**Machine Learning – Regression**

1. **Simple Linear Regression** 
   1. Dataset
      1. SalaryData.csv
   2. One Input
      1. YearsExperience
   3. Output
      1. Salary Prediction

|  |  |  |  |
| --- | --- | --- | --- |
| YearsExperience | weight | bias | Dependent Y |
| 0 | 9360 | 26777 | 26777 |
| 15 | 9360 | 26777 | 167177 |
|  |  |  |  |

r2\_score = 0.974

1. **Multiple Linear Regression**
   1. Dataset – 50\_Salary.csv
   2. One or more Inputs
      1. R & D Spend
      2. Administration
      3. Marketing Spend
      4. State
         1. State\_Florida
         2. State\_NewYork
   3. Output
      1. Profit Prediction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R&D Spend  (X) | Administration  (X) | Marketing Spend  (X) | State\_Florida  (X) | State\_Newyork  (X) | Dependent  (Y)  Profit Prediction |
| 50000 | 50000 | 50000 | 0 | 1 | 85311.26684297 |
| 100000 | 100000 | 100000 | 1 | 0 | 128072.09266009 |

For first Input data,

weight - 7.90840255e-01, 3.01968165e-02, 3.10148566e-02, 4.63028992e+02, 3.04799573e+02

bias - 42403.87087053

r2\_score = 0.935

1. **Support Vector Machine** 
   1. Dataset – 50\_Salary.csv
   2. One or more Inputs
      1. R & D Spend
      2. Administration
      3. Marketing Spend
      4. State
         1. State\_Florida
         2. State\_NewYork
   3. Output
      1. Profit Prediction

|  |  |  |
| --- | --- | --- |
| Support Vector Machine | Standardization | r2\_score Accuracy |
| SVM kernel=’rbf’ | With Standardization | -0.057 |
| SVM kernel=’poly’ | With Standardization | -0.057 |
| SVM kernel=’sigmoid’ | With Standardization | -0.057 |
| **SVM kernel=’linear’** | **Without standardization** | **0.895** |

SVM kernel=’linear’ without standardization gives the best accuracy –r2\_score=0.895

* Bias and support vectors are calculated in SVM.
* Sklearn.svm.SVC
* Hyper tuning parameter
* **kernel*{‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’} or callable, default=’rbf’***
* **C*float, default=1.0 Penalty***

1. **Decision Tree** 
   1. Dataset – 50\_Salary.csv
   2. One or more Inputs
      1. R & D Spend
      2. Administration
      3. Marketing Spend
      4. State
         1. State\_Florida
         2. State\_NewYork
   3. Output
      1. Profit Prediction

|  |  |  |
| --- | --- | --- |
| criterion | splitter | r2\_score |
| criterion=’absolute\_error’ |  | 0.911 |
| criterion=’squared\_error’ |  | 0.935 |
| **criterion=’friedman\_mse’** |  | **0.961** |
| criterion=’poisson’ |  | 0.920 |
|  |  |  |
| criterion=’absolute\_error’ | splitter=’best’ | 0.904 |
| criterion=’squared\_error’ | splitter=’best’ | 0.924 |
| **criterion=’friedman\_mse’** | **splitter=’best’** | **0.960** |
| criterion=’poisson’ | splitter=’best’ | 0.910 |
|  |  |  |
| criterion=’absolute\_error’ | splitter=’random’ | 0.73 |
| criterion=’squared\_error’ | splitter=’random’ | 0.84 |
| criterion=’friedman\_mse’ | splitter=’random’ | 0.915 |
| criterion=’poisson’ | splitter=’random’ | 0.935 |

Best hyper tuning parameter – absolute\_error – r2\_score - 0.96

* Graphs are generated in decision tree using matplotlib
* sklearn.tree.DecisionTreeRegressor
* hyper tuning parameter
* **criterion*{“squared\_error”, “friedman\_mse”, “absolute\_error”, “poisson”}, default=”squared\_error”***
* **splitter*{“best”, “random”}, default=”best”***

1. **Random Forest**
   1. Dataset – 50\_Salary.csv
   2. One or more Inputs
      1. R & D Spend
      2. Administration
      3. Marketing Spend
      4. State
         1. State\_Florida
         2. State\_NewYork
   3. Output
      1. Profit Prediction

|  |  |  |  |
| --- | --- | --- | --- |
| n\_estimators | random\_state | criterion | r2\_score |
| RF without hyper tuning parameter |  |  | 0.930 |
| n\_estimators:100 |  |  | 0.933 |
| n\_estimators:50 |  |  | 0.929 |
| n\_estimators:200 |  |  | 0.934 |
| **n\_estimators:100** | **random\_state=0** |  | **0.946** |
| n\_estimators:50 | random\_state=0 |  | 0.944 |
| n\_estimators:200 | random\_state=0 |  | 0.943 |
| n\_estimators:100 | random\_state=0 | criterion=’absolute\_error’ | 0.945 |
| **n\_estimators:100** | **random\_state=0** | **criterion=’squared\_error’** | **0.946** |
| n\_estimators:100 | random\_state=0 | criterion=’friedman\_mse’ | 0.941 |
| n\_estimators:100 | random\_state=0 | criterion=’poisson’ | 0.941 |

* Best hyper tuning parameter

n\_estimators=100, random\_state=0, criterion=’squared\_error’ 🡪 r2\_score =0.946

* Sklearn.ensemble.RandomForestRegressor
* Hyper tuning parameter
* **n\_estimators: *int, default=100* -** The number of trees in the forest.
* **random\_state: *int, RandomState instance or None, default=None***
* **criterion*{“squared\_error”, “absolute\_error”, “friedman\_mse”, “poisson”}, default=”squared\_error”***

|  |  |  |
| --- | --- | --- |
| S.No | Algorithm | r2\_score - Accuracy |
| 1 | Multiple Linear Regression | 0.93 |
| 2 | Support Vector Machine | 0.89 |
| 3 | **Decision Tree** | **0.96** |
| 4 | **Random Forest** | **0.946** |

1. **Adaptive Boosting – AdaBoost** 
   1. Dataset – 50\_Salary.csv
   2. One or more Inputs
      1. R & D Spend
      2. Administration
      3. Marketing Spend
      4. State
         1. State\_Florida
         2. State\_NewYork
   3. Output
      1. Profit Prediction

|  |  |  |
| --- | --- | --- |
| AdaBoost | Hyper tuning parameter | r2\_score |
| AdaBoost() |  | 0.908 |
| **AdaBootst** | **(n\_estimators=100,random\_state=0)** | **0.926** |

* Best hyper tuning parameter
* *class*sklearn.ensemble.**AdaBoostRegressor**(*estimator=None*, *\**, *n\_estimators=50*, *learning\_rate=1.0*, *loss='linear'*, *random\_state=None*, *base\_estimator='deprecated'*)
* sklearn.ensemble.AdaBoostRegressor

1. **Extreme Gradient Boosting - XGBoost**
   1. Dataset – 50\_Salary.csv
   2. One or more Inputs
      1. R & D Spend
      2. Administration
      3. Marketing Spend
      4. State
         1. State\_Florida
         2. State\_NewYork
   3. Output
      1. Profit Prediction

|  |  |  |
| --- | --- | --- |
| XGBoost | Hyper tuning parameter | r2\_score |
| XGBoost() |  | 0.894 |
| **XGBoost** | **(n\_estimators=100,random\_state=0)** | **0.894** |

1. **Light Gradient Boosting - LGBoost**
   1. Dataset – 50\_Salary.csv
   2. One or more Inputs
      1. R & D Spend
      2. Administration
      3. Marketing Spend
      4. State
         1. State\_Florida
         2. State\_NewYork
   3. Output
      1. Profit Prediction

|  |  |  |
| --- | --- | --- |
| LGBoost | Hyper tuning parameter | r2\_score |
| LGBoost() |  | -0.0367 |
| **LGBoost** | **(n\_estimators=100,random\_state=0)** | **-0.0367** |

https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMRegressor.html