Instituto Tecnológico de Costa Rica

Programa de Ciencia de los Datos - Módulo Big Data

Proyecto Final

- Esteban Sáenz Villalobos (**88548992,** esaenz7@gmail.com)

- Entrega: 6 de septiembre 2021, 23:00.
- Observaciones: Ejecutar el programa siguiendo las instrucciones detalladas a continuación.

Instrucciones

1. Para cargar el contenedor con todos los recursos necesarios, ejecute los archivos:

   a) **clean_docker.sh**. Este script borrará contenedores e imágenes antiguos correspondientes a este proyecto. **Atención: el comando realiza una acción de "prune" para la limpieza.**

   b) **build_image.sh**. Construye una imagen a partir del archivo DockerFile.

   c) **run_image.sh**. Este script creará 2 contenedores y una red local en docker de la siguiente forma:

   ```
    * Red: bigdatanet, IP: 10.7.84.0/24.
    * Host principal (sesión bash donde se ejecutarán los
   comandos): bigdata_proyecto_esv_1, IP: 10.7.84.101.
    * Host secundario (base de datos): bigdata_proyecto_esv_2
   (postgres),  IP: 10.7.84.102.
   ```

   d) Estos parámetros corresponden a la instancia de postgres dentro del ambiente de docker:

   ```
    * Host: 10.7.84.102
    * Puerto: 5432
    * Usuario: postgres
    * Clave: testPassword
   ```

2. Programa principal:

   a) Ejecute el archivo:

   ```
   #run_main.sh
   ```

   Este comando ejecutará las siguientes instrucciones:

   1. Un query psql (*create_tables.sql*) para crear las tablas necesarias en la base de datos postgres y un query psql (*read_tables.sql*) para leer las tablas creadas a manera de confirmación.

2. Seguidamente ejecutará el programa principal (*main.py*) el cual realizará las siguientes tareas:

   a) Crear los conjuntos de datos a partir de los archivos fuentes (*/datasources/*.csv*)

   b) Preprocesar los conjuntos de datos individuales.

   c) Unir los conjuntos de datos en un dataframe principal.

   d) Aplicar ingeniería de características al conjunto de datos principal para el proceso de ml ("machine learning").

   e) Realizar la operaciones de escritura en base de datos.

3. Finalmente se ejecutará el módulo de **pytest** el cual por medio de las librerías **conftest.py** y **test_app.py** realizará la evaluación de los distintos módulos, funciones y datos de la aplicación a través de una batería de pruebas de 4 etapas.

b) Ejecute el archivo:

```
#run_jupyter.sh
```

Este comando cargará el servidor de Jupyter para accesar al directorio en donde se encuentra el notebook con la segunda parte del proyecto (carga desde BD, entrenamiento, pruebas, evaluación y análisis de resultados).

El notebook "BIGDATA_07_2021_ProyectoFinal_ESV.ipynb" almacena los recursos, el código, la documentación y los resultados solicitados. Se ejecuta de forma completa y secuencial. Adicional se incluyen copias en formatos HTML y PDF.

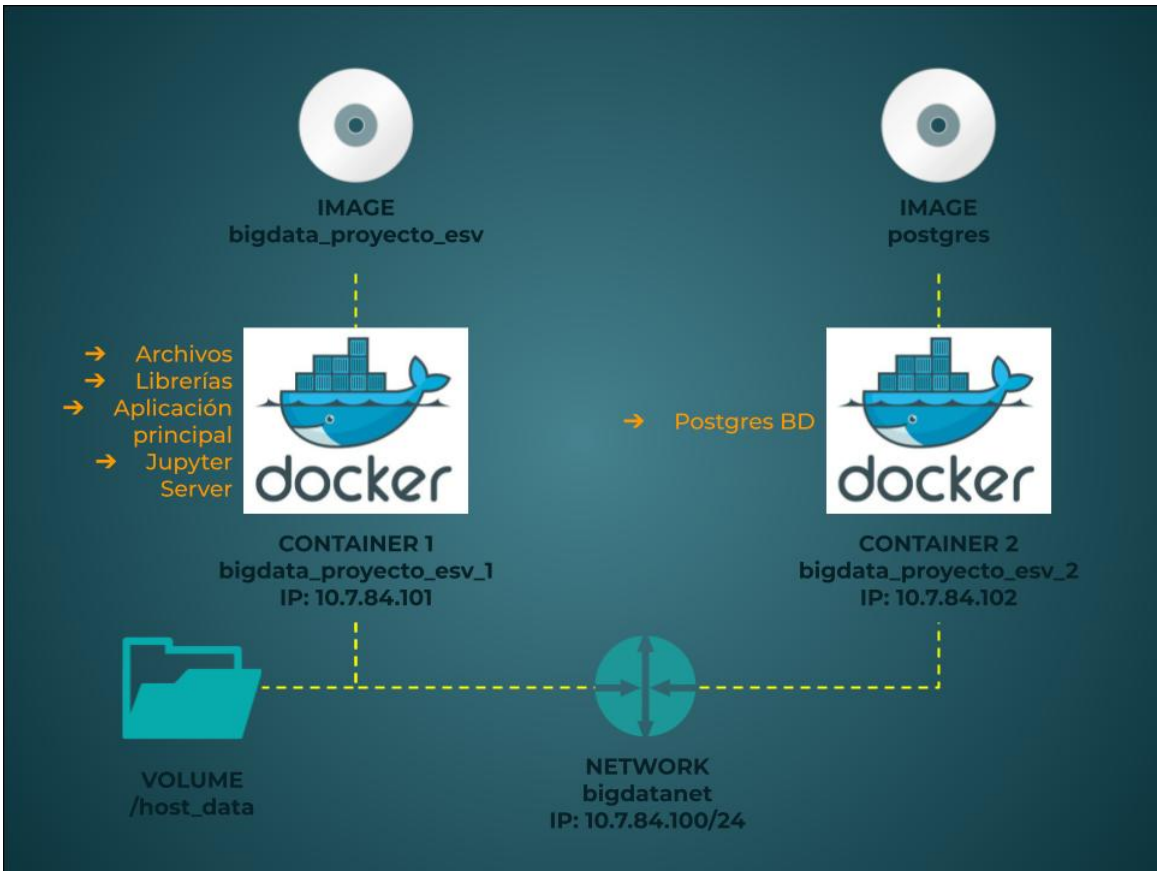
b) Luego de cerrar la sesión del servidor de Jupyter se puede ejecutar el siguiente comando desde la misma sesión BASH del contenedor principal.

```
#run_read.sh
```

Este comando ejecutará un query psql (*read_tables.sql*) para ralizar una lectura de los datos contenidos en las tablas creadas.

c) Una vez haya terminado, puede ejecutar el comando **clean_docker.sh** en el host de docker para limpiar los recursos creados para este trabajo.

- Tanto el programa ejecutado en consola, como el cuaderno de jupyter hacen uso de una librería común llamada **recursos.py**, la cual contiene los módulos, parámetros y funciones globales para las diferentes ejecuciones a lo largo de cada etapa del proyecto.
- El repositorio completo de la tarea se encuentra también en el siguiente enlace github/esaenz7.
- Arquitectura en Docker:



- Base de datos Postgres:

```
                    List of relations
 Schema |     Name      | Type  |  Owner   |  Size   | Description
--------+---------------+-------+----------+---------+-------------
 public | tb_airports   | table | postgres | 104 kB  |
 public | tb_flights    | table | postgres | 5328 kB |
 public | tb_modelolr   | table | postgres | 0 bytes |
 public | tb_modelorf   | table | postgres | 0 bytes |
 public | tb_proyecto   | table | postgres | 7696 kB |
 public | tb_proyectoml | table | postgres | 34 MB   |
 public | tb_weather    | table | postgres | 33 MB   |
(7 rows)
```

  - Las tablas tb_flights, tb_airports y tb_weather corresponden a los conjuntos de datos individuales. La tabla tb_proyecto continene los 3 conjuntos de datos ensamblados previo al pre-procesamiento.
  - La tabla tb_proyectoml corresponden al conjunto de datos preparado después del proceso de ingeniería de características.
  - Las tablas tb_modelolr y tb_modelorf contienen las etiquetas, predicciones y probabilidades, resultado de la evaluación de cada modelo con el conjunto de prueba.
