Gurucharan M K Follow

Jul 5, 2020 · 5 min read ★ · Listen

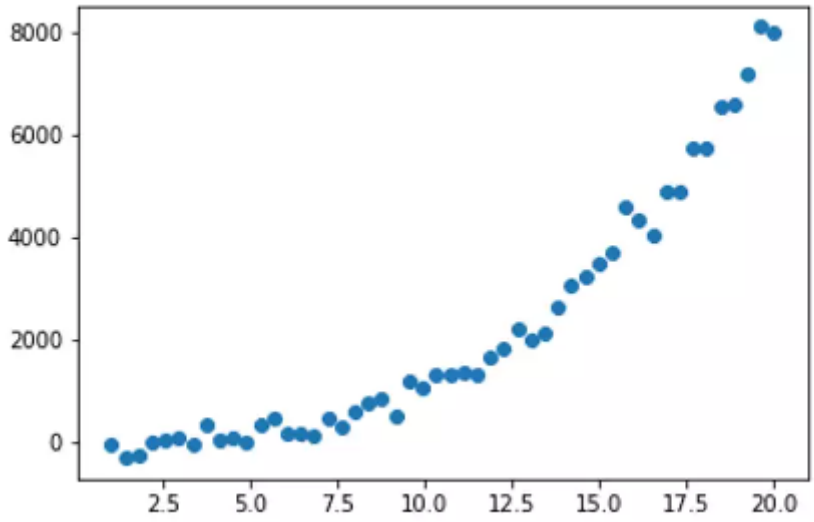
Machine Learning Basics: Polynomial Regression

Learn to build a Polynomial Regression model to predict the values for a non-linear dataset.

In previous stories, I have given a brief of Linear Regression and showed how to perform Simple and Multiple Linear Regression. In this article, we will go through the program for building a Polynomial Regression model based on the non-linear data.

Overview

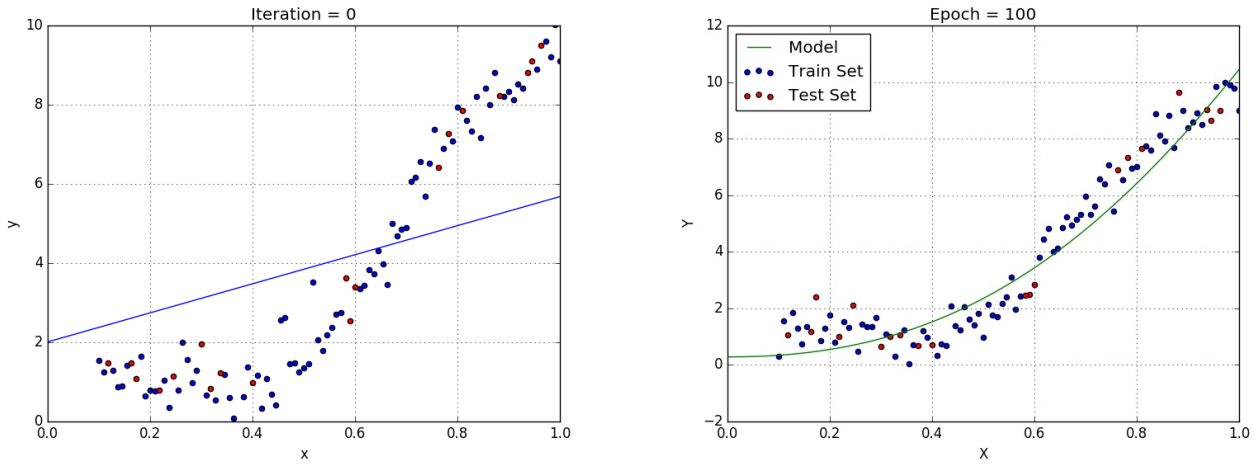
In the previous examples of Linear Regression, when the data is plotted on the graph, there was a linear relationship between both the dependent and independent variables. Thus, it was more suitable to build a linear model to get accurate predictions. What if the data points had the following non-linearity making the linear model giving an error in predictions due to non-linearity?



Non-Linear Data points (Source)

In this case, we have to build a polynomial relationship which will accurately fit the data points in the given plot. This is called Polynomial Regression. The formula for a Polynomial Regression curve is given as $y=w1x+w2x^2+..+b$

Below are the GIFs of fitting both a Linear Regression model and a Polynomial Regression model on a non-linear data.



Left: Linear Regression, Right: Polynomial regression (Source)

As we can see, the Linear regression always tends to make an error however hard it tries to fit in the data. On the other hand, the Polynomial Regression graph manages to

previous company from the salary data for the same position levels in the new company.

Problem Analysis

In this data, we have the two independent variables namely, *Position and Level*. There is one independent variable i.e., *Salary*. So, in this problem we have to train a Polynomial Regression model with this data to understand the correlation between the Level and Salary of the employee data in the company and be able to predict the salary for the new employee based on this data.

Step 1: Importing the libraries

In this first step, we will be importing the libraries required to build the ML model. The ***NumPy*** library and the ***matplotlib*** are imported. Additionally, we have imported the ***Pandas*** library for data analysis.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Step 2: Importing the dataset

In this step, we shall use pandas to store the data obtained from my github repository and store it as a Pandas DataFrame using the function "**pd.read_csv**".

We go through our dataset and assign the independent variable (x) to the second column with the column name "***Level***" and the dependent variable (y) to the last column, which is the "***Salary***" to be predicted.

```
dataset = pd.read_csv('https://raw.githubusercontent.com/mk-gurucharan/Regression/master/PositionSalaries_Data.csv')

X = dataset.iloc[:, 1:-1].values
y = dataset.iloc[:, -1].values

dataset.head(5)
```


```
Senior Consultant  3       60000
Manager            4       80000
Country Manager    5       110000
```

We use the corresponding .iloc function to slice the DataFrame to assign these indexes to X and Y. In this, the *Level* is taken as the independent variable and is assigned to X. The dependent variable that is to be predicted is the last column (-1) which is *Salary* and it is assigned to y. We print the DataFrame "**dataset**" to see if we have got the correct columns for our training data.

Step 3: Training the Polynomial Regression model on the whole dataset

The dataset on which we are using has very few number of rows and hence we train the entire dataset for building the Polynomial Regression model. In this the "***PolynomialFeatures***" function is used to assign the degree of the polynomial line that we are going to plot. In this, the degree is set as ***4***.

The independent variable X is then fitted with the PolynomialFeatures class and is converted to a new variable ***X_poly***. In this, the variable X is converted to a new matrix X_Poly which consists of all the polynomial combinations of features with degree=4.

```
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
poly_reg = PolynomialFeatures(degree = 4)
X_poly = poly_reg.fit_transform(X)
lin_reg = LinearRegression()
lin_reg.fit(X_poly, y)
```

The class "***LinearRegression"*** is also imported and is assigned to the variable ***"lin_reg"*** which is fitted with the X_poly and y for building the model.

Step 4: Predicting the Results

In this step, we are going to predict the values Salary based on the Polynomial Regression model built. The "**regressor.predict**" function is used to predict the values for our independent variable, X_poly. We assign the predicted values as y_pred. We now have two data, y(real values) and y_pred (predicted values).

Step 5: Comparing the Real Values with Predicted Values

In this step , we shall print both the values of y as ***Real Values*** and y_pred values as ***Predicted Values*** of each X_test in a Pandas DataFrame.

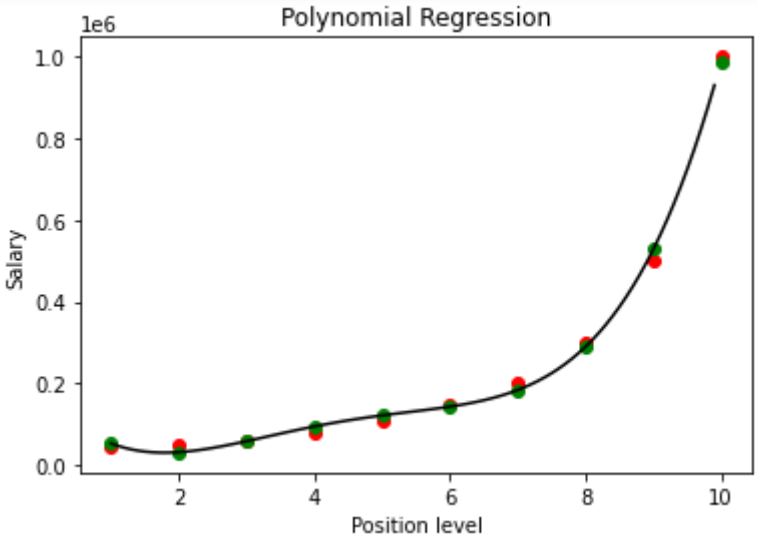
```
df = pd.DataFrame({'Real Values':y, 'Predicted Values':y_pred})
df

>>
Real Values  Predicted Values
45000        53356.643357
50000        31759.906760
60000        58642.191142
80000        94632.867133
110000       121724.941725
150000       143275.058275
200000       184003.496504
300000       289994.172494
500000       528694.638695
1000000      988916.083916
```

We can see that the model has done a great job in fitting the data and predicting the salary of the employee based on the position level.

Step 6: Visualising the Polynomial Regression results

In this last step, we shall visualize the polynomial model that was built using the given data and plot the values of "***y***" and "***y_pred***" on the graph and analyze the results

```
X_grid = np.arange(min(X), max(X), 0.1)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(X, y, color = 'red')
plt.scatter(X, y_pred, color = 'green')
plt.plot(X_grid, lin_reg.predict(poly_reg.fit_transform(X_grid)),
color = 'black')
plt.title('Polynomial Regression')
plt.xlabel('Position level')
plt.ylabel('Salary')
plt.show()
```

Polynomial Regression Model

In this graph, the Real values are plotted in "***Red***" color and the Predicted values are plotted in "***Green***" color. The Polynomial Regression line that is generated is drawn in "***Black***" color.

I am attaching a link of my github repository where you can find the Google Colab notebook and the data files for your reference.

mk-gurucharan/Regression

GitHub is home to over 50 million developers working together to host and review code, manage projects, and build...

github.com

Hope I have been able to clearly explain the program for building a Polynomial Regression model.

You can also find the explanation of the program for other Regression models below:

- Simple Linear Regression
- Multiple Linear Regression
- Polynomial Regression

- Random Forest Regression

We will come across the more complex models of Regression, Classification and Clustering in the upcoming articles. Till then, Happy Machine Learning!