```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.metrics import mean_squared_error, r2_score

import warnings
warnings.filterwarnings('ignore')
```

```
df = pd.read_csv("CO2 Emissions_Canada.csv")
df.head()
```

	Make	Model	Vehicle Class	Engine Size(L)	Cylinders	Transmission	Fuel Type	Fuel Consumption City (L/100 km)	Fuel Consumption Hwy (L/100 km)	Fuel Consumption Comb (L/100 km)	Fuel Consumption Comb (mpg)	Em
0	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	6.7	8.5	33	
1	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	7.7	9.6	29	
2	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	5.8	5.9	48	
3	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	9.1	11.1	25	
4	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	8.7	10.6	27	

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

```
df.columns
```

```
Index(['Make', 'Model', 'Vehicle Class', 'Engine Size(L)', 'Cylinders',
       'Transmission', 'Fuel Type', 'Fuel Consumption City (L/100 km)',
       'Fuel Consumption Hwy (L/100 km)', 'Fuel Consumption Comb (L/100 km)',
       'Fuel Consumption Comb (mpg)', 'CO2 Emissions(g/km)'],
      dtype='object')
```

```
X = df[['Engine Size(L)', 'Cylinders', 'Fuel Consumption Comb (L/100 km)']]
y = df['CO2 Emissions(g/km)']
```

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

```
scaler = StandardScaler()

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

```
lr = LinearRegression()
lr.fit(X_train_scaled, y_train)

y_pred_lr = lr.predict(X_test_scaled)

mse_lr = mean_squared_error(y_test, y_pred_lr)
rmse_lr = np.sqrt(mse_lr)
r2_lr = r2_score(y_test, y_pred_lr)

print("Linear Regression Performance")
print("MSE:", mse_lr)
print("RMSE:", rmse_lr)
print("R2 Score:", r2_lr)
```

```
Linear Regression Performance
MSE: 421.92233190519977
RMSE: 20.540748085335153
R2 Score: 0.8773348735033225
```

```
lasso = Lasso(alpha=0.1)
lasso.fit(X_train_scaled, y_train)

y_pred_lasso = lasso.predict(X_test_scaled)
```

```
mse_lasso = mean_squared_error(y_test, y_pred_lasso)
rmse_lasso = np.sqrt(mse_lasso)
r2_lasso = r2_score(y_test, y_pred_lasso)

print("\nLasso Regression Performance")
print("MSE:", mse_lasso)
print("RMSE:", rmse_lasso)
print("R2 Score:", r2_lasso)
```

```
Lasso Regression Performance
MSE: 421.9906734651132
RMSE: 20.54241157861251
R2 Score: 0.8773150046187961
```

```
ridge = Ridge()

param_grid = {'alpha': [0.001, 0.01, 0.1, 1, 10, 100]}

grid = GridSearchCV(ridge, param_grid, cv=5)
grid.fit(X_train_scaled, y_train)

print("Best Alpha:", grid.best_params_)
```

```
Best Alpha: {'alpha': 10}
```

```
best_ridge = grid.best_estimator_

y_pred_ridge = best_ridge.predict(X_test_scaled)

mse_ridge = mean_squared_error(y_test, y_pred_ridge)
rmse_ridge = np.sqrt(mse_ridge)
r2_ridge = r2_score(y_test, y_pred_ridge)

print("\nBest Ridge Performance")
print("MSE:", mse_ridge)
print("RMSE:", rmse_ridge)
print("R2 Score:", r2_ridge)
```

```
Best Ridge Performance
MSE: 421.94896543141095
RMSE: 20.541396384652405
R2 Score: 0.8773271303605337
```

```
results = pd.DataFrame({
    "Model": ["Linear Regression", "Lasso", "Ridge"],
    "MSE": [mse_lr, mse_lasso, mse_ridge],
    "RMSE": [rmse_lr, rmse_lasso, rmse_ridge],
    "R2 Score": [r2_lr, r2_lasso, r2_ridge]
})

results
```

	Model	MSE	RMSE	R2 Score
0	Linear Regression	421.922332	20.540748	0.877335
1	Lasso	421.990673	20.542412	0.877315
2	Ridge	421.948965	20.541396	0.877327

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

```
coef_df = pd.DataFrame({
    "Feature": X.columns,
    "Linear Regression": lr.coef_,
    "Lasso": lasso.coef_,
    "Ridge": best_ridge.coef_
})

coef_df
```

	Feature	Linear Regression	Lasso	Ridge
0	Engine Size(L)	7.605027	7.605462	7.746834
1	Cylinders	11.702725	11.648532	11.676620
2	Fuel Consumption Comb (L/100 km)	38.330321	38.272358	38.170300

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