sk_polynomialRegression

April 15, 2019

1 Polynomial Regression

In statistics, polynomial regression is a form of regression analysis in which the relationship between the independent variable x and the dependent variable y is modelled as an nth degree polynomial in x.

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

```
In [2]: # dataset
        boston = datasets.load_boston()
        print(boston.data.shape, boston.target.shape)
        print(boston.feature_names)
```

```
(506, 13) (506,)
['CRIM' 'ZN' 'INDUS' 'CHAS' 'NOX' 'RM' 'AGE' 'DIS' 'RAD' 'TAX' 'PTRATIO'
 'B' 'LSTAT']
```

```
In [3]: # using pandas for data handling
        data = pd.DataFrame(boston.data, columns=boston.feature_names)
        data = pd.concat([data, pd.Series(boston.target, name='MEDV')], axis=1)
        data.head()
```

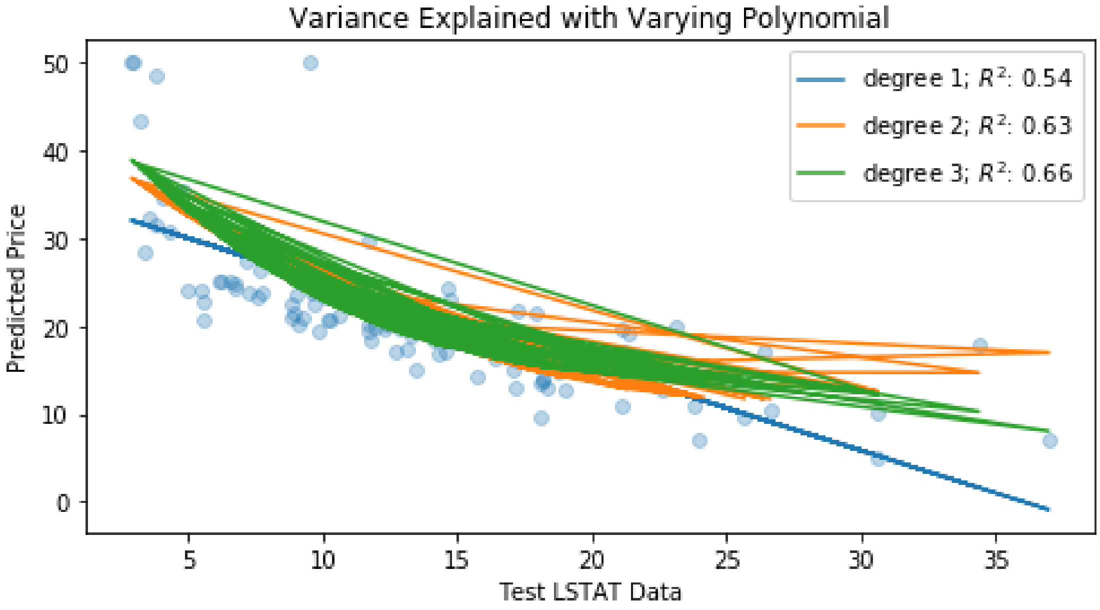
```
Out[3]:       CRIM    ZN  INDUS  CHAS    NOX     RM   AGE     DIS  RAD    TAX  \
        0  0.00632  18.0   2.31   0.0  0.538  6.575  65.2  4.0900  1.0  296.0
        1  0.02731   0.0   7.07   0.0  0.469  6.421  78.9  4.9671  2.0  242.0
        2  0.02729   0.0   7.07   0.0  0.469  7.185  61.1  4.9671  2.0  242.0
        3  0.03237   0.0   2.18   0.0  0.458  6.998  45.8  6.0622  3.0  222.0
        4  0.06905   0.0   2.18   0.0  0.458  7.147  54.2  6.0622  3.0  222.0
```

```
   PTRATIO       B  LSTAT  MEDV
0     15.3  396.90   4.98  24.0
1     17.8  396.90   9.14  21.6
2     17.8  392.83   4.03  34.7
3     18.7  394.63   2.94  33.4
4     18.7  396.90   5.33  36.2
```

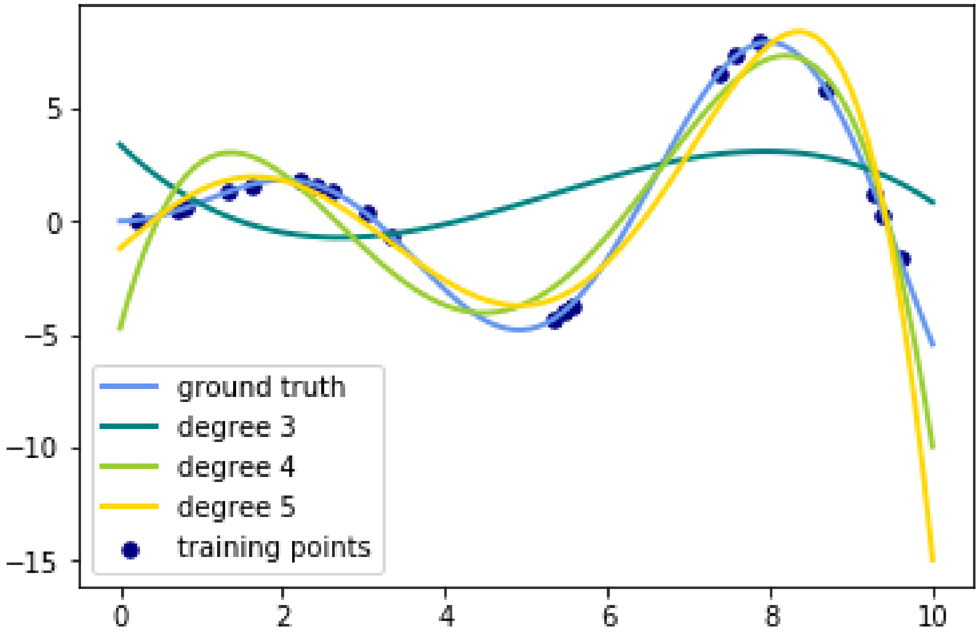
```
In [4]: # Feature Selection
        X = data[['LSTAT']]
        y = data[['MEDV']]
```

```
In [5]: # train test split
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_sta
```

```
In [6]: # Polynomial Regression nth order
        plt.figure(figsize=(8, 4))
        plt.scatter(X_test, y_test,  alpha=0.3)

        for degree in range(1, 4):
            model = make_pipeline(PolynomialFeatures(degree), LinearRegression())
            model.fit(X_train, y_train)

            y_pred = model.predict(X_test)
            plt.plot(X_test, y_pred, label="degree %d" % degree+'; $R^2$: %.2f' % model.score(
            plt.legend(loc='upper right')
            plt.xlabel("Test LSTAT Data")
            plt.ylabel("Predicted Price")
            plt.title("Variance Explained with Varying Polynomial")
```



```
In [7]: # Polynomial interpolation
        def f(x):
            """ function to approximate by polynomial interpolation"""
            return x * np.sin(x)


        # generate points used to plot
        x_plot = np.linspace(0, 10, 100)

        # generate points and keep a subset of them
        x = np.linspace(0, 10, 100)
        rng = np.random.RandomState(0)
        rng.shuffle(x)
        x = np.sort(x[:20])
        y = f(x)

        # create matrix versions of these arrays
        X = x[:, np.newaxis]
        X_plot = x_plot[:, np.newaxis]

        colors = ['teal', 'yellowgreen', 'gold']
        lw = 2
        plt.plot(x_plot, f(x_plot), color='cornflowerblue', linewidth=lw,
                 label="ground truth")
        plt.scatter(x, y, color='navy', s=30, marker='o', label="training points")

        for count, degree in enumerate([3, 4, 5]):
            model = make_pipeline(PolynomialFeatures(degree), LinearRegression())
            model.fit(X, y)
            y_plot = model.predict(X_plot)
            plt.plot(x_plot, y_plot, color=colors[count], linewidth=lw,
                     label="degree %d" % degree)

        plt.legend(loc='lower left')

        plt.show()
```



1.1 References:

1. https://acadgild.com/blog/polynomial-regression-understand-power-of-polynomials
2. https://en.wikipedia.org/wiki/Polynomial_regression
3. https://scikit-learn.org/stable/auto_examples/linear_model/plot_polynomial_interpolation.html