

TEST TRAIN

```
http://127.0.0.1:8000/train
POST
Params
         Authorization Headers (9) Body Pre-request Script Tests Settings
■ none ■ form-data ■ x-www-form-urlencoded ● raw ■ binary ■ GraphQL JSON ∨
          ·"data": ·[
                  "life expentancy": 65.0,
                 "adult_mortality": 263.0,
                 ·"infant_deaths": 62,
                 ·"alcohol": 0.01,
                 "percentage_expenditure": 71.27962362,
                 "hepatitis_B": 65.0,
                 ·"measles": 1154,
                 ·"bmi": 19.1,
                 -"under_five_deaths": 83,
                  "polio": 6.0,
                  "total_expenditure": 8.16,
                 ·"diphtheria": 65.0,
                  "hiv_aids": 0.1,
                  "gdp": 584.25921,
                  "population": 33736494.0,
 19
                  "thinness_10_19_years": 17.2,
                 "thinness_5_9_years": 17.3,
                 "income_composition_of_resources": 0.479,
 21
                 ··"schooling": 10.1
```



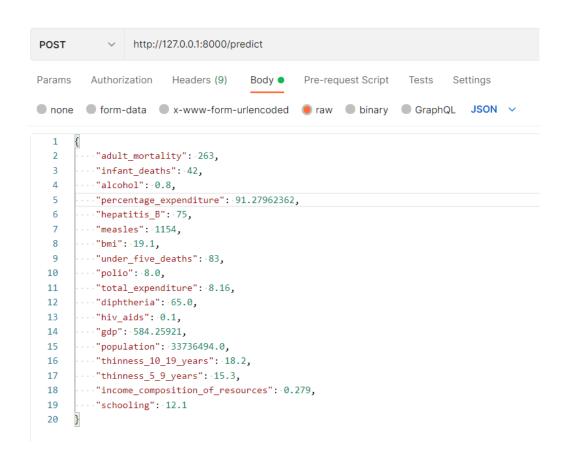
PREDICT 1 - CORRECT

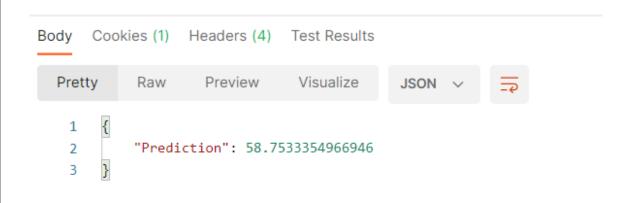
```
POST
           http://127.0.0.1:8000/predict
                                                                 Settinas
        Authorization Headers (9)
                                 Body •
                                          Pre-request Script
1
  2
         "adult_mortality": 263,
         "infant_deaths": 62,
  4
        "alcohol": 0.01,
         "percentage_expenditure": 71.27962362,
         "hepatitis_B": 65,
         "measles": 1154,
         ·"bmi": 19.1,
  9
         "under five deaths": 83,
 10
        "polio": 6.0,
 11
         "total_expenditure": 8.16,
 12
        "diphtheria": 65.0,
 13
        "hiv aids": 0.1,
 14
        gdp": 584.25921,
 15
        "population": 33736494.0,
 16
        "thinness_10_19_years": 17.2,
        · "thinness_5_9_years": 17.3,
 17
 18
        "income composition of resources": 0.479,
 19
        ·"schooling": 10.1
 20
```

```
Pretty Raw Preview Visualize JSON V

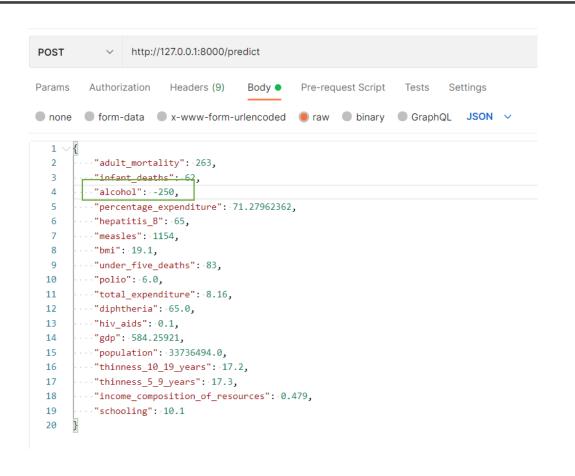
1 {
2 "Prediction": 59.45177556967521
3 }
```

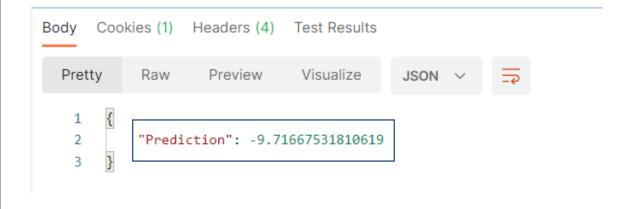
PREDICT 2 - CORRECT



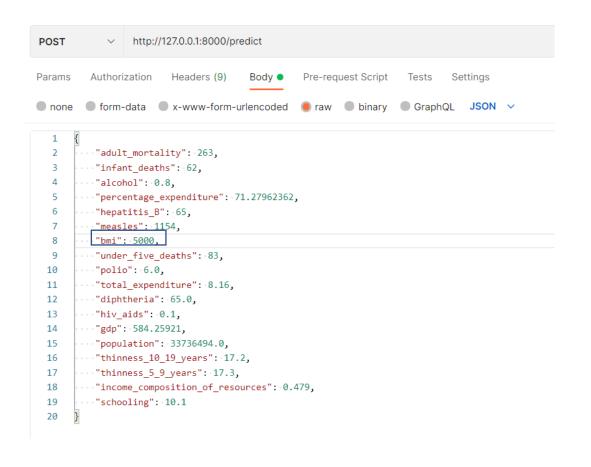


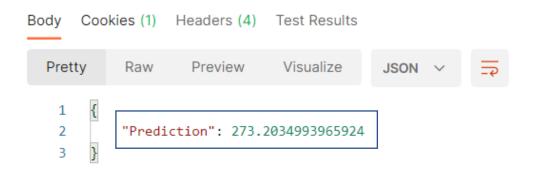
PREDICT 3 - WRONG





PREDICT 4





PREDICT 5

```
http://127.0.0.1:8000/predict
POST
Params
         Authorization
                       Headers (9)
                                      Body •
                                               Pre-request Script
                                                                  Tests
                                                                          Settings
        form-data x-www-form-urlencoded raw binary GraphQL JSON >
         "adult_mortality": 263,
         "infant_deaths": 62,
         "percentage_expenditure": 71.27962362,
         · "hepatitis_B": 65,
         ··"measles": 1154,
         ··"bmi": ·19.1,
        · · "under_five_deaths": 83,
        .."polio": 6.0,
        "total_expenditure": 8.16,
 10
        ... "diphtheria": 65.0,
 11
 12

· "hiv_aids": 0.1,
 13
        ··"gdp": 584.25921,
 14
        · "population": 33736494.0,
        "thinness_10_19_years": 17.2,
 15
        "thinness_5_9_years": 17.3,
 16
 17
         "income_composition_of_resources": 0.479,
 18
         ··"schooling": 10.1
 19
```

```
Cookies (1) Headers (4) Test Results
                   Preview
Pretty
          Raw
                                Visualize
                                             JSON V
 1 \vee \{
          "detail": [
 3
 4
                   "loc": [
 5
                       "body",
 6
                       "alcohol"
 8
                   "msg": "field required",
                   "type": "value error.missing"
 9
10
11
12
```

CONCLUSIONES

- En total se tienen 5 pruebas de predict, y 1 de train.
- 1. Train: Carga una tabla de datos y devuelve la métrica de R^2.
- 2. Predict: Prueba y valor lógico correctos.
- 3. Predict: Prueba y valor lógico correctos.
- 4. Predict: Prueba correcta y valor lógico incorrecto.
- 5. Predict: Prueba correcta y valor lógico incorrecto.
- 6. Predict: Prueba incorrecta.
- Como conclusiones de porque un resultado es incorrecto se tiene un explicación visible de los datos. Cuando una variable esta fuera de su rango normal (definase normal, como dentro de los limites estipulados por el negocio y la naturaleza para dicho componente) la predicción es errorea. Un segundo resultado no evidente por lo obtenido en el análisis del laboratio 3 puede ser entrenar el modelo con datos que tengan problemas dentro de los supuestos.
- Como conclusión cuando la aplicación falla se debe unicamente a la violación de los parametros del **json** pasados en postamn, es decir, falta algun parametro.
- Como medida de solución, se propone: Realizar una validación por medio del API de los datos, buscando comprobar si todas las variables estan dentro del rango
 normal, o en su defecto, usar un SimpleImputer de la libreria Sklearn. Así se logra validar que la información presentada sea correcta. Segundo, programar filtros
 personalizados que permitan remover columnas o transformar los datos cuando un supuesto se viole con el fin de saber que los resultados del modelo son fiables.
 Tercero, y ya respecto al API, permiter que no se pasen ciertos campos sin marcar error, pero que dichos campos sean luego imputados por medio de alguna tecnica.