UNIVERSITAT POLITÈCNICA DE CATALUNYA – UNIVERSITAT DE BARCELONA
Máster en Ingeniería Biomédica

MODELO DE PREDICCIÓN DE INSUFICIENCIA RENAL AGUDA PARA PACIENTES CON RABDOMIÓLISIS

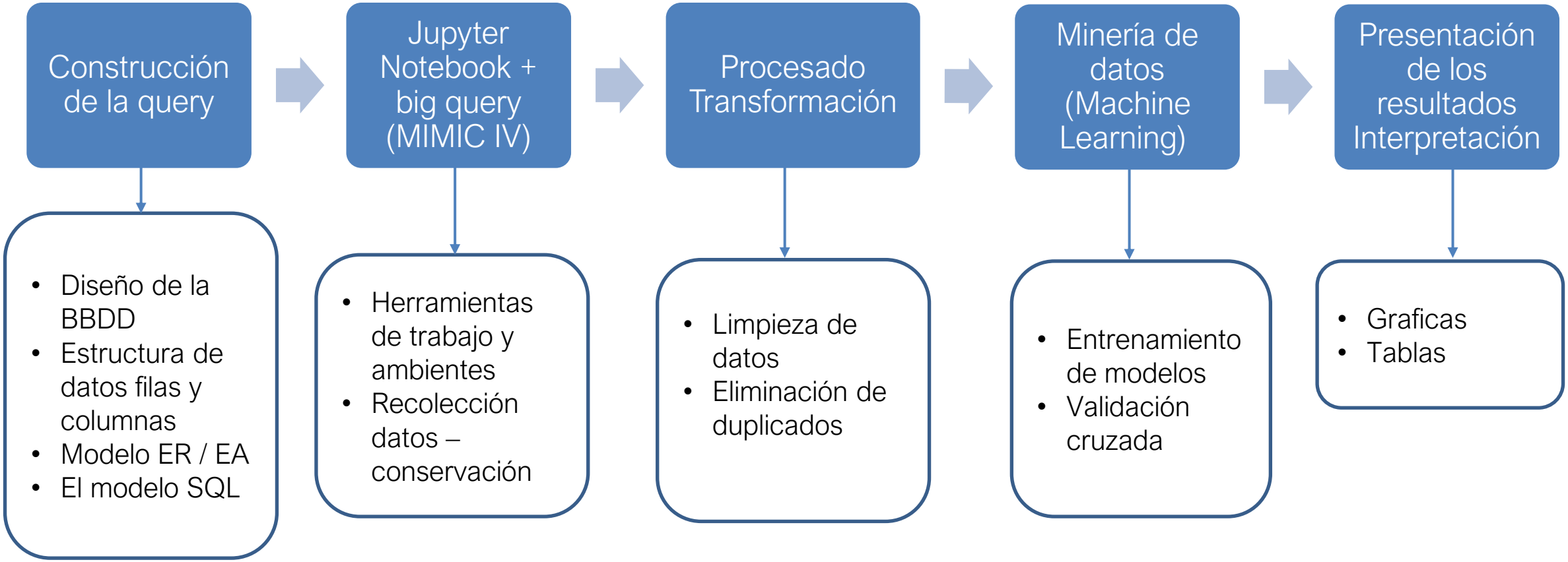
Marc Palomer
June Alberdi
Miranda Silveria
Yanelky Fabian

Tutor: Dr. Guido Muñoz
OGICC – Organización y Gestión de la Información y Conocimientos Clínicos

Mayo 2023

ÍNDICE

1. WIKI


Construcción de la query
Jupyter Notebook + big query (MIMIC IV)
Procesado Transformación
Minería de datos (Machine Learning)
Presentación de los resultados Interpretación
• Diseño de la BBDD
• Estructura de datos filas y columnas
• Modelo ER / EA
• El modelo SQL
• Herramientas de trabajo y ambientes
• Recolección datos – conservación
• Limpieza de datos
• Eliminación de duplicados
• Entrenamiento de modelos
• Validación cruzada
• Graficas
• Tablas

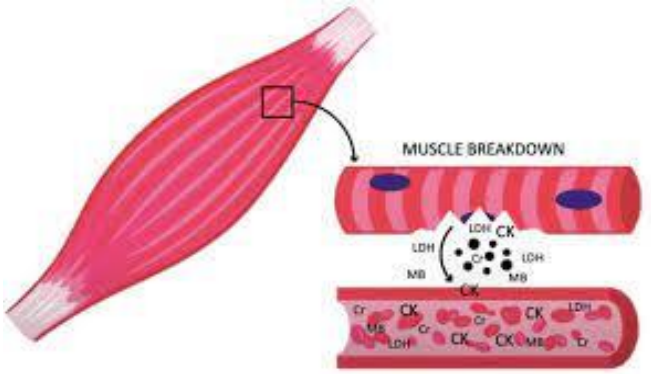
2. CASO CLÍNICO. INTRODUCCIÓN

OBJETIVO: Desarrollar un modelo de predicción de insuficiencia renal aguda en pacientes diagnosticados con rabdomiólisis.

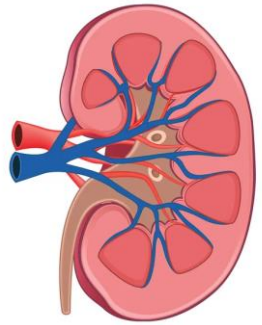
¿Cuál es la problemática?

RABDOMIÓLISIS

- **Traumatismos**
- **Enfermedades musculares**
- **Isquemia**
- **Deshidratación**
- **Cirugías prolongadas**
- **Esfuerzo muscular intenso**
- **Consumo de drogas**
- **Convulsiones**
- **Temperaturas extremas**
- **Infecciones**



LIBERACIÓN DE CREATINA CINASA (CK)



CITOTOXICIDAD DE LAS CK

INSUFICIENCIA RENAL AGUDA

2. CASO CLÍNICO. INTRODUCCIÓN

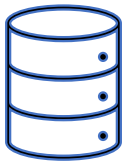
CRITERIOS DE INCLUSIÓN Y EXCLUSIÓN DE LOS PACIENTES

- Pacientes con rabdomiólisis

VARIABLES

Database | Credentialed Access

MIMIC-IV



¿Qué queremos?

• **Género del paciente**
• **Creatina cinasa (CK)** • **Potasio** • **PH**
• **Insuficiencia renal aguda** • **Rabdomiólisis** • **Diabetes** • **Insuficiencia renal crónica** • **Infección**

¿Dónde lo encontramos?

TABLA	ATRIBUTO
patients	**gender**
d_labitems	Buscamos por: label Seleccionamos: **itemid**
d_icd_diagnoses	Buscamos por: long_title Seleccionamos: **Icd_code**

2. CASO CLÍNICO. INTRODUCCIÓN

PROCEDIMIENTOS ÉTICOS Y LEGALES PARA GARANTIZAR LA SEGURIDAD DEL PACIENTE

Pasos para obtener los datos del conjunto de datos MIMIC (Medical Information Mart for Intensive Care):

1. Acceder al sitio web oficial de MIMIC
2. Revisar los requisitos y políticas de acceso
3. Completar los formularios y obtener las aprobaciones necesarias
4. Firmar los acuerdos de uso de datos
5. Descargar los conjuntos de datos

IMPORTANTE:

- Seguir las instrucciones proporcionadas para garantizar una descarga correcta y segura de los datos.
- El acceso a los datos de MIMIC está sujeto a ciertas restricciones y políticas de uso.

2. CASO CLÍNICO. **EXTRACCIÓN,** TRANSFORMACIÓN Y DESCRIPCIÓN

ACCESO A LOS DATOS

- Declaramos **librerías**:

```
import pandas as pd
import numpy as np
import time

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score

from google.oauth2 import service_account
from google.cloud import bigquery
```

```
import xgboost as xgb
import lightgbm as lgb

from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split, cross_val_score
from google.oauth2 import service_account
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier

import pickle
```

- Importamos las **credenciales** (.json) y configuramos el **cliente**.

```
# Definir credenciales
credentials = service_account.Credentials.from_service_account_file("C:/Users/

# Configurar el cliente de Big Query
client = bigquery.Client(credentials=credentials)
```

2. CASO CLÍNICO. **EXTRACCIÓN,** TRANSFORMACIÓN Y DESCRIPCIÓN

ACCESO A LOS DATOS

- Guarda los nombres de los **datasets** tal como los tenemos en BigQuery

```
# Definir el dataset
hosp_dataset_id = 'mimiciv_hosp'
icu_dataset_id = 'mimiciv_icu'
```

- Buscar las **tablas** donde se recogen las variables que nos interesan. ← Analizar el modelo de relaciones que conectan las tablas

```
#Definimos las tablas
admissions_table = 'admissions'
patients_table = 'patients'
d_items_table = 'd_items'
icustays_table = 'icustays'
chartevents_table = 'chartevents'
labevents_table = 'labevents'
procedureevents_table = 'procedureevents'
diagnoses = 'diagnoses_icd'
```

2. CASO CLÍNICO. **EXTRACCIÓN**, TRANSFORMACIÓN Y DESCRIPCIÓN

ACCESO A LOS DATOS

- Para cada variable hacer una **query** sobre la tabla en la que se encuentra para identificar su **código**:

	Dataset	Tabla	Condición	Guardamos
Género	hosp_dataset_id	patient		**gender**
Creatina cinasa (CK)		d_labitems	**Label:** contenga '%creati%' **Fluid:** 'blood'	**itemid**
Potasio			**Label:** contenga '%potassium%' , '%K+%' o '%K%' **Fluid:** 'blood'	
pH			**Label:** contenga '%p_%' y 2 caracteres de longitud	
Insuficiencia renal aguda		diagnoses	**Long_title:** contenga 'acute kidney%'	**icd_code**
Rabdomiólisis			**Long_title:** contenga 'rhab%'	
Diabetes			**Long_title:** contenga '%diabetes%'	
Insuficiencia renal crónica			**Long_title:** contenga 'chronic kidney%'	
Infección			**Long_title:** contenga 'infec%'	

2. CASO CLÍNICO. **EXTRACCIÓN**, TRANSFORMACIÓN Y DESCRIPCIÓN

ACCESO A LOS DATOS

- Utilizamos los códigos para crear una **query** y unificar todas las variables en un dataframe.
- Cálculo del tiempo de consulta.

```
La consulta tardó 35.890196561813354 segundos en ejecutarse.
```

```
data_df.head(5)
```

	subject_id	hadm_id	stay_id	ce_charttime	SEX	crea	ck	potassium	ph	rhabdomiolisis	chronickidneydisease	diab_code	insuf_renal
0	10014136	24097334	30374965	2176-05-12 23:22:00	M	NaN	NaN	NaN	NaN	72888	0	0	5845
1	10014136	24097334	30374965	2176-05-13 00:15:00	M	NaN	NaN	NaN	NaN	72888	0	0	5845
2	10014136	24097334	30374965	2176-05-13 00:41:00	M	NaN	NaN	NaN	NaN	72888	0	0	5845
3	10014136	24097334	30374965	2176-05-13 00:43:00	M	NaN	NaN	NaN	NaN	72888	0	0	5845
4	10014136	24097334	30374965	2176-05-13 00:59:00	M	NaN	NaN	NaN	NaN	72888	0	0	5845

2. CASO CLÍNICO. EXTRACCIÓN, **TRANSFORMACIÓN** Y DESCRIPCIÓN

MANIPULACIÓN DE LA BBDD

Manipulación:

- Eliminar las filas que tienen más de 11 variables vacías
- Eliminar filas duplicadas
- Eliminar filas sin insuficiencia renal aguda (valor a predecir)
- Eliminar la columna de pH

```
# Conteo del número de no-null en cada fila (paciente)
row_non_null_counts = data_df.notnull().sum(axis=1)

# Eliminamos los pacientes con más NaN
df_filtered = data_df[row_non_null_counts >= 11]
df_filtered

#Eliminamos pacientes duplicados
data_dup = df_filtered.drop_duplicates()

#Eliminamos los pacientes con NaN en la columna de insuficiencia renal
data_df=data_dup.dropna(subset='insuf_renal', thresh = 1)

#Eliminamos todos los pH porque la mayoría estan vacíos
data_df = data_df.drop('ph', axis=1)

#DF:
data_df
```

2. CASO CLÍNICO. EXTRACCIÓN, **TRANSFORMACIÓN** Y DESCRIPCIÓN

MANIPULACIÓN DE LA BBDD

- **Estadística descriptiva** de las variables numéricas:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 634 entries, 20 to 351315
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype         
---  ------                --------------  -----         
 0   subject_id            634 non-null    Int64         
 1   hadm_id               634 non-null    Int64         
 2   stay_id               634 non-null    Int64         
 3   ce_charttime          634 non-null    datetime64[ns]
 4   SEX                   634 non-null    object        
 5   crea                  582 non-null    float64       
 6   ck                    465 non-null    float64       
 7   potassium             589 non-null    float64       
 8   rhabdomiolisis        634 non-null    object        
 9   chronickidneydisease  634 non-null    object        
 10  diab_code             634 non-null    object        
 11  insuf_renal           634 non-null    object        
dtypes: Int64(3), datetime64[ns](1), float64(3), object(5)
memory usage: 66.2+ KB
```

	subject_id	hadm_id	stay_id	crea	ck	potassium
count	6.340000e+02	6.340000e+02	6.340000e+02	582.000000	465.000000	589.000000
mean	1.484377e+07	2.514375e+07	3.496694e+07	2.306529	14162.294624	4.262649
std	2.871501e+06	2.937797e+06	2.897955e+06	2.213608	31625.986994	0.989003
min	1.001414e+07	2.000136e+07	3.008916e+07	0.300000	28.000000	1.900000
25%	1.223740e+07	2.272712e+07	3.267639e+07	0.900000	1589.000000	3.700000
50%	1.486092e+07	2.531087e+07	3.463456e+07	1.600000	4486.000000	4.100000
75%	1.739961e+07	2.761291e+07	3.763851e+07	2.775000	12062.000000	4.600000
max	1.999784e+07	2.996045e+07	3.996048e+07	22.700000	285260.000000	10.600000

2. CASO CLÍNICO. EXTRACCIÓN, **TRANSFORMACIÓN** Y DESCRIPCIÓN

MANIPULACIÓN DE LA BBDD

- Revisar **coherencia** de los datos:

```
#Agrupamos por pacientes contamos el número de muestras que tiene de cada variable
a=data_df.groupby('subject_id')
a.count()
```

subject_id	hadm_id	stay_id	ce_charttime	SEX	crea	ck	potassium	rhabdomiolisis	chronickidneydisease	diab_code	insuf_renal
10014136	1	1	1	1	1	1	1	1	1	1	1
10021487	1	1	1	1	1	1	1	1	1	1	1
10027704	1	1	1	1	1	0	1	1	1	1	1
10029874	1	1	1	1	1	1	1	1	1	1	1
10076506	1	1	1	1	1	1	1	1	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...
19882264	1	1	1	1	1	1	1	1	1	1	1
19920096	1	1	1	1	1	1	1	1	1	1	1
19957847	1	1	1	1	1	1	1	1	1	1	1
19970892	1	1	1	1	1	1	0	1	1	1	1
19997843	1	1	1	1	1	1	1	1	1	1	1

539 rows × 11 columns

```
data_df.groupby('hadm_id').count() #Repetimos para las admisiones
```

hadm_id	subject_id	stay_id	ce_charttime	SEX	crea	ck	potassium	rhabdomiolisis	chronickidneydisease	diab_code	insuf_renal
20001361	1	1	1	1	1	0	1	1	1	1	1
20024997	1	1	1	1	1	1	1	1	1	1	1
20044149	2	2	2	2	2	0	2	2	2	2	2
20053897	1	1	1	1	0	1	1	1	1	1	1
20059818	1	1	1	1	1	1	1	1	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...
29927537	1	1	1	1	1	1	1	1	1	1	1
29941974	1	1	1	1	1	1	1	1	1	1	1
29942574	2	2	2	2	1	1	1	2	2	2	2
29949595	1	1	1	1	1	1	1	1	1	1	1
29960453	1	1	1	1	1	1	1	1	1	1	1

541 rows × 11 columns

2. CASO CLÍNICO. EXTRACCIÓN, **TRANSFORMACIÓN** Y DESCRIPCIÓN

MANIPULACIÓN DE LA BBDD

- Binarización de los datos:

```
#Creamos una copia:
data_dup_na_coerced= data_df.copy()

#Binarizamos las variables:

#Si el paciente no tiene el diagnóstico o es NA = 0, si tiene cualquier otro ID =1
data_dup_na_coerced['diab_code']=data_dup_na_coerced['diab_code'].apply(lambda y: 0 if y == '0' else 1)
data_dup_na_coerced['insuf_renal'] = data_dup_na_coerced['insuf_renal'].apply(lambda y: 0 if y == '0' else pd.NA if pd.isna(y) el
data_dup_na_coerced['chronickidneydisease'] = data_dup_na_coerced['chronickidneydisease'].apply(lambda y: 0 if y == '0' else pd.N
data_dup_na_coerced['rhabdomiolisis'] = data_dup_na_coerced['rhabdomiolisis'].apply(lambda y: 0 if y == '0' else pd.NA if pd.isna

#Si es hombre "Male" = 0, si es mujer =1:
data_dup_na_coerced['SEX'] = data_dup_na_coerced['SEX'].apply(lambda y: 0 if y == 'M' else pd.NA if pd.isna(y) else 1)
```

2. CASO CLÍNICO. EXTRACCIÓN, **TRANSFORMACIÓN** Y DESCRIPCIÓN

MANIPULACIÓN DE LA BBDD

- Adecuar el tipo de datos

```
subject_id                 Int64
hadm_id                    Int64
stay_id                    Int64
ce_charttime       datetime64[ns]
SEX                        object
crea                      float64
ck                        float64
potassium                 float64
rhabdomiolisis             object
chronickidneydisease       object
diab_code                  object
insuf_renal                object
dtype: object
```

```
subject_id                 Int64
hadm_id                    Int64
stay_id                    Int64
ce_charttime       datetime64[ns]
SEX                        int64
crea                      float64
ck                        float64
potassium                 float64
rhabdomiolisis             int64
chronickidneydisease       int64
diab_code                  int64
insuf_renal                int64
dtype: object
```

2. CASO CLÍNICO. EXTRACCIÓN, **TRANSFORMACIÓN** Y DESCRIPCIÓN

MANIPULACIÓN DE LA BBDD

- Definir variables dependientes (X) y la variable independiente (Y): Y es la columna de la insuficiencia renal aguda, que es lo que queremos predecir, X es el resto

```
: # Definir las variables dependientes e independientes
#La etiqueta que queremos predecir
y = data_dup_na_coerced['insuf_renal']
#Cogemos todas menos la columna de insuficiencia renal aguda
X = data_dup_na_coerced.loc[:,data_dup_na_coerced.columns != 'insuf_renal']
```

- Imputamos los missing values con la media.

```
#DATA INPUTATION
#print(y.isna().sum())
print(X.isna().sum())
X['crea'].fillna(X['crea'].mean(), inplace=True)
X['ck'].fillna(X['ck'].mean(), inplace=True)
X['potassium'].fillna(X['potassium'].mean(), inplace=True)
```

2. CASO CLÍNICO. EXTRACCIÓN, TRANSFORMACIÓN Y **DESCRIPCIÓN**

HERRAMIENTAS PARA LA DESCRIPCIÓN

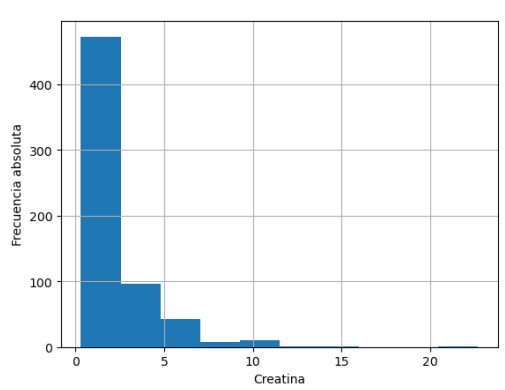
- Después del preprocesado nos quedamos con 632 pacientes.

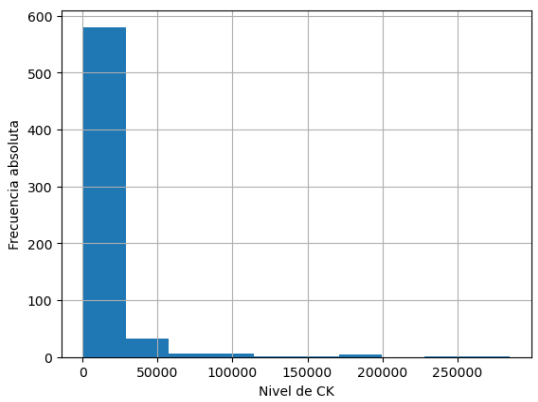
```
: data_dup_na_coerced.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 634 entries, 20 to 351315
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   subject_id            634 non-null    Int64
 1   hadm_id               634 non-null    Int64
 2   stay_id               634 non-null    Int64
 3   ce_charttime          634 non-null    datetime64[ns]
 4   SEX                   634 non-null    int64
 5   crea                  582 non-null    float64
 6   ck                    465 non-null    float64
 7   potassium             589 non-null    float64
 8   rhabdomiolisis        634 non-null    int64
 9   chronickidneydisease  634 non-null    int64
 10  diab_code             634 non-null    int64
 11  insuf_renal           634 non-null    int64
dtypes: Int64(3), datetime64[ns](1), float64(3), int64(5)
memory usage: 66.2 KB
```

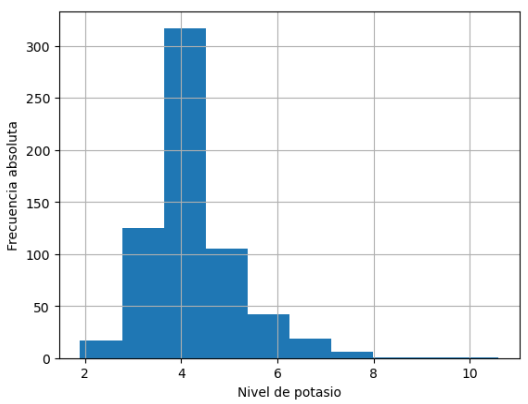
2. CASO CLÍNICO. EXTRACCIÓN, TRANSFORMACIÓN Y **DESCRIPCIÓN**

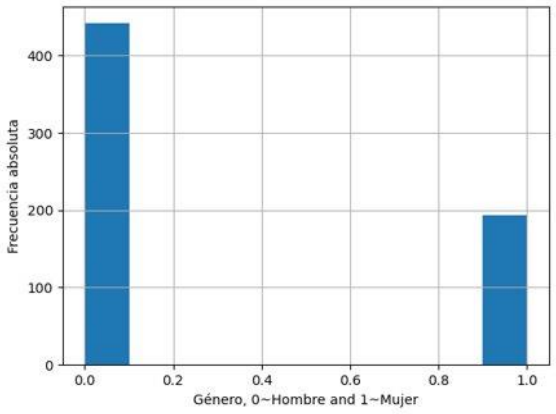
HERRAMIENTAS PARA LA DESCRIPCIÓN

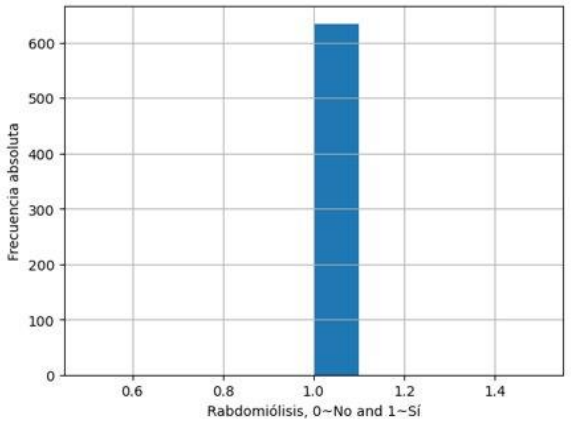
- Histogramas de frecuencias absolutas de las variables independientes X .

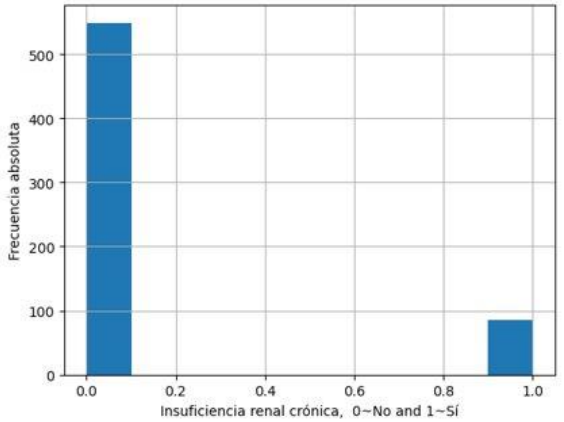








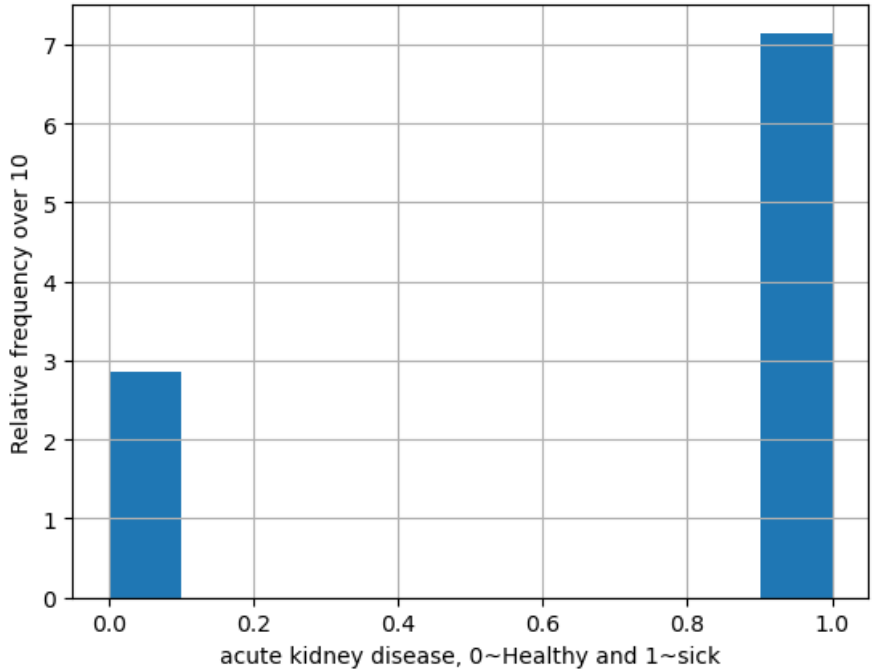




2. CASO CLÍNICO. EXTRACCIÓN, TRANSFORMACIÓN Y **DESCRIPCIÓN**

HERRAMIENTAS PARA LA DESCRIPCIÓN

- Evaluamos el histograma de frecuencia de Y



2. CASO CLÍNICO. MODELIZACIÓN

- Dividimos los datos en conjunto de entrenamiento (70%) y de prueba (30%).
- Elegimos los modelos:

APRENDIZAJE AUTOMÁTICO CLÁSICO

Supervisado de Clasificación:

- Logistic Regression
- Support Vector Machine

```
#Modelos de clasificación
models = dict([("Random Forest", RandomForestClassifier()),
          ('Logistic Regression', LogisticRegression(max_iter=10000)),
          ('Support Vector Machine', SVC()),
          ('XGBoost', XGBClassifier()),
          ('LightGBM', LGBMClassifier())])
```

APRENDIZAJE AUTOMÁTICO MODERNO

Métodos de Ensamble:

Empaquetamiento
-Random Forest

Impulso:
-LightGBM
-XGBoost

2. CASO CLÍNICO. MODELIZACIÓN

- Evaluación de los modelos por validación cruzada con los datos con y sin escalar.

```
# Evaluar los modelos utilizando validación cruzada
for name, model in models.items():
    print(f'{name}:')
    # Volidación cruzada
    scores = cross_val_score(model, X_train, y_train, cv=7, scoring = 'balanced_accuracy')
    print(f'Mean cross-validation score: {scores.mean():.3f}')
    print(f'Standard deviation: {scores.std():.3f}')

    #Ajuste con train:
    model.fit(X_train, y_train)

    #Predicción con test:
    y_pred = model.predict(X_test)

    #Resultados
    accuracy = accuracy_score(y_test, y_pred)#Exactitud
    precision = precision_score(y_test, y_pred, average='weighted')#Precisión
    recall = recall_score(y_test, y_pred, average='weighted')

    print(f'Accuracy score on test data: {accuracy:.3f}')
    print(f'Precision score on test data: {precision:.3f}')
    print(f'Recall score on test data: {recall:.3f}\n')
```

2. CASO CLÍNICO. MODELIZACIÓN

- Resultados sin escalar los datos:

```
Random Forest:
Mean cross-validation score: 0.816
Standard deviation: 0.055
Accuracy score on test data: 0.874
Precision score on test data: 0.885
Recall score on test data: 0.874

Logistic Regression:
Mean cross-validation score: 0.560
Standard deviation: 0.074
Accuracy score on test data: 0.785
Precision score on test data: 0.775
Recall score on test data: 0.785

Support Vector Machine:
Mean cross-validation score: 0.500
Standard deviation: 0.000
Accuracy score on test data: 0.707
Precision score on test data: 0.500
Recall score on test data: 0.707
```

```
XGBoost:
Mean cross-validation score: 0.820
Standard deviation: 0.030
Accuracy score on test data: 0.853
Precision score on test data: 0.853
Recall score on test data: 0.853

LightGBM:
Mean cross-validation score: 0.829
Standard deviation: 0.047
Accuracy score on test data: 0.843
Precision score on test data: 0.849
Recall score on test data: 0.843
```

2. CASO CLÍNICO. MODELIZACIÓN

- Resultados escalando los datos:

```
Random Forest:

Mean cross-validation score: 0.789
Standard deviation: 0.065
Accuracy score on test data: 0.859
Precision score on test data: 0.866
Recall score on test data: 0.859
```

```
Logistic Regression:
Mean cross-validation score: 0.739
Standard deviation: 0.066
Accuracy score on test data: 0.864
Precision score on test data: 0.862
Recall score on test data: 0.864
```

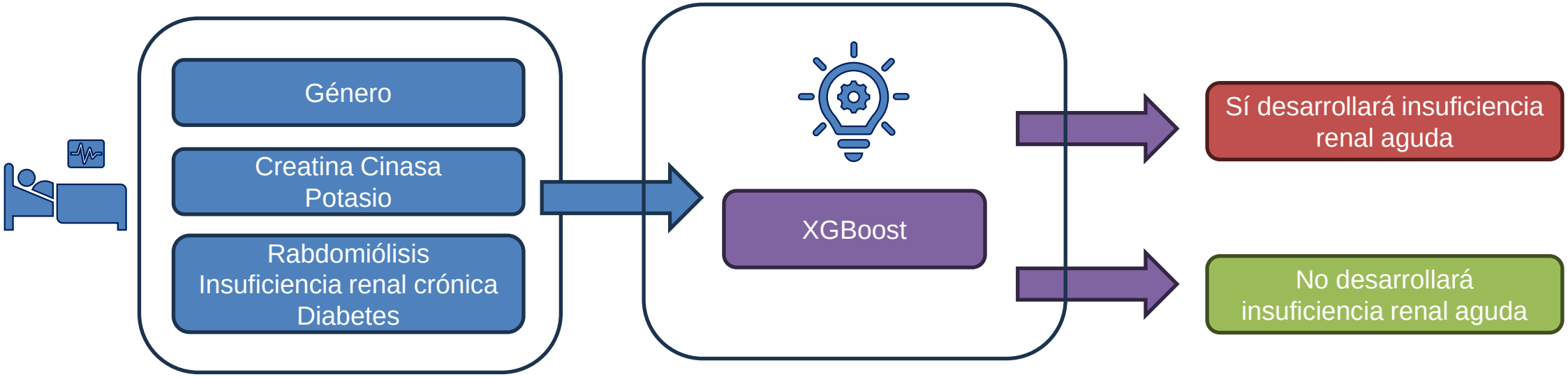
```
Support Vector Machine:
Mean cross-validation score: 0.771
Standard deviation: 0.053
Accuracy score on test data: 0.822
Precision score on test data: 0.834
Recall score on test data: 0.822
```

```
LightGBM:
Mean cross-validation score: 0.803
Standard deviation: 0.050
Accuracy score on test data: 0.827
Precision score on test data: 0.838
Recall score on test data: 0.827
```

```
XGBoost:
Mean cross-validation score: 0.814
Standard deviation: 0.079
Accuracy score on test data: 0.848
Precision score on test data: 0.853
Recall score on test data: 0.848
```

2. CASO CLÍNICO. MODELIZACIÓN

MODELO FINAL: XGBoost




```
Mean cross-validation score: 0.855
Standard deviation: 0.039
Accuracy score on test data: 0.848
Precision score on test data: 0.849
Recall score on test data: 0.848
```

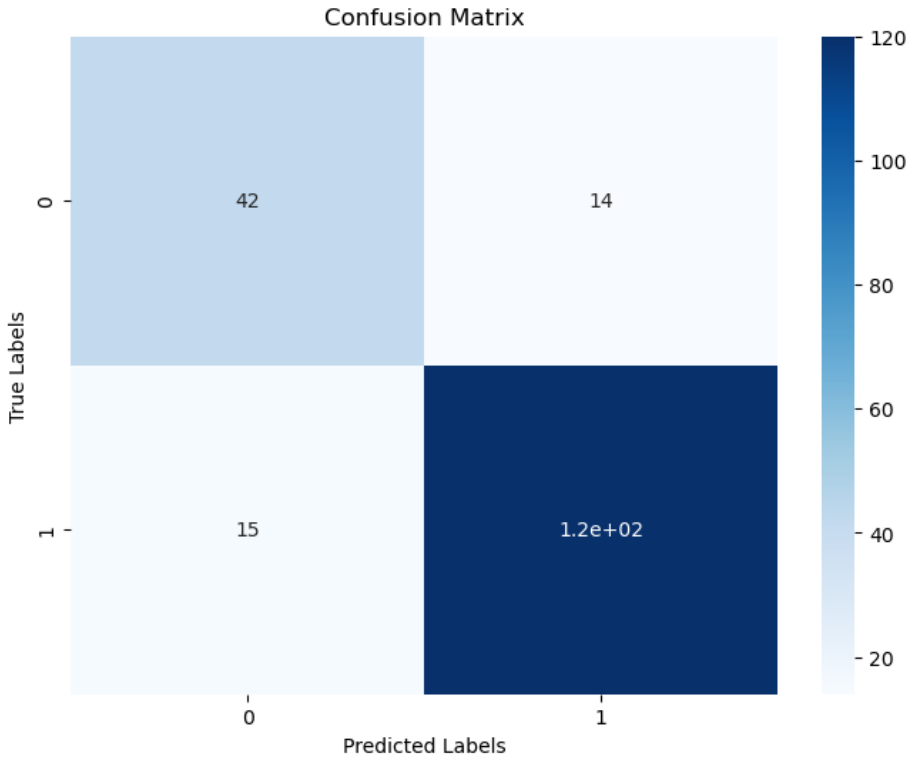
2. CASO CLÍNICO. MODELIZACIÓN

MODELO FINAL: XGBoost

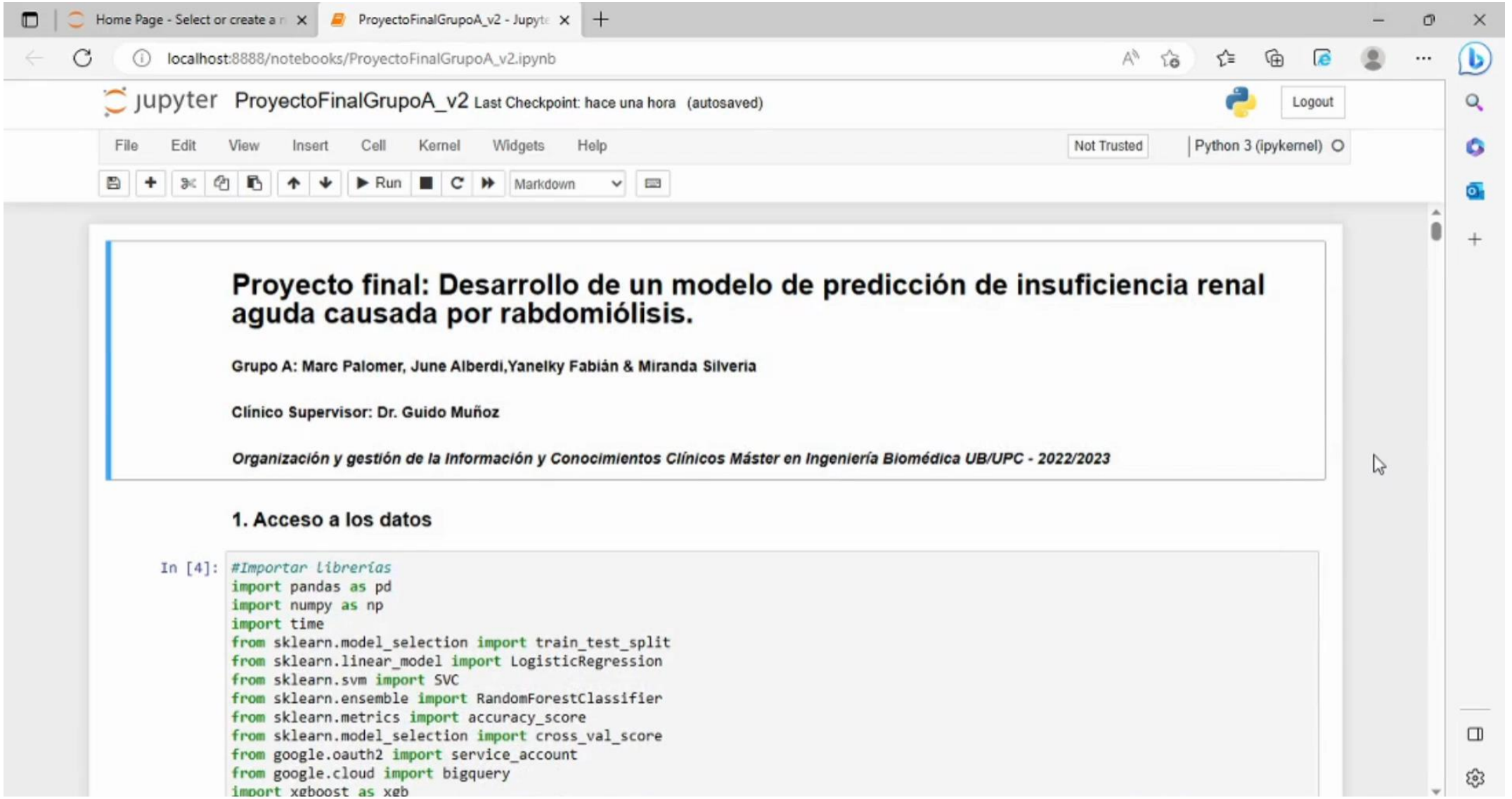
Pacientes clasificados correctamente: 162
- No desarrollan insuficiencia renal aguda (TN): 42
- Sí desarrollan insuficiencia renal aguda (TP): 120

Pacientes clasificados incorrectamente: 29
- Clasificados como no pero sí desarrollan insuficiencia renal aguda (FN): 14
- Clasificados como sí, pero no desarrollan insuficiencia renal aguda (FP): 15

```
Mean cross-validation score: 0.855
Standard deviation: 0.039
Accuracy score on test data: 0.848
Precision score on test data: 0.849
Recall score on test data: 0.848
```



3. NOTEBOOK


Home Page - Select or create a
ProyectoFinalGrupoA_v2 - Jupyte
localhost:8888/notebooks/ProyectoFinalGrupoA_v2.ipynb
jupyter ProyectoFinalGrupoA_v2 Last Checkpoint: hace una hora (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
Proyecto final: Desarrollo de un modelo de predicción de insuficiencia renal aguda causada por rabdomiólisis.
Grupo A: Marc Palomer, June Alberdi,Yanelky Fabián & Miranda Silveria
Clínico Supervisor: Dr. Guido Muñoz
Organización y gestión de la Información y Conocimientos Clínicos Máster en Ingeniería Biomédica UB/UPC - 2022/2023
1. Acceso a los datos
In [4]:
#Importar librerías
import pandas as pd
import numpy as np
import time
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from google.oauth2 import service_account
from google.cloud import bigquery
import xgboost as xgb

UNIVERSITAT POLITÈCNICA DE CATALUNYA – UNIVERSITAT DE BARCELONA

Máster en Ingeniería Biomédica

Muchas gracias

¿Alguna pregunta?