Exercise Collection – Supervised Classification

Contents

Lecture exercises **1**

Exercise 1: logistic regression – thresholding . . . 1
Exercise 2: logistic regression – link function and likelihood . . . 2
Exercise 3: logistic regression – softmax hyperplane . . . 4
Exercise 4: decision boundaries with `mlr3` . . . 4
Exercise 5: k-NN classification . . . 5
Exercise 6: k-NN from scratch . . . 7
Exercise 7: QDA from scratch . . . 8
Exercise 8: naive Bayes . . . 8
Exercise 9: classifying `ionosphere` . . . 10

Further exercises **10**

Exercise 10: WS2020/21, first, question 1 . . . 10
Exercise 11: WS2020/21, first, question 4 . . . 12

Ideas & exercises from other sources **13**

Lecture exercises

Exercise 1: logistic regression – thresholding

In logistic regression, we estimate the probability $\pi(\mathbf{x}) = \mathbb{P}(y = 1 \mid \mathbf{x})$. To decide if $\hat{y}$ is 0 or 1, we follow:

$$\hat{y} = 1 \quad \Leftrightarrow \quad \hat{\pi}(\mathbf{x}) \geq a$$

a) What happens if you are choosing $a = 0.5$? More precisely, from which value of $\boldsymbol{\theta}^T\mathbf{x}$ do you predict $\hat{y} = 1$ rather than $\hat{y} = 0$?

b) Explain (using words) why $a = 0.5$ is a sensible threshold.

Solution 1:

(a) For a binary classification problem the model can be written as:

$$\hat{y} = 1 \quad \Leftrightarrow \quad \pi(\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})} \geq a$$

This can be reformulated, s.t. for $a \in (0, 1)$

$$\begin{aligned} & \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})} \geq a \\ \Leftrightarrow\ & 1 + \exp(-\boldsymbol{\theta}^T \mathbf{x}) \leq a^{-1} \\ \Leftrightarrow\ & \exp(-\boldsymbol{\theta}^T \mathbf{x}) \leq a^{-1} - 1 \\ \Leftrightarrow\ & -\boldsymbol{\theta}^T \mathbf{x} \leq \log(a^{-1} - 1) \\ \Leftrightarrow\ & \boldsymbol{\theta}^T \mathbf{x} \geq -\log(a^{-1} - 1) \end{aligned}$$

For $a = 0.5$ we get:

$$\hat{y} = 1 \quad \Leftrightarrow \quad \boldsymbol{\theta}^T \mathbf{x} \geq -\log(0.5^{-1} - 1) = -\log(2 - 1) = -\log(1) = 0$$

This means the linear decision boundary is defined by a hyperplane equation, i.e., $\boldsymbol{\theta}^T \mathbf{x} = 0$ and it divides the input space in the "1"-space ($\boldsymbol{\theta}^T \mathbf{x} \geq 0$) and in the "0"-space ($\boldsymbol{\theta}^T \mathbf{x} < 0$).

(b) When the threshold $a = 0.5$ is chosen, the losses of missclassified observations, i.e., $L(\hat{y} = 0|y = 1)$ and $L(\hat{y} = 1|y = 0)$, are weighted equally. This means $a = 0.5$ is a sensible threshold if one does not need to avoid one type of misclassification more than the other.
Intuitively it makes sense to cut off at 0.5 because, if the probability for 1 is closer to 1 than to 0, we would intuitively choose 1 rather than 0.

Exercise 2: logistic regression – link function and likelihood

a) What is the relationship between softmax $\pi_k(x) = \frac{\exp(\theta_k^T x)}{\sum_{j=1}^{g} \exp(\theta_j^T x)}$ and the logistic function $\pi(\mathbf{x}) = \frac{1}{1+\exp(-\theta^T x)}$ for $g = 2$ (binary classification)?

b) The likelihood function of a multinomially distributed target variable with g target classes is given by

$$\mathcal{L}_i = \mathbb{P}(Y^{(i)} = y^{(i)}|x^{(i)}, \theta_1, \ldots, \theta_g) = \prod_{j=1}^{g} \pi_j(x^{(i)})^{\mathbb{1}_{\{y^{(i)}=j\}}}$$

where the posterior class probablities $\pi_1(x), \ldots, \pi_g(x)$ are modeled with softmax regression. Derive the likelihood function of n such independent target variables. How can you transform this likelihood function into an empirical risk function?
Hints:

- By following the maximum likelihood principle, we should look for parameters $\theta_1, \ldots, \theta_g$, which maximize the likelihood function.
- The expressions $\prod \mathcal{L}_i$ and $\log \prod \mathcal{L}_i$ (if this expression is defined) are maximized by the same parameters.
- The empirical risk is a *sum* of loss function values, not a *product*.
- Minimizing a scalar function multiplied with -1 is equivalent to maximizing the original function.

State the associated loss function.

c) Explain how the predictions of softmax regression (multiclass classification) look like (probabilities and classes) and define the parameter space.

Solution 2:

a) $\pi_1(x) = \frac{\exp(\theta_1^T x)}{\exp(\theta_1^T x)+\exp(\theta_2^T x)}$

$\pi_2(x) = \frac{\exp(\theta_2^T x)}{\exp(\theta_1^T x)+\exp(\theta_2^T x)}$

$\pi_1(x) = \frac{1}{(\exp(\theta_1^T x)+\exp(\theta_2^T x))/\exp(\theta_1^T x)} = \frac{1}{1+\exp(-\theta^T x)}$ where $\theta = \theta_1 - \theta_2$ and $\pi_2(x) = 1 - \pi_1(x)$

b) When using softmax regression the posterior class probability for the class k is modeled, s.t.

$$\pi_k(x) = \frac{\exp(\theta_k^T x)}{\sum_{j=1}^{g} \exp(\theta_j^T x)}.$$

A single observation is multinomially distributed, i.e.,

$$\mathcal{L}_i = \mathbb{P}(Y^{(i)} = y^{(i)} | x^{(i)}, \theta_1, \ldots, \theta_g) = \prod_{j=1}^{g} \pi_j(x^{(i)})^{\mathbb{1}_{\{y^{(i)}=j\}}}.$$

Under the assumption that the observations are conditionally independent the likelihood of the data can be expressed, s.t.

$$\mathcal{L} = \mathbb{P}(Y^{(1)} = y^{(1)}, \ldots, Y^{(n)} = y^{(n)} | x^{(1)}, \ldots, x^{(n)}, \theta_1, \ldots, \theta_g) = \prod_{i=1}^{n} \prod_{j=1}^{g} \pi_j(x^{(i)})^{\mathbb{1}_{\{y^{(i)}=j\}}}.$$

(By following the maximum likelihood principle, we should look for parameters $\theta_1, \ldots, \theta_g$, which maximize the expression above.)
Now we want the empirical risk to be a *sum* of loss function values, not a *product* recall:

$$\mathcal{R}_{\text{emp}} = \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)}\right)\right).$$

We can turn the product into a sum by taking its log since the same parameters maximize the expressions. Finally, we convert the maximization into minimization by multiplying with -1. So we end up with the so-called cross-entropy loss function:

$$L\left(y, f(\mathbf{x})\right) = -\sum_{j=1}^{g} \mathbb{1}_{\{y=j\}} \log[\pi_j(x)].$$

We see that for the softmax regression the loss function is equal to the negative log-likelihood of one observation. Thus the associated empirical risk is the negative log-likelihood of the complete data set.

c) Since the subtraction of any fixed vector from all θ_k does not change the prediction, one set of parameters is "redundant". Thus we set $\theta_g = (0, \ldots, 0)$. Hence for g classes we get $g - 1$ discriminant functions from the softmax $\hat{\pi}_1(x), \ldots, \hat{\pi}_{g-1}(x)$ which can be interpreted as probability. The probability for class g can be calculated by using $\hat{\pi}_g = 1 - \sum_{k=1}^{g-1} \hat{\pi}_k(x)$. To estimate the class we are using majority vote:

$$\hat{y} = \arg\max_k \hat{\pi}_k(x)$$

The parameter of the softmax regression is defined as parameter matrix where each class has its own parameter vector θ_k, $k \in \{1, \ldots, g-1\}$:

$$\theta = [\theta_1, \ldots, \theta_{g-1}]$$

Exercise 3: logistic regression – softmax hyperplane

Summarize the logistic regression model (shortly!). Show that the decision boundaries, which are defined as the area where the estimated probability for class 1 is exactly a specific threshold $a \in [0, 1]$, is always a hyperplane between both classes. What happens if you are choosing $a = 0.5$? **Solution 3:**

Logistic regression is a classification model, that estimates posterior probabilities $\pi(x)$ by linear functions in x. For a binary classification problem the model can be written as:

$$\hat{y} = 1 \quad \Leftrightarrow \quad \pi(x) = \frac{1}{1 + \exp(-x^T\theta)} \geq a$$

For the decision boundary we have to set $\pi(x) = a$. Solving this equation for x yields:

$$\begin{aligned}
&\frac{1}{1 + \exp(-x^T\theta)} = a \\
\Leftrightarrow\ & 1 + \exp(-x^T\theta) = a^{-1} \\
\Leftrightarrow\ & \exp(-x^T\theta) = a^{-1} - 1 \\
\Leftrightarrow\ & \exp(-x^T\theta) = a^{-1} - 1 \\
\Leftrightarrow\ & -x^T\theta = \log(a^{-1} - 1) \\
\Rightarrow\ & x^T\theta = -\log(a^{-1} - 1)
\end{aligned}$$

For $a = 0.5$ we get:

$$x^T\theta = -\log(0.5^{-1} - 1) = -\log(2 - 1) = -\log(1) = 0$$

Exercise 4: decision boundaries with `mlr3`

Choose some of the classifiers already introduced in the lecture and visualize their decision boundaries for relevant hyperparameters. Use `mlbench::mlbench.spirals` to generate data and use `plot_learner_prediction` for visualization. To refresh your knowledge about `mlr3` you can take a look at `https://mlr3book.mlr-org.com/basics.html`.

Solution 4:

See R code

Exercise 5: k-NN classification

Let the 2D feature vectors in the following figure be with two different class labels (triangles and circles). Classify the point (7,6) - represented by a square in the picture - with a k-nearest neighbor classifier. Distance function should be the L_1 norm (Manhattan distance):

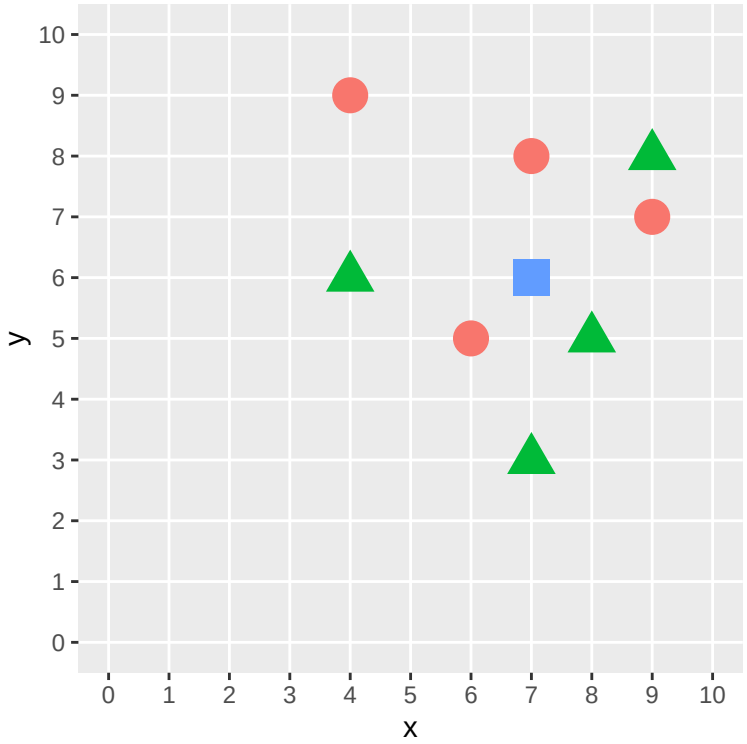
$$d_{\text{manhattan}}(x, \tilde{x}) = \sum_{j=1}^{p} |x_j - \tilde{x}_j|$$

As a decision rule, use the unweighted number of the individual classes in the k-next Neighbor Quantity, i. e. the point is assigned to the class that represents most k-nearest neighbors.

a) $k = 3$

b) $k = 5$

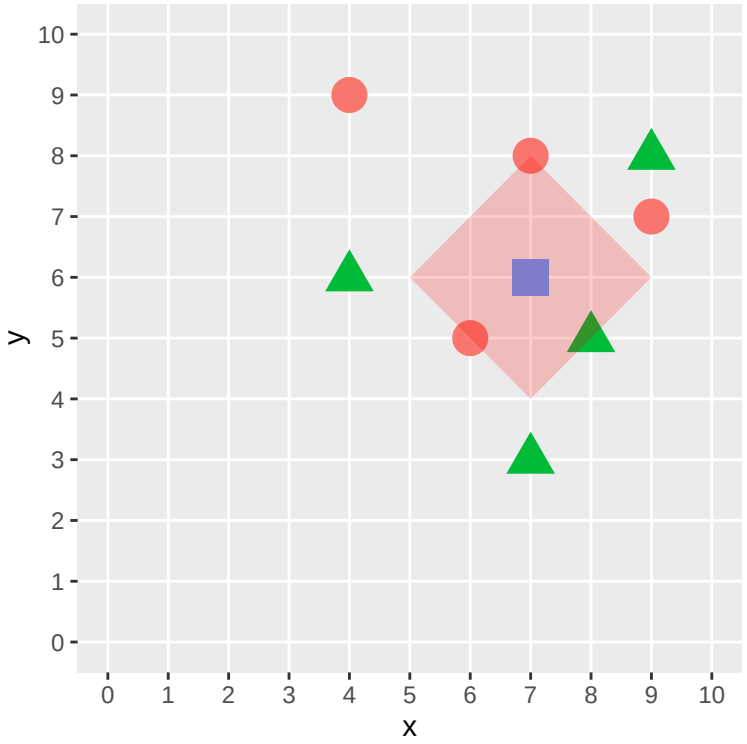
c) $k = 7$



Solution 5:

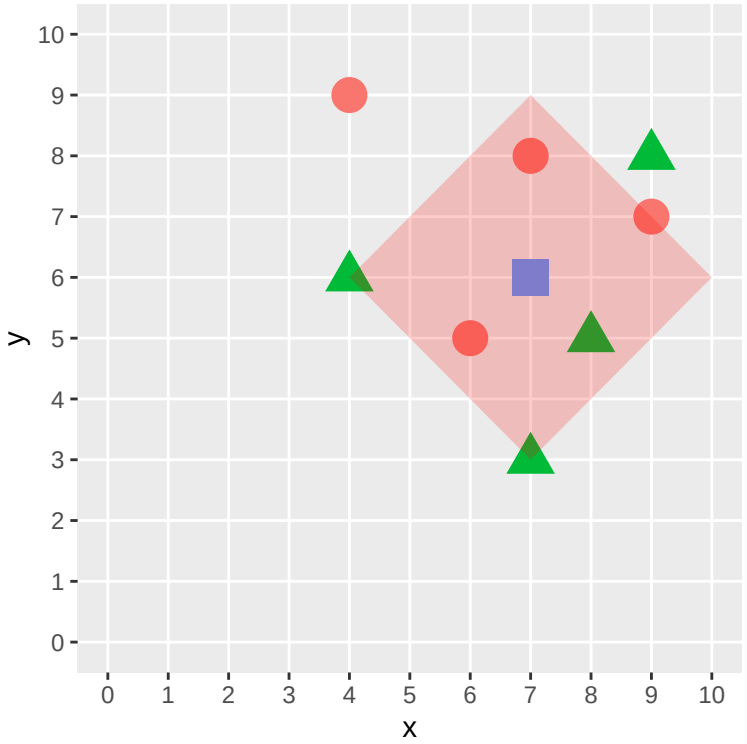
a) $k = 3$

2 circles and 1 triangle, so our point is also a circle



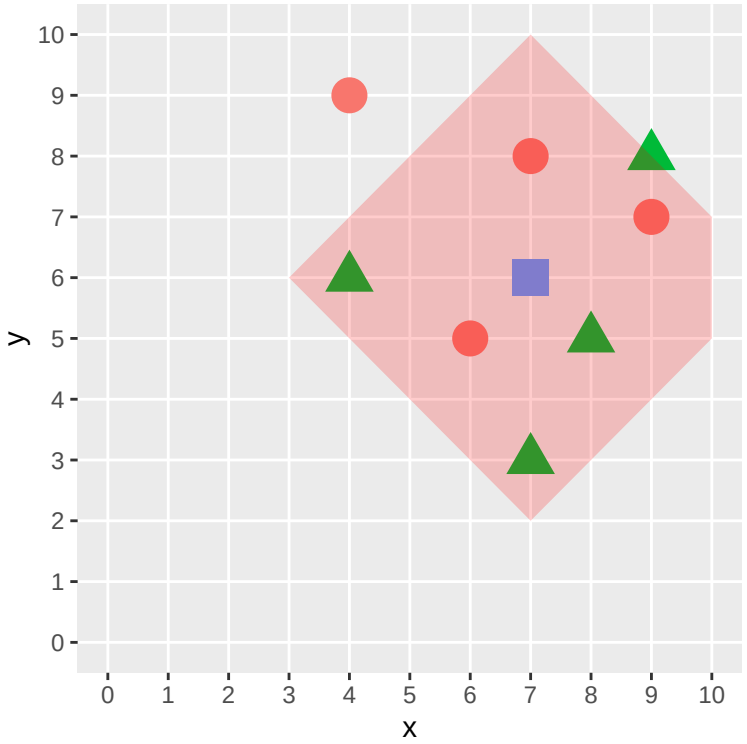
b) $k = 5$

3 circles and 3 triangles, we have to specify beforehand what to do in case of a tie



c) $k = 7$

3 circles and 4 triangles, so our point is also a triangle



Exercise 6: k-NN from scratch

Implement a simple version of a K-nearest-neighbour classifier.

```
myknn = function(target, traindata, testdata, k)
```

The function should return a factor of predicted classes from `testdata`. `target` is the name of the target variable in both `data.frames`. Some hints:

- Your function only needs to work for numeric features.
- Use the euclidean distance.
- Do not overengineer your solution. Keep it simple and do not think too much about efficiency.

Test your implementation on the `iris` data set for $k = 1, 2, 7$. Split your data set 10 times in $\frac{2}{3}$ training and $\frac{1}{3}$ test data. Measure training and test error in each split.

Solution 6:

See R code

Exercise 7: QDA from scratch

Implement you own version of QDA:

```
trainQDA = function(target, data)
predictQDA = function(taget,data)
```

The first function should return the fitted model (use an adequate structure for the model!). The second function should return a factor of classes for `type = 'class'` and a matrix of predicted probabilities for `type = 'prob'`. Your method does only need to work for numeric features. Check your implementation on the `iris` dataset and compare your results of both types with the `qda()` function from the package `MASS`.

Solution 7:

See R code

Exercise 8: naive Bayes

You are given the following table with the target variable `Banana`:

ID	Color	Form	Origin	Banana ?
1	yellow	oblong	imported	yes
2	yellow	round	domestic	no
3	yellow	oblong	imported	no
4	brown	oblong	imported	yes
5	brown	round	domestic	no
6	green	round	imported	yes
7	green	oblong	domestic	no
8	red	round	imported	no

a) We want to use a naive Bayes classifier to predict whether a new fruit is a Banana or not. Calculate the posterior probability $\pi(x)$ for a new observation $(\text{yellow}, \text{round}, \text{imported})$. How would you classify the object?

b) Assume you have an additional feature "Length", which measures the length in cm. Describe in 1-2 sentences how you would handle this numeric feature with Naive Bayes.

Solution 8:

a) When using the naive Bayes classifier, the features $x := (x_{\text{Color}}, x_{\text{Form}}, x_{\text{Origin}})$ given the category $y \in \{\text{yes}, \text{no}\}$ are assumed to be conditionally independent of each other, s.t.

$$p((x_{\text{Color}}, x_{\text{Form}}, x_{\text{Origin}})|y = k) = p(x_{\text{Color}}|y = k) \cdot p(x_{\text{Form}}|y = k) \cdot p(x_{\text{Origin}}|y = k).$$

For the posterior probabilities $\pi_k(x)$ it holds that

$$\begin{aligned} \pi_k(x) &\propto \underbrace{\pi_k \cdot p(x_{\text{Color}}|y = k) \cdot p(x_{\text{Form}}|y = k) \cdot p(x_{\text{Origin}}|y = k)}_{=:\alpha_k(x)} \\ &\iff \exists c \in \mathbb{R} : \pi_k(x) = c \cdot \alpha_k(x), \end{aligned}$$

where π_k is the prior probability of class k. From this and since the posterior probabilities need to sum up to 1, it holds that

$$\begin{aligned} 1 &= c \cdot \alpha_{\text{yes}}(x) + c \cdot \alpha_{\text{no}}(x) \\ \iff c &= \frac{1}{\alpha_{\text{yes}}(x) + \alpha_{\text{no}}(x)}. \end{aligned}$$

This means in order to compute $\pi_{\text{yes}}(x)$ the scores $\alpha_{\text{yes}}(x)$ and $\alpha_{\text{no}}(x)$ are needed.
Now we want to compute for a new fruit the posterior probability $\hat{\pi}_{yes}((\text{yellow}, \text{round}, \text{imported}))$.
Note that we do not know the *true* prior probability and the *true* conditional densities. Here -since the target and the features are categorical- we can estimate them with the relative frequencies encountered in the data, s.t.

$$\begin{aligned} \hat{\alpha}_{\text{yes}}(x) &= \hat{\pi}_{yes} \cdot \hat{p}(\text{yellow}|y = \text{yes}) \cdot \hat{p}(\text{round}|y = \text{yes}) \cdot \hat{p}(\text{imported}|y = \text{yes}) \\ &= \hat{\mathbb{P}}(y = \text{yes}) \cdot \hat{\mathbb{P}}(x_{\text{Color}} = \text{yellow}|y = \text{yes}) \cdot \hat{\mathbb{P}}(x_{\text{Form}} = \text{round}|y = \text{yes}) \cdot \hat{\mathbb{P}}(x_{\text{Origin}} = \text{imported}|y = \text{yes}) \\ &= \frac{3}{8} \cdot \frac{1}{3} \cdot \frac{1}{3} \cdot 1 = \frac{1}{24} \approx 0.042, \\ \hat{\alpha}_{\text{no}}(x) &= \hat{\pi}_{no} \cdot \hat{p}(\text{yellow}|y = \text{no}) \cdot \hat{p}(\text{round}|y = \text{no}) \cdot \hat{p}(\text{imported}|y = \text{no}) \\ &= \hat{\mathbb{P}}(y = \text{no}) \cdot \hat{\mathbb{P}}(x_{\text{Color}} = \text{yellow}|y = \text{no}) \cdot \hat{\mathbb{P}}(x_{\text{Form}} = \text{round}|y = \text{no}) \cdot \hat{\mathbb{P}}(x_{\text{Origin}} = \text{imported}|y = \text{no}) \\ &= \frac{5}{8} \cdot \frac{2}{5} \cdot \frac{3}{5} \cdot \frac{2}{5} = \frac{3}{50} = 0.06. \end{aligned}$$

At this stage we can already see that the predicted label is "no", since $\hat{\alpha}_{\text{no}}(x) = 0.06 > \frac{1}{24} = \hat{\alpha}_{\text{yes}}(x)$.
With this we can calculate the posterior probability

$$\hat{\pi}_{\text{yes}}(x) = \frac{\hat{\alpha}_{\text{yes}}(x)}{\hat{\alpha}_{\text{yes}}(x) + \hat{\alpha}_{\text{no}}(x)} \approx 0.41.$$

Corresponding R-Code:

```
df_banana <- data.frame(
  Color = as.factor(
    c("yellow", "yellow", "yellow", "brown", "brown", "green", "green", "red")),
  Form = as.factor(
    c("oblong", "round", "oblong", "oblong", "round", "round", "oblong", "round")),
  Origin = as.factor(
    c("imported", "domestic", "imported", "imported", "domestic", "imported",
    "domestic", "imported")),
  Banana = as.factor(c("yes", "no", "no", "yes", "no", "yes", "no", "no"))
)

new_fruit <- data.frame(Color = "yellow", Form = "round", Origin = "imported", Banana = NA)
df_banana <- rbind(df_banana, new_fruit)

library(mlr3)
library(mlr3learners)

nb_learner <- lrn("classif.naive_bayes",
                  predict_type = "prob")

banana_task <- TaskClassif$new(
  id = "banana",
  backend = df_banana,
  target = "Banana"
)

nb_learner$train(banana_task, row_ids=1:8)

nb_learner$predict(banana_task, row_ids = 9)
```

```
## <PredictionClassif> for 1 observations:
##  row_ids truth response   prob.no  prob.yes
##        9  <NA>       no 0.5901639 0.4098361
```

b) For the distribution of a numerical feature given the the category we need to specify a probability distribution with continuous support. For example, for the information x_{Length} we could assume that $p(x_{\text{Length}}|y = \text{yes}) \sim \mathcal{N}(\mu_{\text{yes}}, \sigma^2_{\text{yes}})$ and $p(x_{\text{Length}}|y = \text{no}) \sim \mathcal{N}(\mu_{\text{no}}, \sigma^2_{\text{no}})$. (To estimate these normal distributions one would need to estimate their parameters $\mu_{\text{yes}}, \mu_{\text{no}}, \sigma^2_{\text{yes}}, \sigma^2_{\text{no}}$ on the data respectively)

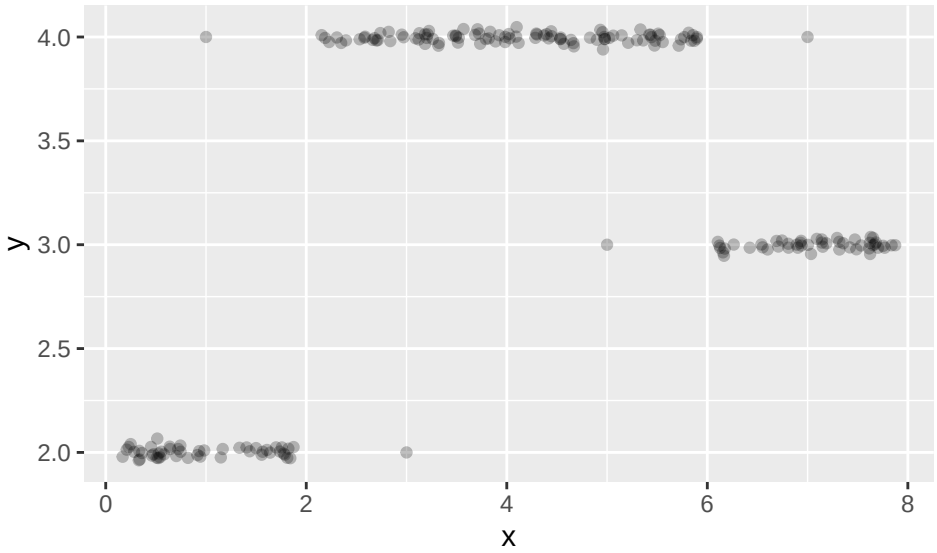
Exercise 9: classifying `ionosphere`

Take a look at the Ionosphere dataset from the `mlbench` package and descripe the dataset shortly. You don't have to summarize all its statistical properties but rather what kind of classification problem it is. Which classifiers from the lecture seem to be applicable for the dataset? Descripe independently of your code how you proceeded to in your analysis and what your results are.
Note: You don't have to check every single classifier in every configuration, more importantly is a smart approach with moderate effort. Use `mlr` for your analysis. **Solution 9:**

No model solution

Further exercises

Exercise 10: WS2020/21, first, question 1



The above plot shows $\mathcal{D} = \left(\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(n)}, y^{(n)}\right)\right)$, a data set with $n = 200$ observations of a continuous target variable y and a continuous, 1-dimensional feature variable $\mathbf{x}$. In the following, we aim at predicting y with a machine learning model that takes $\mathbf{x}$ as input.

(a) Since the data seem to fall in 3 quite well-separable classes, we now want to apply a classification model instead of the regression model in a). To prepare the data for classification, we categorize the target variable y in 3 classes and call the transformed target variable z, as follows:

$$z^{(i)} = \left\{ \begin{array}{ll} 1, \text{ if } & -\infty < y^{(i)} \leq 2.5 \\ 2, \text{ if } & 2.5 < y^{(i)} \leq 3.5 \\ 3, \text{ if } & 3.5 < y^{(i)} < \infty \end{array} \right\}$$

Now we can apply quadratic discriminant analysis (QDA):

(i) Estimate the class means $\mu_k = \mathbb{E}(\mathbf{x}|z = k)$ for each of the three classes $k \in \{1, 2, 3\}$ visually from the plot. Do not overcomplicate this, a rough estimate is sufficient here.

(ii) Make a hand-drawn plot that visualizes the different estimated densities per class.

(iii) How would your drawing from (ii) change if we used linear discriminant analysis (LDA) instead of QDA? Explain your answer.

(iv) Why is QDA for this data preferable over LDA?

(b) Given are two new observations $\mathbf{x}_{*1} = -10$ and $\mathbf{x}_{*2} = 7$. State the prediction for each of the two models

(i) regression tree

(ii) QDA

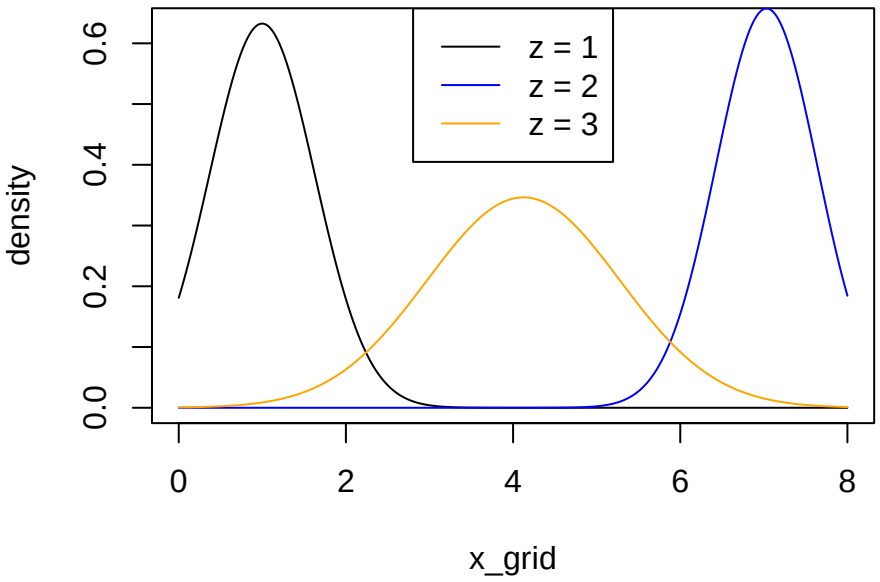
and explain how you derived the predictions.

(c) Discuss in 1-2 sentences which of the 2 models (regression tree, QDA) you would prefer for modeling the data and explain your decision.

Solution 10:

(a)

(i) $\mu_1 = 1$, $\mu_2 = 7$, $\mu_3 = 4$



(ii)

(iii) Variances would be all equal. Assumption of LDA is equal variances, i.e., estimatated models will always have equal variances, no matter if this fits the data or not

(iv) Variances seem not to be equal, this is only captured in QDA

(b)

(i) $\hat{y}_{*1} = 2$, $\hat{y}_{*2} = 3$,

(ii) $\hat{z}_{*1} = 3$, since the variance of class 3 is higher, the density will overshoot the density of class 1. $\hat{z}_{*2} = 2$, obviously highest posterior here.

(c) E.g.,

- CART better than LM because I do not have to specify those indicator functions manually and estimate the split points manually, CART does this data driven
- For QDA we have to throw away information of y, this favors CART
- QDA predicts the middle class (3) for very extreme observations, this does not seem right. However, we do not know how data behave outside the bounds of x.
- QDA assumes gaussian distributions which is clearly not the case.

Exercise 11: WS2020/21, first, question 4

The table below shows $\mathcal{D} = \left(\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(n)}, y^{(n)}\right)\right)$, a data set with $n = 8$ observations of a binary target variable y containing the information if the object is a `Banana` or not and a 4-dimensional feature vector $\mathbf{x}$, containing `Color`, `Form`, `Origin` (categorical features) and `Length` (continuous feature) of the object. In the following, we aim at predicting `Banana` with a machine learning model that takes $\mathbf{x}$ as input.

ID	Color	Form	Origin	Length [cm]	Banana
1	yellow	oblong	imported	15	yes
2	yellow	round	domestic	5	no
3	yellow	oblong	imported	10	no
4	brown	oblong	imported	17	yes
5	brown	round	domestic	16	no
6	green	round	imported	13	yes
7	green	oblong	domestic	25	no
8	red	round	imported	7	no

We want to use a naive Bayes classifier to predict the label of a new observation.

(a) Calculate the posterior probability $\pi(\mathbf{x}_*) = \mathbb{P}(y = \text{yes}|\mathbf{x}_*)$ for a new observation $\mathbf{x}_* = (\text{green}, \text{oblong}, \text{imported}, 14)^\top$. Explain every step thoroughly. (Hint: At some point you will have to compute values of Gaussian densities - use R for this step.)

(b) How would you classify the new observation? Explain your answer.

Solution 11:

(a) The features $\mathbf{x} := (x_{\text{Color}}, x_{\text{Form}}, x_{\text{Origin}}, x_{\text{Length}})$ given the category $y \in \{\text{yes}, \text{no}\}$ are assumed to be conditionally independent of each other (since we are using Naive Bayes), s.t.

$$p(\mathbf{x}|y = k) = p(x_{\text{Color}}|y = k) \cdot p(x_{\text{Form}}|y = k) \cdot p(x_{\text{Origin}}|y = k) \cdot p(x_{\text{Length}}|y = k).$$

For the posterior probabilities $\pi_k(\mathbf{x}) = \mathbb{P}(y = k|\mathbf{x})$ it holds with Bayes' Theorem:

$$\begin{aligned}\pi_k(\mathbf{x}) &\propto \underbrace{\pi_k \cdot p(\mathbf{x}|y = k)}_{=:\alpha_k(\mathbf{x})} \\ &\iff \exists c \in \mathbb{R} : \pi_k(\mathbf{x}) = c \cdot \alpha_k(\mathbf{x}),\end{aligned}$$

where $\pi_k = \mathbb{P}(y = k)$ is the prior probability of class k.

From this and since the posterior probabilities need to sum up to 1, it holds that

$$\begin{aligned}1 &= c \cdot \alpha_{\text{yes}}(\mathbf{x}) + c \cdot \alpha_{\text{no}}(\mathbf{x}) \\ \iff c &= \frac{1}{\alpha_{\text{yes}}(\mathbf{x}) + \alpha_{\text{no}}(\mathbf{x})}.\end{aligned}$$

This means, the scores $\alpha_{\text{yes}}(x)$ and $\alpha_{\text{no}}(x)$ are needed to compute

$$\pi_{\text{yes}}(\mathbf{x}) = \frac{\alpha_{\text{yes}}(\mathbf{x})}{\alpha_{\text{yes}}(\mathbf{x}) + \alpha_{\text{no}}(\mathbf{x})}.$$

For the new observation $\mathbf{x}_*$ we have to estimate the respective probabilities and densities. For the categorical features, we can simply compute relative frequencies. For the continuous features `Length` we have to estimate the densities per class and evaluate those at the value 14. Doing this with R we end up with:

```
## [1] 0.1760327
## [1] 0.04863352
```

k	$\hat{\pi}_k$	$\hat{p}(\text{green}\|y = k)$	$\hat{p}(\text{oblong}\|y = k)$	$\hat{p}(\text{imported}\|y = k)$	$\hat{p}(14\|y = k)$
yes	3/8	1/3	2/3	3/3	0.176
no	5/8	1/5	2/5	2/5	0.049

$$\begin{aligned}\hat{\alpha}_{\text{yes}}(x) &= \hat{\pi}_{yes} \cdot \hat{p}(\text{green}|y = \text{yes}) \cdot \hat{p}(\text{oblong}|y = \text{yes}) \cdot \hat{p}(\text{imported}|y = \text{yes}) \cdot \hat{p}(14|y = \text{yes}) \\ &= \frac{3}{8} \cdot \frac{1}{3} \cdot \frac{2}{3} \cdot 1 \cdot 0.176 \approx 0.0147, \\ \hat{\alpha}_{\text{no}}(x) &= \hat{\pi}_{no} \cdot \hat{p}(\text{green}|y = \text{no}) \cdot \hat{p}(\text{oblong}|y = \text{no}) \cdot \hat{p}(\text{imported}|y = \text{no}) \cdot \hat{p}(14|y = \text{no}) \\ &= \frac{5}{8} \cdot \frac{1}{5} \cdot \frac{2}{5} \cdot \frac{2}{5} \cdot 0.049 \approx 0.00098.\end{aligned}$$

With this we can calculate the posterior probability

$$\hat{\pi}_{\text{yes}}(x) = \frac{\hat{\alpha}_{\text{yes}}(x)}{\hat{\alpha}_{\text{yes}}(x) + \hat{\alpha}_{\text{no}}(x)} \approx 0.937.$$

(b) Classification as yes = `banana`. We have to define a threshold, observations with a posterior probability equal or above the threshold are hard labeled as yes, others as no. The optimal threshold has to be chosen, e.g., inspecting ROC measures. With default 0.5 we end up with the above classification.

Ideas & exercises from other sources