

Camilo Sebastian Duque Cardenas – 202024289

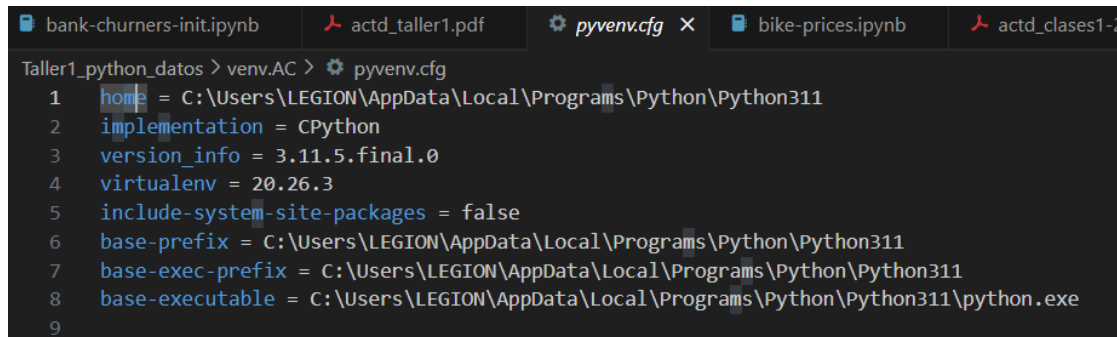
Analítica Computacional IIND-4130

8 de agosto de 2024

## Reporte Taller 1

### 1. Instale su ambiente de desarrollo

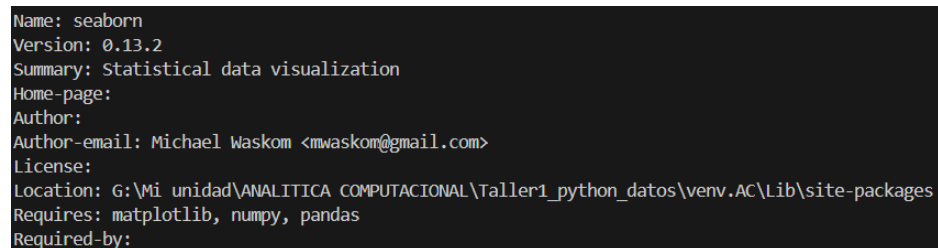
#### 1.1 Ubicación (directorio) local en la que queda instalado Python:



```
bank-churners-init.ipynb  actd_taller1.pdf  pyenvn.cfg  bike-prices.ipynb  actd_clases1-2
Taller1_python_datos > venv.AC > pyenvn.cfg
1  home = C:\Users\LEGION\AppData\Local\Programs\Python\Python311
2  implementation = CPython
3  version_info = 3.11.5.final.0
4  virtualenv = 20.26.3
5  include-system-site-packages = false
6  base-prefix = C:\Users\LEGION\AppData\Local\Programs\Python\Python311
7  base-exec-prefix = C:\Users\LEGION\AppData\Local\Programs\Python\Python311
8  base-executable = C:\Users\LEGION\AppData\Local\Programs\Python\Python311\python.exe
9
```

Figura 1. Directorio Local (base) Instalación de Python 3.11 y Usuario

#### 1.6 Versión instalada de cada paquete y su usuario en el equipo local:



```
Name: seaborn
Version: 0.13.2
Summary: Statistical data visualization
Home-page:
Author:
Author-email: Michael Waskom <mwaskom@gmail.com>
License:
Location: G:\Mi unidad\ANALITICA COMPUTACIONAL\Taller1_python_datos\venv.AC\Lib\site-packages
Requires: matplotlib, numpy, pandas
Required-by:
```

Figura 2. Dirección de ambiente virtual de instalación de paquetes

```
(venv.AC) G:\Mi unidad\ANALITICA COMPUTACIONAL\Taller1_python_datos>pip list
Package Version
-----
contourpy 1.2.1
cyclor 0.12.1
fonttools 4.53.1
kiwisolver 1.4.5
matplotlib 3.9.1.post1
numpy 2.0.1
packaging 24.1
pandas 2.2.2
pillow 10.4.0
pip 24.2
pyparsing 3.1.2
python-dateutil 2.9.0.post0
pytz 2024.1
scipy 1.14.0
seaborn 0.13.2
setuptools 70.1.0
six 1.16.0
tzdata 2024.1
wheel 0.43.0
```

Figura 3. Versión instalada de cada paquete

## 2. Exploración de Datos en Python

### 2.4 Explicar Celdas “bank-churners-init.ipynb”:

- **Celda 1:** Importar las librerías “numpy” y “pandas” y asignarles un alias en este caso se le asigno “np” y “pd” respectivamente para llamarlas de una forma más corta.
- **Celda 2:** Cargar los datos crudos que están en el archivo (“BankChurn.csv”) usando la función “read\_csv()” porque el archivo esta en el formato separado por comas y convertirlo en un “dataframe” que es un objeto de pandas.
- **Celda 3:** Usar la función “shape” del objeto dataframe para saber las dimensiones del dataframe, es decir, el número de filas y columnas respectivamente devuelto como una tupla.
- **Celda 4:** Usar la función “head()” que devuelve los cinco primeros registros del dataframe.
- **Celda 5:** Usar la función “unique” para extraer los valores únicos de la columna “Attrition\_Flag”.

- **Celda 6:** Usar la función contar "count" para contar cuantos datos hay de cada cada columna y "groupby" para agruparlo por la columna "Attrition\_Flag".
- **Celda 7:** Usar la función "value\_counts()" para contar cuantos hay de la columna "Attrition\_Flag".
- **Celda 8:** Usar la función "describe()" para obtener estadísticas descriptivas básicas del dataframe.
- **Celda 9:** Calcular con "value\_counts()" contar la cantidad de veces de cada "Attrition\_Flag" e indexar (filtrar) los que retiraron representado con 1 sobre el total que son la suma de los que existen y los que se retiraron. Calcular la tasa de abandono (churn rate).
- **Celda 10:** Seleccionar y filtrar con la función ".duplicated()" los registros que son idénticos en cada columna, aplicar "len" que es para determinar el número total de registros que hay duplicados.
- **Celda 11:** Usar función "isnull()" para determinar cuántos datos nulos hay en cada columna usar la función "sum()" para sumar cuantos nulos hay en cada columna y "sum()" nuevamente para sumar cuantos nulos hay en total en todo el dataframe.
- **Celda 12:** Usar función "dtypes" sobre el dataframe para determinar que tipo de datos es cada columna y sobre esto aplicar value\_counts() que cuenta la frecuencia de cada tipo de dato.
- **Celda 13:** Usar función "map()" para conversión de las características en caso de "Gender" por variables escalares usando un diccionario poniendo como llave "M" y valor "0" y otro elemento con llave "F" y valor "1", lo mismo para "Attrition\_Flag".
- **Celda 13:** Usar función "select\_dtypes()" para seleccionar las columnas que tengan o no tengan el tipo de dato especificado dependiendo si es "include" o "exclude". Después usar la función "get\_dummies()" para generar variables

dummies para las columnas que no son ni entero ni float sino que son categóricas.

## 2.5 Histograma de Edad del Cliente:

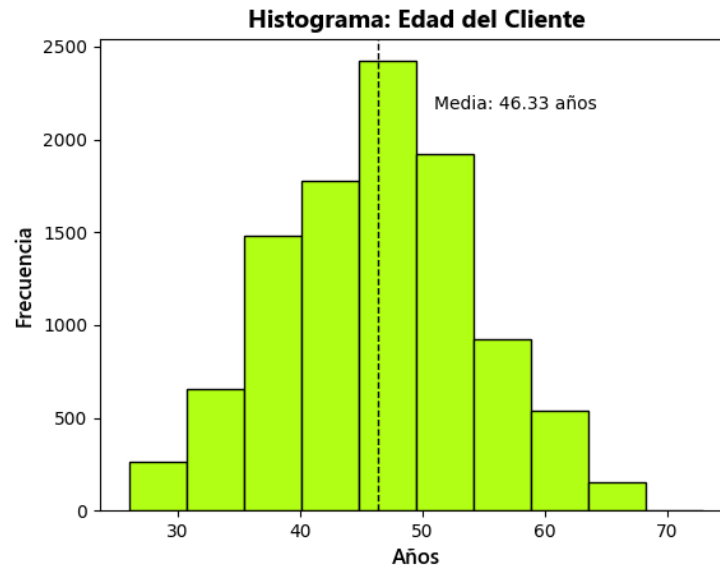


Figura 4. Histograma Edad del Cliente

## 2.5 Histograma de Histograma Precio de Venta de Motocicletas:

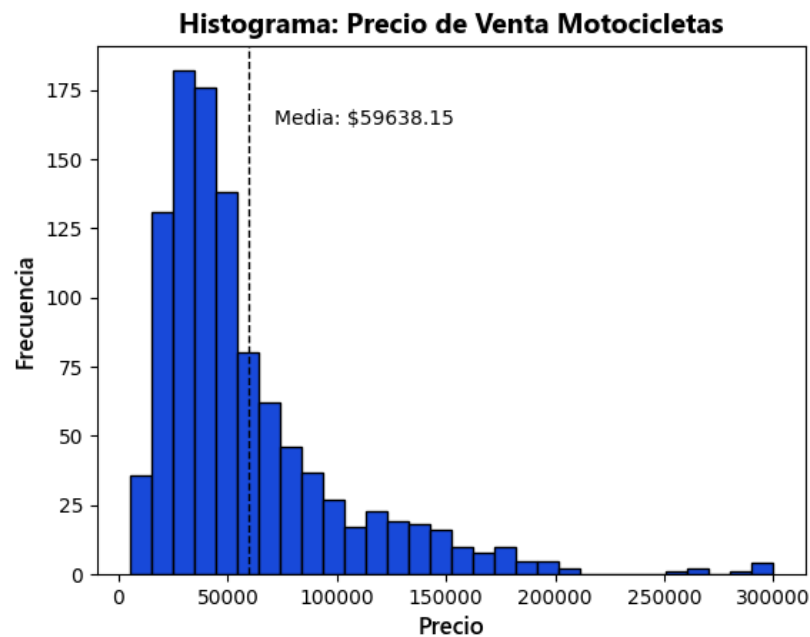


Figura 5. Histograma Precio de Venta Motocicletas

### 3. Números Aleatorios y Bondad de Ajuste

#### 3.4 Explicar Celdas "data-gen.ipynb":

- **Celda 1:** Importar la librería "numpy". Usar la función "random.normal()" para generar 1000 datos que sigan una distribución normal con media = 3 y desviación estándar = 0.5.
- **Celda 2:** Importar la librería "pandas", convertir los datos generados de numpy que están en formato matriz en un objeto de tipo dataframe usando "pd.DataFrame()" y usar "describe()" para sacar estadísticas descriptivas de los datos.
- **Celda 3:** Importar la librería "pyplot" de matplotlib, generar un histograma con "plt.hist()" y almacenar las variables "counts" y "bins", agregar el título con "plt.title()", agregar título del eje x con "plt.xlabel()", agregar título del eje y con "plt.ylabel()", usar "plt.show()" para mostrar la gráfica. Por último, imprimir el rango de los bins y el número de datos que contiene cada rango.
- **Celda 4:** Importar la librería "scipy", "numpy" y "matplotlib", usar np.arange() para generar datos entre el rango (-4, 4) devolviendo valores espaciados uniformemente por 0.001. Usar "norm.pdf()" para generar distribución normal de los datos entre -4 y 4 como no se especifica la media y ni la desviación estándar, entonces por default son 0 y 1 respectivamente. Finalmente, se imprime el gráfico.
- **Celda 5:** Generar gráfica de distribución normal con media (loc) = 3 y usando la desviación estándar default (scale) = 1 entre los valores de 0 y 6.
- **Celda 6:** Generar gráfica de distribución normal con media (loc) = 3 y asignando la desviación estándar (scale) = 0.5 entre los valores de 0 y 6.
- **Celda 7:** Importar la librería "statsmodels", usar la función "qqplot()" para graficar un Quantile-Quantile Plot especificando los datos generados inicialmente, con una distribución normal con media (loc) = 3, desviación estándar (scale) = 0.5 y trazar la línea de 45 grados.

- **Celda 8:** Usar la función "qqplot()" para graficar un Quantile-Quantile Plot especificando los datos generados inicialmente, con una distribución normal con media ( $\text{loc}$ ) = 3, desviación estándar ( $\text{scale}$ ) = 1 y trazar la línea de 45 grados.
- **Celda 9:** Usar la función "qqplot()" para graficar un Quantile-Quantile Plot especificando los datos generados inicialmente, con una distribución normal con media ( $\text{loc}$ ) = 0, desviación estándar ( $\text{scale}$ ) = 1 y trazar la línea de 45 grados.
- **Celda 10:** Importar "matplotlib", generar con "hist()" el histograma de los datos generados inicialmente indicado densidad = True para que el área debajo de la curva o la integral de la curva sea igual a 1 y graficar línea de distribución normal con "plot()" con los valores x generados con media = 3 y desviación estándar = 0.5.

### 3.5 Replicar Ejercicio usando Exponencial:

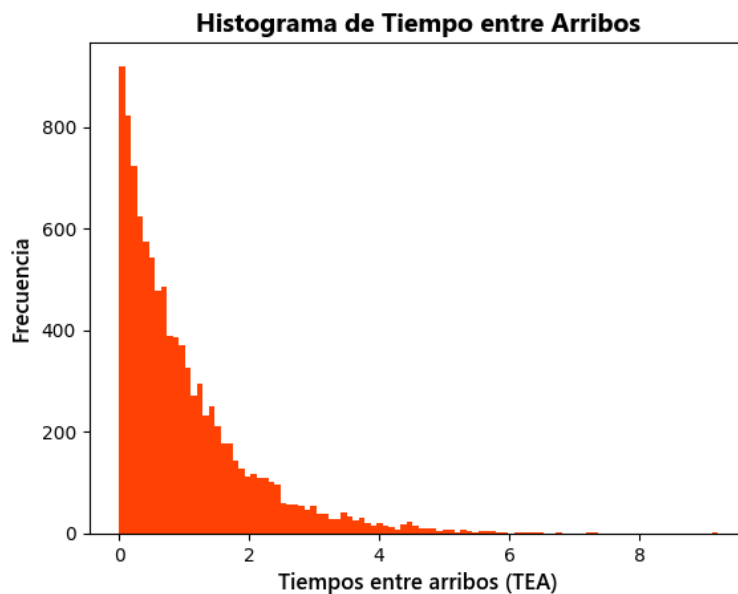


Figura 6. Histograma Tiempo Entre Arribos

En la Figura 6 se generaron 10,000 datos con una distribución exponencial con  $\sigma = 1$ .

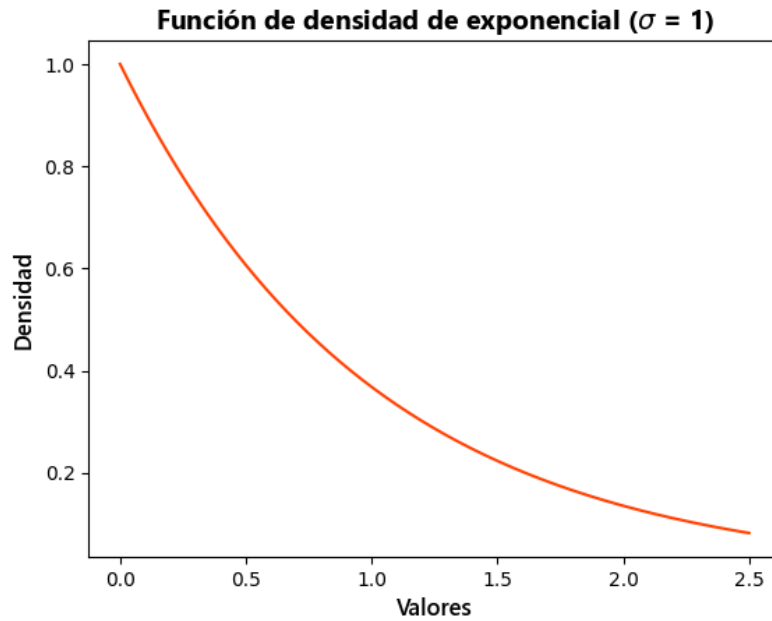


Figura 7. Función de Densidad de Exponencial ( $\sigma = 1$ )

En la Figura 7 se puede evidenciar la función de densidad de probabilidad para una distribución exponencial con  $\sigma = 1$ .

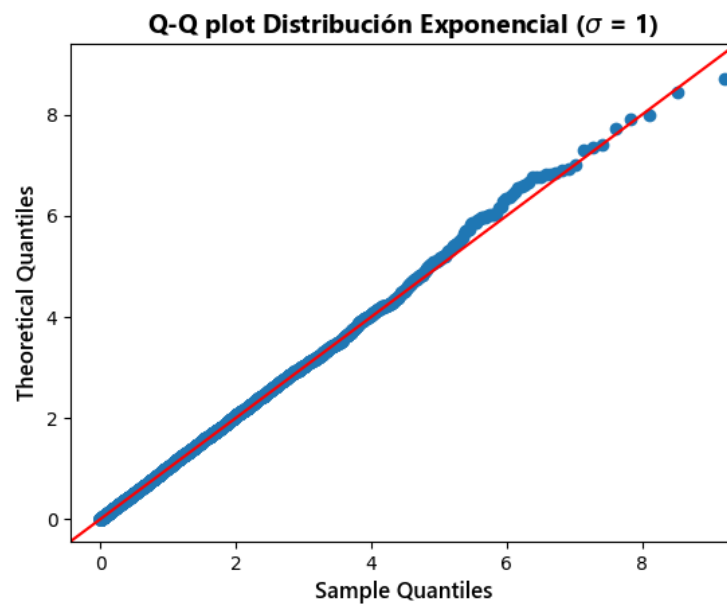


Figura 8. Q-Q Plot Distribución Exponencial ( $\sigma = 1$ )

En la Figura 8 se puede evidenciar un Q-Q Plot para una distribución exponencial con  $\sigma = 1$  y los puntos medios de la gráfica se encuentran sobre la recta, lo

cual es un indicio que los datos posiblemente pudieran seguir esta distribución de probabilidad, pero tocaría realizar una prueba de bondad/ajuste y determinarlo con el p-value para validar.

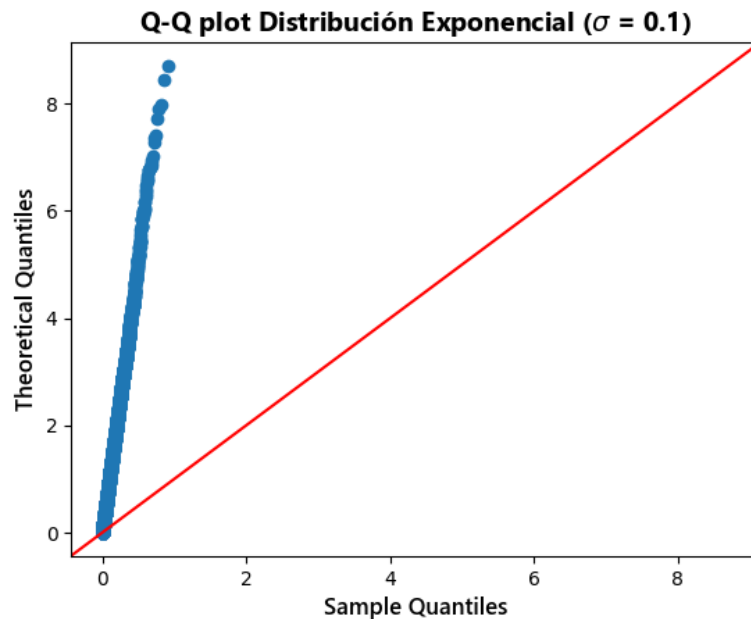


Figura 9. Q-Q Plot Distribución Exponencial ( $\sigma = 0.1$ )

En la Figura 9 se puede evidenciar un Q-Q Plot para una distribución exponencial con  $\sigma = 0.1$  y los puntos medios de la gráfica se encuentran desviados de la recta, lo cual es un indicio que los datos **NO** siguen esta distribución de probabilidad, pero tocaría realizar una prueba de bondad/ajuste y determinarlo con el p-value para validar.



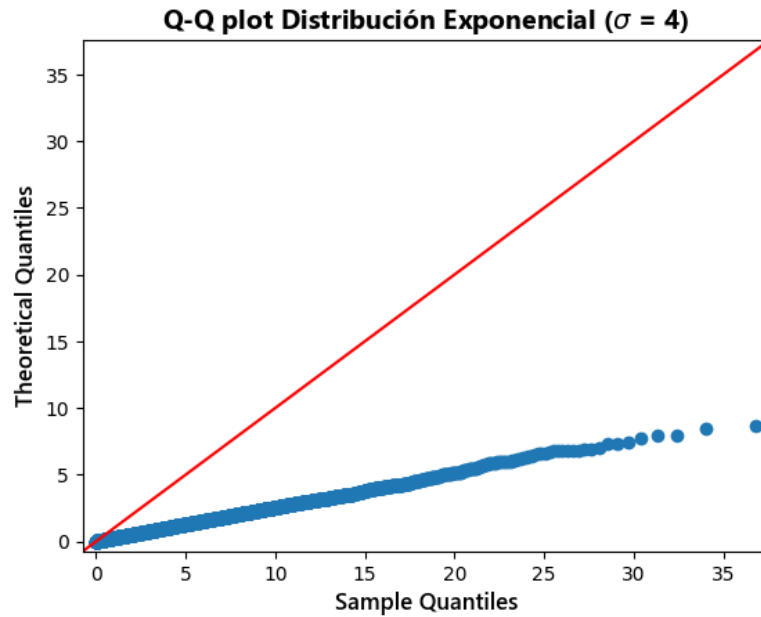


Figura 10. Q-Q Plot Distribución Exponencial ( $\sigma = 4$ )

En la Figura 10 se puede evidenciar un Q-Q Plot para una distribución exponencial con  $\sigma = 4$  y los puntos medios de la gráfica se encuentran desviados de la recta, lo cual es un indicio que los datos **NO** siguen esta distribución de probabilidad, pero tocaría realizar una prueba de bondad/ajuste y determinarlo con el p-value para validar.

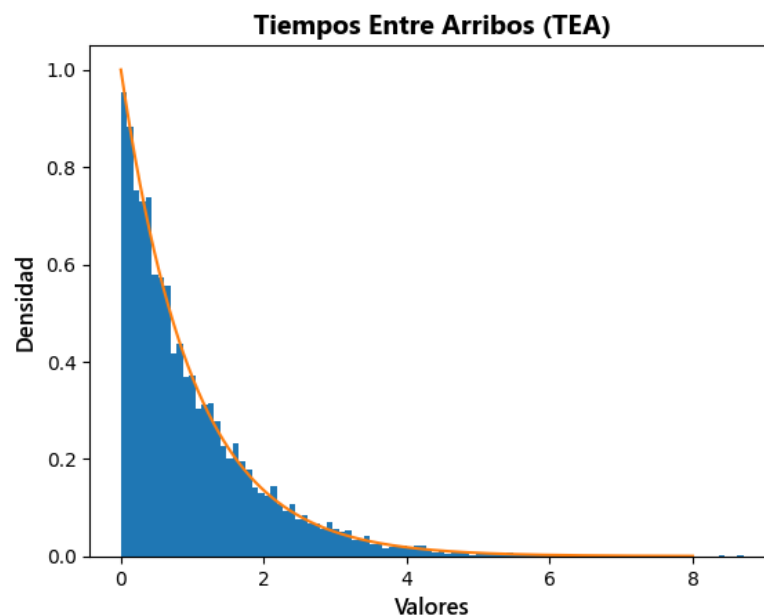


Figura 11. Histograma Tiempo entre Arribos y Función de Densidad de Distribución Exponencial con  $\sigma = 1$

En la Figura 10 se puede evidenciar un Histograma del Tiempo entre arribos de los datos y la gráfica de densidad de probabilidad de una distribución exponencial con  $\sigma=1$ . Claramente, pareciese que la distribución exponencial con  $\sigma = 1$  se ajusta a los datos.