1 HW: Machine Learning in Finance Lab_Week 02

1.1 due 2023-02-05

- Yu-Ching Liao ycliao3@illinois.edu (mailto:ycliao3@illinois.edu)

2 Basic Import

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
```

executed in 2.41s, finished 22:32:00 2023-02-01

In [2]:
```
import warnings
warnings.filterwarnings("ignore")
```

executed in 3ms, finished 22:32:00 2023-02-01

In [3]:
```
T = pd.read_csv('/Users/yu-chingliao/Library/CloudStorage/GoogleDrive-josephliao0127@gmail.com/My Drive/Note/UIUC/Spr
T = T.drop(['rowindex', 'contract'], axis=1)
T.columns[:-1]
```

executed in 12ms, finished 22:32:00 2023-02-01

Out[3]:
```
Index(['price_crossing', 'price_distortion', 'roll_start', 'roll_heart',
       'near_minus_next', 'ctd_last_first', 'ctd1_percent', 'delivery_cost',
       'delivery_ratio'],
      dtype='object')
```

In [4]:
```
X = T.drop("squeeze", axis=1).values
y = T["squeeze"].values
y = np.array(list(map(lambda x: int(x), y)))
```

executed in 4ms, finished 22:32:00 2023-02-01

In [5]:
```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=1, stratify = y)
print(X_train.shape, y_train.shape)
```

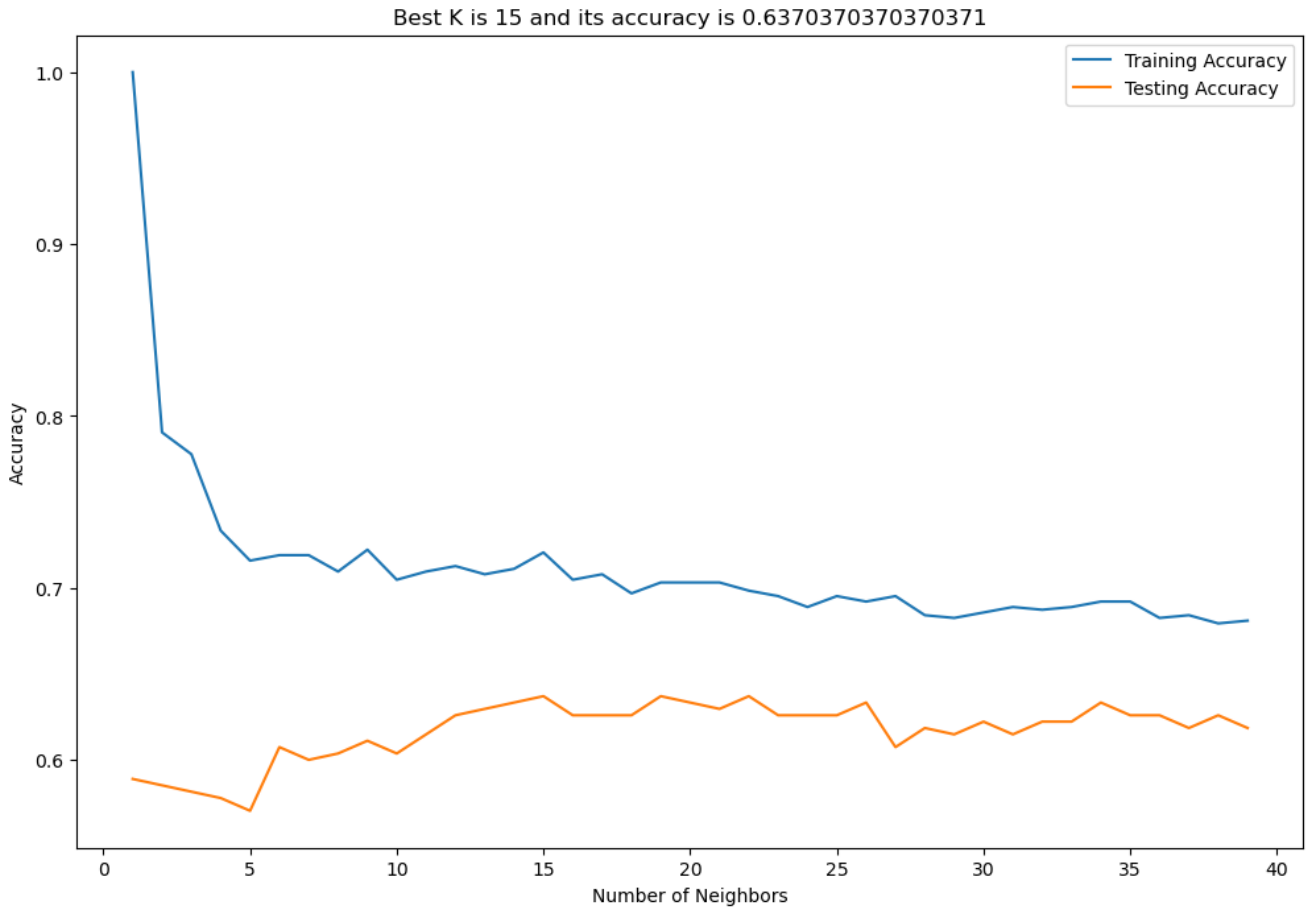
executed in 6ms, finished 22:32:00 2023-02-01

```
(630, 9) (630,)
```

3 KNN Model

In [6]:
```
neighbors = np.arange(1, 40)
train_accuracies = {}
test_accuracies = {}
best_acc = 0
best_idx = 0

for neighbor in neighbors:

    # Set up a KNN Classifier
    knn = KNeighborsClassifier(n_neighbors=neighbor)

    # Fit the model
    knn.fit(X_train, y_train)

    # Compute accuracy
    train_accuracies[neighbor] = knn.score(X_train, y_train)
    test_accuracies[neighbor] = knn.score(X_test, y_test)
    if knn.score(X_test, y_test) > best_acc:
        best_acc = knn.score(X_test, y_test)
        best_idx = neighbor


plt.figure(figsize=[12,8])

s = "Best K is "+ str(best_idx)+ ' and its accuracy is ' +str(best_acc)
# Add a title
plt.title(s)

# Plot training accuracies
plt.plot(neighbors, train_accuracies.values(), label="Training Accuracy")

# Plot test accuracies
plt.plot(neighbors , test_accuracies.values(), label="Testing Accuracy")

plt.legend()
plt.xlabel("Number of Neighbors")
plt.ylabel("Accuracy")

# Display the plot
plt.show()
```

executed in 1.94s, finished 22:32:01 2023-02-01



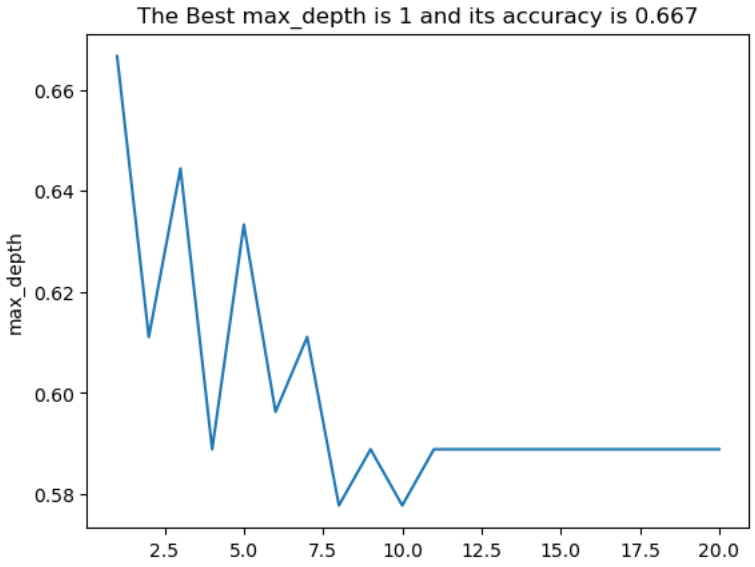
4 Decision Tree (Gini)

In [15]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
import matplotlib.pyplot as plt
from pydotplus import graph_from_dot_data
from sklearn.tree import export_graphviz
import matplotlib.image as mpimg
```

executed in 4ms, finished 22:38:30 2023-02-01

In [12]:
```
best_acc = 0
best_index = 0
X = []
Y = []
for i in range(1, 21):
    tree = DecisionTreeClassifier(criterion='gini',
                                  max_depth=i,
                                  random_state=1)
    tree.fit(X_train, y_train)
    y_pred = tree.predict(X_test)
    X.append(i)
    Y.append(metrics.accuracy_score(y_test, y_pred))
    if metrics.accuracy_score(y_test, y_pred)>best_acc:
        best_acc = metrics.accuracy_score(y_test, y_pred)
        best_index = i
ttl = "The Best max_depth is "+str(best_index)+" and its accuracy is " +str(round(best_acc, 3))
plt.plot(X,Y)
plt.title(ttl)
plt.ylabel("Accuracy")
plt.ylabel("max_depth")
plt.show()
ttl = "The Best max_depth is "+str(best_index)+" and its accuracy is " +str(best_acc)
```

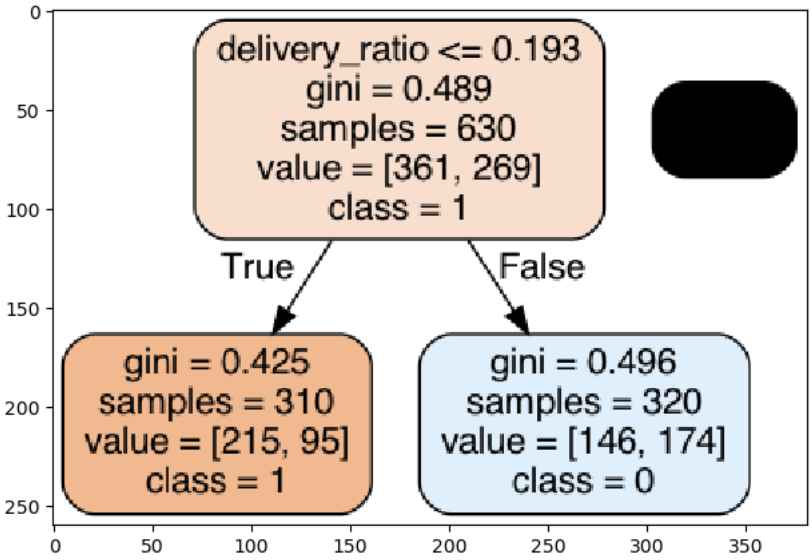
executed in 172ms, finished 22:36:00 2023-02-01



In [16]:
```
best_tree = DecisionTreeClassifier(criterion='gini',
                                   max_depth=best_index,
                                   random_state=1)
best_tree.fit(X_train, y_train)
dot_data = export_graphviz(best_tree,
                           filled=True,
                          rounded = True,
                          class_names = ["1", '0'],
                          feature_names=T.columns[:-1],
                          out_file = None)
graph = graph_from_dot_data(dot_data)
graph.write_png("tree.png")
img = mpimg.imread("tree.png")
plt.figure(figsize=(8, 5))
plt.imshow(img,interpolation='nearest')
```

executed in 351ms, finished 22:38:35 2023-02-01

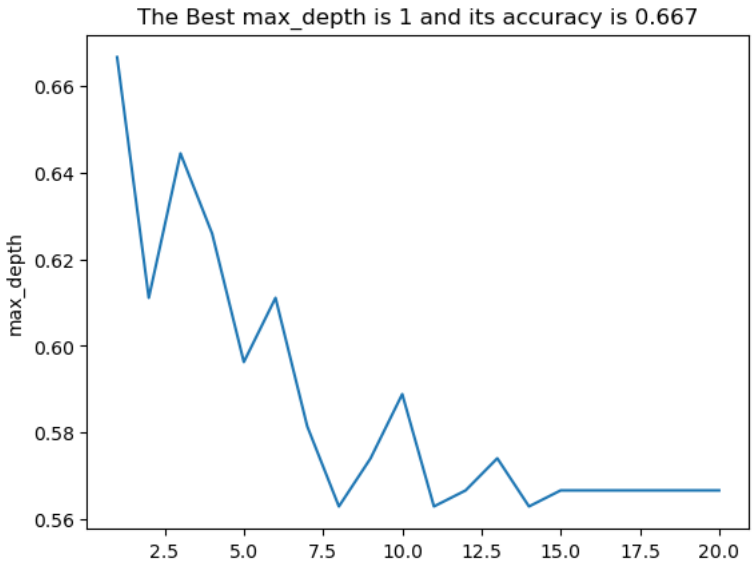
Out[16]: <matplotlib.image.AxesImage at 0x7fd1c8873df0>



5 Decision Tree (Entropy)

In [18]:
```
best_acc = 0
best_index = 0
X = []
Y = []
for i in range(1, 21):
    tree = DecisionTreeClassifier(criterion='entropy',
                                  max_depth=i,
                                  random_state=1)
    tree.fit(X_train, y_train)
    y_pred = tree.predict(X_test)
    X.append(i)
    Y.append(metrics.accuracy_score(y_test, y_pred))

    if metrics.accuracy_score(y_test, y_pred)>best_acc:
        best_acc = metrics.accuracy_score(y_test, y_pred)
        best_index = i
ttl = "The Best max_depth is "+str(best_index)+" and its accuracy is " +str(round(best_acc, 3))
plt.plot(X,Y)
plt.title(ttl)
plt.ylabel("Accuracy")
plt.ylabel("max_depth")
plt.show()
ttl = "The Best max_depth is "+str(best_index)+" and its accuracy is " +str(best_acc)
```

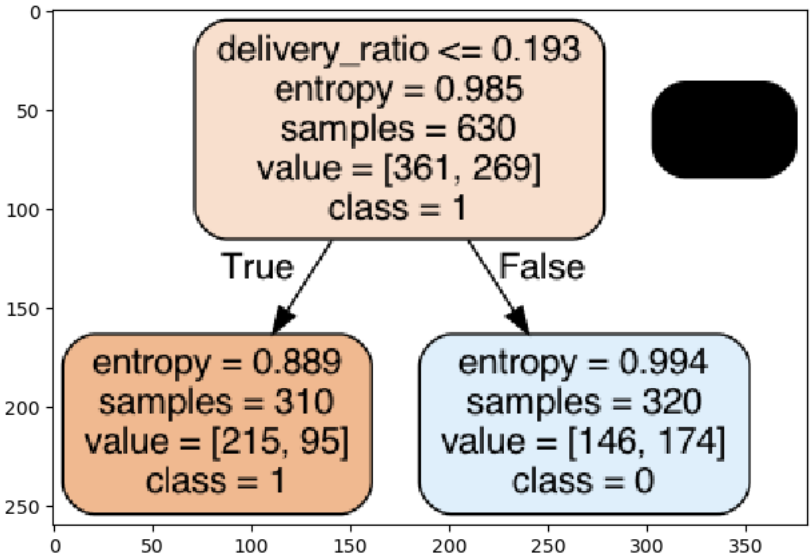
executed in 273ms, finished 22:39:29 2023-02-01



In [19]:
```
best_tree = DecisionTreeClassifier(criterion='entropy',
                                   max_depth=best_index,
                                   random_state=1)
best_tree.fit(X_train, y_train)
dot_data = export_graphviz(best_tree,
                           filled=True,
                          rounded = True,
                          class_names = ["1", '0'],
                          feature_names=T.columns[:-1],
                          out_file = None)
graph = graph_from_dot_data(dot_data)
graph.write_png("tree.png")
img = mpimg.imread("tree.png")
plt.figure(figsize=(8, 5))
plt.imshow(img,interpolation='nearest')
```

executed in 345ms, finished 22:39:32 2023-02-01

Out[19]: <matplotlib.image.AxesImage at 0x7fd1d8ba7640>



6 Signing

In [20]:
```
print("My name is Yu-Ching Liao")
print("My NetID is: 656724372")
print("I hereby certify that I have read the University policy on Academic Integrity and that I am not in violation.")
```

executed in 3ms, finished 22:39:39 2023-02-01

```
My name is Yu-Ching Liao
My NetID is: 656724372
I hereby certify that I have read the University policy on Academic Integrity and that I am not in violation.
```

In [ ]: