# Machine Learning

Lab 1

Irene Natale[1]

[1]Department of Mathematics

June 4, 2018

# Outline

## The problem

- three different binary classification problems MONK-1, MONK-2, MONK-3

$$a_1 \in \{1,2,3\} \quad a_2 \in \{1,2,3\} \quad a_3 \in \{1,2\}$$

$$a_4 \in \{1,2,3\} \quad a_5 \in \{1,2,3,4\} \quad a_6 \in \{1,2\}$$

## The problem

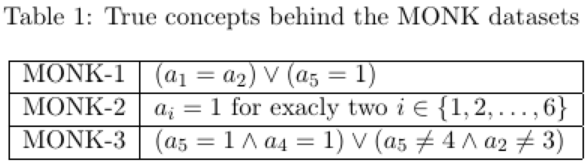


Table 1: True concepts behind the MONK datasets

| | |
|---|---|
| MONK-1 | $(a_1 = a_2) \vee (a_5 = 1)$ |
| MONK-2 | $a_i = 1$ for exacly two $i \in \{1, 2, \ldots, 6\}$ |
| MONK-3 | $(a_5 = 1 \wedge a_4 = 1) \vee (a_5 \neq 4 \wedge a_2 \neq 3)$ |

Figure:

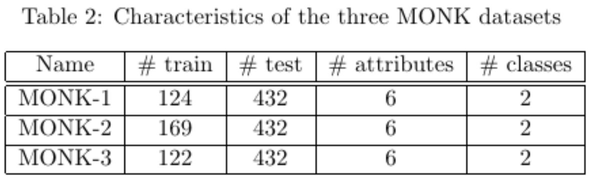


Table 2: Characteristics of the three MONK datasets

| Name | # train | # test | # attributes | # classes |
|---|---|---|---|---|
| MONK-1 | 124 | 432 | 6 | 2 |
| MONK-2 | 169 | 432 | 6 | 2 |
| MONK-3 | 122 | 432 | 6 | 2 |

Figure:

## Assignment 0

Which dataset is the most difficult to learn?

- **Training datasets are small** compared to the test datasets $\rightarrow$ all data sets generally hard to learn
- no thresholds formulation
- decision trees encode logicaal expressions of the conjuction of decisions along the path $\rightarrow$ **boolean functions** $\rightarrow$ **MONK-2** will be the most difficult to learn

## Assignment 1

Calculate the entropy of the training datasets using the function *Entropy*. Entropy of a data set represents the measure of unpredictability, and it is computed as

$$\sum_i -p_i \log_2 p_i$$

Table:

| Dataset | Entropy |
|---|---|
| MONK-1 | 1.0 |
| MONK-2 | 0.957117428265 |
| MONK-3 | 0.999806132805 |

**Entropy around 1** means that the output is far from being predictable $\rightarrow$ a **big variety of outputs** in each dataset

## Assignment 2

Explain entropy for a uniform distribution and a non-uniform distribution, present some example distributions with high and low entropy.

- A **uniform distribution** is a distribution is which every outcome has the same probability to happen. For example uniform distribution:

$$f(x) = \begin{cases} \frac{1}{b-a}, & a < x < b \\ 0, & \text{otherwise} \end{cases}$$

  and the entropy for this case will be computed as follows

$$E = -\int_a^b \frac{1}{b-a} log(\frac{1}{b-a})dx = \int_a^b \frac{1}{b-a} log(b-a)dx = log(b-a)$$

  **domain largeness** becomes wider, the entropy gets higher as well. This makes sense, since the entropy measures the total variance across classes.

## Assignment 2

In a **non-uniform distribution** one or more outputs are more likely to happen then the others. For example the Bernoulli distribution:

$$f(x) = \begin{cases} p, & x = 1 \\ 1 - p, & x = 0 \\ 0, & \text{otherwise} \end{cases}$$

the entropy computation becomes

$$E = -plog(p) - (1 - p)log(1 - p)$$

The **entropy** in this case takes its **maximum value** 1 when $p = 0.5 \rightarrow$ high variance of values.
If, for example, we take $p = 0.2$, the output $x = 0$ is definitely more likely to happen $\rightarrow$ less information $\rightarrow$ less variance

## Assignment 3

Calculate the **expected information gain** corresponding to each of the six attributes. Based on the results, which attribute should be used for splitting the examples at the root node?

Table: Information Gain

| Dataset | a1 | a2 | a3 | a4 | a5 |
|---|---|---|---|---|---|
| MONK-1 | 0.0752725 | 0.0058384 | 0.0047075 | 0.0263116 | 0.2870307 |
| MONK-2 | 0.0037561 | 0.0024584 | 0.0010561 | 0.0156642 | 0.0172771 |
| MONK-3 | 0.0071208 | 0.2937361 | 0.0008311 | 0.0028918 | 0.2559117 |

Information Gain indicates the effectiveness of an attribute in classifying the training data $\rightarrow$ **select the the argument that leads to the higher Information Gain** $\rightarrow$ **5th** argument is chosen for MONK-1 and MONK.2, while the **2nd** argument is selected for MONK-3.

## Assignment 4

How does the entropy of the subsets, Sk, look like when the information gain is maximized? How can we motivate using the information gain as a heuristic for picking an attribute for splitting?

$$Gain(S, A) = Entropy(S) - \sum_{k \in values(A)} \frac{|S_k|}{|S|} Entropy(S_k)$$

when the entropy of the subset $S_k$ decreases, the information gain increases

**max information gain = min entropy** $S_k$

split is effective $\rightarrow$ it leads to a more homogeneous subset of values, with lower entropy

the chosen split is effective $\rightarrow$ we are using a big amount of information

## Assignment 5

Build the full decision trees for all three Monk datasets using buildTree. Measure the performance of the decision tree on both the training and test datasets. Compute the train and test set errors for the three Monk datasets for the full trees. Were your assumptions about the datasets correct?

$$tree1 = d.buildTree(m.monk1, m.attributes)$$

$$tree2 = d.buildTree(m.monk2, m.attributes)$$

$$tree3 = d.buildTree(m.monk3, m.attributes)$$

## Assignment 5

Table: Fraction of correctly classified samples

| | Train samples | Test samples |
|---|---|---|
| MONK-1 | 1.0 | 0.8287037 |
| MONK-2 | 1.0 | 0.6921296 |
| MONK-3 | 1.0 | 0.9444444 |

Table: Error on training and test samples

| | Train samples error | Test samples error |
|---|---|---|
| MONK-1 | 0.0 | 0.1712962 |
| MONK-2 | 0.0 | 0.3078703 |
| MONK-3 | 0.0 | 0.0555555 |

## Assignment 5

- Errors from the train datasets are all 0.0 $\rightarrow$ perfectly fit $\rightarrow$ overfitting?
- high test error $\rightarrow$ **overfitting!** $\rightarrow$ model that is not able to capture the real structure of data because it is overly specialized for the training samples
- wrong assumptions

## Assignment 6

Explain pruning from a bias variance trade-off perspective.

- "allPruned" = picks the pruned tree that gives the best classification performance on the validation test
- total tree is done from training test, classification performance on the validation samples $\rightarrow$ "partition"
- **whole tree = most complex model** $\rightarrow$ **high variance, low bias** $\rightarrow$ we want to decrease the variance
- **pruned tree has lower variance!! (but high bias)**
- we create a **BALANCE** using the validation dataset

## Assignment 7

- we use training set to build the whole tree
- we use validation set to prune
- different partition fractions 0.3, 0.4, 0.5, 0.6, 0.7, 0.8
- we use test samples to compute the error
- we run the computation 1000 times because the SPLIT IS DONE RANDOMLY
- mean error computation

$$mean = \frac{1}{N} \sum_{i=0}^{N}{}^{i=N} v_i$$

- variance computed as standard deviation

$$variance = \frac{\sum (x_i - x_{mean})^2}{N-1}$$

Table: Error mean and variance for MONK-1

| fraction | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 |
|---|---|---|---|---|---|---|
| mean | 0.234701 | 0.206053 | 0.175112 | 0.156120 | 0.142026 | 0.1259 |
| variance | 0.054068 | 0.059863 | 0.062535 | 0.064660 | 0.069641 | 0.0738 |

Table: Error mean and variance for MONK-3

| fraction | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 |
|---|---|---|---|---|---|---|
| mean | 0.132465 | 0.109324 | 0.095868 | 0.088367 | 0.079567 | 0.0707 |
| variance | 0.054846 | 0.043255 | 0.035193 | 0.033341 | 0.040131 | 0.0465 |

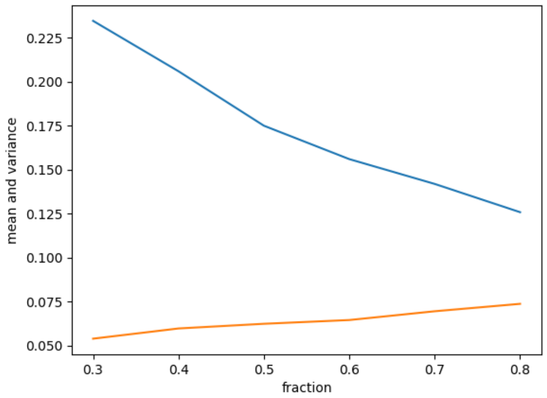


Figure: Error mean (in blue) and error variance (in orange), computed on the validation dataset for the monk-1 case

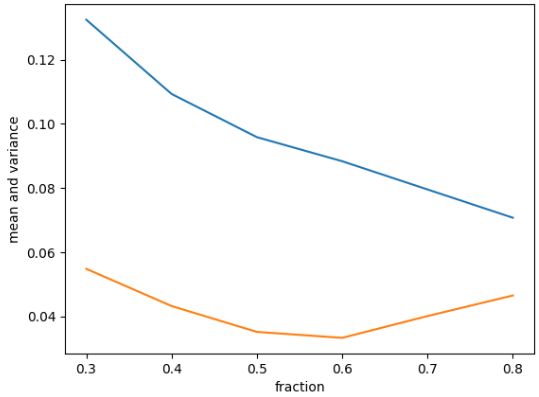


Figure: Error mean (in blue) and error variance (in orange), computed on the validation dataset for the monk-3 case

For both monk sets the error mean is lower when the training set is composed of 80 percent of the whole data (fraction = 0.8).

## Lab2 - Support Vector Machines

Theory Background

- Indicator function $ind(\bar{s}) = \sum_i \alpha_i t_i K(\bar{s}, \bar{x_i}) - b$
- Dual formulation of the problem: find $\alpha_i$ that minimize

$$\frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j t_i t_j K(\bar{x_i}, \bar{x_j}) - \sum_i \alpha_i$$

$$0 \leq \alpha_i \leq C \qquad \sum_i \alpha_i t_i = 0$$

- Kernels

$$K(x, y) = x^T \cdot y$$

$$K(x, y) = (x^T \cdot y + 1)^P$$

$$K(x, y) = e^{-\frac{||x-y||^2}{2\sigma^2}}$$

# Dataset 1

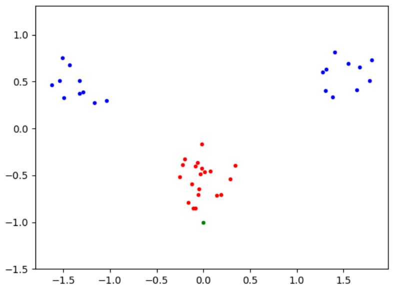


Figure: Data1

# Dataset 1 - Linear kernel

We first try to classify new data points with a linear kernel $\rightarrow$ doesn't work

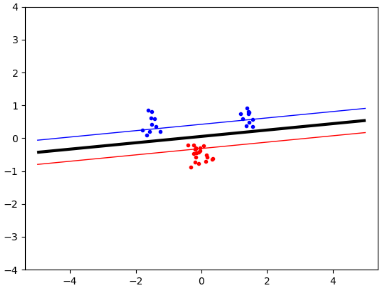
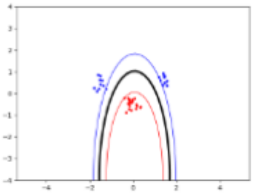
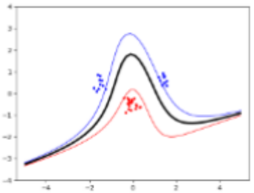


Figure: Data1, Linear Kernel, C=0.7

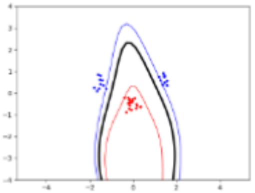
# Dataset 1 - Polynomial kernel



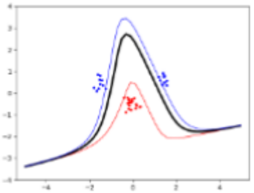
(a) Data1, Polynomial kernel, p=2, C=1



(b) Data1, Polynomial kernel, p=3, C=1



(c) Data1, Polynomial kernel, p=4, C=1

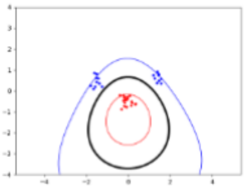


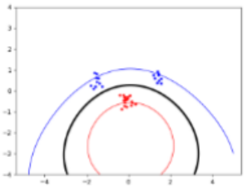
(d) Data1, Polynomial kernel, p=5, C=1

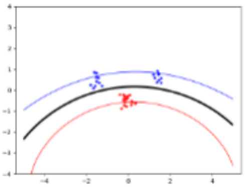
Figure 3: Polynomial kernel

- the degree of the kernel obviously influences the **shape** of the hyperplane
- when **p increases** $\rightarrow$ **low bias but high variance** $\rightarrow$ small modifications in the support vectors could significantly change the final solution

# Dataset 1 - Radial kernel


(a) Data1, Radial kernel, sigma=2, C=1


(b) Data1, Radial kernel, sigma=3, C=1


(c) Data1, Radial kernel, sigma=5, C=3

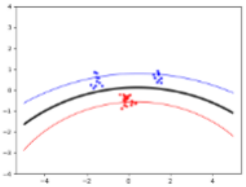

(d) Data1, Radial kernel, sigma=6, C=5

Figure 4: Radial Kernel

Figure: Data1, Polynomial Kernel

- circular hyperplanes
- parameter $\sigma$ controls the **smoothness** of the boundary, and so the slope of the hyperplane
- $\sigma \rightarrow \infty$ then the kernel becomes **linear**
- **small $\sigma \rightarrow$ low bias and a high variance**

# Dataset 2

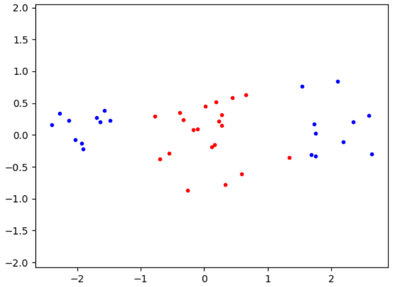
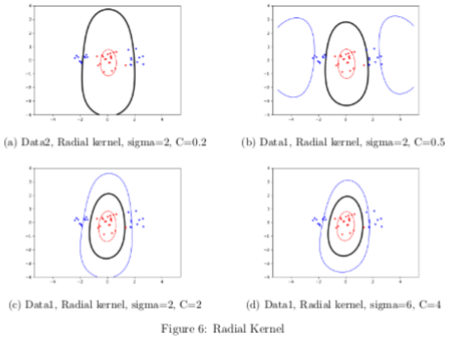


Figure: Data2

division between different classes is less clean $\rightarrow$ more difficult to classify.
A **study of the parameter C** on this data set $\rightarrow$ use of radial kernel with fixed $\sigma = 2$

## Dataset 2 - Radial kernel -study of C


(a) Data2, Radial kernel, sigma=2, C=0.2 (b) Data1, Radial kernel, sigma=2, C=0.5

(c) Data1, Radial kernel, sigma=2, C=2 (d) Data1, Radial kernel, sigma=6, C=4

Figure 6: Radial Kernel

Figure: Data2

- C big $\rightarrow$ large slacks
- C small $\rightarrow$ narrow slacks

## Dataset 3

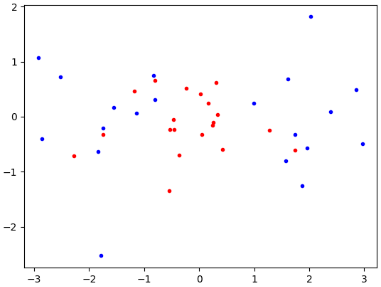
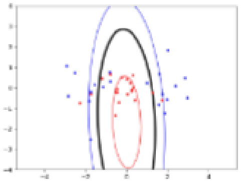
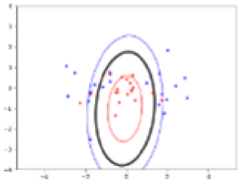
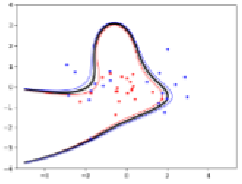


Figure: Data3

- not separable
- When should we opt for more slack rather than going for a more complex model (kernel) and vice versa?


(a) Data3, Polynomial kernel, degree=2, C=0.5


(b) Data3, Polynomial kernel, degree=2, C=2


(c) Data3, Polynomial kernel, degree=4, C=0.5

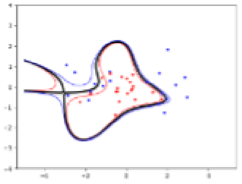
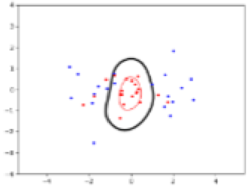

(d) Data3, Polynomial kernel, degree=4, C=2

Figure: Data3

Polynomial **degree 2** does not fit well even if C increases.
Good to increase the degree → **more complex model**


(a) Data3, Radial kernel, sigma = 0.7, C=0.5

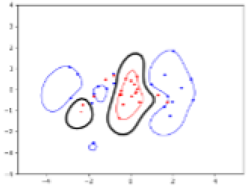

(b) Data3, Radial kernel, sigma = 0.7, C=8

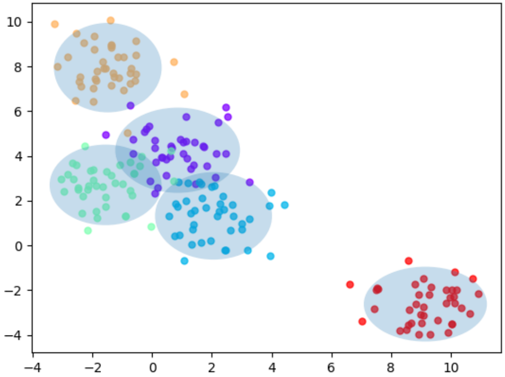
Figure: Data3

We need to pay attention to not use a too complex model → **overfitting!!**
→ croos validation study to see what is the best

## LAB 3 - BAYES + BOOSTING

## Assignment 1

- generation of dataset with the function genBlobs()
- Compute the ML-estimates for the data and plot the 95 percent-confidence interval using the function plotGaussians



## Assignment 3 - Bayes classifier

Now we randomly divide dataset into training and test dataset and

- we **train** the classifier on the training dataset
- **evaluate performance** on the test partition

We do these two steps for the IRIS DATASET :
"Final mean classification accuracy 89 with standard deviation 4.16"
and on the VOWEL DATSET:
"Final mean classification accuracy 64.7 with standard deviation 4.03"

## Assignment 3

1. When can a feature independence assumption be reasonable and when not?
   Assumption of independence among all the features:

$$P(x_1, x_2, ..., x_N|y) = \prod_{i=1}^{N} P(x_i|y)$$

   - **hard to be true in reality but it PERFORMS WELL even** when dependency among classes is high
   - good to solve problems related to CURSE OF DIMENTIONALITY $\rightarrow$ we are modelling a D-dim feature as D 1-dim features
   - Assumptions works well also because Bayes classifier will give a correct classification as long as the correct class is MORE PROBABLE then the others, even if the posterior computation is uncorrect!

## Assignment 3

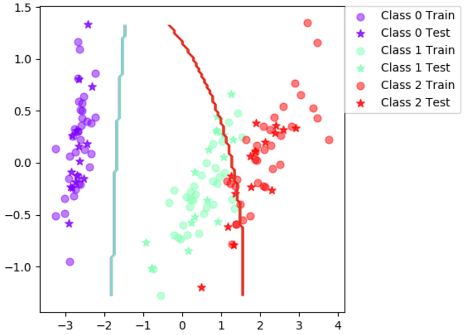


Figure: Iris dataset Decision boundaries

NB: the red boundary does not perform well due to noisy and overlapping classes → solution could be boundaries ith slacks (SVMs) or BOOSTING of Bayes classifier

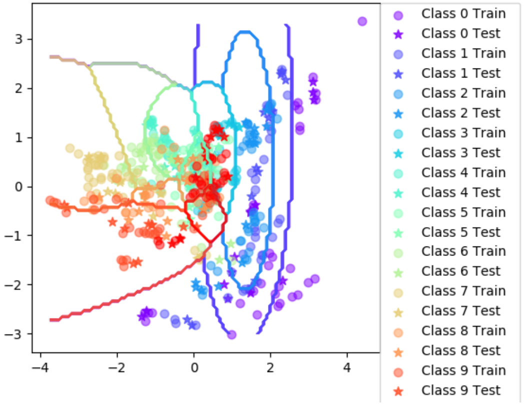


Figure: Vowels dataset Decision boundaries

## Assignment 5

Compute the classification accuracy of the boosted classifier on some data sets using "testClassifier" and compare it with those of the basic classifier on the vowels and iris data sets (Ass. 3)

- testClassifier(BoostClassifier(BayesClassifier(), T=10), dataset='iris',split=0.7)
  "Final mean classification accuracy **94.1** with standard deviation **6.72**"
- normal Bayes classifier on iris dataset:
  "Final mean classification accuracy **89** with standard deviation **4.16**"
- testClassifier(BoostClassifier(BayesClassifier(), T=10), dataset='vowel',split=0.7)
  "Final mean classification accuracy **80.2** with standard deviation **3.52**"
- normal Bayes cassifier on vowels datset:
  "Final mean classification accuracy **64.7** with standard deviation **4.03**"

## Assignment 5

- predictable improvement of performance because we are using boosting!
- weights in the Adaboost algorithm gives more importance to incorrectly classified instances $\rightarrow$ **focus more on the incorrect samples**, increasing classification accuracy

## Assignment 6

the code includes a class DecisionTreeClassifier based upon skLearns decision tree classifier. Test the decision tree classifier on the vowels and iris data sets, with and without boosting.

- NO BOOSTING - IRIS

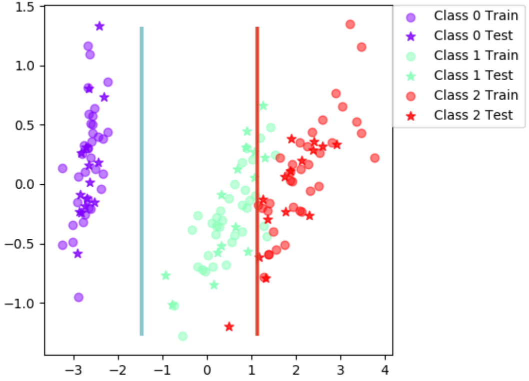


Figure: Decision boundary - iris dataset - DecisionTreeClassifier

"Final mean classification accuracy 92.4 with standard deviation 3.71"

- NO BOOSTING - VOWELS

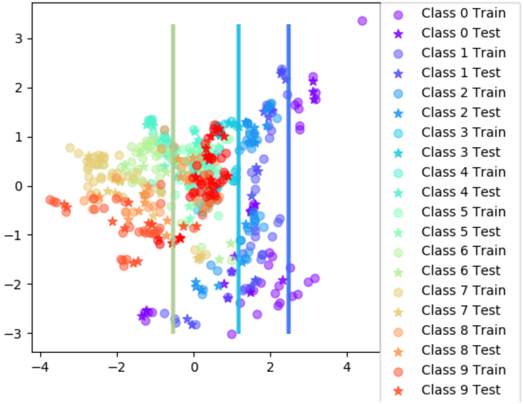


Figure: Decision boundary - vowel dataset - DecisionTreeClassifier

"Final mean classification accuracy 64.1 with standard deviation 4"

- YES BOOSTING - IRIS

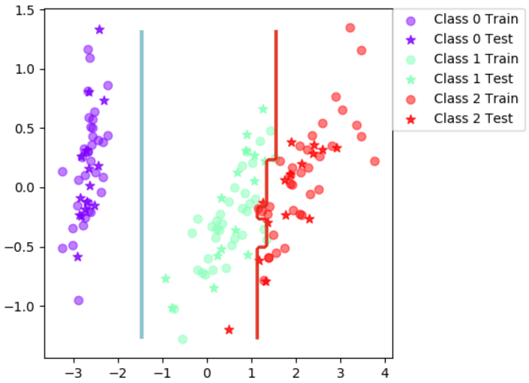


Figure: Decision boundary - iris dataset - boosted DecisionTreeClassifier

"Final mean classification accuracy 94.6 with standard deviation 3.65"

- YES BOOSTING - VOWELS

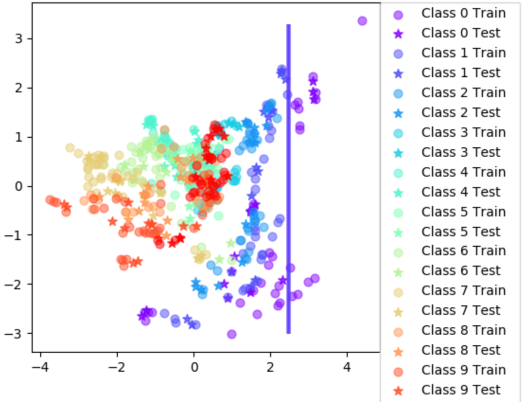


Figure: Decision boundary - vowel dataset - boosted DecisionTreeClassifier

"Final mean classification accuracy 86.6 with standard deviation 3.01"

1. Yes, there is improvement! Wider for vowels - Decision trees in general have high variance $\rightarrow$ the boosting decrease the variance
2. For IRIS dataset it is easy to see the INCREASE OF COMPLEXITY $\rightarrow$ performs better
3. Yes, we are making up for not using a more advanced model in the basic classifier by using boosting $\rightarrow$ the boosted classifier performs better then the basic ones in both cases.

## Assignment 7

Pick a classifier:

- OUTLIERS - Naive Bayes - The boosted version would give too much weight to the outliers, and decision trees tend to overfit the training dataset
- IRRELEVANT INPUTS - Decision trees - because they tend to focus only on relevant features
- PREDICTIVE POWER - Boosted Naive Bayes - it has the best prediction performance
- MIXED TYPES OF DATA - Decision trees - are more flexible and can work on different types of data, while Bayes classifier works better on continuous features - Boosting would also increase performance
- SCALABILITY - Bayes works well also with small datasets, while Decision trees works better when the dataset in bigger. Bayes solves problems about curse of dimensionality very well thanks to the assumption of independence among features.