🍷 Análisis de Calidad del Vino 🍷

Objetivo 🎯

El principal objetivo de este análisis es explorar cómo diferentes características químicas de distintos tipos de vino tinto se relacionan con la calidad que los catadores expertos les asignan. 🍷 En este conjunto de datos, se tienen detalles como cuál es su graduación alcohólica, qué tan ácido es, su nivel de acidez, entre otros. La idea es entender cómo todas estas cosas están conectadas y cómo influyen en cómo la gente percibe el vino. 🍇

Calidad del Vino 🍷

Fuente: Red Wine Quality Dataset

El dataset contiene información detallada sobre diversas propiedades químicas de diferentes variedades de vino tinto, así como calificaciones de calidad asignadas por expertos catadores. 🍷 Cuenta de múltiples atributos, incluidos aspectos como el contenido de alcohol, acidez, pH y concentraciones de diversos componentes. 🧪

Contexto Comercial 🍷

Suponemos que estamos trabajando en una empresa vitivinícola y observamos que el vino tinto argentino se erige como una potencia inigualable en la industria vinícola global, gracias a su excepcional calidad y diversidad de sabores derivados de las variadas regiones vinícolas del país. 🍇🌄 Esta riqueza enológica, respaldada por una tradición arraigada y técnicas modernas de producción, confiere a los vinos tintos argentinos una posición envidiable en los mercados internacionales. 🌍 Con una capacidad única para satisfacer paladares diversos, el vino tinto argentino ostenta un poder exportador que trasciende fronteras y consolida su reputación como un deleite irresistible para los amantes del buen vino en todo el mundo. 🍷

Problema Comercial 🍷

El problema radica en la creación de un modelo capaz de clasificar la calidad del vino en función de ciertas variables. Desde el inicio, nos encontramos con el desafío de que la calidad del vino está determinada por la mirada subjetiva de expertos. Será crucial realizar una selección precisa de las variables objetivas relevantes y estar atentos a posibles sesgos. Además, resultaría útil interpretar los resultados para obtener una comprensión más profunda de las relaciones existentes. 🍷

Contexto Analítico 🍷

El mismo cuenta con variables de entrada obtenidas mediante pruebas fisicoquímicas, las cuales son: acidez fija, acidez volátil, ácido cítrico, azúcar residual, cloruros, anhídrido sulfuroso libre, anhídrido sulfuroso total, densidad, pH, sulfatos, alcohol y por último una variable de salida que se basa en datos sensoriales y es: Calidad. 🍷

Las variables se miden y representan cosas distintas. La acidez fija es el conjunto de ácidos naturales en el vino, se mide en total de gramos por litro de vino. La acidez volátil surge de la diferencia entre la acidez total y la acidez fija. La acidez volátil se pretende que sea lo más baja posible porque afecta al sabor. 🍷

El ácido cítrico es un acidificante para corregir la acidez en mostos y vinos, además posee una acción estabilizante como antioxidante. Contribuye al equilibrio del gusto, aporta frescura y está presente en vinos y uvas en concentraciones entre 0,1-1 g/l. 🍷

El azúcar residual es la cantidad que queda en el vino después de haber sido fermentado. 🍷

El anhídrido sulfuroso libre y total, al igual que los sulfatos, son conservantes que se le añaden al vino y parecen afectar su calidad. 🍷

En cuanto al pH, la mayoría de los vinos se encuentra en un intervalo, según expertos, de 2,8 y 4, siendo 2,8 más ácido y 4 menos ácido. 🍷

acido y 4 menos acido.

Por último, están la densidad, la mayoría de los vinos están en el rango de 990-1160 (Kg/m3) y la graduación alcohólica.

Preguntas/Hipótesis

Este proyecto busca clasificar la calidad del vino según determinadas variables, sería pertinente entonces realizarse algunas preguntas:

- **¿Existe una correlación entre el contenido alcohólico y la calidad del vino?**

Librerias

In [69]:

```
#Librerias
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from bokeh.palettes import Spectral6
from bokeh.io import show, output_notebook
from bokeh.models import HoverTool, CategoricalColorMapper, ColumnDataSource, FactorRang
e
from bokeh.plotting import figure, output_file, show
import warnings
warnings.filterwarnings("ignore", category=UserWarning)
from bokeh.plotting import figure
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn.tree import DecisionTreeClassifier
import plotly.express as px


mpl.style.use('seaborn') # Estilo de los graficos
```

```
<ipython-input-69-3920dc451497>:27: MatplotlibDeprecationWarning:

The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer corr
espond to the styles shipped by seaborn. However, they will remain available as 'seaborn-
v0_8-<style>'. Alternatively, directly use the seaborn API instead.
```

Analisis exploratorio

In [28]:

```
from google.colab import drive
import os
drive.mount('/content/drive')

Calidadvinofile_path = '/content/drive/MyDrive/#42390 Data - Coder - Tripcevich/Calidad d
el vino.csv'

Calidadvino = pd.read_csv(Calidadvinofile_path)

Calidadvino.shape # Shape
```

```
Calidadvino # Ver el dataset
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount
("/content/drive", force_remount=True).
```

Out[28]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

1599 rows × 12 columns

In [3]:

```
Calidadvino.describe().round(2) # Primera vista estadistica y redondeo en dos decimales
```

Out[3]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
count	1599.00	1599.00	1599.00	1599.00	1599.00	1599.00	1599.00	1599.00	1599.00	1599.00	1599.00	1599.00
mean	8.32	0.53	0.27	2.54	0.09	15.87	46.47	1.00	3.31	0.66	10.42	5.64
std	1.74	0.18	0.19	1.41	0.05	10.46	32.90	0.00	0.15	0.17	1.07	0.81
min	4.60	0.12	0.00	0.90	0.01	1.00	6.00	0.99	2.74	0.33	8.40	3.00
25%	7.10	0.39	0.09	1.90	0.07	7.00	22.00	1.00	3.21	0.55	9.50	5.00
50%	7.90	0.52	0.26	2.20	0.08	14.00	38.00	1.00	3.31	0.62	10.20	6.00
75%	9.20	0.64	0.42	2.60	0.09	21.00	62.00	1.00	3.40	0.73	11.10	6.00
max	15.90	1.58	1.00	15.50	0.61	72.00	289.00	1.00	4.01	2.00	14.90	8.00

Análisis de los datos

Observaciones sobre el describe

Al realizar un primer vistazo a los datos, podemos notar que la información está completa y, a pesar de la presencia de algunos valores atípicos, estos parecen ser registros legítimos.

- **El promedio de calidad de los vinos es de 5.64. En términos generales, estamos tratando con vinos de calidad media.**
- **El tercer cuartil, que abarca el 75% de las observaciones, refleja vinos con una calificación de hasta seis puntos. Aquellos vinos que superan los siete puntos son más escasos en la muestra.**
- **Al examinar la columna 'fixed acidity', podemos notar valores atípicos que se distancian significativamente del tercer cuartil. Este fenómeno también se presenta en 'residual sugar', 'free sulfur dioxide', 'chlorides' y 'total sulfur dioxide'.**

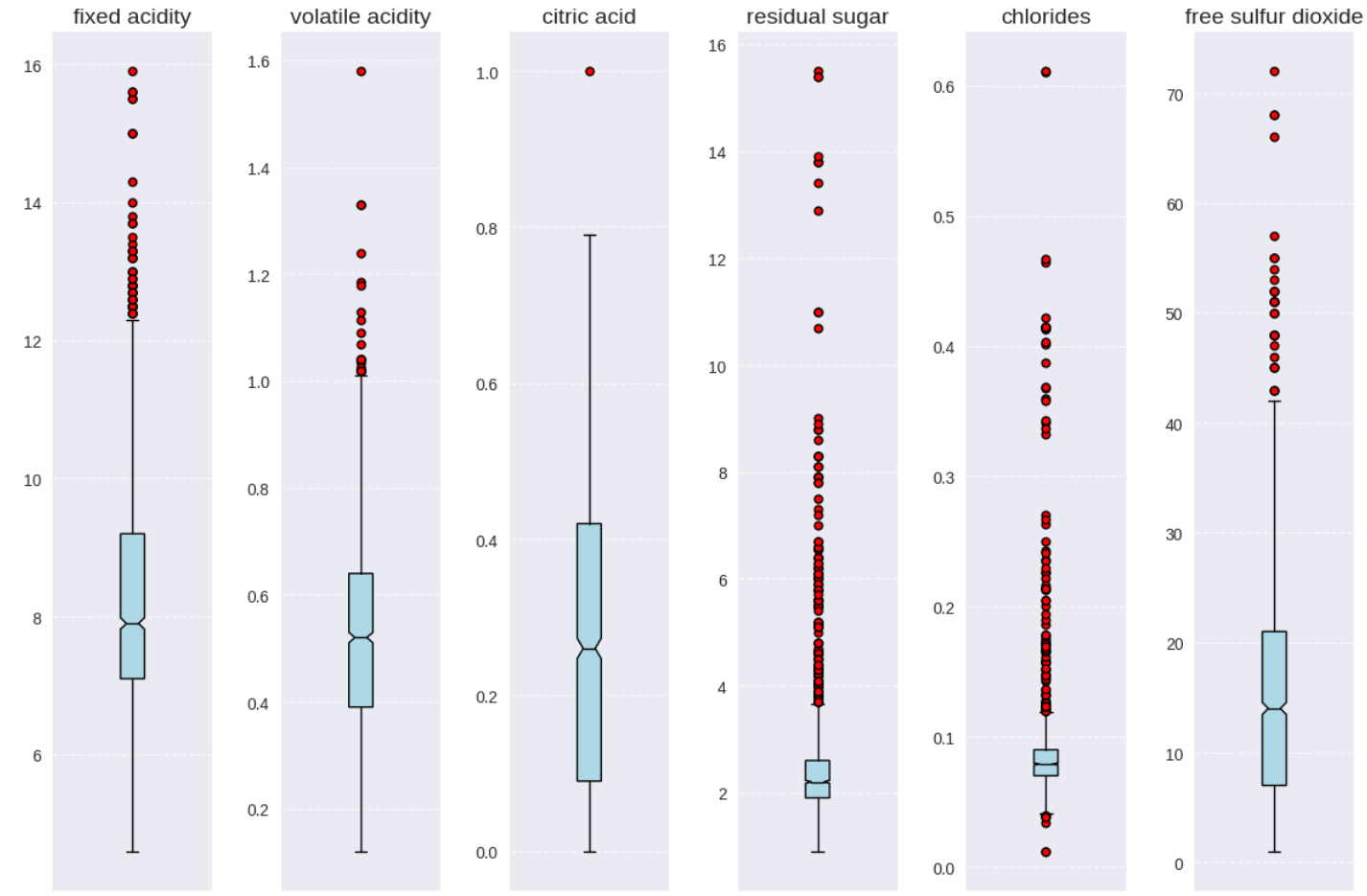
Estos resultados iniciales nos brindan una idea general de la distribución y las características clave de nuestros datos. Continuaremos explorando para obtener una comprensión más profunda y revelar posibles patrones y tendencias ocultas.

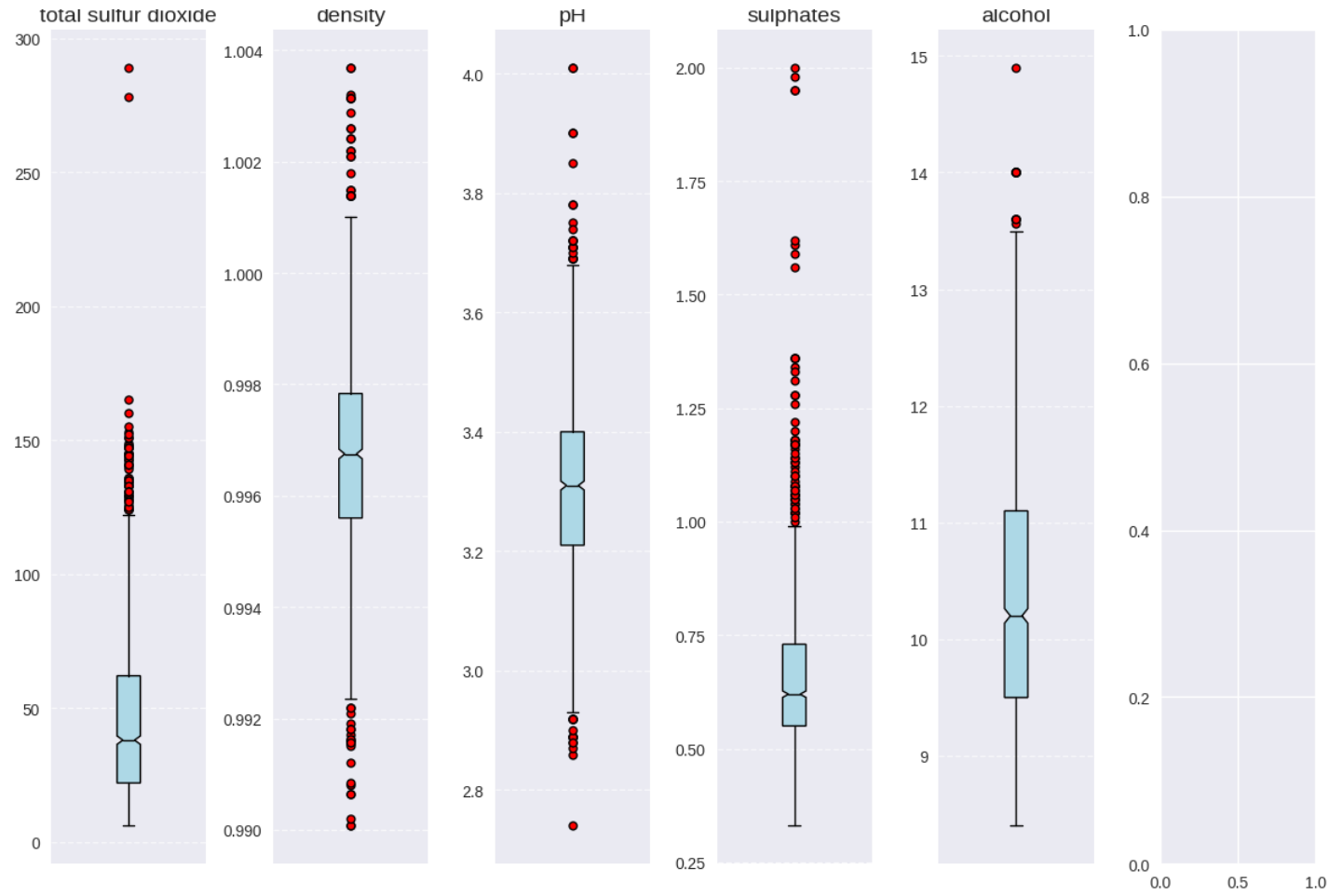
Outliers

Vista de los outliers

In [32]:

```
# Obtener las columnas numéricas
num_columns = Calidadvino.select_dtypes(include=['float64']).columns

# Definir el número de columnas por fila
columns_per_row = 6

# Calcular el número de filas necesarias
num_rows = -(-len(num_columns) // columns_per_row)  # Redondeo hacia arriba

# Generar gráficos organizados en filas y columnas
fig, axes = plt.subplots(num_rows, columns_per_row, figsize=(12, 8*num_rows))

axes = axes.flatten()

for i, column in enumerate(num_columns):
    ax = axes[i]
    ax.boxplot(Calidadvino[column], patch_artist=True, notch=True, boxprops=dict(facecol
or='lightblue', color='black'),
               capprops=dict(color='black'), whiskerprops=dict(color='black'), flierprop
s=dict(marker='o', markersize=5, markerfacecolor='red'),
               medianprops=dict(color='black'))
    ax.set_title(f'{column}', fontsize=14)
    ax.set_xticks([])  # Eliminar etiquetas en el eje X
    ax.set_xlabel('')
    ax.grid(axis='y', linestyle='--', alpha=0.7)

# Ajustar el espacio entre los subplots
plt.tight_layout()
plt.show()
```





Outliers

Conteo de valores atipicos por columna

In [5]:

```
Q1 = Calidadvino['fixed acidity'].quantile(0.25)
Q3 = Calidadvino['fixed acidity'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['fixed acidity'] < lower_bound) | (Calidadvino['fixed acidity']
> upper_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[5]:

```
49
```

In [6]:

```
Q1 = Calidadvino['volatile acidity'].quantile(0.25)
Q3 = Calidadvino['volatile acidity'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['volatile acidity'] < lower_bound) | (Calidadvino['volatile aci
dity'] > upper_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[6]:

```
19
```

In [7]:

```
Q1 = Calidadvino['citric acid'].quantile(0.25)
Q3 = Calidadvino['citric acid'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['volatile acidity'] < lower_bound) | (Calidadvino['volatile aci
dity'] > upper_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[7]:

```
43
```

In [8]:

```
Q1 = Calidadvino['residual sugar'].quantile(0.25)
Q3 = Calidadvino['residual sugar'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['residual sugar'] < lower_bound) | (Calidadvino['residual sugar
'] > upper_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[8]:

```
155
```

In [9]:

```
Q1 = Calidadvino['chlorides'].quantile(0.25)
Q3 = Calidadvino['chlorides'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['chlorides'] < lower_bound) | (Calidadvino['chlorides'] > upper
_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[9]:

```
112
```

In [10]:

```
Q1 = Calidadvino['free sulfur dioxide'].quantile(0.25)
Q3 = Calidadvino['free sulfur dioxide'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['free sulfur dioxide'] < lower_bound) | (Calidadvino['free sulf
ur dioxide'] > upper_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[10]:

```
30
```

In [11]:

```
Q1 = Calidadvino['total sulfur dioxide'].quantile(0.25)
Q3 = Calidadvino['total sulfur dioxide'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
```

```
outliers = ((Calidadvino['total sulfur dioxide'] < lower_bound) | (Calidadvino['total su
lfur dioxide'] > upper_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[11]:

55

In [12]:

```
Q1 = Calidadvino['density'].quantile(0.25)
Q3 = Calidadvino['density'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['density'] < lower_bound) | (Calidadvino['density'] > upper_bou
nd))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[12]:

45

In [13]:

```
Q1 = Calidadvino['pH'].quantile(0.25)
Q3 = Calidadvino['pH'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['pH'] < lower_bound) | (Calidadvino['pH'] > upper_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[13]:

35

In [14]:

```
Q1 = Calidadvino['sulphates'].quantile(0.25)
Q3 = Calidadvino['sulphates'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['sulphates'] < lower_bound) | (Calidadvino['sulphates'] > upper
_bound))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[14]:

59

In [15]:

```
Q1 = Calidadvino['alcohol'].quantile(0.25)
Q3 = Calidadvino['alcohol'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = ((Calidadvino['alcohol'] < lower_bound) | (Calidadvino['alcohol'] > upper_bou
nd))
outliers.sum()
#Suma del total de outliers en la columna
```

Out[15]:

Out[15]:

13

Nulos

In [16]:

```
Calidadvino.info() # Info - Averiguar si hay nulos
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

In [17]:

```
Calidadvino.isnull().any() # Verificar si hay nulos
```

Out[17]:

```
fixed acidity           False
volatile acidity        False
citric acid             False
residual sugar          False
chlorides               False
free sulfur dioxide     False
total sulfur dioxide    False
density                 False
pH                      False
sulphates               False
alcohol                 False
quality                 False
dtype: bool
```

Tratamiento de Outliers y Valores Nulos

Durante la exploración inicial de los datos, se pudo confirmar que los valores atípicos, o 'outliers', representan menos del 10% en cada variable del conjunto. Este hallazgo sugiere que los datos se mantienen mayoritariamente coherentes y dentro de los rangos esperados. Asimismo, la ausencia de valores faltantes en el conjunto refuerza la integridad de los datos. Es crucial tener en cuenta que la presencia de algunos valores anómalos no necesariamente indica la invalidez de los datos. En su conjunto, estos factores señalan que los datos son fiables para su posterior análisis y modelado, proporcionando una base sólida para la toma de decisiones fundamentadas.

Cambio de categorias

De numericas a "Malo", "Normal", "Bueno"

In [18]:

```
bins = [0, 5, 10]
labels = ['Malo a mediocre', 'Mediocre a bueno']
Calidadvino['quality'] = pd.cut(Calidadvino['quality'], bins=bins, labels=labels)
```

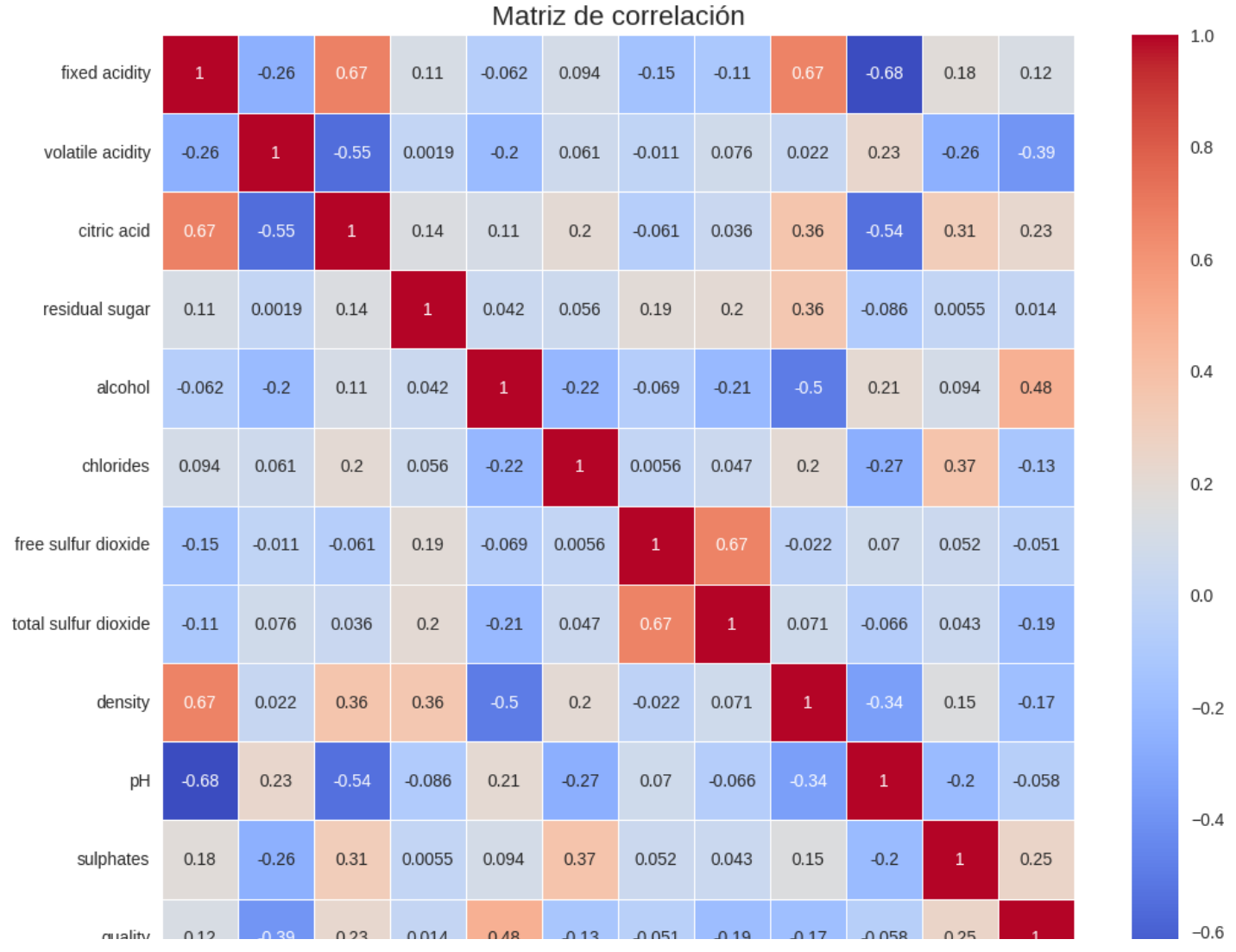
```
Calidadvino.head()
```

Out[18]:

	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	Malo a mediocre
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	Malo a mediocre
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	Malo a mediocre
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	Mediocre a bueno
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	Malo a mediocre

In [33]:

```
columnas_seleccionadas = Calidadvino[['fixed acidity', 'volatile acidity', 'citric acid'
, 'residual sugar',
                                      'alcohol', 'chlorides', 'free sulfur dioxide', 'to
tal sulfur dioxide',
                                      'density', 'pH', 'sulphates', 'quality']]

matriz_correlacion = columnas_seleccionadas.corr()

plt.figure(figsize=(12, 10))
sns.heatmap(matriz_correlacion, cmap='coolwarm', annot=True, annot_kws={"size": 10}, lin
ewidths=0.5)
plt.title('Matriz de correlación', fontsize=16)
plt.show()
```



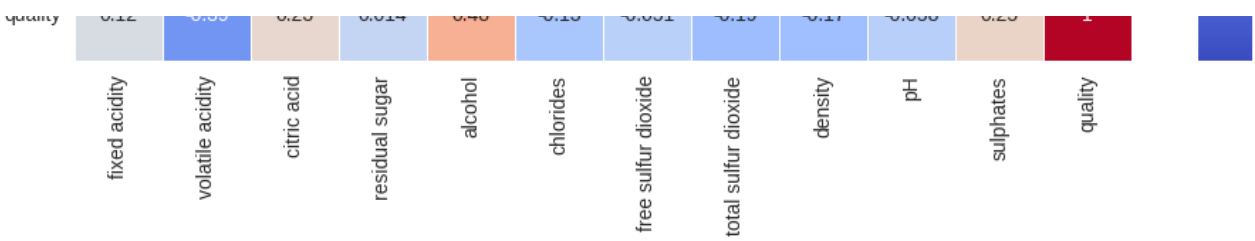


Tabla pivote

La tabla pivote muestra los valores promedios de cada variable segun las categorias que se crearon.

In [20]:

```
# Crear una tabla pivote con las otras características
tabla_pivote = pd.pivot_table(Calidadvino, values=['fixed acidity', 'volatile acidity',
'citric acid', 'residual sugar', 'chlorides', 'free sulfur dioxide', 'total sulfur dioxi
de', 'density', 'pH', 'sulphates', 'alcohol'], index='quality', aggfunc='mean')
print(tabla_pivote)
```

```
                 alcohol  chlorides  citric acid   density  fixed acidity  \
quality
Malo a mediocre   9.926478   0.092989     0.237755  0.997068       8.142204
Mediocre a bueno 10.855029   0.082661     0.299883  0.996467       8.474035

                 free sulfur dioxide        pH  residual sugar  sulphates  \
quality
Malo a mediocre            16.567204  3.311653        2.542070   0.618535
Mediocre a bueno           15.272515  3.310643        2.535965   0.692620

                 total sulfur dioxide  volatile acidity
quality
Malo a mediocre             54.645161          0.589503
Mediocre a bueno            39.352047          0.474146
```

🍷 El Alcohol y el Vino 🍷

Nuestro cliente ha solicitado la creación de un modelo para evaluar la calidad del vino 🍷, y para este propósito, disponemos del conjunto de datos que se ha descrito anteriormente. Es importante señalar que, tradicionalmente, los enólogos no consideran el contenido alcohólico al calificar el vino. Sin embargo, hemos centrado nuestra atención en este atributo.

A través de un análisis de correlación, hemos descubierto que el alcohol muestra una correlación de 0.48 con respecto a la calidad, superando a muchas otras características del vino. Además, hemos realizado una tabla pivote que ha proporcionado resultados interesantes. A continuación, se presentan algunas conclusiones clave de esta tabla pivote:

🍷 Alcohol vs. Otras Características: 🍷

- **Acidez Fija: Ambos grupos de vinos muestran niveles similares.**
- **Acidez Volátil: Los vinos clasificados como "Malo a mediocre" tienden a tener valores ligeramente más altos.**
- **Ácido Cítrico: Los vinos catalogados como "Mediocre a bueno" muestran niveles más altos de ácido cítrico.**
- **Azúcar Residual: Los niveles son comparables en ambos grupos.**
- **Cloruros: Los vinos "Malo a mediocre" tienden a tener niveles ligeramente más elevados.**
- **Dióxido de Azufre Libre: Los niveles son similares en ambos grupos.**
- **Dióxido de Azufur Total: Los vinos "Malo a mediocre" tienden a tener valores promedio más altos, con mayor variabilidad.**
- **Densidad: Los vinos "Mediocre a bueno" exhiben una densidad ligeramente más baja.**
- **pH: Los niveles de pH son similares en ambos grupos.**
- **Sulfatos: Los vinos catalogados como "Mediocre a bueno" tienen niveles más altos de sulfatos.**

🍷 Insights:

Basándonos en las observaciones, se destaca que el contenido de alcohol muestra una correlación significativa

Basándonos en las observaciones, se destaca que el contenido de alcohol muestra una correlación significativa con la calidad del vino, a pesar de la tradición que sugiere lo contrario. Esto sugiere que, en este conjunto de datos, el alcohol desempeña un papel crucial en la determinación de la calidad del vino. Será esencial considerar esta relación al desarrollar un modelo efectivo para evaluar la calidad del vino, asegurándonos de incluir el contenido alcohólico como una variable destacada en el proceso de clasificación.

Las diferencias entre las variables por categoria, ordenadas de mayor a menor

In [21]:

```
# Las diferencias absolutas entre las variables.
diferencia_var = abs(tabla_pivote.loc["Malo a mediocre"] - tabla_pivote.loc["Mediocre a bueno"])

# Ordena las diferencias de medias en orden descendente.
diferencia_var = diferencia_var.sort_values(ascending=False)
print(diferencia_var)
```

```
total sulfur dioxide    15.293115
free sulfur dioxide      1.294690
alcohol                  0.928551
fixed acidity            0.331831
volatile acidity         0.115356
sulphates                0.074085
citric acid              0.062128
chlorides                0.010328
residual sugar           0.006105
pH                       0.001010
density                  0.000602
dtype: float64
```

In [22]:

```
# Obtiene las tres variables con la mayor diferencia
top3 = diferencia_var.head(3)
print(top3)
```

```
total sulfur dioxide    15.293115
free sulfur dioxide      1.294690
alcohol                  0.928551
dtype: float64
```

In [23]:

```
# Crea una nueva tabla con las tres variables y las categorías
nuevo_dataset = Calidadvino[['quality'] + top3.index.tolist()]

# Muestra el nuevo dataset
print(nuevo_dataset.head())
```

```
            quality  total sulfur dioxide  free sulfur dioxide  alcohol
0   Malo a mediocre                  34.0                 11.0      9.4
1   Malo a mediocre                  67.0                 25.0      9.8
2   Malo a mediocre                  54.0                 15.0      9.8
3  Mediocre a bueno                  60.0                 17.0      9.8
4   Malo a mediocre                  34.0                 11.0      9.4
```

In [24]:

```
conteo_calidad = nuevo_dataset['quality'].value_counts() #cuenta como se distribuyen las categorias
print(conteo_calidad)
```

```
Mediocre a bueno    855
Malo a mediocre     744
Name: quality, dtype: int64
```

La distribucion de categorias es muy pareja.

In [25]:

```
# minimo por cada variable
minimos = nuevo_dataset.groupby('quality').min()
print(minimos)
```

```
                  total sulfur dioxide  free sulfur dioxide  alcohol
quality
Malo a mediocre                    6.0                  3.0      8.4
Mediocre a bueno                   6.0                  1.0      8.4
```

In [26]:

```
# maximo por cada variable
maximos = nuevo_dataset.groupby('quality').max()
print(maximos)
```

```
                  total sulfur dioxide  free sulfur dioxide  alcohol
quality
Malo a mediocre                  155.0                 68.0     14.9
Mediocre a bueno                 289.0                 72.0     14.0
```

Existen vinos de las dos categorias con el mismo minimo en el alcohol y tambien existen vinos malos con gran graduacion alcoholica

In [27]:

```
# Cambio de las etiquetas de quality a numero
Etiquetascalidad = Calidadvino['quality'].map({'Malo a mediocre': 0, 'Mediocre a bueno':
1})
```

Histograma PH

El grafico muestra como se distribuye la muestra segun la cantidad de pH en el vino. Vemos que la mayoria de los casos se concentra entre tres y cuatro, siendo estos datos flotantes

In [42]:

```
Alcohol = Calidadvino.alcohol # Arreglo para luego con los datos graficar

Alcohol.values.flatten()
```

Out[42]:

```
array([ 9.4,  9.8,  9.8, ..., 11. , 10.2, 11. ])
```

In [43]:

```
len(PH.values.flatten())
```

Out[43]:

```
1599
```

In [37]:

```
PH = Calidadvino.pH # Arreglo para luego con los datos graficar

PH.values.flatten()
```

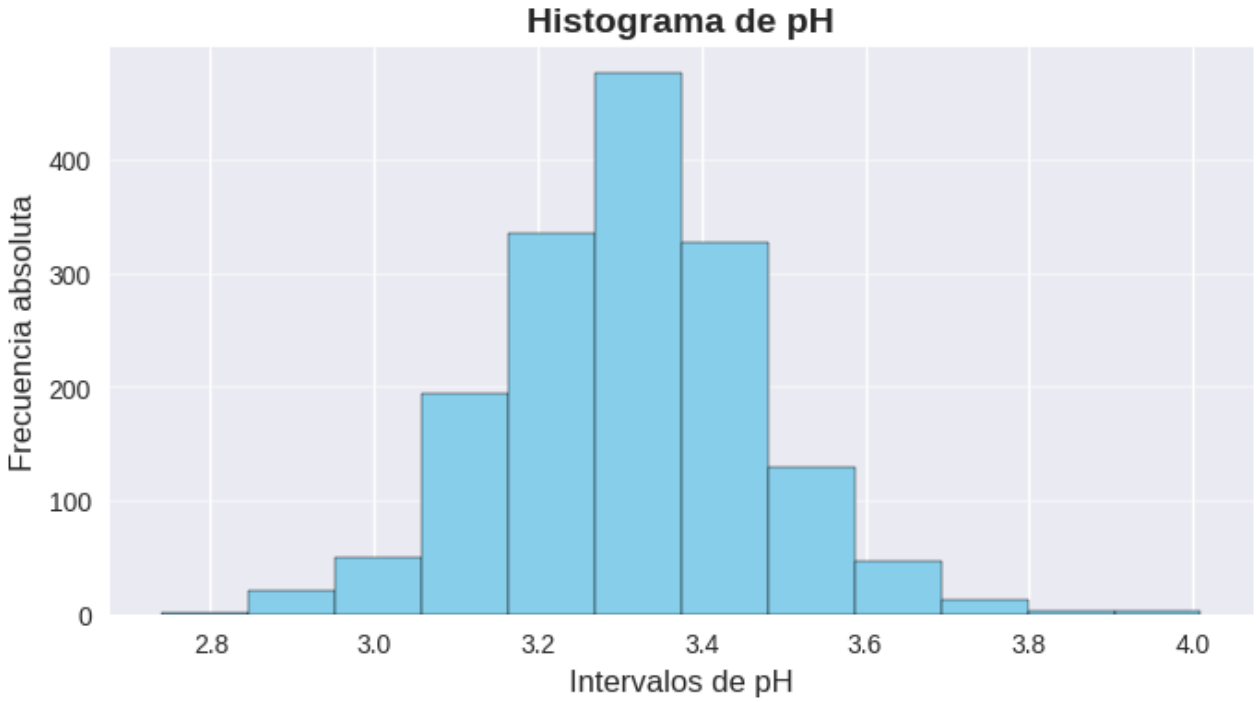
Out[37]:

```
array([3.51, 3.2 , 3.26, ..., 3.42, 3.57, 3.39])
```

In [39]:

```
fig, ax = plt.subplots(figsize=(8, 4))
ax.hist(PH.values.flatten(), bins=12, color='skyblue', edgecolor='black')
ax.set_title('Histograma de pH', fontsize=14, fontweight='bold')
ax.set_xlabel('Intervalos de pH', fontsize=12)
ax.set_ylabel('Frecuencia absoluta', fontsize=12)
```

```
ax.grid(axis='y', alpha=0.5)
plt.show()
```

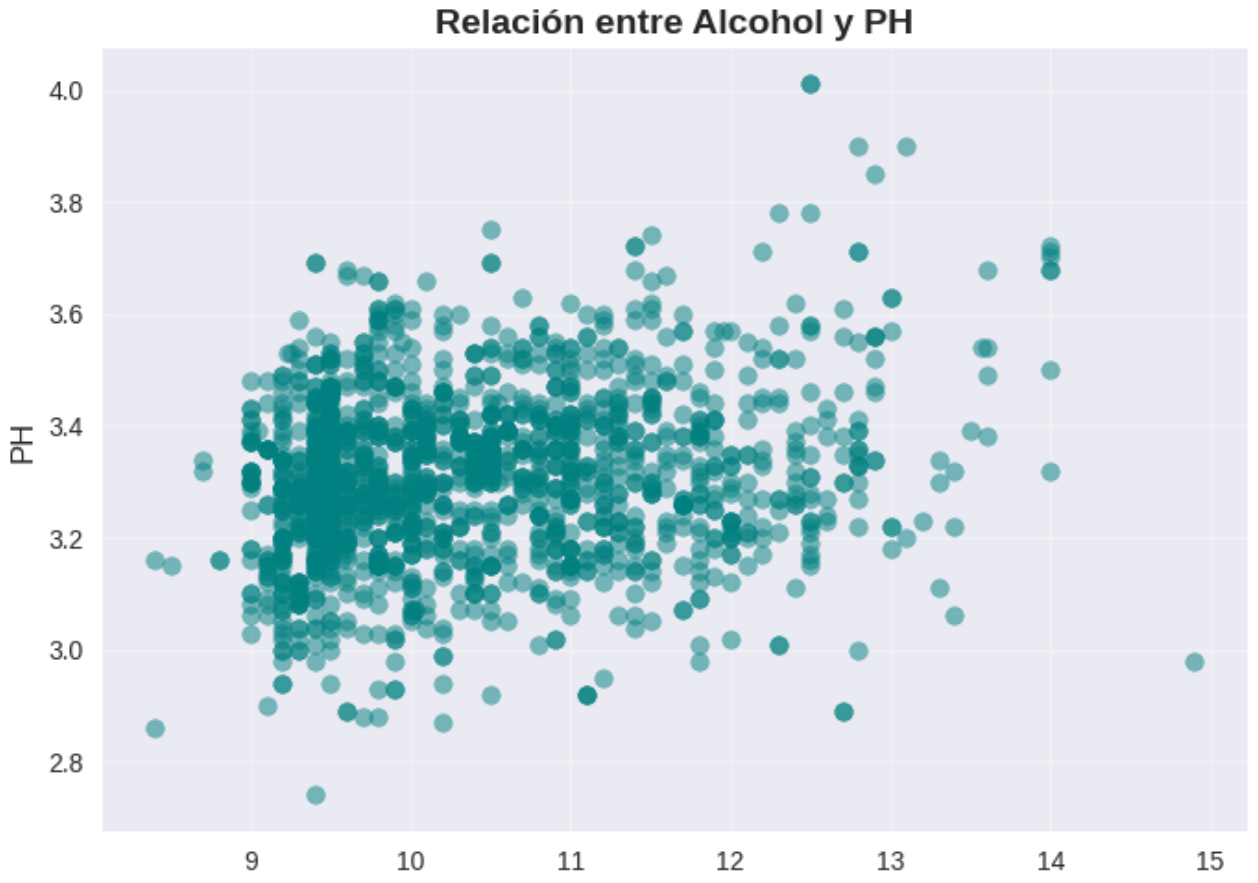


Dispersion

El grafico muestra la relacion entre la graduacion alcoholica y el pH, parece indicar, una baja corralacion directa

In [44]:

```
fig, ax = plt.subplots()
ax.scatter(Alcohol, PH, alpha=0.5, color='teal')
ax.set_title('Relación entre Alcohol y PH', fontsize=14, fontweight='bold')
ax.set_xlabel('Alcohol', fontsize=12)
ax.set_ylabel('PH', fontsize=12)
ax.grid(alpha=0.3)
plt.show()
```



Alcohol

Grafico de torta

Muestra la composicion de la muestra segun la calidad del vino

In [47]:

```
Calidadvino.insert(
    loc=1,
    column = 'IDVino',
    value = [i for i in range(0,1599)]
)

Calidadvino
```

Out[47]:

	fixed acidity	IDVino	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
1	7.8	1	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5
2	7.8	2	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5
3	11.2	3	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6
4	7.4	4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1594	6.2	1594	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1595	5.9	1595	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1596	6.3	1596	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1597	5.9	1597	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1598	6.0	1598	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

1599 rows × 13 columns

In [48]:

```
Puntaje=Calidadvino[['IDVino','quality']].groupby(by='quality').count()

Puntaje
```

Out[48]:

	IDVino
quality	
3	10
4	53
5	681
6	638
7	199
8	18

In [49]:

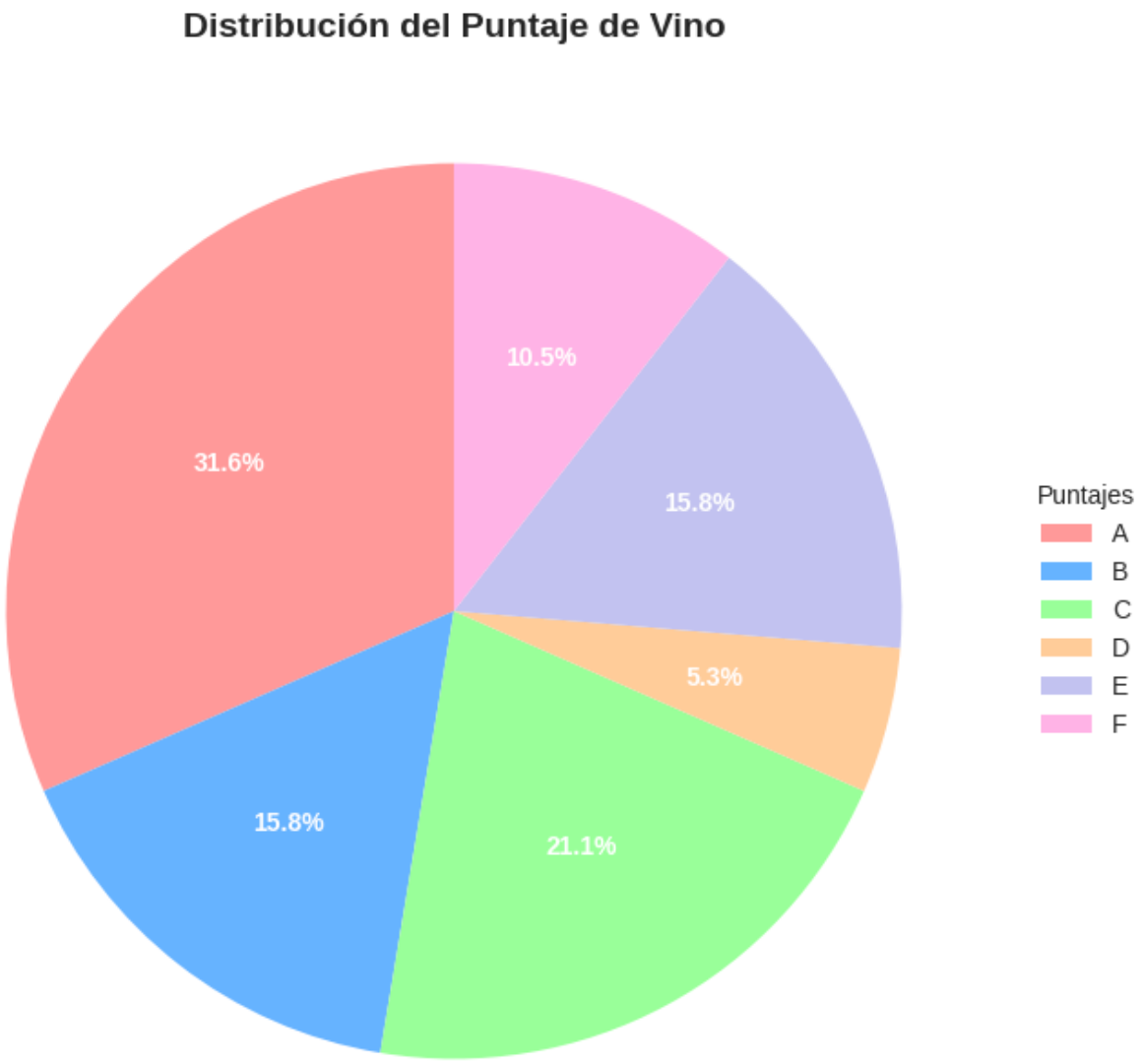
```
Puntaje.index
```

Out[49]:

```
Int64Index([3, 4, 5, 6, 7, 8], dtype='int64', name='quality')
```

In [57]:

```
import matplotlib.pyplot as plt

# Datos de ejemplo para Puntaje.IDVino y Puntaje.index
Puntaje = {'IDVino': [30, 15, 20, 5, 15, 10],
           'index': ['A', 'B', 'C', 'D', 'E', 'F']}

fig, ax = plt.subplots(figsize=(8, 8))

colors = ['#ff9999', '#66b3ff', '#99ff99', '#ffcc99', '#c2c2f0', '#ffb3e6']

wedges, texts, autotexts = ax.pie(Puntaje['IDVino'], colors=colors, startangle=90, autop
ct='%1.1f%%', textprops=dict(color="w"))

ax.legend(wedges, Puntaje['index'], title="Puntajes", loc="center left", bbox_to_anchor=
(1, 0, 0.5, 1))

plt.setp(autotexts, size=10, weight="bold")
ax.set_title('Distribución del Puntaje de Vino', fontsize=14, fontweight='bold')

plt.show()
```



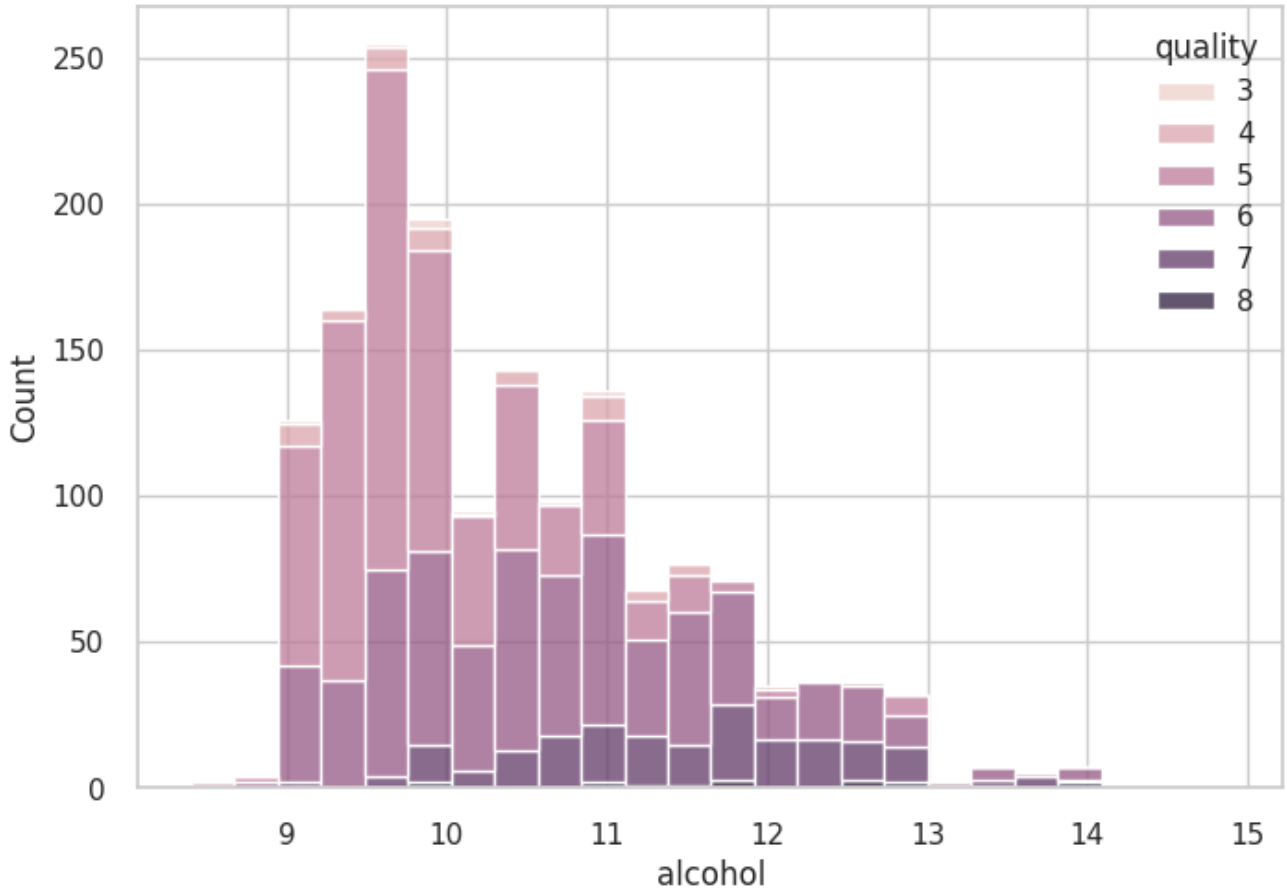
Histograma

Muestra como se distribuye la muestra segun la graduacion alcoholica y por que calidad de vino estan compuestas esas frecuencias. Pareciera indicar que los vinos que tienen mayor graduacion alcoholica son de

mayor calidad.

In [ ]:

```
sns.set_theme(style="whitegrid")
sns.histplot(data=Calidadvino, x="alcohol", hue="quality", multiple="stack",)
plt.show()
```



In [70]:

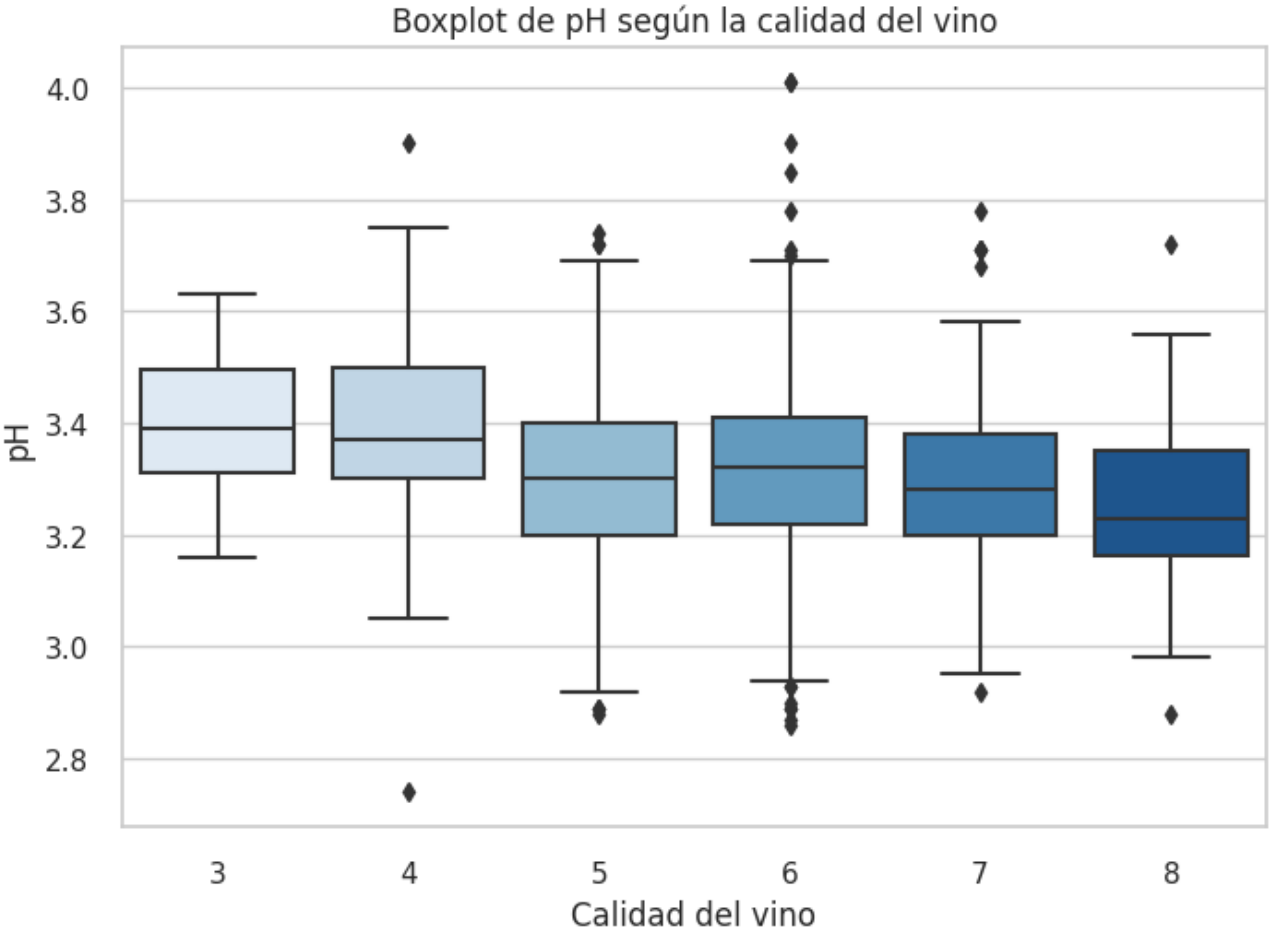
```
fig = px.histogram(Calidadvino, x="alcohol", color="quality", marginal="rug")
fig.update_layout(title_text="Distribución de alcohol por calidad del vino")
fig.show()
```

Caja y bigote

Muestra como es la distribucion del pH para cada calidad de vino

In [65]:

```
sns.set_theme(style="whitegrid")
ax = sns.boxplot(x="quality", y="pH", data=Calidadvino, palette="Blues")
ax.set_title("Boxplot de pH según la calidad del vino")
ax.set_xlabel("Calidad del vino")
ax.set_ylabel("pH")

plt.show()
```

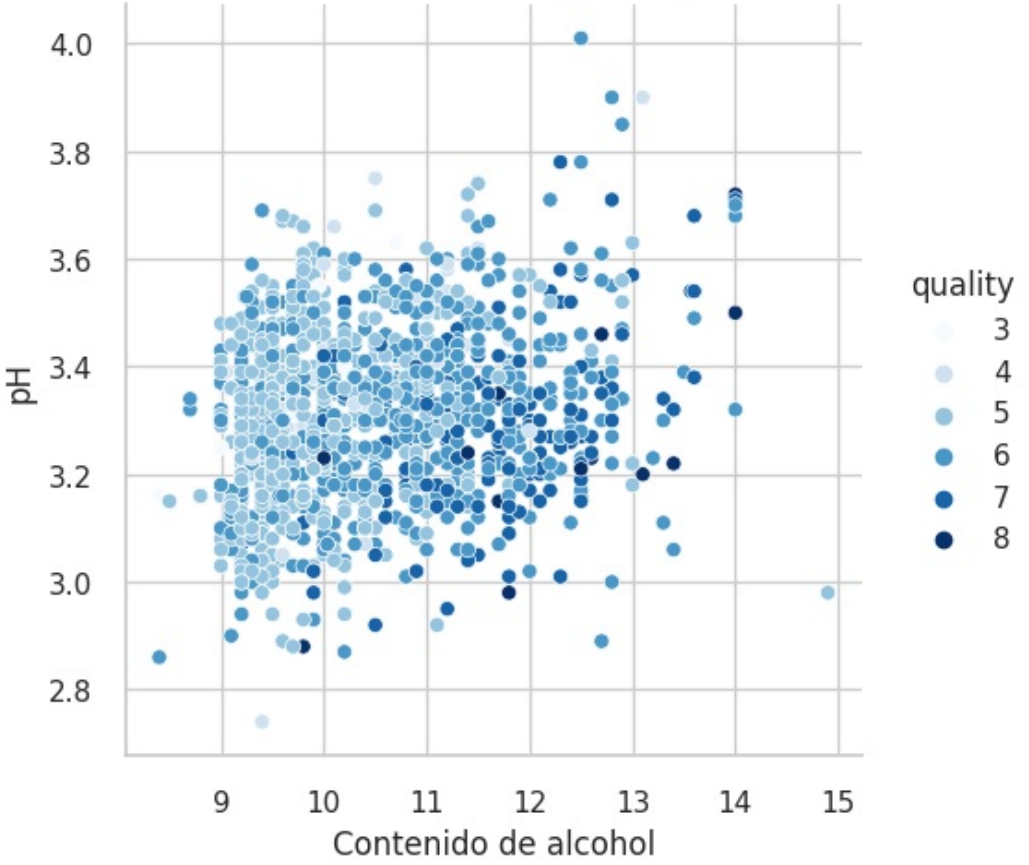


Dispersion

El grafico muestra la relacion entre la graduacion alcoholica y el pH, parece indicar, difereciandolos segun calidad del vino

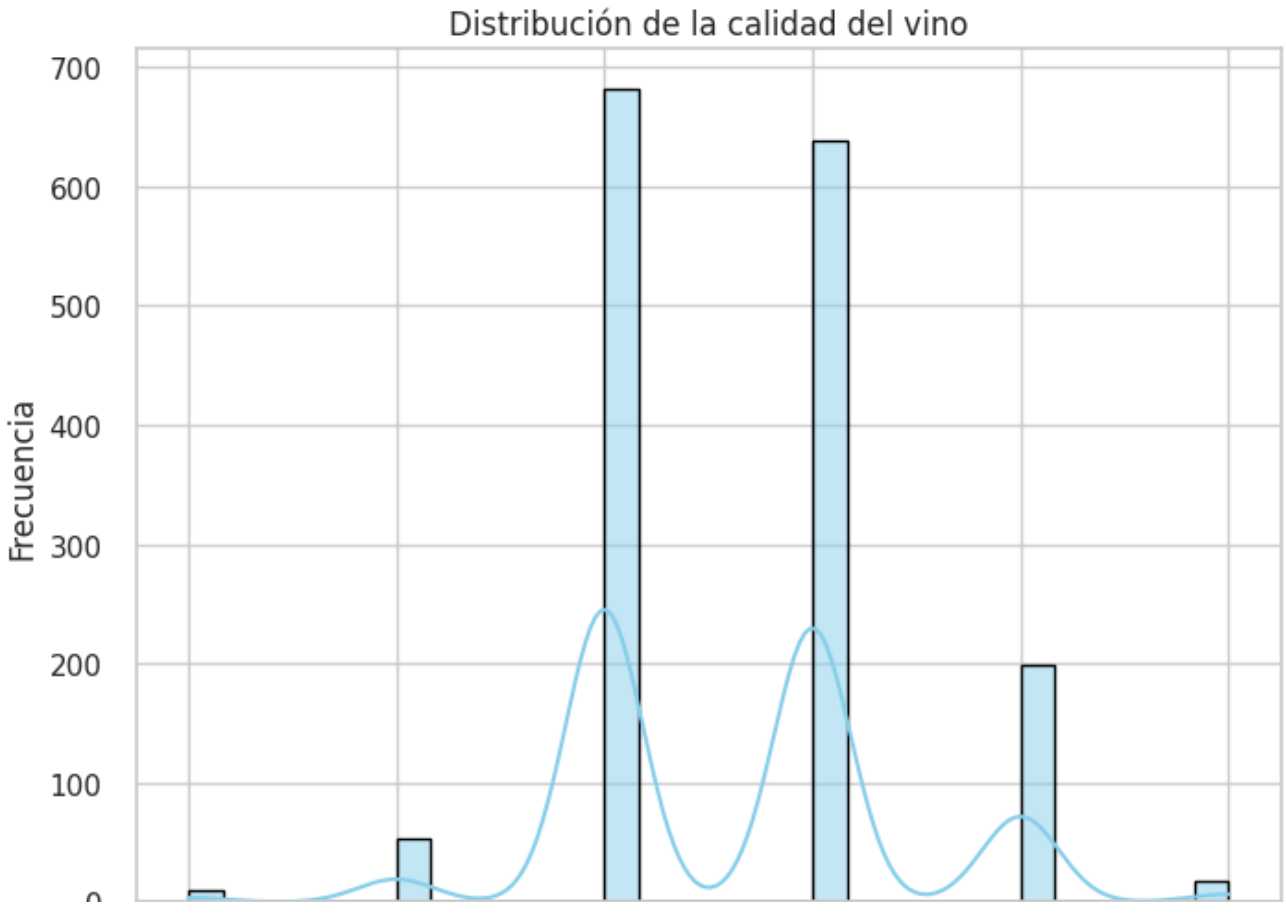
In [67]:

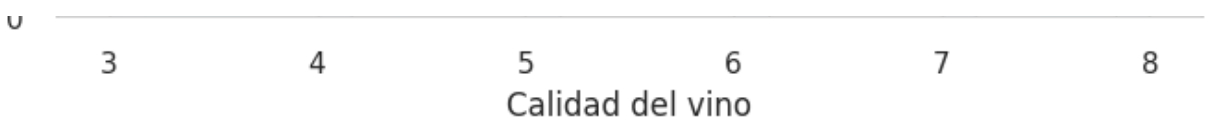
```
sns.set_theme(style="whitegrid")
sns.relplot(data=Calidadvino, x="alcohol", y="pH", hue="quality", kind="scatter", palett
e="Blues")
plt.title("Relación entre el contenido de alcohol y pH según la calidad del vino")
plt.xlabel("Contenido de alcohol")
plt.ylabel("pH")
plt.show()
```



In [68]:

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.set_theme(style="whitegrid")
plt.figure(figsize=(8, 6))
sns.histplot(data=Calidadvino, x="quality", kde=True, color='skyblue', edgecolor='black')
plt.title("Distribución de la calidad del vino")
plt.xlabel("Calidad del vino")
plt.ylabel("Frecuencia")
plt.show()
```





PCA

In [ ]:

```
# Eliminar la columna 'quality' de los datos
Datos = Calidadvino.drop(columns=['quality'])
```

In [ ]:

```
# Estandarizar los datos
scaler = StandardScaler()
Datosescalados = scaler.fit_transform(Datos)
```

In [ ]:

```
# Aplicar PCA
Ncomponentes = 10  # La cantidad de columnas menos el index y quality
Pca = PCA(n_components=Ncomponentes)
Componentesprincipales = Pca.fit_transform(Datosescalados)
```

In [ ]:

```
# Crear un nuevo DataFrame con los componentes principales
Column_names = [f'Componente {i+1}' for i in range(Ncomponentes)]
Componentesdf = pd.DataFrame(data=Componentesprincipales, columns=Column_names)
```

In [ ]:

```
# Combinar los componentes principales con las etiquetas de calidad
Componentesetiquetas_df = pd.concat([Componentesdf, Etiquetascalidad], axis=1)
print(Componentesetiquetas_df)
```

```
      Componente 1  Componente 2  Componente 3  Componente 4  Componente 5  \
0        -1.619530      0.450950     -1.774454      0.043740      0.067014
1        -0.799170      1.856553     -0.911690      0.548066     -0.018392
2        -0.748479      0.882039     -1.171394      0.411021     -0.043531
3         2.357673     -0.269976      0.243489     -0.928450     -1.499149
4        -1.619530      0.450950     -1.774454      0.043740      0.067014
...            ...           ...           ...           ...           ...
1594     -2.150500      0.814286      0.617063      0.407687     -0.240936
1595     -2.214496      0.893101      1.807402      0.414003      0.119592
1596     -1.456129      0.311746      1.124239      0.491877      0.193716
1597     -2.270518      0.979791      0.627965      0.639770      0.067735
1598     -0.426975     -0.536690      1.628955     -0.391716      0.450482

      Componente 6  Componente 7  Componente 8  Componente 9  Componente 10  \
0        -0.913921     -0.161043     -0.282258      0.005098      -0.267759
1         0.929714     -1.009829      0.762587     -0.520707       0.062833
2         0.401473     -0.539553      0.597946     -0.086857      -0.187442
3        -0.131017      0.344290     -0.455375      0.091577      -0.130393
4        -0.913921     -0.161043     -0.282258      0.005098      -0.267759
...            ...           ...           ...           ...            ...
1594      0.054835      0.170812     -0.355866     -0.971524       0.356851
1595     -0.674711     -0.607970     -0.247640     -1.058135       0.478879
1596     -0.506410     -0.231082      0.079382     -0.808773       0.242248
1597     -0.860408     -0.321487     -0.468876     -0.612248       0.779404
1598     -0.496154      1.189132      0.042176      0.404309       0.779440

     quality
0          1
1          1
2          1
3          1
4          1
```

```
...       ...
1594        1
1595        1
1596        1
1597        1
1598        1

[1599 rows x 11 columns]
```

In [ ]:

```
# Aplicar PCA a los datos
Componentesprincipales = Pca.fit_transform(Datos)
```

In [ ]:

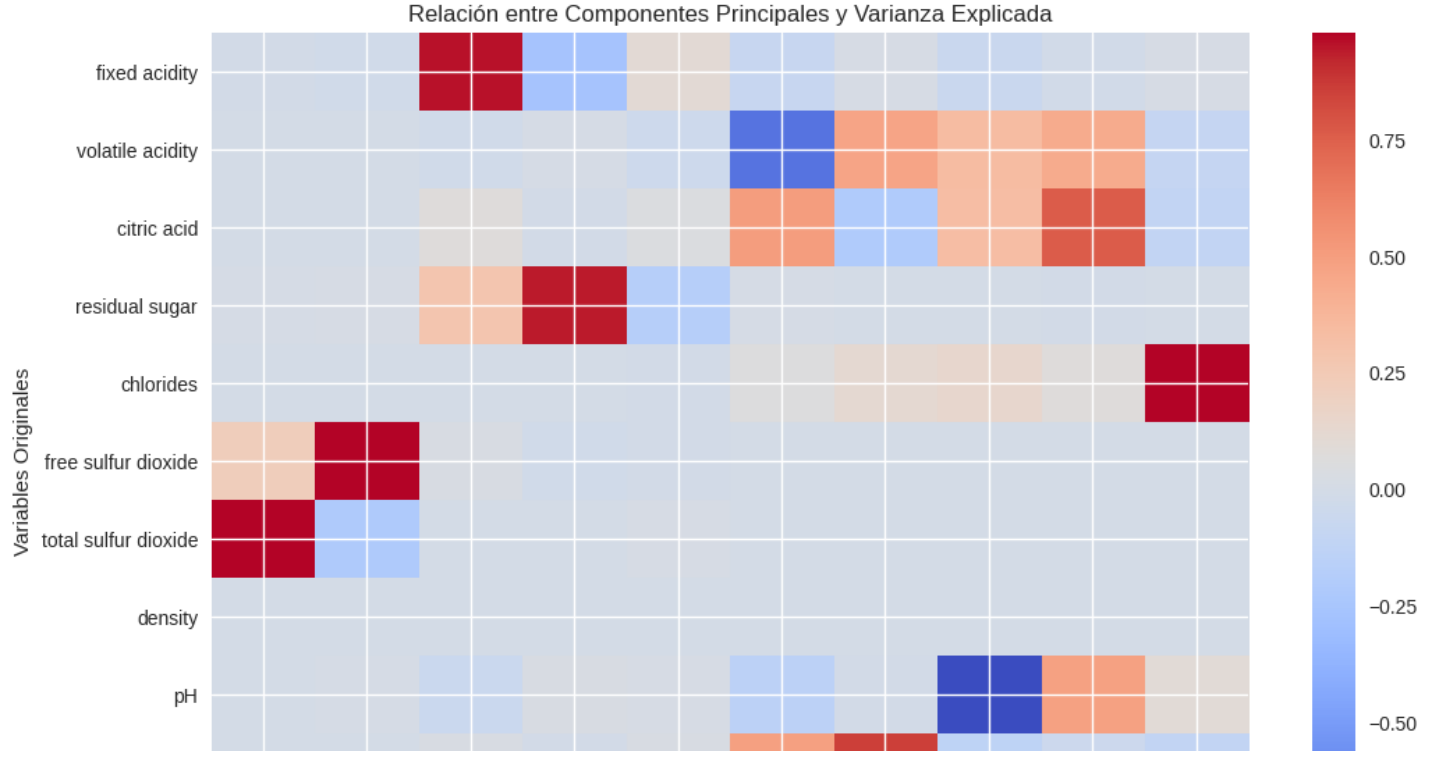
```
# Obtener la varianza explicada
varianza_explicada = Pca.explained_variance_ratio_
varianza_explicada_redondeada = [round(valor, 4) for valor in varianza_explicada]

print(f"Varianza Explicada por Componente Principal: {varianza_explicada_redondeada}")
```

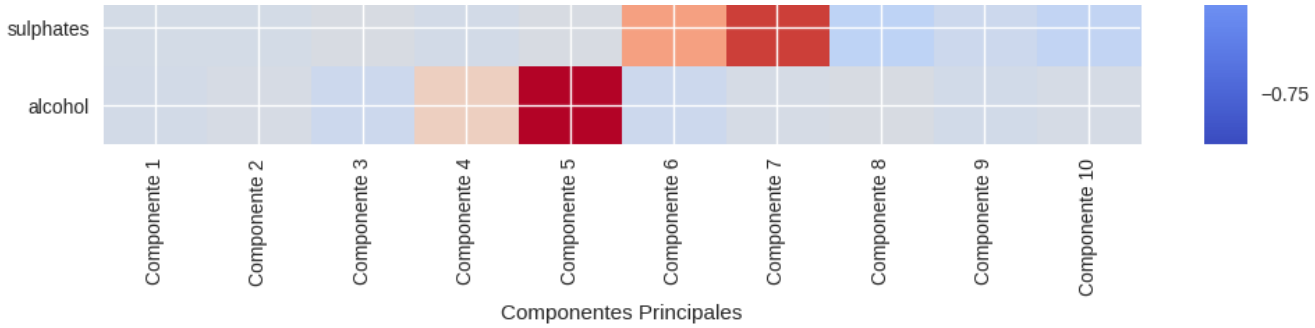
```
Varianza Explicada por Componente Principal: [0.9466, 0.0484, 0.0026, 0.0015, 0.0009, 0.0
, 0.0, 0.0, 0.0, 0.0]
```

El componente uno explica la mayor parte de la varianza

In [ ]:

```
# Obtener los vectores de carga de los componentes
vectores_de_carga = Pca.components_

# Crear un DataFrame para visualizar los vectores de carga
vectores_de_carga_df = pd.DataFrame(data=vectores_de_carga, columns=Datos.columns, index
=Column_names)


plt.figure(figsize=(12, 8))
plt.imshow(vectores_de_carga_df.T, cmap='coolwarm', aspect='auto')
plt.colorbar()
plt.xticks(range(Ncomponentes), Column_names, rotation=90)
plt.yticks(range(len(Datos.columns)), Datos.columns)
plt.xlabel('Componentes Principales')
plt.ylabel('Variables Originales')
plt.title('Relación entre Componentes Principales y Varianza Explicada')
plt.show()
```





Explicacion matriz

El gráfico de calor en la matriz de varianza-componentes nos muestra cómo las cosas que medimos originalmente (como acidez o contenido de azúcar) están relacionadas con los nuevos grupos que encontramos llamados "Componentes principales". Cada cuadrante en el gráfico nos dice cuánto ayuda cada medición a entender esos grupos nuevos. Si el cuadrante está de un color más fuerte, significa que esa cosa que medimos tiene más que ver con uno de los grupos nuevos. En resumen, nos ayuda a saber qué mediciones son más importantes para entender estos grupos especiales que encontramos.

K-nearest neighbor

In [ ]:

```
# Eliminar la columna 'quality' de los datos
Datos = Calidadvino.drop(columns=['quality'])
```

In [ ]:

```
# Estandarizar los datos
scaler = StandardScaler()
Datosescalados = scaler.fit_transform(Datos)
```

In [ ]:

```
# Dividir el conjunto de datos en entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(Datosescalados, Etiquetascalidad, te
st_size=0.2, random_state=42)
```

Tamaño de entrenamiento 80% Tamaño test 20%

In [ ]:

```
# Crear un modelo K-NN
knn = KNeighborsClassifier(n_neighbors=3)
```

In [ ]:

```
# Entrenar el modelo
knn.fit(X_train, y_train)
```

Out[ ]:

```
▾ KNeighborsClassifier
KNeighborsClassifier(n_neighbors=3)
```

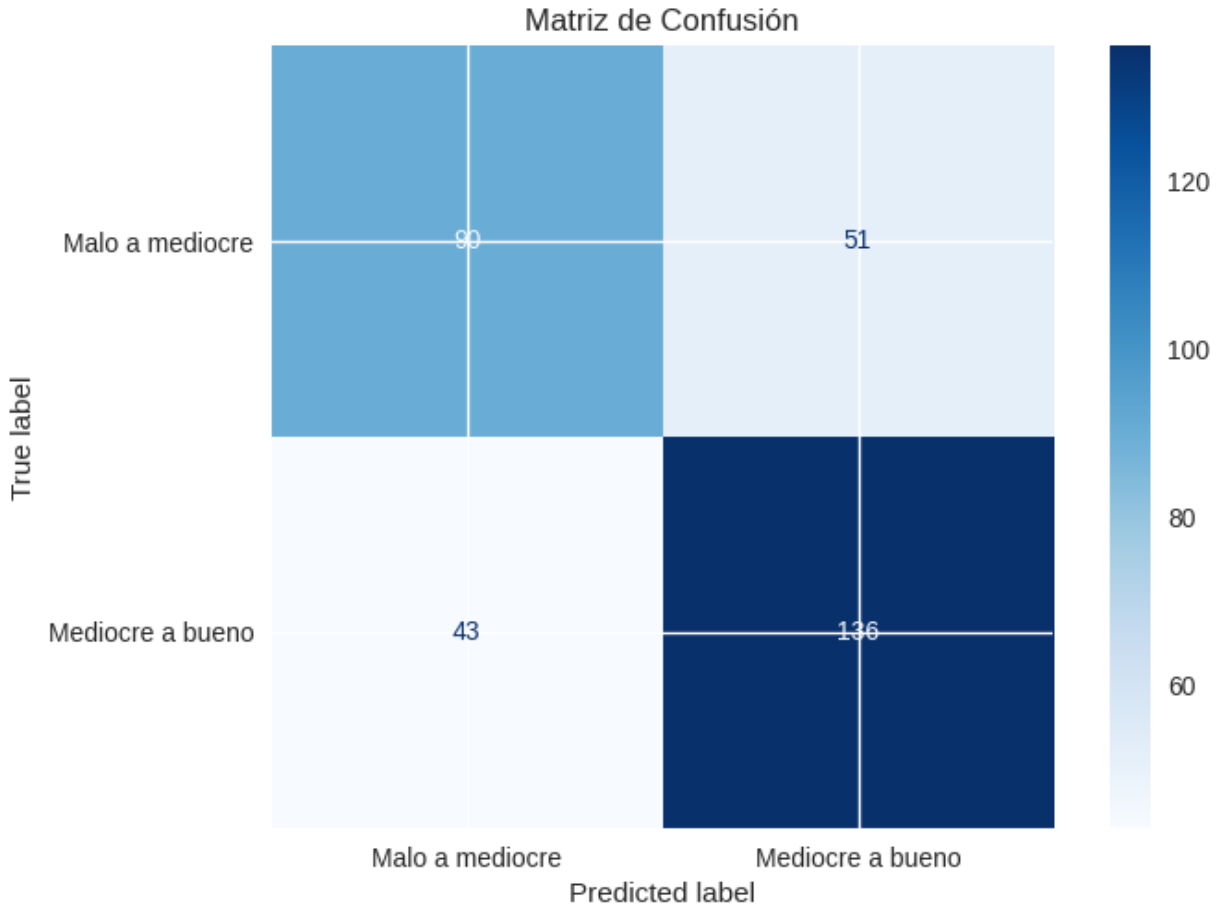
In [ ]:

```
# Hacer predicciones en el conjunto de prueba
y_pred = knn.predict(X_test)
```

In [ ]:

```
# Calcular la matriz de confusión
```

```
conf_matrix = confusion_matrix(y_test, y_pred)

# Crear y mostrar la matriz de confusión utilizando
disp = ConfusionMatrixDisplay(conf_matrix, display_labels=['Malo a mediocre', 'Mediocre
a bueno'])
disp.plot(cmap=plt.cm.Blues)
plt.title('Matriz de Confusión')
plt.show()
```



Conclusiones de la matriz

El modelo predijo bien 132 casos en que eran mediocre bueno y en 90 casos que eran Malo a mediocre.

In [ ]:

```
# Evaluar el rendimiento del modelo
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.68      0.64      0.66       141
           1       0.73      0.76      0.74       179

    accuracy                           0.71       320
   macro avg       0.70      0.70      0.70       320
weighted avg       0.70      0.71      0.71       320
```

Arbol de decision

In [ ]:

```
# Eliminar la columna 'quality' de los datos
datos = Calidadvino.drop(columns=['quality'])
```

In [ ]:

```
# Estandarizar los datos (si es necesario)
```

```
scaler = StandardScaler()
datos_escalados = scaler.fit_transform(datos)
```

In [ ]:

```
# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(datos_escalados, Etiquetascalidad, t
est_size=0.2, random_state=42)
```

In [ ]:

```
# Crear el modelo de Árbol de Decisión
tree_model = DecisionTreeClassifier()
```

In [ ]:

```
# Entrenar el modelo
tree_model.fit(X_train, y_train)
```

Out[ ]:

▾ DecisionTreeClassifier

DecisionTreeClassifier()

In [ ]:

```
# Hacer predicciones en el conjunto de prueba
y_pred_tree = tree_model.predict(X_test)
```

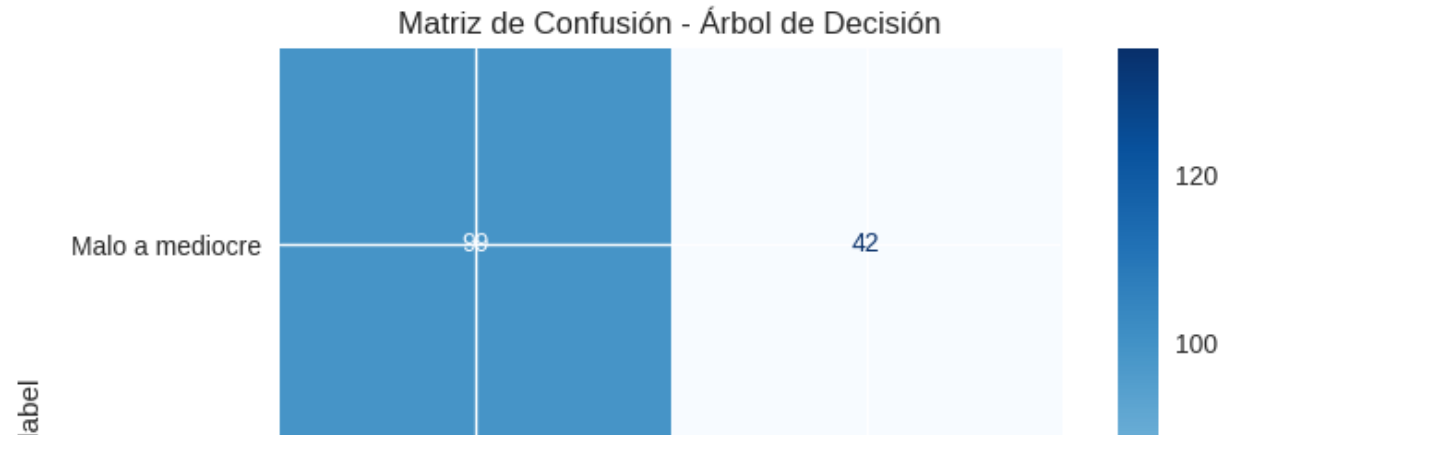
In [ ]:

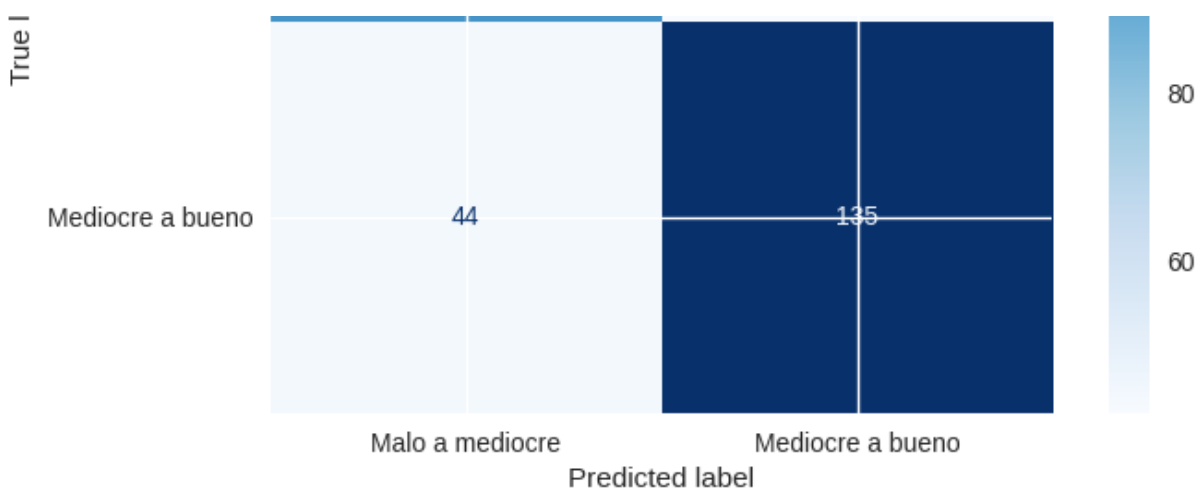
```
# Evaluar el modelo
print(classification_report(y_test, y_pred_tree))
```

```
              precision    recall  f1-score   support

           0       0.69      0.70      0.70       141
           1       0.76      0.75      0.76       179

    accuracy                           0.73       320
   macro avg       0.73      0.73      0.73       320
weighted avg       0.73      0.73      0.73       320
```

In [ ]:

```
# Calcular la matriz de confusión para los resultados del árbol de decisión
conf_matrix_tree = confusion_matrix(y_test, y_pred_tree)

# Crear y mostrar la matriz de confusión utilizando
disp_tree = ConfusionMatrixDisplay(conf_matrix_tree, display_labels=['Malo a mediocre',
'Mediocre a bueno'])
disp_tree.plot(cmap=plt.cm.Blues)
plt.title('Matriz de Confusión - Árbol de Decisión')
plt.show()
```




True l
Mediocre a bueno
44
135
80
60
Malo a mediocre
Mediocre a bueno
Predicted label