Repaso W5 – *Machine Learning*

5.1. ¿Qué es un ETL?

ETL son siglas de *Extract*, *Transform* y *Load*, donde *extract*, que significa "extraer" en castellano, nos indica la parte de extracción de datos, de recoger los mayores datos relevantes posibles. *Transform* es transformar los datos en información relevante para poder convertir los datos en material útil para un beneficio. *Load* significa "cargar", la carga de los datos ya extraídos y transformados en un almacén de datos o un sistema de gestión de bases de datos para su posterior análisis. Los almacenamos en un formato que facilita su consulta y análisis, lo que permite a los usuarios realizar informes, análisis y tomar decisiones basadas en datos.

5.2. Enumera las características de *Python* explicando cada una de ellas.

- Es un **lenguaje de alto nivel**. Esto significa que es un lenguaje de programación que se entiende perfectamente, que se utiliza con palabras en inglés legibles que se puede entender sin mucha experiencia en lenguajes de programación.
- Es un lenguaje de **tipado dinámico**. *Python* no es un lenguaje compilado por lo que una de sus fuertes características es su dinamismo.
- Es **multiparadigma**. No es un lenguaje de programación que solamente se utilice para un sólo problema sino que abarca mucho rango en cuanto a funciones, ya sea para el análisis de datos, generar aplicaciones web, etc.
- Contiene dentro una **biblioteca estándar muy extensa** que proporciona módulos y paquetes para realizar una variedad de tareas comunes, desde manipulación de archivos y redes hasta análisis de datos y creación de interfaces gráficas.

5.3. ¿Cuáles son los tipos primitivos en *Python* y qué valores pueden contener cada uno de ellos?

Los tipos primitivos en *Python* se dividen en 3:

- Tipado de **texto**: este tipo en concreto se conoce como *string*, con la función *str()* para declarar una variable como tipo texto. Toda variable *string* tiene que ir entrecomillado, sea simple o doble. Ej.: var = "Hola mundo"
- Tipo ***booleano***: este tipo en concreto tiene dos valores en concreto, *True* o *False*. Podemos declarar una variable *booleana* con *bool(var)* y siempre devolverá uno de los 2 valores aclarados anteriormente.
- Tipo **número**: aquí tenemos 2 tipos, **enteros** y **flotantes**, o lo que es lo mismo, *integer* o *float*, que podemos generarlos con *int(var)* y *float(var)*, donde *int* devuelve un número entero, sin decimales, y *float* devuelve un número decimal flotante, por defecto con un decimal.

5.4. Menciona alguna estructura de datos más compleja que los tipos primitivos que conozcas.

Listas, tuplas, conjuntos y diccionarios.

5.5. Escribe la sintaxis para crear variables.

var =

5.6. Escribe la sintaxis para crear funciones.

```
def function():
    # aquí la función
```

5.7. Escribe la sintaxis para llamar a variables.

```
var
```

5.8. Escribe la sintaxis para llamar a funciones.

```
function()
```

5.9. Explica con tus palabras para qué sirven las librerías: Pandas y Numpy.

- **Pandas**: Es una librería de *Python* en la cuál podemos crear *data frames* y estructuras de bases de datos para una mejor visualización de datos
- **Numpy**: Es la librería de *Python* la cuál se encarga de la creación de *arrays*, por ejemplo. Es una librería de números, de matermáticas.

5.10. ¿Cómo representamos el valor vacío en Python?

Dependerá de la variable que queramos representar como valor vacío.

- En *string*: *var =*
- En *bool*: *var = False*
- En *int*: *var = 0*
- En *float*: *var = 0.0*
- En listas: *vars* = [ ]
- En tuplas: *vars* = ( )
- En conjuntos *vars = set()*
- En diccionarios: *vars* = { }

5.11. ¿Qué es un IDE?

Un *IDE* es un editor de *scipts* de *Python*. Ahí se crea el programa a ejecutar por la terminal. Es también un editor de código, tiene herramientas de depuración, se puede gestionar proyectos, tiene autocompletación de código, tiene una consola interactiva, además de soporte para Bibliotecas y Módulos Externos, etc.

5.12. ¿Qué es el CRISP-DM?

CRISP-DM (Cross-Industry Standard Process for Data Mining) es un proceso estándar utilizado en la minería de datos y la ciencia de datos para guiar proyectos desde la comprensión del problema hasta la implementación de soluciones. Fue desarrollado por un

grupo de expertos en minería de datos y se ha convertido en un marco ampliamente aceptado para la gestión de proyectos de análisis de datos.

El proceso CRISP-DM consta de las siguientes etapas: Comprensión del Negocio (Business Understanding), Comprensión de los Datos (Data Understanding), Preparación de los Datos (Data Preparation), Modelado (Modeling), Evaluación (Evaluation), Despliegue (Deployment), Seguimiento (Monitoring).

5.13. Explica cada uno de los tipos de Machine Learning.

Tenemos cuatro tipos de *Machine Learning*:

- **Supervisado:** En el aprendizaje supervisado el conjunto de datos que se utiliza para entrenar al algoritmo contiene la solución que el algoritmo debería dar a dichos datos. Por eso es supervisado, porque ya se sabe y se entrena al algoritmo con esos datos para que, por prueba y error, mejore los datos, ya sabiéndolos.
- **No supervisado:** En el aprendizaje no supervisado, el conjunto de datos no tiene porque etiquetado, el modelo intenta aprender sin que le digan que tiene que aprender. No se le dan las respuestas a lo que tiene que solucionar.
- **Semisupervisado:** Un agente que es capaz de observar el entorno, realizar acciones y recibir premios o penalizaciones como respuesta a sus actos.
- **Aprendizaje reforzado:** Como su propio nombre indica, mezvla ambos métodos de aprendizaje. Si el modelo falla, es corregido por el humano y el modelo aprende.
- ***Batch Learning***: Se entrena el modelo con un conjunto de datos completo en una sola pasada. Adecuado para datos estáticos.
- ***Online Learning***: Se entrena el modelo incrementalmente a medida que llegan nuevos datos. Adecuado para datos en constante evolución y aplicaciones en tiempo real.

5.14. ¿Con qué tipo de Machine Learning hemos estado trabajando nosotros?

Únicamente con supervisado.

5.15. ¿Qué es el *Prophet*? ¿Y *Scikit Learn*, *Keras* y *Tensorflow*?

- ***Prophet***: es un algoritmo proporcionado por *FaceBook* para predecir valores futuros en un intervalo de tiempo, para predecir series temporales.
- ***Scikit Learn***: es una de las bibliotecas más populares de *machine learning* en *Python*. Proporciona una amplia gama de algoritmos de aprendizaje supervisado y no supervisado, herramientas para la preparación de datos, evaluación de modelos y selección de características. Es ampliamente utilizado en la comunidad de ciencia de datos y *machine learning* para tareas como clasificación, regresión, agrupación y más.
- ***Keras***: es una API de alto nivel para el desarrollo de redes neuronales, que se ejecuta sobre otras bibliotecas de *deep learning* como *TensorFlow*. *Keras* simplifica la creación y el entrenamiento de redes neuronales, lo que la hace especialmente útil para desarrolladores que desean prototipar y experimentar rápidamente con modelos de *deep learning*. *Keras* es compatible con varios *backends*, siendo *TensorFlow* uno de los más comunes.
- ***TensorFlow***: es una biblioteca de código abierto desarrollada por *Google* para realizar cálculos numéricos y construir y entrenar modelos de *machine learning* y *deep learning*. Es ampliamente utilizado en aplicaciones de inteligencia artificial y *machine learning*, incluyendo procesamiento de lenguaje natural, visión por computadora, y muchas otras áreas. *TensorFlow* es conocido por su flexibilidad y escalabilidad, y se utiliza en una variedad de escenarios, desde aplicaciones móviles hasta despliegues en la nube.

5.16. ¿Qué significa en ML regresión? Responde brevemente.

En *Machine Learning*, "regresión" se refiere a un tipo de problema en el que el objetivo es predecir un valor numérico o continuo, como un precio, una temperatura o una puntuación, en función de un conjunto de características o variables de entrada. La regresión implica encontrar una relación matemática que mejor se ajuste a los datos para hacer estas predicciones.

5.17. ¿Cuál es el tipo de predicción más sencilla, pero a la vez la más usada?

La predicción más sencilla y ampliamente utilizada en *Machine Learning* es la regresión lineal simple. En este tipo de predicción, se busca establecer una relación lineal entre una variable de entrada y una variable de salida. A pesar de su simplicidad, la regresión lineal se usa en una variedad de aplicaciones en la vida cotidiana, como la predicción de precios de viviendas, la estimación de ingresos futuros, y muchas otras situaciones en las que se busca predecir un valor numérico en función de una variable independiente.

5.18. ¿Cuáles son los problemas principales del machine Learning? Justifica tu respuesta.

Los problemas principales del Machine Learning incluyen:

- ***Overfitting***: El sobreajuste ocurre cuando un modelo se ajusta demasiado a los datos de entrenamiento y no generaliza bien a nuevos datos. Esto puede llevar a predicciones inexactas en situaciones del mundo real. Ocurre cuando el modelo es demasiado complejo o cuando hay ruido en los datos.

- ***Underfitting***: Es un problema que ocurre cuando un modelo es demasiado simple para capturar la complejidad de los datos subyacentes. En otras palabras, el modelo no se ajusta lo suficiente a los datos de entrenamiento y no puede hacer predicciones precisas.
- **Falta de Datos**: La falta de datos suficientes puede limitar la capacidad de los algoritmos para aprender patrones significativos y reducir la precisión del modelo.
- **Interpretabilidad**: Los modelos a menudo son cajas negras, lo que dificulta la interpretación de sus decisiones, lo que puede ser problemático en aplicaciones críticas.

5.19. ¿Qué es el residuo en Machine Learning?

En *Machine Learning*, un "residuo" es la diferencia entre la predicción de un modelo y el valor real en los datos. Se utiliza para evaluar el rendimiento del modelo.