7_Life_Expectancy_Modeling

January 29, 2023

Table of Contents

1 Modeling

1.1 NOTE 1:

1 Life_Expectancy_WHO_UN_Analysis_Modeling

1.1 Modeling

To: Magnimind

From: Matt Curcio, matt.curcio.ri@gmail.com

Date: 2023-01-29

Re: NOTEBOOK #6

---

[1]:
```
# Common Python Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

# import warnings
import warnings
warnings.filterwarnings("ignore")

# Loading Regression and Modeling Libaries
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.linear_model import SGDRegressor
```

[2]:
```
!ls *.csv
```

```
Clean_LE_Data_FEng_4.csv      Life_Expectancy_Data.csv  y_test.csv
Clean_LE_Data_Post_EDA_3.csv  x_test.csv                y_train.csv
Clean_LE_Data_w_Means_2.csv   x_train.csv
```

1.1.1 NOTE 1:

- **Model # 1 - Use: Income, Education, HIV, DTP, Polio, lt5yD, AdultMort**

[3]:
```
# Load X-Train
df = pd.read_csv('x_train.csv', header=0)

# Obtain predictors from Notebook #5:
# 5_Life_Expectancy_Recursive_Feature_Elimination.ipynb

predictors = ['Income', 'Education', 'HIV', 'DTP', 'Polio', 'lt5yD',␣
 ↪'AdultMort']

x_train = df[predictors]

x_train.head()
```

[3]:
```
   Income  Education   HIV   DTP  Polio  lt5yD  AdultMort
0   0.782       15.0   0.1  98.0   98.0      6       28.0
1   0.835       13.7   0.1  96.0   95.0      0       83.0
2   0.897       16.6   0.1  93.0   93.0      0       92.0
3   0.673       13.6   0.1  97.0   97.0      4       15.0
4   0.567       11.8  31.9  96.0   96.0      4      693.0
```

[4]:
```
# Load Y-Train
y_train = pd.read_csv("y_train.csv", header=0)

y_train.head()
```

[4]:
```
   LifeExpectancy
0            69.5
1            76.6
2            78.1
3            74.0
4            46.4
```

[5]:
```
# Load training and testing datasets
x_train = pd.read_csv("x_train.csv", header=0)
x_test = pd.read_csv("x_test.csv", header=0)

y_train = pd.read_csv("y_train.csv", header=0)
y_test = pd.read_csv("y_test.csv", header=0)
```

1.2 Four algorythms were tested accuracy alone was used as a benchmark:

Model	Average % Accuracy (cv=5)
Gradient Boosting Regressor	94.7
Decision Tree Regressor	88.5
Linear Regression	81.0
Support Vector Regressor	19.0

[6]:
```
# Linear Regression

lm = LinearRegression()
cv = cross_val_score(lm,x_train,y_train,cv=5)
print(cv)
print(cv.mean())
```

```
[0.77353212 0.81239152 0.82021745 0.83378051 0.8123496 ]
0.8104542396676632
```

[7]:
```
# Decision Tree Regressor

dt = DecisionTreeRegressor(max_depth=5)
cv = cross_val_score(dt,x_train,y_train,cv=5)
print(cv)
print(cv.mean())
```

```
[0.89449861 0.86730021 0.88371289 0.89929528 0.88145511]
0.8852524181702497
```

[8]:
```
# Support Vector Regressor

regr = SVR()
cv = cross_val_score(regr,x_train,y_train,cv=5)
print(cv)
print(cv.mean())
```

```
[0.24553878 0.16638795 0.22588448 0.12750496 0.18453894]
0.1899710214514779
```

[9]:
```
# CatBoost Regressor

model = GradientBoostingRegressor(random_state=100)
cv = cross_val_score(model,x_train,y_train,cv=5)
print(cv)
print(cv.mean())
```

```
[0.93777862 0.94625967 0.94826961 0.95304168 0.94819837]
0.9467095920842518
```