**Project 1 Advanced Apllications of Machine Learning**

Overview

The algorithm uses Random Forest to categorize into the classes circRNA encoding and not circRNA encoding. The algorithm precomputes the testdata (see below). It uses the features k-mers (size 4), length of the sequence, chromosome and tandem repeats. The effectivity of these features is explained in the papers [ADD NAMES] and was found by grid search as well as local considerations.

Training Data Used

For Training the AI, I used the following way to generate positive and negative examples of training data:

Positive Example: A single gene that is encoding a circRNA

“hsa\_hg19\_Rybak2015.bed” stores the loci of circRNA encoding genes

Negative training data: A single, whole exon that does not contain any part of a circRNA encoding gene

The whole genome is used for positive and negative exmaples.

The training data is generated in the folder training\_data by the script traing\_data.py

Algorithm used

Discussion

Choice of training data

The algorithm assumes, that the exons that do not contain any circRNA parts have the same DNA structure as parts of an exon that do not encode a circRNA, this is because of the choice of training data. This assumption seems to be reasonable, but it might have a slight effect.

I assume that the input to classify will be a whole exon. Thus whole exons are used as negative examples. That means that the length of the input has an effect on the classification. If only genes with similar length as the circRNAs are used this criterion will not be so effective.

Choice of features

For the length feature see discussion above.

The tandem repeat feature is supposed to be a good indicator for the encoding of a circRNA. However in my implementation it did not make a strong difference and was due to performance reasons disabled. The effectiveness is expounded in the paper “PredcircRNA: computational classification of circular RNA from other long non-coding RNA using hybrid features†”

The k-mer feature is a known and effective feature for dna classification. Its effectiveness was confirmed by the test data. We use k-mers of length 3 for performance reasons. The algorithm is flexible in this regard and increasing will increase the accuracy of the predictions. Since this is the key feature, the algorithm is very flexible.

The chromosome which stores the gene is also a good indicator for the class. 35 out of 60 exon genes do not encode circRNAs at all. Those chromosomes are a lot smaller, but this is a very reliable feature.

Choice of classification algorithm

The classification algorithm and its parameters were found by grid search. The accuracy can further be increased by decreasing the learning speed, which has been set to 0.4 for performance reasons, which is quite high.

Time performance of algorithm

The algorithm runs fast by precomputing the test data and saving them to be reused. Also the kmers and the learning rate are quite low. This results in a fast performance of the algothm which will finish after 10 minutes on a standard machine.

Performance evaluation

Test Run of algorithm

I plot the output of a test run of my algorithm, so the performance can be seen quickly.

(C:\Program Files\Anaconda2) C:\Users\NoahH\_000\Desktop\project1>python predict.py

Fetching the training data...

## Loaded 65673 positive examples and 65673 negatives

Building features...

extracting repeat feature

Set 0 with score: 0.901408450704

Set 1 with score: 0.896003045299

Set 2 with score: 0.890216977541

Set 3 with score: 0.889912447659

Set 4 with score: 0.891968024362

Set 5 with score: 0.897830224591

Set 6 with score: 0.89920060906

Set 7 with score: 0.899961933765

Set 8 with score: 0.892044156833

Set 9 with score: 0.889455652836

ACLU: 0.967520400932, Average AUC (ROC): 0.894814858084 Average F1 Score: 0.894753548328