

PROYECTO DATA SCIENCE:

Encuesta Clientes Bancarios - Marketing para Depósito a Plazo



Motivación:



LOS DATOS ESTÁN RELACIONADOS CON CAMPAÑAS DE MARKETING DIRECTO DE UNA ENTIDAD BANCARIA PORTUGUESA. LAS CAMPAÑAS DE MARKETING SE BASARON EN LLAMADAS TELEFÓNICAS. A MENUDO, SE REQUERÍA MÁS DE UN CONTACTO CON EL MISMO CLIENTE, PARA PODER RECONOCER SI EL CLIENTE DESEA SUSCRIBIRSE AL PRODUCTO (DEPÓSITO BANCARIO A PLAZO).



EN EL PRESENTE PROYECTO SE EXPLORARÁ INFORMACIÓN RESPECTO A LA CAMPAÑA DE MARKETING EN EL DATASET "BANK-FULL.CSV" CON TODOS LOS EJEMPLOS Y 17 ENTRADAS, ORDENADAS POR FECHA.

ESTE DATASET ES MULTIVARIADO Y FUE DONADO EN: 2012-02-14



DEFINIR:

Tarea:

La tarea a realizar es de clasificación entre 2 clases posibles 'yes' , 'no'.

Variable Objetivo:

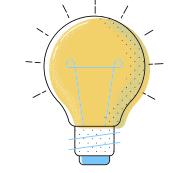
El conjunto de datos tiene información sobre los clientes y las campañas realizadas por el banco hacia el depósito a plazo. Hay una columna llamada 'y' en el conjunto de datos que indica si el cliente optó por un depósito a plazo o no.

Fuente de datos:

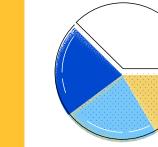
Los datos fueron recopilados de UCI
<https://archive.ics.uci.edu/ml/datasets/Bank+Marketing>



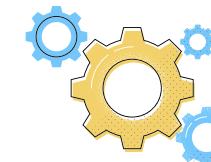
Metodología del proyecto



Entendimiento de
features



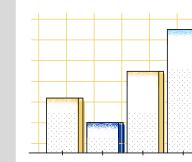
Exploración y análisis de datos
CORRELACION | GRÁFICOS |
DETECCIÓN DE VALORES ATÍPICOS |
FEATURE SELECTION



Pre-procesamiento de
de datos
RECORTE DE ATÍPICOS |
CODIFICACION DE COLUMNAS
CATEGÓRICAS |
NORMALIZACIÓN



Evaluación de modelos V1



BALANCE DE CLASES

UTILIZANDO SMOTEEN



Evaluación de modelos V2 y
Explicabilidad del modelo

Entendimiento de Features

Significado de columnas:

- Age : numérico
- Job : Tipo de trabajo (categórico)
- Marital : Estado civil
- Education : Nivel de educación del cliente
- default : ¿Tiene el crédito en default? (binario: "yes", "no")
- Balance : saldo medio anual, en euros (numérico)
- Housing : tiene préstamo de vivienda? (binario: "yes", "no")
- Loan : tiene préstamo personal? (binario: "yes", "no")
- Contact: tipo de comunicación del contacto (categórico: "desconocido", "teléfono", "celular")
- Day : último día de contacto del mes (numérico)
- Month : último mes de contacto del año (categóricos: "ene", "feb", "mar", ..., "nov", "dec")
- Duration : duración del último contacto, en segundos (numérico), importante
- Campaign: número de contactos realizados durante esta campaña y para este cliente (numérico, incluye último contacto)
- Pdays : número de días que pasaron después de que se contactó al cliente por última vez desde una campaña anterior (numérico, -1 significa que el cliente no fue contactado previamente)
- Previous : número de contactos realizados antes de esta campaña y para este cliente (numérico)
- Poutcome : resultado de la campaña de marketing anterior (categórico: "desconocido", "otro", "fracaso", "éxito")
- Y : ¿El cliente ha suscrito un depósito a plazo? (binario: yes, no) , TARGET



Análisis Exploratorio de datos

Exploración general

```
bank.isnull().sum()

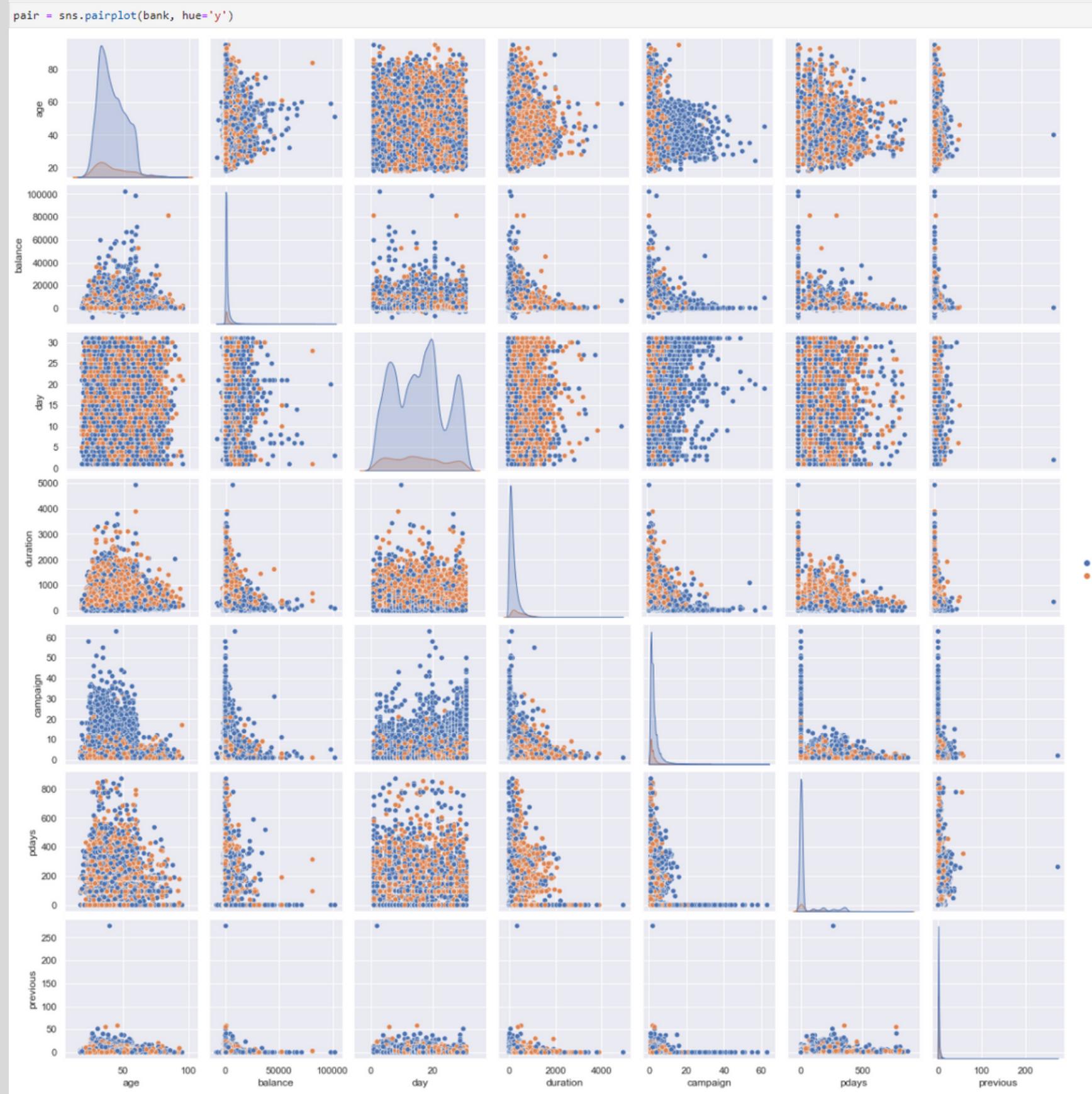
age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan          0
contact      0
day           0
month         0
duration     0
campaign     0
pdays         0
previous     0
poutcome     0
y              0
dtype: int64
```

Analizando las columnas numéricas por separado según el valor de 'Y'							
	age	balance	day	duration	campaign	pdays	previous
count	5289.000000	5289.000000	5289.000000	5289.000000	5289.000000	5289.000000	5289.000000
mean	41.670070	1804.267915	15.158253	537.294574	2.141047	68.702968	1.170354
std	13.497781	3501.104777	8.501875	392.525262	1.921826	118.822266	2.553272
min	18.000000	-3058.000000	1.000000	8.000000	1.000000	-1.000000	0.000000
25%	31.000000	210.000000	8.000000	244.000000	1.000000	-1.000000	0.000000
50%	38.000000	733.000000	15.000000	426.000000	2.000000	-1.000000	0.000000
75%	50.000000	2159.000000	22.000000	725.000000	3.000000	98.000000	1.000000
max	95.000000	81204.000000	31.000000	3881.000000	32.000000	854.000000	58.000000

bank[bank['y']=='no'].describe()							
	age	balance	day	duration	campaign	pdays	previous
count	39922.000000	39922.000000	39922.000000	39922.000000	39922.000000	39922.000000	39922.000000
mean	40.838986	1303.714969	15.892290	221.182806	2.846350	36.421372	0.502154
std	10.172662	2974.195473	8.294728	207.383237	3.212767	96.757135	2.256771
min	18.000000	-8019.000000	1.000000	0.000000	1.000000	-1.000000	0.000000
25%	33.000000	58.000000	8.000000	95.000000	1.000000	-1.000000	0.000000
50%	39.000000	417.000000	16.000000	164.000000	2.000000	-1.000000	0.000000
75%	48.000000	1345.000000	21.000000	279.000000	3.000000	-1.000000	0.000000
max	95.000000	102127.000000	31.000000	4918.000000	63.000000	871.000000	275.000000

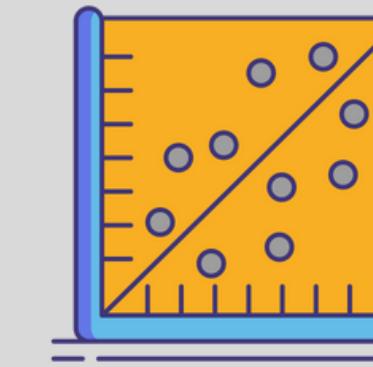
- No tenemos valores nulos en la data.
- Podemos notar que tenemos 5289 registros donde el valor de Y es yes, y 39922 registros donde el valor de Y es No, por lo tanto tenemos un desbalance de clases de aproximadamente: (Yes:12% vs No:88%).
- Podemos ver también que la duración es de 221 seg en promedio para la clase "NO" y de 537 seg en promedio para la clase "Yes".
- Asimismo, podemos ver que el número de contactos previos a la campaña máximo que se le hizo a algún cliente que aceptó es de 58 , mientras que para los que rechazaron es de 275.

Análisis Exploratorio de datos



Correlación entre variables

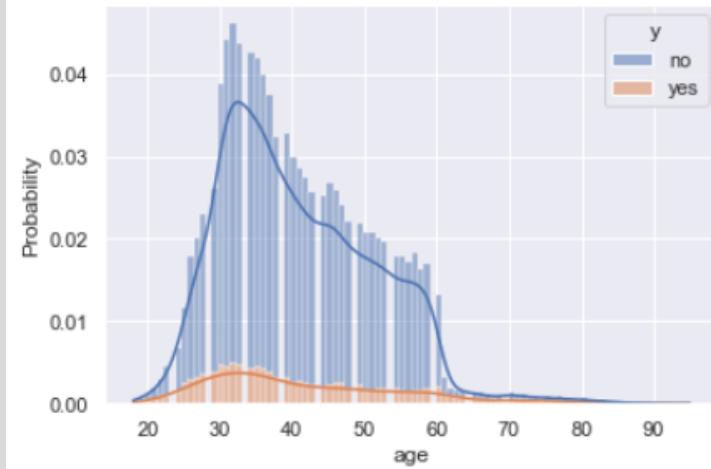
Al parecer las variables no tienen una correlación lineal significativa, esto nos impulsa a buscar otras formas de relación entre las características



Análisis Exploratorio de datos

Histogramas

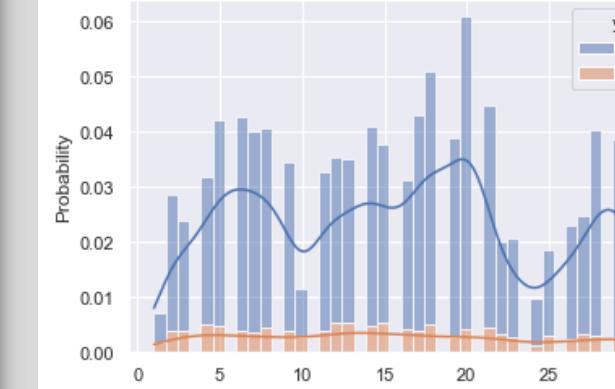
```
plot=sns.histplot(data=bank, x="age", hue="y", stat='probability', multiple="stack", kde=True)
```



Podemos ver que en el intervalo de 30 - 40 años se acumula la mayor probabilidad de que la persona acepte suscribirse a un depósito a plazo.

```
sns.histplot(data=bank, x="day", hue="y", stat='probability', multiple="stack", kde=True)
```

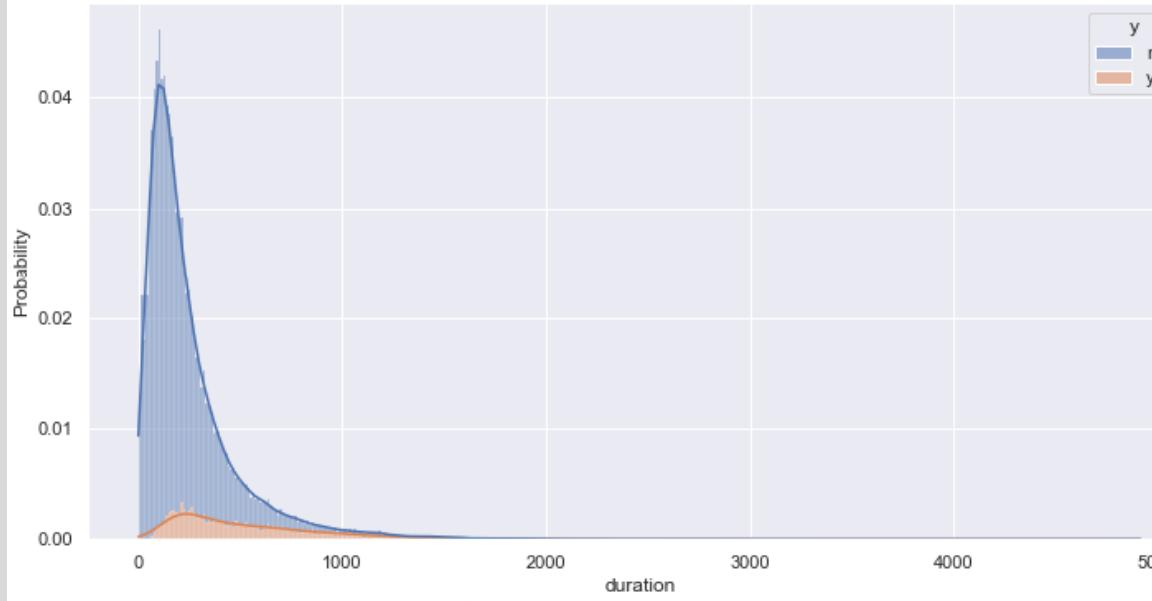
```
<AxesSubplot:xlabel='day', ylabel='Probability'>
```



Según este gráfico parece que la variable "day" tiene una distribución casi uniforme, pues la probabilidad en los días parece tener valores similares a excepción de los días 24 y 31 donde la probabilidad parece ser menor.

```
fig, ax = plt.subplots(figsize = (12, 6))
sns.histplot(data=bank, x="duration", hue="y", stat='probability', multiple="stack", kde=True)
```

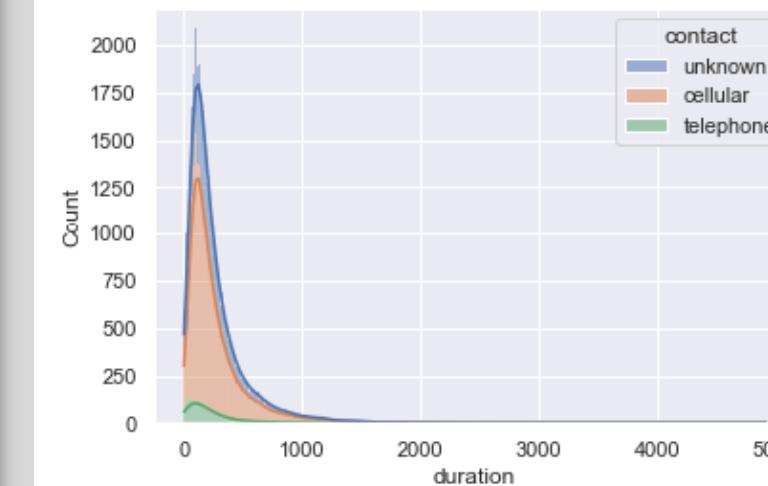
```
<AxesSubplot:xlabel='duration', ylabel='Probability'>
```



Podemos notar que la mayor probabilidad de que el cliente acepte el depósito a plazo se acumula en el intervalo de 0 a 1000 segundos de duración de contacto.

```
sns.histplot(data=bank, x="duration", hue="contact", multiple="stack", kde=True)
```

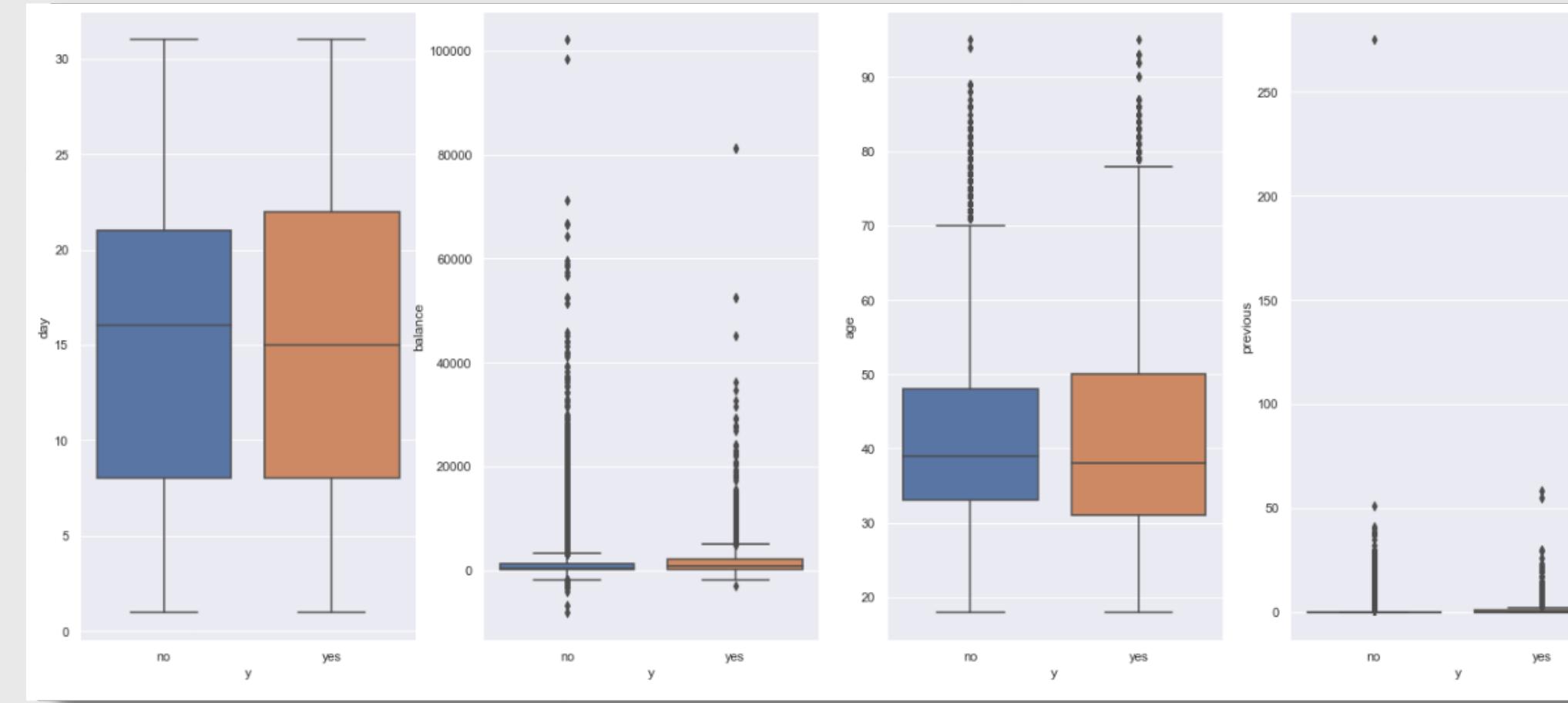
```
<AxesSubplot:xlabel='duration', ylabel='Count'>
```



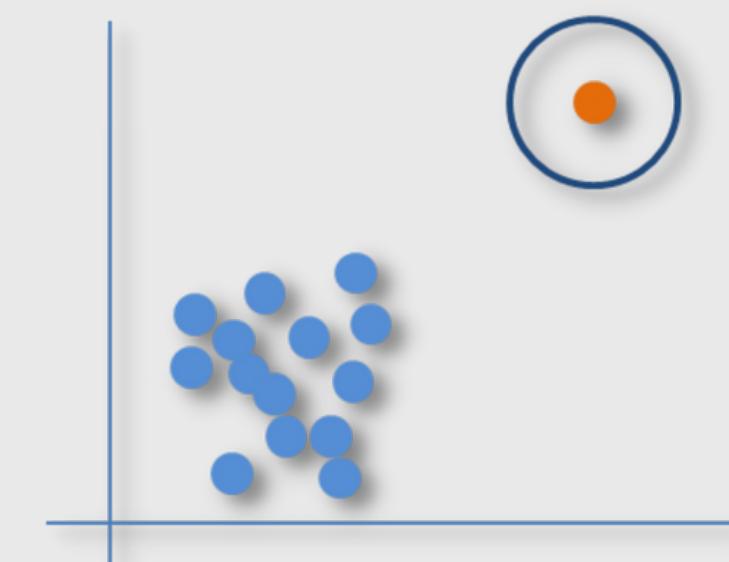
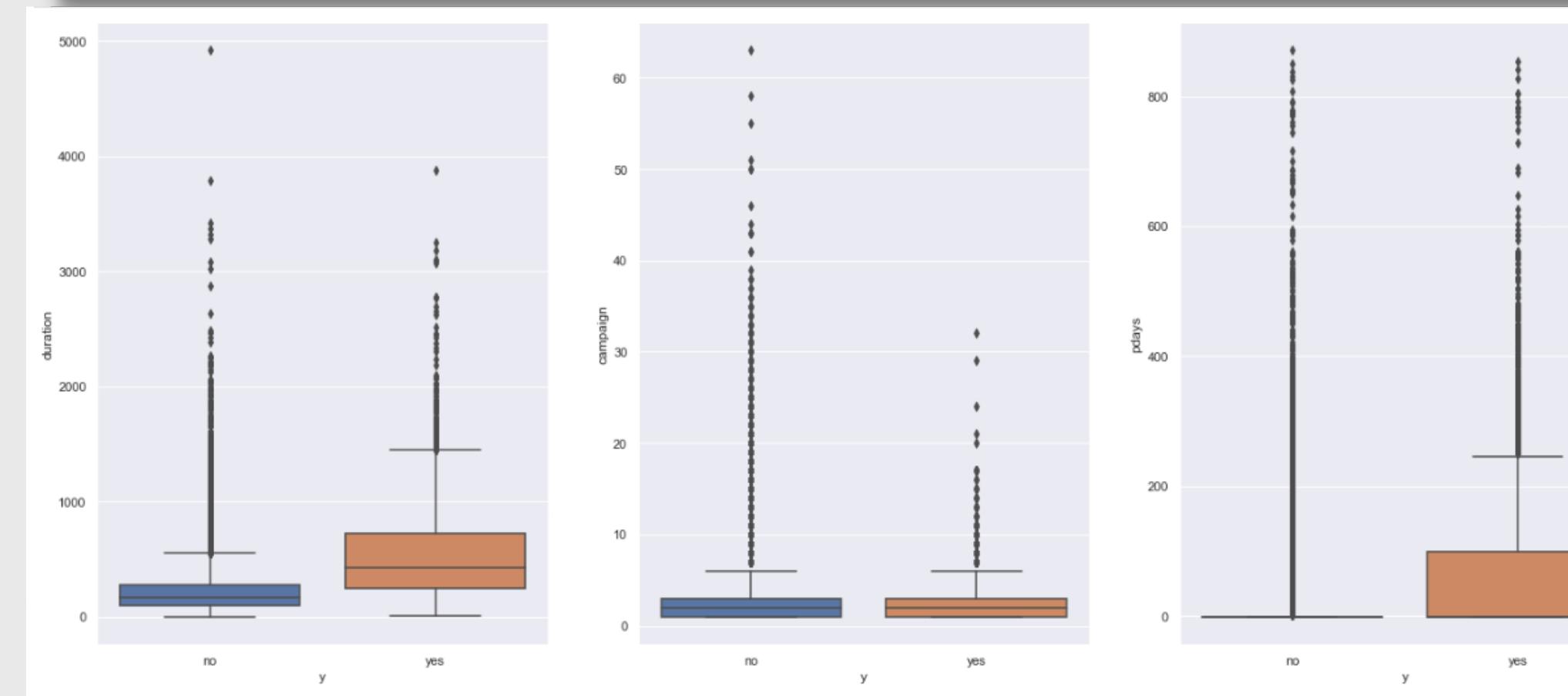
La mayor cantidad de contactos es de tipo desconocida, en segundo lugar tenemos al contacto mediante llamada por celular y en el último lugar al contacto por teléfono.

Análisis Exploratorio de datos

BOXPLOTS

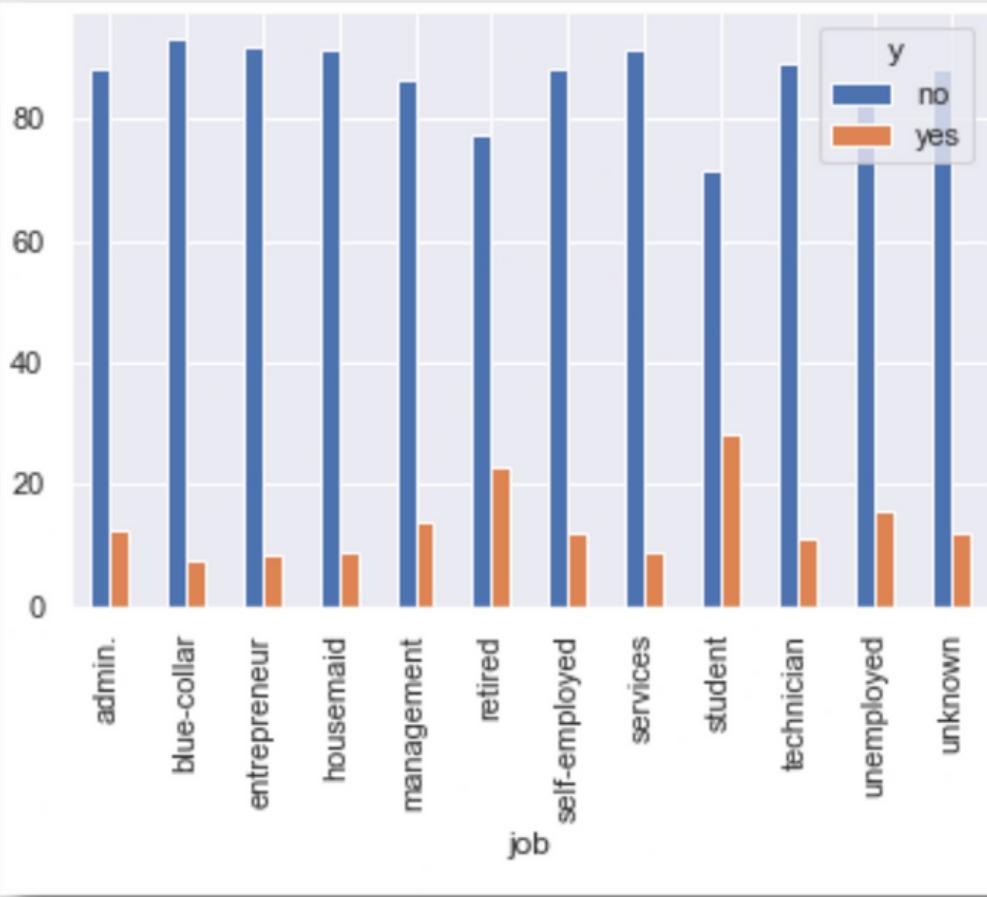


Podemos notar la presencia de Outliers en la data numérica , por ejemplo en las variables "duration", "balance", "previous", "campaign"

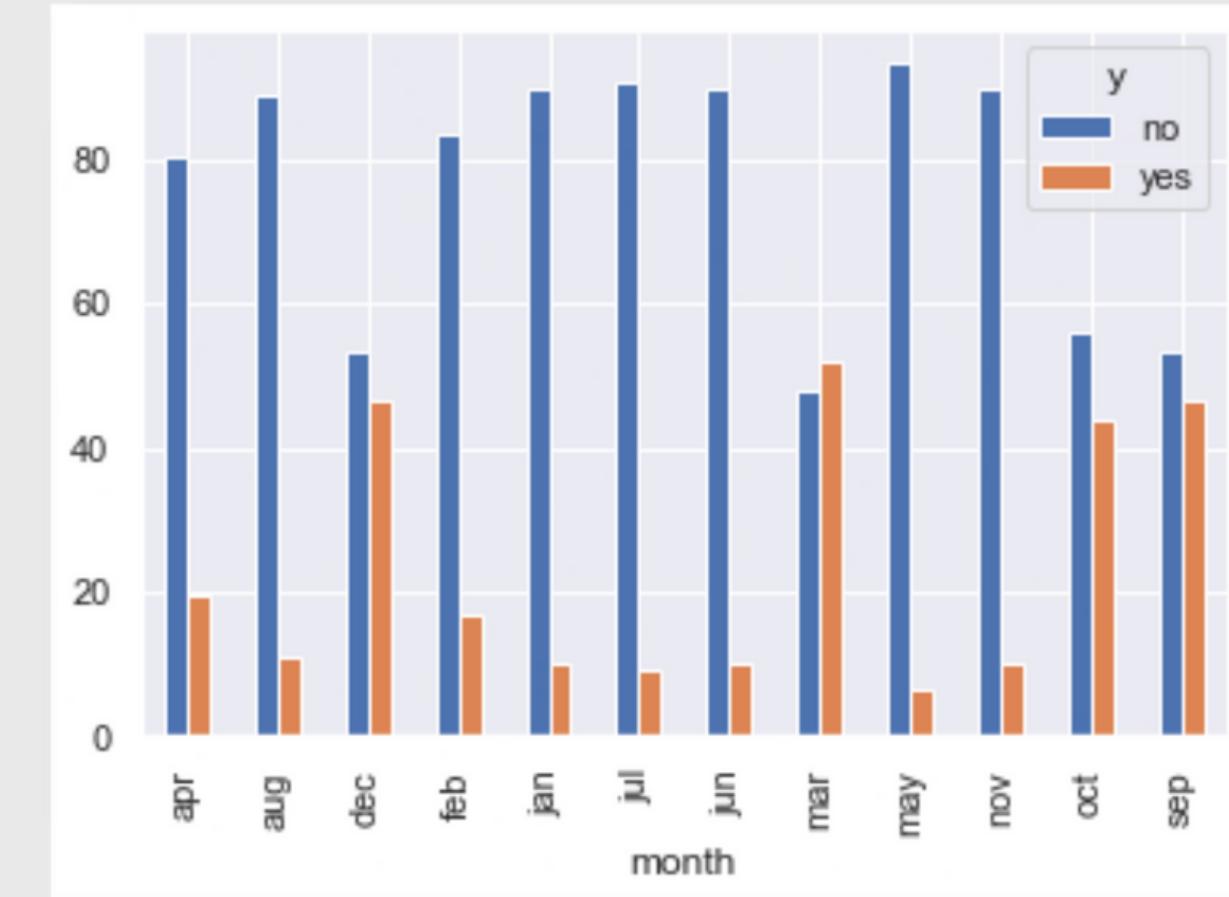


Análisis Exploratorio de datos

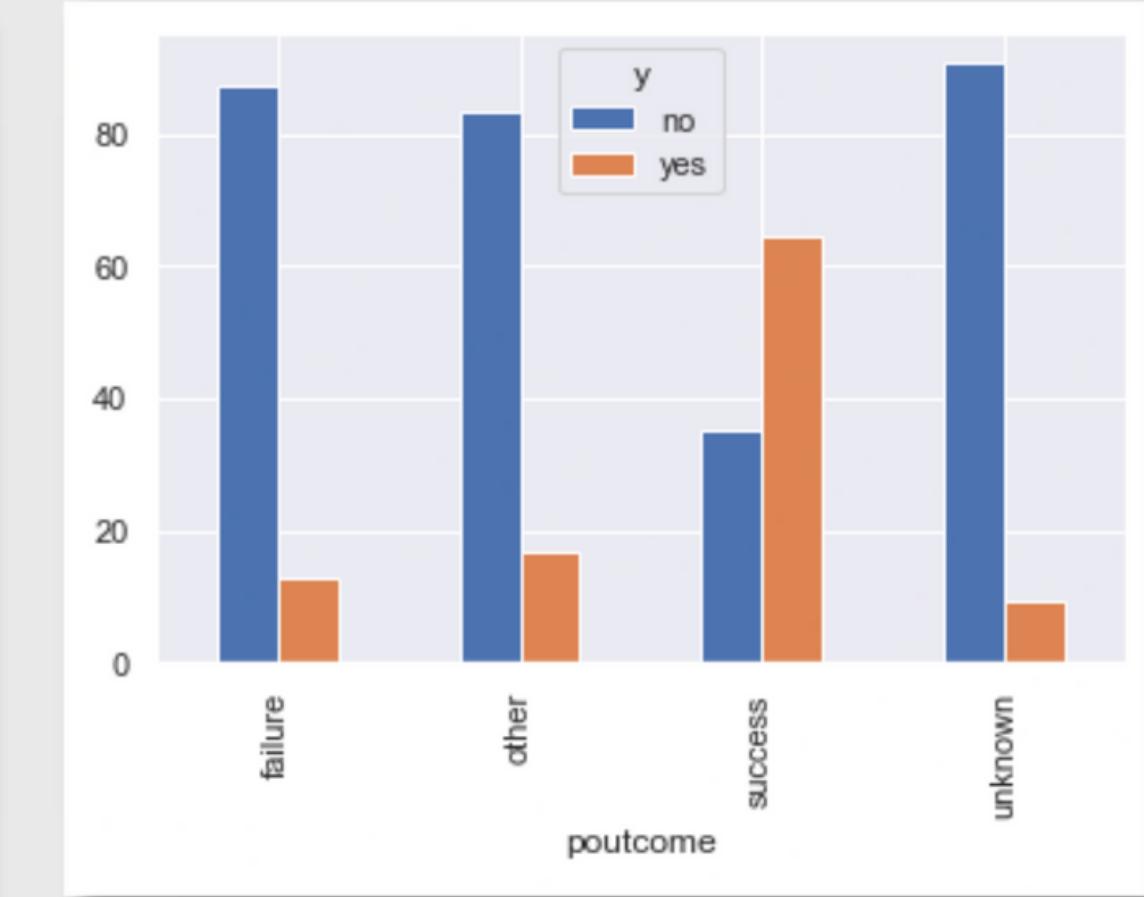
Gráficos de Barras



Los estudiantes son quienes en mayor cantidad aceptan el depósito a plazo y los "blue-collar" (obreros, agricultores, etc) quienes más rechazan este depósito.

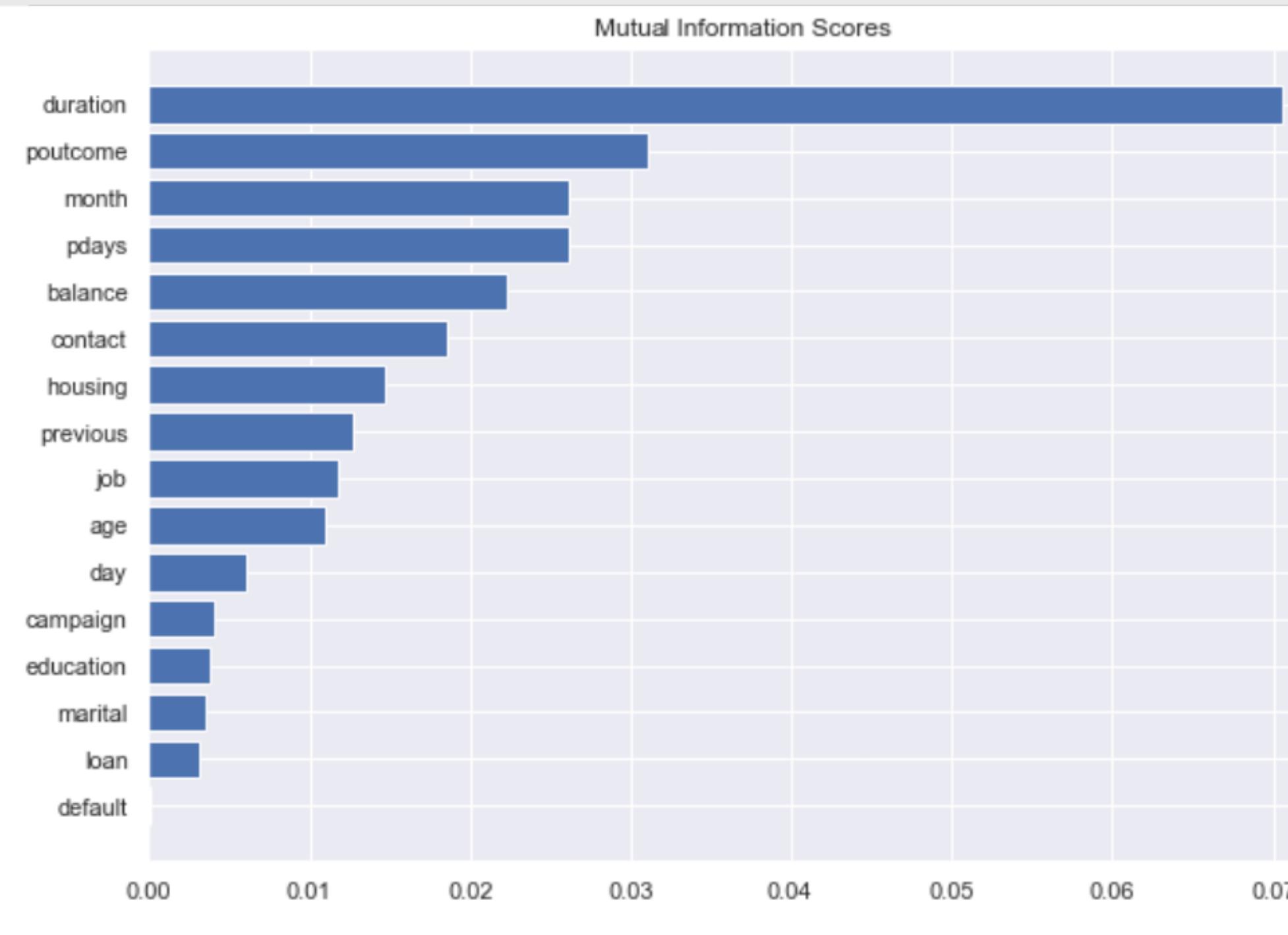


Podemos ver que el mes con mayor probabilidad de aceptación es marzo, seguido por septiembre, octubre y diciembre. Asimismo, el mes con mayor probabilidad de rechazo es mayo, seguido de noviembre, julio y junio.



Si el resultado de la campaña de marketing anterior fue "éxito" lo más probable es que en la actual la persona también acepte

Feature Selection: Mutual Information



¿Por qué usar "Mutual Information" para encontrar alguna relación entre la característica y la variable target?

Mutual Information puede predecir cualquier tipo de relación mientras que la correlación solo predice la relación lineal entre la característica y la variable target

Eligiré mantener las 10 primeras variables más significativas

```
: to_drop = ['default','loan','marital','education','campaign','day']
bank.drop(columns=to_drop, axis=1, inplace = True)
bank.head()
```

	age	job	balance	housing	contact	month	duration	pdays	previous	poutcome	y
0	58	management	2143	yes	unknown	may	261	-1	0	unknown	no
1	44	technician	29	yes	unknown	may	151	-1	0	unknown	no
2	33	entrepreneur	2	yes	unknown	may	76	-1	0	unknown	no
3	47	blue-collar	1506	yes	unknown	may	92	-1	0	unknown	no
4	33	unknown	1	no	unknown	may	198	-1	0	unknown	no

Pre-Procesamiento de datos

Codificando columnas de tipo Object:

```
for col in bank.select_dtypes(['object']):  
    bank[col],unique = bank[col].factorize(sort= True)
```

```
bank.head()
```

	age	job	balance	housing	contact	month	duration	pdays	previous	poutcome	y
0	58	4	2143	1	2	8	261	-1	0	3	0
1	44	9	29	1	2	8	151	-1	0	3	0
2	33	2	2	1	2	8	76	-1	0	3	0
3	47	1	1506	1	2	8	92	-1	0	3	0
4	33	11	1	0	2	8	198	-1	0	3	0

Pandas posee el método "factorize" que nos ayuda a asignarle un valor entero a cada clase que tengamos en las columnas categóricas.



Pre-Procesamiento de datos

Tratamiento de Outliers:

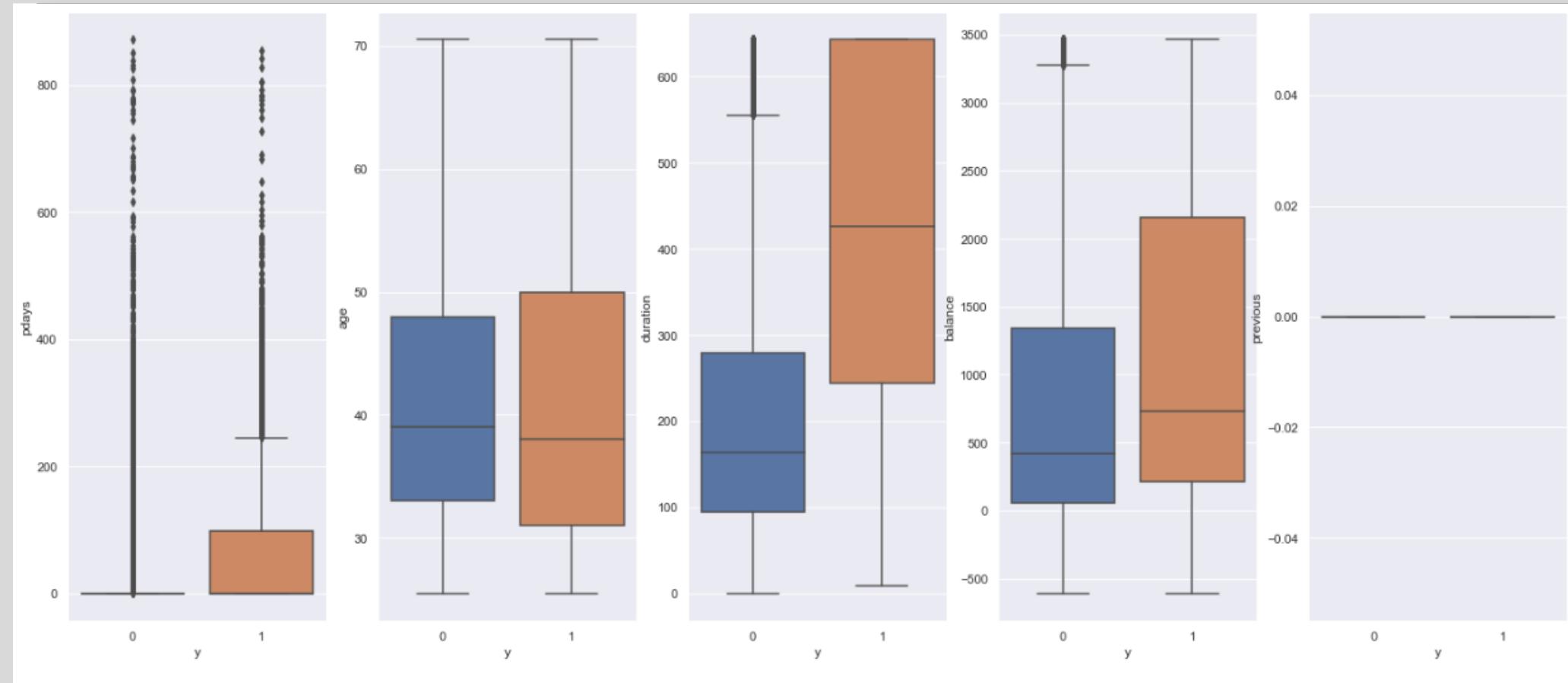
Realizré un recorte de los datos basandome en los límites superior e inferior de cada variable utilizando los cuartiles 1 , 3 y el rango intercuartílico.

```
# funcion para calcular el limite superior e inferior de las variables
def limites(columna):
    c = [.25,.75]
    q1,q3=columna.quantile(c)
    ## calculamos
    lim_inf = q3 - 1.5* (q3-q1)
    lim_sup = q3 + 1.5* (q3-q1)
    return(lim_inf ,lim_sup)
```

```
# Analizamos las variables numericas
a,b = limites(bank["age"])
print (' el limite inferior es:', a)
print (' el limite superior es:', b)

#recorte
bank.loc[bank["age"]<a,"age"] = a
bank.loc[bank["age"]>b,"age"] = b

el limite inferior es: 25.5
el limite superior es: 70.5
```



De este modo recortamos los valores atípicos utilizando los límites superiores e inferiores.



NumPy

Pre-Procesamiento de datos

Normalización de datos

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
prueba = scaler.fit_transform(prueba)
```

$$x_{\text{norm}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \in [0, 1]$$

	age	balance	duration	pdays	previous
0	58.0	2143	261	0	0
1	44.0	29	151	0	0
2	33.0	2	76	0	0
3	47.0	1506	92	0	0
4	33.0	1	198	0	0



	age	balance	duration	pdays	previous
0	0.722222	0.675762	0.405910	0.0	0.0
1	0.411111	0.156096	0.234837	0.0	0.0
2	0.166667	0.149459	0.118196	0.0	0.0
3	0.477778	0.519174	0.143079	0.0	0.0
4	0.166667	0.149213	0.307932	0.0	0.0



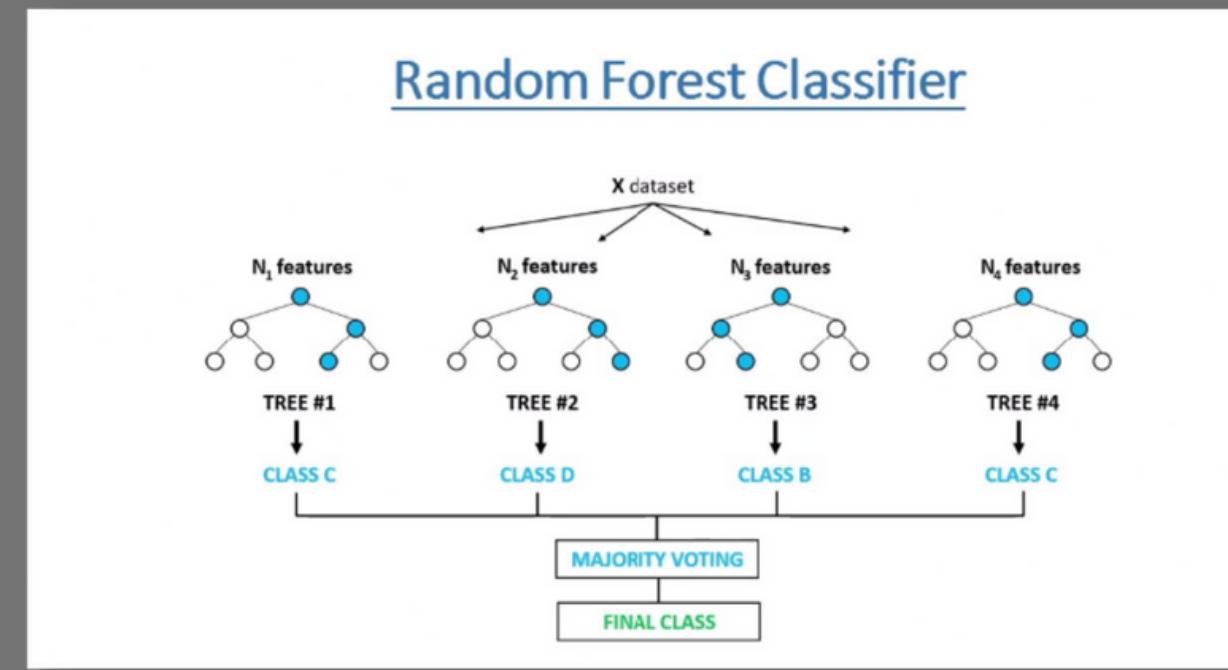
Evaluación de Modelos V1:

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
from sklearn.model_selection import StratifiedKFold
import xgboost as xgb
from xgboost import XGBClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.metrics import f1_score

X = df_bank.drop('y',axis=1)
y = df_bank['y']

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.4, stratify=y,random_state = 0)
```

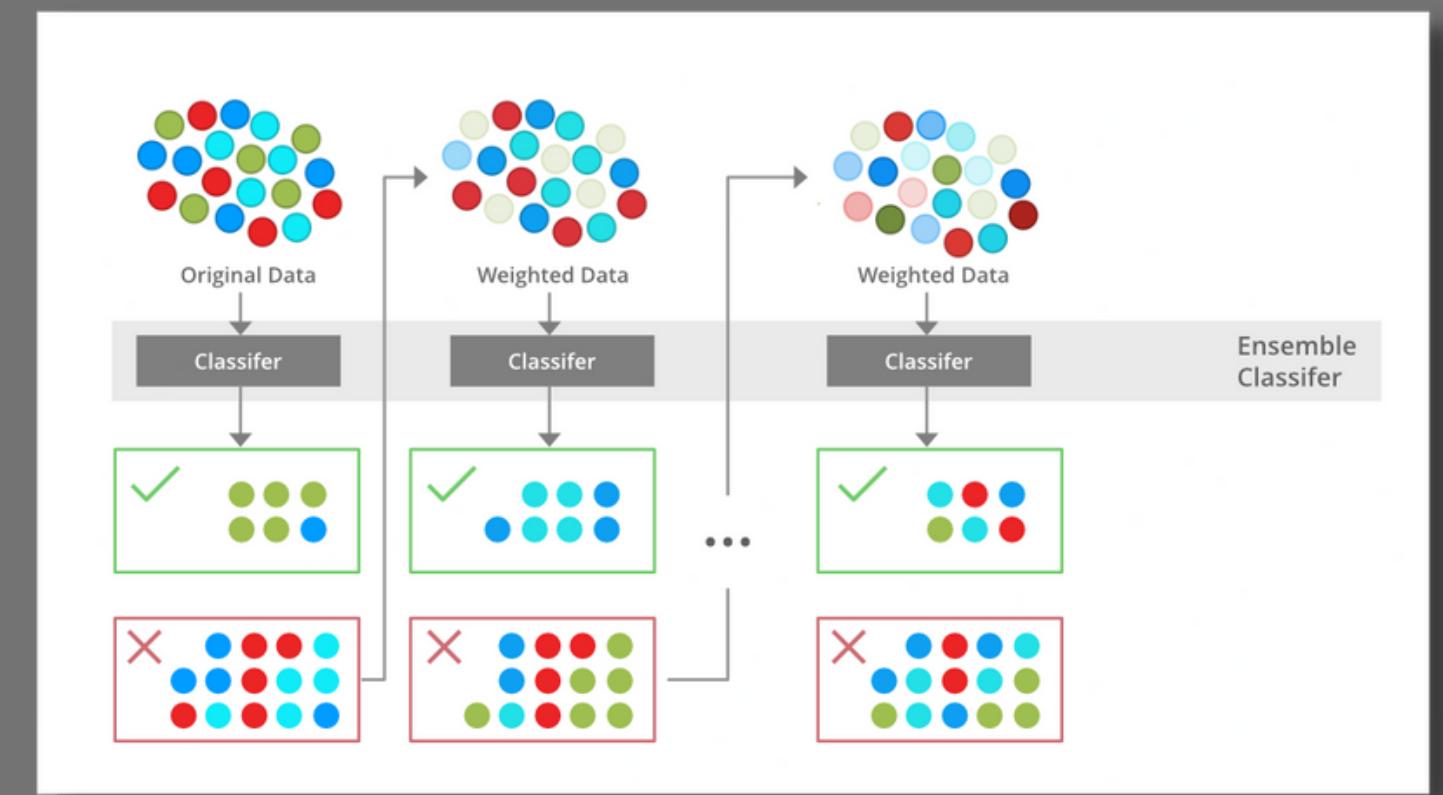
RANDOM FOREST CLASSIFIER



XGBOOST CLASSIFIER

```
def evaluar_modelo(model,xtrain,xtest,ytrain,ytest):

    print("En train")
    model.fit(xtrain, ytrain)
    predicted_train=model.predict(xtrain)
    print(confusion_matrix(ytrain, predicted_train))
    print(classification_report(ytrain,predicted_train))
    print("Ahora en test")
    predicted_test=model.predict(xtest)
    print(confusion_matrix(ytest, predicted_test))
    print(classification_report(ytest,predicted_test))
```

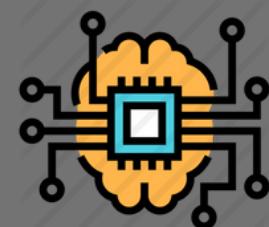


Evaluación de Modelos V1:

RANDOM FOREST CLASSIFIER



Ahora en test				
[[15403 566]				
[1226 890]]				
	precision	recall	f1-score	support
0	0.93	0.96	0.95	15969
1	0.61	0.42	0.50	2116
accuracy				
macro avg	0.77	0.69	0.72	18085
weighted avg	0.89	0.90	0.89	18085



XGB CLASSIFIER



Ahora en test				
[[15335 634]				
[1176 940]]				
	precision	recall	f1-score	support
0	0.93	0.96	0.94	15969
1	0.60	0.44	0.51	2116
accuracy				
macro avg	0.76	0.70	0.73	18085
weighted avg	0.89	0.90	0.89	18085

La métrica elegida para evaluar los modelos es F1-score, ya que esta representa la media armónica entre precision y recall:
F1-SCORE= $2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$

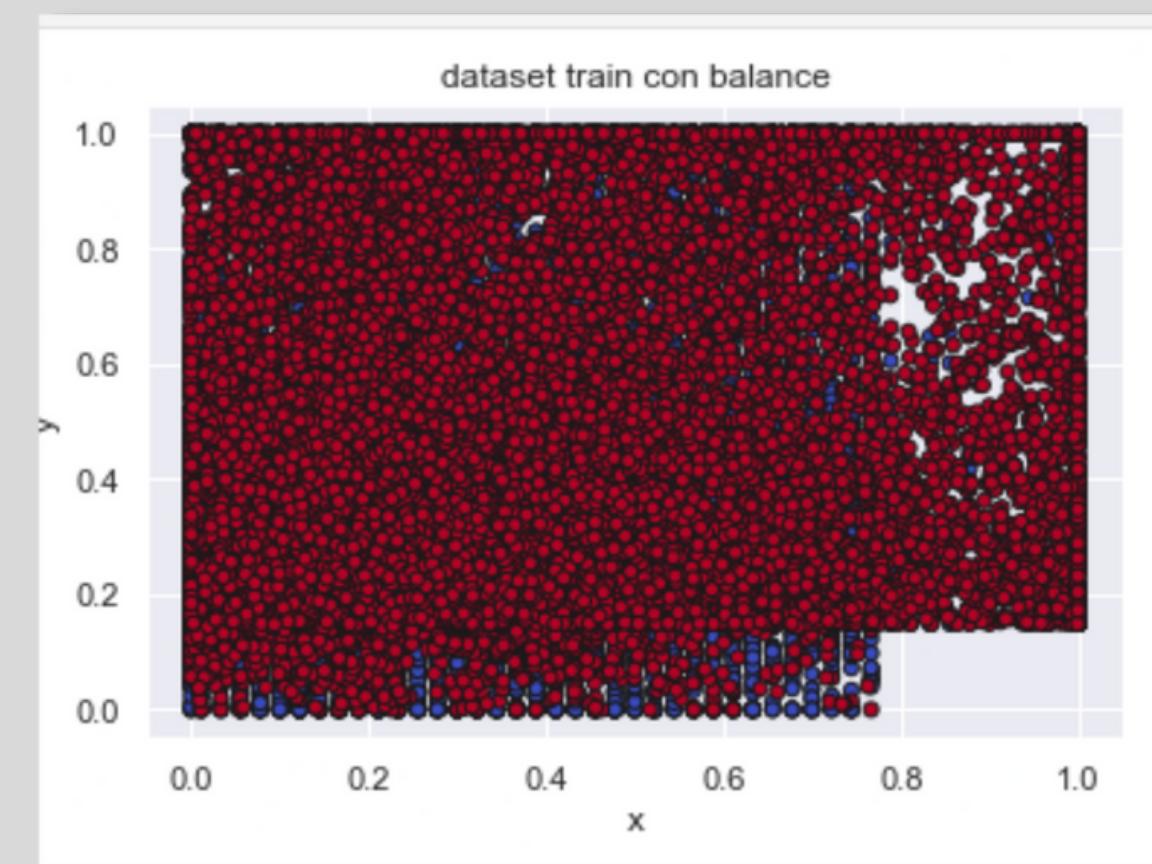
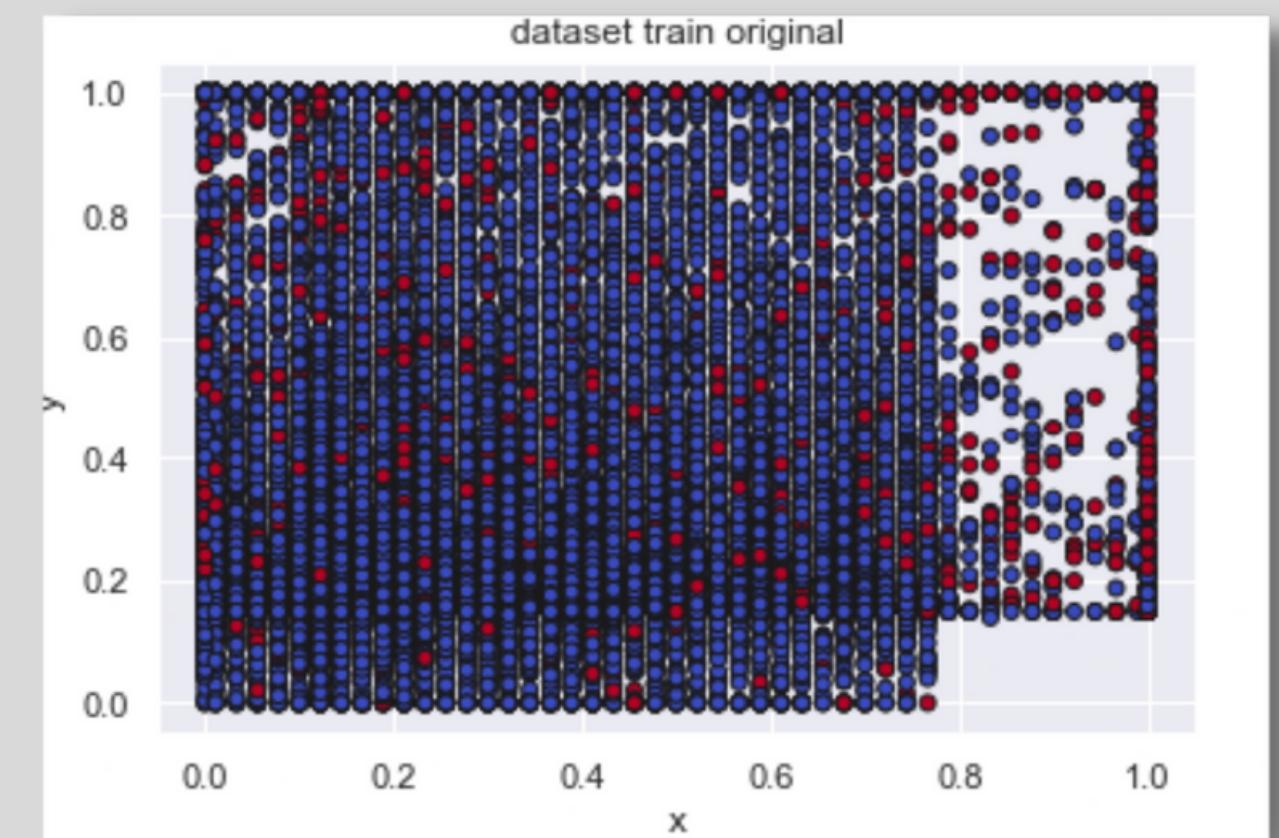
Quiero considerar la misma importancia para ambas y enfocarme en no tener muchos falsos positivos y falsos negativos, ya que estos generalmente tienen costos comerciales sean tangibles o intangibles. Además mi data es desbalanceada con un gran número de negativos reales.

Balance de clases:

```
SMT=SMOTEENN()
x_bal, y_bal= SMT.fit_resample(X_train,y_train)
from collections import Counter
print('Resampled dataset shape %s' % Counter(y_bal))

Resampled dataset shape Counter({1: 20870, 0: 19286})
```

SMOTEEN: este método combina la capacidad de SMOTE para generar ejemplos sintéticos para la clase minoritaria y la capacidad de ENN para eliminar algunas observaciones de ambas clases que se identifican como de clase diferente entre la clase de la observación y su clase mayoritaria vecina más cercana.



Evaluación de Modelos V2:

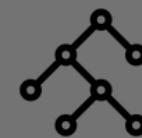
RANDOM FOREST CLASSIFIER



```
Ahora en test
[[13486  2483]
 [ 341 1775]]
      precision    recall  f1-score   support
          0       0.98      0.84      0.91     15969
          1       0.42      0.84      0.56      2116

   accuracy                           0.84      18085
  macro avg       0.70      0.84      0.73     18085
weighted avg       0.91      0.84      0.86     18085
```

XGB CLASSIFIER



```
Ahora en test
[[13985  1984]
 [ 434 1682]]
      precision    recall  f1-score   support
          0       0.97      0.88      0.92     15969
          1       0.46      0.79      0.58      2116

   accuracy                           0.87      18085
  macro avg       0.71      0.84      0.75     18085
weighted avg       0.91      0.87      0.88     18085
```

GRID SEARCH PARA XGBOOSTCLASSIFIER

```
parameters = {
    'min_child_weight': [1,3,5],
    'gamma': [0 ,0.05, 0.1,0.5],
    'max_depth': [4,5,6,8],
    'subsample' : [0.8],
    'learning_rate': [0.01,0.05,0.1,0.5],
    'n_estimators':[150,200,250],
}

model_xgb = xgb.XGBClassifier(objective = 'binary:logistic', seed=27)
xg_CV = GridSearchCV(model_xgb, parameters , n_jobs=4, cv=3,verbose=1)
xg_CV.fit(x_bal,y_bal)
```

```
xg_CV.best_estimator_
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
gamma=0.05, gpu_id=-1, importance_type=None,
interaction_constraints='', learning_rate=0.1, max_delta_step=0,
max_depth=8, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=200, n_jobs=16,
num_parallel_tree=1, predictor='auto', random_state=27,
reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=27,
subsample=0.8, tree_method='exact', validate_parameters=1,
verbosity=None)
```



```
Ahora en test
[[13938  2031]
 [ 407 1709]]
      precision    recall  f1-score   support
          0       0.97      0.87      0.92     15969
          1       0.46      0.81      0.58      2116

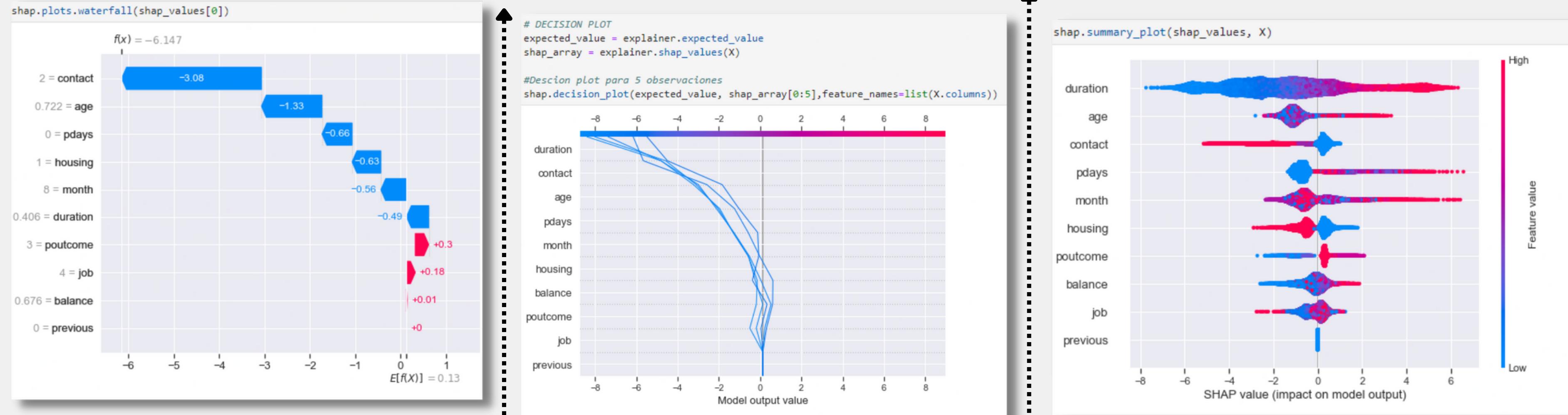
   accuracy                           0.87      18085
  macro avg       0.71      0.84      0.75     18085
weighted avg       0.91      0.87      0.88     18085
```

Observamos que disminuye la cantidad de False Negative detectados. También aumentó el recall para la clase I. Existe una pequeña mejora respecto a la versión anterior en cierto modo.

Explicabilidad del modelo:



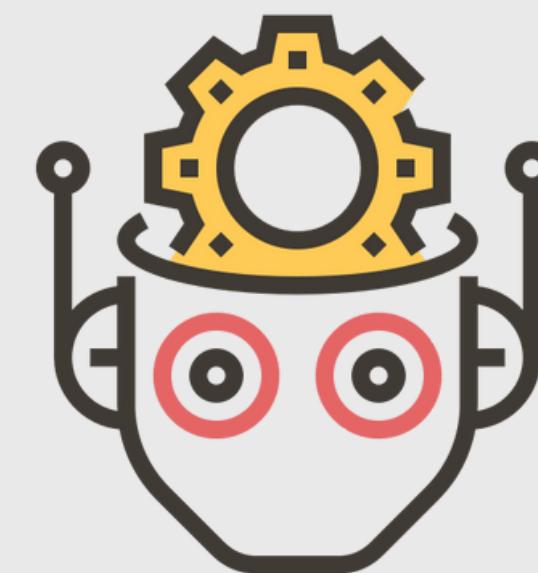
SHAP



- Este gráfico sirve para saber cuánto ha aportado cada feature a la predicción del modelo aumentando o reduciendo la probabilidad. Podemos ver que la variable contacto tiene el mayor impacto en el modelo y la variable que tiene menos impacto es previous.
- Podemos ver que variables como duración, contacto, edad disminuyen el valor de la predicción. Por otro lado, poutcome, job y balance aumentan el valor de la predicción.
- Observamos que los valores altos de contacto hacen que el output del modelo tenga un valor más bajo, a diferencia de la duración del contacto en donde al parecer el modelo asume que a valores más altos hay más probabilidad de que la clase sea 'I'.
- Notamos que previous no aporta al modelo.
- A menor valor de duración será menor el valor del output del modelo.
- El impacto de la edad generalmente es para reducir el valor output, puesto que se observa un gran cúmulo a la derecha del eje central.
- También se observa que para el modelo los valores de balance no aportan mucho al output, ya que sus valores se acumulan en el eje central en mayor proporción.

Conclusiones:

- El modelo con mejor desempeño fue XGBCLASSIFIER en la 2da evaluación. Se pudo conseguir un accuracy de 87% en test, reconociendo mejor a la clase '1' y valores F1-score de 0.92 y 0.58 para las clases 0 y 1 respectivamente.
- El recorte de datos atípicos en las columnas numéricas utilizando los cuartiles 1 y 3 para hallar el límite superior e inferior fue de bastante ayuda para el modelo.
- La normalización Min-Max Scaler fue punto clave en el pre-procesamiento, pues ayudó a mejorar el rendimiento en comparación con la utilización de la data sin escalar.
- Balancear la data ayudó a mejorar el desempeño de los modelos. Específicamente la utilización de SMOTEENN me resultó mejor que SMOTE.
- Se deben realizar pruebas nuevas quitando aquellas variables que no tuvieron un aporte grande en la decisión del modelo como: 'previous' y 'balance'.
- Se podría probar el uso de Redes neuronales con el dataset balanceado para evaluar si es mejor que los modelos vistos en el presente proyecto.



GRACIAS

Referencias:

- Wai On.(2020).Visualizing AI: Deconstructing and Optimizing the SHAP Summary Plot. Recuperado de: <https://towardsdatascience.com/visualizing-ai-8fad4ea70b87>
- Jain,A.(2016).Complete Guide to Parameter Tuning in XGBoost. Recuperado de: <https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/>
- Acevedo,N.(2020).Matriz de confusión en Machine Learning. Recuperado de: <https://nataliaacevedo.com/matriz-de-confusion-en-machine-learning-explicado-paso-a-paso/>
- Andhika, R (2021).Imbalanced Classification in Python: SMOTE-ENN Method. Recuperado de: <https://towardsdatascience.com/imbalanced-classification-in-python-smote-enn-method-db5db06b8d50>