Lab 6 - Ensembles

Artificial Intelligence and Pattern Recognition

In this exercise we will use some ensemble processes to enhance performance as well as some models that use ensembles inherently. Most of the new RapidMiner operators that we will use in this exercise can be found under Modeling->Predictive->Ensembles. Most of these operators work by putting one or more models inside that will then be trained individually and when used for prediction the ensemble process will aggregate the predictions of all the models to one final prediction.

> **If you're doing this lab outside the lab session**
> It is recommended that you find one or more of your fellow students to do the lab with in order to discuss your findings. If you have any remaining questions, feel free to ask them in the Discussion on Canvas.

One of the initial concerns one might have about ensembling methods is that it would just lead to getting the average performance of the many models. A quick way to test this is to train one ensemble as well as many models outside the ensemble and then compare the test performance of the ensemble to the average test performance (using the "Average" operator to collect performance averages).

The first form of ensemble that one usually thinks of is simple majority voting. Just train a number of classifiers on the data and then predict the class that has the highest average prediction score.

> **Note**
> The tasks in this exercise build on each other so **don't** reset the RapidMiner process between each task.

Task 1

- Create a training and testing setup for the Sonar dataset (validation can be skipped as we will do no hyper-parameter tuning).
- Create five Decision Tree models in the process that are trained and tested separately and then average the performance.
- Create a Vote-ensemble with five Decisions Trees in it as a model that is trained and tested.
- Inspect the structure of the trained Decision Trees, do you notice anything?
- Slightly vary the parameters between the Decision Trees so that you have five trees with different parameters setups (to make a fair comparison, make sure that the five averaged trees are the same as the five voting trees).

- Compare the performance of the averaged Decision Trees with the voting Decision Trees. What do you notice?
- You might want to try to use a few different seeds to see if the effects are constant.

A problem with training ensembles is that many machine learning models will become very similar if they're trained with the same data. What point is there to vote if every Decision Tree is the same? One method to prevent the models from becoming the same is called Bagging. In Bagging each classifier in the ensemble is trained with only part of the training data (commonly around 80 to 90%, but the best choice will depend on the dataset). This way they won't have all the same data and thus be less likely to become similar. Read about **Bagging on Wikipedia** if you want to understand it better.

Task 2

- Add training and testing of a Bagging-ensemble with five decision trees (selected in the iterations parameter of Bagging) to the process from Task 1.
- What do you notice?

Another popular type of ensemble is Boosting. There are many variants of Boosting, but the general idea is that after one model is trained the data that it performs poorly at is given an increased importance so the next model will train "more" on that data. Through this procedure each model can be made to specialize on a certain part of the data. This is elaborated on in the article about **Boosting on Wikipedia**.

Task 3

- Add training and testing of an AdaBoost-ensemble with five decision trees (selected in the iterations parameter) to the process from Task 2.
- What do you notice?
- Try increasing the number of models in the Bagging and AdaBoost ensembles (to something really high). What do you notice?

A special ensemble technique for Decision Trees has become so popular that it has been given its own name; "Random Forest". The Random Forest operator in RapidMiner behaves like any other model, but under the hood it is actually a form of Bagging for Decision Trees. Read about **Random Forest on Wikipedia** to better understand it, especially the "From bagging to random forests"-section.

Task 4

- Add training and testing of Random Forest to the process from Task 3.
- What do you notice?

Task 5

- Add a slower non-ensemble method to the process (like one of the neural network models).
- How many times slower is the model to train compared with a single Decision Tree?

- How many times slower is it to test?
- Make an ensemble with so many Decision Trees that it takes about as long to train and another that takes about as long to test. Compare their performance, what do you notice?
- Which comparison do you think is more fair?

Task 6

- Read the descriptions of some of the ensemble parameters and try changing them around to see how it affects the performance. It can often be useful to take parameters to extremes to see what impact that has.

Task 7

- Experiment some more with the different models in the ensembles and perhaps different datasets.
- Which ensemble method seems best with simple models and which seems best with complex models?
- Which ensemble method seems best when there are few models in the ensemble (<10) and which seems best when there are many models in the ensemble (>100)?
- Can you find any experts online (perhaps in blogs or openly available books) that have any opinions and arguments for which is better when? Is there a general consensus?
    - It can be helpful to search for terms like “X vs. Y”, “When is X better than Y” or “When to use X?”.
    - Can you find the credentials of the experts answering?