Problem Set 08: Decision Trees

Johannes C. B. Dietschreit, Sascha Mausenberger

Due: 2.12.2024

The problems below are meant to be solved using Python functions and libraries. You can do so by either writing short scripts or using Jupyter notebooks. **Report your results in the Quiz on Moodle** to obtain a grade.

Introduction

In this PSet you will compare single decision trees and ensembles of trees, also called Random Forests. Using trees and forests we will take a closer look at feature importance analysis. Note, that while we are using sklearn functions in this problem set, many researchers rely on SHAP feature importance analysis, which is based on game theory and which we won't cover in this course.

The Problems

1 Decision Trees

In this exercise we will use another of the standard scikit learn data sets, the breast cancer set. We will learn decision trees and discuss how to evaluate feature importance.
Load the breast cancer data set using the following code

```
from sklearn.datasets import load_breast_cancer
breast_cancer = load_breast_cancer()
X, y = breast_cancer.data, breast_cancer.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=42)
```

1.1 Train a Decision Tree Classifier with the settings

```
clf = tree.DecisionTreeClassifier(criterion="gini",
                                  max_depth=3,
                                  random_state=42)
# visualize the model with
dot_data = tree.export_graphviz(clf, out_file=None)
graph = graphviz.Source(dot_data)
# or render the tree using
graph.render("DTC_breastcancer")
```

1.1.a) How many different features were used for decisions in the tree? (0.5 P)

1.1.b) Out of the train samples, how many patients tested positive for cancer? You can, e.g. read the number off the visualization. (0.5 P)

1.1.c) How many leaves did not manage to fully separate the classes? (0.5 P)

```
# A leave does not have a left and right child
is_leaf = (clf.tree_.children_left == -1) & (clf.tree_.
    children_right == -1)
# Leaves can then be selected with
clf.tree_.impurity[is_leaf]
```

1.1.d) To what do you need to set `max_depth` to achieve pure leaves (while keeping `random_state=42`)? (0.5 P)

1.1.e) Which model has the better accuracy, the decision tree of depth 3 or the one that could separate or training samples? (0.5 P)

- clearly depth 3
- almost the same, but depth 3 shows slightly better generalization
- their performance on the test set is identical
- almost the same, but the deeper tree can get a few more samples right
- clearly the deeper tree

1.2 We will use the model with depth 3. When inspecting the feature importance (an attribute of decision trees), which are the features that have a predicted importance above 5%? (1 P)
The feature names can be accessed with *breast_cancer.feature_names*

1.3 The importance of a feature is computed as the total reduction of the criterion (in our case "gini impurity") brought by that feature. Hence, this would be the Gini importance of a feature. However, impurity-based feature importances can be misleading. Therefore, we will test a more general strategy that can be applied to any architecture, not just decision trees, namely "perturbation importance". The idea is, feature by feature we break the connection between the feature and the label, by randomly shuffling this particular feature and test how the model performs. If a feature is irrelevant, then this shuffling will have no effect. If the feature is important, then the shuffling will lead to a completely deteriorated model performance. Use:

```
from sklearn.inspection import permutation_importance
r = permutation_importance(clf, X_test, y_test,
                           n_repeats=50,
                           random_state=0,
                           scoring='accuracy')
```

what feature is now considered the most important? (0.5 P)

1.4 While our model's accuracy is relatively good (accuracy = rate of correct prediction), it might be more important to look at other metrics. One can argue that it is more important to correctly identify all cancerous cells than to get all those correct that do not have cancer, i.e., a low false negative rate (recall) is more important than a high positive predictive power (precision).

Compare the feature importance for each of the three scores (accuracy, recall, precision). You will notice that the value of the feature importance is scoring dependent. Hint: You can pass a list of strings to the ‘scoring‘ argument.

Which of the following statements is true: (1 P)

- The most important 4 features are the same for all three scores.
- "worst radius" is always among the top 2 features.
- Accuracy and recall has a clear top 2, whereas precision does not.

- "area error" is always among the top 4 features.
- The feature importance values for precision are the largest.

1.5 Previously in the lecture we have seen that Support Vector Machines are very powerful, kernel-based classifiers. Train two SVC models, one with a linear the other with an rbf kernel. Use again `permutation_importance` to perform a feature analysis. Which of the following statements are true: (1 P)

- The features that were the most important for the decision tree are also important to the SVC models.
- One can identify a top 2 for the linear SVC model, as for the decision tree (though it's a different top 2).
- The top 1 for the rbf kernel is among the top 3 for the linear kernel.
- The top 1 for the rbf kernel is not among the top features of the decision tree.
- The rbf kernel model relies on less features than the linear model.

2 Random Forests

Ensembles of models can be better than a single model, which is a concept that is even true for deep learning. And even more important for decision trees. Training a set of different decision trees is called a random forest. The expectation is that the forest is better than a single tree (it is also more expensive to train and when performing inference). Make sure to use:

```
RandomForestClassifier(n_estimators = XXX, criterion="gini",random_state=42,
   max_depth=3)
```

2.1 The question is how many trees do we need. The more the better, however, the more the more expensive. Since we are striving for perfect recall, we should monitor how it improves with the number of trees in our estimator. How many trees do I need until I do not see any further improvement w.r.t. recall on the test set? (1 P)

2.2 The question is now, what features does the ensemble rely on? Are those closer to the single tree studied before or rather identical to the SVM models? Train the following model

```
RandomForestClassifier(n_estimators = 25,
                       criterion="gini",
                       max_depth = 3,
                       random_state=42)
```

and evaluate its feature importance using accuracy or precision.

Looking at the top features, which of the following statements are true: (1 P)

- The Forest has the same top 2 features as the tree.
- The feature importance values are all very small
- The forest seems to rely on different features than the tree or the linear SVC.
- The forest has the same top 2 features as the linear SVC model.
- "mean area" is the only feature with an importance larger than 2-times its standard deviation.

2.3 To understand why we received those feature importances. Code a loop that does the following. First select randomly only one, then two, then three, ..., then finally all features. Shuffle all entries of the randomly selected features (only within that specific feature) and compute the accuracy.

Repeat this 50 times and take the average over the repetitions. Plot the averaged values vs the number of features you shuffled. The last point is basically an average of giving the forest random numbers for all features.

What do you observe, select the correct answer: (2 P)

- The forest is biased towards predicting bening cells.
- One can shuffle 5-6 features without the forests accuracy changing much.
- When presented with completely randomized features, the forest's accuracy is equal to just predicting the more probable label.
- The forest gets monotonously worse with each feature that is shuffled.
- When all features are shuffled, the forest performs worse than if one had flipped a coin (chosen randomly).