DD2434 Machine Learning, Advanced Course
Assignment 1

Pawel Herman (adapted from Carl Henrik Ek)

Deadline 12:00 (noon) December 2nd, 2019

You will present the assignment by a written report that you should submit via Canvas. *As the last resort for those who* **do not have access to Canvas**, you can mail to me at paherman@kth.se before the deadline (please include "[mladv18]" in the subject line). From the report it should be clear what you have done and you need to support your claims with results. You are supposed to write down the answers to the specific questions detailed for each task. This report should clearly show how you have drawn the conclusions and come up with the derivations. Your assumptions, if any, should be stated clearly. For the practical part of the task you are not expected to include your code in the report but rather demonstrate and focus on the results of your experiments, i.e. show images and graphs together with your analysis.

Being able to communicate your results and conclusions is a key aspect of any scientific practitioner. It is up to you as an author to make sure that the report clearly shows what you have done. Based on this, and only this, we will decide if you pass the task. No detective work should be needed on our side. Therefore, neat and tidy reports, please! Avoid copying the question content in your reports, just write down the answers using the same numbering as for the questions (Q1-24).

As far as the implementation is concerned, please use any programming/scripting language/environment that you feel comfortable with. I can just recommend Python with its suitable libraries and all the clues regarding the implementation are given here with Python in mind. Matlab is another good option.

The grading of the assignments is as follows,

E Completed Task 1.1 and 1.2.

D E + Completed Task 1.3.

C D + Completed Task 1.4.

A/B C + Completed Tasks 1.5 and 1.6.

As you can notice, it is necessary for you to complete the first two Tasks, 1.1 and 1.2, to pass the Assignment 1. Then for D you should additionally provide correct answers to theoretical questions in Task 1.3, and for C - perform practical Task 1.4. Finally, the completion of Tasks 1.1-1.4 paves the way for the highest grades A (successful completion of all Tasks) and B (builds on C and allows for partial completion of Tasks 1.5 and 1.6).

Abstract

In this assignment we will examine several different aspects of building models of data. In the first task we will look at a supervised scenario where we work with a model of a specific relationship between two different domains. This is a very common problem where we have observations in one domain, say an image of a face and then wish to infer the identity of the person. The second task we look at how we can perform unsupervised learning and learn a new representation of the data. This is related to finding hidden structures or patterns in the data which might contain important information. Finally we will end with a look at how we can approach model selection. This is very important as it gives us the tool to design different models and then choose the one that best represents our data. The important message that these exercises tries to convey is how we can integrate our beliefs with observations using a set of simple rules. The assignments are aimed at showing the key aspects of data modeling in a simple scenario such that our insights about the models are not "clouded" by the complexity data. It is left to you as a student to extend this knowledge to a realistic scenario with real data.

Remark Please bear in mind that the notation here is a bit different from what you are used to in the lectures. In short:

- $\mathbf{X}$ and $\mathbf{T}$ describe matrices of data $\mathcal{D}$ (vectors of inputs $\mathbf{x}_i$ and outputs $\mathbf{t}_i$, respectively, stacked together)
- matrix $\mathbf{W}$ describes parameters (not $\mathbf{w}$) since we allow a general case of multidimensional outputs ($\mathbf{t}$ not t)

I The Prior $p(\mathbf{X})$, $p(\mathbf{W})$, $p(f)$

1.1 Theory

Regression is the task of estimating a continuous target variable $\mathbf{T}$ from an observed variate $\mathbf{X}$. The target and the observed variates are related to each other through a mapping,

$$f : \mathbf{X} \to \mathbf{T}, \tag{1}$$

where f indicates the mapping. Given input output pairs $\{\mathbf{x}_i, \mathbf{t}_i\}_1^N$ our task is to estimate the mapping f such that we can infer the associated $\mathbf{t}_i$ from previously unseen $\mathbf{x}_i$. In this task we will work with real vectorial data such that $\mathbf{x}_i \in \mathbf{X}$ where $\mathbf{x}_i \in \mathbb{R}^q$ and $\mathbf{t}_i \in \mathbf{T}$ where $\mathbf{t}_i \in \mathbb{R}^D$. Being probabilistic means that we need to consider the uncertainty in both the observations as well as the relationship between the variates. Starting with the relationship between two *single* points $\mathbf{x}_i$ and $\mathbf{t}_i$ we can assume the following form of the likelihood,

$$p(\mathbf{t}_i|f, \mathbf{x}_i) \sim \mathcal{N}(f(\mathbf{x}_i), \sigma^2\mathbf{I}). \tag{2}$$

Question 1: *Explain why Gaussian form of the likelihood is a sensible choice and what assumptions we make by this choice. What assumptions do we make about the data by choosing a spherical covariance matrix for the likelihood?*

Assuming that each output point is conditionally independent given the input and the mapping we can write the likelihood of the data as follows:

$$p(\mathbf{T}|f, \mathbf{X}) = \prod_{i=1}^{N} p(\mathbf{t}_i|f, \mathbf{x}_i). \tag{3}$$

Question 2: *If we do* **not** *assume that the data points are independent how would the likelihood look then? Remember that* $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]$

The task of regression means that we wish to infer $\mathbf{t}_i$ from its corresponding variate $\mathbf{x}_i$. These two variates are related to each other by the mapping f so from a probabilistic view point we wish to find the mapping from the observed data. More specifically, taking uncertainty into account, what we wish to reach is the posterior distribution over the mapping given the observations,

$$p(f|\mathbf{X}, \mathbf{T}). \tag{4}$$

1.1.1 Linear Regression

Please read first Ch.3 in (Bishop 2006). Let's make an assumption about the mapping and model the relationship between the variates as a linear mapping. Next, let's assume that the structure of the noise in the observations follows additive Gaussian distribution ($i = 1, .., N$):

$$\mathbf{t}_i = \mathbf{W}\mathbf{x}_i + \boldsymbol{\epsilon}, \tag{5}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. From this we can formulate the likelihood of the data,

$$p(\mathbf{T}|\mathbf{X}, \mathbf{W}) = \tag{6}$$

Question 3: *Please, complete the right-hand side of the expression in (6).*

The inference task we are interested in here is to learn the mapping, i.e. to infer $\mathbf{W}$ from the data,

$$p(\mathbf{W}|\mathbf{X}, \mathbf{T}) = \frac{1}{Z} p(\mathbf{T}|\mathbf{X}, \mathbf{W}) p(\mathbf{W}). \tag{7}$$

In the above equation we can see that we need to formulate our belief of the model parameters $\mathbf{W}$ in a prior $p(\mathbf{W})$. We can make many different choices of priors, but a sensible choice would be to pick the conjugate prior i.e. a Gaussian prior over the parameters:

$$p(\mathbf{W}) = \mathcal{MN}(\mathbf{W}_0, \mathbf{I}, \tau^2\mathbf{I})^1 \tag{8}$$

The prior distribution describes how likely or "how far" a parameter is from our belief.

Question 4: *The prior over each row of* $\mathbf{W}$ *in Eq.8 is a spherical Gaussian:* $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}_0, \tau^2\mathbf{I})$. *This means that the preferred model is encoded in terms of* L_2 *distance in the parameter space.*

- *What would be the effect of encoding the preferred model with* L_1 *norm (for model parameters)?*
- *Discuss how these two types of priors affect the posterior from the regularization perspective. Write down the penalization term, i.e. the negative log-prior, and illustrate for a two-dimensional problem (in the two-dimensional parameter space).*

The posterior is the object that integrates our prior beliefs with the data. In the next section we will see how this works in practice for the linear regression model derived above.

Question 5: *Assuming conditional independence of the target variables in* $\mathbf{t}$, *derive the posterior over the parameters. Please, do these calculations by hand as it is very good practice. To pass the assignment you only need to outline the calculation and highlight the important steps.*
In summary, please complete the following tasks

- *Derive the posterior over the parameters and explain the final form in terms of the mean and*

[1] This is a special case of matrix normal distribution, which implies that $\mathbf{w}$, rows of $\mathbf{W}$, follow normal distributions with the corresponding $\mathbf{w}_0$ means and diagonal covariances. In general, this can be usually handled using vectorization with the normal distribution.

covariance.

- *How does the posterior form relate to the least square estimator of* $\mathbf{W}$ *(equivalent to the maximum likelihood approach) for this linear regression problem?*
- *How does the constant* Z *(Eq.7) affect the solution? Are we interested in it?*

1.1.2 Non-parametric Regression

In the previous task we made the assumption that the relationship between the two variates was linear. This is quite a strong assumption that severely restricts the representative power of our model. The obvious way to proceed would be to add more parameters to f and use a higher-degree polynomial, but which one should we pick, degree 3 or 4, should we add a trigonometric function? These are tricky questions that require a lot of knowledge about the specific data that we are looking at. We want to stay general however, and the whole idea about Machine Learning is that we want the *data* to tell us all that without any need for us to specify it before we start building our model (beyond what prior can account for).

Let's take a step back and think how a Bayesian would think in this situation. The above argument just says that we have a large uncertainty in, not only in the the parameters of the mapping, but also in the actual *form* of the mapping. Bayesian reasoning allows us to deal with this, it is actually *exactly* this scenario that it was designed to deal with. We just need to formulate our uncertainty about the mapping in a prior over mappings and then use Bayes rule to reach the posterior. The problem is just that we need to somehow formulate a prior over a space of functions, which is quite a lot stranger mathematical object compared to the scalar valued parameters $\mathbf{W}$ in the previous task. Before proceeding with this task please read (Bishop 2006, p. 303-311).

We know from the lecture that Gaussian Processes ($\mathcal{GP}$s) can be used to represent prior distributions over the space of functions. Rather than specifying a concrete parametric form of the function, $\mathcal{GP}$s is a non-parametric model. We will now proceed to look at the regression problem where we replace the linear assumption in the mapping to use a non-parametric prior over the space of functions. Let's make the same assumption about the observations as we did in the linear case,

$$\mathbf{t}_i = f(\mathbf{x}_i) + \boldsymbol{\epsilon}. \tag{9}$$

This allows us to formulate the likelihood in the same manner as before. However, in the linear example we could easily formulate the relationship between $\mathbf{x}$, $\mathbf{t}$ and f as the latter had a simple parametric form. Now we cannot do this anymore. To proceed, let's define the output of the function f as its own random variable,

$$\mathbf{t}_i = f_i + \boldsymbol{\epsilon}, \tag{10}$$

where f_i is the *output* of the function at input location $\mathbf{x}_i$. The next step is to formulate the prior over the output of the function. This we can do using a $\mathcal{GP}$,

$$p(f|\mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, k(\mathbf{X}, \mathbf{X})) \tag{11}$$

where $k(\cdot, \cdot)$ is the covariance function and $\boldsymbol{\theta}$ is its parameters. We will refer to $\boldsymbol{\theta}$ as the *hyper-parameters* of the process.It is assumed that the data have been translated to have zero-mean such that we do not need to have a mean function in the prior.

Question 6: *Explain what this prior does. Motivate the choice of this prior and use images to show your reasoning. Clue: use the marginal distribution to explain the prior*

Given this formulation we can now formulate the full model.

Question 7: *Formulate the joint likelihood of the full model defined above,*

$$p(\mathbf{T}, \mathbf{X}, f, \boldsymbol{\theta})$$

and draw a simple graphical model reflecting the assumptions that you have made.

Unfortunately, we have added to our model a new variable that we are not really interested in. Specifically we have modeled the relationship between $\mathbf{T}$ and f and also f and $\mathbf{X}$ but we really are interested in the relationship between $\mathbf{T}$ and $\mathbf{X}$. The motivation behind this is that we now have the possibility to have uncertainty in each of these stages, in our beliefs of the functions, and in how we believe the output of the function have generated the observed data. But again, we are not interested in f and therefore the variable should be marginalised out. Performing the marginalisation implies calculating a specific integral.

$$p(\mathbf{T}|\mathbf{X}, \boldsymbol{\theta}) = \tag{12}$$

Question 8: *Complete the marginalisation formula in Eq.12 (general form) and discuss the following:*

- *Explain how it connects the prior and the data.*
- *How does the uncertainty "filter" through the marginalisation?*
- *Why do we still condition on $\boldsymbol{\theta}$ after the marginalisation?*

1.2 Practical

Now we will implement the approach we studied in the previous part. Remember to save images and figures to support your claims in part 1 as this will make the presentation much easier to examine. There are a couple of packages in Python that are really useful:

```
import pylab as pb
import numpy as np
from math import pi
from scipy.spatial.distance import cdist

# To sample from a multivariate Gaussian
f = np.random.multivariate_normal(mu,K);
# To compute a distance matrix between two sets of vectors
D = cdist(x1,x2)
# To compute the exponetial of all elements in a matrix
E = np.exp(D)
```

1.2.1 Linear Regression

In this task we will implement the linear regression that we looked at in the previous task. We will examine both the prior and the posterior over the parameters $\mathbf{W} = [w_0, w_1]$ and evaluate the effect this will have on the model. To do so we will need to have some data to experiment with. What we want to show is that the methodology that we have learned is capable of recovering the true underlying mapping from the observed data. Therefore let's generate some data $(\mathbf{x}, \mathbf{t})$ and then simply throw the generating parameters away.

$$t_i = w_0 x_i + w_1 + \epsilon = 0.5 x_i - 1.5 + \epsilon \tag{13}$$
$$\mathbf{x} = [-1, -0.99, \ldots, 0.99, 1] \tag{14}$$
$$\epsilon \sim \mathcal{N}(0, 0.2) \tag{15}$$
$$\tag{16}$$

Question 9:

1. *Set the prior distribution over* $\mathbf{W}$ *and visualise it.*
2. *Pick a single data point (x,t) and visualise the posterior distribution over* $\mathbf{W}$*.*
3. *Draw 5 samples from the posterior and plot the resulting functions.*
4. *Repeat* $2-3$ *by adding additional data points up to 7.*
5. *Given the plots explain the effect of adding more data on the posterior as well as the functions. How would you interpret this effect?*
6. *Finally, test the exercise for different values of* σ*, e.g. 0.1, 0.4 and 0.8. How does your model account for data with varying noise levels? What is the effect on the posterior?*

1.2.2 Non-parametric Regression

In this task we will implement and evaluate the effect of a $\mathcal{GP}$-prior. Specifically we will look at the squared exponential covariance function,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 e^{-\frac{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}{l^2}} \tag{17}$$

You will need to first formulate the prior distribution and then the posterior. How to do this can be found in (Bishop 2006, p. 306-308). First we will look at the prior.

Question 10:

1. *Create a $\mathcal{GP}$-prior with a squared exponential covariance function.*
2. *For each of 4 different length scales, please draw 10 samples from this prior and visualise them. Explain the observed consequences of altering the length-scale of the covariance function.*

This has been said many times before but it is something that cannot be stressed enough, priors are very important as they allow us to formulate our uncertainty in our beliefs in a principled manner. However, more important is that we can combine our beliefs with observations, this is what facilitates learning. The object that contains this is the posterior distribution. We will now perform a simple experiment on the posterior.
Let's generate some data $(\mathbf{x}, \mathbf{t})$ that we know would not work particularly well using a linear model as in the previous task,

$$t_i = (2 + (0.5x_i - 1)^2) * \sin(3x_i) + \epsilon_i \tag{18}$$

$$\mathbf{x} = [-4, -3, -2, -1, 0, 2, 3, 5]^{\mathrm{T}} \tag{19}$$

$$\epsilon_i \sim \mathcal{N}(0, 3). \tag{20}$$

These observations of a noisy function can be used together with the prior to obtain the posterior distribution over the functions.

Question 11:

1. *What is the posterior before we observe any data?*
2. *Compute the predictive posterior distribution of the model.*
3. *Sample from this posterior with points both close to and far away from the observed data. Explain the observed effects.*
4. *Plot the data, the predictive mean and the predictive variance of the posterior from the data.*
5. *Compare the samples of the posterior with the ones from the prior. Is the observed behavior desirable?*
6. *What would happen if you added a diagonal covariance matrix to the squared exponential?*

II The Posterior $p(\mathbf{X}|\mathbf{Y})$

In the previous task we looked at learning a relationship between two variates $\mathbf{X}$ and $\mathbf{T}$ such that we could infer one from the other. One way of thinking about this is that given the mapping f and the input $\mathbf{X}$ we specify the outputs $\mathbf{T}$, you can think of $\mathbf{X}$ as a "representation" of $\mathbf{T}$, i.e. that the former have generated the later. Actually this is exactly what we did in the linear regression task, we generated some data using a set of parameters $\mathbf{w}$ then we threw them away and later recovered them back, but we retained $\mathbf{x}$ in this process.
In this task we will make things a bit more complicated by looking at representation learning. In this context, to be consistent with the notation used in the lecture and in most literature, let's use $\mathbf{Y}$ to denote output data instead of $\mathbf{T}$ for targets. This means we will only observe the outputs $\mathbf{Y}$ and want to learn input $\mathbf{X}$ that can represent $\mathbf{Y}$. Why would we ever want to do this? Let's take the example of an image. Images are very high-dimensional objects, a typical HD image $\mathbf{y}_i$ concatenated into a vector will live in a space of $\mathbb{R}^{1920\times1080\times3}$. However, does the image actually have that many degrees-of-freedom? To simplify, given that you know the place the image was taken, the weather, the exact camera angle, the objects in the image wouldn't you be able to generate the pixel data? This is of course a massive simplification but ponder how many parameters you can come up with and this will be less than the number of pixels in the image. Let's call all these factors that we came up with and refer to them as *generating parameters* just as we said that $\mathbf{X}$ through f generated $\mathbf{Y}$ ($\mathbf{T}$) in Section I. Representation learning allows you to recover these generating parameters directly from the data. More specifically this relates to building a model of the data $\mathbf{Y}$ and then looking at the posterior distribution over the input to the model $\mathbf{X}$.

The other new thing that we will introduce in this task is *learning*. This means that we specify a model like in Section I and then *fit* this model to the data. It implies that the model has a set of parameters, which we now will infer from the data.

1.3 Theory

The focus here will be on the same linear models as in the first part (though non-parametric Gaussian Processes in Part I can also be handled in the similar framework). The main difference is that the input locations $\mathbf{X}$ are not known a priori but rather we want to infer them from data. We will refer to the input locations as the *latent representation* of the observed data $\mathbf{Y}$. Think about how this relates to the latent space models that you worked on in the first part of the course, where you used discrete latent *states* to represent continuous data. This is very much the same thing, except for that we want to find the latent space from data and that rather than being a discrete variable it is continuous.

Let's start with the linear model:

$$p(\mathbf{Y}, \mathbf{X}, \mathbf{W}) = p(\mathbf{Y}|\mathbf{X}, \mathbf{W})p(\mathbf{X})p(\mathbf{W}). \tag{21}$$

The next step needs a deeper reflection, being fully Bayesian we would like to invert the model above and look at the conditional distribution over the variables that we want to infer. Think about this, does it make sense? Actually, there is a simple relationship between $\mathbf{X}$ and $\mathbf{W}$: having one implies having the other one. As an example, if each $\mathbf{x}_i$ is multipled by a constant, it is the same as dividing the $\mathbf{W}$ by the same constant. To get away from this is we should only look at a single variable. But as our model contains both $\mathbf{X}$ and $\mathbf{W}$, how can we do this? In the Bayesian spirit, we can specify a prior over the variable that we are not interested in and marginalise it out from the model.

What does this actually mean? Previously we have been using prior distributions as a mean of encoding our beliefs about a variable before seeing data. Another equally valid explanation is as encoding our *preference* of a variable.

Let's specify the prior over the latent variables as a spherical Gaussian:

$$p(\mathbf{X}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{22}$$

Question 12: *What type of "preference" for the latent variable* $\mathbf{X}$ *does this prior encode?*

Now we can combine this prior with the likelihood, integrate out $\mathbf{X}$ and reach the marginal distribution,

$$p(\mathbf{Y}|\mathbf{W}) = \int p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{X})\mathrm{d}\mathbf{X}. \tag{23}$$

Question 13: *Perform the marginalisation in Eq. 23 and write down the expression. As previously, it is recommended that you do this by hand. In the answer outline the calculations and highlight the important steps.*
Hint: *The marginal can be computed by integrating out* $\mathbf{X}$ *with the use of Gaussian algebra we exploited in the exercise derivations and, in particular, by completing the square. However, it is much easier to derive the mean and covariance, knowing that the marginal is Gaussian, from the linear equation of* $\mathbf{Y}(\mathbf{X})$.

1.3.1 Learning

So far we have only created models and looked at the posterior. Now we will take one step further and learn the parameters of the model. A good background to what we will go through here can be found in (Bishop 2006, p. 9,23,26,30,165, sec. 1.2.4-1.2.6). Let's start to do learning in a probabilistic model with the maximum-likelihood (ML) approach. So, we formulate the likelihood of the data and find the parameters that maximise it,

$$\hat{\mathbf{W}} = \mathrm{argmax}_{\mathbf{W}} p(\mathbf{Y}|\mathbf{X},\mathbf{W}). \tag{24}$$

The next level is to perform maximum-a-posteriori (MAP) estimation. This means that we find the parameters that maximise the posterior distribution,

$$\hat{\mathbf{W}} = \mathrm{argmax}_{\mathbf{W}} \frac{p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W})}{\int p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W})\mathrm{d}\mathbf{W}} = \mathrm{argmax}_{\mathbf{W}} p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W}). \tag{25}$$

There is also an in-between stage which is often referred to as Type-II Maximum-Likelihood which implies maximisation of the marginal likelihood where you integrate out one parameter and then maximise over another, as follows:

$$\hat{\mathbf{W}} = \mathrm{argmax}_{\mathbf{W}} \int p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{X})\mathrm{d}\mathbf{X}. \tag{26}$$

Question 14: *Compare the three different estimation procedures above in log-space.*

1. *What are their distinctive features and how are they different when we observe more data?*
2. *Why are the two last expressions of Eq. 25 equal?*
3. *Explain why Type-II Maximum-Likelihood is a sensible approach to learn the model.*

Practical Optimisation In practice when performing optimisation on probabilistic models we often have to deal with exponentials. Exponentials are nice in many ways, they are for example infinitely differentiable, but they are a bit tricky to play with. Often we have the case that our parameters are actually in the exponents and then we can do a neat trick that makes life much easier. Rather than working directly on the exponent we perform all our learning in the log-space instead. The reason that we can do that is because $\log(\cdot)$ is a monotonic function and therefore it will not alter the location of the extremes of the function. Further, most optimisation packages are designed to minimise a function rather than maximising it. This means that in practice we often formulate our optimisation problem as a minimisation of the *negative log* of a probablitiy,

$$\hat{\theta} = \text{argmax}_\theta p(\mathbf{Y}|\theta) = \text{argmin}_\theta - \log\left(p(\mathbf{Y}|\theta)\right). \tag{27}$$

Now we will compute the objective function and its gradients which means we need to do some matrix algebra. There are plenty of literature resources helpful when working with matrices. The following two references, (Petersen and Pedersen 2006) and (Magnus and Neudecker 1988), are particularly worth recommending.

Question 15:

1. *Compute the objective function* $-log(p(\mathbf{Y}|\mathbf{W})) = \mathcal{L}(\mathbf{W})$ *for the marginal distribution in Eq. 23.*
2. *Compute the gradients of the objective with respect to the parameters* $\frac{\delta \mathcal{L}}{\delta \mathbf{W}}$

In the practical section of this task you will perform the optimisation above for a real dataset.

1.4 Practical

This practical part includes what is considered the bread and butter for a machine learning scientist, i.e. working with data. Let's generate some data so that we know what we are looking to recover,

$$\mathbf{Y} = f_{\text{lin}}(f_{\text{non-lin}}(\mathbf{x})) \tag{28}$$
$$\mathbf{x} = [0, \ldots, 4\pi]^{\text{T}} \tag{29}$$
$$|\mathbf{x}| = N \tag{30}$$
$$f_{\text{non-lin}}(x_i) = [\sin(x_i) - x_i\cos(x_i), \cos(x_i) + x_i\sin(x_i)] \tag{31}$$
$$f_{\text{lin}}(x') = \mathbf{x}'\mathbf{A}^{\text{T}} \tag{32}$$
$$\mathbf{A} \in \mathbb{R}^{10\times 2} \tag{33}$$
$$\mathbf{a}_{ij} \sim \mathcal{N}(0, 1). \tag{34}$$

The values in the linear mapping are drawn from an independent Gaussian as we do not really care about the specific form of the mapping we only care about its `rank`.

Now we have generated a dataset $\mathbf{Y} \in \mathbb{R}^{N\times 10}$ from a one-dimensional *generating parameter* $\mathbf{x} \in \mathbb{R}^{N\times 1}$ (N=200). The aim is now to recover $\mathbf{x}$, i.e a single line, given only $\mathbf{Y}$. This is a very general and incredibly important task in machine learning, i.e. how to discover the parameters that have generated some observations. It is important as many types of data are represented in high-dimensional spaces, which are hard to interpret, and providing the true generating parameters allows us to analyse the data and hopefully find the casual behavior in the data.

1.4.1 Linear Representation Learning

We have an objective function and we have the gradients with respect to the parameters that we want to learn. The actual optimisation can be done with the use of gradient descent. This is well implemented in `scipy.optimize`. Have a look at reference manual for the different methods that are available. Below is the simple structure that you need to implement in order to get the `fmin` function working,

```
import numpy as np
import scipy as sp
import scipy.optimize as opt

def f(x, *args):
    # return the value of the objective at x
    return val

def dfx(x,*args):
    # return the gradient of the objective at x
    return val

x_star = opt.fmin_cg(f,x0,fprime=dfx, args=args)
```

Question 16:

1. *Plot the representation that you have learned (hint: plot* $\mathbf{X}$ *as a two-dimensional representation).*
2. *Explain the outcome and discuss key features, elaborate on any invariance you observe. Did you expect this result?*
3. *How is the effect of representation learning dependent on the number of available samples? Please test lower values of* N *and discuss the observed implications.*

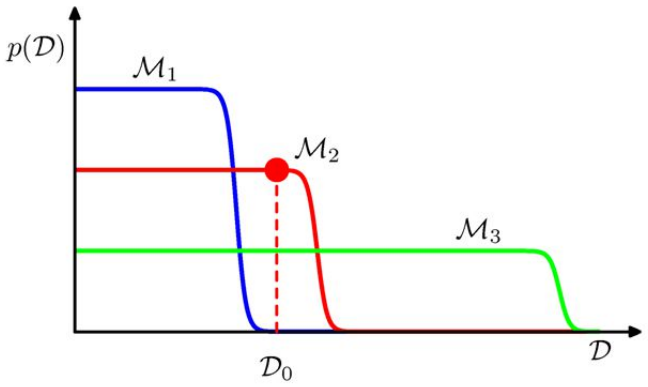

Figure 1: *The above figure (Figure* 3.13 *in (Bishop 2006) shows the idea of model complexity and Occam's razor that was introduced in (Mackay 1991). Occam's Razor tells us that we should always choose the "simplest" model that explains all of our data. In the figure the data domain is ordered on the* `x-axis` *by increasing complexity and the evidence* $p(\mathcal{D})$ *is shown on the* `y-axis`*. Given that we want to model the data* $\mathcal{D}_0$ *which model should we choose? Model* $\mathcal{M}_1$ *places no probability mass over* $\mathcal{D}_0$ *so it is a bad choice. Both models,* $\mathcal{M}_2$ *and* $\mathcal{M}_3$*, place probability mass over* $\mathcal{D}_0$*. However,* $\mathcal{M}_2$ *places more as it is less complex and models only a part of the data domain, and should therefore be preferred according to Occam's Razor.*

III The Evidence $p(\mathcal{D})$

One of the main arguments behind Bayesian reasoning is that it automatically implements Occam's razor, i.e. that automatically chooses the "correct" model complexity to perform a specific task. In this part of the assignment we will perform a study which shows that sometimes things are not as obvious as one might think. We will use the evidence of the data under the model as a means of measuring the complexity of the model.

1.5 Theory

In the practical part of the task we will perform the experiments outlined in (Murray and Ghahramani, 2005). It is recommended that you read this paper and familiarise yourself with their discussion surrounding complexity. Do not expect any clear answers in this but you should grasp the discussion and be able to argue about what the results shows.

1.5.1 Data

Consider a very simple data domain $\mathcal{D} = \{t^i\}_{i=1}^9$ where $t^i \in \{-1, 1\}$ (we return to the notation t for the outputs to remain consistent with the lecture). This data is structured according to a grid whose locations can be parametrised by $\mathcal{X} = \{\mathbf{x}^i\}_{i=1}^9$ where $\mathbf{x}^i = (\{-1, 0, +1\}, \{-1, 0, +1\})$. This means that our data domain $\mathcal{D}$ contains $2^9 = 512$ different elements which is small enough for us to reason about but still complicated enough that it requires a sensible model.

1.5.2 Models

Given the data defined above we wish to create a model, i.e. something that will explain the statistical variations that are possible in $\mathcal{D}$. The simplest model is something that simply takes all its probability mass and places it uniformly over the whole data space,

$$p(\mathcal{D}|M_0, \boldsymbol{\theta}_0) = \frac{1}{512}. \tag{35}$$

Question 17: *Why is this the simplest model, and what does it actually imply? What makes it a bad model on the one hand, and a good model on the other hand?*

The first model in Eq. 35 does not take any parameters at all which means it has no flexibility and uses no information about $\mathcal{D}$ except for its cardinality. We can use what we know about the data in order to specify something slightly more representative. If we assume that all t^n are independent we can factorise the model into 9 separate models,

$$p(\mathcal{D}|M_1, \boldsymbol{\theta}_1) = \prod_{n=1}^{9} p(t^n|M_1, \boldsymbol{\theta}_1), \tag{36}$$

where θ_i^j means the j:th element of the parameter vector for the i:th model. Each model can be expressed using an exponential function which relates the value t^i to its location $\mathbf{x}^i$,

$$p(\mathcal{D}|M_1, \boldsymbol{\theta}_1) = \prod_{n=1}^{9} \frac{1}{1 + e^{-t^n \theta_1^1 x_1^n}}, \tag{37}$$

Question 18: *Explain how each separate model works. In what way is this model more or less flexible compared to M_0? How does this model spread its probability mass over $\mathcal{D}$?*

We can continue to add more parameters and create further models,

$$p(\mathcal{D}|M_2, \boldsymbol{\theta}_2) = \prod_{n=1}^{9} \frac{1}{1 + e^{-t^n (\theta_2^1 x_1^n + \theta_2^2 x_2^n)}} \tag{38}$$

$$p(\mathcal{D}|M_3, \boldsymbol{\theta}_3) = \prod_{n=1}^{9} \frac{1}{1 + e^{-t^n (\theta_3^1 x_1^n + \theta_3^2 x_2^n + \theta_3^3)}}, \tag{39}$$

Question 19: *Discuss and compare the models. In particular, please address the following questions in your discussion*

- *How have the choices we made above restricted the distribution of the model?*
- *What datasets is each model suited to model? What does this actually imply in terms of uncertainty?*
- *In what way are the different models more flexible and in what way are they more restrictive?*

1.5.3 Evidence

The evidence of a model M_i is the distribution $p(\mathcal{D}|M_i)$. This distribution tells us how and where the model spreads its probability mass. Occam's razor can be interpreted in terms of the evidence such as we should choose a model which places most of its mass where we will see data and as little as possible elsewhere. In the previous section we have defined a small simple data domain $\mathcal{D}$ and we will now evaluate where the different models defined above places their probability mass.

In order to "reach" the evidence of a model we need to first remove the dependency of the variable $\boldsymbol{\theta}$. This can be done by marginalising out the parameters from the model,

$$p(\mathcal{D}|M_i) = \int_{\forall \boldsymbol{\theta}} p(\mathcal{D}|M_i, \boldsymbol{\theta}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}. \tag{40}$$

Question 20: *Explain the process of marginalisation and briefly discuss its implications in the given context of the model evidence.*

The marginalisation above requires one more object that we have not seen before $p(\boldsymbol{\theta}|M_i)$. This is the *prior* over the parameters of the model. Being Bayesian implies that you need to take uncertainty into account in all steps of your calculations. This is true for the data but also true for the parameters. As we do not really know much at all about the parameters, we would like to be uncertain and allow for a large range of possible values of $\boldsymbol{\theta}$. One prior would be to choose a simple Gaussian with zero mean and a very large variance, e.g.:

$$\begin{aligned} p(\boldsymbol{\theta}|M_i) &= \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ \boldsymbol{\mu} &= \mathbf{0} \\ \boldsymbol{\Sigma} &= \sigma^2 \boldsymbol{I} \\ \sigma^2 &= 10^3 \end{aligned} \tag{41}$$

Question 21: *What does this choice of prior imply? How does the choice of the parameters of the prior $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ affect the model?*

Now when we have defined the prior $p(\boldsymbol{\theta})$ we just need to perform the marginalisation in Eq. 40 to be able to evaluate the evidence. However, this integration is rather tricky to do analytically, which means that we will here use an approximate integral using a naïve Monte Carlo approach,

$$p(\mathcal{D}|M_i) \approx \frac{1}{S} \sum_{s=1}^{S} p(\mathcal{D}|M_i, \boldsymbol{\theta}^s), \tag{42}$$

$$\boldsymbol{\theta}^s \sim p(\boldsymbol{\theta}|M_i) \tag{43}$$

where s indexes the samples from the prior of the parameters.

We will now proceed to implement the procedure explained above and see what implications the different models and the choices of prior distributions have in terms of the evidence.

1.6 Practical

Even though the functionality that we need to do the calculations described above are simple there is, as always with high-level languages, lots of useful packages available in `Python` that will make our life much easier. You may find the following libraries very useful,

```
import itertools as it
from math import exp, sqrt, pi
import scipy.stats
```

Below is just a suggestion of how and in what order to implement the code to be able to answer the questions for this task. For grading your results reflected in plots matter, not the implementation itself.

1. First create the code that generates the dataset $\mathcal{D}$ and the locations of the data `x`, `itertools` will be very useful here. Write some simple functionality to visualise a single element of the data on the 3×3 grid defined by `x`.
2. Create the code to represent each model M_0 to M_3.
3. Create the code to sample from the prior $p(\boldsymbol{\theta}|M_i)$.
4. Write the code to perform the Monte Carlo integration given a model, returning the evidence.
5. Write the code to index the datasets such that you can easily compare the models.

Question 22: *Plot the evidence over the whole dataset for each model (and sum the evidence for the whole of $\mathcal{D}$, explain the numbers you get). The* `x-axis` *index the different instances in $\mathcal{D}$ and each models evidence is on the* `y-axis`*. How do you interpret this? Relate this to the parametrisation of each model.*

Question 23: *Find* `using np.argmax and np.argmin` *which part of the $\mathcal{D}$ that is given most and least probability mass by each model. Plot the datasets which are given the highest and lowest evidence for each model. Discuss these results, do the findings make sense?*

Question 24: *What is the effect of the prior $p(\boldsymbol{\theta})$.*

- *What happens if we change its parameters?*
- *What happens if we use a non-diagonal covariance matrix for the prior?*
- *Alter the prior to have a non-zero mean, such that $\mu = [5, 5]^T$?*
- *Redo evidence plot for these and explain the changes compared to using zero-mean.*

Good Luck!

References

[1] C.M. Bishop. *Pattern recognition and machine learning.* 2006

[2] K.P. Murphy. *Machine Learning: A Probabilistic Perspective* The MIT Press, 2012.

[3] K.B. Petersen and M.S. Pedersen. (2006) The matrix cookbook. Technical report. Technical University of Denmark. https://www.ics.uci.edu/~welling/teaching/KernelsICS273B/MatrixCookBook.pdf

[4] J.R. Magnus *et al. Matrix differential calculus with applications in statistics and econometrics.* Wiley, 1998.

[5] J. Hensman *et al.* (2014) GPy: A Gaussian process framework in Python. https://github.com/SheffieldML/GPy

[6] D.J.C. Mackay. (1991) Bayesian methods for adaptive models. PhD thesis. California Institute of Technology. http://thess.library.caltech.edu/25/

[7] I. Murray and Z. Ghahramani. (2005) A note on evidence and Bayesian Occam's razor. Technical report. http://homepages.inf.ed.ac.uk/imurray2/pub/05occam/occam.pdf