```
#Jeffery Dirden
#W214801986
#ITAI-1371


import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LassoCV
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, r2_score

# Generate a synthetic dataset
np.random.seed(42)  # For reproducibility

# Assume 1000 samples with 10 features
num_samples = 1000
num_features = 10

# Create random features and a target variable
X = np.random.rand(num_samples, num_features) * 100  # Features scaled between 0 and 100
y = X[:, 0] * 0.5 + X[:, 1] * 0.3 + X[:, 2] * 0.2 + np.random.randn(num_samples) * 5  # Linear relationship with noise

# Convert to a DataFrame for compatibility with pandas
columns = [f'feature_{i}' for i in range(1, num_features + 1)]
data = pd.DataFrame(X, columns=columns)
data['margin'] = y  # Add target variable

print("Synthetic Dataset Preview:")
print(data.head())

# Separate features and target
X = data.drop(['margin'], axis=1)  # Drop target column
y = data['margin']  # Isolate the target variable

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Scale numerical features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Lasso Regression
print("\n--- Lasso Regression ---")
lasso = LassoCV(cv=5, random_state=0)
lasso.fit(X_train_scaled, y_train)

# Lasso Evaluation
y_pred_lasso = lasso.predict(X_test_scaled)
lasso_rmse = np.sqrt(mean_squared_error(y_test, y_pred_lasso))
lasso_r2 = r2_score(y_test, y_pred_lasso)
print(f"Lasso Regression RMSE: {lasso_rmse}")
print(f"Lasso Regression R^2: {lasso_r2}")

# Support Vector Machine Regression
print("\n--- SVM Regression ---")
svm = SVR(kernel='rbf')

# GridSearch for best parameters
param_grid = {'C': [0.1, 1, 10], 'gamma': [1, 0.1, 0.01]}
grid_search = GridSearchCV(svm, param_grid, cv=5, scoring='neg_mean_squared_error')
grid_search.fit(X_train_scaled, y_train)

# Best model from GridSearch
best_svm = grid_search.best_estimator_
print("Best SVM Parameters:", grid_search.best_params_)

# SVM Evaluation
y_pred_svm = best_svm.predict(X_test_scaled)
svm_rmse = np.sqrt(mean_squared_error(y_test, y_pred_svm))
svm_r2 = r2_score(y_test, y_pred_svm)
print(f"SVM Regression RMSE: {svm_rmse}")
print(f"SVM Regression R^2: {svm_r2}")
```

```
# Final Summary
print("\n--- Model Performance Summary ---")
print(f"Lasso RMSE: {lasso_rmse}, R^2: {lasso_r2}")
print(f"SVM RMSE: {svm_rmse}, R^2: {svm_r2}")
```

```
Synthetic Dataset Preview:
   feature_1  feature_2  feature_3  feature_4  feature_5  feature_6  \
0  37.454012  95.071431  73.199394  59.865848  15.601864  15.599452
1   2.058449  96.990985  83.244264  21.233911  18.182497  18.340451
2  61.185289  13.949386  29.214465  36.636184  45.606998  78.517596
3  60.754485  17.052412   6.505159  94.888554  96.563203  80.839735
4  12.203823  49.517691   3.438852  90.932040  25.877998  66.252228

   feature_7  feature_8  feature_9  feature_10     margin
0   5.808361  86.617615  60.111501   70.807258  54.449005
1  30.424224  52.475643  43.194502   29.122914  41.149443
2  19.967378  51.423444  59.241457    4.645041  42.564448
3  30.461377   9.767211  68.423303   44.015249  30.924632
4  31.171108  52.006802  54.671028   18.485446  27.208161

--- Lasso Regression ---
Lasso Regression RMSE: 5.304533013175062
Lasso Regression R^2: 0.8960878604006127

--- SVM Regression ---
Best SVM Parameters: {'C': 10, 'gamma': 0.01}
SVM Regression RMSE: 5.2842766850530625
SVM Regression R^2: 0.8968799600855704

--- Model Performance Summary ---
Lasso RMSE: 5.304533013175062, R^2: 0.8960878604006127
SVM RMSE: 5.2842766850530625, R^2: 0.8968799600855704
```