Making Real-time predictions with H2O on storm

1. Summary

I used the following tutorial to successfully implement a storm topology and make real-time predictions with H2O.

http://learn.h2o.ai/content/tutorials/streaming/storm/index.html

A Gradient Boosting Machine (GBM) model was produced using H2O. The model was emitted as a Java POJO which was embedded in Storm and then used for scoring.

2. Installation and setup of H20 and Storm

- H2O model was built using R. H2O package for R was installed using 'install.packages("h2o")' in R.
- Storm was installed by cloning the storm repository from Github using 'git clone https://github.com/apache/storm.git'

3. Data and Model creation in H20

The modeling problem was a classification problem. The training dataset consisted of 1000 observations with a binary response variable. The response labels were 'dog' and 'cat'. The ratio of dogs to cats is 3:1. The features used for classification were:

- Has4Legs
- CoatColor
- HairLength
- TailLength
- EnjoysPlay
- StairsOutWindow
- HoursSpentNapping
- RespondsToCommands
- EasilyFrightened
- Age
- Noise1
- Noise2
- Noise3
- Noise4
- Noise5

R was used to build a GBM model in H2O. The output consisted of two files: A 'h2o-genmodel.jar' file containing the interface definition, and a 'GBMPojo.java' file containing the Java code for the POJO model.

4. Setting up a simple application in Storm

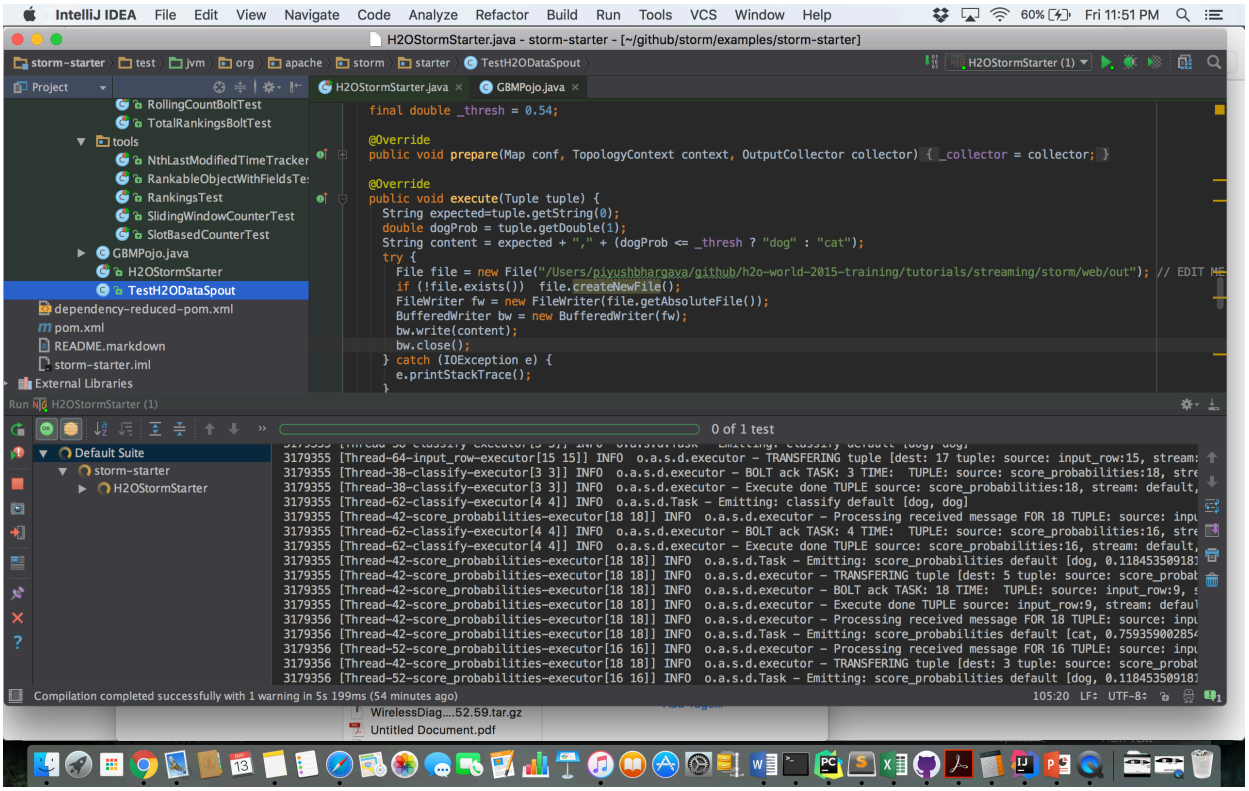
To setup/build a Storm project, Storm jars need to be added to classpath to develop Storm topologies. Apache Maven was used as it was an easier way to accomplish this purpose.

Once Storm was built, 'IntelliJ IDEA' was used to build the H2O streaming topology. The topology has one spout 'TestH2ODataSpout.java' which reads data from the test data file and passes each observation to the prediction bolt one observation at a time. The topology also had two bolts (H2OStormStarter.java): 'a prediction' bolt and a 'classifier' bolt.

A live dataset was used to feed the spout in the Storm topology on which predictions were made. The parallelism was introduced for spout by using 10 executors and for bolts by using 3 executors each.

5. Integration of H2O with Storm environment

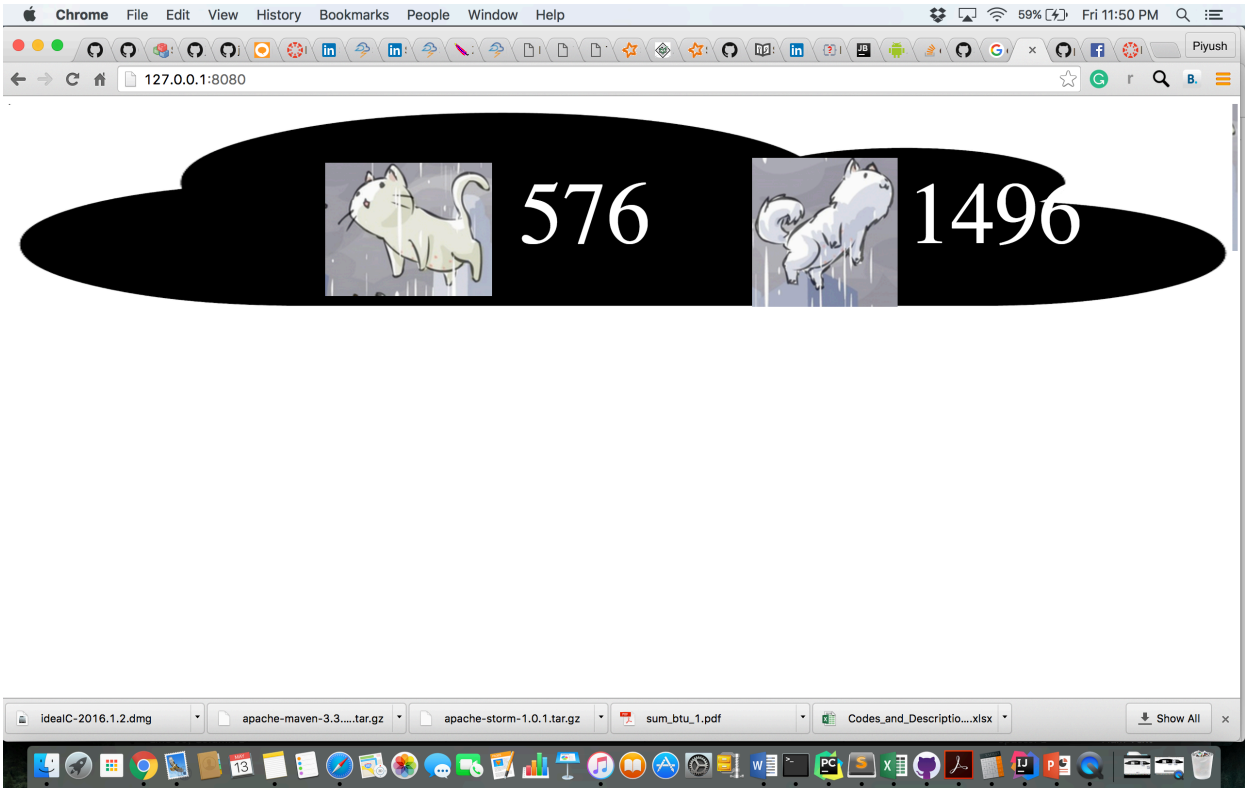
The h2o-genmodel.jar file (interface definition) and the GBMPojo.java file (model) from H2O model creation were added to the Storm environment using IntelliJ. The screenshot below shows the implementation of storm topology using IntelliJ



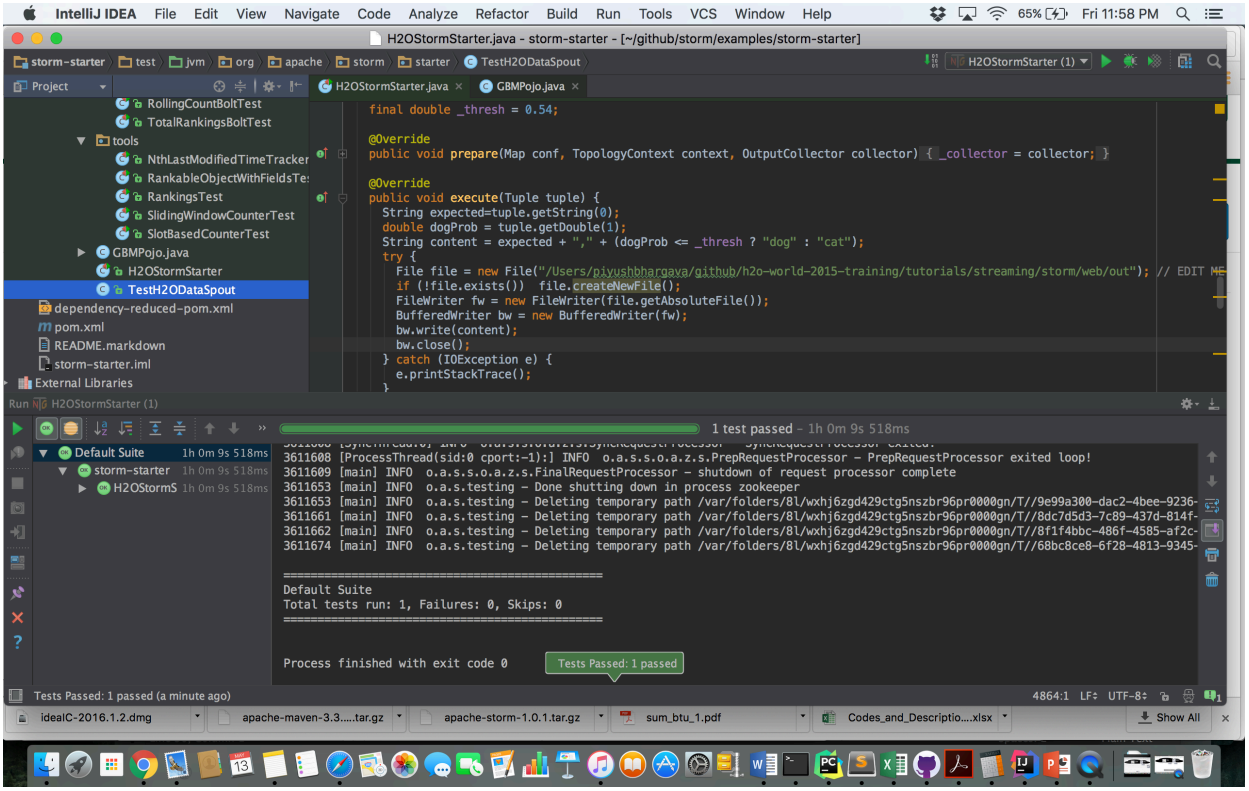
6. Results:

After integration, the Storm topology was run with H2O model deployed. The ‘prediction’ bolt emitted the probability of a test observation being a 'dog'. The ‘classifier’ bolt was then used to decide whether the observation is of type 'cat' or 'dog' depending on a threshold of probability. The threshold was chosen using AUC curve.

The predictions could be watched in real time on ‘localhost’. The observations were passed to the prediction bolt one observation at a time which simulated the arrival of data in real-time. A screenshot is shown below:



A screenshot showing the successful run (Test passed) is shown below. The parallelization was also successful.



A video displaying the execution of the model and real-time predictions is posted on the following link. Important files are also posted
https://github.com/piyushbhargava7/Predictions-with-H2O-on-Storm

7. Challenges:

As the tutorial is slightly dated, some modifications had to be made to the code. For e.g. Migrating the APIs to org.apache.storm. Though the changes required were minor, lack of understanding of Java and relative unfamiliarity with Storm made the errors time consuming.