Solution 1:

(a) 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:

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

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

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

For $a = 0.5$ we get:

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

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

(b) When the threshold $a = 0.5$ is chosen the Bernoulli loss, which penalizes the missclassifications $L(\hat{y} = 0|y = 1)$ and $L(\hat{y} = 1|y = 0)$ equally, is minimized. This means $a = 0.5$ is a sensible threshold if one does not need to avoid one type of missclassification 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.

Solution 2:
See R code **sol_mlr_decision_boundaries.R**

Solution 3:

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 $\theta = \theta_2 - \theta_1$ 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:

$$\hat{y} = \operatorname*{arg\,max}_k \hat{\pi}_k(x)$$

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

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

Solution 4:

a) When using the naive Bayes classifier here 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 $\pi_k(x)$ we are interested in it holds that

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

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

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

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:

```r
df_banana <- data.frame(
  Color = c("yellow", "yellow", "yellow", "brown", "brown", "green", "green", "red"),
  Form = c("oblong", "round", "oblong", "oblong", "round", "round", "oblong", "round"),
  Origin = c("imported", "domestic", "imported", "imported", "domestic", "imported",
    "domestic", "imported"),
  Banana = 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)
```

```r
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_id 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)