TEST PLAN

1. INTRODUCTION

1.1 PURPOSE

This test plan describes the testing approach and overall updation(Or manipulation) of used datasets that will derive the testing of Flight Delay Prediction. The idea is to manipulate given different datasets so that our algorithm can work properly on those datasets.

The method of manipulation of datasets consists different steps like merging datasets, deleting different columns, updating columns and rows etc. Our testing approach also take care of handling the nan values in datasets that can affect accuracy of our algorithm.

1.2 PROJECT OVERVIEW

Our Project analyses the given dataset which consists the information of flights as well as the weather condition at that point of time. After analysing the datasets the trained algorithm will be able to predict (Or classify) the flight delay.

2. TEST STRATEGY

2.1 Test Objectives

The objective of the test is to verify that the our merged dataset is perfect i.e should not contain any nan values, should not have any useless columns etc.
The final Product of the test is:

1.A perfect Dataset without conating any vague values.
2. Ready to be used to train our algorithm.

2.2 Assumptions

Certain assumption are made while testing This include :

- The data types of the columns are same, before passing this data into the machine learning classifier.

3. Test Cases

The main Test Cases which we use in the testing of our project include :

1. Accuracy testing: Here we test if the accuracy of our model is actually correct. Suppose the accuracy obtained by our model is 70 %. However because of the way we have defined the classes, there might be the case that the number of instances in the test data which predict to a particular class is very high (say 70%).

Now there might be the case that our model is totally biased towards that particular class. This can be checked by generating inputs where we know the output and checking the actual output.

2. Checking for NaN values: The database has to be checked for NaN values before using it to train our model. This can be easily be done by using df.describe() function which will tell the number of correct rows for each column and thus matching it with the actual number of rows.