In [123…

```
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
from sklearn.linear_model import LinearRegression
import warnings
from sklearn.metrics import mean_squared_error
from scipy.stats import f as fisher_f
warnings.filterwarnings('ignore')
df=pd.read_csv('NVDA.csv')
df_week_2017=df.loc[df['Year']==2017,:]
df_week_2018=df.loc[df['Year']==2018,:].reset_index()
month_arr=np.array(range(13))


def get_sse(df,k,month):
    month_len=df.loc[df["Month"]==month,:]["Adj Close"].count()
    month_df=df.loc[df["Month"]==month,:]
    day_arr=np.array(month_len)
    cut_front_df=month_df[:k]
    cut_back_df=month_df[k:]
    a=np.array(range(k)).reshape(-1,1)
    b=np.array(range(k,month_len)).reshape(-1,1)
    c=np.array(range(month_len)).reshape(-1,1)
    # a=np.array(range(10)).reshape(-1,1)
    #regr = LinearRegression()
    regr_front=LinearRegression().fit(a, cut_front_df["Adj Close"])
    regr_back=LinearRegression().fit(b, cut_back_df["Adj Close"])
    regr_orig=LinearRegression().fit(c, month_df["Adj Close"])
    #print(mean_squared_error(a,cut_front_df["Adj Close"])+mean_squared_error(b
    sse_front = mean_squared_error(cut_front_df["Adj Close"],regr_front.predict
    sse_back = mean_squared_error(cut_back_df["Adj Close"],regr_back.predict(b)
    sse_total=sse_front+sse_back
    sse_orig= mean_squared_error(month_df["Adj Close"],regr_orig.predict(c))/le
    return sse_front,sse_back,sse_total,sse_orig
def get_k(m,df):
    month_len=df.loc[df["Month"]==m,:]["Adj Close"].count()
    k=0
    min_sse=1000000
    p_value=0
    for i in range(1,month_len):
        sse_front,sse_back,sse_total,sse_orig=get_sse(df,i,m)
        #print(i,sse_front,sse_back,sse_total,sse_orig)

        if sse_total<min_sse:
                min_sse=sse_total
                k=i
                F = ((sse_orig - sse_total)/2)/(sse_total/(month_len-4))
                p_value = fisher_f.cdf(F ,2, month_len-4)
                #print(i,sse_total,p_value)
```

```
    #print(k,mid_sse)
    return k,p_value,min_sse
```

1. take years 1 and 2. For each month, compute the ″candi- date″ days and decide whether there is a significant change of pricing trend in each month. Use 0.1 as critical value.

In [124…

```
for i in range(1,13):
    k,p_value,min_sse=get_k(i,df_week_2017)
    print("In 2017,the best k in NVDA at",i,"month is",k,"the p_value is",p_val
for i in range(1,13):
    k,p_value,min_sse=get_k(i,df_week_2018)
    print("In 2018,the best k in NVDA at",i,"month is",k,"the p_value is",p_val
```

```
In 2017,the best k in NVDA at 1 month is 19 the p_value is 0.0 the minimun SSE
is 0.01394602517373766
In 2017,the best k in NVDA at 2 month is 1 the p_value is 0.29496065055301984
the minimun SSE is 0.0273138741309823
In 2017,the best k in NVDA at 3 month is 1 the p_value is 0.9752882878665425 t
he minimun SSE is 0.006264990363571046
In 2017,the best k in NVDA at 4 month is 2 the p_value is 0.9976491841616215 t
he minimun SSE is 0.012659362914716053
In 2017,the best k in NVDA at 5 month is 7 the p_value is 0.999814570266828 th
e minimun SSE is 0.03523430600367122
In 2017,the best k in NVDA at 6 month is 20 the p_value is 0.8263513961834199
the minimun SSE is 0.04983671541353387
In 2017,the best k in NVDA at 7 month is 9 the p_value is 0.9989290395157632 t
he minimun SSE is 0.03783143443412791
In 2017,the best k in NVDA at 8 month is 22 the p_value is 0.19531143273403576
the minimun SSE is 0.035300225161112725
In 2017,the best k in NVDA at 9 month is 15 the p_value is 0.4401722736018504
the minimun SSE is 0.10092594497354496
In 2017,the best k in NVDA at 10 month is 21 the p_value is 0.0 the minimun SS
E is 0.02851579595763274
In 2017,the best k in NVDA at 11 month is 19 the p_value is 0.9999846064394357
the minimun SSE is 0.019964715944987173
In 2017,the best k in NVDA at 12 month is 1 the p_value is 0.938998333620168 t
he minimun SSE is 0.024456624386450892
In 2018,the best k in NVDA at 1 month is 1 the p_value is 0.9569928549921554 t
he minimun SSE is 0.026395981578947274
In 2018,the best k in NVDA at 2 month is 2 the p_value is 0.931332511319953 th
e minimun SSE is 0.11511953914783903
In 2018,the best k in NVDA at 3 month is 19 the p_value is 0.7991998925238537
the minimun SSE is 0.1523978417650774
In 2018,the best k in NVDA at 4 month is 1 the p_value is 0.0 the minimun SSE
is 0.11925988421052625
In 2018,the best k in NVDA at 5 month is 8 the p_value is 0.987527919431964 th
e minimun SSE is 0.0945200339493626
In 2018,the best k in NVDA at 6 month is 9 the p_value is 0.7110651325740263 t
he minimun SSE is 0.10654476960368145
In 2018,the best k in NVDA at 7 month is 19 the p_value is 0.9622944247428371
the minimun SSE is 0.03098222287019491
In 2018,the best k in NVDA at 8 month is 12 the p_value is 0.398026231259641 t
he minimun SSE is 0.1036091698185502
In 2018,the best k in NVDA at 9 month is 18 the p_value is 0.9957719011921956
the minimun SSE is 0.03191828175498038
In 2018,the best k in NVDA at 10 month is 22 the p_value is 0.810217015577737
the minimun SSE is 0.17306261835541376
In 2018,the best k in NVDA at 11 month is 11 the p_value is 0.9182939390539447
the minimun SSE is 0.3891383878787877
In 2018,the best k in NVDA at 12 month is 1 the p_value is 0.8800342261847197
the minimun SSE is 0.04693643716529293
```

1. how many months exhibit significant price changes for your sotck ticker

For 2017 is 2 month, for 2018 is 1 month

3,are there more "changes" in year 1 or in year 2

no