PW_09: Classification Trees and Random Forests

Students

- Liechti Matthieu
- Loup Olivia

Date

17.11.24

Ex 1 : Classification trees

Q1.1: At which frequencies does the electrical activity mostly occur?

As we can observe in the graph below, the moste concetration of electrical activities is situed in low frequencies between 2 and 10 Hz.

Q1.2: Can you easily distinguish between classes visually? What can you say about the inter- vs intra-class variability?

Even though it can be challenging to visually distinguish between the states using the EEG spectra, several differences are apparent: • The consistent pattern of the non-REM sleep (n) spectra, which shows a sharp peak at lower frequencies, suggests that there is little variability within this state (low intra-class variability). • The awake (w) state also exhibits a consistent spectral pattern, like the non-REM sleep, characterized by minimal variability within this condition (low intra-class variability). This consistency in the spectrum of the non-REM sleep state could be used as a visual cue for identification. However, because both states lack notable peaks and retain a lower amplitude across the spectrum, it is difficult to distinguish between the awake state and non-REM sleep based only on visual inspection (high inter-class variability).

Q1.3: Describe both of these criteria. What does a gini impurity of 0 means? What does an entropy of 1 mean?

The Gini impurity is a metric thas shows the likelihood of inaccurate classification with randomly picked items and classifed according to the distribution of the classes in the dataset. The range is form 0 to 0.5, 0.5 correspond to 50/50 split in a binary classification. A gini impurity of 0 means the the set is perfectly homogeneous. All elements blong to the same class.

The Entropy of 1 represents the maximum disorder. it happens when the node contains an equal proportion of both classes. It become completely unpredictable.

Q1.4: This problem suffers from a common issue in machine learning. What is this problem called? What could be its causes? How can it be resolved?

We found an overfitting problem in this system. It happens when the model is too mutch trained. It cpatures the noise and the details that do not generalize to new unseen datas. The causes of this problem are as follows : too many iteration for the training, too complex model or too small data. To solve this kind of problem, we can do some cross validation or just stop training erlier.

Q1.5: Use the visualization of this tree to show and explain:

- What is a node? What is an edge? What is a leaf?
- What are the two additional hyperparameters doing? Do you think that both are necessary in this particular case (min_samples_leaf = 20, max_depth= 4)? Why?
- What does the color of each node represent?

Node: Is a point in the decision tree where the data is split. In this tree, each rectangle represents a node. Each node will test an attribute and split the data according to the outcome of the test. • Edge: An edge is the line that connects one node to another. • Leaf: A leaf is a terminal node that does not split any further. This is where a prediction is made

min_samples_leaf : this is the minimum number of sampels to create a leaf node. It prevents form overfitting. max_depth : It stops the tree growth. It represents the longest path form the root to a leaf. We think that thoses parameters are essential in our case because it prevents overfitting. Specialy the max depth. In general both can be necessary, but their necessity values would usually be determined through a process of experimentation.

The color in each node represents the majority class in that node after the split, with different colors corresponding to different classes. In this case, they play with the intensity of the color to say if there are more or less portions of the class.

Q1.6: Choose one of the nodes. Explain precisely the information given on each line of text in this node.

We choose second last on the right of the tree. The first ligne describe the condition to split data. In our case, if x[14] is lower or equal than 0.04 the tested vleu goes to the left and in the orther case, on the right. (0.14)The second lign is the gini and we already descibe the the answer 1.3. (2061) The sample is juste the number of samples that were thrown at this node. [1911, 15, 135] The fourth lign represent the distribution in all classes of the samples. The last lign shows the majority class among this node, in our case (n).

Q1.7: Does model 2 still have the same problem as model 1? Explain based on the classification reports and the confusion matrices.

Model 2's accuracy is 0.813 on the training set and 0.812 on the test set, both of which are extremely close. Comparing the training and test sets to Model 1, the precision, recall, and f1-score for every class are likewise more consistent. This proves that the Model 2

does not have the same overfitting issue as Model 1 because it performs better when applied to data that has not yet been observed. We can see that while the numbers across the diagonal in the matrix are rather high, they are not flawless.

Q1.8: One of the class seems more difficult to predict than others? Which one? Where could this difficulty come from in your opinion?

The 'r' class shows a much lower precision and recall score. Ithink this issue comme form the imbalanced dataset. Compared to the tow other classes, this one havec only a few samples.

Q1.9: What does this hyperparameter do? Explain giving examples from this dataset.

When there are substantially more instances of one class in a dataset than instances of another class, the class_weight='balanced' hyperparameter is used to handle the class imbalance problem. When 'balanced' is selected, this hyperparameter modifies the

class weights that are inversely proportionate to the input data's frequency of each class. Classes are given weights in such a way that larger classes receive a lower weight and smaller classes receive a higher weight. With this option, even though instances of class 'r' are less common, the model places greater emphasis on correctly classifying them (it will likely multiply the number of values by the amount it needs to equal the others).

Q1.10: Compare result

Pros Model 2 : • Higher accuracy on both the training (0.750) and test (0.749) sets compared to Model 3. • The f1-scores for classes 'n' and 'w' are higher in Model 2 than in Model 3, suggesting better overall performance for these classes. Cons Model 2: • The recall for class 'r' is significantly lower than in Model 3, demonstrating that Model 2 misses a larger number of actual 'r' instances. Like seen in the precedent exercices. Pros Model 3: • Significantly improved recall for class 'r' on both training and test sets Cons Model 3: • Lower accuracy overall on both training and test sets compared to Model 2

Model 2 is generally more accurate and precise, but it struggles with recall for the minority class 'r'. Model 3, with balanced class weights, sacrifices overall accuracy and precision to improve the recall for the minority class 'r'.

Ex 2 : Random forests

Q2.1: For each of the hyperparameter: Is there a range of value giving particularly good results? Or particularly bad results?

- **'max_depth'**:
  - Good result : [0;10]
  - Bad result : [30;50]
- **'max_features'**: (some similarities with all value)
  - Good result : > 0
  - Bad result :[0]
- **'min_samples_leaf'**: (some similarities with all value)
  - Good result : >15
  - Bad result : [>5;<15]
- **'n_estimators'**: (Have some bad results at some few points)
  - Good result : > 15
  - Bad result : < 15

Q2.2: These representations give valuable information about hyperparameters. It is nevertheless insufficient. What are/is the main problem(s) with those graphs in your opinion?

In two case, these reprensentations don't give enough differences between results. With this, it's difficult to see the real bests parameters.

Q2.3: What do the following plots represent?

It's the mean test score (mean of number of good value predict) depend of the value and hyperparameter type.

Q2.4: What do the white spots (=empty spots) in the heatmaps mean?

The result fewer than the representation on the array.

Q2.5: How do those plots address the limitations of the previous visualizations?

For the most visualisable values.

Q2.6: What is grid search? Explain by giving real examples from this specific task.

Grid search is used to try out different parameter/hyperparameter to choose the best one or the best combinations of parameters (if more than one).
In example with this case, we have differents hyperparameter : 'max_depth', 'max_features','n_estimators', 'min_samples_leaf'. We should test all combinations, to find best fit of values to create model.

Q2.7: Use the plots above to narrow the range of hyperparameters you want to explore. Which values did you choose to test for each parameter? Justify your choices.

the plots above "*Pair-wise Comparison of Hyperparameters*"show the result of mean test score depend of hyperparameter. This show us than they have more good result :

- 'max_depth': better around [3;7]
- 'min_samples_leaf': better around [21;24] (less obvious than 'max_depth')
- 'max_features' : depend most of other hyperparameters, with 'n_estimators'and 'min_samples_leaf', this is better around [15;20] (no difference with 'max_depth')
- 'n_estimators' : 181 (no difference with 'n_estimators', good result with this value and 'max_features' or 'min_samples_leaf')

- 'min_samples_leaf': [1,18,19] (not big stability of good value)

Q2.8: Which value did you choose for each hyperparameter?

The values of hyperparameters are :

- **'max_depth'**: 7
- **'max_features'**: 23
- **'min_samples_leaf'**: 18
- **'n_estimators'**: 181

This values depend of the graphics of the pair-wire comparison of hyperparmeters and the results of the previous part after several test.

Q2.10: The test set should be used only at this stage, and it is theoretically important not to change the hyperparameters based on the performance on the test set. Why?

Because this parameters depend also of training set, change parameters may not correpsond with previous dataset. It must be conciderate like biais to change on the test set.

Q2.11: Comment your results. -> How well does the model generalize on unseen data? Is a random forest better than a single classification tree in this case? What is the main challenge of this dataset? ...

The features could have an impact on result because of resut of mesurable proprety.
The resutl give :

- **Accuracy**: 93.213%
- **F1 score**: 93.461%

	precision	recall	f1-score	support
Wake	0.94	0.94	0.94	1376
NREM	0.65	0.90	0.76	271
REM	0.97	0.93	0.95	2405
accuracy			0.93	4052
macro avg	0.86	0.92	0.88	4052
weighted avg	0.94	0.93	0.93	4052

With the result on F1-score and accuracy, the model give a good result. The main challenge is the number of hyperparameters and the number of good values. It could take time to find the good ones because there were severall values of hyperparameters seems corrects before.
The most good result are with the REM classes and the worst with the NREM classes.
In this case, the random forest is better because the score of the test set is better than classification tree.

Q2.11: How is this importance calculated?

they are computed as the mean and standard deviation of accumulation of the impurity decrease within each tree.

Q2.12: What can you conclude from this graph?

In this graph, the highter the result are, the best is for the model. In this case, the best result seems to be with the 17 firsts features in depend of the features between 8 and 12 than are lower.

Ex 3 : Gradient boosting

Q3.1: Two additional hyperparameters were added compared to the RandomForestClassifier. What are these hyperparameters, and what roles do they play?

The two nes hyperparameters are: learning_rate and subsample. The learning_rate parameter scales the contribution of each tree to the final model. It essentially controls the step size at each iteration while moving toward a minimum of a loss function. The subsample parameter define the fraction of the used data. the datas are selected randomly. It prevent overfitting and increase the training speed.

Q3.2: Comment the results. Compare these results with the ones obtained with the RandomForestClassifier. Compare more specifically the precision, the recall and the f1-score of the 'r' class obtained with GradientBoostingClassifier and RandomForestClassifier. What are your conclusions?

For The results, we obtain a slightly less preformace with the global accuracy of the gradient Boosting on the test samples. GB : 91% and RF : 93% . When we look at

the result for the 'r' class, we have the following reslults. GB : Precision : 75%, Recall : 41%, f1-score : 53% RF : Precision : 65%, Recall : 90%, f1-score : 76%

In conculison, we can say that Gradient Boosting shows itself to be a strong model with better overall performance, as evidenced by more balanced metrics across classes and improved accuracy. This dominance is particularly noticeable when it comes to how well it predicts instances of class "r." In this situation, precision means that the model identifies 'r' instances correctly out of all instances that it labels as 'r'. Recall, a metric that assesses the model's capacity to recognize every real instance of class "r," is sacrificed in order to achieve this high precision. In this case, Random Forest outperforms Gradient Boosting, demonstrating a stronger recall for class "r." This means that Random Forest labels more non-'r' instances as 'r', in fact we have a higher recall. Choosing the right model become a prioritizing question around the recall. It depends on the specifc requirements.