In [4]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import learning_curve
from sklearn.model_selection import train_test_split
```

In [5]:
```python
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/wine
data = pd.read_csv(url, sep=';')
data
```

Out[5]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcol
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	0.45	
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	0.49	
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	0.44	1
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.40	
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.40	
...	...	...	...	...	...	...	...	...	...	...	
4893	6.2	0.21	0.29	1.6	0.039	24.0	92.0	0.99114	3.27	0.50	1
4894	6.6	0.32	0.36	8.0	0.047	57.0	168.0	0.99490	3.15	0.46	
4895	6.5	0.24	0.19	1.2	0.041	30.0	111.0	0.99254	2.99	0.46	
4896	5.5	0.29	0.30	1.1	0.022	20.0	110.0	0.98869	3.34	0.38	1
4897	6.0	0.21	0.38	0.8	0.020	22.0	98.0	0.98941	3.26	0.32	1

4898 rows × 12 columns

In [6]:
```python
# Load the wine quality dataset
#data = pd.read_csv('winequality-white.csv', delimiter=';')

# Split the dataset into training and testing sets
X = data[['alcohol']].values
y = data['quality'].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_st
```

In [ ]:

In [ ]:

In [7]:
```python
# Fit a linear regression model to the data
lin_regressor = LinearRegression()
lin_regressor.fit(X_train, y_train)

# Fit a polynomial regression model to the data
poly = PolynomialFeatures(degree=2)
X_poly_train = poly.fit_transform(X_train)
X_poly_test = poly.transform(X_test)
```

```python
poly_regressor = LinearRegression()
poly_regressor.fit(X_poly_train, y_train)

# Predict the quality of the wine for the test data using both models
y_pred_lin = lin_regressor.predict(X_test)
y_pred_poly = poly_regressor.predict(X_poly_test)

# Print the performance metrics for both models
print('Linear Regression Metrics:')
mse_lin = mean_squared_error(y_test, y_pred_lin)
rmse_lin = np.sqrt(mse_lin)
r2_lin = r2_score(y_test, y_pred_lin)
print('Mean Squared Error: ', mse_lin)
print('Root Mean Squared Error: ', rmse_lin)
print('R-squared: ', r2_lin)
```

```
Linear Regression Metrics:
Mean Squared Error:  0.7306442340192562
Root Mean Squared Error:  0.8547773008329457
R-squared:  0.1710201454832172
```

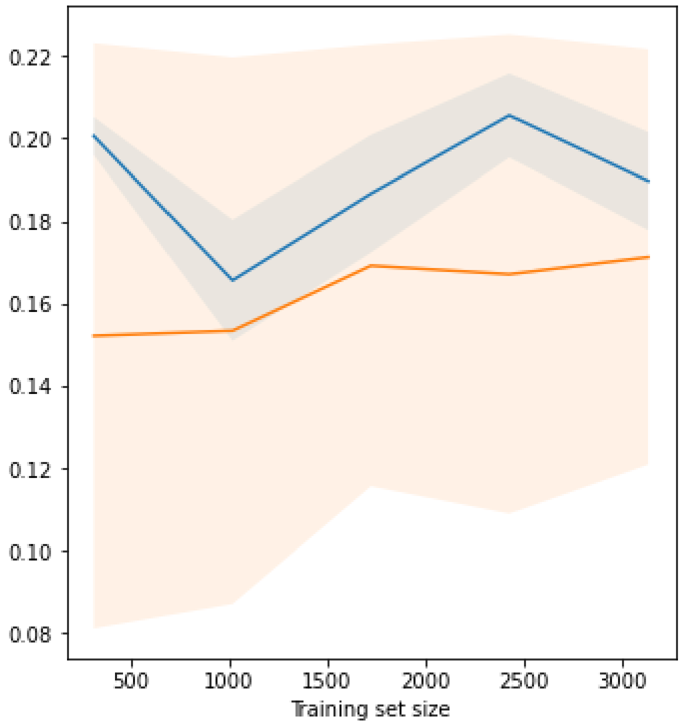
In [8]:

```python
print('Polynomial Regression Metrics:')
mse_poly = mean_squared_error(y_test, y_pred_poly)
rmse_poly = np.sqrt(mse_poly)
r2_poly = r2_score(y_test, y_pred_poly)
print('Mean Squared Error: ', mse_poly)
print('Root Mean Squared Error: ', rmse_poly)
print('R-squared: ', r2_poly)
```

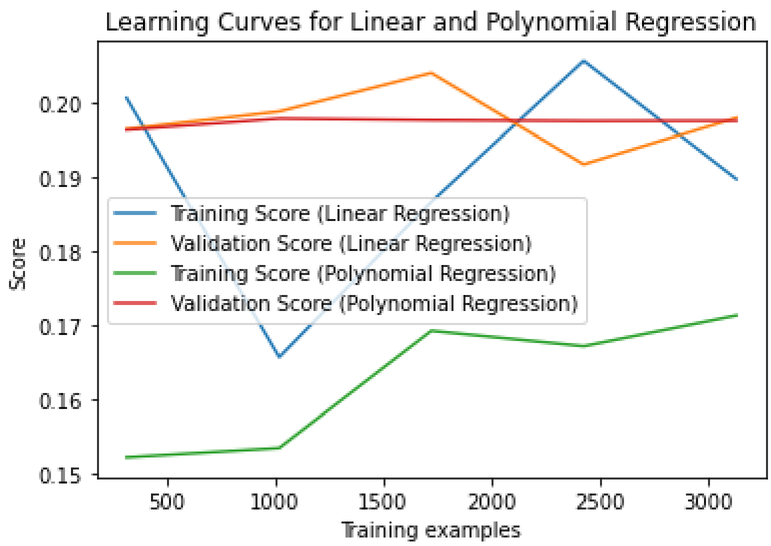
```
Polynomial Regression Metrics:
Mean Squared Error:  0.7321933575255221
Root Mean Squared Error:  0.8556829772325274
R-squared:  0.16926252923298202
```

In [9]:

```python
# Plot the learning curves for both models
train_sizes, train_scores_lin, test_scores_lin = learning_curve(lin_regressor, X, y
train_sizes, train_scores_poly, test_scores_poly = learning_curve(poly_regressor, X
train_mean_lin = np.mean(train_scores_lin, axis=1)
train_std_lin = np.std(train_scores_lin, axis=1)
test_mean_lin = np.mean(test_scores_lin, axis=1)
test_std_lin = np.std(test_scores_lin, axis=1)
train_mean_poly = np.mean(train_scores_poly, axis=1)
train_std_poly = np.std(train_scores_poly, axis=1)
test_mean_poly = np.mean(test_scores_poly, axis=1)
test_std_poly = np.std(test_scores_poly, axis=1)

plt.figure(figsize=(12,6))
plt.subplot(1,2,1)
plt.plot(train_sizes, train_mean_lin, label='Training score')
plt.plot(train_sizes, test_mean_lin, label='Cross-validation score')
plt.fill_between(train_sizes, train_mean_lin - train_std_lin, train_mean_lin + trai
plt.fill_between(train_sizes, test_mean_lin - test_std_lin, test_mean_lin + test_st
plt.xlabel('Training set size')
plt.show()
```



In [11]:
```python
import matplotlib.pyplot as plt

# Plot the learning curves for Linear Regression
plt.plot(train_sizes, train_mean_lin, label='Training Score (Linear Regression)')
plt.plot(train_sizes, train_mean_poly, label='Validation Score (Linear Regression)

# Plot the learning curves for Polynomial Regression
plt.plot(train_sizes, test_mean_lin, label='Training Score (Polynomial Regression)
plt.plot(train_sizes, test_mean_poly, label='Validation Score (Polynomial Regressi

# Set the plot title and labels
plt.title("Learning Curves for Linear and Polynomial Regression")
plt.xlabel("Training examples")
plt.ylabel("Score")

# Set the legend
plt.legend(loc="best")

# Show the plot
plt.show()
```



```
In [ ]:
```