ML_LAB_06

2403A52006

MD.Mustafa

Import Required Libraries

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

Load the Dataset

```
# Upload insurance.csv to Colab before running
df = pd.read_csv("/content/insurance.csv")

df.head()
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

Next steps: Generate code with df | New interactive sheet

Encode Categorical Variables

```
df_encoded = pd.get_dummies(df, drop_first=True)

df_encoded.head()
```

	age	bmi	children	charges	sex_male	smoker_yes	region_northwest	region_
0	19	27.900	0	16884.92400	False	True	False	
1	18	33.770	1	1725.55230	True	False	False	
2	28	33.000	3	4449.46200	True	False	False	
3	33	22.705	0	21984.47061	True	False	True	
4	32	28.880	0	3866.85520	True	False	True	

Next steps: Generate code with `df_encoded` New interactive sheet

Select Features and Target

```
X = df_encoded.drop('charges', axis=1)  # Input features
y = df_encoded['charges']               # Target (insurance cost)
```

Train–Test Split (80/20)

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

Apply Feature Scaling

```
scaler = StandardScaler()

X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Train SVR Model (RBF Kernel)

```
svr_rbf = SVR(kernel='rbf')
svr_rbf.fit(X_train_scaled, y_train)

y_pred_rbf = svr_rbf.predict(X_test_scaled)
```

Evaluate SVR (RBF Kernel)

```
mae_rbf = mean_absolute_error(y_test, y_pred_rbf)
rmse_rbf = np.sqrt(mean_squared_error(y_test, y_pred_rbf))
r2_rbf = r2_score(y_test, y_pred_rbf)
```

```
print("SVR (RBF Kernel)")
print("MAE:", mae_rbf)
print("RMSE:", rmse_rbf)
print("R2 Score:", r2_rbf)
```

```
SVR (RBF Kernel)
MAE: 8612.408423351833
RMSE: 12889.096314656128
R2 Score: -0.07008155372454805
```

Train SVR Model (Polynomial Kernel)

```
svr_poly = SVR(kernel='poly', degree=2)
svr_poly.fit(X_train_scaled, y_train)

y_pred_poly = svr_poly.predict(X_test_scaled)
```

Evaluate SVR (Polynomial Kernel)

```
mae_poly = mean_absolute_error(y_test, y_pred_poly)
rmse_poly = np.sqrt(mean_squared_error(y_test, y_pred_poly))
r2_poly = r2_score(y_test, y_pred_poly)

print("SVR (Polynomial Kernel)")
print("MAE:", mae_poly)
print("RMSE:", rmse_poly)
print("R2 Score:", r2_poly)
```

```
SVR (Polynomial Kernel)
MAE: 8628.582335026595
RMSE: 12897.50266305726
R2 Score: -0.07147783653474571
```

Compare Results

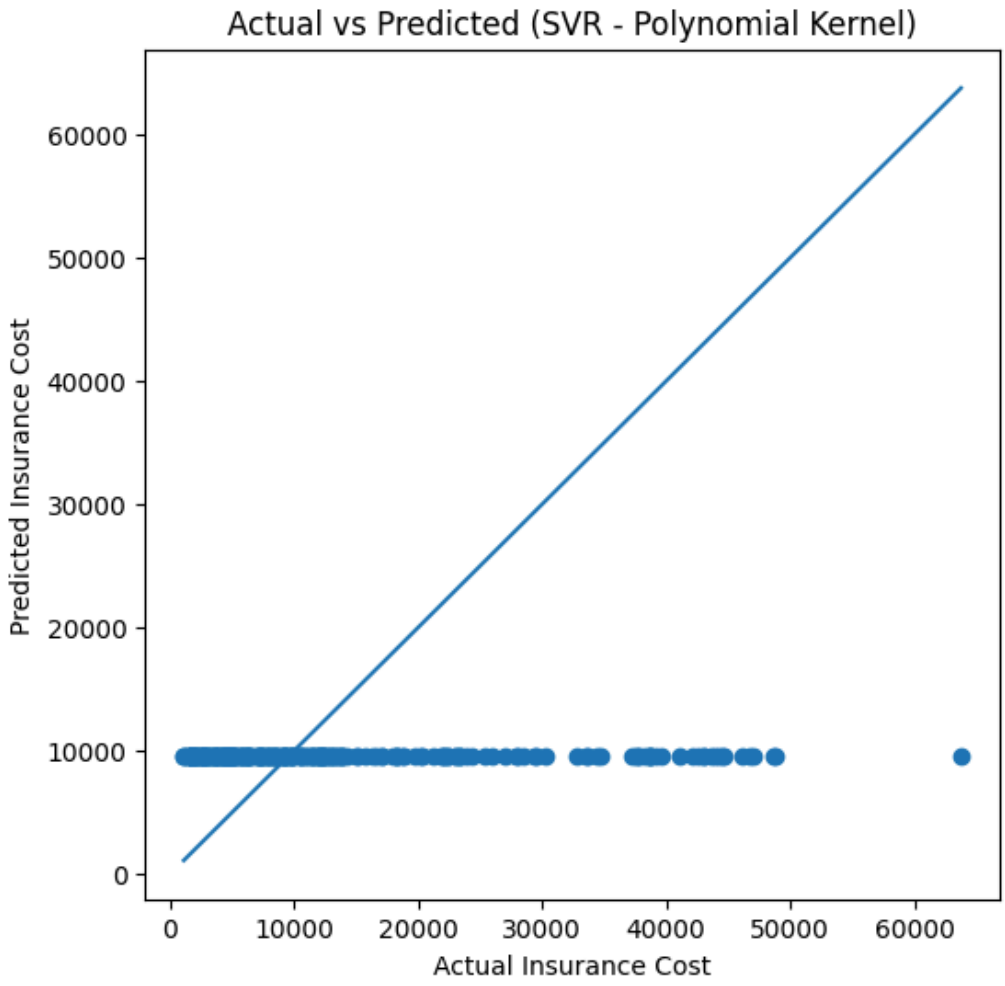
```
results = pd.DataFrame({
    "Kernel": ["RBF", "Polynomial"],
    "MAE": [mae_rbf, mae_poly],
    "RMSE": [rmse_rbf, rmse_poly],
    "R2 Score": [r2_rbf, r2_poly]
})

results
```

	Kernel	MAE	RMSE	R2 Score
0	RBF	8612.408423	12889.096315	-0.070082
1	Polynomial	8628.582335	12897.502663	-0.071478

Next steps: Generate code with `results` New interactive sheet

Actual vs Predicted (SVR – Polynomial Kernel)

```
plt.figure(figsize=(6,6))
plt.scatter(y_test, y_pred_poly)
plt.xlabel("Actual Insurance Cost")
plt.ylabel("Predicted Insurance Cost")
plt.title("Actual vs Predicted (SVR - Polynomial Kernel)")
plt.plot([y_test.min(), y_test.max()],
         [y_test.min(), y_test.max()])
plt.show()
```



Close

```
### Error Comparison (MAE & RMSE)
```

Error Comparison (MAE & RMSE)

```
kernels = ['RBF', 'Polynomial']
mae_values = [mae_rbf, mae_poly]
rmse_values = [rmse_rbf, rmse_poly]

x = np.arange(len(kernels))
width = 0.35
```

```
plt.figure(figsize=(7,5))
plt.bar(x - width/2, mae_values, width, label='MAE')
plt.bar(x + width/2, rmse_values, width, label='RMSE')

plt.xticks(x, kernels)
plt.xlabel("Kernel Type")
plt.ylabel("Error Value")
plt.title("Error Comparison of SVR Kernels")
plt.legend()
plt.show()
```