Clasificación de vinos



Adrián Yared Armas de la Nuez

**Contenido**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

[**1. Objetivo 2**](#_2c35568x6wg8)

[**2.1 Librerías 2**](#_t8cwbo6n4466)

[**2.1.1 Código 2**](#_aig92oe7k2m0)

[**3 Vino tinto 3**](#_o6k30b2mhp4j)

[**3.1 Carga del dataset 3**](#_w847desd5rel)

[**3.1.1 Código 3**](#_9mprq48csecu)

[**3.1.2 Separado de características 3**](#_27g4tag5tno)

[**3.2 Matriz de correlación 3**](#_cbbjv1azzpnt)

[**3.2.1 Código 3**](#_6ys1ts9gjfob)

[**3.2.2 Resultado 4**](#_po93tupql549)

[**3.3 Selección de características (SelectKBest) 4**](#_v5eju7pjd8c6)

[**3.3.1 Código 4**](#_dpkf335iwqps)

[**3.3.2 Resultado 4**](#_pmb3iw4predc)

[**3.4 comparativa de la precisión KNN y Naive Bayes 5**](#_qfh1qqp3fz1b)

[**3.4.1 Código 5**](#_jk8rtubam59u)

[**3.4.2 Naive Bayes 5**](#_lrjwh1cu04p3)

[**3.4.3 Knn 5**](#_vno7et1toq1s)

[**3.4.4 Selección del mejor modelo 5**](#_kboumpuout9d)

[**3.5 Tras elegir el mejor modelo 5**](#_hay85s1r7o1y)

[**3.5.1 Entrenamiento y matriz de confusión 5**](#_fepkgvms3hlo)

[**3.5.2 Exportar el modelo 6**](#_ng4whlptn776)

[**3.5.3 Importar el modelo 6**](#_urilrf13wyda)

[**3.5.4 Predicción y matriz de confusión 6**](#_wng5lgc52mt5)

[**3.6 Comparar el resultado 7**](#_4fd9xmw6hnz)

[**4 Vino blanco 7**](#_1zs0p34ko43u)

[**4.1 Carga del dataset 7**](#_lz3gzmf2ltwk)

[**4.1.1 Separado de características 7**](#_3whfa2ux7yad)

[**4.2 Matriz de correlación 8**](#_anrhqbvqd7x)

[**4.3 Selección de características 8**](#_8az2l8s3jbtt)

[**4.4 Comparación de precisión 8**](#_x3mc6ebj4sux)

[**4.4.1 Naive bayes 9**](#_e2fd1ri6x1zr)

[**4.4.2 Knn 9**](#_s72u97fa1jad)

[**4.4.3 Comparación de modelos 9**](#_if667hf9f3ie)

[**4.5 Entrenamiento y matriz de confusión 9**](#_bjyz9w679bci)

[**4.5.2 Exportar el modelo 10**](#_cfwgq7fsy7am)

[**4.5.3 Importar el modelo 10**](#_gww45nf0hmdt)

[**4.5.4 Aplicar la predicción y la matriz de confusión 10**](#_xlvkr2swwekw)

[**4.6 Comparación del resultado 11**](#_gmmhlk9pknyw)

[**5. Github y Colab 11**](#_lmdngf8dlin3)

## 

## 

## **1. Objetivo**

El objeto de esta actividad es poner en práctica los conocimientos adquiridos hasta el

momento para ellos vamos a utilizar el siguiente dataset que contiene una serie de

características físico-químicas que determina la calidad del vino en una escala de valores del 1 al 10.

El enlace donde se encuentran los dataset es el siguiente:

<https://archive.ics.uci.edu/ml/datasets/Wine+Quality>

Como proyecto de partida se puede utilizar el ejemplo:

Título: Ejemplo\_3\_3\_Clasificación\_con\_Naive\_Bayes\_(Heart\_Diseases)

Url: <https://colab.research.google.com/drive/1hwri6X-N_cHmpZs31-zyK2XwRyfA4EGN?usp=sharing>

### **2.1 Librerías**

#### **2.1.1 Código**

import pandas as pd # Data handling

import numpy as np # Numerical calculations

import matplotlib.pyplot as plt # Data visualization

import seaborn as sns # Statistical graphics

from sklearn.model\_selection import train\_test\_split, cross\_val\_score # Data splitting and validation

from sklearn.naive\_bayes import GaussianNB # Naïve Bayes model

from sklearn.neighbors import KNeighborsClassifier # KNN model

from sklearn.metrics import confusion\_matrix, accuracy\_score # Evaluation metrics

from sklearn.feature\_selection import SelectKBest, f\_classif # Feature selection

import joblib # Save and load models

import seaborn as sns # Confusion matrix visualization

from sklearn.preprocessing import StandardScaler # Data scaling

## **3 Vino tinto**

### **3.1 Carga del dataset**

#### **3.1.1 Código**

# Load datasets

red\_wine = pd.read\_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv', sep=';') # Red Wine

#### **3.1.2 Separado de características**

X\_red = red\_wine.drop(columns=['quality'])

y\_red = red\_wine['quality']

### **3.2 Matriz de correlación**

#### **3.2.1 Código**

# Display the correlation matrix

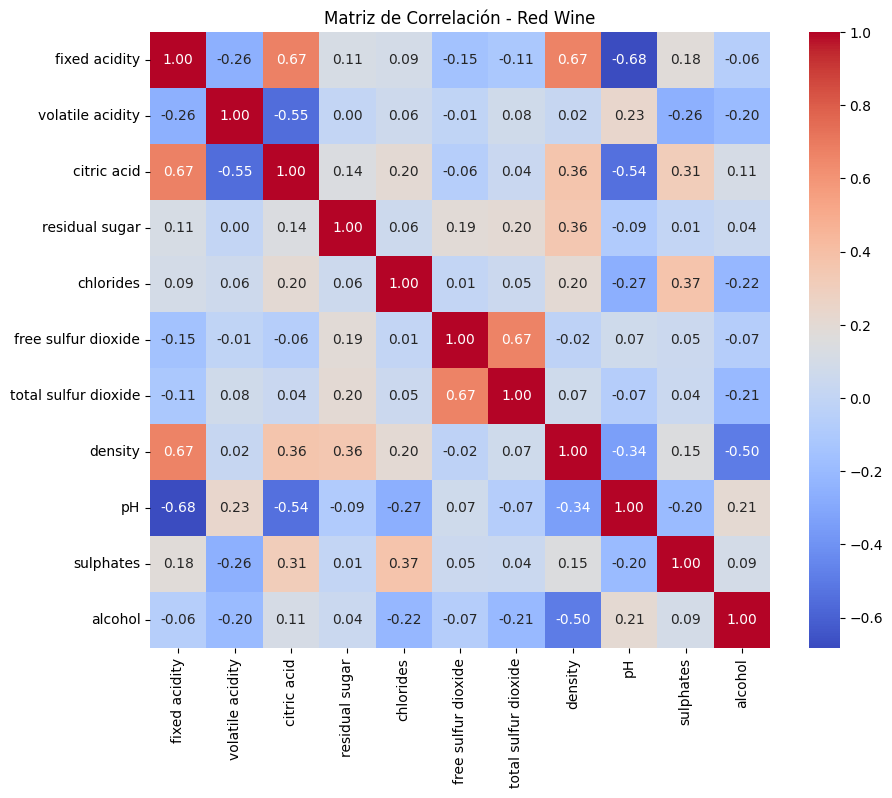
plt.figure(figsize=(10, 8))

sns.heatmap(X\_red.corr(), annot=True, cmap='coolwarm', fmt='.2f')

plt.title("Matriz de Correlación - Red Wine")

plt.show()

#### **3.2.2 Resultado**



### **3.3 Selección de características (SelectKBest)**

#### **3.3.1 Código**

# Feature selection using SelectKBest

k = 8 # Select the top 8 features

selector = SelectKBest(score\_func=f\_classif, k=k)

X\_red\_selected = selector.fit\_transform(X\_red, y\_red)

selected\_features = X\_red.columns[selector.get\_support()]

print(f"Selected features: {selected\_features.tolist()}")

#### **3.3.2 Resultado**



### **3.4 comparativa de la precisión KNN y Naive Bayes**

#### **3.4.1 Código**

# Comparison of Naïve Bayes and KNN using cross-validation

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_red\_selected, y\_red, test\_size=0.2, random\_state=42)

# Data scale for a better presition

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

#### **3.4.2 Naive Bayes**

nb = GaussianNB()

nb\_scores = cross\_val\_score(nb, X\_train, y\_train, cv=5, scoring='accuracy')

print(f"Naïve Bayes - Average Accuracy: {nb\_scores.mean():.4f}")



#### **3.4.3 Knn**

knn = KNeighborsClassifier(n\_neighbors=5)

knn\_scores = cross\_val\_score(knn, X\_train, y\_train, cv=5, scoring='accuracy')

print(f"KNN (k=5) - Average Accuracy: {knn\_scores.mean():.4f}")



#### **3.4.4 Selección del mejor modelo**

best\_model = nb if nb\_scores.mean() > knn\_scores.mean() else knn

best\_model.fit(X\_train, y\_train)

### **3.5 Tras elegir el mejor modelo**

#### **3.5.1 Entrenamiento y matriz de confusión**

# Train the best model and obtain the confusion matrix

y\_pred = best\_model.predict(X\_test)

cm = confusion\_matrix(y\_test, y\_pred)

plt.figure(figsize=(6,5))

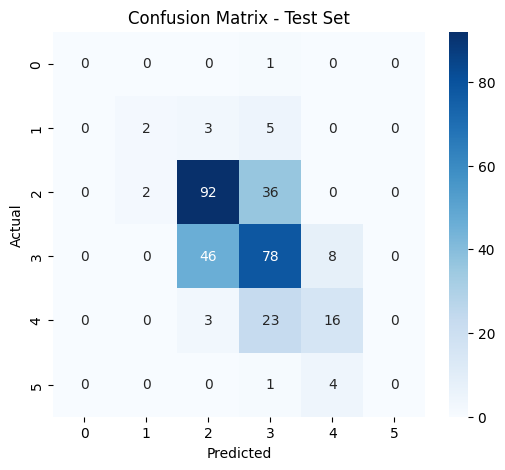
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.title("Confusion Matrix - Test Set")

plt.show()



#### **3.5.2 Exportar el modelo**

# Export the model

joblib.dump(best\_model, 'best\_wine\_model.pkl')



#### **3.5.3 Importar el modelo**

# Import the model

loaded\_model = joblib.load('best\_wine\_model.pkl')

#### **3.5.4 Predicción y matriz de confusión**

# Apply the model to the entire dataset and obtain the confusion matrix

y\_all\_pred = loaded\_model.predict(scaler.transform(X\_red\_selected))

overall\_cm = confusion\_matrix(y\_red, y\_all\_pred)

plt.figure(figsize=(6,5))

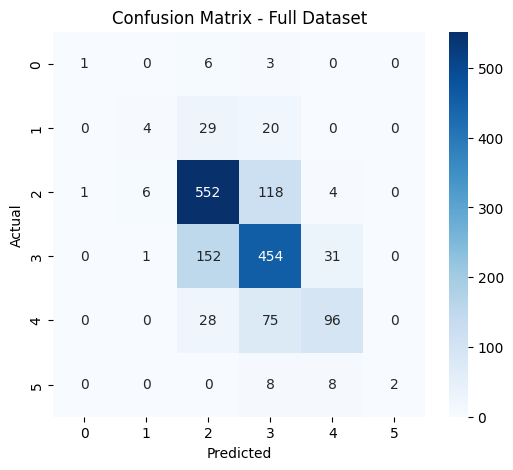
sns.heatmap(overall\_cm, annot=True, fmt='d', cmap='Blues')

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.title("Confusion Matrix - Full Dataset")

plt.show()



### **3.6 Comparar el resultado**

# Compare accuracy

accuracy = accuracy\_score(y\_red, y\_all\_pred)

print(f"Final Accuracy: {accuracy:.4f}")



## **4 Vino blanco**

### **4.1 Carga del dataset**

# Fetch dataset

white\_wine = pd.read\_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv', sep=';')

#### **4.1.1 Separado de características**

# data (as pandas dataframes) (white wine)

X\_white = white\_wine.drop(columns=['quality'])

y\_white = white\_wine['quality']

### **4.2 Matriz de correlación**

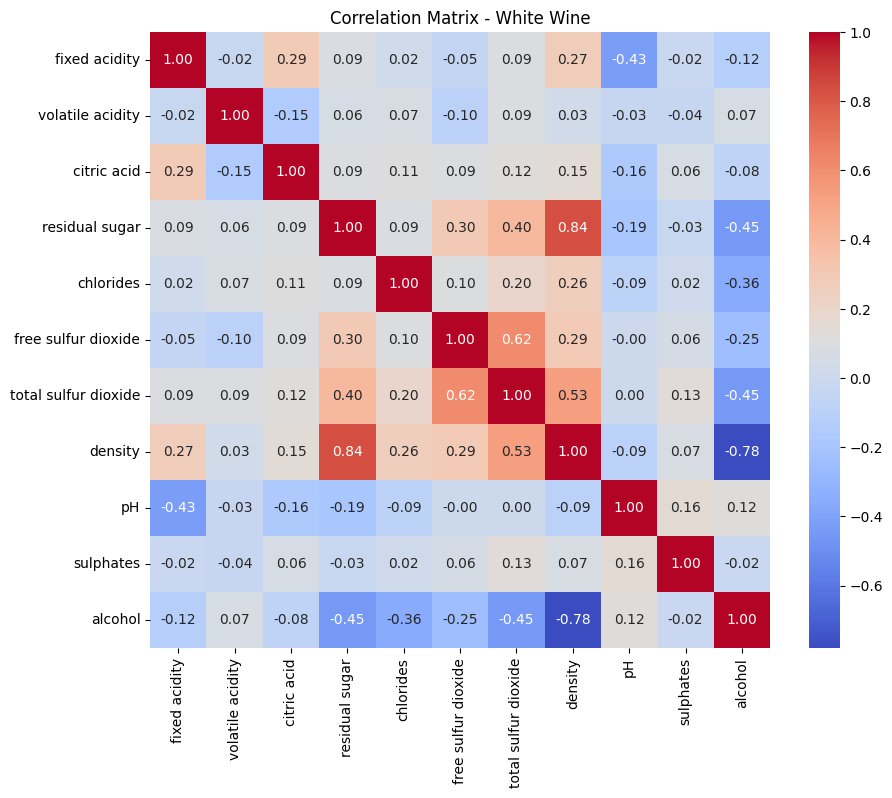
# Display the correlation matrix

plt.figure(figsize=(10, 8))

sns.heatmap(X\_white.corr(), annot=True, cmap='coolwarm', fmt='.2f')

plt.title("Correlation Matrix - White Wine")

plt.show()



### **4.3 Selección de características**

# Feature selection using SelectKBest

k = 8 # Select the top 8 features

selector = SelectKBest(score\_func=f\_classif, k=k)

X\_white\_selected = selector.fit\_transform(X\_white, y\_white)

selected\_features = X\_white.columns[selector.get\_support()]

print(f"Selected features: {selected\_features.tolist()}")

### **4.4 Comparación de precisión**

# Comparison of Naïve Bayes and KNN using cross-validation

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_white\_selected, y\_white, test\_size=0.2, random\_state=42)

# Data scale for a better presition

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

#### **4.4.1 Naive bayes**

nb = GaussianNB()

nb\_scores = cross\_val\_score(nb, X\_train, y\_train, cv=5, scoring='accuracy')

print(f"Naïve Bayes - Average Accuracy: {nb\_scores.mean():.4f}")



#### **4.4.2 Knn**

knn = KNeighborsClassifier(n\_neighbors=5)

knn\_scores = cross\_val\_score(knn, X\_train, y\_train, cv=5, scoring='accuracy')

print(f"KNN (k=5) - Average Accuracy: {knn\_scores.mean():.4f}")



#### **4.4.3 Comparación de modelos**

best\_model = nb if nb\_scores.mean() > knn\_scores.mean() else knn

best\_model.fit(X\_train, y\_train)



### **4.5 Entrenamiento y matriz de confusión**

# Train the best model and obtain the confusion matrix

y\_pred = best\_model.predict(X\_test)

cm = confusion\_matrix(y\_test, y\_pred)

plt.figure(figsize=(6,5))

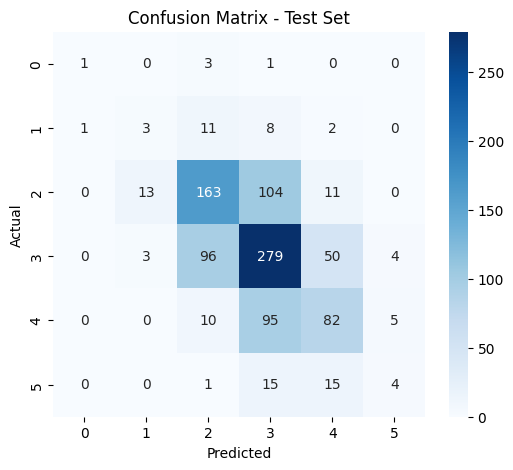
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.title("Confusion Matrix - Test Set")

plt.show()



#### **4.5.2 Exportar el modelo**

# Export the model

joblib.dump(best\_model, 'best\_wine\_model\_white.pkl')



#### **4.5.3 Importar el modelo**

# Import the model

loaded\_model = joblib.load('best\_wine\_model\_white.pkl')

#### **4.5.4 Aplicar la predicción y la matriz de confusión**

# Apply the model to the entire dataset and obtain the confusion matrix

y\_all\_pred = loaded\_model.predict(scaler.transform(X\_white\_selected))

overall\_cm = confusion\_matrix(y\_white, y\_all\_pred)

plt.figure(figsize=(6,5))

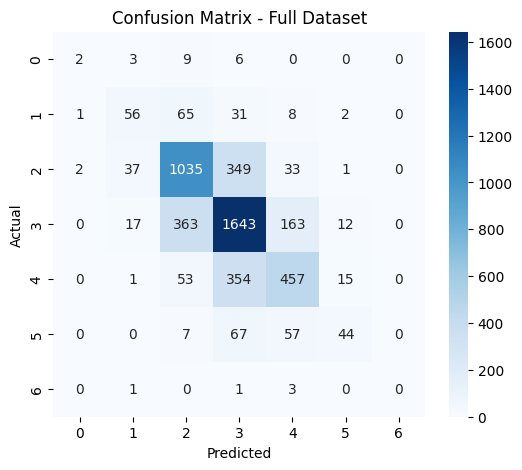
sns.heatmap(overall\_cm, annot=True, fmt='d', cmap='Blues')

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.title("Confusion Matrix - Full Dataset")

plt.show()



## 

### **4.6 Comparación del resultado**

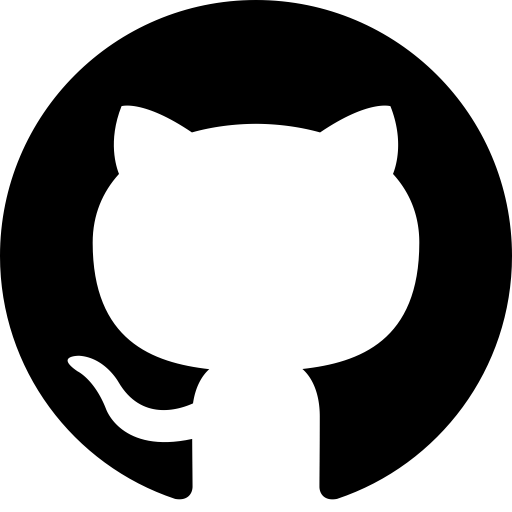
# Compare accuracy

accuracy = accuracy\_score(y\_white, y\_all\_pred)

print(f"Final Accuracy: {accuracy:.4f}")



## **5. Github y Colab**

[](https://github.com/AdrianYArmas/IaBigData/tree/main/SNS/3%20%20-%20Algoritmos%20y%20herramientas%20para%20el%20aprendizaje%20supervisado%20/3.5%20Clasificaci%C3%B3n%20de%20vinos) [](https://colab.research.google.com/drive/1Bnn3C-Av2Ock5uS__BOkJbfKgw3aYca_?usp=sharing)