Or Samimi Golan 305641045

Shajaf Gonen 204067268

Unless specified otherwise in an alternative, we generated our classifiers in the following form:

1. SNP-wise. Meaning, that we generated a classifier for each missing SNP in the test set (300 classifiers in total in each alternative). The classifier of each SNP is the one that achieved the best success rate for the SNP amongst all alternatives(see section 2).
2. Each sample in the train (test) set, is the 100 SNPs before and after the SNP to be learned (200 feature in the feature vector in total). The matching label of a sample, is of course the value of the SNP. (Known in the train set = {0, 1, 2}, unknown in the test set = {-1}). These details are generated out of 'extracted\_train' and 'extracted\_test'
3. The success rate for each SNP is calculated using 10-fold validation, with a 0-1 loss function.
4. The total success rate is the average of all SNPs success rate.

Section 1 – Same classifier for each SNP

At first , each of us generated a classifier, and tested it.

Alternative 1 - SVM – one vs one using libsvm

Libsvm handles mutli-class classification, using one-vs-one technique (learned in class).

By running: svmtrain, using the following options.

Success rates

Linear SVM

C = 1: 69.47%

C = 0.1: 72.62%

C = 0.01: 70.69%%

Polynomial SVM

C = 1, degree = 4: 73.25%

C = 0.1, degree = 10: 72.37%

C = 1, degree = 10: 72.30%

C = 100, degree = 10: 72.30%

Radial basis SVM

C = 1, gamma = 1/features\_number (default value)= 67.51%

C = 100000, gamma = 1/features\_number (default value) = 73.69%

Alternative 2 – Multiclass Adaboost

We modeled the missing SNP value as dependent on some of the neighbor SNPs. For each missing SNP, we want to find the

Section 2 – Different classifier types for different SNPs

We understood that learning a classifier f

Section 3 – Our script usage and output

Summary

|  |  |  |
| --- | --- | --- |
| Script Name | Description Summary | Success Rate (10-fold-validation) |
|  |  |  |
|  |  |  |
|  |  |  |