INFORME SOBRE EL PROYECTO FINAL: PREDICCIÓN DE APARICIONES DE POKÉMON EN POKÉMONGO

UN PROBLEMA DE CLASIFICACIÓN

MARCO TULIO CANCELAS MENDICUTI
CURSO CIENCIA DE DATOS IT ACADEMY (BARCELONA ACTIVA)
2023

Introducción:

El proyecto final se centra en la predicción de las apariciones de Pokémon en el juego móvil de realidad aumentada PokémonGo. Desarrollado por Niantic Inc., PokémonGo permite a los jugadores usar la capacidad de GPS de sus dispositivos móviles para localizar, capturar, combatir y entrenar criaturas virtuales llamadas Pokémon en el mundo real. El objetivo del proyecto es entrenar un algoritmo de aprendizaje automático para predecir dónde aparecerán los Pokémon en el futuro utilizando un conjunto de datos que consta de aproximadamente 293,000 avistamientos históricos de Pokémon, con diversas características que describen cada aparición.

Descripción del Conjunto de Datos:

El conjunto de datos se compone de los siguientes atributos que se utilizarán para entrenar el algoritmo de aprendizaje automático:

1. `pokemonId`: El identificador numérico del Pokémon. Este atributo debe eliminarse para evitar influir en las predicciones, ya que es el objetivo (rango entre 1 y 151).

2. `latitude` y `longitude`: Las coordenadas geográficas del avistamiento de Pokémon (atributos numéricos).

3. `appearedLocalTime`: La hora exacta del avistamiento en formato "yyyy-mm-dd'T'hh-mm-ss.ms'Z'" (nominal).

4. `cellId 90-5850m`: La posición geográfica proyectada en una celda S2, con tamaños de celda que van desde 90 a 5850 metros (atributo numérico).

5. `appearedTimeOfDay`: La hora del día del avistamiento (noche, tarde, tarde, mañana).

6. `appearedHour` y `appearedMinute`: La hora y el minuto locales del avistamiento (atributos numéricos).

7. `appearedDayOfWeek`: El día de la semana del avistamiento (lunes, martes, miércoles, jueves, viernes, sábado, domingo).

8. `appearedDay`, `appearedMonth` y `appearedYear`: El día, mes y año del avistamiento (atributos numéricos).

9. `terrainType`: El tipo de terreno donde apareció el Pokémon, descrito con la ayuda de GLCF Modis Land Cover (atributo numérico).

10. `closeToWater`: Indica si el Pokémon apareció cerca del agua (a 100 metros o menos) (booleano).

11. `city`: La ciudad donde ocurrió el avistamiento (nominal).

12. `continent`: El continente del avistamiento, aunque no siempre se analiza correctamente (nominal).

13. `weather`: El tipo de clima durante el avistamiento, que incluye diversas categorías climáticas (Foggy Clear, PartlyCloudy, MostlyCloudy, Overcast, Rain, BreezyandOvercast, LightRain, Drizzle, BreezyandPartlyCloudy, HeavyRain, BreezyandMostlyCloudy, Breezy, Windy, WindyandFoggy, Humid, Dry, WindyandPartlyCloudy, DryandMostlyCloudy, DryandPartlyCloudy, DrizzleandBreezy, LightRainandBreezy, HumidandPartlyCloudy, HumidandOvercast, RainandWindy) (fuente de todos los atributos climáticos).

14. `temperature`: La temperatura en grados Celsius en la ubicación del avistamiento (atributo numérico).

15. `windSpeed`: La velocidad del viento en kilómetros por hora en la ubicación del avistamiento (atributo numérico).

16. `windBearing`: La dirección del viento (atributo numérico).

17. `pressure`: La presión atmosférica en bares en la ubicación del avistamiento (atributo numérico).

18. `weatherIcon`: Una representación compacta del clima en la ubicación del avistamiento (fog, clear-night, partly-cloudy-night, partly-cloudy-day, cloudy, clear-day, rain, wind).

19. `sunriseMinutesMidnight` a `sunsetMinutesBefore`: La hora de aparición en relación con el amanecer y el atardecer.

20. `population density`: La densidad de población por kilómetro cuadrado en la ubicación del avistamiento (atributo numérico).

21. `urban-rural`: Indica cuán urbana es la ubicación donde apareció el Pokémon (booleano, basado en la densidad de población).

22. `gymDistanceKm` y `pokestopDistanceKm`: La distancia al gimnasio o la parada de Pokémon más cercanos en kilómetros desde el avistamiento (atributos numéricos).

23. `gymIn100m` y `pokestopIn5000m`: Indica si hay un gimnasio o parada de Pokémon en 100 metros o 5000 metros, respectivamente (booleano).

24. `cooc 1-cooc 151`: La co-ocurrencia con cualquier otro Pokémon (ID de Pokémon que van del 1 al 151) dentro de una distancia de 100 metros y en las últimas 24 horas (booleano).

25. `class`: Indica a qué Pokémon corresponde el avistamiento, y es la variable objetivo que se desea predecir.

Metodología:

Para abordar este proyecto, se seguirán los siguientes pasos:

1. Preprocesamiento de Datos: Se realizará una limpieza de datos para manejar valores nulos, codificar variables categóricas y normalizar datos numéricos si es necesario.

2. Exploración de Datos: Se llevará a cabo un análisis exploratorio de datos para comprender las distribuciones y relaciones entre las variables.

3. División de Datos: El conjunto de datos se dividirá en conjuntos de entrenamiento y prueba para la validación del modelo.

4. Modelado: Se entrenarán varios modelos de aprendizaje automático, como regresión, clasificación y modelos de ensamble, para predecir el Pokémon en función de las características proporcionadas.

6. Ajuste de Hiperparámetros: Se realizará la optimización de hiperparámetros para mejorar el rendimiento del modelo.

7. Evaluación del Modelo: Se evaluará el rendimiento del modelo utilizando métricas adecuadas, como precisión, recall, F1-score y según el tipo de modelo.

Preprocesamiento de datos:

- pokemonId: Esta columna contiene el identificador numérico del Pokémon, y se elimina porque es la variable objetivo que se desea predecir. Por lo tanto, no se necesita como característica.
- city y continent: Estas columnas contienen información sobre la ciudad y el continente de la ubicación del avistamiento de Pokémon. Dado que ya se disponen de las coordenadas geográficas (latitud y longitud) para representar la ubicación, estas columnas se consideran redundantes y se eliminan.
- weatherIcon: Esta columna también se elimina porque se considera redundante con la columna weather, que ya describe el tipo de clima durante un avistamiento. Por lo tanto, no es necesario mantener ambos atributos.
- appearedYear: Esta columna contiene el año del avistamiento de Pokémon. Dado que solo se dispone de datos de un año (2016), esta columna no agrega información relevante y se elimina.
- appearedMonth: Similar al caso anterior, esta columna contiene el mes del avistamiento y también es redundante, ya que solo hay datos de un mes.
- appearedDayOfWeek: La columna que indica el día de la semana del avistamiento se elimina porque esta información ya está contenida en la columna "appearedLocalTime".
- appearedTimeOfDay: Al igual que las columnas anteriores, esta columna se elimina porque la información de la hora del día ya se encuentra en "appearedLocalTime".
- appearedDay, appearedHour, y appearedMinute: Estas columnas se eliminan debido a que la información relacionada con la fecha y la hora del avistamiento se maneja de manera más completa en la columna "appearedLocalTime" .
- Se calcula la frecuencia de las categorías en las columnas "weather" y "terrainType" para comprender cuántas veces aparece cada categoría en el conjunto de datos. Luego, se recodifican las categorías con una frecuencia baja en ambas columnas. Aquellas categorías que aparecen menos de 1,000 veces en la columna "weather" se reemplazan por "Otro", y aquellas que aparecen menos de 3,000 veces en la columna "terrainType" también se reemplazan por "Otro". Esto simplifica las categorías y agrupa las menos frecuentes en una categoría común, "Otro".

Estas operaciones ayudan a reducir la complejidad de las categorías en las columnas "weather" y "terrainType, lo que facilita el análisis de los datos y puede mejorar el rendimiento de los modelos de aprendizaje automático al reducir la dimensionalidad del conjunto de datos.

Se está crea una nueva columna llamada "tipo" en el DataFrame "poke" para clasificar las clases de Pokémon en categorías de acuerdo a su rareza. Las categorías son las siguientes:

- "Common" (Común): Se asigna a las clases de Pokémon cuyos identificadores ("class") se encuentran en el rango de 13 a 20, o son igual a 41 o 42. Estas clases se consideran comunes.
- "Uncommon" (Poco Común): Se asigna a las clases de Pokémon que cumplen ciertas condiciones específicas. Esto incluye varias clases que tienen identificadores en diferentes rangos.
- "Very Rare" (Muy Raro): Se asigna a clases de Pokémon específicas que tienen identificadores determinados, como 88, 89, 106, 107, 108, 113, 129, 130, 137 y 142. Estas clases se consideran muy raras.
- "Super Rare" (Súper Raro): Se asigna a clases de Pokémon también específicas, con identificadores como 83, 132, 144, 145, 146, 150, 151, 115, 122 y 131. Estas clases se consideran súper raras.
- "Rare" (Raro): Cualquier clase de Pokémon que no cumple con las condiciones anteriores se clasifica como rara.

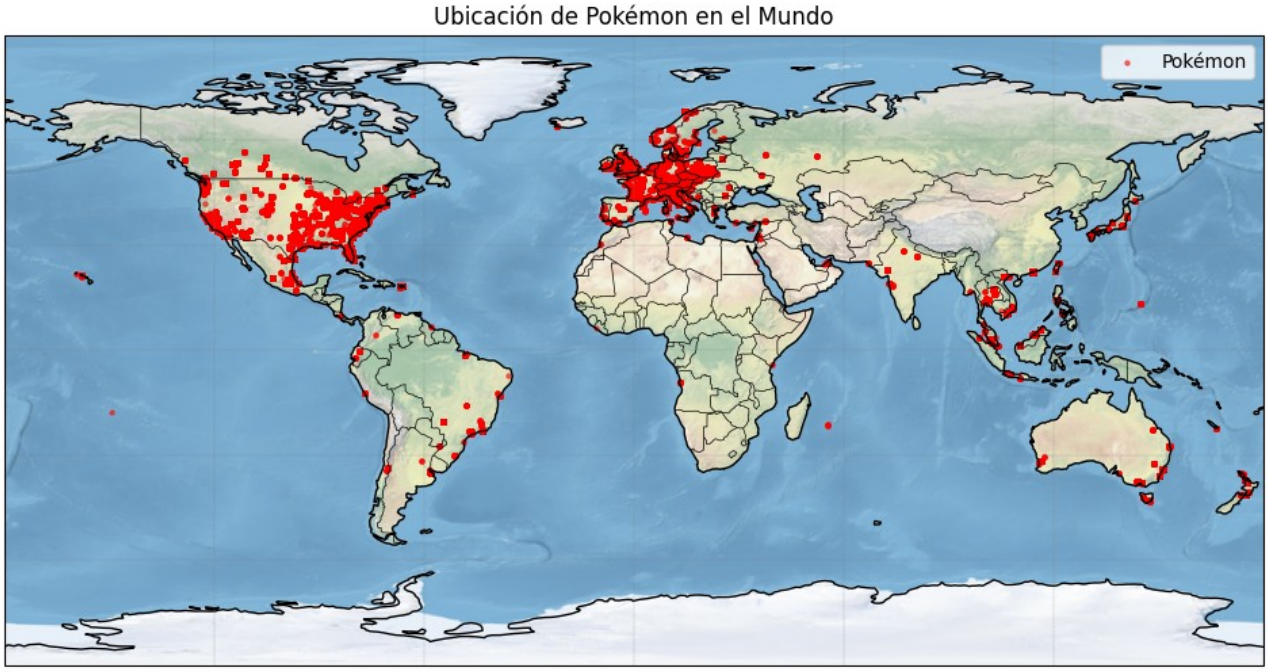
Se cambia el tipo de datos de las columnas "terrainType", "weather", "class" y "tipo" en el DataFrame "poke" para que sean del tipo "categórico" (category).

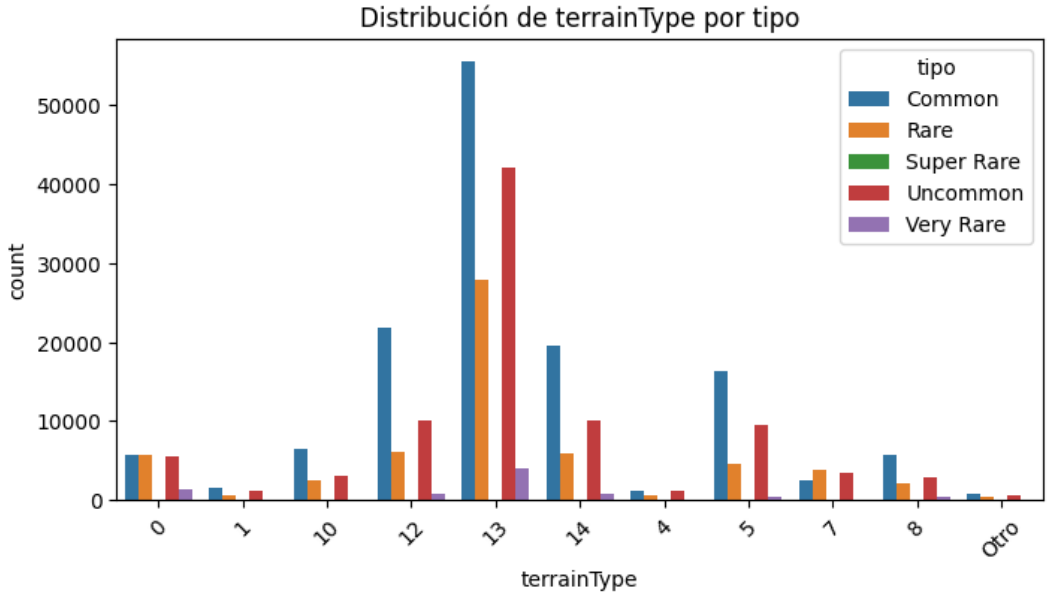
Se convierte la columna "pokestopDistanceKm" a valores numéricos, trata los valores que no se pueden convertir y rellena los valores faltantes con la media de los valores no faltantes en la misma columna. Esto asegura que la columna esté lista para su posterior análisis sin valores faltantes.

Se verificar la presencia de valores nulos en el DataFrame "poke" y muestra un mensaje apropiado en función del resultado de la verificación.

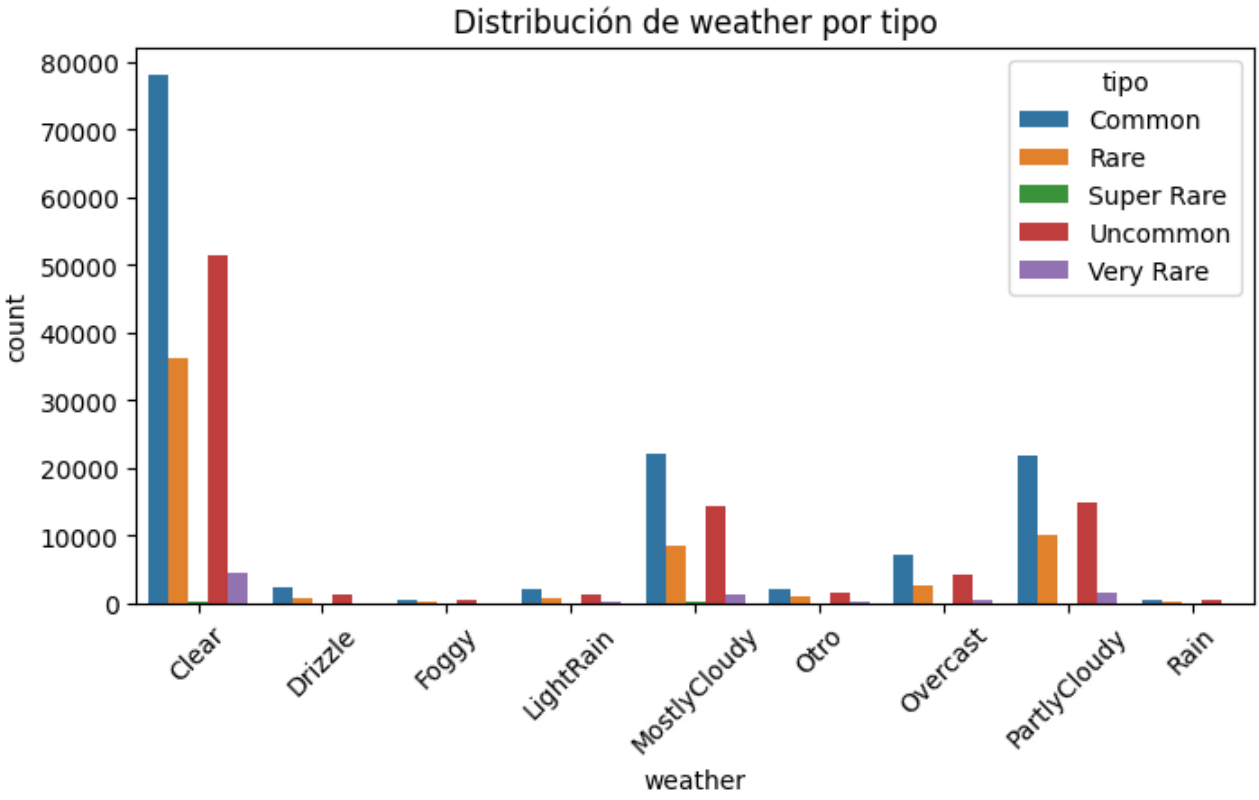
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 296021 entries, 0 to 296020
Data columns (total 19 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   latitude            296021 non-null  float64
 1   longitude           296021 non-null  float64
 2   terrainType         296021 non-null  category
 3   closeToWater        296021 non-null  bool
 4   weather             296021 non-null  category
 5   temperature         296021 non-null  float64
 6   windSpeed           296021 non-null  float64
 7   windBearing         296021 non-null  int64
 8   pressure            296021 non-null  float64
 9   population_density  296021 non-null  float64
 10  urban               296021 non-null  bool
 11  suburban            296021 non-null  bool
 12  midurban            296021 non-null  bool
 13  rural               296021 non-null  bool
 14  gymDistanceKm       296021 non-null  float64
 15  pokestopDistanceKm  296021 non-null  float64
 16  tipo                296021 non-null  category
 17  hour                296021 non-null  int32
 18  minute              296021 non-null  int32
dtypes: bool(5), category(3), float64(8), int32(2), int64(1)
memory usage: 24.8 MB
```

Exploración de Datos:

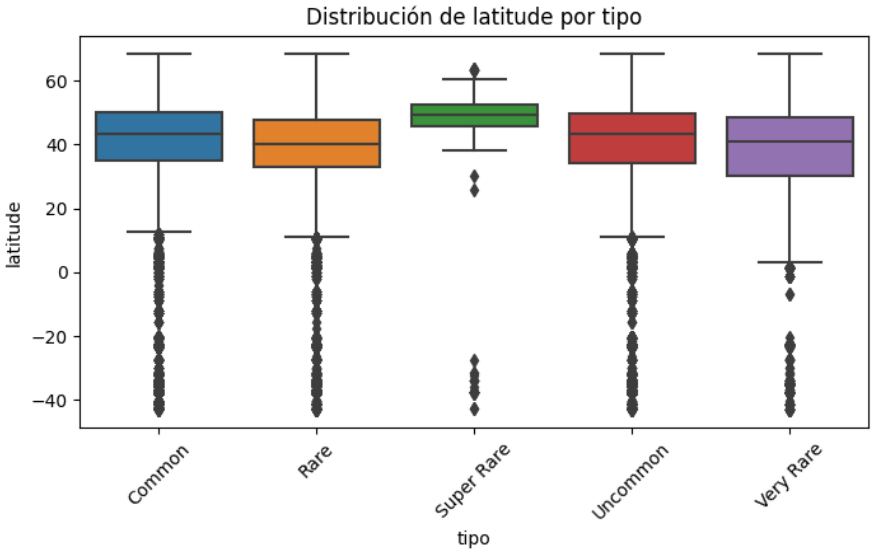




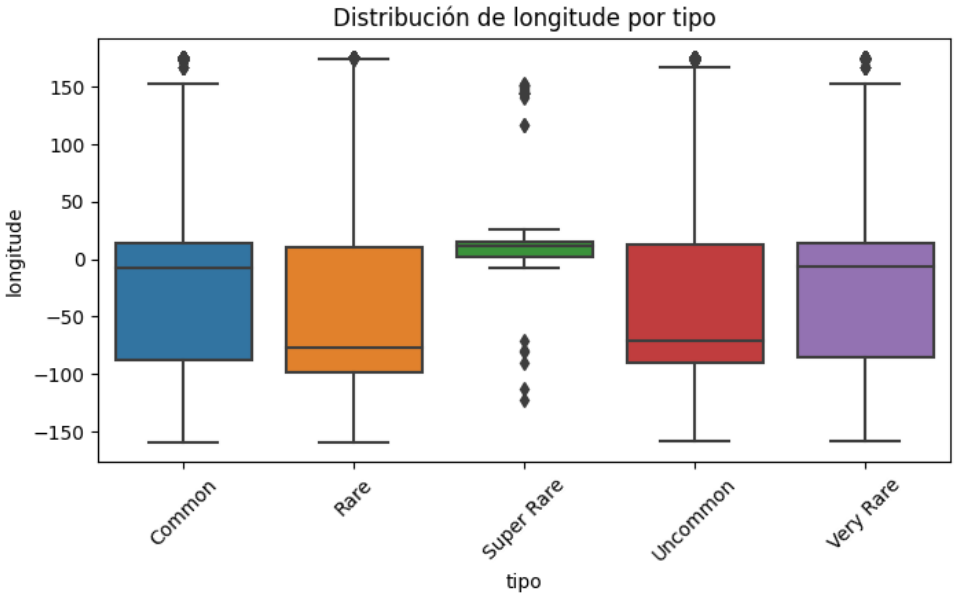
- En esta tabla, se muestra cómo se distribuyen las categorías de rareza de los Pokémon en función del tipo de terreno en el que aparecen.
- El tipo de terreno "13" es donde se encuentra la mayoría de los Pokémon, con una alta presencia de "Common" y "Uncommon," pero también con una proporción significativa de "Rare" y "Very Rare."
- Los tipos de terreno "0," "1," y "10" también tienen una presencia notable de "Common" y "Uncommon," pero tienden a tener menos "Rare" y "Super Rare."
- Los tipos de terreno "4" y "8" son los que muestran menos diversidad en términos de rareza, con una mayoría de "Common" y muy pocos "Super Rare" y "Very Rare."



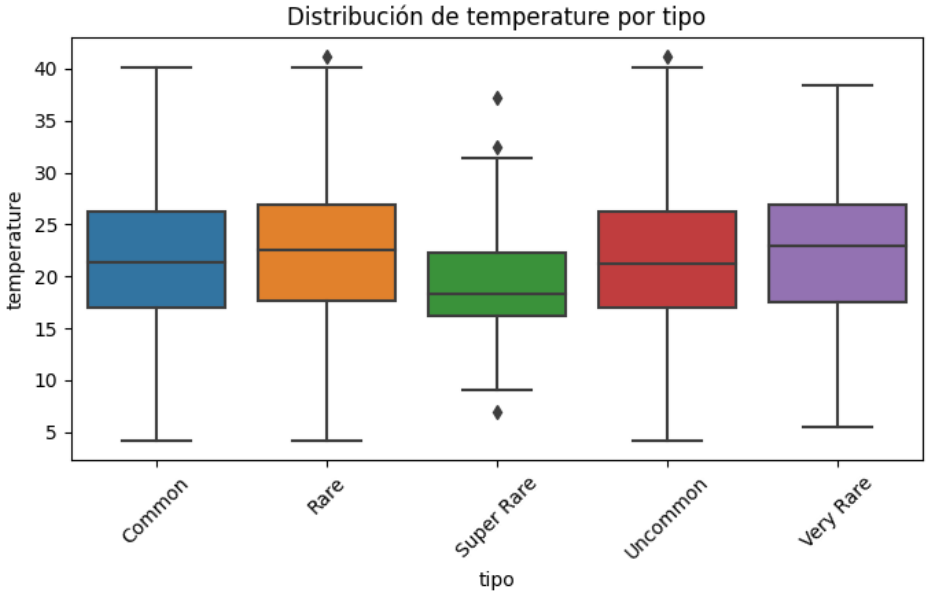
- Esta tabla muestra cómo la rareza de los Pokémon se relaciona con las diferentes condiciones climáticas en las que aparecen.
- En condiciones climáticas "Clear," la mayoría de los Pokémon son de las categorías "Common" y "Uncommon," lo que sugiere que es más probable encontrar Pokémon comunes en días despejados.
- Las condiciones climáticas "MostlyCloudy" también tienen una alta presencia de "Common" y "Uncommon," pero también una cantidad considerable de "Rare."
- En condiciones climáticas "Foggy," "Drizzle," y "Rain," la rareza de los Pokémon tiende a ser más equilibrada entre las categorías.
- "PartlyCloudy" muestra una mayor presencia de "Rare" en comparación con otras condiciones.
- "Otro" incluye varias condiciones climáticas no específicas y parece tener una distribución diversa de rareza.



- Para la variable "latitude," se observa que la media varía ligeramente entre las categorías de rareza de Pokémon. La categoría "Super Rare" tiene la media más alta de latitud, lo que sugiere que estos Pokémon tienden a aparecer en ubicaciones más septentrionales.
- Las desviaciones estándar también varían, lo que indica la dispersión de las observaciones. Las categorías "Rare" y "Super Rare" tienen desviaciones estándar relativamente altas, lo que significa que sus ubicaciones pueden ser más variadas.
- En general, se pueden identificar tendencias en la latitud de aparición de Pokémon en función de su rareza.



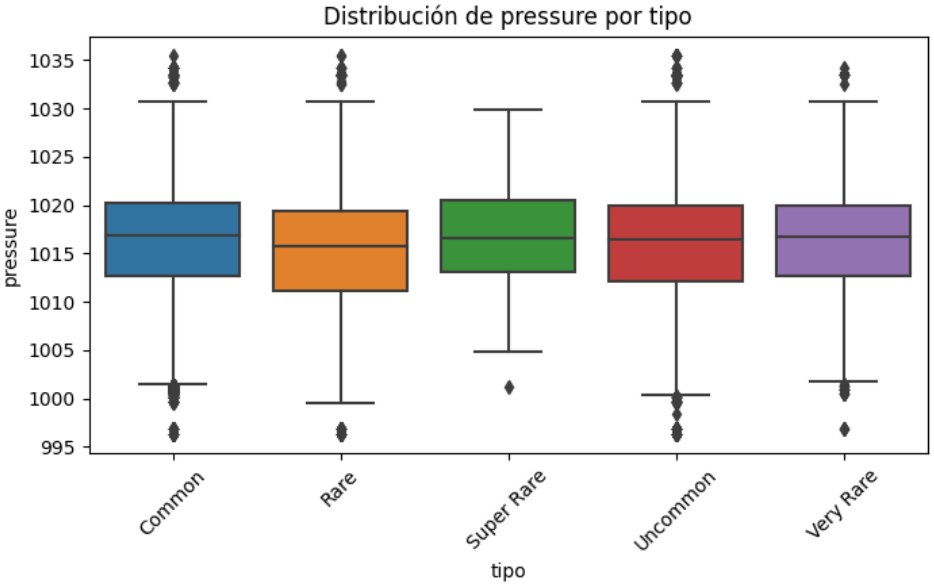
- Al igual que con la latitud, la media de la longitud varía entre las categorías de rareza. La categoría "Super Rare" tiene la media más alta de longitud, lo que sugiere que estos Pokémon tienden a aparecer en ubicaciones más orientales.
- Las desviaciones estándar también varían, lo que indica la dispersión de las observaciones. Nuevamente, las categorías "Rare" y "Super Rare" tienen desviaciones estándar relativamente altas.
- Las diferencias en la longitud de aparición de Pokémon pueden influir en su disponibilidad en diferentes regiones geográficas.



- Las

medias de temperatura varían ligeramente entre las categorías de rareza. Las categorías "Rare" y "Very Rare" tienen medias más altas de temperatura, lo que podría sugerir que estos Pokémon aparecen con más frecuencia en condiciones climáticas cálidas.

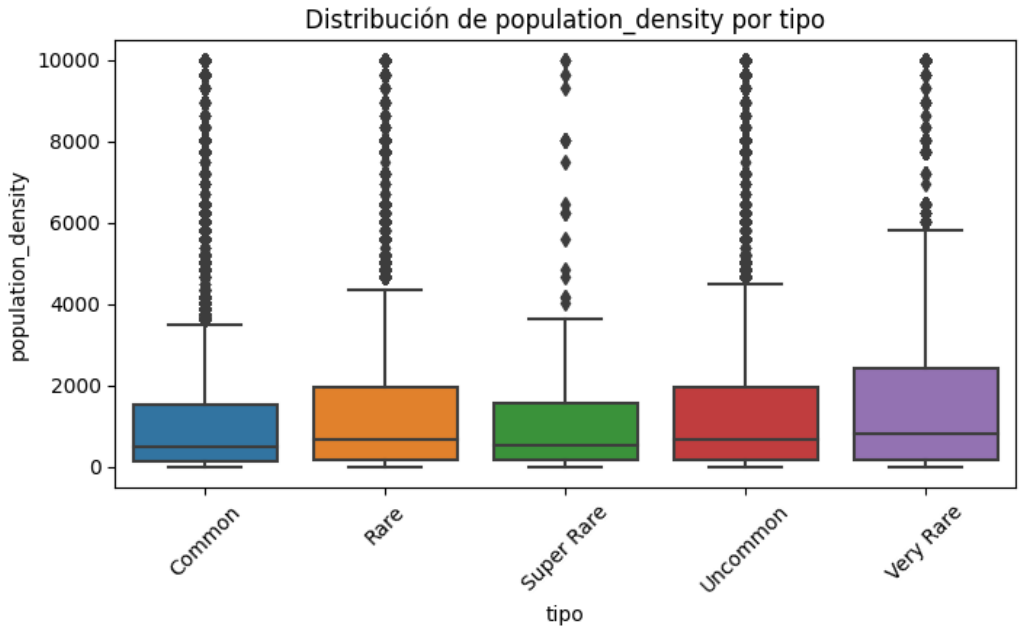
- Las desviaciones estándar indican la variabilidad de la temperatura en cada categoría de rareza. Nuevamente, "Rare" y "Very Rare" muestran una mayor variabilidad.
- Estas diferencias en la temperatura pueden influir en la preferencia de ciertos Pokémon por climas específicos.



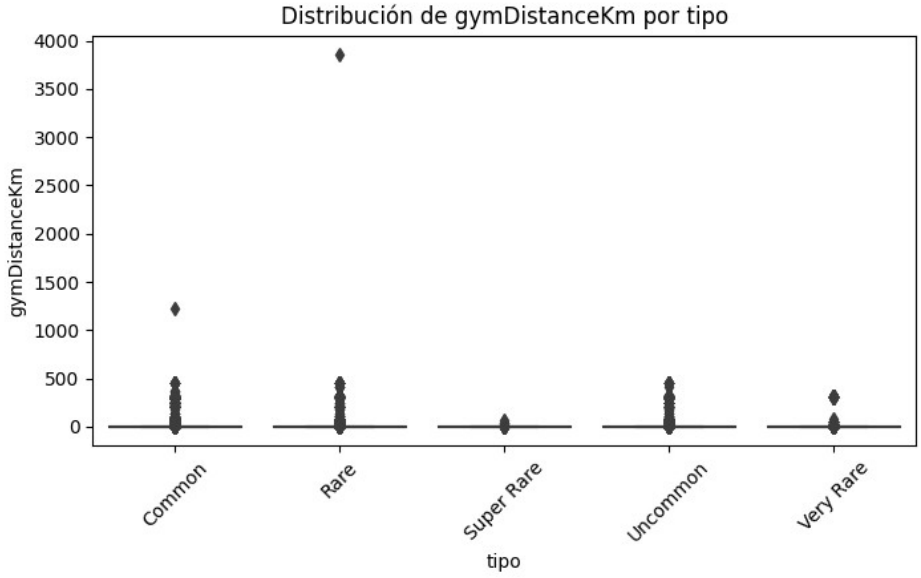
- La

media de la presión atmosférica varía ligeramente entre las categorías de rareza. Las categorías "Common" y "Super Rare" tienen medias ligeramente más altas de presión atmosférica.

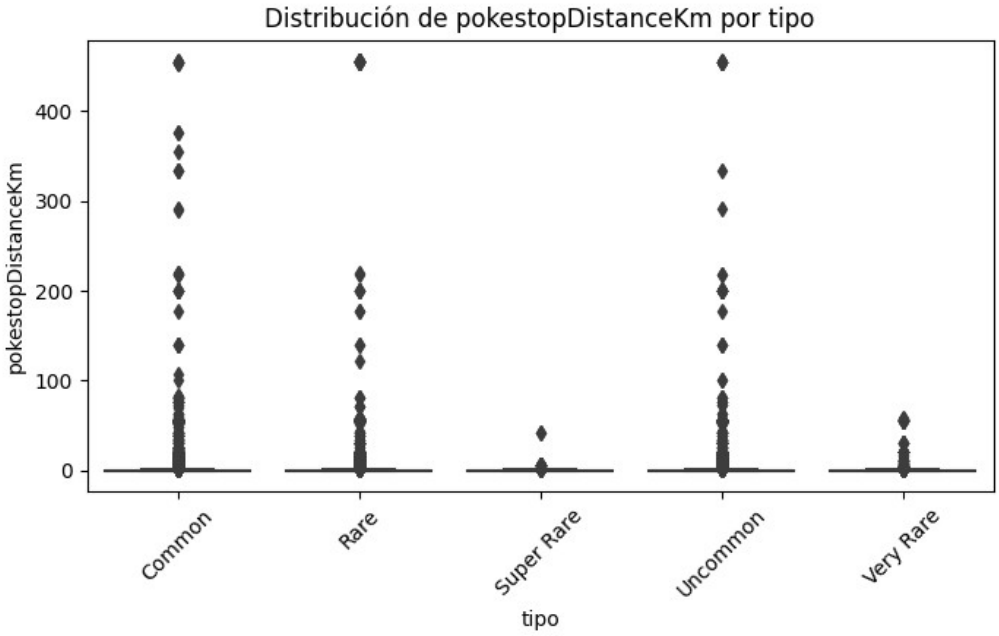
- Las desviaciones estándar también varían, lo que indica la dispersión de las observaciones. La categoría "Rare" muestra una desviación estándar más alta.
- Estas diferencias en la presión atmosférica pueden estar relacionadas con las condiciones climáticas en las que aparecen los Pokémon.



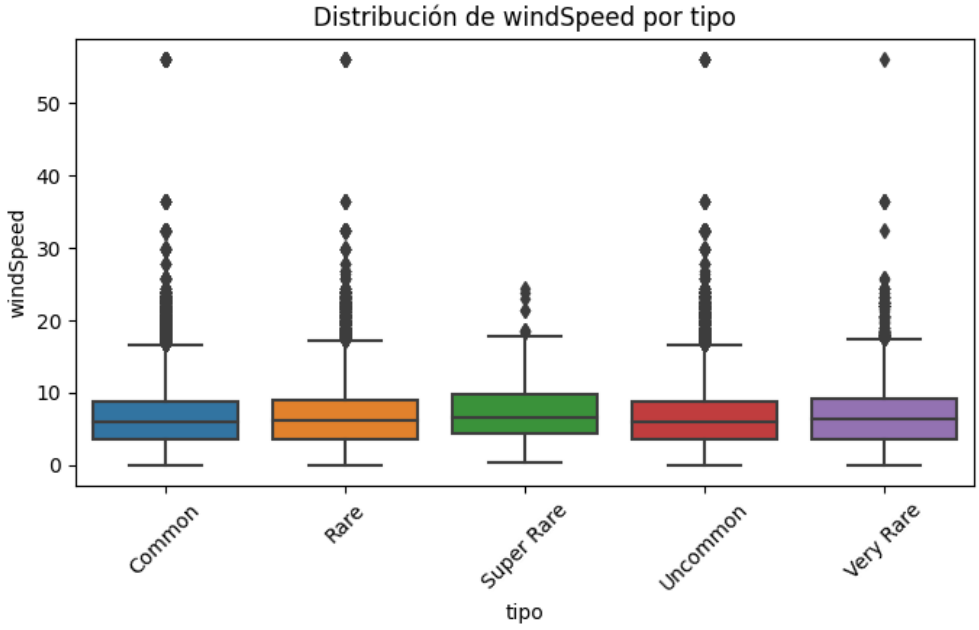
- Las medias de densidad de población varían entre las categorías de rareza. "Very Rare" tiene la media más alta de densidad de población.
- Las desviaciones estándar varían, y "Very Rare" muestra una desviación estándar relativamente alta.
- Las diferencias en la densidad de población pueden indicar que ciertas categorías de rareza prefieren aparecer en áreas más o menos densamente pobladas.



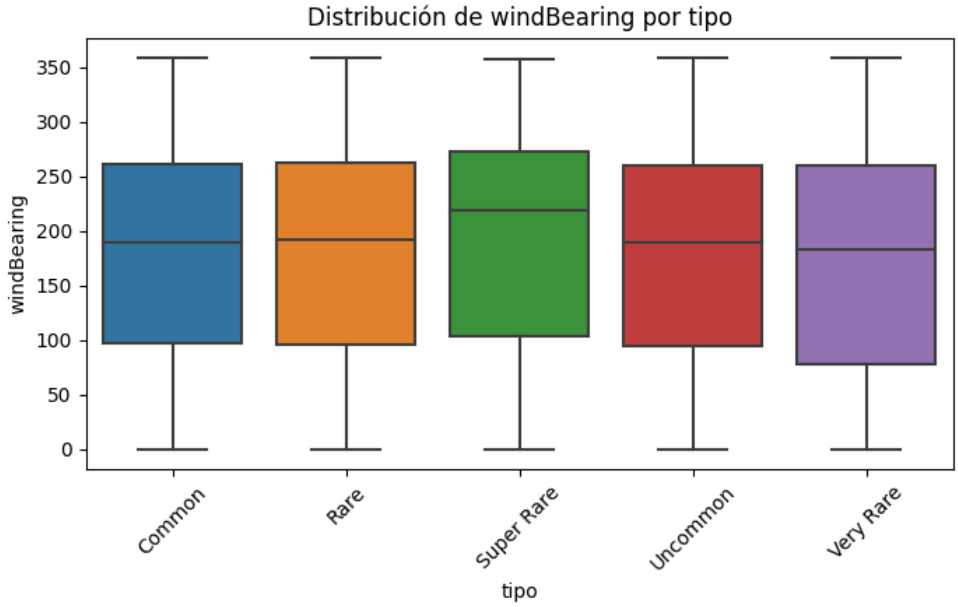
- La "media" de la distancia a gimnasios para todos los tipos de Pokémon es bastante similar, en el rango de 1.34 km a 2.41 km.
- La "desviación estándar" es relativamente alta para todos los tipos, lo que sugiere una variabilidad significativa en las distancias a los gimnasios dentro de cada tipo.
- El "mínimo" y el "máximo" indican las distancias más cortas y más largas, respectivamente, dentro de cada tipo.



Al igual que con la distancia a los gimnasios, la media de distancia a las Pokeparadas es similar entre los tipos de Pokémon, pero la variabilidad es alta. Esto significa que algunos Pokémon de cada tipo pueden estar ubicados mucho más lejos de las Pokeparadas que otros.



La velocidad del viento varía ligeramente entre los tipos de Pokémon, pero en general, los valores se mantienen en el rango de cero a unos pocos metros por segundo. La desviación estándar es relativamente baja, lo que sugiere que la velocidad del viento no varía mucho entre las muestras de diferentes tipos de Pokémon.



La dirección del viento se mide en grados y varía entre 0 y 359 grados, donde 0 representa el norte. Los resúmenes estadísticos indican que la dirección del viento tiende a ser diversa para todos los tipos de Pokémon, con medias alrededor de 180 grados, lo que sugiere que no hay una tendencia clara hacia una dirección específica del viento para ningún tipo

División de Datos:

En el proceso de preparación de datos, se han convertido variables categóricas en valores numéricos utilizando Label Encoding. También se han codificado la variable objetivo 'tipo'. Además, dividimos los datos en conjuntos de entrenamiento y prueba, siendo el 80% para entrenar modelos y el 20% para evaluar su rendimiento.
(X_train, y_train) y prueba (X_test, y_test)

Se aplicó `LabelEncoder` a las variables categóricas 'hour', 'minute', 'weather' y 'terrainType' en el conjunto de datos 'pok'. Luego, se codificó la variable objetivo 'tipo'. Finalmente, se dividió el conjunto de datos en conjuntos de entrenamiento y prueba (80% entrenamiento, 20% prueba) con un valor de `random_state` de 42 para garantizar la reproducibilidad.

(X_trainx, y_trainx) y prueba (X_testx, y_testx)

Modelado:

1. `model1`:
 - Configuración: Clasificador XGBoost con hiperparámetros específicos.
 - Entrenamiento en `X_train` y `y_train`.
 - Evaluación en conjuntos de entrenamiento y prueba.

2. `model2`:
 - Similar a `model1`, pero con una función objetivo diferente ('multi:softprob').

3. `model3`:
 - Lo mismo que `model2`, pero con pesos de muestra (pesos de clase) tenidos en cuenta durante el entrenamiento.

4. `model4`:
 - Similar a `model2`, pero utilizando datos diferentes (`X_trainx` y `y_trainx`).

(1Er búsqueda de hiperparámetros para un clasificador XGBoost con un conjunto de parámetros y luego aplicando la validación cruzada estratificada (Stratified K-Fold))

5. `best_model`:
 - Configuración con hiperparámetros específicos.
 - Entrenamiento en `X_train` y `y_train` con pesos de muestra.
 - Utiliza un conjunto personalizado de parámetros (`best_params`).

6. `best_model2`:
 - Similar a `best_model`, pero entrenado en diferentes datos (`X_trainx` y `y_trainx`).

7. `best_model3`:
 - Similar a `best_model`, pero sin pesos de muestra.

8. `best_model4`:
 - Otra versión de `best_model2`, pero sin pesos de muestra.

9. `best_modelx4`:
 - Similar a `best_model4`, pero con pesos de muestra (`wb`) y hiperparámetros específicos adicionales.

10. `best_modelxw4`:
 - Similar a `best_modelx4`, pero con hiperparámetros adicionales.

(búsqueda exhaustiva de hiperparámetros para un clasificador XGBoost. Se emplea la estrategia de validación cruzada estratificada con 10 divisiones. Se exploran múltiples hiperparámetros, como 'subsample', 'num_class', 'colsample_bynode', 'learning_rate', 'max_depth', 'min_child_weight', 'gamma', 'n_estimators', 'booster', 'max_delta_step', 'eval_metric', 'max_leaves', 'max_bin', 'alpha', y 'lambda'. El objetivo es encontrar la combinación de hiperparámetros que minimice la pérdida logarítmica negativa ('neg_log_loss').)

11. `best_modelg`:
 - Configuración con hiperparámetros específicos y cambios adicionales en comparación con los modelos anteriores.
 - Entrenamiento en `X_trainx` con pesos de muestra.

12. `best_modelgg`:
    - Similar a `best_modelg`, pero sin pesos de muestra.

13. Regresión Logística (Logistic Regression), Bosque Aleatorio (Random Forest), k-Vecinos Más Cercanos (KNN) y Máquinas de Soporte Vectorial (SVM) a partir de las matrices X_train y X_test.

13. Regresión Logística (Logistic Regression), Bosque Aleatorio (Random Forest), k-Vecinos Más Cercanos (KNN) y Máquinas de Soporte Vectorial (SVM) a partir de las matrices X_trainx y X_testx.

Evaluación de los Modelos:

Modelo 1:

- Precisión promedio: 0.37
- Recall promedio: 0.28
- Valor F1 promedio: 0.28

Modelo 2:

- Precisión promedio: 0.37
- Recall promedio: 0.27
- Valor F1 promedio: 0.27

Modelo 3:

- Precisión promedio: 0.30
- Recall promedio: 0.37
- Valor F1 promedio: 0.29

Modelo 4:

- Precisión promedio: 0.37
- Recall promedio: 0.27
- Valor F1 promedio: 0.27

Modelo 5:

- Precisión promedio: 0.30
- Recall promedio: 0.37
- Valor F1 promedio: 0.30

Modelo 6:

- Precisión promedio: 0.30
- Recall promedio: 0.37
- Valor F1 promedio: 0.30

Modelo 7:

- Precisión promedio: 0.37
- Recall promedio: 0.28
- Valor F1 promedio: 0.28

Modelo 8:

- Precisión promedio: 0.37
- Recall promedio: 0.28
- Valor F1 promedio: 0.28

Modelo 9:

- Precisión promedio: 0.37
- Recall promedio: 0.28
- Valor F1 promedio: 0.28

Modelo 10:

- Precisión promedio: 0.37
- Recall promedio: 0.28
- Valor F1 promedio: 0.28

Modelo 11:

- Precisión promedio: 0.29
- Recall promedio: 0.39
- Valor F1 promedio: 0.23

Modelo 12:

- Precisión promedio: 0.27
- Recall promedio: 0.25
- Valor F1 promedio: 0.23

Logistic Regression 1:

- Precisión promedio: 0.23
- Recall promedio: 0.20
- Valor F1 promedio: 0.14

Random Forest 1:

- Precisión promedio: 0.36
- Recall promedio: 0.26
- Valor F1 promedio: 0.26

KNN 1:

- Precisión promedio: 0.27
- Recall promedio: 0.25
- Valor F1 promedio: 0.25

SVM 1:

- Precisión promedio: 0.17
- Recall promedio: 0.21
- Valor F1 promedio: 0.15

Logistic Regression 2:

- Precisión promedio: 0.22
- Recall promedio: 0.20
- Valor F1 promedio: 0.14

Random Forest 2:

- Precisión promedio: 0.36
- Recall promedio: 0.26
- Valor F1 promedio: 0.26

KNN 2:

- Precisión promedio: 0.28
- Recall promedio: 0.25
- Valor F1 promedio: 0.25

SVM 2:

- Precisión promedio: 0.17
- Recall promedio: 0.21
- Valor F1 promedio: 0.15

Conclusión:

Basado en estas métricas y sin conocer el contexto específico del problema, parece que los "Modelos 1, 2 y4 " y "Modelos Random Forest 1 y 2" son los que tienen un rendimiento relativamente mejor. Estos modelos tienen una precisión promedio y un valor F1 más altos en comparación con otros modelos. Sin embargo, el rendimiento general de los modelos parece ser modesto.

Si se tuviera que escoger alguno sería:

Modelo 2:

- Tiene un rendimiento equilibrado en términos de precisión y recall para la mayoría de las clases.
- Tiene un desempeño razonable para la clase 0 y la clase 3.

Sin embargo el conjunto de datos presenta una distribución de clases altamente desequilibrada, donde las clases "Common" y "Uncommon" son mayoritarias, mientras que las clases "Super Rare" son extremadamente escasas. Este desequilibrio en la distribución de clases plantea un desafío para los modelos de clasificación.

Hay que añadir las complicaciones que han habido con la capacidad computacional que requieren estos modelos de clasificación para esta base de datos.