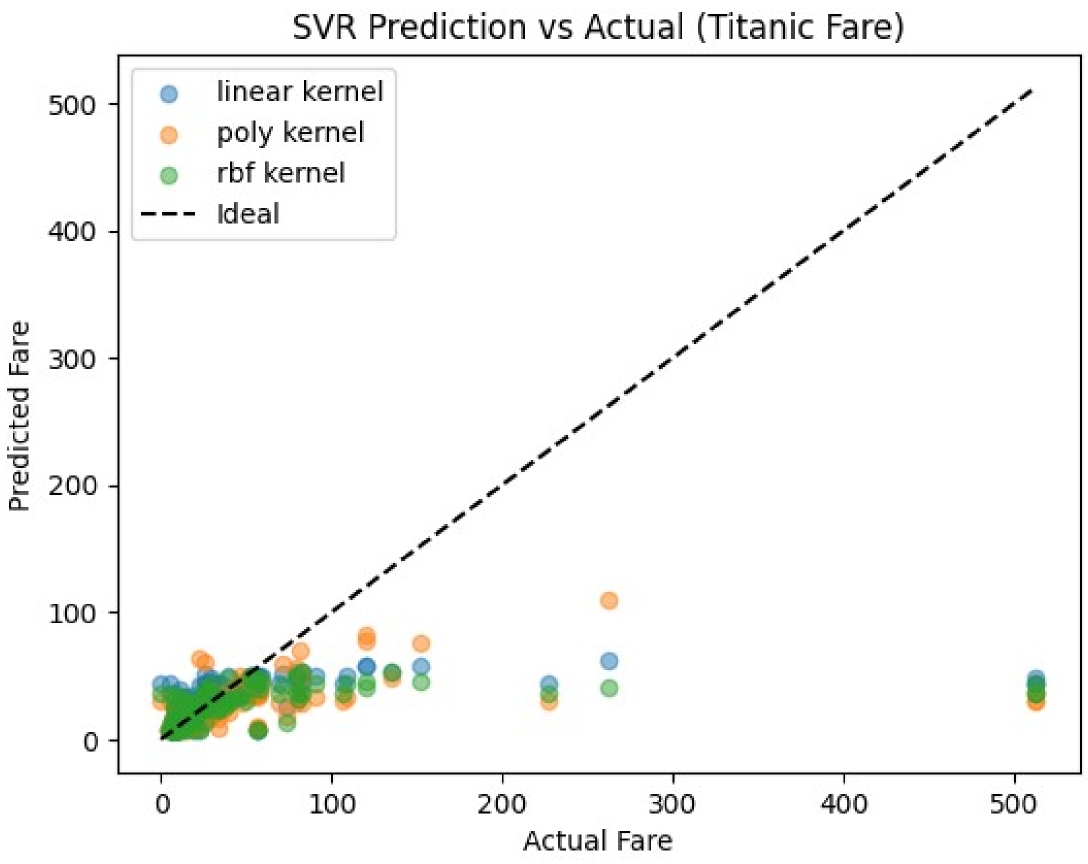
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC, SVR
from sklearn.metrics import accuracy_score, precision_score,
classification_report, mean_squared_error
from sklearn.preprocessing import StandardScaler, LabelEncoder

import seaborn as sns
titanic = sns.load_dataset('titanic')

# === Preprocessing for Classification (SVC) ===
# Predict Survived using relevant features
df_class = titanic[['survived', 'pclass', 'sex', 'age', 'sibsp',
'parch', 'fare']]
df_class = df_class.dropna()

# Encode categorical variable
df_class['sex'] = LabelEncoder().fit_transform(df_class['sex'])

X_class = df_class.drop('survived', axis=1)
y_class = df_class['survived']

scaler = StandardScaler()
X_class_scaled = scaler.fit_transform(X_class)

X_train_c, X_test_c, y_train_c, y_test_c =
train_test_split(X_class_scaled, y_class, test_size=0.2,
random_state=42)

kernels = ['linear', 'poly', 'rbf']
print("=== Support Vector Classification on Titanic ===")
for kernel in kernels:
    print(f"\nKernel: {kernel}")
    clf = SVC(kernel=kernel)
    clf.fit(X_train_c, y_train_c)
    y_pred_c = clf.predict(X_test_c)

    print("Accuracy:", accuracy_score(y_test_c, y_pred_c))
    print("Precision (macro avg):", precision_score(y_test_c,
y_pred_c, average='macro'))
    print("Classification Report:\n", classification_report(y_test_c,
y_pred_c))


# === Preprocessing for Regression (SVR) ===
# Predict Fare using other numerical features
df_reg = titanic[['fare', 'pclass', 'age', 'sibsp', 'parch']]
df_reg = df_reg.dropna()
```

```
X_reg = df_reg.drop('fare', axis=1)
y_reg = df_reg['fare']

scaler = StandardScaler()
X_reg_scaled = scaler.fit_transform(X_reg)

X_train_r, X_test_r, y_train_r, y_test_r =
train_test_split(X_reg_scaled, y_reg, test_size=0.3, random_state=42)

print("\n=== Support Vector Regression on Titanic ===")
for kernel in kernels:
    print(f"\nKernel: {kernel}")
    svr = SVR(kernel=kernel)
    svr.fit(X_train_r, y_train_r)
    y_pred_r = svr.predict(X_test_r)

    rmse = np.sqrt(mean_squared_error(y_test_r, y_pred_r))
    print("Root Mean Squared Error (RMSE):", rmse)

    plt.scatter(y_test_r, y_pred_r, label=f'{kernel} kernel',
alpha=0.5)

plt.plot([0, max(y_test_r)], [0, max(y_test_r)], 'k--', label='Ideal')
plt.title('SVR Prediction vs Actual (Titanic Fare)')
plt.xlabel('Actual Fare')
plt.ylabel('Predicted Fare')
plt.legend()
plt.show()
```

```
=== Support Vector Classification on Titanic ===

Kernel: linear
Accuracy: 0.7342657342657343
Precision (macro avg): 0.7209737827715356
Classification Report:
               precision    recall  f1-score   support

           0       0.78      0.79      0.78        87
           1       0.67      0.64      0.65        56

    accuracy                           0.73       143
   macro avg       0.72      0.72      0.72       143
weighted avg       0.73      0.73      0.73       143


Kernel: poly
Accuracy: 0.7972027972027972
Precision (macro avg): 0.7920104211897525
Classification Report:
               precision    recall  f1-score   support
           0       0.81      0.87      0.84        87
           1       0.78      0.68      0.72        56

    accuracy                           0.80       143
   macro avg       0.79      0.78      0.78       143
weighted avg       0.80      0.80      0.79       143


Kernel: rbf
Accuracy: 0.8041958041958042
Precision (macro avg): 0.7986021505376344
Classification Report:
               precision    recall  f1-score   support

           0       0.82      0.87      0.84        87
           1       0.78      0.70      0.74        56

    accuracy                           0.80       143
   macro avg       0.80      0.78      0.79       143
weighted avg       0.80      0.80      0.80       143


=== Support Vector Regression on Titanic ===

Kernel: linear
Root Mean Squared Error (RMSE): 60.727990015468635

Kernel: poly
Root Mean Squared Error (RMSE): 61.763167277096606

Kernel: rbf
Root Mean Squared Error (RMSE): 62.148575982379704
```


SVR Prediction vs Actual (Titanic Fare)
linear kernel
poly kernel
rbf kernel
Ideal
Predicted Fare
Actual Fare
0
100
200
300
400
500
0
100
200
300
400
500