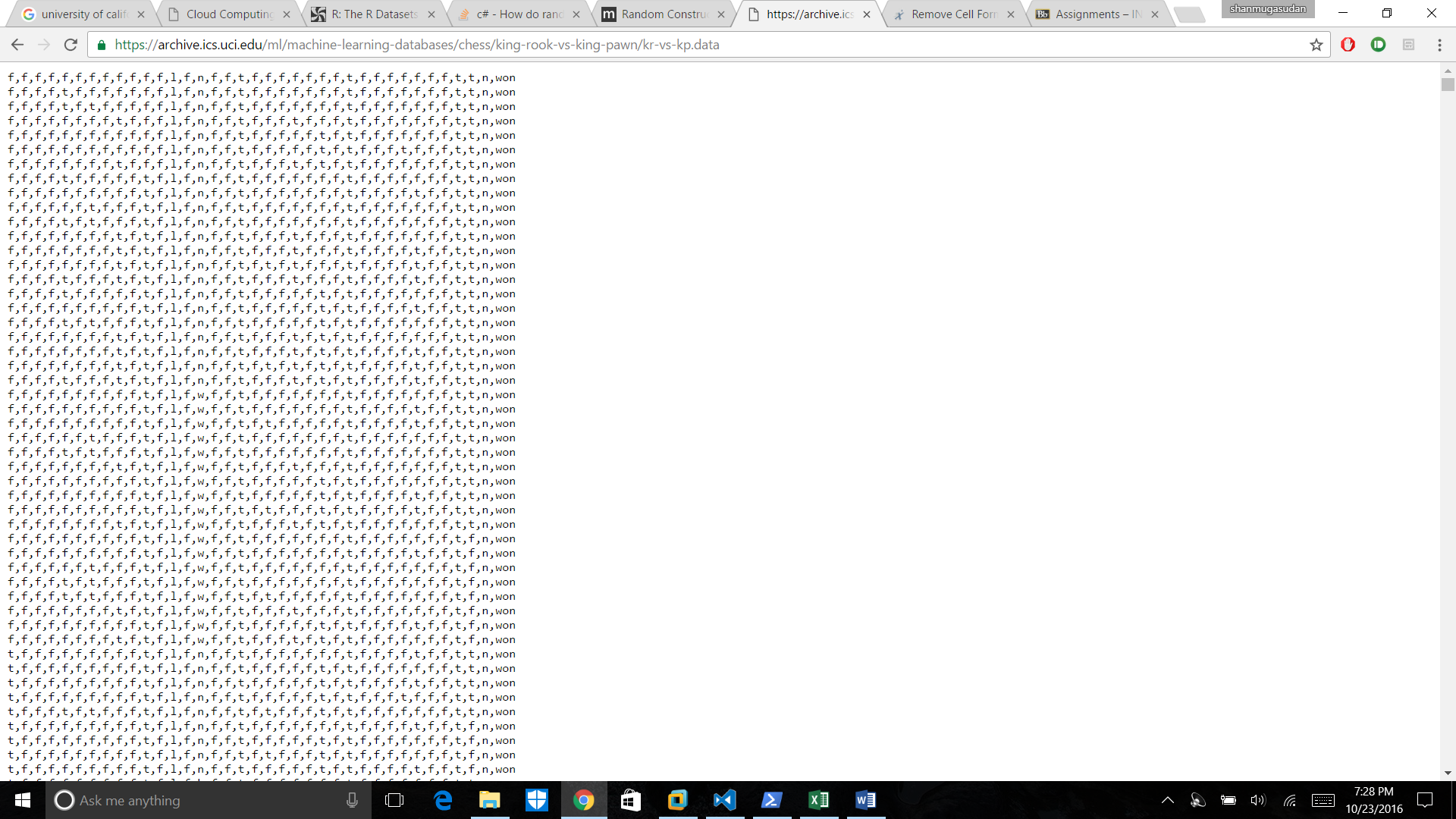
**Winnow Algorithm Problem:**

My Dataset is a **Chess (King-Rook vs. King-Pawn) Data Set** available at ***University of California, Irvine machine learning repository.***

Please find the [link](https://archive.ics.uci.edu/ml/datasets/Chess+%28King-Rook+vs.+King-Pawn%29) for the same here.

**Data set Screenshot:**

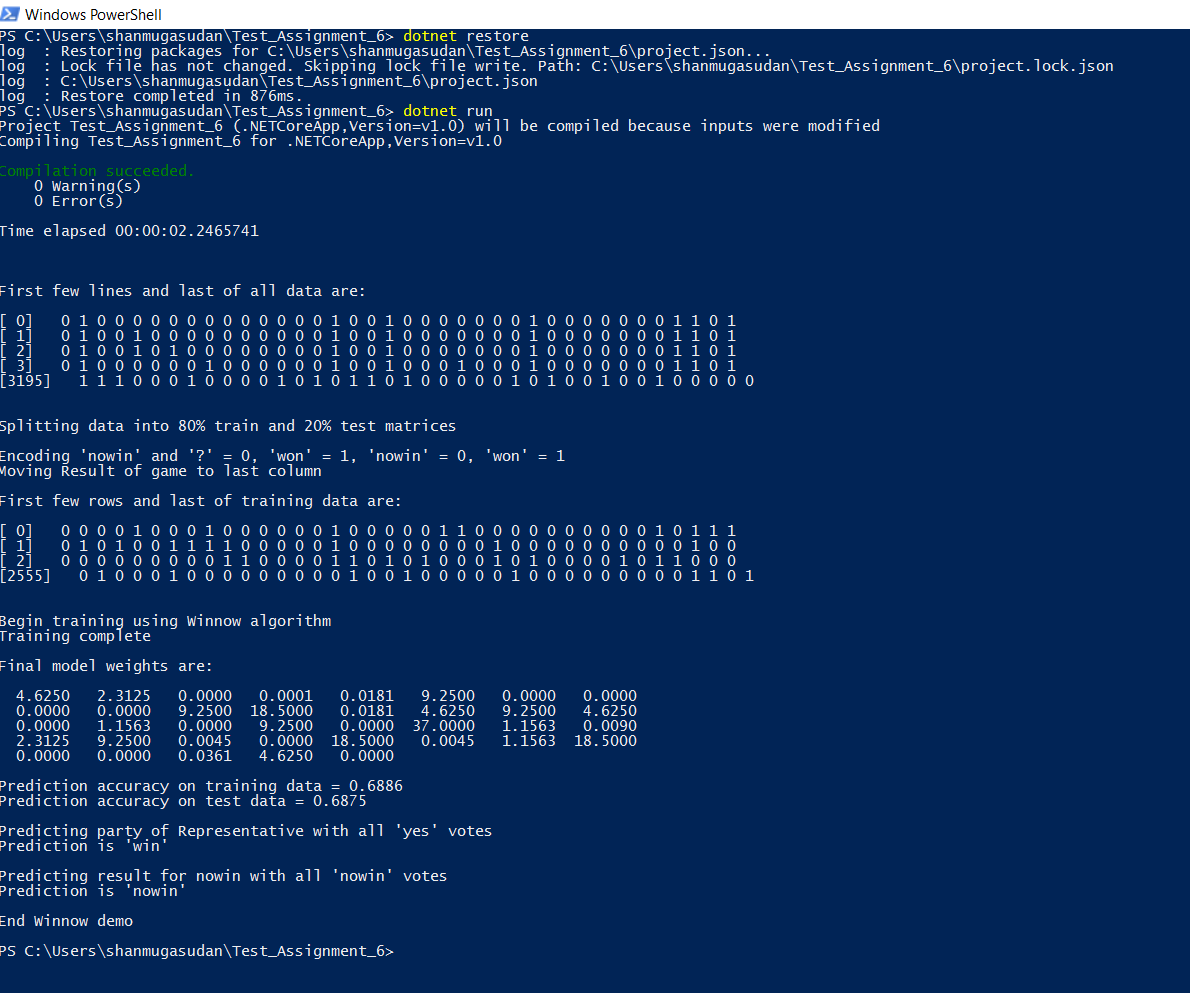


**Corrections made in the code:**

**Integrated the code provided by professor and used the chess dataset for my problem.**

1. My Dataset contains 36 attributes in total and predicts who wins a chess game.
2. One of the attribute values have got three possibilities (value: b, t, n) and so I have chosen binary value for them as 0,0 0,1 1,0 respectively.
3. Consequently, my total input values have to be increased by 1 and so it is 37.
4. A Total of 3195 data sets are considered for input and predicts the results with an accuracy of 68.7%.

**Screenshot of Output:**



Question: How to improve Winnow Algorithm:

Answer:

1. Winnow algorithm works on the principle of upper bounding the number of mistakes. This means that, it learns from the mistake to approach the true value, in this process it modifies the weights considered for computing the predictions.
2. Introduce a new limit by duplicating the upper and lower limit values of independent variables.
3. This gives a broader window for consideration and now applying *Online Learning Algorithm* helps in improvement.
4. Ultimately, applying Widrow-Hoff algorithm gives significant improvement in reducing the number of mistakes and probability of making mistakes.

Read this information from a [link](https://www.cs.princeton.edu/courses/archive/spring14/cos511/scribe_notes/0408.pdf) online.

Question: What is the dataset size after which the algorithm become unwieldly?

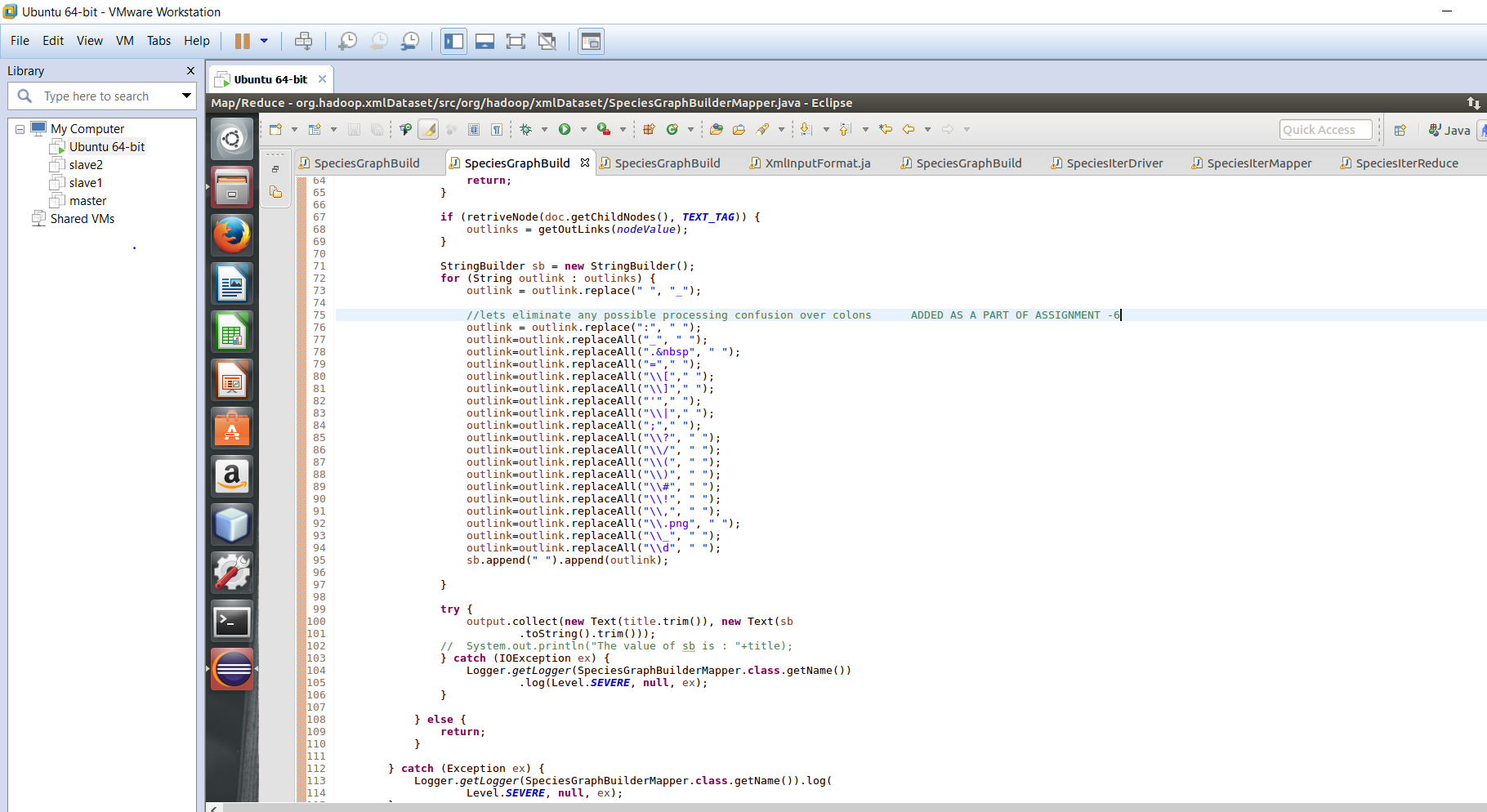
Answer: If the input dataset contains a lot of non-linear data in it, it gives a lot of outliers and causes winnow algorithm to lose its significance. Since Winnow is a linear classification algorithm, the dataset considered for selection should also be linear in consideration. Found the details from this [Linear Algorithms-Perceptron](https://en.wikipedia.org/wiki/Perceptron) link.

XML Species Dataset:

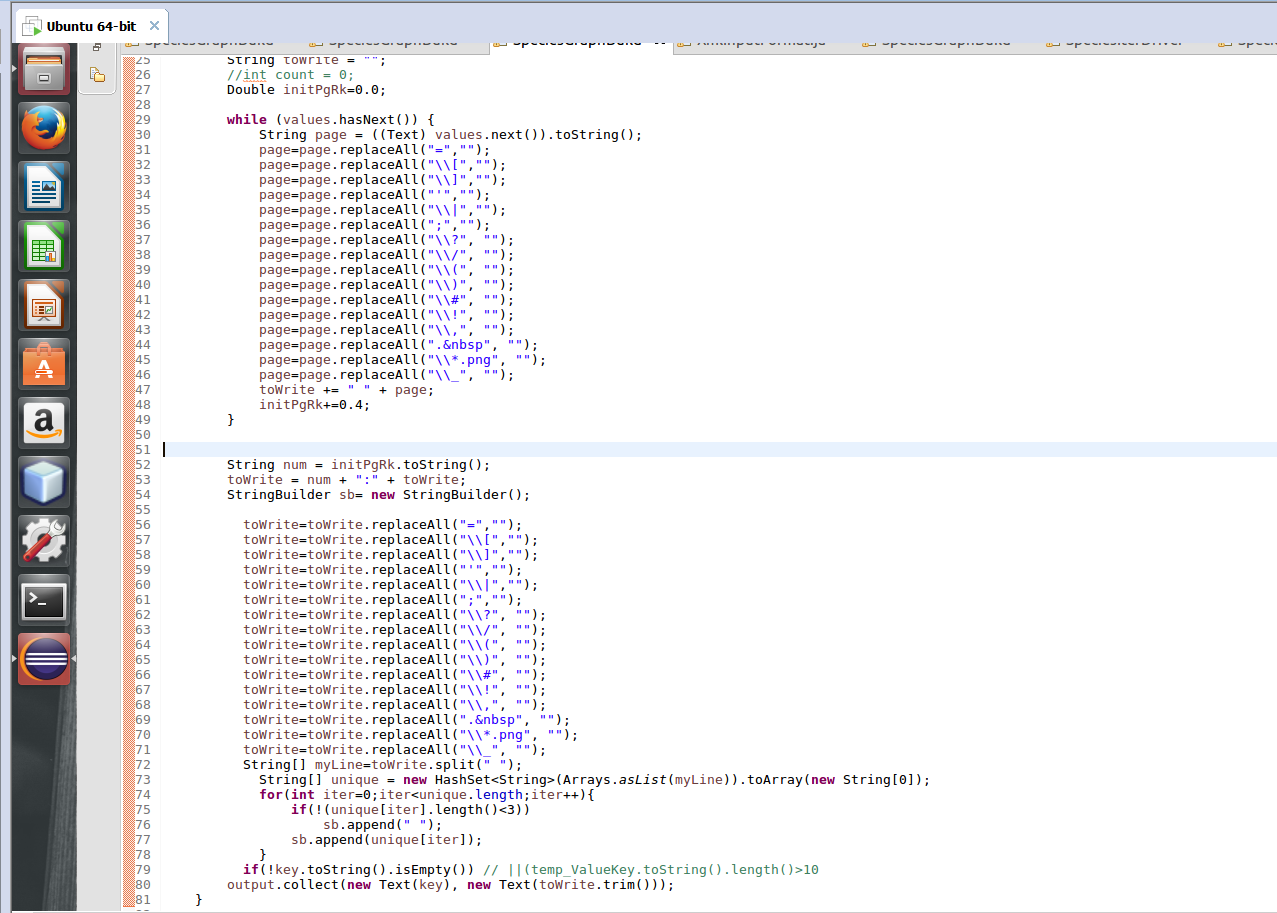
The Wiki Species data is given in XML format. We have been provided with XML parsing code.

The parsed code contained junk values. Hence the following

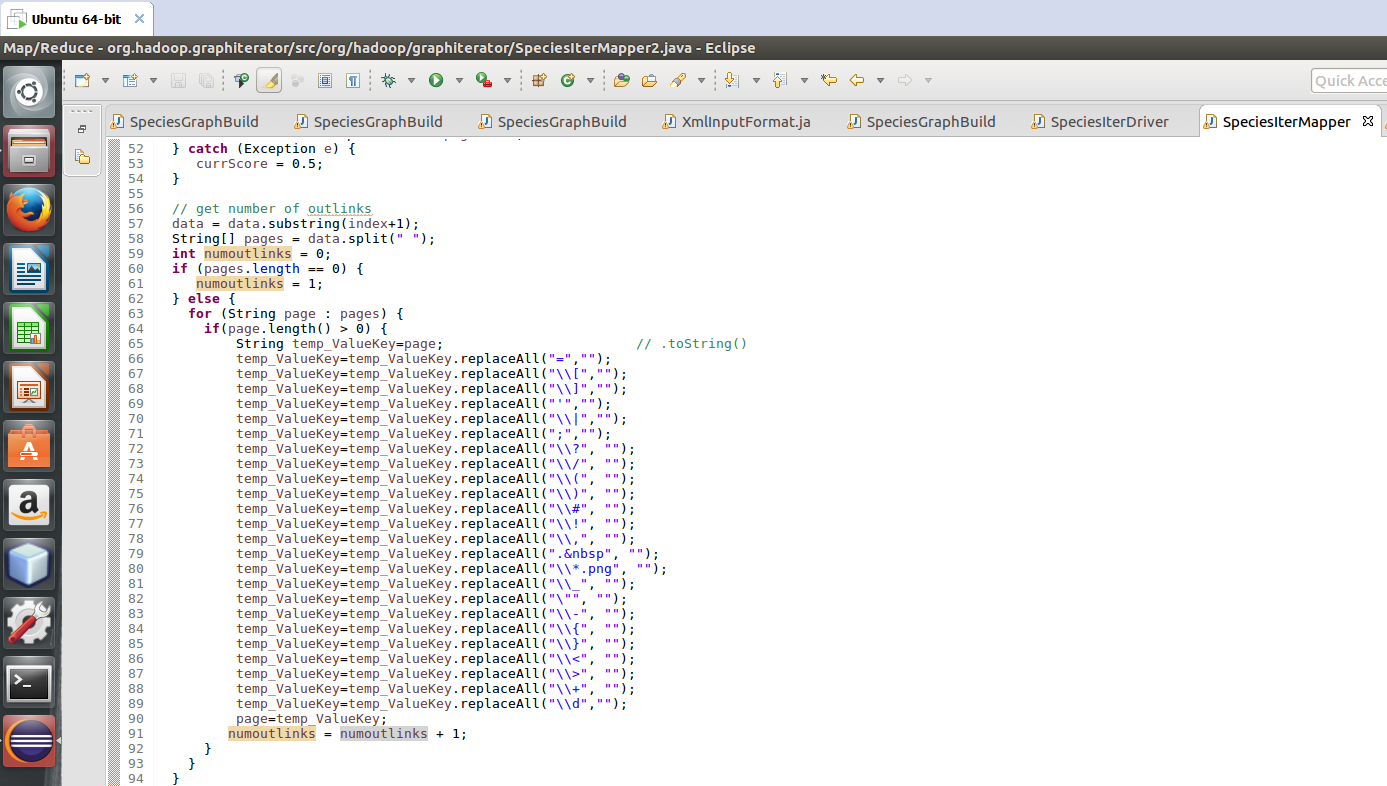
Builder Mapper:



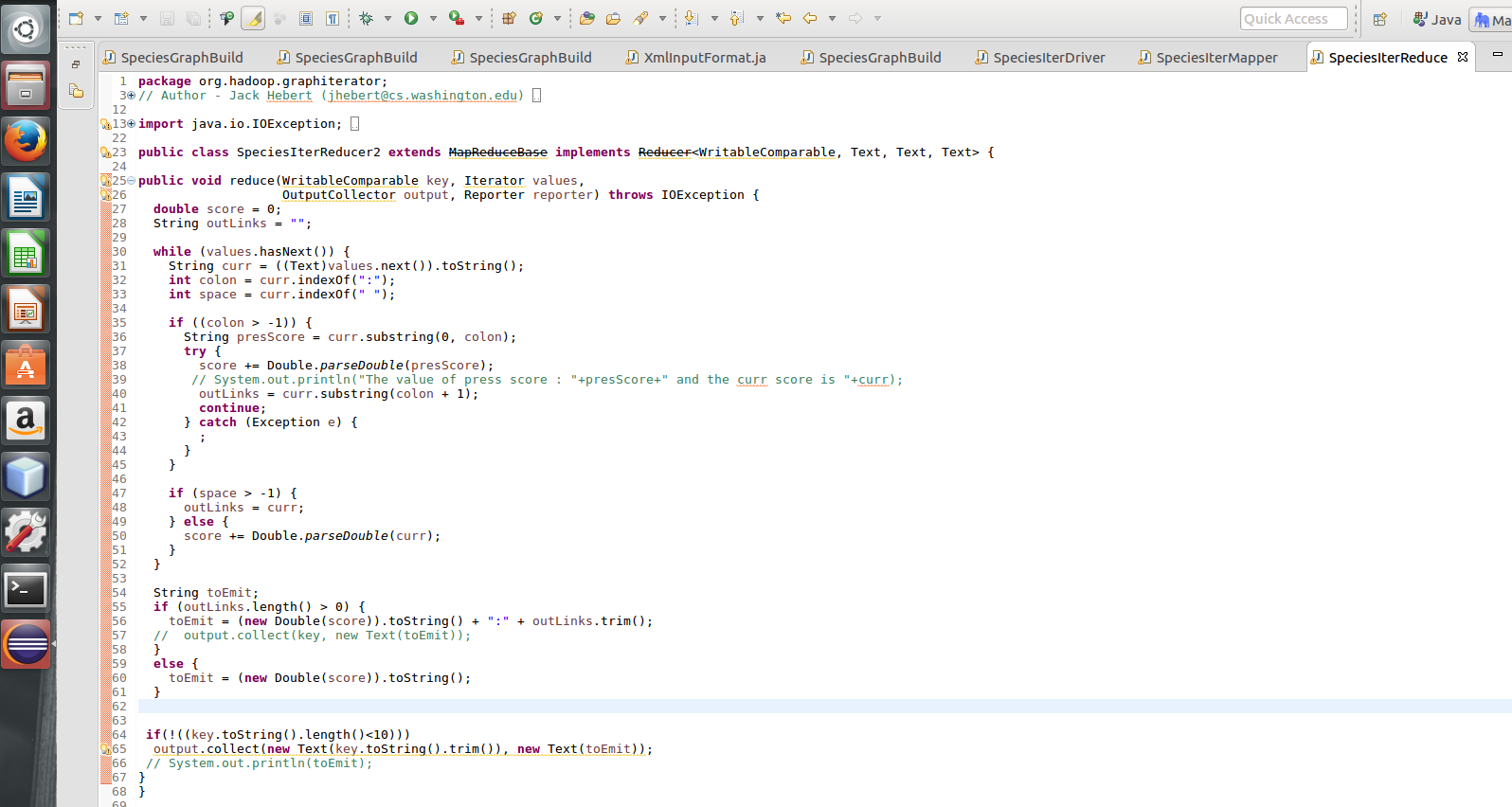
Builder Reducer: I was getting a duplicates in values and so decided to use a hash set to filter the values. That is the reason I have a hash set for values in reducer. Since reducer only performs a group by on key for values, all the results are added to a key without removal of duplicates.



Iterator Mapper:

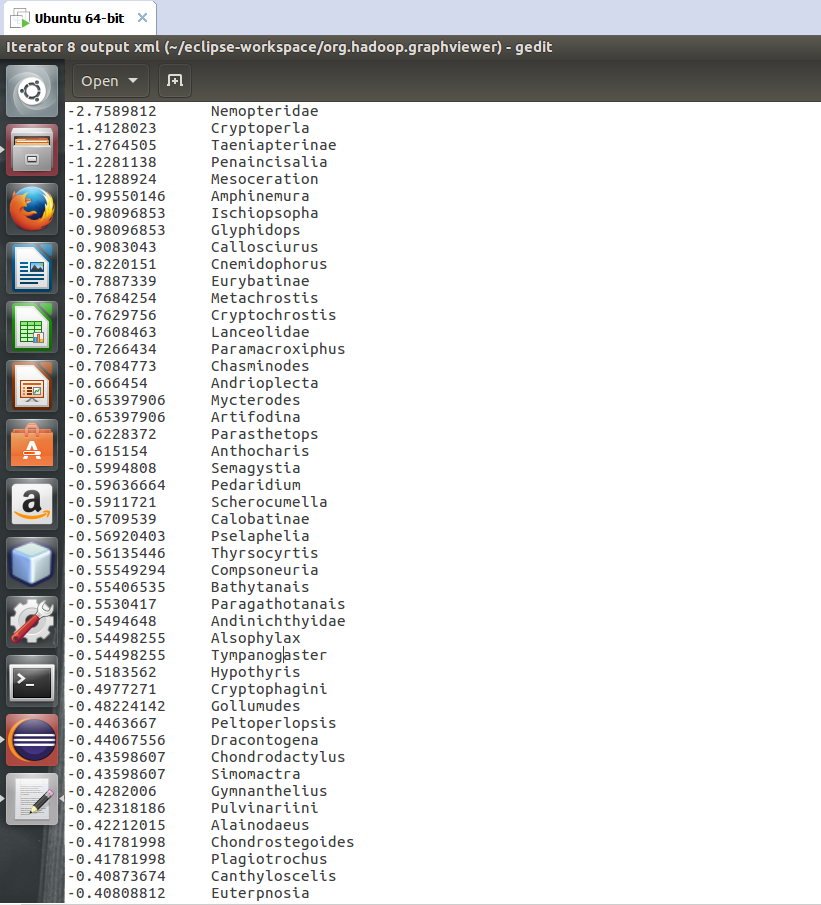


Iterator Reducer: Was having keys with size less than 10.

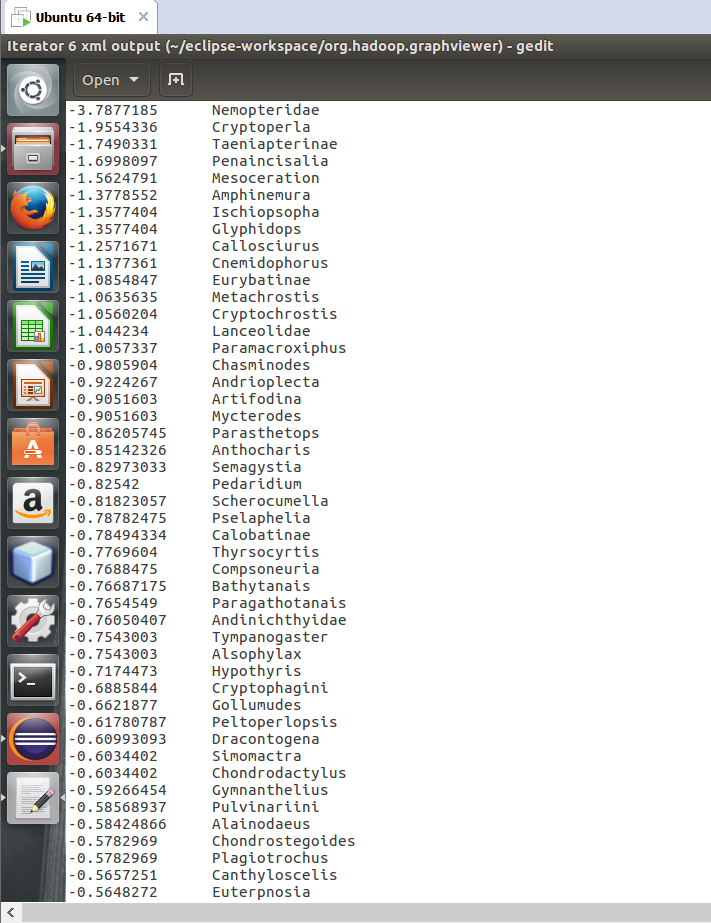


**Output Files:**

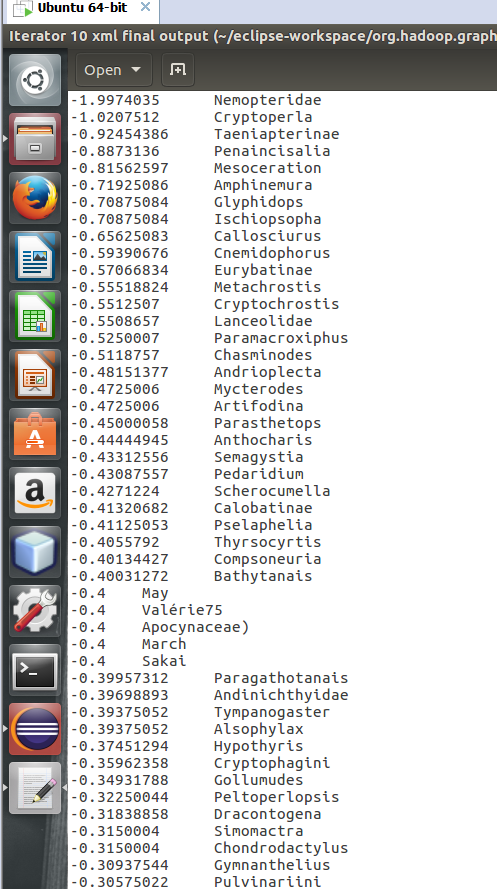
**Iterator 8-XML viewer Output:**



**Iterator -6 XML viewer output:**



**Iterator 10 XML viewer Output:**



**Observation:**

From all these values, I conclude that the most popular species is “**Nemopteridae**”.