```
import numpy as np
import pandas as pd
```

```
df=pd.read_csv('fuel_consumption_dataset.csv')
```

```
df.head()
```

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	FUELCONSUM
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	

Next steps: Generate code with df New interactive sheet

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1067 entries, 0 to 1066
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   MODELYEAR                 1067 non-null   int64  
 1   MAKE                      1067 non-null   object 
 2   MODEL                     1067 non-null   object 
 3   VEHICLECLASS              1067 non-null   object 
 4   ENGINESIZE                1067 non-null   float64
 5   CYLINDERS                 1067 non-null   int64  
 6   TRANSMISSION              1067 non-null   object 
 7   FUELTYPE                  1067 non-null   object 
 8   FUELCONSUMPTION_CITY      1067 non-null   float64
 9   FUELCONSUMPTION_HWY       1067 non-null   float64
 10  FUELCONSUMPTION_COMB      1067 non-null   float64
 11  FUELCONSUMPTION_COMB_MPG  1067 non-null   int64  
 12  CO2EMISSIONS              1067 non-null   int64  
dtypes: float64(4), int64(4), object(5)
memory usage: 108.5+ KB
```

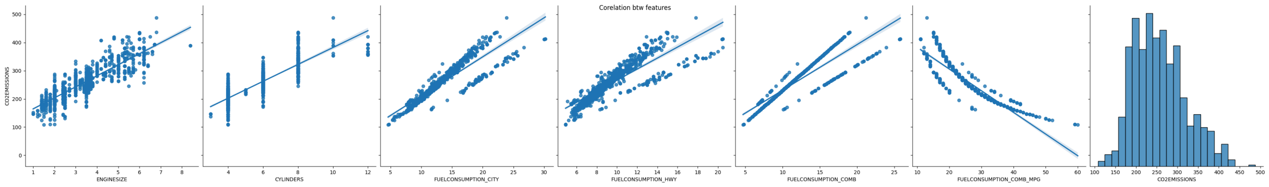
```
df.isnull().sum()
```

	0
MODELYEAR	0
MAKE	0
MODEL	0
VEHICLECLASS	0
ENGINESIZE	0
CYLINDERS	0
TRANSMISSION	0
FUELTYPE	0
FUELCONSUMPTION_CITY	0
FUELCONSUMPTION_HWY	0
FUELCONSUMPTION_COMB	0
FUELCONSUMPTION_COMB_MPG	0
CO2EMISSIONS	0

dtype: int64

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
sns.pairplot(data=df,
             x_vars=['ENGINESIZE', 'CYLINDERS',
                'FUELCONSUMPTION_CITY', 'FUELCONSUMPTION_HWY',
                'FUELCONSUMPTION_COMB', 'FUELCONSUMPTION_COMB_MPG',
                'CO2EMISSIONS'],
             y_vars=['CO2EMISSIONS'],
             height=5,
             aspect=1,
             kind='reg')
plt.suptitle("Corelation btw features")
plt.show()
```



```
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
import numpy as np

numerical_data=df.select_dtypes(include=np.number)
categorical_data=df.select_dtypes(exclude=np.number)

preprocessor=ColumnTransformer(
   transformers=[
       ('num',StandardScaler(),numerical_data.columns),
       ('cat',OneHotEncoder(handle_unknown='ignore'),categorical_data.columns)
   ]
)

x=preprocessor.fit_transform(df)
y=df['CO2EMISSIONS']
```

```
x.shape
```

```
(1067, 752)
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

```
from sklearn.tree import DecisionTreeRegressor

model=DecisionTreeRegressor()
model.fit(x_train,y_train)
```

```
▾ DecisionTreeRegressor  i ?
DecisionTreeRegressor()
```

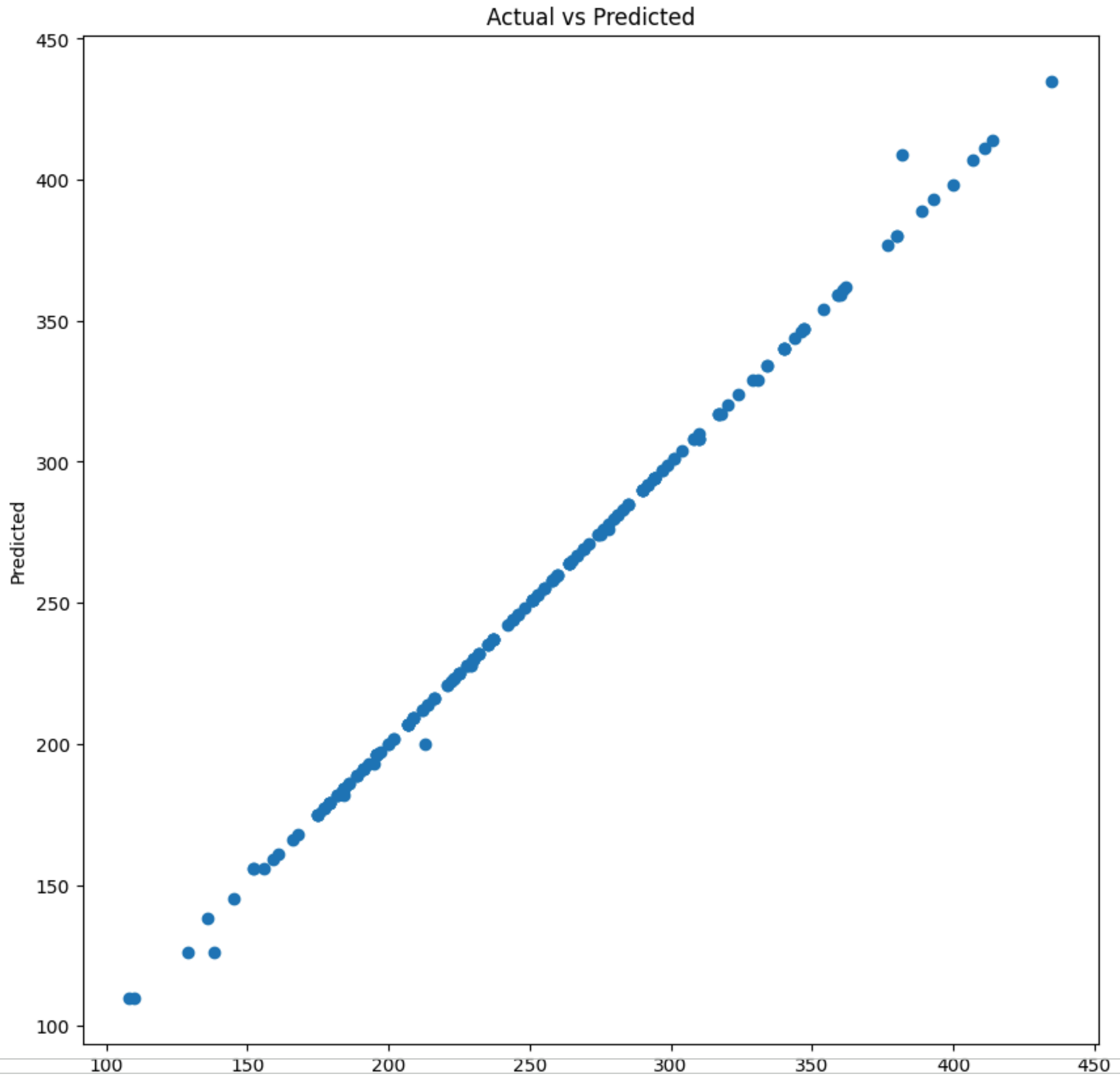
```
y_pred=model.predict(x_test)
```

```
from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error

print(r2_score(y_test,y_pred))
print(mean_absolute_error(y_test,y_pred))
print(mean_squared_error(y_test,y_pred))
```

```
0.9987252484849758
0.411214953271028
5.271028037383178
```

```
plt.figure(figsize=(10,10))
plt.scatter(y_test,y_pred)
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Actual vs Predicted')
plt.show()
```



Start coding or generate with AI.