AML CLASSIFICATION
EARLY TERM PROJECT

Group Members: Jaswanth Sai Pyneni, Gulay Bengu Ulukaya, Manasa Vegesna

Normalizing and Preprocessing the data:

We first normalized the FCS measurements per file by normalizing each with respect to the other FSC values within the same file, as explained in lecture. We then took a mean features approach and averaged all features in each file to create 7 cells per file. Per patient, each files average was appended onto each other into one row to create a 56 column (7*8) feature vector per patient. Thus, we combined all the data into a 359 (rows) * 56 (columns) matrix to represent the data, with each row being all the features of one patient and each column being the same measurement category for each patient.

After the data was normalized, we split the training data randomly using a 80/20 split ratio. The split data was used to train the machine learning model on 80% of data points and validate its performance on the remaining 20%.

Method for defining features:

The features used in the training data are the means over each column in each file per patient, with the averages being appended from all files related to the patient. This measurement is the mean FSC and SSC per tube and mean expression of each of the five proteins per tube.

Algorithm used:

We tested out three different machine learning classification algorithms - Logistic Regression, KNN and Random Forest algorithms on the training data. From the 80/20 split on the training data, we trained each model on the 80% split and then used a validation data set (the remaining 20%) to compare the performance of each model on data it did not see during training. Once we chose the best model, we re-evaluated the entire training data before producing the final predictions.

First, we tested and validated linear and logistic regression models on the training data and found that the model validation accuracy was low (<76%). Second, we used a K-Nearest Neighbors (KNN) model on the training data which yielded a 97.2% accuracy. For the KNN model, I hyperparameter tuned by testing multiple values for k. I tested 2 through 9 as the value for k and got the same performance on the validation set for each k. As such, to limit model complexity, we used k=3 for the final KNN model. Third, we tested out a Random Forest machine learning model which had a similar accuracy rate of 97.2%. Since the Random Forest and the KNN model had the same accuracy scores, we have to rely on other performance metrics to decide them.

We calculated the F1 score, balanced accuracy score and ROC-AUC from prediction scores for both the models and got the same results for both: 0.900 balanced accuracy score, 0.972 accuracy score, 0.889 F1 score and 0.900 roc-auc score. Therefore, in order to decide which model to use for predicting on the test data set, we trained both models with all of the training data, preprocessed the test data similar to the train data, and then ran both models on the test data and produced confidence scores from each model for each individual prediction using sklearn.model.predict_proba(). We compared the predictions of each test sample by the KNN and Random Forest models and found 4 differences. To further discern the models on these 4 different predictions, we compared the prediction probabilities of each prediction in both the models and found that the KNN model had a higher prediction probability in all 4 cases (it was more confident in each of its answers than the Random Forest model was). Therefore, we used the KNN model to create our predictions.

Level of confidence:

Assessing the prediction probabilities on each test sample using KNN model, we see that 170 of the test cases had 1.0 prediction probability. However, the prediction probabilities varied between 0.62 and 0.74 for the remaining 10 predictions. Of these 10 cases, 4 of them were predicted differently in the Random Forest model. There are a total of 176 samples which produced the same prediction between both the KNN and the RF model. This included a combination of the 170 from KNN with 1.0 prediction probability and the other 6 that the two models did not disagree on. As a result, we are highly confident in 176 predictions and are slightly less confident in the other 4 predictions (samples 239, 250, 337, and 357).

Results:

Number of positive cases identified in test data: 20 out of 180 samples