Polynomial Regression

January 27, 2026

```
[1]: # TODO: Add import statements
     import numpy as np
     import pandas as pd
     from sklearn.linear_model import LinearRegression
     from sklearn.preprocessing import PolynomialFeatures
     import matplotlib.pyplot as plt
```

```
[2]: # Assign the data to predictor and outcome variables
     # TODO: Load the data
     train_data = pd.read_csv('data.csv')
     #X = train_data ['Var_X']
     Y = np.array(train_data ['Var_Y']).reshape(-1,1)
     X = np.array(train_data['Var_X']).reshape(-1,1)
     print(X.shape)
     print(X)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\AppData\Roaming\Python\Python311\site-packages\pandas\core\indexes\base.
  ↪py:3805, in Index.get_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.
  ↪PyObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.
  ↪PyObjectHashTable.get_item()

KeyError: 'Var_Y'

The above exception was the direct cause of the following exception:
```

```
KeyError                                  Traceback (most recent call last)
Cell In[2], line 5
      3 train_data = pd.read_csv('data.csv')
      4 #X = train_data ['Var_X']
----> 5 Y = np.array(train_data ['Var_Y']).reshape(-1,1)
      6 X = np.array(train_data['Var_X']).reshape(-1,1)
      7 print(X.shape)

File ~\AppData\Roaming\Python\Python311\site-packages\pandas\core\frame.py:4102
 ↪in DataFrame.__getitem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File ~\AppData\Roaming\Python\Python311\site-packages\pandas\core\indexes\base.
 ↪py:3812, in Index.get_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'Var_Y'
```

[ ]:
```
# Create polynomial features
# TODO: Create a PolynomialFeatures object, then fit and transform the
# predictor feature
poly_feat = PolynomialFeatures(degree=5)
X_poly = poly_feat.fit_transform(X)
print(X_poly)
```

```
[[ 1.00000000e+00 -3.35320000e-01  1.12439502e-01 -3.77032139e-02
   1.26426417e-02 -4.23933061e-03]
 [ 1.00000000e+00  2.16000000e-02  4.66560000e-04  1.00776960e-05
   2.17678234e-07  4.70184985e-09]
 [ 1.00000000e+00 -1.19438000e+00  1.42654358e+00 -1.70383513e+00
   2.03502660e+00 -2.43059507e+00]
 [ 1.00000000e+00 -6.50460000e-01  4.23098212e-01 -2.75208463e-01
   1.79012097e-01 -1.16440208e-01]
 [ 1.00000000e+00  -2.80010000e-01   7.84056001e-02  -2.19543521e-02
   6.14743813e-03  -1.72134415e-03]
 [ 1.00000000e+00   1.93258000e+00   3.73486546e+00   7.21792628e+00
   1.39492200e+01   2.69579835e+01]
 [ 1.00000000e+00   1.22620000e+00   1.50356644e+00   1.84367317e+00
   2.26071204e+00   2.77208510e+00]
 [ 1.00000000e+00   7.47270000e-01   5.58412453e-01   4.17284874e-01
   3.11824468e-01   2.33017070e-01]
 [ 1.00000000e+00   3.32853000e+00   1.10791120e+01   3.68771565e+01
   1.22746722e+02   4.08566146e+02]
 [ 1.00000000e+00   2.87457000e+00   8.26315268e+00   2.37530108e+01
   6.82796923e+01   1.96274755e+02]
 [ 1.00000000e+00  -1.48662000e+00   2.21003902e+00  -3.28548821e+00
   4.88427249e+00  -7.26105717e+00]
 [ 1.00000000e+00   3.76290000e-01   1.41594164e-01   5.32804680e-02
   2.00489073e-02   7.54420333e-03]
 [ 1.00000000e+00   1.43918000e+00   2.07123907e+00   2.98088585e+00
   4.29003130e+00   6.17412724e+00]
 [ 1.00000000e+00   2.41830000e-01   5.84817489e-02   1.41426413e-02
   3.42011495e-03   8.27086399e-04]
 [ 1.00000000e+00  -2.79140000e+00   7.79191396e+00  -2.17503486e+01
   6.07139232e+01  -1.69476845e+02]
 [ 1.00000000e+00   1.08176000e+00   1.17020470e+00   1.26588063e+00
   1.36937903e+00   1.48133946e+00]
 [ 1.00000000e+00   2.81555000e+00   7.92732180e+00   2.23197709e+01
   6.28424310e+01   1.76936006e+02]
 [ 1.00000000e+00   5.49240000e-01   3.01664578e-01   1.65686253e-01
   9.10015174e-02   4.99816734e-02]
 [ 1.00000000e+00   2.36449000e+00   5.59081296e+00   1.32194213e+01
   3.12571896e+01   7.39073121e+01]
 [ 1.00000000e+00  -1.01925000e+00   1.03887056e+00  -1.05886882e+00
   1.07925205e+00  -1.10002765e+00]]
```

[ ]:

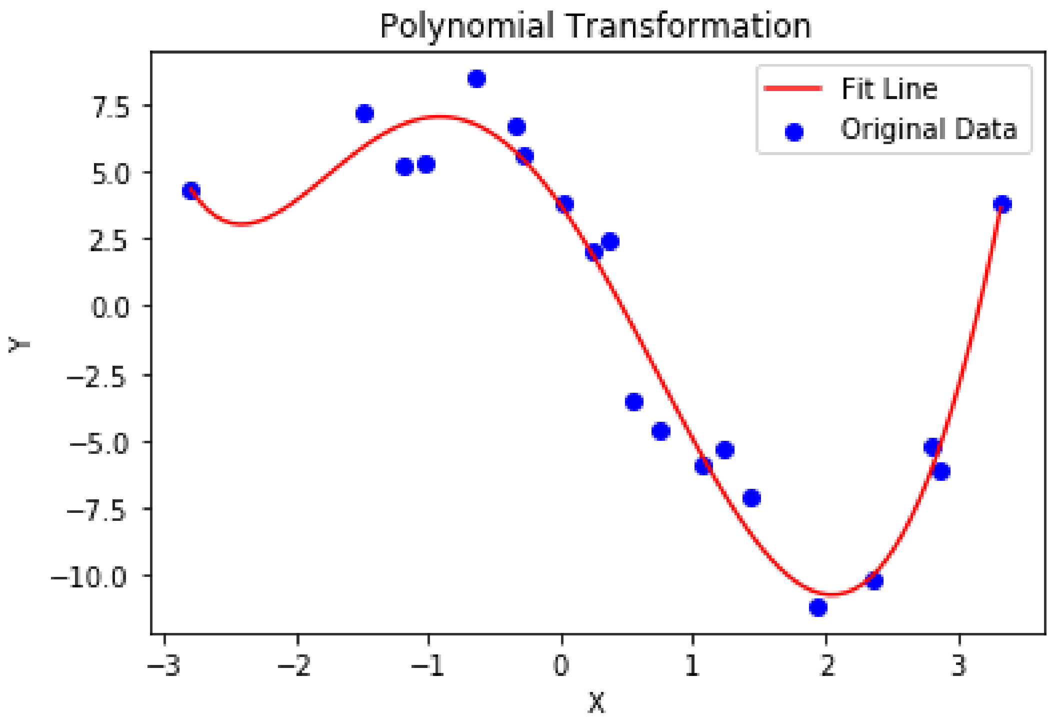
```
# Make and fit the polynomial regression model
# TODO: Create a LinearRegression object and fit it to the polynomial predictor
# features
model = LinearRegression()
poly_model = model.fit(X_poly, Y)

print(f"Coefficients: {poly_model.coef_}")
print(f"Intercept: {poly_model.intercept_}")
```

```
Coefficients: [[ 0.         -6.9182434  -3.03904779  0.99773696  0.31205509
-0.04531437]]
Intercept: [ 3.75718445]
```

[ ]:

```
# Plot the original data points
plt.scatter(X, Y, color='blue', label='Original Data')
```

```
#coefficients = poly_model.coef_
X = X.reshape(1,-1)
coefficients = np.polyfit(X.flatten(), Y.flatten(), deg = 4)
fit_line = np.poly1d(coefficients)


# Generate x values for the fit line
x_fit = np.linspace(X.min(), X.max(), 100)

# Plot the fit line
plt.plot(x_fit, fit_line(x_fit), color='red', label='Fit Line')

# Add labels and title to the plot
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Polynomial Transformation')

# Add legend
plt.legend()

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



```
[ ]:
```