Testing Document

Air Traffic Analytics Senior Project

1 Testing Overview

Very light testing was done on this project since, in line with team member interests, focus was placed more on development and the smoke testing that accompanied that development. As such, the system is not fully tested and more robust testing would be a focus if this project were to continue on for longer to ensure that all functions are working as intended.

2 Testing

2.1 Smoke Testing:

-> This type of testing was used along with development to verify that expected outcomes code were being produced, in a pass/fail way. Front end development for this project lends itself to this type of testing as code can be written and the effects can be seen in a very visual way. In addition to front end, this type of testing was heavily used in many sections of the project.

2.2 Unit Testing:

-> Unit tests were used to increase confidence in expected functionality of certain functions during development but those functions were ultimately not used in the final version of the project.These tests were used somewhat during development of the routing section of the project. Unit tests can be used in instances where, given certain input, the expected output is known.

To incorporate unit testing, users can use Air-Traffic-Analysis/testing/UnitTestSuite.py in the GitHub as an example of testing. Users may add this file to a project, include files you want to test, and create tests accordingly. There should be no need to change anything about the code to be tested and the code will still run like normal except have extra test report at end. The test output is listed by test function alphabetically. If one test fails, the rest will still execute. Reference for unittest can be found at `https://docs.python.org/2/library/unittest.html.`

2.3 Benchmarking:

-> This type of test was primarily used in the machine learning part of the project to inspect performance of different classifiers in relation to each other. These benchmarking tests were less about verifying accuracy of written code and more about the accuracy of the code functionally.

The testing information that follows is also included in the white paper.

The following table shows results of training and testing the MLP used in this project over each month in 2015, using the whole month of data as the training set for each model. Features tested: Month, Day of the Week, Listed Departure Time, Distance. Label tested for: Airtime of Flight. Rounding: 20 minute intervals. Possible MLP configurations for each month (Alpha Value, Hidden Layer composition): [(0.005, (500, 100)), (0.1, (500, 100)), (0.005, 2000), (0.01, 500), (0.01, 2000)]. Rounding was set to 20 minute intervals.

Month	Alpha Value	Hidden Layer Composition	Accuracy (%)
September	0.1	(500, 100)	70.16152
September	0.005	(500, 100)	69.97355
August	0.01	2000	69.66886
September	0.01	500	69.55801
June	0.01	500	69.47578
August	0.005	2000	69.40587
July	0.005	(500, 100)	69.36842
May	0.005	(500, 100)	69.34699
July	0.005	2000	69.29422
July	0.1	(500, 100)	69.14921
July	0.01	2000	69.08648
June	0.005	(500, 100)	68.99374
August	0.01	500	68.96913
September	0.01	2000	68.90079
June	0.01	2000	68.87441
June	0.005	2000	68.8039
July	0.01	500	68.76273
August	0.1	(500, 100)	68.73927
May	0.1	(500, 100)	68.60717
August	0.005	(500, 100)	68.45397
September	0.005	2000	68.15564
May	0.01	500	67.60155
June	0.1	(500, 100)	66.43841
May	0.005	2000	65.24513
October	0.01	500	65.05112
October	0.005	2000	64.8637

October	0.005	(500, 100)	64.71659
October	0.01	2000	64.70786
October	0.1	(500, 100)	64.41834
May	0.01	2000	63.39052
November	0.01	2000	58.32167
March	0.1	(500, 100)	58.29203
December	0.1	(500, 100)	58.16665
April	0.005	(500, 100)	58.1619
December	0.01	500	58.10958
November	0.005	2000	58.10833
November	0.1	(500, 100)	58.10563
November	0.01	500	58.09419
November	0.005	(500, 100)	58.08948
December	0.005	2000	58.04028
December	0.005	(500, 100)	57.98932
April	0.01	500	57.8891
April	0.005	2000	57.86418
April	0.01	2000	57.8406
December	0.01	2000	57.72095
April	0.1	(500, 100)	57.46137
March	0.005	2000	57.45262
January	0.01	2000	56.9988
March	0.005	(500, 100)	56.90118
February	0.005	2000	56.88022
February	0.005	(500, 100)	56.84846
March	0.01	500	56.81267
March	0.01	2000	56.6663
February	0.1	(500, 100)	56.62333
February	0.01	500	56.50548
February	0.01	2000	56.19425
January	0.1	(500, 100)	49.46126
January	0.005	2000	49.29513
January	0.01	500	44.89635
January	0.005	(500, 100)	28.90653

Random guessing accuracy within the max and min of the label in the dataset, averaging over >10 iterations:

Rounding Interval (Minutes)	Accuracy (%)
30	5.0
20	3.3
10	1.7
5	0.82
1	0.17

3 Limitations

In regards to testing this team dealt with lack of experience in regards to testing, a desire to focus on development work as the project is not being used for actual application, and a lack of ability to verify solutions. One major limitation is that the expected functionality of certain functions may not be that which was intended. Though this is the case, this project still provides a base that can be improved and further tested.

4 Suggestions For Further Testing

The following methods for testing would be useful for validation of this project:

- User testing for front end performance
- Concentration on testing for the routing section of the code is suggested due to multitude of edge cases
- It is suggested that a file similar UnitTestSuite.py is used for any unit testing which is, as discussed above, located at Air-Traffic-Analysis/testing/UnitTestSuite.py in the GitHub repository
- Speed Testing to determine the expected performance given software implementation and the user's particular hardware configuration
- Load Testing to determine the limits the system can be pushed to given software implementation and the user's particular hardware configuration
- Security Testing to mitigate risk of website vulnerability if hosting this on the web.