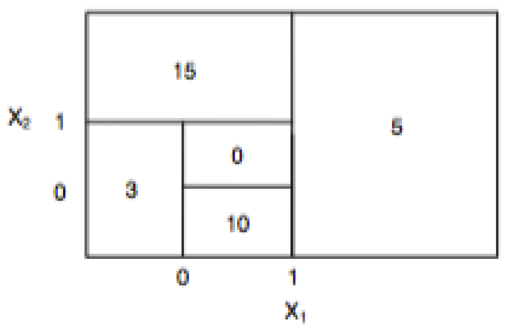
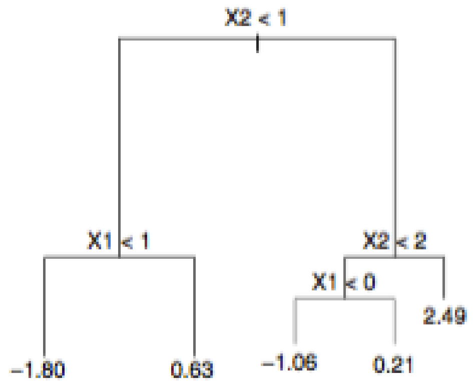
ISEP
Machine Learning
October 7th 2024

Lab 3 : Decision trees and Support Vector Machines (SVM)

1 Part I : Exercises

Exercise 1. Consider the following figures :





a) Sketch the tree corresponding to the partition of the predictor space illustrated in the left-hand panel of the Figure. The numbers inside the boxes indicate the mean of Y within each region, that is $\hat{y}_m$ for region m.

b) Consider the tree in the right-hand panel of the figure, create the diagram of the predictor space partition (similar to that of the left-hand panel). You should also indicate the mean response value for each region.

Exercise 2. Charlie wants to predict if he will pass or not the machine learning final exam based on his results in previous exams. Up to now there have been 9 exams. He considers two features : "*whether to stay up late the day before the exam or not* (S)" and " "*whether to attend the session or not*" (A). We considered, there are 9 sessions, one per each exam. Consider the following information :

— He passed 5 exams out of 9.

— He studied late the day before for 5 exams, among these 5 he passed 3 exams and failed 2 exams. In contrast, the times he went to bed early, he passed the two exams and failed the others.

— He attended 6 sessions out of 9. The times he attended he succeeded the exam 5 times and the times he missed the session he failed all the exams.

1. He decides to use decision trees to fit his model. Which feature would you choose at the top most node to split the data, S or A ? Why ? Build the decision tree for both cases. For both trees calculate the entropy (deviance) at each level and deduce the information gain (decrease of entropy from one level to he next one). You may use the following approximations :

N	3	5	7
$\log_2 N$	1.58	2.32	2.81

2. Answer the previous question using the Gini index instead of entropy. Compare the results.

Exercise 3.* A French Region wants to understand the investment differences in several towns and villages. To tackle this problem, several criteria were considered :

— Existence of a Train Station : yes/no
— Status : Town/Village
— Distance to Paris : close/far
— Investment level : small/medium/high

Build a classification tree using the following data :

Station	Status	Distance	Investment
No	Village	Close	Medium
Yes	Town	Close	High
No	Village	Far	Small
Yes	Village	Close	High
No	Town	Far	Small
Yes	Town	Far	Medium
No	Town	Close	high

If a village has a train station and it is situated far away from Paris, predict the investment level.

Exercise 4. We mentioned in the lecture that when performing bagging, on average, each bagged tree makes use of around $\frac{2}{3}$ of the observations. The purpose of this exercise is to prove this result by calculating the probability that a given observation is part of a bootstrap sample.

Suppose that we obtain a bootstrap sample from a set of n observations.

a) What is the probability that the first bootstrap observation is *not* the jth observation from the original sample? What is the probability that neither the first one nor the second bootstrap selected observation is the jth observation from the original sample? Justify your answer.

b) Argue that the probability that the jth observation is not in the bootstrap sample is $\left(1-\frac{1}{n}\right)^n$.

c) When $n = 5$, what is the probability that the jth observation is in the bootstrap sample? What about $n = 100$ and $n = 10000$?

d) Create a plot that displays, for each integer value of n from 1 to 100000, the probability that the jth observation is in the bootstrap sample. In Python you can run the following code :

```
import matplotlib.pyplot as plt
import numpy as np
x = range(1, 100000)
p =(1 - (1-np.divide(1,x))**x)
plt.ylim(0, 1)
plt.plot(x,p,'.',color="black")
```

Comment on what you observe.

e) We will now investigate numerically the probability that a bootstrap sample of size $n = 100$ contains the jth observation. Here $j = 4$. We repeatedly create bootstrap samples, and each time we record whether or not the fourth observation is contained in the bootstrap sample. You can execute the following code in Python :

```
from random import choices
B=10000
```

```
sum4=0
for b in range(1,B+1):
    bsample=np.random.choice(range(1,101),100)
    if np.count_nonzero(bsample==4)>0:
        sum4+=1
sum4/=B
print(sum4)
```

Execute the code. On average, how often the observation 4 is sampled ?

Exercise 5. Here you will explore the maximal margin classifier on a toy data set.

1. Consider the following observations in $p = 2$ dimensions. and a class label variable.

Obs	X_1	X_2	Y
1	3	4	Red
2	2	2	Red
3	4	4	Red
4	1	4	Red
5	2	1	Blue
6	4	3	Blue
7	4	1	Blue

   Sketch the observations.

2. Sketch the optimal separating hyperplane, and provide its equation.
3. Describe the classification rule for the maximal margin classifier.
4. On your sketch, indicate the margin for the maximal margin hyperplane.
5. Indicate the support vectors for the maximal margin classifier.
6. Argue that a slight movement of the seventh observation would not affect the maximal margin hyperplane.

2 Part II : Practical applications graded !

Do you remember the `bankrupt` data set studied during the Lab 2 ? In this part, you are going to fit decision trees and SVMs in order to predict whether a company goes bankrupt or not as a function of financial features.

[graded question] First of all, use the *Pandas* library to import the file *bankrupt.txt* to an object called `bankrupt`. You can use the command `bankrupt["Bankrupt?"].describe()` to calculate descriptive statistics for the target variable. Comment on the results. What is the percentage of companies that went bankrupt ? Is the data set balanced ? You can use the command `value_counts()`.

You will need to import the files `x_train.csv`, `x_test.csv`, `y_train.csv` and `y_test.csv`. These files contain the train and test set respectively. Import these data sets. Standardize the feature variables using the `StandardScaler()` function from the `preprocessing` library of `sklearn`. Create a new dataframe of the standardized variables contained in `x_train` and in `x_test`. Verify that the new dataframes have the same length as the original ones.

2.1 Classification Trees

Now you will use classification trees to predict whether a given company will go bankrupt or not. You will need to import the class `tree` from `sklearn`. To fit a classification tree you will use the function `DecisionTreeClassifier()`. For instance to fit a classification tree of maximum depth 3 you can run the code :

```
from sklearn import tree
class_tree_bankrupt=tree.DecisionTreeClassifier(max_depth=3)
class_tree_bankrupt.fit(x_train,y_train)
```

You can visualize the tree with the `plot_tree` function.

```
import matplotlib.pyplot as plt
plt.figure(figsize=(20,20))
tree.plot_tree(class_tree_bankrupt,feature_names=x_train.columns,filled=True,class_names= True)
```

1. [*graded question*] Plot the tree and answer the following questions :
   (a) Each node of the plotted tree contains information about the fitted tree. Interpret the information for the topmost node of the tree. How was the value of the Gini index calculated ?
   (b) How many terminal nodes does the tree have ? This information can also be obtained with the method `get_n_leaves()]`.
   (c) Make a prediction for a company with a Working Capital to Total Assets (WKTA) equal to -1.73, a Debt ratio (DR) of about 50% and a Net Income to total assets (NITA) equal to 0.75 (all the values were standardized).
2. [*graded question*] Use the function `predict()` to make predictions for the train and for the test set. Calculate the confusion matrix and the classification report. You will need to import the functions `confusion_matrix` and `classification_report` from `sklearn.metrics`. Compare the metrics and conclude if there is overfitting. If so, propose a solution (without calculating) to reduce the overfitting.

Now you will build a very large tree and then use the *minimal cost-complexity pruning* algorithm to prune it. As seen in the lecture this technique has a parameter α which controls the size of the tree, that is, great values of α increase the number of pruned nodes. Follow the steps :

a) The method `cost_complexity_pruning_path` returns the effective values of α and the corresponding total leaf impurities at each step of the pruning process. To get the pruning path run the following code :

```
prun_tree_bankrupt = tree.DecisionTreeClassifier(random_state=10)
path = prun_tree_bankrupt.cost_complexity_pruning_path(x_train, y_train)
alphas, impurities = path.ccp_alphas, path.impurities
```

Warning ! Always use the indicated value of the parameter `random_state` to reproduce the same outputs.

b) Now, fit a decision tree for each value of α :

```
class_tree_bankrupt_list = []
for alpha in alphas:
    clf = tree.DecisionTreeClassifier(random_state=0, ccp_alpha=alpha)
    clf.fit(x_train, y_train)
    class_tree_bankrupt_list.append(clf)
```

The list `class_tree_bankrupt_list` contains the fitted trees for each value of α. The last element in this list corresponds to the trivial classifier with only one node. So, remove it from the list as well as the last element of the list `alphas`.

```
class_tree_bankrupt_list = class_tree_bankrupt_list[:-1]
alphas = alphas[:-1]
```

c) [*graded question*] Now, you are going to choose the best value on α based on the *balanced accuracy score*. You will need to import the `balanced_accuracy_score` function from the `sklearn.metrics` library. Calculate the balanced accuracy for each fitted tree in the train and the test set. Plot the train and test balanced accuracy versus the parameter α. Comment on the graphic and choose the best value of α. How many leaves does the chosen tree have ?

d) [*graded question*] Calculate the confusion matrix and the classification report and interpret the performance metrics for the chosen classifier for the train and test data sets.

2.2 Ensemble methods : Bagging, Random forest and Boosting

In this part you will use the ensemble learning methods treated in the lecture in order to predict if the company goes bankrupt or not. First of all, you will need to import the suitable packages `RandomForestClassifier` and `AdaBoostClassifier` from `sklearn.ensemble`. For all the ensemble methods set the parameter `random_state =1`. This will allow to get the same results.

1. [*graded question*] Recall that bagging is a special case of a random forest where the number of randomly selected predictors m is equal to the full set of predictors p. Perform bagging to the `bankrupt` data set with the code :

```
bagging_bankrupt = RandomForestClassifier(max_features = 12, random_state = 1)
bagging_bankrupt.fit(x_train,np.ravel(y_train))
```

   Calculate the confusion matrix and the classification report and interpret the results.

2. [*graded question*] Modify the code of the previous question in order to perform random forests with $m = \sqrt{p}$. You will to set the parameter `max_features = 'sqrt'`. Calculate the confusion matrix and the classification report and interpret the results. In there any improvement in comparison to the bagging method ?

3. [*graded question*] Perform Adaptive boosting (Adaboost) setting the maximum number of estimators to 4. Calculate the confusion matrix and the classification report and interpret the results.

4. [*graded question*] Although ensemble methods lose in interpretability in comparison to a single decision tree, it is possible to measure the importance of each predictor. This information is stored in the attribute `feature_importances_`. For instance, for the bagging classifier you can obtain the variable importance with the command `bagging_bankrupt.feature_importances_`. Calculate the variable importance for the three classifiers (bagging, random classifier and Adaboost). Interpret and comment on the results. In addition you can make a barplot chart of the variable importance :

```
print(bagging_bankrupt.feature_importances_)
Importance = pandas.DataFrame({'Importance':bagging_bankrupt.feature_importances_*100},
index = x_train.columns)
Importance.sort_values(by = 'Importance', axis = 0,ascending = True).plot(kind = 'barh')
plt.xlabel('Variable␣Importance')
```

2.3 Support Vector Machines (SVM)

In this section you will use SVMs to predict bankruptcy. Previously you will need to import the `SVC` module from the `sklearn.svm` package. For instance, to fit a support vector classifier with a linear kernel you can use the code :

```
svm_bankrupt = SVC(C=1, kernel='linear')
svm_bankrupt.fit(x_train, np.ravel(y_train))
```

The cost parameter `C` of the `SVC()` function is inversely proportional to the tuning parameter defined in the lecture. It allows to specify the penalty of a violation to the margin. When the cost argument is small, then the margins will be wide and many support vectors will be on the margin or will violate the margin. When the penalty argument is large, then the margins will be narrow and there will be few support vectors on the margin or violating the margin.

1. [*graded question*] You will perform cross-validation with the function `GridSearchCV()` of the module `sklearn.model_selection` in order to select the best value of the parameter C. The value of the parameter `scoring` of the function `GridSearchCV()` denotes the metric used to evaluate the performance of the model. You will use the *balanced accuracy* as performance metric. What is the formula of the balanced accuracy ? When is it preferable to consider the balanced accuracy instead of the overall accuracy ?
2. [*graded question*] The following code allows to perform 10-fold cross-validation for the following range of values of the cost parameter [0.001, 0.01, 0.1, 1, 5, 10] (running this code might take some time) :

   ```
   from sklearn.model_selection import GridSearchCV
   tuned_parameters = [{'C': [0.001, 0.01, 0.1, 1, 5, 10]}]
   svm_bankrupt_CV = GridSearchCV(SVC(kernel='linear'), tuned_parameters, cv=10, scoring='balanced_accuracy')
   svm_bankrupt_CV.fit(x_train, np.ravel(y_train))
   ```

   The best parameter value can be obtained with command `svm_bankrupt_CV.best_params_`. What is the best value of C in terms of accuracy ? The command `svm_bankrupt.support_` outputs a list of the support vectors. How many support vectors does the model have ?
3. [*graded question*] Calculate the confusion matrix and the classification report and comment on the results. Plot the ROC curve of the linear SVM model, which value of classification threshold would you choose to get a good compromise for the two classes ? Calculate the classification report for the chosen threshold and comment on the results.

In some situations the data can not be linearly separable. Then, it is necessary to fit SVM with non-linear kernels. For instance, consider the following simulated data :

```
np.random.seed(8)
X = np.random.randn(200,2)
X[:100] = X[:100] +2
X[101:150] = X[101:150] -2
Y = np.concatenate([np.repeat(-1, 150), np.repeat(1,50)])
plt.scatter(X[:,0], X[:,1], s=70, c=Y, cmap=plt.cm.Paired)
plt.xlabel('X1')
plt.ylabel('X2')
```

One can easily deduce from the plot that the best choice is to use a radial kernel :

```
svm_radial = SVC(C=1.0, kernel='rbf', gamma=1)
svm_radial.fit(X, Y)
plt.scatter(X[:,0], X[:,1], s=70, c=Y, cmap=plt.cm.Paired)
plt.xlabel('X1')
```

```
plt.ylabel('X2')
h=0.02
pad=0.25
x_min, x_max = X[:, 0].min()-pad, X[:, 0].max()+pad
y_min, y_max = X[:, 1].min()-pad, X[:, 1].max()+pad
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = svm_radial.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
plt.contourf(xx, yy, Z, cmap=plt.cm.Paired, alpha=0.2)
```

4. [*graded question*] Fit an SVM with a radial kernel. You will need to specify the value of the parameter γ. Perform 10-fold-CV in order to select the best values of the tuning parameters C and γ. You can use the following code :

   ```
   tuned_parameters = [{'C': [0.01, 0.1, 1, 10, 100], 'gamma': [0.5, 1,2,3,4]}]
   svm_radial_bankrupt_CV = GridSearchCV(SVC(kernel='rbf'), tuned_parameters, cv=10, scoring='balanced_accuracy')
   svm_radial_bankrupt_CV.fit(x_train, np.ravel(y_train))
   svm_radial_bankrupt_CV.best_params_
   ```

   What are the best values of C and γ? Fit the SVM classifier to the train set with the chosen value of C and γ. Calculate performance metrics (confusion matrix and classification report). Comment on the results. Plot the ROC curve and find out if there is a classification threshold that would allow to get a good compromise for the two classes. If so, calculate the classification report for the chosen threshold and comment on the results.

2.4 Conclusion

[*graded question*] Make a summary table with the performance metrics of all the classifiers fitted in this lab and make your general conclusions.

References

— UCI Machine Learning Repository (2020) "*Taiwanese Bankruptcy Prediction*". `https://doi.org/10.24432/C5004D`. Consulted on September 15th, 2023.

— James, Gareth ; Witten, Daniela ; Hastie, Trevor and Tibshirani, Robert. "An Introduction to Statistical Learning with Applications in R", 2nd edition, New York : "Springer texts in statistics", 2021. Site web : `https://hastie.su.domains/ISLR2/ISLRv2_website.pdf`.

— J Crouser : "SDS 293 - Machine Learning labs". `http://www.science.smith.edu/~jcrouser/SDS293/`. Visited on October 18th.

— UCI Machine Learning Repository (2020) "*Taiwanese Bankruptcy Prediction*". `https://doi.org/10.24432/C5004D`. Consulted on September 15th, 2023.