Summary Report 1 on ML Course

Yahya Ramzy [02-SBIRfeatureX]

March 2023

1 Multiple Linear Regression

1.1 Data Set

R Profit	D Spend	Administration	Marketing Spend	State
165349.2	136897.8	471784.1	New York	192261.83
162597.7	151377.59	443898.53	California	191792.06
153441.51	101145.55	407934.54	Florida	191050.39
144372.41	118671.85	383199.62	New York	182901.99
142107.34	91391.77	366168.42	Florida	166187.94
131876.9	99814.71	362861.36	New York	156991.12
134615.46	147198.87	127716.82	California	156122.51
130298.13	145530.06	323876.68	Florida	155752.6
120542.52	148718.95	311613.29	New York	152211.77
123334.88	108679.17	304981.62	California	149759.96
101913.08	110594.11	229160.95	Florida	146121.95
100671.96	91790.61	249744.55	California	144259.4
93863.75	127320.38	249839.44	Florida	141585.52
91992.39	135495.07	252664.93	California	134307.35
119943.24	156547.42	256512.92	Florida	132602.65
114523.61	122616.84	261776.23	New York	129917.04
78013.11	121597.55	264346.06	California	126992.93
94657.16	145077.58	282574.31	New York	125370.37
91749.16	114175.79	294919.57	Florida	124266.9
86419.7	153514.11	0	New York	122776.86
76253.86	113867.3	298664.47	California	118474.03
78389.47	153773.43	299737.29	New York	111313.02
73994.56	122782.75	303319.26	Florida	110352.25
67532.53	105751.03	304768.73	Florida	108733.99
77044.01	99281.34	140574.81	New York	108552.04
64664.71	139553.16	137962.62	California	107404.34
75328.87	144135.98	134050.07	Florida	105733.54
72107.6	127864.55	353183.81	New York	105008.31
66051.52	182645.56	118148.2	Florida	103282.38
65605.48	153032.06	107138.38	New York	101004.64
61994.48	115641.28	91131.24	Florida	99937.59
61136.38	152701.92	88218.23	New York	97483.56
63408.86	129219.61	46085.25	California	97427.84
55493.95	103057.49	214634.81	Florida	96778.92
46426.07	157693.92	210797.67	California	96712.8
46014.02	85047.44	205517.64	New York	96479.51
28663.76	127056.21	201126.82	Florida	90708.19
44069.95	51283.14	197029.42	California	89949.14
20229.59	65947.93	185265.1	New York	81229.06
38558.51	82982.09	174999.3	California	81005.76
28754.33	118546.05	172795.67	California	78239.91
27892.92	84710.77	164470.71	Florida	77798.83
23640.93	96189.63	148001.11	California	71498.49
15505.73	127382.3	35534.17	New York	69758.98
22177.74	154806.14	28334.72	California	65200.33
1000.23	124153.04	1903.93	New York	64926.08
1315.46	115816.21	297114.46	Florida	49490.75
0	135426.92	0	California	42559.73
542.05	51743.15	0	New York	35673.41
0	116983.8	45173.06	California	14681.4

1.2 Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
```

1.3 Preprocessing Data

```
dataset = pd.read_csv('50_Startups.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values

ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [3])],
remainder='passthrough')
X = np.array(ct.fit_transform(X))

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
random_state = 0)
```

1.4 Building Model

```
from sklearn.linear_model import LinearRegression
regresser = LinearRegression()
regresser.fit(X_train,y_train)
```

1.5 Predicting Test Results

```
y_pred = regresser.predict(X_test)
np.set_printoptions(precision=2)
print(np.concatenate((y_pred.reshape(len(y_pred),1),
y_test.reshape(len(y_test),1)), 1))
```

Result :
[103015.2 103282.38]
[132582.28 144259.4]
[132447.74 146121.95]
[71976.1 77798.83]
[178537.48 191050.39]
[116161.24 105008.31]
[67851.69 81229.06]
[98791.73 97483.56]
[113969.44 110352.25]
[167921.07 166187.94]
R2 Score : 0.934

2 Polynomial Regression

2.1 Data Set

Position	Level	Salary
Business Analyst	1	45000
Junior Consultant	2	50000
Senior Consultant	3	60000
Manager	4	80000
Country Manager	5	110000
Region Manager	6	150000
Partner	7	200000
Senior Partner	8	300000
C-level	9	500000
CEO	10	1000000

2.2 Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
```

2.3 Importing Data Set

```
dataset = pd.read_csv('Position_Salaries.csv')
X = dataset.iloc[:, 1:-1].values
y = dataset.iloc[:, -1].values
```

2.4 Training the Linear Regression model on the whole dataset

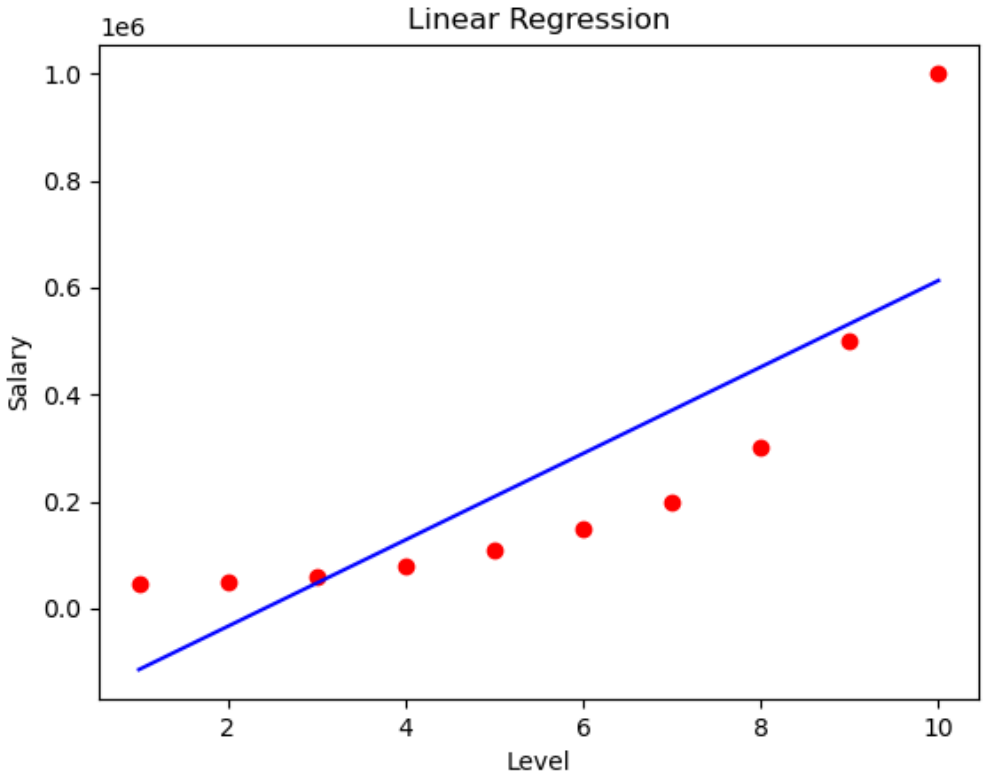
```
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression()
lin_reg.fit(X, y)
```

2.5 Training the Polynomial Regression model on the whole dataset

```
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree = 4)
X_poly = poly_reg.fit_transform(X)
lin_reg_2 = LinearRegression()
lin_reg_2.fit(X_poly, y)
```

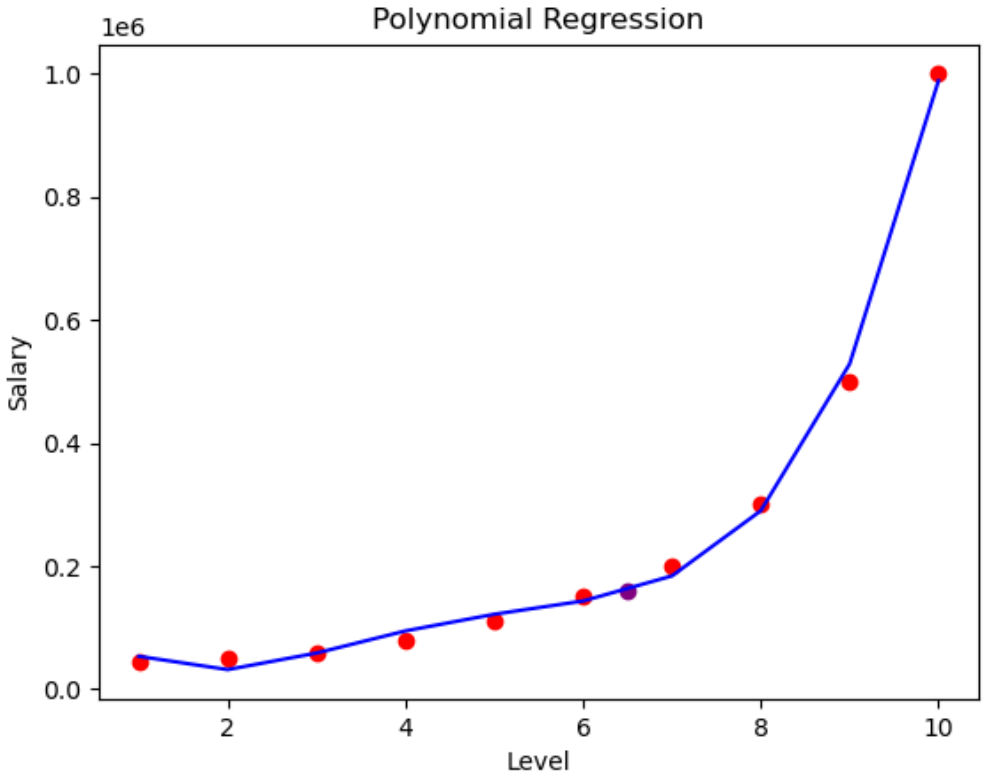
2.6 Visualising the Linear Regression results

```
plt.scatter(X, y, color = 'red')
plt.plot(X, lin_reg.predict(X), color = 'blue')
plt.title('Linear Regression')
plt.xlabel('Position Level')
plt.ylabel('Salary')
plt.show()
```



2.7 Visualising the Polynomial Regression results

```
plt.scatter(X, y, color = 'red')
plt.plot(X, lin_reg_2.predict(poly_reg.fit_transform(X)), color = 'blue')
plt.title('Truth or Bluff (Polynomial Regression)')
plt.xlabel('Position level')
plt.ylabel('Salary')
plt.show()
```



3 Support Vector Regression (SVR)

3.1 Data Set

Position	Level	Salary
Business Analyst	1	45000
Junior Consultant	2	50000
Senior Consultant	3	60000
Manager	4	80000
Country Manager	5	110000
Region Manager	6	150000
Partner	7	200000
Senior Partner	8	300000
C-level	9	500000
CEO	10	1000000

3.2 Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.svm import SVR
from sklearn.preprocessing import StandardScaler
```

3.3 Importing Dataset

```
dataset = pd.read_csv('Position_Salaries.csv')
X = dataset.iloc[:, 1:-1].values
y = dataset.iloc[:, -1].values

y = y.reshape(len(y),1)
```

3.4 Feature Scaling

```
sc_X = StandardScaler()
sc_y = StandardScaler()
X = sc_X.fit_transform(X)
y = sc_y.fit_transform(y)
```

3.5 Training the SVR model on the whole dataset

```
reg = SVR(kernel='rbf')
reg.fit(X,y)
```

3.6 Predicting a new result

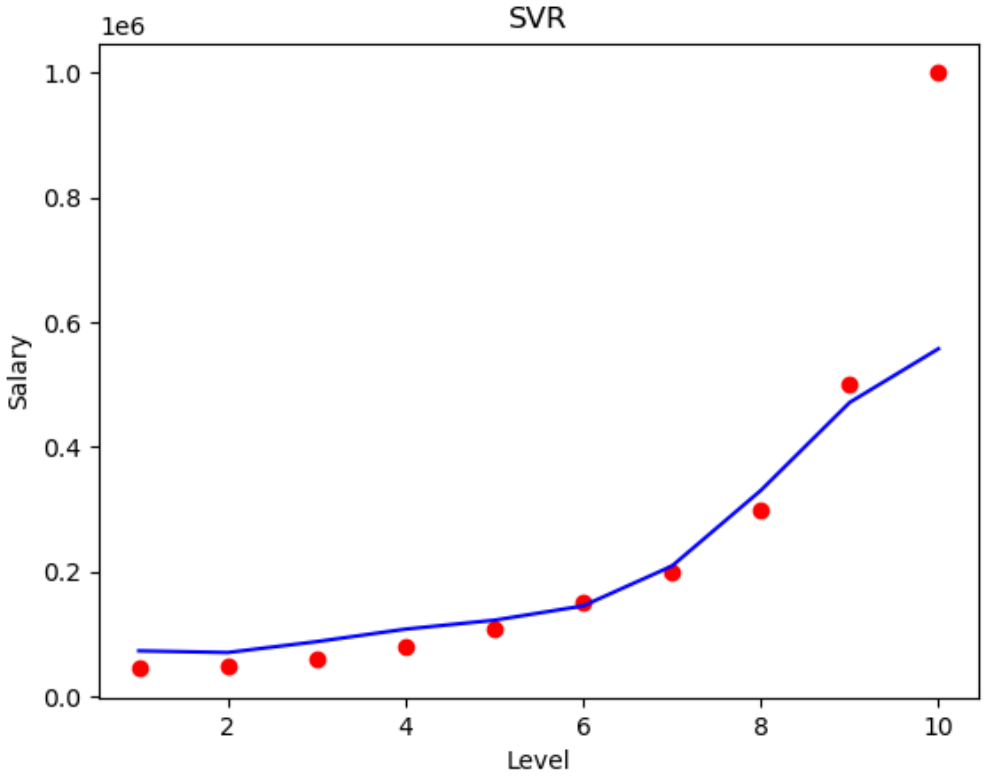
```
print(sc_y.inverse_transform(reg.predict(sc_X.transform([[6.5]])).reshape(-1,1)))
```

Result : 170370.0204065

3.7 Visualising the SVR results

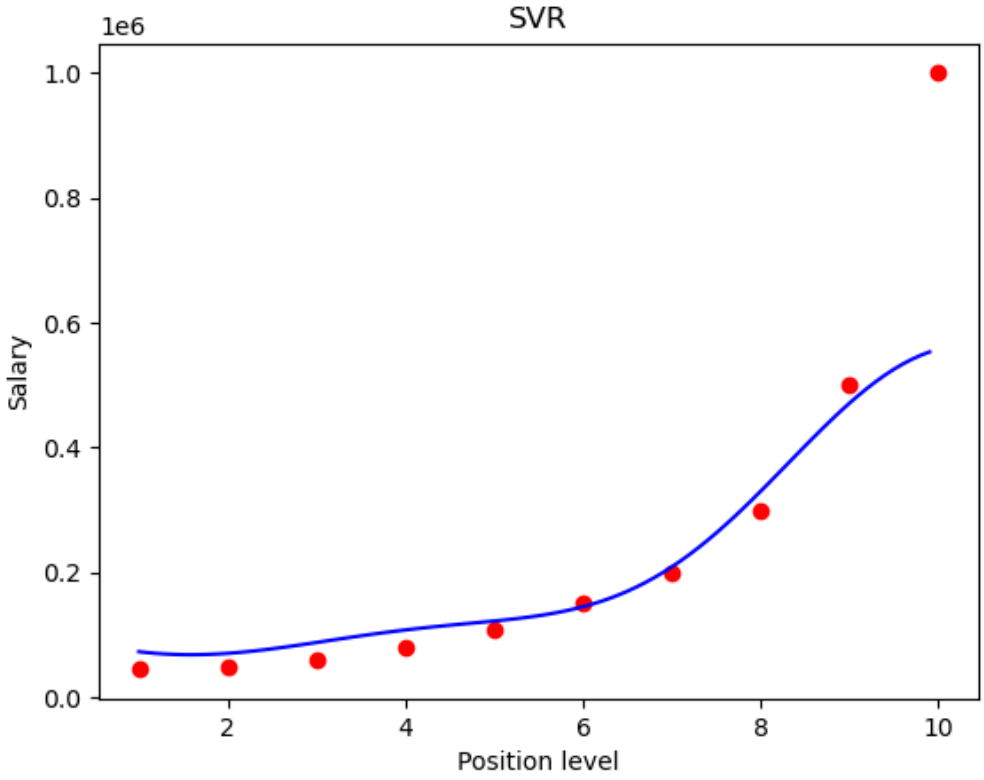
3.7.1 Visualising SVR results Low Resolution

```
plt.scatter(sc_X.inverse_transform(X),sc_y.inverse_transform(y),color = 'red')
plt.plot(sc_X.inverse_transform(X),sc_y.inverse_transform(reg.predict(X)
.reshape(-1,1)),color = 'blue')
plt.title('SVR')
plt.xlabel('Level')
plt.ylabel('Salary')
plt.show()
```



3.7.2 Visualising SVR results High Resolution

```
X_grid = np.arange(min(sc_X.inverse_transform(X)),
max(sc_X.inverse_transform(X)), 0.1)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(sc_X.inverse_transform(X), sc_y.inverse_transform(y),
color = 'red')
plt.plot(X_grid, sc_y.inverse_transform(reg.predict(sc_X.transform(X_grid))
.reshape(-1,1)), color = 'blue')
plt.title('SVR')
plt.xlabel('Position level')
plt.ylabel('Salary')
plt.show()
```



4 Decision Tree Regression

4.1 Data Set

Position	Level	Salary
Business Analyst	1	45000
Junior Consultant	2	50000
Senior Consultant	3	60000
Manager	4	80000
Country Manager	5	110000
Region Manager	6	150000
Partner	7	200000
Senior Partner	8	300000
C-level	9	500000
CEO	10	1000000

4.2 Importing Libraries

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

4.3 Importing Dataset

```
dataset = pd.read_csv('Position_Salaries.csv')
X = dataset.iloc[:, 1:-1].values
y = dataset.iloc[:, -1].values
```

4.4 Training the Decision Tree Regression model on the whole dataset

```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor(random_state=0)
model.fit(X,y)
```

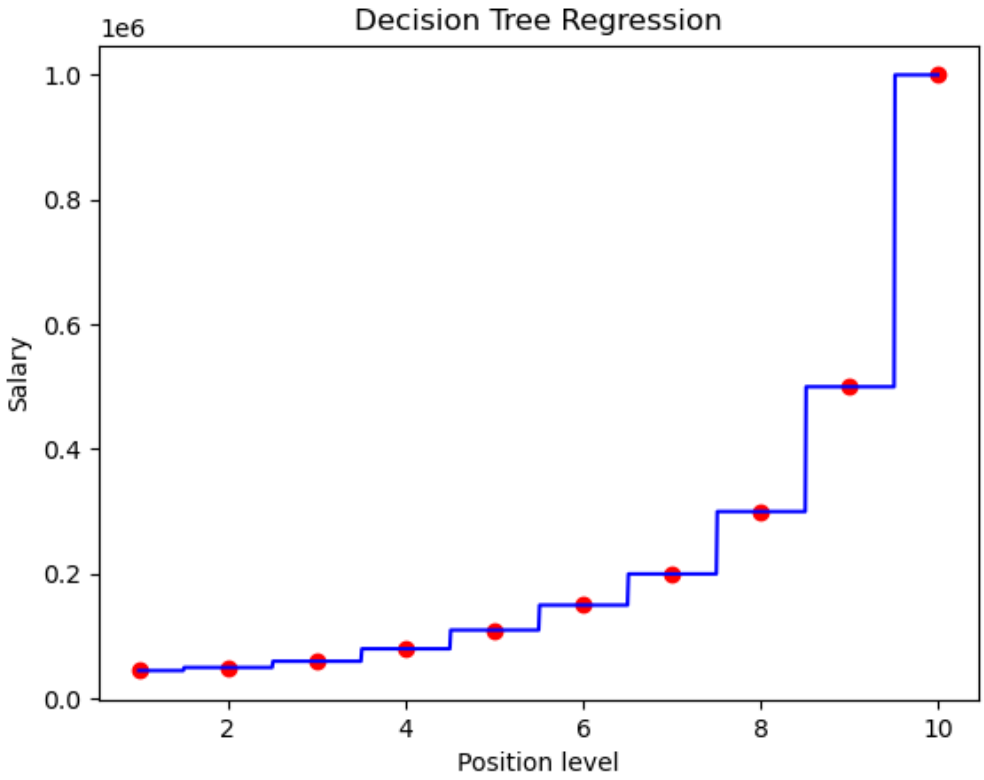
4.5 Predicting a new result

```
print(model.predict([[6.5]]))
```

Result : 150000

4.6 Visualising the Decision Tree Regression results (higher resolution)

```
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(X, y, color = 'red')
plt.plot(X_grid, model.predict(X_grid), color = 'blue')
plt.title('Decision Tree Regression')
plt.xlabel('Position level')
plt.ylabel('Salary')
plt.show()
```



5 Random Forest Regression

5.1 Data Set

Position	Level	Salary
Business Analyst	1	45000
Junior Consultant	2	50000
Senior Consultant	3	60000
Manager	4	80000
Country Manager	5	110000
Region Manager	6	150000
Partner	7	200000
Senior Partner	8	300000
C-level	9	500000
CEO	10	1000000

5.2 Importing Libraries

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

5.3 Importing Dataset

```
dataset = pd.read_csv('Position_Salaries.csv')
X = dataset.iloc[:, 1:-1].values
y = dataset.iloc[:, -1].values
```

5.4 Training the Random Forest Regression model on the whole dataset

```
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor(n_estimators = 10, random_state = 0)
model.fit(X, y)
```

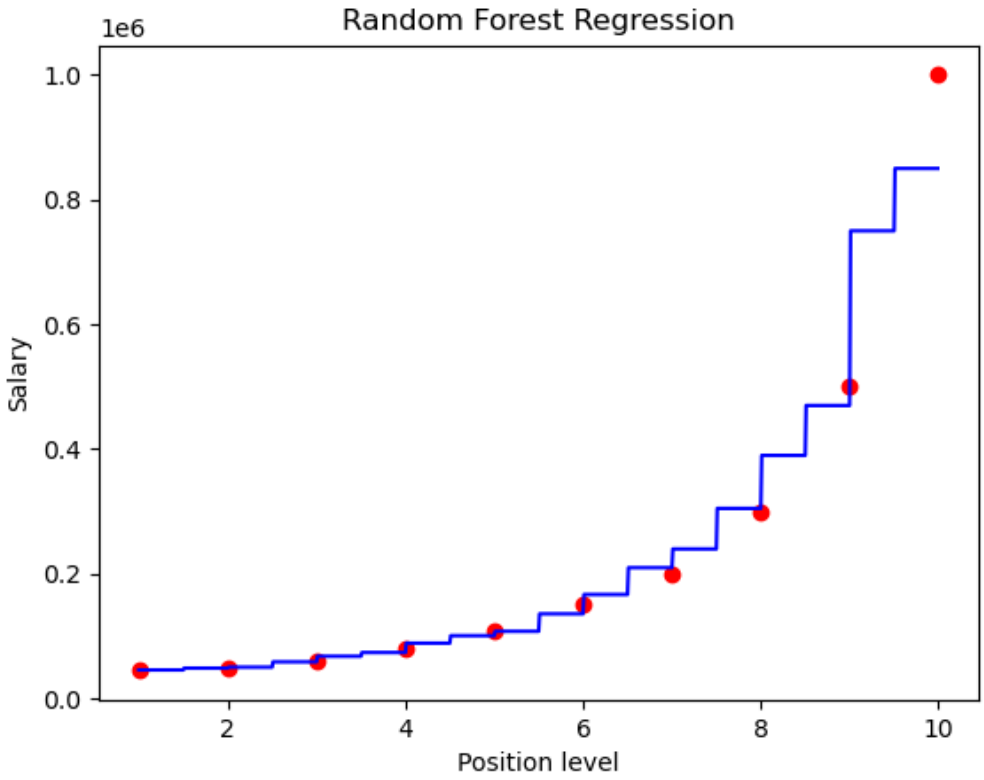
5.5 Predicting a new result

```
print(model.predict([[6.5]]))
```

Result : 167000

5.6 Visualising the Random Forest Regression results (higher resolution)

```
X_grid = np.arange(min(X), max(X), 0.01)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(X, y, color = 'red')
plt.plot(X_grid, model.predict(X_grid), color = 'blue')
plt.title('Random Forest Regression')
plt.xlabel('Position level')
plt.ylabel('Salary')
plt.show()
```



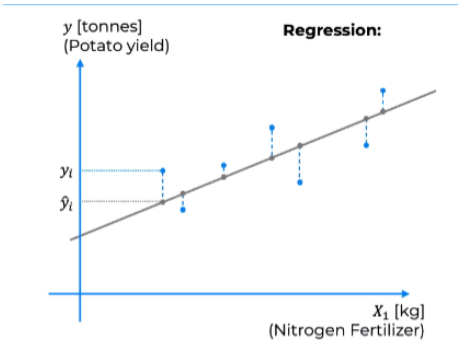
6 Evaluating Regression Models

R-squared, also known as the coefficient of determination, is a statistical measure used to evaluate the performance of a machine learning model. It is calculated by comparing the predicted values of the model to the actual values and determining the amount of variation in the dependent variable that can be explained by the independent variables.

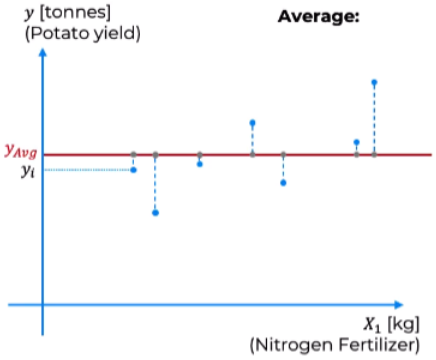
The R-squared value ranges from 0 to 1, with higher values indicating a better fit between the model and the data. A value of 1 means that the model perfectly predicts the dependent variable, while a value of 0 means that the model does not explain any of the variation in the dependent variable.

Intuitively, R-squared can be thought of as the proportion of the variance in the dependent variable that is explained by the independent variables. A high R-squared value indicates that the model is able to capture a large portion of the variation in the data, while a low R-squared value indicates that the model is not able to explain much of the variation.

R-squared is an important metric for evaluating machine learning models because it provides a measure of the model's accuracy and helps to determine whether the model is useful for making predictions. However, it should be used in conjunction with other metrics, such as mean squared error, to get a more complete picture of the model's performance.



$$SS_{res} = SUM(y_i - \hat{y}_i)^2$$



$$SS_{tot} = SUM(y_i - y_{avg})^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

Rule of thumb (for our tutorials)*:
1.0 = Perfect fit (suspicious)
~0.9 = Very good
<0.7 = Not great
<0.4 = Terrible
<0 = Model makes no sense for this data