# Application Setup Model Training and Prediction

## Application Setup and Running the Api

Download/clone the code from github using the link <https://github.com/RaghunathChandran/BertSentimentAnalysis.git>

Create a new conda environment with python version 3.8.5

Once the environment is created and install the packages using the below command.

>pip install -r requirements.txt

Make sure you set the folder SentimentAnalysis as your working directory.

## Model Training and Prediction

Set the folder SentimentAnalysis as your working directory

##### Training Script:

Run the script train.py from the working directory using the command

> python train.py

The model will be trained and the saved to the folder -SentimentAnalysis\saved\_models

##### Predict Script:

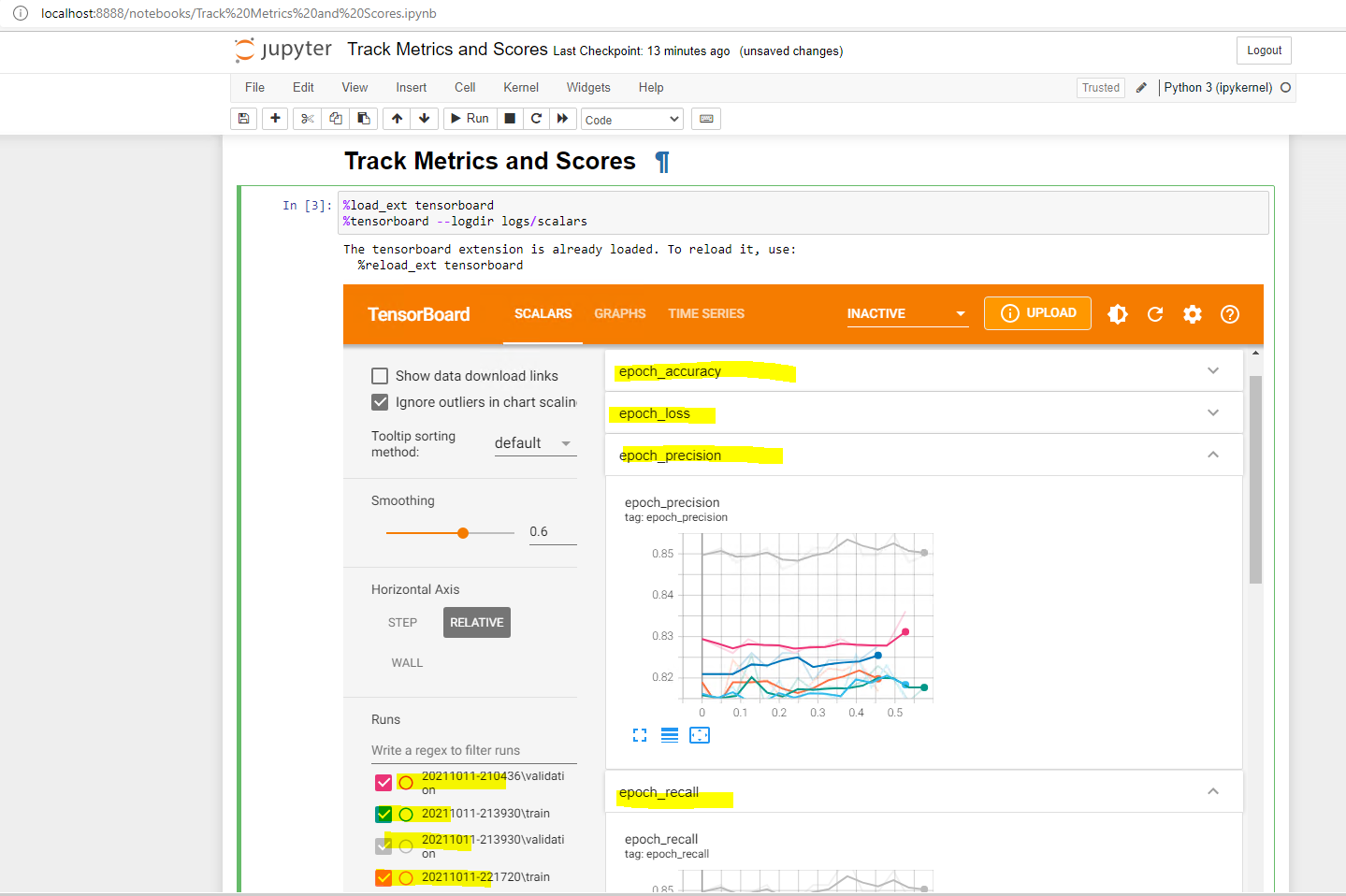
Run the script predict.py from the working directory using the command

> python predict.py

The model predicts the sentiments for the file “Entity\_sentiment\_testV2” in the “***data***” folder and generates an excel file with the prediction in the folder “***results***”.

## Tracking Metrics using TensorBoard.

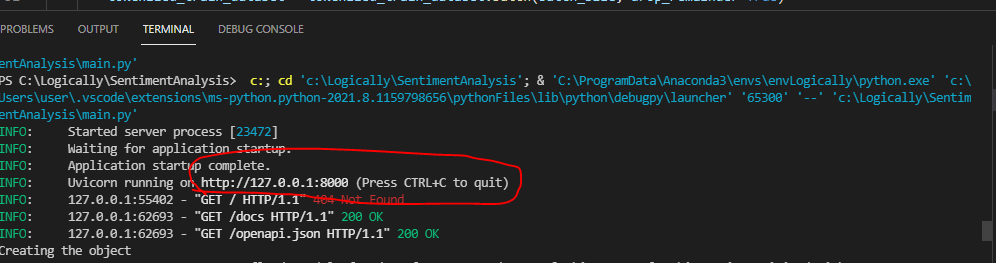
To check the metrics open the Jupyter notebook file “**Track Metrics and Scores.ipynb**” in the working folder “SentimentAnalysis” and run the cells to see all the metrics. See screenshot below.



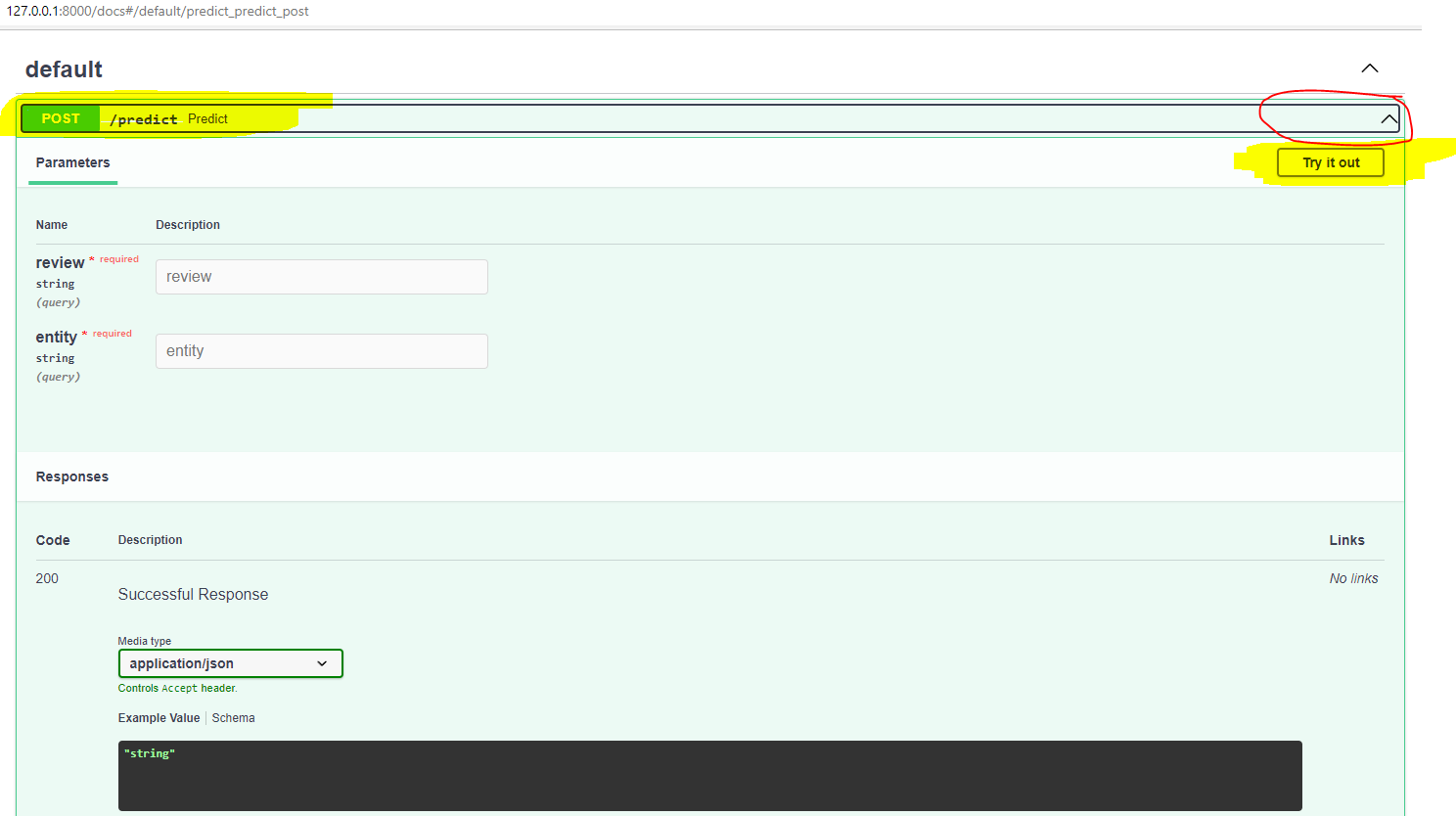
Run the FastApi application using the command.

>python main.py

click the url <http://127.0.0.1:8000> to open the api in the browser.

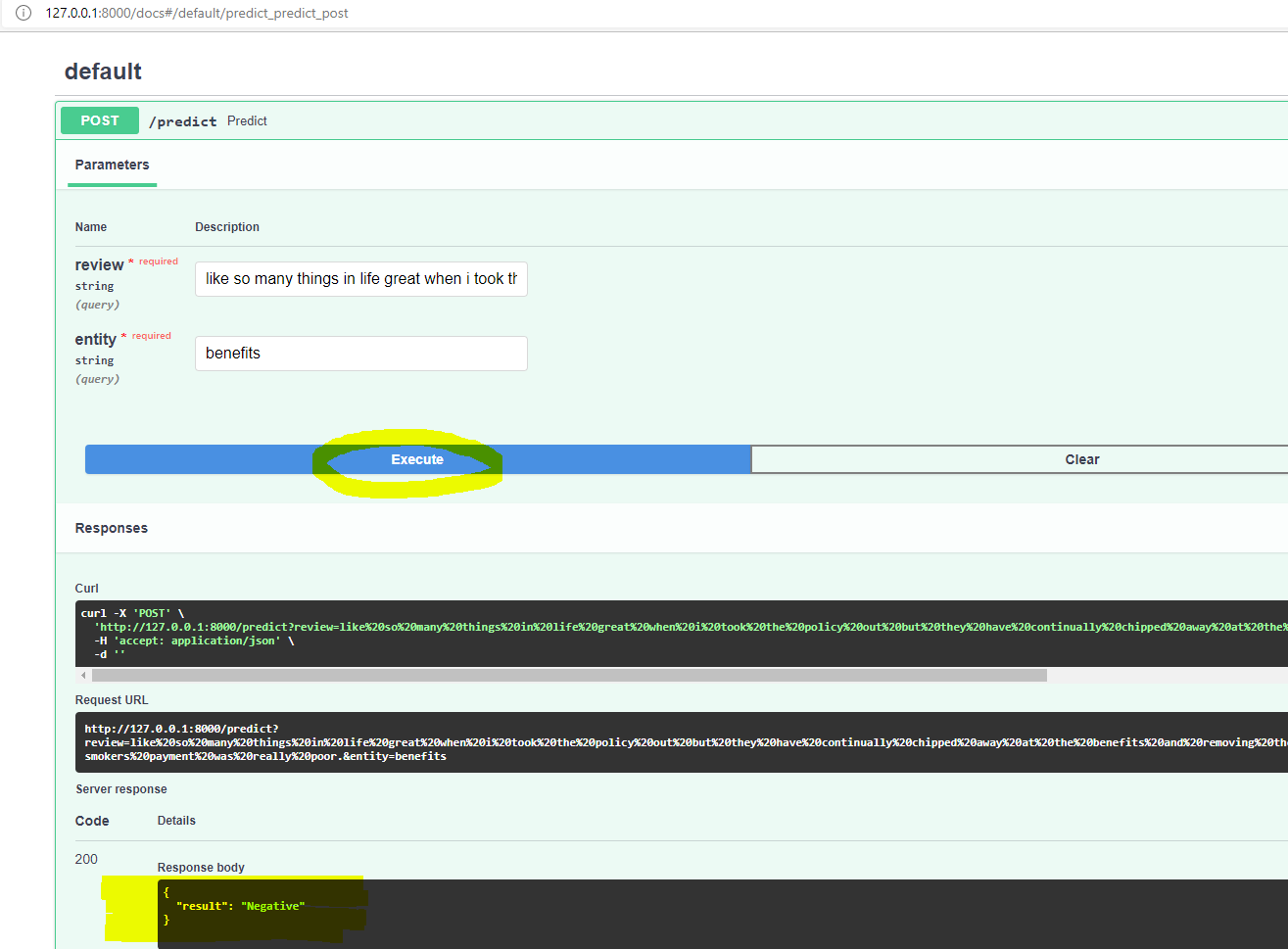


To check the model for a single review and entity, open the OpenApi UI in FastApi using the url <http://127.0.0.1:8000/docs> to see the below screen. (important to type /docs after port 8000)



Expand the tab “POST /predict” and click the button “Try it out” to enable the text box for review and entity and should see the button “Execute”. Enter proper text in the text box and click the button to see the result in the “Response Body” area under “Responses” see screen shot below.

**NOTE:** The first response will be slow, then the performance improves for subsequent calls.



Improvization in Code and Architecture

1. Create a better pipeline to embed “Sentence” and “Entity” separately.
2. Error Logging to be implemented.
3. Create TF dataset objects and load them for training for better utilization of GPU.