Importing the Dependencies

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

Data Collection

```
# loading the dataset to a Pandas DataFrame
wine_dataset = pd.read_csv('/content/winequality-red.csv')
# number of rows & columns in the dataset
wine_dataset.shape
```

```
(1599, 12)
```

```
# first 5 rows of the dataset
wine_dataset.head()
```

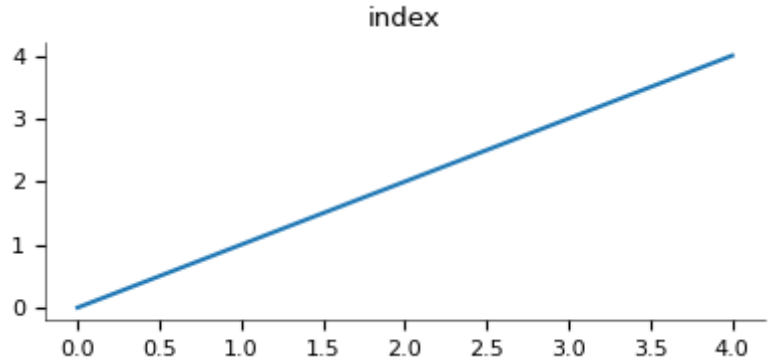
1 to 5 of 5 entries Filter

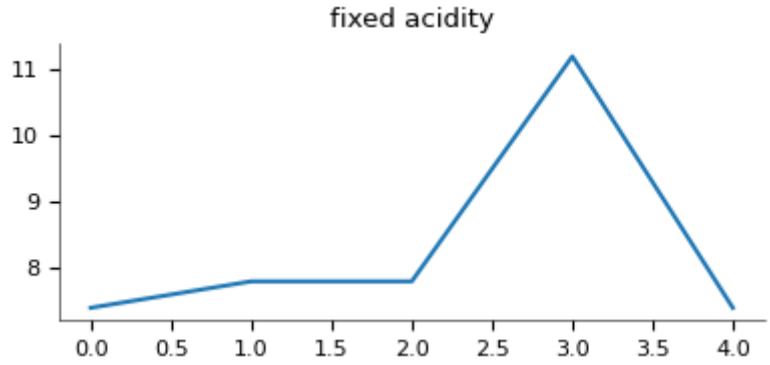
index	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total s
0	7.4	0.7	0.0	1.9	0.076	11.0	
1	7.8	0.88	0.0	2.6	0.098	25.0	
2	7.8	0.76	0.04	2.3	0.092	15.0	
3	11.2	0.28	0.56	1.9	0.075	17.0	
4	7.4	0.7	0.0	1.9	0.076	11.0	

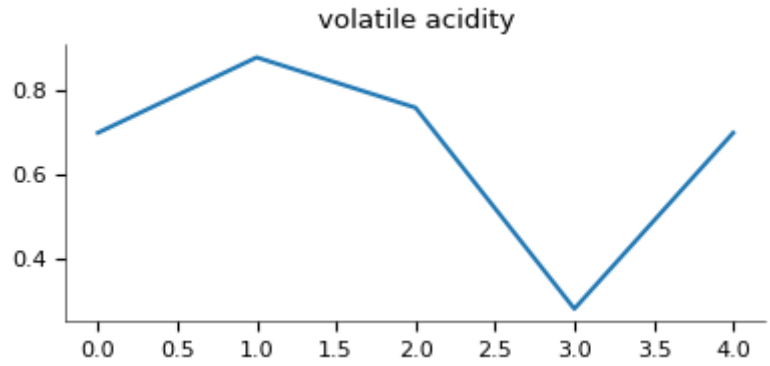
Show 25 per page

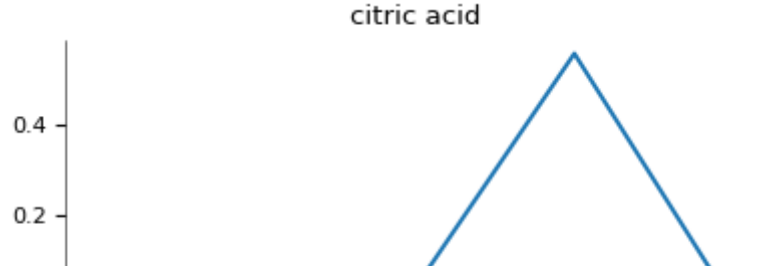
Like what you see? Visit the data table notebook to learn more about interactive tables.

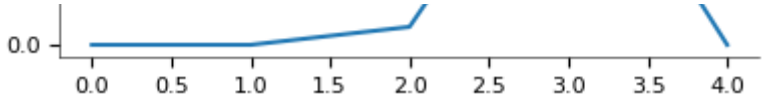
Values



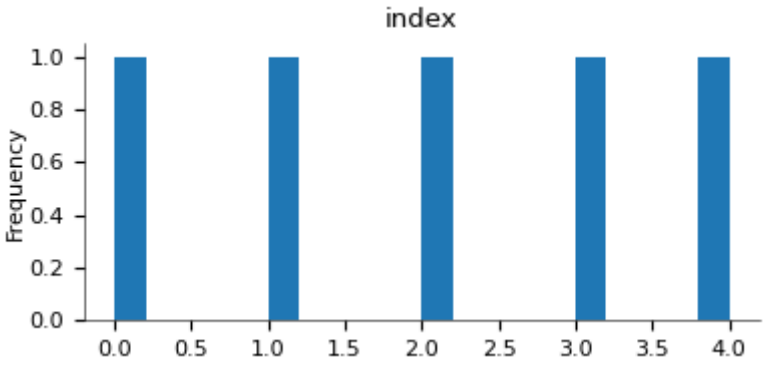


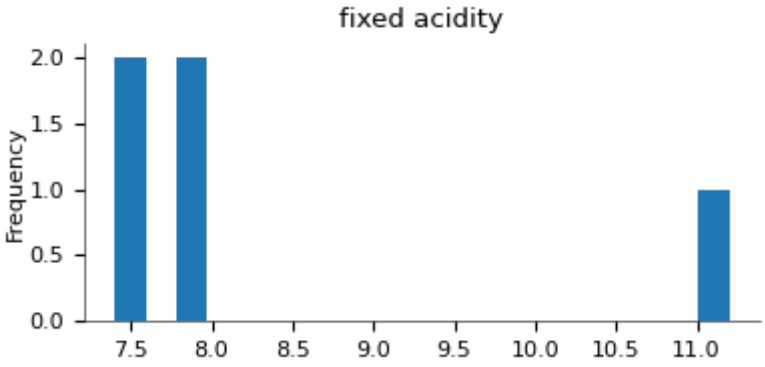


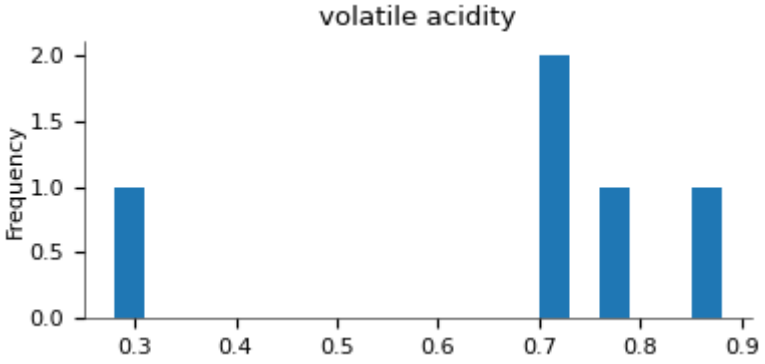


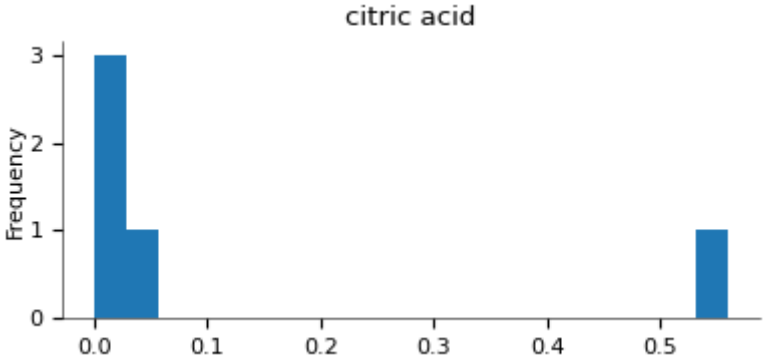


Distributions

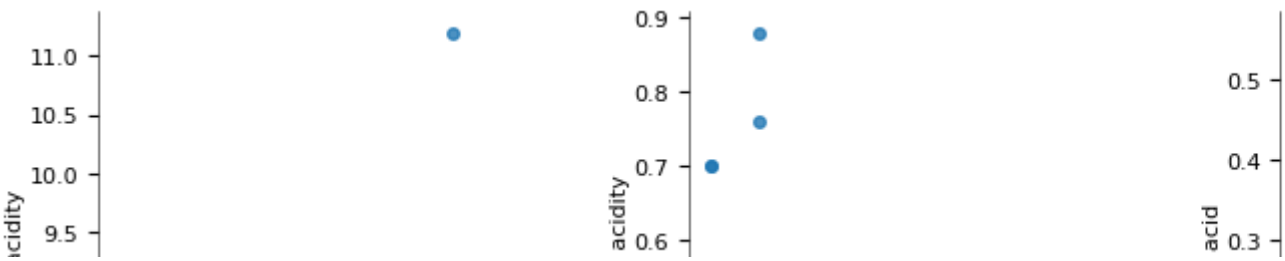


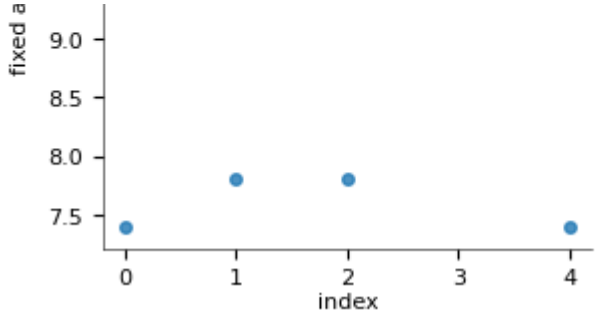


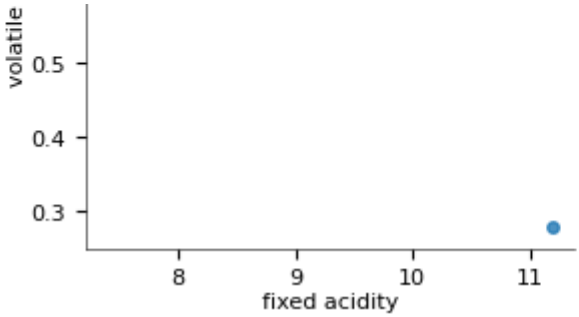


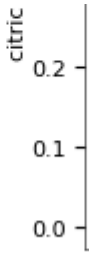


2-d distributions

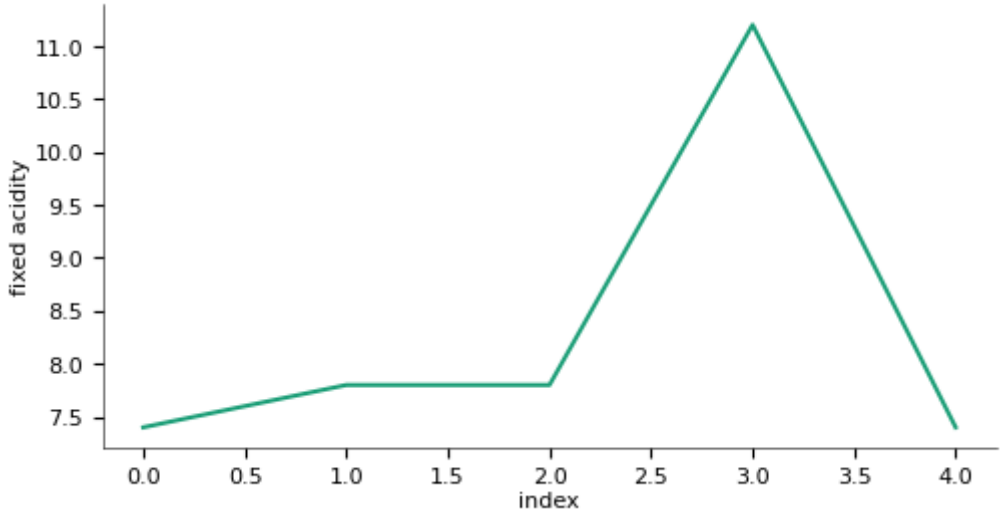


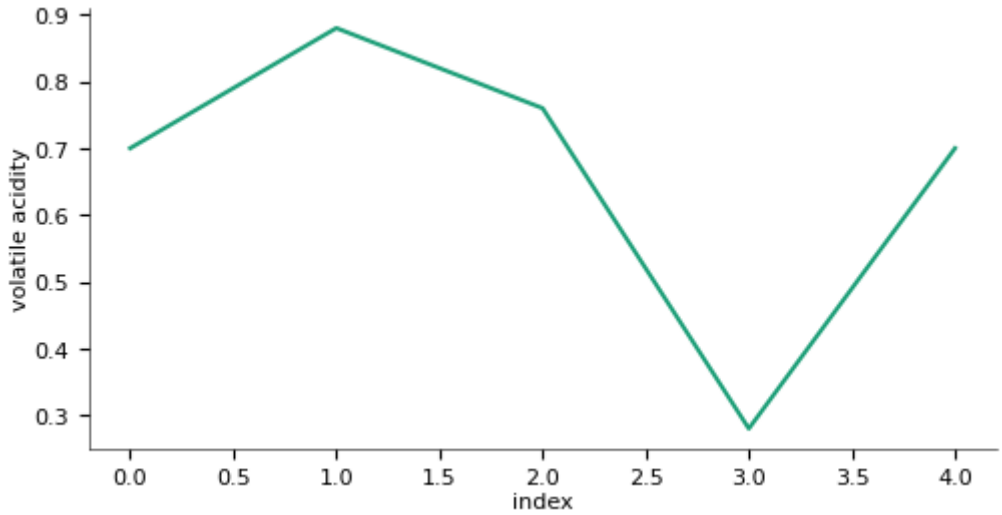






Time series





```
# checking for missing values
wine_dataset.isnull().sum()
```

```
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
```

Data Analysis and Visulaization

```
# statistical measures of the dataset
wine_dataset.describe()
```

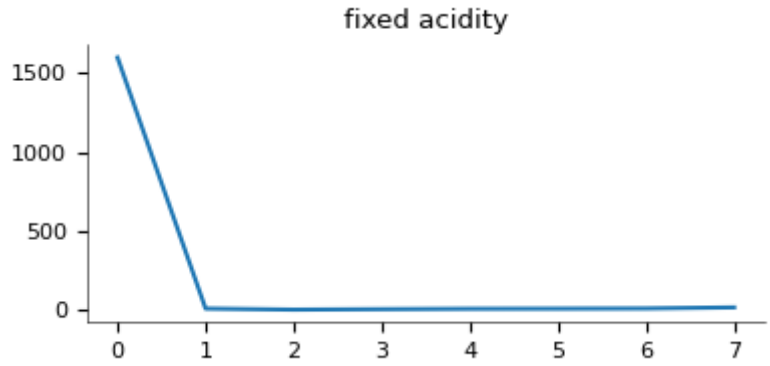
1 to 8 of 8 entries Filter

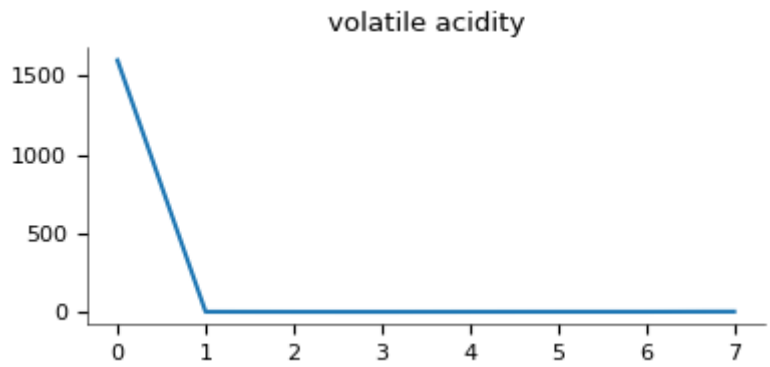
index	fixed acidity	volatile acidity	citric acid	residual sugar	
count	1599.0	1599.0	1599.0	1599.0	
mean	8.31963727329581	0.5278205128205128	0.2709756097560976	2.53880550343965	0.08
std	1.7410963181276953	0.17905970415353537	0.19480113740531857	1.4099280595072798	0.0
min	4.6	0.12	0.0	0.9	
25%	7.1	0.39	0.09	1.9	
50%	7.9	0.52	0.26	2.2	
75%	9.2	0.64	0.42	2.6	
max	15.9	1.58	1.0	15.5	

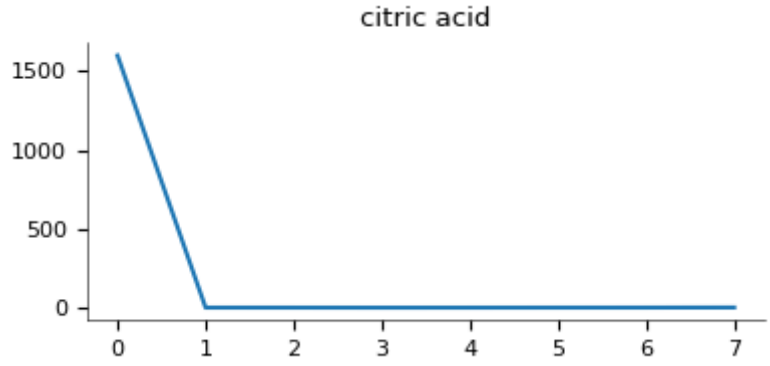
Show 25 per page

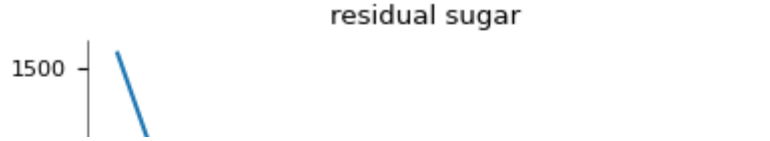
Like what you see? Visit the data table notebook to learn more about interactive tables.

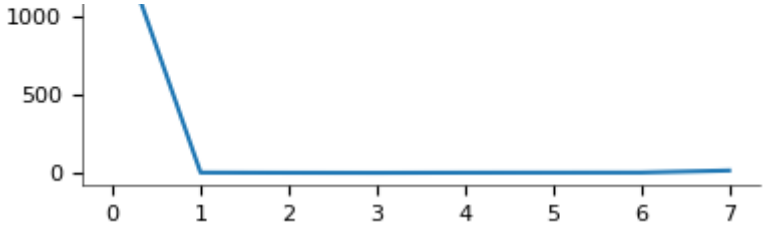
Values



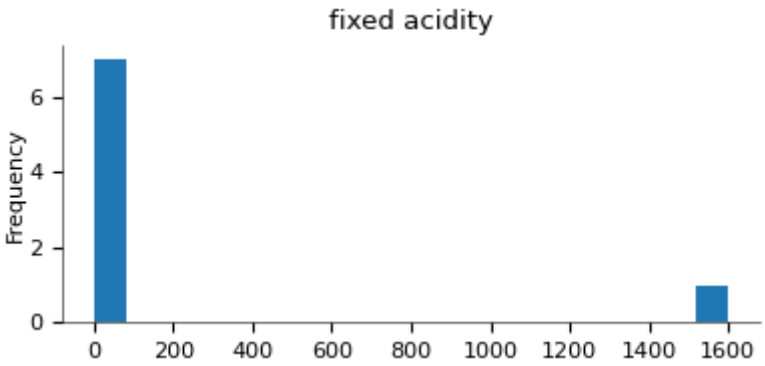


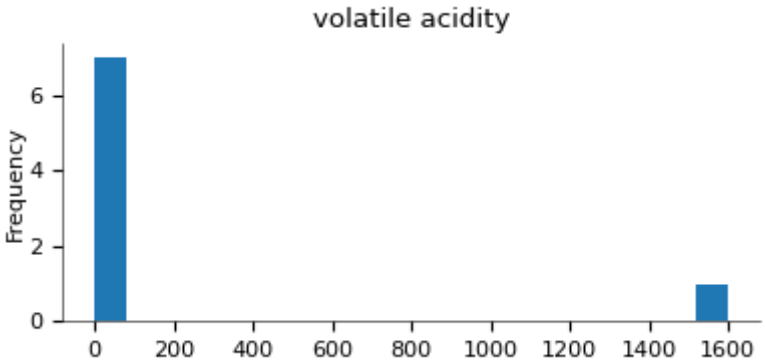


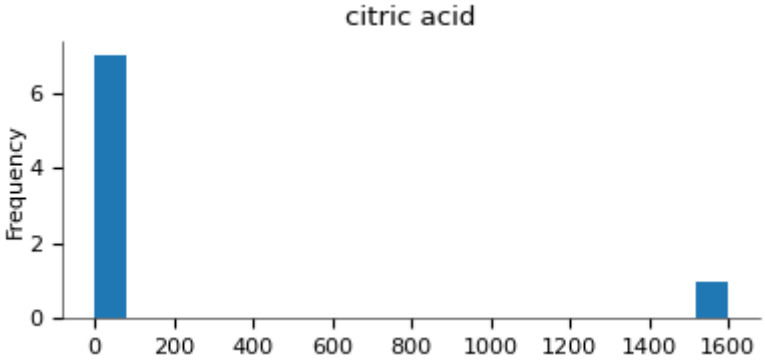


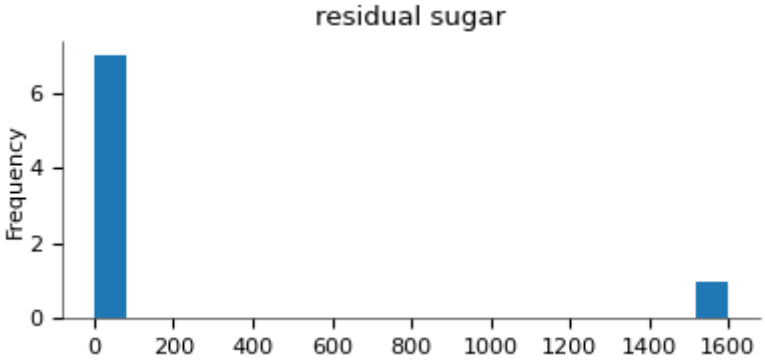


Distributions

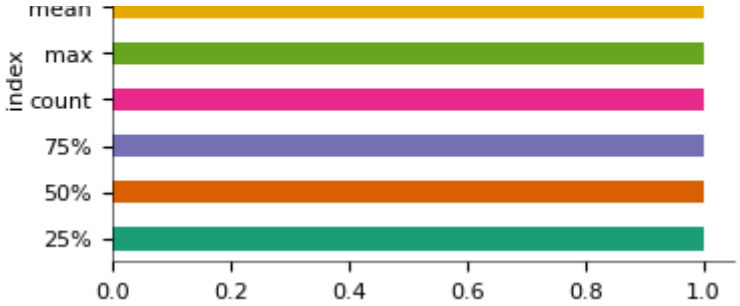




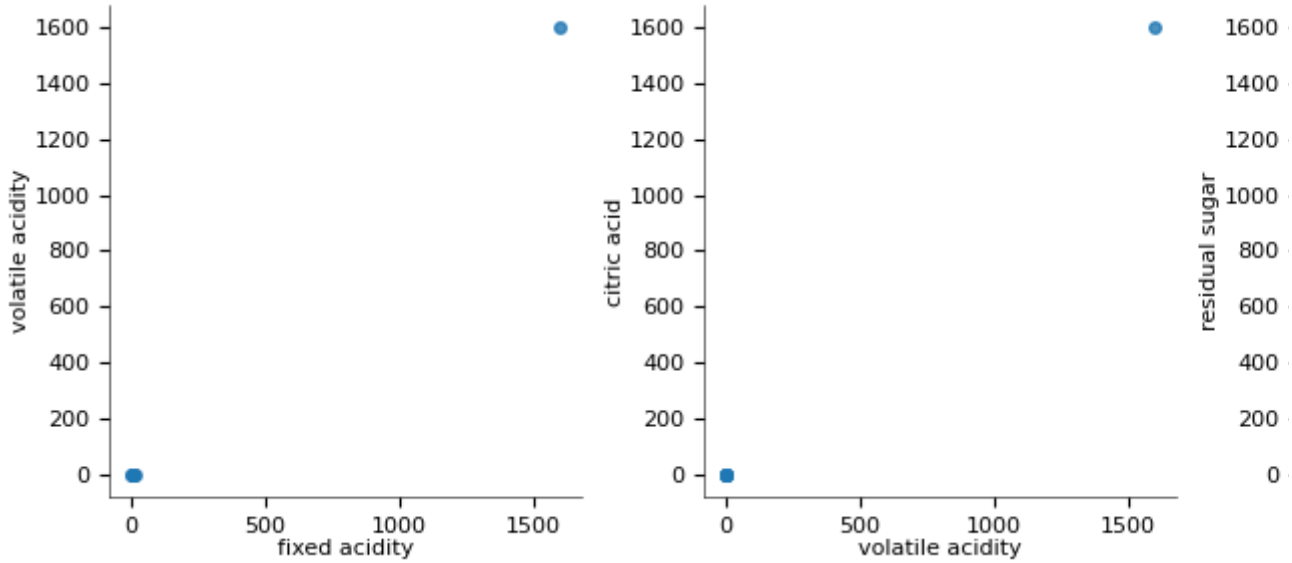




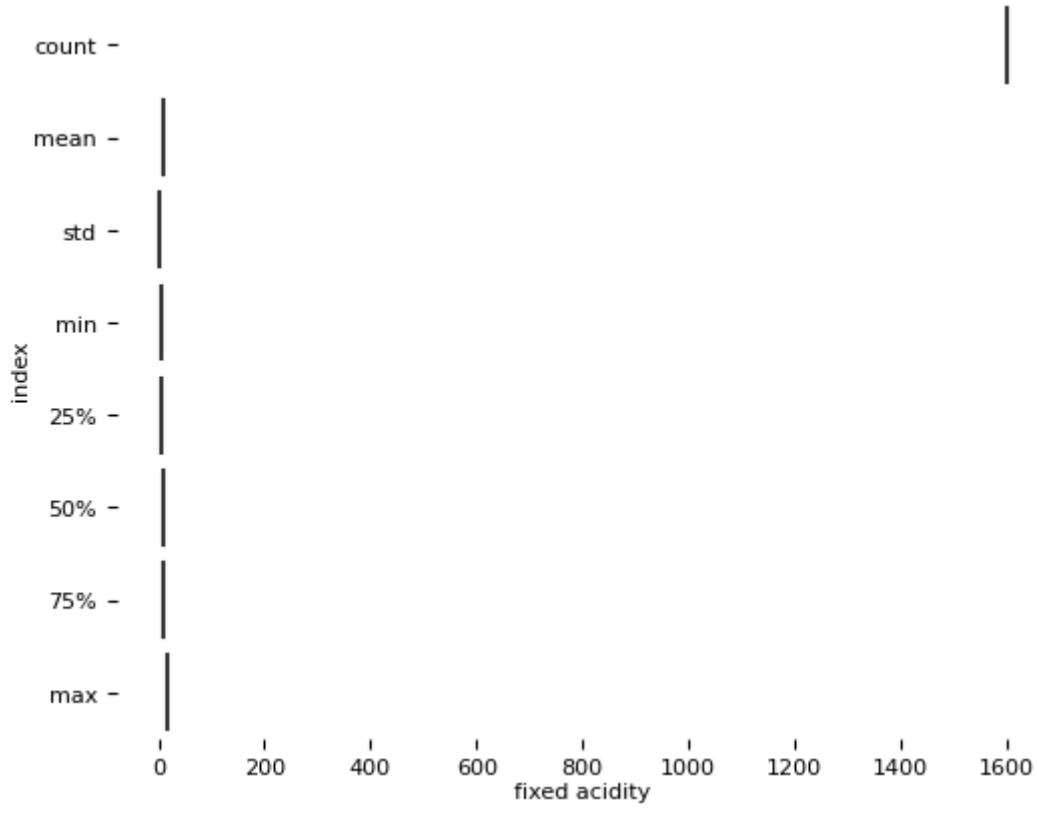
Categorical distributions

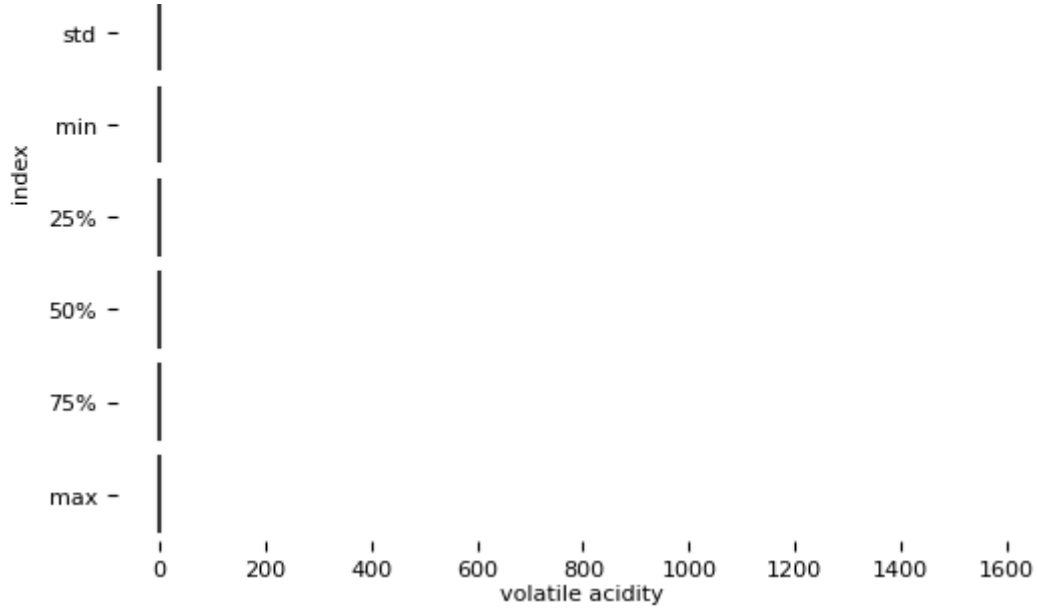


2-d distributions



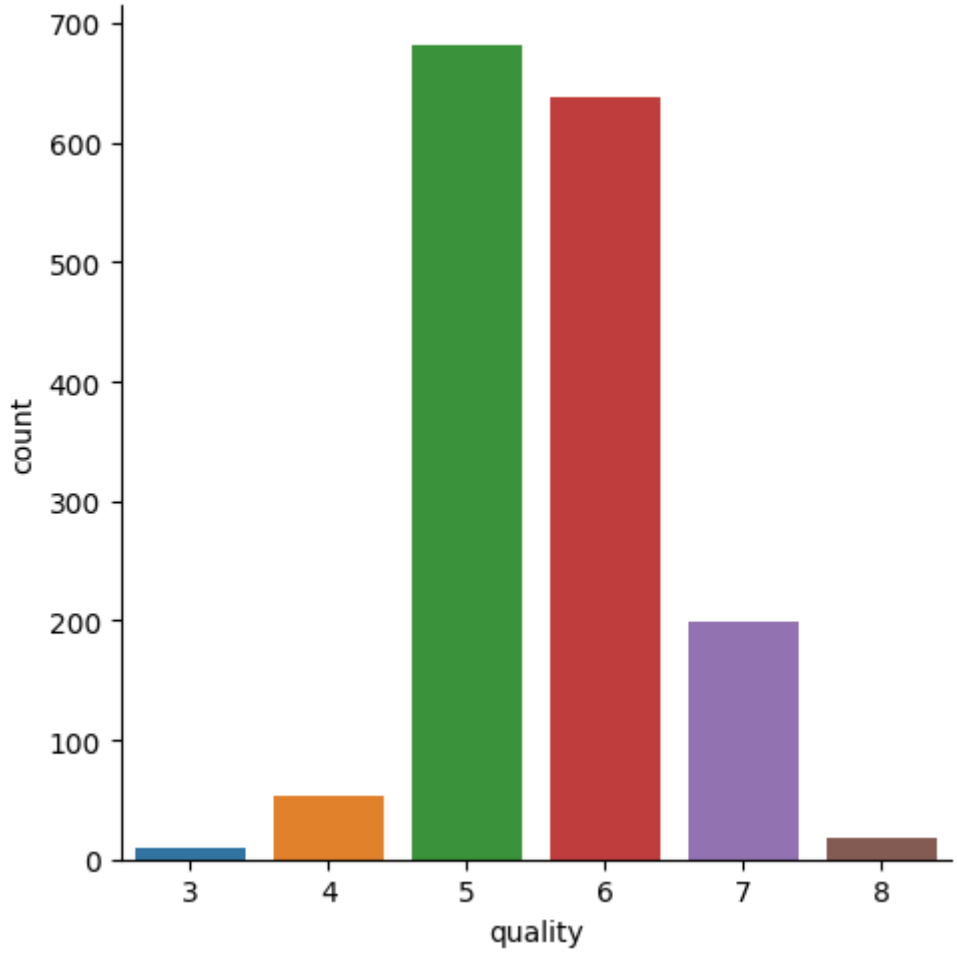
Faceted distributions

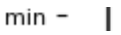





```
# number of values for each quality
sns.catplot(x='quality', data = wine_dataset, kind = 'count')
```

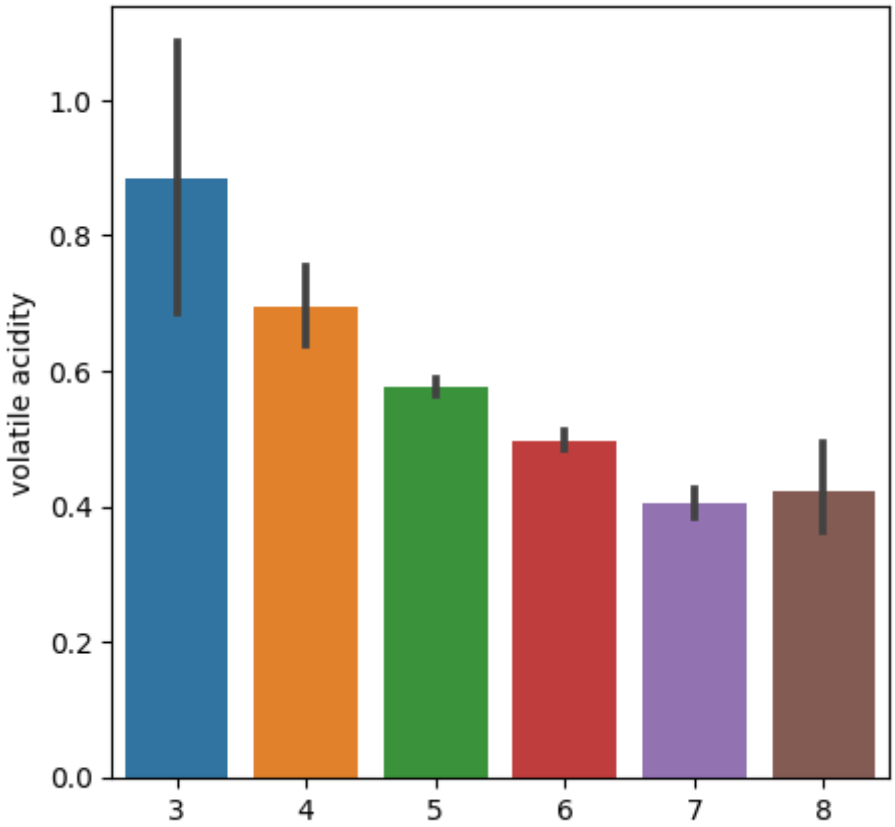
```
<seaborn.axisgrid.FacetGrid at 0x7b3abdf8f460>
```





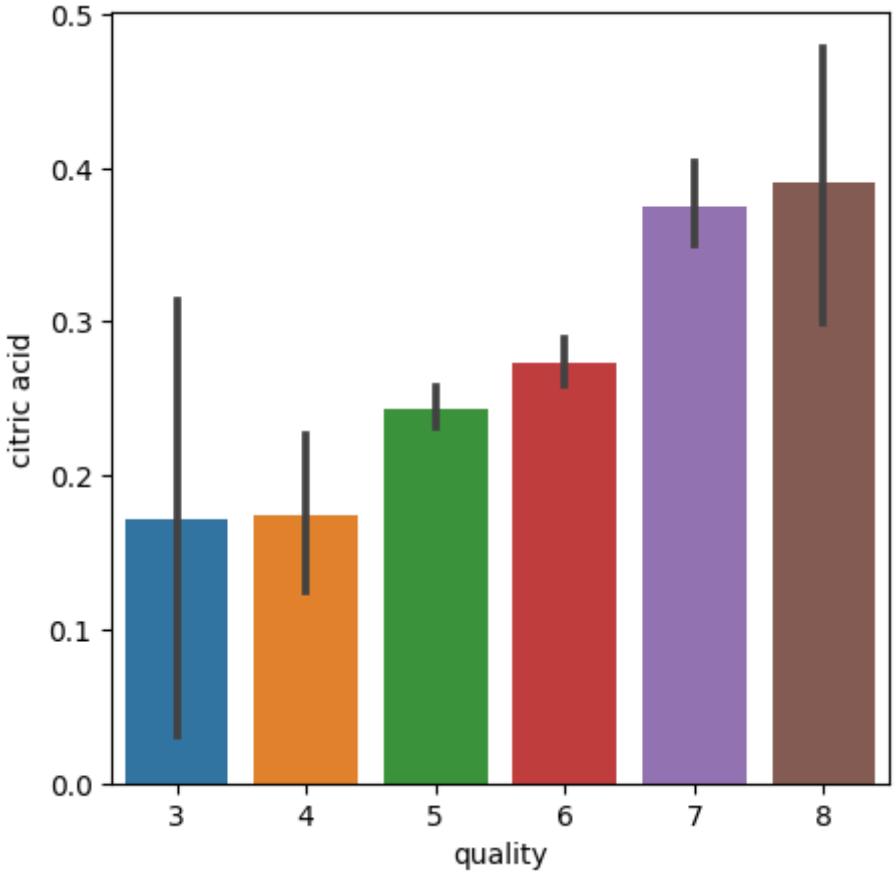

```
# volatile acidity vs Quality
plot = plt.figure(figsize=(5,5))
sns.barplot(x='quality', y = 'volatile acidity', data = wine_dataset)
```

```
<Axes: xlabel='quality', ylabel='volatile acidity'>
```




```
# citric acid vs Quality
plot = plt.figure(figsize=(5,5))
sns.barplot(x='quality', y = 'citric acid', data = wine_dataset)
```

```
<Axes: xlabel='quality', ylabel='citric acid'>
```



Correlation

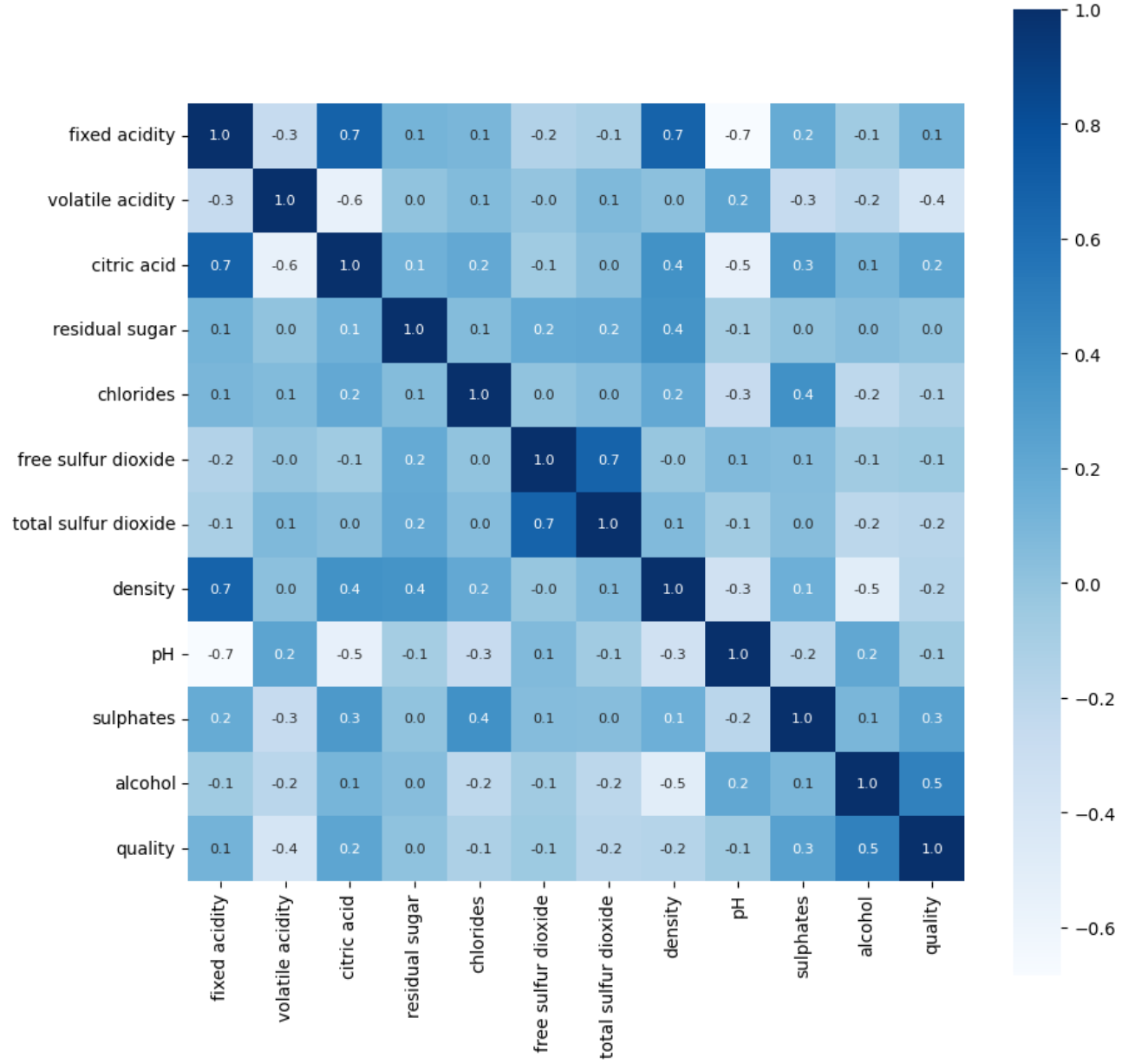
1. ***Positive Correlation***
2. ***Negative Correlation***

```
correlation = wine_dataset.corr()
```

```
tand the correlation between the columns
```

```
, square=True, fmt = '.1f', annot = True, annot_kws={'size':8}, cmap = 'Blues')
```

```
<Axes: >
```



Data Preprocessing

```
# separate the data and Label
X = wine_dataset.drop('quality',axis=1)
print(X)
```

```
      fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0               7.4             0.700         0.00             1.9      0.076
1               7.8             0.880         0.00             2.6      0.098
2               7.8             0.760         0.04             2.3      0.092
3              11.2             0.280         0.56             1.9      0.075
4               7.4             0.700         0.00             1.9      0.076
...             ...               ...          ...             ...        ...
1594            6.2             0.600         0.08             2.0      0.090
1595            5.9             0.550         0.10             2.2      0.062
1596            6.3             0.510         0.13             2.3      0.076
1597            5.9             0.645         0.12             2.0      0.075
1598            6.0             0.310         0.47             3.6      0.067

      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                    11.0                  34.0  0.99780  3.51       0.56
1                    25.0                  67.0  0.99680  3.20       0.68
2                    15.0                  54.0  0.99700  3.26       0.65
3                    17.0                  60.0  0.99800  3.16       0.58
4                    11.0                  34.0  0.99780  3.51       0.56
...                   ...                   ...      ...   ...        ...
1594                 32.0                  44.0  0.99490  3.45       0.58
1595                 39.0                  51.0  0.99512  3.52       0.76
1596                 29.0                  40.0  0.99574  3.42       0.75
1597                 32.0                  44.0  0.99547  3.57       0.71
1598                 18.0                  42.0  0.99549  3.39       0.66

      alcohol
0         9.4
1         9.8
2         9.8
3         9.8
4         9.4
...       ...
1594     10.5
1595     11.2
1596     11.0
1597     10.2
1598     11.0

[1599 rows x 11 columns]
```

Label Binarizaton

```
Y = wine_dataset['quality'].apply(lambda y_value: 1 if y_value>=7 else 0)
print(Y)
```

```
0       0
1       0
2       0
3       0
4       0
       ..
1594    0
1595    0
1596    0
1597    0
```

```
1598    0
Name: quality, Length: 1599, dtype: int64
```

Train & Test Split

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=3)
print(Y.shape, Y_train.shape, Y_test.shape)
```

```
(1599,) (1279,) (320,)
```

Model Training:

Random Forest Classifier

```
model = RandomForestClassifier()
model.fit(X_train, Y_train)
```

```
RandomForestClassifier
RandomForestClassifier()
```

Model Evaluation

Accuracy Score

```
# accuracy on test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy : ', test_data_accuracy)
```

```
Accuracy :  0.93125
```

Building a Predictive System

```
input_data = (7.5,0.5,0.36,6.1,0.071,17.0,102.0,0.9978,3.35,0.8,10.5)

# changing the input data to a numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the data as we are predicting the label for only one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = model.predict(input_data_reshaped)
print(prediction)

if (prediction[0]==1):
  print('Good Quality Wine')
else:
  print('Bad Quality Wine')
```

```
[0]
Bad Quality Wine
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not
  warnings.warn(
```

✓ 0s completed at 12:06 PM