

# **Análisis de Datos FIFA Players**

Autor: Bryam David Vega Moreno
Maestro: Diego Quisi
Materia: Simulación
Universidad: Universidad Politécnica Salesiana
Carrera: Ciencias de la computación

#### Introducción

En este informe presentamos el uso de librerias para visualización. Si bien matplotlib es una de las librerias más populares para la impresión de gráficos en python, actualmente existen nuevas librerias que permiten realizar un mejor desempeño en cuanto a graficas se refiere. En esta ocasión presentamos dos librerias llamadas plotly y altair las cuales permiten realizar gráficos con interacción, a su vez estas se estan volviendo populares debido a su diseño y facilidad de uso al momento de utiliar dichas librerias. Esta es una alternativa nueva para viejas librerias como matplotlib.

Los datos que tenemos para realizar las gráficas son sacadas de la página data.world la cual contiene información de los jugadores fifa con sus respectivas características.

### Librerias a importar

### Para análisis de datos

In [8]: import pandas as pd

## Para gráficos

In [9]: import altair as alt import plotly.express as px import plotly

### plotly.offline.init\_notebook\_mode(connected=True)

# Lectura de datos

Mediante la libreria de **pandas** podremos realizar la lectura del conjunto de datos que tenemos y a la vez poder realizar algunas operaciones que pandas nos ofrece con su potencialidad.

# In [10]: df = pd.read\_csv('./in/fifa.csv') df.head()

### Out[10]:

	id	name	full_name	birth_date	age	height_cm	weight_kgs	positions	nationality	overall_rating	 LWB	LDM	CDM	RDM	RWB	LB	I
0	158023	L. Messi	Lionel Andrés Messi Cuccittini	1987-06- 24	31	170.18	72.1	CF,RW,ST	Argentina	94	 64+2	61+2	61+2	61+2	64+2	59+2	4
1	190460	C. Eriksen	Christian Dannemann Eriksen	1992-02- 14	27	154.94	76.2	CAM,RM,CM	Denmark	88	 71+3	71+3	71+3	71+3	71+3	66+3	5
2	195864	P. Pogba	Paul Pogba	1993-03- 15	25	190.50	83.9	CM,CAM	France	88	 76+3	77+3	77+3	77+3	76+3	74+3	7
3	198219	L. Insigne	Lorenzo Insigne	1991-06- 04	27	162.56	59.0	LW,ST	Italy	88	 63+3	58+3	58+3	58+3	63+3	58+3	4
4	201024	K. Koulibaly	Kalidou Koulibaly	1991-06- 20	27	187.96	88.9	СВ	Senegal	88	 73+3	77+3	77+3	77+3	73+3	76+3	8

5 rows × 92 columns

Una vez leido los datos, procedemos a realizar los análisis planteados para esta tarea

### **Análisis de datos**

### Primer enunciado:

Con los datos FIFA, organizar a los jugadores de acuerdo al peso en las siguientes escalas y generar un caudro tipo PIE

- Debajo de 125 lbs
- 125-150
- 150-175175 o superior

## Solución

Primera debemos tomar en cuenta que el conjunto de datos tiene los pesos en kgs por lo cual tenemos que hacer una conversión de kgs a lbs. A continuación mostramos el procedimiento para realizarlo

# In [11]: df['weight\_lbs'] = df['weight\_kgs']\*2.20462 df[['weight\_lbs', 'weight\_kgs']].head()

## Out[11]: weight\_lbs weight\_

**3** 130.072580

**4** 195.990718

 weight\_lbs
 weight\_kgs

 0
 158.953102
 72.1

 1
 167.992044
 76.2

 2
 184.967618
 83.9

Con dicha conversión pcedemos a crear un pequeño dataset con las escalas que se han específicado, con la finalidad de graficarlo con un PIE.

, df.query('wei
df\_weight=pd.DataFrame(dict\_weight)
df\_weight

59.0

88.9

Out[12]: weight\_lbs values

**0** <125 0.002005 **1** >=125 and <150 0.163752

**2** >=150 and <175 0.558928

**3** >=175 0.275315

Una vez creado el conjunto de datos con las escalas correspondientes, procedemos a graficarlo utilizando la librería plotly que tiene la potencialidad de generar **graficos pastel** muy buenos. A continuación mostramos el resultado

color\_discrete\_sequence=px.colors.sequential.Aggrnyl,width=500,height=500)
fig.update\_layout(title\_text='Weight Soccer Players FIFA', title\_x=0.5)

Weight Soccer Players FIFA

>=150 and <175 >=175 >=125 and <150 <125

tenermos a los jugadores que pesan entre 125 y 150. Por ultimo lugar y con casi nada del porcentaje, tenemos a jugadores que pesan menos de 125 lbs lo cual representa un 0.20% del total de datos.

En dicho gráfico podemos notar que la mayoría de jugadores estan entre un peso de 150 a 169 lbs, le sigue el peso de 175 lbs o superior y en tercero lugar

# Segundo enunciado: Generar un gráfico de barras (histograma) de acuerdo a su habilidad (overall) en base a los siguientes segmentos contando el número de jugadores:

4050

506070

708090

• 100

# Solución

Para este enunciado se utiliza el poder de pandas utilizando la función groupby con la finalidad de agrupar los datos por su overall y hacer un count de cada uno de dichos valores y así obtener el count de cada overall. A continuación se muestra el resultado

df\_overall.columns = ['count']
df\_overall.reset\_index(inplace=True)
df\_overall.head()

In [6]: | df\_overall = pd.DataFrame(df.groupby(['overall\_rating'])['overall\_rating'].count())

# Out[6]:

47 21
48 33
49 45
50 98
51 120
Por ultimo, una vez realizado el groupby y obtenido los resultados, procedemos a graficar el histograma tomando en cuenta los segmentos especificados en

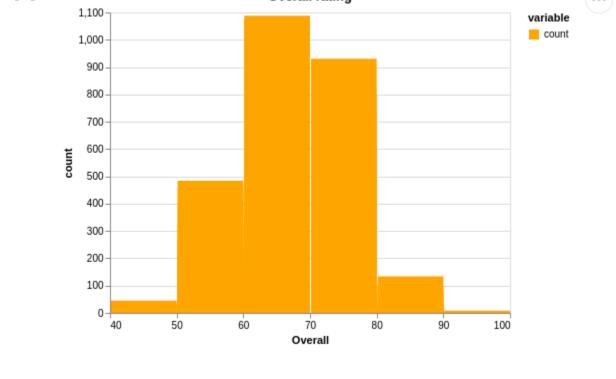
overall\_rating count

In [7]: alt.Chart(df\_overall\_melt('overall\_rating')).mark\_bar().encode(

el enunciado. Con la libraría altair, es muy facil gráficar de una manera interactiva el histograma y ademas de ello se tiene un mejor diseño a comparación de

x=alt.X('overall\_rating', bin=alt.Bin(maxbins=5), title='Overall'),
 y=alt.Y('value', title='count'),
 color=alt.Color('variable', scale=alt.Scale(domain=['count'], range=['orange']))
).properties(title='Overall rating').interactive()

Out[7]:
 Overall rating
 variable





# **Uso de Papermill para parametrizar cuadernos Jupyter**

**Autor:** Bryam David Vega Moreno Maestro: Diego Quisi Materia: Simulación Universidad: Universidad Politécnica Salesiana Carrera: Ciencias de la computación

### Introducción, Instalación y uso

Los cuadernos de jupyter actualmente son un estandar muy utilizado para realizar análisis exploratorios (EDA) y a su vez es una excelente herramienta para documentar proyectos de Ciencia de Datos.

#### Pros

- Compartible
- · Facil de leer Documentación con código
- Salidas como resultados
- Interfaz familiar Multi-lenguaje

### Contras

- Dificultad para realizar Test
- Dificil de parametrizar Documento mutable

Con todo estos pros y contras en cuenta, nace la necesidad de parametrizar variables con la finalidad de generar reportes de jupyter de mánera más dinámica. Para ello nace papermill





## Qué es papermill y por que se necesita?

Papermill es una herramienta que nos permite parametrizar y ejecutar cuadernos jupyter. Esta herramienta transforma tu jupyter notebook en una herramienta de data workflow, va ejecutando cada celda secuencialmente sin la necesidad de tener jupyterLab (o notebook) abierto. Papermill te pemite a ti:

- Parametrizar cuadernos
- Ejecutar y recolectar metricas a traves de cuadernos
- Resumir lo recolectado de tus cuadernos Usa los cuadernos como llamar funciones

#### **Como instalarlo?**

Para poder instalar papermill solamente se debe escribir la siguiente linea de comando:

\$ pip install papermill

Con papermill instalado, existen varias maneras de ejecutar dicha herramienta

### Como usarlo?

# **Ejecutando via Python API**

```
import papermill as pm
pm.execute_notebook(
    'path/to/input.ipynb',
    'path/to/output.ipynb',
    parameters = dict(a=2,b=3)
```

#### **Ejecutando via Cliente**

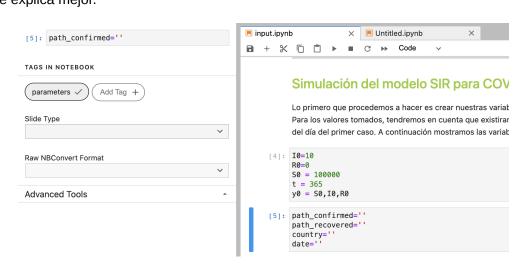
En este ejemplo estamos ejecutando un archivo local y tendra como salida un cuaderno nuevo en una cuenta de amazon S3:

\$ papermill local/input.ipynb s3://bkt/output.ipynb -p a 2 -p b 3

Para este ejemplo que mostraremos a continuación vamos a mostrar como parametrizar valores y ejecutar el cuaderno parametrizado para obtener un nuevo cuaderno como un reporte. Para este caso utilizaremos la simulación del modelo SIR utilizando los datos de COVID-19. A continuación explicamos como realizamos este ejemplo.

### Creación del cuaderno jupyter parametrizado utilizando jupyterLab

Para poder realizar este cuaderno, simplemente creamos un nuevo jupyter notebook y procedemos a crear los tags para poder parametrizar el cuaderno. Para poder esto en jupyterLab simplemente nos dirijimos a la celda a la que le vamos a poner el tag y ponemos como variables los parámetros del cuaderno. Con la siguiente imagen se explica mejor.



Como podemos ver en la imagen que se muestra, en la celda con número de ejecución 5 podemos ver que creamos un tag llamado "parameters", por lo tanto cuando vayamos a utilizar papermill, tendremos que llamar al tag parameters y pasar como parámetros las variables que estan en esa celda, con esto quedá más claro el uso de papermill en jupyterLab. Después de hacer ello, lo único que hacemos es realizar el modelo SIR para poder obtener nuestros resultados. A continuación mostramos como vamos a ejecutar nuestro cuaderno parametrizado, para obtener un cuaderno ejecutado con dichos parámetros.

# Usando papermill para ejecutar el cuaderno creado

Para poder ejecutar dicho cuderno, haremos uso de papermill, lo único que debemos hacer es crear un archivo python que puede tener cualquier nombre y escribir el siguiente código:

```
import papermill as pm
pm.execute_notebook(
    'input.ipynb',
    'output.ipynb',
   parameters = dict(path_confirmed='./in/time_series_covid19_confirmed_global.csv',
                    path_recovered='./in/time_series_covid19_recovered_global.csv',
                    country='Ecuador',
                    date='3/1/20')
```

Como podemos darnos cuenta estamos diciendo que vamos a ejecutar el cuaderno input.ipynb el cual esta parametrizado y después ejecutar el cuaderno y guardar los resultados en un cuaderno llamado output.ipynb. Posteriormente nos damos cuenta que utilizamos el tag parameters para mandar los parametros de ese tag OJO: Podemos tener varios tags, pero en este caso solo utilizamos un tag. Una vez realizado eso ejecutamos la siguiente línea de comando para ejecutar el programa.

\$ python name\_file.py

Con ello vamos a obtener una salida en la terminal como esta: \$ python main.py

```
Executing: 100%
5 [05:52<00:00, 6.40s/cell]
```

Una vez ejecutado el programa lo único que hacemos es verificar que se genero correctamente el archivo de salida. En la siguiente imagen presentamos el resultado obtenido.

```
path_confirmed='
path_recovered='
path_confirmed = "./in/time_series_covid19_confirmed_global.csv"
path_recovered = "./in/time_series_covid19_recovered_global.csv"
date = "3/1/20"
```

Como podemos ver, en el archivo **output.ipynb** se le asigna a cada variable del tag los parámetros mandados en el papermill. Con ello generamos un nuevo cuaderno con resultados como los que estan presentados en esta imagen.



un cuaderno del modelo SIR para diferentes paises y así agilizar disminuir el trabajo de ejecutar a cada momento de manera personal el cuaderno una y otra vez.

## **Conclusiones** Mediante este trabajo hemos revisado nuevas herramientas de visualización con la finalidad de ver nuevas opciones para graficar datos y dejar matplotlib de

lado poco a poco. Por otro lado hemos visto la gran potencialidad de papermill al momento de parametrizar cuadernos facilitando la ejecución del cuaderno con diferentes valores y obtener reportes diferentes con el mismo diseño pero con diferentes resultados.

# Referencias

- datos de fifa players: https://data.world/raghav333/fifa-players información papermill:

https://www.datacouncil.ai/hubfs/DataEngConf/Data%20Council/Slides%20SF%2019/Notebooks%20as%20Functions%20with%20papermill.pdf