Laboratorio 4: Despliegue de modelos de Machine Learning

Gabriela Cagua Bolívar - 201812944

Juan Andrés Méndez Galvis - 20181580138

Juan Andrés Romero Colmenares – 202013449

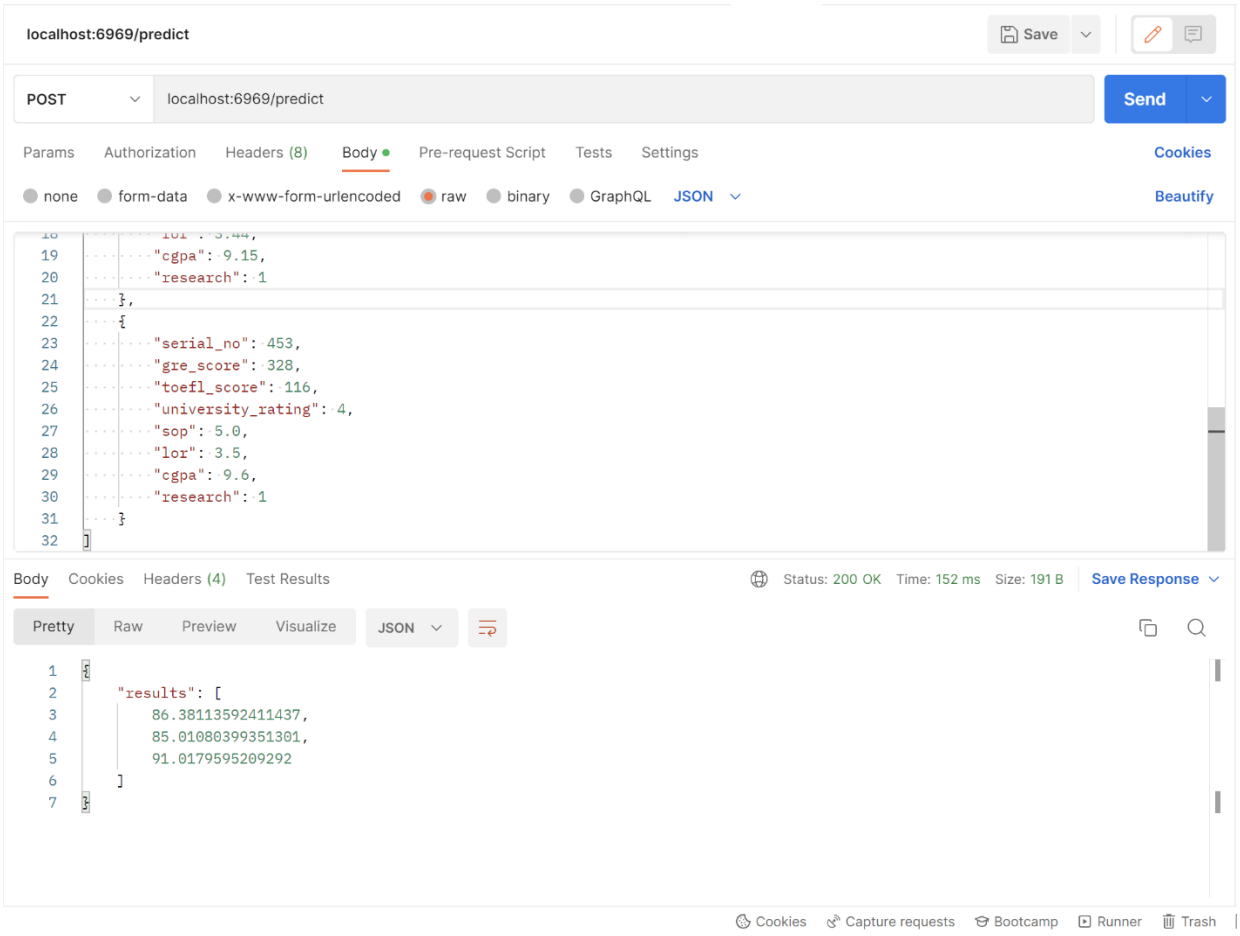
1. Escenarios de Prueba

Para este laboratorio se desarrollaron distintos escenarios de prueba del API implementada.

Pruebas del endpoint /predict

Prueba 1: En esta prueba se introducirán datos coherentes y se revisará el resultado del modelo.

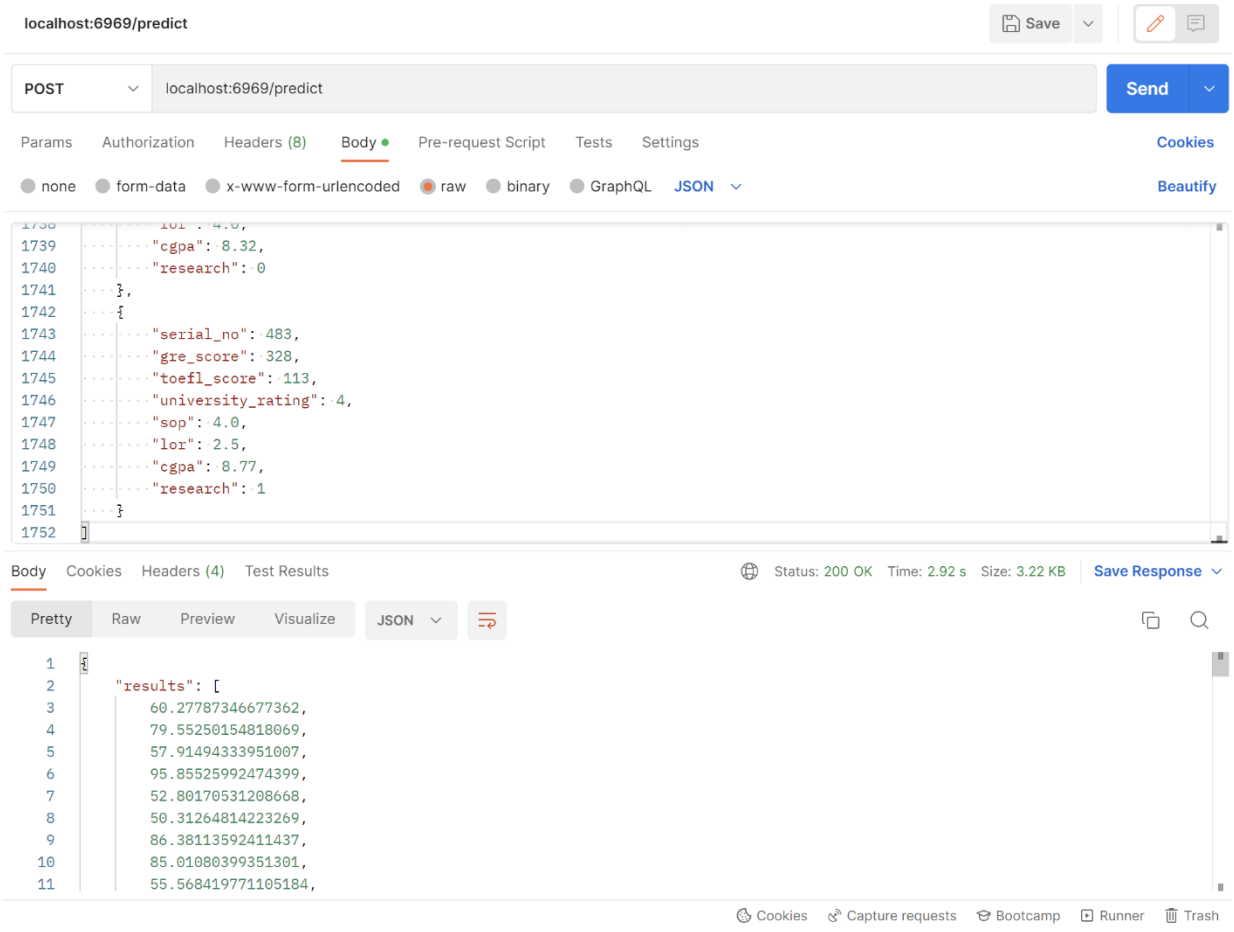
Datos utilizados: Escenario 1



Explicación: En este escenario se encontró que la respuesta del API fue coherente y está en linea con los resultados obtenidos en el laboratorio anterior.

Prueba 2: En esta prueba se introducirá un alto volumen de datos (generado del laboratorio 3) y se revisará el resultado del modelo

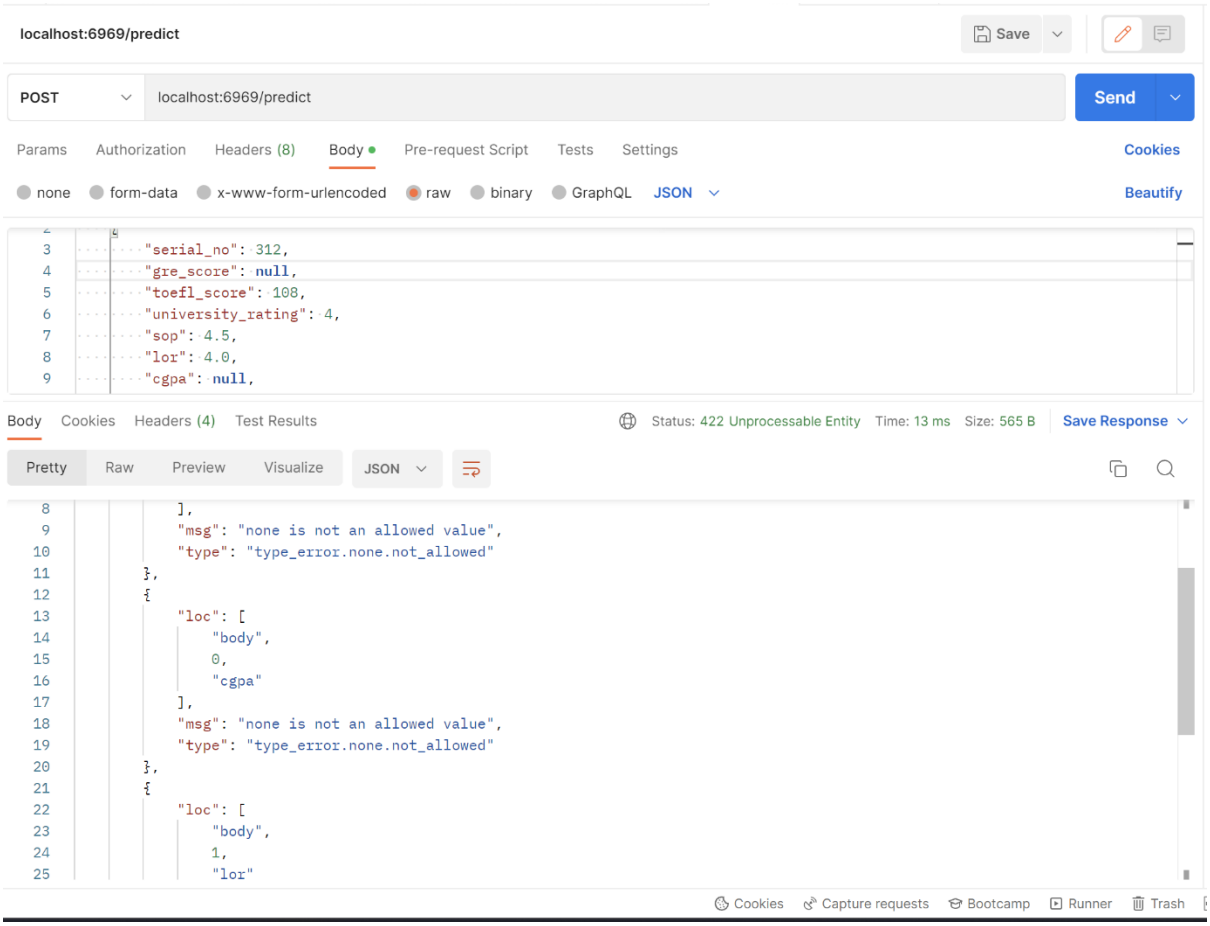
Datos utilizados: Escenario 2



Explicación: En este escenario se puede ver que el API aceptó y procesó un volumen alto de entidades para predecir. Los resultados son coherentes y están en línea con lo encontrado en el laboratorio anterior.

Prueba 3: En esta prueba se intentará introducir valores nulos a diferentes columnas

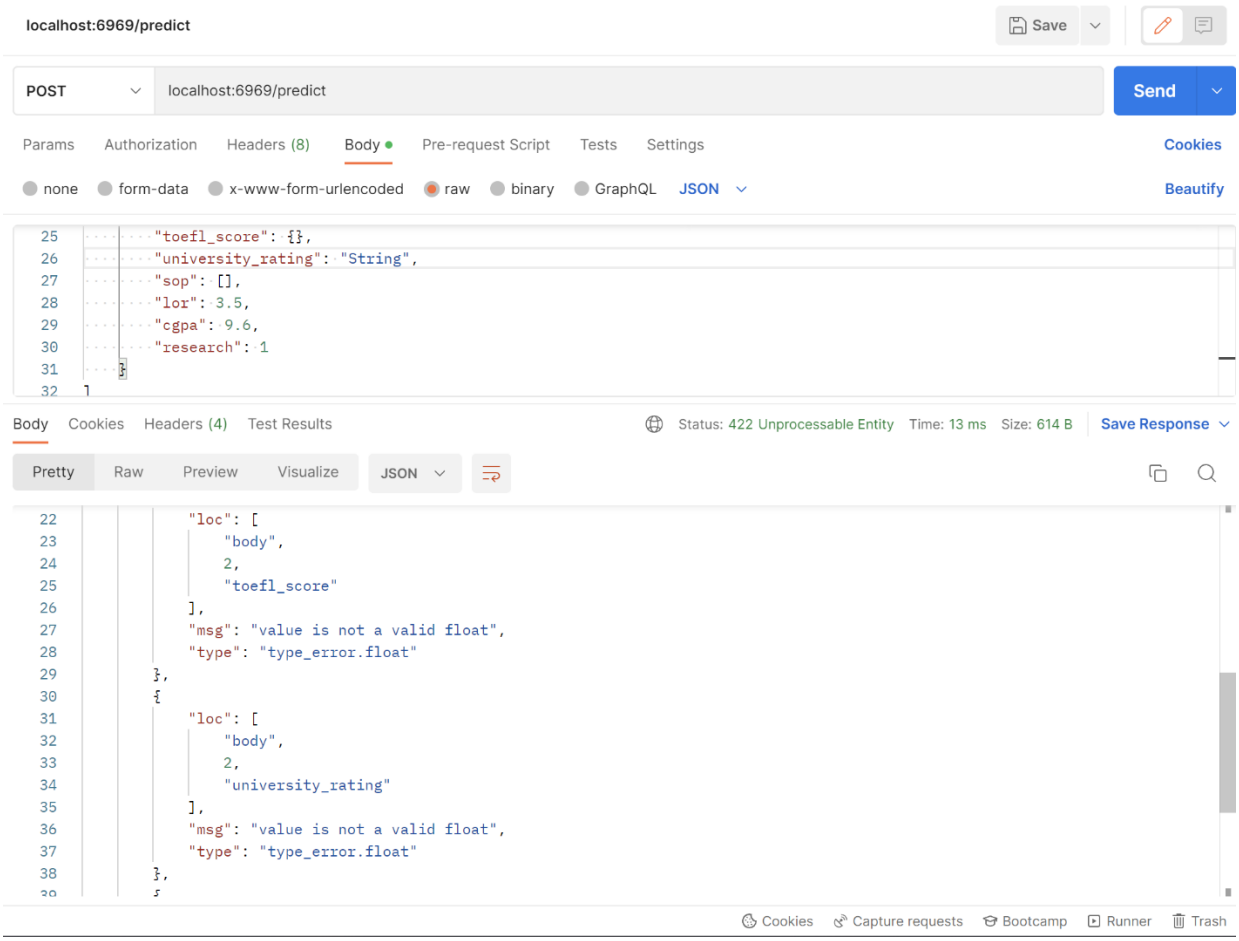
Datos utilizados: [Escenario 3]



Explicación: Para este escenario, el API identificó que existían valores nulos en el body de la prueba y devolvió una respuesta de error indicando que no acepta valores de este tipo. Si el API hubiera aceptado los datos, es posible que se hubieran producido errores en el mejor caso o predicciones erróneas en el peor.

Prueba 4: En esta prueba se intentarán introducir valores que no tienen sentido a las features elegidas (valores con otros tipos de dato)

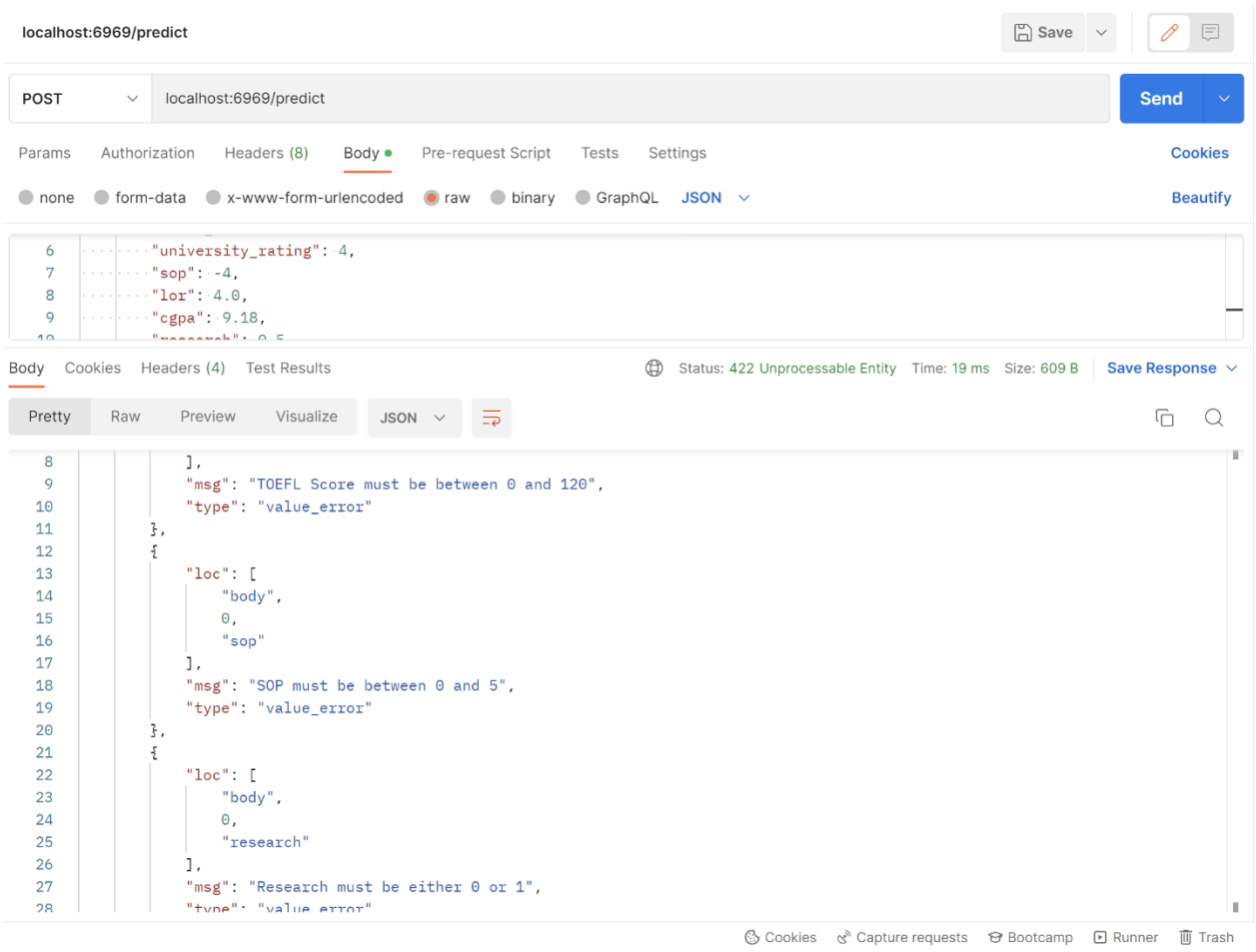
Datos utilizados: Escenario 4



Explicación: Similar al escenario anterior, el API encontró que había tipos de datos no coherentes en el body de la petición recibida, debido a esto, se decidió retornar un mensaje de error que mostrara los campos de cada predictor que tuvieran dicho error. Si esto no hubiera sido manejado, el modelo fallaría y causaría una excepción.

Prueba 5: En esta prueba se intentarán introducir valores con tipo de dato correcto, pero en rangos que no tienen sentidos para el negocio

Datos utilizados: Escenario 5

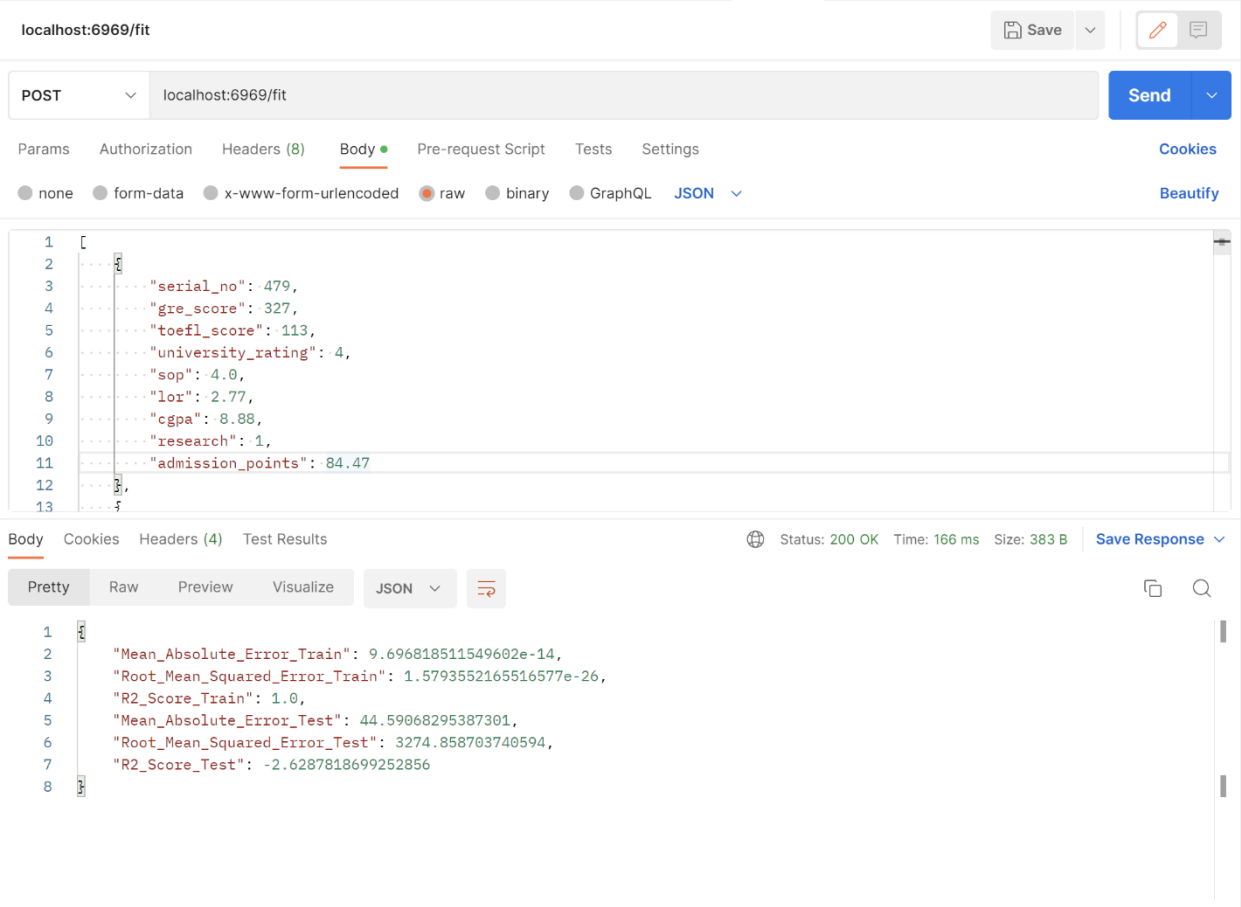


Explicación: Para este escenario se ve cómo el API, aunque reciba tipos de datos coherentes, no acepta valores que estén fuera de los rangos establecidos por el negocio. Si se aceptaran, se llegarían a producir resultados incoherentes, los cuales serían totalmente incorrectos.

Pruebas del endpoint /fit

Prueba 1: En esta prueba se introducirán datos coherentes y se revisará el resultado del modelo.

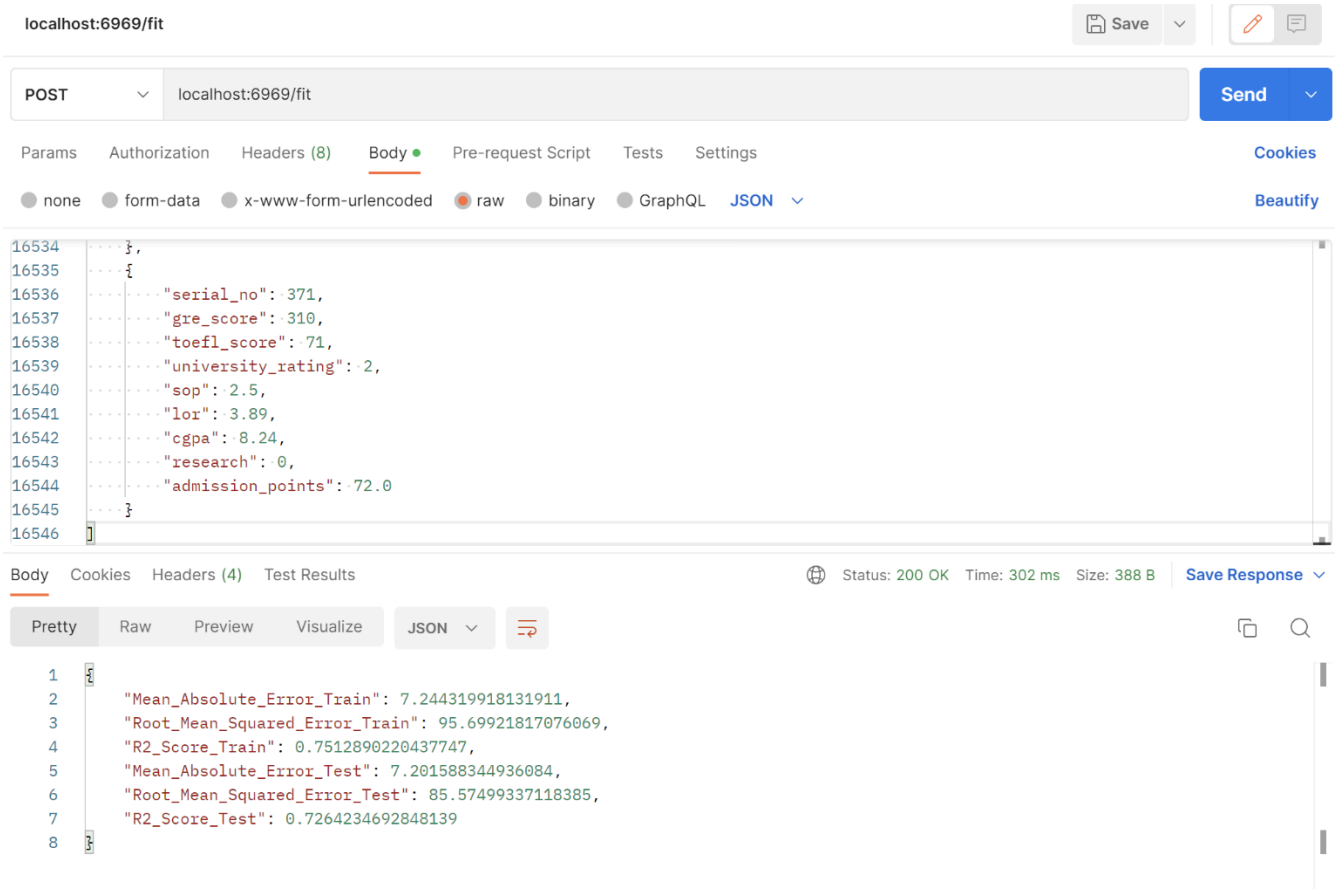
Datos utilizados: Escenario 1



Explicación: En este caso, se presentaron datos coherentes en el body de la petición del fit, sin embargo, se obtuvieron métricas extrañas debido al bajo volumen de entidades enviadas para el entrenamiento (se enviaron alrededor de 22 entidades). Este resultado es coherente, aunque se recomienda no entrenar con una cantidad tan baja de elementos.

Prueba 2: En esta prueba se introducirá un alto volumen de datos (generado del laboratorio 3) y se revisará el resultado del modelo

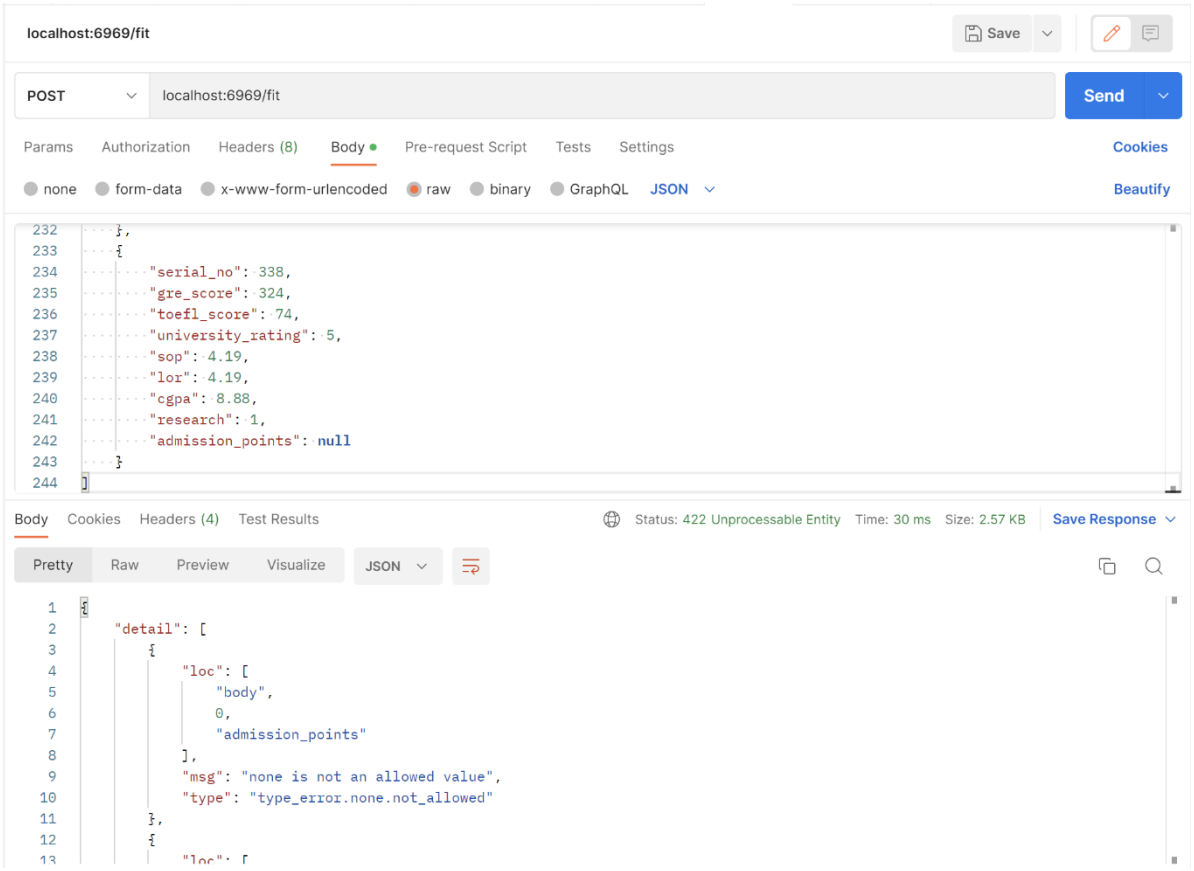
Datos utilizados: Escenario 2



Explicación: En este caso, se le mandó al endpoint alrededor de 719 entidades, con las cuales se dieron unas métricas más coherentes y similares a las obtenidas en el laboratorio 3. Se puede ver que los resultados obtenidos están en linea con lo que se obtendría normalmente en el notebook del lab 3.

Prueba 3: En esta prueba se intentará introducir valores nulos a la variable objetivo

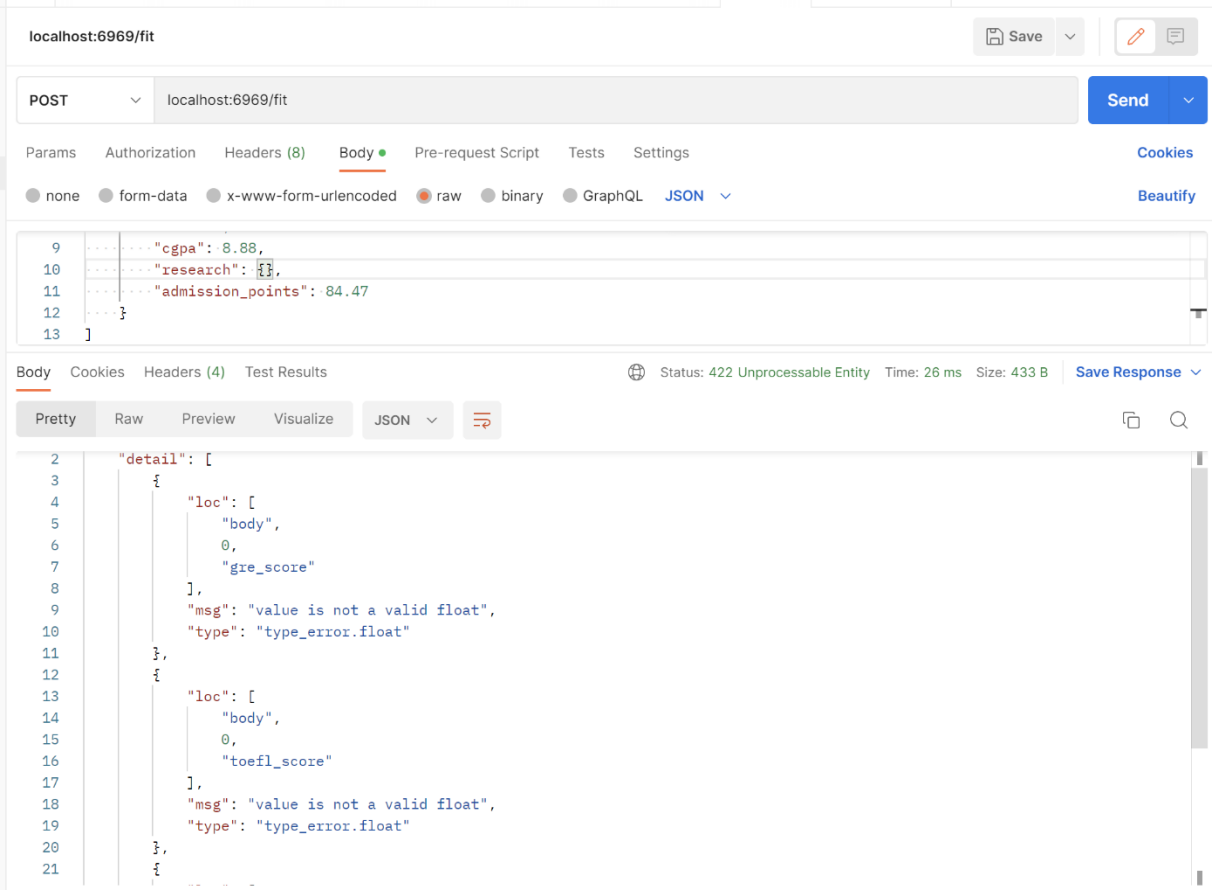
Datos utilizados: Escenario 3



Explicación: Eneste escenario, el API identificó que existían valores nulos en la variable objetivo de la prueba, por lo que devolvió una respuesta de error indicando que no acepta valores de este tipo. Si el API hubiera aceptado los datos, es posible de que se hubieran producido errores en el mejor caso o un entrenamiento completamente incorrecto en el peor.

Prueba 4: En esta prueba se intentarán introducir valores que no tienen sentido a las features elegidas (valores con otros tipos de dato)

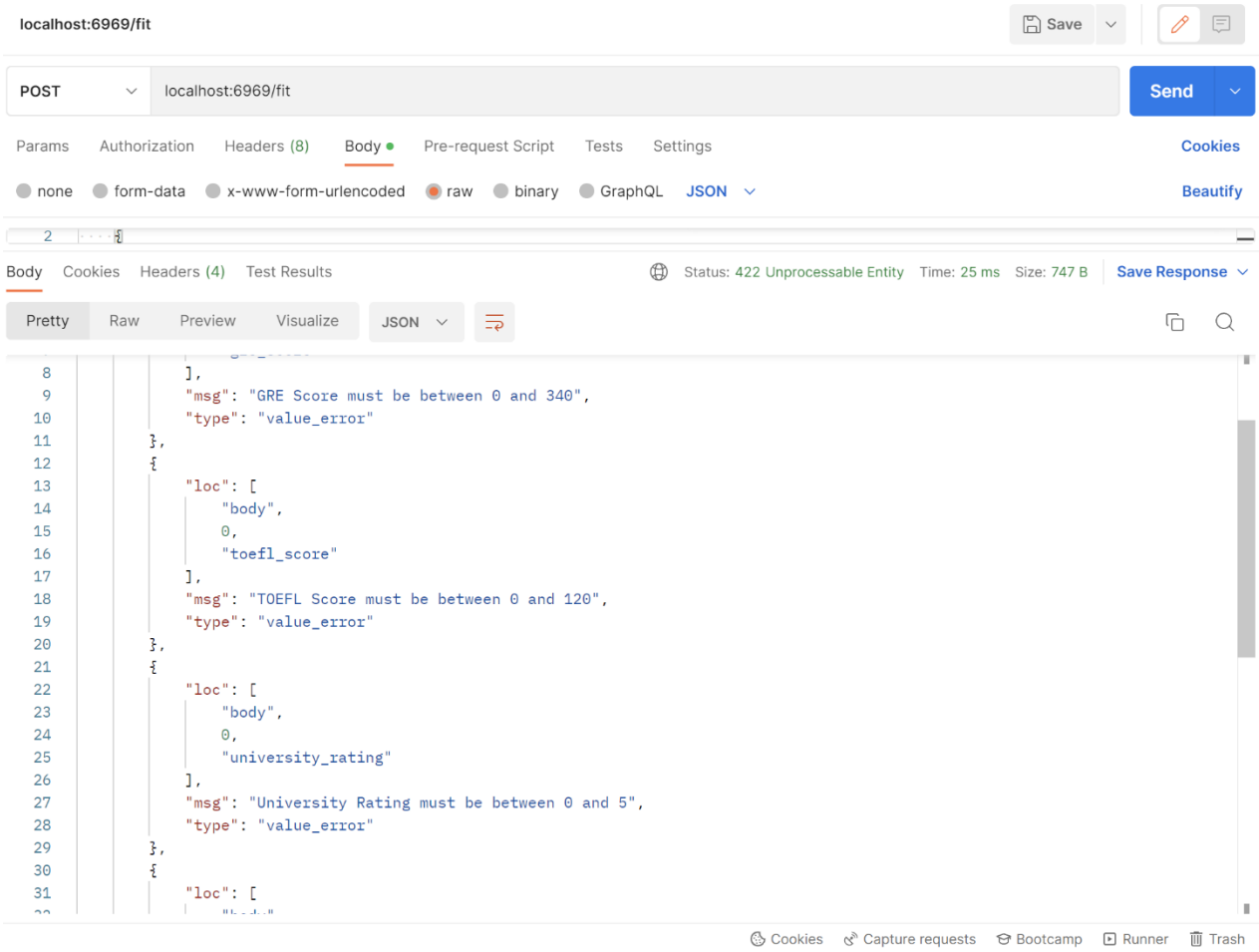
Datos utilizados: Escenario 4



Explicación: Similar al escenario anterior, el API encontró que habían tipos de datos no coherentes en el body de la petición recibida, debido a esto, se decidió retornar un mensaje de error que mostrara los campos de cada elemento que tuvieran dicho error. Si esto no hubiera sido manejado, el modelo fallaría y causaría una excepción.

Prueba 5: En esta prueba se intentarán introducir valores con tipo de dato correcto, pero en rangos que no tienen sentidos para el negocio

Datos utilizados: Escenario 5



Explicación:

Para este escenario se ve cómo el API, aunque reciba tipos de datos coherentes, no acepta valores que estén fuera de los rangos establecidos por el negocio. En su lugar, retorna mensajes de error indicando los campos que violan dichas reglas y no acepta la entrada. Si se aceptaran, se llegaría a entrenar un modelo incorrecto que realizaría predicciones que están muy fuera de lugar de lo que quiere el negocio.

Estrategia para mitigar incoherencias y errores de ejecución:

Para mitigar cualquier tipo de incoherencias y errores generados en el API se sugiere introducir reglas personalizadas dentro del programa que analice la entrada recibida y no permita la predicción/entrenamiento del modelo si encuentra problemas o anomalías con los datos recibidos. Para implementar dichas restricciones, en nuestra implementación del programa decidimos de usar la librería de Python llamada Pydantic para el análisis de las peticiones recibidas. En general, esta librería sirve para

validación de datos y confirmación de la entrada. Al crear una instancia de una clase de esta librería, se permite verificar los tipos de datos recibidos y de igual manera, confirmar reglas adicionales que se le agreguen a dicha clase con forma de “validators”. De esta manera, se pueden capturar casi todos los errores e incoherencias causados por la entrada del usuario.

Adicionalmente, se sugiere (aunque no está restringido) el entrenamiento del modelo con suficiente volumen de datos, para que de esta manera los resultados entregados por el API sean coherentes con lo pedido por el negocio.