Derek Graves

08/01/2024

Lab 6: Deploying a Model

* Progress 1 - Screen shot of pipeline model results

A screenshot of a computer program

Description automatically generated

* Analysis 2 - Compare linear with pipeline performance
  + The pipeline regression model seems to perform better than the linear regression model on both training and test data. It has a lower RMSE and higher R2 scores, which indicates a better fit and better generalization to new data. We can see this in the graph for our pipeline model as well, in that it better captures the non-linear relationships in the data that the simple model can’t. The pipeline model with polynomial features significantly improves performance and is the better choice for predicting weight based on height in this data set.
* Progress 3 - Screen shot of scatter plot with pipeline model curve overlay

A screen shot of a graph

Description automatically generated

* Progress 4 - Screen shot of set of predicted value for pipeline model

A computer screen shot of a program

Description automatically generated

* Analysis 5 - Discussion on deciding input limits for pipeline model
  + There are a couple of considerations I would have when determining input limits for the pipeline model:
    - The model was trained on height data ranging from approximately 60 cm to 180 cm. Keeping inputs within this range is more likely to provide accurate predictions since the model has learned from data in this range. This is particularly important when working with polynomial models, especially at higher degrees, as they can have more erratic predictions outside the training data range.
    - We also want our predictions to be reasonable. At the extremes of our data range for height, we should ensure we are getting weight predictions that are consistent with what we would normally observe for those given heights. Predictions that are significantly off from expected values could indicate that the input is outside the valid range or that the model is overfitting poorly.
* Progress 6 - Screen shot directory listing with saved model

A screenshot of a computer

Description automatically generated

* Progress 7 - Screen shot predicted values for restored pipeline model. Verify match.

A screenshot of a computer program

Description automatically generated

* Progress 8 - Screen shot of flask installation

A screen shot of a computer

Description automatically generated

* Progress 9 - Screen shot of server start

A black screen with text

Description automatically generated

* Progress 10 - List of server responses from browser or CURL
* [**http://127.0.0.1:5000/weight**](http://127.0.0.1:5000/weight)
  + Response: Bad Request - Missing height query parameter
* [**http://127.0.0.1:5000/weight?height=33.3**](http://127.0.0.1:5000/weight?height=33.3)
  + Response: {"height": 33.3, "weight": -25.8606098982665}
* [**http://127.0.0.1:5000/weight?height=0**](http://127.0.0.1:5000/weight?height=0)
  + Response: {"height": 0.0, "weight": -166.29034127566825}
* [**http://127.0.0.1:5000/weight?height=50**](http://127.0.0.1:5000/weight?height=50)
  + Response: {"height": 50.0, "weight": -1.3934699237888282}
* [**http://127.0.0.1:5000/weight?height=100**](http://127.0.0.1:5000/weight?height=100)
  + Response: {"height": 100.0, "weight": 13.326036231749782}
* [**http://127.0.0.1:5000/weight?height=150**](http://127.0.0.1:5000/weight?height=150)
  + Response: {"height": 150.0, "weight": 41.40189303869}
* [**http://127.0.0.1:5000/weight?height=175**](http://127.0.0.1:5000/weight?height=175)
  + Response: {"height": 175.0, "weight": 56.42986106265152}
* [**http://127.0.0.1:5000/weight?height=200**](http://127.0.0.1:5000/weight?height=200)
  + Response: {"height": 200.0, "weight": 44.77557652322908}
* NOTE: Include a screenshot of your app running in the browser showing the address bar in the image.

A screenshot of a computer

Description automatically generated

* Analysis 11 - Verify match. Explain discrepancies.
  + When reviewing the server responses and the initial Jupyter notebook predictions, we can see the values match up exactly. We can confirm that our pipeline was serialized and deserialized correctly and our Flask API is functioning correctly, providing errors when no valid input is given, and accurate predictions based on valid height inputs. By first verifying that our model is working correctly, followed up by testing our server responses against our known model data, we have demonstrated that the model can be used for predictions with a web application.