Análisis de datos: **Calidad del agua**

Gonzalo Gómez

Universidad Nacional Andrés Bello
Facultad de Ingeniería
Magister en Ingeniería Informática

1.- Montar Drive e importar librerías

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

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import datasets
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
```

2.- Leer y describir CSV

```
           Ph      Dureza      Solidos  Cloraminas     Sulfato  \
0         NaN  204.890456  20791.31898    7.300212  368.516441
1    3.716080  129.422921  18630.05786    6.635246         NaN
2    8.099124  224.236259  19909.54173    9.275884         NaN
3    8.316766  214.373394  22018.41744    8.059332  356.886136
4    9.092223  181.101509  17978.98634    6.546600  310.135738
...       ...         ...          ...         ...         ...
3271 4.668102  193.681736  47580.99160    7.166639  359.948574
3272 7.808856  193.553212  17329.80216    8.061362         NaN
3273 9.419510  175.762646  33155.57822    7.350233         NaN
3274 5.126763  230.603758  11983.86938    6.303357         NaN
3275 7.874671  195.102299  17404.17706    7.509306         NaN

      Conductividad  Carbono Organico  Trihalometanos  Turbiedad Potabilidad
0        564.308654         10.379783       86.990970   2.963135  No Potable
1        592.885359         15.180013       56.329076   4.500656  No Potable
2        418.606213         16.868637       66.420093   3.055934  No Potable
3        363.266516         18.436525      100.341674   4.628771  No Potable
4        398.410813         11.558279       31.997993   4.075075  No Potable
...             ...               ...             ...        ...         ...
3271     526.424171         13.894419       66.687695   4.435821     Potable
3272     392.449580         19.903225             NaN   2.798243     Potable
3273     432.044783         11.039070       69.845400   3.298875     Potable
3274     402.883113         11.168946       77.488213   4.708658     Potable
3275     327.459761         16.140368       78.698446   2.309149     Potable

[3276 rows x 10 columns]
```

```
Ph                 float64
Dureza             float64
Solidos            float64
Cloraminas         float64
Sulfato            float64
Conductividad      float64
Carbono_Organico   float64
Trihalometanos     float64
Turbiedad          float64
Potabilidad         object
dtype: object
```

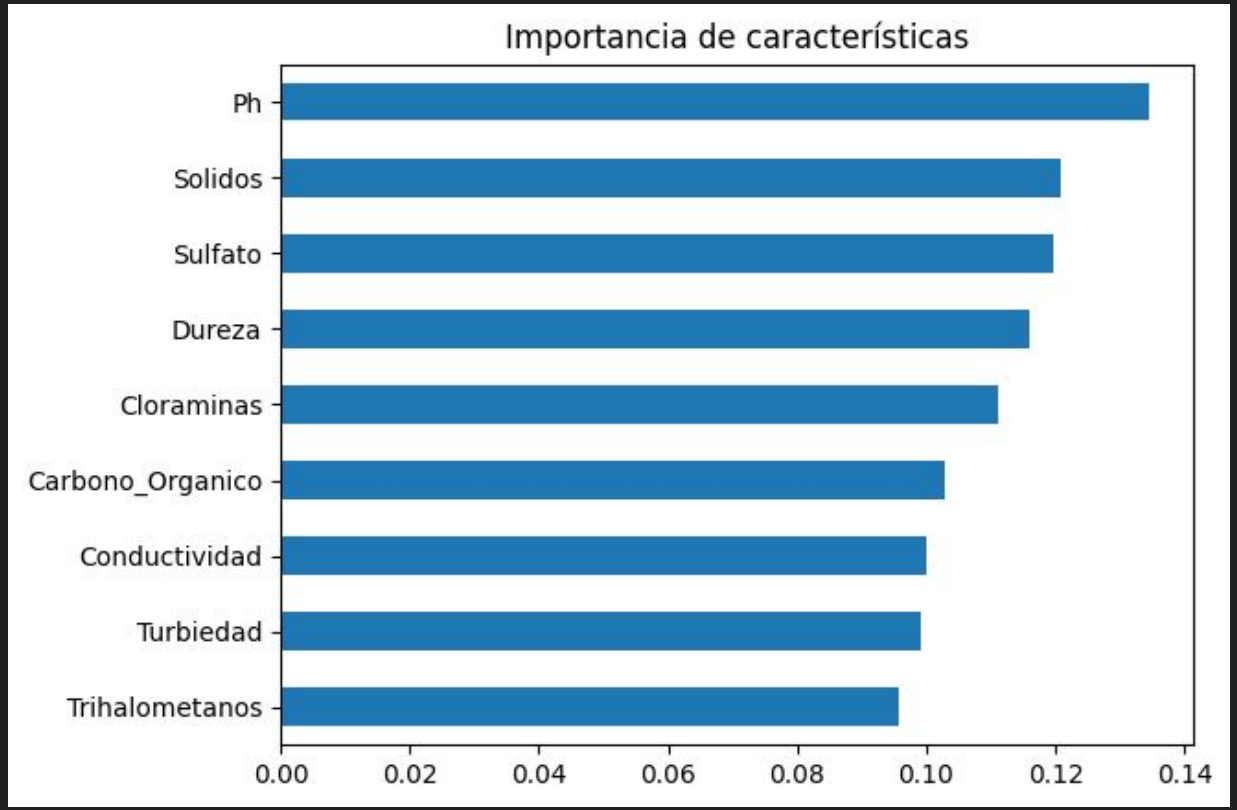
2.- Leer y describir CSV

```
      Ph        Dureza       Solidos   Cloraminas      Sulfato  \
count  2785.000000  3276.000000   3276.000000  3276.000000  2495.000000
mean      7.080795   196.369496  22014.092526     7.122277   333.775777
std       1.594320    32.879761   8768.570828     1.583085    41.416840
min       0.000000    47.432000    320.942611     0.352000   129.000000
25%       6.093092   176.850538  15666.690300     6.127421   307.699498
50%       7.036752   196.967627  20927.833605     7.130299   333.073546
75%       8.062066   216.667456  27332.762125     8.114887   359.950170
max      14.000000   323.124000  61227.196010    13.127000   481.030642
```

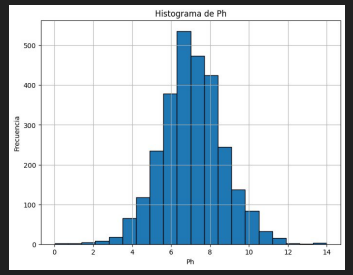
```
       Conductividad  Carbono_Organico  Trihalometanos     Turbiedad
count    3276.000000       3276.000000     3114.000000  3276.000000
mean      426.205111         14.284970       66.396293     3.966786
std        80.824064          3.308162       16.175008     0.780382
min       181.483754          2.200000        0.738000     1.450000
25%       365.734414         12.065801       55.844536     3.439711
50%       421.884968         14.218338       66.622485     3.955028
75%       481.792305         16.557652       77.337473     4.500320
max       753.342620         28.300000      124.000000     6.739000
```

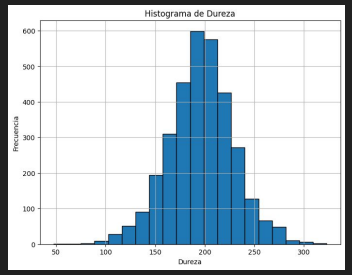
```
Data columns (total 10 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Ph                2785 non-null   float64
 1   Dureza            3276 non-null   float64
 2   Solidos           3276 non-null   float64
 3   Cloraminas        3276 non-null   float64
 4   Sulfato           2495 non-null   float64
 5   Conductividad     3276 non-null   float64
 6   Carbono_Organico  3276 non-null   float64
 7   Trihalometanos    3114 non-null   float64
 8   Turbiedad         3276 non-null   float64
 9   Potabilidad       3276 non-null   object
dtypes: float64(9), object(1)
memory usage: 256.1+ KB
```

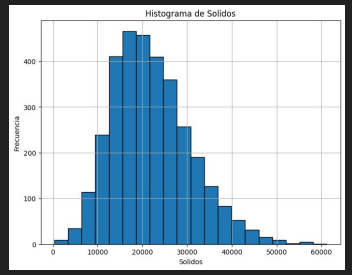
2.- Lectura y descripción del CSV

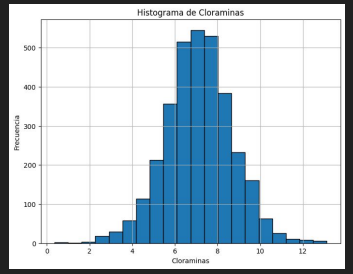


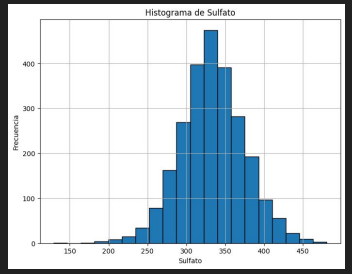
3.- Histogramas de variables

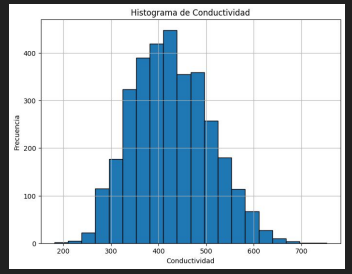


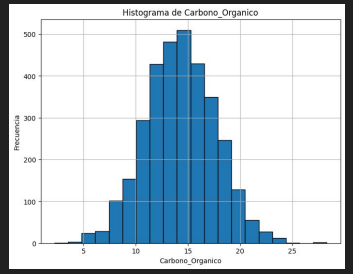


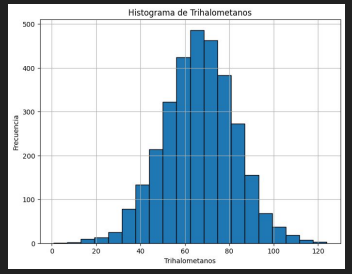


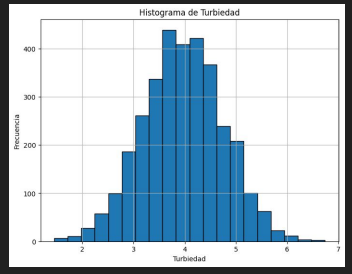


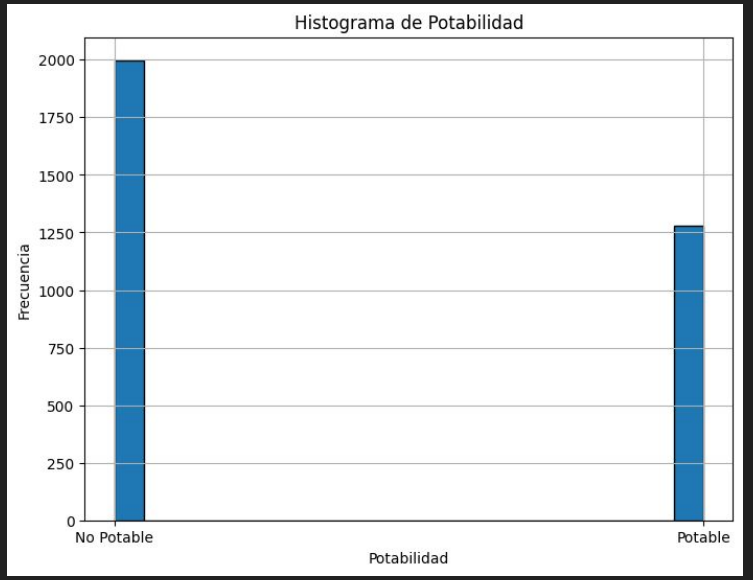




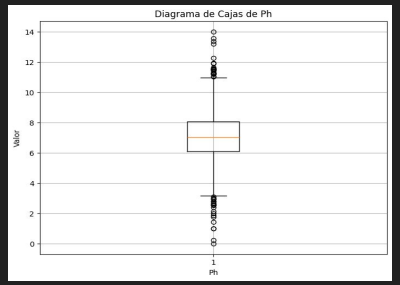


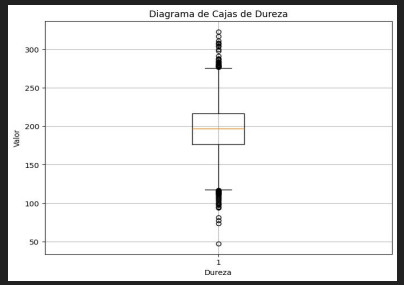


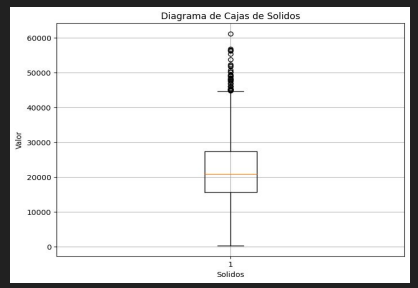


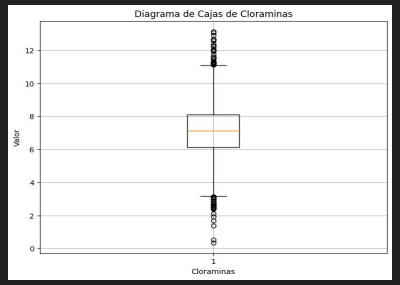


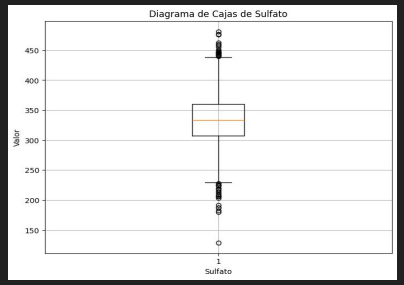
3.1.- Boxplot de variables

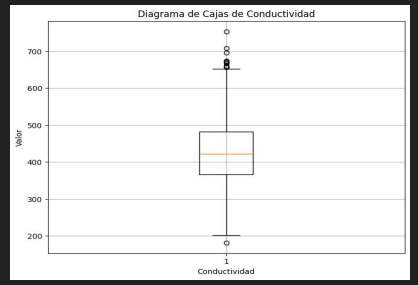


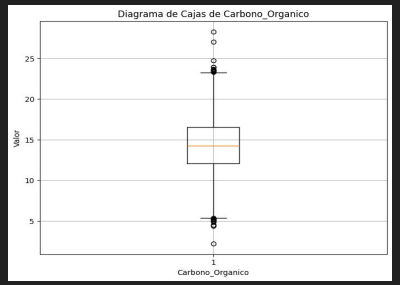


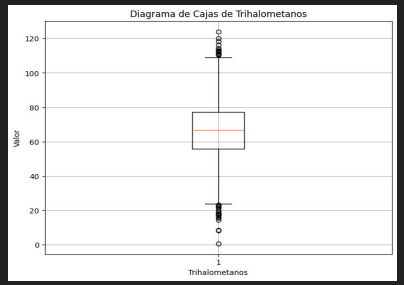


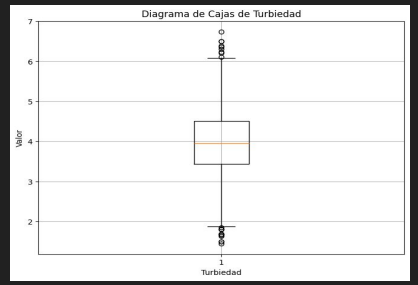




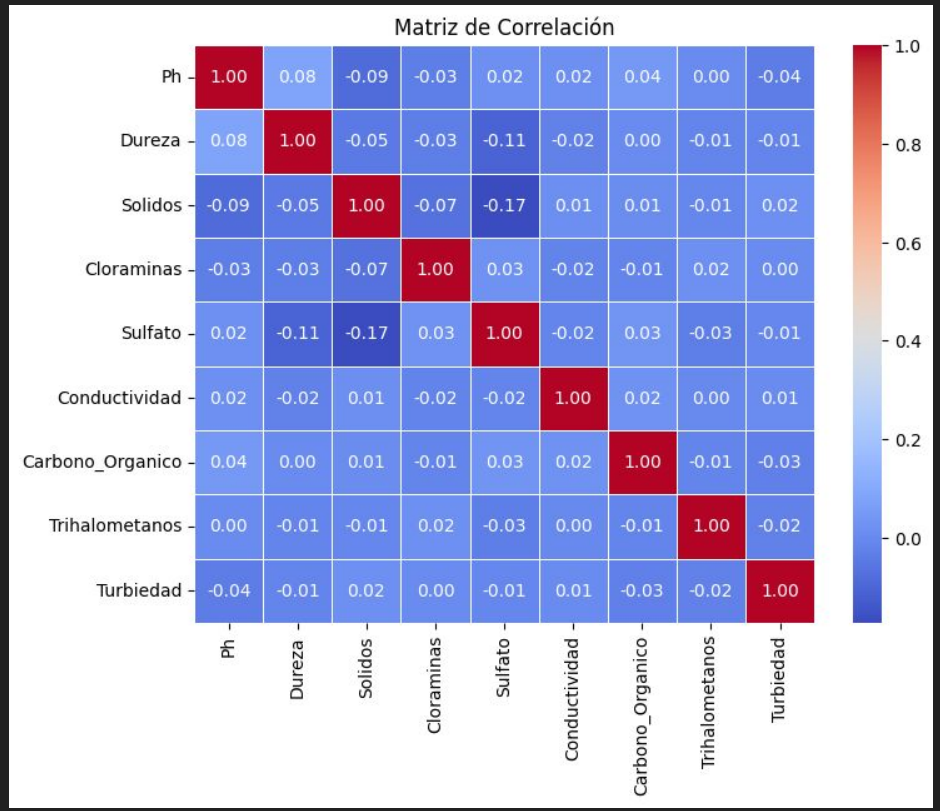








4.- Matriz de correlación



5.- Valores nulos

```
Ph                 2785
Dureza             3276
Solidos            3276
Cloraminas         3276
Sulfato            2495
Conductividad      3276
Carbono_Organico   3276
Trihalometanos     3114
Turbiedad          3276
Potabilidad        3276
dtype: int64
```

```
Ph                 491
Dureza               0
Solidos              0
Cloraminas           0
Sulfato            781
Conductividad        0
Carbono_Organico     0
Trihalometanos     162
Turbiedad            0
Potabilidad          0
dtype: int64
```

6.- Imputación de Datos

```
from sklearn.impute import SimpleImputer
# Seleccionar las columnas que deseas imputar
columnas_a_imputar = ['Ph', 'Dureza', 'Solidos',
'Cloraminas', 'Sulfato',
'Conductividad','Carbono_Organico',
'Trihalometanos', 'Turbiedad']

# Crear un objeto SimpleImputer
imputer = SimpleImputer(strategy='median') # ,mean,
median , most_frequent

# Imputar solo las columnas seleccionadas
df[columnas_a_imputar] =
imputer.fit_transform(df[columnas_a_imputar])
```

```
from fancyimpute import IterativeImputer
import pandas as pd

# Seleccionar columnas a imputar
columnas_a_imputar = ['Ph', 'Dureza', 'Solidos',
'Cloraminas', 'Sulfato', 'Conductividad',
       'Carbono_Organico', 'Trihalometanos',
'Turbiedad']
data = df[columnas_a_imputar]

# Imputación con MICE
imputer = IterativeImputer()
data_imputado = imputer.fit_transform(data)

# Reemplazar en el DataFrame original
df[columnas_a_imputar] = data_imputado
```

7.- Balance de datos usando SMOTE

```
from imblearn.over_sampling import SMOTE
smote = SMOTE(random_state=42)
X_res, y_res = smote.fit_resample(X, y)
```

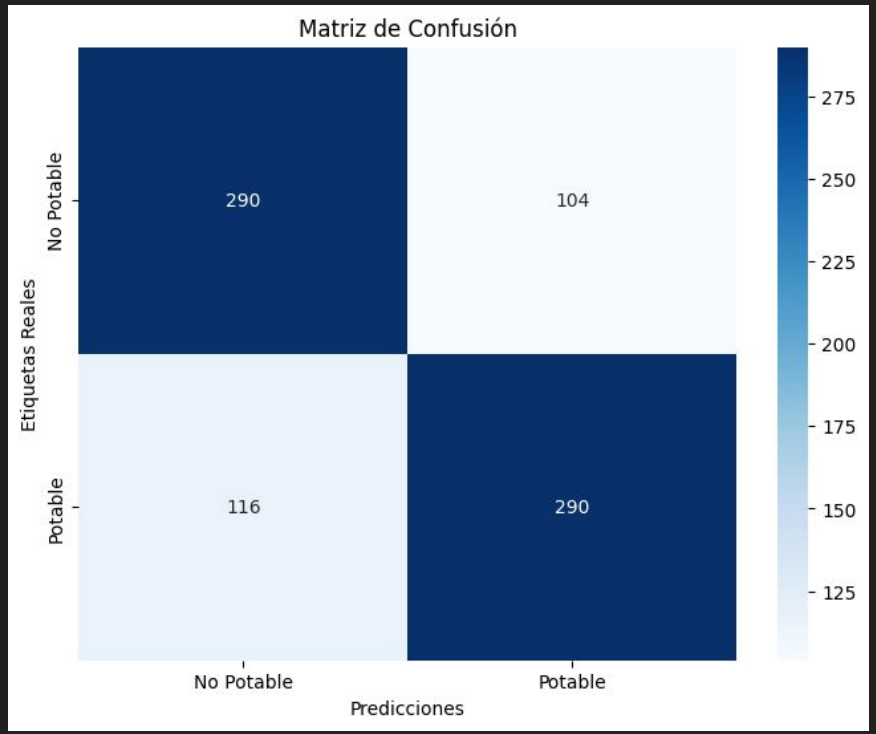
```
Valores en y: Potabilidad
No Potable    1998
Potable       1278
Name: count, dtype: int64
¿Hay valores nulos aún? 0 0

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

Valores en y: Potabilidad
No Potable    1998
Potable       1998
Name: count, dtype: int64
¿Hay valores nulos aún? 0 0
```

8.- Entrenamiento y pruebas

```
# Dividir los datos en conjunto de entrenamiento y conjunto de prueba
X_train, X_test, y_train, y_test = train_test_split(X_res, y_res, test_size=0.2, random_state=42)

# Crear el modelo de Árbol de Decisión
from sklearn.ensemble import RandomForestClassifier

modelo = RandomForestClassifier(n_estimators=100,class_weight='balanced', random_state=42)
#modelo = DecisionTreeClassifier(random_state=42)

# Entrenar el modelo con los datos de entrenamiento
modelo.fit(X_train, y_train)

# Hacer predicciones sobre los datos de prueba
y_pred = modelo.predict(X_test)
# Evaluar el rendimiento del modelo
print("Precisión del modelo:", accuracy_score(y_test, y_pred))
print("\nReporte de clasificación:")
print(classification_report(y_test, y_pred))
```

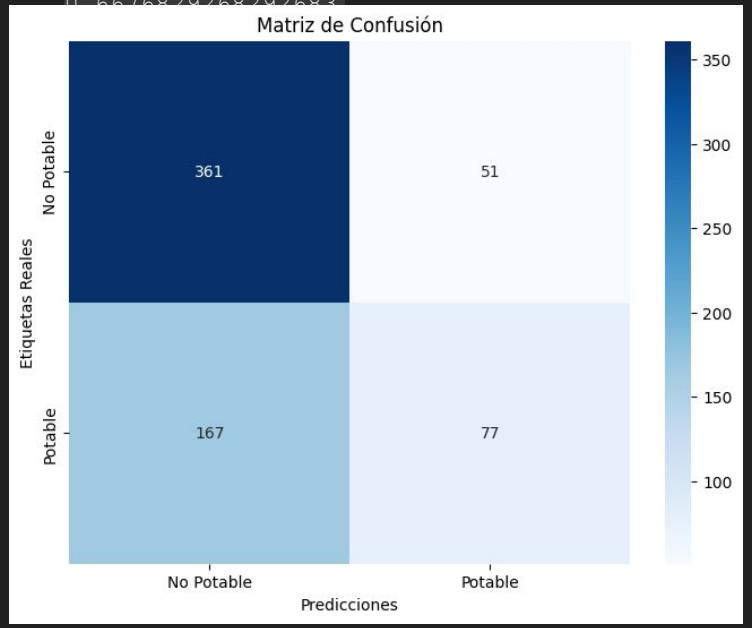
9.- Evaluación Métricas y Matriz de Confusión

```
Precisión del modelo: 0.725

Reporte de clasificación:
              precision    recall  f1-score   support

  No Potable       0.71      0.74      0.72       394
     Potable       0.74      0.71      0.72       406

    accuracy                           0.72       800
   macro avg       0.73      0.73      0.72       800
weighted avg       0.73      0.72      0.72       800
```

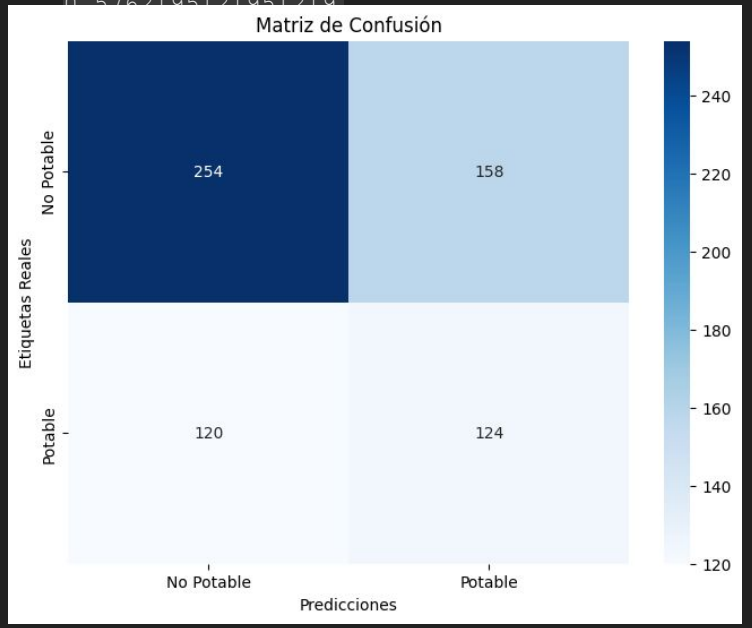


9.1.- Errores que tuve

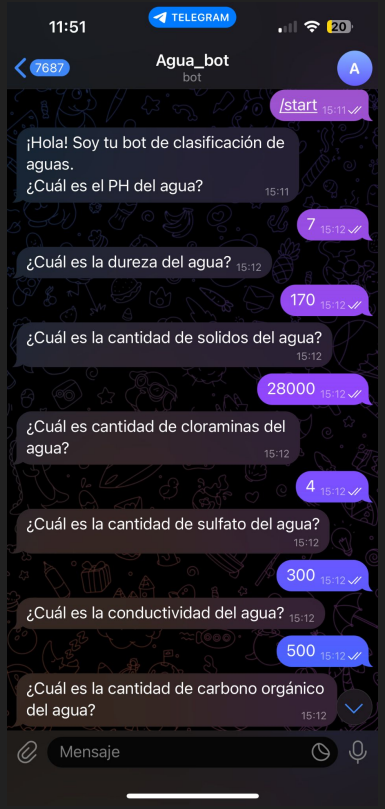
```
SIN SMOTE - RandomForestClassifier
Precisión del modelo:
0.6676829268292683
```

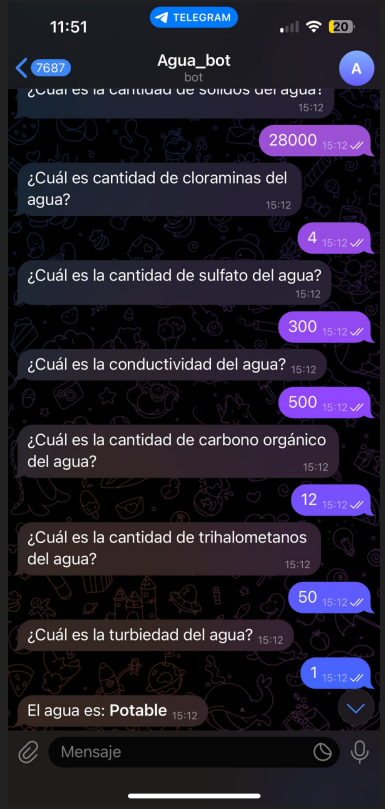



```
SIN SMOTE - DecisionTreeClassifier
Precisión del modelo:
0.5762195121951219
```



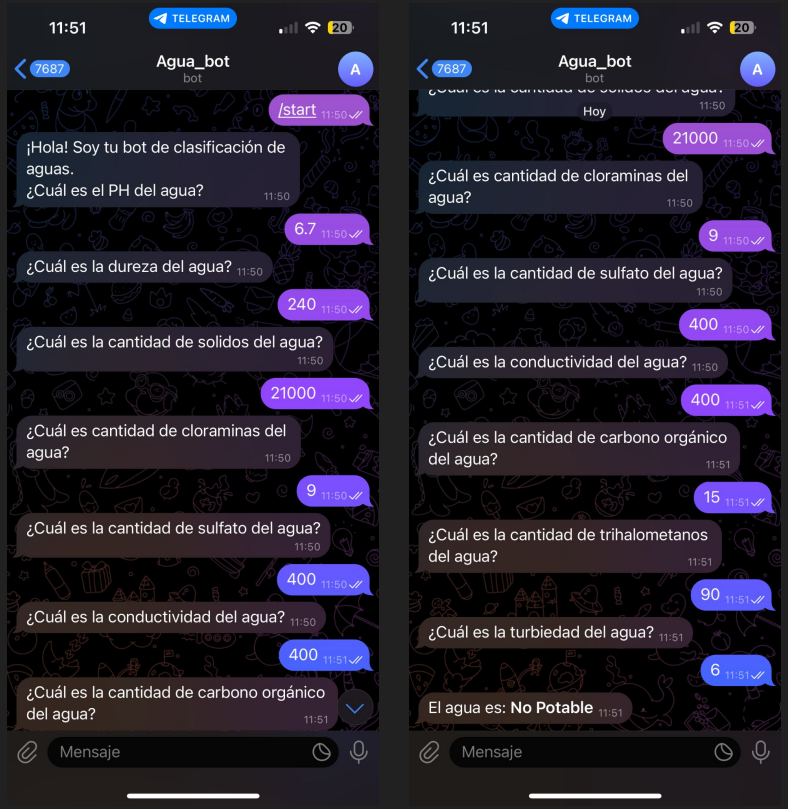
11.- Prototipo Chatbot Telegram: *Ejemplo agua potable*





UNAB_Agua_bot

11.- Prototipo Chatbot Telegram: *Ejemplo agua **no** potable*



UNAB_Agua_bot