Web app link:
https://share.streamlit.io/avangarde2225/seoul_bike_count_prediction/Seoul_Bike_Prediction_Model.py

Test Folder:
https://github.com/Avangarde2225/Seoul_Bike_Demand_Prediction_Linear_Regression/tree/master/Test

You need tests for:

- Validating input data.
- Validating feature engineering.
- Validating quality of new model versions.
- Validating serving infrastructure.
- Testing integration between pipeline components.

QA process test

A requirement it's a need, functionality or characteristic of a system.

Requirements:

- All numeric features are scaled, for example, between 0 and 1.
- One-hot encoded vectors only contain a single 1 and N-1 zeroes.
- Missing data is replaced by mean or default values.
- Outliers are handled, such as by scaling or clipping.

QA test scenario writing

Although there are different approaches to testing, I have followed the approach that suggests separating tests pre-training and post-training of the model. "Pre-train tests—which can be run without trained parameters—check if our written logic is correct. For example, is classification probability between 0 to 1? Post-train tests check if the learned logic is expected."

Pre-train tests:

Negative Test Case :
Test Scenario: Sending incorrect number of features and getting incorrect result.
Link in the repo:
https://github.com/Avangarde2225/Seoul_Bike_Demand_Prediction_Linear_Regression/blob/master/Test/test.py

Post-train tests

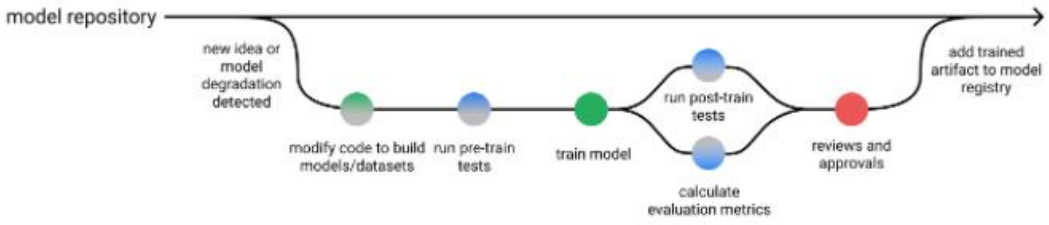
- API test scenario for the deployed code to UI
Test Scenario : Simple API response check
Link in repo:
https://github.com/Avangarde2225/Seoul_Bike_Demand_Prediction_Linear_Regression/blob/master/Test/test_deployment_api.py

- Checking the mse and mae boundaries for the model using `drifter_ml library`
Link in repo:
https://github.com/Avangarde2225/Seoul_Bike_Demand_Prediction_Linear_Regression/blob/master/Test/simple_test.py

- Testing the model with the random data
Link in repo:
https://github.com/Avangarde2225/Seoul_Bike_Demand_Prediction_Linear_Regression/blob/master/Test/test_linear_model.py

Automation architecture

Despite different models suggested by the ML scholars I have found the below diagram most intriguing in terms of the approach in testing.



A proposed workflow for developing high-quality models.

Resource: https://www.jeremyjordan.me/testing-ml/