Tarea Nº 2:

Clasificación de Calidad de Vino

Luis Miranda De la Guarda
Ingeniería Civil Computación, Escuela de Ingeniería, Universidad de O'Higgins
26 de 04 del 2023, Rancagua, Chile
Email: luis.miranda@pregrado.uoh.cl

***Abstract*—Este informe tiene como objetivo analizar, visualizar y clasificar datos de la calidad de vinos, utilizando el conjunto de datos Wine Quality Data Set del UC Irvine Machine Learning Repository, usando las librerías pandas, numpy, matplotlib y seaborn de Python. En la primera parte, se realiza la visualización y análisis de datos, dividiendo los datos en dos subconjuntos de calidad alta y baja, generando histogramas y una matriz de correlación. En la segunda parte, se lleva a cabo la clasificación de los datos utilizando el modelo de clasificación LogisticRegression() y se modifican las funciones de pérdida para observar cómo afectan al rendimiento del modelo. Se encontró que utilizando Logistic Regression con los parámetros por defecto, se logró el mejor accuracy correspondiente a un 73,15% de precisión en el conjunto de validación. Este resultado sugiere que el modelo podría ser útil en la predicción de la variable objetivo en futuros conjuntos de datos, pero con resultados no tan precisos.**

I. Introducción

La visualización y análisis de datos son procesos críticos en el análisis de datos y la toma de decisiones informadas. Las librerias pandas, junto con numpy, matplotlib y seaborn, son herramientas esenciales para realizar estas tareas. En este informe se utilizará el conjunto de datos Wine Quality Data Set del UC Irvine Machine Learning Repository para hacer análisis exploratorio de datos. Además, se realizará la clasificación de los datos utilizando el modelo de LogisticRegression() y se modificarán las funciones de pérdida para observar su efecto en el rendimiento del modelo.

El objetivo es comparar los diferentes rendimientos de los modelos, según la configuración de los parámetros de este. Se analizarán 2 métricas para evaluar el rendimiento de los algoritmos: Accuracy y matriz de confusión. Además, se deben reportar para cada uno de los modelos, el coeficiente de intercepción y los coeficientes de las variables/caracteristicas.

II. Marco Teórico

A. Base de datos; Wine Quality:

La base de datos Wine Quality es un conjunto de datos que contiene información sobre características físico-químicas de diferentes muestras de vino procedente de Portugal, Los datos se encuentran etiquetas con una variable de calidad que va de 0 a 10. Este conjunto fue recolectado desde el 2004 hasta el 2007 usando solo muestras de denominación de origen que fueron analizadas en el organismo de certificación oficial (CVRVV). El set se encuentra disponible en el repositorio de UCI Machine Learning. Las características presentes en el data set son las siguientes:

- fixed acidity.
- volatile acidity.
- citric acid.
- residual sugar.
- chlorides.
- free sulfur dioxide.
- total sulfur dioxide.
- density.
- pH.
- sulphates.
- alcohol.
- quality.

Siendo la columna quality, la que se usará como columna objetivo.

B. Librerías de Python:

Se utilizarán diversas librerías de Python para realizar las tareas de análisis exploratorio de datos y los modelos de clasificación y desempeño. A continuación, se describen brevemente cada una de ellas:

- Pandas: Esta librería proporciona estructuras de datos flexibles y de alto rendimiento para el análisis de datos. En particular, se utilizará para leer y manipular los datos.
- NumPy: Es una librería para realizar cálculos numéricos con alta eficiencia. En esta tarea, se utilizará principalmente para realizar operaciones matemáticas.
- Matplotlib: Es una librería de gráficos en 2D, que produce figuras de calidad de publicación en una variedad de formatos impresos y entornos interactivos en todas las plataformas. Se utilizará para visualizar los datos mediante la creación de diferentes tipos de gráficos.
- Seaborn: Es una librería de visualización de datos basada en matplotlib. Proporciona una interfaz de alto nivel para dibujar gráficos estadísticos, atractivos e informativos. Se utilizará para crear gráficos más avanzados, como mapas de calor y gráficos de dispersión.
- Scikit-learn: Es una librería de aprendizaje automático para clasificación, regresión y agrupamiento de datos. Se utilizará en esta tarea para entrenar y evaluar diferentes modelos de aprendizaje automático en los datos de la base de datos.

C. Análisis Exploratorio de Datos:

Es un enfoque para analizar y resumir las características principales de un conjunto de datos. El objetivo del EDA

es descubrir patrones, identificar valores atípicos y establecer relaciones entre las variables.

En lugar de seguir un proceso estructurado y rígido para analizar los datos, el EDA es más flexible y se basa en la intuición del analista de datos para explorar y visualizar los datos de diversas formas. Algunas técnicas de EDA comunes incluyen la elaboración de gráficos de dispersión, histogramas, diagramas de caja y bigotes, y la realización de pruebas de normalidad y correlación.

1) Histogramas: Los histogramas son herramientas gráficas que permiten visualizar la distribución de frecuencias de una variable continua en un conjunto de datos. En un histograma, la variable se divide en intervalos o clases, y se cuenta la frecuencia con que se observa cada valor dentro de cada clase. Luego, se grafican las frecuencias como barras que se levantan desde la base del intervalo correspondiente. El ancho de las barras es igual al ancho del intervalo y la altura de las barras representa la frecuencia observada.

2) Gráficos de Dispersión: Los gráficos de dispersión, por otro lado, son utilizados para analizar la relación entre dos variables continuas. En un gráfico de dispersión, cada punto representa una observación de ambas variables. El eje horizontal representa una variable y el eje vertical representa la otra variable. Si los puntos se agrupan en una forma lineal, esto indica una fuerte relación entre las dos variables.

3) Matriz de Correlación: Una matriz de correlación es una tabla que muestra la relación entre varias variables. Esta tabla se compone de una serie de coeficientes de correlación que miden la relación entre dos variables. Los coeficientes de correlación pueden variar entre -1 y 1, donde -1 indica una correlación negativa perfecta, 0 indica la falta de correlación y 1 indica una correlación positiva perfecta.

En una matriz de correlación, las variables se colocan en las filas y columnas de la tabla. La diagonal de la tabla siempre está llena de 1, ya que una variable siempre tiene una correlación perfecta consigo misma. Las celdas fuera de la diagonal contienen los coeficientes de correlación para la combinación de dos variables correspondientes.

La matriz de correlación es una herramienta útil para explorar las relaciones entre variables y para identificar patrones y tendencias en los datos. También se utiliza para reducir la dimensionalidad de los datos, ya que puede mostrar qué variables están altamente correlacionadas y, por lo tanto, pueden ser redundantes en un modelo.

D. Regresión Logística:

La regresión logística es una técnica importante en el campo de la inteligencia artificial y el machine learning (AI/ML). Los modelos de ML son programas de software que se pueden entrenar para realizar tareas complejas de procesamiento de datos sin intervención humana. Esta técnica se basa en un modelo estadístico que utiliza la función logística, o función logit, en matemáticas como la ecuación entre x e y. La función logit mapea y como una función sigmoidea de x con la siguiente ecuación:

$$f(x) = \frac{1}{1 + e^{-x}}$$

Al representar esta ecuación de regresión logística, se obtendrá una curva en S como la que se muestra a continuación.

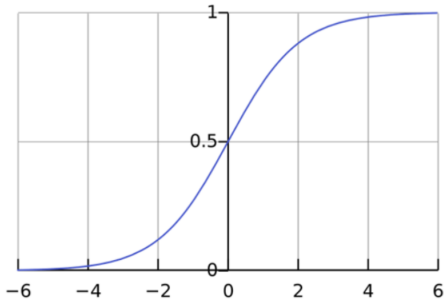

Fig. 1: Función Sigmoidea

La función devuelve solo valores entre 0 y 1 para la variable dependiente, al margen de los valores de la variable independiente. Así es como la regresión logística estima el valor de la variable objetivo. Los métodos de regresión logística también modelan ecuaciones entre múltiples variables independientes y una variable dependiente, como es el caso de este informe.

La regresión logística se implementará con **sklearn.linear_model.LogisticRegression**. A pesar de su nombre, se implementa como un modelo lineal para clasificación en lugar de regresión en términos de la nomenclatura scikit-learn/ML.

Esta implementación puede adaptarse a la regresión logística binaria, uno contra el resto, multinomial con $l1, l2$ opcionales, o regularización de Elastic-Net.

1) Caso de regresión logística entre 2 clases: Para facilitar la notación, se asume que el vector objetivo y_i puede tomar valores en el conjunto $\{0,1\}$ para el punto i. Una vez ajustado el modelo, el método predict_proba predice la probabilidad de la clase positiva $P(y_i = 1|X_i)$ como:

$$\hat{p}(X_i) = expit(X_i w + w_0) = \frac{1}{1 + e^{-X_i w - w_0}}$$

Como problema de optimización, la regresión logística de clase binaria con término de regularización $r(w)$ minimiza la siguiente función de costo:

$$min_w C \sum_{i=1}^{n} (-y_i log(\hat{p}(X_i)) - (1 - y_i) log(1 - \hat{p}(X_i))) + r(w)$$

2) Penalty: Sklearn provee actualmente 4 opciones para el término de regularización $r(w)$, con el argumento **penalty**.

Penalty	r(w)
None	0
$l1$	$\|w\|_1$
$l2$	$\frac{1}{2}\|w\|_2^2 = \frac{1}{2} w^T w$
ElasticNet	$\frac{1-\rho}{2} w^T w + \rho \|w\|_1$

TABLE I: Opciones parámetro Penalty.

Para ElasticNet, ρ (que corresponde al parámetro l_1_ratio) controla la fuerza de la regularización $l1$ vs. la regularización $l2$. Elastic-Net es equivalente a $l1$ cuando $\rho = 1$ y equivalente a $l2$ cuando $\rho = 0$.

3) Solvers: Los algoritmos solucionadores del problema de optimización presentado, implementados en la clase LogisticRegression son “lbfgs”, “liblinear”, “newton-cg”, “newton-cholesky”, “sag” y “saga”. A continuación se presenta una tabla donde se puede observar el detalle de los tipos de penalty que acepta cada uno.

Solver	Penalty aceptado
’lbfgs’	[’l2’, None]
’liblinear’	[’l1’,’l2’]
’newton-cg’	[’l2’, None]
’newton-cholesky’	[’l2’, None]
’sag’	[’l2’, None]
’saga’	[’elasticnet’, ’l1’, ’l2’, None]

TABLE II: Solvers y sus penalty aceptados.

III. Metodología

A. Primera Parte; Visualización y análisis de datos:

A continuación se describe la metodología a seguir para realizar la primera parte de la tarea:

1) **Carga y Lectura de los datos:** Se conectó el notebook de colab de la tarea con Google Drive, donde está alojado el set de datos, luego se lee usando la función **pd.read_csv()** con la ruta especificada.
2) **Etiquetado de los Datos según Calidad:** Se asignaron etiquetas categóricas según la nota de calidad de cada vino, con la siguiente regla:
   - high: Todos los vinos con nota de calidad mayor o igual a 6.
   - low: Todos los vinos con nota de calidad menor a 6.

```
# Crear una nueva columna llamada 'class' con valores 'high' o 'low' dependiendo de la calidad
data_all['class'] = data_all.apply(lambda row: 'high' if row['quality'] >= 6 else 'low', axis=1)

# Eliminar la columna 'quality'
data_all.drop('quality', axis=1, inplace=True)
data_all

```

3) **Agrupación de datos según class:** Luego de etiquetar las instancias según su nota, se ha dividido el data set en 2 subconjuntos, cada uno con los datos de vinos high y low respectivamente.

```
# Separar los datos en dos conjuntos diferentes
high_data = data_all.groupby('class').get_group('high')
low_data = data_all.groupby('class').get_group('low')

```

4) **Gráfico de Histogramas:** Con los datos divididos en dos subconjuntos, se procedió a graficar un histograma para cada característica de los datos, superponiendo los gráficos de ambos subconjuntos en uno solo.

```
fig, axs = plt.subplots(4, 3, figsize=(12, 12))
axs = axs.flatten()

for i, col in enumerate(high_data.columns):
    axs[i].hist(low_data[col], bins=20, alpha=0.8, color='red', label='Low Quality')
    axs[i].hist(high_data[col], bins=20, alpha=0.8, color='blue', label='High Quality')
    axs[i].set_title(col)
    axs[i].set_xlabel(col)
    axs[i].set_ylabel('Frecuencia')
    axs[i].legend()

plt.tight_layout()
plt.show()

```

5) **Matriz y Gráfico de Correlación** En este paso, se obtuvo la matriz de correlación de los datos usando la función corr() de Pandas, además se usó la función heatmap() de Seaborn para graficar, es importante notar que la columna categórica se etiquetó numéricamente con la siguiente regla y que se le aplicó la función de valor absoluto a la matriz completa:
   - 1: Para todos los datos clasificados como high.
   - 0: Para todos los datos clasificados como low.

```
# Codificar clase
data_all['class'] = data_all['class'].apply(lambda q: 1 if q == 'high' else 0)

# Calcular matriz de correlacion
corr_matrix = data_all.corr().abs()

# Graficar matriz de correlacion
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='YlGnBu')
plt.show()

```

6) **Características más relacionadas con la Clase**: En esta sección se analizó la matriz de correlación y se buscaron los atributos más relacionados con la columna objetivo.

```
# Ordenando las correlaciones con la clase y mostrando los resultados.
data_sorted = data_all.sort_values(by='class')
corr_matrix = data_sorted.corrwith(data_sorted['class']).abs().sort_values(ascending=False)

print("Caracteristicas mas correlacionadas con class")
print(corr_matrix[1:])

```

7) **5 Pares de Características más relacionadas entre sí:** Luego se ha procedido a analizar las 5 características más correlacionadas entre sí, buscando los 5 valores más altos en la parte superior de la matriz de correlación absoluta.

```
# Nombres de las caracteristicas
feature_names = data_all.columns.to_list()

# Calculamos la matriz de correlacion de los datos
correlation_matrix = np.array(data_all.corr().abs().T)

# Ignoramos la varianza de cada caracteristica en si misma
```

```
np.fill_diagonal(correlation_matrix, 0)

# Obtenemos los nombres de los 5 pares de caracteristicas mas relacionadas entre si
pairs = []
for i in range(5):
    max_value = np.max(np.triu(correlation_matrix))
    row, col = np.where(correlation_matrix == max_value)
    feature_pair = (feature_names[row[0]], feature_names[col[0]])
    pairs.append((feature_pair, max_value))
    correlation_matrix[row[0], col[0]] = 0
    correlation_matrix[col[0], row[0]] = 0

# Ordenamos los pares de caracteristicas por su valor de covarianza (de mayor a menor)
pairs.sort(key=lambda x: x[1], reverse=True)

# Imprimimos los pares de caracteristicas mas relacionadas
for i in range(len(pairs)):
    feature_pair = pairs[i][0]
    covariance_value = pairs[i][1]
    print("Pareja de caracteristicas {}: {} y {}: {}".format(i+1, feature_pair[0], feature_pair[1], covariance_value))

```

8) **Graficar las 2 características más relacionadas entre sí:** Una vez obtenido los 5 pares de atributos más correlacionados entre sí, se procedió a graficar los 2 atributos con el coeficiente más alto. Al gráfico se le ha agregado una línea que muestra la relación lineal de estros atributos, además de los histogramas de cada uno, usando la función joinplot() de Seaborn.

```
# Crear el grafico de puntos con histogramas en los ejes x e y y la linea de regresion
sns.jointplot(data=data_all, x='free sulfur dioxide', y='total sulfur dioxide', kind='reg', color='purple',
              line_kws={'linewidth': 2, 'color': 'blue'}, scatter_kws={'s':30, 'color': 'purple', 'alpha': 0.5})

# Mostrar el grafico
plt.show()

```

9) **5 Pares de características menos relacionadas entre sí y gráfico del par menos relacionado:** Se ha hecho el mismo ejercicio anterior, pero para los 5 pares de características menos relacionadas, se ha seguido la misma lógica que para el punto anterior, pero usando funciones de mínimo en vez de máximo, por lo que el detalle en código se omite.

B. Segunda Parte; Clasificación:

En esta sección, se describe la metodología para generar los experimentos requeridos de clasificación usando **sklearn.linear_model.LogisticRegression** donde se crearon 5 modelos diferentes, cambiando los siguientes parámetros:

- **Penalty.**
- **Solver.**
- **l1_ratio.**

Se realizará una prueba de proporción de cada uno de los conjuntos de datos para conocer si las clases mantienen la proporcionan normal, en el conjunto de entrenamiento y validación, esto calculando el promedio de la clase positiva en el total de los conjuntos.

A continuación se describe en detalle, la construcción de cada uno de los experimentos. Se implementó una función esencial para llevar a cabo el trabajo de entrenamiento y testing de cada uno de los modelos, además se implementó una función que permite graficar las matrices de confusión.

1) **plot_confusion_matrix():**

```
def plot_confusion_matrix(cm, classes,
    normalize=False, title='Confusion matrix',
    cmap=plt.cm.Blues):
    """
    Esta funcion grafica la matriz de confusion.
    Se puede aplicar normalizacion cambiando el parametro a normalize =True `.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in np.ndindex(cm.shape):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh
                                 else "black")
    plt.tight_layout()
    plt.ylabel('Etiquetas Verdaderas')
    plt.xlabel('Etiquetas Predichas')
    plt.show()

```

2) **entrenar_modelo():**

```
def entrenar_modelo(modelo_log, X_train, X_val, y_train, y_val):
    """
    Esta funcion entrena y prueba el desempe o de un modelo de aprendizaje.
    """
    # Entrenamos el modelo con los datos de entrenamiento
    modelo_log.fit(X_train, y_train)

    # Hacemos predicciones con los conjuntos de entrenamiento y validacion
    pred_train = modelo_log.predict(X_train)
    pred_val = modelo_log.predict(X_val)

    # Evaluamos la precisi n del modelo en los conjuntos de entrenamiento y validacion
    acc_train = accuracy_score(y_train, pred_train)
    acc_val = accuracy_score(y_val, pred_val)
    print('-'*70)
    print(f'Accuracy en el conjunto de entrenamiento:\t{acc_train:.2%}')
    print(f'Accuracy en el conjunto de validacion:  \t{acc_val:.2%}')
```

```
    print('-'*70)

    # Obtenemos el parametro de intercepcion y los coeficientes de las variables/ caracteristicas
    intercep = modelo_log.intercept_
    coef = modelo_log.coef_

    # Creamos una tabla con los coeficientes de cada caracteristica
    coef_df = pd.DataFrame({'feature': X_train.columns, 'coef': coef[0]})

    # Mostramos la tabla de coeficientes
    print(f'Parametro de intercepcion: {intercep}\n')
    print('Coeficientes de las variables/ caracteristicas:')
    print(coef_df)
    print('-'*70)

    # Obtenemos la matriz de confusion de los conjuntos de entrenamiento y validacion
    cm_train = confusion_matrix(y_train, pred_train)
    cm_val = confusion_matrix(y_val, pred_val)

    print('Matriz de confusion del conjunto de entrenamiento:')
    print(cm_train)
    # Graficamos la matriz de confusion del conjunto de entrenamiento
    plot_confusion_matrix(cm_train, classes=['clase 0', 'clase 1'])
    print('-'*70)

    print('Matriz de confusion del conjunto de validacion:')
    print(cm_val)
    # Graficamos la matriz de confusion del conjunto de validacion
    plot_confusion_matrix(cm_val, classes=['clase 0', 'clase 1'])
    print('-'*70)
```

Con el uso de estas funciones se entrenó y calculó los desempeños y matrices de confusión de los modelos de LogisticRegression() configurados de las siguientes formas:

1) Modelo 1: En el primer experimento, se utilizó la configuración por default del modelo, lo cual corresponde a los siguientes parámetros:

- **penalty='l2'**
- dual=False
- tol=0.0001
- C=1.0
- fit_intercept=True
- intercept_scaling=1
- class_weight=None
- random_state=None
- **solver='lbfgs'**
- max_iter=100
- multi_class='auto'
- verbose=0
- warm_start=False
- n_jobs=None
- **l1_ratio=None**

```
modelo_1 = LogisticRegression()
entrenar_modelo(modelo_1,X_train,X_val,y_train,y_val)
```

2) Modelo 2: En el segundo experimento, se cambió el parámetro requerido en el enunciado, pero para hacer este cambio fue necesario actualizar el solver para que la función de perdida pudiera ser soportada:

- **penalty='l1'**
- **solver='liblinear'**

```
modelo_2 = LogisticRegression(penalty='l1', solver='liblinear')
entrenar_modelo(modelo_2, X_train, X_val, y_train, y_val)

```

3) Modelo 3: El tercer experimento, corresponde a tener fijo el mismo penalty del modelo 2, pero probando el otro solver posible que soporta la función de perdida l1:

- **penalty='l1'**
- **solver='saga'**

```
modelo_3 = LogisticRegression(penalty='l1', solver='saga')
entrenar_modelo(modelo_3, X_train, X_val, y_train, y_val)

```

4) Modelo 4: En el cuarto experimento, se cambió el parámetro requerido en el enunciado nuevamente y se dejó fijo el solver, pero en esta ocasión se quiso experimentar cambiando un tercer parámetro:

- **penalty='elascticnet'**
- **solver='saga'**
- **l1_ratio=1**

```
modelo_4 = LogisticRegression(penalty='elasticnet', solver='saga', l1_ratio=1)
entrenar_modelo(modelo_4, X_train, X_val, y_train, y_val)

```

5) Modelo 5: El último experimento, es similar al modelo 4, pero cambiando el último parámetro agregado:

- **penalty='elascticnet'**
- **solver='saga'**
- **l1_ratio=0**

```
modelo_5 = LogisticRegression(penalty='elasticnet', solver='saga', l1_ratio=0)
entrenar_modelo(modelo_5, X_train, X_val, y_train, y_val)

```

Con los experimentos realizados, en la siguiente sección se procederá a generar y mostrar los resultados obtenidos.

IV. Resultados

A continuación se presentan los resultados obtenidos en la parte 1 y 2 de este trabajo, se comenzará mostrando los resultados de la fase de Análisis Exploratorio de Datos, para luego mostrar los resultados obtenidos en los experimentos de Clasificación con cada modelo de regresión logística.

A. Resultados Parte 1; Análisis Exploratorio de Datos:

La base de datos de calidad de vino cuenta con **6497 filas** y **12 columnas**, en las cuales no existen valores NaN. Además, existen **4113 Vinos de calidad High** y **2384 Vinos de Calidad low**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6497 entries, 0 to 6496
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         6497 non-null   float64
 1   volatile acidity      6497 non-null   float64
 2   citric acid           6497 non-null   float64
 3   residual sugar        6497 non-null   float64
 4   chlorides             6497 non-null   float64
 5   free sulfur dioxide   6497 non-null   float64
 6   total sulfur dioxide  6497 non-null   float64
 7   density               6497 non-null   float64
 8   pH                    6497 non-null   float64
 9   sulphates             6497 non-null   float64
 10  alcohol               6497 non-null   float64
 11  quality               6497 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 609.2 KB
```

Fig. 2: Información Base de Datos.

Los resultados de graficar los histogramas para cada una de las calidades de vino para todas sus características se muestran a continuación:

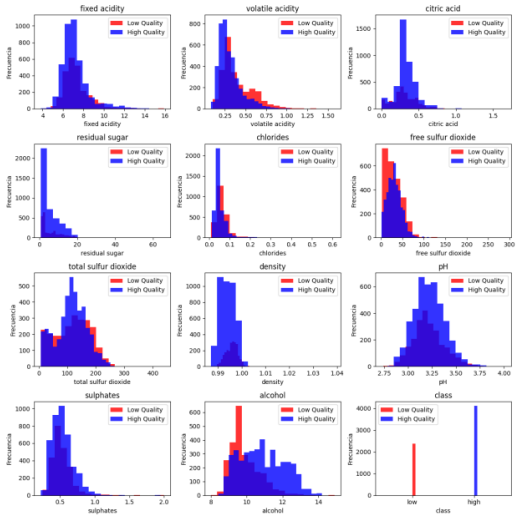

Fig. 3: Histogramas Superpuestos

El resultado de calcular y graficar la matriz de correlación, codificando las clases high y low a 1 y 0 respectivamente, se muestra a continuación:

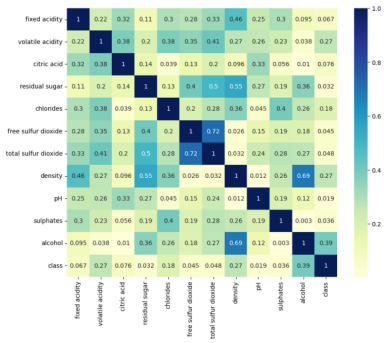

Fig. 4: Gráfico Matriz de Correlación.

Las características más relacionadas con la columna class, ordenadas de forma descendente:

Característica	Coeficiente
alcohol	0.3947
density	0.2689
volatile acidity (v.a)	0.2670
chlorides	0.1819
citric acid (c.a)	0.0757
fixed acidity (f.a)	0.0674
total sulfur dioxide (t.s.d)	0.0476
free sulfur dioxide (f.s.d)	0.0448
sulphates	0.0358
residual sugar (r.s)	0.0325
pH	0.0188

Los 5 pares de características más relacionadas entre sí en el set de datos ordenados de forma descendente, son los siguientes:

Par de Características	Coeficiente
f.s.d y t.s.d	0.7209
density y alcohol	0.687
r.s y density	0.5525
r.s y t.s.d	0.4954
f.a y density	0.4589

El gráfico de dispersión entre las dos características más correlacionadas, se muestra a continuación:

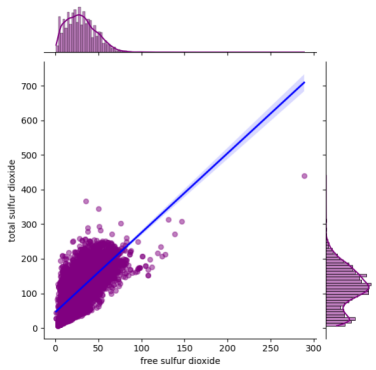

Fig. 5: Gráfico de dispersión Free Sulfur Dioxide vs Total Sulfur Dioxide.

Los 5 pares de características menos correlacionados entre sí, ordenados de forma descendente, se muestran a continuación:

Par de Características	Coeficiente
sulphates y alcohol	0.0030
c.a y alcohol	0.0105
density y pH	0.0117
pH y class	0.0188
f.s.d y density	0.0257

El gráfico de dispersión de las características menos correlacionadas del data set, se muestra a continuación:

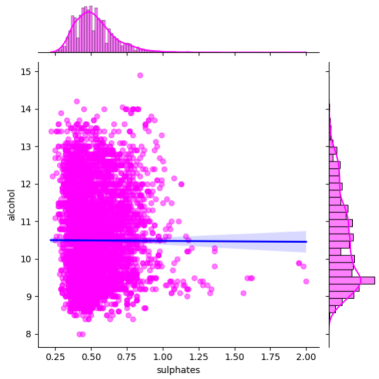

Fig. 6: Gráfico de Dispersión alcohol vs sulphates.

B. Resultados Parte 2; Experimentos de Clasificación:

1) **Prueba de Representatividad de los datos**: Al calcular el promedio de la clase positiva en los distintos conjuntos de datos, se obtuvo lo siguiente:

Conjunto de datos	Proporción de clase positiva (%)
Completo	63.31
Entrenamiento	62.81
Validación	65.31

2) **Resultados Modelo 1:** Los resultados del modelo 1 para cada uno de los coeficientes requeridos en el enunciado son:

```
--------------------------------------------------------
Accuracy en el conjunto de entrenamiento:       74.22%
Accuracy en el conjunto de validación:          73.15%
--------------------------------------------------------
Parámetro de intercepción: [-1.7527268]

Coeficientes de las variables/características:
                 feature      coef
0          fixed acidity -0.098616
1       volatile acidity -4.825566
2            citric acid -0.437381
3         residual sugar  0.051595
4              chlorides -0.443221
5    free sulfur dioxide  0.018744
6   total sulfur dioxide -0.010033
7                density -1.648390
8                     pH -1.333902
9              sulphates  2.284737
10               alcohol  0.957046
--------------------------------------------------------
```

Fig. 7: Resultados Modelo 1.

Luego, el gráfico de la matriz de confusión para el modelo 1, en el conjunto de entrenamiento, se presenta en la Figura 8.

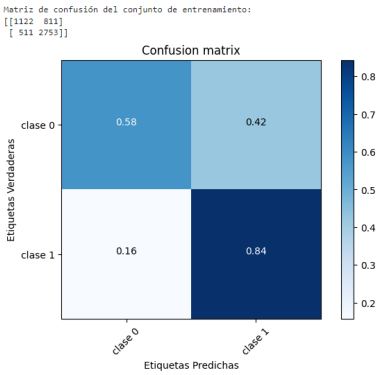

Fig. 8: Matriz de Confusión Modelo 1 Entrenamiento.

El gráfico de la matriz de confusión para el modelo 1, en el conjunto de validación, se presenta en la Figura 9.

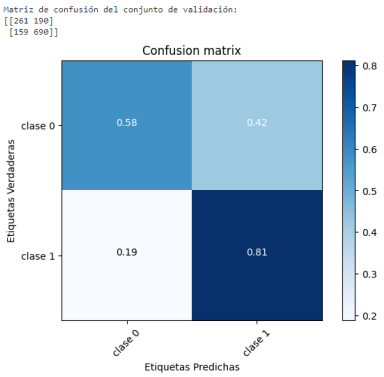

Fig. 9: Matriz de Confusión Modelo 1 Validacion.

3) **Resultados Modelo 2:** Los resultados del modelo 2 para cada uno de los coeficientes requeridos en el enunciado son:

```
--------------------------------------------------------
Accuracy en el conjunto de entrenamiento:       74.56%
Accuracy en el conjunto de validación:          72.38%
--------------------------------------------------------
Parámetro de intercepción: [-4.58651098]

Coeficientes de las variables/características:
                 feature      coef
0          fixed acidity  0.014240
1       volatile acidity -4.712872
2            citric acid -0.684501
3         residual sugar  0.070637
4              chlorides  0.000000
5    free sulfur dioxide  0.018109
6   total sulfur dioxide -0.008265
7                density -4.619341
8                     pH  0.148077
9              sulphates  1.991199
10               alcohol  0.962766
--------------------------------------------------------
```

Fig. 10: Resultados Modelo 2.

Luego la matriz de confusión para el modelo 2, en el conjunto de entrenamiento, se presenta en la Figura 11
El gráfico de la matriz de confusión para el modelo 2, en el conjunto de validación, se presenta en la Figura 12

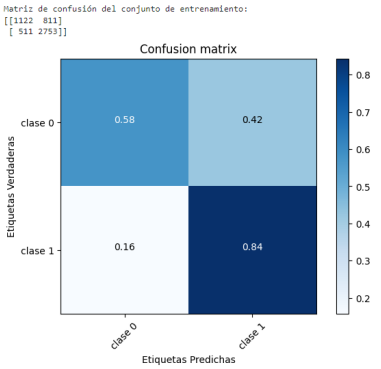

Fig. 11: Matriz de Confusión Modelo 2 Entrenamiento.

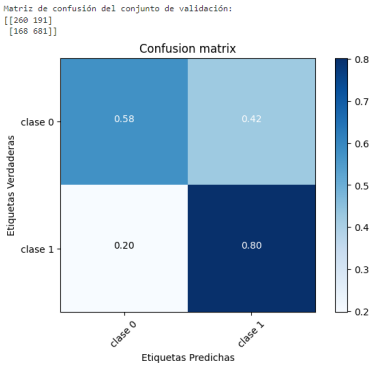

Fig. 12: Matriz de Confusión Modelo 2 Validación.

4) **Resultados Modelo 3:** Los resultados del modelo 3 para cada uno de los coeficientes requeridos en el enunciado son:

```
------------------------------------------------------------
Accuracy en el conjunto de entrenamiento:       65.58%
Accuracy en el conjunto de validación:          65.69%
------------------------------------------------------------
Parámetro de intercepción: [-0.07231035]

Coeficientes de las variables/características:
                 feature      coef
0          fixed acidity -0.290437
1       volatile acidity -0.195955
2            citric acid  0.050103
3         residual sugar  0.022044
4              chlorides -0.020417
5    free sulfur dioxide  0.018273
6   total sulfur dioxide -0.009278
7                density -0.072011
8                     pH -0.254991
9              sulphates -0.003060
10               alcohol  0.393767
------------------------------------------------------------
```

Fig. 13: Resultados Modelo 3.

La matriz de confusión para el modelo 3, en el conjunto de entrenamiento, se presenta en la Figura 14

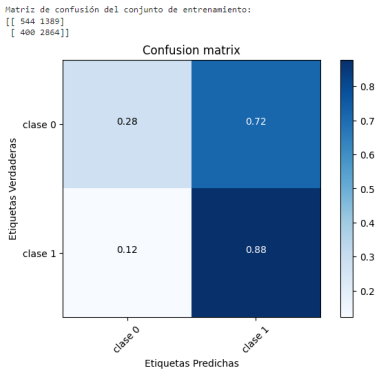

Fig. 14: Matriz de Confusión Modelo 3 Entrenamiento.

La matriz de confusión para el modelo 3. en el conjunto de validación, se presenta en la Figura 15

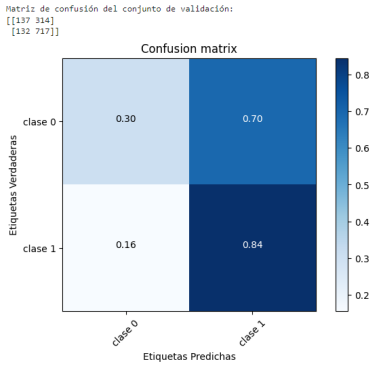

Fig. 15: Matriz de Confusión Modelo 3 Validación.

5) **Resultados Modelo 4:** Los resultados del modelo 4 para cada uno de los coeficientes requeridos en el enunciado son:

```
------------------------------------------------------------
Accuracy en el conjunto de entrenamiento:       65.56%
Accuracy en el conjunto de validación:          65.69%
------------------------------------------------------------
Parámetro de intercepción: [-0.07224636]

Coeficientes de las variables/características:
                 feature      coef
0          fixed acidity -0.290672
1       volatile acidity -0.196075
2            citric acid  0.050119
3         residual sugar  0.022064
4              chlorides -0.020475
5    free sulfur dioxide  0.018278
6   total sulfur dioxide -0.009277
7                density -0.071949
8                     pH -0.254893
9              sulphates -0.003045
10               alcohol  0.393914
------------------------------------------------------------
```

Fig. 16: Resultados Modelo 4.

Luego matriz de confusión para el modelo 4, en el conjunto de entrenamiento, se presenta en la Figura 17.

La matriz de confusión para el modelo 4, en el conjunto de validación, se presenta en la Figura 18.

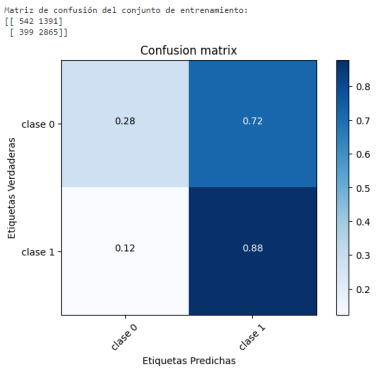

Fig. 17: Matriz de Confusión Modelo 4 Entrenamiento.

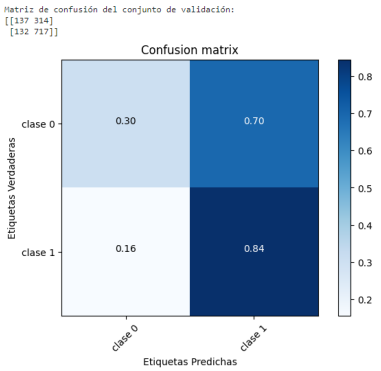

Fig. 18: Matriz de Confusión Modelo 4 Validación.

6) **Resultados Modelo 5:** Los resultados del modelo 5 para cada uno de los coeficientes requeridos en el enunciado son:

```
------------------------------------------------------------
Accuracy en el conjunto de entrenamiento:       65.58%
Accuracy en el conjunto de validación:          65.69%
------------------------------------------------------------
Parámetro de intercepción: [-0.07223006]

Coeficientes de las variables/características:
                feature      coef
0         fixed acidity -0.290706
1      volatile acidity -0.197389
2           citric acid  0.051911
3        residual sugar  0.022143
4             chlorides -0.022115
5   free sulfur dioxide  0.018279
6  total sulfur dioxide -0.009283
7               density -0.073383
8                    pH -0.256045
9             sulphates -0.004260
10              alcohol  0.394478
------------------------------------------------------------
```

Fig. 19: Resultados Modelo 5.

Luego matriz de confusión para el modelo 4, en el conjunto de entrenamiento, se presenta en la Figura 20.

La matriz de confusión para el modelo 5, en el conjunto de validación, se presenta en la Figura 21.

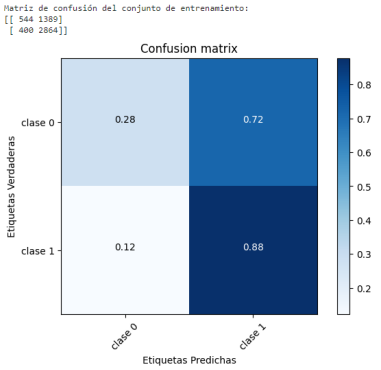

Fig. 20: Matriz de Confusión Modelo 5 Entrenamiento.

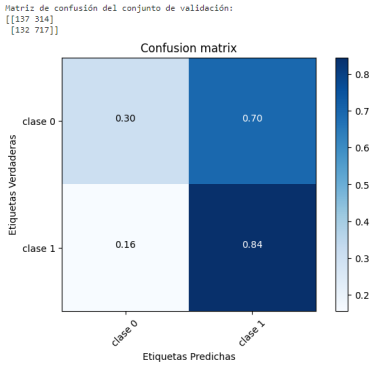

Fig. 21: Matriz de Confusión Modelo 5 Validación.

V. Análisis

En esta sección se analizarán cada uno de los resultados presentados en la sección anterior, se comenzará analizando la parte 1 de análisis exploratorio de datos, para finalizar analizando los resultados de la parte 2 de clasificación.

A. Análisis Resultados Parte 1:

En la parte 1 se realizaron una serie de cálculos y gráficos para reconocer la verdadera naturaleza de los datos, sus distribuciones y sus correlaciones entre características, al respecto se puede analizar lo siguiente:

- **Análisis Exploratorio de Datos:** Con los resultados de la cantidad de vinos de calidad high y low, se puede establecer que existen más vinos con notas iguales o superiores a **6**, por lo que la viña que los produce, en general, genera vinos con altos estándares de calidad.
- **Histogramas:** Al analizar los histogramas para cada característica, se puede observar que los vinos de calidad high tienen una menor cantidad de dióxido de azufre libre, además tienen menores cantidades de acidez volátil, también el grado alcohólico de los vinos high en general es más alto que los vinos low.
- **Correlaciones:** En cuanto al análisis de correlaciones, este confirma el análisis previo de los histogramas, debido a que la característica más correlacionada con la columna class es el alcohol, con un valor de **0.3946**. Respecto al par de características más correlacionadas, el resultado es bastante esperable, debido a que mientras más exista

Dióxido de Azufre Libre en los vinos, el Total del Dióxido de Azufre aumentará. Luego, respecto al par de características menos correlacionadas, es un poco más difícil de analizar, pero se entiende, que el nivel de alcohol no varía ni tiene nada que ver con los sulfatos, que son conservantes, antioxidantes y antimicrobianos.

B. Análisis Resultados Parte 2:

En la parte 2, se realizaron distintos experimentos de regresión logística para clasificar los vinos en 2 clases distintas, cambiando la función de perdida y los solvers, a continuación se muestra el análisis de los resultados para esta parte:

- **Prueba de Representatividad de los conjuntos de datos:** Se puede establecer que **si** existe una buena representatividad en cada conjunto de datos de entrenamiento y validación, ya que se mantuvo una proporción cercana al **60%** en ambos conjuntos, lo cuál es similar al **63.31%** real de proporción de la clase positiva en el conjunto completo.
- **Análisis del Mejor Modelo:** En cuanto a la elección del mejor modelo para la tarea de clasificación, se ha escogido el **Modelo 1**, con todos los parámetros en default, debido a que consiguió el mejor reporte de Accuracy en el conjunto de validación **73.15%**. Luego, se pudo notar que al cambiar la función de perdida a $l1$ y debido a esto, cambiar el solver a **liblinear/saga**, los resultados de accuracy fueron **72.38% y 65.69%** respectivamente, por lo que los modelos 2 y 3 no pudieron superar al modelo 1.
  Finalmente, al utilizar la función de perdida **elasticnet** junto a su único solver posible **saga**, los modelos 4 y 5 tampoco pudieron superar en accuracy al modelo 1. Además, se puede notar que el mejor modelo para clasificar ambas clases, las cuales se encuentran desbalanceadas, fue el modelo 1, debido a que en las matrices de confusión de los modelos con las funciones de perdida **l1 y elasticnet**, cometieron muchos errores al reconocer la clase 0 (low), debido a que había menos muestras de esta clase, por lo tanto, clasificaron muchos Falsos Positivos.

VI. Conclusiones

En este informe, se trabajó con un conjunto de datos de vinos procedentes de Portugal, había vinos blancos y tintos, la columna objetiva era la calidad de estos, que variaban desde 1 a 10. La primera parte del trabajo implicó un análisis exploratorio de datos para reconocer la verdadera naturaleza de los datos, sus distribuciones y las correlaciones entre las características. La segunda parte del trabajo consistió en experimentar con modelos de regresión logística para clasificar los vinos en dos clases distintas, utilizando diferentes funciones de pérdida y solucionadores.

El análisis Exploratorio de Datos, reveló que el viñedo generalmente produce vinos de alta calidad, con más vinos con calificaciones iguales o superiores a 6. Los vinos de alta calidad tienen niveles más bajos de dióxido de azufre libre y acidez volátil, mientras que su contenido de alcohol es generalmente más alto que los vinos de baja calidad. El análisis de correlación confirmó estos hallazgos, siendo el alcohol la característica más correlacionada con la columna class. Además, las características menos correlacionadas fueron el alcohol y los sulfatos, lo que indica que la presencia de sulfatos, no afecta en nada el nivel de alcohol en los vinos.

También se evaluó la capacidad de los modelos de regresión logística para clasificar los vinos con precisión. El análisis mostró que los conjuntos de datos de entrenamiento y validación tenían una buena representación de ambas clases, con una proporción similar al conjunto de datos real. El mejor modelo para la clasificación fue el Modelo 1, que logró una precisión del 73,15% en el conjunto de validación, utilizando los parámetros predeterminados. Los modelos con diferentes funciones de pérdida y solucionadores no pudieron superar la precisión del Modelo 1. El análisis reveló que los modelos con funciones de pérdida l1 y elasticnet clasificaron muchos falsos positivos en la clase 0 (baja) debido a las clases desbalanceadas. Por lo tanto, el Modelo 1 fue el mejor modelo para clasificar los vinos con precisión.

Bibliografía

[1] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*. Morgan Kaufmann, 2011.
[2] A. K. Jain, M. N. Murty, and P. J. Flynn, *Data clustering: 50 years beyond K-means*. CRC Press, 2010.
[3] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009.