Project midway report

Anupam Samanta, Deepak Gupta, Keshav Gupta, Soumyadeep Chakraborty

Problems faced:

1. Feeding highly-dimensional feature vectors representing genome samples to any machine learning algorithm is expensive both in terms of space and time complexity.
2. Selecting the correct machine learning model, given that **Random Forest** is known to produce an accuracy of **66%** when applied here.
3. Avoiding **overfitting**: Because of limited sample space, there is a fine line between supposedly correct results and "overfitting"
4. Using the **fragment ambiguity graph** to figure out which transcripts are unique to certain individuals (enabling their use as identifiers of the population that they represent), and sifting out which transcripts are expressed across individuals.

Solutions:

1. To solve problem 1: We used the following **dimensionality reduction** techniques:
    a. **PCA** - PCA restritcs us to use **369** principal components and hence there was a lot of information loss in features.
    b. **Tree based feature selection model (Scikit-learn)** - This uses tree based estimators to measure feature importances which can be used to eliminate irrelevant features. With this, we could reduce the feature vector to a size of **828** features. Accuracy achieved **= 67%**
2. To solve problem 2: **Machine learning model selection**-
   Deep learning model - Used a **2-layer fully connected neural network**. The first layer had **800 neurons** followed by **10 neurons** in the second layer. Initially, the resultant accuracy was **40%**. However, after inspection we found out that this poor outcome was a result of non-standardized data. Thus, we used **Z-score standardization** (**Scikit-learn**) to center and scale each feature independently. Subsequently, we achieved an accuracy of **77.3%**. We also used regularization to overcome the problem of less number of dataset samples.
3. Since we used a deep learning model, we discovered that longer epoch execution resulted in the deep learning model was memorizing the training data (During the training the accuracy > **95%**, which fell to 77% over 1000 epochs). We applied **Stratified KFold Validation** with **number of splits = 3** to add a validation step during training. This led to a test accuracy of **81.1%** and a train accuracy of **91.9%**. We still have to improve this by more hyper-paramter tuning.
4. We are yet to incorporate this information into our framework and this will constitute our **future direction** of work.