MARINE LIFE FORECASTING

David González Pérez

1. Introducción

Hace algunos años tuve la oportunidad de obtener el título de buceo, y desde entonces se ha convertido en una de mis aficiones preferidas.

Existen varias razones por las que se ha convertido en una de mis aficiones: la sensación de ingravidez junto con la posibilidad de moverte en tres dimensiones en lugar de dos; el silencio, que te permite quedarte solo con tus pensamientos; y la más importante, la vida bajo el agua.

Bajo el agua los animales parecen de otro planeta, completamente diferentes a lo que hay en la superficie.

Esta última parte es la que ha motivado el objetivo de mi proyecto.

Siempre me ha interesado saber qué vida me iba a encontrar en las zonas de buceo a las que voy. Antes de cada inmersión hay un briefing donde se explica el recorrido y la vida que se va a ver. Algunas de estas especies son diferentes según la época del año e incluso de la hora, pues no es lo mismo una inmersión diurna que una nocturna.

El objetivo de este trabajo es tratar de construir un **modelo de clasificación multi-etiqueta** que prediga los animales que se pueden encontrar en una zona de buceo dependiendo de la localidad, la profundidad, la hora y la época del año.

1. Fuentes de datos

Los datos han sido obtenidos de las siguientes fuentes.

1.1. GBIF

GBIF es una organización internacional y una red de datos financiada por gobiernos de todo el mundo, destinada a proporcionar a cualquier persona, en cualquier lugar, acceso abierto y gratuito a datos sobre cualquier tipo de forma de vida que hay en la Tierra.

GBIF organiza todas esas fuentes mediante el uso del estándar Darwin Core.

De entre toda la información que proporciona esta red de datos, he utilizado el dataset “Diveboard - Scuba diving citizen science observations” y la API de GBIF para ayudarme en el desarrollo de este proyecto.

1.1.1. Diveboard - Scuba diving citizen science observations

Este dataset contiene los datos principales utilizados para este trabajo.

Está formado por todas las observaciones hechas por los usuarios de Diveboard desde 1970 hasta finales de 2020 a lo largo de todo el mundo.

La finalidad de estas observaciones es reflejar aquello que les ha llamado la atención durante las inmersiones, lo que produce un sesgo en los datos hacia especies conocidas, interesantes o extrañas.

1.1.2. Species API

La API de GBIF es una API basada en RESTful JSON.

Se ha utilizado para enriquecer el dataset con los nombres comunes de las especies, ya que en los datos aparecen la especie y sus diferentes taxonomías.

1.2. Natural Earth

Natural Earth se compone de un conjunto de datasets de mapas de dominio público a diferentes escalas.

Estos datos se han descargado mediante la librería Cartopy para comprobar si las coordenadas están sobre el agua o sobre tierra. En la página de descargas se pueden ver los datos disponibles.

1.3. Open Street Map

OpenStreetMap es un mapa del mundo, de uso libre bajo una licencia abierta.

Los datos de Open Street Map se han descargado mediante la librería Geopy, más concretamente utilizando Nominatum, y se han utilizado para obtener las coordenadas de los lugares que inserta el usuario en la aplicación final.

1.4. Wikipedia

Wikipedia es un proyecto enciclopedia web multilingüe de contenido libre basado en un modelo de edición abierta.

La Wikipedia se ha utilizado para descargar los datos y fotos de los animales que se muestran en la aplicación, mediante la librería wikipedia de Python.

2. Análisis de datos y preprocesado

2.1. Información general

“Diveboard - Scuba diving citizen science observations” es un dataset de 38.324 filas y 249 columnas.

Las observaciones están categorizadas por diferentes taxonomías:

- Reino (kingdom)
- Filo (phylum)
- Clase (class)
- Orden (order)
- Familia (family)
- Género (genus)
- Especie (species)

Como el objeto de este trabajo está centrado en la clasificación de animales, se filtran los datos, manteniendo únicamente aquellas observaciones del reino animal. Éste es el número de observaciones por Reino:

- **Animalia** **36.235**
- Incertae sedis 45
- Plantae 29
- Chromista 15

2.2. Calidad de los datos

2.2.1. Columnas vacías

En un primer vistazo se aprecia enseguida que hay muchas columnas vacías, en concreto, 167. Esto es debido a que el fichero sigue el estándar Darwin Core y los datos han sido recogidos por usuarios de la aplicación Diveboard, que no sigue estos estándares.

2.2.2. Columnas con pocos valores

A pesar de eliminar las columnas vacías, siguen existiendo columnas que contienen información poco valiosa, como, por ejemplo, las que se muestran a continuación:

```
license values: ['CC_BY_NC_4_0']
rights values: ['http://creativecommons.org/publicdomain/zero/1.0/']
rightsHolder values: ['Diveboard']
type values: ['Event']
datasetID values: ['https://ipt.diveboard.com/resource.do?r=diveboard-occurrences'
 'http://ipt.diveboard.com/resource.do?r=diveboard-occurrences']
institutionCode values: ['Diveboard']
collectionCode values: ['Diveboard']
datasetName values: ['Diveboard - Scuba diving citizen science']
ownerInstitutionCode values: ['Diveboard']
basisOfRecord values: ['HUMAN_OBSERVATION']
occurrenceStatus values: ['PRESENT']
occurrenceRemarks values: ['Non validated citizen science record']
samplingProtocol values: ['citizen science scuba dive observation']
coordinateUncertaintyInMeters values: [100.]
georeferenceSources values: ['Google Maps']
nameAccordingTo values: ['EOL']
kingdom values: ['Animalia']
taxonRank values: ['SPECIES' 'FAMILY' 'GENUS' 'ORDER' 'SUBSPECIES' 'FORM' 'VARIETY' 'CLASS'
 'PHYLUM']
taxonomicStatus values: ['ACCEPTED' 'SYNONYM' 'DOUBTFUL']
datasetKey values: ['66f6192f-6cc0-45fd-a2d1-e76f5ae3eab2']
publishingCountry values: ['FR']
elevationAccuracy values: [0.]
hasCoordinate values: [ True]
hasGeospatialIssues values: [False  True]
kingdomKey values: [1]
protocol values: ['DWC_ARCHIVE']
lastCrawled values: ['2020-12-11T15:02:29.629Z']
repatriated values: [True False nan]
```

Es por ello que estas columnas se descartan.

2.2.3. Columnas con información redundante

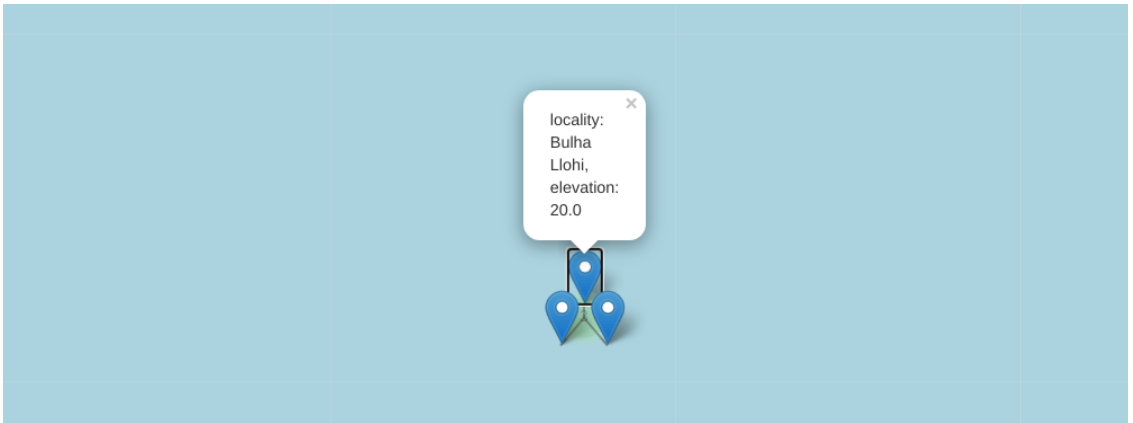
Las columnas *depth* y *depthAccuracy* tienen la misma información, y lo mismo ocurre con la fecha de *eventDate* y *dateIdentified*, por lo que únicamente se mantienen *depth* y *eventDate*.

2.2.4. Columnas con información incorrecta

2.2.4.1. Elevation

Al representar los datos en un mapa se puede comprobar que existen valores superiores a 0 metros en zonas sobre el mar:

Ejemplo:



Por esta razón se descarta el uso de esta columna.

2.2.4.2. Location, Water body y Country code

En estos campos existen valores nulos. Para imputar estos valores se ordenan los registros ascendentemente por longitud y latitud, y se completan con el valor de registro anterior.

2.2.4.3. Coordenadas

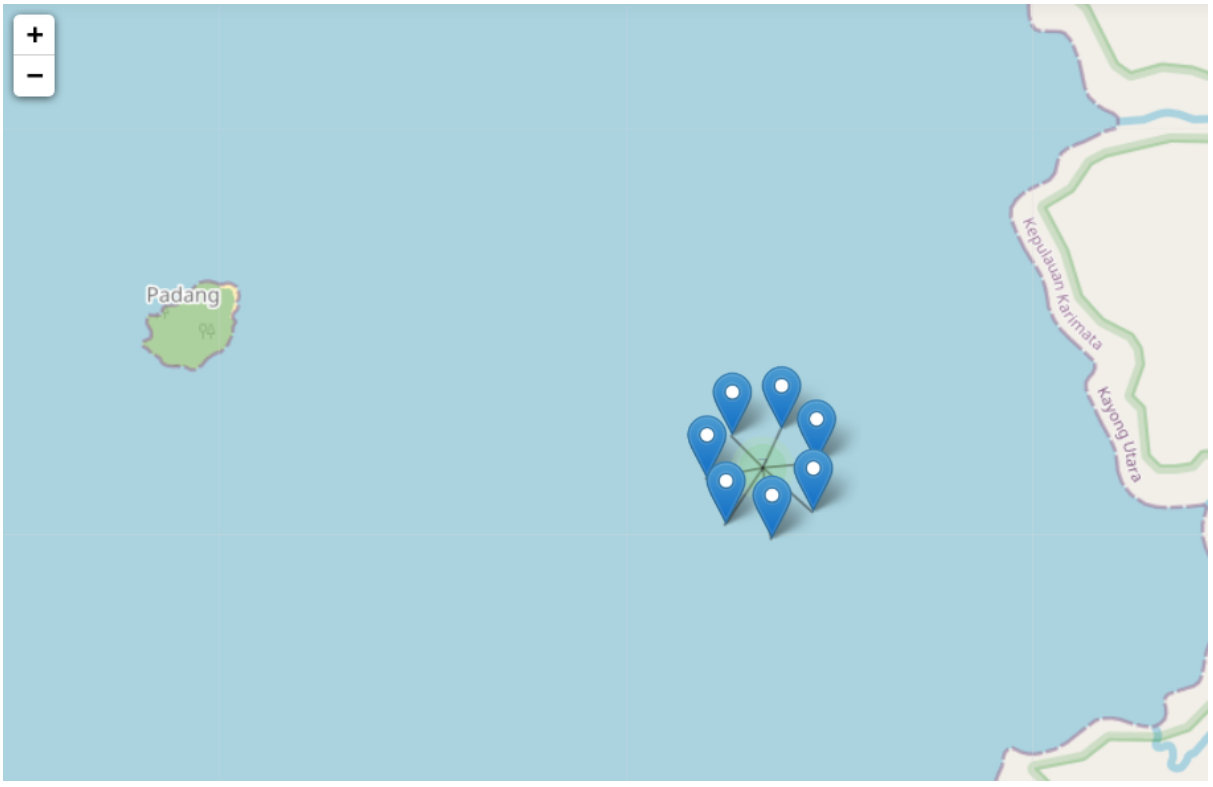
Estos dos valores, latitud y longitud, han sido los campos con mayor dificultad para imputar sus valores.

Existen 158 filas con coordenadas (0,0). Para estos registros se han buscado otros registros con el mismo valor de código de país y localidad, y se les ha asignado esas coordenadas.

Una vez imputadas las coordenadas (0,0), se han eliminado los registros que se han quedado con dichas coordenadas.

Posteriormente, y utilizando la cartografía de Natural Earth mediante la librería Cartopy, se ha intentado detectar las coordenadas que están fuera del agua, pero el mapa no es lo suficientemente preciso como para utilizarlo (ya que aparecen coordenadas que están sobre el mar que marca como tierra). Además, hay observaciones que se han etiquetado en el centro de buceo o en la localidad, en lugar de en el punto o zona de buceo, así que no tiene sentido filtrar por este criterio.

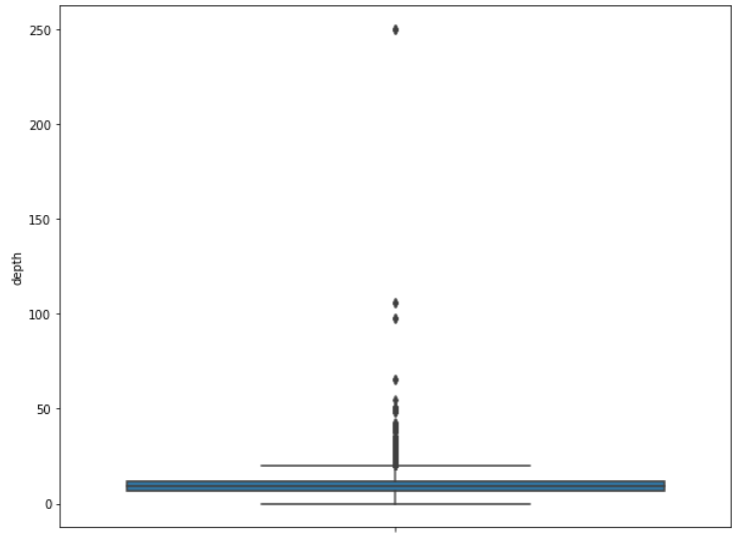
Ejemplo de coordenadas marcadas como tierra y que pertenecen realmente a zona de agua:



2.2.4.4. Depth

La profundidad máxima para el buceo recreativo son 40 metros, y sólo se puede sobrepasar en caso de ser buceador técnico, hasta los 65 metros.

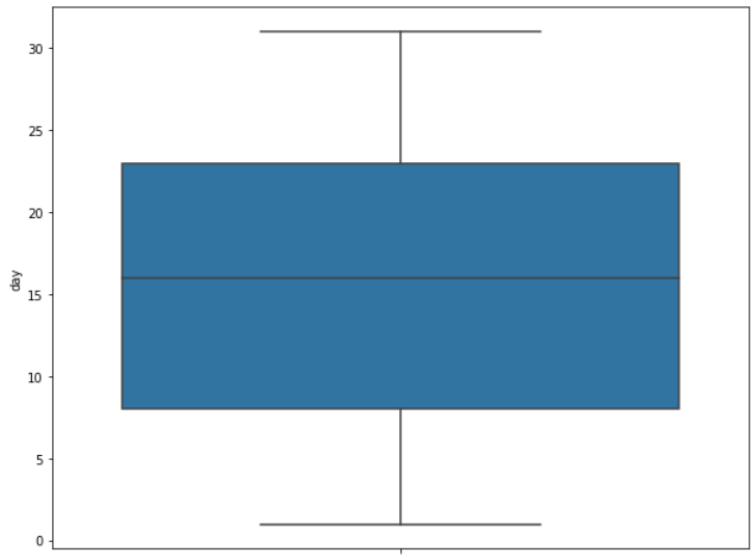
Los valores de la profundidad se reparten de la siguiente manera:



Como el público objetivo de la aplicación va a ser el de buceadores recreativos, se establece el límite máximo en 40 metros. Los valores que lo superen se consideran incorrectos, y se les va a imputar la profundidad máxima, que son 40 metros.

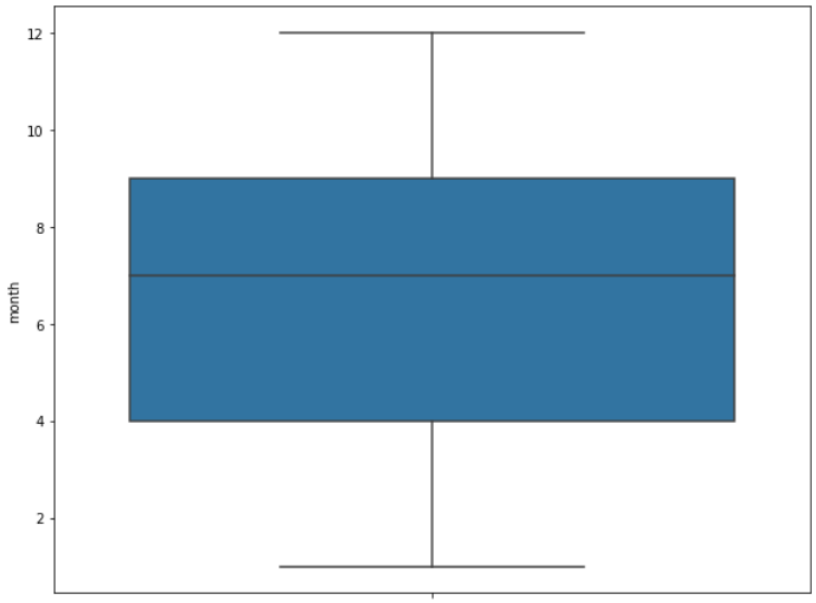
2.2.4.5. Day

El campo day tiene valores comprendidos entre 1 y 31 y su distribución es la siguiente:



2.2.4.6. Month

Sus valores están entre 1 y 12. Se distribuyen de la siguiente manera:



2.3. Asignación de un identificador a cada inmersión

El target del modelo es clasificar los animales por inmersión, y como el dataset está desglosado a nivel de observación (avistamiento), hay que agregar los registros por inmersión.

Para ello nos podemos fijar en el campo *references*. Este campo contiene una url única por cada inmersión (y que conduce a los datos de esa inmersión en Diveboard - ejemplo: https://www.diveboard.com/zylantha/D3D8Nbu).

Para crear el identificador se utiliza la opción de pandas de convertir a categoría el campo *references* y asignarle un código por cada categoría del campo.

Para los registros que tienen valor nulo se agregan los registros por los campos *eventDate*, *decimalLatitude*, *decimalLongitude* y *depth*, y se les asigna un identificador cuyo primer valor es el del último valor asignado en el paso anterior, incrementado en 1.

2.4. Selección de columnas definitivas

Tras todo este análisis, se desechan las columnas ya mencionadas y todas las que tienen información sobre el dataset pero no de las observaciones.

Las columnas finales seleccionadas del fichero son:

- eventDate - para obtener la hora
- month
- day
- waterBody
- countryCode
- locality
- dive_id

- decimalLatitude - necesario para el KNNImputer que se usará en la app
- decimalLongitude - necesario para el KNNImputer que se usará en la app
- depth
- orderKey - necesario para obtener los nombres comunes de los animales
- familyKey - necesario para obtener los nombres comunes de los animales
- genusKey - necesario para obtener los nombres comunes de los animales
- speciesKey - necesario para obtener los nombres comunes de los animales

3. Enriquecimiento del dataset

Debido a que el usuario final de la aplicación va a ser un buceador y no un biólogo, se hace necesario convertir los datos sobre la taxonomía de los animales en nombres comunes, que todos puedan entender.

En este paso, se va a descargar la información correspondiente a los nombres comunes de los animales utilizando la API de GBIF. Se agregarán los datos por inmersión y se asignará la profundidad para aquellas inmersiones que tengan valores erróneos que no se hicieron en el paso anterior.

Una vez hecho esto, se hará una exploración de los datos para buscar *insights*.

3.1. Nombres comunes de los animales

Es en este punto donde se aprovecha la información de las claves de la taxonomía que se mantuvieron en el paso anterior. Aprovechando los campos:

- orderKey
- familyKey
- genusKey
- speciesKey

se pasan como parámetro a la API para descargar el nombre común de cada una de las taxonomías de cada animal.

La intención es obtener el nombre común genérico que permita distinguir unos animales de otros.

Como existen valores nulos en todas las taxonomías, el orden de preferencia para establecer el nombre común es el siguiente:

1. Familia
2. Orden
3. Género
4. Especie

3.2. Extracción de la hora

Se emplea el campo *eventDate* para extraer la hora de inmersión (*hour)*.

3.3. Selección de target

Como target para el modelo predictivo se eligen los cinco animales más frecuentes del dataset. Estos son:

1. Moray eels
2. Firefishes
3. Damselfishes
4. Sea turtles
5. Groupers

3.4. Agregación del dataset por inmersión

Como ya se ha dicho, el dataset está a nivel de observación de animales, pero el objetivo es la inmersión. Por eso, se agrega el dataset por el identificador de inmersión asignado anteriormente y se establecen las columnas de los target de la siguiente manera: si uno de los 5 animales es visto en una inmersión se informa con 1, en caso contrario, con 0.

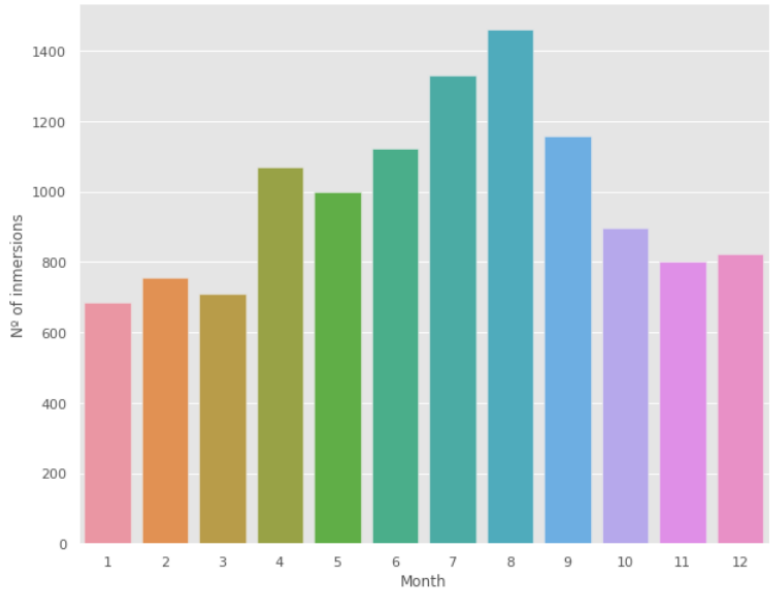
4. Insights

Para una mejor comprensión del dataset, se busca información que no se vea únicamente mirando el contenido de sus columnas.

4.1. Inmersiones

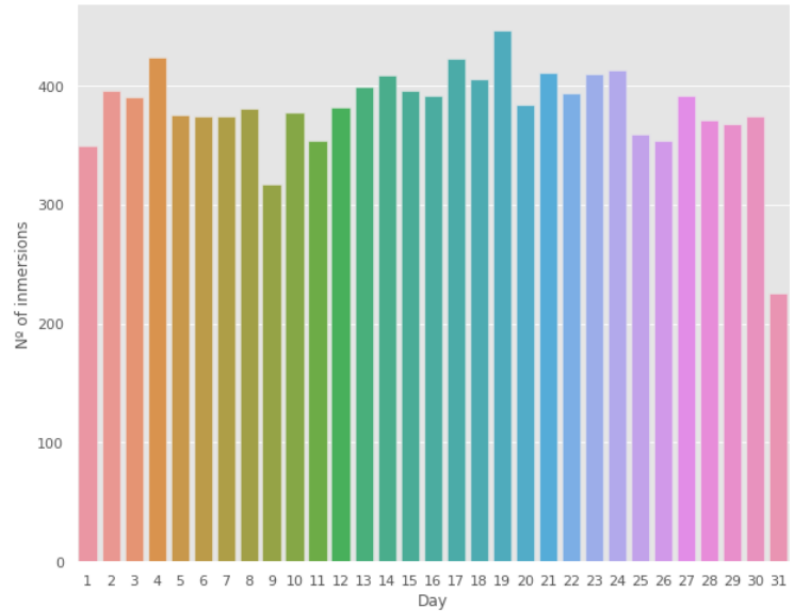
4.1.1. Inmersiones por mes

La mayoría de las inmersiones se realizan en los meses de verano, que suele coincidir con las vacaciones en el hemisferio norte. Esto tiene sentido, puesto que son los meses en los que mejor está la temperatura del agua para bucear, y además, la gente tiene más facilidad para ir.



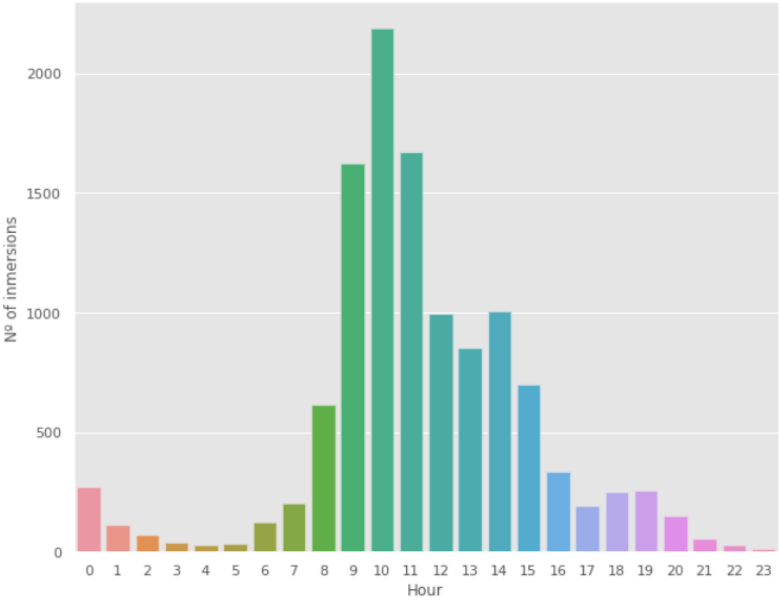
4.1.2. Inmersiones por día del mes

Se puede observar que se trata de una distribución uniforme de las inmersiones, exceptuando el día 31, ya que no todos los meses tienen 31 días.



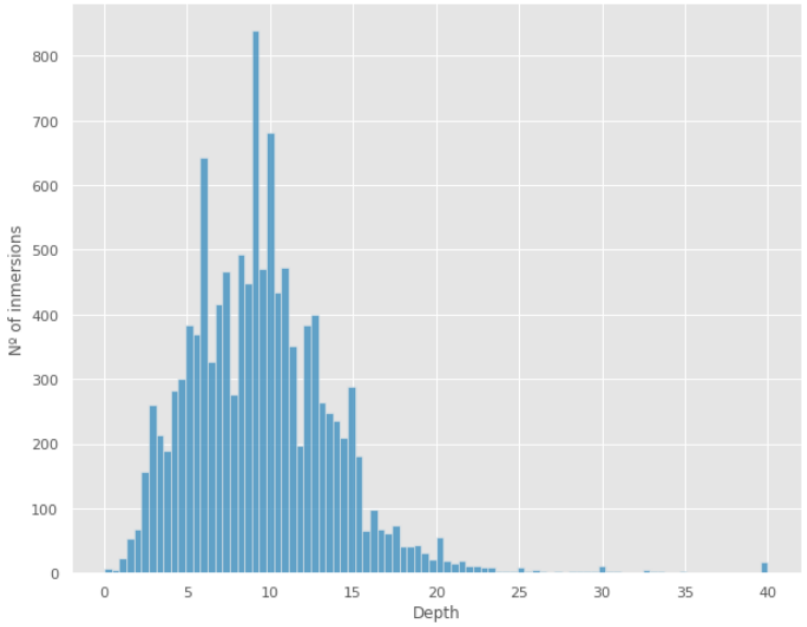
4.1.3. Inmersiones por horas del día

La mayoría de las inmersiones se producen por la mañana, entre las 9 y las 12 horas.



4.1.4. Distribución de las inmersiones por profundidad

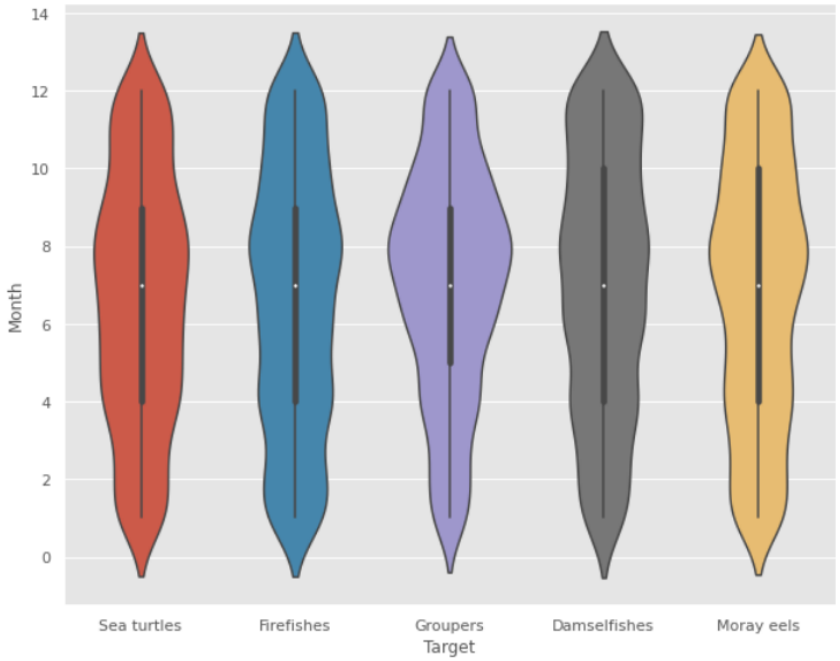
Una gran parte de las inmersiones se mueven entre los 5 y los 15 metros de profundidad máxima.



4.2. Target

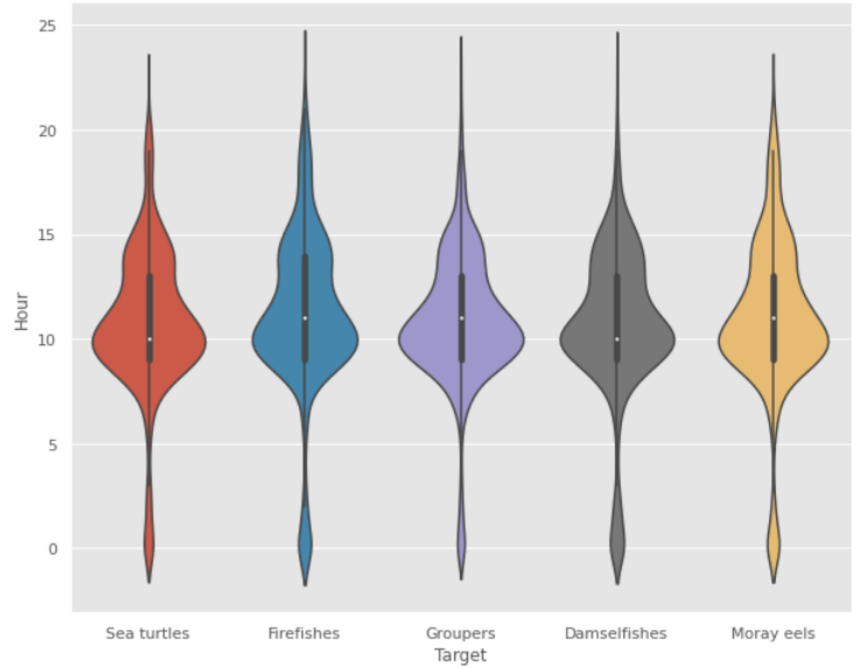
4.2.1. Meses

Con las observaciones de los animales ocurre lo mismo que con las inmersiones, son más habituales en los meses de verano. La única pequeña variación es que los meros (groupers) son más habituales en agosto.



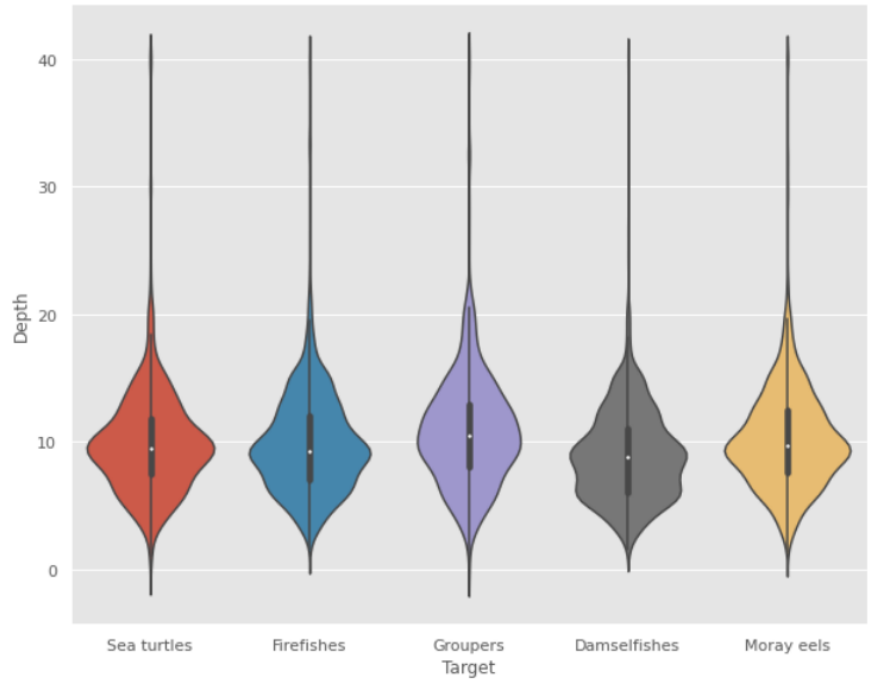
4.2.2. Horas

Igual que las inmersiones, las observaciones se dan más a menudo entre las 9 y las 12 de la mañana.



4.2.3. Profundidad

También tienen la misma distribución que las inmersiones, y son más habituales las observaciones entre los 5 y los 15 metros de profundidad.



5. Feature engineering

En esta parte del proceso se van a preparar las variables para extraer la máxima información disponible para entrenar el modelo.

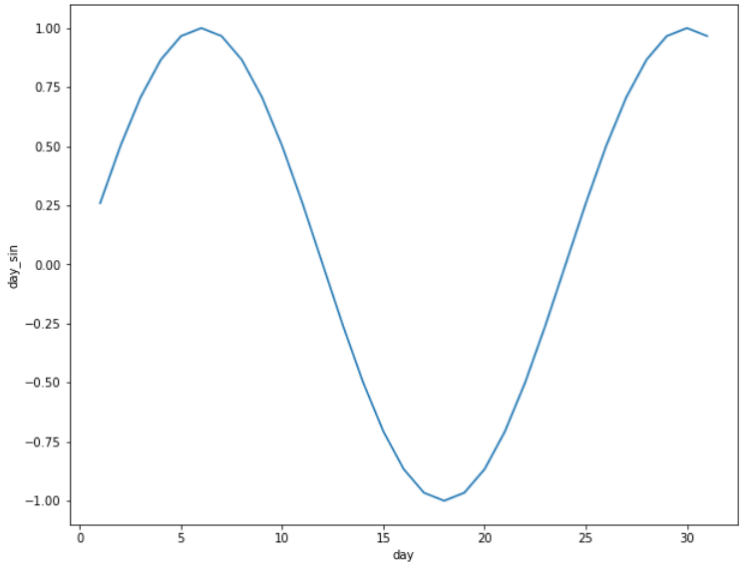
5.1. Conversión de los valores cíclicos a seno y coseno

Existen variables cuyo comportamiento es cíclico, pero sus valores no lo reflejan. Ése es el caso de la hora, el día del mes y el mes del año. Cuando llegan a su valor máximo comienzan de nuevo por el primer valor. Así, la hora pasa de 23 a 0, el día pasa de 28, 29, 30 ó 31 a 1, dependiendo del mes del año, y el mes pasa de 12 a 1.

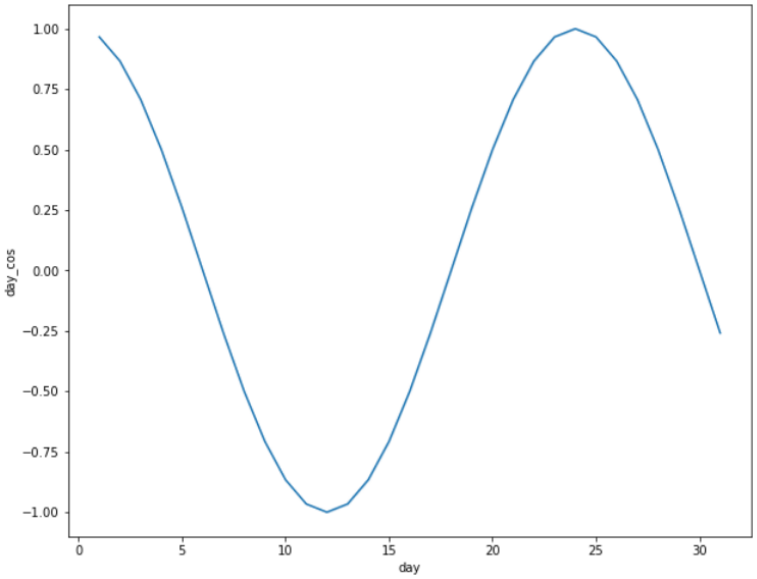
Para poder reflejar este comportamiento cíclico es necesario convertir estos valores a una función cíclica, por eso se ha elegido la función seno.

Además, para compensar la importancia de los valores se va a usar también la función coseno.

Ejemplo de función seno:



Ejemplo de función coseno:



5.2. Variables categóricas

Las variables categóricas son *waterBody*, *locality* y *countryCode*. Debido a su alta cardinalidad se ha elegido el método *mean target encoding* para tratarlas.

Además, este proyecto tiene la dificultad de que se trata de un modelo multi-etiqueta, por lo que hay que calcular la media de cada columna para cada uno de los 5 valores del target.

Para poder utilizarlo en un pipeline, se ha creado una clase que devuelve el valor de *mean target encoding* para cada uno de los valores del target.

5.3. Variables continuas

La única variable continua que falta por transformar es *depth*. Como se prueban algunos modelos paramétricos, se estandariza la variable.

6. Importancia de las variables

Tras haber convertido las variables de entrada, se han realizado diferentes test de hipótesis a las variables categóricas y las variables continuas, para comprobar su influencia en el target.

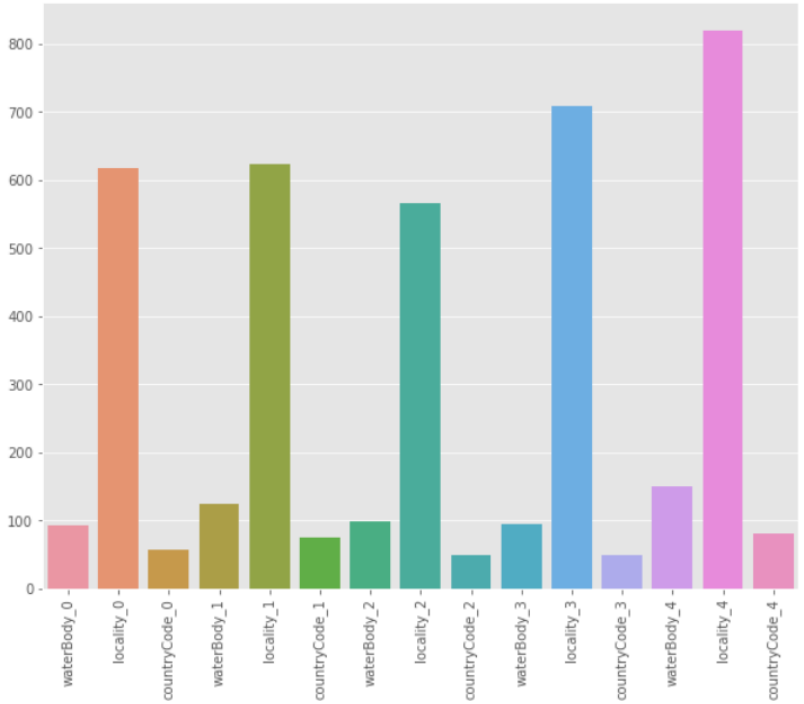
6.1. Variables categóricas

Las variables categóricas son *locality, countryCode* y *waterBody.*

6.1.1. Test Chi cuadrado

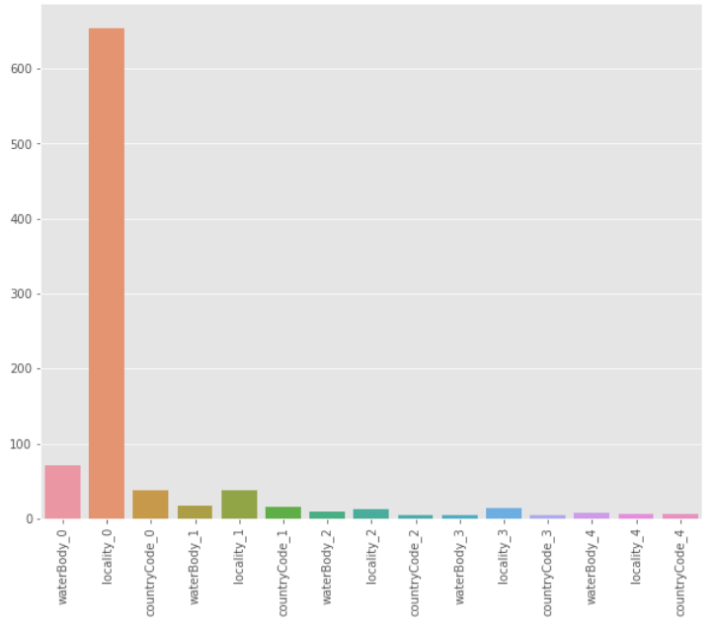
Al hacer el test con los vectores se puede ver que todas las variables son importantes para el modelo, pero especialmente la variable *locality*:

```
Feature waterBody_0 score: 93.605657 p_valor: 2.2555030139248376e-19
Feature locality_0 score: 617.229329 p_valor: 2.891849571398031e-132
Feature countryCode_0 score: 57.630703 p_valor: 9.122037093491549e-12
Feature waterBody_1 score: 123.683397 p_valor: 8.724480551714297e-26
Feature locality_1 score: 622.735603 p_valor: 1.859294462472237e-133
Feature countryCode_1 score: 75.028315 p_valor: 1.9652999349579875e-15
Feature waterBody_2 score: 99.309105 p_valor: 1.380130849286163e-20
Feature locality_2 score: 565.429441 p_valor: 4.692874025144407e-121
Feature countryCode_2 score: 48.652620 p_valor: 6.899055159583764e-10
Feature waterBody_3 score: 94.909925 p_valor: 1.190995135359265e-19
Feature locality_3 score: 709.219840 p_valor: 3.5143286626947474e-152
Feature countryCode_3 score: 48.769377 p_valor: 6.522831291675752e-10
Feature waterBody_4 score: 149.466596 p_valor: 2.648670572397405e-31
Feature locality_4 score: 820.196335 p_valor: 3.2402186948086864e-176
Feature countryCode_4 score: 81.517143 p_valor: 8.308505999291465e-17
```

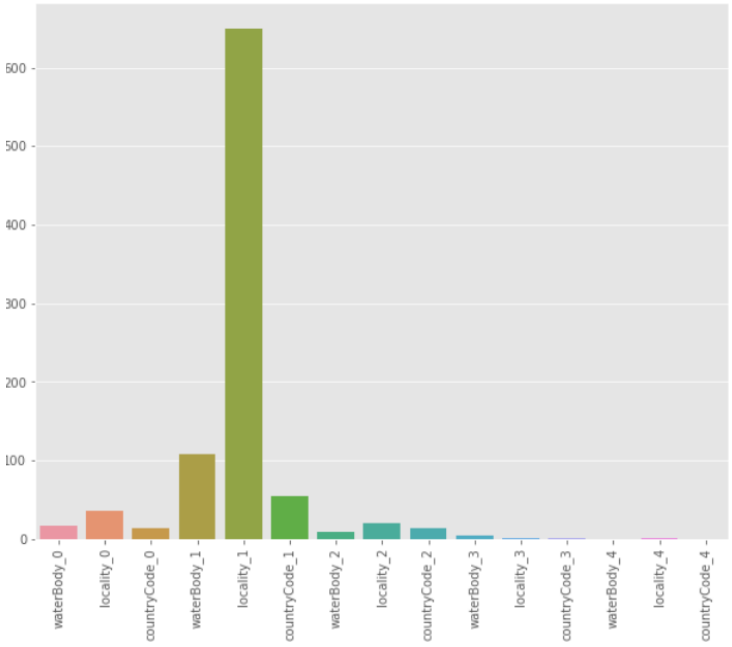


Al realizar el test con cada uno de los animales por separado se ve que el resultado es el mismo (en este caso cada una de las barras que destaca respecto a las otras es porque se trata del mean target encoding de ese animal concreto).

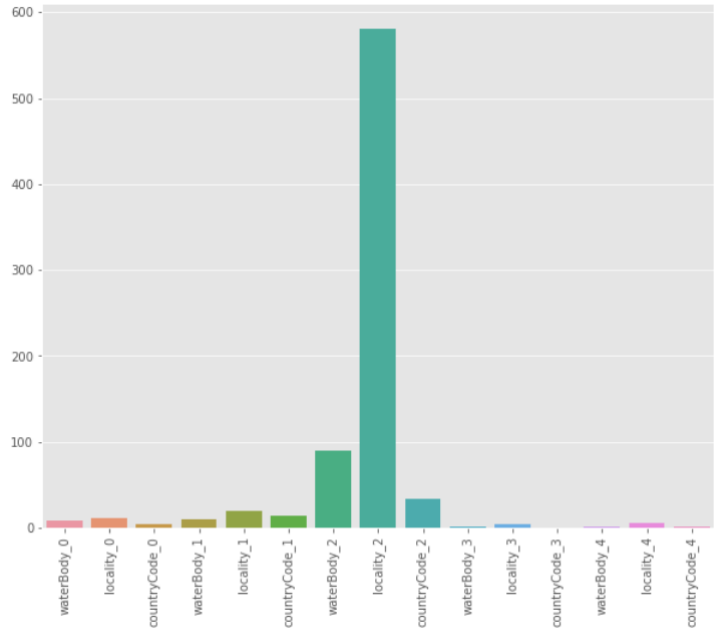
Moray eels:



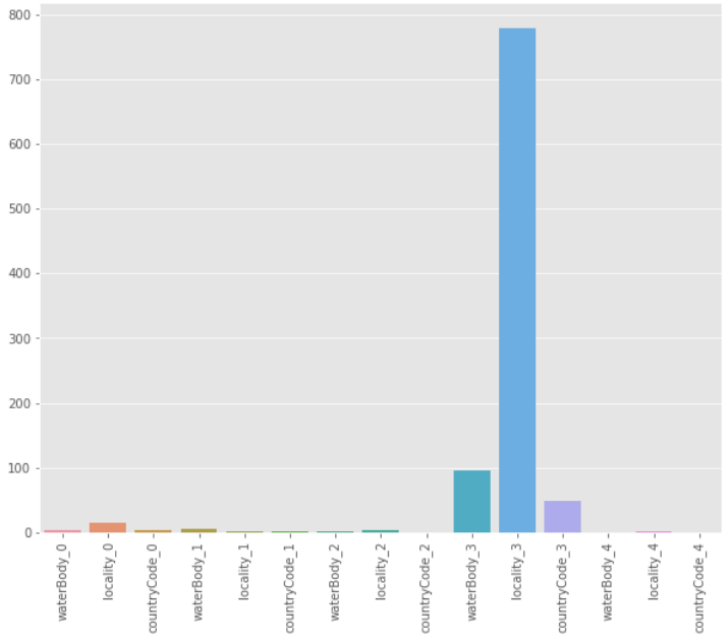
Firefishes:



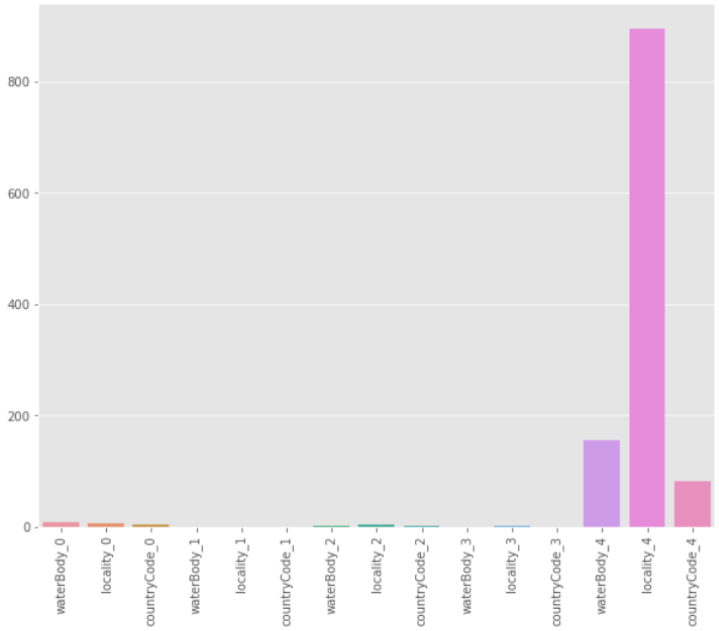
Damselfishes:



Sea turtles:



Groupers:



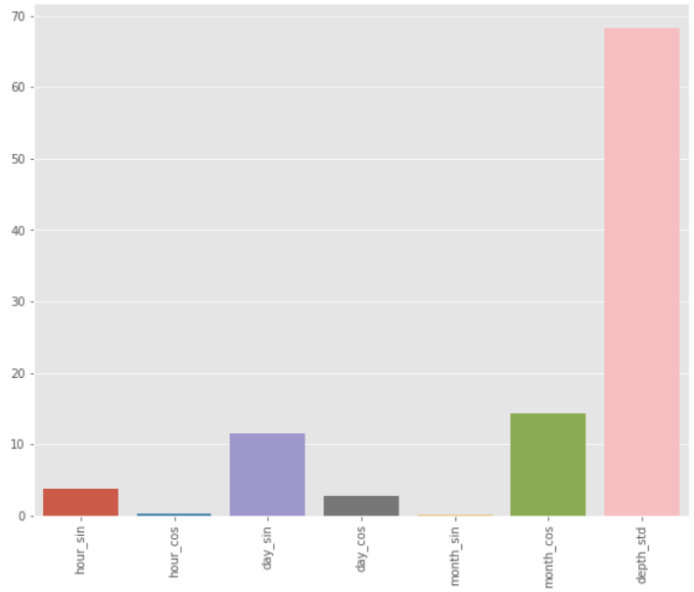
6.2. Variables numéricas

Las variables numéricas son *hour, day, month* y *depth*.

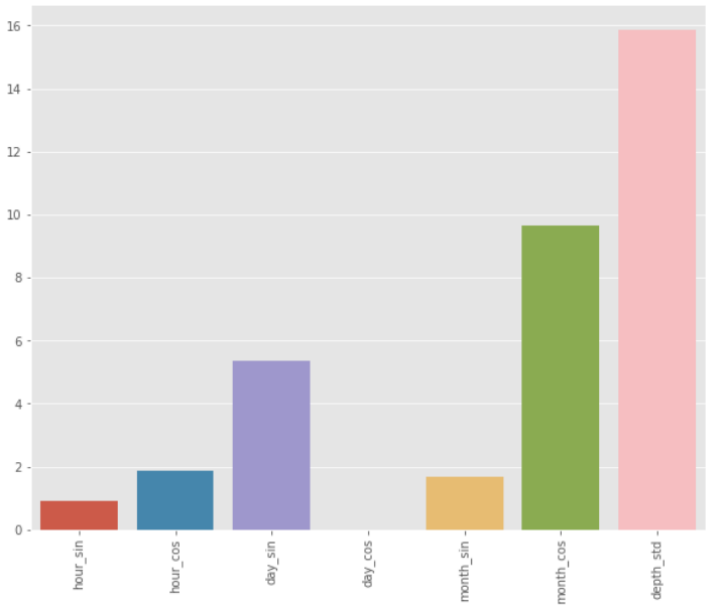
6.2.1. Anova

En este caso, la importancia de las variables es diferente según el animal.

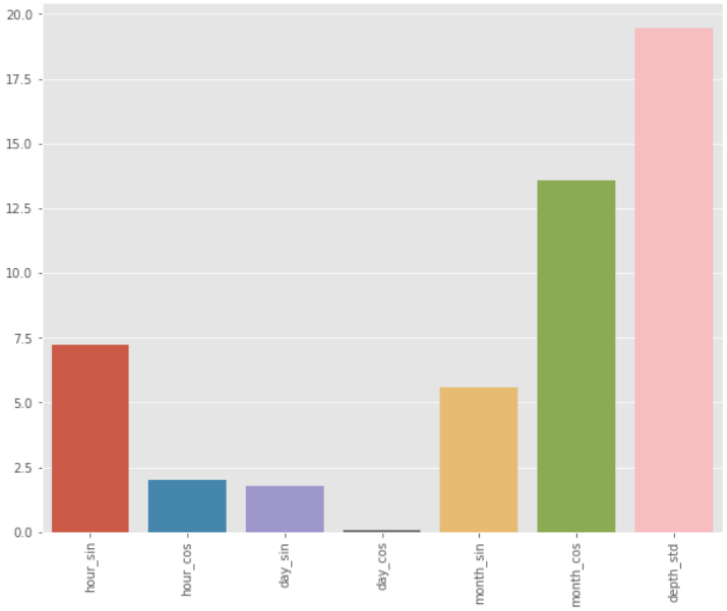
Moray eels: aquí son significativamente importantes el día y la profundidad.



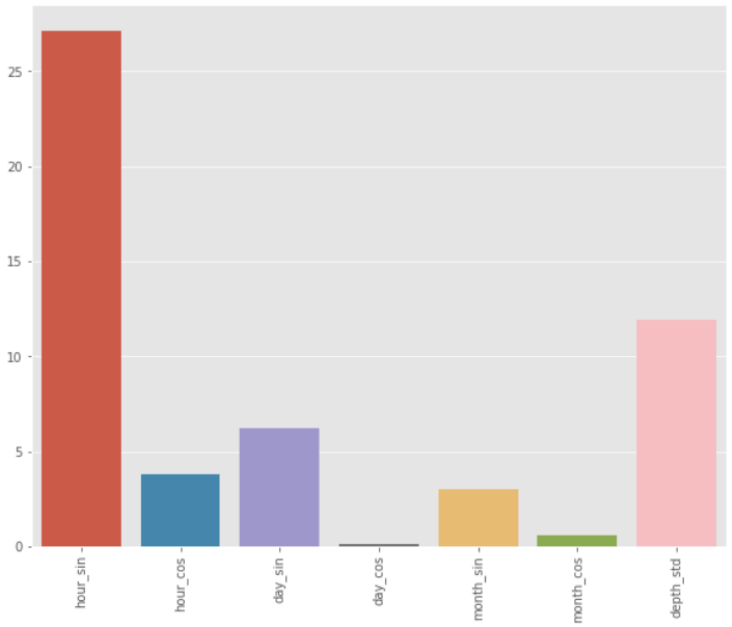
Firefishes: en este caso influyen el día, el mes y la profundidad.



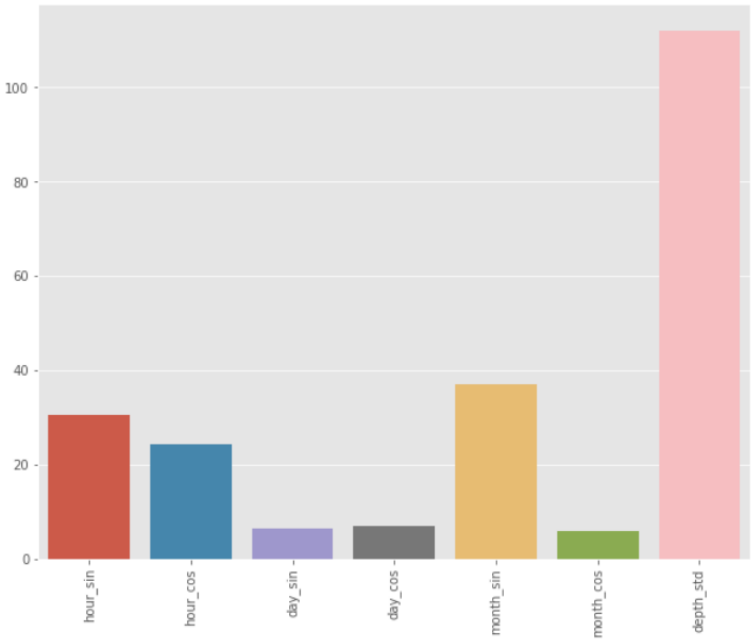
Damselfishes: hora, mes y profundidad.



Sea turtles: hora, día y profundidad.



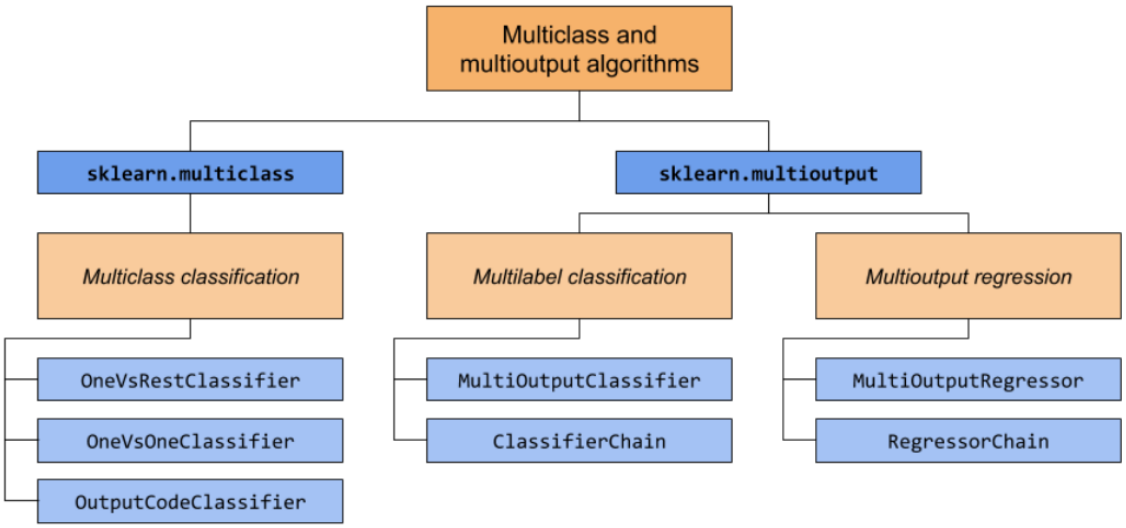
Groupers: todas las variables.



7. Selección de modelo

Debido a la naturaleza del problema que se intenta resolver en este trabajo, el tipo de modelo de predicción tiene que ser un clasificador multi-etiqueta.

Para realizar un clasificador multi-etiqueta, existen en scikit-learn dos aproximaciones, como son utilizar un meta estimador Multi Output Classifier o un Classifier Chain:



Ambas aproximaciones se van a probar en la selección del modelo para cada uno de los estimadores.

7.1. Multi Ouput Classifier

Este meta estimador consiste en crear un estimador para cada una de las clases del target. Tiene el problema de que no es capaz de detectar relaciones entre las distintas clases.

7.2. Classifier Chain

Este meta estimador sí intenta encontrar relaciones entre las distintas clases que componen el target. Para ello, a la hora de entrenar el modelo, crea un primer estimador para la primera clase. Después crea un estimador para la segunda clase añadiendo a las variables de entrada el valor real del clasificador de la primera clase. A continuación hace lo mismo para la tercera clase pero aprovechando los valores reales de los dos primeros clasificadores, y así para todas las clases, de modo que el último clasificador tiene como entrada los resultados de los clasificadores de las clases anteriores.

En el caso de las predicciones, el comportamiento es similar, pero como no se conoce el valor real de los clasificadores anteriores, lo que se utiliza es el valor predicho para cada una de las clases.

La contrapartida de este meta estimador es que es difícil encontrar el orden correcto de las clases del target para encontrar la relación entre ellos, y depende del azar ser capaz de obtener la distribución más adecuada.

7.3. Estimadores

Se van a probar 3 estimadores, que son Logistic Regression, Support Vector Machine y xgboost.

7.3.1. Logistic Regression

Se va a utilizar este estimador paramétrico como referencia. Tiene la ventaja de que es fácil ver la relación entre las variables de entrada y el target.

7.3.2. Support Vector Machine

Este estimador se ha utilizado ampliamente en la predicción de migración de especies, por eso se ha elegido en este trabajo.

7.3.3. XGBoost

No podía faltar uno de los estimadores más populares de los últimos tiempos.

7.4. Métricas

Las métricas elegidas han sido las siguientes.

7.4.1. Hamming Loss

Esta métrica calcula la media de la distancia de Hamming entre los valores reales y los predichos:

$$L_{Hamming}(y, \hat{y}) = \frac{1}{n_{\text{labels}}} \sum_{j=0}^{n_{\text{labels}}-1} 1(\hat{y}_j \neq y_j)$$

El resultado representa el porcentaje de etiquetas erróneas de todas las clases, y su valor oscila entre 0 y 1, significando 1 que se han errado todas las etiquetas y 0 que se han acertado todas.

La ventaja de esta métrica es que tiene en cuenta los aciertos parciales de las predicciones.

7.4.2. Micro Average F1 Score

Calcula el F1 Score como si fuera una clasificación binaria, teniendo en cuenta todos los valores de las matrices de confusión de todas las clases del target, al contrario del macro average, que realiza la media de cada F1 Score de cada clase.

7.4.3. Micro Average Precision

Igual que Micro Average F1 Score, tiene en cuenta todos los valores para hacer el cálculo y no las medias de cada clase.

7.4.4. Micro Average Recall

Lo mismo que en los dos casos anteriores.

7.4.5. Accuracy score

Esta métrica calcula el ratio entre etiquetas correctamente predichas y las etiquetas reales. Es mucho más estricta que Hamming Loss, ya que se penalizan los aciertos parciales.

Para la elección del modelo final se ha tenido en cuenta el valor más bajo de la métrica Hamming loss y, en caso de haber más de 1 con el mismo resultado, el que tenga el valor más alto de Accuracy.

7.5. Selección de hiperparámetros

Para este paso se ha creado una pipeline que contiene todo el feature engineering explicado en el punto 6 (excepto la extracción de hora) incluido en un ColumnTransformer, el meta estimador y el estimador, para hacer la búsqueda de hiperparámetros con un GridSearchCV al que se le pasan todos los valores de los hiperparámetros a probar.

7.6. Selección de modelo final

Para la selección del modelo final, se han establecido las métricas Hamming Loss en primera instancia y, en caso de empate, Accuracy.

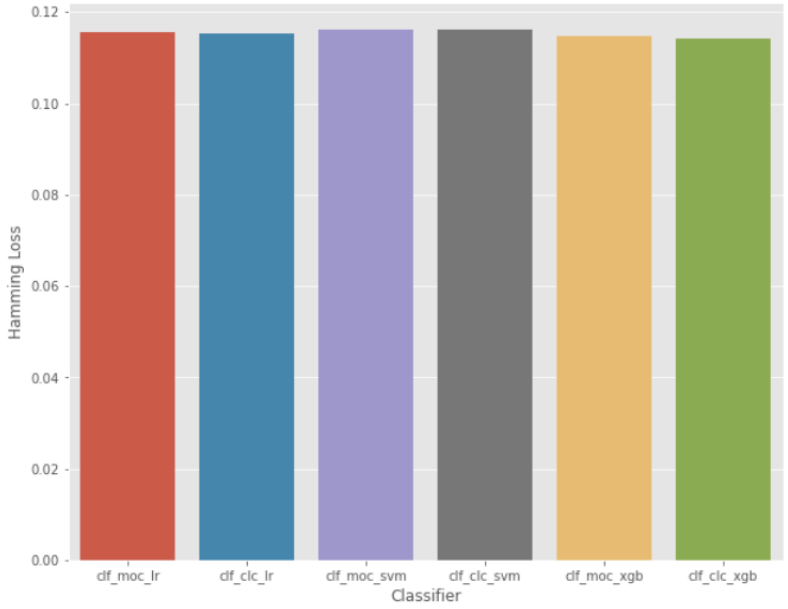
En las gráficas, los modelos han sido nombrados de la siguiente manera:

- clf_moc_lr: Multi Output Classifier Logistic Regression
- clf_clc_lr: Classifier Chain Logistic Regression
- clf_moc_svm: Multi Output Classifier Support Vector Machine
- clf_clc_svm: Classifier Chain Support Vector Machine
- clf_moc_xgb: Multi Output Classifier XGBoost
- clf_clc_xgb: Classifier Chain XGBoost

Los valores de las métricas de los modelos han sido las siguientes.

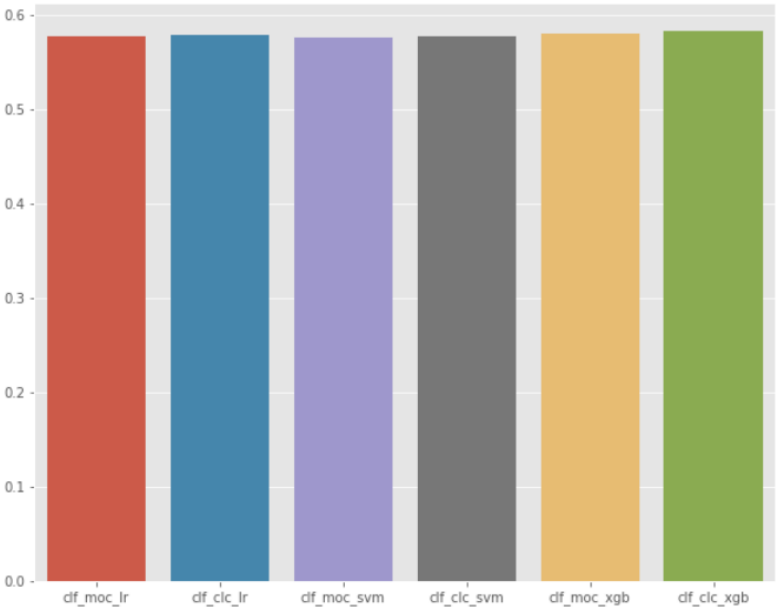
7.6.1. Hamming Loss

Para esta métrica, el mínimo valor es el más adecuado. Como se puede ver el clasificador que mejor métrica tiene es el Chain Classifier XGBoost, aunque por un margen muy pequeño.



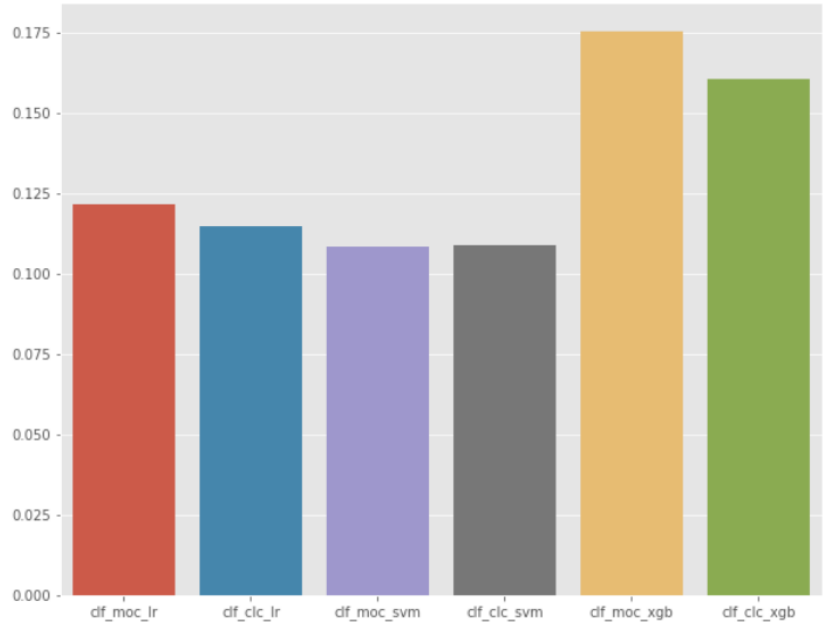
7.6.2. Accuracy

Por un escaso margen, el Classifier Chain XGBoost vuelve a ser el que mejor valor tiene.



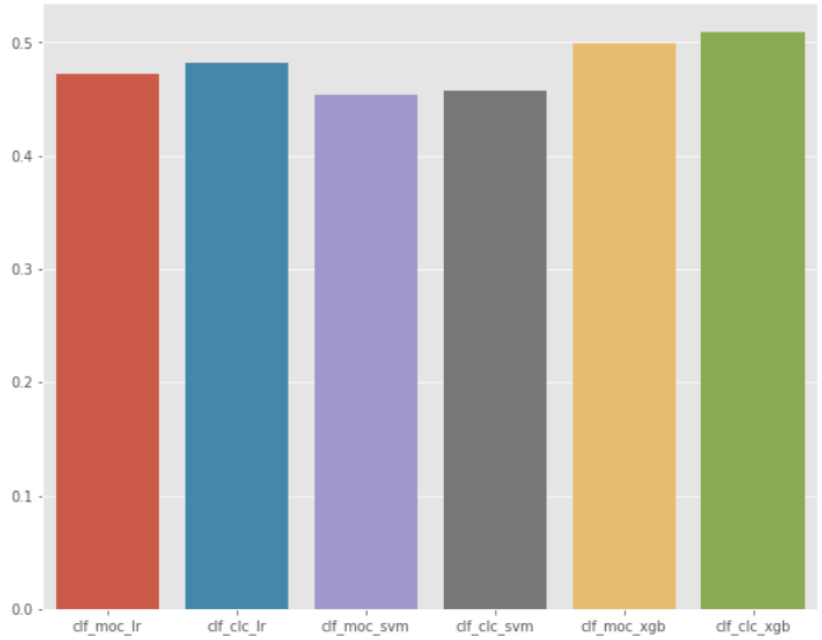
7.6.3. Micro Average F1 Score

Se puede observar que ambos clasificadores XGBoost obtienen un mejor resultado, pero en esta ocasión, el mejor es el Multi Output Classifier XGBoost.



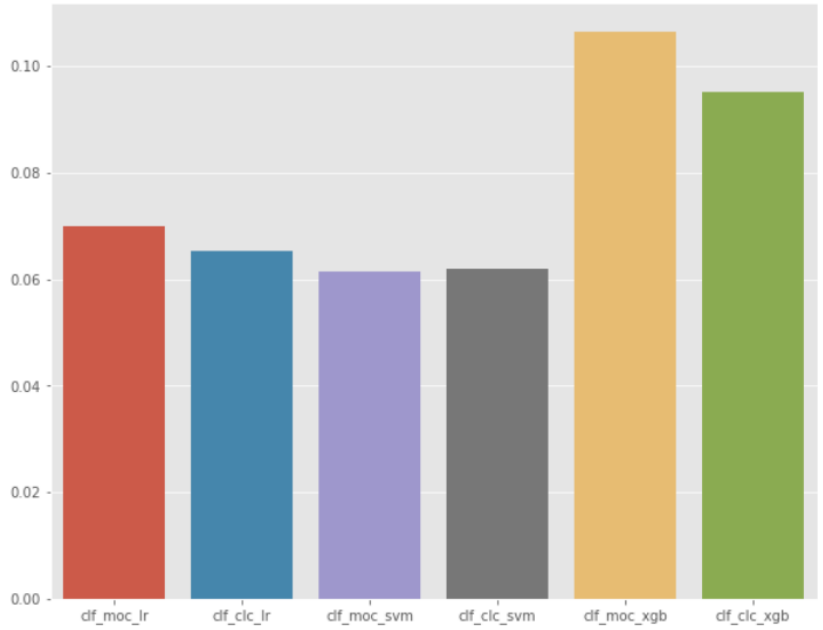
7.6.4. Micro Average Precision

En esta métrica, todos los modelos están más cerca en valores, pero el Classifier Chain XGBoost vuelve a ser el que mejor valor tiene.



7.6.5. Micro Average Recall

En este caso ambos XGBoost vuelven a tener mejores valores que el resto de modelos, siendo el Multi Output Classifier el que mejor valor tiene.



Parámetros y modelo final

Como se ha podido comprobar, las diferencias no son muy grandes, pero los dos clasificadores XGBoost han tenido la mejor puntuación en todas las métricas.

De los dos XGBoost, el que mejor valores ha obtenido en las métricas de Hamming Loss y Accuracy ha sido el Classifier Chain XGBoost, y por eso ha sido el modelo final elegido.

Sus parámetros son:

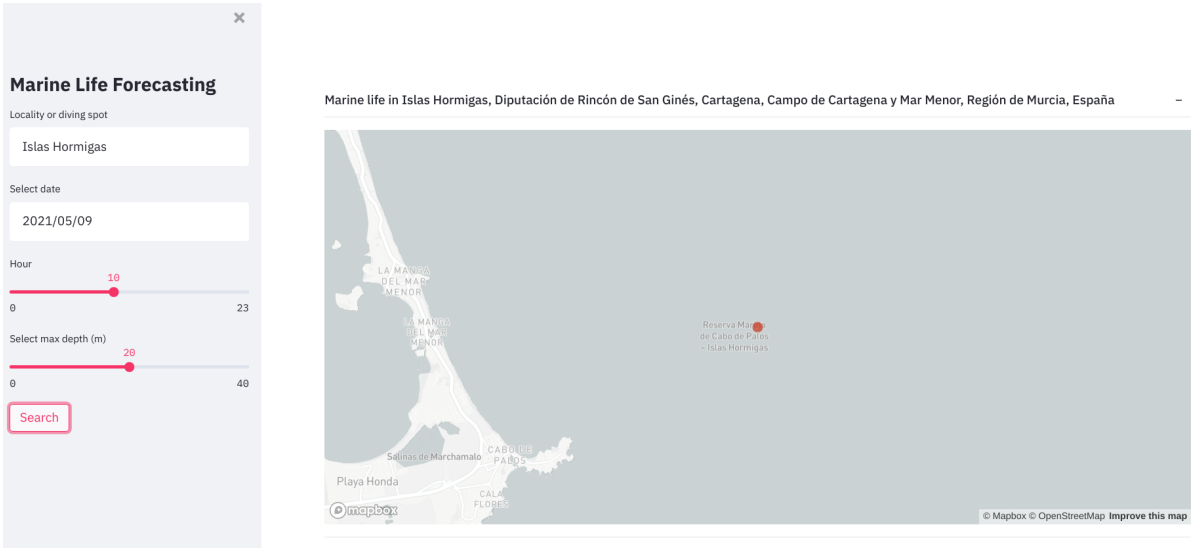
```
{'clc__base_estimator__colsample_bytree': 0.3,
 'clc__base_estimator__eta': 0.05,
 'clc__base_estimator__gamma': 0.2,
 'clc__base_estimator__max_depth': 10,
 'clc__base_estimator__min_child_weight': 1,
 'clc__order': None,
 'clc__random_state': 17}
```

8. Aplicación

Es aquí donde converge todo el trabajo desarrollado.

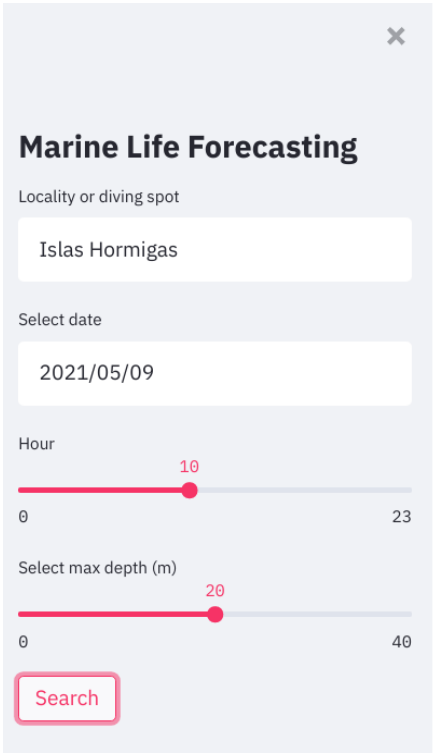
Se ha decidido realizar una aplicación web desarrollada con Python. Para ello se ha elegido el framework Streamlit.

8.1. Interfaz



La interfaz de la aplicación está separada en 2 partes: un panel lateral izquierdo donde introducir los parámetros de búsqueda, y la parte principal, donde aparece el mapa geolocalizando la zona y los datos de los animales predichos.

8.1.1. Panel lateral



El panel lateral está dividido en las siguientes partes:

- Locality or diving spot: campo de texto en el que se introduce la zona de buceo
- Select date: calendario en el que se elige la fecha de la inmersión
- Hour: barra deslizante para elegir la hora planificada

- Select max depth (m): barra deslizante para elegir la profundidad máxima
- Search: botón que, al presionarlo, y con todos los datos rellenos, realiza la búsqueda.

8.1.2. Zona principal de visualización

Sea turtles (39%)

Sea turtles (superfamily Chelonioidea), sometimes called marine turtles, are reptiles of the order Testudines and of the suborder Cryptodira. The seven existing species of sea turtles are the green sea turtle, loggerhead sea turtle, Kemp's ridley sea turtle, olive ridley sea turtle, hawksbill sea turtle, flatback sea turtle, and leatherback sea turtle. All six of the sea turtle species present in U.S waters (Loggerhead, Green sea turtle, Hawksbill, Kemps Ridley, Olive Ridley, and Leatherback) are listed as endangered and/or threatened under the Endangered Species Act. The seventh sea turtle species is the Flatback, which exists in the waters of Australia, Papua New Guinea and Indonesia. Sea turtles can be separated into the categories of hard-shelled (cheloniid) and leathery-shelled (dermochelyid). There is only one dermochelyid species which is the leatherback sea turtle.

En esta parte se muestra el mapa de la zona elegida, la información, las fotos y la probabilidad de ver los animales predicha por el modelo.

8.2. Funcionamiento

Para el correcto funcionamiento de la aplicación hay que rellenar todos los campos de la interfaz y pulsar el botón *Search*.

8.2.1. Mapa

Para el mapa inicial se han establecido unas coordenadas estándar (42.2376602, -8.7247205).

Una vez que se introduce el lugar a buscar en el campo de texto, y se presiona el botón buscar, se lanza una consulta a la API de Open Street Map con el método geocode. Esta consulta devuelve un objeto del que se aprovechan tres de sus atributos: la latitud, la longitud y la dirección.

La latitud y la longitud se utilizan para mostrar la localización en la función map de Streamlit, y la dirección se utiliza para mostrar la dirección en el título.

8.2.2. Predicciones

Para realizar las predicciones se utilizan los datos introducidos en la interfaz, es decir, la localidad, la fecha elegida, la hora y la profundidad máxima.

De la localidad se obtienen la latitud y la longitud de la forma descrita en el apartado de Mapa, y con el KNNImputer se obtiene la localidad, la masa de agua y el código de país más próximos en el dataset original.

De la fecha se obtiene el día del mes y el mes del año.

Con esto se obtienen todas las variables que se pasan al modelo para hacer la predicción:

- Localidad devuelta por el KNNImputer
- Masa de agua devuelta por el KNNImputer
- Código de país devuelto por el KNNImputer
- Día
- Mes
- Hora
- Profundidad (desde los 0 metros hasta la profundidad elegida)

Se calculan todas las probabilidades desde los 0 metros hasta la profundidad máxima de inmersión y, una vez obtenidas todas las probabilidades, se elige la máxima probabilidad de cada animal.

Por último se ordenan las probabilidades de los animales de mayor a menor para mostrarlos luego en ese orden.

8.2.3. Imágenes e información de los animales

En este paso se filtran las probabilidades del paso anterior que tengan un valor de 10% o superior. Con la API de Wikipedia, se obtiene su información con la función summary y la foto con el atributo imagen que devuelve la función page.

9. Conclusiones

Como se puede observar en los resultados de las métricas del modelo, no se trata de un estimador muy preciso.

Esto es debido a que la información de partida está muy sesgada por la idiosincrasia del buceo. Para la gran mayoría de los buceadores se trata de un hobby, lo que produce que la mayoría de ellos realice sus inmersiones en sus vacaciones.

Otra particularidad es que la mayoría de los buceos se suelen hacer de mañana. Esto provoca que, debido al nivel de nitrógeno que se acumula en el cuerpo durante el buceo, no es recomendable realizar demasiadas inmersiones continuadas para permitir al cuerpo eliminar este exceso, lo que hace disminuir el número de inmersiones por la tarde (eso y que, como ya se ha dicho, se suelen hacer en periodo vacacional, así que se suele aprovechar las tardes para otras cosas).

Todo esto hace que los datos estén muy sesgados y no permiten establecer un modelo más ajustado a la realidad.