Question1

In [1]:

```python
import pandas as pd
import numpy as np
```

In [2]:

```python
# Read file and add column 'TrueLabel' of SPY
dfSPY = pd.read_csv("C:/Users/Lee/iCloudDrive/Document/Boston University/CS677 DS with Python/Homework/Assignme
tLabel = np.zeros_like(dfSPY.shape[0])
dfSPY['TrueLabel'] = tLabel
# Read file and add column 'TrueLabel' of Google
dfGOO = pd.read_csv("C:/Users/Lee/iCloudDrive/Document/Boston University/CS677 DS with Python/Homework/Assignme
tLabel = np.zeros_like(dfSPY.shape[0])
dfGOO['TrueLabel'] = tLabel
```

In [3]:

```python
# Assigning '-' or '+' based on column 'Return'
# I personally define Return = 0 as '-'

# SPY
profitReturn = (dfSPY.Return > 0)
lossReturn = (dfSPY.Return <= 0)
dfSPY.loc[profitReturn, 'TrueLabel'] = '+'
dfSPY.loc[lossReturn, 'TrueLabel'] = '-'
# GOOGL
profitReturn = (dfGOO.Return > 0)
lossReturn = (dfGOO.Return <=0)
dfGOO.loc[profitReturn, 'TrueLabel'] = '+'
dfGOO.loc[lossReturn, 'TrueLabel'] = '-'
```

In [4]:

```python
# Locate the index of the last date of the third year
temp = dfSPY[dfSPY.Year == 2019 ].index.values[0]

# Creat sub dataframe of first 3 years and sub dataframe of last 2 years
# For SPY
dfSPY3 = dfSPY.loc[:temp-1, ['Date', 'Return', 'TrueLabel']]
dfSPY2 = dfSPY.loc[temp:, ['Date', 'Return', 'TrueLabel']]
dfSPY2 = dfSPY2.reset_index(drop=True) # Reset the index of testing data

# For GOOGL
dfGOO3 = dfGOO.loc[:temp-1, ['Date', 'Return', 'TrueLabel']]
dfGOO2 = dfGOO.loc[temp:, ['Date', 'Return', 'TrueLabel']]
dfGOO2 = dfGOO2.reset_index(drop=True) # Reset the index of testing data
```

In [5]:

```python
Lplus = sum(( dfSPY3.Return > 0 ))
Lminus = sum(( dfSPY3.Return <= 0 ))
totalDay = dfSPY3.shape[0]
print('### Default probability p* = {:.1f}% that the next day is a "up" day'.format(Lplus*100/totalDay))
```

```
### Default probability p* = 54.8% that the next day is a "up" day
```

In [6]:

```python
def labelSlicing(key, labelArray):
    k = key
    predictLabel = ["NA" for i in range(k)]
    index = k
    lastIndex = len(label) - 1
    while index <= lastIndex:
        i = index
        sub = labelArray[i-k : i-k+k]
        index = index + 1
        temp = ''
        for e in sub:
            temp = temp + e
        predictLabel.append(temp)
    return predictLabel

if __name__ == "__main__":

    # Creat label with k = 1, 2, 3, 4
    ## For APY
    label = np.array(dfSPY3.TrueLabel)
    ### K = 1, 2, 3, 4
    k1 = labelSlicing( key = 1 , labelArray = label )
    k2 = labelSlicing( key = 2 , labelArray = label )
    k3 = labelSlicing( key = 3 , labelArray = label )
    k4 = labelSlicing( key = 4 , labelArray = label )
    ### Add three columns into the dfSPY3 dataframe
    dfSPY3['K=1'] = k1
    dfSPY3['K=2'] = k2
    dfSPY3['K=3'] = k3
    dfSPY3['K=4'] = k4

    ## For GOOGL
    label = np.array(dfGOO3.TrueLabel)
    ### K = 1, 2, 3, 4
    k1 = labelSlicing( key = 1 , labelArray = label )
    k2 = labelSlicing( key = 2 , labelArray = label )
    k3 = labelSlicing( key = 3 , labelArray = label )
    k4 = labelSlicing( key = 4 , labelArray = label )
    ### Add three columns into the dfSPY3 dataframe
    dfGOO3['K=1'] = k1
    dfGOO3['K=2'] = k2
    dfGOO3['K=3'] = k3
    dfGOO3['K=4'] = k4
```

In [7]:

```python
# Creat label on last 2 years data with K = 1,2,3,4
## For SPY
label = np.array(dfSPY2.TrueLabel)
### K = 1, 2, 3, 4
k1 = labelSlicing( key = 1 , labelArray = label )
k2 = labelSlicing( key = 2 , labelArray = label )
k3 = labelSlicing( key = 3 , labelArray = label )
k4 = labelSlicing( key = 4 , labelArray = label )
### Add three columns into the dfSPY3 dataframe
dfSPY2['K=1'] = k1
dfSPY2['K=2'] = k2
dfSPY2['K=3'] = k3
dfSPY2['K=4'] = k4

## For GOOGL
### K = 1, 2, 3, 4
label = np.array(dfGOO2.TrueLabel)
k1 = labelSlicing( key = 1 , labelArray = label )
k2 = labelSlicing( key = 2 , labelArray = label )
k3 = labelSlicing( key = 3 , labelArray = label )
k4 = labelSlicing( key = 4 , labelArray = label )
### Add three columns into the dfSPY3 dataframe
dfGOO2['K=1'] = k1
dfGOO2['K=2'] = k2
dfGOO2['K=3'] = k3
dfGOO2['K=4'] = k4
```

In [8]:

```python
def keyMatch(preKey, posKey, preLable, posLable):
    matchCount = 0
    for (pre, pos) in zip(preLable, posLable):
        if (pre, pos) == (preKey, posKey):
            matchCount = matchCount + 1
    return matchCount

if __name__ == "__main__":

    # For SPY
    ### For K consecutive 'down days'
    tLabel = np.array(dfSPY3['TrueLabel'])
    kP = []
    ### Count the probability of '- +' as opposed to '- -'
    preLabel = np.array(dfSPY3['K=1'])
    upCount = keyMatch(preKey= '-', posKey= '+', preLable= preLabel, posLable= tLabel)
    downCount = keyMatch(preKey= '-', posKey= '-', preLable= preLabel, posLable= tLabel)
    temp = upCount /(downCount + upCount)
    kP.append( ('K=1', temp))
    ### Count the probability of '-- +' as opposed to '-- -'
    preLabel = np.array(dfSPY3['K=2'])
    upCount = keyMatch(preKey='--', posKey='+', preLable=preLabel, posLable=tLabel)
    downCount = keyMatch(preKey='--', posKey='-', preLable=preLabel, posLable=tLabel)
    temp = upCount / (downCount + upCount)
    kP.append( ('K=2', temp))
    ### Count the probability of '--- +' as opposed to '--- -'
    preLabel = np.array(dfSPY3['K=3'])
    upCount = keyMatch(preKey='---', posKey='+', preLable=preLabel, posLable=tLabel)
    downCount = keyMatch(preKey='---', posKey='-', preLable=preLabel, posLable=tLabel)
    temp = upCount / (downCount + upCount)
    kP.append( ('K=3', temp))
    ### Print
    print('### For SPY, k cosnsecutive "down days"')
    for p in kP:
        print('### for {} , the probability of the next day is a ''up day'' is {:.1f}%'.format(p[0], p[1]*100))
    print()

    # For GOO
    ### For K consecutive 'down days'
    tLabel = np.array(dfGOO3['TrueLabel'])
    kP = []
    ### Count the probability of '- +' as opposed to '- -'
    preLabel = np.array(dfGOO3['K=1'])
    upCount = keyMatch(preKey= '-', posKey= '+', preLable= preLabel, posLable= tLabel)
    downCount = keyMatch(preKey= '-', posKey= '-', preLable= preLabel, posLable= tLabel)
    temp = upCount /(downCount + upCount)
    kP.append( ('K=1', temp))
    ### Count the probability of '-- +' as opposed to '-- -'
    preLabel = np.array(dfGOO3['K=2'])
    upCount = keyMatch(preKey='--', posKey='+', preLable=preLabel, posLable=tLabel)
    downCount = keyMatch(preKey='--', posKey='-', preLable=preLabel, posLable=tLabel)
    temp = upCount / (downCount + upCount)
    kP.append( ('K=2', temp))
    ### Count the probability of '--- +' as opposed to '--- -'
    preLabel = np.array(dfGOO3['K=3'])
    upCount = keyMatch(preKey='---', posKey='+', preLable=preLabel, posLable=tLabel)
    downCount = keyMatch(preKey='---', posKey='-', preLable=preLabel, posLable=tLabel)
    temp = upCount / (downCount + upCount)
    kP.append( ('K=3', temp))
    ### Print
    print('### For GOOGL,  k cosnsecutive "down days"')
    for p in kP:
        print('### for {} , the probability of the next day is a ''up day'' is {:.1f}%'.format(p[0], p[1]*100))
    print()



    # For SPY
    ### For consecutive 'up days'
    tLabel = np.array(dfSPY3['TrueLabel'])
    kP1 = []
    ### Count the probability of '+ +' as opposed to'+ -'
    preLabel = np.array(dfSPY3['K=1'])
    upCount = keyMatch(preKey= '+', posKey= '+', preLable= preLabel, posLable= tLabel)
    downCount = keyMatch(preKey= '+', posKey= '-', preLable= preLabel, posLable= tLabel)
    temp = upCount /(downCount + upCount)
    kP1.append( ('K=1', temp))

    ### Count the probability of '++ +' as opposed to'++ -'
    preLabel = np.array(dfSPY3['K=2'])
    upCount = keyMatch(preKey= '++', posKey= '+', preLable= preLabel, posLable= tLabel)
    downCount = keyMatch(preKey= '++', posKey= '-', preLable= preLabel, posLable= tLabel)
    temp = upCount /(downCount + upCount)
    kP1.append( ('K=2', temp))
    ### Count the probability of '+++ +' as opposed to'+++ -'
    preLabel = np.array(dfSPY3['K=3'])
    upCount = keyMatch(preKey= '+++', posKey= '+', preLable= preLabel, posLable= tLabel)
    downCount = keyMatch(preKey= '+++', posKey= '-', preLable= preLabel, posLable= tLabel)
    temp = upCount /(downCount + upCount)
    kP1.append( ('K=3', temp))
    ### Print
    print('### For k cosnsecutive "up days"')
    for p in kP1:
        print('### for {} , the probability of the next day is a ''up day'' is {:.1f}%'.format(p[0], p[1]*100))
    print()

    # For GOOGL
    ### For consecutive 'up days'
    tLabel = np.array(dfGOO3['TrueLabel'])
    kP1 = []
    ### Count the probability of '+ +' as opposed to'+ -'
    preLabel = np.array(dfGOO3['K=1'])
    upCount = keyMatch(preKey= '+', posKey= '+', preLable= preLabel, posLable= tLabel)
    downCount = keyMatch(preKey= '+', posKey= '-', preLable= preLabel, posLable= tLabel)
    temp = upCount /(downCount + upCount)
    kP1.append( ('K=1', temp))

    ### Count the probability of '++ +' as opposed to'++ -'
    preLabel = np.array(dfGOO3['K=2'])
    upCount = keyMatch(preKey= '++', posKey= '+', preLable= preLabel, posLable= tLabel)
    downCount = keyMatch(preKey= '++', posKey= '-', preLable= preLabel, posLable= tLabel)
    temp = upCount /(downCount + upCount)
    kP1.append( ('K=2', temp))
    ### Count the probability of '+++ +' as opposed to'+++ -'
    preLabel = np.array(dfGOO3['K=3'])
    upCount = keyMatch(preKey= '+++', posKey= '+', preLable= preLabel, posLable= tLabel)
    downCount = keyMatch(preKey= '+++', posKey= '-', preLable= preLabel, posLable= tLabel)
    temp = upCount /(downCount + upCount)
    kP1.append( ('K=3', temp))
    ### Print
    print('### For GOOGL,  k cosnsecutive "up days"')
    for p in kP1:
        print('### for {} , the probability of the next day is a ''up day'' is {:.1f}%'.format(p[0], p[1]*100))
    print()
```

```
### For SPY, k cosnsecutive "down days"
### for K=1 , the probability of the next day is a up day is 58.7%
### for K=2 , the probability of the next day is a up day is 58.2%
### for K=3 , the probability of the next day is a up day is 61.0%

### For GOOGL,  k cosnsecutive "down days"
### for K=1 , the probability of the next day is a up day is 51.4%
### for K=2 , the probability of the next day is a up day is 47.1%
### for K=3 , the probability of the next day is a up day is 54.9%

### For k cosnsecutive "up days"
### for K=1 , the probability of the next day is a up day is 51.7%
### for K=2 , the probability of the next day is a up day is 49.8%
### for K=3 , the probability of the next day is a up day is 45.3%

### For GOOGL,  k cosnsecutive "up days"
### for K=1 , the probability of the next day is a up day is 53.9%
### for K=2 , the probability of the next day is a up day is 53.7%
### for K=3 , the probability of the next day is a up day is 53.9%
```

Question2

In [9]:

```python
# Make a prediction dictionary based on the training data (first 3 years data)

from itertools import permutations
def possiblePreKey(w):
    # Creat possible preKey of w
    # Eg. if w = 2, then return ['-+', '++', '--', '+-']
    pool = '+' * w + '-' * w
    preSet = list(set(permutations(pool, w)))
    for i in range(len(preSet)):
        preSet[i] = ''.join(preSet[i])
    return preSet

def predictModel(wValue, preKeyPool, preLabel, posLabel):
    # Provide the prediction outcome based on w
    # Eg. for w = 2, return {'-+':'+', '+-':'+', '++': '-', '--':'+'}
    w = wValue
    poolList = possiblePreKey(w)
    modelDic = dict.fromkeys(poolList,'NA')
    for p in poolList:
        upCount = keyMatch(preKey=p, posKey='+', preLable=preLabel, posLable=posLabel)
        downCount = keyMatch(preKey=p, posKey='-', preLable=preLabel, posLable=posLabel)
        if upCount > downCount:
            modelDic[p] = '+'
        else:
            modelDic[p] = '-'
    return modelDic

if __name__ == "__main__":
    # For SPY
    ## W = 2
    w = 2
    posLabel = np.array(dfSPY3['TrueLabel'])
    preKeyPool = possiblePreKey(w)
    preLabel = np.array(dfSPY3['K=2'])
    w2Model = predictModel(wValue= w, preKeyPool= preKeyPool, preLabel= preLabel, posLabel= posLabel )
    ## W = 3
    w = 3
    posLabel = np.array(dfSPY3['TrueLabel'])
    preKeyPool = possiblePreKey(w)
    preLabel = np.array(dfSPY3['K=3'])
    w3Model = predictModel(wValue= w, preKeyPool= preKeyPool, preLabel= preLabel, posLabel= posLabel )
    ## W = 4
    w = 4
    posLabel = np.array(dfSPY3['TrueLabel'])
    preKeyPool = possiblePreKey(w)
    preLabel = np.array(dfSPY3['K=4'])
    w4Model = predictModel(wValue= w, preKeyPool= preKeyPool, preLabel= preLabel, posLabel= posLabel )

    # For GOOGL
    ## W = 2
    w = 2
    posLabel = np.array(dfGOO3['TrueLabel'])
    preKeyPool = possiblePreKey(w)
    preLabel = np.array(dfGOO3['K=2'])
    w2ModelG = predictModel(wValue= w, preKeyPool= preKeyPool, preLabel= preLabel, posLabel= posLabel )
    ## W = 3
    w = 3
    posLabel = np.array(dfGOO3['TrueLabel'])
    preKeyPool = possiblePreKey(w)
    preLabel = np.array(dfGOO3['K=3'])
    w3ModelG = predictModel(wValue= w, preKeyPool= preKeyPool, preLabel= preLabel, posLabel= posLabel )
    ## W = 4
    w = 3
    posLabel = np.array(dfGOO3['TrueLabel'])
    preKeyPool = possiblePreKey(w)
    preLabel = np.array(dfGOO3['K=4'])
    w4ModelG = predictModel(wValue= w, preKeyPool= preKeyPool, preLabel= preLabel, posLabel= posLabel )
```

In [10]:

```python
# Creat prediction lable based on W = 2,3,4

def makePrediction(testData, wValue, trainModel):
    preDictLable = ['na' for i in range(wValue)]
    preKey = testData
    for pk in preKey[wValue:]:
        preDictLable.append(trainModel[pk])
    return preDictLable

if __name__ == "__main__":
    # For SPY
    ## W = 2
    w2PredictLabel = makePrediction(testData = np.array(dfSPY2['K=2']), wValue= 2, trainModel= w2Model)
    dfSPY2['w2PredictLabel'] = w2PredictLabel
    ## W = 3
    w3PredictLabel = makePrediction(testData= np.array(dfSPY2['K=3']), wValue= 3, trainModel= w3Model)
    dfSPY2['w3PredictLabel'] = w3PredictLabel
    ## W = 4
    w4PredictLabel = makePrediction(testData= np.array(dfSPY2['K=4']), wValue= 4, trainModel= w4Model)
    dfSPY2['w4PredictLabel'] = w4PredictLabel

    # For GOOGL
    ## W = 2
    w2PredictLabel = makePrediction(testData = np.array(dfGOO2['K=2']), wValue= 2, trainModel= w2Model)
    dfGOO2['w2PredictLabel'] = w2PredictLabel
    ## W = 3
    w3PredictLabel = makePrediction(testData= np.array(dfGOO2['K=3']), wValue= 3, trainModel= w3Model)
    dfGOO2['w3PredictLabel'] = w3PredictLabel
    ## W = 4
    w4PredictLabel = makePrediction(testData= np.array(dfGOO2['K=4']), wValue= 4, trainModel= w4Model)
    dfGOO2['w4PredictLabel'] = w4PredictLabel
```

Question3

In [11]:

```python
def ensemLabel(data):
    eLab = ['na' for i in range(4)]
    temp = data.loc[4:, ['w2PredictLabel', 'w3PredictLabel', 'w4PredictLabel']]
    for (a, b, c) in zip(temp.w2PredictLabel, temp.w3PredictLabel, temp.w4PredictLabel):
        value = (a == '+') + (b == '+') + (c == '+')
        if value >= 2:
            p = '+'
        else:
            p = '-'
        eLab.append(p)
    return eLab

if __name__ == "__main__":

    # For SPY
    data = dfSPY2
    eLab = ensemLabel(data)
    dfSPY2['ensembleLabel'] = eLab

    # For GOOGL
    data = dfGOO2
    eLab = ensemLabel(data)
    dfGOO2['ensembleLabel'] = eLab
```

In [12]:

```python
# Compute the accuracy of ensembleLabel
# For SPY
size = dfSPY2.shape[0]
e = sum(dfSPY2['TrueLabel'] == dfSPY2['ensembleLabel']) / size
print("### For SPY")
print("### The accuracy of ensembleLable is {:.1f}%".format(e*100))
print()
# For GOOGL
size = dfSPY2.shape[0]
e = sum(dfGOO2['TrueLabel'] == dfGOO2['ensembleLabel']) / size
print("### For GOOGL")
print("### The accuracy of ensembleLable is {:.1f}%".format(e*100))
```

```
### For SPY
### The accuracy of ensembleLable is 57.9%

### For GOOGL
### The accuracy of ensembleLable is 53.2%
```

In [13]:

```python
# Prediction '-' labels
## For SPY
size = sum(dfSPY2['TrueLabel'] == '-')
w2 = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w2PredictLabel == '-')) / size
w3 = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w3PredictLabel == '-')) / size
w4 = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w4PredictLabel == '-')) / size
eL = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.ensembleLabel == '-')) / size
print("For SPY")
print("The accuracy on predicting '-' of W = 2 is {:.1f}%".format(w2*100))
print("The accuracy on predicting '-' of W = 3 is {:.1f}%".format(w3*100))
print("The accuracy on predicting '-' of W = 4 is {:.1f}%".format(w4*100))
print("The accuracy on predicting '-' of ensembleLabel is {:.1f}%".format(eL*100))
print()
## GOOGL
size = sum(dfGOO2['TrueLabel'] == '-')
w2 = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w2PredictLabel == '-')) / size
w3 = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w3PredictLabel == '-')) / size
w4 = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w4PredictLabel == '-')) / size
eL = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.ensembleLabel == '-')) / size
print("For GOOGL")
print("The accuracy on predicting '-' of W = 2 is {:.1f}%".format(w2*100))
print("The accuracy on predicting '-' of W = 3 is {:.1f}%".format(w3*100))
print("The accuracy on predicting '-' of W = 4 is {:.1f}%".format(w4*100))
print("The accuracy on predicting '-' of ensembleLabel is {:.1f}%".format(eL*100))
print()

# Prediction '+' labels
## For SPY
size = sum(dfSPY2['TrueLabel'] == '+')
w2 = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w2PredictLabel == '+')) / size
w3 = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w3PredictLabel == '+')) / size
w4 = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w4PredictLabel == '+')) / size
eL = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.ensembleLabel == '+')) / size
print("For SPY")
print("The accuracy on predicting '+' of W = 2 is {:.1f}%".format(w2*100))
print("The accuracy on predicting '+' of W = 3 is {:.1f}%".format(w3*100))
print("The accuracy on predicting '+' of W = 4 is {:.1f}%".format(w4*100))
print("The accuracy on predicting '+' of ensembleLabel is {:.1f}%".format(eL*100))
print()
## For GOOGL
size = sum(dfGOO2['TrueLabel'] == '+')
w2 = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w2PredictLabel == '+')) / size
w3 = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w3PredictLabel == '+')) / size
w4 = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w4PredictLabel == '+')) / size
eL = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.ensembleLabel == '+')) / size
print("For GOO")
print("The accuracy on predicting '+' of W = 2 is {:.1f}%".format(w2*100))
print("The accuracy on predicting '+' of W = 3 is {:.1f}%".format(w3*100))
print("The accuracy on predicting '+' of W = 4 is {:.1f}%".format(w4*100))
print("The accuracy on predicting '+' of ensembleLabel is {:.1f}%".format(eL*100))
print()
```

```
For SPY
The accuracy on predicting '-' of W = 2 is 33.0%
The accuracy on predicting '-' of W = 3 is 23.0%
The accuracy on predicting '-' of W = 4 is 36.8%
The accuracy on predicting '-' of ensembleLabel is 23.0%

For GOOGL
The accuracy on predicting '-' of W = 2 is 31.0%
The accuracy on predicting '-' of W = 3 is 16.4%
The accuracy on predicting '-' of W = 4 is 27.9%
The accuracy on predicting '-' of ensembleLabel is 16.4%

For SPY
The accuracy on predicting '+' of W = 2 is 67.1%
The accuracy on predicting '+' of W = 3 is 83.1%
The accuracy on predicting '+' of W = 4 is 71.2%
The accuracy on predicting '+' of ensembleLabel is 82.7%

For GOO
The accuracy on predicting '+' of W = 2 is 70.1%
The accuracy on predicting '+' of W = 3 is 83.1%
The accuracy on predicting '+' of W = 4 is 72.3%
The accuracy on predicting '+' of ensembleLabel is 83.1%
```

Question 4

In [14]:

```python
# True Positive
## For SPY
w2TPSPY = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w2PredictLabel == '+'))
w3TPSPY = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w3PredictLabel == '+'))
w4TPSPY = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w4PredictLabel == '+'))
eLTPSPY = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.ensembleLabel == '+'))
print("For SPY")
print('True Positives for W = 2 is {}'.format(w2TPSPY))
print('True Positives for W = 3 is {}'.format(w3TPSPY))
print('True Positives for W = 4 is {}'.format(w4TPSPY))
print('True Positives for ensembleLabel is {}'.format(eLTPSPY))
print()

## For GOOGL
w2TPG = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w2PredictLabel == '+'))
w3TPG = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w3PredictLabel == '+'))
w4TPG = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w4PredictLabel == '+'))
eLTPG = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.ensembleLabel == '+'))
print("For GOOGL")
print('True Positives for W = 2 is {}'.format(w2TPG))
print('True Positives for W = 3 is {}'.format(w3TPG))
print('True Positives for W = 4 is {}'.format(w4TPG))
print('True Positives for ensembleLabel is {}'.format(eLTPG))
print()
```

```
For SPY
True Positives for W = 2 is 198
True Positives for W = 3 is 245
True Positives for W = 4 is 210
True Positives for ensembleLabel is 244

For GOOGL
True Positives for W = 2 is 195
True Positives for W = 3 is 231
True Positives for W = 4 is 201
True Positives for ensembleLabel is 231
```

In [15]:

```python
# False Positive
## For SPY
w2FPSPY = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w2PredictLabel == '+'))
w3FPSPY = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w3PredictLabel == '+'))
w4FPSPY = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w4PredictLabel == '+'))
eLFPSPY = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.ensembleLabel == '+'))
print("For SPY")
print('False Positives for W = 2 is {}'.format(w2FPSPY))
print('False Positives for W = 3 is {}'.format(w3FPSPY))
print('False Positives for W = 4 is {}'.format(w4FPSPY))
print('False Positives for ensembleLabel is {}'.format(eLFPSPY))
print()

## For GOOGL
w2FPG = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w2PredictLabel == '+'))
w3FPG = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w3PredictLabel == '+'))
w4FPG = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w4PredictLabel == '+'))
eLFPG = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.ensembleLabel == '+'))
print("For GOOGL")
print('False Positives for W = 2 is {}'.format(w2FPG))
print('False Positives for W = 3 is {}'.format(w3FPG))
print('False Positives for W = 4 is {}'.format(w4FPG))
print('False Positives for ensembleLabel is {}'.format(eLFPG))
print()
```

```
For SPY
False Positives for W = 2 is 139
False Positives for W = 3 is 160
False Positives for W = 4 is 131
False Positives for ensembleLabel is 160

For GOOGL
False Positives for W = 2 is 155
False Positives for W = 3 is 188
False Positives for W = 4 is 161
False Positives for ensembleLabel is 187
```

In [16]:

```python
# True Negetive
## For SPY
w2TNSPY = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w2PredictLabel == '-'))
w3TNSPY = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w3PredictLabel == '-'))
w4TNSPY = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.w4PredictLabel == '-'))
eLTNSPY = sum((dfSPY2.TrueLabel == '-') & (dfSPY2.ensembleLabel == '-'))
print("For SPY")
print('True Negative for W = 2 is {}'.format(w2TNSPY))
print('True Negative for W = 3 is {}'.format(w3TNSPY))
print('True Negative for W = 4 is {}'.format(w4TNSPY))
print('True Negative for ensembleLabel is {}'.format(eLTNSPY))
print()

## For GOOGL
w2TNG = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w2PredictLabel == '-'))
w3TNG = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w3PredictLabel == '-'))
w4TNG = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.w4PredictLabel == '-'))
eLTNG = sum((dfGOO2.TrueLabel == '-') & (dfGOO2.ensembleLabel == '-'))
print("For GOOGL")
print('True Negative for W = 2 is {}'.format(w2TNG))
print('True Negative for W = 3 is {}'.format(w3TNG))
print('True Negative for W = 4 is {}'.format(w4TNG))
print('True Negative for ensembleLabel is {}'.format(eLTNG))
print()
```

```
For SPY
True Negative for W = 2 is 69
True Negative for W = 3 is 48
True Negative for W = 4 is 77
True Negative for ensembleLabel is 48

For GOOGL
True Negative for W = 2 is 70
True Negative for W = 3 is 37
True Negative for W = 4 is 63
True Negative for ensembleLabel is 37
```

In [17]:

```python
# False Negative
## For SPY
w2FNSPY = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w2PredictLabel == '-'))
w3FNSPY = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w3PredictLabel == '-'))
w4FNSPY = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.w4PredictLabel == '-'))
eLFNSPY = sum((dfSPY2.TrueLabel == '+') & (dfSPY2.ensembleLabel == '-'))
print("For SPY")
print('False Negative for W = 2 is {}'.format(w2FNSPY))
print('False Negative for W = 3 is {}'.format(w3FNSPY))
print('False Negative for W = 4 is {}'.format(w4FNSPY))
print('False Negative for ensembleLabel is {}'.format(eLFNSPY))
print()

## For GOOGL
w2FNG = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w2PredictLabel == '-'))
w3FNG = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w3PredictLabel == '-'))
w4FNG = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.w4PredictLabel == '-'))
eLFNG = sum((dfGOO2.TrueLabel == '+') & (dfGOO2.ensembleLabel == '-'))
print("For GOOGL")
print('False Negative for W = 2 is {}'.format(w2FNG))
print('False Negative for W = 3 is {}'.format(w3FNG))
print('False Negative for W = 4 is {}'.format(w4FNG))
print('False Negative for ensembleLabel is {}'.format(eLFNG))
print()
```

```
For SPY
False Negative for W = 2 is 96
False Negative for W = 3 is 48
False Negative for W = 4 is 82
False Negative for ensembleLabel is 48

For GOOGL
False Negative for W = 2 is 82
False Negative for W = 3 is 45
False Negative for W = 4 is 75
False Negative for ensembleLabel is 45
```

In [18]:

```python
# TPR = TP/(TP+FN)
## For SPY
w2TPR_SPY = w2TPSPY/(w2TPSPY + w2FNSPY)
w3TPR_SPY = w3TPSPY/(w3TPSPY + w3FNSPY)
w4TPR_SPY = w4TPSPY/(w4TPSPY + w4FNSPY)
eLTPR_SPY = eLTPSPY/(eLTPSPY + eLFNSPY)
print("TPR for SPY")
print("w2 = {:.1f}%".format(w2TPR_SPY*100))
print("w3 = {:.1f}%".format(w3TPR_SPY*100))
print("w4 = {:.1f}%".format(w4TPR_SPY*100))
print("eL = {:.1f}%".format(eLTPR_SPY*100))
## For GOOGL
w2TPR_G = w2TPG/(w2TPG + w2FNG)
w3TPR_G = w3TPG/(w3TPG + w3FNG)
w4TPR_G = w4TPG/(w4TPG + w4FNG)
eLTPR_G = eLTPG/(eLTPG + eLFNG)
print("TPR for GOOGL")
print("w2 = {:.1f}%".format(w2TPR_G*100))
print("w3 = {:.1f}%".format(w3TPR_G*100))
print("w4 = {:.1f}%".format(w4TPR_G*100))
print("eL = {:.1f}%".format(eLTPR_G*100))
```

```
TPR for SPY
w2 = 67.3%
w3 = 83.6%
w4 = 71.9%
eL = 83.6%
TPR for GOOGL
w2 = 70.4%
w3 = 83.7%
w4 = 72.8%
eL = 83.7%
```

In [19]:

```python
# TNR = TN/(TN+FP)
## For SPY
w2TNR_SPY = w2TNSPY/(w2TNSPY + w2FPSPY)
w3TNR_SPY = w3TNSPY/(w3TNSPY + w3FPSPY)
w4TNR_SPY = w4TNSPY/(w4TNSPY + w4FPSPY)
eLTNR_SPY = eLTNSPY/(eLTNSPY + eLFPSPY)
print("TPR for SPY")
print("w2 = {:.1f}%".format(w2TNR_SPY*100))
print("w3 = {:.1f}%".format(w3TNR_SPY*100))
print("w4 = {:.1f}%".format(w4TNR_SPY*100))
print("eL = {:.1f}%".format(eLTNR_SPY*100))
print()
## For GOOGL
w2TNR_G = w2TNG/(w2TNG + w2FPG)
w3TNR_G = w3TNG/(w3TNG + w3FPG)
w4TNR_G = w4TNG/(w4TNG + w4FPG)
eLTNR_G = eLTNG/(eLTNG + eLFPG)
print("TPR for SPY")
print("w2 = {:.1f}%".format(w2TNR_G*100))
print("w3 = {:.1f}%".format(w3TNR_G*100))
print("w4 = {:.1f}%".format(w4TNR_G*100))
print("eL = {:.1f}%".format(eLTNR_G*100))
```

```
TPR for SPY
w2 = 33.2%
w3 = 23.1%
w4 = 37.0%
eL = 23.1%

TPR for SPY
w2 = 31.1%
w3 = 16.4%
w4 = 28.1%
eL = 16.5%
```