# Unsupervised learning: latent variables, expectation maximization and k-means clustering using Spark in a Hadoop environment

Varderes Barsegyan

## Unsupervised Learning & My Research Goals

Machine learning algorithms learn from data and make predictions and inferences based on the knowledge that is gained from that analysis. *Supervised* machine learning algorithms, such as neural networks and support vectors machines, discover knowledge from the data in which each datum has been labeled as belonging to a specific group, state, or class. In contrast, *unsupervised* learning algorithms are commonly employed due to their ability to identify classifications, or find *hidden structures,* without having a previously mapped model on which to base decisions [LeCun].

Unsupervised learning alone is a vast and far-reaching paradigm of machine learning. Many textbooks have been written and many classes have been taught on this category of learning. It would be impossible to discuss the many concepts, theories and algorithms that make up such a beautiful subject. Therefore, this report will cover a few important fundamental concepts in the unsupervised realm. I am hoping that the reader will not only understand these concepts but also apply to them to problems that they may encounter in the future.

I will start by exploring the idea of hidden, or latent, variables. I will then extend this concept by introducing expectation-maximization (EM) and how it can be applied to a Gaussian mixture model (GMM) to cluster unlabeled data. Afterwards I will show that the famous and widely-used k-means clustering algorithm is a special case of EM applied to GMM, where data points are "hard-assigned" to cluster points instead of following a probability distribution of assignment. Finally, I will run a parallel version of k-means clustering using the Spark computing environment. In my case, Spark resides in a Hadoop cluster and runs via the Hadoop YARN resource management system built into the Hadoop cluster. As a final analysis, I will compare the running time of k-means in a Spark environment with k-means in a single-threaded, local environment. I am using the Luteo protein data set generated by molecular dynamics simulations in Folding@Home. It is an 8-dimensional data set with about 1.5 million points.

There are two main reason I am exploring these topics and working on my project:

1. Unsupervised learning algorithms are widely-applied in industry *and* research, yet they need much more exploration. Many questions are left unanswered, and many datasets are left unanalyzed.
2. I am heavily involved in bioinformatics research in Dr. Eric Sorin's research lab at the CSULB's Molecular and Life Sciences Center. In our lab, we simulate the complex processes of proteins associated with Alzheimer's disease. We then analyze the resulting

complex and rich datasets to pinpoint stable energy states of the proteins of interest. This way we can have a deep understanding of proteins and their binding sites. If we can map binding sites to energetically stable states, we can design and deploy disease-inhibiting drugs efficiently and effectively.

## Latent Variables & Latent Variable Models

Latent variable models are used in a wide-range fields such as psychology, physics, bioinformatics, and of course, machine learning. Latent, or unobserved, variables are those that are not directly observed, but are direct or probabilistic mappings of observed variables [Goodfellow]. In other words, observed variables are directly measured (e.g. gravitational waves and survey results) and latent variables are indirect structural or conclusive observations of those measurements (e.g. existence of exoplanets and customer satisfaction levels).

It is important to note that the distinction between independent and dependent variables is not the same as the distinction between observed and latent variables, and it is especially important to isolate these distinctions when examining unsupervised learning algorithms. If data points are not labeled with dependent variables then supervised learning models do not have any values by which to base their decisions. Thus, we are left with the task of finding structure in the data set of interest. One can systemically find latent structures using one of the following methods: dimensionality reduction, category finding, unsupervised classification, factor analysis, autoencoing and many others.

The task of finding latent variables underlie many important algorithms. For example, the units of the hidden layers of a deep neural network are latent variables that represent hierarchical structures of a data set:

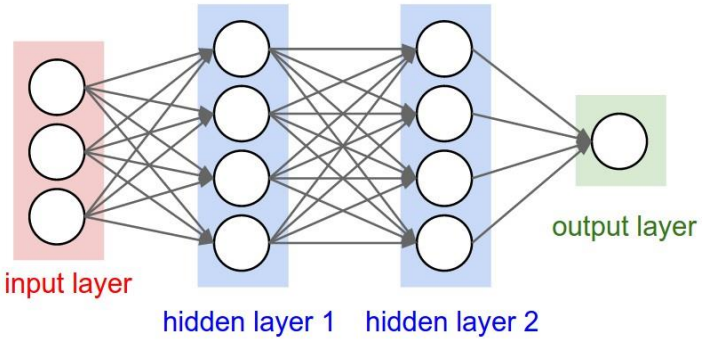


*Figure 1: A generic representation of an artificial neural network with hidden layers*

In the case of clustering, the latent variables are cluster points to which all points are in some way assigned to:

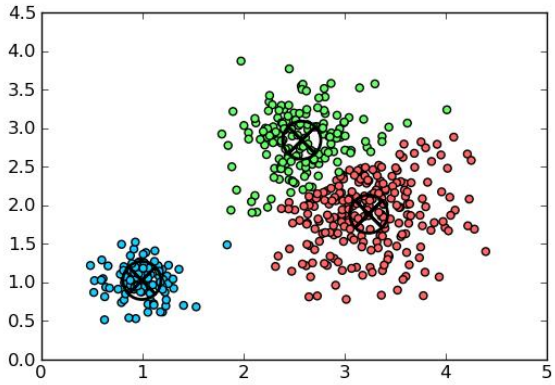


*Figure 2: Results of k-means. The black crosshairs are the cluster centers.*

## Expectation Maximization & The Mixture of Gaussians

Applying the expectation-maximization (EM) algorithm to a gaussian mixture model (MOG) is an example of a probabilistic unsupervised learning model with hidden variables. Suppose we have a dataset **Y** that can be separated into **observed** variables **X** and **hidden** variables **Z**. Suppose further that we have a learning model with parameters $\boldsymbol{\theta}$. Then $\boldsymbol{\theta^t}$ would be the estimate of $\boldsymbol{\theta}$ in the t-th iteration of the algorithm. In the EM algorithm, we want to calculate two quantities at every iteration:

1. The expectation, $E[\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta^t}]$ of the hidden variables given not only the observed variables, **X**, but also the current values of the parameters, $\boldsymbol{\theta^t}$.
2. The maximization of $P(Y|\boldsymbol{\theta})$ over the model parameters $\boldsymbol{\theta}$.

We know that **Y** is composed of **X** and **Z**, and therefore we need to use the previously calculated expectations of Z and the observed values of X to maximize

$$P(\boldsymbol{X} \cup E[\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta^t}] \mid \boldsymbol{\theta})$$

If we assume that the logarithm of the above probability is linear with respect to Y, then we can use the following useful property:

$$f(E[Z]) = E[f(Z)]$$

Appling this property to the probability we get,

$$\ln P(\boldsymbol{X} \cup E[\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta^t}] \mid \boldsymbol{\theta}) = E[\ln P(\boldsymbol{X} \cup \boldsymbol{Z} \mid \boldsymbol{\theta}) \mid \boldsymbol{X},\boldsymbol{\theta^t}] = E[\ln P(\boldsymbol{Y} \mid \boldsymbol{\theta}) \mid \boldsymbol{X},\boldsymbol{\theta^t}]$$

The expression is typically represented as $Q(\boldsymbol{\theta}|\boldsymbol{\theta^t})$. Finally, we are trying to calculate the next $\boldsymbol{\theta^t}$ by maximizing Q with respect to $\boldsymbol{\theta}$:

$$\boldsymbol{\theta^{t+1}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, Q(\boldsymbol{\theta} \mid \boldsymbol{\theta^t}) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, E[\ln P(\boldsymbol{Y} \mid \boldsymbol{\theta}) \mid \boldsymbol{X},\boldsymbol{\theta^t}]$$

The Expectation-Maximization algorithm in its most general form is represented by this expression. One way to start the algorithm would be to randomly initialize the parameters and/or the hidden variables. The expectation and maximization steps run at each iteration until the system reaches a stable state. Now that the mechanism of EM has been established, we will apply it to a gaussian mixture model.

If data points are generated by K normal distributions, then each distribution would have a corresponding mean $\boldsymbol{\mu}_j$ and covariance matrix $\boldsymbol{\Sigma}_j$. To apply EM to MOG we need the vector $\tau = (\tau_1, \dots, \tau_K)$, which represents the proportion of points coming from each distribution. The goal is to find the hidden classes by reconstructing the observed features.

Say for each data point we have a Boolean vector $z_i = (z_{i1}, \dots, z_{iK})$, where only one value in this vector is set to 1 and the rest are set to 0. One way to interpret this is that the i-th point comes from the j-th Gaussian. Using the expression for a **multivariate normal distribution** with the parameters above, we have the general expression for a Gaussian mixture model [Adebisi]:

$$P(x_i, z_i | \boldsymbol{\theta}) = \sum_{j=1}^{K} z_{ij}\tau_j \frac{1}{(2\pi)^{\frac{d}{2}}\sqrt{|\boldsymbol{\Sigma}_j|}} \exp(-\frac{1}{2}(x_i - \mu_i)^T \boldsymbol{\Sigma}_j^{-1}(x_i - \mu_i))$$

In other words, the probability that $x_i$ corresponds to the Boolean vector $z_i$ given the model parameters.

To apply EM to MOG we must put the above expression the form $Q(\boldsymbol{\theta}|\boldsymbol{\theta^t})$:

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta^t}) = E[\ln P(\boldsymbol{Y} \mid \boldsymbol{\theta}) \mid \boldsymbol{X}, \boldsymbol{\theta^t}]$$

$$= E[\ln \prod_{i=1}^{n} P(x_i \cup z_i \mid \boldsymbol{\theta}) \mid \boldsymbol{X}, \boldsymbol{\theta^t}]$$

$$= E[\sum_{i=1}^{n} \ln P(x_i \cup z_i \mid \boldsymbol{\theta}) \mid \boldsymbol{X}, \boldsymbol{\theta^t}]$$

Inserting the mixture model and simplifying we get,

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta^t}) = \sum_{i=1}^{n}\sum_{j=1}^{K} E[z_{ij}|\, \boldsymbol{X}, \boldsymbol{\theta^t}]\, (\ln \tau_j - \frac{d}{2}\ln 2\pi - \frac{1}{2} ln|\boldsymbol{\Sigma}_j| - \frac{1}{2}(x_i - \mu_i)^T \boldsymbol{\Sigma}_j^{-1}(x_i - \mu_i))$$

The model now is in desired form: it includes the expectations over the hidden variables given the data and the model parameters from the previous round. We can find $\boldsymbol{\theta^{t+1}}$ by maximizing over $\boldsymbol{\theta^t}$.

The expectation step calculates the expected values of the indicator variables $z_{ij}$. Since we are calculating the expectation of Boolean variables, we will have some value on the interval [0,1]. The constraint here is the following:

$$\sum_{j=1}^{K} z_{ij} = 1$$

In the expectation step alone we can see how EM applied to a Gaussian mixture model is a general case of k-means. **The cluster assignments for points have continuous values between 0 and 1 (soft assignments), whereas in k-means they are either 0 or 1 (hard assignments).** I will now show the generality in the maximization step.

Since the model parameters are $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$, we need to maximize $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^t)$ over them for each cluster center:

$$\mu_j^{t+1},\ \Sigma_j^{t+1} = \underset{\boldsymbol{\mu}_j,\ \boldsymbol{\Sigma}_j}{\operatorname{argmax}} \sum_{i=1}^{n} E[z_{ij}|\boldsymbol{X},\boldsymbol{\theta}^t](-\frac{1}{2} ln|\boldsymbol{\Sigma}_j| - \frac{1}{2}(x_i-\mu_i)^T\boldsymbol{\Sigma}_j^{-1}(x_i-\mu_i))$$

$$= \underset{\boldsymbol{\mu}_j,\ \boldsymbol{\Sigma}_j}{\operatorname{argmin}} \sum_{i=1}^{n} E[z_{ij}|\boldsymbol{X},\boldsymbol{\theta}^t](\frac{1}{2} ln|\boldsymbol{\Sigma}_j| + \frac{1}{2}(x_i-\mu_i)^T\boldsymbol{\Sigma}_j^{-1}(x_i-\mu_i))$$

In other words, we want to find cluster points that minimizes the sum of squared Euclidean distances to the corresponding points. We find the optimal values by taking weighted average over all the points (Flach):

$$\mu_j^{t+1} = \frac{1}{E_j}\sum_{i=1}^{n} E\left[z_{ij}\middle|\boldsymbol{X},\boldsymbol{\theta}^t\right]x_i = \frac{\sum_{i=1}^{n} E[z_{ij}|\boldsymbol{X},\boldsymbol{\theta}^t]x_i}{\sum_{i=1}^{n} E[z_{ij}|\boldsymbol{X},\boldsymbol{\theta}^t]}$$

$$\Sigma_j^{t+1} = \frac{1}{E_j}\sum_{i=1}^{n} E\left[z_{ij}\middle|\boldsymbol{X},\boldsymbol{\theta}^t\right]\left(x_i-\mu_j^{t+1}\right)\left(x_i-\mu_j^{t+1}\right)^T$$

$$= \frac{\sum_{i=1}^{n} E\left[z_{ij}\middle|\boldsymbol{X},\boldsymbol{\theta}^t\right]\left(x_i-\mu_j^{t+1}\right)\left(x_i-\mu_j^{t+1}\right)^T}{\sum_{i=1}^{n} E\left[z_{ij}\middle|\boldsymbol{X},\boldsymbol{\theta}^t\right]}$$

**The general form of the covariance calculation allows for cluster shapes of any size. In k-means cluster shapes are strictly spherical, and the maximization step does not depend on the covariance matrix:**

$$\mu_j^{t+1} = \underset{\boldsymbol{\mu}_j}{\operatorname{argmin}} \sum_{i=1}^{n} \|x_i-\mu_i\|^2$$

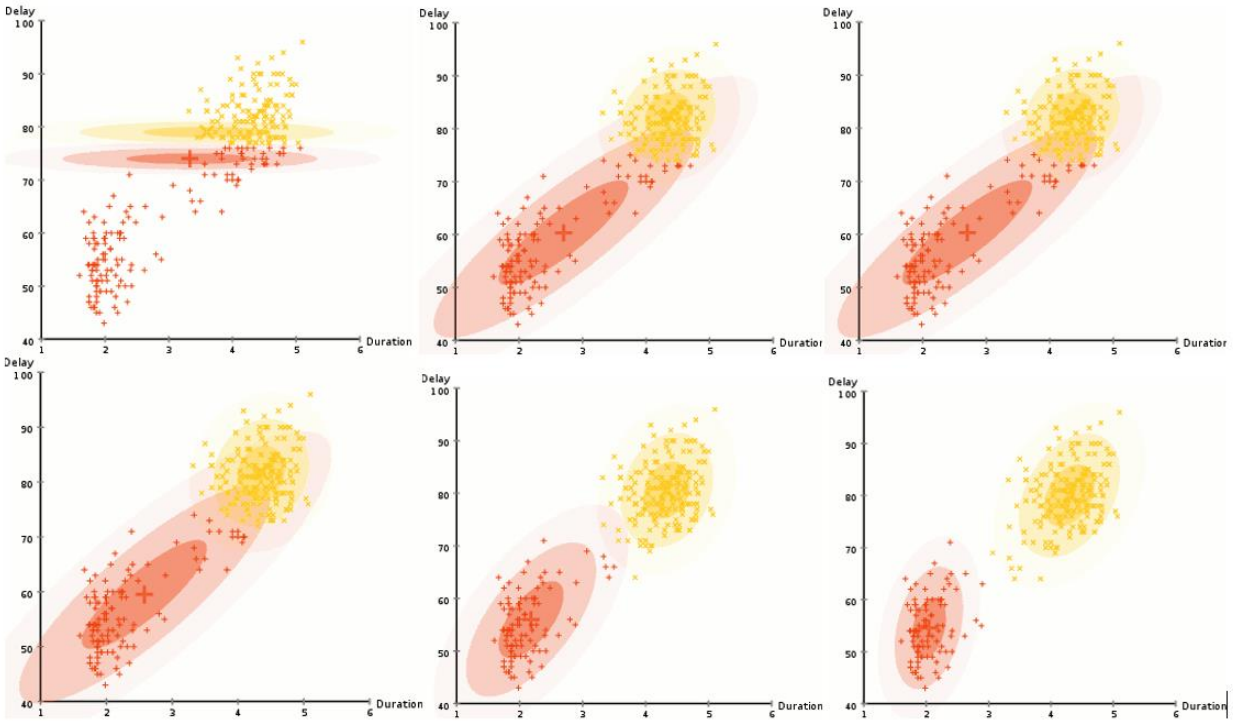


*Figure 3: A visual example of the expectation-maximization algorithm applied to a Gaussian mixture model.*

One can easily see the overlapping Gaussians and the non-spherical shapes. On the other hand, a visual example of k-means verifies that clusters can only be spherically shaped:

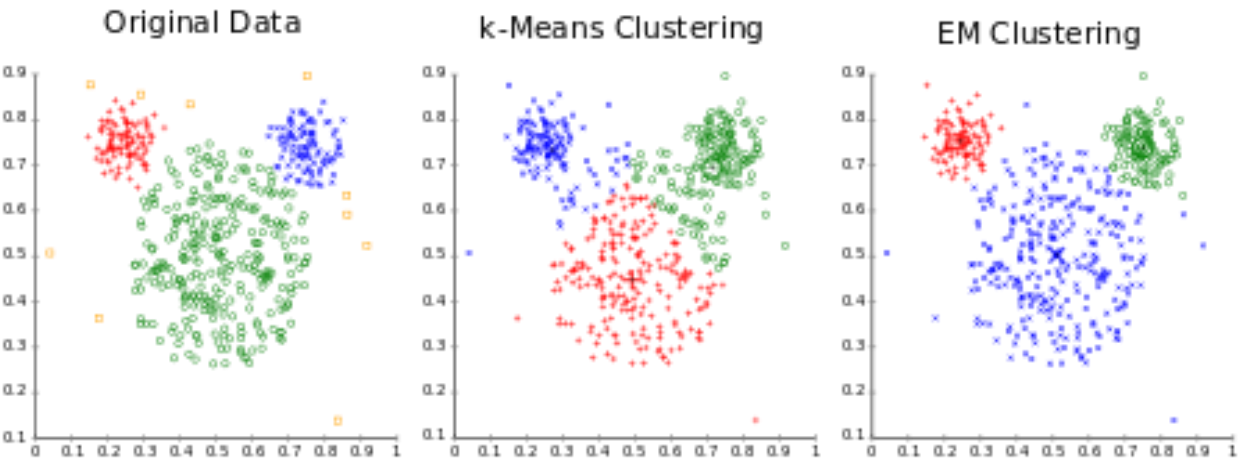


*Figure 4: Comparison of k-means and EM using a non-spherical covariance matrix.*

## Bibliography

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. Deep Learning. Cambridge, MA: MIT, 2017.

Flach, Peter A. Machine Learning: The Art and Science of Algorithms That Make Sense of Data. Cambridge: Cambridge UP, 2015.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep Learning." Nature 521.7553 (2015): 436-44.

Adebisi, Adigun, Omidiora Olusayo, and Olabiyisi Olatunde. "An Exploratory Study of K-Means and Expectation Maximization Algorithms." *British Journal of Mathematics & Computer Science* 2.2 (2012): 62-71.

## Other Sources

http://courses.ee.sun.ac.za/Pattern_Recognition_813/lectures/lecture03/node4.html

http://athena.ecs.csus.edu/~vanggs/177_finalpres.pdf

https://www.cs.cmu.edu/~guestrin/Class/10701-S05/slides/EM-MixGauss4-4-2005.pdf

http://research.ijcaonline.org/icquest/number1/icquest1510.pdf

https://www.slideshare.net/phvu/kmeans-em-and-mixture-models

http://cseweb.ucsd.edu/~atsmith/project1_253.pdf

https://alliance.seas.upenn.edu/~cis520/dynamic/2016/wiki/index.php?n=Lectures.EM#toc1

https://measuringu.com/latent-variables/

https://lagunita.stanford.edu/c4x/HumanitiesScience/StatLearning/asset/unsupervised.pdf

## Project Diary

**04/24/2017**

1. I started narrowing down possible project paths. I knew that I wanted to explore topics in unsupervised learning, but it was difficult deciding what to do.
2. My research is in the field of bioinformatics. I am given the task of finding methods to cluster large, complex and rich biological data sets.
3. Thus, I decided to explore some theoretical foundations of unsupervised learning. Although I do not have to explain the theoretical depths of unsupervised learning in my thesis and potential publications, this project would be an incredible opportunity to explore some of the underlying mechanisms of such methods.
4. I decided to explore the following topics: latent variables, expectation maximization and mixture of gaussians, and autoencoders.
5. I also decided to apply k-means clustering to a protein data set in a distributed Hadoop cluster that I have built.

**05/01/17**

1. Although I had already built a Hadoop cluster, I decided to scratch everything and start over. This is mainly because there was a major update that fixed many bugs. I also wanted to add a few more slave nodes.
2. The main complication with my desired set up was that I wanted my main lab computer to be the master node. However, my lab computers are in their own private network with limited root access.
3. Nonetheless I fresh installed Debian on all of the lab computers and connected all of them via a network switch.
4. I downloaded and configured Hadoop on the master node.
5. I then set up passwordless SSH to all the slave nodes from the master node.
6. There were many networking complications that I ran into. These are not worth mentioning, but they were problems I had never encountered.
7. I was not able to finish the set up today. It was 1am and I was no longer able to stay awake.

**05/02/17**

1. I came right back into the lab the next day with very little sleep. I had to get this thing done today.
2. I learned that simple network switches do not conveniently assign IP addresses to computers. I had to set up my own DHCP server on the master node.
3. After hours of research I learned that I need a router. Routers have DHCP servers built into them. I found an old, unused router and eventually got the network set up.

4. I could now launch Hadoop and all the respective slave daemons from the master node. I was also able to monitor the status of slaves using Hadoop's built-in web UI. This made me very happy.

**05/07/2017**

1. With the Hadoop cluster up and running, I decided to finally start on the write-up of my project.
2. I gathered my initial set of sources:
    a. **Machine Learning by Peter Flach**
    b. **Deep Learning by Ian Goodfellow**
3. I have decided to migrate over to Spark (as opposed to Map Reduce).
    a. Datasets are smaller
    b. Data processing/analysis is done in memory
    c. Spark runs up to 100x faster than MapReduce in memory or 10x faster in disk
    d. Provides high-level APIs in Java, Scala, Python, and R
    e. Has a well-known machine learning library called MLlib
4. I should mention that Spark still runs in a Hadoop cluster.
5. I downloaded, installed, and configured Spark.
6. By the end of the day I was able to successfully run Spark in the Hadoop cluster.

**05/08/2017**

1. I established the sections of my final report.
2. I started and completed introduction section of my report.
3. I gathered a few more resources:
    a. **Deep Learning; Nature; LeCun**
    b. **Latent variable modeling; Shangai Archives of Psychiatry; Li CAI**
    c. **Overview of K-means and Expectation Maximization Algorithm for Document Clustering; Umale**
    d. **Clustering With EM and K-Means; Alldrin**
    e. **Other class notes from credible university courses (UCSD, CMU, Stanford)**
4. I started and completed the latent variables sections of my report.

**05/09/2017**

1. I started taking a deep look at expectation maximization.
2. I read the EM section in Machine Learning by Flach.
3. I started working on the derivation for the EM algorithm.
4. I updated my report with my newly acquired knowledge.

**05/10/2017**

1. I completed the derivation of the EM algorithm.
2. I started the application of EM to a Gaussian mixture model.
3. This derivation was quite complication so I was only able to finish about half of it.
4. I updated my report with my newly acquired knowledge.

**05/11/2017**

1. I completed the derivation for the application of EM to the Gaussian mixture model.
2. I also showed that k-means is a special case of EM.
3. I then updated my report.

**05/14/2017**

1. I ran k-means locally 30 times, each time with a different number of starting clusters.
2. I ran k-means on Spark 30 times, each time with a different number of starting clusters.
3. I created plots of the running times vs. number of clusters.
4. The results are quite impressive; I discovered that Spark is an order of magnitude fast.

**05/15/2017**

1. I put together my presentation.
2. I realized that it would be beneficial to briefly go over Hadoop and Spark as well as my journey into building a Hadoop cluster.
3. The presentation now only needs screenshots of the Web UIs for Hadoop/Spark as proof of a running cluster.
4. I made the final edits to my report and put everything together.

## Source Code

All source code can be found here:

https://github.com/Vardominator/CSULBProjects/tree/master/CECS551_AdvancedAI/FinalProject