Objective:

1. Fitting KNN model to Treasury Data Set
2. Predicting with diffrent value of K
3. Fitting Decision Tree model to Treasury Data Set
4. Predicting with diffrent value of Depth
5. Determine the best Model by measuring performance of KNN and Decision Tree

In [1]:
```
#Import Libraries
import csv
import numpy as np
from numpy.random import randn
import pandas as pd
from pandas import Series, DataFrame
import matplotlib.pyplot as plt
import os
import math
import re
pd.options.display.max_columns=40
#Import KNeighborsClassifier from sklearn.neighbors
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
### Import Descision Tree Classifier
from sklearn.tree import DecisionTreeClassifier
```

In [2]:
```
### Load
path_hw2="C:\\Users\\farbo\\IE 598 Machine Learning\\HW\\HW 2\\"
df_ts=pd.read_csv(path_hw2+"Treasury Squeeze test - DS1.csv").drop(["rowindex","contract"],axis=1)
```

In [3]:
```
### Missing Values
df_ts.isnull().sum().sum()==0
```

Out[3]: True

In [4]:
```
X=df_ts.drop("squeeze",axis=1).values
#y=df_ts["squeeze"].values
y=df_ts["squeeze"].astype(int).values
```

In [5]:
```
### Test and Train
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = .2, random_state=42, stratify=y)
```

Fit KNN for different values of K (Neighbours)

In [6]:
```
list_of_models_score=[]
list_of_y_pred=[]
list_of_neighbors=[]
for i in range(1,22):
    ### Create the Model
    knn = KNeighborsClassifier(n_neighbors=i)
    ### Fit values to the model
    knn.fit(X_train,y_train)
    ### Prediction
    y_pred=knn.predict(X_test)
    y_pred=list(y_pred)
    ### Store the prediction of each model
    list_of_y_pred.append(y_pred)
    #### Evaluate the score of each model
    knn_model_score=knn.score(X_test,y_test)
    ### Store the Scores of each model
    list_of_models_score.append(knn_model_score)
    ### Store Neghbour Number
    list_of_neighbors.append(i)
```

In [7]:
```
### Store the Performance of all models fitted
df_performance=pd.DataFrame({"Neighbours":list_of_neighbors,
             "KNN Model Accuracy%":list_of_models_score})
df_performance["KNN Model Accuracy%"]=round(df_performance["KNN Model Accuracy%"]*100,2)
df_performance.sort_values("KNN Model Accuracy%",ascending=False).head(7)
```

Out[7]:

	Neighbours	KNN Model Accuracy%
19	20	65.56
20	21	63.33
15	16	62.22
4	5	61.11
16	17	61.11
5	6	60.56
14	15	60.00

In [8]:
```
### Load the Predictions
df_prediction=pd.DataFrame(list_of_y_pred).T
### Add the neighbors number to the columns
df_prediction.columns=["K"+str(c) for c in list_of_neighbors]
### Merge Prediction with Y_test
df_prediction=pd.merge(df_prediction.reset_index(),
         pd.DataFrame({"Y Test":y_test}).reset_index(),
         left_on="index",right_on="index",how="left").drop('index',axis=1)
df_prediction.head()
```

Out[8]:

	K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12	K13	K14	K15	K16	K17	K18	K19	K20	K21	Y Test
0	1	0	1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	0	0
3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

In [9]:
```
best_k=df_performance.sort_values("KNN Model Accuracy%",ascending=False)["Neighbours"].values[0]
best_k_accuracy=round(df_performance.sort_values("KNN Model Accuracy%",ascending=False)["KNN Model Accuracy%"].values[
```

In [10]:
```
print("The most accurate KNN model is the one with " + str(best_k) + " neighbours with " + str(best_k_accuracy)+"% acc
```

```
The most accurate KNN model is the one with 20 neighbours with 65.56% accuracy
```

Decision Tree

In [11]:
```
list_of_models_score_dt=[]
list_of_y_pred_dt=[]
list_of_depth_dt=[]
for i in range(1,22):
    ### Create the Model
    Decision_Tree=DecisionTreeClassifier(max_depth=i,random_state=1)
    ### Fit values to the model
    Decision_Tree.fit(X_train,y_train)
    ### Prediction
    y_pred_dt=Decision_Tree.predict(X_test)
    ### Store the prediction of each model
    list_of_y_pred_dt.append(y_pred_dt)
    #### Evaluate the score of each model
    dt_model_score=Decision_Tree.score(X_test,y_test)
    ### Store the Scores of each model
    list_of_models_score_dt.append(dt_model_score)
    ### Store Neghbour Number
    list_of_depth_dt.append(i)
```

In [12]:
```
df_dt_performance=DataFrame({"Depth":list_of_depth_dt,"DT Model Accuracy%":list_of_models_score_dt})
df_dt_performance["DT Model Accuracy%"]=round(df_dt_performance["DT Model Accuracy%"]*100,2)
```

In [13]:
```
best_depth=df_dt_performance.sort_values("DT Model Accuracy%",ascending=False)["Depth"].values[0]
best_depth_accuracy=df_dt_performance.sort_values("DT Model Accuracy%",ascending=False)["DT Model Accuracy%"].values[0
```

Print Accuracy High Level Report

In [14]:
```
print("The most accurate KNN model is the one with " + str(best_k) + " neighbours with " + str(best_k_accuracy)+"% acc

print("The most accurate Decision Tree \
model is the one with " + str(best_depth) + " Depth with " + str(best_depth_accuracy)+"% accuracy")
```

```
The most accurate KNN model is the one with 20 neighbours with 65.56% accuracy
The most accurate Decision Tree model is the one with 1 Depth with 66.67% accuracy
```

Top 5 Accuracy of each Model

In [15]:
```
df_performance.sort_values("KNN Model Accuracy%",ascending=False).head(5)
```

Out[15]:

	Neighbours	KNN Model Accuracy%
19	20	65.56
20	21	63.33
15	16	62.22
4	5	61.11
16	17	61.11

In [16]:
```
df_dt_performance.sort_values("DT Model Accuracy%",ascending=False).head(5)
```

Out[16]:

	Depth	DT Model Accuracy%
0	1	66.67
2	3	65.00
4	5	65.00
3	4	61.67
1	2	61.67

In [17]:
```
print("My name is Farbod Baharkoush")
print("My NetID is: fbahar2")
print("I hereby certify that I have read the University policy on Academic Integrity and that I am not in violation.")
```

```
My name is Farbod Baharkoush
My NetID is: fbahar2
I hereby certify that I have read the University policy on Academic Integrity and that I am not in violation.
```

In [ ]:

In [ ]:

In [ ]: