

STAT406- Methods of Statistical Learning Lecture 15

Matias Salibian-Barrera

UBC - Sep / Dec 2017

Classification as prediction

- Most classifiers can be thought of as different ways to estimate or model

$$P(G = \mathbf{c}_j | \mathbf{X} = \mathbf{x})$$

- One way to model these probabilities is via Bayes' Thrm:

$$P(G = \mathbf{c}_j | \mathbf{X} = \mathbf{x}) = \mathbf{f}(\mathbf{x} | \mathbf{c}_j) P(G = \mathbf{c}_j) / \mathbf{f}(\mathbf{x})$$

LDA vs QDA

LDA classifies an observation into the class c_j for which the estimated (under a normality assumption)

$$P(G = c_j | \mathbf{X} = \mathbf{x})$$

is highest.

If $\mathbf{X} | G = c_j \sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma})$ it means the class for which

$$\delta_j(\mathbf{x}) = \mathbf{x}' \mathbf{a}_j + b_j$$

is highest, where

$$\mathbf{a}_j = \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_j \quad \text{and} \quad b_j = -\frac{1}{2} \boldsymbol{\mu}_j' \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_j + \log(p_j)$$

QDA

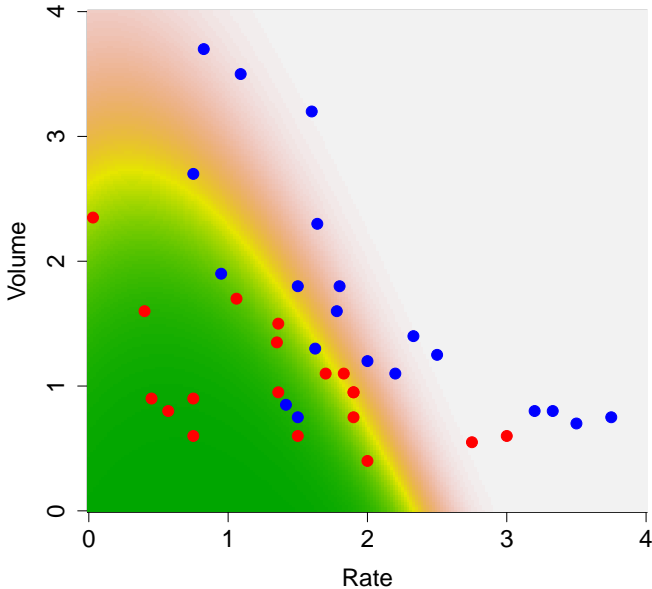
If $\mathbf{X} | G = c_j \sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ then, the estimated posterior probabilities look different.

QDA classifies an observation into the class c_j that maximizes

$$\delta_j(\mathbf{x}) = \mathbf{x}' \mathbf{A}_j \mathbf{x} + \mathbf{x}' \mathbf{d}_j + b_j$$

(see the book for details)

QDA-based probabilities



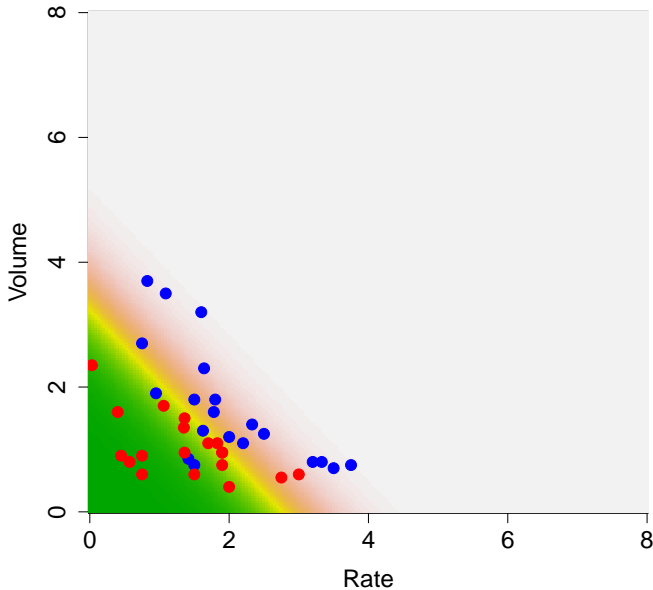
LDA - QDA - Logistic classifiers

- Multiclass: LDA, QDA and Logistic classifiers extend to the case of more than 2 classes
- Examples in R (Zip-code hand-written digits)

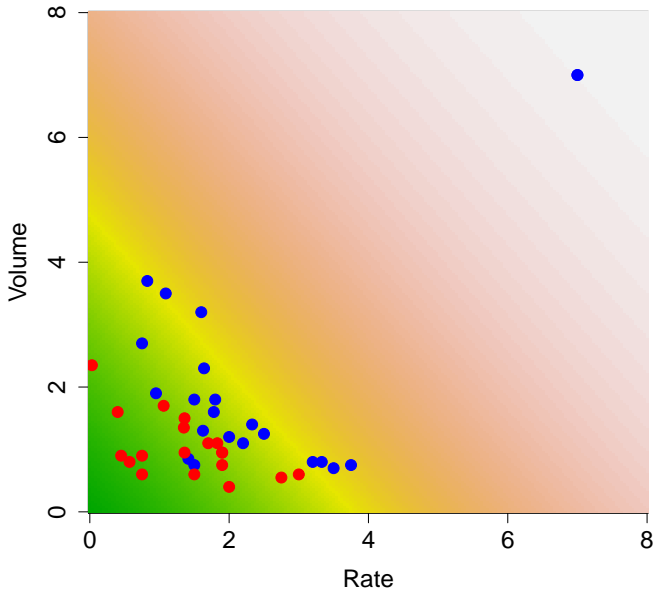
LDA - QDA - Logistic classifiers

- LDA vs QDA: presents the usual “flexibility vs. variability” trade-off
- LDA & QDA vs Logistic classifiers:
Gaussian MLE estimates
(non-robust, sensitive to the Gaussian assumption) vs. **Binomial MLE estimates** (no distributional assumption of $\mathbf{X} | G = c_j$ required).

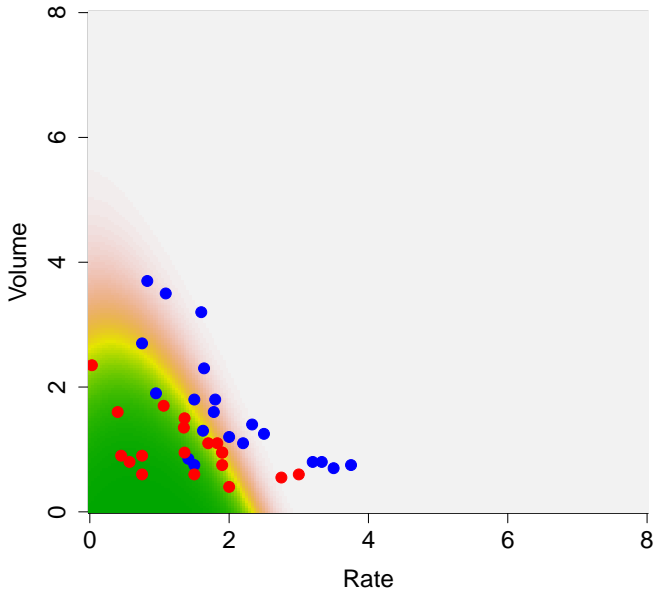
LDA



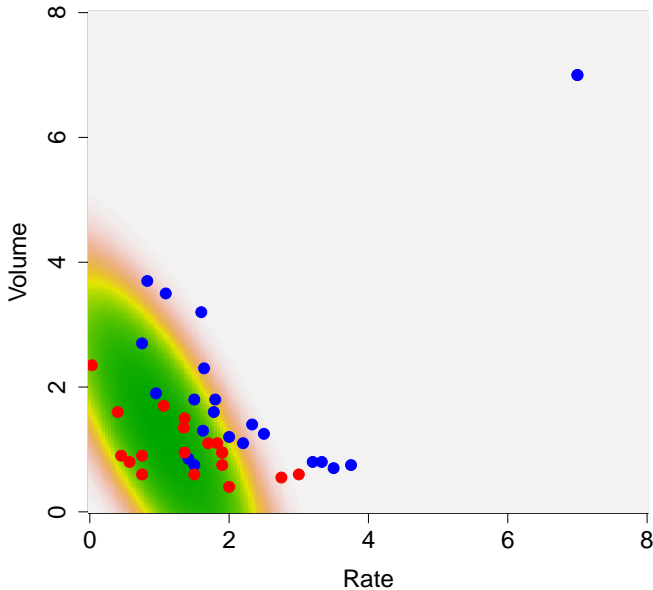
LDA + 2 “good outliers”



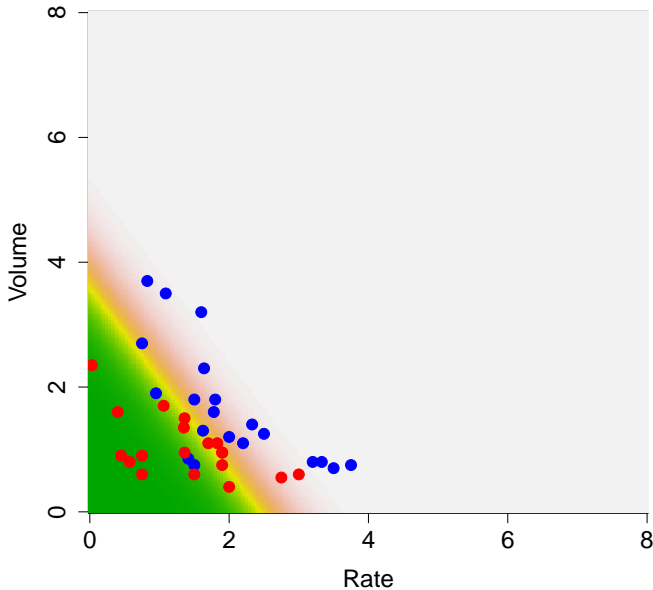
QDA



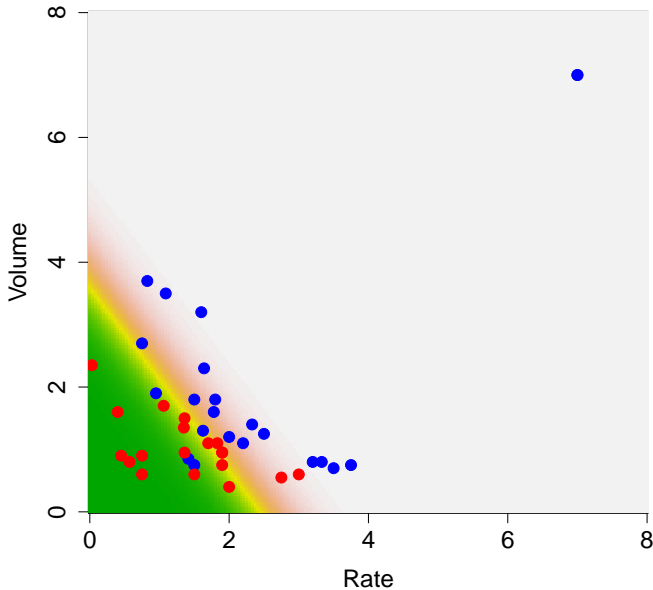
QDA + 2 “good outliers”



Logistic



Logistic + 2 “good outliers”



LDA - QDA - Logistic classifiers

Examples with R

Nearest-neighbours

- We need to estimate

$$P(G = \mathbf{g} | \mathbf{X} = \mathbf{x})$$

- An intuitive and “model-free” estimator is the nearest-neighbours estimator

Nearest-neighbours

- Same spirit as the local-constant (kernel) regression estimator
- The K-NN estimator is

$$\hat{P}(G = \mathbf{g} | \mathbf{X} = \mathbf{x}) = \frac{1}{|N_{\mathbf{x}}^K|} \sum_{j \in N_{\mathbf{x}}^K} \mathbf{I}\{Y_j = \mathbf{g}\}$$

where

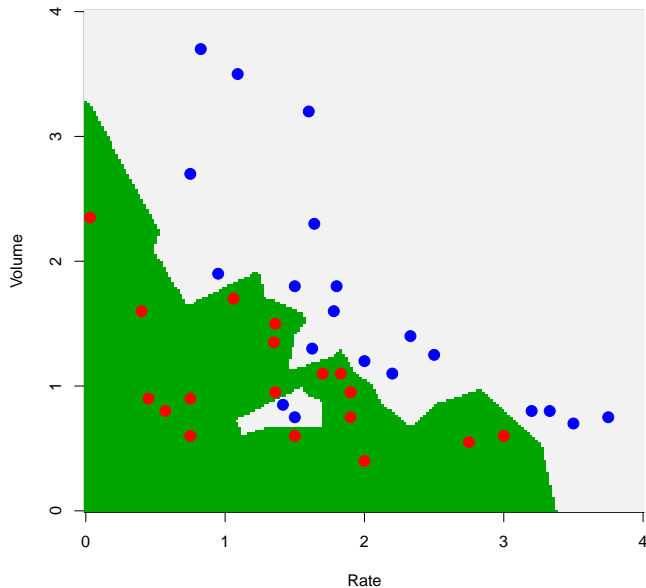
$$N_{\mathbf{x}}^K = \{i : d(\mathbf{X}_i, \mathbf{x}) \leq d_{(K)}\}$$

and $d_{(K)}$ is the distance from \mathbf{x} to the K -th closest point in the sample (\mathbf{X}_i)

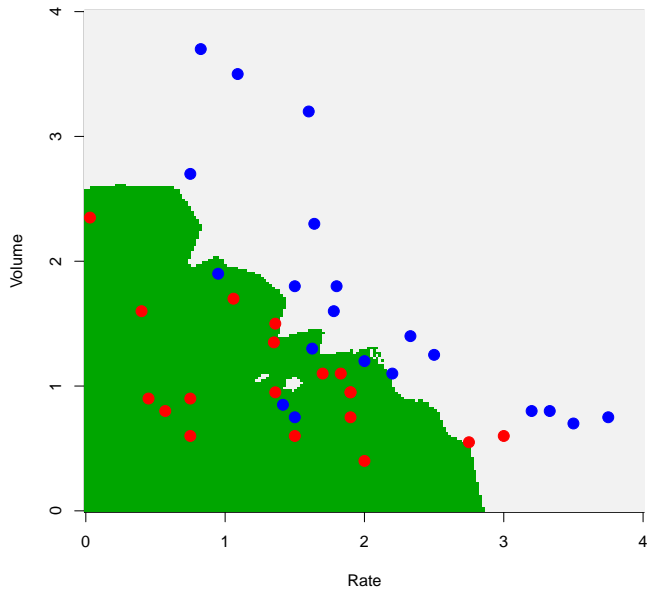
Nearest-neighbours

- The K-NN estimator is the proportion of observations from class **g** among the closest K neighbours
- The K-NN classifier assigns a point to the class most represented among its K neighbours (“peer pressure”)

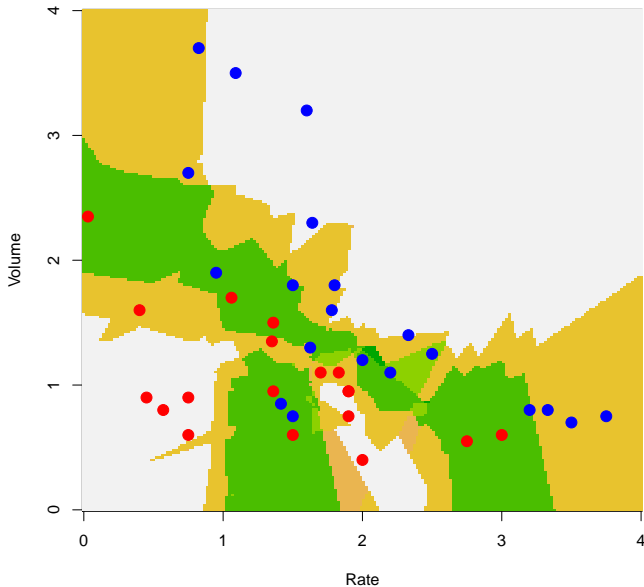
Vaso - 1-NN



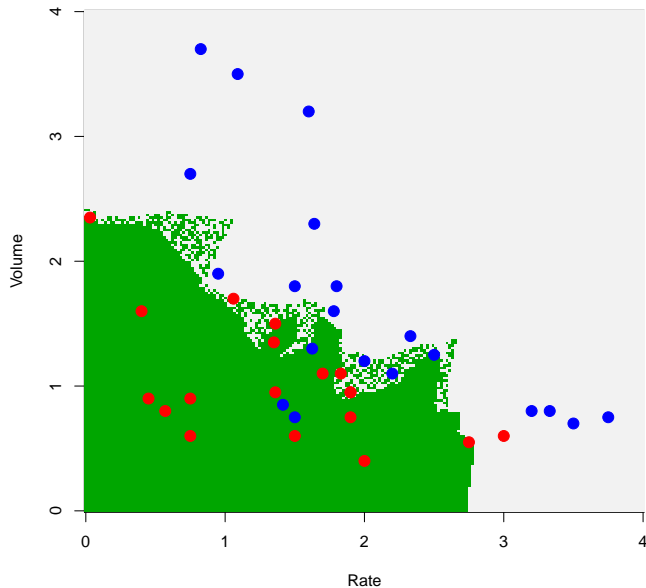
Vaso - 5-NN



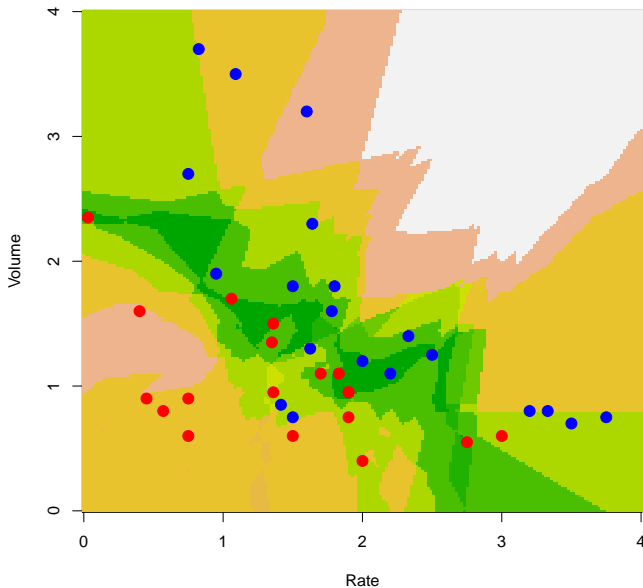
Vaso - 5-NN - votes



Vaso - 10-NN



Vaso - 10-NN - votes

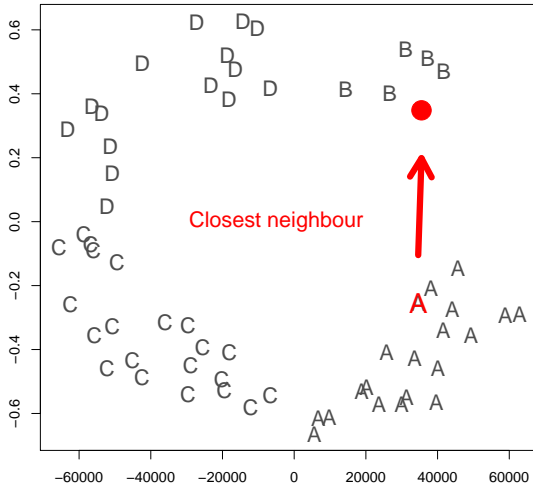


Nearest-neighbours

- How can we select the number K of neighbours?
- Zip-code example

Nearest-neighbours - Challenge

- What's wrong with this picture?



Classification Trees

- To estimate the probabilities of each class **g** for a particular value of the feature vector **x**, **nearest neighbours** constructs estimates

$$\begin{aligned}\hat{P}(\mathbf{g} = \mathbf{g} \mid \mathbf{X} = \mathbf{x}) \\ = \text{prop. of objects of class } \mathbf{g} \\ \text{among } \mathbf{x}'\text{'s neighbours}\end{aligned}$$

Classification Trees

- Drawback: may not be “local” for moderate number of features (hence, may not represent the distribution of \mathbf{g} for $\mathbf{X} = \mathbf{x}$)
- Yet another incarnation of the “curse of dimensionality”

Classification Trees

- If we assume a model for the distribution of the features \mathbf{X} for each class

$$f(\mathbf{x} | \mathbf{g} = \mathbf{g}) = f_{\mathbf{g}}(\mathbf{x})$$

then we have a formula for

$$P(\mathbf{g} = \mathbf{g} | \mathbf{X} = \mathbf{x}) = \frac{f_{\mathbf{g}}(\mathbf{x}) P(\mathbf{g} = \mathbf{g})}{\sum_{\mathbf{g}} f_{\mathbf{g}}(\mathbf{x}) P(\mathbf{g} = \mathbf{g})}$$

Classification Trees

- We typically have estimates for these probabilities
- Only the numerator is needed
- Drawback: the model may not be correct.
- Model-based inferences are typically more stable than model-free ones
- But they may be biased if the model is a poor approximation to the truth

Classification Trees

- Classification trees provide another family of estimates for $P(\mathbf{g} = \mathbf{g} | \mathbf{X} = \mathbf{x})$
- They do not need a model
- Instead of estimating the local proportion (probability) of each class around a point \mathbf{x} , CART's attempt to find regions of the feature space that are “dominated” by one class.

Classification Trees

- Classification trees try to identify **regions** of the domain where **one class** clearly **dominates** the others (i.e. where the class proportions are far from being “uniform”)
- They **search** for these regions in a **very specific** way.

Classification Trees

- These regions are searched using a sequential algorithm that at each step **partitions** the current level (“leaf”) into two “leaves” / “children” **according** to the value of **one of the feature variables**, for example:

$$X_j \leq \mathbf{a} \quad \text{versus} \quad X_j > \mathbf{a}$$

Classification Trees

- At every step, the algorithm searches for the variable X_j and level a that produce the largest increase in “homogeneity” (alternatively: the largest decrease in “heteroscedasticity”)
- We need a measure of “homogeneity” (or lack of it)

Classification Trees

Some practical considerations in building (spanning) the tree:

- Do not partition nodes / leaves with fewer elements than a fixed threshold (say, 5)
- Do not partition a node if the “gain” in less homogeneity is less than a certain percentage of the current value
- Or both...

Classification Trees

Let **N** denote a “node”, that is: a subset of the data.

Let \hat{p}_j , $j = 1, \dots, K$ be the proportion of observations of each class in this node

$$\hat{p}_j = \frac{\# \text{ of observations of class } j \text{ in node } \mathbf{N}}{\text{total } \# \text{ of observations in node } \mathbf{N}}$$

Classification Trees

Maximum homogeneity when

$$\hat{p}_1 \approx \hat{p}_2 \approx \dots \approx \hat{p}_K$$

Minimum homogeneity when $\hat{p}_r \approx 1$ for
some $1 \leq r \leq K$

Classification Trees

Measures of homogeneity

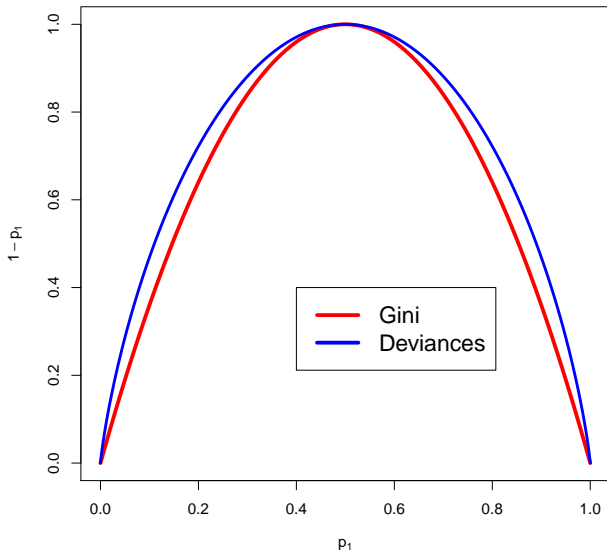
- Gini index:

$$Q_G(\hat{p}_1, \dots, \hat{p}_K) = \sum_{j \neq i}^K \hat{p}_j \hat{p}_i = \sum_{j=1}^K \hat{p}_j (1 - \hat{p}_j)$$

- Entropy or deviance:

$$Q_D(\hat{p}_1, \dots, \hat{p}_K) = -2 \sum_{j=1}^K \hat{p}_j \log(\hat{p}_j)$$

Gini & Deviances - 2 groups



Classification Trees

Define the “homogeneity” of a node **N** as

$$Q(\mathbf{N}) = Q(\hat{p}_1, \dots, \hat{p}_K)$$

where

$$\hat{p}_j = \frac{\# \text{ of observations of class } j \text{ in node } \mathbf{N}}{\text{total } \# \text{ of observations in node } \mathbf{N}}$$

Q could be Q_G or Q_D , for example.

Classification Trees

- (a) Start with a node **N** containing all data points
- (b) Find the variable X_j and value **a** that minimize

$$\mathbf{n}_L Q(\mathbf{X} \in \mathbf{N} : X_j \leq \mathbf{a}) + \mathbf{n}_R Q(\mathbf{X} \in \mathbf{N} : X_j > \mathbf{a})$$

$$\text{where } \mathbf{n}_L = \#\{\mathbf{X} \in \mathbf{N} : X_j \leq \mathbf{a}\}$$

- (c) Define the corresponding two “children” of node **N**
- (d) Apply the same to each “child” / “leaf”

Classification Trees

- In practice, we need to weight the homogeneity by the number of observations in each leaf.
- This reflects a probabilistic model for the tree, and represents the probability that a point is observed in this leaf
- In other words: minimize homogeneity more in “larger” (in average) leaves.

Classification Trees

- Let n_c be the size of the node \mathbf{N}_c , its homogeneity is

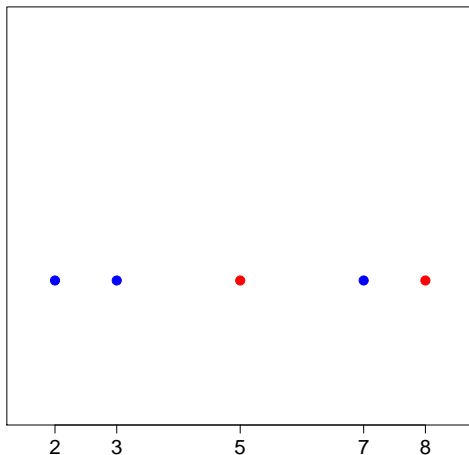
$$n_c Q(\mathbf{N}_c) \propto \frac{n_c}{n} Q(\mathbf{N}_c)$$

- If we split \mathbf{N}_c into \mathbf{N}_1 and \mathbf{N}_2 the homogeneity is

$$n_1 Q(\mathbf{N}_1) + n_2 Q(\mathbf{N}_2) \propto \frac{n_1}{n} Q(\mathbf{N}_1) + \frac{n_2}{n} Q(\mathbf{N}_2)$$

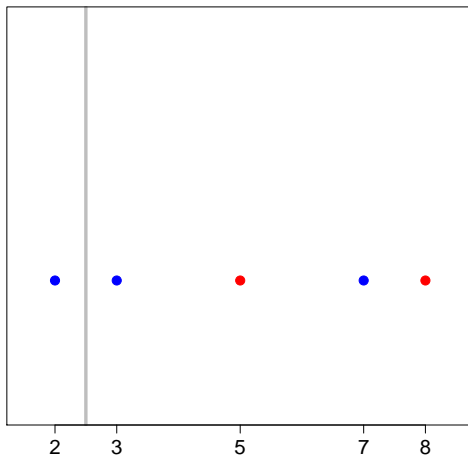
The homogeneity of any split is less than that of the parent node (Breiman, Friedman, Olshen, Stone; 1984).

Classification Trees



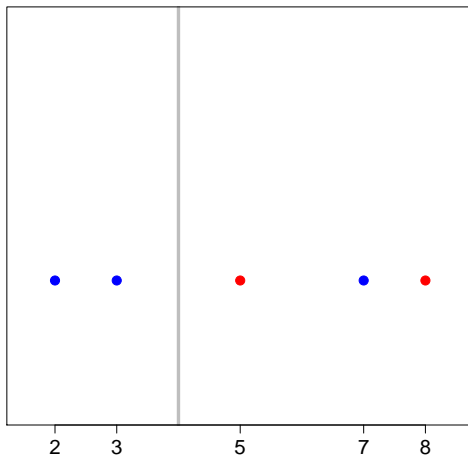
$$(\hat{p}_R = 2/5 \quad \hat{p}_B = 3/5) \quad Q_G = \mathbf{2.4}$$

Classification Trees



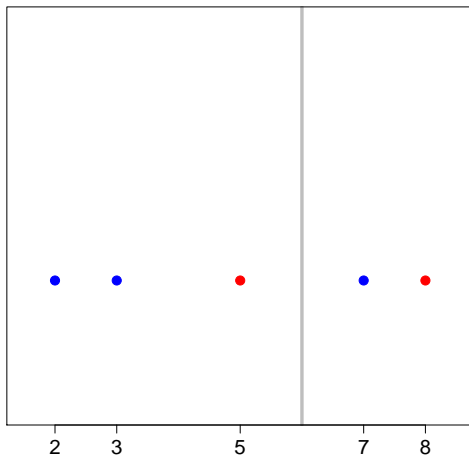
$$(\hat{p}_R = 0 \quad \hat{p}_B = 1) \quad (\hat{p}_R = 1/2 \quad \hat{p}_B = 1/2) \quad Q_G = \mathbf{2}$$

Classification Trees



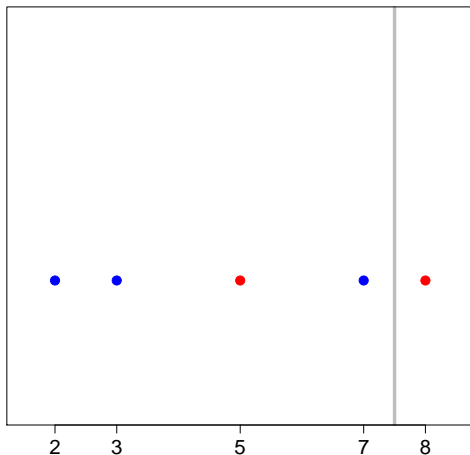
$$(\hat{p}_R = 0 \quad \hat{p}_B = 1) \quad (\hat{p}_R = 2/3 \quad \hat{p}_B = 1/3) \quad Q_G = \mathbf{1.33}$$

Classification Trees



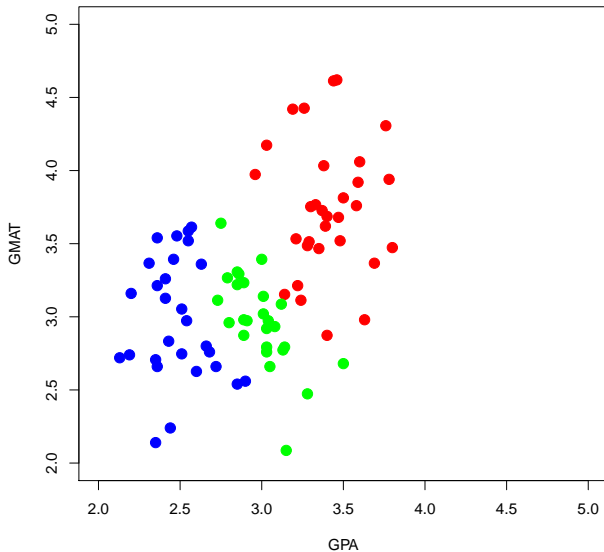
$$(\hat{p}_R = 1/3 \quad \hat{p}_B = 2/3) \quad (\hat{p}_R = 1/2 \quad \hat{p}_B = 1/2) \quad Q_G = \mathbf{2.33}$$

Classification Trees

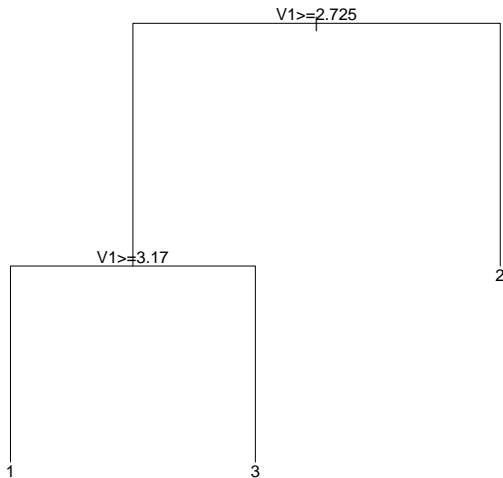


$$(\hat{p}_R = 1/4 \quad \hat{p}_B = 3/4) \quad (\hat{p}_R = 1 \quad \hat{p}_B = 0) \quad Q_G = \mathbf{1.5}$$

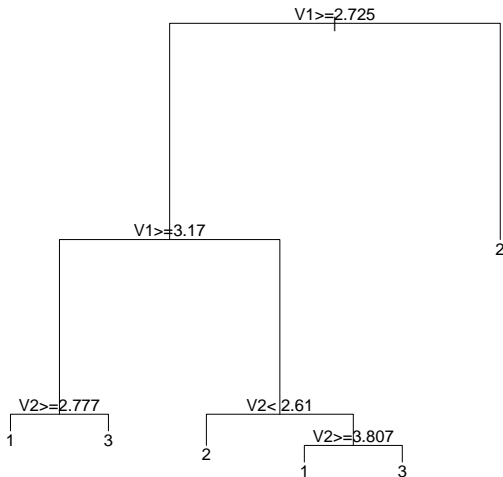
Grad admissions



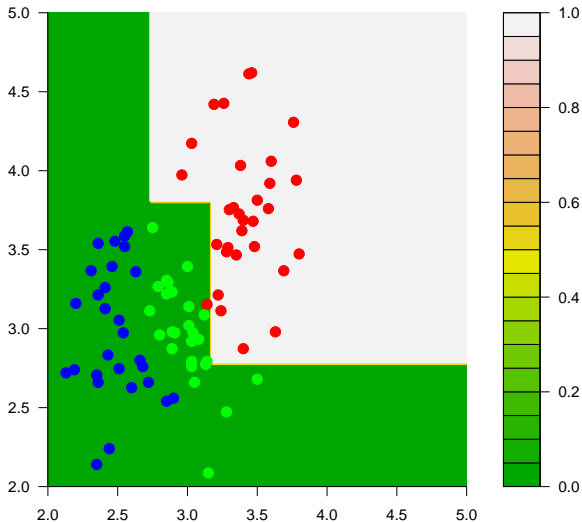
Grad admissions - Gini tree



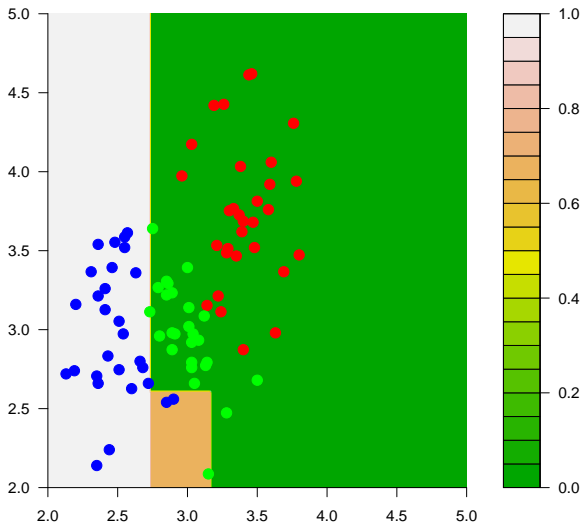
Grad admissions - Deviance tree



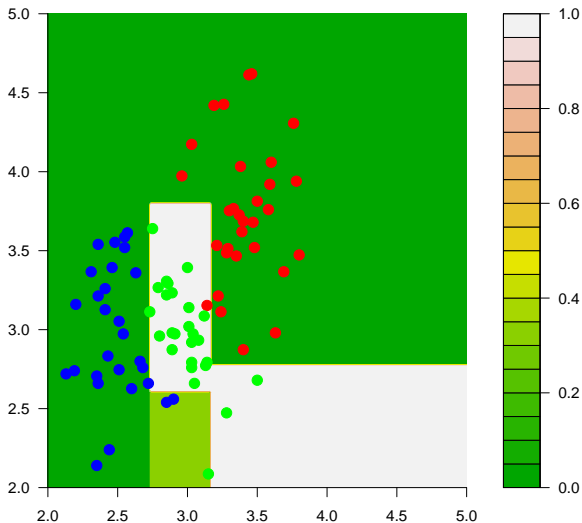
Grad admissions



Grad admissions



Grad admissions



Classification example

- ISOLET data
<http://archive.ics.uci.edu/ml/datasets/ISOLET>
- 150 subjects spoke the name of each letter twice
- 52 samples from each subject, 300 samples for each letter

Classification example

- 617 features (explanatory variables). They include: spectral coefficients; contour features, sonorant features, pre-sonorant features, and post-sonorant features.
- Data are split into training ($n = 6238$) and test ($n = 1559$)
- Mission: build a classifier to identify future spoken letters

Classification example

- I separated the letters C and Z

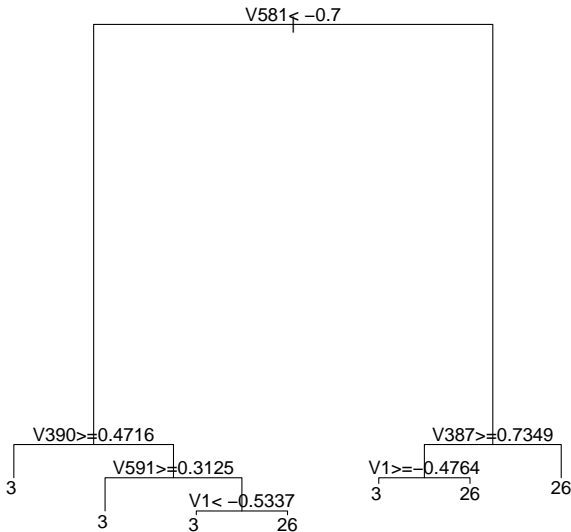
```
> x <- read.table('isolet-train.data', sep=',')
> # 3 and 26 --- "C" and "Z"
>
> xa <- x[ x$V618 == 3, ]
> xb <- x[ x$V618 == 26, ]
>
> xx <- rbind(xa, xb)
> xx$V618 <- as.factor(xx$V618)
```

Classification example

- Fit a classification tree

```
> myc <- rpart.control(minsplit=7, cp=1e-7,  
                        xval=10)  
>  
>  
> set.seed(123)  
> a.t <- rpart(V618~., data=xx,  
               parms = list(split = 'information'),  
               control=myc)  
> b <- a.t$cptable[  
        which.min(a.t$cptable[, "xerror"]), "CP"]  
> d.r <- prune(a.t, cp=b)
```


Classification example



Classification example

- Predict on the test data

```
> d.pr <- predict(d.r, newdata=dd,  
                  type='class')  
> table(truth, d.pr)
```

	d.pr	
truth	3	26
	3	59 1
	26	0 60

- And we only used 5 explanatory variables!

Classification example

- The problem is not trivial
- Let's try a 1-NN nearest neighbour classifier

```
> u1 <- knn(train=xx[, -618], test=dd[, -618],  
            cl=xx[, 618], k = 1)  
>  
> table(truth, u1)  
      u1  
truth  3 26  
      3 57  3  
     26  9 51
```

Classification example

- With 5-NN is not much better

```
> u5 <- knn(train=xx[, -618],  
             test=dd[, -618], cl=xx[, 618],  
             k = 5)  
>  
> table(truth, u5)  
      u5  
truth  3 26  
      3 58  2  
     26  5 55
```

Classification example

- With a logistic classifier

```
> xx$V619 <- as.numeric(xx$V618==3)
> d.glm <- glm(V619 ~ . - V618, data=xx,
               family=binomial)
```

Warning message:

glm.fit: algorithm did not converge

[...]

```
> table(truth, pr.glm)
```

	pr.glm	
truth	0	1
3	25	35
26	33	27

- Can we explain the problem?

Classification example

- How about the approach based on the Gaussian distribution of the features (within each class)?

```
> library(MASS)
>
> d.lda <- lda(V618 ~ ., data=xx)
Warning message:
In lda.default(x, grouping, ...) :
  variables are collinear
```

- Can we explain the problem?