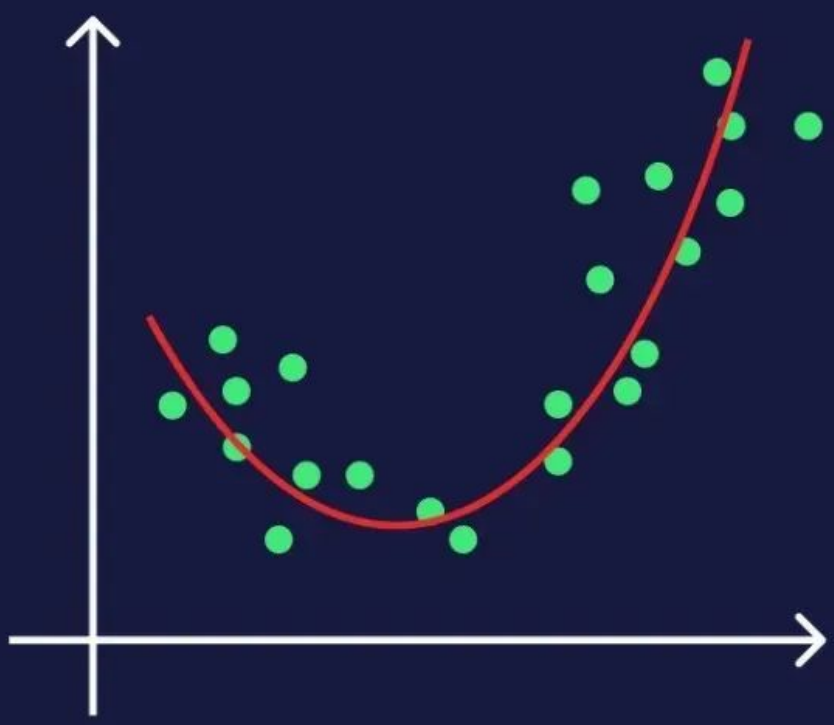
Polynomial Regression in ML



What is Polynomial Regression?

A regression procedure known as polynomial regression models the relationship between the independent variable (x) and dependent variable (y) as an nth degree polynomial.

- By making the necessary adjustments, a linear regression model is transformed into a polynomial regression model.

Polynomial Regression Algorithm:

The basic goal of any regression study is to represent an independent variable's expression in terms of the expected value of a dependent variable.

Polynomial Regression Equation:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n + \varepsilon.$$

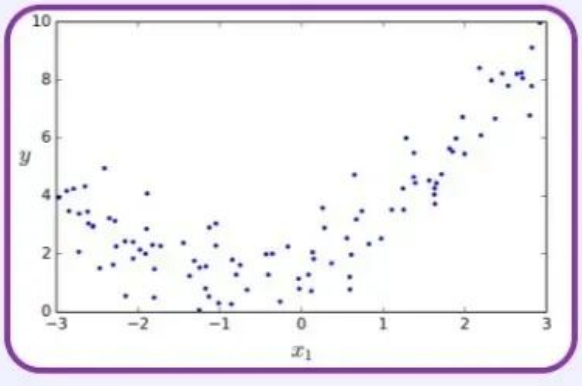
How it works?

Using a straightforward quadratic equation, let's create some nonlinear data.

Example:

```
m = 100
X = 6 * np.random.rand(m, 1) - 3
y = 0.5 * X**2 + X + 2 +
np.random.randn(m, 1)
```

Result:



Now, we should fit a model that passes through the datapoints.

Adding the square (2nd-degree polynomial) of each feature in the training set as new features

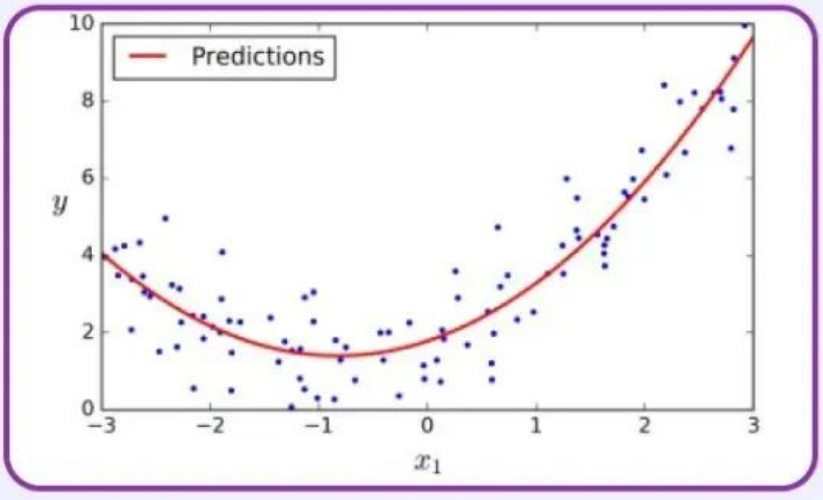
```
>>> from sklearn.preprocessing import PolynomialFeatures
>>> poly_features = PolynomialFeatures(degree=2,
include_bias=False)
>>> X_poly = poly_features.fit_transform(X)
>>> X[0]
array([-0.75275929])
>>> X_poly[0]
array([-0.75275929, 0.56664654])
```

The original feature of X as well as its square are now both present in X poly.

Example:

```
>>> lin_reg = LinearRegression()
>>> lin_reg.fit(X_poly, y)
>>> lin_reg.intercept_, lin_reg.coef_
(array([ 1.78134581]), array([[ 0.93366893, 0.56456263]]))
```

Result:



PolynomialFeatures(degree=d) transforms an array containing n features into an array containing (n+d)!/n!d! features.

Original model:

y = 0 . 5x^2 + 1 . 0x1 + 2 . 0 +
Gaussian noise.

Predicted Model:

y = 0 . 56x1^2 + 0 . 93x1 + 1 . 78

This has predicted very very closely.

Did you find it useful ?

Let us know in the comments

FOLLOW FOR MORE