Universidad Nacional de Rosario

Facultad de Ciencias Exactas, Ingeniería y Agrimensura

Tecnicatura Universitaria en Inteligencia Artificial

Árboles de decisión y Clasificación supervisada

TUIA - IA 4.3 Minería de Datos

Trabajo Práctico Nro. 2

***Fecha*:** 17/10/2023

Docentes:

- *Flavio Spetale*
- *Dolores Biondo*
- *Facundo Vasquez*
- *Sofia Errecarte*

Grupo:

- *Revello Simon*
- *Giampaoli Fabio*

Objetivo:

El objetivo de este trabajo práctico es integrar los conocimientos adquiridos en las unidades 4 (Árboles de decisión) y 5 (Clasificación con Aprendizaje Automático) en dos problemas reales asociados uno al comportamiento financiero de 1000 empresas y otro a un juego de cartas.

Actividades:

1. Descargar el conjunto de datos, 1000_Companies.csv1, para realizar el trabajo práctico. Analizar los atributos del conjunto de datos (distribuciones, valores, outliers, tipos de datos, etc.) y elegir un método de estandarización.

2. Realizar la estimación del atributo Profit utilizando árboles de decisión (Regresión) analizando los parámetros máximo profundidad, número mínimo de observaciones, número mínimo de observaciones por separación y criterio de separación. Graficar el árbol obtenido en el proceso de entrenamiento y mostrar los resultados sobre dos conjuntos de test (Error Absoluto Medio, Error Cuadrático Medio y Raíz del Error Cuadrático Medio).

3. Descargar el conjunto de datos, PokemonDB.csv2, para realizar el trabajo práctico. Analizar los atributos del conjunto de datos (distribuciones, valores, outliers, tipos de datos, etc.)

4. Realizar la estimación del atributo Type utilizando árboles de decisión (Clasificación) analizando los parámetros máximo profundidad, número mínimo de observaciones, número mínimo de observaciones por separación y criterio de separación. Graficar el árbol obtenido en el proceso de entrenamiento y mostrar los resultados sobre dos conjuntos de test (Precisión, Exhaustividad y Exactitud).

5. Realizar la estimación del atributo Type utilizando Bayes Ingenuo. Aquí deberá considerar un criterio de división de los atributos para discretizar los. Mostrar los resultados sobre dos conjuntos de test (Precisión, Exhaustividad y Exactitud).

6. Realizar la estimación del atributo Type utilizando k-NN analizando los parámetros cantidad de vecinos, métrica y valor de p. Mostrar los resultados sobre un conjunto de test (Precisión, Exhaustividad y Exactitud).

Resolución - Regresión:

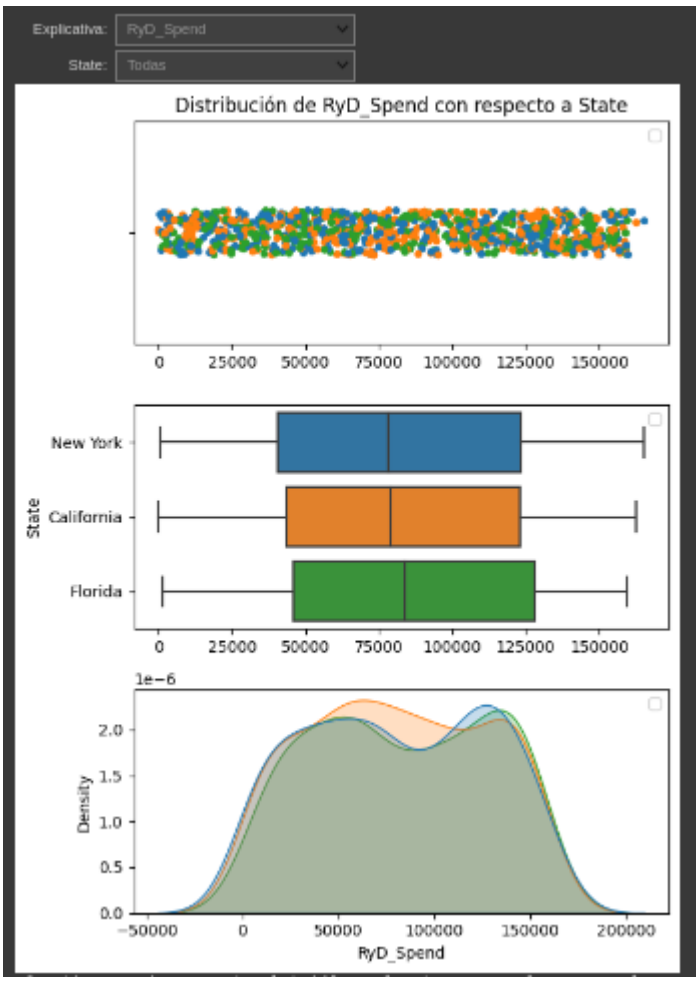
El conjunto de datos contiene las siguientes características:

- **R&D Spend (float64)**: Esta característica representa la cantidad de dinero que una startup gasta en actividades de investigación y desarrollo. Indica la inversión realizada en crear nuevos productos, mejorar los existentes o mejorar las capacidades tecnológicas generales.
- **Administration (float64)**: Esta característica representa la cantidad de dinero que una startup gasta en tareas administrativas y gastos generales. Incluye los costos asociados con el espacio de oficina, los servicios públicos, los salarios del personal no productivo y otros gastos operativos generales.
- **Marketing Spend (float64)**: Esta característica representa la cantidad de dinero que una startup invierte en actividades de marketing y promoción. Incluye los gastos relacionados con las campañas publicitarias, los esfuerzos de marketing en línea, las relaciones públicas y otras iniciativas destinadas a aumentar la visibilidad de la marca y atraer clientes.
- **State (object)**: Esta característica captura el estado en el que opera cada startup. Se representa como un tipo de objeto, lo que sugiere que contiene información categórica o textual sobre la ubicación geográfica de la empresa. La información del estado puede ser útil para analizar tendencias regionales o identificar posibles variaciones en los costos operativos y las ganancias en diferentes ubicaciones.
- **Profit (float64)**: Esta característica representa la rentabilidad de cada startup. Indica la ganancia o pérdida financiera generada por la empresa dentro de un período específico. El beneficio se mide típicamente como los ingresos obtenidos menos los costos operativos totales incurridos, incluyendo el gasto en I+D, los gastos de administración, el gasto en marketing y otros factores relevantes.

Podemos ver una muestra del conjunto:

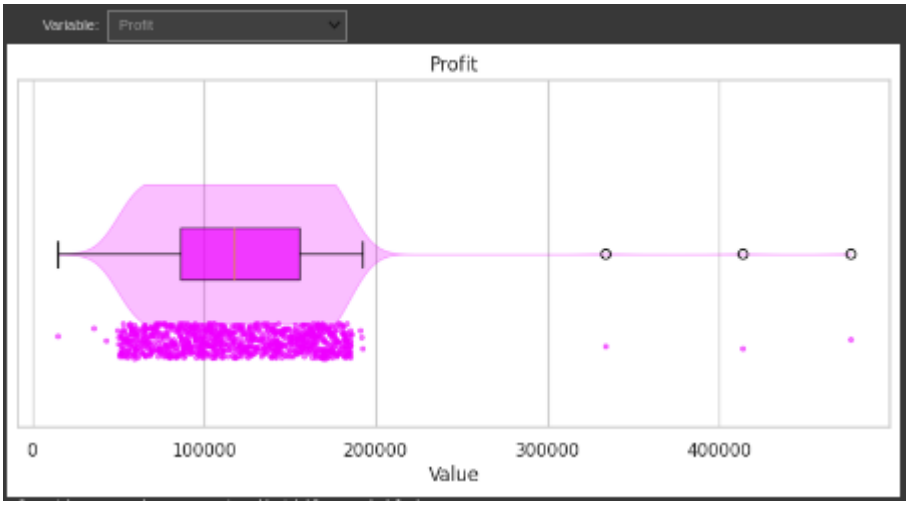
RyD_Spend	Administration	Marketing Spend	State	Profit
165349.20	136897.80	471784.10	New York	192261.83
162597.70	151377.59	443898.53	California	191792.06
153441.51	101145.55	407934.54	Florida	191050.39

Se inicia un análisis exploratorio de las variables. Probamos identificar si las variables varían dependiendo del estado de la compañía. Es decir, si hay estados o ciudades más propensos a ciertos gastos o ganancias que otros. Para ello se genera un panel interactivo de selección de la variable explicativa y el estado o estados.



Con esto notamos que no hay diferencias significativas entre los números de cada estado.

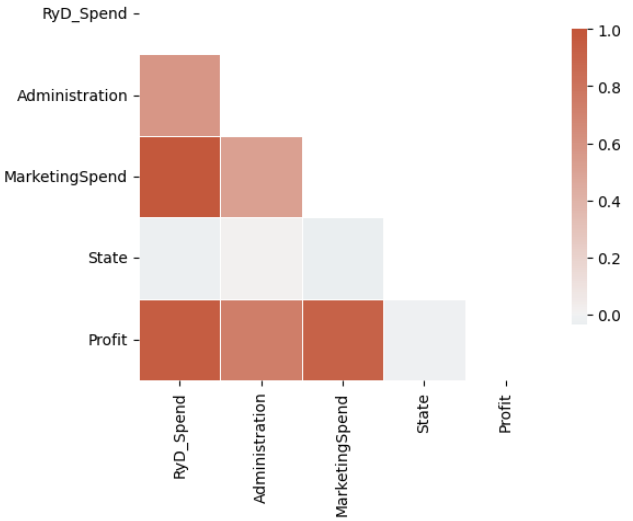
También se estudian las distribución de cada variable a nivel global para comprender los valores normales para cada tipo de gastos o ganancia nuevamente con un panel interactivo que genera gráficos de RainCloud:



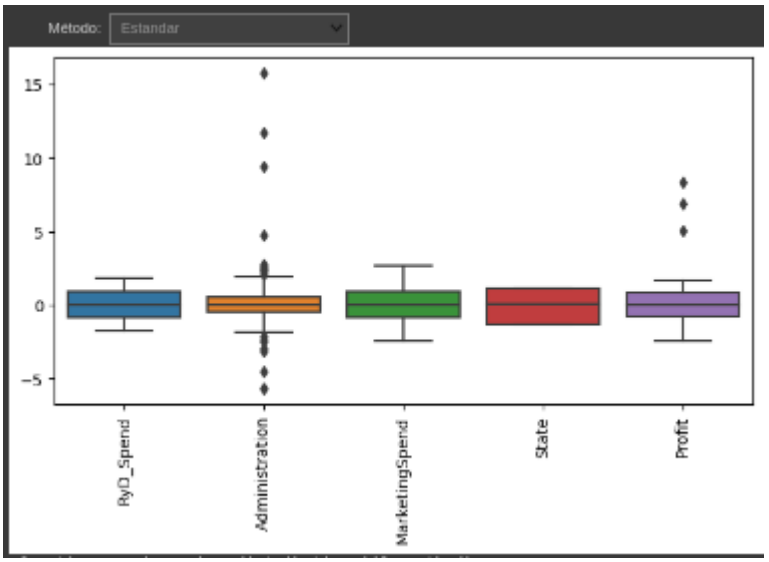
En general notamos que hay presencia de valores atípicos en algunas de las variables. Pero en general las distribuciones tienden a ser normales con mucha concentración en ciertos rangos, y colas más o menos densas según la variable.

Como el modelo de interés está basado en árboles de decisión, los valores atípicos no tienen influencia en la forma de tomar decisiones. Por ello, no se realiza un tratamiento exhaustivo de los mismos.

Notamos en una matriz de correlación que en general hay buenas correlaciones lineales entre la variables numéricas:

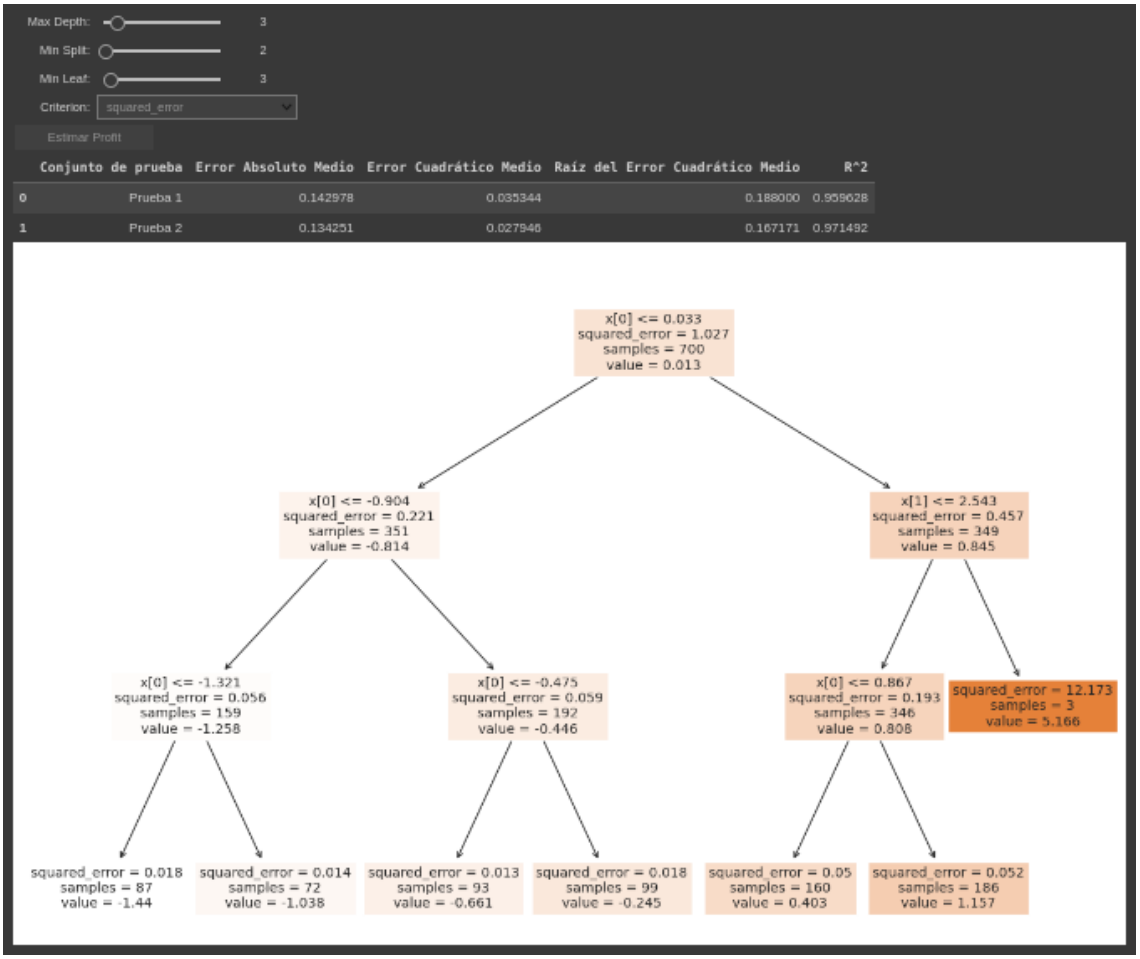


Elegimos un método de escalado estándar para llevar todas la variables un mismo rango de valores:



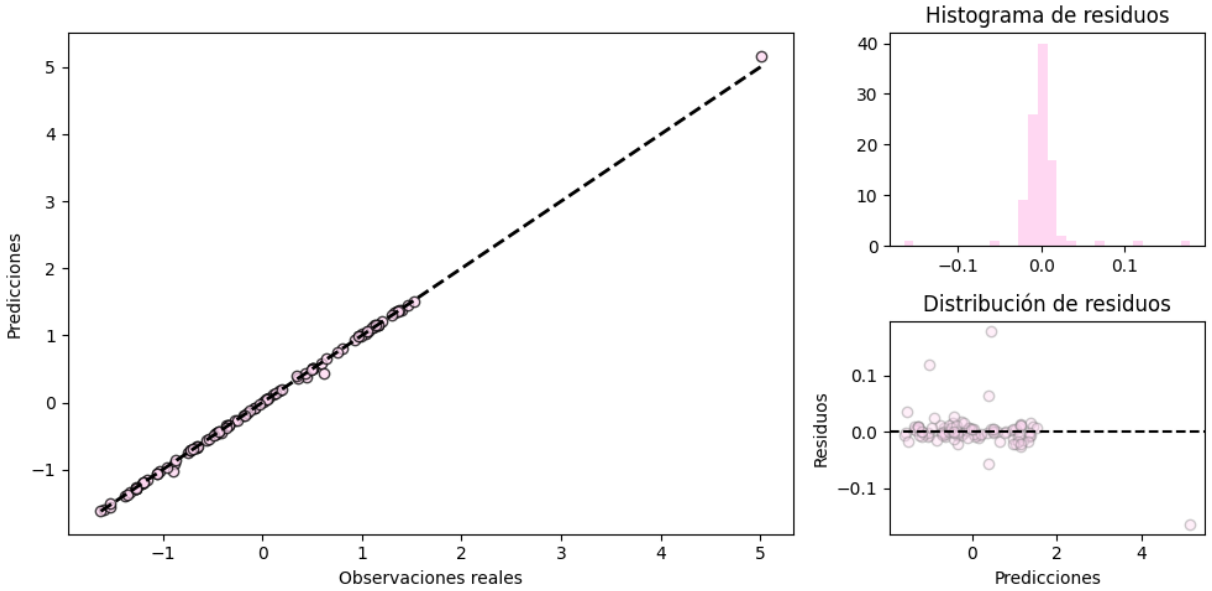
Como el objetivo es estimar la variable continua Profit usando un modelo de Decision Tree Regressor. Para ello se ha creado un panel interactivo de ajuste de hiper parámetros del modelo que se ajustan manualmente para encontrar las mejores métricas de validación sobre las predicciones, y generar un gráfico del árbol de decisión que se ha conseguido en cada entrenamiento.

Aquí, se deben ajustar los parámetros y presionar el botón de estimar Profit para generar un dataframe con las métricas de validación de cada conjunto de test y mostrar el árbol de decisión utilizado para el entrenamiento del modelo con dichos parámetros:



Como se nota un gran rendimiento de este modelo para estimar la variable Profit, se crea un modelo definitivo con estos hiperparametros encontrados manualmente y se entrena con un conjunto de entrenamiento más grande.

Se obtiene un modelo como el siguiente:



Notamos un ajuste casi perfecto de los datos. La línea diagonal representa el ideal entre valores reales y predichos, y vemos que en su mayoría los puntos están sobre dicha línea, inclusive los valores que podrían resultar atípicos.

Los gráficos de residuos indican que el modelo no tiende a subestimar ni sobreestimar de manera desproporcionada respecto una de la otra.

Resolución - Clasificación

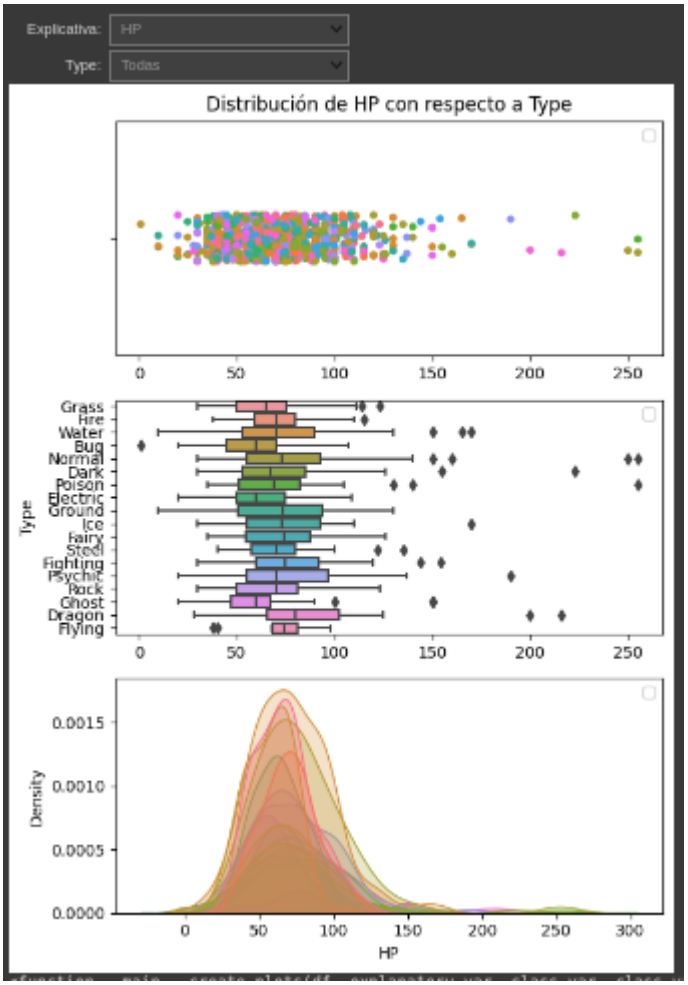
El segundo conjunto de datos es sobre un juego de cartas que presenta propiedades de distintos personajes del juego. Las variables presentes en el conjunto son:

- **Name**: Nombre de cada pokemon
- **Type**: Cada pokémon tiene un tipo, esto determina la debilidad/resistencia a los ataques
- **HP**: puntos de golpe, o salud, define cuánto daño puede soportar un pokémon antes de desmayarse
- **Attack**: el modificador base para ataques normales
- **Defense**: la resistencia al daño base contra ataques normales
- **SP Atk**: ataque especial, el modificador base para ataques especiales (por ejemplo, fire blast, bubble beam)
- **SP Def**: la resistencia al daño base contra ataques especiales
- **Speed**: determina qué pokémon ataca primero en cada ronda

Podemos notar una muestra del conjunto aquí:

Name	Type	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed
Bulbasaur	Grass	45	49	49	65	65	45
Ivysaur	Grass	60	62	63	80	80	60
Venusaur	Grass	80	82	83	100	100	80

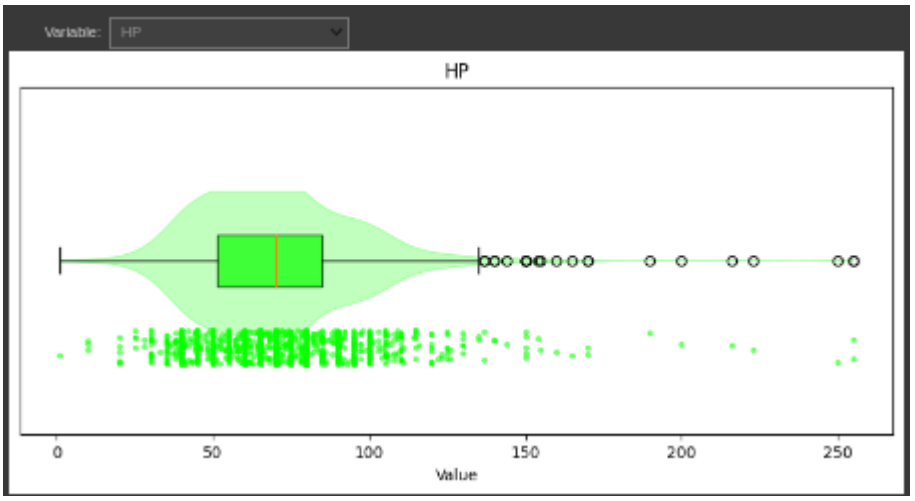
Iniciamos estudiando las distribuciones de cada variable acorde al tipo de pokémon en un gráfico interactivo que permite elegir un tipo específico o ver todos a la vez enfrentados a una variable numérica explicativa del conjunto:



Notamos en general para todas la variables que las distribuciones de cada clase de la variable categórica son muy similares entre sí. Es decir, sería difícil identificar un tipo de pokémon basándose solamente en las distribuciones de las variables presentadas debido a su similaridad con las demás.

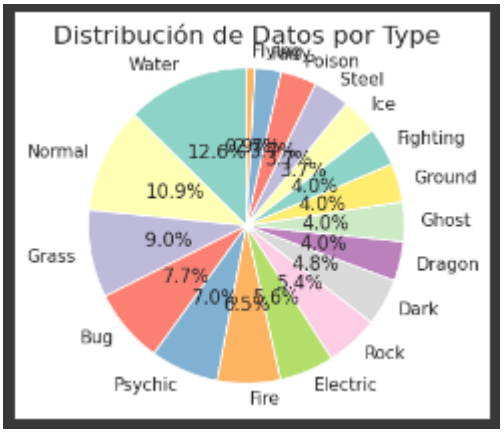
También notamos presencia de valores atípicos en muchas de las clases, pero en general se concentran valores normales. Es decir, las colas de las distribuciones normales presentan baja curtosis.

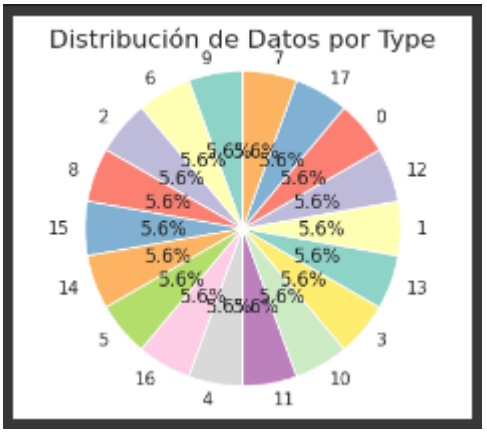
Se genera un gráfico para notar las distribuciones de las variables a nivel global:



Notamos principalmente que las distribuciones presentan valores atípicos que solamente resultan altos. Es decir, por ejemplo, se encuentran valores atípicos altos de HP con respecto a los demás pokémons, pero ninguno tiene valores mucho más bajos que los demás. Este comportamiento ocurre igual en todas las variables.

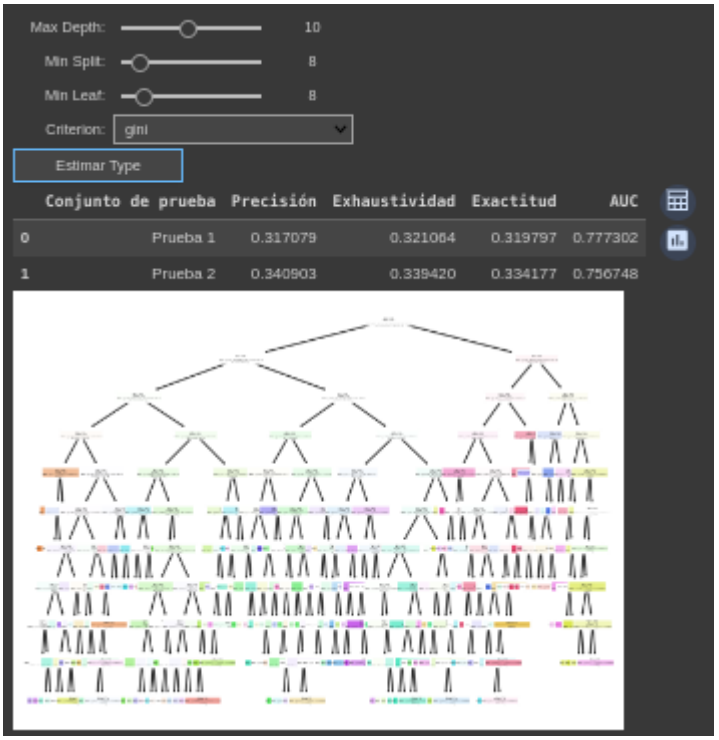
Para problemas de clasificación es importante contar con datasets balanceados en cuanto a las clases a predecir debido a que los modelos pueden generar sesgo hacia las clases mayoritarias en cuanto a cantidad de datos. Debajo se muestra el dataset desbalanceado en su forma inicial, y a la derecha el dataset con las variables codificadas y balanceadas de forma proporcional.





Para balancear el dataset se ha utilizado un enfoque de oversampling SMOTE. Este método implica la creación de muestras sintéticas de la clase minoritaria en lugar de simplemente duplicar ejemplos existentes. Esto se hace seleccionando muestras que están cerca en el espacio de características, trazando una línea entre los ejemplos en el espacio de características y dibujando una muestra a lo largo de esa línea. Por lo que el dataset balanceado presenta mayor cantidad de registros donde muchos son datos artificiales pero creados de forma estratégica.

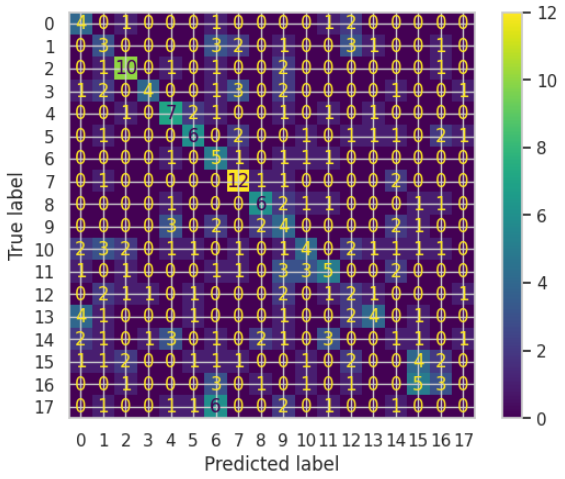
Para estimar la clase Tipo de un conjunto de validación se crea un menú interactivo de selección de hiperparametros que retorna un modelo entrenado con su gráfico y las métricas de validación en dos conjuntos de test sobre las predicciones de las clases:



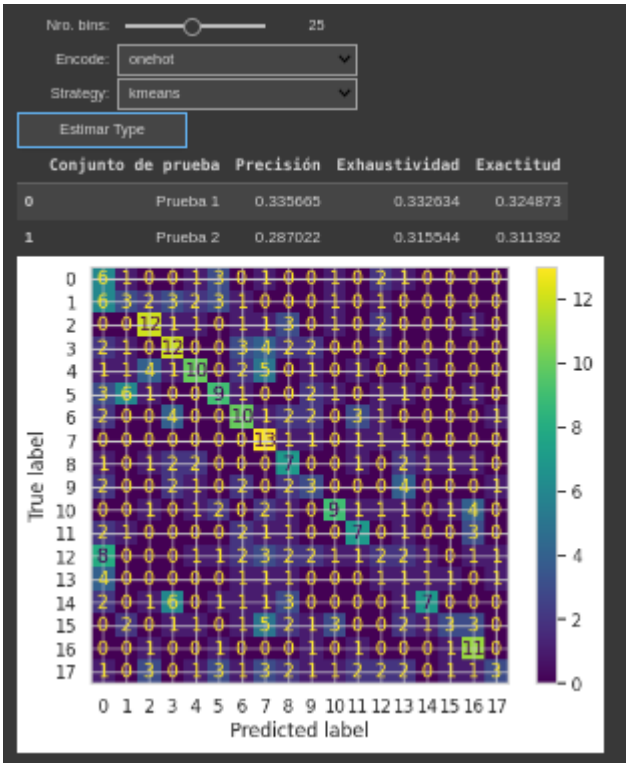
	Conjunto de prueba	Precisión	Exhaustividad	Exactitud	AUC
0	Prueba 1	0.317079	0.321084	0.319797	0.777302
1	Prueba 2	0.340903	0.339420	0.334177	0.756748

Con los parámetros encontrados manualmente no se han conseguido gran rendimiento en las estimaciones. En general para ambos conjuntos de test las métricas de clasificación son bajas, por lo que el modelo clasifica erróneamente los tipos en gran parte de los casos.

Notamos en una matriz de confusión que en general las aserciones son erroneas a excepcion de unas pocas clases, como los tipos 7 y 2 (es un tipo de pokemon codificado con LabelEncoder), que para clases 7 y 2 han acertado 12 y 10 veces con las predicciones de dichas clases.



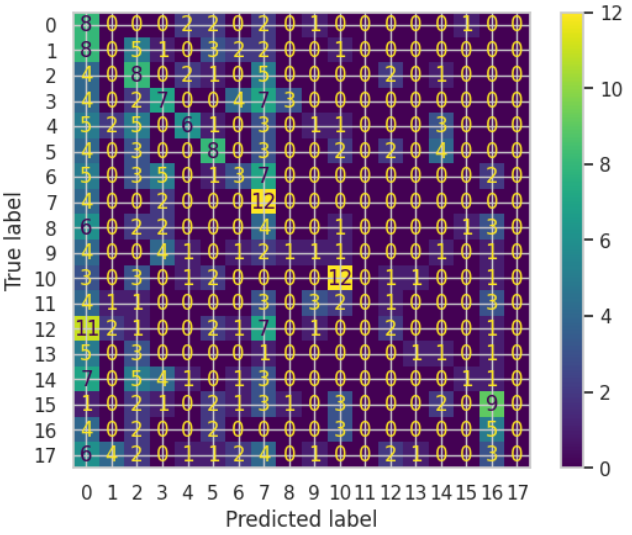
También probamos las estimación de esta variable categórica usando algoritmos de Bayes Ingenuos con dos enfoques diferentes. El primero utilizando un modelo multinomial. Es decir que trabaja obteniendo las probabilidades condicionales de cada clase. Para ello es necesario discretizar en clases las variables numéricas del dataset. Por lo que se crea un menú de selección para el discretizador y mostrar las métricas del modelo multinomial acorde al dataset transformado. Notar que Bayes Ingenuo no tiene hiperparametros para elegir en este caso.



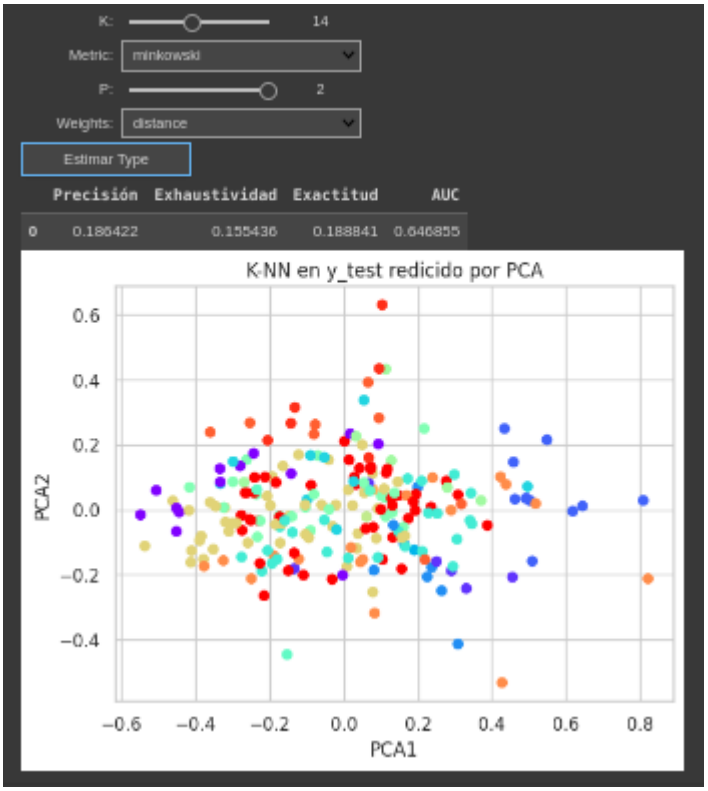
En este caso, discretizar las variables en 25 clases utilizando k means ha demostrado un mejor rendimiento que otros parámetros. Pero aun así las métricas de validación de aserción de las predicciones son bajas para un modelo que debe realizar esta tarea.

En el otro enfoque, se utiliza un modelo gaussiano que interpreta las variables numéricas como tal, tomando decisiones probabilísticas basadas en las distribuciones de las variables.

Este modelo tiende más a equivocarse que el enfoque multinomial, por lo que el anterior es preferido para hacer esta tarea de clasificación.



Finalmente se prueba esta misma tarea de estimación del tipo de pokémon utilizando un modelo basado en cálculo de distancias, K Nearest Neighbours. Se genera un bloque interactivo de selección manual de hiperparametros para el mismo. Al presionar el botón se entrena el modelo con los parámetros seleccionados, realiza predicciones y métricas de validación, y agrega un gráfico que son los datos del conjunto de validación (x_test) reducidos a dos dimensiones con PCA para facilitar la visualización. Cada color de los puntos representa una clase de pokemon:

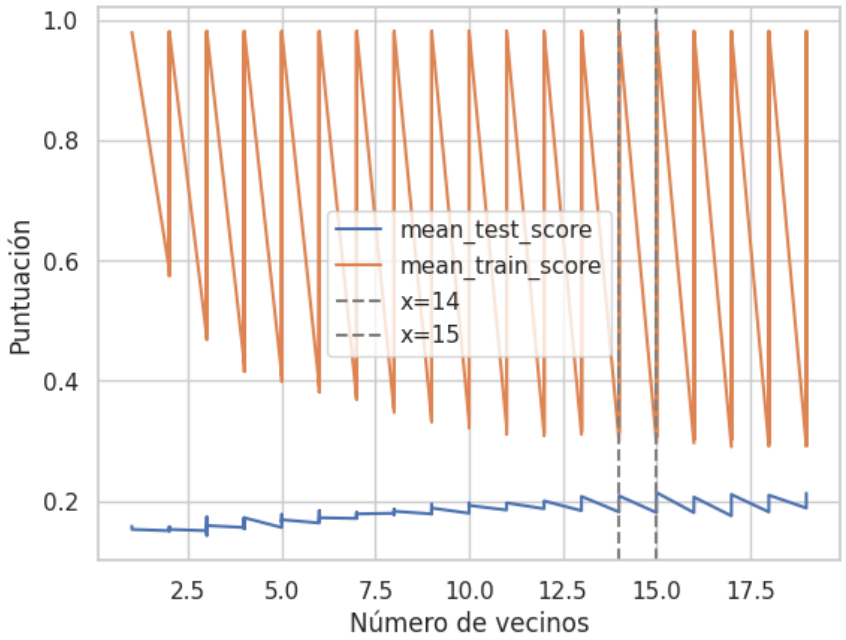


Vemos que en general las métricas son muy bajas para la tarea de clasificación. Y el gráfico nos permite ver rápidamente que el modelo no está encontrado relaciones directas entre las clases debido a que los colores en los puntos están muy dispersos.

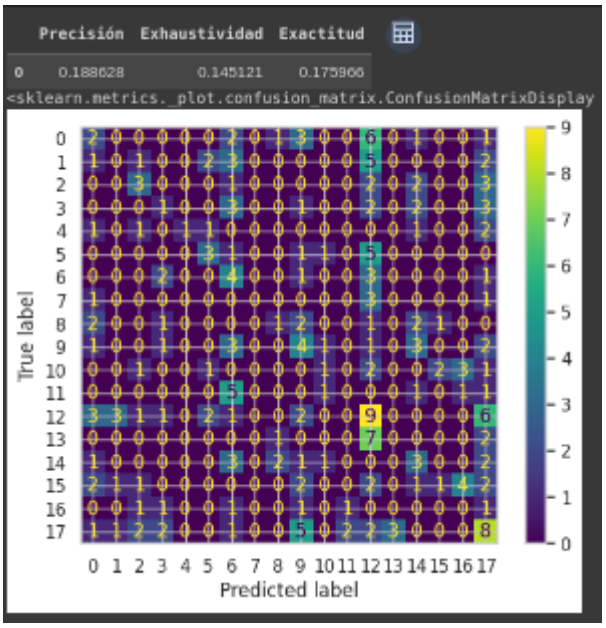
Probamos hacer la selección de los hiperparamétricos de este último modelo de forma automática usando un método de grilla. El mismo a través de 380 entrenamientos diferentes con rangos predefinidos de hiperparametros por lo que optar, selecciona esta combinación:

```
{'n_neighbors': 15, 'p': 2, 'weights': 'distance'}
```

Como vemos, es casi la misma selección que hemos encontrado manualmente. Podemos notar el proceso de optimización de hiperparametros en el siguiente gráfico y comparar la selección de 14 y 15 vecinos cercanos como parámetro para el modelo:



Tanto las métricas de validación como de test son muy similares para ambos valores. Por lo que finalmente se crea un modelo con 14 vecinos cercanos como parámetro, y se genera su matriz de confusión:



Nuevamente notamos métricas muy bajas a excepción de algunas pocas clases. Por lo que este modelo no rinde lo suficientemente bien para realizar estimaciones sobre la categoría tipo de pokémon en este dataset.

Conclusiones

Se han trabajado dos conjuntos de datos diferentes, con objetivos diferentes.

Para el primero el objetivo fue estimar una variable continua usando una regresión basada en árboles. Los resultados fueron muy positivos y se han conseguido métricas de validación de predicciones muy buenas con un modelo relativamente sencillo.

Para el segundo conjunto de datos, el objetivo fue de estimar clases, es decir, realizar clasificaciones de tipos de personajes. Se han probado 3 enfoques de modelos diferentes y cambiado los hiperparametros de cada uno. En los tres modelos de clasificación las métricas de validación sobre las predicciones resultaron insatisfactorias debido a las altas probabilidades de que se equivoquen al estimar una clase en base a otros atributos.

Aun así se han probado tres enfoques de clasificación diferentes donde algunos demostraron un mejor desempeño, aunque no significativo para ser elegido como modelo preferido para esta tarea.

Para lograr estimaciones de las clases con mayor precisión es posible que se requieran modelos más sofisticados o un dataset que aporte más información, ya sea con más variables explicativas o más cantidad de registros para 'enriquecer' el entrenamiento de los modelos de clasificación.