Predicting with SVM models

Ralph Schlosser, ralph.schlosser@gmail.com

Introduction

In this report it is intended to apply *Support Vector Machines* (SVMs) to a classification problem. Specifically I am interested in comparing the classification accuracy of four different kernels and see how well they do against Random Forest classification. The `kernlab` library provides the implementation for kernels and SVM algorithms used in this report. The data set we are considering is the 1984 US Congress election data (see URL below), and we are looking at how accurately we can predict the political party of a candidate based on number on their received votes.

Methodology

The election data set records 435 rows with 16 columns for the votes (yes or no) and a 17th column indicating the political party a candidate belongs to.

The original file containing the data can be retrieved from:

`http://archive.ics.uci.edu/ml/machine-learning-databases/voting-records/house-votes-84.data`.

Before running the algorithms all the votes recorded as `"?"` were changed into `"n"`.

The principal comparison methodology is as follows:

1. Divide data in to training, validation and test set with a split of $50\%, 25\%, 25\%$, respectively.
2. Fit SVMs using different kernels on training data.
3. Use fitted SVMs to predict on validation data and calculate **validation set accuracy**. At this stage we could have evaluated the accuracy to determine the kernel that works best for the given data. But instead of selecting a single best classifier, I am recording the validation set accuracy for all SVMs.
4. Using the fitted SVMs from above, predict on the test set and calculate **test set accuracy**.
5. Build a Random Forest classifier, predict on test and validation set and calculate the same accuracy measures as above.

The above steps are repeated 20 times so as to be able to look at summary statistics later.

The following kernels were used with the SVMs: rbfdot, polydot, vanilladot, tanhdot.[1]

Results

For the test set accuracy we get the following summary:

```
##         rbfdot polydot vanilladot tanhdot randomForest
## Min.    0.9182  0.9182     0.9182  0.5636       0.9182
## 1st Qu. 0.9432  0.9455     0.9455  0.5909       0.9432
## Median  0.9545  0.9545     0.9545  0.6227       0.9545
## Mean    0.9500  0.9532     0.9532  0.6186       0.9527
## 3rd Qu. 0.9636  0.9636     0.9636  0.6455       0.9636
## Max.    0.9727  0.9909     0.9909  0.7000       0.9818
```

It is clear that rbfdot, polydot, and vanilladot are very similar in their results and achieve the same test set prediction accuracy on average. Also, the random forests approach predicts the political party with – again – a very similar accuracy, so the mentioned SVMs don't seem to have an advantage / disadvantage here as far as the results are concerned.

[1] The vignette for `kernlab` as well as the help file discusses various "tuning" parameters that can be supplied to kernels. Due to time constraints, I couldn't spend more time on studying their effects on results, so only **default** settings, like e.g. automatic choice of the `sigma` parameter for `rbfdot` were used.

Only the tanhdot kernel seems to be doing much worse in contrast to the others, and Random Forest, with a much lower average accuracy.

Similar results are achieved when considering the validation set accuracy:

```
##         rbfdot polydot vanilladot tanhdot randomForest
## Min.    0.9074  0.9074     0.9074  0.5370       0.9167
## 1st Qu. 0.9444  0.9421     0.9421  0.5741       0.9444
## Median  0.9537  0.9537     0.9537  0.6019       0.9583
## Mean    0.9551  0.9542     0.9542  0.6023       0.9560
## 3rd Qu. 0.9722  0.9722     0.9722  0.6296       0.9653
## Max.    0.9907  0.9907     0.9907  0.6852       0.9907
```

We find that rbfdot, polydot, and vanilladot are marginally better on the validation data, while tanhdot seems to be doing a little worse. As before, Random Forest is on the same level as the first four kernels mentioned.

Finally, in Figure 1, we can get an impression as to the fluctuation of results for test and validation set accuracy when plotting the achieved accuracy for each run. We again see the very good results described above for Random Forest as well as the SVMs, with the exception of tanhdot which also displays a higher variability.

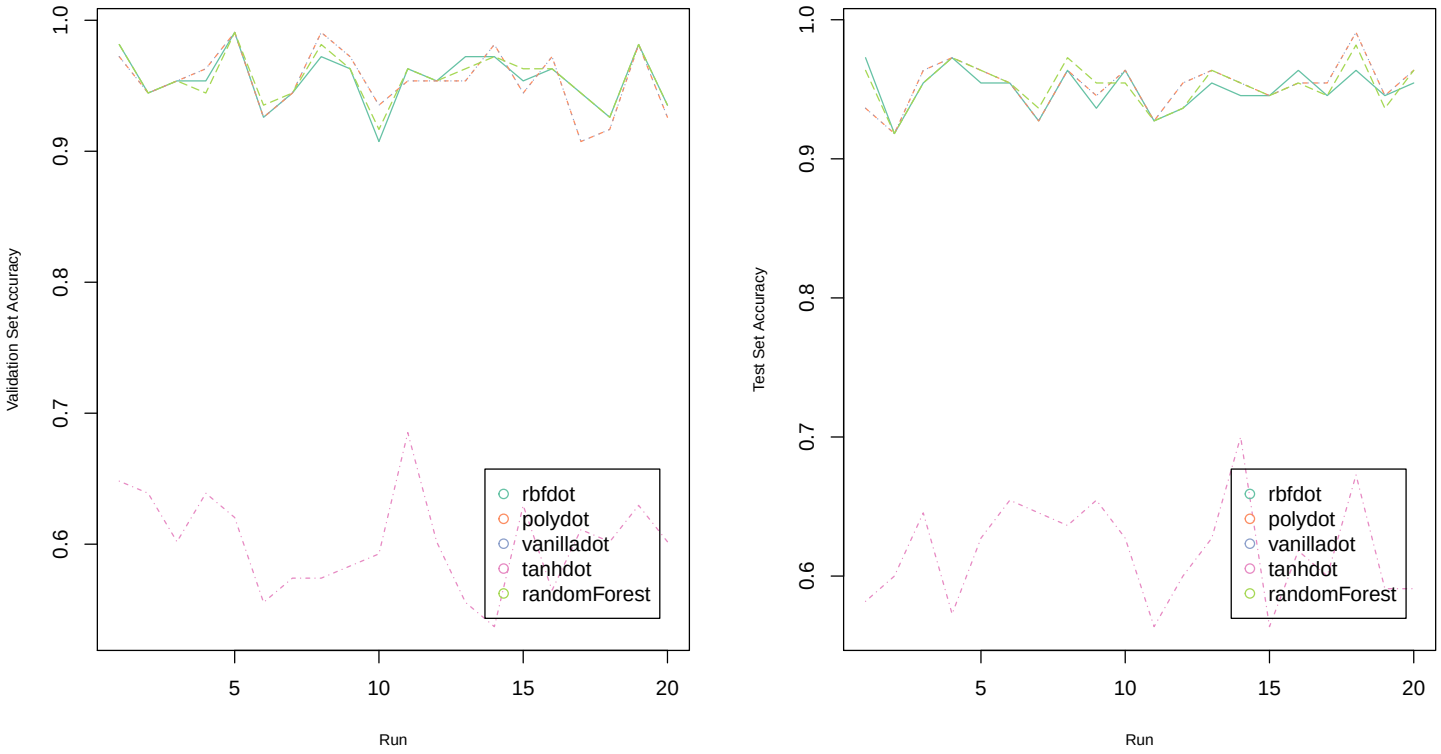

Figure 1: Accuracy Plot

Conclusion

- Most of the SVM kernels tested did very well at predicting the political party of a candidate based on the votes.
- Only the tanhdot kernel would seem to be a poor choice for this prediction problem.
- In this example, there is no clear winner in terms of prediction accuracy, however there could be other advantages with certain algorithms, like e.g. computational efficiency, stability and so on. But I haven't looked into these further.
- Also I haven't considered what effect differnt parameters for kernels (called *hyper-parameters* in the `ksvm` help) have on the prediction accuracy.
- Due to above it is feasible that tanhdot could be improved with a "tuned" set of parameters.