In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Step 1: Load the medical dataset
# Replace 'medical_dataset.csv' with your actual dataset file
data = pd.read_csv('data[1].csv')
data
```

Out[1]:

	id	gender	dob	zipcode	employment_status	education	marital_status	children	ancestry	avg_commute	daily_internet_use	available_vehicles	military_service
0	Amelia Nixon	female	1944-03-09	89136	retired	bachelors	married	1	Portugal	13.38	2.53	2	no
1	Clara Hicks	female	1966-07-02	94105	employed	phd/md	married	4	Sweden	15.16	6.77	2	no
2	Mason Brown	male	1981-05-31	89127	employed	masters	married	2	Germany	23.60	3.63	1	no
3	Michael Rice	male	1945-02-13	44101	retired	bachelors	married	2	Denmark	19.61	5.00	3	no
4	Eleanor Ritter	female	1939-09-03	89136	retired	masters	married	3	Austria	36.55	7.75	1	no
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1995	Ethan Johnson	male	1942-04-13	89127	retired	masters	married	1	Switzerland	28.48	5.88	1	no
1996	Natalia Dominguez	female	1963-05-10	60612	unemployed	highschool	married	4	Denmark	21.09	5.92	0	no
1997	Joseph Zuniga	male	1965-07-12	94110	employed	bachelors	married	2	Russia	30.80	4.91	2	no
1998	Daniel Murphy	male	1926-08-10	90015	retired	bachelors	married	3	Finland	37.56	2.69	0	no
1999	Samuel Harris	male	1948-11-22	43210	retired	bachelors	married	5	Scotland	49.34	4.77	3	no

2000 rows × 14 columns

In [2]:
```python
X= data[['daily_internet_use']]
y=data['available_vehicles']
```

In [3]:
```python
# Split the data 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)
```

In [4]:
```python
# Specify the degree of the polynomial (e.g., quadratic, cubic)
degree = 2

# Create polynomial features
poly_features = PolynomialFeatures(degree=degree)
X_train_poly = poly_features.fit_transform(X_train)
X_test_poly = poly_features.transform(X_test)
```

In [5]:
```python
# Create a polynomial regression model
model = LinearRegression()

# Train the model on the polynomial features
model.fit(X_train_poly, y_train)
```

Out[5]:
```
▾ LinearRegression
LinearRegression()
```

In [6]:
```python
# Make predictions on the testing data
y_pred = model.predict(X_test_poly)

# Evaluate the model's performance
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)
```

```
Mean Squared Error (MSE): 1.2090248183750671
R-squared (R2): -0.011080528004906443
```

In [7]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Step 1: Load the medical dataset
# Replace 'medical_dataset.csv' with your actual dataset file
data = pd.read_csv('city_temperature[1].csv')
data
```

```
C:\Users\nishi\AppData\Local\Temp\ipykernel_7736\2357946565.py:11: DtypeWarning: Columns (2) have mixed types. Specify dtype option on import or set low_memory=False.
  data = pd.read_csv('city_temperature[1].csv')
```

Out[7]:

	Region	Country	State	City	Month	Day	Year	AvgTemperature
0	Africa	Algeria	NaN	Algiers	1	1	1995	64.2
1	Africa	Algeria	NaN	Algiers	1	2	1995	49.4
2	Africa	Algeria	NaN	Algiers	1	3	1995	48.8
3	Africa	Algeria	NaN	Algiers	1	4	1995	46.4
4	Africa	Algeria	NaN	Algiers	1	5	1995	47.9
...	...	...	...	...	...	...	...	...
2906322	North America	US	Additional Territories	San Juan Puerto Rico	7	27	2013	82.4
2906323	North America	US	Additional Territories	San Juan Puerto Rico	7	28	2013	81.6
2906324	North America	US	Additional Territories	San Juan Puerto Rico	7	29	2013	84.2
2906325	North America	US	Additional Territories	San Juan Puerto Rico	7	30	2013	83.8
2906326	North America	US	Additional Territories	San Juan Puerto Rico	7	31	2013	83.6

2906327 rows × 8 columns

In [8]:
```python
X= data[['Day']]
y=data['AvgTemperature']
```

In [9]:
```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [10]:
```python
# Specify the degree of the polynomial (e.g., quadratic, cubic)
degree = 2

# Create polynomial features
poly_features = PolynomialFeatures(degree=degree)
X_train_poly = poly_features.fit_transform(X_train)
X_test_poly = poly_features.transform(X_test)
```

In [11]:
```python
# Create a polynomial regression model
model = LinearRegression()

# Train the model on the polynomial features
model.fit(X_train_poly, y_train)
```

Out[11]:
```
▾ LinearRegression
LinearRegression()
```

In [12]:
```python
# Make predictions on the testing data
y_pred = model.predict(X_test_poly)

# Evaluate the model's performance
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)
```

```
Mean Squared Error (MSE): 1028.5467147230481
R-squared (R2): 8.186790901754115e-06
```

EXAMPLES

In [13]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn .linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error
```

In [14]:
```python
data=pd.read_csv('FuelConsumption[1].csv')
data
```

Out[14]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	FUELCONSUMPTION_HWY	FUELCONSUM
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	6.7	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	7.7	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	5.8	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	9.1	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	8.7	
...	...	...	...	...	...	...	...	...	...	...	
1062	2014	VOLVO	XC60 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	9.8	
1063	2014	VOLVO	XC60 AWD	SUV - SMALL	3.2	6	AS6	X	13.2	9.5	
1064	2014	VOLVO	XC70 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	9.8	
1065	2014	VOLVO	XC70 AWD	SUV - SMALL	3.2	6	AS6	X	12.9	9.3	
1066	2014	VOLVO	XC90 AWD	SUV - STANDARD	3.2	6	AS6	X	14.9	10.2	

1067 rows × 13 columns

In [15]:
```python
X=data[['ENGINESIZE']]
y=data['CO2EMISSIONS']
```

In [16]:
```python
X_train,X_test,y_train,y_test= train_test_split(X,y,test_size=0.2,random_state=42)
```

In [17]:
```python
Degree=2
poly_feature=PolynomialFeatures(degree=degree)
X_train_poly=poly_feature.fit_transform(X_train)
X_test_poly=poly_feature.transform(X_test)
```

In [18]:
```python
model = LinearRegression()
model.fit(X_train_poly, y_train)
```

Out[18]:
```
▾ LinearRegression
LinearRegression()
```

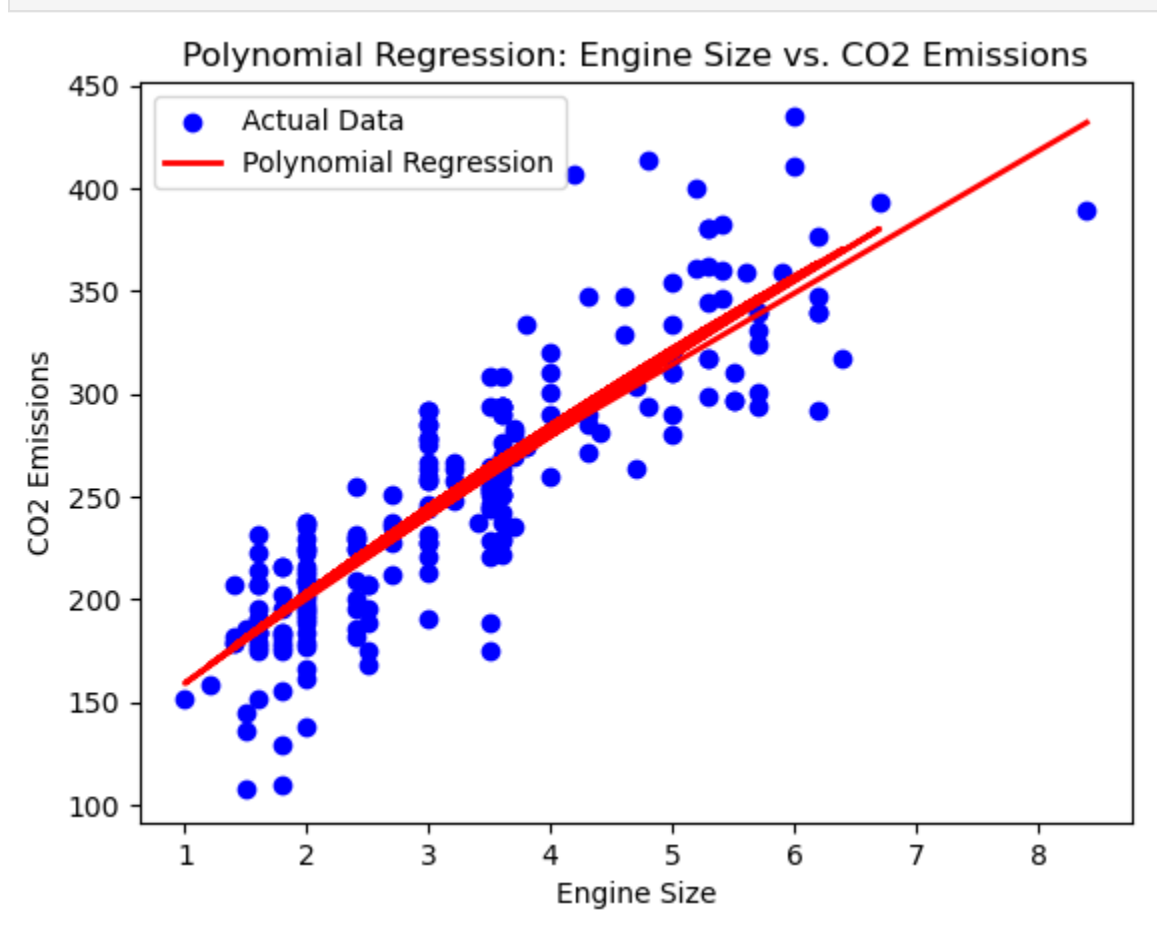
In [19]:
```python
y_pred=model.predict(X_test_poly)
```

In [20]:
```python
mse=mean_squared_error(y_test,y_pred)
r2=r2_score(y_test,y_pred)
```

In [21]:
```python
print(mse)
print(r2)
```

```
960.8705832028328
0.7676219471812655
```

In [22]:
```python
# Step 9: Visualize the Regression Curve and Data
plt.scatter(X_test, y_test, color='blue', label='Actual Data')
plt.plot(X_test, y_pred, color='red', linewidth=2, label='Polynomial Regression')
plt.xlabel('Engine Size')
plt.ylabel('CO2 Emissions')
plt.title('Polynomial Regression: Engine Size vs. CO2 Emissions')
plt.legend()
plt.show()
```



Example

In [23]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [24]:
```python
# Step 1: Load the Fuel Consumption Dataset
data = pd.read_csv('FuelConsumption[1].csv')
data
```

Out[24]:

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	FUELCONSUMPTION_HWY	FUELCONSUM
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	6.7	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	7.7	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	5.8	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	9.1	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	8.7	
...	...	...	...	...	...	...	...	...	...	...	
1062	2014	VOLVO	XC60 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	9.8	
1063	2014	VOLVO	XC60 AWD	SUV - SMALL	3.2	6	AS6	X	13.2	9.5	
1064	2014	VOLVO	XC70 AWD	SUV - SMALL	3.0	6	AS6	X	13.4	9.8	
1065	2014	VOLVO	XC70 AWD	SUV - SMALL	3.2	6	AS6	X	12.9	9.3	
1066	2014	VOLVO	XC90 AWD	SUV - STANDARD	3.2	6	AS6	X	14.9	10.2	

1067 rows × 13 columns

In [25]:
```python
# Step 2: Select Features and Target Variable
X = data[['ENGINESIZE']]  # Feature (e.g., engine size)
y = data['CO2EMISSIONS']  # Target variable (e.g., CO2 emissions)
```

In [26]:
```python
# Step 3: Split the Data 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)
```

In [27]:
```python
degree = 2  # Adjust the degree as needed
poly_features = PolynomialFeatures(degree=degree)
X_train_poly = poly_features.fit_transform(X_train)
X_test_poly = poly_features.transform(X_test)
```

In [28]:
```python
# Step 5: Create and Train the Polynomial Regression Model
model = LinearRegression()
model.fit(X_train_poly, y_train)
```

Out[28]:
```
▾ LinearRegression
LinearRegression()
```

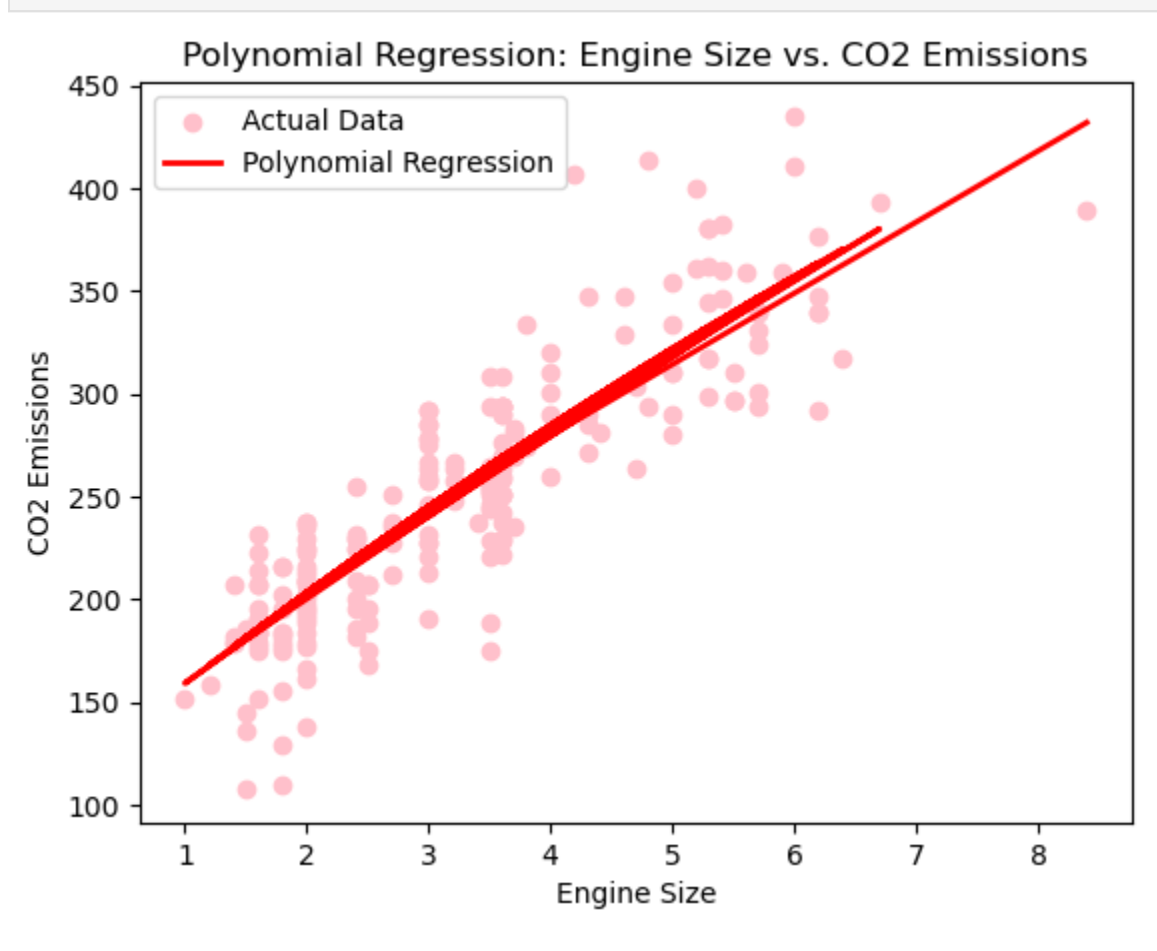
In [29]:
```python
# Step 6: Make Predictions
y_pred = model.predict(X_test_poly)
```

In [30]:
```python
# Step 7: Evaluate the Model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

In [31]:
```python
# Step 8: Print Evaluation Metrics
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)
```

```
Mean Squared Error (MSE): 960.8705832028328
R-squared (R2): 0.7676219471812655
```

In [32]:
```python
# Step 9: Visualize the Regression Curve and Data
plt.scatter(X_test, y_test, color='pink', label='Actual Data')
plt.plot(X_test, y_pred, color='red', linewidth=2, label='Polynomial Regression')
plt.xlabel('Engine Size')
plt.ylabel('CO2 Emissions')
plt.title('Polynomial Regression: Engine Size vs. CO2 Emissions')
plt.legend()
plt.show()
```



EXAMPLE

In [34]:
```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Load the Wine Quality Dataset
data = pd.read_csv('WineQT[1].csv')
data
```

Out[34]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5	0
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5	1
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5	2
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6	3
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5	4
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1138	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6	1592
1139	6.8	0.620	0.08	1.9	0.068	28.0	38.0	0.99651	3.42	0.82	9.5	6	1593
1140	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5	1594
1141	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6	1595
1142	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5	1597

1143 rows × 13 columns

In [35]:
```python
# Select relevant features and target variable (e.g., 'alcohol' as a feature and 'quality' as the target)
X = data[['alcohol']]  # Feature
y = data['quality']    # Target variable
```

In [36]:
```python
# Split the data 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)
```

In [37]:
```python
# Specify the degree of the polynomial
degree = 2

# Create polynomial features
poly_features = PolynomialFeatures(degree=degree)
X_train_poly = poly_features.fit_transform(X_train)
X_test_poly = poly_features.transform(X_test)
```

In [38]:
```python
# Create a polynomial regression model
model = LinearRegression()

# Train the model on the polynomial features
model.fit(X_train_poly, y_train)
```

Out[38]:
```
▾ LinearRegression
LinearRegression()
```

In [39]:
```python
# Make predictions on the testing data
y_pred = model.predict(X_test_poly)
```

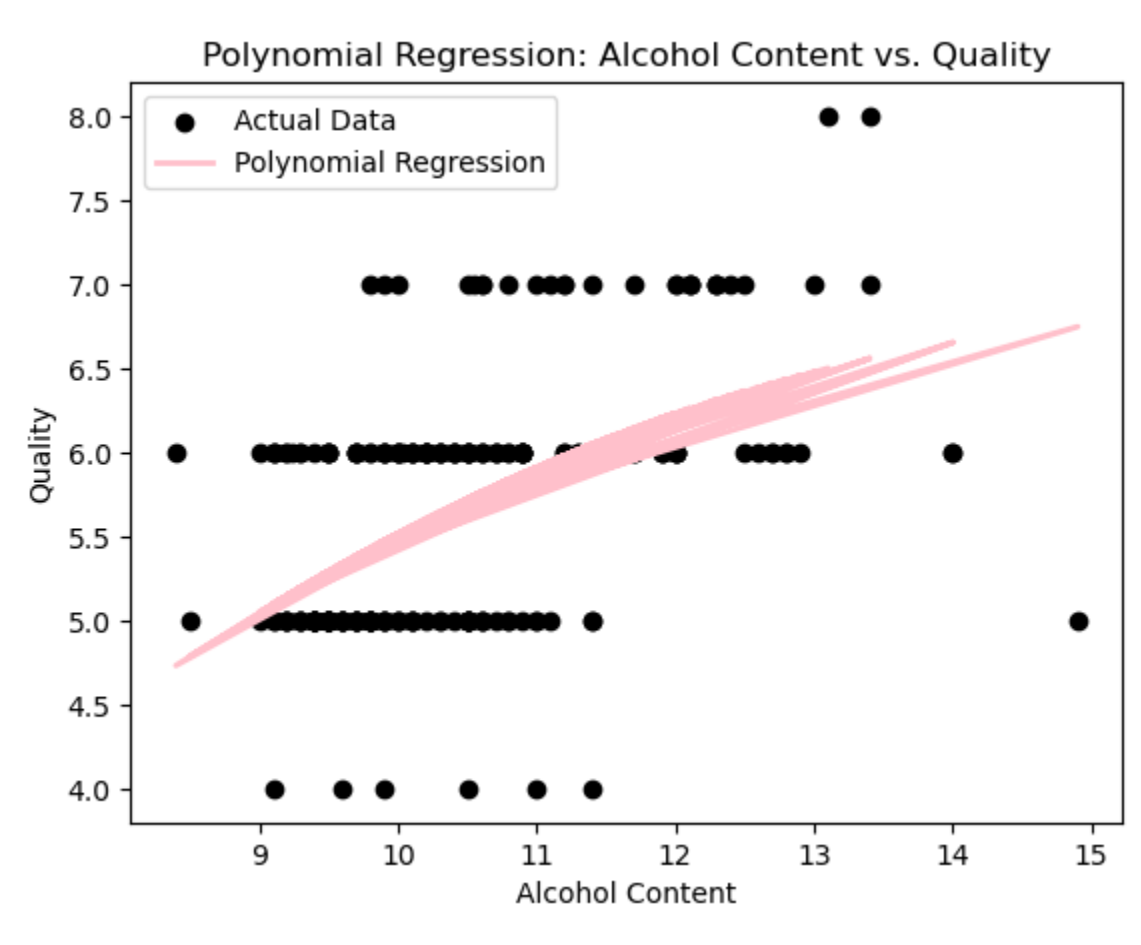
In [40]:
```python
# Evaluate the model's performance
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)
```

```
Mean Squared Error (MSE): 0.41093136962180976
R-squared (R2): 0.26154300752733484
```

In [43]:
```python
plt.scatter(X_test, y_test, color='black', label='Actual Data')
plt.plot(X_test, y_pred, color='pink', linewidth=2, label='Polynomial Regression')
plt.xlabel('Alcohol Content')
plt.ylabel('Quality')
plt.title('Polynomial Regression: Alcohol Content vs. Quality')
plt.legend()
plt.show
```

Out[43]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [44]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [45]:
```python
data=pd.read_csv('concrete_data[1].csv')
data
```

Out[45]:

	cement	blast_furnace_slag	fly_ash	water	superplasticizer	coarse_aggregate	fine_aggregate	age	concrete_compressive_strength
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.99
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	28	61.89
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	270	40.27
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	365	41.05
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	360	44.30
...	...	...	...	...	...	...	...	...	...
1025	276.4	116.0	90.3	179.6	8.9	870.1	768.3	28	44.28
1026	322.2	0.0	115.6	196.0	10.4	817.9	813.4	28	31.18
1027	148.5	139.4	108.6	192.7	6.1	892.4	780.0	28	23.70
1028	159.1	186.7	0.0	175.6	11.3	989.6	788.9	28	32.77
1029	260.9	100.5	78.3	200.6	8.6	864.5	761.5	28	32.40

1030 rows × 9 columns

In [47]:
```python
X = data[['cement']]  # Feature
y = data['concrete_compressive_strength']  # Target variable
```

In [48]:
```python
# Split the data 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)
```

In [49]:
```python
# Specify the degree of the polynomial
degree = 2

# Create polynomial features
poly_features = PolynomialFeatures(degree=degree)
X_train_poly = poly_features.fit_transform(X_train)
X_test_poly = poly_features.transform(X_test)
```

In [50]:
```python
# Create a polynomial regression model
model = LinearRegression()

# Train the model on the polynomial features
model.fit(X_train_poly, y_train)
```

Out[50]:
```
▾ LinearRegression
LinearRegression()
```

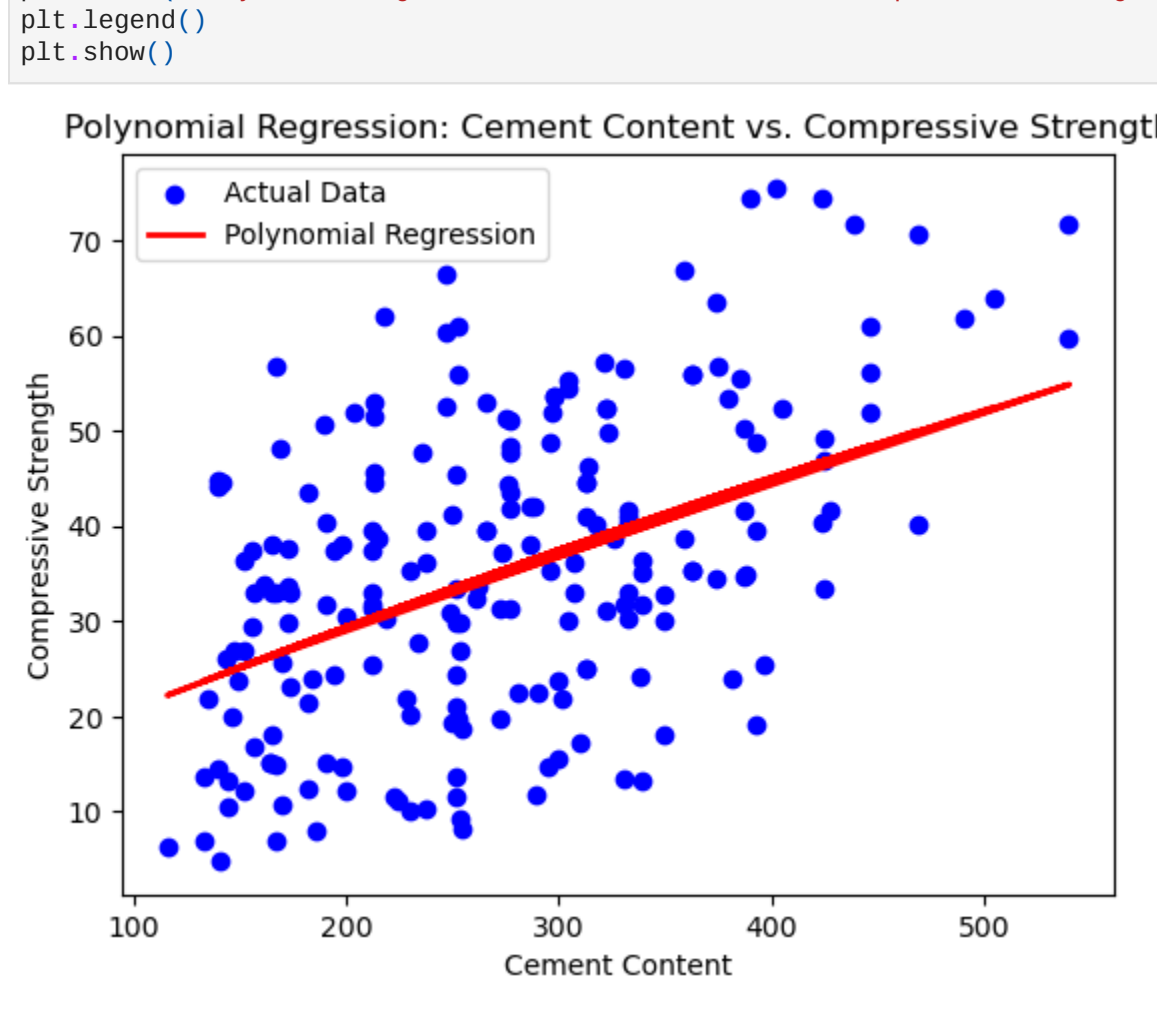
In [51]:
```python
# Make predictions on the testing data
y_pred = model.predict(X_test_poly)
```

In [52]:
```python
# Evaluate the model's performance
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print evaluation metrics
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)
```

```
Mean Squared Error (MSE): 193.14486285131377
R-squared (R2): 0.2504377882674438
```

In [53]:
```python
plt.scatter(X_test, y_test, color='blue', label='Actual Data')
plt.plot(X_test, y_pred, color='red', linewidth=2, label='Polynomial Regression')
plt.xlabel('Cement Content')
plt.ylabel('Compressive Strength')
plt.title('Polynomial Regression: Cement Content vs. Compressive Strength')
plt.legend()
plt.show()
```



In [ ]: