

Sofía Challenge

Francisco Sanhueza Matamala, para postular al cargo de Data Engineer

21/05/2024



Todo el código está subido en el repo

<https://github.com/Pan-con-queso/sofia-challenge>



Contexto

- Tenemos 25000 datos de usuarios que están siguiendo el protocolo de ejercicios y nutrición Blueprint de Brian Johnson y queremos hacer la regresión de que valores son mas influyentes en el $vo2_max$
- El $vo2_max$ es un buen predictor de longevidad. Indica el volumen de oxigeno consumido al hacer un minuto ejercicio intenso. Se mide en $ml/min/kg$ (Mililitros por minuto dividido en la masa corporal)



Brian Johnson,
creador del programa
Blueprint

El Dataset

- Datos de suplementos y cantidad de ejercicio hecho (98 columnas)
- Edad, Género y Nombre



Consideraciones Generales

- Todo se programó en Python
- Se ejecutan todos los códigos en un computador Lenovo Thinkpad T480 año 2018 con Windows 10.
- Se optan por soluciones sencillas
- Los códigos están documentados y haremos referencia a ellos a lo largo de la presentación.



Estructura de la solución

1. Data Engineering
2. Predictive Modelling
3. Data Visualization
4. Bonus

1. Data Engineering

Partes de la solución

1

Descarga de Datos

Usar API de Google Drive para poder
descargar

2

Descomprimir

Uso de librería ZipFile

3

Creación de archivo csv

Uso de pathlib, jsonpath y tqdm

4

Creación de BBDD

Se crea en SQLite

Descarga de Datos

- Tenemos que habilitar la API de Google Drive
- Autorizar consentimiento de credenciales OAuth 2.0 para aplicación en Google.
- Agregar correo personal como usuario test de la aplicación que va a usar la API de google drive
- Crear cliente OAuth 2.0
- Descargar archivo de credenciales
- Modificar quickstart en python

<https://github.com/googleworkspace/python-samples/blob/main/drive/quickstart/quickstart.py>

gdrive_downloader.py

Descomprimir datos

- Se usa la librería zipfile
- El código es una aplicación directa de la librería
- Demora 9 minutos

zipfile — Work with ZIP archives ¶

Source code: [Lib/zipfile/](#)

The ZIP file format is a common archive and compression standard. This module provides tools to create, read, write, append, and list a ZIP file. Any advanced use of this module will require an understanding of the format, as defined in [PKZIP Application Note](#).

This module does not currently handle multi-disk ZIP files. It can handle ZIP files that use the ZIP64 extensions (that is ZIP files that are more than 4 GiB in size). It supports decryption of encrypted files in ZIP archives, but it currently cannot create an encrypted file. Decryption is extremely slow as it is implemented in native Python rather than C.

The module defines the following items:

zip_extractor.py

Escritura en csv

La estructura del zip descomprimido es la siguiente:

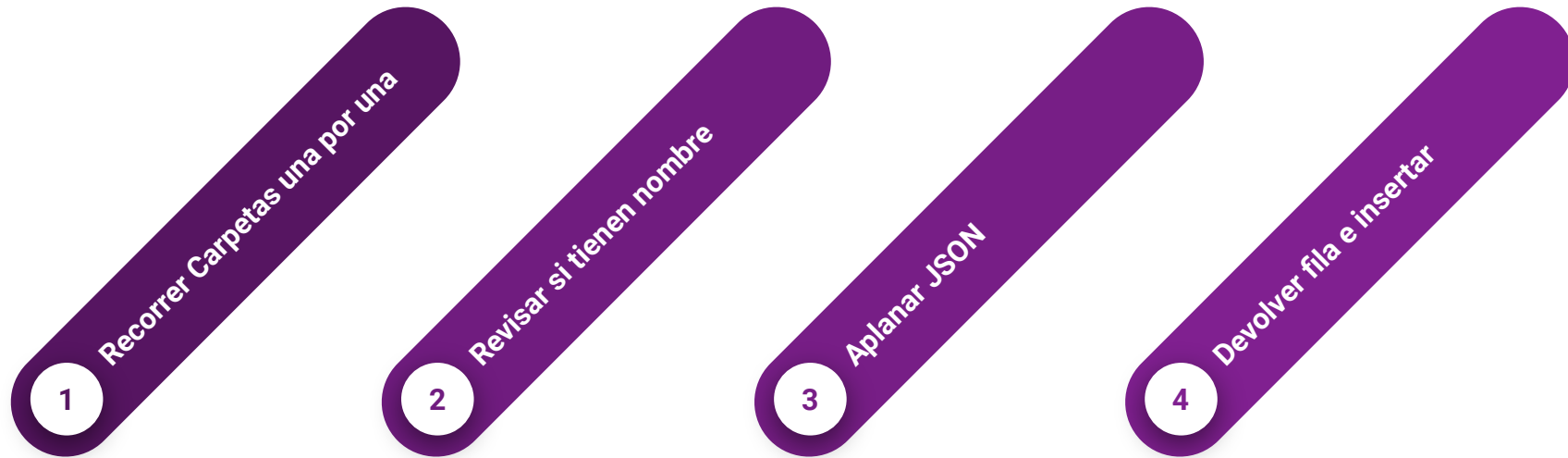
- uuid de usuario 1
 - archivo json con el nombre del uuid
- uuid del usuario 2
 - archivo json con el nombre del uuid
- etc

0fd36158-45c4-44f4-91bf-134893ffbd05	16-05-2024 20:19	Carpeta de archivos
0fd82005-a234-42f9-b244-24ee61aaadf6	16-05-2024 20:19	Carpeta de archivos
0fda0a92-6dbe-4749-87e1-10cbc10847d9	16-05-2024 20:19	Carpeta de archivos
0fdaaaab-7c03-4faa-a922-ff3d2dc0bc6f	15-05-2024 21:49	Carpeta de archivos
0fdd3894-52d7-451f-8b91-3661f81ca1a7	16-05-2024 20:19	Carpeta de archivos
0fdd6525-9658-47f6-b015-bf1e4ca33a14	16-05-2024 20:19	Carpeta de archivos
0fdecece-d9f8-4741-a633-54ad6c7c63d1	16-05-2024 20:19	Carpeta de archivos
0fe8f295-d437-4253-962b-4b05ac00b239	16-05-2024 20:19	Carpeta de archivos
0fe57f7b-a9d2-41ac-99fd-fc74cbdd229c	16-05-2024 20:19	Carpeta de archivos
0fe18362-eb93-4adf-8320-361e4ee9084c	16-05-2024 20:19	Carpeta de archivos
0fe63393-87e1-446e-8d86-a87e14bfd9e5	16-05-2024 20:19	Carpeta de archivos
0fec9803-cc64-4acd-ab41-efc6da1ecedb	15-05-2024 21:46	Carpeta de archivos
0fed22bc-de9c-4e1e-a30a-4a4897ef3b70	16-05-2024 20:19	Carpeta de archivos
0fef92bc-1b9d-46f3-a736-33100354c8fd	16-05-2024 20:19	Carpeta de archivos
0ff0bd53-d843-476f-8b96-9a67cfa45697	15-05-2024 21:46	Carpeta de archivos
0ff6d6f1-2d6e-42ce-afbe-fd3db95ca456	15-05-2024 21:46	Carpeta de archivos
0ff07ef5-fbe2-440c-b305-1ddf19aef8e4	16-05-2024 20:19	Carpeta de archivos
0ff013d2-eb07-4eea-9ab7-bcb0626bd600	16-05-2024 20:19	Carpeta de archivos
0ff86a69-c993-4d87-97df-372876fe6a91	16-05-2024 20:19	Carpeta de archivos
0ffcb763-245c-4ab8-91cb-8e796ac18a51	16-05-2024 20:19	Carpeta de archivos
0ffd23ce-671b-49bd-b780-9288cf4759e7	16-05-2024 20:19	Carpeta de archivos

```
Users > Lenovo > Documents > Python > Sofia_Challenge > Scripts > vo2_max_data > 0fe57f7b-a9d2-41ac-99fd-fc74cbdd229c > {}  
{  
  "personal_data": [  
    {  
      "age": 27,  
      "gender": "F",  
      "name": "Mrs. Lelah Kunde DDS"  
    }  
  ],  
  "fitness_data": [  
    {  
      "Backwards Sled, 2 min": 4,  
      "Posture exercises": 0,  
      "Tricep extensions": 0,  
      "Face pulls": 3,  
      "Butterfly": 1,  
      "Band pull apart (back muscles)": 0,  
      "Back extensions (on a hyperextension)": 2,  
      "Obliques (each side, on a hyperextension)": 3,  
      "StretchesKneeling shin_Kneeling shin_": 1,  
      "StretchesHip flexor_Hip flexor_": 1,  
      "StretchesCouch_Couch_": 1,  
      "Leg raises (for abdomen)": 4,  
      "Seated calf raises": 3,  
      "Poliquin step ups": 1,  
      "Slant board squats": 3,  
      "ATG Split squats": 3,  
      "Nordics": 3,  
      "Reverse Nordics": 2,  
      "Tibialis raises,"": 4,  
    }  
  ]  
}
```

Y los JSON se veían así, como podemos ver, son diccionarios anidados

Partes de la solución



build_dataset.py

Inserción en SQLite

- Se usa cliente de SQLite para python .
- Se procesa la misma fila que nos entrega el recorrido por carpetas, pero se cambian todos los caracteres especiales por guiones bajos y se cambian los nombres de las columnas que empiezan con número.
- Se crea tabla solo si no existe
 - `f"CREATE TABLE IF NOT EXISTS vo2_max({row_string})"` Donde Row_string es la fila procesada como un string.
- Para insertar datos `"INSERT INTO vo2_max VALUES ({row_string})"`

build_dataset.py

Últimas consideraciones

- Cada vez que se revisa un archivo, se crea una copia vacía con extensión .DONE en caso de que tengamos que repetir la inserción desde algún punto
- Demoró 28 minutos la creación de todo el dataset en csv, pero se insertan exactamente los 25000 datos.
- Existen datos con nombres repetidos.

Mejoras

- Orquestar los 3 scripts ocupando herramientas como Airflow o un script que llame a los 3 scripts secuencialmente
- Definir esquema de SQLite de forma explícita, definir llaves primarias y hacer chequeos sobre los datos.
- Paralelizar iteración por las carpetas o probar técnicas de batch inserting.
- Arreglar inserción de archivo csv pues si se crea desde cero cada vez que se corre el script.

2. Predictive Modelling

- Todo el código de esta parte está en el jupyter notebook `predictive_modelling.ipynb`

Explorar datos

- Se exploran datos usando `df.explore` y `df.types`
- No hay datos faltantes ni columnas con tipos de datos inconsistentes, es un muy buen dataset!

Preprocesamiento y columnas nuevas

- Se hace one-hot-enconding para la columna de género
- Se elimina la columna de nombre, no tiene valor predictivo
- Se agrega columna is_elderly para personas mayores de 60 años

Modelos entrenados

- Se prueba una regresión lineal y un árbol de decisión.
- No se escalan los datos para la regresión lineal debido a que están en escalas distintas.
- Se prueba un XGBoost Regressor pero no se prueba más porque la ejecución toma varios minutos, y se privilegian modelos rápidos de revisar
- Se prueban las regresiones con el dataset completo, puesto que el interés **para esta parte de la presentación** es ver que variables son más importantes.

Subconjuntos de datos entrenados

- Solo Hombres
- Sólo Mujeres
- Solo mayores de 60 años

Se prueba con ambos modelos

Resultados generales

- Tanto para la regresión lineal como para el árbol de decisión, las variables mas importantes corresponden a algunos suplementos y la edad
- Para el árbol de decisión la edad es muy determinante en el vo2_max
- Para mayores de 60 años la edad empieza a ser ligeramente menos importante y aumentan más las influencias de ciertos suplementos alimenticios.
- No es notoria la diferencia entre la importancia de los atributos para hombres y mujeres
- En la regresión lineal no aparece la edad como variable más importante, pero puede ser por la escala de los datos.

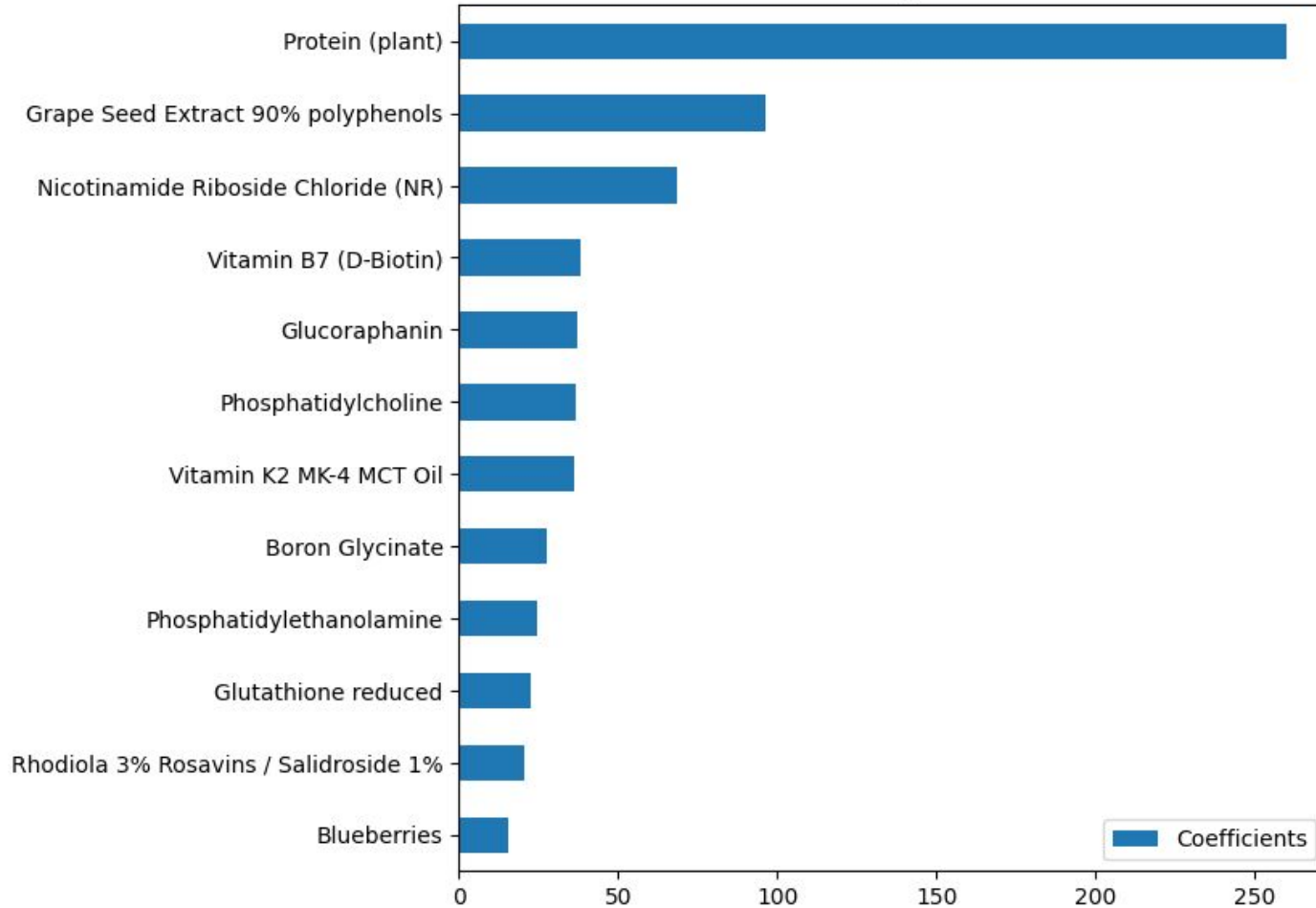
3. Data visualization

- Todo el código de esta parte está en el jupyter notebook `predictive_modelling.ipynb`

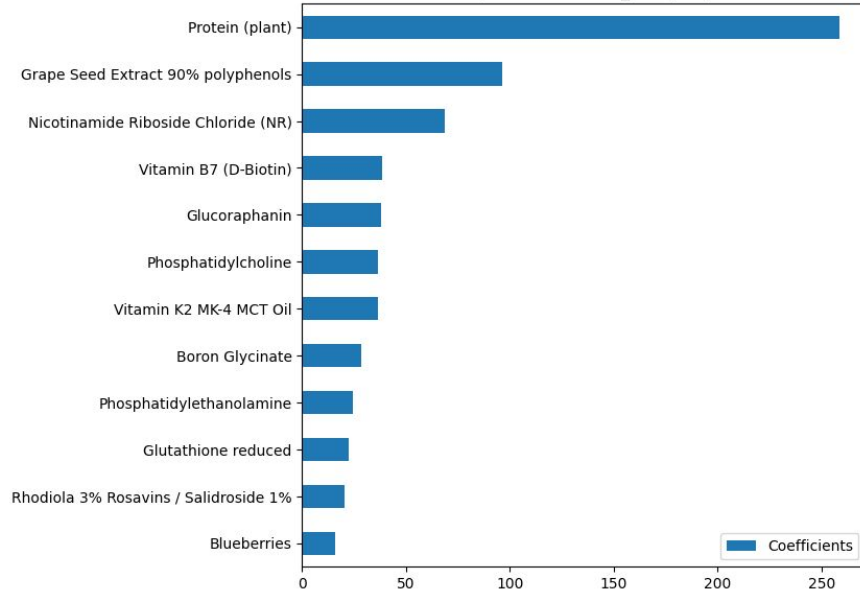
Regresión lineal



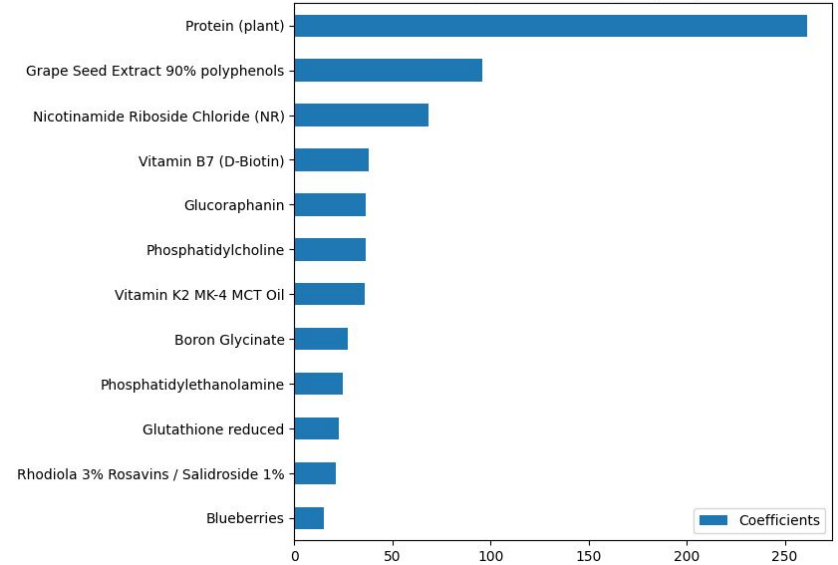
Feature importance



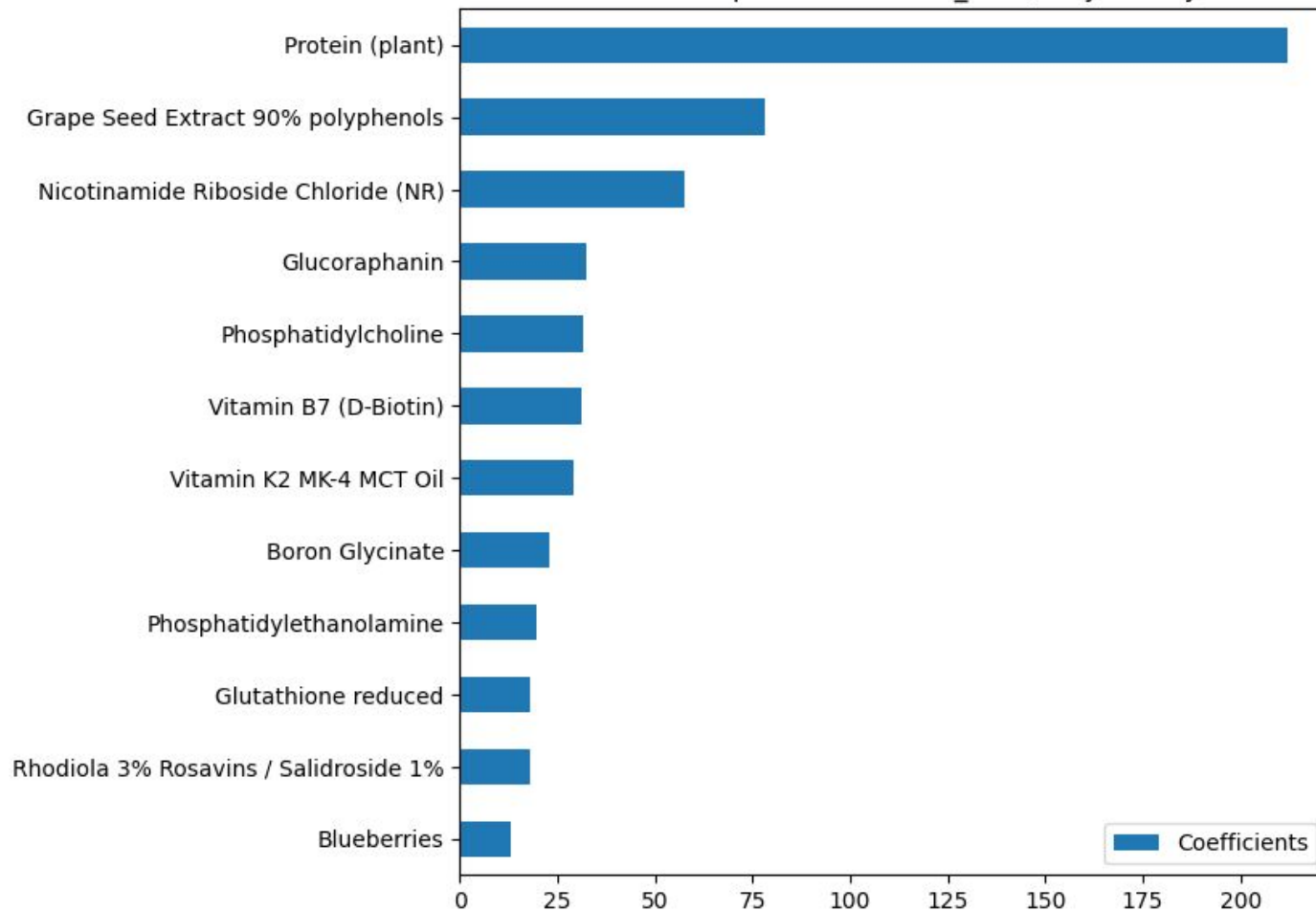
Feature importance for vo2_max, only women



Feature importance for vo2_max, only men

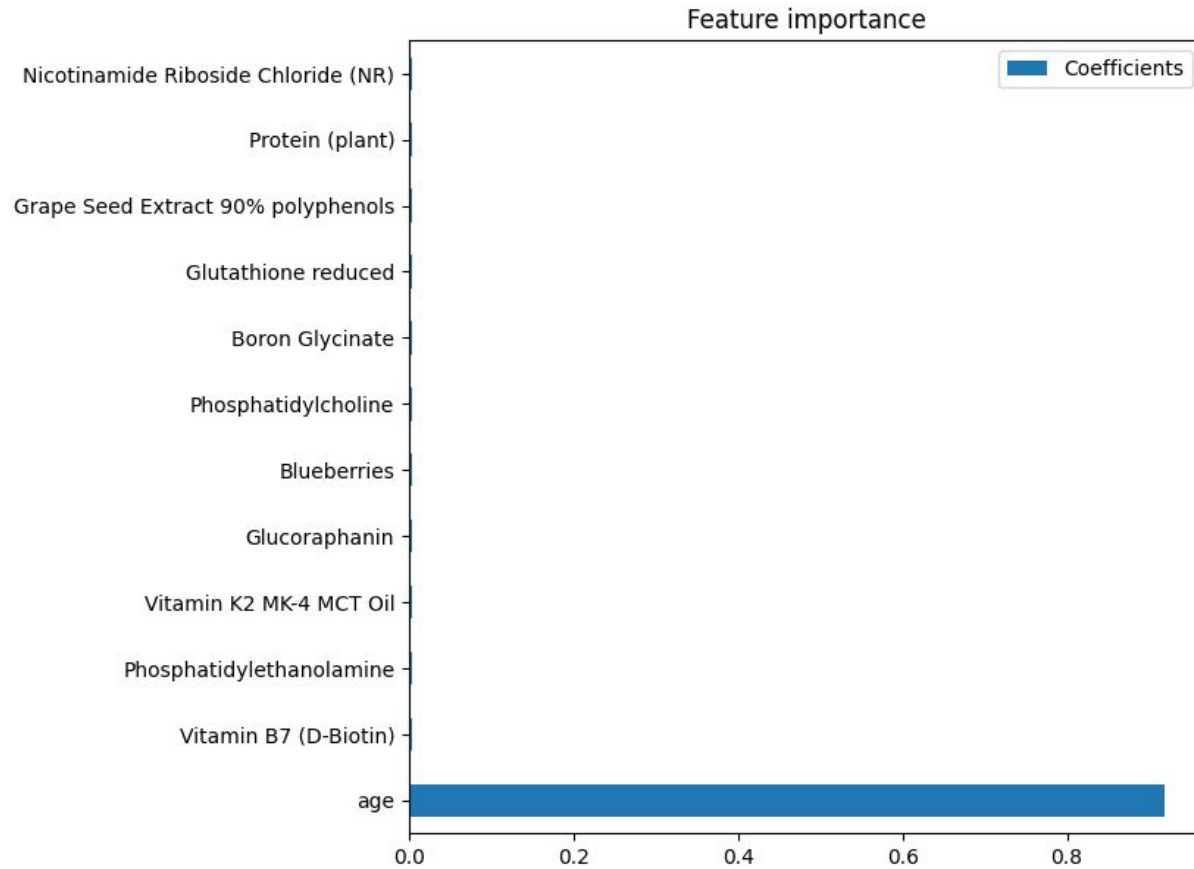


Feature importance for vo2_max, only elderly

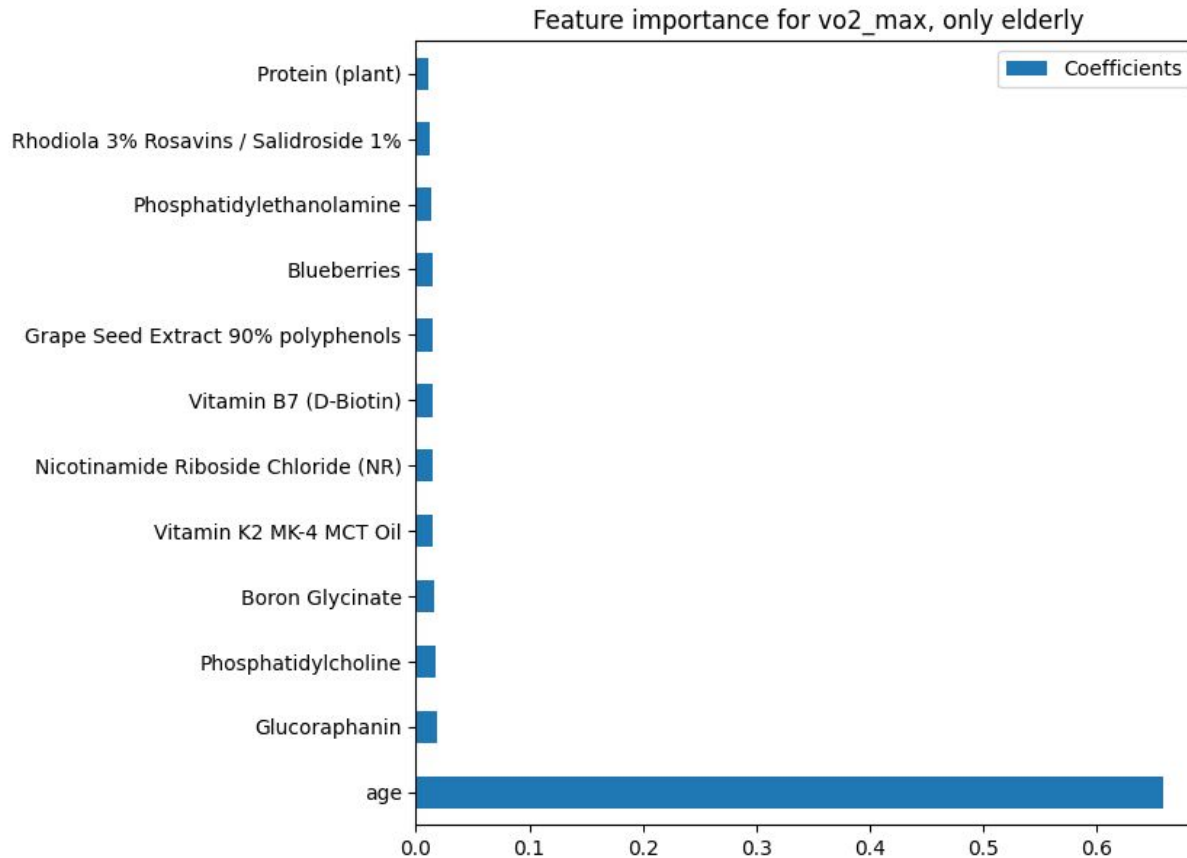


Decision Trees

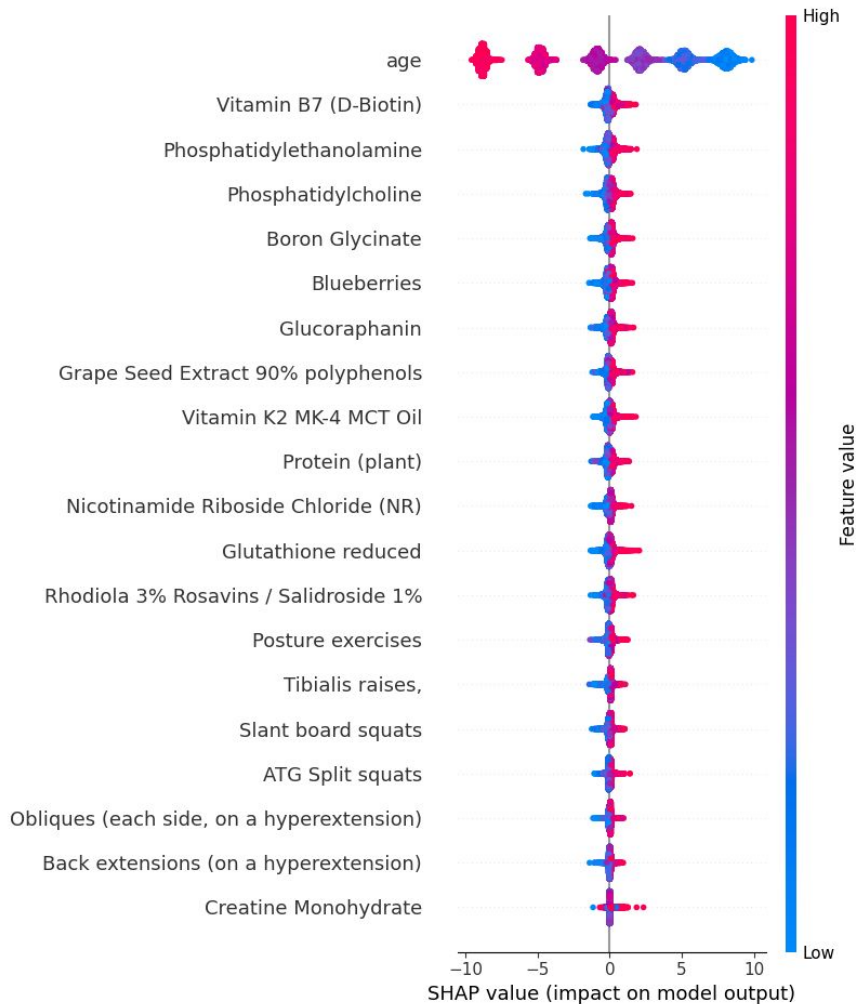




Variables más importantes para Arbol de Decisión, dataset completo



Variables más importantes para Arbol de Decisión, sólo mayores



Intenté sacar la importancia de los features del modelo de regresión lineal usando Shapley Values, y me entregó un resultado muy parecido, la edad es altamente determinante en el resultado final, tanto negativamente como positivamente y las otras variables son las de suplementos alimenticios.

No calculé importancia de conjuntos de features.

4. Bonus

- Todo el código de esta parte está en

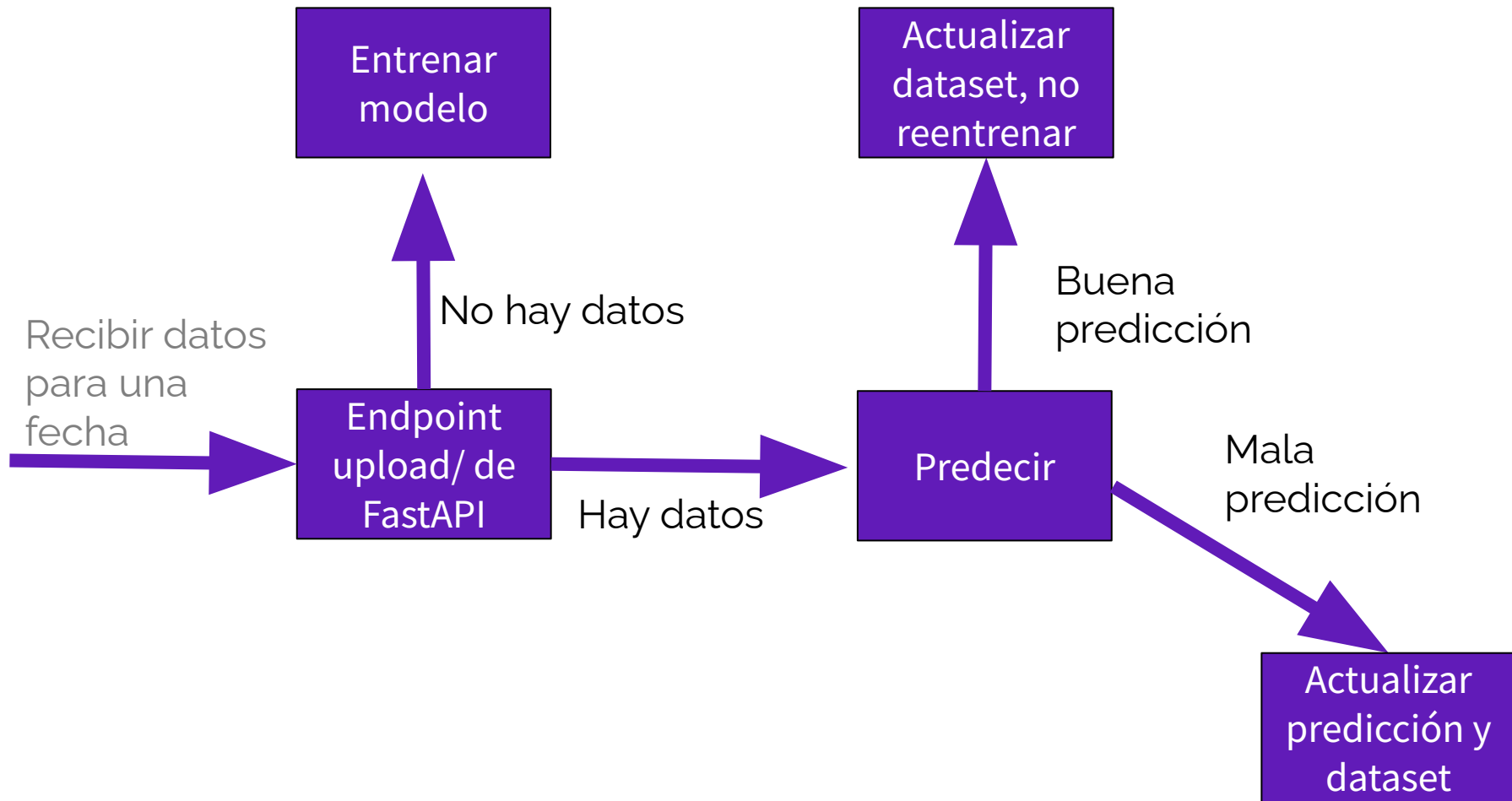
Síntesis

- Se implementa de inicio a fin una app de FastAPI y una clase Model que puede recibir un modelo pre entrenado.
- Se reescribe gran parte de lo que hice en el Jupyter Notebook como métodos de la clase Model.
- Se prueba el modelo dividiendo los datos completos en 25 archivos de 1000 filas porque simular los datos nos daban cosas poco realistas.

Generador de datos sintéticos

- Programa de Python que recibe una fecha en string y el número de datos.
- Todos los valores son random entre 0 y el optimal dose del spreadsheet de excel linkeado.

```
○ (sofiachallenge) panconqueso@DESKTOP-0CND4T9:/mnt/c/Documents and Settings/Lenovo/Mis documentos/Python/Sofia_Challenge$ python3 generate_synthetic_data.py 2024/05/17 1000
```



¿Cómo actualizar la predicción?

- Actualizar cada vez que baje el score

```
INFO:      127.0.0.1:46196 - "POST /update HTTP/1.1" 200 OK  
0.9628190003817867  
Updating dataset for future training  
INFO:      127.0.0.1:46198 - "POST /update HTTP/1.1" 200 OK  
0.9691671700746417  
INFO:      127.0.0.1:46200 - "POST /update HTTP/1.1" 200 OK  
0.9651416722960895  
Updating dataset for future training  
INFO:      127.0.0.1:46202 - "POST /update HTTP/1.1" 200 OK  
0.9696559798564769  
INFO:      127.0.0.1:46204 - "POST /update HTTP/1.1" 200 OK  
0.9621519274270373  
Updating dataset for future training
```

Y por si fuera poco

Se crea un Dockerfile que está en el repo.

Con ese se puede subir a alguna nube de preferencia para poner el modelo en producción!

Posibles mejoras

- Probar otros modelos
- Predicciones con ventanas de tiempo
- Tipado fuerte del modelo con Pydantic.

Demo

¿Preguntas?