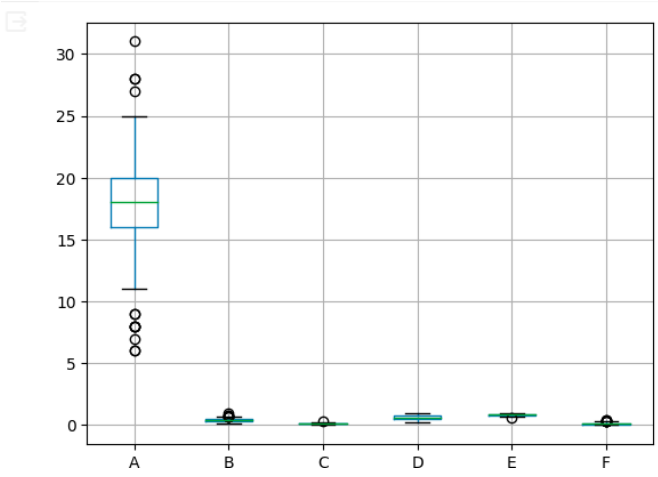
In Lab Tasks

Task 1

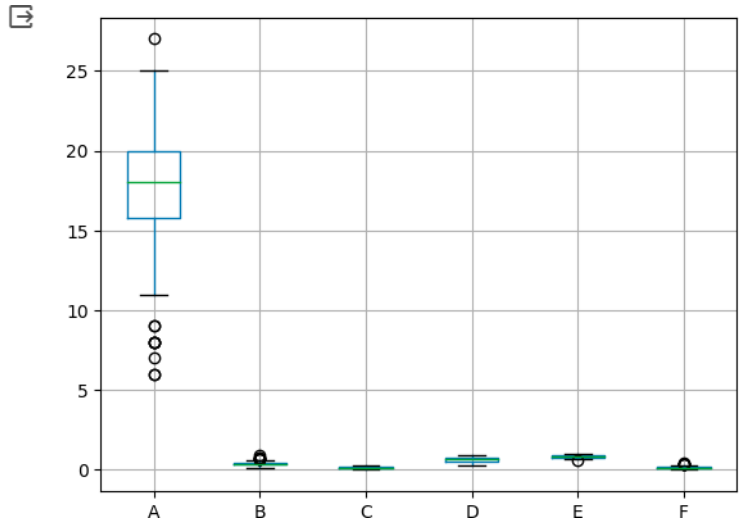
```
from keras.models import Sequential
from keras.layers import Dense, Dropout
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import numpy as np
from sklearn import linear_model
from sklearn import preprocessing
from sklearn import tree
from sklearn.ensemble import RandomForestRegressor,
GradientBoostingRegressor
import pandas as pd
import csv
import matplotlib.pyplot as plt

np.random.seed(7)
df = pd.read_csv("Alumni Giving Regression (Edited) (1).csv",
delimiter=",")
dd_df_1=df.head()

print(dd_df_1)
```

```
    A     B     C     D     E     F
0  24  0.42  0.16  0.59  0.81  0.08
1  19  0.49  0.04  0.37  0.69  0.11
2  18  0.24  0.17  0.66  0.87  0.31
3   8  0.74  0.00  0.81  0.88  0.11
4   8  0.95  0.00  0.86  0.92  0.28
```

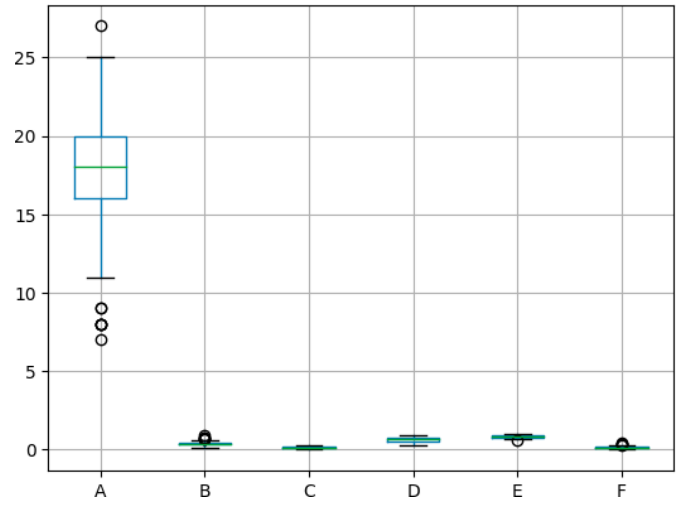
```
import seaborn as sns
import pandas as pd
boxplot = pd.DataFrame(df).boxplot()
```



Task 2




```
quantile1 = df.iloc[:, 0].quantile(0.01)
quantile99 = df.iloc[:, 0].quantile(0.99)
df2 = df[(df.iloc[:, 0] > quantile1) & (df.iloc[:, 0] <
quantile99)]
df2.boxplot()
plt.show()
```




```
df.dropna(inplace=True)
```

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split

#'A' is the target variable
X = df.drop('A', axis=1)
y = df['A']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

model3 = RandomForestRegressor()
model3.fit(X_train, y_train)

importances = model3.feature_importances_
std = np.std([tree.feature_importances_ for tree in
model3.estimators_], axis=0)
indices = np.argsort(importances)[::-1]

print("Feature ranking:")
for f in range(X.shape[1]):
    print("%d. feature (Column index) %s (%f)" % (f + 1,
indices[f], importances[indices[f]]))
```

```
Feature ranking:
1. feature (Column index) 0 (0.523148)
2. feature (Column index) 2 (0.136422)
3. feature (Column index) 3 (0.128585)
4. feature (Column index) 1 (0.128034)
5. feature (Column index) 4 (0.083811)
```

Task 3

```
indices_top3= indices[:3]
print(indices_top3)
dataset=df
df = pd.DataFrame(df)
Y_position = 5
TOP_N_FEATURE = 3
X = dataset.iloc[:, indices_top3]
Y = dataset.iloc[:,Y_position]

X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.20, random_state=2020)
#Model 1 linear regression
model1 = linear_model.LinearRegression()
model1.fit(X_train, y_train)

y_pred_train1 = model1.predict(X_train)
RMSE_train1 = mean_squared_error(y_train, y_pred_train1)
print("Regression TrainSet: RMSE {}".format(RMSE_train1))
```

```
y_pred1 = model1.predict(X_test)
RMSE_test1 = mean_squared_error(y_test,y_pred1)
print("Regression Testset: RMSE {}".format(RMSE_test1))
```

```
[0 2 1]
Regression TrainSet: RMSE 0.003698847883733275
Regression Testset: RMSE 0.005388812554401423
```