

**UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS**

*MAESTRIA EN DATA SCIENCE*

CURSO:

**GESTION DE DATOS**

**ANALISIS DE DATOS POKÉMON**

DOCENTE:

**OSCAR RAMOS PONCE**

APELLIDOS Y NOMBRES CODIGO

PISCO OSTOS TABITA 202010719

CANLLA LINARES JASHIR ALEJANDRO 202010597

ÑIQUE CHACÓN CARMEN 202010603

MERCADO GARCÍA RAÚL ANTONIO 202010561

OPORTO D'UGARD CÉSAR 208032582

**2021**

TABLA DE CONTENIDO

[INTRODUCCIÓN 4](#_Toc78238356)

[1 METODOLOGÍA 5](#_Toc78238357)

[1.1 FUENTE DE DATOS 5](#_Toc78238358)

[1.2 ALMACENAMIENTO 5](#_Toc78238359)

[1.3 PROCESAMIENTO 7](#_Toc78238360)

[1.3.1 Conexión a AWS EC2 desde R 7](#_Toc78238361)

[1.3.2 Datos perdidos e Imputación de datos perdidos 7](#_Toc78238362)

[1.3.3 Análisis univariado de valores atípicos (outliers) 7](#_Toc78238363)

[1.3.4 Análisis multivariado de valores atípicos (outliers) 7](#_Toc78238364)

[1.4 VISUALIZACIÓN DE DATOS 8](#_Toc78238365)

[1.5 DIAGRAMA DE ARQUITECTURA 8](#_Toc78238366)

[2 RESULTADO 9](#_Toc78238367)

[2.1 DATOS PERDIDOS E IMPUTACIÓN DE DATOS PERDIDOS 9](#_Toc78238368)

[2.2 ANÁLISIS UNIVARIADO DE VALORES ATÍPICOS (OUTLIERS) 10](#_Toc78238369)

[2.3 ANÁLISIS MULTIVARIADO DE VALORES ATÍPICOS (OUTLIERS) 12](#_Toc78238370)

[2.4 RESULTADO DE DASHBOARD SHINY 14](#_Toc78238371)

[3 CONCLUSIONES 17](#_Toc78238372)

[4 BIBLIOGRAFÍA 18](#_Toc78238373)

[5 ANEXOS 19](#_Toc78238374)

[5.1 CODIGO DE CONEXIÓN A AWS EC2 19](#_Toc78238375)

[5.2 CODIGO DE ANALISIS DE DATOS R 21](#_Toc78238376)

[5.3 CODIGO DE DASHBOARD SHINY 24](#_Toc78238377)

**TABLA DE ILUSTRACIÓN**

[Ilustración 1. Data set Pokémon 5](#_Toc78150668)

[Ilustración 2. Máquina Virtual AWS EC2 5](#_Toc78150669)

[Ilustración 3: Datos de Servidor Ubuntu 6](#_Toc78150670)

[Ilustración 4.Almacenamiento EC2 - Servidor Ubuntu 6](#_Toc78150671)

[Ilustración 5. Configuración de Puertos AWS EC2 - Servidor Ubuntu 6](#_Toc78150672)

[Ilustración 6.Creación de tablas MySQL 6](#_Toc78150673)

[Ilustración 7. Conexión a AWS EC2 desde R 7](#_Toc78150674)

[Ilustración 8: Datos perdidos 7](#_Toc78150675)

[Ilustración 9.Arquitectura Solución de Pokémon 8](#_Toc78150676)

[Ilustración 10. Datos perdidos campo Type2 9](#_Toc78150677)

[Ilustración 11. Grafica Matrix plot 9](#_Toc78150678)

[Ilustración 12. Análisis de correlación de las variables 10](#_Toc78150679)

[Ilustración 13. Histograma de Variables cuantitativas 10](#_Toc78150680)

[Ilustración 14. Detección de Outliers con Puntaje Z 11](#_Toc78150681)

[Ilustración 15. Grafica de Box Plot ajustado 11](#_Toc78150682)

[Ilustración 16. BarPlot de Distancia de Mahalanobis 12](#_Toc78150683)

[Ilustración 17: Outliers detectados por el punto de corte Chi Cuadrado 12](#_Toc78150684)

[Ilustración 18. Identificación de Outliers con Grafica de Ojiva 13](#_Toc78150685)

[Ilustración 19. Detección de Outliers Multivariados 13](#_Toc78150686)

[Ilustración 20. Vista Dashboard Shiny - Bienvenida 14](#_Toc78150687)

[Ilustración 21..Vista Dashboard Shiny - Tipos de p Pokémon 14](#_Toc78150688)

[Ilustración 22. Vista Dashboard Shiny - Análisis Descriptivo 15](#_Toc78150689)

[Ilustración 23. Vista Dashboard Shiny - Probabilidad 15](#_Toc78150690)

[Ilustración 24. Vista Dashboard Shiny - Duelo Pokémon 15](#_Toc78150691)

# INTRODUCCIÓN

El presente proyecto se enfocará en el análisis de datos de la serie Pokémon, el cual es un anime metaserial que contiene diversos personajes, y especies de Pokémon con diferentes poderes de ataques y defensa. Asimismo, la serie se ha visto expandida en video juegos, juegos de cartas, películas, artículos de juegos y otros.

Esta información es importante para toda la comunidad de fans, debido que le permite tener información detalladas de los tipos de Pokémon y sus atributos principales de combate. Asimismo, se ha observado que existe 1,000 millones de descargas en Android y IOS del juego Pokémon Go, y 150 millones de jugadores mensuales aproximadamente (El país, 2019).

Para el análisis de datos se considera el data set POKEMON publicado en Kaggle (Michael Metter, 2019), el cual se subirá a la plataforma de Amazon Web Services - EC2 en un servidor de Ubuntu; el análisis se realizará con el software libre R y para la visualización de datos se desarrolla un dashboard interactivo con Shiny Web Apps y Flexdashboard.

Como parte del procesamiento de datos, se analiza los valores perdidos con gráficas de agregación y de matriz, se aplica imputación de datos perdidos, se analiza los valores atípicos a nivel univariado con las gráficas de histograma, cajas ajustadas (Hubert) y puntuación Z, y para los valores atípicos a nivel multivariado se utiliza la distancia de Mahalanobis con puntos de corte chi-cuadrado. Asimismo, se utilizaron gráficos de ojiva y de cuantil cuantil (QQ).

Finalmente, para la visualización de los datos se desarrolla un dashboard interactivo que contiene histogramas, gráficos de cajas, gráficos de dispersión y gráficos radiales, los cuales nos permite analizar por tipo de cada Pokémon.

# METODOLOGÍA

## FUENTE DE DATOS

El dataset a trabajar es la información de Pokémones las cuales provienen de la página de Kaggle.

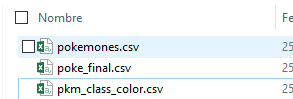


Ilustración 1. Data set Pokémon

Los datos son estructurados, se muestra la descripción y tipo de dato de las tablas:

Tabla: pokemones

|  |  |  |  |
| --- | --- | --- | --- |
| Campo | Descripción de campo | Tipo de Variable | Tipo de dato |
| Id | Clave primaria auto incremental | Entero | INTEGER |
| Name | El nombre en inglés del Pokémon | Categórico | VARCHAR(100) |
| Type1 | El tipo principal de Pokémon | Categórico | VARCHAR(100) |
| Type2 | El tipo secundario de Pokémon | Categórico | VARCHAR(100) |
| HP | El HP base del Pokémon | Entero | INTEGER |
| Attack | El ataque base de los Pokémon | Entero | INTEGER |
| Defense | La defensa base de los Pokémon | Entero | INTEGER |
| SpAtk | El ataque especial base del Pokémon | Entero | INTEGER |
| SpDef | La Defensa Especial Base del Pokémon | Entero | INTEGER |
| Speed | la velocidad base del Pokémon | Entero | INTEGER |
| generación | la generación numerada en la que se introdujo por primera vez el Pokémon | Entero | INTEGER |
| Legendary | indica si el Pokémon es legendario. | Binario | VARCHAR(100) |

## ALMACENAMIENTO

Para el almacenamiento de datos, se utilizó el servicio de Amazon Web Service de EC2:



Ilustración 2. Máquina Virtual AWS EC2

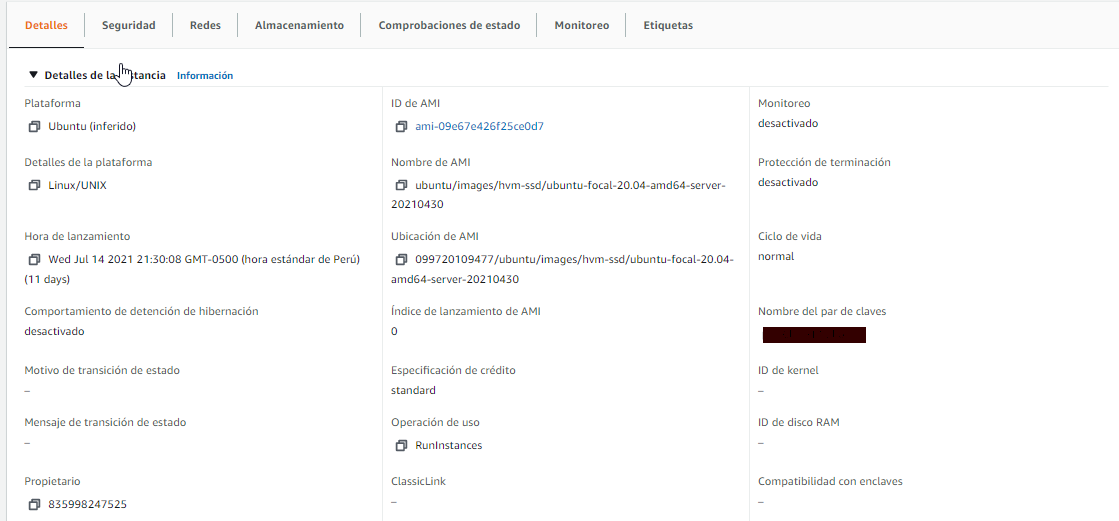


Ilustración 3: Datos de Servidor Ubuntu

1. Se creó un máquina virtual Ubuntu con 8GB de almacenamiento

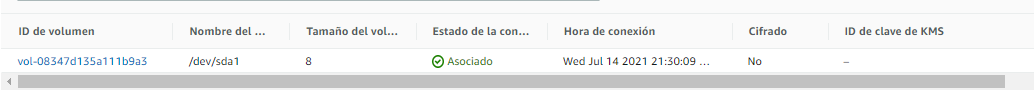


Ilustración 4.Almacenamiento EC2 - Servidor Ubuntu

1. Se habilitaron puertos para SSH y MySQL de acceso público (sin restricciones).



Ilustración 5. Configuración de Puertos AWS EC2 - Servidor Ubuntu

1. En el Servidor EC2 se instaló la base de datos MYSQL.
2. Configuraciones en la máquina local para la carga del archivo .csv en la plataforma de AWS EC2

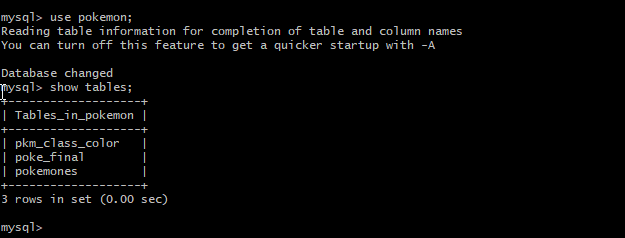


Ilustración 6.Creación de tablas MySQL

## PROCESAMIENTO

### Conexión a AWS EC2 desde R

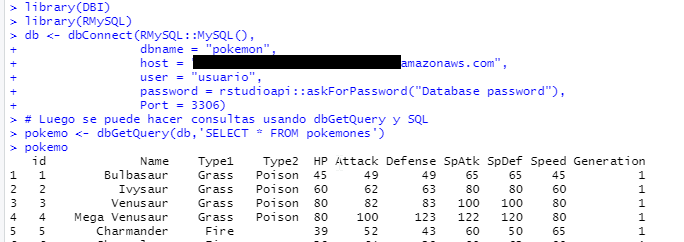


Ilustración 7. Conexión a AWS EC2 desde R

### Datos perdidos e Imputación de datos perdidos

Se observa que la tabla tiene datos perdidos en la variable Type2 (386 registros), los cuales se imputaron con la descripción **“no aplica”,** debido que hay pokemones que son de un tipo en específico.

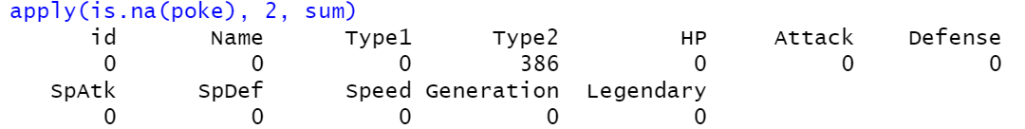


Ilustración 8: Datos perdidos

Para las variables cualitativas del dataset se transformó en variables cuantitativas los campos: **Type1, Type2 y Legendary** para poder utilizar los gráficos de agregación y matriz.

### Análisis univariado de valores atípicos (outliers)

Se analizó las variables de forma independiente con gráficos de histogramas para determinar las distribuciones, luego se utilizó los gráficos de cajas de Hubert y por último se aplica la metodología de puntaje Z para la detección de variables atípicas (outliers).

### Análisis multivariado de valores atípicos (outliers)

Se calcula la distancia de Mahalanobis al cuadrado donde se determinó un umbral de K=3 para la detección de outliers multivariados. También se estimó un punto de corte asociado a la distribución chi cuadrado considerando el 99.9% de la data.

Por otra parte, de forma visual se utiliza la gráfica de ojiva y el grafico cuantil cuantil donde se observarían los mismos resultados.

## VISUALIZACIÓN DE DATOS

Para poder realizar la visualización de los datos, se desarrolló un dashboard usando Shiny Web Apps y FlexDashboard en R Studio, el cual contiene 4 secciones:

* Introducción
* Tipos de Pokémon
* Data por Clase
* Análisis Descriptivo
* Probabilidad
* Duelo Pokémon

## DIAGRAMA DE ARQUITECTURA

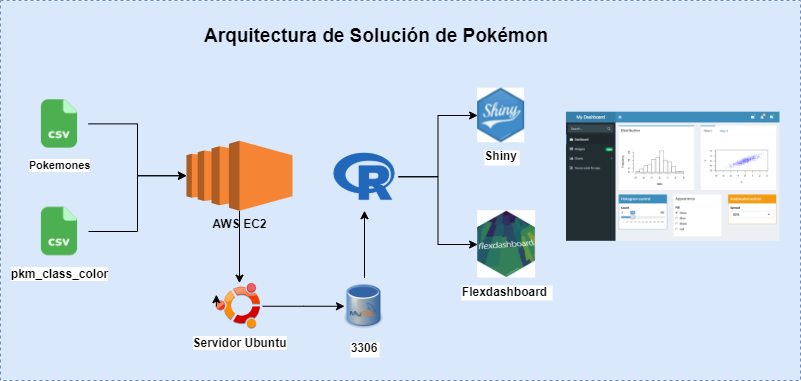
****

Ilustración 9.Arquitectura Solución de Pokémon

# RESULTADO

## DATOS PERDIDOS E IMPUTACIÓN DE DATOS PERDIDOS

**Diagrama de Agregación**

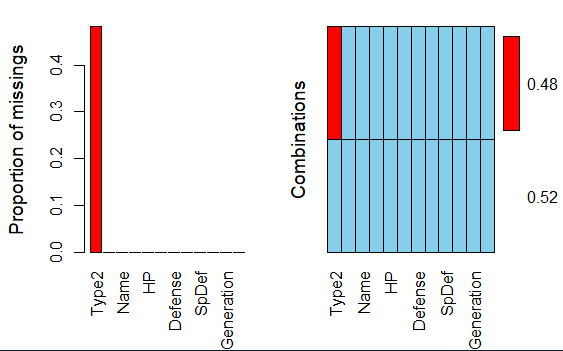


Ilustración 10. Datos perdidos campo Type2

**Interpretación:** Se observa que el 48% de la variable **“Type2”** son valores perdidos.

**Matrix Plot**

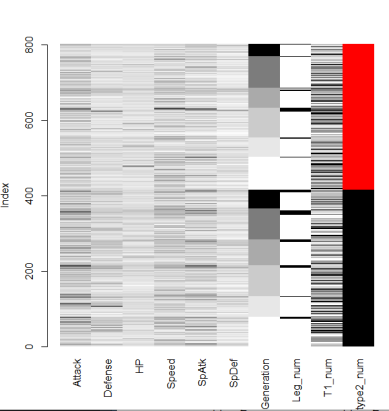


Ilustración 11. Grafica Matrix plot

**Interpretación:** Se observa que no hay ningún patrón de los valores perdidos por ende se dice que son completamente aleatorios.

## ANÁLISIS UNIVARIADO DE VALORES ATÍPICOS (OUTLIERS)

**Gráfico de correlación Bivariada**

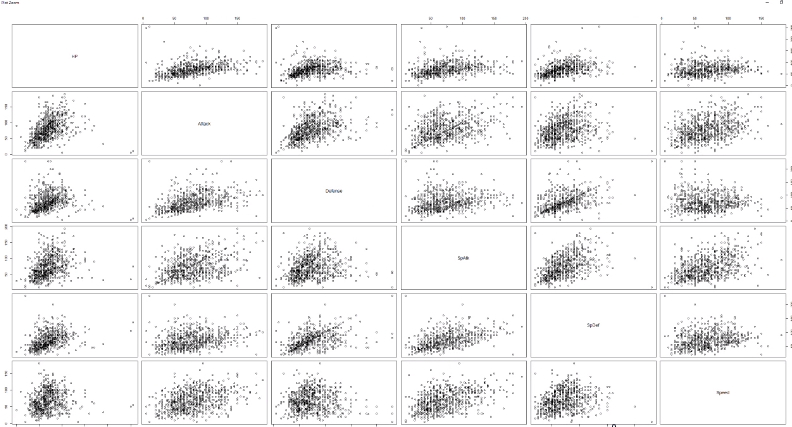


Ilustración 12. Análisis de correlación de las variables

**Interpretación:** Las variables cuantitativas del dataset no tienen una relación fuerte, es decir no hay una clara correlación entre cada par de variables.

**Histograma**

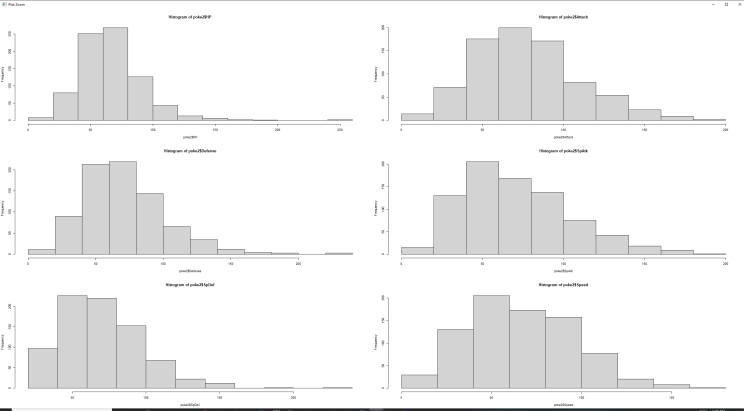


Ilustración 13. Histograma de Variables cuantitativas

Interpretación: Se observa que el histograma tiene una asimetría positiva y probablemente se vengan de una distribución normal, por tanto se aplicaría la regla de Puntuación Z.

**Outliers usando la Puntuación Z**

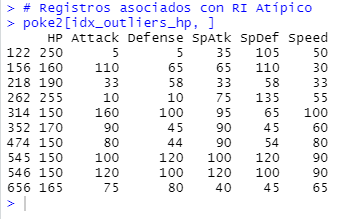


Ilustración 14. Detección de Outliers con Puntaje Z

**Interpretación:** De la evaluación de la metodología de detección de outliers por puntuación Z, se observa 10 registros outliers para la variable HP.

**Grafica de Box Plot**

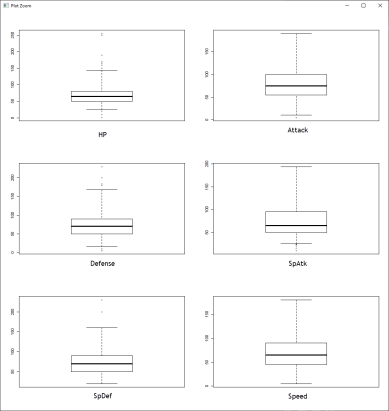


Ilustración 15. Grafica de Box Plot ajustado

**Interpretación**: Dado que la distribución de las variables es asimétrica se utiliza el grafico de cajas ajustado en donde se observa que para las variables Hp y Defense se registran valores atípicos confirmando lo encontrado en la metodología de puntuación Z.

## ANÁLISIS MULTIVARIADO DE VALORES ATÍPICOS (OUTLIERS)

**Mahalanobis**

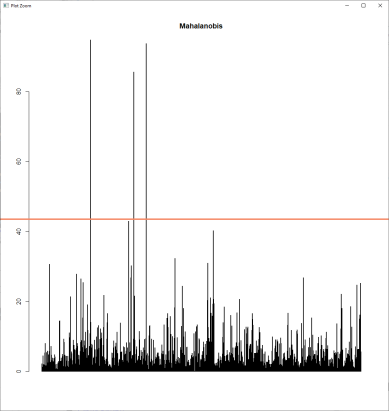


Ilustración 16. BarPlot de Distancia de Mahalanobis

Interpretación: De acuerdo a la distancia de Mahalanobis al cuadrado se observa que 3 registros despuntan como posibles outliers multivariados.

**Punto de corte Chi cuadrado**

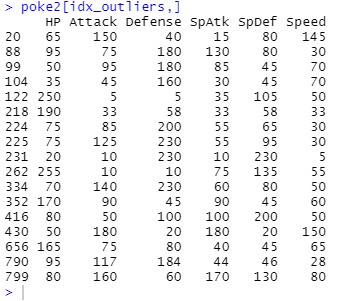


Ilustración 17: Outliers detectados por el punto de corte Chi Cuadrado

**Interpretación:** Según la metodología de punto de corte chi cuadrado se deben evaluar las siguientes observaciones de la ilustración 17 como registros atípicos multivariados

**Grafica Ojiva**

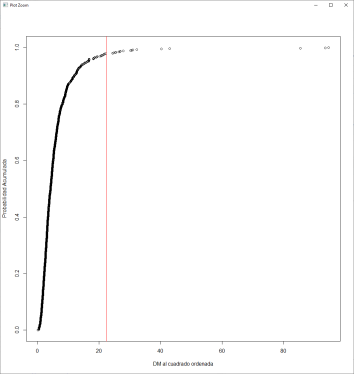


Ilustración 18. Identificación de Outliers con Grafica de Ojiva

Interpretación: gráficamente se pueden observar los registros con posibilidad de ser outliers multivariados (Se considera el 99.9% de la distribución).

**Grafica Cuantil Cuantil (qq)**

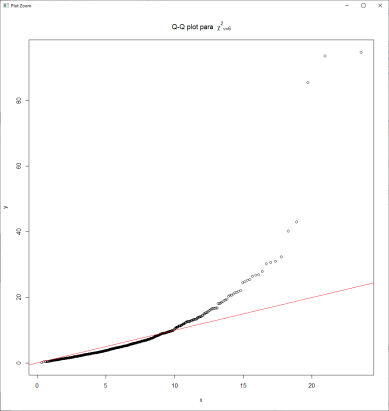
+

Ilustración 19. Detección de Outliers Multivariados

Interpretación: Se observa de igual manera con el gráfico QQ-PLOT de chi cuadrado la presencia de registros outliers multivariados.

## RESULTADO DE DASHBOARD SHINY

* **Introducción.** Aquí se visualiza el menú principal que permite conocer el resto de la información dentro del dashboard.

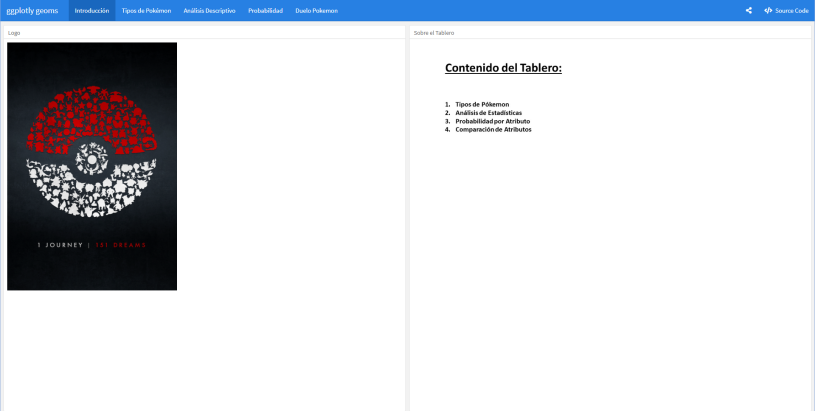


Ilustración 20. Vista Dashboard Shiny - Bienvenida

* **Tipos de Pokémon.** Se visualizan diagramas de barras por tipos de Pokémon. Esto permite conocer la cantidad de pokemones por tipo en el dataset. Cada Pokémon puede contener dos tipos de habilidades, las cuales son presentadas en estos diagramas.

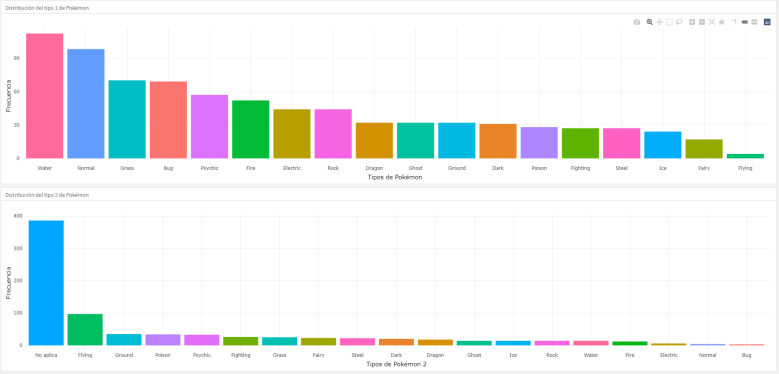


Ilustración 21..Vista Dashboard Shiny - Tipos de p Pokémon

* **Análisis Descriptivo.** Aquí se visualizan los atributos básicos, de batalla y el resumen de las características de la familia a la que pertenece cada Pokémon.

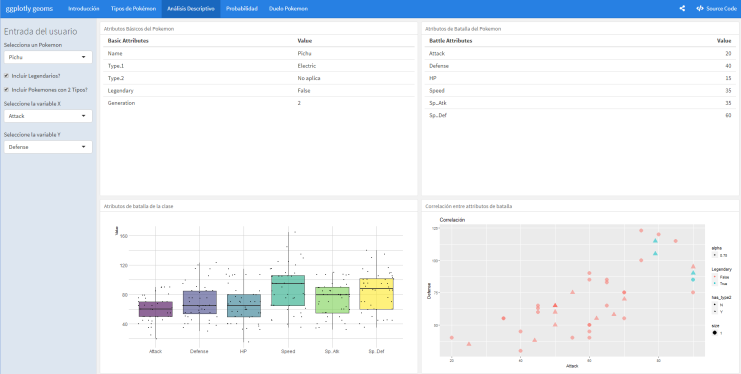


Ilustración 22. Vista Dashboard Shiny - Análisis Descriptivo

* **Mapa de Calor por Variable y Tipo.** Aquí se visualizan los atributos de batalla por cada tipo de Pókemon. Tanto numéricamente como en un mapa de calor.

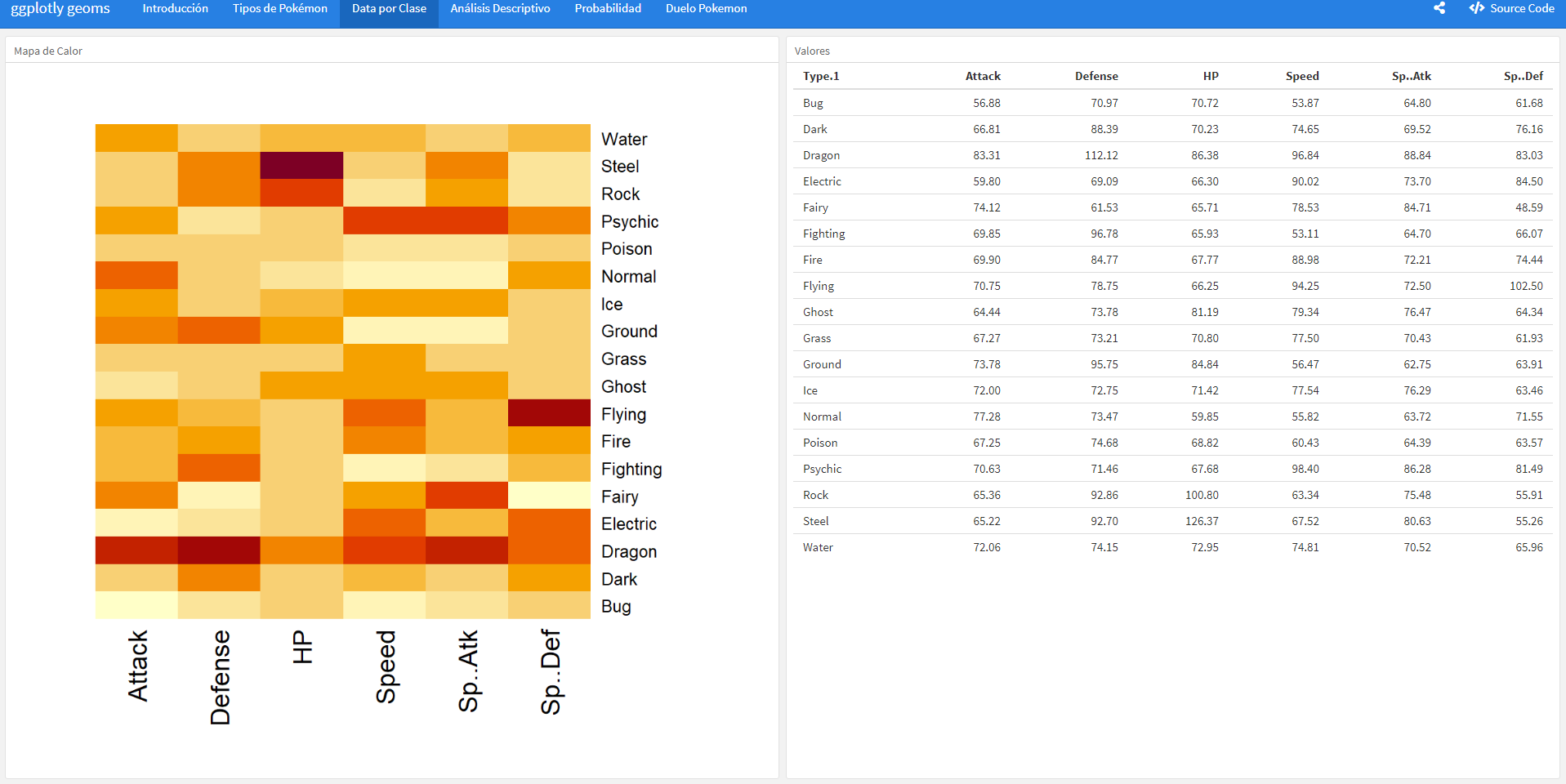


Ilustración 23. Vista Dashboard Shiny – Mapa de Calor por Variable y Tipo

* **Probabilidad.** Se muestran los histogramas de cada atributo del Pokémon. Cabe indicar que solo se han considerado valores cuantitativos.

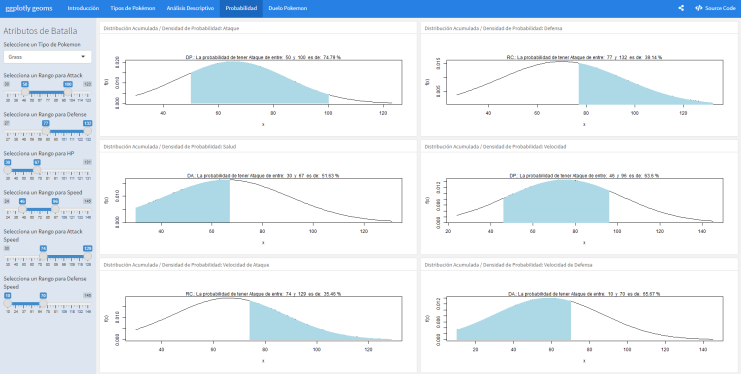


Ilustración 24. Vista Dashboard Shiny - Probabilidad

* **Duelo Pokémon.** Se muestra un diagrama radial en el que se resaltan los atributos de cada Pokémon en un escenario de combate. Asimismo, se visualizan las imágenes de los pokemones.

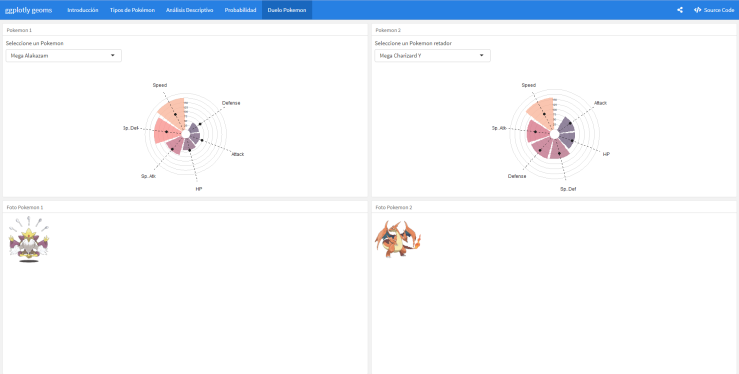


Ilustración 25. Vista Dashboard Shiny - Duelo Pokémon

# CONCLUSIONES

* Los datos de 799 Pokémon contaban con 386 valores perdidos los cuales fueron imputados creando una categoría más al tipo2 de los mismos (Categoría: No aplica). Esto quiere decir que 386 ejemplares de Pokémon tienen tan solo 1 tipo de poder mientras el resto tienen hasta 2 tipos diferentes de poder.
* Se encontraron valores atípicos Univariados y multivariados, sin embargo, como parte de la naturaleza de la información no se decidieron quitarlos debido que existen Pokémones mucho más fuertes que el resto.
* Se encontraron 18 tipos de pokémones diferentes dentro de los cuales el más frecuente son los pokémones de tipo Agua con 112 ejemplares y le sigue los de tipo normal con 98 ejemplares.
* Existen Pokémon legendarios en determinados tipos, por ejemplo, los tipos de Dark, Dragon, Eléctrico, Fantasmas, Hielo, Fuego, etc. Siendo los de categoría Psiquicos el tipo de Pokémon que tiene más ejemplares legendarios (14), seguido del tipo Dragón (12 ejemplares).
* En promedio el tipo de Pokémon que tiene mayores puntos de Defensa son los de tipo Agua con 126.3 puntos, seguidos de los Pokémon de roca con 100.8 puntos.
* En promedio el tipo de Pokémon que tiene mayores puntos de Velocidad son los de tipo Volador con 102.5 puntos, seguidos de los Pokémon de eléctricos con 84.5 puntos.
* En promedio el tipo de Pokémon que tiene mayores puntos de Ataque son los de tipo Dragón con 112.1 puntos, seguidos de los Pokémon de Pelea con 96.8 puntos.
* En promedio el tipo de Pokémon que tiene mayores puntos de Vida son los de tipo Dragón con 83.3 puntos, seguidos de los Pokémon de tipo normal con 77.2 puntos.
* En promedio el tipo de Pokémon que tiene mayores puntos de velocidad de ataque son los de tipo Psíquico con 98.4 puntos, seguidos de los Pokémon de tipo Dragón con 96.8 puntos.
* En promedio el tipo de Pokémon que tiene mayores puntos de velocidad de Defensa son los de tipo Dragón con 88.8 puntos, seguidos de los Pokémon de tipo Psíquico con 86.3 puntos.

# BIBLIOGRAFÍA

* Michael Metter. (mayo del 2019). Pokémon Data Analysis

<https://www.kaggle.com/mmetter/pokemon-data-analysis-tutorial>

* El País. (septiembre del 2019). Por qué juegan a Pokémon GO millones de personas tres años después de su lanzamiento

<https://elpais.com/tecnologia/2019/09/19/actualidad/1568906241_530163.html>

* Amazon Web Service – EC2

<https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Home>:

* Tydiverse- GGplot2

<https://ggplot2.tidyverse.org/>

* The R Graph Gallery

<https://www.r-graph-gallery.com/index.html>

* Irizarry, R. A. (2019). Introduction to data science: Data analysis and

prediction algorithms with R. CRC Press.

* Horgan, J. M. (2019). Probability with R: an introduction with

computer science applications. John Wiley & Sons.

* Ugarte, M. D., Militino, A. F., & Arnholt, A. T. (2016). Probability and

Statistics with R. CRC press.

* Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). Database system concepts (Vol. 4). New York: McGraw-Hill.
* Hoffer, J. A., Topi, H., & Venkataraman, R. (2013). Essentials of Database Management. Pearson Higher Ed.
* Erl, T., Puttini, R., & Mahmood, Z. (2013). Cloud computing: concepts, technology, & architecture. Pearson Education.

# ANEXOS

## CODIGO DE CONEXIÓN A AWS EC2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CONECTARSE A AWS EC2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- Ubicarse en la ruta donde se tienen los archivos d clave - csv y otros.

cd Documents

ssh -i "Clave14julio.pem" ubuntu@ec2-18-207-181-49.compute-1.amazonaws.com

yes

sudo apt update

sudo apt upgrade

Y

===========================================================

-- Para verificar si esta instalado

systemctl status mysql.service

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Crear una DB y Tabla MYSQL\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

sudo mysql

create database pokemon;

use pokemon;

CREATE TABLE pokemones (

id INTEGER AUTO\_INCREMENT,

Name VARCHAR(100),

Type1 VARCHAR(100),

Type2 VARCHAR(100),

HP INTEGER,

Attack INTEGER,

Defense INTEGER,

SpAtk INTEGER,

SpDef INTEGER,

Speed INTEGER,

Generation INTEGER,

Legendary VARCHAR(100),

PRIMARY KEY (id)

);

CREATE TABLE pkm\_class\_color (

nombre\_poke VARCHAR(100),

color\_class VARCHAR(100),

PRIMARY KEY (nombre\_poke)

);

CREATE TABLE poke\_final (

id INTEGER AUTO\_INCREMENT,

Name VARCHAR(100),

Type1 VARCHAR(100),

Type2 VARCHAR(100),

HP INTEGER,

Attack INTEGER,

Defense INTEGER,

SpAtk INTEGER,

SpDef INTEGER,

Speed INTEGER,

Generation INTEGER,

Legendary VARCHAR(100),

PRIMARY KEY (id)

);

exit;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SUBIR ARCHIVO CSV A AWS EC2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Importar archivo:Se tiene que regresar al directorio origen para copiar a servidor (Documentos)

Antes de subir los archivos se debe eliminar la cabecera

Propietario@DESKTOP-71B4IOR MINGW64 ~/Documents

-- Se tiene que ingresar desde documentos para esta opción:

scp -i "Clave14julio.pem" pokemones.csv ubuntu@ec2-18-207-181-49.compute-1.amazonaws.com:/home/ubuntu

scp -i "Clave14julio.pem" pkm\_class\_color.csv ubuntu@ec2-18-207-181-49.compute-1.amazonaws.com:/home/ubuntu

scp -i "Clave14julio.pem" poke\_final.csv ubuntu@ec2-18-207-181-49.compute-1.amazonaws.com:/home/ubuntu

--Comectarse a Ubuntu:

ssh -i "Clave14julio.pem" ubuntu@ec2-18-207-181-49.compute-1.amazonaws.com

Desde la instancia EC2: Copiar el archivo a la ruta indicada

- Copiar el archivo:

sudo cp pokemones.csv /var/lib/mysql-files/

sudo cp pkm\_class\_color.csv /var/lib/mysql-files/

sudo cp poke\_final.csv /var/lib/mysql-files/

-- Ingresar a base de datos

sudo mysql

use pokemon;

-- Para cargar la tabla CSV a AWS EC2

LOAD DATA INFILE '/var/lib/mysql-files/pokemones.csv' INTO TABLE pokemones COLUMNS TERMINATED BY ';' LINES TERMINATED BY '\n';

LOAD DATA INFILE '/var/lib/mysql-files/pkm\_class\_color.csv' INTO TABLE pkm\_class\_color COLUMNS TERMINATED BY ',' LINES TERMINATED BY '\n';

LOAD DATA INFILE '/var/lib/mysql-files/poke\_final.csv' INTO TABLE poke\_final COLUMNS TERMINATED BY ';' LINES TERMINATED BY '\n';

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Crear usuario MySQL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CREATE USER usuario@'%' IDENTIFIED WITH mysql\_native\_password BY 'User1234+';

GRANT ALL PRIVILEGES ON \*.\* TO usuario@'%' WITH GRANT OPTION;

- Salir de MySQL y desde el terminal:

• Abrir el archivo mysqld.cnf:

sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf

• Comentar (con #) la línea que contiene “bin-address 127.0.0.1”:

#bind-address 127.0.0.1

• Reiniciar el servicio de MySQL:

sudo service mysql restart

## CODIGO DE ANALISIS DE DATOS R

#=====================================================================

#El host se muestra en el "dashboard" de la instancia como "DNS de IPv4 p?blica"

library(DBI)

library(RMySQL)

library(VIM)

db <- dbConnect(RMySQL::MySQL(),

dbname = "pokemon",

host = "ec2-18-207-181-49.compute-1.amazonaws.com",

user = "usuario",

password = rstudioapi::askForPassword("Database password"),

Port = 3306)

#Clave: User1234+

# Luego se puede hacer consultas usando dbGetQuery y SQL

poke <- dbGetQuery(db,'SELECT \* FROM pokemones')

poke

###### Realizando analisis univariado #############

summary(poke)

class(poke)

# Conversi?n de valores perdidos a NA (por defecto est?n como 0)

poke$Type2[poke$Type2==""] <- NA

# Revisando datos perdidos #

apply(is.na(poke), 2, mean)

apply(is.na(poke), 2, sum)

# Visualizando graficamente

aggr(poke,number=TRUE, sortComb=TRUE, sortVar=TRUE, only.miss=TRUE)

# para numericas

poke$type2\_num<-ifelse(is.na(poke$Type2),NA,1)

poke$type1\_num<-ifelse(is.na(poke$Type2),NA,1)

library(dplyr)

poke$T1\_num=case\_when(

poke$Type1 == 'Bug' ~ 1,

poke$Type1 == 'Dark' ~ 2,

poke$Type1 == 'Dragon' ~ 3,

poke$Type1 == 'Electric' ~ 4,

poke$Type1 == 'Fairy' ~ 5,

poke$Type1 == 'Fighting' ~ 6,

poke$Type1 == 'Fire' ~ 7,

poke$Type1 == 'Flying' ~ 8,

poke$Type1 == 'Ghost' ~ 9,

poke$Type1 == 'Grass' ~ 10,

poke$Type1 == 'Ground' ~ 11,

poke$Type1 == 'Ice' ~ 12,

poke$Type1 == 'Normal' ~ 13,

poke$Type1 == 'Poison' ~ 14,

poke$Type1 == 'Psychic' ~ 15,

poke$Type1 == 'Rock' ~ 16,

poke$Type1 == 'Steel' ~ 17,

poke$Type1 == 'Water' ~ 18

)

poke$Leg\_num=case\_when(

poke$Legendary == 'False\r' ~ 0,

poke$Legendary == 'True\r' ~ 1)

names(poke)

table(poke$Legendary)

pks\_btl\_attr<- c('Attack','Defense','HP','Speed','SpAtk','SpDef','Generation', 'Leg\_num','T1\_num','type2\_num')

poke[,pks\_btl\_attr]

x11()

matrixplot(poke[,pks\_btl\_attr])

# Immputando los NA (debido a que estos NA corresponden aquellos que no tienen poder TIPO2 creamos una nueva categoria)#

poke[is.na(poke)] <- "No aplica"

# Analisis de Outliers #

library(ggplot2)

# ExploraciÃÂ³n de Datos

# Para determinar si se hace un anÃÂ¡lisis de outlier por chi-square o Mahalanobis

poke2 = poke[-c(1,2,3,4,11,12,13,14,15,16)]

head(poke2,5)

summary(poke2)

boxplot(poke2)

pairs(poke2)

par(mfrow = c(3,2))

hist(poke2$HP)

hist(poke2$Attack)

hist(poke2$Defense)

hist(poke2$SpAtk)

hist(poke2$SpDef)

hist(poke2$Speed)

# Outliers usando la PuntuaciÃÂ³n Z

# """""""""""""""""""""""""""""""

is.outlier\_z <- function(x, k=3) {

return(abs(scale(x)) > k) # scale: (x-media)/desv\_est

}

is.outlier\_z(poke2)

################# Variable HP ####################

# ÃÂndices (T/F) del Rango Intercuartilico

idx\_outliers\_hp <- is.outlier\_z(poke2$HP, k=3)

which(idx\_outliers\_hp)

# RI atÃÂ­picos

poke2$HP[idx\_outliers\_hp]

# Registros asociados con RI AtÃÂ­pico

poke2[idx\_outliers\_hp, ]

# SegÃÂºn la puntuaciÃÂ³n Z se identifican 3 observaciones con valores atÃÂ­picos

# para Al, con K = 3 desv estÃÂ¡ndar.

# Al:

par(mfrow=c(1,1))

#################################################

install.packages('robustbase')

library(robustbase)

# RI:

par(mfrow=c(3,2))

adjbox(poke2$HP) # Se observan 2 observaciones outliers

adjbox(poke2$Attack) # Se observan 1 observaciones outliers

adjbox(poke2$Defense) # Se observan 1 observaciones outliers

adjbox(poke2$SpAtk) # Se observan 4 observaciones outliers

adjbox(poke2$SpDef) # Se observan 2 observaciones outliers y algunas observaciones con valores atÃÂ­picos

adjbox(poke2$Speed) # Se observan 4 observaciones outliers

#Evaluar valores outlier con la distancia de Mahalanobis

#a) calculamos la distancia de mahalanobis para nuestros datos.

par(mfrow=c(1,1))

dm2 <- mahalanobis(poke2, colMeans(poke2), cov(poke2))

barplot(dm2, main="Mahalanobis")

which.max(dm2)

# DistribuciÃÂ³n Chi-Cuadrado: Punto de Corte

p <- 1-0.001

dof = ncol(poke2)

k <- (qchisq(p, dof))

idx\_outliers <- which(dm2 > k)

idx\_outliers

poke2[idx\_outliers,]

#Con el 99.9% de la distribuciÃÂ³n se encuentran 5 observaciones posiblemente

#outliers multivariados.

# GrÃÂ¡fico de Ojiva

plot(sort(dm2), ppoints(nrow(poke2)), xlab="DM al cuadrado ordenada",

ylab="Probabilidad Acumulada")

abline(v = qchisq(p,dof), col = "red")

# QQ-plot:

x <- qchisq(ppoints(nrow(poke2)), dof)

y <- dm2

qqplot(x, y, main=expression("Q-Q plot para"~~{chi^2}[nu==6]))

abline(0, 1, col="red")

#Con estos grÃÂ¡ficos se confirma la existencia de observaciones outliers multivariadas.

#Se crea la tabla final la cual serÃ¡ importada a la instancia EC2

poke3=poke[,c(1:12)]

poke3$Legendary=case\_when(

poke$Legendary == 'False\r' ~ 'False',

poke$Legendary == 'True\r' ~ 'True')

write.csv(poke3,"poke\_fin.csv")

################################# FIN

## CODIGO DE DASHBOARD SHINY

---

title: "ggplotly geoms"

output:

flexdashboard::flex\_dashboard:

orientation: rows

social: menu

source\_code: embed

runtime: shiny

---

```{r setup, include=FALSE}

library(flexdashboard)

library(shiny)

library(tidyverse)

library(hrbrthemes)

library(viridis)

library(reshape2)

library (thematic)

library(ggplot2)

library(plotly)

library(dplyr)

library(e1071)

library(modeest)

library(DBI)

library(RMySQL)

library(VIM)

#Conexión a la BD de la máquina virtual EC2

db <- dbConnect(RMySQL::MySQL(),

dbname = "pokemon",

host = "ec2-18-207-181-49.compute-1.amazonaws.com",

user = "usuario",

#password = rstudioapi::askForPassword("Database password"),

password = 'User1234+',

Port = 3306)

# Se obtiene el arreglo de datos final desde la base SQL

# El tratamiento de los datos se realizó en la etapa previa de procesamiento y exploración

pkms<- dbGetQuery(db,'SELECT \* FROM poke\_final')

# Eliminar los caracteres que se incluyeron en el campo Legendary al subir el arreglo de datos final

pkms$Legendary=case\_when(

pkms$Legendary == 'False\r' ~ 'False',

pkms$Legendary == 'True\r' ~ 'True'

)

# Renombrar columnas para que coincidan con las del desarrollo inicial

names(pkms)<- c('id','Name','Type.1','Type.2','Attack','Defense','HP','Speed','Sp..Atk','Sp..Def','Generation','Legendary')

# incluir un campo genérico "has\_type2" para indicar que pókemones tienen un segund tipo.

# se utilizará posteriormente en el panel

pkms$has\_type2 <- ifelse(pkms$Type.2=='No aplica','N','Y')

# Se obtiene una lista de nombres de las variables categóricas

pks\_bsc\_attr<- c('Name','Type.1','Type.2','Legendary','Generation')

# Se obtiene una lista de nombres de las variables numéricas

pks\_btl\_attr<- c('Attack','Defense','HP','Speed','Sp..Atk','Sp..Def')

# Se transpone la dataframe original (pkms), sólo variables categóricas: atributos básicos

pkms\_bsc <- melt(

pkms,

id.vars='Name',

measure.vars=pks\_bsc\_attr,

variable.name='Basic Attributes',

value.name='Value',

na.rm=TRUE)

# Generar una matriz para comparar los promedios de las variables numericas de cada tipo de pokemon, se graficará posetriormente

btl <- pkms[,c('Type.1',pks\_btl\_attr)]

btl\_avr<- btl%>%

group\_by(Type.1)%>%

summarise(Attack=mean(Attack),

Defense=mean(Defense),

HP=mean(HP),

Speed=mean(Speed),

Sp..Atk=mean(Sp..Atk),

Sp..Def=mean(Sp..Def)

)

mtx <- as.matrix(btl\_avr[,pks\_btl\_attr])

row.names(mtx)<-btl\_avr$Type.1

# Se transpone la dataframe original (pkms), sólo variables numéricas: atributos de batalla

pkms\_btl <- melt(

pkms,

id.vars=c('Name','Type.1','Type.2','Legendary','has\_type2'),

measure.vars=pks\_btl\_attr,

variable.name='Battle Attributes',

value.name='Value',

na.rm=TRUE)

# Funcion para reindexar un DF

ridx <- function(df){

#Devuelve el dataframe de ingreso reindexado

row.names(df)<-1:nrow(df)

return(df)

}

# Funcion para obtener los promedios de cada columna numérica

get\_fields\_avgs <- function(df,nameOfNumCols){

avgrs<-c()

attbs\_num <-length(nameOfNumCols)

for (i in 1:attbs\_num){

avg<- sum(df[nameOfNumCols[i]])/nrow(df[nameOfNumCols[i]])

avgrs[i]<-avg

}

result <- as.data.frame(matrix(avgrs,1,attbs\_num))

names(result)<-nameOfNumCols

result

}

# Función para generar los gráficos de barras circulares de la sección 5 del panel

# transposed\_data: Data transpuesta, cada atributo/variable debe ser una fila

# crcl\_rad: Radio máximo del circulo

# Fuente: https://www.r-graph-gallery.com/index.html

circular\_bar\_plot <- function(transposed\_data,crcl\_rad){

units <- 25 # Se utiliza para el radio de los circulos internos

crcl\_rad\_units <- crcl\_rad/units # num radios

p<-ggplot(transposed\_data) +

# Make custom panel grid

geom\_hline(aes(yintercept = y),data.frame(y = c(0:crcl\_rad\_units) \* units),color = "lightgrey") +

# Add bars to represent the cumulative track lengths

# str\_wrap(region, 5) wraps the text so each line has at most 5 characters

# (but it doesn't break long words!)

#Barras

geom\_col(aes(x = reorder(str\_wrap(Attributes, 5), Value),y = Value,fill = Value),position = "dodge2",show.legend = TRUE,alpha = .75) +

#Puntos para los promedios por clase

# Add dots to represent the mean of class

geom\_point(aes(x = reorder(str\_wrap(Attributes,5),Avrgs),y = Avrgs),size = 3,color = "gray12") +

# Lineas punteeadas de cada categoría

# Lollipop shaft for mean gain per region

geom\_segment(aes(x = reorder(str\_wrap(Attributes, 5), Value),y = 0,xend = reorder(str\_wrap(Attributes, 5), Value),yend = crcl\_rad - 25),linetype = "dashed",color ="gray12")+

# Make it circular!

coord\_polar()+

# Valores de los radios internos

# Annotate custom scale inside plot

annotate(x = 0, y = 25, label = "25", geom = "text", color = "gray12", size=2.5) +

annotate(x = 0, y = 50, label = "50", geom = "text", color = "gray12", size=2.5) +

annotate(x = 0, y = 75, label = "75", geom = "text", color = "gray12", size=2.5) +

annotate(x = 0, y = 100, label = "100", geom = "text", color = "gray12", size=2.5) +

annotate(x = 0, y = 125, label = "125", geom = "text", color = "gray12", size=2.5) +

annotate(x = 0, y = 150, label = "150", geom = "text", color = "gray12", size=2.5) +

# Scale y axis so bars don't start in the center

scale\_y\_continuous(limits = c(-25, crcl\_rad),expand = c(0, 0),breaks = c(0:crcl\_rad\_units)\*units) +

# New fill and legend title for number of tracks per region

scale\_fill\_gradientn("Attribute value",colours = c( "#6C5B7B","#C06C84","#F67280","#F8B195")) +

# Make the guide for the fill discrete

guides(fill = guide\_colorsteps(barwidth = 15, barheight = .5, title.position = "top", title.hjust = .5))+

theme(

# Remove axis ticks and text

axis.title = element\_blank(),

axis.ticks = element\_blank(),

axis.text.y = element\_blank(),

# Use gray text for the region names

axis.text.x = element\_text(color = "gray12", size = 12),

# Move the legend to the bottom

legend.position = "bottom",

)+

theme(

# Set default color and font family for the text

text = element\_text(color = "gray12"),

# Make the background white and remove extra grid lines

panel.background = element\_rect(fill = "white", color = "white"),

panel.grid = element\_blank(),

panel.grid.major.x = element\_blank()

)

p

}

# Generar una lista de pókemons por cada typo 1

Bug <- pkms$Name[pkms['Type.1']=='Bug']

Dark <- pkms$Name[pkms['Type.1']=='Dark']

Dragon <- pkms$Name[pkms['Type.1']=='Dragon']

Electric <- pkms$Name[pkms['Type.1']=='Electric']

Fairy <- pkms$Name[pkms['Type.1']=='Fairy']

Fighting <- pkms$Name[pkms['Type.1']=='Fighting']

Fire <- pkms$Name[pkms['Type.1']=='Fire']

Flying <- pkms$Name[pkms['Type.1']=='Flying']

Ghost <- pkms$Name[pkms['Type.1']=='Ghost']

Grass <- pkms$Name[pkms['Type.1']=='Grass']

Ground <- pkms$Name[pkms['Type.1']=='Ground']

Ice <- pkms$Name[pkms['Type.1']=='Ice']

Normal <- pkms$Name[pkms['Type.1']=='Normal']

Poison <- pkms$Name[pkms['Type.1']=='Poison']

Psychic <- pkms$Name[pkms['Type.1']=='Psychic']

Rock <- pkms$Name[pkms['Type.1']=='Rock']

Steel <- pkms$Name[pkms['Type.1']=='Steel']

Water <- pkms$Name[pkms['Type.1']=='Water']

# Generar histogramas de probabilidad acumulada y Densidad de probabilidad

# Grafica la distribución por variable e indica la probabilidad segun los valores ingresados en los controles

# de tipo Slider. se actualiza el control slider para que tenga el min y max de la variable

# x:variable numerica

# stp: steps del control slider

# sldr\_ctrl\_name: nombre del control slider

# sldr\_ctrl: control slider, como objerto

get\_pop\_hist <- function(x,stp,sldr\_ctrl\_name,sldr\_ctrl){

mn <- min(x) # valor minimo de la variable x

mx <-max(x) # valor máximo de la variable x

mu <- median(x) # media de la variable x

sdv <- sd(x) # desv standard de la variable x

med <- median(x) # media de la variable x

skwnss <- skewness(x) # skewness (asimetría) de la variable x

updateSliderInput(session, sldr\_ctrl\_name,min=mn,max = mx, step = 1)

aval<-min(sldr\_ctrl) # valor minimo seleccionado

bval<-max(sldr\_ctrl) # valor maximo seleccionado

mthd<-"" # Valor defecto del metodo

prob<-1.00 # valor defecto de la probabilidad

# seleccion del metodo y calculo de la probabilidad

if(aval == mn & bval==mx){

prob<-1

mthd<-"test"

}else if (aval == mn){

prob<-pnorm(bval, mu, sdv)

mthd<-"DA.:"

}else if (bval == mx){

prob<-1-pnorm(aval, mu, sdv)

mthd<-"RC.:"

}else{

prob<-pnorm(bval, mu, sdv) - pnorm(aval, mu, sdv)

mthd<-"DP.:"

}

# Mensaje sobre el histograma

propmsg <- paste(mthd,'La probabilidad de tener Ataque de entre: ',aval,' y ',round(bval,2),' es de: ',round(prob\*100,2),'%')

# Graficar curva

p<-curve(dnorm(x, mu, sdv), mn, mx, xlab = "x", ylab = "f(x)")

# Graficar sombra

pshdw <- seq(aval, bval, 0.01)

lines (pshdw, dnorm(pshdw, mean = mu, sd = sdv), type = "h", col = "lightblue")

# Colocar el mensaje sobre el histograma

mtext(propmsg, side=3)

return(p)

}

```

Introducción

=======================================================================

Row

-----------------------------------------------------------------------

### Logo

```{r}

renderImage({

#Mostrar imagen en la portada del panel

filename <- normalizePath(file.path('images/',paste('Pokemon\_front.png', sep='')))

list(

src = filename,

contentType = "image/png",

width = 400,

alt = paste("Image ", input$cmbPokemon)

)

}, deleteFile = FALSE)

```

### Sobre el Tablero

```{r}

renderImage({

#Mostrar imagen en la portada del panel - contenido

filename <- normalizePath(file.path('images/',paste('Pokemon\_front\_Content.jpg', sep='')))

list(

src = filename,

contentType = "image/jpeg",

width = 400,

alt = paste("Image ", input$cmbPokemon)

)

}, deleteFile = FALSE)

```

Tipos de Pokémon

=======================================================================

Row

-----------------------------------------------------------------------

### Distribución del tipo 1 de Pokémon

```{r}

dat2=na.omit(pkms)

library(forcats)

# Gráfico de barras, cantidades por tipo 1

p <- ggplot(dat2, aes(x=fct\_infreq(dat2$Type.1),..count..,fill=dat2$Type.1)) +

geom\_bar()+

theme\_minimal()+

labs(x="Tipos de Pokémon",y="Frecuencia")+

theme(legend.position = "none")

ggplotly(p)

```

Row

-----------------------------------------------------------------------

### Distribución del tipo 2 de Pokémon

```{r}

dat2=na.omit(pkms)

library(forcats)

# Gráfico de barras, cantidades por tipo 2

p <- ggplot(dat2, aes(x=fct\_infreq(dat2$Type.2),..count..,fill=dat2$Type.2)) +

geom\_bar()+

theme\_minimal()+

labs(x="Tipos de Pokémon 2",y="Frecuencia")+

theme(legend.position = "none")

ggplotly(p)

```

Data por Clase

=======================================================================

Row

-----------------------------------------------------------------------

### Mapa de Calor

```{r}

heatmap(mtx, Colv = NA, Rowv = NA, scale="column")

```

### Valores

```{r}

renderTable(

btl\_avr

)

```

Análisis Descriptivo

=======================================================================

Row

-----------------------------------------------------------------------

### Atributos Básicos del Pokemon

```{r}

# Tabla con los atributos básicos del pokemon selecionado, se obtienen de la tabla transpuesta de atributos básicos

# se filtra la data por nombre

renderTable(

na.omit(pkms\_bsc[which(pkms\_bsc$Name == input$cmbPokemon\_sel),c('Basic Attributes','Value')])

)

```

### Atributos de Batalla del Pokemon

```{r}

# Tabla con los atributos de batalla del pokemon selecionado, se obtienen de la tabla transpuesta de atributos básicos

# se filtra la data por nombre

renderTable(

na.omit(pkms\_btl[which(pkms\_btl$Name == input$cmbPokemon\_sel),c('Battle Attributes','Value')])

)

```

Row

-----------------------------------------------------------------------

### Atributos de batalla de la clase

```{r}

renderPlot({

# Se obtiene el tipo 1 segun el pokemon seleccionado

sel\_type <- pkms$Type.1[pkms['Name']==input$cmbPokemon\_sel]

#Se obtiene solo los pokemon que pertenecen al mismo tipo 1, se reindexa con la función ridx()

selpkms<-ridx(pkms\_btl[which(pkms\_btl$Type.1 == sel\_type),])

if(input$chkbox\_Legendary==FALSE){

#Se filtran del arreglo los pokemon legendarios si se retira el check en el control para este fin

#se reindexa el arreglo con la funcion ridx

selpkms <-ridx(selpkms[which(selpkms$Legendary == 'False'),])

}

if(input$chkbox\_BiType==FALSE){

#Se filtran del arreglo los pokemon que tienen un tipo 2 si se retira el check en el control para este fin

#se reindexa el arreglo con la funcion ridx

selpkms<- ridx(selpkms[which(selpkms$has\_type2 == 'N'),c('Battle Attributes','Value')])

}

selpkms <- na.omit(selpkms)

# Grafico de boxplot con jitter, muestra disperción de cada uno de los atributos

# fuente: https://www.r-graph-gallery.com/index.html

ggplot(selpkms,aes(x=`Battle Attributes`,y=Value,fill=`Battle Attributes`)) +

geom\_boxplot() +

scale\_fill\_viridis(discrete = TRUE, alpha=0.6) +

geom\_jitter(color="black", size=0.4, alpha=0.9) +

theme\_ipsum() +

theme(

legend.position="none",

plot.title = element\_text(size=11)

) +

xlab("")

})

```

### Correlación entre attributos de batalla

```{r}

renderPlot({

# Nombre del pokemon selecionado

pkmName <- input$cmbPokemon\_sel

#Se obtiene solo los pokemon que pertenecen al mismo tipo 1, se reindexa con la función ridx()

sel\_type <- pkms$Type.1[pkms['Name']==pkmName]

#pkm<-pkms[which(pkms$Name==pkmName),c(input$var\_1,input$var\_2,pks\_bsc\_attr)]

# se extrae una parte del dataframe original, se filtra sólo pokemones del mismo tipo y

# sólo los campos:

# - seleccionados en el combo 1

# - seleccionados en el combo 2

# - Atritubos básicos (categírocs)

# - has\_type2

selpkms <- pkms[which(pkms$Type.1 == sel\_type),c(input$var\_1,input$var\_2,pks\_bsc\_attr,'has\_type2')]

#Se obtiene solo los pokemon que pertenecen al mismo tipo 1, se reindexa con la función ridx()

if(input$chkbox\_Legendary==FALSE){

#Se filtran del arreglo los pokemon legendarios si se retira el check en el control para este fin

#se reindexa el arreglo con la funcion ridx

selpkms <- ridx(selpkms[which(selpkms$Legendary != 'True'),])

}

if(input$chkbox\_BiType==FALSE){

#Se filtran del arreglo los pokemon que tienen un tipo 2 si se retira el check en el control para este fin

#se reindexa el arreglo con la funcion ridx

selpkms <- ridx(selpkms[which(selpkms$has\_type2 == 'N'),])

}

# Graficar gráfico de dispersión

ggplot(selpkms, aes(x=selpkms[,1], y=selpkms[,2], color=Legendary, pch=has\_type2, cex=1, alpha=.75)) +

geom\_point()+

labs(title = "Correlación ", x = input$var\_1, y = input$var\_2)

})

```

Input{.sidebar}

----------------------------------------------------------------

### Entrada del usuario

```{r}

# Se genera un combobox con agrupaciones por tipo 1

selectInput("cmbPokemon\_sel", "Selecciona un Pokemon",

choices =

list(`Bug` = Bug,

`Dark` = Dark,

`Dragon` = Dragon,

`Electric` = Electric,

`Fairy` = Fairy,

`Fighting` = Fighting,

`Fire` = Fire,

`Flying` = Flying,

`Ghost` = Ghost,

`Grass` = Grass,

`Ground` = Ground,

`Ice` = Ice,

`Normal` = Normal,

`Poison` = Poison,

`Psychic` = Psychic,

`Rock` = Rock,

`Steel` = Steel,

`Water` = Water))

```

```{r}

# Checkbox para incluir pokemones legendarios en el analisis

checkboxInput("chkbox\_Legendary", "Incluir Legendarios?", value = TRUE)

```

```{r}

# Checkbox para incluir pokemones legendarios en el analisis

checkboxInput("chkbox\_BiType", "Incluir Pokemones con 2 Tipos?", value = TRUE)

```

```{r}

# Combo para seleccionar la variable x para incluir en el analisis bivariado

# pks\_btl\_attr: se le pasa la lista de variables numéricas

selectInput("var\_1", "Seleccione la variable X", pks\_btl\_attr,selected = "Attack")

```

```{r}

# Combo para seleccionar la variable y para incluir en el analisis bivariado

# pks\_btl\_attr: se le pasa la lista de variables numéricas

selectInput("var\_2", "Seleccione la variable Y", pks\_btl\_attr,selected = "Defense")

```

Probabilidad

=======================================================================

Input{.sidebar}

----------------------------------------------------------------

### Atributos de Batalla

```{r}

# Combobox para seleccionar el tipo de pokemon

# Se le pasa la lista de Tipos unicos y ordenados

selectInput("cmd\_Type", "Seleccione un Tipo de Pokemon", unique(sort(pkms$Type.1)), selected = 1)

```

```{r}

# Slider para seleccionar el rango de puntos de ataque para los cuales se desea conocer la probabilidad

sliderInput("sldr\_atk", "Selecciona un Rango para Ataque", value=c(50, 100),min=0, max=250, step=5)

```

```{r}

# Slider para seleccionar el rango de puntos de defensa para los cuales se desea conocer la probabilidad

sliderInput("sldr\_def", "Selecciona un Rango para Defensa", value=c(50, 100),min=0, max=250, step=5)

```

```{r}

# Slider para seleccionar el rango de puntos de puntos de vida para los cuales se desea conocer la probabilidad

sliderInput("sldr\_hp", "Selecciona un Rango para los Puntos de Vida", value=c(50, 100),min=0, max=250, step=5)

```

```{r}

# Slider para seleccionar el rango de puntos de Velocidad para los cuales se desea conocer la probabilidad

sliderInput("sldr\_spd", "Selecciona un Rango para Velocidad", value=c(50, 100),min=0, max=250, step=5)

```

```{r}

# Slider para seleccionar el rango de puntos de Velocidad de Ataque para los cuales se desea conocer la probabilidad

sliderInput("sldr\_spd\_atk", "Selecciona un Rango para Velocidad de Ataque", value=c(50, 100),min=0, max=250, step=5)

```

```{r}

# Slider para seleccionar el rango de puntos de Velocidad de Defensa para los cuales se desea conocer la probabilidad

sliderInput("sldr\_spd\_def", "Selecciona un Rango para Velocidad de Defensa", value=c(50, 100),min=0, max=250, step=5)

```

row {data-width=650}

-----------------------------------------------------------------------

### Distribución Acumulada / Densidad de Probabilidad: Ataque

```{r}

renderPlot({

# Genera un histograma de probabilidad

# se le pasa un dataframe con pókemones de un tipo determinado

sel\_pkms <- filter(pkms, pkms$Type.1 == input$cmd\_Type)

x<-sel\_pkms$Attack

get\_pop\_hist(x,5,'sldr\_atk',input$sldr\_atk)

})

```

### Distribución Acumulada / Densidad de Probabilidad: Defensa

```{r}

renderPlot({

# Genera un histograma de probabilidad

# se le pasa un dataframe con pókemones de un tipo determinado

sel\_pkms <- filter(pkms, pkms$Type.1 == input$cmd\_Type)

x<-sel\_pkms$Defense

get\_pop\_hist(x,5,'sldr\_def',input$sldr\_def)

})

```

row {data-width=650}

-----------------------------------------------------------------------

### Distribución Acumulada / Densidad de Probabilidad: Salud

```{r}

renderPlot({

# Genera un histograma de probabilidad

# se le pasa un dataframe con pókemones de un tipo determinado

sel\_pkms <- filter(pkms, pkms$Type.1 == input$cmd\_Type)

x<-sel\_pkms$HP

get\_pop\_hist(x,5,'sldr\_hp',input$sldr\_hp)

})

```

### Distribución Acumulada / Densidad de Probabilidad: Velocidad

```{r}

renderPlot({

# Genera un histograma de probabilidad

# se le pasa un dataframe con pókemones de un tipo determinado

sel\_pkms <- filter(pkms, pkms$Type.1 == input$cmd\_Type)

x<-sel\_pkms$Speed

get\_pop\_hist(x,5,'sldr\_spd',input$sldr\_spd)

})

```

row {data-width=650}

-----------------------------------------------------------------------

### Distribución Acumulada / Densidad de Probabilidad: Velocidad de Ataque

```{r}

renderPlot({

# Genera un histograma de probabilidad

# se le pasa un dataframe con pókemones de un tipo determinado

sel\_pkms <- filter(pkms, pkms$Type.1 == input$cmd\_Type)

x<-sel\_pkms$Sp..Atk

get\_pop\_hist(x,5,'sldr\_spd\_atk',input$sldr\_spd\_atk)

})

```

### Distribución Acumulada / Densidad de Probabilidad: Velocidad de Defensa

```{r}

renderPlot({

# Genera un histograma de probabilidad

# se le pasa un dataframe con pókemones de un tipo determinado

sel\_pkms <- filter(pkms, pkms$Type.1 == input$cmd\_Type)

x<-sel\_pkms$Sp..Def

get\_pop\_hist(x,5,'sldr\_spd\_def',input$sldr\_spd\_def)

})

```

Duelo Pokemon

=======================================================================

row {data-width=650}

-----------------------------------------------------------------------

### Pokemon 1

```{r}

# Genera un combo box con los nombres de los pókemones, para seleccionar el pokemon 1

selectInput("cmbPokemon", "Seleccione un Pokemon", pkms$Name, selected = 'Pikachu')

# Genera un gráfico de barras circular que muestra los atributos de batalla del pokemon seleccionado

# Compara sus valores contra la media de la clase

renderPlot({

# Se obtienen los atributos de batalla desde la tabla transpuesta de variables numéricas

# Se filtra sólo los registros que corresponden al pokemon seleccionado

pkm <- pkms\_btl[which(pkms\_btl$Name ==input$cmbPokemon),c('Name','Battle Attributes','Value')]

# Se renombran las columnas del dataframe generado

names(pkm)<-c('Name','Attributes','Value')

# Se obtiene el tipo 1 del pokemon seleccionado

pkm\_type <- pkms$Type.1[pkms$Name==input$cmbPokemon]

# Se obtiene una lista de los pokemones de la misma clase

pkms\_bytype <- pkms[which(pkms$Type.1==pkm\_type),c(pks\_btl\_attr)]

# Se obtiene los promedios de los atributos de batalla (variables numericas) de la clase

pkms\_bytype\_avrgs<-get\_fields\_avgs(pkms\_bytype,pks\_btl\_attr)

# Se transponen los promedios

pkms\_bytype\_avrgs\_t <- melt(

pkms\_bytype\_avrgs,

measure.vars=pks\_btl\_attr,

variable.name='Attributes',

value.name='Avrgs',

na.rm=TRUE)

# Se combinan los dataframes de atributos del pokemon seleccionado con el dataframe de promedios.

# la llave es el nombre del atributo

pkm<-merge(pkm, pkms\_bytype\_avrgs\_t, by = "Attributes")

#Se establece el radio del grafico circular

crcl\_rad <- ceiling(max(pkm[,c('Avrgs','Value')]\*1.1/25))\*25+25

#Se muestra el gráfico

circular\_bar\_plot(pkm,crcl\_rad)

})

```

### Pokemon 2

```{r}

# Genera un combo box con los nombres de los pókemones, para seleccionar el pokemon 2

selectInput("cmbPokemon\_2", "Seleccione un Pokemon retador", pkms$Name, selected = 'Charmander')

renderPlot({

# Genera un gráfico de barras circular que muestra los atributos de batalla del pokemon seleccionado

# Compara sus valores contra la media de la clase

pkm <- pkms\_btl[which(pkms\_btl$Name == input$cmbPokemon\_2),c('Name','Battle Attributes','Value')]

# Se renombran las columnas del dataframe generado

names(pkm)<-c('Name','Attributes','Value')

# Se obtiene el tipo 1 del pokemon seleccionado

pkm\_type <- pkms$Type.1[pkms$Name==input$cmbPokemon\_2]

# Se obtiene una lista de los pokemones de la misma clase

pkms\_bytype <- pkms[which(pkms$Type.1==pkm\_type),c(pks\_btl\_attr)]

# Se obtiene los promedios de los atributos de batalla (variables numericas) de la clase

pkms\_bytype\_avrgs<-get\_fields\_avgs(pkms\_bytype,pks\_btl\_attr)

# Se transponen los promedios

pkms\_bytype\_avrgs\_t <- melt(

pkms\_bytype\_avrgs,

measure.vars=pks\_btl\_attr,

variable.name='Attributes',

value.name='Avrgs',

na.rm=TRUE)

# Se combinan los dataframes de atributos del pokemon seleccionado con el dataframe de promedios.

# la llave es el nombre del atributo

pkm<-merge(pkm, pkms\_bytype\_avrgs\_t, by = "Attributes")

#Se establece el radio del grafico circular

crcl\_rad <- ceiling(max(pkm[,c('Avrgs','Value')]\*1.1/25))\*25+25

#Se muestra el gráfico

circular\_bar\_plot(pkm,crcl\_rad)

})

```

row {data-width=650}

-----------------------------------------------------------------------

### Foto Pokemon 1

```{r}

renderImage({

# Muestra la imagen que corresponde al nombre del pokemon 1 seleccionado.

# Se tiene una carpeta con los archivos de las imagenes, los nombres deben coincidir con los del combobox

imgsz <- 120# Se establece el tamaño de la imagen

filename <- normalizePath(file.path('images/',paste(input$cmbPokemon,'.png', sep='')))

list(

src = filename,

contentType = "image/png",

width = imgsz,

height = imgsz,

alt = paste("Image ", input$cmbPokemon)

)

}, deleteFile = FALSE)

```

### Foto Pokemon 2

```{r}

renderImage({

# Muestra la imagen que corresponde al nombre del pokemon 1 seleccionado.

# Se tiene una carpeta con los archivos de las imagenes, los nombres deben coincidir con los del combobox

imgsz <- 120 # Se establece el tamaño de la imagen

filename <- normalizePath(file.path('images/',paste(input$cmbPokemon\_2,'.png', sep='')))

list(

src = filename,

contentType = "image/png",

width = imgsz,

height = imgsz,

alt = paste("Image ", input$cmbPokemon)

)

}, deleteFile = FALSE)

``````