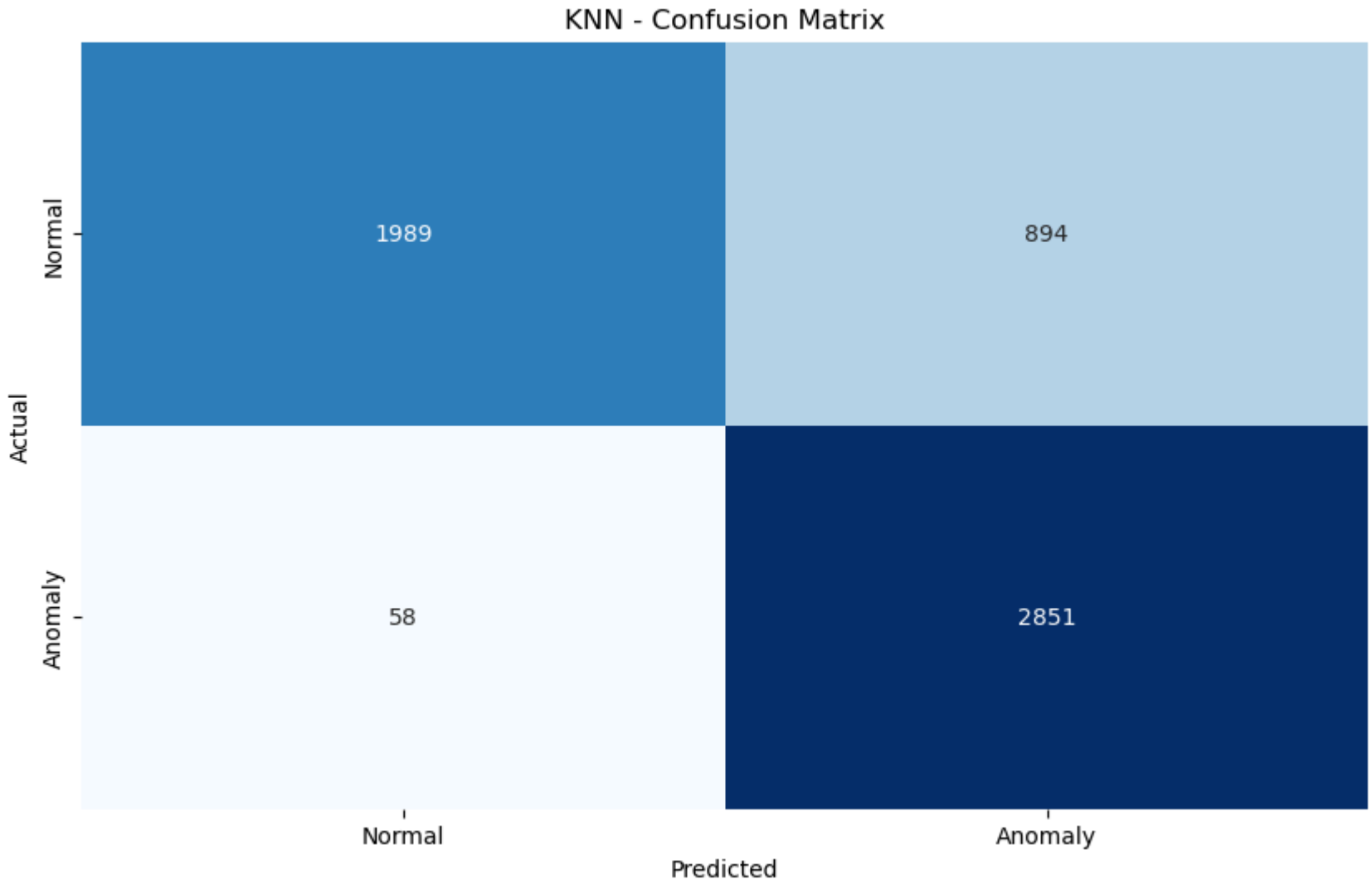
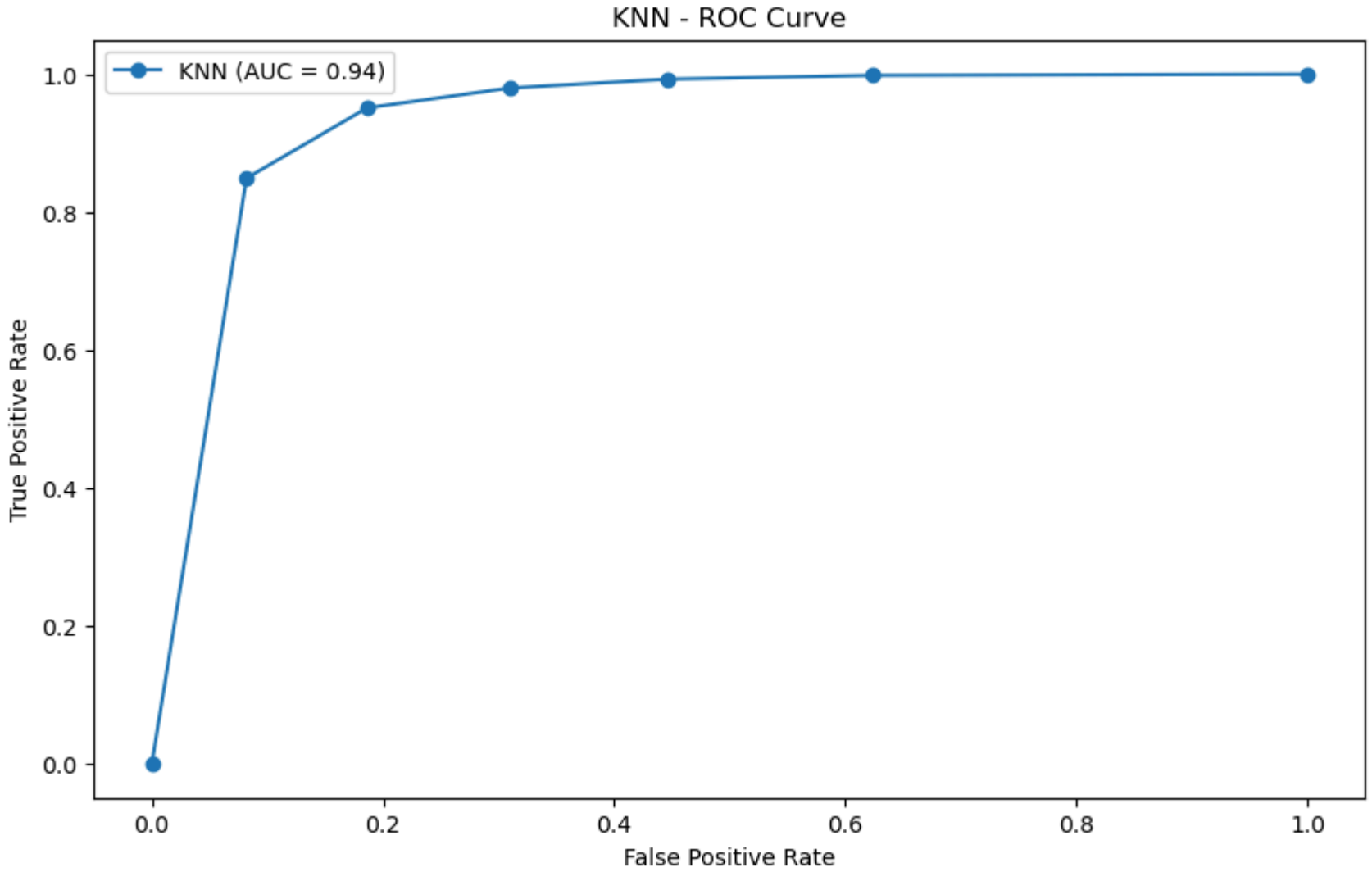
In [1]:

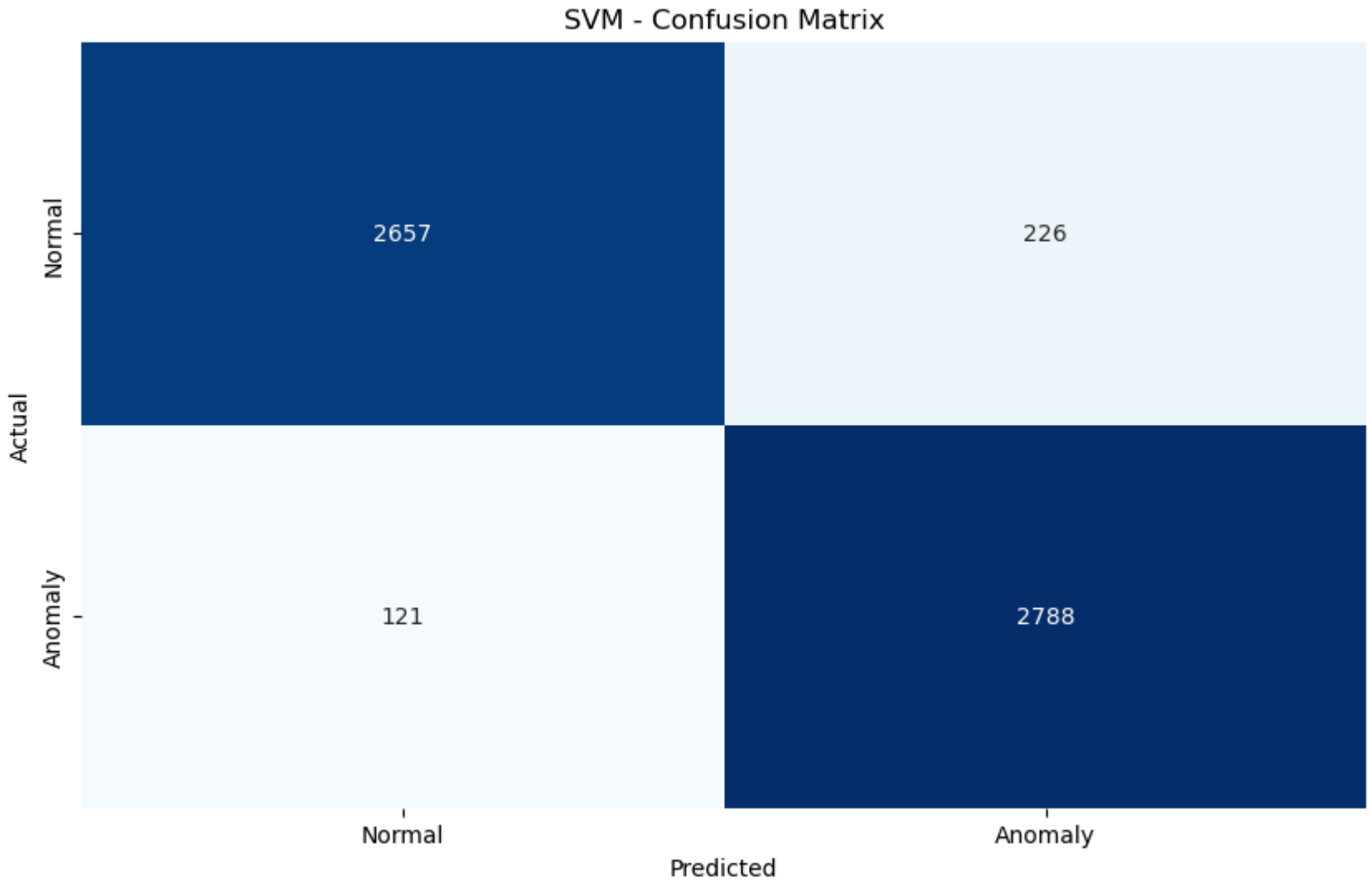
```
# -*- coding: utf-8 -*-
"""
Created on Tue Jul 30 23:35:49 2024

@author: Ramana_
"""

import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, roc_curve, roc_auc_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
import matplotlib.pyplot as plt
import seaborn as sns
# Load the dataset
data = pd.read_csv("C:/Users/Admin/Desktop/Dissertation/avall (1)/dataset_pca.csv")

# Define categorical columns
categorical_columns = ['latitude', 'longitude', 'code', 'Occurrence_Code', 'acft_model', 'light_cond', 'eng_type']

# Encode categorical variables
label_encoders = {}
for col in categorical_columns:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col].astype(str))
    label_encoders[col] = le

# Select features for anomaly detection
features = ['vis_sm', 'crew_age', 'num_eng', 'wx_temp', 'wind_dir_deg', 'wind_vel_kts', 'altimeter'] + categorical_columns
X = data[features]

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Generate synthetic normal data points
normal_data = np.random.normal(loc=0, scale=1, size=(X.shape[0], X.shape[1]))
```

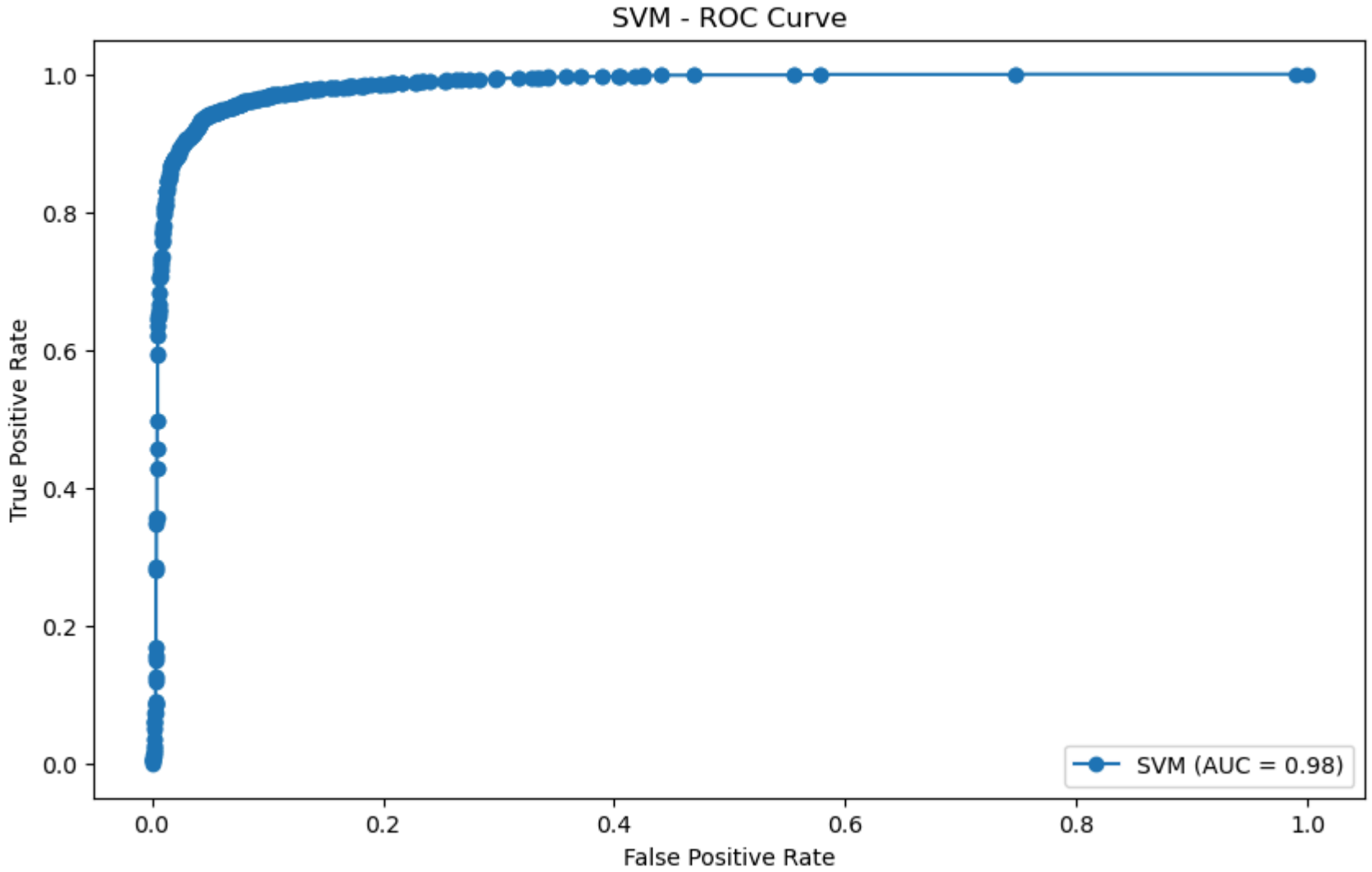
```python
# Combine normal data and anomaly data
X_combined = np.vstack([normal_data, X_scaled])
y_combined = np.hstack([np.zeros(normal_data.shape[0]), np.ones(X_scaled.shape[0])])  # 0 for normal, 1 for anomalies

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_combined, y_combined, test_size=0.2, random_state=42)

# Define models
models = {
    'KNN': KNeighborsClassifier(n_neighbors=5),
    'SVM': SVC(kernel='rbf', gamma='auto'),
    'Logistic Regression': LogisticRegression(random_state=42),
    'Naive Bayes': GaussianNB()
}

# Train, predict, and evaluate each model
results = {}
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    results[name] = {
        'accuracy': accuracy,
        'confusion_matrix': confusion_matrix(y_test, y_pred),
        'classification_report': classification_report(y_test, y_pred)
    }

    # Plot Confusion Matrix
    plt.figure(figsize=(10, 6))
    cm = results[name]['confusion_matrix']
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False,
                xticklabels=['Normal', 'Anomaly'], yticklabels=['Normal', 'Anomaly'])
    plt.title(f'{name} - Confusion Matrix')
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.show()

    # Plot ROC Curve
    fpr, tpr, _ = roc_curve(y_true, y_prob)
    plt.figure(figsize=(10, 6))
    plt.plot(fpr, tpr, marker='o', label=f'{name} (AUC = {results[name]["roc_auc"]:.2f})')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title(f'{name} - ROC Curve')
```

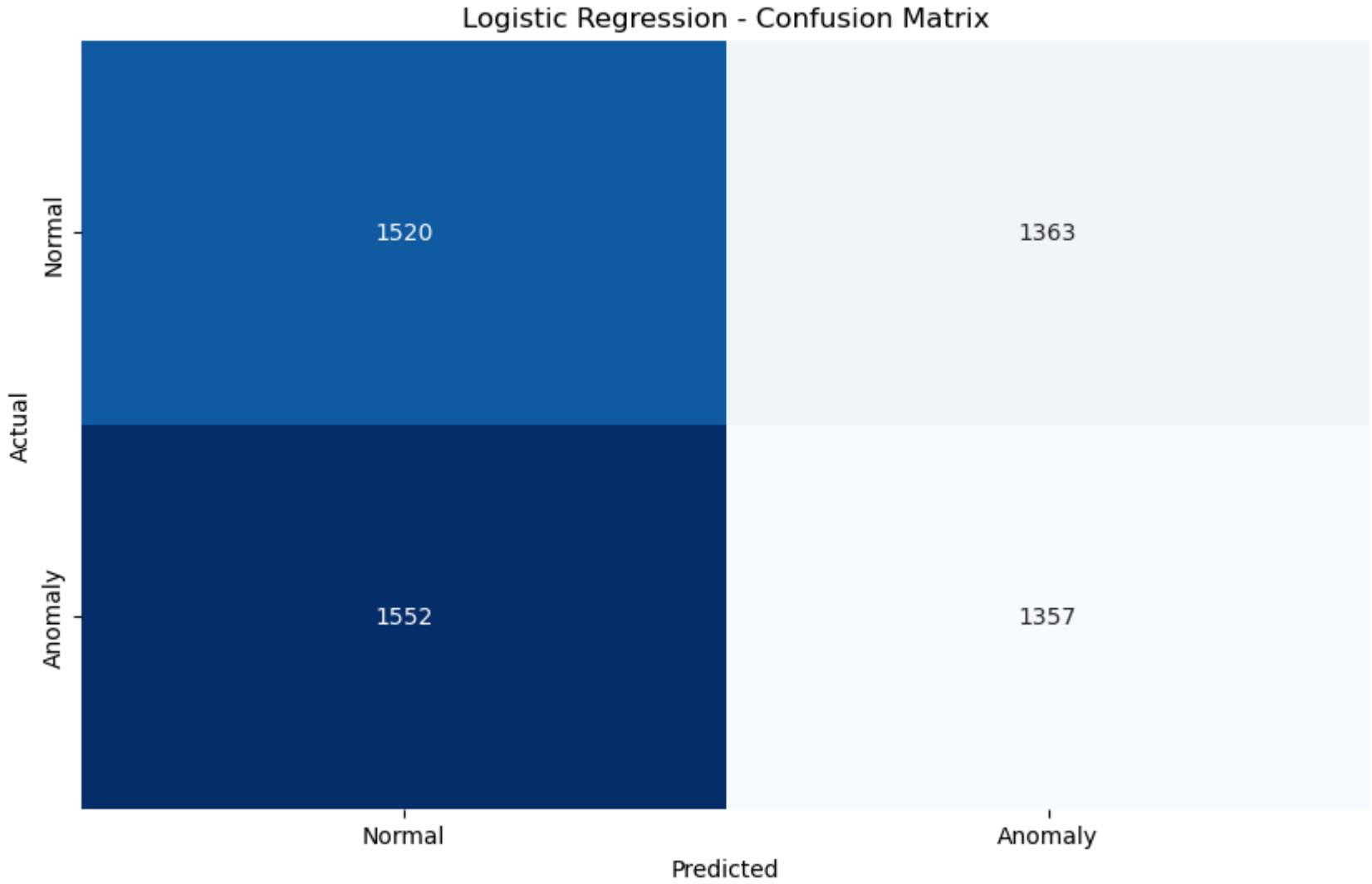
```
    plt.legend()
    plt.show()

# Accuracy Comparison Plot
plt.figure(figsize=(10, 6))
model_names = list(results.keys())
accuracies = [results[name]['accuracy'] for name in model_names]
sns.barplot(x=model_names, y=accuracies, palette='viridis')
plt.title('Model Accuracy Comparison')
plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.ylim(0, 1)
plt.show()

# Print results
for name, result in results.items():
    print(f"{name}:\n")
    print("Confusion Matrix:\n", result['confusion_matrix'])
    print("\nClassification Report:\n", result['classification_report'])
    print(f"Accuracy: {result['accuracy']:.4f}")
    print(f"ROC AUC: {result['roc_auc']:.4f}\n")
```

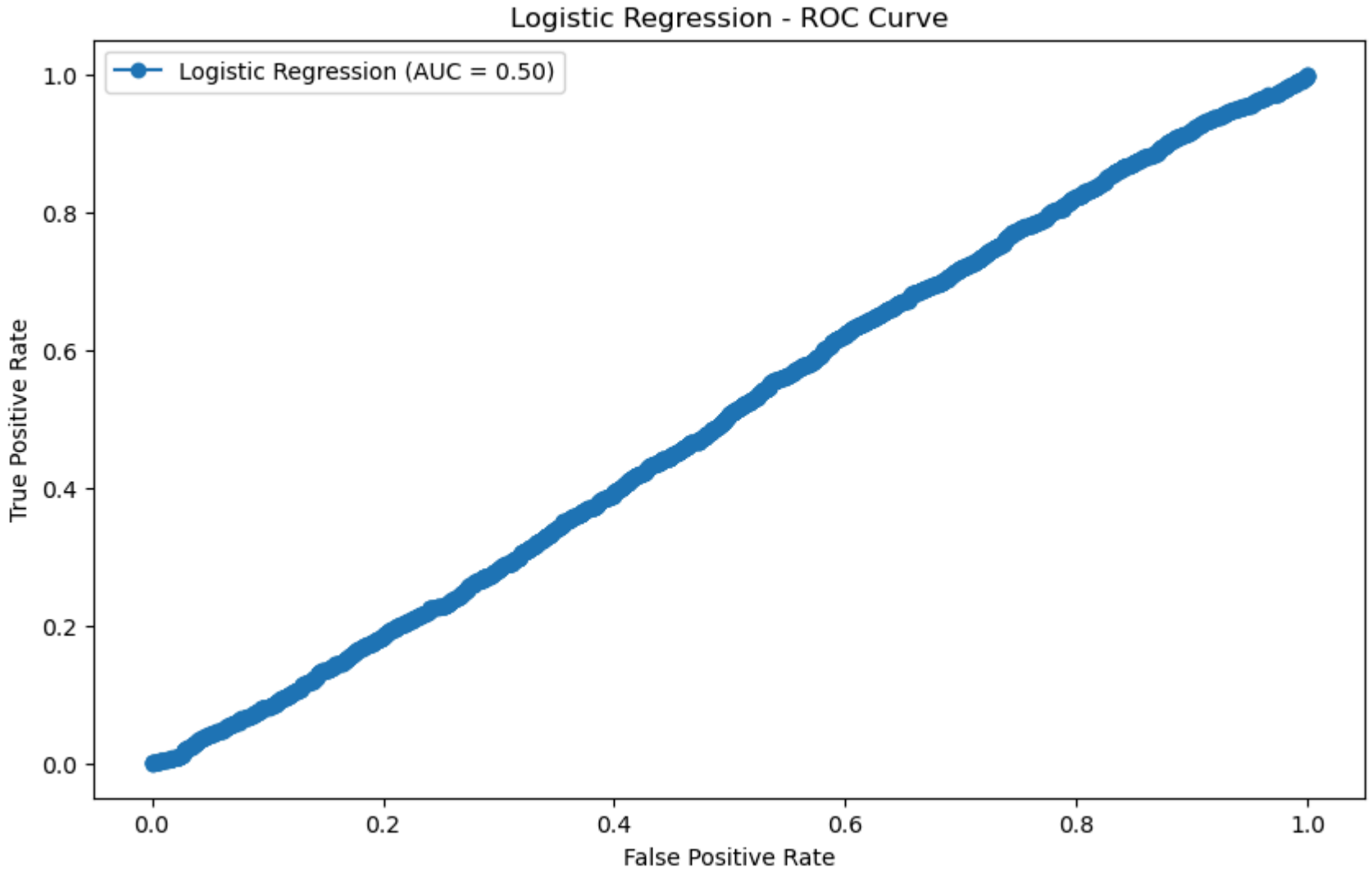

KNN - Confusion Matrix
Normal
Anomaly
Actual
1989
894
58
2851
Normal
Anomaly
Predicted

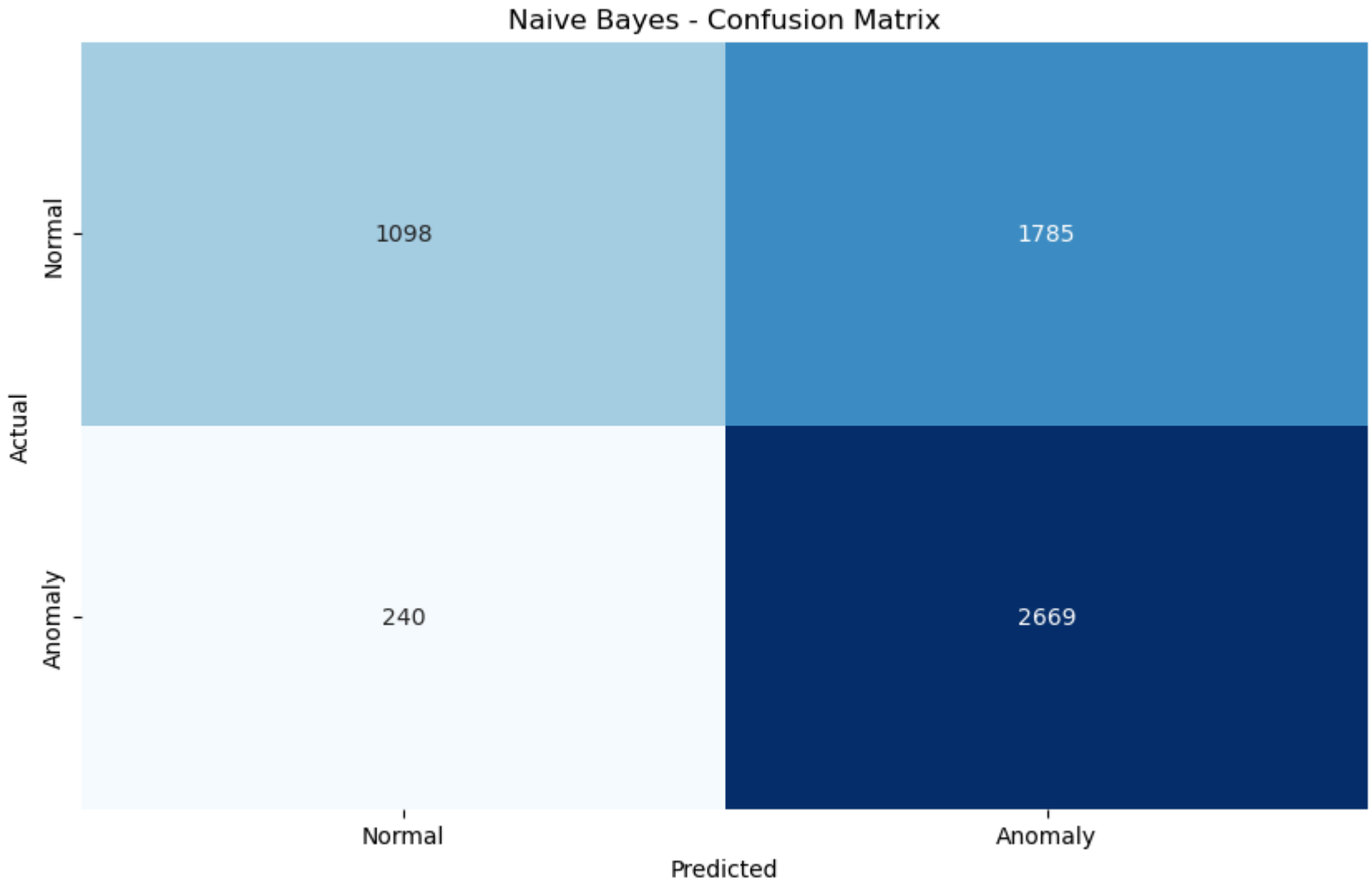

KNN - ROC Curve
KNN (AUC = 0.94)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

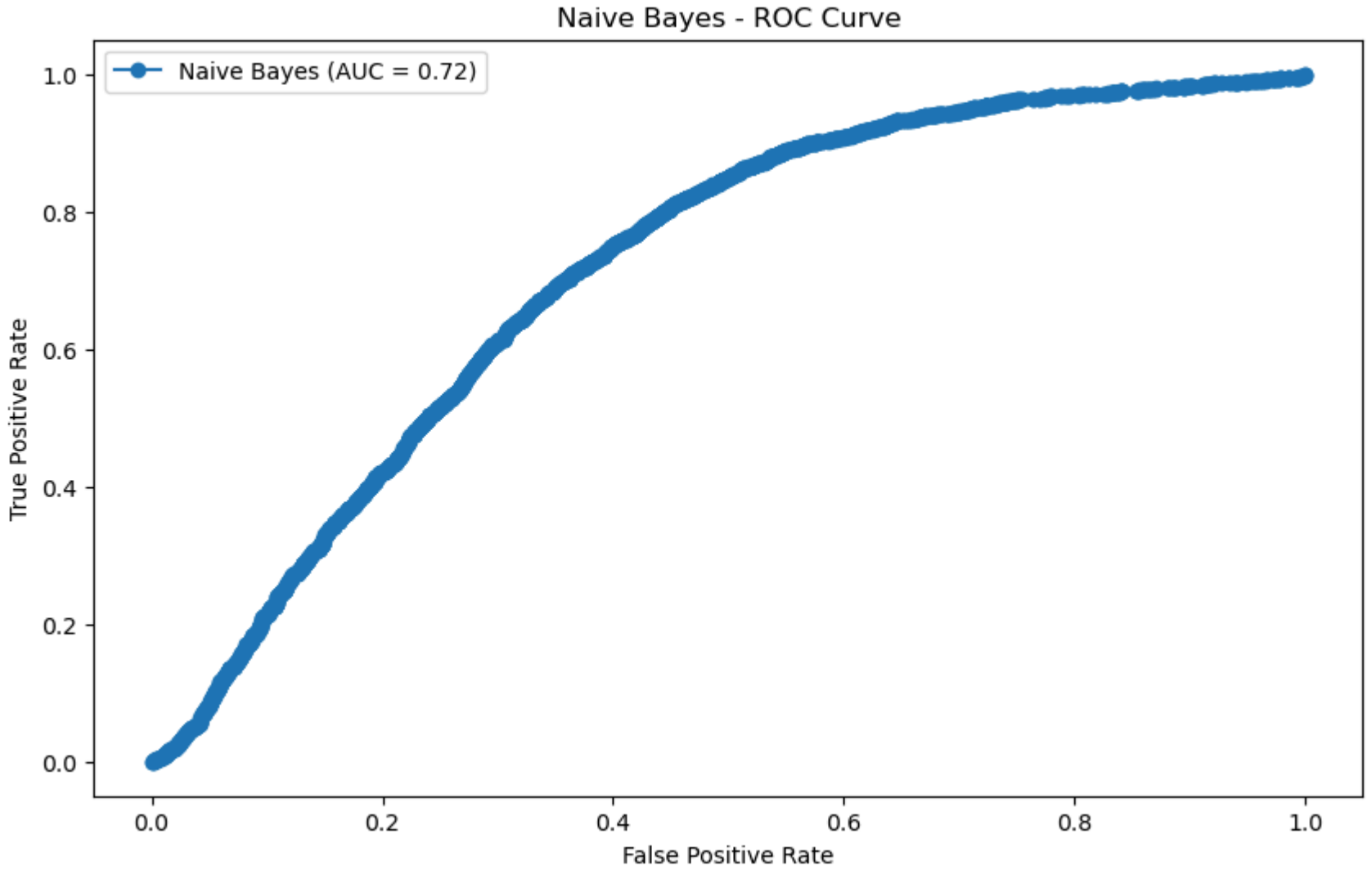

SVM - Confusion Matrix
Normal
Anomaly
Actual
2657
226
121
2788
Normal
Anomaly
Predicted

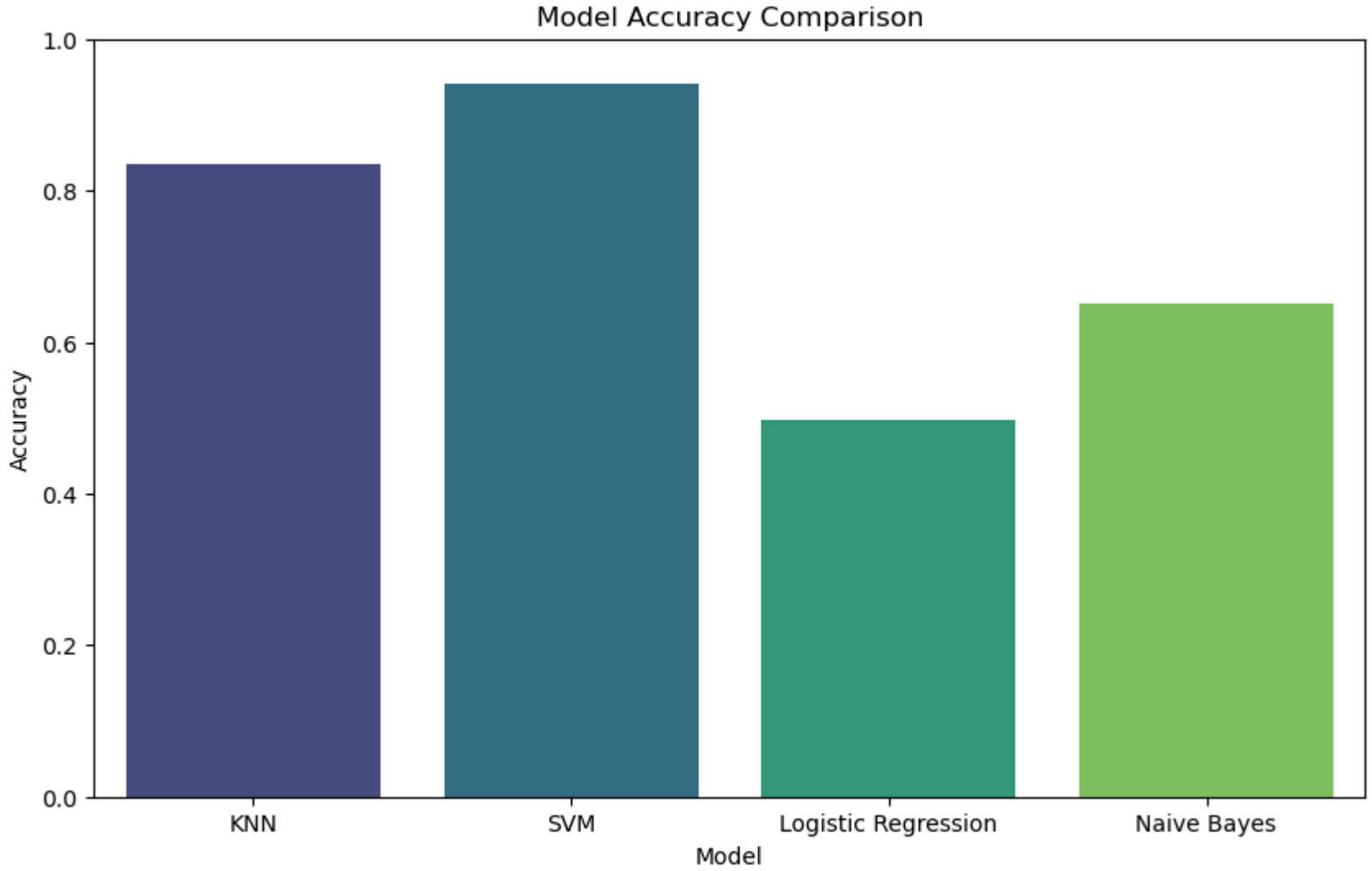

SVM - ROC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
SVM (AUC = 0.98)


Logistic Regression - Confusion Matrix
Normal
Anomaly
Actual
1520
1363
1552
1357
Normal
Anomaly
Predicted


Logistic Regression - ROC Curve
Logistic Regression (AUC = 0.50)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


Naive Bayes - Confusion Matrix
Normal
Anomaly
Actual
1098
1785
240
2669
Normal
Anomaly
Predicted


Naive Bayes - ROC Curve
Naive Bayes (AUC = 0.72)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


Model Accuracy Comparison
1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
KNN
SVM
Logistic Regression
Naive Bayes
Model

```
KNN:

Confusion Matrix:
 [[1989  894]
 [  58 2851]]

Classification Report:
              precision    recall  f1-score   support

         0.0       0.97      0.69      0.81      2883
         1.0       0.76      0.98      0.86      2909

    accuracy                           0.84      5792
   macro avg       0.87      0.83      0.83      5792
weighted avg       0.87      0.84      0.83      5792

Accuracy: 0.8356
ROC AUC: 0.9353

SVM:

Confusion Matrix:
 [[2657  226]
 [ 121 2788]]

Classification Report:
              precision    recall  f1-score   support

         0.0       0.96      0.92      0.94      2883
         1.0       0.93      0.96      0.94      2909

    accuracy                           0.94      5792
   macro avg       0.94      0.94      0.94      5792
weighted avg       0.94      0.94      0.94      5792

Accuracy: 0.9401
ROC AUC: 0.9842

Logistic Regression:

Confusion Matrix:
 [[1520 1363]
 [1552 1357]]
```

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

         0.0       0.49      0.53      0.51      2883
         1.0       0.50      0.47      0.48      2909

    accuracy                           0.50      5792
   macro avg       0.50      0.50      0.50      5792
weighted avg       0.50      0.50      0.50      5792

Accuracy: 0.4967
ROC AUC: 0.5020

Naive Bayes:

Confusion Matrix:
 [[1098 1785]
 [ 240 2669]]

Classification Report:
              precision    recall  f1-score   support

         0.0       0.82      0.38      0.52      2883
         1.0       0.60      0.92      0.72      2909

    accuracy                           0.65      5792
   macro avg       0.71      0.65      0.62      5792
weighted avg       0.71      0.65      0.62      5792

Accuracy: 0.6504
ROC AUC: 0.7157
```

In [ ]: