A logo for college computing

Description automatically generated

**Assessment**

|  |  |
| --- | --- |
| *Ana Paula de Oliveira Damasceno* |  |
| *Student Number: 2024079* |  |
| *Module Title: Statistical Techniques for Data Analysis* |  |
| *Students Performance in Exams* |  |
| *Assessment Due Date: 21/04/2024* |  |
| *Date of Submission: 21/04/2024i* |  |

**Declaration**

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.

Summary

[*Module Title: Statistical Techniques for Data Analysis* 1](#_Toc164597372)

[1. INTRODUCTION 3](#_Toc164597373)

[2. IMPORTING LIBRARIES 3](#_Toc164597374)

[3. DATA CHARACTERIZATION 3](#_Toc164597375)

[3.1 Load the Dataset 3](#_Toc164597376)

[3.2 Check information about variables and data types 3](#_Toc164597377)

[4. CALCULATE DESCRIPTIVE STATISTICS 4](#_Toc164597378)

[5. DATA CLEANING 5](#_Toc164597379)

[6. STATISTICAL ANALYSYS 7](#_Toc164597380)

[7. DATA EXPLORATION 12](#_Toc164597381)

[8. CONCLUSION 18](#_Toc164597382)

[9. TASK 2 AND TASK 19](#_Toc164597383)

[9.1 Task 2 - Probability (Discrete): 19](#_Toc164597384)

[9.2 Task 3 - Probability (Continuous): 21](#_Toc164597385)

[10. REFERENCES 23](#_Toc164597386)

# INTRODUCTION

In this project, we explore a wine dataset consisting of 2993 entries. Our focus lies in the statistical analysis of variables such as region, price, and judges' ratings. We employ measures like mean, median, and mode to gain a deeper understanding of these wines' characteristics. Through this analysis, we aim to uncover insights that can enrich our understanding of the dynamics behind wine appreciation.

# IMPORTING LIBRARIES

import pandas as pd

import seaborn as sns

import numpy as np

import matplotlib.image as plt

import matplotlib.pyplot as plt

import plotly.express as px

%matplotlib inline

# DATA CHARACTERIZATION

# Load the Dataset

Load the Dataset

df=pd.read\_csv("PortugueseWines.csv")

Head

Viewing the First 10 Rows of the dataset

Table

Auto-generated description

# Check information about variables and data types

df.shape

(2993, 16)

<flokkur 'pandas.core.frame.DataFrame'>

RangeIndex: 2993 entries, 0 to 2992

Data columns (total 16 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Unnamedu: 0 2993 Non-Nuck Int64

1 Non-Null Object in Name 2993

2 Region 2989 non-null object

3 Year 2993 non-null int64

4 Color 2993 non-null object

5 Castes 2989 non-null object

6 AlcoholPercentage 2993 non-null float64

7 Producer 2993 non-null object

8 MinimunPrice 2993 non-null float64

9 MaximumPrice 2993 non-null float64

10 Judge 2993 non-null object

11 JudgeRating 2993 non-null float64

12 Date 2970 non-null object

13 JudgeNotes 2992 non-null object

14 Label 2992 non-null object

15 Link 2993 Non-within Object

dtypes: float64(4), int64(2), object(10)

memory usage: 374.3+ KB

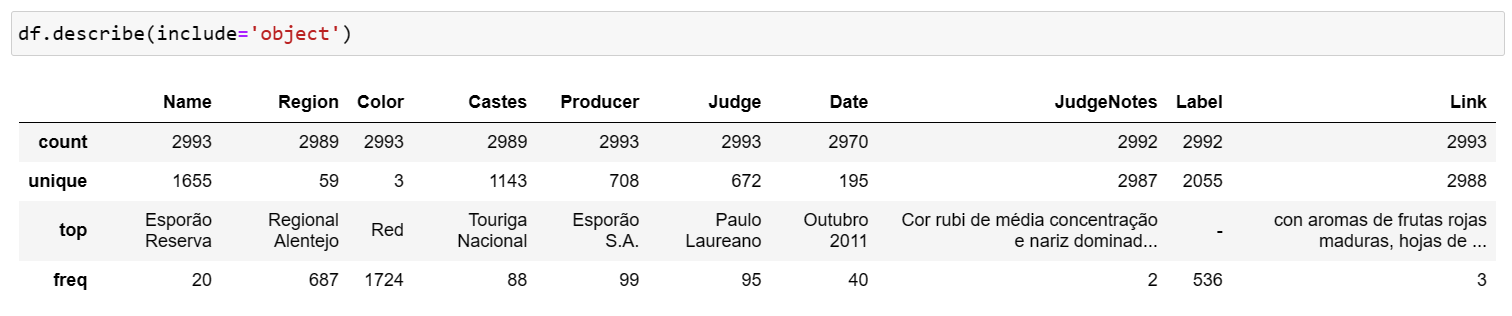
# CALCULATE DESCRIPTIVE STATISTICS

The describe function provides a statistical summary of the numeric columns in a dataset, including counts, means, standard deviations, quartiles, and minimum and maximum values.

DF.describe()



* 1. **Statistical summary of categorical variables**



# DATA CLEANING

Missing values handling \_ Data cleaning

Missing values handling refers to the process of dealing with missing/null values in a dataset. In this task, we use three approaches, namely:

Imputation Models:Rather than using basic statistics to replace missing values, these models predict missing values based on other variables in the datase

Imputation:Replaces missing values with calculated values. This can be done using simple statistics like mean, median, or mode, or through more advanced techniques like regression or machine learning models.

Removal of Rows or Columns): Removes observations or variables that contain missing values.

The columns last\_review and reviews\_per\_month will not be used, therefore, they will be deleted

Identifying Null Values in Data Set

df.isnull().sum()

33

m\_v\_f = ["n.a", ","," "?", "NA", "n/a", "na", "--"]

df = pd.read\_csv("PortugueseWines.csv", na\_values=m\_v\_f)

D.F. Isnul() and ().

columns\_with\_nulls= df.columns[df.isnull().any()]

print("Columns with null values:")

print(columns\_with\_nulls)

null\_count\_per\_column = df.isnull().sum()

print("Count of null values per column:")

print(null\_count\_per\_column)

Count of null values per column:

Unnamed: 0 0

Name 0

Region 4

Year 0

Color 0

Castes 4

Alcohol Content 0

Producer 0

MinimunPrice 0

MaximumPrice 0

Judge 0

JudgeRating 0

Date 23

JudgeNotes 1

Label 1

Link 0

dtype: int64

total\_rows = len(df)

missing\_values\_per\_column = df.isnull().sum()

percentage\_missing\_values = (missing\_values\_per\_column / total\_rows) \* 100

print("Percentage of Missing Values over total rows:")

print(percentage\_missing\_values)

Percentage of Missing Values over total rows:

Unnamed: 0 0.000000

Name 0.000000

Region 0.133645

Year 0.000000

Color 0.000000

Castes 0.133645

Alcohol Content 0.0000000

Producer 0.000000

MinimunPrice 0.000000

MaximumPrice 0.000000

Judge 0.000000

JudgeRating 0.000000

Date 0.768460

JudgeNotes 0.033411

Label 0.033411

Link 0.000000

dtype: float64

After identifying the missing values, we assess that the representativeness of this data in the total data set is irrelevant, so we delete the missing values from the database.

df = df.dropna(axis=0)

D.F. Isnul() and ().

0

df.shape

(2961, 16)

Another analysis we did in the data cleaning stage was the identification of duplicate values, in the case of this base, the values are irrelevant.

duplicates = df.duplicated()

print(df[duplicates])

Empty DataFrame

Columns: [Unnamed: 0, Name, Region, Year, Color, Castes, AlcoholPercentage, Producer, MinimunPrice, MaximumPrice, Judge, JudgeRating, Date, JudgeNotes, Label, Link]

Index: []

df.drop\_duplicates()

df.shape

(2961, 16)

df = df.drop(['Unnamed: 0', 'Link'], axis=1)

df.shape

(2961, 14)

# STATISTICAL ANALYSYS

The statistical analysis of mean, median, and mode will be performed for the variables minimum price, maximum price, alcohol percentage, and Judge Rating. Then, an exploratory data analysis will be conducted.

**MEAN**

The mean (mean) is a measure of central tendency that represents the typical value of a dataset. To calculate the mean, we sum all the values in the dataset and divide by the total number of values.

In the codes below, we calculate the mean for the variables minimum price, maximum price, alcohol percentage, and Judge Rating.

def calc\_mean(variable):

#In any funtion you create

#You should implement some

#Security logic - Error mitigation

#Create a conditional statement to

#process the error

if len(variable) == 0:

#Check the length of the data

#If we dont have enough data to process

#Terminate by returning nothing

return None

#Otherwise we assume we have data

#Since we do hacve data let's check the mean

#Using the formula above mean = Summation Xi / N

#Create the mean variable to store the mean value

mean = sum(variable) / len(variable)

#Now we return the output to the user

return mean

calc\_mean(df["MinimunPrice"])

9.607058426207363

## Calculate the average minimum price by color

average\_min\_price\_by\_color = df.groupby('Color')['MinimunPrice'].mean()

# Display the average minimum price by color

print("Average minimum price by color:")

print(average\_min\_price\_by\_color)

Average minimum price by color:

Color

Red 11.219376

Rosé 3.805921

White 7.936544

Name: MinimunPrice, dtype: float64

calc\_mean(df['Alcohol Percentage'])

13.405111927831607

calc\_mean(df['JudgeRating'])

15.639492148346141

average\_max\_price\_by\_color = df.groupby('Color')['MaximumPrice'].mean()

# Display the average minimum price by color

print("Average maximum price by color:")

print(average\_max\_price\_by\_color)

Average maximum price by color:

Color

Red 16.334629

Rosé 6.526316

White 13.258729

Name: MaximumPrice, dtype: float64

calc\_mean(df["MaximumPrice"])

14.688573337788172

## Calculate the average Judge Rating by Region

average\_JudgeRating\_by\_color = df.groupby('Region')['JudgeRating'].mea()

**MEDIAN**

The median is a measure of central tendency that represents the middle value of a dataset when it is sorted in ascending order. If there is an odd number of data points, the median is the middle value. If there is an even number of data points, the median is the average of the two middle values.

In the codes below, we calculate the median for the variables minimum price, maximum price, alcohol percentage, and Judge Rating.

def calc\_median(variable):

#In any function you create, you should implement some

#Security logic - Error mitigation

#Create a conditional statement to process the error

if len(variable) == 0:

#Check the lenght of the data

#If we don't have enought data to process

#Terminate by returning nothing

return None

#In order to get the median value the array must be sorted

#In order to have the values sorted in the same array we need to ignore the previous values indexes as they

#will be given a new index number

variable = variable.sort\_values(ignore\_index = True)

n = len(variable)

if n % 2 == 0:

mid1 = variable[n // 2 - 1]

mid2 = variable[n // 2]

median = ((mid1 + mid2) / 2)

return median

else:

median = variable[n // 2]

return median

calc\_median(df["MinimunPrice"])

7.5

## Calculate the average minimum price by color

median\_min\_price\_by\_color = df.groupby('Color')['MinimunPrice'].median()

# Display the median minimum price by color

print("median minimum price by color:")

print(median\_min\_price\_by\_color)

median minimum price by color:

Color

Red 10.0

Rosé 4.0

White 7.5

Name: MinimunPrice, dtype: float64

calc\_median(df["MaximumPrice"])

10.0

## Calculate the median maximum price by color

median\_max\_price\_by\_color = df.groupby('Color')['MaximumPrice'].median()

# Display the median maximum price by color

print("median maximum price by color:")

print(median\_min\_price\_by\_color)

median maximum price by color:

Color

Red 10.0

Rosé 4.0

White 7.5

Name: MinimunPrice, dtype: float64

calc\_median(df['Alcohol Percentage'])

13.5

calc\_median(df['JudgeRating'])

15.5

**MODE**

The mode is a statistical measure that represents the most frequently occurring value in a dataset. It's the value that appears with the highest frequency among all the data points.

In the codes below, we calculate the mode for the variables minimum price, maximum price, alcohol percentage, and Judge Rating.

def calc\_mode(variable):

if len(variable) == 0:

return None

num\_freq = {}

for num in variable:

if num in num\_freq:

num\_freq[num] +=1

else:

num\_freq[num] = 1

max\_count = max(num\_freq.values())

mode = [num for num, count in num\_freq.items() if count == max\_count]

if len(mode) == len(variable):

return "No mode available"

return mode

calc\_mode(df["MinimunPrice"])

[4.0]

## Calculate the mode minimum price by color

mode\_min\_price\_by\_color = df.groupby('Color')['MinimunPrice'].apply(lambda x: x.mode().iloc[0])

# Display the mode minimum price by color

print("Mode minimum price by color:")

print(mode\_min\_price\_by\_color)

Mode minimum price by color:

Color

Red 7.5

Rosé 2.0

White 7.5

Name: MinimunPrice, dtype: float64

calc\_mode(df["MaximumPrice"])

[7.5]

## Calculate the mode maximum price by color

mode\_max\_price\_by\_color = df.groupby('Color')['MaximumPrice'].apply(lambda x: x.mode().iloc[0])

# Display the mode maximum price by color

print("Mode maximum price by color:")

print(mode\_max\_price\_by\_color)

Mode maximum price by color:

Color

Red 10.0

Rosé 4.0

White 10.0

Name: MaximumPrice, dtype: float64

calc\_mode(df['Alcohol Percentage'])

[13.5]

calc\_mode(df['JudgeRating'])

[15.5]

# DATA EXPLORATION

We start the analysis of data exploration by the correlation matrix.

# Select numerical columns

numerical\_columns = df[['AlcoholPercentage', 'MinimunPrice', 'MaximumPrice', 'Year', 'JudgeRating']]

# Calculate correlation matrix

correlation\_matrix = numerical\_columns.corr()

# Display correlation matrix

print("Correlation Matrix:")

print(correlation\_matrix)

# Alternatively, you can visualize the correlation matrix using a heatmap

import seaborn as sns

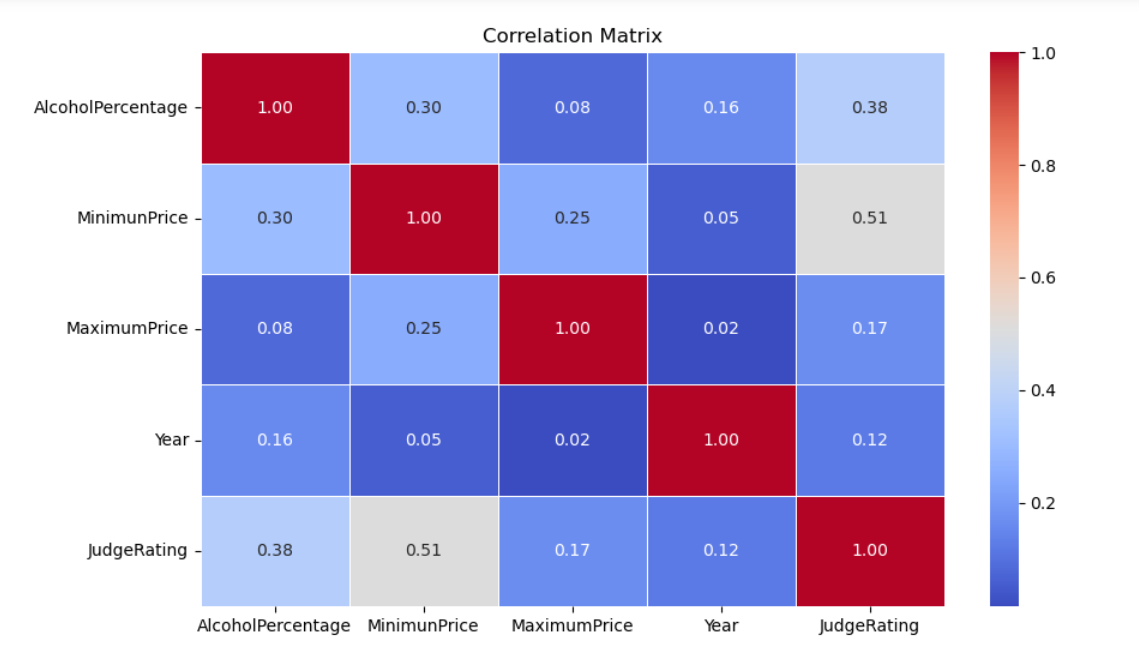
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)

plt.title('Correlation Matrix')

plt.show()



The results indicate that alcohol content tends to have a moderately positive correlation with judges' ratings, suggesting that wines with higher alcohol content generally receive higher ratings. Additionally, there is a moderate correlation between minimum price and judges' ratings, indicating that wines with higher minimum prices tend to receive higher ratings. However, the correlation between maximum price and judges' ratings is weaker compared to minimum price. This suggests that while wines with higher maximum prices may receive higher ratings, this relationship is not as strong as observed with minimum price. Additionally, there are weaker correlations between other variables, such as year and alcohol content, indicating that wines from more recent years tend to have slightly higher alcohol content, and between maximum price and alcohol content, suggesting that wines with higher maximum prices may have slightly higher alcohol content. These results provide valuable insights into how these variables are related to each other in the context of your wine data.

Next, we analyzed the Heatmap matrix for the numerical variables.

# Filter out non-numeric columns

numeric\_df = df.select\_dtypes(include='number')

# Defining the correlation matrix

correlation\_matrix = numeric\_df.corr()

# Creating the heatmap with a different colormap

plt.figure(figsize=(8, 5))

plt.title('Correlation Heatmap')

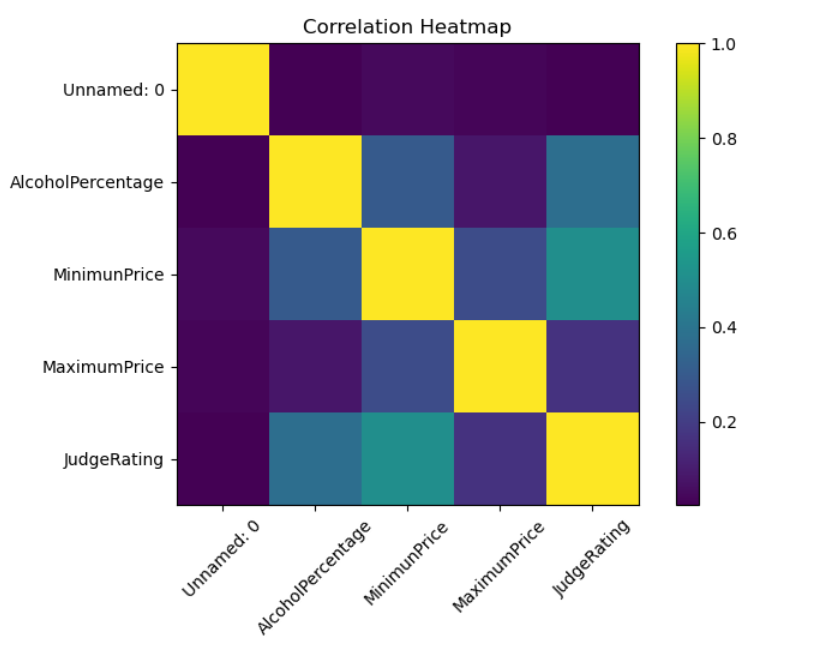
heatmap = plt.imshow(correlation\_matrix, cmap='viridis', interpolation='nearest') # Adjust the colormap here

PLT. Colorbar

plt.xticks(range(len(correlation\_matrix.columns)), correlation\_matrix.columns, rotation=45)

plt.yticks(range(len(correlation\_matrix.columns)), correlation\_matrix.columns)

plt.show()



We pretty much had the same result as the first matrix.

Next, we create a graph to evaluate the average minimum price and average maximum price per color of the wine.

# Plot the bar plot

means\_by\_color.plot(kind='bar', figsize=(8, 6))

plt.title('Mean Minimum and Maximum Value by Wine Color')

plt.xlabel('Wine Color')

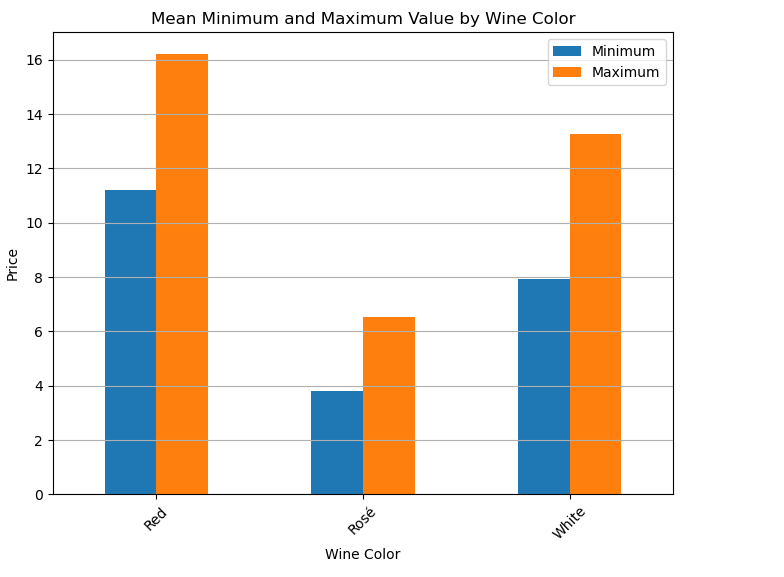
plt.ylabel('Price')

plt.xticks(rotation=45)

plt.legend(['Minimum', 'Maximum'])

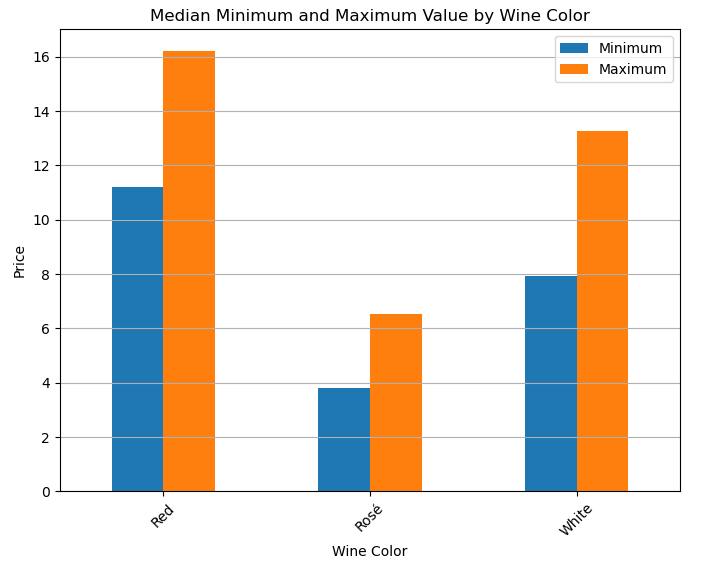
plt.grid(axis='y')

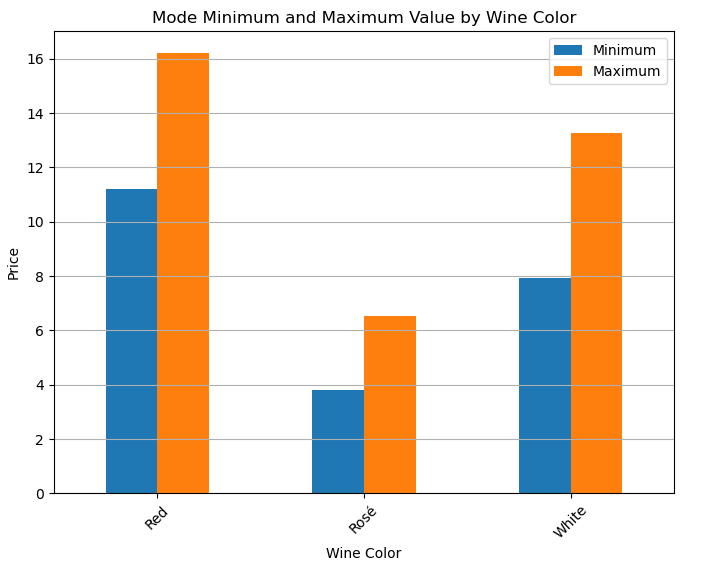
plt.show()



Through the graph, we can identify that red wine has the highest average maximum price.

We created the same graph for the median and mode





Another graph to demonstrate the average percentage of alcohol by Judge Rating.

Chart, Bar Chart

Auto-generated description

DOC Península de Setúbal: This region has the highest average alcohol percentage, at 14.00%. The judges' average rating for wines from this region is 15.60. Uruguay: In second place, we have Uruguay, with an average of 13.875% alcohol and an average judges' rating of 16.00. Argentina: Argentina ranks third, with an average of 13.828% alcohol and an average judges' rating of 16.14. Spain: Next, we have Spain, with an average of 13.797% alcohol and an average judges' rating of 16.06.

The next chart is the average Judge Rating for the top 10 regions

top\_10\_regions = df['Region'].value\_counts().nlargest(10).index.tolist()

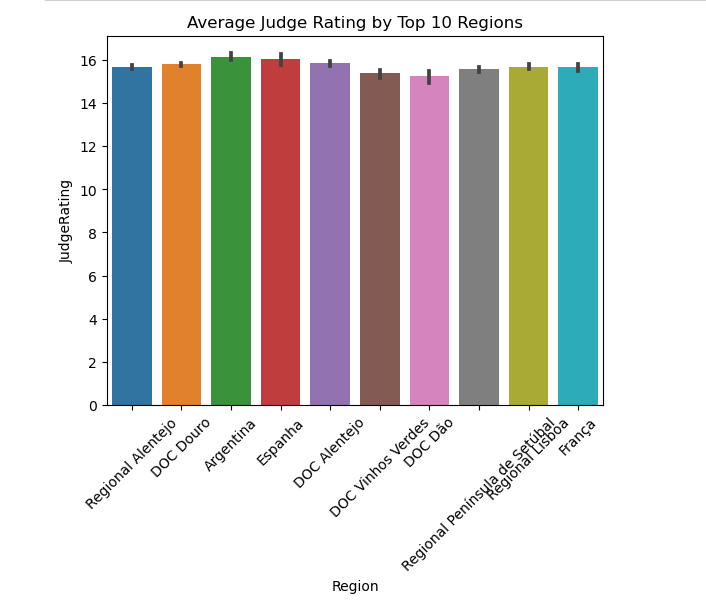
df\_top\_10 = df[df['Region'].isin(top\_10\_regions)]

sns.barplot(x='Region', y='JudgeRating', data=df\_top\_10)

plt.title('Average Judge Rating by Top 10 Regions')

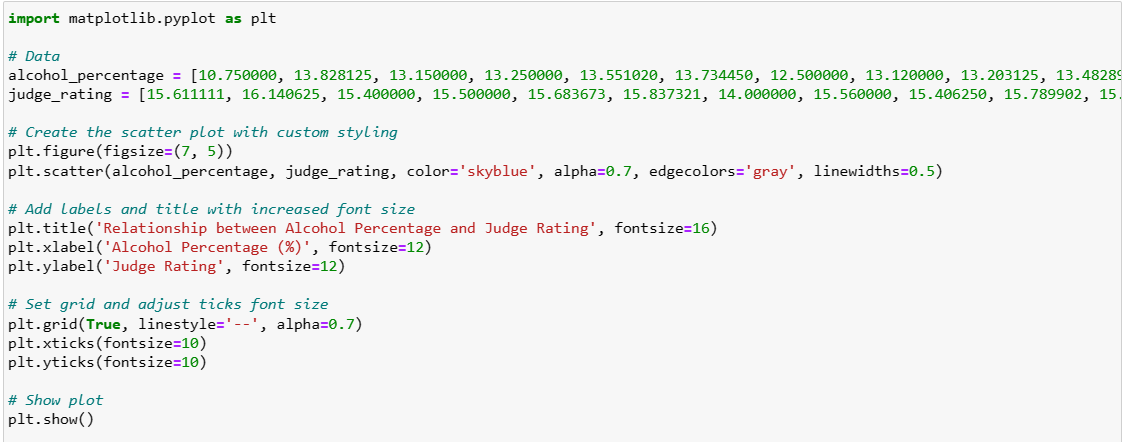
plt.xticks(rotation=45) # Rotate x-axis labels for better readability

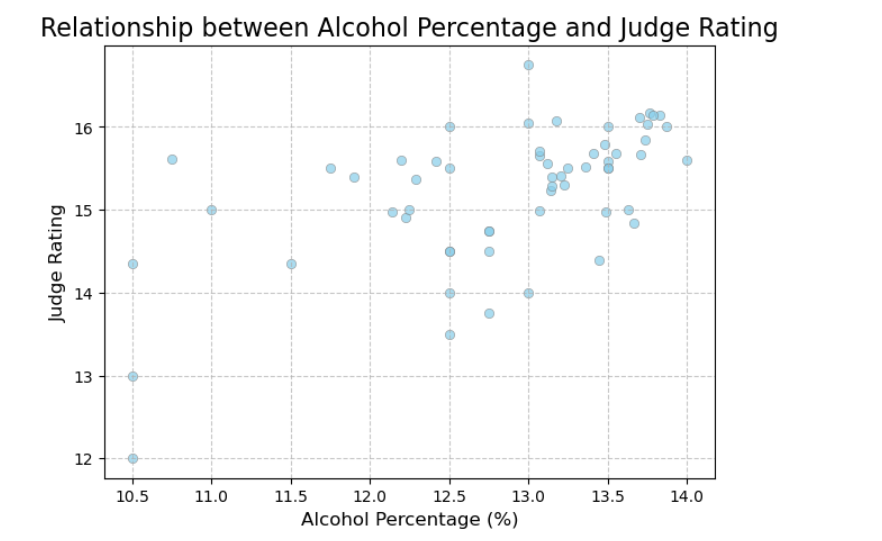
plt.show()



The top 10 regions with the highest number of reviews are Regional Alentejo, DOC Douro, DOC Alentejo, Regional Península de Setúbal, Spain, DOC Dão, DOC Vinhos Verdes, Argentina, France, and Regional Lisbon.

Below we plot a graph with the relationship between alcohol percentage and Judge Rating





These results demonstrate the relationship between the alcohol percentage in wines and judges' ratings. Generally, it is observed that wines with a higher alcohol percentage tend to receive higher ratings from judges, while wines with a lower alcohol percentage tend to receive lower ratings. However, there are exceptions, and other factors such as flavor, aroma, and complexity also influence judges' ratings.

# CONCLUSION

Through this introductory data analysis project, I was able to explore and better understand the relationship between different variables in a wine dataset. I learned that minimum price and alcohol percentage significantly influence judges' ratings, suggesting that wines with specific characteristics tend to receive better evaluations. Additionally, data visualization highlighted the importance of considering multiple factors, such as region and wine type, when analyzing and understanding consumer and wine expert preferences. This initial experience provided valuable insights into the data analysis process and motivated me to deepen my knowledge in this fascinating field.

# TASK 2 AND TASK

# Task 2 - Probability (Discrete):

• What is the probability of rolling exactly two 6s on five rolls of a fair die?

• The average number of workplace accidents per working week in a factory is 0.75.

Assuming that the distribution of injuries follows a Poisson distribution, find the probability that in a given week there will be no more than two accidents.

P(X=k) is the probability of getting exactly

k successes,

n is the total number of trials,

k is the number of successes,

p is the probability of success in a single trial.

In this case, we have n=5, k=2 (we want exactly two 6s) and (probability of getting a 6 on a single roll)

Probability of there being no more than two accidents in a given week, following a Poisson distribution:

Where is the probability of events occurring, λ is the average number of events in a fixed interval, and is the basis of the natural logarithm (approximately 2.71828).

In this case, we have λ=0.75 (average number of accidents per week) and we want to find the probability of there being no more than two accidents, i.e. P(X≤2).

Probability of getting exactly two 6's out of five rolls of a fair dice:

*P*(*X=*2)=

* is the binomial coefficient, which calculates the number of ways to choose 2 successes in 5 trials,
* is the probability of getting two 6's on a single roll,
* is the probability of not getting a 6 on the other three rolls.

Probability of there being no more than two accidents in a given week, following a Poisson distribution:

Calculation of odds:

Probability of getting exactly two 6's out of five rolls of a fair dice:

Probability of getting exactly two 6's out of five rolls of a fair dice:

= ≈ **0.1608**

Probability of there being no more than two accidents in a given week, following a Poisson distribution:

= ≈ 0.4724

= S 0.75 ≈ 0.3543

≈ 0.1329

Probability of no more than two accidents in a week:

p(c≤2)=p(c=0)+p(c=1)+p(c=2)

P(X≤2)≈0.4724+0.3543+0.1329

P(X≤2)≈0.9596

Therefore, the probability of there being no more than two accidents in a given week is approximately **0.9596.**

I used the Poisson distribution to model the number of workplace accidents in a given week, where the average number of accidents is known (0.75 per week). The formula of the Poisson distribution allows us to calculate the probability of a certain number of rare events occurring in a fixed interval.

The probability of there being no more than two accidents in a given week, following a Poisson distribution, is approximately 0.9596. This means that there is a high probability (approximately 95.96%) that, in a typical week, the number of workplace accidents does not exceed two, based on the historical average of accidents. This information is valuable for companies and safety authorities as it provides an understanding of the risk of accidents within a specific time period and can aid in the implementation of preventive measures.

# Task 3 - Probability (Continuous):

The time a person spends at Dublin Zoo is typically spread out with an average of 90 minutes and a standard deviation of 10 minutes.

Using this distribution, answer the following:

• If a visitor is selected at random, find the probability that they will spend a maximum of 85

minutes visiting the zoo.

• If a visitor is randomly selected, find the probability that they will spend at least 100

minutes visiting the zoo.

*0.8413 = 0.1587*

Given that you know that a particular visitor has spent more time than average visiting the zoo, how likely is it that they have spent more than 100 minutes there?

1. **Probability of a visitor spending a maximum of 85 minutes at the zoo:**

The probability of a visitor spending a maximum of 85 minutes at the zoo is approximately 0.3085. This means that there is about a 30.85% chance that a randomly selected visitor will spend 85 minutes or less at the zoo. This makes sense because we're looking at the area under the normal distribution curve up to the 85-minute point, and that area represents the proportion of visitors who spend less than 85 minutes at the zoo.

1. **Probability of a visitor spending at least 100 minutes at the zoo:**

The probability of a visitor spending at least 100 minutes at the zoo is approximately 0.1587. This means that there is about a 15.87% chance that a randomly selected visitor will spend 100 minutes or more at the zoo. This makes sense because we're looking at the area under the normal distribution curve from the 100-minute point to infinity, and this area represents the proportion of visitors who spend 100 minutes or more at the zoo.

1. **Probability of a visitor who spent more than 100 minutes at the zoo than average:**

The probability that a visitor who has spent more time than average will spend more than 100 minutes at the zoo is approximately 0.3174. This means that since we know that a visitor is already at the right tail end of the distribution (i.e., has spent more time than average), there is about a 31.74% chance that they will spend more than 100 minutes at the zoo. This makes sense because we're comparing the conditional probability of spending more than 100 minutes, given that they've already spent more time than average, versus the likelihood of spending more than 100 minutes overall.

# REFERENCES

McKinney, W. (2018). "Python for Data Analysis." O'Reilly Media.

Downey, A. (2014). "Think Stats: Exploratory Data Analysis in Python." O'Reilly Media.

Millman, K. J., & Aivazis, M. (2011). "Python for Scientists and Engineers." Computing in Science & Engineering.