dt-regression

April 5, 2024

0.1 Predicting Concrete Compressive Strength

[2]:
```
## importing necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.tree  import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

[22]:
```
## Loading dataset
data = pd.read_csv(r'C:\Users\ntpc\Desktop\Slump.csv',sep= '\t')
data.head()
```

[22]:

	Cement	Slag	Fly ash	Water	SP	Coarse Aggr.	Fine Aggr.	SLUMP(cm)	\
0	273.0	82.0	105.0	210.0	9.0	904.0	680.0	23.0	
1	163.0	149.0	191.0	180.0	12.0	843.0	746.0	0.0	
2	162.0	148.0	191.0	179.0	16.0	840.0	743.0	1.0	
3	162.0	148.0	190.0	179.0	19.0	838.0	741.0	3.0	
4	154.0	112.0	144.0	220.0	10.0	923.0	658.0	20.0	

	FLOW(cm)	Compressive Strength (28-day)(Mpa)
0	62.0	34.99
1	20.0	41.14
2	20.0	41.81
3	21.5	42.08
4	64.0	26.82

[23]:
```
#checking for categorical variables
data.dtypes
```

[23]:
```
Cement                                float64
Slag                                  float64
Fly ash                               float64
Water                                 float64
SP                                    float64
Coarse Aggr.                          float64
Fine Aggr.                            float64
SLUMP(cm)                             float64
```

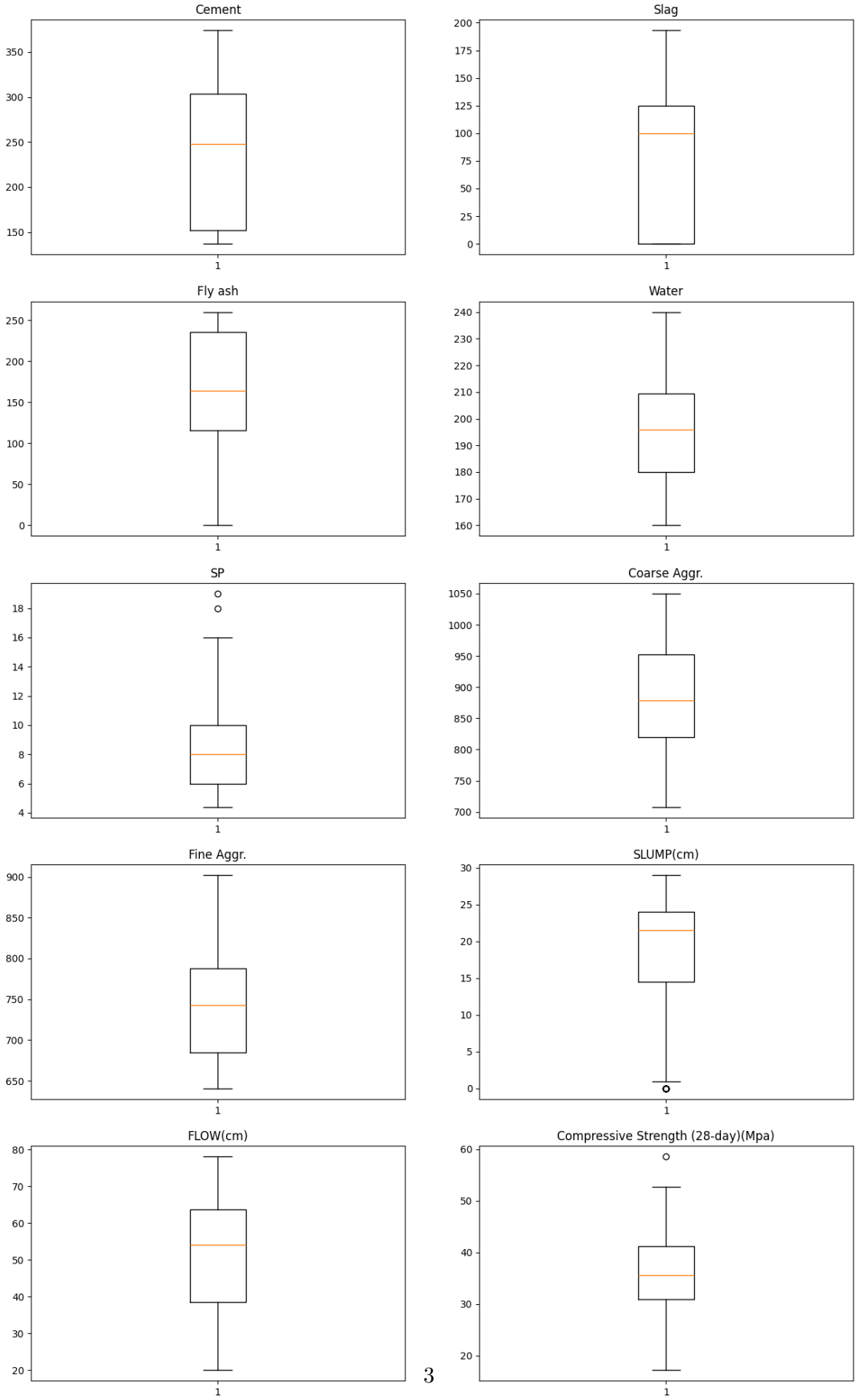
```
FLOW(cm)                                float64
Compressive Strength (28-day)(Mpa)      float64
dtype: object
```

[24]:
```
#checking for missing values
data.isnull().sum()
```

[24]:
```
Cement                                  0
Slag                                    0
Fly ash                                 0
Water                                   0
SP                                      0
Coarse Aggr.                            0
Fine Aggr.                              0
SLUMP(cm)                               0
FLOW(cm)                                0
Compressive Strength (28-day)(Mpa)      0
dtype: int64
```

[25]:
```
import matplotlib.pyplot as plt
#checking for outliers
plt.figure(figsize = (15,25))
count = 1
for col in data:
    plt.subplot(5,2,count)
    plt.boxplot(data[col])
    plt.title(col)
    count +=1

plt.show()
```


Cement
Slag
Fly ash
Water
SP
Coarse Aggr.
Fine Aggr.
SLUMP(cm)
FLOW(cm)
Compressive Strength (28-day)(Mpa)

[26]:
```
# Splitting the data into features (X) and target variable (y)
X = data.drop(['Compressive Strength (28-day)(Mpa)'], axis=1)
y = data['Compressive Strength (28-day)(Mpa)']
```

[27]:
```
# Performing feature scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

[28]:
```
# Splitting the scaled data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,␣
↪random_state=42)
```

[29]:
```
dt_regressor = DecisionTreeRegressor(random_state = 0)
dt_regressor .fit(X_train,y_train)
y_pred = dt_regressor .predict(X_test)
```

[30]:
```
r2_score(y_test,y_pred)
```

[30]:
```
0.4944091917964002
```

[31]:
```
from sklearn.model_selection import GridSearchCV
param_grid = {'max_depth': [None, 5, 10, 15, 20],
              'min_samples_split': [2, 5, 10],
              'min_samples_leaf': [1, 2, 4]}
grid_search = GridSearchCV(dt_regressor, param_grid, cv=5)
grid_search.fit(X_train, y_train)
```

[31]:
```
GridSearchCV(cv=5, estimator=DecisionTreeRegressor(random_state=0),
             param_grid={'max_depth': [None, 5, 10, 15, 20],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10]})
```

[32]:
```
best_model = GridSearchCV(estimator,param_grid,cv=5)
```

[33]:
```
best_model.fit(X_train,y_train)
```

[33]:
```
GridSearchCV(cv=5, estimator=DecisionTreeRegressor(random_state=0),
             param_grid={'max_depth': [None, 5, 10, 15, 20],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10]})
```

[34]:
```
best_params = grid_search.best_params_
```

[35]:
```
best_dt_regressor = DecisionTreeRegressor(**best_params)
best_dt_regressor.fit(X_train, y_train)
```

[35]:
```
DecisionTreeRegressor(min_samples_leaf=2)
```

[36]:
```
y_pred = best_dt_regressor.predict(X_test)
```

[37]:
```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Best Hyperparameters: {best_params}')
print(f'Mean Squared Error (MSE): {mse}')
print(f'R-squared (R^2): {r2}')
```

```
Best Hyperparameters: {'max_depth': None, 'min_samples_leaf': 2,
'min_samples_split': 2}
Mean Squared Error (MSE): 19.718233730158726
R-squared (R^2): 0.610290192541469
```