Polynomial Regression

# Polynomial Regression in Python (PART 1)

Note: Polynomial regression is not a linear aggressor.

Business Problem:

* A HR team working for a big company and they need to hire a new employee
* About to make an offer to this new employee
* Now to negotiate the salary of employee
* He has 20 years of experience and earning 160k in previous role
* HR team gathers information from previous company to confirm above information, and the information gathered is a table of salaries for 10 different positions in previous company.

Summary Mission:

* New employee is telling that its annual salary was 160k, let’s predict if it’s truth or not by building a bluffing detector using polynomial regression.
* Dependent variable: Level of position
* Independent variable: Salary

Interesting Knowledge Notes:

* When building a matrix, if you get a shape (10,). This shows it’s a vector, not a matrix. We want to specify it’s a matrix not a vector so when importing the dataset again, chance X = dataset.iloc[:, 1:2] NOT dataset.iloc[:,1]. This prevents errors

Why do we not split the dataset into training set and test set in this data preprocessing?

1. We have a small number of observations in dataset, it doesn’t make sense to split training set into test set because we don’t have enough information to train the model.
2. To make an accurate prediction, we need the maximum information available, so our model perfectly understands the correlation of this dataset.

Why do we not do feature scaling on data preprocessing step?

1. PR is actually adding some polynomial terms into MR equation and therefore we will use same linear regression library we used to build our simple linear regression models.

# Polynomial Regression in Python (PART 2)

Objectives:

* Build a linear regression model to the dataset.
* Fitting PR to the dataset
* Compare the two models above. Making the linear regression model as a reference to compare to results of PR model.

Fitting Linear Regression to the Dataset:

* We need to import a linear regression class from preprocessing library (scalar and linear normal library) sklearn

Fitting Polynomial Regression to the dataset:

* Below polymorphic object is a transformer tool that will transform our matrix of features X into a new matrix of features named X\_poly
* When creating X\_poly, why do we use the fit\_transform command with our Polynomial Features object poly\_reg?
* We’re transforming X into X\_poly by fitting our object poly\_reg to X and then transform X into X\_poly

Final Step:

* Create a new linear regression object will to be fit that we made with poly\_reg object into our linear regression model
* Built 2 regressors already to compare PR with reference base regression model

# Polynomial Regression in Python (PART 3)

Objectives:

* Visualize the results of both models

Notes Visualising Linear Regression Results

* In visualizing the linear regression results, we plot the real observation points given to us on the data set and the predictions that is predicted salaries of the 10 levels predicted by our linear regression model
* Use plt.scatter or plt.plot
* Results? The real observations points is far from the linear regression line.
* We need to make a better model that matches our real observation dataset
* Make a non-linear model to make accurate predictions on salaries to aid on negotiations for salary

Notes Visualizing Polynomial Regression results:

* We have a curve (PR results) which is due to the PR model is not a linear regression model
* Make distinction between linear models and non-linear models (curve etc.)
* To improve on this model, we are going to add a degree when we made our poly\_reg object. When graph is observed, it approaches the observation points between with this new polynomial regression results than with degree = 2. Keep adding the degree until you get a great model to predict the salary associated

Making model better:

* Instead of having predictions from level 1 – 10 incremented by 1, we will increment a higher resolution like 0.1 step with X\_grid

# Polynomial Regression in Python (PART 4)

Objectives:

* Predict truth or bluff by predicting the previous salary of employee (160K) to our predictions to see if his/her lying.