Oct 21st 2021

CS 194-172 – Computational Genomics – HW 3

Stefan Bielmeier

Problem 4a) and b)

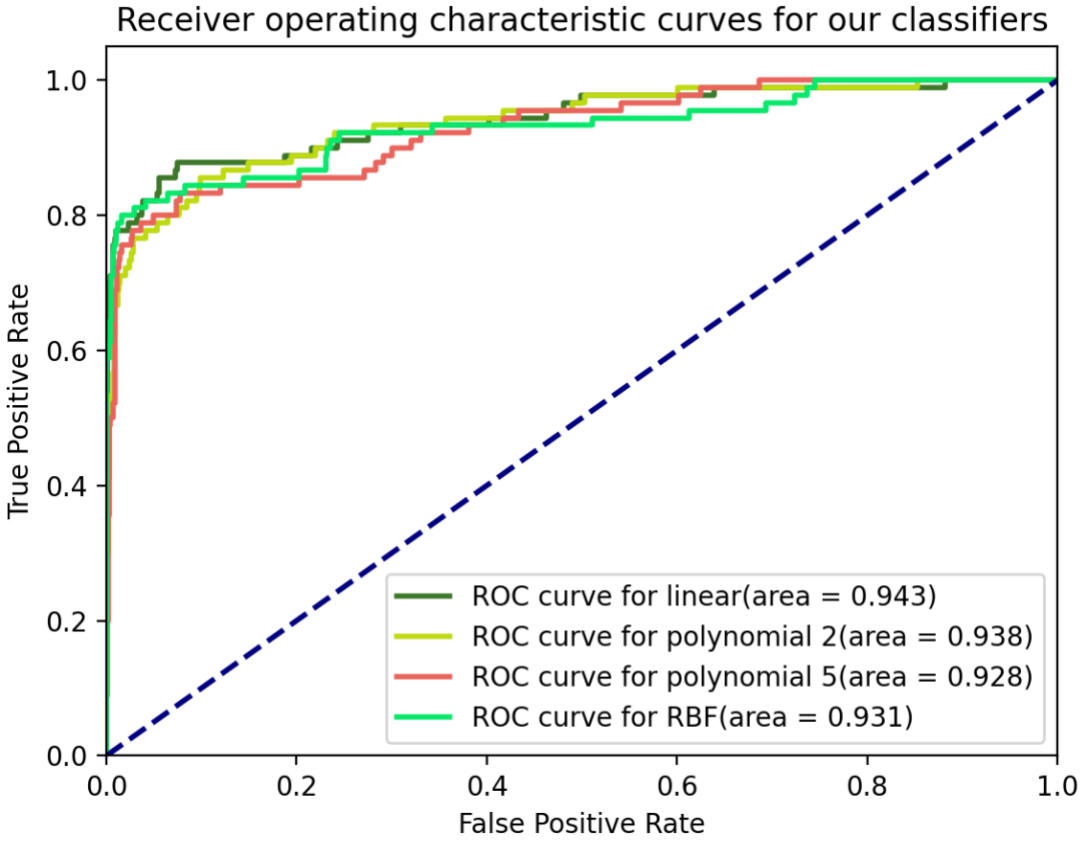
SVM Kernel Type	**Best value of C**	**Mean cross-validation accuracy for that C**	**Test accuracy for that C**	**Total running time (secs) of cross-validation for all possible Cs** (0.1, 1, 10, 100, 1000)
Linear	0.1	0.981	0.974	306.0
Polynomial degree 2	0.1	0.981	0.953	274.9
Polynomial degree 5	1000	0.973	0.963	0.4
RBF Kernel	1000	0.98	0.97	< 0.0

Raw data

```
acc for poly2 for C=0.1 0.9810000000000001
acc for poly2 for C=1.0 0.9800000000000001
acc for poly2 for C=10.0 0.9790000000000001
acc for poly2 for C=100.0 0.9800000000000001
acc for poly2 for C=1000.0 0.976
cross-validation took 274.9 secs
acc for poly5 for C=0.1 0.913
acc for poly5 for C=1.0 0.9400000000000001
acc for poly5 for C=10.0 0.968
acc for poly5 for C=100.0 0.9709999999999999
acc for poly5 for C=1000.0 0.9730000000000001
cross-validation took 0.4 secs
acc for rbf for C=0.1 0.945
acc for rbf for C=1.0 0.97
acc for rbf for C=10.0 0.975
acc for rbf for C=100.0 0.9799999999999999
acc for rbf for C=1000.0 0.9800000000000001
cross-validation took 0.0 secs
acc for linear for C=0.1 0.9810000000000001
acc for linear for C=1.0 0.9800000000000001
acc for linear for C=10.0 0.9790000000000001
 acc for linear for C=100.0 0.9800000000000001
acc for linear for C=1000.0 0.976
cross-validation took 306.0 secs
```

Problem 4c)

ROC curves for each classifier and respective AUROC in label as “area”.



Problem 4d)

The linear Kernel probably trains the longest because it tries to separate the data into zero and ones with the most constrained assumption of separation, a single Hyperplane. Finding a hyperplane of dimension $K - 1$, where k is the *number of columns*, that almost linearly separates the data into two classes – zero (0) and one (1) probably needs quite the number of iterations, because it is quite unlikely that the data is linearly separable at all, the bigger K gets for the same number of observations N.

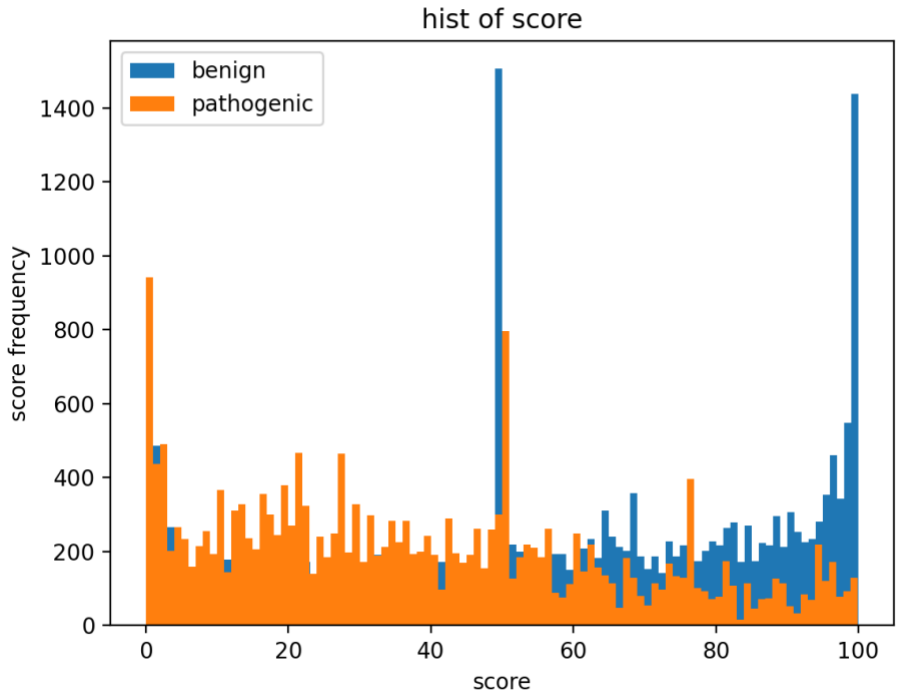
In contrast, a polynomial of 2nd degree SVM may converge quicker as it basically is not constrained to one, but **two** Hyperplanes / Support vectors that separate the data. That gives more flexibility to train the model and may lead it to converge quicker (find a local optimum) to appropriate parameters. A polynomial of 5th degree SVM has 5 possible support vectors to separate the data into 0s and 1s, given that K dimensions and the number of observations stays the same.
This is also why we observe inversely correlated model training times with dimensionality of the model.

Problem 5a)
After this filtering, 24346 benign and 20596 pathogenic variants remain.

Problem 5b)
After randomly splitting into 80% training data, and 20% validation data, there are

- 16517 pathogenic and 19436 benign in the training set
- 4079 pathogenic and 4910 benign variants in the validation set

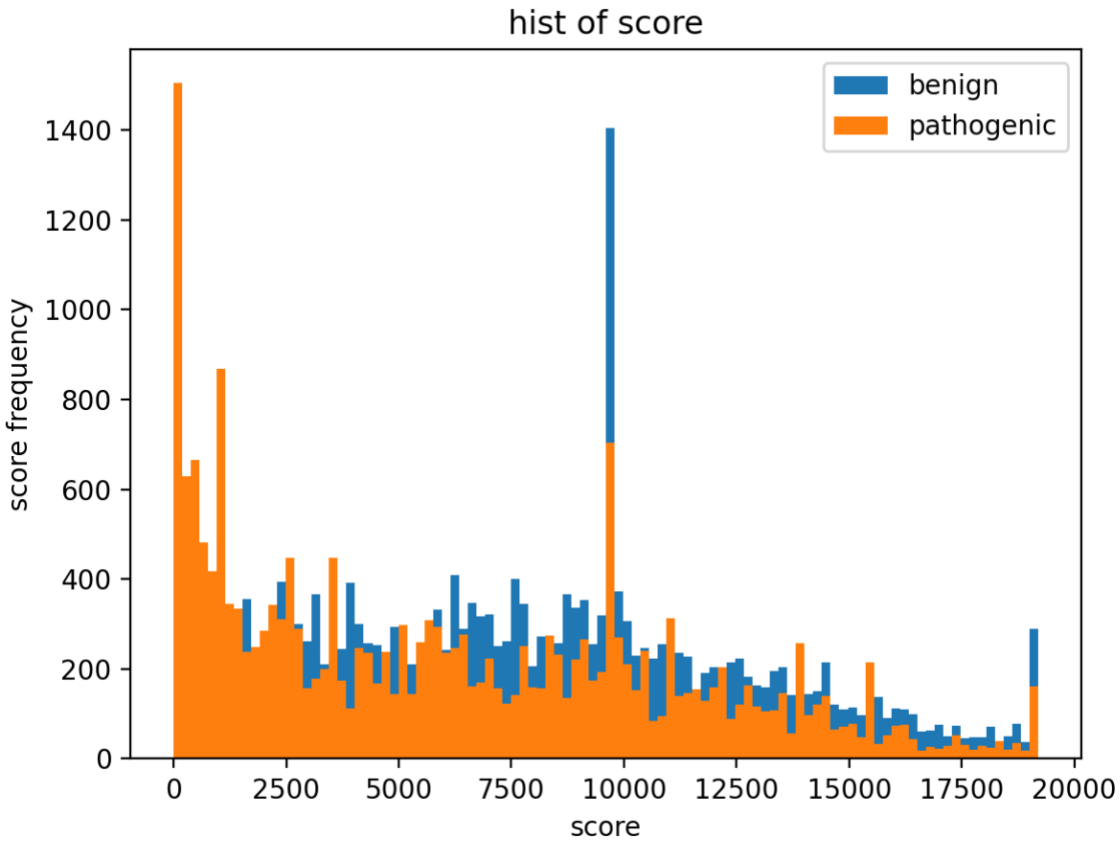
Problem 5c)



Based on the plot, do you think RVIS is a good feature to add to our model?

A gene with a positive score has more common functional variation, and a gene with a negative score has less and is referred to as "intolerant". The percentiles work the same way, the higher, the more common, the lower, the less common. Therefore, RVIS seems like a good feature to add to the model because it can help the model distinguish more common function variants (benign) from less common functional variants (potentially pathogenic).

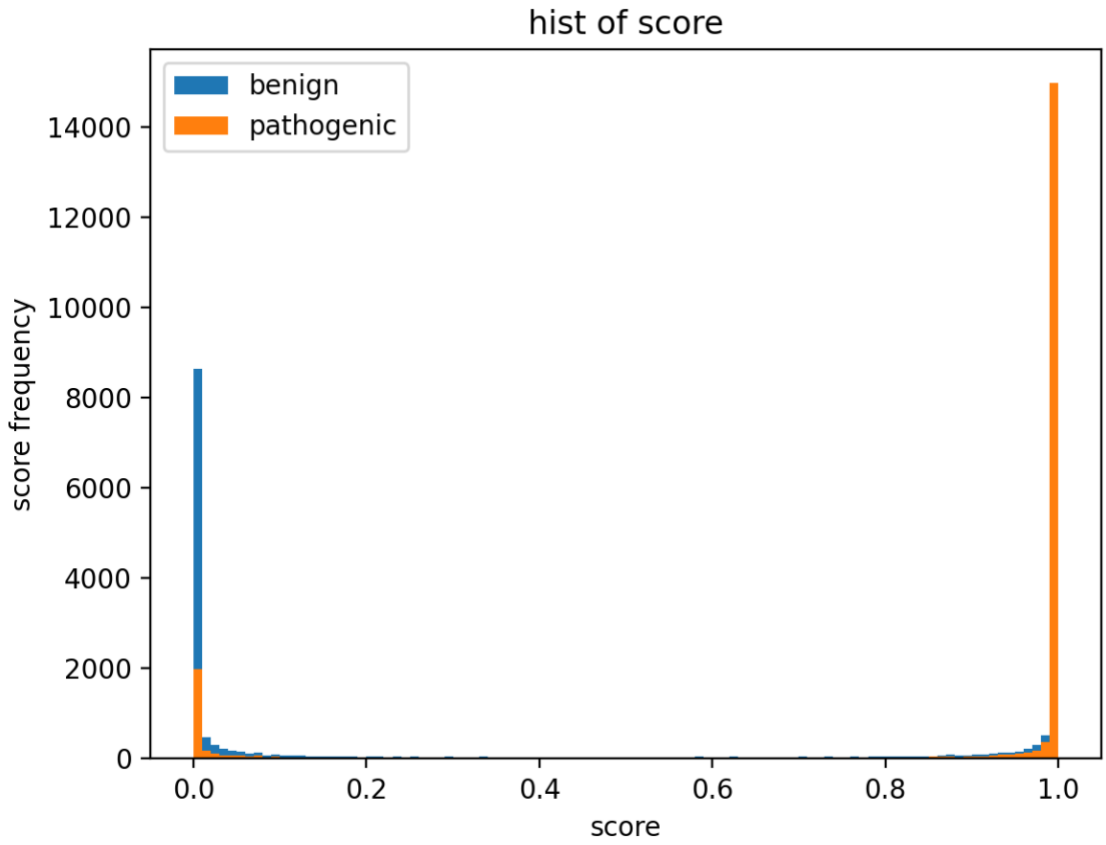
Problem 5d)



This seems like an okay metric. Very high ranks (i.e. number 1, 2) usually are only assigned to pathogenic variants, while high to very low ranks (1000 – 20000, etc.) are assigned to benign or pathogenic variants in a seemingly similar distribution.

Usually high ranks are associated more with pathogenic scores than benign, any other rank says: doesn't matter, which may help our model predict as well.

Problem 5e)



The phastCons score for each nucleotide describes the probability that the nucleotide is part of a conserved region.

- ⇨ High phastCons score = highly conserved = a mutation in highly conserved (i.e. important) regions is probably damaging
- ⇨ Low phastCons score = not conserved at all = probably not so damaging if a mutation occurs but can also be damaging. That's why we see both pathogenic and benign variants, but way more benign ones.

I think this is an OK feature, as it does not separate the pathogenic or benign scores well-enough so that the machine learning model can make clear distinctions because of low scores showing both categories. However, a high score very clearly correlates with pathogenicity.

Problem 5f)

I chose a Random-Forest Classifier. I trained the model on 60% training data to achieve a better accuracy. I used the data compiler Brainome (CS 294-082, brainome.ai). I did not need to perform cross-validation, as the data compiler trains the model with training data that approximately resembles the split of classes in the entire dataset. In our case, it's about 54.17% benign variants, and 45.83% pathogenic variants. This is why I used a split into 60% training data to achieve best results for training accuracy and validation accuracy (combined model accuracy: 85.57%).

Validation accuracy on the 40% test set: 79.35%

The chosen hyperparameters for the Random Forest are:

- n_estimators = number of trees in the foreset
- max_features = max number of features considered for splitting a node
- max_depth = max number of levels in each decision tree
- min_samples_split = min number of data points placed in a node before the node is split
- min_samples_leaf = min number of data points allowed in a leaf node
- bootstrap = method for sampling data points (with or without replacement)

When I split into a randomly selected 20% validation set. My model achieves the following ROC curve on that randomly selected 20% validation set, and an AUROC of 0.933.