

# Proyecto Final

# Data Science

**Introducción al Proyecto - Bank of Argentina**

Preparado por Nicolas Moses & Nicolas Mollo & Gabriel Mazzitelli

# Aspectos Clave

**Un breve vistazo a lo que debatiremos en este informe**

- Introducción
- Objetivos
- Data Set
- Contacto

# ¿Quienes somos?

## Data Specialist Team - Bank of Argentina

Nuestro proyecto se basa en realizar un análisis de la ultima campaña de marketing del banco. Nuestro DataSet contiene todos los datos de la campaña 2021 (variables categoricas y númericas) de nuestros clientes, y en el mismo se puede observar si aceptaron o no realizar una inversion (El target se encuentra en la columna "y")

## Inversiones de renta fija

Queremos aumentar las inversiones en renta fija a largo plazo que nuestros clientes realizan en el banco.

## Algoritmos de Machine Learning

Luego de realizar los analisis correspondientes, aplicaremos algoritmos de Machine Learning para lograr mejores predicciones en la campaña siguiente.



# Objetivos

Campaña 2022

## Predicciones mas eficientes.

AUMENTAR LAS INVERSIONES QUE REALIZAN NUESTROS CLIENTES

Con nuestro proyecto, buscamos mejorar el nivel de predicciones a la hora de armar la próxima campaña de venta. En nuestro dataset actual, el porcentaje de rechazos fue de 89% por lo que no es un número optimo para una campaña comercial. Vamos a orientar las técnicas utilizadas, para entender cuales son las variables que influyen a la hora de que un cliente acepte suscribir una inversión a largo plazo con nosotros.

**BANK OF ARGENTINA - DATA SPECIALIST TEAM**

# Información del Proyecto



En nuestro data set contamos con datos de clientes, datos de la campaña e indicadores económicos / sociales.

# DataSet

Campaña 2021

**41188**

LLAMADOS TELÉFONICOS

Cantidad de llamados en nuestra  
campaña 2021.

**4640**

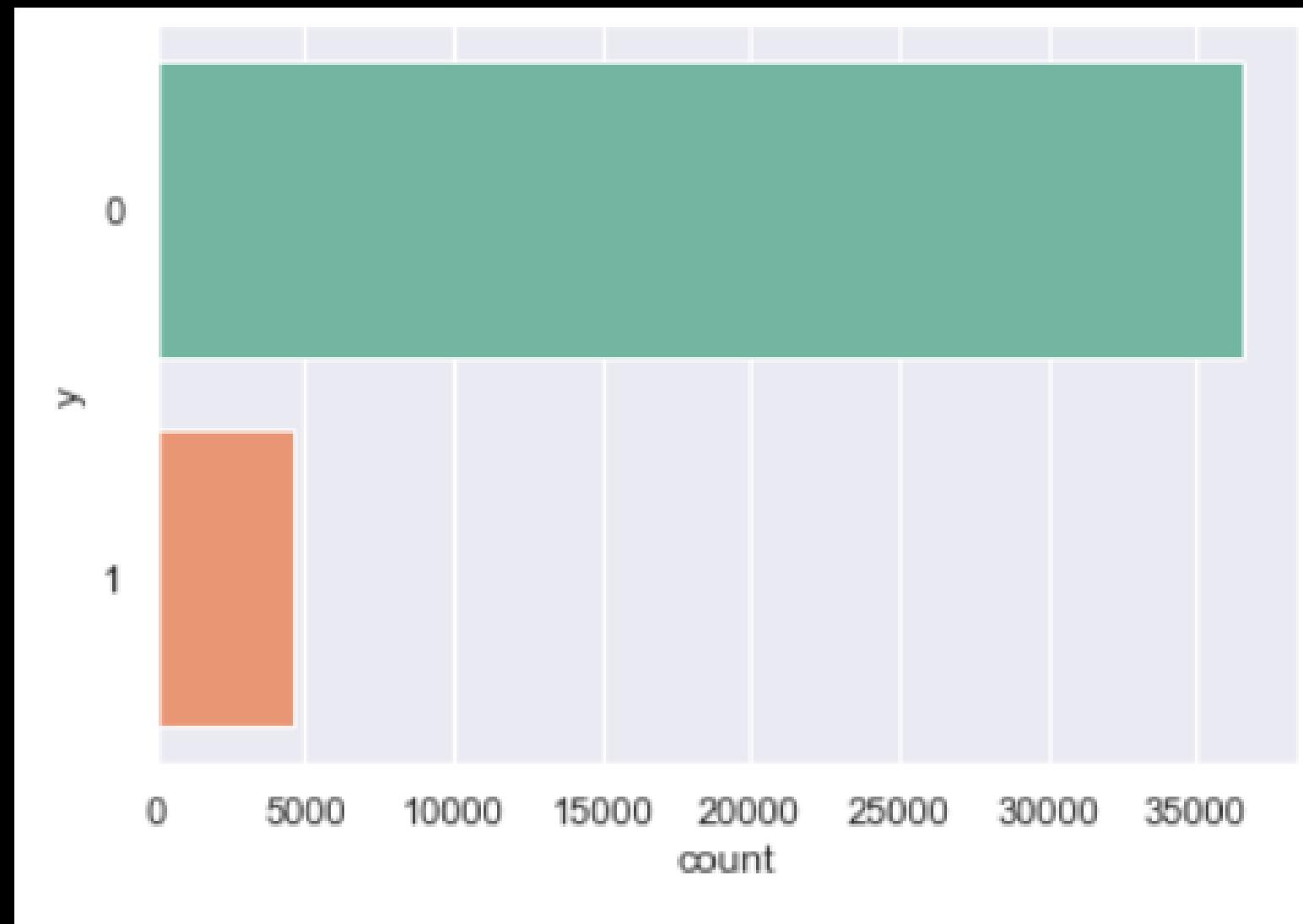
INVERSIONES REALIZADAS

Logramos esa cantidad en plazos  
fijos. Buscaremos mejorar estos  
resultados, para la campaña  
entrante.

**11%**

EFFECTIVIDAD

# Resultados 2021



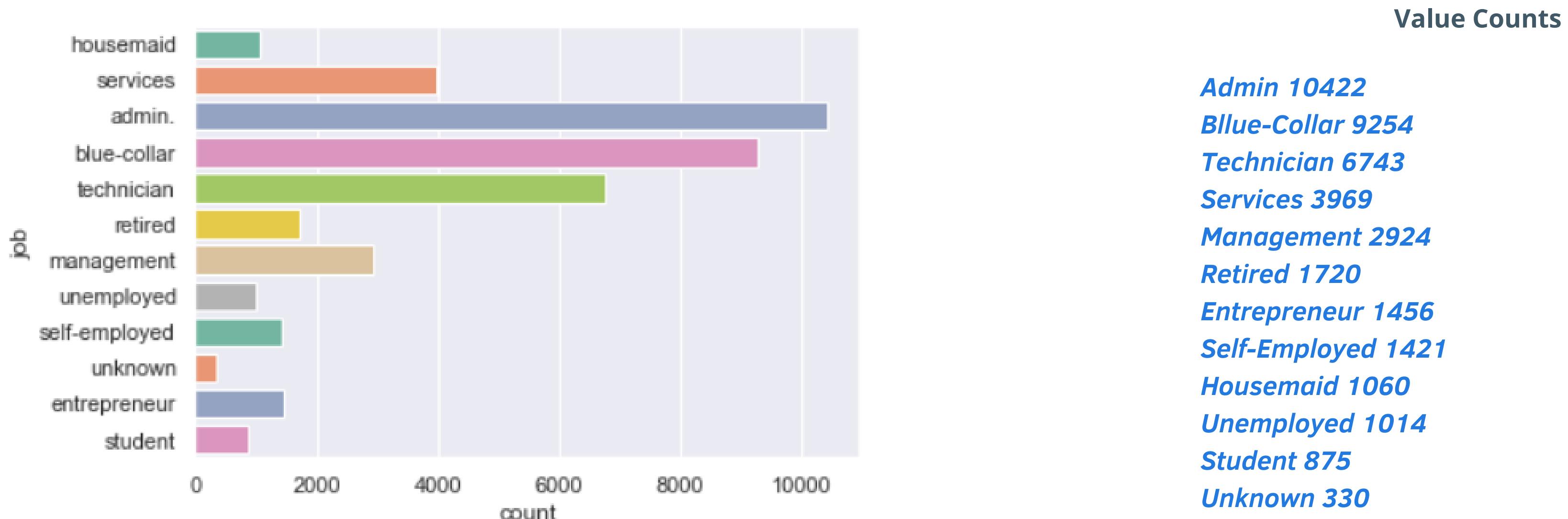
VISUALIZAREMOS LA DISTRIBUCIÓN DE LOS VALORES DE NUESTRO TARGET "Y"

Esta variable contiene 2 valores en función de si acepto realizar la inversión en la campaña:

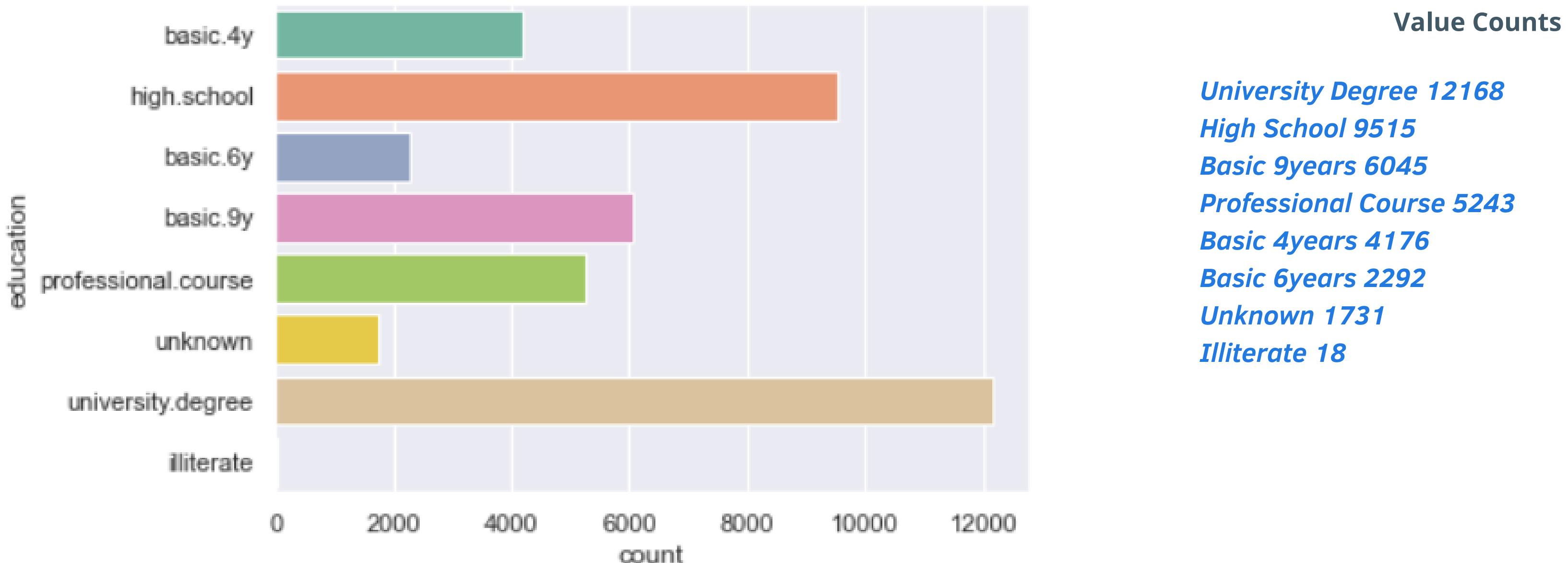
- 1 (yes)
- 0 (no)

Esto es lo que buscaremos mejorar cuando apliquemos los algoritmos de machine learning. Vemos como la mayoría fueron negativos, por lo que se encuentran desbalanceados de cara a futuro.

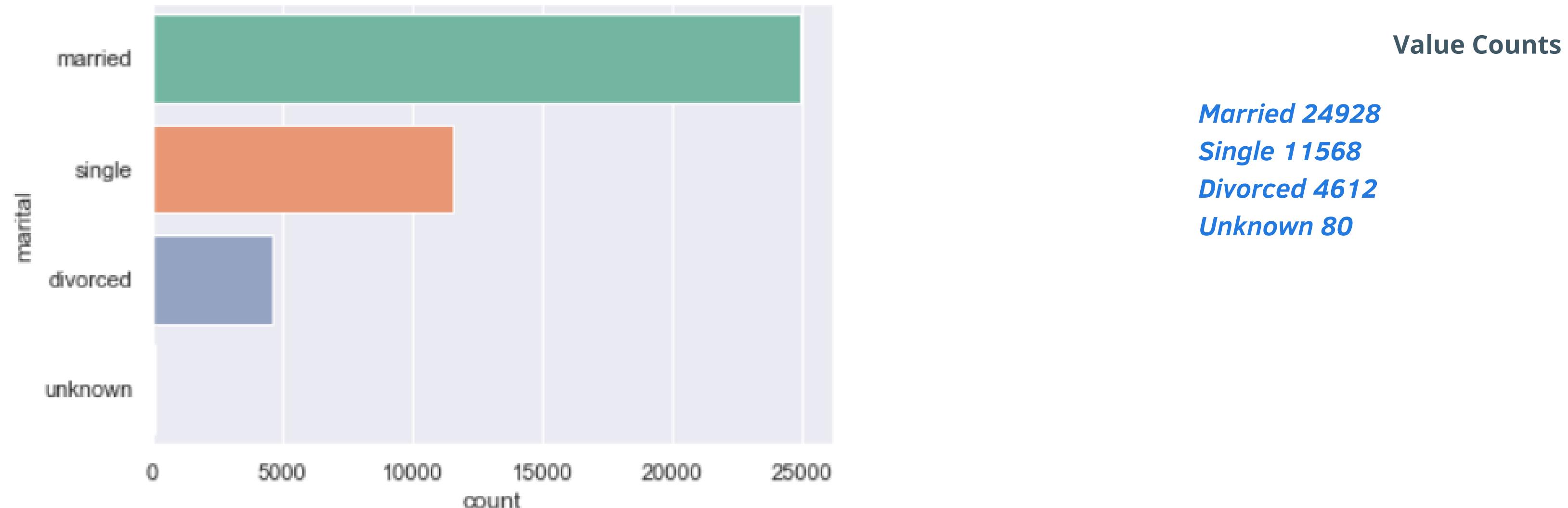
# Variables Categoricas



# Variables Categoricas

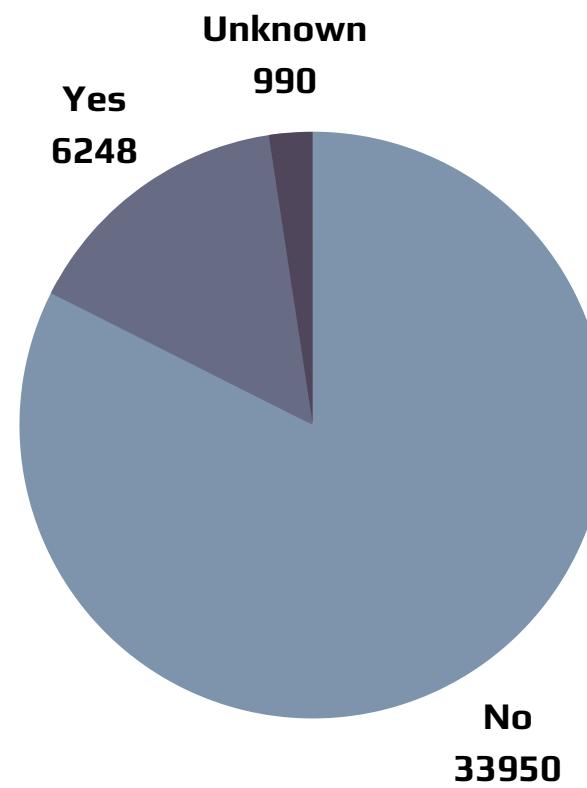


# Variables Categoricas

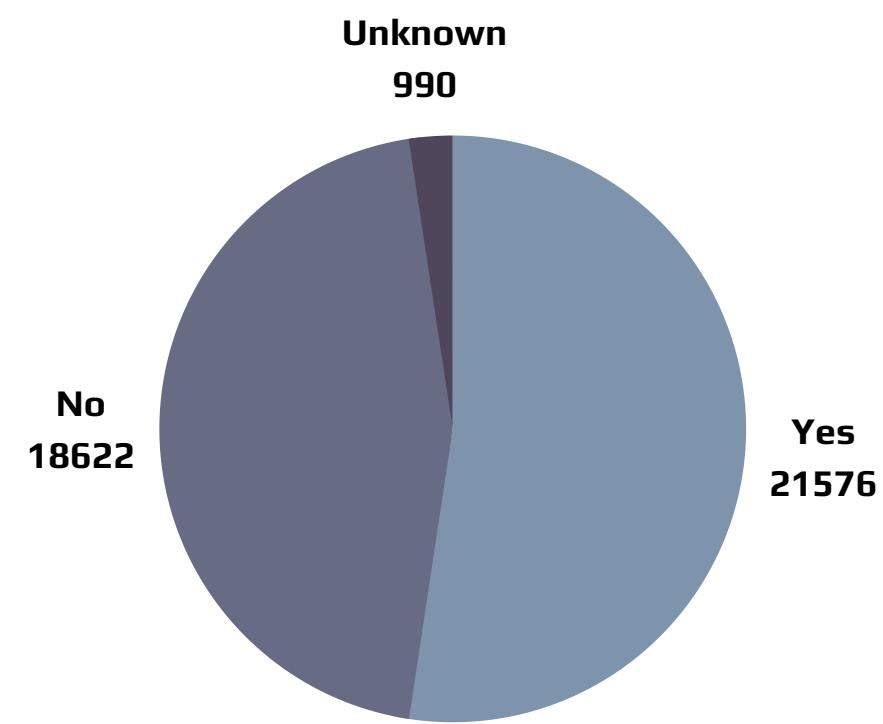


# Variables Categoricas

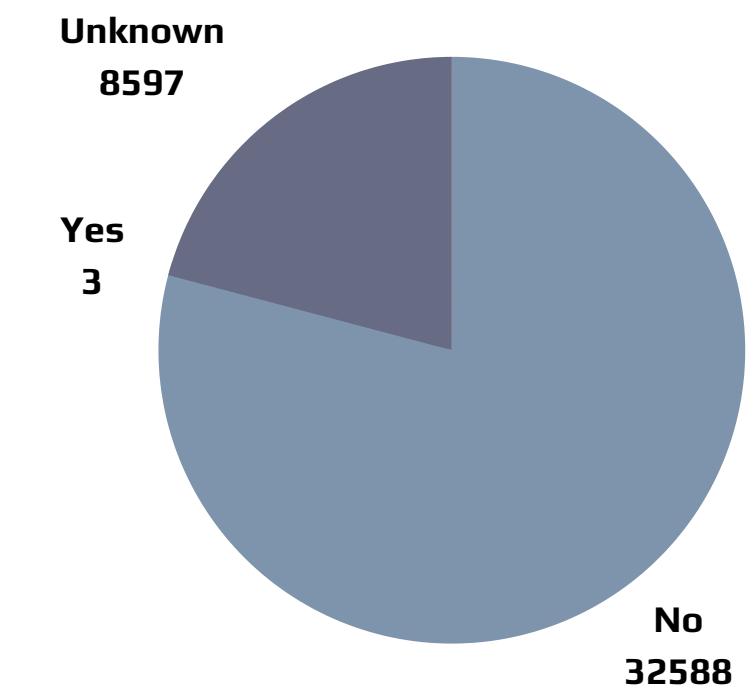
## Métricas Financieras



¿Has a Loan?



¿Has a housing loan?



Financial Default

# Variables Categoricas

## Metodo de Contacto

**26144**

**CELULAR**

**63.47%**

---

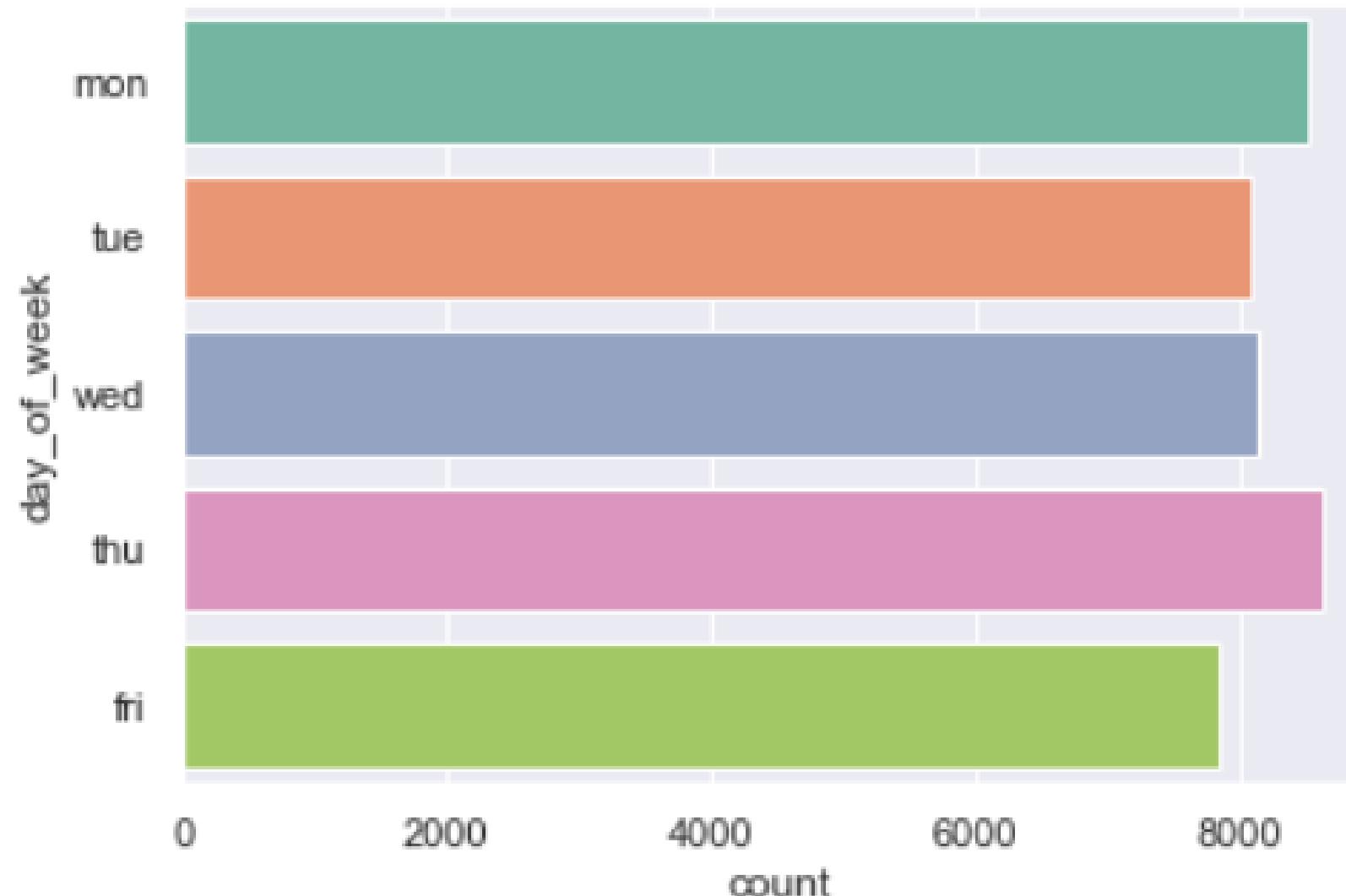
**15044**

**TELÉFONO**

**36,53%**

---

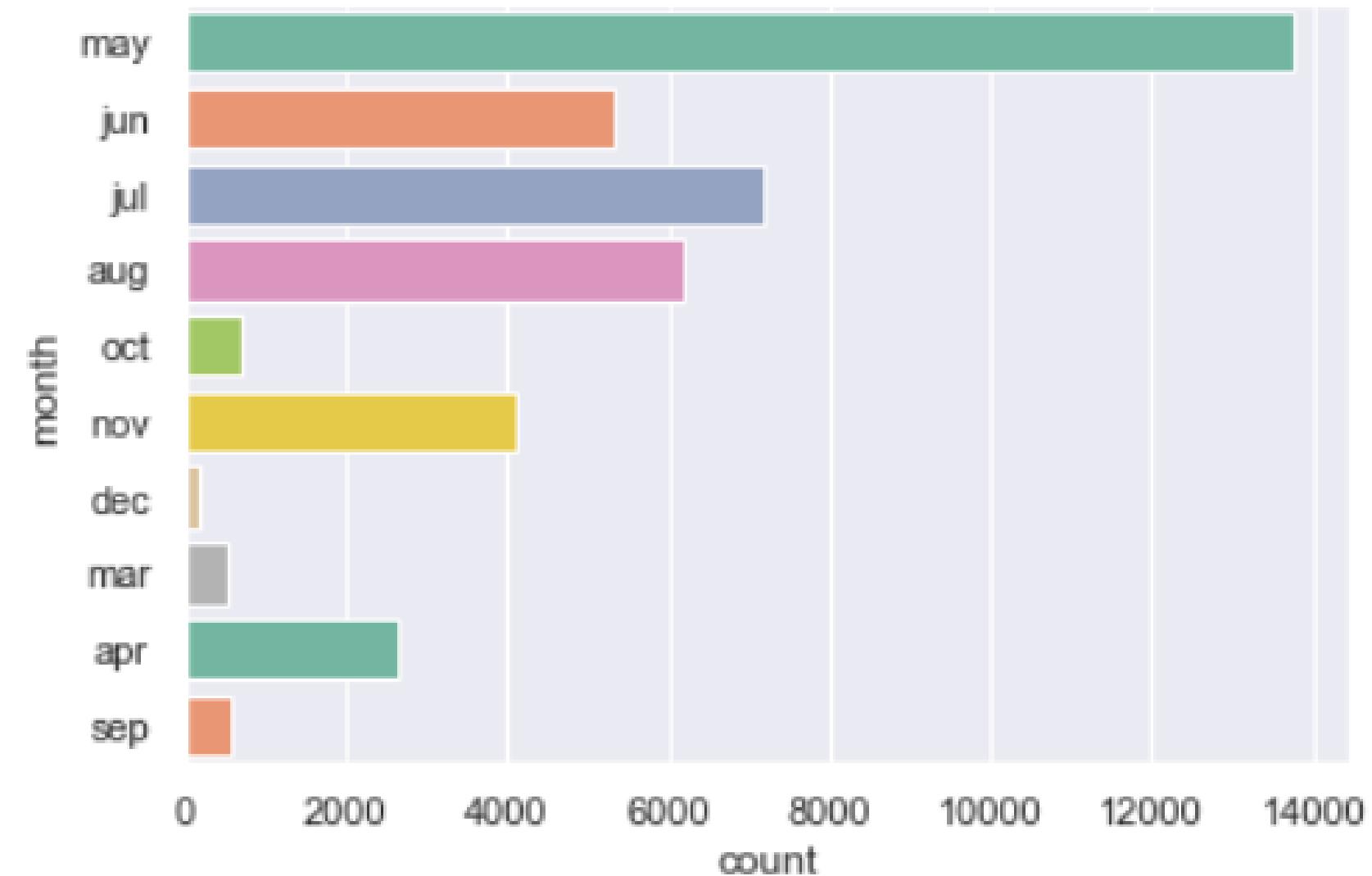
# Variables Categoricas



## Dias de la semana

Podemos observar que los valores de las ventas de inversiones en los días de la semana son similares. Esto nos indica que tendremos que focalizar nuestra atención en ver que meses son mas rentables que otros.

# Variables Categoricas



- Campaña realizada entre los meses de Marzo y Diciembre
  - Marzo 546*
  - Abril 2632*
  - Mayo 13769*
  - Junio 5318*
  - Julio 7174*
  - Agosto 6178*
  - Septiembre 570*
  - Octubre 718*
  - Noviembre 4101*
  - Diciembre 182*

# Variables Numericas

Información general relacionada a los contactos con clientes

**39673**

CLIENTES CONTACTADOS POR  
PRIMERA VEZ

-Representan el 96,32% de nuestro  
data set.

- Variable "pdays" (999 en el df)

**258**

TIEMPO DE LLAMADO

-Promedio de la duración de los  
llamados. Expresado en segundos.

- Variable "Duration"

**40**

EDAD PROMEDIO

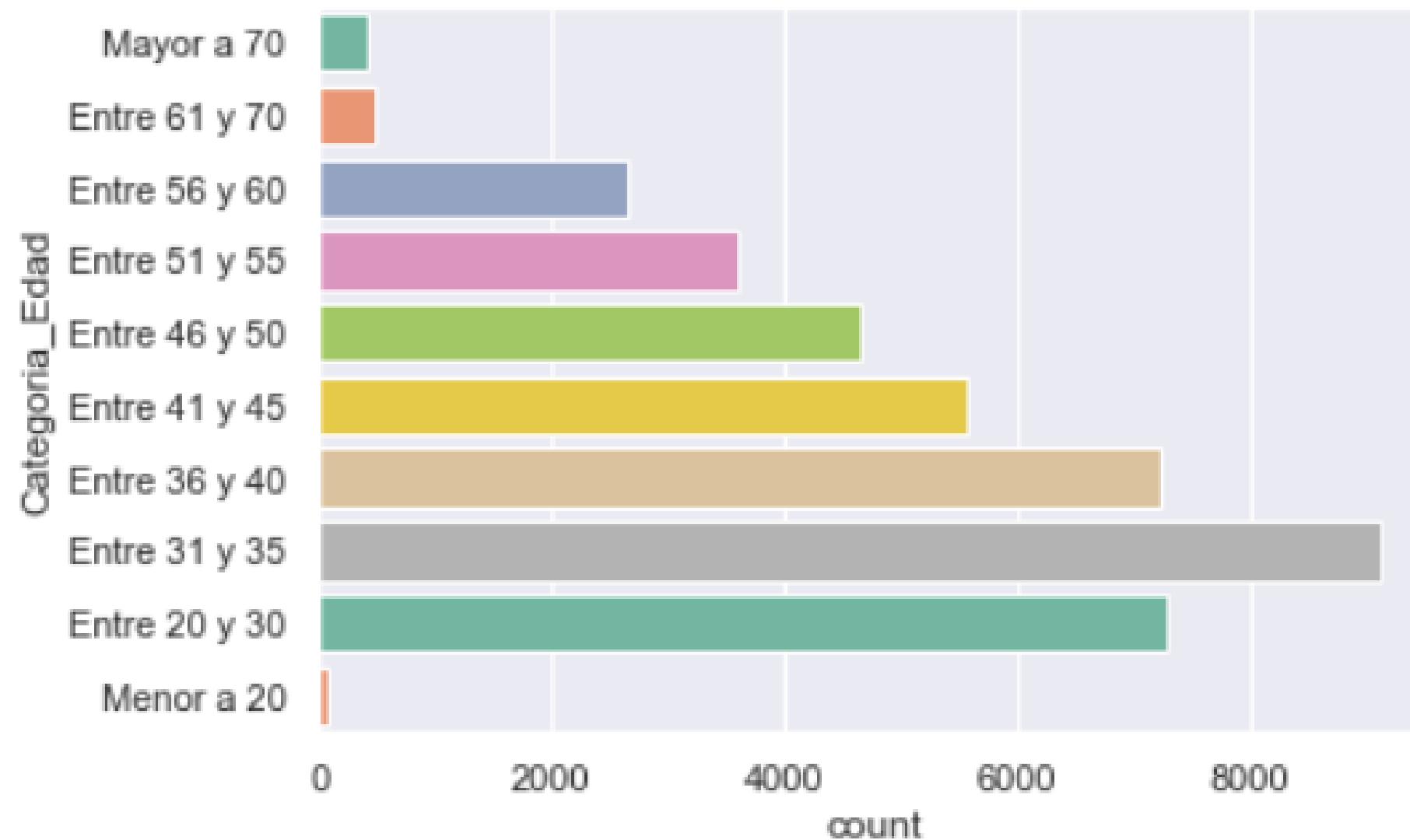
- Oscilan en un rango de 17 a 98  
años con un promedio de 40.

-Variable "age"

# Variables Numericas

AGE

## **DISTRIBUCIÓN POR GRUPO DE EDADES**



# Variables Numericas

## Indicadores Económicos

01

### TASA DE EMPLEO

Proporción de personas empleadas respecto a la población total

02

### INDICE DE PRECIOS AL CONSUMIDOR

Evaluación de los cambios de precios asociados con el costo de vida

03

### CONFIANZA DEL CONSUMIDOR

Grado de optimismo de los consumidores con respecto a sus perspectivas financieras

04

### ÍNDICE DE NUMERO DE EMPLEADOS

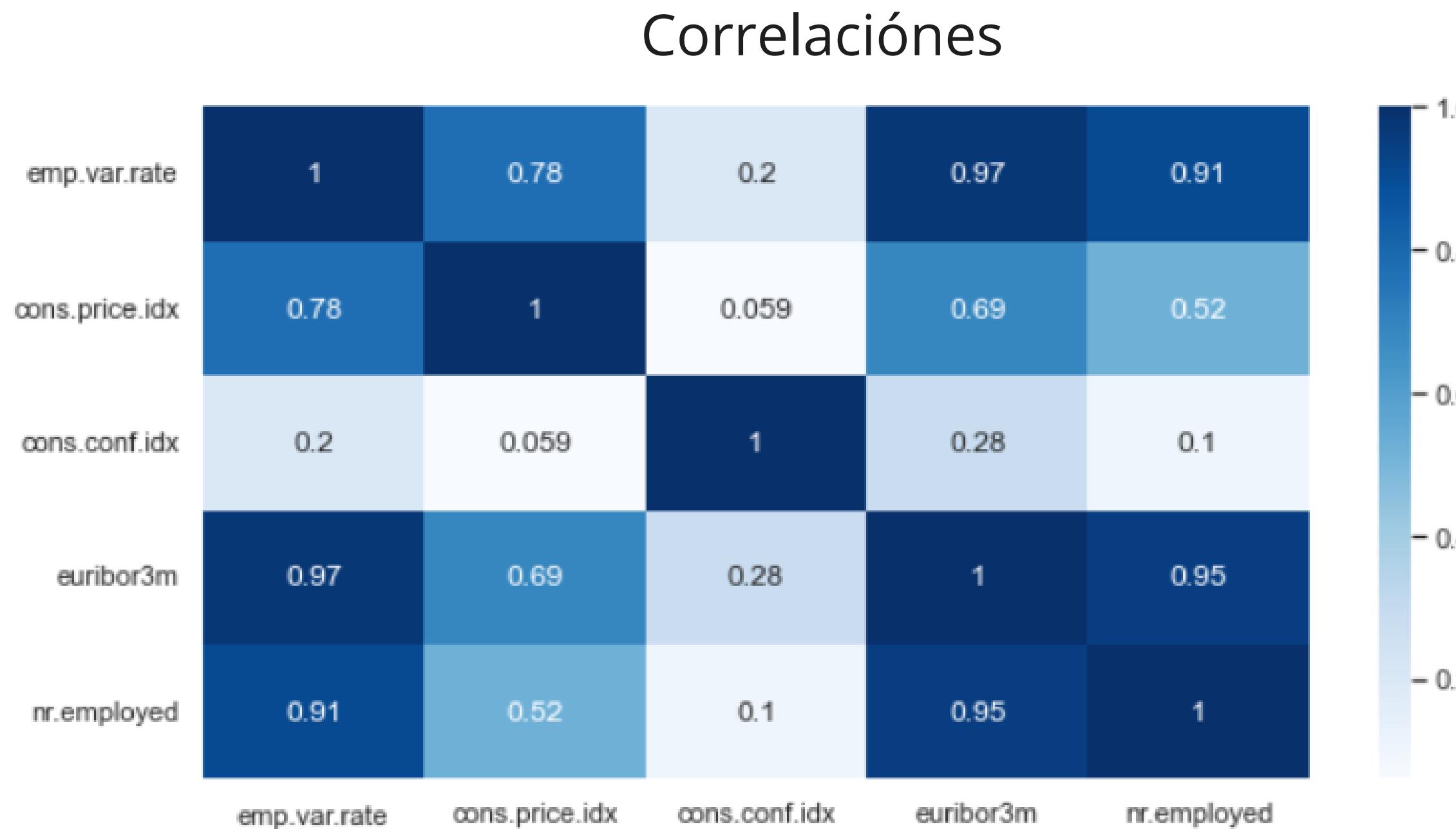
Aproximación del total de personas empleadas con contrato de trabajo.

05

### Tasa Euribor (3 meses)

Tasa de Interes que utilizan los Bancos para prestarse entre si.

# Heatmap - Indicadores Económicos



# One Hot Encoding

**Aplicamos esta técnica debido a la gran cantidad de variables categoricas que poseemos.**

Para esto, utilizamos el metodo get\_dummies de la libreria Pandas

**Variables categorias a convertir :**

- marital, education, job, contact, month y poutcome
- Estas 6 columnas luego del one hot encoding pasan a ser 39

# One Hot Encoding

Nuevas columnas

```
['marital_divorced', 'marital_married', 'marital_single',
'marital_unknown', 'education_basic.4y', 'education_basic.6y',
'education_basic.9y', 'education_high.school', 'education_illiterate',
'education_professional.course', 'education_university.degree',
'education_unknown', 'job_admin.', 'job_blue-collar',
'job_entrepreneur', 'job_housemaid', 'job_management', 'job_retired',
'job_self-employed', 'job_services', 'job_student', 'job_technician',
'job_unemployed', 'job_unknown', 'contact_cellular',
'contact_telephone', 'month_apr', 'month_aug', 'month_dec', 'month_jul',
'month_jun', 'month_mar', 'month_may', 'month_nov', 'month_oct',
'month_sep', 'poutcome_failure', 'poutcome_nonexistent',
'poutcome_success'],
```

# Machine Learning

**Algoritmos y librerías utilizadas**

Presentaremos detalles de lo aplicado  
en el notebook del proyecto

# Librerias

```
from sklearn.feature_selection import chi2
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import precision_recall_fscore_support
from sklearn.metrics import classification_report,roc_auc_score,roc_curve,auc
from sklearn import metrics
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from imblearn.over_sampling import RandomOverSampler
```

# X & Y Training y Test

80%

20%

**X Train = 30792,44**

**X Test = 7699,44**

**Y Train = 30792**

**Y Test = 7699**

# Algoritmos de clasificación

Con cross validation

## 01 KNN

```
knnclassifier = KNeighborsClassifier(n_neighbors = 4)
knn_score = cross_val_score(knnclassifier, x, y, cv = 10, scoring = 'precision').mean()
```

## 02 Regresion Logistica

```
RegresionLog=LogisticRegression(max_iter=1000)
RL_score = cross_val_score(RegresionLog, x, y, cv = 10, scoring = 'precision').mean()
```

## 03 Random Forest

```
RandomF = RandomForestClassifier(n_estimators = 20, criterion = 'entropy')
RandomF_score = cross_val_score(RandomF, x, y, cv = 10, scoring = 'precision').mean()
```

## 04 Bayes

```
Bayes = GaussianNB()
Bayes_score = cross_val_score(Bayes, x, y, cv = 10, scoring = 'precision').mean()
```

## 05 Arbol Decisión

```
ArbolDD = DecisionTreeClassifier()
ArbolDD_score = cross_val_score(ArbolDD, x, y, cv = 10, scoring = 'precision').mean()
```

# Algoritmos de clasificación

Con cross validation

Aplicamos Cross Validation para comparar los algoritmos:

| score               |          |
|---------------------|----------|
| modelo              |          |
| KNN                 | 0.020938 |
| Regresion_Logistica | 0.016811 |
| Random_forest       | 0.010082 |
| Bayes_ingenuo       | 0.145590 |
| Arbol de decisiones | 0.011059 |



Elegiremos esto debido a que tuvo el mejor score

# Algoritmos de clasificación

Con cross validation

## Bayes Ingenuo

```
Bayes = GaussianNB()  
Bayes.fit(x_train, y_train)  
  
prediccionBayes = Bayes.predict(x_test_Std)  
Bayes.score(x_test_Std,y_test)|
```

Score **0.895 %**

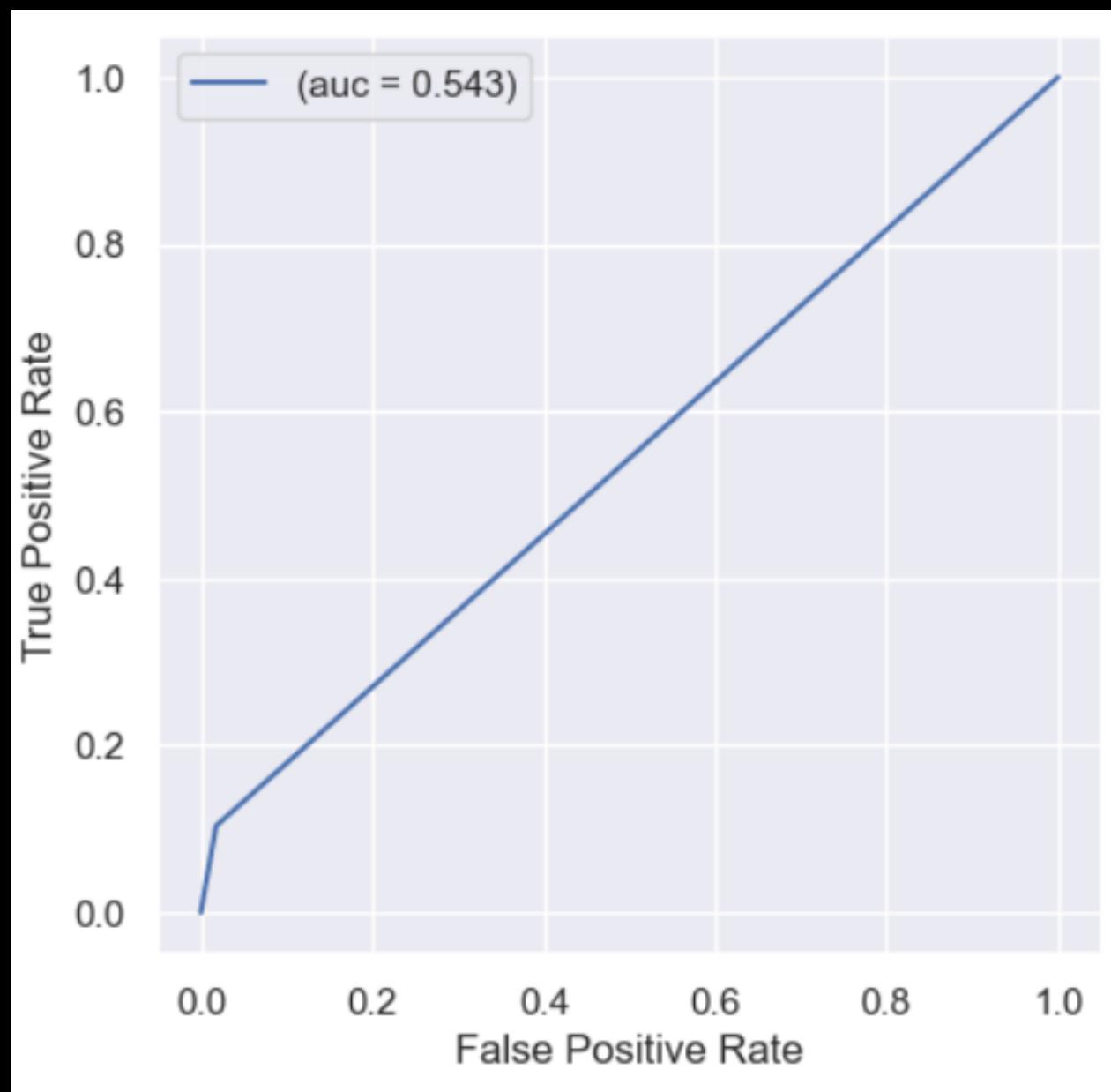
## Confusion Matrix

|            | Predecido N | Predecido P | %         |
|------------|-------------|-------------|-----------|
| Realidad N | 6814        | 123         | 98.226899 |
| Realidad P | 683         | 79          | 10.367454 |

# Algoritmos de clasificación

Con cross validation

## Bayes Ingenuo - ROC



Precisión en la predicción = **0.649 %**

- **10 % de verdaderos positivos.**
- **Este número no es óptimo, por lo que aplicaremos técnicas para balancear los datos y optimizar nuestro modelo.**

# Balanceo de Datos

## Random Over Sample

```
ros = RandomOverSampler()  
ros_x_train, ros_y_train = ros.fit_resample(x_train_Std, y_train)
```

## Datos Balanceados

0 (no) = 27900

1 (yes) = 27900

## Aplicamos el re sample

```
x_balanceado, y_balanceado = ros.fit_resample(x, y)
```

# Algoritmos de clasificación

Con cross validation y datos balanceados

## 01 KNN

```
knnclassifier = KNeighborsClassifier(n_neighbors = 4)
knn_score = cross_val_score(knnclassifier, x, y, cv = 10, scoring = 'precision').mean()
```

## 02 Regresion Logistica

```
RegresionLog=LogisticRegression(max_iter=1000)
RL_score = cross_val_score(RegresionLog, x, y, cv = 10, scoring = 'precision').mean()
```

## 03 Random Forest

```
RandomF = RandomForestClassifier(n_estimators = 20, criterion = 'entropy')
RandomF_score = cross_val_score(RandomF, x, y, cv = 10, scoring = 'precision').mean()
```

## 04 Bayes

```
Bayes = GaussianNB()
Bayes_score = cross_val_score(Bayes, x, y, cv = 10, scoring = 'precision').mean()
```

## 05 Arbol Decisión

```
ArbolDD = DecisionTreeClassifier()
ArbolDD_score = cross_val_score(ArbolDD, x, y, cv = 10, scoring = 'precision').mean()
```

# Algoritmos de clasificación

Con cross validation

**Aplicamos Cross Validation para comparar los algoritmos:**

| modelo              | score    |
|---------------------|----------|
| KNN                 | 0.762704 |
| Regresion_Logistica | 0.782532 |
| Random_forest       | 0.674416 |
| Bayes_ingenuo       | 0.756092 |
| Arbol de decisiones | 0.648872 |



Elegiremos esto debido a que tuvo el mejor score

# Algoritmos de clasificación

Con cross validation y datos balanceados

## Regresion Logistica

```
RegresionLog=LogisticRegression()  
RegresionLog.fit(ros_x_train, ros_y_train)  
  
prediccionRegLog = RegresionLog.predict(x_test_Std)  
RegresionLog.score(x_test_Std,y_test)
```

Score **0.74 %**

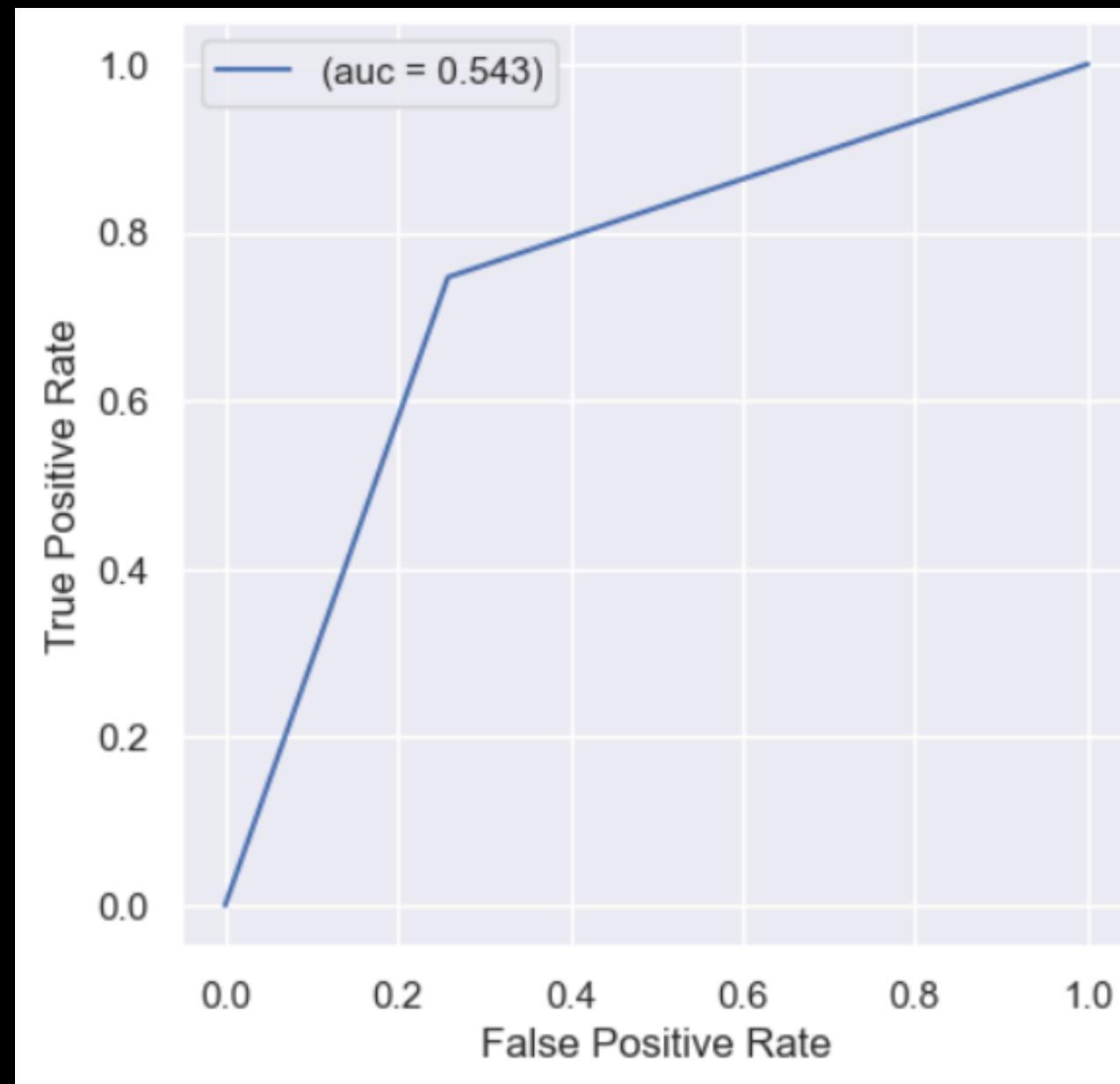
## Confusion Matrix

|            | Predecido N | Predecido P | %         |
|------------|-------------|-------------|-----------|
| Realidad N | 5144        | 1793        | 74.153092 |
| Realidad P | 193         | 569         | 74.671916 |

# Algoritmos de clasificación

Con cross validation

## Regresión Logística - ROC



Precisión en la predicción = **0.602 %**

Disminuye la precisión debido a que luego de balancear los datos (98% vs 74%) los casos de negativo predicho correctamente también bajaron.

# Conclusiones

## TESTEO DEL MODELO EN DATOS BALANCEADOS VS DESBALANCEADOS

En próximas entregas buscaremos elegir la manera mas optima de aplicar los datos para predecir resultados

El modelo es efectivo en identificar casos positivos, y lo optimizaremos para evitar overfitting a la hora de tomar decisiones.

Realizaremos mejoras en los algoritmos utilizados para mejorar los resultados y aplicaremos nuevas técnicas de balanceo de datos

# ¿Tenés alguna pregunta?

**Presentación entregada con el notebook correspondiente. El mismo cuenta con el análisis exploratorio realizado (univariado,bivariado y multivariado).**

**Para la segunda entrega del proyecto, adicionamos los algoritmos de Machine Learning aplicados y explicados en detalle.**

## NÚMERO TELEFÓNICO

1131061104  
1139400927  
1159905651

## SITIO WEB

<https://github.com/NicolasMos>  
[es/Trabajo-final-Coderhouse](#)

## E-MAIL

nikoamoses@gmail.com  
mollonicolas95@gmail.com  
gabrielmazzitelli86@gmail.com