**Trabajos previos para el modelo de ML**

Antes de realizar cualquier programa se probaron multitud de enfoques y de modelos de ML sobre cuadernos de Jupyter.

-Predicción de las ventas de una tienda

Utilizando el modelo de RRNN se trabajó con los datos de venta de una tienda para comprobar la validez del modelo

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente con confianza media

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Gráfico, Gráfico de líneas

Descripción generada automáticamente

-Cálculos sobre la calidad del aire de Madrid

Los modelos de Series temporales y redes neuronales. Se utilizaron distintas metodologías como VariableLag y escalados y transformación a entrenamientos supervisados

Gráfico, Gráfico de barras, Gráfico de líneas

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Aplicación

Descripción generada automáticamente con confianza baja

Interfaz de usuario gráfica, Gráfico

Descripción generada automáticamente

-Cuadernos de Jupyter para comprobar la viabilidad de las Series Temporales

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Gráfico

Descripción generada automáticamente

Gráfico, Gráfico de líneas

Descripción generada automáticamente

¡¡¡!!! Predicción sin parcial

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Con parcial:

Gráfico, Gráfico de líneas

Descripción generada automáticamente

-SARIMAX

Este modelo se descartó por ser simple y tener poca capacidad de ampliación. Su ejecución también devolvía varios Warning

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente Imagen que contiene Texto

Descripción generada automáticamente

**Aplicación para la captura de datos.**

La empresa Hopu pone a disposición los datos del aforador y de la calidad del aire en una API accesible desde distintas URL.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

El servicio tiene autenticación por tockens.

<https://fiware.hopu.eu/keycloak/auth/realms/fiware-server/protocol/openid-connect/token>

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

El resto de servicios se dividen en calidad del aire, aforamiento e información del dispositivo.

Todos tienen el protocolo GET para recoger los datos y el POST para modificar información de la ubicación de del dispositivo

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

URL datos calidad del aire: [https://fiware.hopu.eu/orion/v2/entities?limit=1000&attrs=\*,dateModified&options=keyValues](https://fiware.hopu.eu/orion/v2/entities?limit=1000&attrs=*,dateModified&options=keyValues)

URL Datos aforamiento:

[https://fiware.hopu.eu/orion/v2/entities?limit=1000&attrs=\*,dateModified&options=count,keyValues](https://fiware.hopu.eu/orion/v2/entities?limit=1000&attrs=*,dateModified&options=count,keyValues)

Para realizar la llamada a esta API, le crea un programa en Java. Este guardará cada 30 segundos un registro en un fichero CSV. de esta manera los datos quedarán guardados para ser procesador por el modelo de ML

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Dentro del paquete calidadAire.rest: la clase RestHopu contiene las llamadas a la API, la clase FicheroCSV se encarga de guardar el fichero mencionado, la clase Registro es utilizada para guardar un log del funcionamiento de la aplicación, la clase App es la encargada de sincronizar todas las funciones de las clases anteriores.

Dentro del paquete gui: se encuentra una pequeña interfaz gráfica FeedBack, realizada con WindowBuilder, utilizada para reiniciar, iniciar y detener la aplicación, muestra un pequeño feedback para conocer el estado del programa y permite forzar el reinicio de la recolección de datos

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

El proyecto Java es un Maven con las dependencias de json y okhttp

<dependency>

<groupId>com.squareup.okhttp</groupId>

<artifactId>okhttp</artifactId>

<version>2.7.5</version>

</dependency>

<dependency>

<groupId>org.json</groupId>

<artifactId>json</artifactId>

<version>20210307</version>

</dependency>

El encabezado del fichero CSV es:

fecha;pm25;pm10;co2;humedad;temperatura;t\_terabee;presencia;

La primera fecha es la fecha que devuelve el REST de calidad del aire, después vienen las partículas PM10, PM2,5, co2, humedad y temperatura. La segunda fecha (t\_terabee) es la fecha que devuelve el dispositivo de presencia, seguido del dato de presencia. Este último se consigue restando el valor OUT al valor IN. Obteniendo la gente que hay dentro

Ejemplo de datos:

16-11-2021 15:13:50;10.0;11.0;527.57635;38.62974;24.140154;16-11-2021 15:13:56;11.0;

16-11-2021 15:14:22;9.0;10.0;527.57635;38.62974;24.140154;16-11-2021 15:13:56;11.0;

16-11-2021 15:15:02;10.0;11.0;527.57635;38.62974;24.140154;16-11-2021 15:14:56;11.0;

16-11-2021 15:15:41;12.0;13.0;527.57635;38.62974;24.140154;16-11-2021 15:14:56;11.0;

En los ficheros logs la aplicación genera uno nuevo por día y hay dos tipos, el de funcionamiento habitual, “logRestHopu-fecha” y “logCritico\_fecha” . El primero solo indica el buen funcionamiento de la aplicación, el segundo solo se genera si se ha producido un error que detenga la captura de datos.

**Programa Python para la ejecución del modelo de ML**

Se realiza un programa .py para la ejecución de los modelos desde un menú y para evitar el reentrenamiento continuo de los modelos. Este se ejecuta en línea de comandos

Texto

Descripción generada automáticamente

Hacemos también distintas clases. La case FicheroCSV se encarga de la lectura del fichero y la carga como DataFrame de panda, la clase Menu contiene la interacción por línea de comandos y el funcionamiento de las distintas opciones, la case Modelos, tiene todos los distintos modelos de ML disponibles. El archivo CONSTANTES, guarda la configuración para el programa y el archivo main es el encargado de levantar la aplicación.

Todos los requisitos de librerías se guardan en el fichero requeriments.txt, para poder resolverlas con el comando : pip install -r requeriments.txt

Las librerías necesarias son:

Texto

Descripción generada automáticamente

También se necesita la compilación 3.9.6 de Python

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Los distintos modelos utilizados son: Forecast con una sola variable, Forecast con variable exógena, ambos con RandomForest como regresión y Forecast MultiOutput con exógena y regresión Lasso

Menús de la aplicación: Texto

Descripción generada automáticamente

El primer menú de la aplicación es donde elegimos el modelo con el que trabajar, dependiendo de si ya esté entrenado o no, pondrá READY / NOT READY.

La cuarta opción ejecuta todos los modelos de seguido con una interacción inicial del usuario.

La quinta opción define los pasos que el modelo utiliza en la predicción, cada paso equivale a 5 minutos

La séptima opción guarda un gráfico con todos los datos que contenga el DataSet

La octava opción instala la librerías que necesita la aplicación para funcionar, pensado para primera ejecuciones en la máquina

La novena cierra el programa

El segundo menú de la aplicación es el que nos permite interactuar con el modelo en específico, el menú es el mismo para los distintos modelos

Texto

Descripción generada automáticamente

La primera opción realiza la primera carga del modelo en memoria, el primer entrenamiento y la búsqueda de los mejores hiperparámetros

La segunda opción hace un reentrenamiento del modelo sin búsqueda de hiperparámetros

La tercera opción ejecutará la predicción

La cuarta opción recarga el fichero de datos CSV

La quinta opción ejecuta la predicción con los datos de Train/test para conocer el error de nuestro modelo, esto generará un gráfico en la carpeta Output

La sexta opción guarda un gráfico de la predicción

Con la novena opción volveremos al menú anterior

Estás ejecuciones tienes que efectuarse en un orden concreto, no podremos hacer la predicción sin ejecutar el paso 1 primero.

**Cada vez que nos movamos a menús de modelos distintos deberemos ejecutar la opción 4, recarga de datos**. Esto se debe a que el programa con el objetivo de ahorrar memoria guarda los datos en la misma ubicación en la memoria, por lo que los distintos planteamientos de los datos pueden provocar error en modelos distintos

**Análisis de datos y distintos modelos**

Datos recopilados en CTCON durante la configuración del dispositivo

La línea azul es el CO2, podemos ver como se registra perfectamente las caídas y subidas de CO2 conforme la oficina va llenándose y vaciándose.

El fin de semana apenas tiene cambio en los niveles de CO2

Gráfico

Descripción generada automáticamente

Predicción VS datos de Test

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Gráfico

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Instalación en el cliente.

Estos datos del Grafana son los datos de urdecon

Imagen que contiene Gráfico

Descripción generada automáticamente

Predicciones de los distintos modelos con datos de ventana, la ventana le da un punto de partida al modelo para la predicción. Como se puede observar los datos son más acertados ya que hace la predicción a partir de la subida de CO2

Sin ventana

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Con ventana

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Comparación entre modelos

Este primer modelo solo tiene los datos de la ventana y del CO2, como se puede observar predice una linealidad en el CO2

Gráfico, Gráfico de líneas

Descripción generada automáticamente

En el caso del modelo MultiOutput con Lasso, en el que añadimos también las personas que habrá durante la predicción (se marca con 45 personas), la curva es ascendente.Gráfico, Gráfico de líneas

Descripción generada automáticamente

Utilizando también la presencia pero con el modelo de RandomForest, la predicción vuelve a ser lineal pero nos dibuja la predicción a continuación de la última lectura

Gráfico, Gráfico de líneas

Descripción generada automáticamente