Lab 3 – Trees

Question 1 – Decision Trees

Let us use decision trees on the breast cancer dataset. Again, use stratified sampling.

1. Train the model sklearn.tree.DecisionTreeClassifier with random_state=0 and compute the scores on the training and test sets.
2. Visualize the tree.

```
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
plt.figure() #parameters: figsize, dpi
plot_tree(clf) #parameters: filled, class_names, feature_names
plt.show()
```

3. Repeat the previous steps (score and visualization), with a maximum depth of 3 (max_depth=3). What do you observe? What happens if you change the value of the min_samples_leaf or min_samples_split?
4. Instead of looking at the tree, which can be a bit difficult, we can consider other properties, such as *feature importance* in the decision taken by the tree: from 0 meaning "not used" to 1 meaning "perfectly predicts the class". The sum of feature importances sums to 1. Observe these values with:

```
<model>.feature_importances_
```

5. Visualize feature importance with the following code.

```
n_features = cancer.data.shape[1]
plt.barh(range(n_features), tree.feature_importances_,
align='center')
plt.yticks(np.arange(n_features), cancer.feature_names)
plt.xlabel("Feature Importance")
plt.ylabel("Feature")
plt.ylim(-1, n_features)
```

   A low importance does not necessarily mean that the feature is not useful or informative, rather it means that the feature was not used in this particular model (which can be the case when some other feature represents the same information).
6. In order to be able to visualize the decision boundary (in 2 dimensions), build a new decision tree model based on the 2 most important features identified previously. Draw the decision boundaries for trees with depth 2 to 6.

Question 2 – Bagging and Boosting

Let us use random forests and a couple of techniques boosting on the breast cancer dataset. Again, use stratified sampling.

1. Train the model sklearn.ensemble.RandomForestClassifier containing 100 trees (n_estimators=100) with random_state=0 and compute the scores on the training and test sets.
2. Visualize feature importance. What are the similarities and differences compared to a single tree?
3. Now try sklearn.ensemble.AdaBoostClassifier and sklearn.ensemble.GradientBoostingClassifier.