README for the performance measurement part

Group 2

December, 2nd, 2019

Abstract

This Chapter is describes the evaluation of the best sampling and model combination, which is computed in the script 'PerformanceMeasurement.R'.

The goal of our performance measurement

After training a model it is important to check how well it performes. Since we not only trained one model but 140 in total (7 algos * 5 datasets * 4 positions), it is even more important to compare all the different models to see which one is the best. For our business case, this means that we want to see which combination of method and sampling we should take to predict which position, or if we even have a model, which does it better when we leave them together.

How to compare the method/sampling combinations for all positions

After training the models with 10-fold cross-validation, we applied them on all the (unsampled) data from 2007 to 2013 to obtain the true positives, true negatives, false positives and false negatives. Since we used the cross-validation for training, this will not give us the training error, but the testing error we get with all the available data prior to the 2014 NFL Draft. At the end of every model-script, we save this information separately, to bring it all together in the script 'PerformanceMeasurement.R'.

The first step, is to bring them all into one dataframe, to use them easier. Then we make sure, that we used the same, unsampled data by computing the sums of TP, TN, FP, FN, for every method/sampling/position combination. In the CheckTibble we now see, that for every position the whole column contains always the same number, which means this is the case.

Then we calculate the ratio of correct classification, which is equal to:

$$\frac{\text{Correct Classifications}}{\text{All Classifications}} = \frac{TP + TN}{TP + TP + FP + FN}$$

The result is a table showing the percentage of correct classification for all the 140 model/sampling/position combination. This table is quite big, but it is still interesting to have a look at it, to see how well which combination performes.

Table 1: Ratio of correct classification

Method	Sampling	QB	WR	RB	Together
ClassificationTree	no_sampling	0.9098	0.9208	0.9156	0.8939
ClassificationTree	oversampling	0.8780	0.8415	0.8631	0.7930
ClassificationTree	undersampling	0.7098	0.8254	0.8137	0.7938
ClassificationTree	Rose_both	0.8317	0.7985	0.8057	0.7746
ClassificationTree	Smote	0.8439	0.8062	0.7850	0.7998
KNN	no_sampling	0.8488	0.8892	0.8646	0.8755
KNN	oversampling	0.8366	0.8915	0.8519	0.8725
KNN	undersampling	0.6463	0.8331	0.7962	0.7994
KNN	Rose_both	0.7317	0.8192	0.7834	0.7981
KNN	Smote	0.8610	0.9023	0.8726	0.8918
NaiveBayes	no_sampling	0.8268	0.8908	0.8758	0.8901

Method	Sampling	QB	WR	RB	Together
NaiveBayes	oversampling	0.7000	0.7746	0.8137	0.7998
NaiveBayes	undersampling	0.6878	0.7946	0.8328	0.7968
NaiveBayes	Rose_both	0.7341	0.7231	0.8615	0.7930
NaiveBayes	Smote	0.7317	0.8077	0.7213	0.8798
randomForest	no_sampling	1.0000	0.9977	1.0000	0.9979
randomForest	oversampling	1.0000	0.9985	1.0000	0.9996
randomForest	undersampling	0.7415	0.8838	0.8344	0.8499
randomForest	Rose_both	0.9073	0.9462	0.9395	0.9371
randomForest	Smote	1.0000	0.9962	1.0000	0.9983
ANN	no_sampling	0.8488	0.9054	0.8838	0.8828
ANN	oversampling	0.8593	0.8664	0.8574	0.8531
ANN	undersampling	0.8559	0.8771	0.8708	0.8440
ANN	Rose_both	0.8923	0.8570	0.8558	0.8512
ANN	Smote	0.8639	0.8853	0.8692	0.8628
LogisticRegression	no_sampling	0.8510	0.9072	0.8938	0.8980
LogisticRegression	oversampling	0.7876	0.8608	0.8246	0.8432
LogisticRegression	undersampling	0.7678	0.8637	0.8167	0.8432
LogisticRegression	Rose_both	0.7879	0.8606	0.8248	0.8451
LogisticRegression	Smote	NA	NA	NA	NA

We can see these two tendencies in the models:

- Better performance with unsampled data, than with sampled data
- Better performance when we split the positions manually

Our best models

With regards to the business case, we now need to decide which model/models we will use to predict the 2014 NFL Draft. Here you can see the best model/sampling combination for all positions.

Table 2: The best model/sampling combinations by position

Position	Method	Sampling	Accuracy
QB	randomForest	no_sampling	1.0000
WR	randomForest	oversampling	0.9985
RB	randomForest	no_sampling	1.0000
Together	randomForest	oversampling	0.9996

Discussion

The great and sometimes perfect results of the random forest model show a high risk of a massive overfit. The other models with accuracies between 89% and 93% seem to be very good at the first sight. But we have to keep the reason for filtering out the players with <10 played games and sampling the data in mind. Here we applied the models to the unsampled but filtered data, which only contains 12.4% of drafted players. This means, that a model predicting "not drafted" for every player would not perform much worse, since it would still have an accuracy of 87.6%.

Interpretations of models are always quite difficult, but the following thoughts are pretty likely to be true, according to the TP/TN/FP/FN. Our models are pretty good at predicting the likelihood of players being drafted that didn't perform very well in college football being low. It probably also does not too bad in predicting great players to be drafted, that nearly must be picked (and probably are picked early in the draft).

But there is probably much room for improvement for all the players that still performed well in college, and might or might not be drafted.

We would like to close the circle to one of our fist lessons in machine learning, in which we were taught the following very high level formula for models (the right part). Y denotes the true outcome (in our case whether a player is drafted or not), f(X) is the true pattern that describes it, and ϵ is the noise, which appears to be random. With our models (the left part) we try to predict a $\hat{Y}$, which shall be as close as possible to the true Y.

$$\hat{f}(X) = \hat{Y} \approx Y = f(X) + \epsilon$$

Looking at this inequation we can think of three possibilities, why our models make so many mistakes:

- Our models $\hat{f}_i(X)$ do not include enough variables and/or are not sophisticated enough
- The data is not good enough
- The NFL draft contains a pretty large ϵ