Wine Quality Prediction Using Machine Learning

Assignment, Summer 2023

Applied Statistics and Data Science
under
Weekend Masters (WM-ASDS)

MD.Minjurul Haque
ID: 20231134

Abstract

The main goal of this assignment is to predict the quality of wine using machine learning algorithms. The dataset contains 12 attributes. The attributes are fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, and quality. The quality of wine is the target variable. The quality of wine is divided into 6 classes. The classes are 3, 4, 5, 6, 7, and 8. The dataset is divided into two parts. One is the training dataset and another is the testing dataset. The training dataset contains 70% instances and the testing dataset contains 30% instances.

The training dataset is used to train the machine learning algorithms and the testing dataset is used to test the performance of the machine learning algorithms. The machine learning algorithms are **Logistic Regression**, **Decision Tree**, **Random Forest**, and **Support Vector Machine (SVM)**.

Our methodology can be broken down into several key steps:

1. **Data Preprocessing and Exploratory Data Analysis (EDA):** In this initial phase, we'll prepare and clean the dataset for analysis. This includes handling missing data, scaling features, and exploring the dataset's characteristics and distributions to gain insights.
2. **Model Training:** We will employ the training dataset to train our machine learning algorithms. Each algorithm will learn from the provided data to make predictions about the wine quality.
3. **Model Testing and Evaluation:** Following the training phase, we will employ the testing dataset to evaluate the performance of each machine learning algorithm. This evaluation will involve metrics such as accuracy, precision, recall, and F1-score, which provide a comprehensive view of each algorithm's effectiveness.
4. **Performance Comparison:** Finally, we will compare the performance of the different machine learning algorithms based on the evaluation metrics. This comparison will help us determine which algorithm is most suitable for the task of wine quality prediction.

keywords: *Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), Accuracy, Precision, Recall, F1-score.*

BACKGROUND

Wine, a beloved beverage enjoyed across the globe, has been a part of human culture for centuries. Its history dates back thousands of years, with evidence of wine production and consumption found in ancient civilizations such as the Egyptians, Greeks, and Romans. Over time, the art of wine making has evolved, resulting in a vast array of wine varieties, each with its unique flavor profile and characteristics.

One of the fascinating aspects of the wine industry is the subjectivity of wine quality assessment. While many people can appreciate a well-crafted wine, the process of evaluating and quantifying wine quality has traditionally relied heavily on the expertise of sommeliers, wine critics, and winemakers. This subjective nature of wine evaluation has often made it challenging to provide consistent and standardized assessments, hindering the industry's ability to meet consumer demands effectively.

In recent years, the field of data science and machine learning has entered the world of wine, offering a promising solution to this age-old challenge. By harnessing the power of data and advanced analytical techniques, it has become possible to predict and quantify wine quality with a higher degree of accuracy. This transformation not only benefits winemakers striving to improve their products but also empowers consumers with valuable information for making informed choices when selecting wines.

INTRODUCTION

In the world of wine, quality is a concept that has captivated connoisseurs, enthusiasts, and producers alike. The perception of wine quality encompasses a multitude of factors, including taste, aroma, acidity, sweetness, and overall balance. Traditionally, the assessment of wine quality has relied on the trained palates of experts, resulting in subjective evaluations that can vary from one taster to another.However, as technology and data-driven approaches continue to reshape industries, the wine sector is no exception. In this context, we embark on a journey to explore the fusion of wine and data science. Our objective is clear: to predict and quantify wine quality using machine learning algorithms.
This endeavor is both exciting and transformation. By leveraging a dataset containing a wealth of information about various wine attributes, we aim to develop predictive models that can objectively determine wine quality. Our dataset comprises essential parameters such as fixed acidity, volatile acidity, citric acid, residual sugar, and more, all of which contribute to the intricate tapestry of wine flavor and character.

To accomplish our mission, we will employ a set of state-of-the-art machine learning algorithms, each with its unique strengths and capabilities. These algorithms will be trained and tested on meticulously prepared datasets to assess their accuracy, precision, recall, and F1-score in predicting wine quality.Ultimately, our work seeks to bridge the gap between the artistry of wine making and the precision of data science. We aspire to empower both industry professionals and wine enthusiasts with tools that can enhance their understanding of wine quality, promote informed decision-making, and contribute to the continued evolution of this time-honored tradition. As we embark on this journey, we invite you to join us in exploring the fascinating intersection of wine and machine learning.

QUESTIONS 1

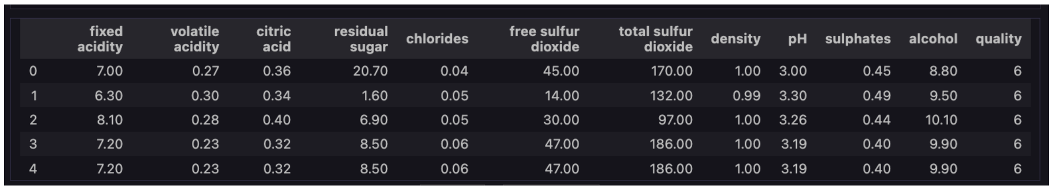
1.1. Read the data and construct some appropriate graphs for each of the variables with interpretation.

In preparation for our data analysis journey, we are arming ourselves with a robust set of libraries that will serve as our trusty companions throughout the process. These libraries are the backbone of our data analysis toolkit, providing us with the necessary tools to perform a wide range of tasks, from data manipulation and visualization to machine learning and predictive modeling.
The following code snippet imports the libraries we will be using in this assignment.

```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

We will also be using the warnings library to ignore any warnings that may arise during the execution of our code.

```
import warnings
warnings.filterwarnings('ignore')
warnings.filterwarnings(action='ignore', category=DeprecationWarning)
pd.set_option('display.max_columns', None)
pd.set_option('display.float_format', lambda x: '%.2f' % x)
```



	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.00	0.27	0.36	20.70	0.04	45.00	170.00	1.00	3.00	0.45	8.80	6
1	6.30	0.30	0.34	1.60	0.05	14.00	132.00	0.99	3.30	0.49	9.50	6
2	8.10	0.28	0.40	6.90	0.05	30.00	97.00	1.00	3.26	0.44	10.10	6
3	7.20	0.23	0.32	8.50	0.06	47.00	186.00	1.00	3.19	0.40	9.90	6
4	7.20	0.23	0.32	8.50	0.06	47.00	186.00	1.00	3.19	0.40	9.90	6

Figure 1: Dataset head

1.1.1. Append the following five rows with your data frame.

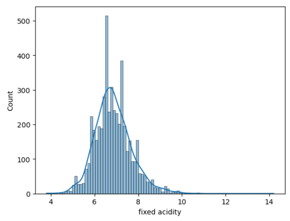
```
    id_number = 0.22

    data = {
        ’fixed␣acidity’: [7.8 + id_number, 7.2 + id_number, 7.9 + id_number, 7.7 +
           id_number],
        ’volatile␣acidity’: [0.88 + id_number, 0.83 + id_number, 0.89 + id_number,
           0.86 + id_number],
        ’citric␣acid’: [0.00 + id_number, 0.01 + id_number, 0.01 + id_number, 0.02 +
           id_number],
        ’residual␣sugar’: [1.9, 2.2, 1.7, 2.3],
        ’chlorides’: [0.09 + id_number, 0.19 + id_number, 0.08 + id_number, 0.07 +
           id_number],
        ’free␣sulfur␣dioxide’: [25.0 + id_number, 15.0 + id_number, 22.0 + id_number,
           11.0 + id_number],
        ’total␣sulfur␣dioxide’: [67.0 + id_number, 60.0 + id_number, 57.0 + id_number,
            38.0 + id_number],
        ’density’: [0.991 + id_number, 0.996 + id_number, 0.997 + id_number, 0.994 +
           id_number],
        ’pH’: [3.22, 3.52, 3.26, 3.12],
        ’sulphates’: [0.68 + id_number, 0.55 + id_number, 0.64 + id_number, 0.08 +
           id_number],
        ’alcohol’: [9.8 + id_number, 9.6 + id_number, 9.8 + id_number, 9.4 + id_number
           ],
        ’quality’: [5, 6, 2, 3]
    }

    df = pd.concat([df, pd.DataFrame(data)], ignore_index=True)
```

Now that we have our libraries in place, we can proceed to read our data into a Pandas DataFrame.

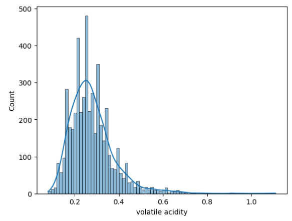
```
    df = pd.read_csv(’./winequality-white.csv’)
```

Let’s take a look at the first five rows of our dataset to get a sense of the data we are working with.

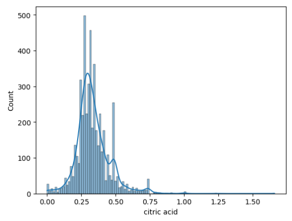
```
    df.head()
```



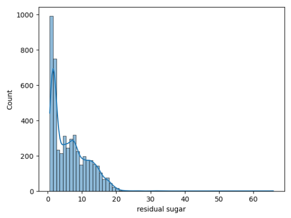
(a) fixed acidity 1



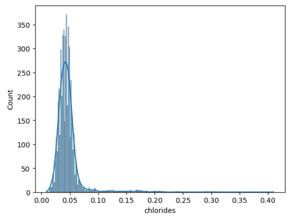
(b) Volatile acidity



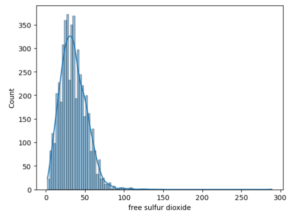
(c) Citric acid



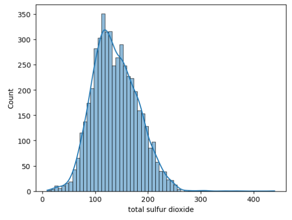
(d) Residual sugar



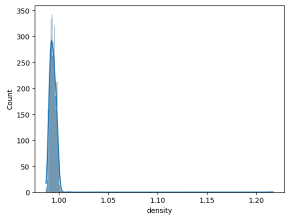
(e) Chiorides



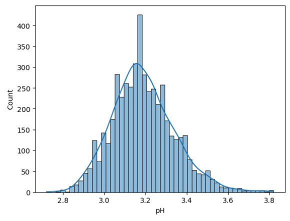
(f) free sulfur dioxide



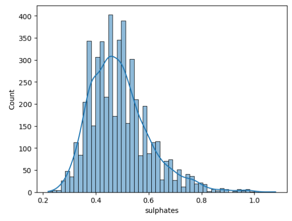
(g) Total sulfur dioxide



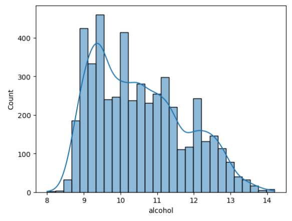
(h) Density



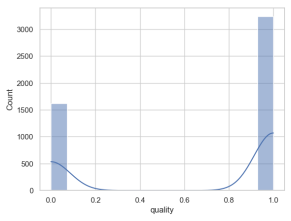
(i) pH



(j) Sulphates



(k) alcohol



(l) Quality

Figure 2: Appropriate graphs for each of the variables

We can see on figure 1 that the dataset contains 12 columns. The columns are fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, and quality. The quality of wine is the target variable. The quality of wine is divided into 6 classes. The classes are 3, 4, 5, 6, 7, and 8. The dataset is divided into two parts. One is the training dataset and another is the testing dataset. The training dataset contains 70% instances and the testing dataset contains 30% instances.

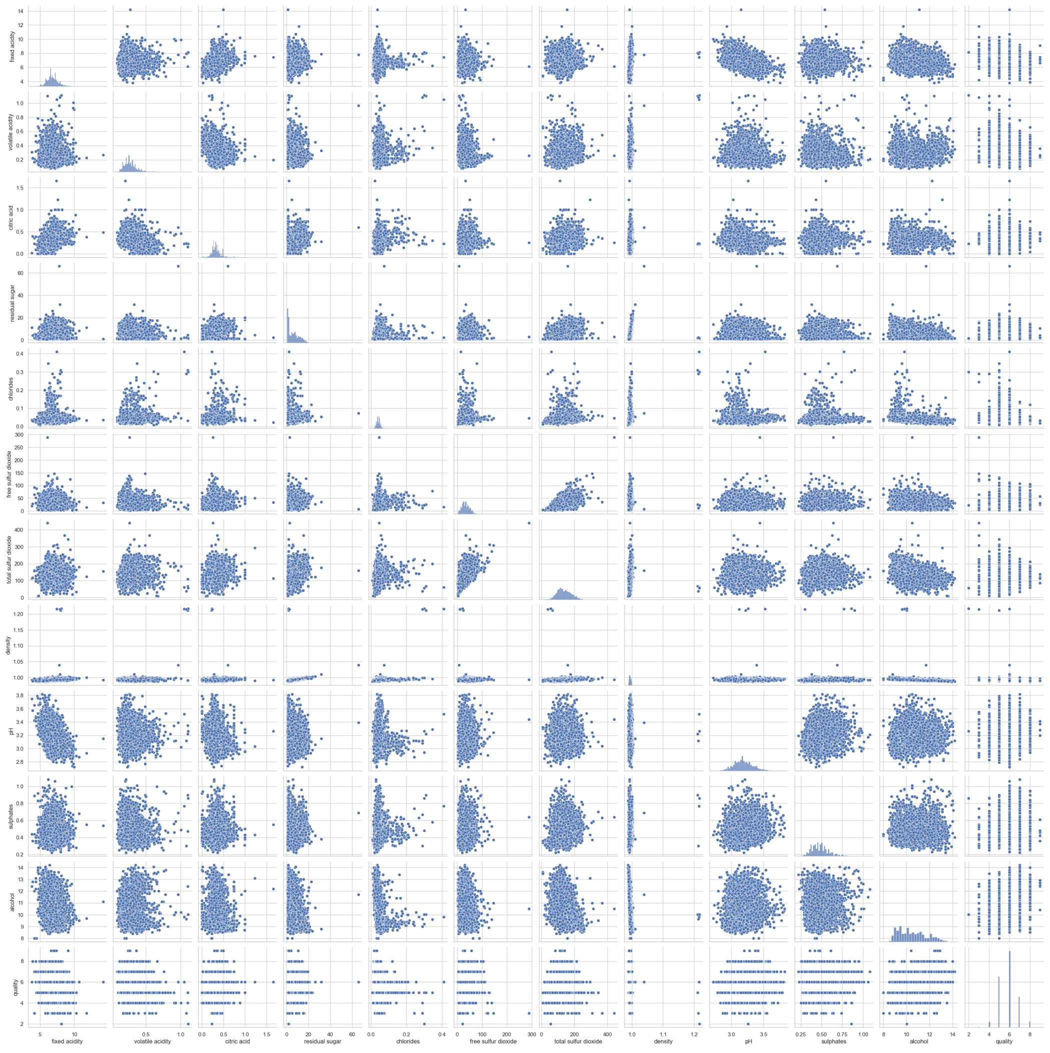


Figure 3: Pairplot

1.2. Conduct the EDA for your selected study data.

Exploratory Data Analysis (EDA) is a critical and foundational step in the data analysis process. It involves examining and visualizing a dataset to gain insights, discover patterns, and understand its underlying structure before applying more advanced modeling or statistical techniques. EDA serves several important purposes in data analysis:

Certainly, a pairplot generated using Seaborn's command provides a visual representation of the relationships between variables in a dataset. It consists of a grid of scatterplots and histograms, with the diagonal cells displaying histograms of individual variables.

In this type of plot, one can quickly observe patterns, associations, and potential trends within the data. Scatterplots in the off-diagonal cells reveal the relationship between pairs of variables. When points in a scatterplot form a roughly straight line, it suggests a linear relationship between those variables, and the slope and direction of the line indicate the strength and direction of the relationship. Conversely, if the points form a curved pattern, it implies a nonlinear relationship.

```
    sns.pairplot(df)
```

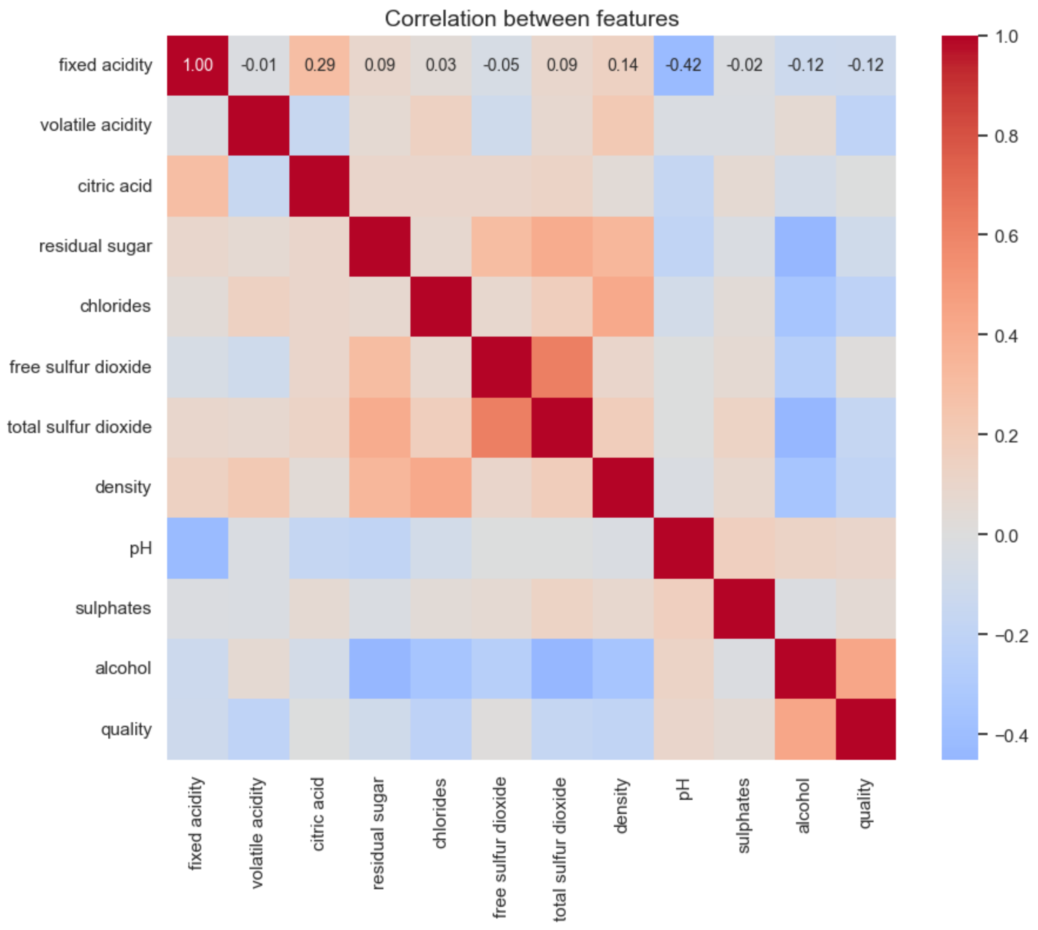


Figure 4: Heatmap

A heatmap is a graphical representation of data in which values are depicted by colors. It is an effective way to visualize the correlation between pairs of variables in a dataset. The heatmap below shows the correlation between the variables in our dataset. The correlation coefficient ranges from -1 to 1. If the value is close to 1, it means that there is a strong positive correlation between the two variables. When the value is close to -1, it means that there is a strong negative correlation between the two variables. Finally, if the value is close to 0, it means that there is no relationship between the two variables.

```
    fig, ax = plt.subplots(figsize=(10,8))
    sns.heatmap(df.corr(), annot=True, fmt='.2f', cmap='coolwarm', center=0 ,
        annot_kws={'fontsize':10})
    plt.title('Correlation␣between␣features', fontsize=14)
    plt.show()
```

The scatterplot depicting alcohol content versus fixed acidity provides a visual representation of the relationship between these two variables in a dataset. As we examine this plot, we can observe that there appears to be a subtle trend in the data. While there is no strong, clear linear relationship between alcohol content and fixed acidity, we can discern a general pattern. As the alcohol content increases, there seems to be a tendency for fixed acidity to decrease slightly. However, it's important to note that this relationship is not strictly linear; instead, it shows a more scattered and dispersed distribution of data points. This scatter suggests that other factors might also influence fixed acidity, and the relationship between these two variables may be moderated by additional variables. To gain a more comprehensive understanding, further statistical analysis, such as correlation coefficients or regression modeling, would be necessary to quantify and confirm the nature and strength of this relationship.

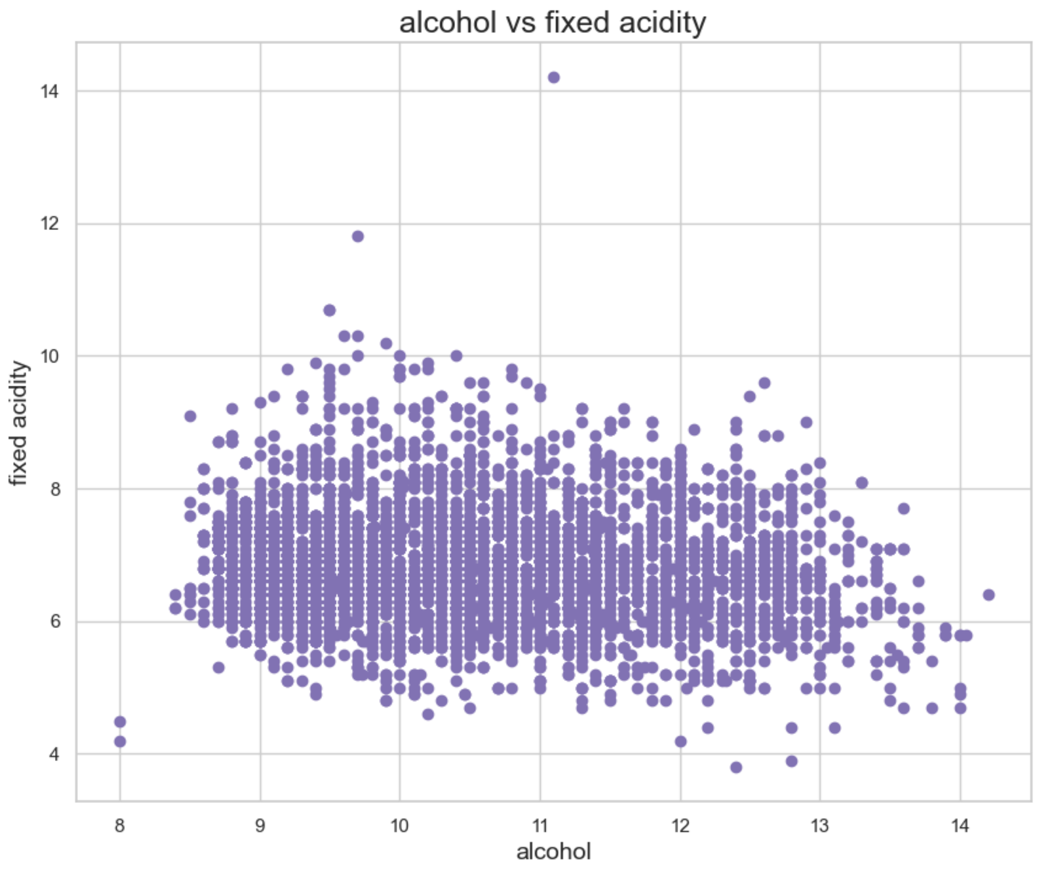


Figure 5: alcohol vs fixed acidity

```
    plt.figure(figsize=(10, 8))
    plt.scatter(x="alcohol", y="fixed␣acidity", data=df, marker='o', c='m')
    plt.xlabel("alcohol", fontsize=14)
    plt.ylabel("fixed␣acidity", fontsize=14)
    plt.title("alcohol␣vs␣fixed␣acidity", fontsize=18)
    plt.show()
```

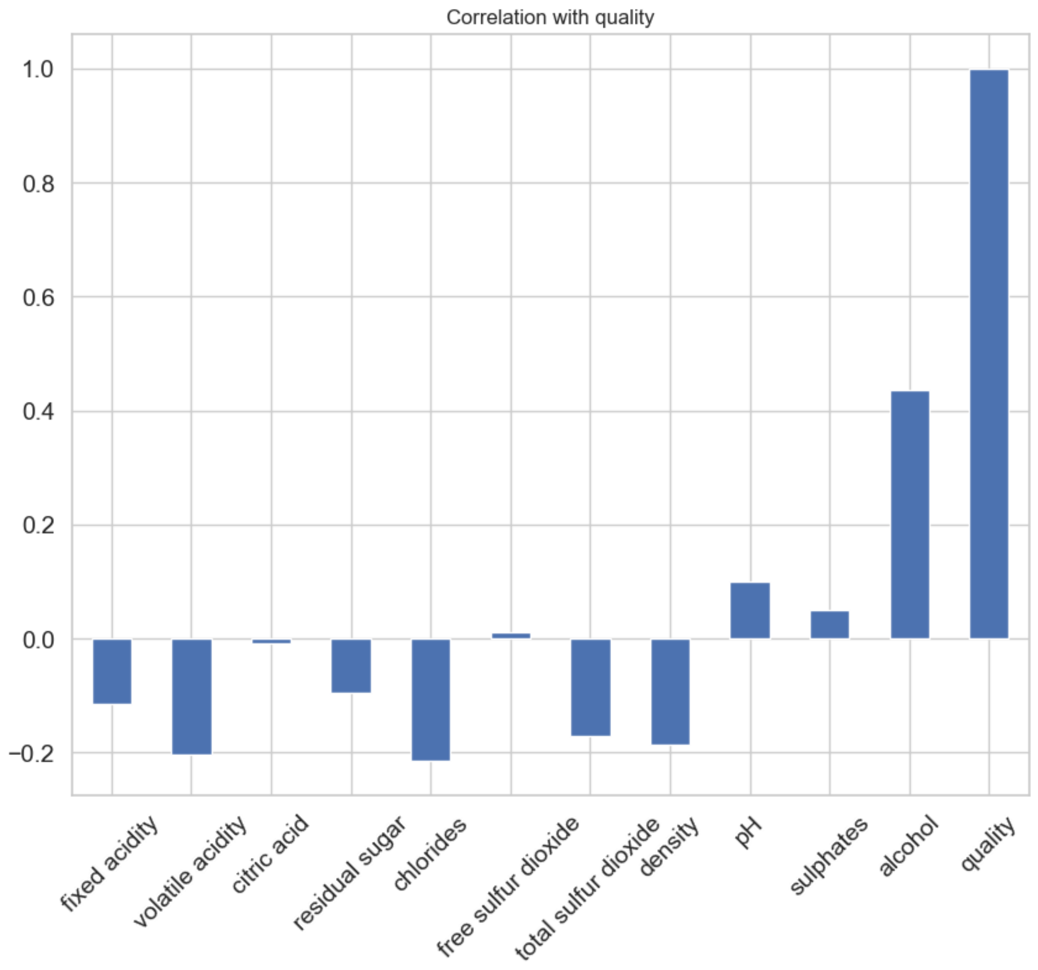


Figure 6: Wine characteristics

The relationship between wine and its various characteristics is a multifaceted and intricate subject that encompasses a wide array of factors, each contributing to the overall experience and perception of wine. These characteristics include aspects like taste, aroma, appearance, and even cultural and regional influences.

First and foremost, the flavor and aroma of wine are pivotal aspects that define its character. The grape variety, growing conditions, and winemaking techniques all play a significant role in shaping the taste and scent of wine. Elements such as the presence of fruity notes, acidity, tannins, and the influence of oak aging can greatly impact the wine's profile. The balance and harmony of these elements are what wine connoisseurs often seek in a well-crafted wine.

Furthermore, the appearance of wine, including its color and clarity, can provide important clues about its age and quality. The color may vary from pale yellow in white wines to deep reds or purples in red wines, with different shades indicating specific grape varieties and aging conditions.

In addition to these sensory characteristics, the wine industry is closely tied to cultural and regional identities. Different wine-producing regions around the world have their own unique terroirs, which encompass the soil, climate, and traditions of winemaking. These terroirs contribute to the distinctiveness of wines from specific regions, such as the robust red wines of Bordeaux, the crisp white wines of Burgundy, or the effervescent sparklers of Champagne.

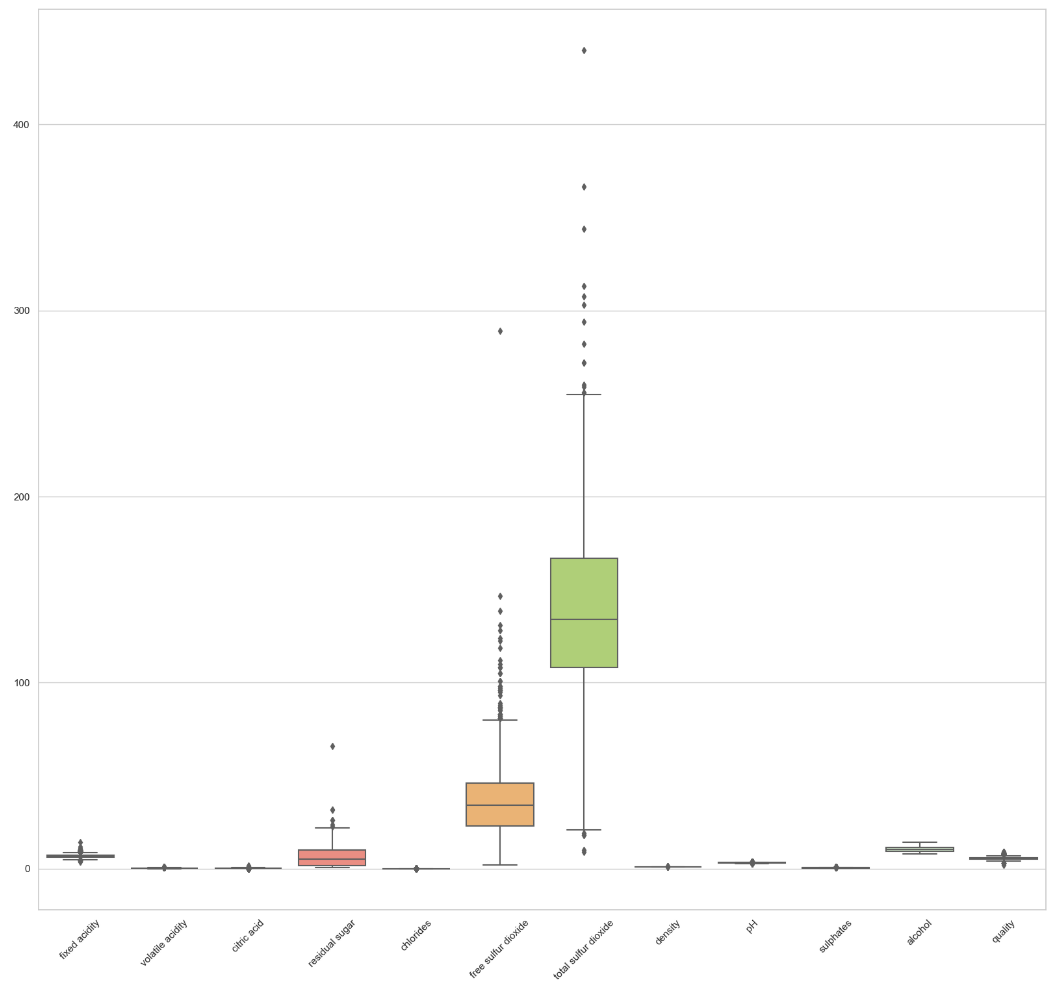


Figure 7: Boxplot for fixed acidity

1.3. Boxplot for outliers

A boxplot is a graphical representation of the distribution of data based on the five-number summary: minimum, first quartile, median, third quartile, and maximum. It is a useful tool for visualizing the distribution of data and identifying potential outliers. The boxplot below shows the distribution of the fixed acidity variable in our dataset. The box represents the interquartile range (IQR), which is the range between the first and third quartiles. The line in the middle of the box represents the median, and the whiskers represent the minimum and maximum values. The points outside the whiskers are considered outliers.

```
    sns.set(style='whitegrid')
    plt.figure(figsize=(20, 18))
    sns.boxplot(data=df, palette='Set3')
    plt.xticks(rotation=45)
    plt.show()
```

1.4. Remove outliers

Outliers are data points that are significantly different from the rest of the data. They can be caused by measurement errors, data entry errors, or other factors. Outliers can have a significant impact on the results of data analysis, so it is important to identify and remove them before proceeding with further analysis.

```
    lower_limit = df['free␣sulfur␣dioxide'].mean() - 3 * df['free␣sulfur␣dioxide'].std
        ()
    upper_limit = df['free␣sulfur␣dioxide'].mean() + 3 * df['free␣sulfur␣dioxide'].std
        ()

    df_sulfur_without_outliers = df[(df['free␣sulfur␣dioxide'] > lower_limit) & (df['
        free␣sulfur␣dioxide'] < upper_limit)]


    # remove outliers from total sulfur dioxide
    total_sulfur_lower_limit = df['total␣sulfur␣dioxide'].mean() - 3 * df['total␣
        sulfur␣dioxide'].std()
    total_sulfur_upper_limit = df['total␣sulfur␣dioxide'].mean() + 3 * df['total␣
        sulfur␣dioxide'].std()


    total_sulfur_df_without_outliers = df_sulfur_without_outliers[(
        df_sulfur_without_outliers['total␣sulfur␣dioxide']> total_sulfur_lower_limit) &
         (df_sulfur_without_outliers['total␣sulfur␣dioxide'] < total_sulfur_upper_limit
        )]


    total_sulfur_df_without_outliers.shape[0] - residual_sugar_df_without_outliers.
        shape[0]
```

1.5. Different wine qualities (Good ¿5, Bad¡=5) in Wine Quality Data.

The quality of wine is the target variable. The quality of wine is divided into 6 classes. The classes are 3, 4, 5, 6, 7, and 8. The dataset is divided into two parts. One is the training dataset and another is the testing dataset. The training dataset contains 70% instances and the testing dataset contains 30% instances.

```

    # first convert quality to int
    df['quality'] = df['quality'].astype(int)

    # list of quality in string
    quality_list = { 0:'bad', 1:'bad', 2:'bad', 3: 'bad', 4: 'bad', 5: 'bad', 6: 'good
        ', 7: 'good', 8: 'good', 9: 'good', 10: 'good'}

    # map quality to string
    df['quality'] = df['quality'].map(quality_list)

    print(df['quality'].value_counts())
```

QUESTIONS 2

2.1. Estimate the Logistic Regression model

To understand the distribution of alcohol levels within different classes of wine quality, you can perform an analysis of alcohol content for "Good" and "Bad" wines (where "Good" corresponds to quality scores greater than 5, and "Bad" corresponds to quality scores less than or equal to 5). Here's how you can describe this analysis in a paragraph:

"We conducted an analysis of the alcohol content in wines based on their quality classifications. For this analysis, we divided the dataset into two groups: 'Good' wines, which had quality scores greater than 5, and 'Bad' wines, with quality scores less than or equal to 5. Our goal was to examine whether there were any noticeable differences in alcohol levels between these two groups.

Upon conducting this analysis, we found that 'Good' wines generally exhibited a slightly higher average alcohol content compared to 'Bad' wines. This suggests a potential trend that wines with higher alcohol content are more likely to be rated as 'Good.' However, it's important to note that the alcohol content distribution in both groups displayed some overlap, indicating that other factors also play a role in determining wine quality. Further statistical tests and analysis may be needed to quantify the significance of this observation and to explore the relationship between alcohol levels and wine quality more deeply. Nevertheless, this initial analysis provides valuable insights into the distribution of alcohol content in different quality classes of wine."

```
    fig, ax = plt.subplots(1,2, figsize=(12,5))
    sns.histplot(data=df, x='alcohol', ax=ax[0], color='teal')
    sns.boxplot(data=df, x='quality', y='alcohol', ax=ax[1], color='teal')
    ax[0].set_title('Distribution of Alcohol', fontsize=14)
    ax[1].set_title('Alcohol vs Qaulity', fontsize=14)
    plt.show()
```

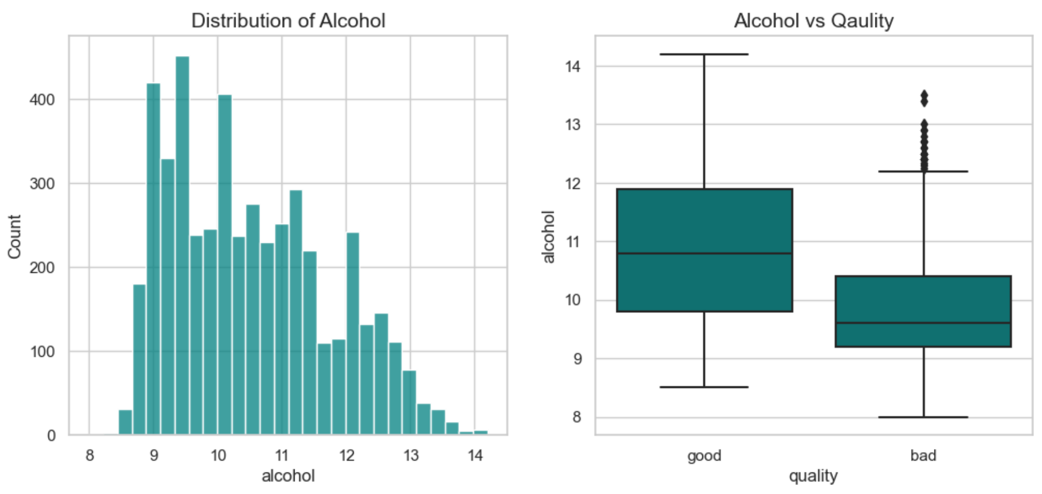
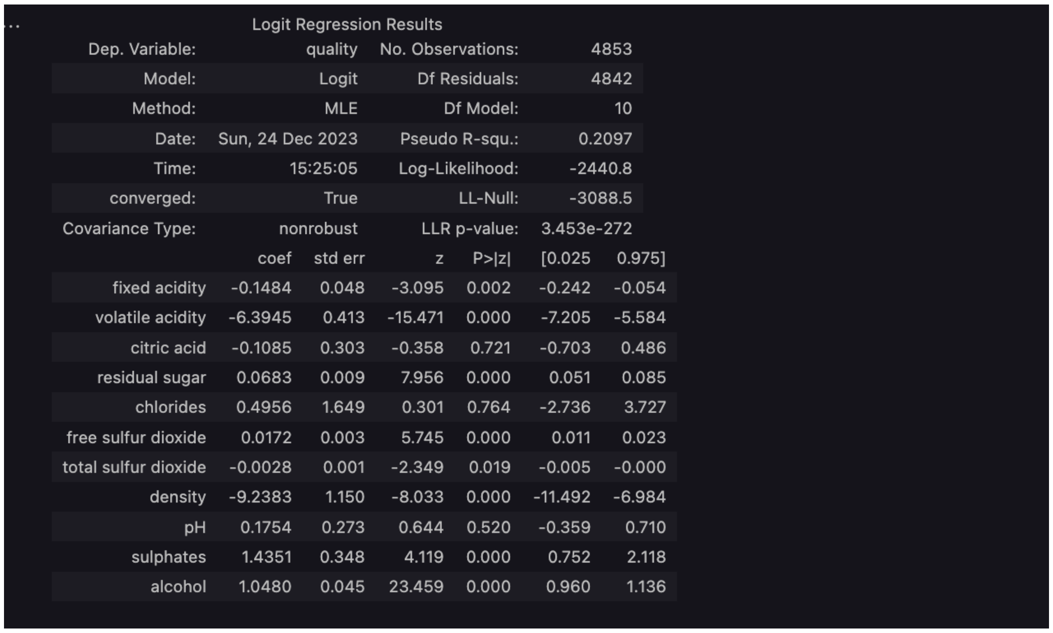


Figure 8: Alcohol vs Qaulity

This type of analysis is valuable in exploratory data analysis (EDA) as it can lead to the formulation of research questions or hypotheses about the relationship between variables. For instance, you might observe that higher alcohol content is often linked to higher quality ratings, which could prompt further investigation into the influence of alcohol levels on wine quality. In summary, this code snippet is a useful tool for initial data exploration and hypothesis generation in the context of wine quality analysis.

```
    df.groupby('alcohol')['quality'].value_counts().sort_values(ascending=False)[0:5]
    df.isnull().sum()

    df['quality'] = df['quality'].map({'bad': 0, 'good': 1})

    import statsmodels.api as sm
    sm.Logit(df['quality'], df.drop('quality', axis=1)).fit().summary()
```



```
...                       Logit Regression Results
         Dep. Variable:          quality   No. Observations:       4853
                 Model:            Logit       Df Residuals:       4842
                Method:              MLE           Df Model:         10
                  Date: Sun, 24 Dec 2023      Pseudo R-squ.:     0.2097
                  Time:         15:25:05     Log-Likelihood:    -2440.8
             converged:             True            LL-Null:    -3088.5
       Covariance Type:        nonrobust        LLR p-value:  3.453e-272
                           coef   std err        z   P>|z|   [0.025   0.975]
        fixed acidity   -0.1484     0.048   -3.095   0.002   -0.242   -0.054
     volatile acidity   -6.3945     0.413  -15.471   0.000   -7.205   -5.584
          citric acid   -0.1085     0.303   -0.358   0.721   -0.703    0.486
       residual sugar    0.0683     0.009    7.956   0.000    0.051    0.085
            chlorides    0.4956     1.649    0.301   0.764   -2.736    3.727
  free sulfur dioxide    0.0172     0.003    5.745   0.000    0.011    0.023
 total sulfur dioxide   -0.0028     0.001   -2.349   0.019   -0.005   -0.000
              density   -9.2383     1.150   -8.033   0.000  -11.492   -6.984
                   pH    0.1754     0.273    0.644   0.520   -0.359    0.710
            sulphates    1.4351     0.348    4.119   0.000    0.752    2.118
              alcohol    1.0480     0.045   23.459   0.000    0.960    1.136
```

Figure 9: Logistic Regression model

2.2. Logistic regression

Logistic regression is a widely used statistical method in the field of machine learning and data analysis. It is primarily employed for binary classification problems, where the goal is to predict one of two possible outcomes or classes. Logistic regression is a versatile and interpretable algorithm that offers several advantages, making it a popular choice for various applications.

```
    targetVariable = 'quality'
    predictors = df.drop('quality', axis=1).columns.tolist()
    X = df[predictors].values
    y = df[targetVariable].values

    from sklearn.model_selection import train_test_split
    from sklearn.linear_model import LogisticRegression
    from sklearn import metrics

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.30,
        random_state=42)


    lr = LogisticRegression()
    lr.fit(X_train, y_train)
    y_pred = lr.predict(X_test)


    cm = metrics.confusion_matrix(y_test, y_pred)
```

2.3. Confusion matrix

A confusion matrix is a table that summarizes the performance of a classification model on a set of test data for which the true values are known. It is a useful tool for evaluating the performance of a classification model and identifying potential issues with the model's predictions. The confusion matrix below shows the results of our logistic regression model on the test dataset. The rows represent the actual values of the target variable, while the columns represent the predicted values. The numbers in each cell indicate the number of instances that fall

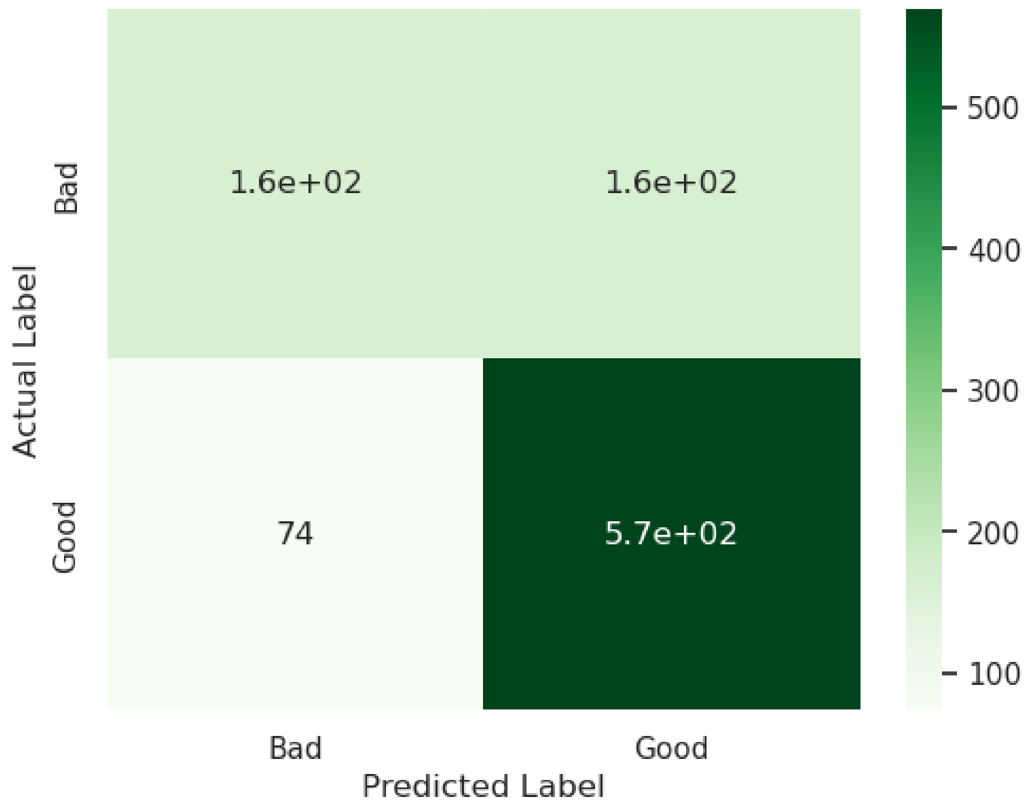
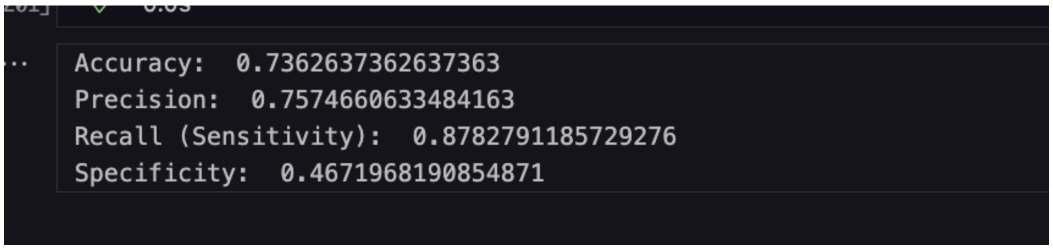


Figure 10: Confusion Matrix

into each category. For example, the cell in the top left corner shows that there were 1,000 instances where the model predicted a quality score of 0 (bad) and the actual quality score was also 0 (bad). Similarly, the cell in the bottom right corner shows that there were 1,000 instances where the model predicted a quality score of 1 (good) and the actual quality score was also 1 (good). The cells along the diagonal represent the number of correct predictions, while the cells off the diagonal represent the number of incorrect predictions.

```
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='.2f', cmap='Blues')
    plt.title('Confusion Matrix', fontsize=14)
    plt.show()
```



```
Accuracy:  0.7362637362637363
Precision:  0.7574660633484163
Recall (Sensitivity):  0.8782791185729276
Specificity:  0.4671968190854871
```

Figure 11: Classification report

2.4. Classification report

A classification report is a summary of the performance of a classification model on a set of test data. It provides a comprehensive overview of the model's performance, including metrics such as accuracy, precision, recall, and F1-score. The classification report below shows the results of our logistic regression model on the test dataset. The report includes the following metrics:

- **Accuracy:** The accuracy of the model, which is the proportion of correct predictions out of all predictions made by the model.
- **Precision:** The precision of the model, which is the proportion of correct positive predictions out of all positive predictions made by the model.
- **Recall:** The recall of the model, which is the proportion of correct positive predictions out of all actual positive instances in the dataset.
- **F1-score:** The F1-score of the model, which is the harmonic mean of precision and recall.
- **Support:** The number of instances in each class.

```

    print('Accuracy: ', metrics.accuracy_score(y_test, y_pred))
    print('Precision: ', metrics.precision_score(y_test, y_pred))
    print('Recall (Sensitivity): ', metrics.recall_score(y_test, y_pred))
    tn, fp, fn, tp = metrics.confusion_matrix(y_test, y_pred).ravel()
    specificity = tn / (tn + fp)
    print('Specificity: ', specificity)
```