Problem 1. (Naive Bayes)
In this problem we aim to predict the possibility of failure during manufacturing an item based on the pressure and temperature applied to the object. Consider the following hypothetical dataset consisting of 10 measurements:

temperature, $°F$	281	126	111	244	288	428	124	470	301	323
pressure, MPa	262	125	282	226	119	155	209	291	292	281
failure	1	0	0	0	0	1	0	1	0	1

1. Given each of the two classes 1 (failure) and 0 (no failure), determine the mean and the variance for each feature vector. Based on this, determine the corresponding 2 Gaussian distributions for each of the two features.

   Solution: We can compute the mean $\mu_{F,1}$ of the first feature vector given that the manufactured item is a failure as follows:

   $$\mu_{F,1} = \frac{1}{4}(281 + 428 + 470 + 323) = 375.5.$$

   The mean $\mu_{MPa,1}$ of the second feature vector given that the manufactured item is a failure and the means $\mu_{F,0}$ and $\mu_{MPa,0}$ of both feature vectors given that the manufactured item is not a failure can be computed analogously and are given by:

   $$\mu_{F,1} = 375.5,\ \mu_{MPa,1} = 247.25,\ \mu_{F,0} = 199,\ \mu_{MPa,0} \approx 208.83.$$

   We can compute the unbiased variance $s^2_{F,1}$ of the first feature vector given that the manufactured item is a failure as follows:

   $$\sigma^2_{F,1} = \frac{1}{3}\left((281 - 375.5)^2 + (428 - 375.5)^2 + (470 - 375.5)^2 + (323 - 375.5)^2\right) = 7991.0$$

   Note that in practice, using the biased variance estimation

   $$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N}$$

   or unbiased variance estimation

   $$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N - 1}$$

   give close results as N is generally large.

   The variances of $\sigma^2_{MPa,1}$, $\sigma^2_{F,0}$ and $\sigma^2_{MPa,0}$ are computed analogously and are given by:

   $$\sigma^2_{MPa,1} \approx 3926.917,\ \sigma^2_{F,0} = 3836.5,\ \sigma^2_{MPa,0} \approx 3457.69.$$

   Therefore, the Gaussian distribution of the feature temperature conditioned on the manufactured item being a failure is $\mathcal{N}(\mu_{F,1}, \sigma^2_{F,1})$ and of the feature pressure conditioned on the manufactured item being a failure is $\mathcal{N}(\mu_{MPa,1}, \sigma^2_{MPa,1})$. The Gaussian distribution of the feature temperature conditioned on the manufactured item not being a failure is $\mathcal{N}(\mu_{F,0}, \sigma^2_{F,0})$ and of the feature pressure conditioned on the manufactured item not being a failure is $\mathcal{N}(\mu_{MPa,0}, \sigma^2_{MPa,0})$.

2. Use the Gaussian Naive Bayes approach to determine whether the test point $x^{\text{test}} = (270, 170)$ would lead to a failure or not.

   Solution: The probability $P\left(\text{failure} = 1 \mid x^{\text{test}}\right)$ that the test point is a failure can be computed as follows:

$$\begin{aligned} P\left(\text{failure} = 1 \mid x^{test}\right) =& \frac{f_{x^{\text{test}}|\text{failure}=1}(x^{\text{test}})P(\text{failure} = 1)}{f_{x^{\text{test}}(x^{\text{test}})}} \\ \propto& f_{x_F^{\text{test}}|\text{failure}=1}(x_F^{\text{test}}) f_{x_{MPa}^{\text{test}}|\text{failure}=1}(x_{MPa}^{\text{test}}) P(\text{failure} = 1) \\ \propto& f(270 \mid \mu_{F,1}, \sigma_{F,1}^2) * f(170 \mid \mu_{MPa,1}, \sigma_{MPa,1}^2) * \frac{4}{10} \\ \propto& 2.64 * 10^{-6}, \end{aligned}$$

   where

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

   is the Gaussian distribution with mean μ and standard deviation σ. The probability $P\left(\text{failure}{=}0 \mid x_{test}\right)$ that the test point is not a failure can be computed as follows:

$$\begin{aligned} P\left(\text{failure} = 0 \mid x^{test}\right) =& \frac{f_{x^{\text{test}}|\text{failure}=0}(x^{\text{test}})P(\text{failure} = 0)}{f_{x^{\text{test}}(x^{\text{test}})}} \\ \propto& f_{x_F^{\text{test}}|\text{failure}=0}(x_F^{\text{test}}) f_{x_{MPa}^{\text{test}}|\text{failure}=0}(x_{MPa}^{\text{test}}) P(\text{failure} = 0) \\ \propto& f(270 \mid \mu_{F,0}, \sigma_{F,0}^2) * f(170 \mid \mu_{MPa,0}, \sigma_{MPa,0}^2) * \frac{6}{10} \\ \propto& 1.05 * 10^{-5}. \end{aligned}$$

   Therefore, the product is more likely to be a success.

Problem 2. (kNN)
In class, we discussed that the kNN approach can be used for both regression and classification. Consider a regression problem with a dataset $\{x^i, y^i\}_{i=1}^N$, $x_i \in \mathbb{R}^d$, $y^i \in \mathbb{R}$.

1. Explain why you need to normalize the data. Write the equation for data normalization based on the so-called Z-score.

   Solution: Some features may have significantly larger range than others and thus dominate the distance calculation and classification. We use $\{x_j^i\}_{i=1}^N$ to denote the j-th feature. We can then normalize the input by:

$$z_j^i = \frac{x_j^i - \mu_j}{\sigma_j},$$

   where $\mu_j = \frac{1}{N}\sum_{i=1}^N x_j^i$ is the mean (average) and $\sigma_j = (\frac{1}{N}\sum_{i=1}^N (x_j^i - \mu_j)^2)^{1/2}$ is the standard deviation of $\{x_j^i\}_{i=1}^N$.

2. Given a test point x, write the formula for determining the estimated label $\hat{y} \in \mathbb{R}$ using $K = 1$ nearest neighbor, and 2-norm distance (referred to as Euclidean distance) as well as the 1-norm distance (referred to as Manhattan distance).

Solution: For each case, we have to find the closest data point, where closeness is based on the distance metric we are using.

Recall that 2-norm and 1-norm distances between two vectors, $x, x' \in \mathbb{R}^d$ are defined as follows:

$$\|x - x'\|_2 = \big(\sum_{j=1}^{d}(x_j - x'_j)^2\big)^{1/2},$$

$$\|x - x'\|_1 = \sum_{j=1}^{d}|x_j - x'_j|$$

Let $\hat{i}$ denote the index of the closest data point. The label $\hat{y}$ is the label of the closest data point, i.e. $\hat{y} = y^{\hat{i}}$.

For the L2 distance we have $\hat{i} = \arg\min_{i\in\{1,\dots,d\}} \|x^i - x\|_2$. For the L1 distance we have $\hat{i} = \arg\min_{i\in\{1,\dots,d\}} \|x^i - x\|_1$.

3. Repeat the same task as with $K = 2$ nearest neighbors. Hint: for regression, we take the label corresponding to the average of the K neighbor labels.

   Solution: For each case, we find the two closest data points based on the distance metric. Let i_1 and i_2 denote the label of the two closest data points. Then, $\hat{y} = \frac{y^{i_1}+y^{i_2}}{2}$.

4. How would you determine which distance metric and which K to use?

   Solution: The distance metric and the K are hyper-parameters of the kNN approach that we need to determine based on our data set. We can divide the data set into a training, validation, and test set. We can select a set of candidate distance metrics (e.g., l_1, l_2 and l_∞ distances) and a set of candidate nearest neighbors (e.g., $K = 1$, $K = 2$, ...). For each distance metric and value of K, we can evaluate the validation error. After enumerating all the candidate distances and candidate K's, we can select the distance measure and K that achieves the lowest error.

Problem 3. (Naive Bayes, k-Nearest Neighbor, taken from Exam 2023)
Company X is hiring employees and aims to hire those who spend less time watching videos online. Thus, it wants to predict an applicant's potential to watch online videos based on past employee data. In particular, for each of the 1000 past employees, it has recorded whether they have a high GPA, and whether they watch online videos. Among those who do not watch videos, some play sports (none of those who watch videos play sports). The survey is summarized below.

	Employees	Sport	No Video
High GPA	600	150	300
low GPA	400	50	100

Let event A denote having a high GPA, event B denote playing sports, and event C denote not watching videos.

1. Calculate the following: (a) empirical probability of having a high GPA; (b) empirical probability of having a high GPA and playing sports.

   Solution.

$$P(A) = \frac{600}{1000} = \frac{3}{5}$$
$$P(A \cap B) = \frac{150}{1000} = \frac{3}{20}$$

2. Show that the events A and B are not independent.

   Solution.

$$P(A) = \frac{3}{5}$$
$$P(B) = \frac{150 + 50}{1000} = \frac{1}{5}$$
$$P(A \cap B) = \frac{3}{20}$$

   Thus

$$P(A)P(B) = \frac{3}{5} * \frac{1}{5} \neq \frac{3}{20} = P(A \cap B)$$

   which shows that events A and B are not independent.

3. Calculate the following: (a) the empirical conditional probability of event A given event C; (b) the empirical conditional probability of events A and B given event C.

   Solution.
   (a)

$$P(A|C) = \frac{P(A \cap C)}{P(C)} = \frac{300}{400} = \frac{3}{4}.$$

   (b)

$$P(A \cap B|C) = \frac{P(A \cap B \cap C)}{P(C)} = \frac{150}{400} = \frac{3}{8}.$$

4. Show that conditioned on C, the events A and B are independent.

   Solution.

$$P(A|C) = \frac{3}{4}$$
$$P(B|C) = \frac{P(B \cap C)}{P(C)} = \frac{150 + 50}{400} = \frac{1}{2}$$
$$P(A \cap B|C) = \frac{3}{8}$$

Thus $P(A \cap B|C) = P(A|C) * P(B|C) = \frac{3}{8}$ and conditioned on C, events A and B are independent.

The company aims to have a classifier for an employee by using the exact GPA ($x_1 \in \mathbb{R}_+$) and the average number of hours of sport played per week ($x_2 \in \mathbb{R}_+$). Based on past data, it fits two probability density functions conditioned on Y, where Y corresponds to watching videos ($Y = 1$) or not ($Y = 0$). These are denoted by $f_{x_1|Y} : \mathbb{R} \to \mathbb{R}$ and $f_{x_2|Y} : \mathbb{R} \to \mathbb{R}$.

5. Formulate the Naive Bayes classifier.

   Solution. The probability that Y is 1 given x is given by

   $$P(Y = 1|x) = \frac{f_{x_1|1}(x_1) * f_{x_2|1}(x_2) * P(Y = 1)}{f_x(x)}.$$

   Analogously, the probability that Y is 0 given x is given by

   $$P(Y = 0|x) = \frac{f_{x_1|0}(x_1) * f_{x_2|0}(x_2) * P(Y = 0)}{f_x(x)}.$$

   Naive Bayes classifier classifies $Y = 1$ given x if

   $$f_{x_1|1}(x_1) * f_{x_2|1}(x_2) * P(Y = 1) \geq f_{x_1|0}(x_1) * f_{x_2|0}(x_2) * P(Y = 0)$$

   and $Y = 0$ otherwise.

6. For an applicant, the company has obtained x_1, x_2 from which it has evaluated $f_{x_1|0}(x_1) = 0.25$, $f_{x_2|0}(x_2) = 2.00$, $f_{x_1|1}(x_1) = 0.20$ and $f_{x_2|1}(x_2) = 2.20$. Based on the Naive Bayes classifier, would this person be likely to watch online videos at work?

   Solution. The probability that Y is 1 given x is given by

   $$\begin{aligned} P(Y = 1|x) &\propto f_{x_1|1}(x_1) * f_{x_2|1}(x_2) * P(Y = 1) \\ &\propto 0.2 * 2.2 * \frac{600}{1000} \propto 0.264. \end{aligned}$$

   Analogously, the probability that Y is 0 given x is given by

   $$\begin{aligned} P(Y = 0|x) &\propto f_{x_1|1}(x_1) * f_{x_2|1}(x_2) * P(Y = 0) \\ &\propto 0.25 * 2 * \frac{400}{1000} \propto 0.2. \end{aligned}$$

   Based on the Naive Bayes classifier this person is more likely to watch online videos when they start the job.

7. On a test set obtained from recently hired employees, it was found that the classifier has more false positives than false negatives. The company decided to change the prior on the probability of watching videos (perhaps the new generation has been bored of all the online videos). What term(s) in your Naive Bayes classifier would you change and how?

   Solution. You could reduce the prior of class 1, namely decrease $P(Y = 1)$ and in turn increase the prior on class 0, namely increase $P(Y = 0)$. If the priors are updated based on new data

and the conditional densities are calculated based on this data, then the conditional density functions $f_{x_1|Y}$ and $f_{x_2|Y}$ will change. However, since it is not specified how the conditional density functions $f_{x_1|Y}$ and $f_{x_2|Y}$ are computed in this problem we can not say whether they are affected or not by the change in the priors.

Problem 4. (Neural networks function class expressiveness)
Consider a data set $\{x^i, y^i\}_{i=1}^N$, with $x^i \in \mathbb{R}^d$ and $y^i \in \mathbb{R}^m$. Let our predictor be a neural network $f : \mathbb{R}^d \to \mathbb{R}^m$.

1. Consider a neural network with one hidden layer having 3 nodes and an output layer having m nodes, with activation function $g : \mathbb{R} \to \mathbb{R}$ for each node. How many parameters need to be determined in this network?

   Solution. There are $3(d+1)$ parameters in the first layer, with $3d$ weights and 3 biases. There are $m(3+1)$ parameters in the second layer, with $3m$ weights and m biases.

2. Recall the definition of an affine function from your "Background and notations.pdf" file (posted on Moodle, 9 September - 15 September). Show that if the activation function for each node in the hidden layer and each node in the output layer is the identity, $g(z) = z$, $\forall z \in \mathbb{R}$, then the neural network predictor is the same as a linear predictor with $m(d+1)$ parameters to be determined. Hence, the problem is the same as linear regression.

   Solution. Let $W^{[1]} \in \mathbb{R}^{d\times 3}$, $W^{[0]} \in \mathbb{R}^{3\times m}$ denote the weights from the input to the hidden layer and from the hidden layer to the output layer, respectively. Let $b^{[1]} \in \mathbb{R}^3$ and $b^{[0]} \in \mathbb{R}^m$ denote the biases on the hidden layer and the output layer, respectively. Notice that if $g(z) = z$, then the neural network predictor can be written as

   $$f(x) = (W^{[0]})^T \left((W^{[1]})^T x + b^{[1]}\right) + b^{[0]} = {W^{[0]}}^T {W^{[1]}}^T x + {W^{[0]}}^T b^{[1]} + b^{[0]},$$

   where we are using $x = (x_1, x_2, \ldots, x_d)$. Let $W^T := {W^{[0]}}^T {W^{[1]}}^T \in \mathbb{R}^{m\times d}, b := {W^{[0]}}^T b^{[1]} + b^{[0]} \in \mathbb{R}^m$. Then we have $f(x) = W^T x + b$. We can indeed verify this is an affine function since W^T is a matrix and b is a vector.

   Note: Recall that for any matrix $M \in \mathbb{R}^{a\times b}$, $f(x) = Mx$ is linear: for any $x, x' \in \mathbb{R}^b$, $\alpha, \beta \in \mathbb{R}$, $f(\alpha x + \beta x') = M(\alpha x + \beta x') = \alpha Mx + \beta Mx' = \alpha f(x) + \beta f(x')$.

3. For the case of linear regression above, consider using batches of size N_b for training the network. Consider the square-error loss function. Write the pseudo-code for stochastic gradient descent with this batch size. For simplicity, let $m = 1$.

Solution. Recall the empirical loss function is $L(W) = \frac{1}{N}\sum_{i=1}^N e(\hat{y}^i, y^i)$, where e is the squared error evaluated at a data point (x^i, y^i) and is computed as $e(\hat{y}^i, y^i) = \|\hat{y}^i - y^i\|_2^2$, where $\hat{y}^i = W^T x^i$ with x defined as $(1, x_1, x_2, \ldots, x_d)$ augmented with a constant 1 corresponding to the bias term, and $W \in \mathbb{R}^{d+1}$. By applying chain rule, the gradient can be derived as $\frac{\partial e}{\partial W} \in \mathbb{R}^{d+1}$ evaluated at this data point is

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial \hat{y}^i}\frac{\partial \hat{y}^i}{\partial W} = 2(\hat{y}^i - y^i)x^i. \tag{0.1}$$

Furthermore, the gradient of the loss for just the samples in the batch size is simply $\sum_{i=1}^{N_b} 2(\hat{y}^i - y^i)x^i$, where $i = 1, 2, \ldots, N_b$ denotes the index of the data points in the batch. We can then give the pseudo-code for stochastic gradient descent with this batch size.

$K \leftarrow$ number of training iterations, W initialized randomly, $\alpha \leftarrow$ step size
for $1 \leq i \leq K$ **do**
  Shuffle training points between batches of size b randomly.
  for each batch, let sample indices be $1, 2, \ldots, N_b$ **do**
    Calculate the gradient over the samples in the batch: $\sum_{i=1}^{N_b} 2(\hat{y}^i - y^i)x^i$
    $W \leftarrow W - \frac{\alpha}{N_b} \sum_{i=1}^{N_b} 2(\hat{y}^i - y^i)x^i$
  end for
end for

Problem 5. (Neural networks expressive power)
Consider a neural network

$$f : [-2, 2] \to \mathbb{R}, \quad f(x) = W^{[1]T} g\left(W^{[0]T} x + b^{[0]}\right) + b^{[1]},$$

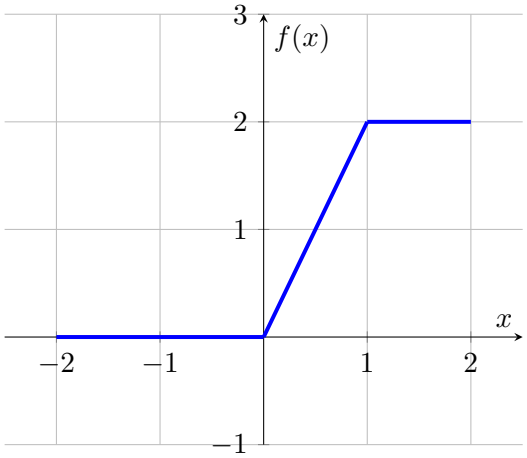
with a single hidden layer and the ReLU activation function[1] $g(x) := \max(0, x)$. Supposing that there are two nodes in the hidden layer, determine the weight matrices $W^{[0]} \in \mathbb{R}^{1\times 2}, W^{[1]} \in \mathbb{R}^{2\times 1}$ and the biases $b^{[0]} \in \mathbb{R}^2, b^{[1]} \in \mathbb{R}$ such that:

1. $f(x) = x$.

   Solution. Since $g(x) - g(-x) = x$, we can choose

   $$W^{[0]} = \begin{bmatrix} 1 & -1 \end{bmatrix}, \; b^{[0]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \; W^{[1]} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \; b^{[1]} = \begin{bmatrix} 0 \end{bmatrix}.$$

2. f has the following graph:



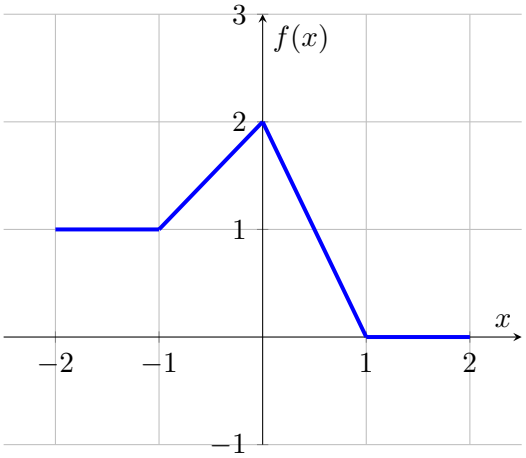
[1] Activation functions are applied elementwise to each node of a hidden layer.

Solution. We can choose

$$W^{[0]} = \begin{bmatrix} 1 & 1 \end{bmatrix}, \; b^{[0]} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \; W^{[1]} = \begin{bmatrix} 2 \\ -2 \end{bmatrix}, \; b^{[1]} = \begin{bmatrix} 0 \end{bmatrix}.$$

Supposing that there are three nodes in the hidden layer, determine the weights $W^{[0]} \in \mathbb{R}^{1\times 3}$, $W^{[1]} \in \mathbb{R}^{3\times 1}$ and the biases $b^{[0]} \in \mathbb{R}^3, b^{[1]} \in \mathbb{R}$ such that:

3. f has the following graph:



Solution. We can choose

$$W^{[0]} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}, \; b^{[0]} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}, \; W^{[1]} = \begin{bmatrix} 1 \\ -3 \\ 2 \end{bmatrix}, \; b^{[1]} = \begin{bmatrix} 1 \end{bmatrix}.$$

Problem 6. (Convolutional neural networks from the python exercise)
Consider the convolutional neural network exercise with the MNIST dataset.

1. Write the dimensionals of the input (features) and the output (labels) for this problem.

   Solution. The input is an image defined by a matrix of dimensions 28×28, thus the total dimension is 784, while the output is the corresponding digit. There are 10 labels (the numbers from 0 to 9), so the output dimensional is 10

2. Our goal is to use the images in the training data to learn a classifier that gives the label of a new handwritten digit. Suppose that a neural network has been trained for this task. How would you measure its accuracy and error rate?

   Solution. The accuracy is the number of correct predictions divided by the number of total predictions. The value is then expressed as a percentage. The error rate is the number of wrong predictions divided by the number of total predictions. It is also equal to 1 minus the accuracy. We would use the test set to determine accuracy and error rate.

3. In the python exercise, you divided the dataset to a training and test set. And then, you used a batch size of 32 for stochastic gradient descent. Write the steps of stochastic gradient descent with a batch size of 32. How many iterations would be in each epoch?

   Solution. Let us denote $\theta = (w, b)$ the set of all weight and biases. To compute the stochastic gradient descent, we iterate the following steps:

   (a) pass a batch through the model to compute the predictions $\hat{y}_i$.

   (b) use a loss function L to compute the difference between the predicted values and the true ones. You can use, for example, the mean squared error:

   $$L(\theta_t) = \frac{1}{32}\sum_{i=1}^{32}(y_i - \hat{y}_i)^2. \tag{0.2}$$

   (c) backward pass the loss to calculate the gradient $\frac{\partial L}{\partial \theta} = \frac{1}{32}\sum_{i=1}^{32}\frac{\partial}{\partial \theta}(y_i - \hat{y}_i)^2$.

   (d) update the parameters using gradient descent with step size α_t:

   $$\theta_{t+1} = \theta_t - \frac{\alpha_t}{32}\sum_{i=1}^{32}\frac{\partial}{\partial \theta}(y_i - \hat{y}_i)^2. \tag{0.3}$$

   The number of iterations needed in each epoch is equal to the training dataset size divided by the batch size:

   $$\text{iterations} = \frac{\text{training dataset size}}{\text{batch size}} = \frac{60000}{32} = 1875.$$

4. Suppose you were to use a logistic regression to learn a classifier that takes an input image and gives the label. What would be the number of parameters (weights and biases) you would have to learn?

   Solution. The logistic regression needs one weight for each feature and one bias. As said above, we have 784 features and 10 regressors (one for each possible class), thus in total we have 10×784 (weights) $+ 10 \times 1$ (biases) $= 7850$ parameters.

5. Now, consider the case that the images get corrupted by noise and the pixel values get permuted. Which of the following approaches is more likely to suffer from accuracy and why? logistic regression, convolutional neural network.

   Solution. The approach more likely to suffer is the convolutional neural network. In fact, it relies on neighborhood properties and the values of pixels nearby. By permutating the pixels, the image loses its structure. You also observed this in your python exercise 9 where you applied permutation to the MNIST dataset.