# Regression-BoostedTree

April 9, 2022

[2]:
```
import pandas as pd

data = pd.read_csv('C:\\Users\\MSI␣
 ↪Stealth\\Downloads\\BMEN415Project\\regression\\Volumetric_features.csv')

print(data.head())
```

```
   S.No  Left-Lateral-Ventricle  Left-Inf-Lat-Vent  \
0     1                 22916.9              982.7
1     2                 22953.2              984.5
2     3                 23320.4             1062.1
3     4                 24360.0             1000.5
4     5                 25769.4             1124.4

   Left-Cerebellum-White-Matter  Left-Cerebellum-Cortex  Left-Thalamus  \
0                       15196.7                 55796.4         6855.5
1                       15289.7                 55778.6         6835.1
2                       15382.1                 55551.2         7566.0
3                       14805.4                 54041.8         8004.6
4                       16331.1                 54108.6         6677.4

   Left-Caudate  Left-Putamen  Left-Pallidum  3rd-Ventricle  …  \
0        2956.4        4240.7         2223.9         2034.4  …
1        3064.2        4498.6         2354.1         1927.1  …
2        3231.7        4456.2         1995.4         2064.7  …
3        3137.3        4262.2         1983.4         2017.7  …
4        2964.4        4204.6         2409.7         2251.8  …

   rh_supramarginal_thickness  rh_frontalpole_thickness  \
0                       2.408                     2.629
1                       2.417                     2.640
2                       2.374                     2.601
3                       2.366                     2.639
4                       2.381                     2.555

   rh_temporalpole_thickness  rh_transversetemporal_thickness  \
0                      3.519                            2.009
1                      3.488                            2.111
```

```
2                     3.342                           2.146
3                     3.361                           2.056
4                     3.450                           2.052

   rh_insula_thickness  rh_MeanThickness_thickness  BrainSegVolNotVent.2  \
0                2.825                     2.33635                1093846
1                2.720                     2.34202                1099876
2                2.684                     2.31982                1097999
3                2.700                     2.29215                1070117
4                2.574                     2.30397                1075926

        eTIV.1  Age  dataset
0  1619602.965   85        1
1  1624755.130   85        1
2  1622609.518   86        1
3  1583854.236   87        1
4  1617375.362   89        1

[5 rows x 141 columns]
```

[3]:
```
# Separate Target Variable and Predictor Variables
y=data['Age'].values
X=data[['Left-Lateral-Ventricle', 'Left-Inf-Lat-Vent',
            'Left-Cerebellum-White-Matter', 'Left-Cerebellum-Cortex',
            'Left-Thalamus', 'Left-Caudate', 'Left-Putamen',
           'Left-Pallidum','3rd-Ventricle','4th-Ventricle',
           'Brain-Stem','Left-Hippocampus','Left-Amygdala',
           'CSF','Left-Accumbens-area','Left-VentralDC',
           'Left-vessel','Left-choroid-plexus','Right-Lateral-Ventricle',
          ␣
↪'Right-Inf-Lat-Vent','Right-Cerebellum-White-Matter','Right-Cerebellum-Cortex',
           'Right-Thalamus','Right-Caudate','Right-Putamen',
          ␣
↪'Right-Pallidum','Right-Hippocampus','Right-Amygdala','Right-Accumbens-area',
           'Right-VentralDC','Right-vessel','Right-choroid-plexus',
           '5th-Ventricle','WM-hypointensities','Left-WM-hypointensities',
          ␣
↪'Right-WM-hypointensities','non-WM-hypointensities','Left-non-WM-hypointensities',
          ␣
↪'Right-non-WM-hypointensities','Optic-Chiasm','CC_Posterior','CC_Mid_Posterior',
          ␣
↪'CC_Central','CC_Mid_Anterior','CC_Anterior','BrainSegVol','BrainSegVolNotVent']].
↪values
```

[4]:
```
train = data[:4000]
test = data[4000:]
X_test = X[4000:]
```

```
y_test = y[4000:]
X_train = X[:4000]
y_train = y[:4000]
```

[7]:
```
from sklearn import datasets
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.decomposition import PCA
from sklearn.metrics import mean_squared_error
import math
```

[11]:
```
# Standardize the dataset
#
sc = StandardScaler()
X_train_std = sc.fit_transform(train)
X_test_std = sc.transform(test)
#
# Hyperparameters for GradientBoostingRegressor
#
gbr_params = {'n_estimators': 20,
          'max_depth': 2,
          'min_samples_split': 2,
          'learning_rate': 0.1,
          'loss': 'ls'}
#
# Create an instance of gradient boosting regressor
#
gbr = GradientBoostingRegressor(**gbr_params)
#
# Fit the model
#
gbr.fit(X_train_std, y_train)

predictions = gbr.predict(X_test_std)
total_cases = len(y_test) # size of validation set

for i in range(total_cases):
    value = y_test[i]
    predict = predictions[i]
    #print(value, '--------- ' , predict)

#
# Print Coefficient of determination R^2
#
print("Model Accuracy: %.3f" % gbr.score(X_test_std, y_test))
```

```
#
# Create the mean squared error
#
mse = mean_squared_error(y_test, gbr.predict(X_test_std))
rmse = math.sqrt(mse)
print("The rooted mean squared error (MSE) on test set:{:.4f}", rmse)
```

```
Model Accuracy: 0.859
The rooted mean squared error (MSE) on test set:{:.4f} 2.609012118675888
```

[ ]: