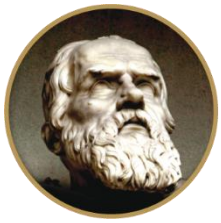
Universidad Galileo
Postgrado en Análisis y Predicción de Datos
Product Development, Sección L
Catedrático: Ing Preng Biba.
Auxiliar: Milton Godinez.
Nombre: Kimberly Alejandra Rivera González
Carné: 20001676



Dataset a utilizar en proyecto

El dataset que utilizaré en el proyecto se llama: Customer Personality Analysis marketing_campaign.csv

El contexto descrito en el dataset es:
Planteamiento del problema: El análisis de la personalidad del cliente es un análisis detallado de los clientes ideales de una empresa. Ayuda a una empresa a comprender mejor a sus clientes y les facilita la modificación de productos de acuerdo con las necesidades, los comportamientos y las preocupaciones específicas de los diferentes tipos de clientes.

El análisis de la personalidad del cliente ayuda a una empresa a modificar su producto en función de sus clientes objetivo de diferentes tipos de segmentos de clientes. Por ejemplo, en lugar de gastar dinero para comercializar un nuevo producto para cada cliente en la base de datos de la empresa, una empresa puede analizar qué segmento de clientes es más probable que compre el producto y luego comercializar el producto solo en ese segmento en particular.

Objetivo: Necesita realizar agrupaciones para resumir los segmentos de clientes.

Cantidad de registros: 2240 filas.

Cuenta con una cantidad de columnas de: 29 columnas.

Descripción de cada una de las columnas: Contenido de Atributos
Persona

ID: identificador único del cliente
Year_Birth: año de nacimiento del cliente
Education: nivel de educación del cliente
Marital_Status: estado civil del cliente
Income: ingresos familiares anuales del cliente
Kidhome: número de niños en el hogar del cliente
Teenhome: Número de adolescentes en el hogar del cliente
Dt_Customer: fecha de inscripción del cliente en la empresa
Recency: número de días desde la última compra del cliente
Queja: 1 si el cliente se quejó en los últimos 2 años, 0 en caso contrario

Productos

MntWines: cantidad gastada en vino en los últimos 2 años
MntFruits: Cantidad gastada en frutas en los últimos 2 años
MntMeatProducts: Cantidad gastada en carne en los últimos 2 años
MntFishProducts: Cantidad gastada en pescado en los últimos 2 años
MntSweetProducts: Cantidad gastada en dulces en los últimos 2 años
MntGoldProds: cantidad gastada en oro en los últimos 2 años

Promoción

NumDealsPurchases: Número de compras realizadas con descuento
AcceptedCmp1: 1 si el cliente aceptó la oferta en la primera campaña, 0 en caso contrario
AcceptedCmp2: 1 si el cliente aceptó la oferta en la segunda campaña, 0 en caso contrario
AcceptedCmp3: 1 si el cliente aceptó la oferta en la tercera campaña, 0 en caso contrario
AcceptedCmp4: 1 si el cliente aceptó la oferta en la cuarta campaña, 0 en caso contrario
AcceptedCmp5: 1 si el cliente aceptó la oferta en la quinta campaña, 0 en caso contrario
Response: 1 si el cliente aceptó la oferta en la última campaña, 0 en caso contrario

Lugar

NumWebPurchases: número de compras realizadas a través del sitio web de la empresa
NumCatalogPurchases: número de compras realizadas mediante un catálogo
NumStorePurchases: número de compras realizadas directamente en las tiendas
NumWebVisitsMonth: número de visitas al sitio web de la empresa en el último mes

La fuente de donde se tomó es de Kaggle:
https://www.kaggle.com/imakash3011/customer-personality-analysis

Visualización de los datos en Jupyter:

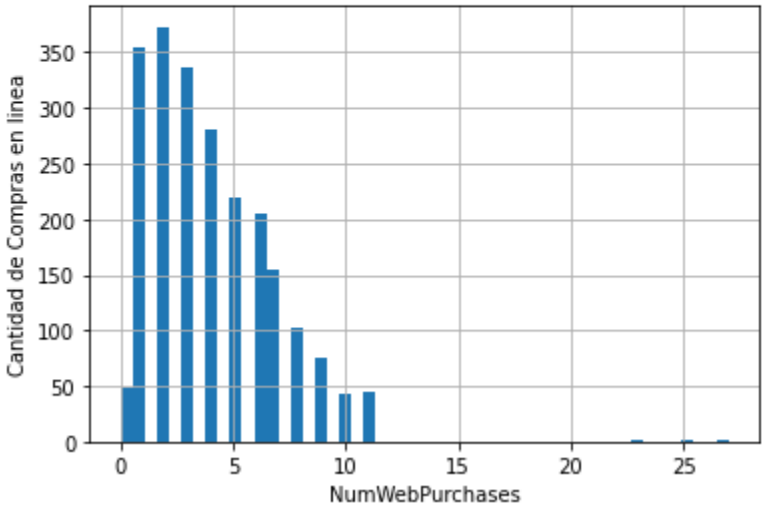
```
data = pd.read_csv('marketing_campaign.csv', sep='\t')
data
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumWebVisitsMonth	AcceptedCmp3
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	58	635	...	7	0
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	38	11	...	5	0
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	26	426	...	4	0
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	26	11	...	6	0
4	5324	1981	PhD	Married	58293.0	1	0	19-01-2014	94	173	...	5	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2235	10870	1967	Graduation	Married	61223.0	0	1	13-06-2013	46	709	...	5	0
2236	4001	1946	PhD	Together	64014.0	2	1	10-06-2014	56	406	...	7	0
2237	7270	1981	Graduation	Divorced	56981.0	0	0	25-01-2014	91	908	...	6	0
2238	8235	1956	Master	Together	69245.0	0	1	24-01-2014	8	428	...	3	0
2239	9405	1954	PhD	Married	52869.0	1	1	15-10-2012	40	84	...	7	0

2240 rows × 29 columns

Target: Se exploró y analizó la data y el target que elegí para el proyecto es NumWebPurchases: número de compras realizadas a través del sitio web de la empresa, esa será la variable que se va a predecir.

```
data['NumWebPurchases'].hist(bins=50, density=False)
plt.xlabel("NumWebPurchases")
plt.ylabel("Cantidad de Compras en linea")
plt.show()
```



Adjunto a este documento el dataset para su visualización:
marketing_campaign.csv

Pasos realizados:
Fueron tres notebooks.

- El primer notebook es el de Proyecto 1 Kimberly Rivera 20001676.ipynb, el cual contiene todo el análisis y toda la limpieza de datos y el llenado de las variables con data faltante.
- Luego el siguiente notebook que se encontrará en la carpeta es llamado Variable_Selection_And_Model_Training P1 - Kimberly Rivera 20001676, el cual contiene todos los modelos que se realizaron y que se entrenaron para validar cual era el mejor modelo que se tomaría para el pipeline, yo utilicé neuronas y elegí la neurona que me dio mejor resultado.
- El tercer notebook es llamado Machine_Learning_Pipeline P1 - Kimberly Rivera 20001676, donde se encuentra toda el esquema y la estructura del pipeline, así como generar el archivo PKL a utilizar.
- En cada uno de los notebooks se encuentran anotaciones y títulos de lo realizado.

API desarrollada.

El API se encuentra en la carpeta **apidemo**
En esta carpeta se encuentran los archivos de:

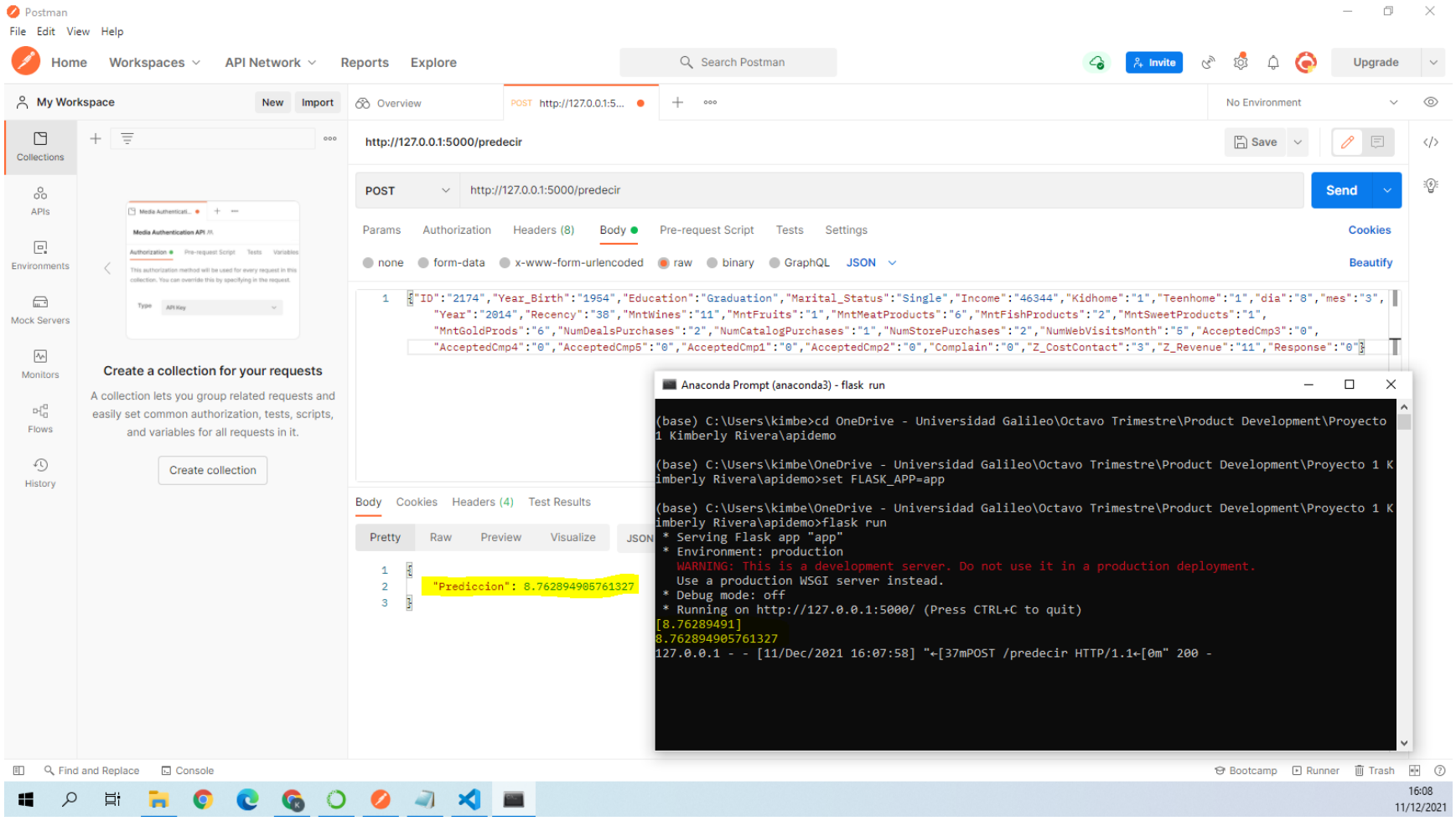
- App.py = archivo donde está el llamado desde postman.
- Config.py = archivo donde está toda la estructura del pipeline.
- My_preprocessors.py = archivo que creamos al inicio para las variables de tiempo.
- Pipeline_predict2.py = es el archivo donde se encuentra el llamado al archivo anteriormente guardado PKL que lleva por nombre WebPurchase_pipeline.pkl
- marketing_campaign.csv = archivo de prueba
- Pasos en el Prompt.txt = Un archivo que yo cree para realizar las pruebas en el postman y en el prompt de anaconda y contiene esta información:

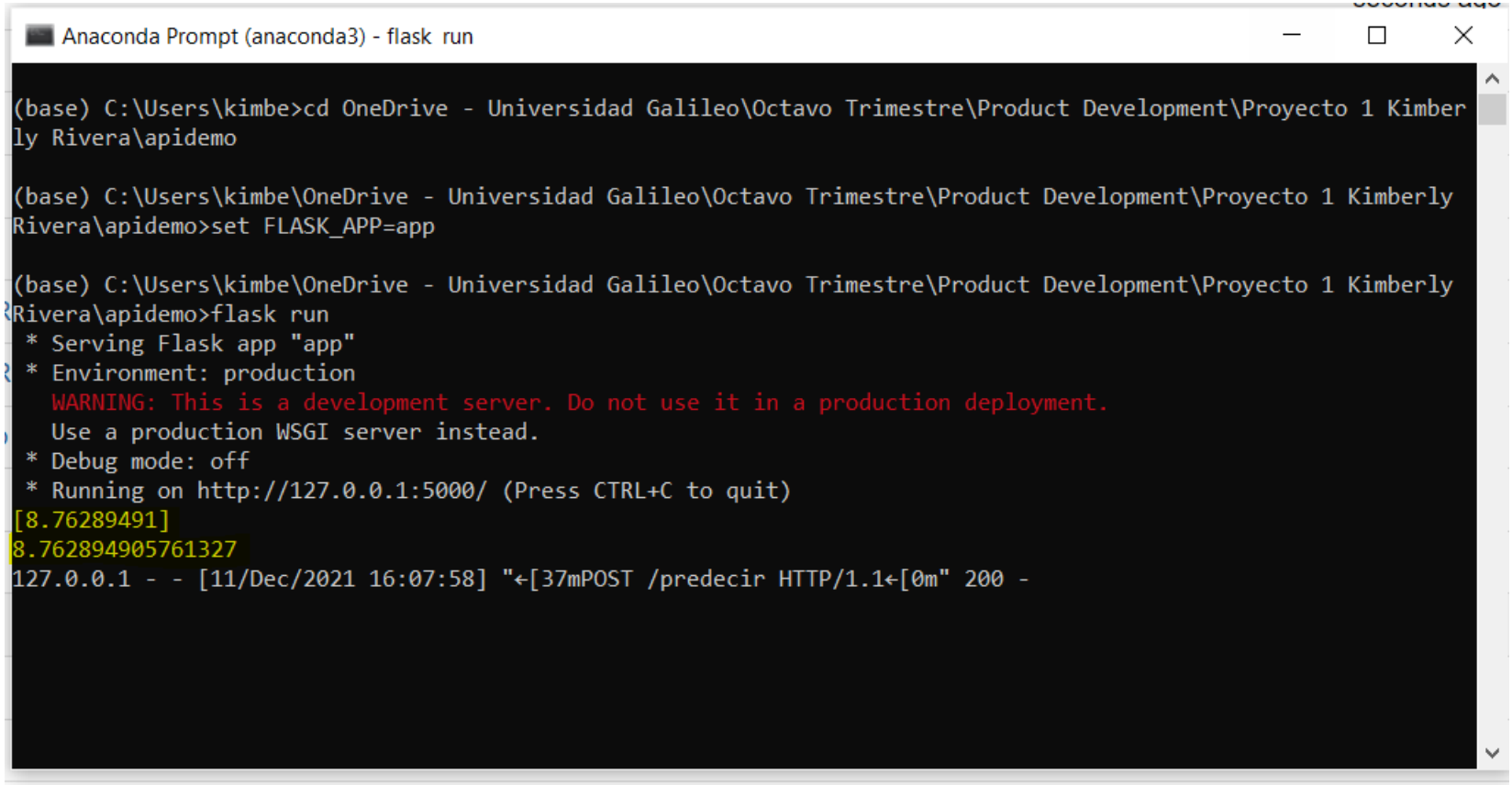
  C:\Users\kimbe\
  cd OneDrive - Universidad Galileo
  cd Octavo Trimestre
  cd Product Development
  cd Proyecto 1 Kimberly Rivera
  cd apidemo
  python pipeline_predict2.py
  OneDrive - Universidad Galileo\Octavo Trimestre\Product Development\Proyecto 1 Kimberly Rivera\apidemo

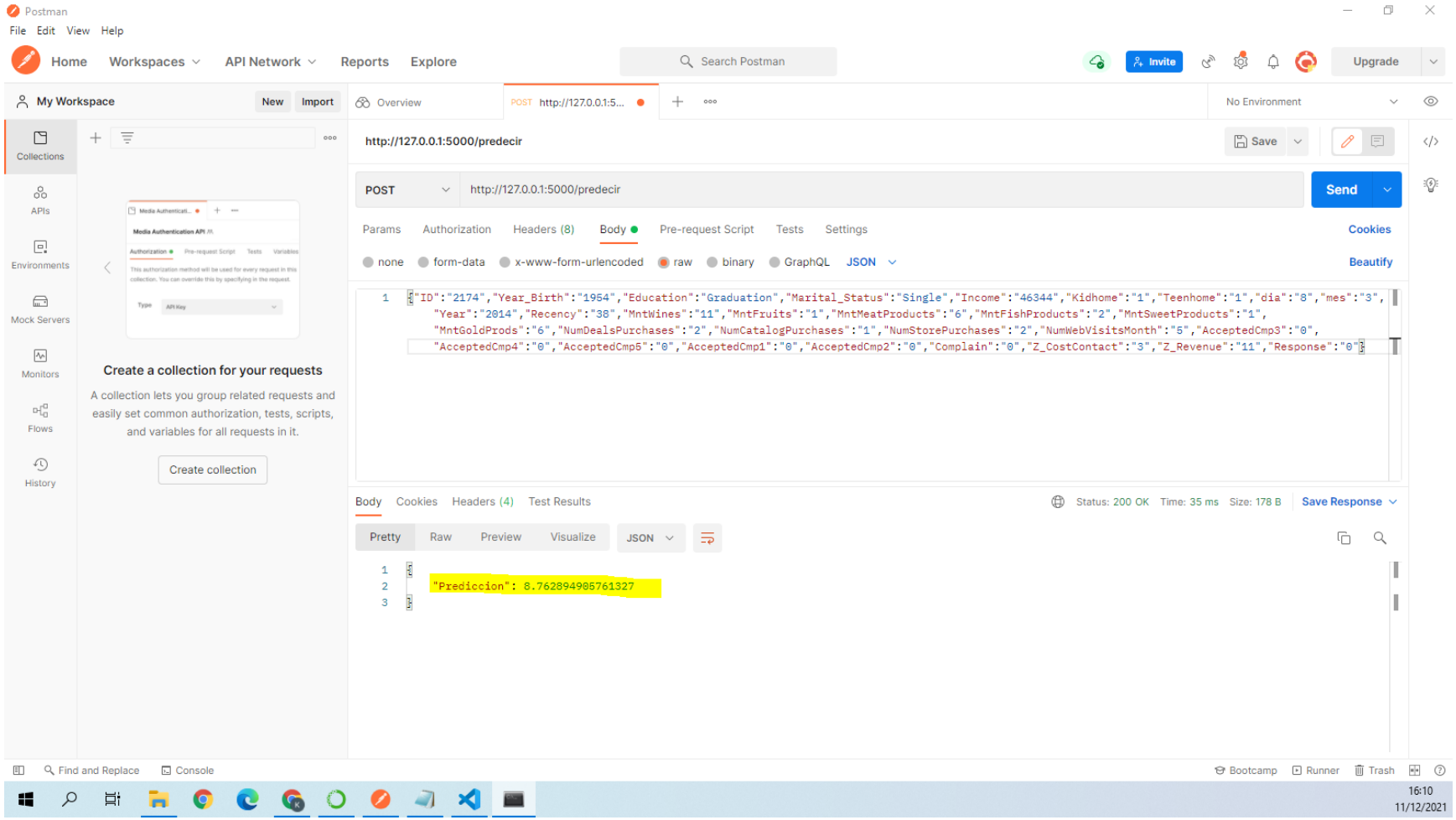
  set FLASK_APP=app
  flask run

  Prueba en postman:
  {"ID":"2174","Year_Birth":"1954","Education":"Graduation","Marital_Status":"Single","Income":"46344","Kidhome":"1","Teenhome":"1","dia":"8","mes":"3","Year":"2014","Recency":"38","MntWines":"11","MntFruits":"1","MntMeatProducts":"6","MntFishProducts":"2","MntSweetProducts":"1","MntGoldProds":"6","NumDealsPurchases":"2","NumCatalogPurchases":"1","NumStorePurchases":"2","NumWebVisitsMonth":"5","AcceptedCmp3":"0","AcceptedCmp4":"0","AcceptedCmp5":"0","AcceptedCmp1":"0","AcceptedCmp2":"0","Complain":"0","Z_CostContact":"3","Z_Revenue":"11","Response":"0"}

- bbu5l-oriqg.json = el archivo .json que se utiliza en postman.
- Imágenes que confirman la conexión entre el postman y el api creada:
  - API Y POSTMAN.PNG
  - API.PNG
  - POSTMAN.PNG


Postman
File Edit View Help
Home Workspaces API Network Reports Explore
Search Postman
Invite
Upgrade
My Workspace New Import
Overview
POST http://127.0.0.1:5...
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Create a collection for your requests
A collection lets you group related requests and easily set common authorization, tests, scripts, and variables for all requests in it.
Create collection
http://127.0.0.1:5000/predecir
Save
POST
http://127.0.0.1:5000/predecir
Send
Params Authorization Headers (8) Body Pre-request Script Tests Settings
Cookies
none form-data x-www-form-urlencoded raw binary GraphQL JSON
Beautify
1 {"ID":"2174","Year_Birth":"1954","Education":"Graduation","Marital_Status":"Single","Income":"46344","Kidhome":"1","Teenhome":"1","dia":"8","mes":"3",
"Year":"2014","Recency":"38","MntWines":"11","MntFruits":"1","MntMeatProducts":"6","MntFishProducts":"2","MntSweetProducts":"1",
"MntGoldProds":"6","NumDealsPurchases":"2","NumCatalogPurchases":"1","NumStorePurchases":"2","NumWebVisitsMonth":"5","AcceptedCmp3":"0",
"AcceptedCmp4":"0","AcceptedCmp5":"0","AcceptedCmp1":"0","AcceptedCmp2":"0","Complain":"0","Z_CostContact":"3","Z_Revenue":"11","Response":"0"}
Body Cookies Headers (4) Test Results
Pretty Raw Preview Visualize JSON
1 {
2 "Prediccion": 8.762894905761327
3 }
Anaconda Prompt (anaconda3) - flask run
(base) C:\Users\kimbe>cd OneDrive - Universidad Galileo\Octavo Trimestre\Product Development\Proyecto 1 Kimberly Rivera\apidemo
(base) C:\Users\kimbe\OneDrive - Universidad Galileo\Octavo Trimestre\Product Development\Proyecto 1 Kimberly Rivera\apidemo>set FLASK_APP=app
(base) C:\Users\kimbe\OneDrive - Universidad Galileo\Octavo Trimestre\Product Development\Proyecto 1 Kimberly Rivera\apidemo>flask run
* Serving Flask app "app"
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
[8.76289491]
8.762894905761327
127.0.0.1 - - [11/Dec/2021 16:07:58] "←[37mPOST /predecir HTTP/1.1←[0m" 200 -
Find and Replace Console
Bootcamp Runner Trash
16:08
11/12/2021


Anaconda Prompt (anaconda3) - flask run
(base) C:\Users\kimbe>cd OneDrive - Universidad Galileo\Octavo Trimestre\Product Development\Proyecto 1 Kimberly Rivera\apidemo
(base) C:\Users\kimbe\OneDrive - Universidad Galileo\Octavo Trimestre\Product Development\Proyecto 1 Kimberly Rivera\apidemo>set FLASK_APP=app
(base) C:\Users\kimbe\OneDrive - Universidad Galileo\Octavo Trimestre\Product Development\Proyecto 1 Kimberly Rivera\apidemo>flask run
* Serving Flask app "app"
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
[8.76289491]
8.762894905761327
127.0.0.1 - - [11/Dec/2021 16:07:58] "←[37mPOST /predecir HTTP/1.1←[0m" 200 -


Postman
File Edit View Help
Home Workspaces API Network Reports Explore
Search Postman
Invite
Upgrade
My Workspace New Import
Overview
POST http://127.0.0.1:5...
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Create a collection for your requests
A collection lets you group related requests and easily set common authorization, tests, scripts, and variables for all requests in it.
Create collection
http://127.0.0.1:5000/predecir
Save
POST
http://127.0.0.1:5000/predecir
Send
Params Authorization Headers (8) Body Pre-request Script Tests Settings
Cookies
none form-data x-www-form-urlencoded raw binary GraphQL JSON
Beautify
{"ID":"2174","Year_Birth":"1954","Education":"Graduation","Marital_Status":"Single","Income":"46344","Kidhome":"1","Teenhome":"1","dia":"8","mes":"3",
"Year":"2014","Recency":"38","MntWines":"11","MntFruits":"1","MntMeatProducts":"6","MntFishProducts":"2","MntSweetProducts":"1",
"MntGoldProds":"6","NumDealsPurchases":"2","NumCatalogPurchases":"1","NumStorePurchases":"2","NumWebVisitsMonth":"5","AcceptedCmp3":"0",
"AcceptedCmp4":"0","AcceptedCmp5":"0","AcceptedCmp1":"0","AcceptedCmp2":"0","Complain":"0","Z_CostContact":"3","Z_Revenue":"11","Response":"0"}
Body Cookies Headers (4) Test Results
Status: 200 OK Time: 35 ms Size: 178 B
Save Response
Pretty Raw Preview Visualize JSON
{
"Prediccion": 8.762894905761327
}
Find and Replace Console
Bootcamp Runner Trash
16:10
11/12/2021