

C:\VS Code\Spyder\Comparison of all regressions for emp salary.py

```
1 # ----- Comparing all Regression Models ----- #
2
3 #importing libraries
4
5 import numpy as np
6 import pandas as pd
7 import matplotlib.pyplot as plt
8 import pickle
9
10 #importing the dataset
11
12 dataset = pd.read_csv(r"C:\Users\Jan Saida\Downloads\emp_sal.csv")
13 x=dataset.iloc[:, 1:2].values #independent variable
14 y=dataset.iloc[:,2].values     #dependent variable
15
16 #Linear reg model -- linear algorithm (bydefault degree - 1)
17
18 from sklearn.linear_model import LinearRegression
19 lin_reg=LinearRegression()
20 lin_reg.fit(x,y)
21
22
23 #Linear regression Predictions
24
25 lin_model_pred=lin_reg.predict([[6.5]])
26 lin_model_pred
27
28 #Linear regression visualization
29
30 plt.scatter(x,y,color='red')
31 plt.plot(x,lin_reg.predict(x),color='blue')
32 plt.title('Linear Regression graph')
33 plt.xlabel('Position level')
34 plt.ylabel('Salary')
35
36 #polynomial regression model (bydefault degree - 2)
37
38 from sklearn.preprocessing import PolynomialFeatures
39 poly_reg=PolynomialFeatures(degree=6)
40 X_poly=poly_reg.fit_transform(x)
41
42 poly_reg.fit(X_poly,y)
43
44 lin_reg_2=LinearRegression()
45
46 lin_reg_2.fit(X_poly,y)
47
48 #Polynomial regression Predictions
49
50 poly_model_pred=lin_reg_2.predict(poly_reg.fit_transform([[6.5]]))
51 poly_model_pred
```

```

52
53 #polynomial visualisation
54
55 plt.scatter(x,y,color='red')
56 plt.plot(x,lin_reg_2.predict(poly_reg.fit_transform(x)),color='blue')
57 plt.title('Polynomial Regression graph')
58 plt.xlabel('Position level')
59 plt.ylabel('Salary')
60
61 #support vector regression model (degree=5)
62
63 from sklearn.svm import SVR
64 svr_reg=SVR(kernel='poly', gamma='scale', degree=5,C=1)
65 svr_reg.fit(x,y)
66
67 #Support vector regression Predictions
68
69 svr_reg_pred=svr_reg.predict([[6.5]])
70 svr_reg_pred
71
72
73 # Support Vector regressor Visualizations
74
75 plt.scatter(x,y,color='red')
76 plt.plot(x,svr_reg.predict(x),color='blue')
77 plt.title('Simple Vector Regression graph')
78 plt.xlabel('Position level')
79 plt.ylabel('Salary')
80
81 # knn regression model
82
83 from sklearn.neighbors import KNeighborsRegressor
84 knn_reg = KNeighborsRegressor(n_neighbors=4,weights='distance')
85 knn_reg.fit(x,y)
86
87 #knn regression Predictions
88
89 knn_reg_pred=knn_reg.predict([[6.5]])
90 knn_reg_pred
91
92 # Knn regressor Visualizations
93
94 plt.scatter(x,y,color='red')
95 plt.plot(x,knn_reg.predict(x),color='blue')
96 plt.title('Knn Regression graph')
97 plt.xlabel('Position level')
98 plt.ylabel('Salary')
99
100
101 #desicion tree algorithm
102
103 from sklearn.tree import DecisionTreeRegressor
104 tree_reg = DecisionTreeRegressor( splitter="best", random_state=0)
105 tree_reg.fit(x,y)

```

```

106
107 #decision tree Predictions
108
109 tree_reg_pred=tree_reg.predict([[6.5]])
110 tree_reg_pred
111
112 # Desicion Tree regressor Visualizations
113
114 plt.scatter(x,y,color='red')
115 plt.plot(x,tree_reg.predict(x),color='blue')
116 plt.title('Decision Tree Regression graph')
117 plt.xlabel('Position level')
118 plt.ylabel('Salary')
119
120 # Random foreest Regression model
121
122 from sklearn.ensemble import RandomForestRegressor
123 forest_regressor=RandomForestRegressor(n_estimators=15)
124 forest_regressor.fit(x,y)
125
126 # Random Forest regressor Predictions
127
128 forest_pred=forest_regressor.predict([[6.5]])
129 forest_pred
130
131 # Random Forest Regressor Visualizations
132
133 x_grid=np.arange(min(x),max(x),0.01)
134 x_grid=x_grid.reshape(len(x_grid),1)
135 plt.scatter(x,y,color='red')
136 plt.plot(x_grid,forest_regressor.predict(x_grid),color='blue')
137 plt.title('Random Forest Regression graph')
138 plt.xlabel('Position level')
139 plt.ylabel('Salary')
140 plt.show()
141
142 # Extreme Gradient Boosting Regressor Model
143
144 import xgboost as xg
145 xgb_r = xg.XGBRegressor(objective = 'reg:linear', n_estimators = 10, seed = 0)
146 xgb_r.fit(x,y)
147
148 # Extreme Gradient Boosting Model Predictions
149
150 xgb_reg_pred = xgb_r.predict([[6.5]])
151 xgb_reg_pred
152
153
154
155 # Pickle Linear Regression Model
156 with open('linear_regression_model.pkl', 'wb') as file:
157     pickle.dump(lin_reg, file)
158
159 # Pickle Polynomial Regression Model

```

```
160 with open('polynomial_regression_model.pkl', 'wb') as file:
161     pickle.dump(lin_reg_2, file)
162
163 # Pickle Support Vector Regression Model
164 with open('svr_model.pkl', 'wb') as file:
165     pickle.dump(svr_reg, file)
166
167 # Pickle KNN Regression Model
168 with open('knn_model.pkl', 'wb') as file:
169     pickle.dump(knn_reg, file)
170
171 # Pickle Decision Tree Model
172 with open('decision_tree_model.pkl', 'wb') as file:
173     pickle.dump(tree_reg, file)
174
175 # Pickle Random Forest Model
176 with open('random_forest_model.pkl', 'wb') as file:
177     pickle.dump(forest_regressor, file)
178
179 # Pickle Extreme Gradient Boosting Model
180 with open('xgboost_model.pkl', 'wb') as file:
181     pickle.dump(xgb_r, file)
182
183
```

Streamlit\Comparison_of_all_regs_emp_salary.py

```
1 import streamlit as st
2 import pickle
3 import numpy as np
4
5 # Load the pickled models
6 with open(r'C:\Users\Jan Saida\linear_regression_model.pkl', 'rb') as file:
7     lin_reg = pickle.load(file)
8
9 with open(r'C:\Users\Jan Saida\polynomial_regression_model.pkl', 'rb') as file:
10     lin_reg_2 = pickle.load(file)
11
12 with open(r'C:\Users\Jan Saida\svr_model.pkl', 'rb') as file:
13     svr_reg = pickle.load(file)
14
15 with open(r'C:\Users\Jan Saida\knn_model.pkl', 'rb') as file:
16     knn_reg = pickle.load(file)
17
18 with open(r'C:\Users\Jan Saida\decision_tree_model.pkl', 'rb') as file:
19     tree_reg = pickle.load(file)
20
21 with open(r'C:\Users\Jan Saida\random_forest_model.pkl', 'rb') as file:
22     forest_regressor = pickle.load(file)
23
24 with open(r'C:\Users\Jan Saida\xgboost_model.pkl', 'rb') as file:
25     xgb_r = pickle.load(file)
26
27 # Polynomial Features (this is the same across different models)
28 from sklearn.preprocessing import PolynomialFeatures
29 poly_reg = PolynomialFeatures(degree=6)
30
31 # Streamlit UI
32 st.title('Salary Prediction using Multiple Regression Models')
33
34 # Input for Position level
35 position_level = st.number_input('Enter the position level (e.g., 6.5):', min_value=1.0,
36                                   max_value=10.0, value=6.5)
37
38 # Polynomial Regression Prediction
39 position_level_poly = poly_reg.fit_transform([[position_level]]) # Apply transformation
40 poly_pred = lin_reg_2.predict(position_level_poly) # Predict using the transformed features
41
42 # Linear Regression Prediction
43 lin_pred = lin_reg.predict([[position_level]])
44
45 # Support Vector Regression Prediction
46 svr_pred = svr_reg.predict([[position_level]])
47
48 # KNN Regression Prediction
49 knn_pred = knn_reg.predict([[position_level, 0]])
50
51 # Decision Tree Prediction
52 tree_pred = tree_reg.predict([[position_level]])
```

```
52 |
53 | # Random Forest Prediction
54 | forest_pred = forest_regressor.predict([[position_level]])
55 |
56 | # XGBoost Prediction
57 | xgb_pred = xgb_r.predict([[position_level]])
58 |
59 | # Display the results
60 | st.subheader(f'Predicted Salary for Position Level {position_level}')
61 | st.write(f"Linear Regression Prediction: {lin_pred[0]}")
62 | st.write(f"Polynomial Regression Prediction: {poly_pred[0]}")
63 | st.write(f"Support Vector Regression Prediction: {svr_pred[0]}")
64 | st.write(f"KNN Regression Prediction: {knn_pred[0]}")
65 | st.write(f"Decision Tree Prediction: {tree_pred[0]}")
66 | st.write(f"Random Forest Prediction: {forest_pred[0]}")
67 | st.write(f"XGBoost Prediction: {xgb_pred[0]}")
68 |
```

Salary Prediction using Multiple Regression Models

Enter the position level (e.g., 6.5):

6.50

- +

Predicted Salary for Position Level 6.5

Linear Regression Prediction: 330378.78787878784

Polynomial Regression Prediction: 174192.81930661248

Support Vector Regression Prediction: 164079.01344549266

KNN Regression Prediction: 1

Decision Tree Prediction: 150000.0

Random Forest Prediction: 157333.33333333334

XGBoost Prediction: 158778.203125

Salary Prediction using Multiple Regression Models

Enter the position level (e.g., 6.5):

2.20

- +

Predicted Salary for Position Level 2.2

Linear Regression Prediction: -17400.0

Polynomial Regression Prediction: 51187.390522576505

Support Vector Regression Prediction: 100009.9473626854

KNN Regression Prediction: 1

Decision Tree Prediction: 50000.0

Random Forest Prediction: 55333.333333333336

XGBoost Prediction: 68077.109375