LAB 2: Classification

Assignment 1 In this assignment you will experiment with five basic classification models from machine learning and statistical learning. The models are provided in the table below together with their names in scikit-learn:

Model Name	Scikit-learn name	Scikit-learn module
Decision Trees	`sklearn.tree`	`DecisionTreeClassifier()` *`[criterion='entropy']`*
Nearest Neighbor	`sklearn.neighbors`	`KNeighborsClassifier()`
Gaussian Naïve Bayes	`klearn.naive_bayes`	`GaussianNB()`
Logistic Regression	`sklearn.linear_model`	`LogisticRegression()`
Support Vector Machines	`sklearn.svm`	`SVC` *`[kernel='linear']`*

Table 1: Five basic classification models. Note that: (1) decision trees in our experiments employ entropy measure for data purity (indicated with *[criterion='entropy']*), and (2) the support vector machine in our experiments is a linear support vector machine (indicated with *[kernel='linear']*).

The classification models will be applied on nine data sets. The data sets are separated into three clusters:

Cluster	Data Sets

Experiment 1 data sets	`exp1a.csv`
	`exp1b.csv`
	`exp1c.csv`

Experiment 2 data sets	`exp2a.csv` `exp2b.csv`

Experiment 3 data sets	`exp3a.csv`
	`exp3b.csv`
	`exp3c.csv`
	`exp3d.csv`

Table 2: Data Sets in three experimental clusters

The data sets between the clusters differ in terms of *class separability, class balances, input variable dependencies, input variable noise, and redundancy*. A detailed description can be found in Appendix: Data.

To estimate the generalization performance of the classification models you can use accuracy rate that can be estimated using the training data and cross validation. Possible scenarios how this can be implemented in Python are provided in Appendix: Estimating Accuracy Rates.

Assignment 1.1 Run the Nearest Neighbor classifier for the parameter *k* equal to 1, 5, 11, and 21, and record 10-fold cross-validation accuracy rate and confusion matrices for the data sets from Experiment 1. (Note that the parameter *k* is the parameter *n_neighbors* in sklearn.neighbors.KNeighborsClassifier().)

(a) Visualize the data sets from the Experiment 1 cluster (they are given with two input numeric variables).
(b) For each of the data sets, how does the 10-fold cross-validation accuracy rates and confusion matrices vary as k increases? Explain this trend.
(c) For each k Nearest Neighbor classifier (k in {1, 5, 11, 21}), how does the 10-fold cross-validation accuracy rates and confusion matrices vary over the three data sets? Explain this trend.
(d) Repeat (b) and (c) for the Nearest Neighbor classifier that employs distance weighting by setting option weights ='distance'. Comment on the change in the generalization performance of the classifiers from the unweighted version for the same values of k and the reason for this change.
(e) Run decision trees and logistic regression classifiers on the three data sets from the Experiment 1 cluster, and compare their generalization performance with that of the various k-NN classifiers from (b), (c), and (d). Explain the reasons for the observed differences in the generalization performance. Visualize the errors of the decision trees and logistic regression classifiers on the data sets from the Experiment 1 cluster.

Assignment 1.2 Run all the five classifiers from Table 1 on the `exp2a.csv` data from the Experiment 2 cluster (see Table 2), and report 10-fold cross-validation accuracy rates and confusion matrices obtained.

(a) What classifiers have a bad generalization performance and what classifiers have a good generalization performance? Comment on the collective characteristics of the variables in the `exp2a.csv` data that lead to such performance.
(b) Can you improve the classifiers from (a) that have a bad generalization performance by changing the parameters when training those classifiers?

Run all the five classifiers from Table 1 on the exp2b.csv data from the Experiment 2 cluster (see Table 2), and report 10-fold cross-validation accuracy rates and confusion matrices obtained.

(c) Which classifier shows the *biggest* drop of the generalization performance compared to the exp2a.csv data, and what is the reason for this drop?
(d) Which classifier shows the *smallest* drop of the generalization performance compared to the exp2a.csv data, and what is the reason for this drop compared with the classifier identified in (c)?

1.3 Assignment Run Gaussian Naive Bayes, Nearest Neighbor (k in {1, 5}) and decision tree on all the datasets from the Experiment 3 cluster (see Table 2) and record 10-fold cross-validation accuracy rates obtained.

(a) How does the performance of the different classifiers compare for each dataset?

(b) How does the performance of each classifier vary as the number of variables is increased from datasets exp3a.csv to exp3d.csv? Which properties of the classifiers lead to such a variation in performance?

Assignment 2. (Extra) To illustrate the problem of model overfitting, generate a two-dimensional dataset containing 1500 labeled instances, each of which is assigned to one of two classes, 0 or 1. Instances from each class have to be generated as follows:

- Instances from class 1 are generated from a mixture of 3 Gaussian distributions, centered at [6,14], [10,6], and [14 14], respectively.
- Instances from class 0 are generated from a uniform distribution in a square region, whose sides have a length equals to 20.

Plot the generated data so that: all instances from class 1 are shown in red while those from class 0 are shown in **black**.

Reserve 80% of the labeled data for training and the remaining 20% for testing. Then fit decision trees of different maximum depths (from 2 to 50 with step of 2) to the training set and plot their respective accuracy rates when applied to the training and test sets.

(a) Identify on the plot the region of underfitting, overfitting and optimality.

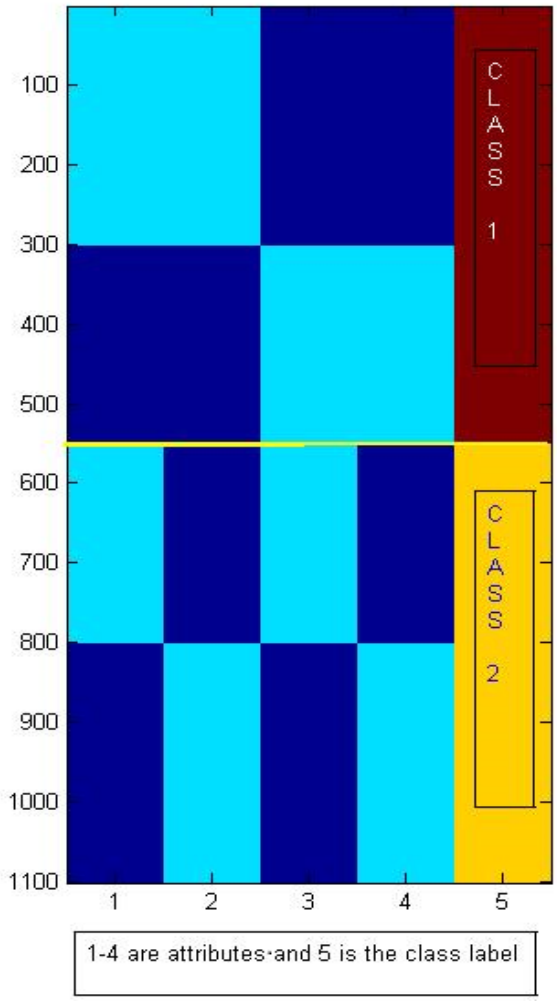
(b) Repeat the assignment for nearest neighbor (k from 1 to 101 with step of 2), and support vector machines for (C from 0.01 to 2 with step of 0.01).

Report. Submit a report with your answers to the assignments (2-6 pages).

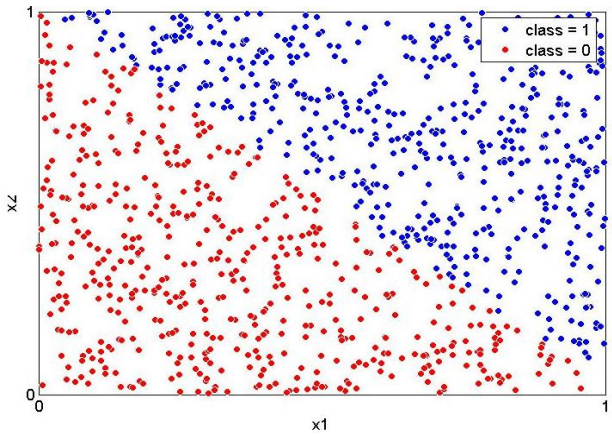
Appendix: Data

Data Cluster	Data Sets	Description

Experiment 1 data sets	`exp1a.csv`
	`exp1b.csv`
	`exp1c.csv`
		The data sets are given with two input numeric variables *X* and *Y*, and one class discrete variable '*class*' with two values 1 and -1. The "1" class consists of normally distributed data points in the rectangle determined by points (-1,-1), (-1,1), (1,1),(1,-1). The class "-1" consists of normally distributed data points outside that rectangle. The `exp1a.csv` set consists of 200 data points for each class. The `exp1b.csv` set consists of 100 data points for class "1" and 200 data points for class "-1". The `exp1c.csv` set consists of 20 data points for class "1" and 200 data points for class "-1".

Experiment 2 data sets	`exp2a.csv` `exp2b.csv`	 The exp2a.csv set is given on the diagram. . The exp2b.csv set includes the 4 variables in the exp2a.csv set and 95 additional noisy variables for the same set of instances. The values for the noisy variables are randomly assigned from the uniform distribution on [0,1000].

Data Cluster	Data Sets	Description

Experiment 3 data sets	`exp3a.csv`
	`exp3b.csv`
	`exp3c.csv`
	`exp3d.csv`
		The exp3a.csv dataset contains two variables (X1, X2) whose values are taken from a uniform distribution between 0 and 1. The class label 1 is assigned for the instances whose x1+x2 is greater than the median of the x1+x2 values computed from all the rows. Class label 0 is assigned to the remaining instances. Check the visualization below.
		 
		
		The exp3b.csv dataset has the same structure as the exp3a.csv dataset except that it has 10 variables.
		
		The exp3c.csv dataset has the same structure as the exp3a.csv dataset except that it has 20 variables.
		
		The exp3c.csv dataset has the same structure as the exp3a.csv dataset except that it has 50 variables.

Appendix: Estimating Accuracy Rates

For estimating accuracy rates you use function cross_val_score from the sklearn.model_selection module.

For estimating confusion matrices you use function confusion_matrix from the sklearn.metrics module. Note that it has to be used in a combination with function cross_val_predict from the sklearn.model_selection module. Example of code:

```
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
Y_pred = cross_val_predict(MyClassifier, X, Y, cv=10)
# X is the input data matrix and Y is output vector
conf_mat = confusion_matrix(Y, Y_pred)
```