In [381…
```
import pandas as pd
import numpy as np
import sklearn
import matplotlib.pyplot as plt
from sklearn.model_selection \
import train_test_split
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler , LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
```

In [382…
```
df=pd.read_csv('NVDA.csv')
df_week=df.groupby(["Year","Week_Number"]).nth(-1)
df_week=df_week.reset_index()
df_week_2017=df_week.loc[df_week['Year']==2017,:]
df_week_2018=df_week.loc[df_week['Year']==2018,:].reset_index()
#print(df_week_2018)
df_week_2017["ture_label"]=df_week_2017['Return'].apply(lambda x:'green' if x>(
# plt.plot(df_week_2017["Close"])
# plt.show()
df_week_2018["ture_label"]=df_week_2018['Return'].apply(lambda x:'green' if x>(

print(df_week_2018)
```

```
    index  Year  Week_Number        Date  Month  Day   Weekday Year_Week  \
0      52  2018            0  2018-01-05      1    5    Friday   2018-00   
1      53  2018            1  2018-01-12      1   12    Friday   2018-01   
2      54  2018            2  2018-01-19      1   19    Friday   2018-02   
3      55  2018            3  2018-01-26      1   26    Friday   2018-03   
4      56  2018            4  2018-02-02      2    2    Friday   2018-04   
5      57  2018            5  2018-02-09      2    9    Friday   2018-05   
6      58  2018            6  2018-02-16      2   16    Friday   2018-06   
7      59  2018            7  2018-02-23      2   23    Friday   2018-07   
8      60  2018            8  2018-03-02      3    2    Friday   2018-08   
9      61  2018            9  2018-03-09      3    9    Friday   2018-09   
10     62  2018           10  2018-03-16      3   16    Friday   2018-10   
11     63  2018           11  2018-03-23      3   23    Friday   2018-11   
12     64  2018           12  2018-03-29      3   29  Thursday   2018-12   
13     65  2018           13  2018-04-06      4    6    Friday   2018-13   
14     66  2018           14  2018-04-13      4   13    Friday   2018-14   
15     67  2018           15  2018-04-20      4   20    Friday   2018-15   
16     68  2018           16  2018-04-27      4   27    Friday   2018-16   
17     69  2018           17  2018-05-04      5    4    Friday   2018-17   
18     70  2018           18  2018-05-11      5   11    Friday   2018-18   
19     71  2018           19  2018-05-18      5   18    Friday   2018-19   
20     72  2018           20  2018-05-25      5   25    Friday   2018-20   
21     73  2018           21  2018-06-01      6    1    Friday   2018-21   
22     74  2018           22  2018-06-08      6    8    Friday   2018-22   
23     75  2018           23  2018-06-15      6   15    Friday   2018-23   
24     76  2018           24  2018-06-22      6   22    Friday   2018-24   
25     77  2018           25  2018-06-29      6   29    Friday   2018-25   
26     78  2018           26  2018-07-06      7    6    Friday   2018-26   
27     79  2018           27  2018-07-13      7   13    Friday   2018-27   
28     80  2018           28  2018-07-20      7   20    Friday   2018-28   
29     81  2018           29  2018-07-27      7   27    Friday   2018-29   
30     82  2018           30  2018-08-03      8    3    Friday   2018-30   
31     83  2018           31  2018-08-10      8   10    Friday   2018-31   
32     84  2018           32  2018-08-17      8   17    Friday   2018-32   
33     85  2018           33  2018-08-24      8   24    Friday   2018-33   
34     86  2018           34  2018-08-31      8   31    Friday   2018-34   
35     87  2018           35  2018-09-07      9    7    Friday   2018-35   
36     88  2018           36  2018-09-14      9   14    Friday   2018-36   
37     89  2018           37  2018-09-21      9   21    Friday   2018-37   
38     90  2018           38  2018-09-28      9   28    Friday   2018-38   
39     91  2018           39  2018-10-05     10    5    Friday   2018-39   
40     92  2018           40  2018-10-12     10   12    Friday   2018-40   
41     93  2018           41  2018-10-19     10   19    Friday   2018-41   
42     94  2018           42  2018-10-26     10   26    Friday   2018-42   
43     95  2018           43  2018-11-02     11    2    Friday   2018-43   
44     96  2018           44  2018-11-09     11    9    Friday   2018-44   
45     97  2018           45  2018-11-16     11   16    Friday   2018-45   
46     98  2018           46  2018-11-23     11   23    Friday   2018-46   
47     99  2018           47  2018-11-30     11   30    Friday   2018-47   
48    100  2018           48  2018-12-07     12    7    Friday   2018-48   
49    101  2018           49  2018-12-14     12   14    Friday   2018-49   
50    102  2018           50  2018-12-21     12   21    Friday   2018-50   
51    103  2018           51  2018-12-28     12   28    Friday   2018-51   
52    104  2018           52  2018-12-31     12   31    Monday   2018-52   

     Open   High    Low  Close    Volume  Adj Close    Return   Short_MA  \
0   53.55  54.23  52.77  53.85  58012400      53.32  0.008474  49.506429   
1   55.90  56.25  55.33  55.74  35991600      55.20 -0.004909  51.917857   
2   57.02  57.77  56.75  57.53  60938000      56.96  0.025263  53.900714   
3   59.53  60.83  59.40  60.83  51635600      60.23  0.029532  56.619286   
```

4	59.25	59.49	57.79	58.38	71846400	57.81	-0.029023	58.294286
5	59.56	59.72	54.38	58.02	167460400	57.45	0.066936	58.022143
6	61.35	62.50	60.87	60.96	63765600	60.36	-0.010791	57.866429
7	61.14	61.48	60.63	61.48	41530000	60.92	0.015610	58.157143
8	56.97	59.20	55.46	59.13	91342800	58.59	0.018647	59.685714
9	60.78	61.46	60.61	61.33	50552400	60.77	0.017207	59.941429
10	62.50	62.81	62.12	62.62	39945600	62.04	0.004572	60.313571
11	60.60	60.62	58.13	58.24	73562000	57.71	-0.036717	60.693571
12	56.03	58.88	55.17	57.90	91662800	57.36	0.046261	59.852143
13	54.31	55.40	53.27	53.56	66298800	53.07	-0.032207	57.415000
14	59.29	59.38	57.39	57.88	50303600	57.34	-0.013214	56.030000
15	57.17	58.03	56.86	57.18	38620000	56.65	-0.001441	56.369286
16	57.38	57.58	56.15	56.58	40084800	56.06	0.004928	56.557857
17	57.96	59.80	57.78	59.76	40066000	59.21	0.026052	56.534286
18	63.19	64.95	62.63	63.63	121445600	63.05	-0.021528	58.551429
19	62.45	63.09	61.44	61.49	48371600	60.92	-0.007146	60.785000
20	62.05	62.49	61.69	62.32	29211200	61.78	0.006419	61.806429
21	63.50	64.47	63.41	64.40	42196800	63.85	0.021531	61.649286
22	64.99	66.00	64.80	65.57	36045600	65.01	-0.002358	63.073571
23	66.15	66.87	65.84	66.32	43226000	65.74	-0.006182	64.606429
24	64.49	64.62	62.58	62.74	43416000	62.20	-0.023958	64.950000
25	60.87	61.00	59.21	59.22	39230000	58.72	-0.016441	62.800714
26	60.44	61.92	60.22	61.83	29635200	61.30	0.018951	61.242857
27	62.99	62.99	61.90	62.33	24698800	61.79	-0.007603	60.446429
28	62.98	63.38	62.61	62.72	22235200	62.18	-0.004523	61.542143
29	64.08	64.15	62.46	63.01	29542000	62.46	-0.011066	62.222857
30	62.90	63.26	62.73	63.03	21428800	62.48	0.005905	62.022143
31	63.29	64.03	63.17	63.70	25639600	63.15	-0.006512	62.427143
32	63.24	63.24	60.93	61.21	114318800	60.68	-0.049021	62.910714
33	66.79	68.20	66.75	68.06	53151200	67.47	0.020162	63.883571
34	69.25	70.30	69.15	70.17	30659200	69.60	0.010331	65.741429
35	67.25	69.23	66.80	67.96	29542000	67.42	-0.003153	67.237857
36	68.75	69.78	68.38	69.11	38693600	68.55	0.018759	68.321429
37	66.69	67.15	65.53	65.86	43512800	65.33	-0.010627	67.608571
38	68.18	70.48	67.90	70.25	70939600	69.69	0.050935	67.042143
39	69.57	70.20	66.89	67.46	42663600	66.92	-0.033764	67.910000
40	61.38	62.38	59.91	61.63	60823600	61.14	0.048526	66.497857
41	60.44	60.64	56.92	57.29	61360800	56.83	-0.043251	63.306429
42	49.58	51.21	48.28	49.57	66478400	49.17	-0.045949	57.455714
43	54.43	55.50	52.55	53.73	45296000	53.30	-0.014626	53.987143
44	50.60	52.33	50.26	51.42	41324000	51.00	-0.001553	51.498571
45	40.83	42.67	40.40	41.11	196352000	40.78	-0.187559	50.447857
46	35.83	37.40	35.70	36.25	41196800	35.96	0.002004	45.782857
47	39.44	40.97	38.93	40.86	72956400	40.57	0.038574	41.170714
48	39.62	39.72	36.40	36.90	68167600	36.64	-0.067471	38.429286
49	36.80	37.65	36.38	36.61	47182000	36.35	-0.016388	38.377857
50	34.04	34.38	32.12	32.39	86374000	32.16	-0.040933	36.721429
51	33.00	34.35	32.58	33.41	62872800	33.18	0.018907	34.806429
52	33.85	34.18	33.06	33.38	46514000	33.14	-0.001122	34.480714

	Long_MA	ture_label
0	50.1008	green
1	50.6628	red
2	51.0022	green
3	51.6632	green
4	52.4006	red
5	52.6238	green
6	53.7292	red
7	54.7766	green
8	56.0202	green

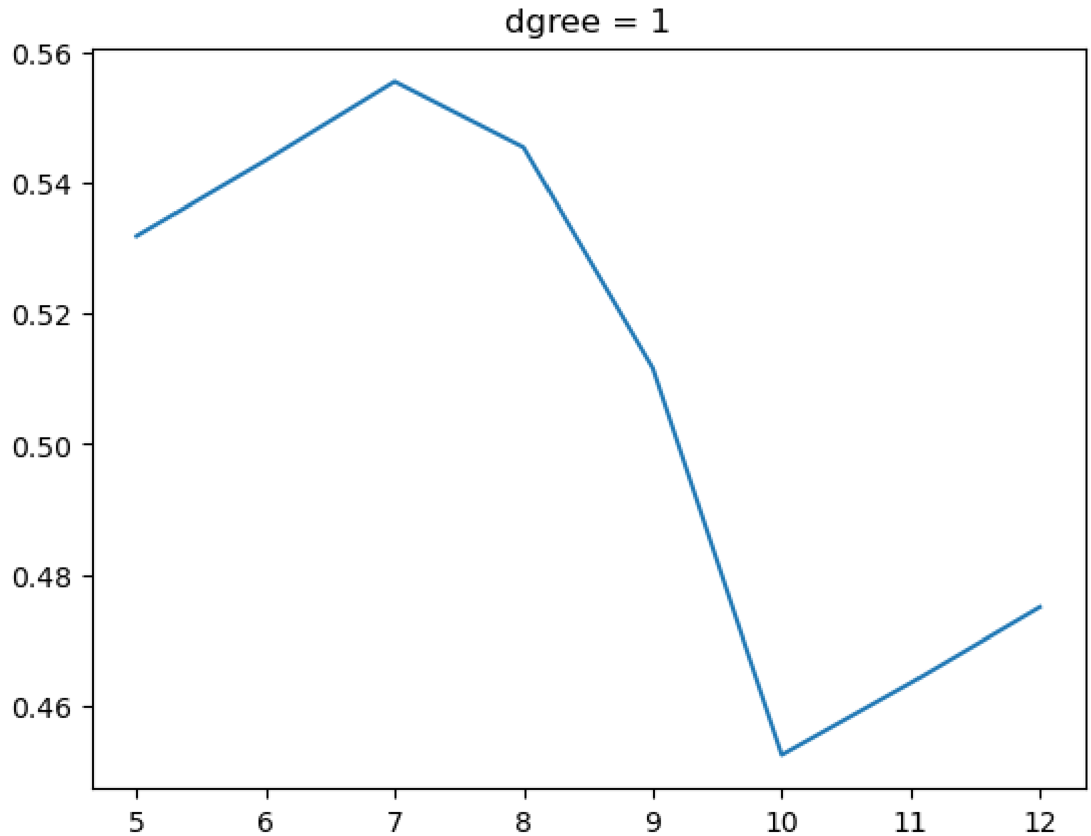
```
9   57.1360      green
10  58.3582      green
11  58.9430        red
12  59.0994      green
13  58.8320        red
14  58.4424        red
15  58.5502        red
16  58.3634      green
17  58.0064      green
18  58.2874        red
19  58.5016        red
20  58.4572      green
21  58.5680      green
22  59.3354        red
23  60.2914        red
24  61.1730        red
25  61.3188        red
26  61.7116      green
27  62.3132        red
28  62.4638        red
29  62.4442        red
30  62.4970      green
31  62.6454        red
32  62.4634        red
33  62.4052      green
34  62.7908      green
35  63.4822        red
36  64.2984      green
37  64.7880        red
38  65.2600      green
39  66.0370        red
40  66.1424      green
41  65.7244        red
42  64.5820        red
43  63.3794        red
44  61.7426        red
45  59.5878        red
46  57.0902      green
47  54.2546      green
48  52.1180        red
49  48.8152        red
50  45.6818        red
51  43.5128      green
52  42.9702        red
```

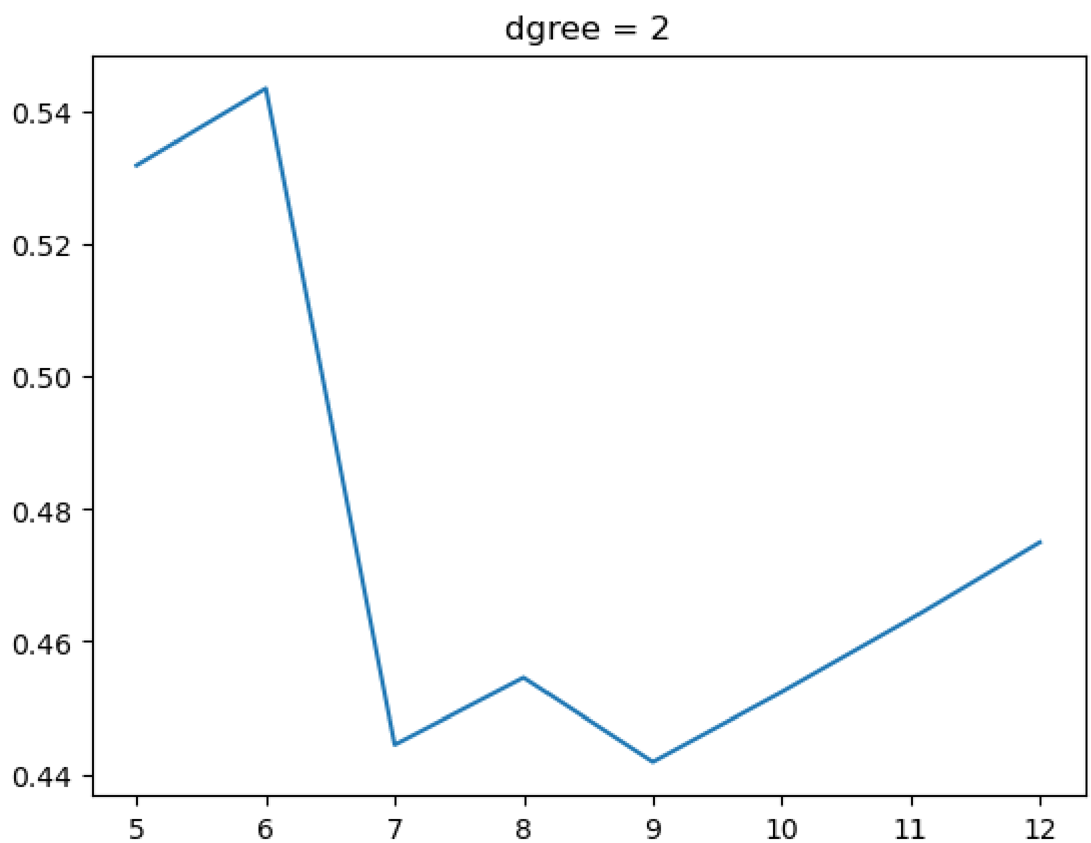
```
/var/folders/by/3jf2bh0x2ks63rycd0ctbzsh0000gn/T/ipykernel_6093/1886444954.py:
7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_week_2017["ture_label"]=df_week_2017['Return'].apply(lambda x:'green' if
x>0 else 'red')
```

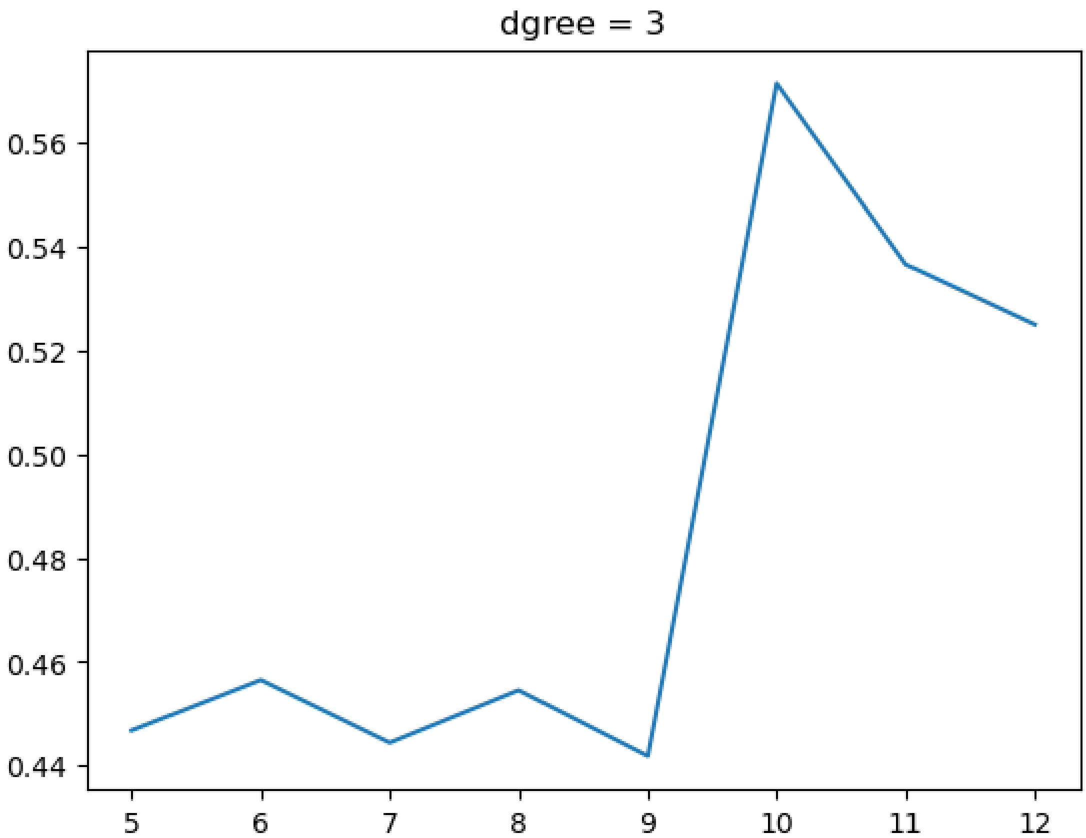
1. take weekly data for year 1. For each W = 5,6,...,12 and for each d = 1, 2, 3 construct the corresponding polynomials Use these polynomials to predict weekly labels. Plot the accuracy - on x axis you have W and you plot three curves for accuracy (separate curve for each d)

In [383…

```
def q1_predect(w,d,df_week):
    X_train = list(range(0,w))
    X_test = list(range(w,len(df_week)))
    y_train = df_week['Close'][:w]
    y_test = df_week['Close'][w:]
    weights= np.polyfit(X_train,y_train, d)
    #print(weights)
    model = np.poly1d(weights)
    predict=model(X_test)
    pred_arr=[]
    for i in range(len(predict)):
        if predict[i]>y_train[w-1]:
            pred_arr.append('green')
        elif predict[i]<y_train[w-1]:
            pred_arr.append('red')
        else:
             pred_arr.append(X_train[w-1])
    res=accuracy_score(df_week[w:]['ture_label'],pred_arr)
    return(res)
degree_1=[]
degree_2=[]
degree_3=[]
week_list=list(range(5,13))
for i in week_list:
    degree_1.append(q1_predect(i,1,df_week_2017))
    degree_2.append(q1_predect(i,2,df_week_2017))
    degree_3.append(q1_predect(i,3,df_week_2017))
plt.plot(week_list,degree_1)
plt.title('dgree = 1')
plt.show()
plt.plot(week_list,degree_2)
plt.title('dgree = 2')
plt.show()
plt.plot(week_list,degree_3)
plt.title('dgree = 3')
plt.show()
```


dgree = 1
0.56
0.54
0.52
0.50
0.48
0.46
5
6
7
8
9
10
11
12


dgree = 2
0.54
0.52
0.50
0.48
0.46
0.44
5
6
7
8
9
10
11
12



1. for each d take the best W that gives you the highest accu- racy. Use this W to predict labels for year 2. What is your accuracy?

In [384…

```
print('for year 2,the degree = 1,the best w in year 1 is 7, and the accurcy is
print('for year 2,the degree = 2,the best w is year 1 is 6, and the accurcy is
print('for year 2,the degree = 3,the best w is year 1 is 10, and the accurcy is
```

```
for year 2,the degree = 1,the best w in year 1 is 7, and the accurcy is  0.413
04347826086957
for year 2,the degree = 2,the best w is year 1 is 6, and the accurcy is  0.595
7446808510638
for year 2,the degree = 3,the best w is year 1 is 10, and the accurcy is  0.37
209302325581395
```

In [385…

```
def q3_predect(w,d,df_week):
    X_train = list(range(0,w))
    X_test = list(range(w,len(df_week)))
    y_train = df_week['Close'][:w]
    y_test = df_week['Close'][w:]
    weights= np.polyfit(X_train,y_train, d)
    #print(weights)
    model = np.poly1d(weights)
    predict=model(X_test)
    pred_arr=[]
    for i in range(len(predict)):
        if predict[i]>y_train[w-1]:
            pred_arr.append('green')
        elif predict[i]<y_train[w-1]:
            pred_arr.append('red')
        else:
             pred_arr.append(X_train[w-1])
```

```
    temp=np.array(df_week[w:]['ture_label'])
    TN,FP,FN,TP=confusion_matrix(temp,np.array(pred_arr)).ravel()
    return(TN, FP, FN, TP)
for i in week_list:
    for j in [1,2,3]:
        print('degree:',j,"week",i,"TN, FP, FN, TP:",q3_predect(i,j,df_week_201
```

```
degree: 1 week 5 TN, FP, FN, TP: (20, 0, 28, 0)
degree: 2 week 5 TN, FP, FN, TP: (0, 20, 0, 28)
degree: 3 week 5 TN, FP, FN, TP: (0, 20, 0, 28)
degree: 1 week 6 TN, FP, FN, TP: (19, 0, 28, 0)
degree: 2 week 6 TN, FP, FN, TP: (0, 19, 0, 28)
degree: 3 week 6 TN, FP, FN, TP: (0, 19, 0, 28)
degree: 1 week 7 TN, FP, FN, TP: (19, 0, 27, 0)
degree: 2 week 7 TN, FP, FN, TP: (0, 19, 0, 27)
degree: 3 week 7 TN, FP, FN, TP: (19, 0, 27, 0)
degree: 1 week 8 TN, FP, FN, TP: (18, 0, 27, 0)
degree: 2 week 8 TN, FP, FN, TP: (0, 18, 0, 27)
degree: 3 week 8 TN, FP, FN, TP: (18, 0, 27, 0)
degree: 1 week 9 TN, FP, FN, TP: (17, 0, 27, 0)
degree: 2 week 9 TN, FP, FN, TP: (0, 17, 0, 27)
degree: 3 week 9 TN, FP, FN, TP: (17, 0, 27, 0)
degree: 1 week 10 TN, FP, FN, TP: (16, 0, 27, 0)
degree: 2 week 10 TN, FP, FN, TP: (0, 16, 0, 27)
degree: 3 week 10 TN, FP, FN, TP: (16, 0, 27, 0)
degree: 1 week 11 TN, FP, FN, TP: (15, 0, 27, 0)
degree: 2 week 11 TN, FP, FN, TP: (0, 15, 0, 27)
degree: 3 week 11 TN, FP, FN, TP: (15, 0, 27, 0)
degree: 1 week 12 TN, FP, FN, TP: (15, 0, 26, 0)
degree: 2 week 12 TN, FP, FN, TP: (1, 14, 0, 26)
degree: 3 week 12 TN, FP, FN, TP: (1, 14, 0, 26)
```

1. implement three trading strategies for year 2 (for each d using the "best" values for W from year 1 that you have computed)

In [386…

```
def q4_predect(w,d,df_week):
    X_train = list(range(0,w))
    X_test = list(range(w,len(df_week)))
    y_train = df_week['Close'][:w]
    y_test = df_week['Close'][w:]
    weights= np.polyfit(X_train,y_train, d)
    #print(weights)
    return(weights)
print(q4_predect(7,1,df_week_2018))
print(q4_predect(6,2,df_week_2018))
print(q4_predect(10,3,df_week_2018))
```

```
[ 0.955       55.03642857]
[-0.50375     3.43503571 53.42178571]
[ 2.50077700e-02 -4.48059441e-01  2.81864608e+00  5.37467133e+01]
```