How do I use plumber to deploy R model

Asked 5 months ago Modified 5 months ago Viewed 52 times

0

I want to deploy R model that help to predict patients drug_administration using plumber. The scripts runs on its own effortlessly.
I have a local server in 4 facilities that I want to use plumber to deploy the model.
The saved rds script file **model_patients.rds** is meant to fetch data directly from the local server, then use plumber to predict the model.

```
library(plumber)
model <- readr::read_rds("model_patients.rds")
#* @apiTitle Patient Prediction Analysis
#* @get /model
function(predicted,model){
  model <- glm(drug_adminstered~.,family = "binomial", data = train)
   predicted <- predict(model, test, type="response")
  predicted
}
```

r plumber

Share Edit Follow Flag

edited Feb 24 at 21:08

r2evans
112k 6 72 123

asked Feb 24 at 21:01

francism
1

This looks like a reasonable plumber endpoint, what's the question? Is it how to host `plumber` in general? – r2evans Feb 24 at 21:09

The problem is that the R model on its own can run effortlessly and get the predictions done but cannot produce any output using plumber. Is there anything that i am not getting right. could be my function. – francism Feb 24 at 21:15

What output have you tried to generate? What is required? Why is `predicted` here insufficient? – r2evans Feb 24 at 21:20

> head(predicted, n=10) 1 2 3 4 5 6 7 8 9 10 0.7001394 0.9631952 0.9676103 0.7045982 0.7005794 0.9579745 0.9654531 0.6856937 0.6983624 0.9311313 – francism Feb 24 at 21:29

That does not address my question, but look at my answer and see if it gives you a good start. Your question is missing a lot of detail, so it's very difficult to give you something actionable. – r2evans Feb 24 at 21:41

1 Answer

Sorted by: Highest score (default)

0

It looks as if you copied code from an R console with many other variables defined and didn't adapt it to an austere/fresh environment.

I see a few problems:

1. You define `model` in the main environment as the contents of the `.rds` file,
   ... *and* you define it as the second argument of the endpoint function,
   ... *and* you immediately overwrite it with the return value from `glm(..)`.
   It makes sense to me that it would be defined by the `.rds` file, which means it should not be a function argument and it should not be overwritten by the call to `glm`.
2. You call `glm(.., data=train)` but you do not define `train`. Perhaps this should be `data=model`? Similarly, you use `test` but never defined it.
3. Wouldn't the model already be defined? Why do you need to call `glm(..)`, instead just using `predict(..)`? Perhaps I'm missing something in the structure of your model-deployment.

A complete guess, but try this:

```
library(plumber)
model <- readr::read_rds("model_patients.rds")
#* @apiTitle Patient Prediction Analysis
#* @get /model
function(predicted){
  # model <- glm(drug_adminstered~.,family = "binomial", data = train)
  predict(model, predicted, type="response")
}
```

Share Edit Follow Flag

answered Feb 24 at 21:41

r2evans

112k 6 72 123

Thank you so much for your support so far. This looks different from my own code and can only show predicted. However, in the provided space for predicted, the model is suppose to pick a new file from server and then perform prediction on it. I tried that but got some errors – francism Feb 24 at 21:53

{ "error": "500 - Internal server error", "message": "Error in model.frame.default(Terms, newdata, na.action = na.action, xlev = object$xlevels): 'data' must be a data.frame, environment, or list\n" } – francism Feb 24 at 22:00

Please, how do I provide input for the predicted. I specify the path to the new file but this throws up errors which was what I posted above. Once again , thank you. – francism Feb 24 at 22:05

Since you had gotten that far, I assumed you knew how to transfer the `predicted` object to your plumber endpoint. I don't understand *"the model is suppose to pick a new file"*, it makes no sense to me. The question of "how to pass a frame as input" depends wholly on what you're using to *call* this endpoint. Are you using R and `httr::GET` ? Are you using a shiny interface? Is it agnostic and can be called from any prog-lang? How are you intending this to be used? What does a model look like? What are these other "files" you referenced? Too much unknown. – r2evans Feb 24 at 22:13

If `predicted` is supposed to be a frame, then the *caller* needs to pass it likely as `application/json` or similar, and you may need to use `jsonlite::parse_json` on the received data to transform it to a `data.frame` . – r2evans Feb 24 at 22:33