Data Exploring

In [45]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [46]:
```
df = pd.read_csv('winequality-red.csv')
```

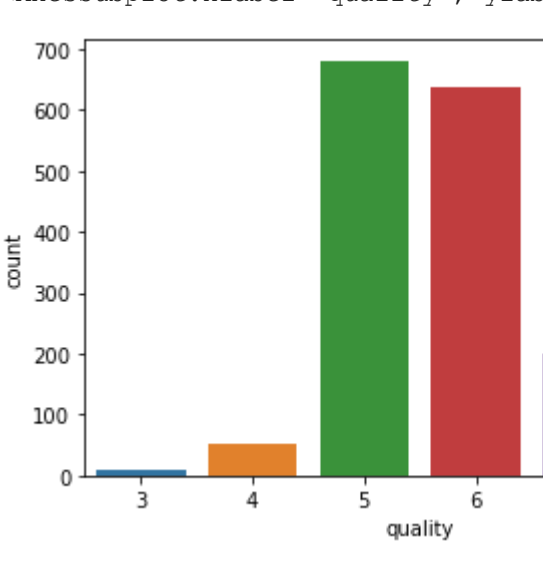
In [47]:
```
df.head()
```

Out[47]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

In [48]:
```
sns.countplot(x=df['quality'])
```

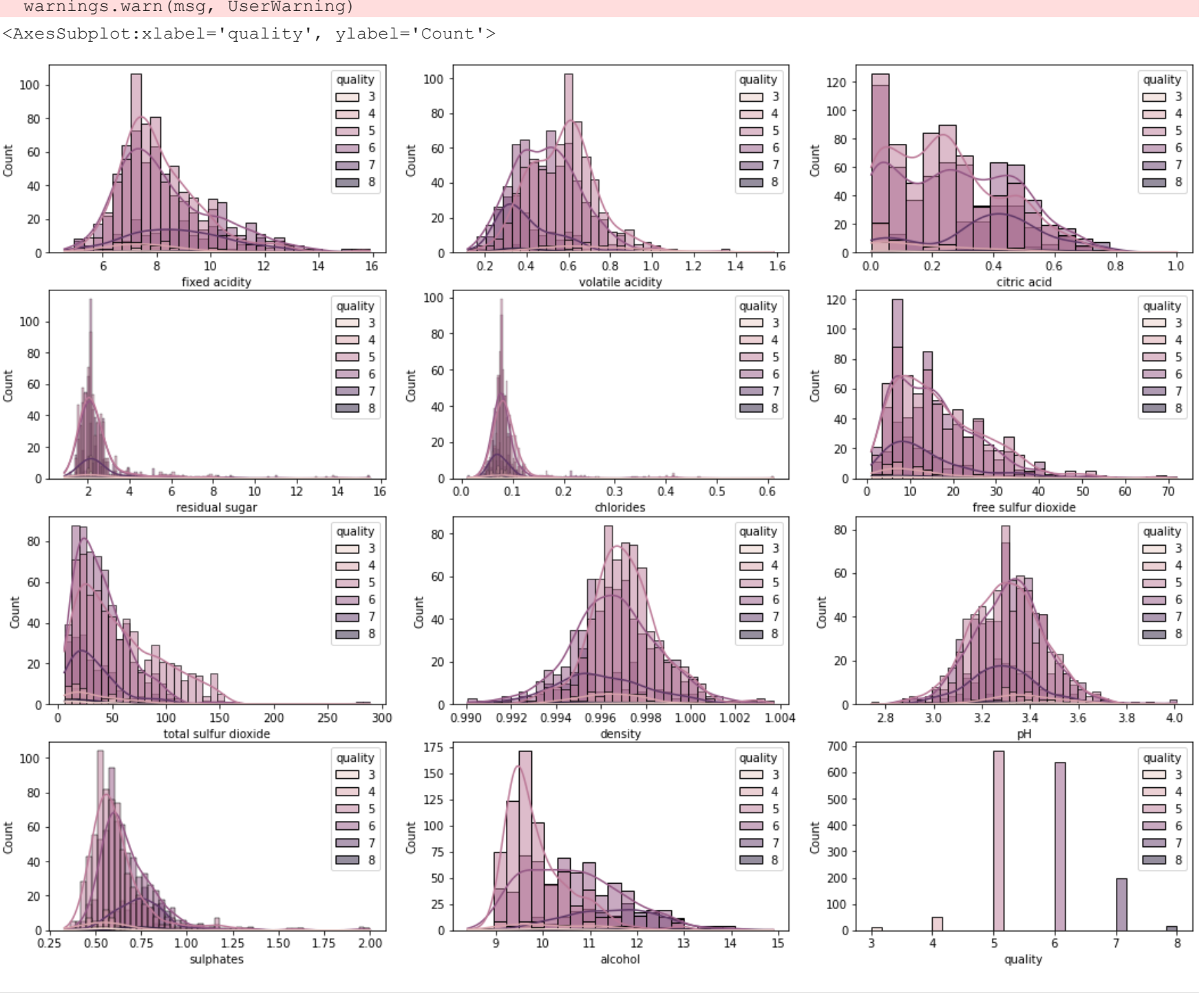
Out[48]: <AxesSubplot:xlabel='quality', ylabel='count'>



In [49]:
```
fig, axes = plt.subplots(nrows=4, ncols=3, figsize=(18, 14))
sns.histplot(data=df, x="fixed acidity", kde=True,ax=axes[0][0],hue='quality')
sns.histplot(data=df, x="volatile acidity", kde=True,ax=axes[0][1],hue='quality')
sns.histplot(data=df, x="citric acid", kde=True,ax=axes[0][2],hue='quality')
sns.histplot(data=df, x="residual sugar", kde=True,ax=axes[1][0],hue='quality')
sns.histplot(data=df, x="chlorides", kde=True,ax=axes[1][1],hue='quality')
sns.histplot(data=df, x="free sulfur dioxide", kde=True,ax=axes[1][2],hue='quality')
sns.histplot(data=df, x="total sulfur dioxide", kde=True,ax=axes[2][0],hue='quality')
sns.histplot(data=df, x="density", kde=True,ax=axes[2][1],hue='quality')
sns.histplot(data=df, x="pH", kde=True,ax=axes[2][2],hue='quality')
sns.histplot(data=df, x="sulphates", kde=True,ax=axes[3][0],hue='quality')
sns.histplot(data=df, x="alcohol", kde=True,ax=axes[3][1],hue='quality')
sns.histplot(data=df, x="quality", kde=True,ax=axes[3][2],hue='quality')
```

```
C:\Users\LENOVO\anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\LENOVO\anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\LENOVO\anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\LENOVO\anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\LENOVO\anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\LENOVO\anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
```

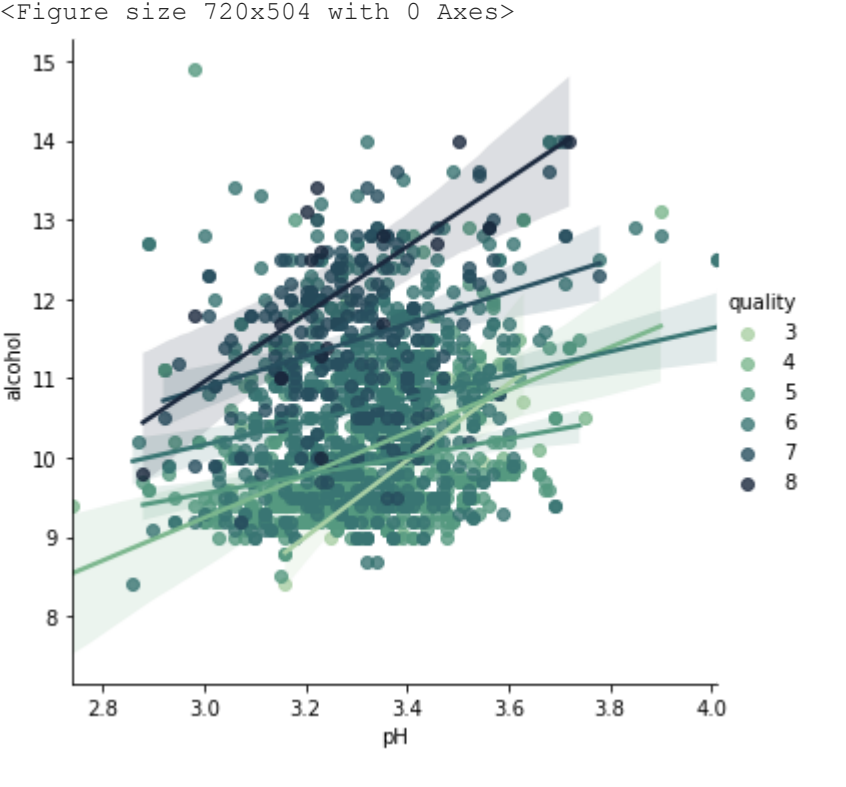
Out[49]: <AxesSubplot:xlabel='quality', ylabel='Count'>



In [93]:
```
plt.figure(figsize=(10,7))
sns.lmplot(x='pH',y='alcohol', data=df, hue='quality', palette="ch:r=-.5,l=.75")
```

Out[93]: <seaborn.axisgrid.FacetGrid at 0x2273026b6d0>

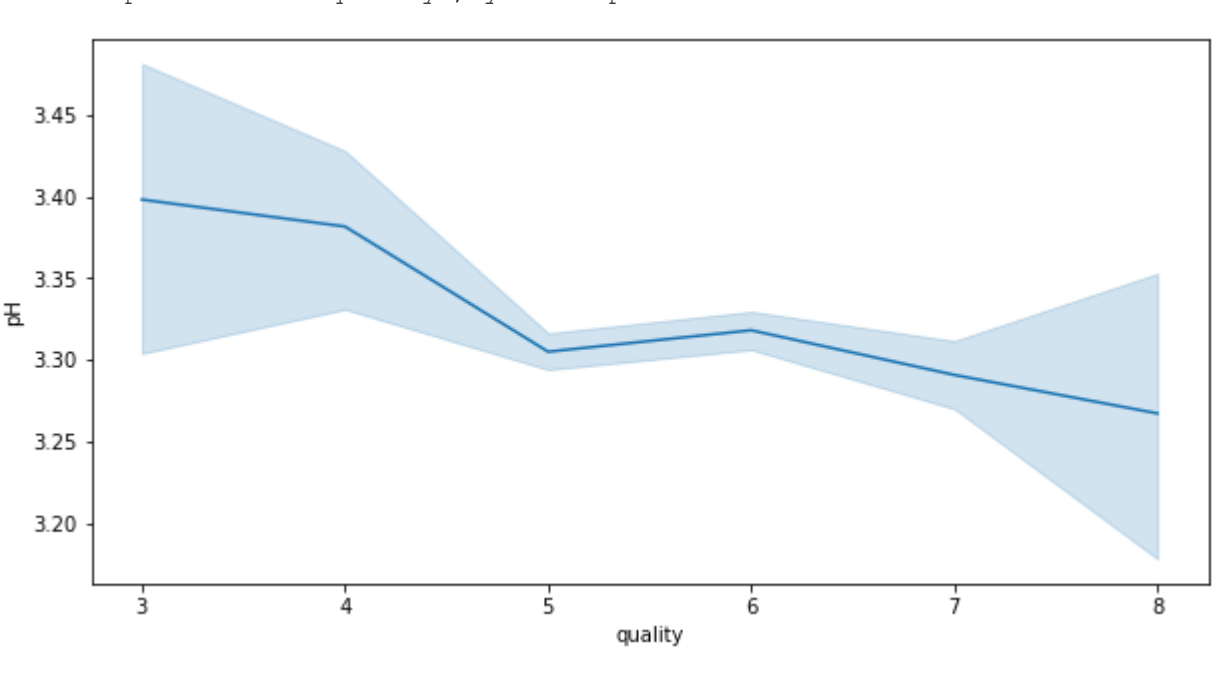
<Figure size 720x504 with 0 Axes>



i cant see any promising relation among them

In [61]:
```
plt.figure(figsize=(10,5))
sns.lineplot(x=df['quality'],y=df['pH'])
```

Out[61]: <AxesSubplot:xlabel='quality', ylabel='pH'>



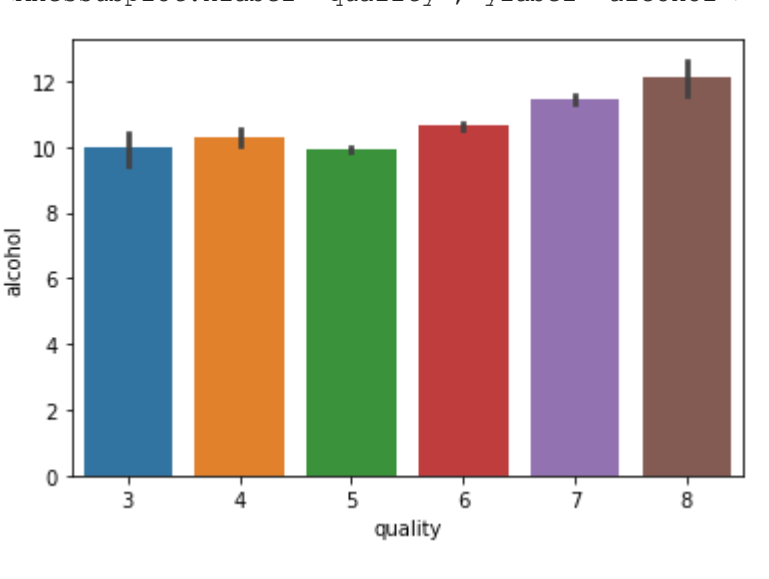
looks like as quality increase ph decreases

In [64]:
```
df.corr()['quality']
```

Out[64]:
```
fixed acidity           0.124052
volatile acidity       -0.390558
citric acid             0.226373
residual sugar          0.013732
chlorides              -0.128907
free sulfur dioxide    -0.050656
total sulfur dioxide   -0.185100
density                -0.174919
pH                     -0.057731
sulphates               0.251397
alcohol                 0.476166
quality                 1.000000
Name: quality, dtype: float64
```

In [69]:
```
sns.barplot(x=df['quality'],y=df['alcohol'])
```

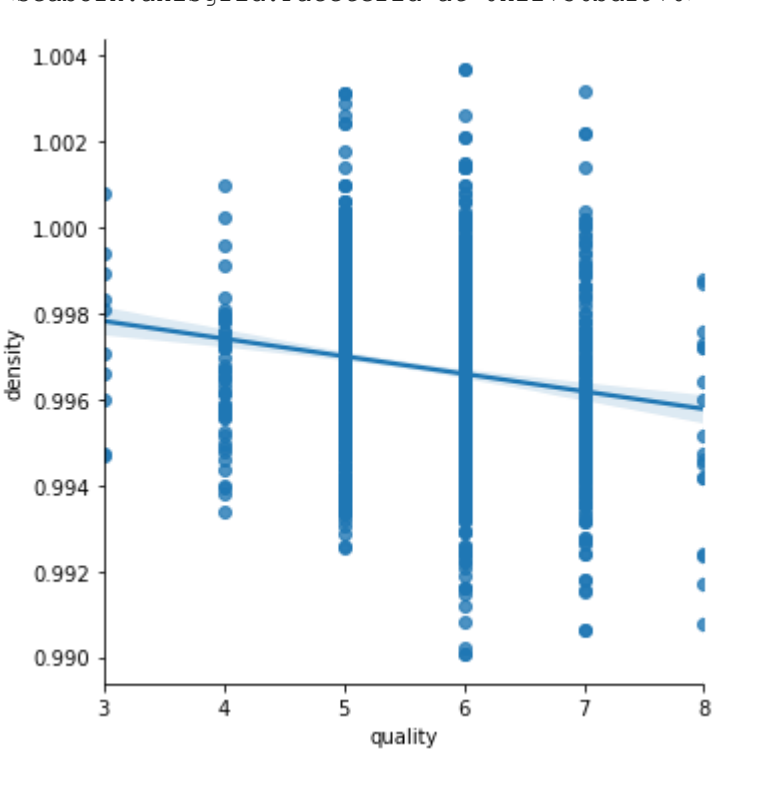
Out[69]: <AxesSubplot:xlabel='quality', ylabel='alcohol'>



looks like alcohol and quality are directly related

In [75]:
```
sns.lmplot(x='quality',y='density',data=df)
```

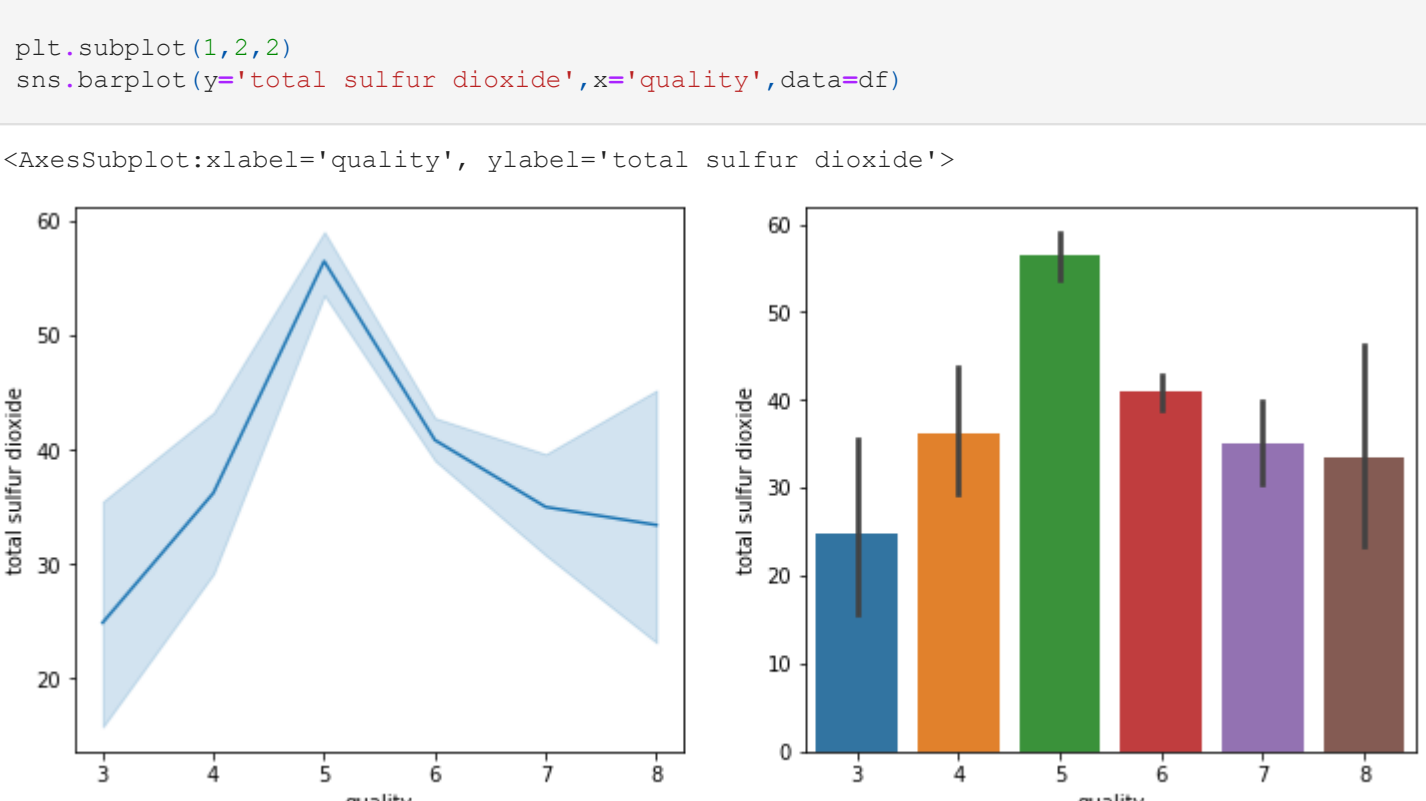
Out[75]: <seaborn.axisgrid.FacetGrid at 0x22736bdf970>



as density decrease quality increases. we are not able to see much varies because the value in variance of density is very less

In [82]:
```
plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
sns.lineplot(y='total sulfur dioxide',x='quality',data=df)

plt.subplot(1,2,2)
sns.barplot(y='total sulfur dioxide',x='quality',data=df)
```

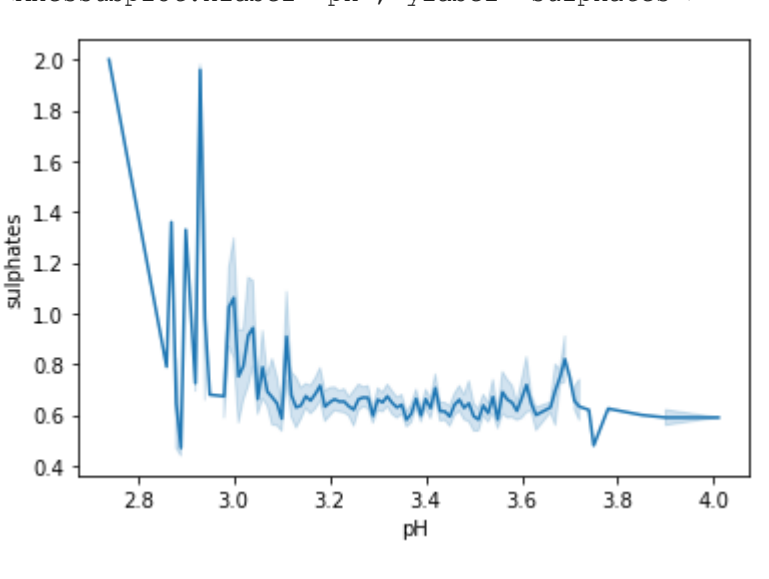
Out[82]: <AxesSubplot:xlabel='quality', ylabel='total sulfur dioxide'>



i dont see any promissing relation between them

In [95]:
```
sns.lineplot(y='sulphates',x='pH',data=df)
```

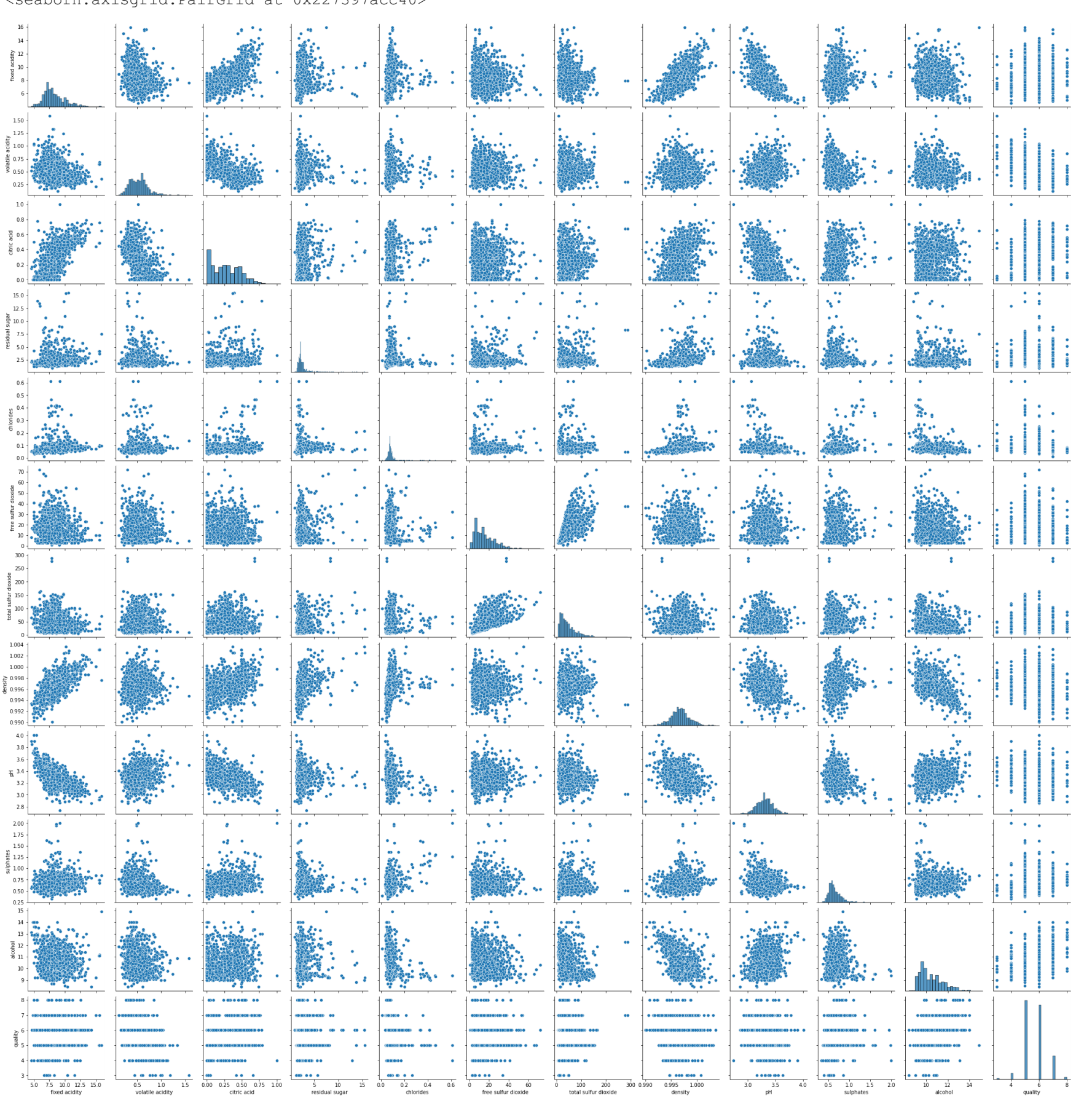
Out[95]: <AxesSubplot:xlabel='pH', ylabel='sulphates'>



we can clearly see as the sulphate value decreases pH increases look like a negative realtion among them

In [97]:
```
sns.pairplot(df)
```

Out[97]: <seaborn.axisgrid.PairGrid at 0x227397acc40>



> fixed acidity had denisty have a direct relation.
>
> simiraly fixed acidity and pH have negative relation.
>
> looks like alcohol and density also has a negative relation

In [103…
```
df.isna().sum()
```

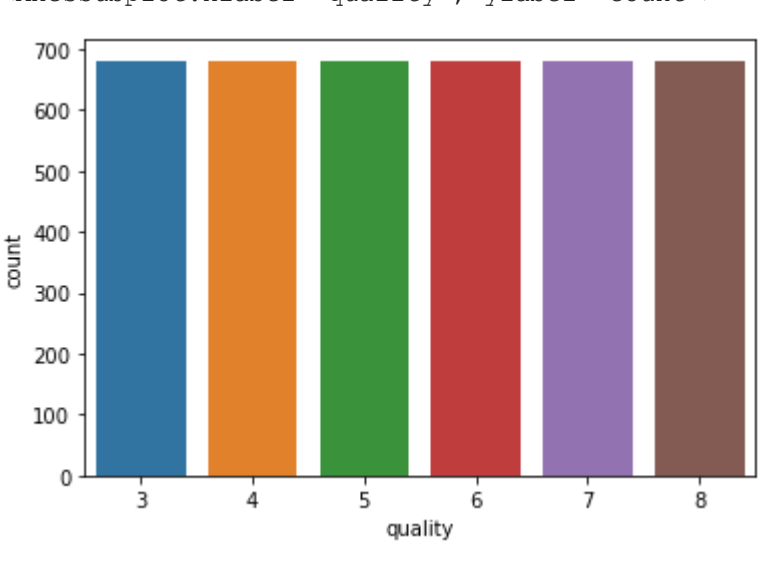
Out[103…
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

Handling imbalance

In [98]:
```
from sklearn.utils import resample,shuffle

five =df[df['quality']==5]
six = df[df['quality']==6]
three = df[df['quality']==3]
four = df[df['quality']==4]
seven = df[df['quality']==7]
eight = df[df['quality']==8]

upsampled1 = resample(three, replace=True, n_samples=five.shape[0])
upsampled2 = resample(four, replace=True, n_samples=five.shape[0])
upsampled3 = resample(six, replace=True, n_samples=five.shape[0])
upsampled4 = resample(seven, replace=True, n_samples=five.shape[0])
upsampled5 = resample(eight, replace=True, n_samples=five.shape[0])

df = pd.concat([five,upsampled1,upsampled2,upsampled3,upsampled4,upsampled5])

df = shuffle(df)
```

In [100…
```
sns.countplot(x=df['quality'])
```

Out[100… <AxesSubplot:xlabel='quality', ylabel='count'>



Model Building

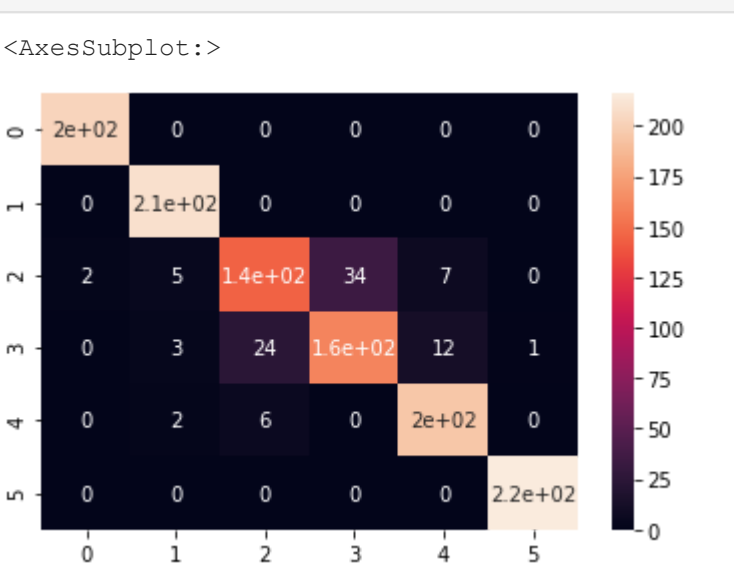
In [101…
```
x = df.drop('quality',axis=1)
y = df['quality']
```

In [104…
```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(x,y,train_size=0.7,random_state=101)
```

1)Decision Tree

In [106…
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,confusion_matrix

dt = DecisionTreeClassifier()
dt.fit(x_train,y_train)
dt_predict = dt.predict(x_test)
dt_cm = confusion_matrix(y_test,dt_predict)
sns.heatmap(dt_cm,annot=True)
```

Out[106… <AxesSubplot:>



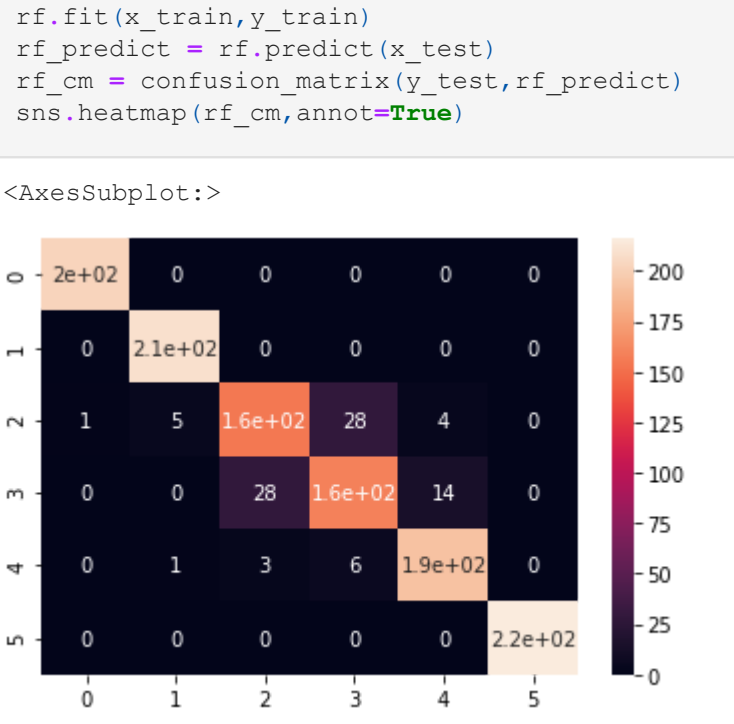
In [107…
```
accuracy_score(y_test,dt_predict)
```

Out[107… 0.9216965742251223

2)Random forest

In [124…
```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()
rf.fit(x_train,y_train)
rf_predict = rf.predict(x_test)
rf_cm = confusion_matrix(y_test,rf_predict)
sns.heatmap(rf_cm,annot=True)
```

Out[124… <AxesSubplot:>



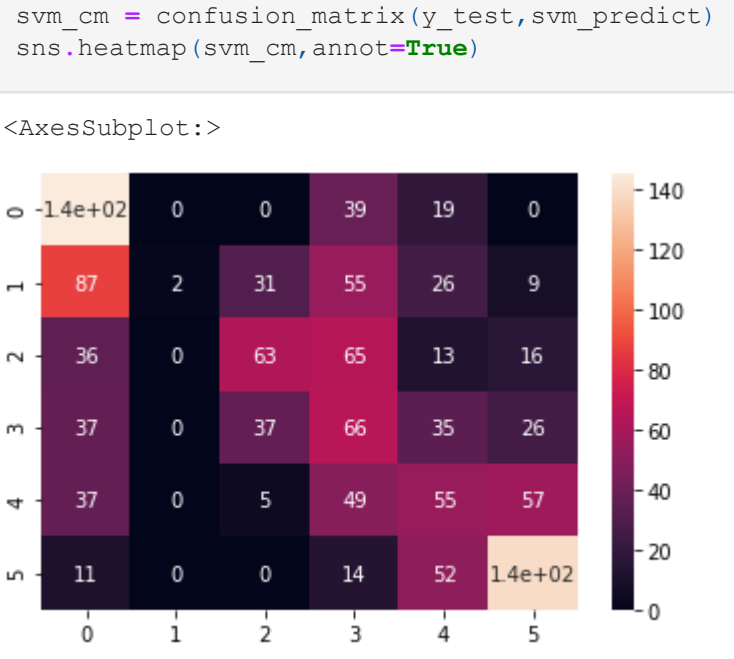
In [125…
```
accuracy_score(y_test,rf_predict)
```

Out[125… 0.9265905383360522

3)support vector machine

In [110…
```
from sklearn.svm import SVC

svm = SVC()
svm.fit(x_train,y_train)
svm_predict = svm.predict(x_test)
svm_cm = confusion_matrix(y_test,svm_predict)
sns.heatmap(svm_cm,annot=True)
```

Out[110… <AxesSubplot:>



In [111…
```
accuracy_score(y_test,svm_predict)
```

Out[111… 0.3833605220228385

4)K-Nearest Neighbor

from sklearn.neighbors import KNeighborsClassifier

kn = KNeighborsClassifier(n_neighbors=4) kn.fit(x_train,y_train) kn_predict = kn.predict(x_test) kn_cm = confusion_matrix(y_test,kn_predict) sns.heatmap(kn_cm,annot=True)

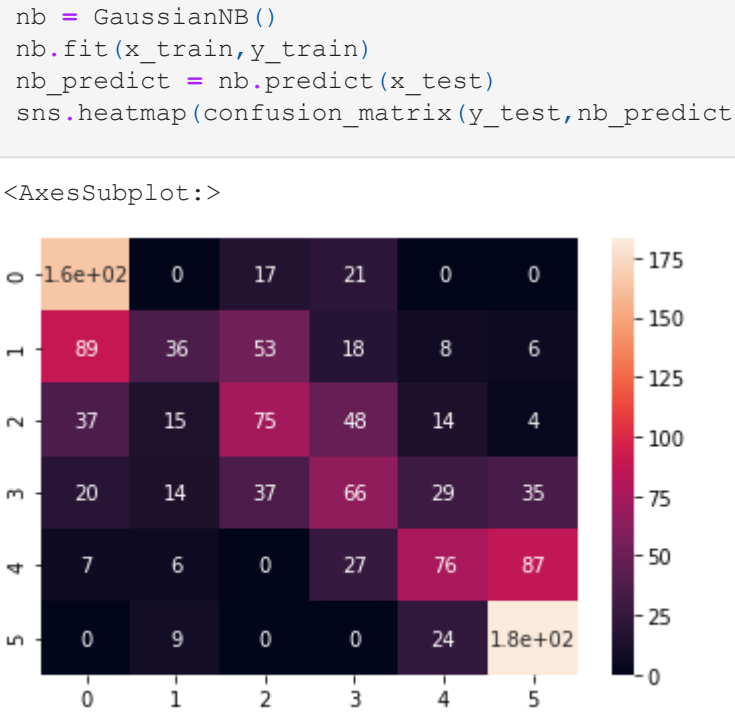
In [123…
```
accuracy_score(y_test,kn_predict)
```

Out[123… 0.800978792822186

5)Naive Bayes

In [126…
```
from sklearn.naive_bayes import GaussianNB

nb = GaussianNB()
nb.fit(x_train,y_train)
nb_predict = nb.predict(x_test)
sns.heatmap(confusion_matrix(y_test,nb_predict),annot=True)
```

Out[126… <AxesSubplot:>



In [127…
```
accuracy_score(y_test,nb_predict)
```

Out[127… 0.4902120717781403

from all the models above Random forest is the best one as it accuracy score is 92.65%

In [ ]: