Final Project Report

By: Max Barshay, Riley Smith, Rupal Totale

Explanation

In this project we implemented the k-means algorithm. This is an unsupervised machine learning algorithm which means that it works with data when the true labels are unknown. We used a dataset containing 32,561 people and 5 attributes all relating more or less to their income. The 5 numeric columns that we clustered on were age, years of education, capital gain, capital loss, and number of hours per week they worked. There is also a column in the dataset for whether or not that observation had an income of more than $50k with 1 meaning they had incomes of more than $50k and 0 otherwise. We decided to exclude this >$50k column, and wanted to see if our k-means algorithm could cluster on that attribute.

In order for this to work, there would need to be some relationship between income and the 5 columns that I mentioned earlier. Intuitively, it would make sense that older people who work more hours and are more educated would tend to have higher incomes. In our dataset, there were 24,720 observations who did not have an income of greater than $50k and 7,841 observations who did have an income of greater than $50k. This is not an even split, but I think it is balanced enough for k-means to perform well.

Implementation

Our implementation consisted of several steps. In brief, we

- standardized the data
- used the dataset to generate random centroids with k set to 2 (since there are two groups that people can fall into for our dataset)
- used the randomly generated centroids to assign each observation a cluster id by finding the cluster closest to it (using sum of squared differences)
- generated a new set of centroids using the dataset with cluster ID assigned to each observation
- Finally, while (loop) the old and new centroids were far aparts (distance determined using sum of squared differences approach and compared with epsilon)
    - updated the observations with the cluster closest to them in new centroids
    - set old centroids to new centroids
    - updated new centroids after updating cluster IDs.
- For ease of testing, we returned the cluster ID and observation Id as key value pairs in and RDD.

We decided to implement a function that standardizes the data because each column was on a very different scale. For example, the minimum value for capital gain was 0 and the maximum value was 99,999 while on the other hand for years of education, the minimum was 1 and the maximum was 16. This large discrepancy will cause the distance metric to be dominated by these large values and will pretty much ignore the education column. By standardizing each column, the result is that each column has equal weight in the distance metric.

Results

The results were highly sensitive to the initial centroids, which is a flaw of the k-means algorithm. The results from one iteration had us group 14,235 observations into cluster 0 and 18,326 into cluster 1. Another “flaw” of clustering is that the labels of the clusters, meaning “1” or “0”, are random. Since k-kmeans is an unsupervised machine learning algorithm, it is intended to be used in situations where labels are unknown. It is just supposed to group points that are similar to each other. That is why I put flaw in quotation marks.

In order for me to assess how well our clustering algorithm performed, I can look at the true positives, false positives, false negatives, and true negatives. For one iteration (same iteration as in the previous paragraph) we had 2,570 true positives, 15,756 false positives, 5,271 false negatives, and 8,964 true negatives. There are likely better ways to evaluate clustering performance in the situation where the labels are known, but we did not implement those. I still believe that these 4 numbers give us some information. I will now interpret these numbers a bit. 2,570 true positives means that out of all of our observations 2,570 observations were in cluster 1 and had an income of over $50k. Recall that cluster “1” was an arbitrary decision.

Despite the apparent flaws of these metrics, imagine a situation where all observations fell into either all true positive or true negative or all false positive or false negative. In this case we would have completely separated the classes.While it is difficult for us to evaluate our performance based on these metrics, they still allow us to get a glimpse of what our algorithm is doing.

Challenges

We faced two primary challenges throughout the life of this project. The first challenge was finding a data set of appropriate size that contains information that we can use with our k-means algorithm. We ended up settling with a dataset that was a bit smaller than we had hoped at just 3 megabytes of data. If given more time for this project, we likely would have spent more time searching for a dataset that is larger. We don’t feel that this smaller dataset in any way degrades the results that we found with our algorithm. It just doesn’t highlight the distributed aspect of our functions as much as a larger dataset would, that is to say that an undistributed program could have likely computed a solution on our dataset in the same or less amount of time.

The second challenge that our group faced was standardizing the drastically varying ranges of the numeric columns in our Income dataset. For example, our capital gain and loss categories featured a much larger range of values than most other categories. Capital gain covered anywhere from 0 to 100,000 while capital loss covered 0 to 4,356. In some of our other categories like education, the range only sat between 0 and 16. Because of these differences, we decided to standardize our data so that our sum of squares wouldn't give a much larger weight to the categories that cover a much larger range. Our data standardization function made sure that each category only held data between 0 and 1 so that our sum of squares would be equally weighted by each variable. Due to time constraints, however, although we implemented this function, we were unable to incorporate it in our algorithm successfully since it would require changes to most other functions. A takeaway from this is that we should have either started earlier or decided on what dataset we would be using before implementing the algorithm so that we could take into account any quirks about the dataset while implementing the algorithm.