Modelling

Assignment in COMS30007 Machine Learning

Carl Henrik Ek

October 29, 2018

Welcome to the first assignment in the machine learning course. You will present the assignment with by a written report that you should submit on SAFE, try to leave some space for submission and don't wait till to close to the deadline. 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, should be stated clearly. For the practical part of the task you should not show any of your code in the report and only show the results of your experiments using images and graphs together with your analysis. You should still submit your code though, remember that you are free to use whatever language that you want.

Being able to communicate your results and conclusions is key for any scientific practitioner. It is up to you as a author to make sure that the report clearly shows what you have done and what your understanding is. 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! Each report can be up to 10 pages long including references.

If you have not already done so, learn LaTeX. It is an amazing tool that you will find very useful in your further endeavours as a scientist. In the `git` repo there is a style file for the report. If anyone is an `Emacs` connoisseur I recommend using the excellent `org-mode` and the `ox-latex` engine to prepare your report.

The grading of the assignments will be as follows,

50% Question 1-14, 30

70% Question 1-22, 30

80% Question 1-30

90% Question 1-30 and Non-Parametric Representation Learning

This means that if you have achieved all the assignments this is your potential top mark if all the questions are answered correctly and the discussion is as it should be. For the 90% mark, you have to do the non-parametric representation learning bit which should include the derivations and the explanations for this not just the results. The last 10% of the mark is for me to sprinkle onto reports when I see especially nice arguments or solutions.

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 try and learn a model of a specific relationship between two different variates. 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 modelling 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.

1 The Prior

1.1 Theory

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

$$f : \mathbf{X} \to \mathbf{Y},$$

where f indicates the mapping. Given input output pairs $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$ our task is to estimate the mapping f such that we can infer the associated $\mathbf{y}_i$ 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 $\mathbf{x}_i \in \mathcal{R}^q$ and $\mathbf{y}_i \in \mathbf{Y}$ where $\mathbf{y}_i \in \mathcal{R}^D$. 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 $\mathbf{y}_i$ we can assume the following form of the likelihood,

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

Question 1

1. What assumption does a Gaussian likelihood encode, i.e. what motivates the choice of this likelihood function?
2. What does it mean that we have chosen a spherical covariance matrix for the likelihood, contrast with the non-spherical case?[a]

[a] A spherical co-variance implies a diagonal matrix where each diagonal element is the same, i.e. a matrix of the form $\mathbf{M} = \beta \cdot \mathbf{I}$

Assuming that each output point is independent given the input and the mapping we can write the likelihood of the data in the following factorised form,

$$p(\mathbf{Y}|f, \mathbf{X}) = \prod_i^N p(\mathbf{y}_i|f, \mathbf{x}_i).$$

Question 2

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

The task of regression means that we wish to infer a continous $\mathbf{y}_i$ from its corresponding variate $\mathbf{x}_i$. These two variates are related to each other by the mapping f. Taking uncertainty into account, what we wish to reach is the posterior distribution over the mapping given the observations,

$$p(f|\mathbf{X}, \mathbf{Y}).$$

1.1.1 Linear Regression

I recommend that you read Chapter 3 in [1] before you start this part of the task. In order to proceed lets make an assumption about the mapping and model the relationship between the variates as a linear mapping. Further, lets make an assumption about the structure of the noise in the observations, assuming that they have been corrupted by additive Gaussian noise,

$$\mathbf{y}_i = \mathbf{W}\mathbf{x}_i + \boldsymbol{\epsilon},$$

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

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

Question 3

What is the specific form of the likelihood above, complete the right-hand side of the expression.

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

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

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 a conjugate prior i.e. a Gaussian prior over the parameters,

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

Question 4

Explain the concept of conjugate distributions, why do they help us compute the posterior distribution?

The prior in Eq.1 is a spherical Gaussian. The prior distribution is a tool that tells us how likely a specific parameter choise is under our belief. One way to think about concepts such as likely is to use a geometric interpretations where a distance function acts as a proxy for our semantic encoding a dissimilarity measure.

Question 5

Reason about the Gaussian distribution in this context, which distance function does it encode with a spherical co-variance matrix.

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 that we have derived above.

Question 6

Write out the posterior over the parameters $\mathbf{W}$. I recommend that you do these calculations by hand as it is very good practice and provides important intuitons. However, in order to pass the assignment you only need to outline the calculation and highlight the important steps.

- Justify the the posterior by providing an intuition of its form.

1.1.2 Non-parametric Regression

In the previous section we made the assumption that the relationship between the two variates where linear. This is quite a strong assumption that severely restricts the representative power of our model. The obvious way to proceed is would be to add more parameters to f and use a higher-degree polynomial, but which one should we pick, degree 3 or 4, maybe we should we add a trigonometric function? These are very tricky questions that requires a lot of knowledge about the specific data that we are looking at. Another way to think about this problem is to say that when learning a linear model we place zero probability on all other models. Importantly we should only place zero probability on things which we can deductively reason cannot happen. With this argument the linear restriction is not just several limiting it is in most cases an assumption that is very hard to justify.

Lets 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 of the actual *form* of the mapping. We know how to deal with this, we just need to formulate our uncertainty about the mapping in a prior over mappings and then use Bayes rule to reach the posterior, job done! The problem is that we need to somehow formulate a prior over a the 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 I recommend that you read [1] page 303-311.

We know from the lecture that Gaussian Processes can be used to represent prior distributions over the space of functions. Rather than specifying a specific parametric form of the function GPs are non-parametric models.

Question 7

What is a non-parametric model and what is the difference between non-parametrics and parametrics? In specific discuss these two aspects of non-parametrics,

- Representation/parametrisation of data?
- Interpretability[a] of models?

[a] I'm not even sure if this is a word but hopefully you get what I mean. Write down what it means to you.

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. Lets make the same assumption about the observations as we did in the linear case,

$$\mathbf{y}_i = f(\mathbf{x}_i) + \boldsymbol{\epsilon}.$$

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 $\mathbf{x}$, $\mathbf{y}$ and f as the latter had a simple parametric form. Now we cannot do this anymore. To proceed, lets define the output of the function f as its own random variable,

$$\mathbf{y}_i = f_i + \boldsymbol{\epsilon},$$

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 Gaussian process (GP),

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

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

Question 8

Explain what this prior represents and how it places structure on the space of functions?

Question 9

Does this prior encode all possible functions or only a subset?

Given this formulation we can now formulate the full model.

Question 10

Formulate the joint distribution of the full model that you have defined above,

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

Draw the graphical model and clearly state the assumptions that has been made in bullet list.

Annoyingly we have added a new variable to our model which we are not really interested in. Specifically we have modeled the relationship between $\mathbf{Y}$ and f and also f and $\mathbf{X}$ but what we really are interested in is the relationship between $\mathbf{Y}$ 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 the following integral,

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \int p(\mathbf{Y}|f)p(f|\mathbf{X}, \boldsymbol{\theta})\mathrm{d}f. \tag{2}$$

Question 11

Explain the marginalisation in Eq.2

- Explain how this connects the prior and the data?
- How does the uncertainty "filter" through this?
- What does it imply that $\boldsymbol{\theta}$ is left on the left-hand side of the expression after 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 for me, and therefore also for you. There are a couple of packages that I found really useful when I implemented this code,

Code

```
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 look at both the prior and the posterior over the parameters $\mathbf{W}$ 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 observed data, therefore lets generate some data and then simply throw the generating parameters away.

$$y_i = w_0 x_i + w_1 + \epsilon \tag{3}$$
$$\mathbf{x} = [-1, -0.99, \ldots, 0.99, 1] \tag{4}$$
$$\epsilon \sim \mathcal{N}(0, 0.3) \tag{5}$$
$$\mathbf{W} = [-1.3, 0.5] \tag{6}$$

Question 12

1. Visualise the prior distribution over $\mathbf{W}$.
2. Pick a single data-point from the data and visualise the posterior distribution over $\mathbf{W}$
3. Sample from the posterior and show a couple of functions
4. Repeat $2-3$ by adding additional data points
5. Describe the plots, and the behavior when adding more data? Is this a desirable behavior?
6. Relate to the expression of the posterior why you see the behaviour that you do when you add more data

1.2.2 Non-parametric Regression

In this task we will implement and evaluate the effect of a GP-prior. In specific we will look at the squared exponential covariance function,

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

You will need to first formulate the prior distribution and then the posterior. How to do this can be found in [1] p. 306-308. First we will look at the prior,

Question 13

1. Create a GP-prior with a squared exponential co-variance function.
2. Sample from this prior and visualise the samles
3. Show samples using different length-scale for the squared exponential
4. Explain the behavior of altering the length-scale of the covariance function.
5. What assumption does the lengthscale encode?

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 assumptions, belief and our uncertainty in a principled manner. However, in order for this to have any value we need to be able to combine our prior with data, this process is what facilitates learning. The object that contains this is the posterior distribution. We will now perform

a simple experiment on the posterior. Lets generate some data that we know would not work particularly well using a linear model as in the previous task,

$$y_i = \sin(x_i) + \epsilon_i \tag{7}$$
$$\mathbf{x} = [-\pi, \dots, \pi]^{\mathrm{T}} \tag{8}$$
$$\epsilon_i \sim \mathcal{N}(0, 0.5), \tag{9}$$

where the cardinality of $\mathbf{x}$ is 7, i.e. we have 7 data-points. Now we have some observations of a noisy sin-wave which we can use together with our prior to see the posterior distribution over the functions.

Question 14

1. Compute the predictive posterior distribution of the model
2. Sample from this posterior with points both close to the data and far away from the observed data.
3. Plot the data, the predictive mean and the predictive variance of the posterior from the data

Explain the behavior of the samples and compare the samples of the posterior with the ones from the prior. Is this behavior desirable? What would happen if you would add a diagonal co-variance matrix to the squared exponential?

2 The Posterior

In the previous task we looked at learning a relationship between two variates $\mathbf{X}$ and $\mathbf{Y}$ 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 $\mathbf{Y}$, you can think of $\mathbf{X}$ as a "representation" of $\mathbf{Y}$, 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 representation learning. 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? Lets 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 I recon it will be less than the number of pixels in the image. Lets 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 $\mathbf{Y}$ in Task~**??**. What representation learning is ideally about is 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 Task 1 and then *fit* this model to the data. What this means is that the model have a set of parameters which we now will infer from the data.

2.1 Theory

The models that we will look like are much the same as in the first task, a linear model and a non-parametric Gaussian Process. The difference now is that the input locations $\mathbf{X}$ are not known a-prior 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.

Lets start with the linear model,

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

The next step will take a bit of thinking, 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 $\mathbf{W}$, if I have one I have the other right? As an example, if I multiply each $\mathbf{x}_i$ with a constant that is the same as dividing the $\mathbf{W}$ with the same constant. The way to get away from this is to only look at one variable. But as our model contains both $\mathbf{X}$ and $\mathbf{W}$ how can we do this? We can specify a prior over the variable that we are not interested in an marginalise it out from the model, right?

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.

Question 15

Elaborate on the relationship between assumptions, belief and preference.[a]

[a] There is no right or wrong answer here, I want to hear your thoughts on the what these words means to you in the context of what we have been doing.

Lets specify the prior over the latent variables as a spherical gaussian,

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

Question 16

What is the assumption/preference we have encoded with this prior?

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

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

Question 17

Perform the marginalisation in Eq.2.1 and write down the expression. As previously, I do recommend that you do this by hand but to pass the assignment you only need to outline the calculations and show the approach that you would take.

2.1.1 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 [1, page 9,23,26,30,165 & Section 1.2.4-1.2.6]. The most straight forward manner to do learning in a probabilistic model is called Maximum-Likelihood or ML. This does exactly what it says on the box, we formulate the likelihood of the data and then we find the parameters that maximises the likelihood,

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

The next level up is to perform maximum-a-posteriori or *MAP* estimation. This means that we find the parameters that maximises the posterior distribution,

$$\hat{\mathbf{W}} = \text{argmax}_{\mathbf{W}} \frac{p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W})}{\int p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W})\mathrm{d}\mathbf{W}} = \text{argmax}_{\mathbf{W}} p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W}). \tag{10}$$

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,

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

Question 18

- How are the different?
- How are MAP and ML different when we observe more data?
- Why are the two expressions in Eq. 10 equal?

In the representation task we have a model with two variables $\mathbf{W}$ and $\mathbf{X}$ that interact. This means that a Type-II ML estimation is a sensible approach to learn the model.

2.1.2 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 (however that can also be an issue), but they are a bit tricky to play with. Often we have the case that our parameters are actually in the exponents therefore 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,

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

Now we will write down the objective function and its gradients which means we need to do some matrix algebra. Two really good references that I find incredibly useful when working with matrices is [2, 3] which you can find online. You are allowed to make assumptions that simplifie your calculations, for example setting the mean of the data to zero is one of them. Have a look at [1] Eq. 12.43-44 and verify your objective function before you start finding the derivatives.

Question 19

- Write down the objective function $-\log(p(\mathbf{Y}|\mathbf{W})) = \mathcal{L}(\mathbf{W})$.
- Write down the gradients of the objective with respect to the parameters $\frac{\delta \mathcal{L}}{\delta \mathbf{W}}$ [a]

[a] Look in the appendix for help to compute the matrix derivatives. It is important that you follow how they are computed as this is something we often have to do.

In the practical section of this task we will perform the optimisation above for a real data-set.

2.1.3 Non-parametric

We will now move on to the non-linear case using GPs to model the generative mapping from the latent space to the observed data. The procedure will be much the same as in the linear case where we compute the marginal likelihood of the model and then fit the parameters.

In the linear case we integrated out the latent locations $\mathbf{X}$ and performed Type-II Maximum Likelihood estimation of the parameters of the mapping $\mathbf{W}$. It is not as straight forward in the non-parametric case. But as discussed before, rather than marginalising over $\mathbf{X}$ we can marginalise out the mapping f,

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \int p(\mathbf{Y}|f)p(f|\mathbf{X}, \boldsymbol{\theta})\mathrm{d}f$$

Question 20

Marginalisation of f is much simpler to do than marginalising out X, the latter is actually in most cases analytically intractable. Provide a simple reason why this is, the argument should be general about marginalisation and not about this model in specific. A good idea is to draw the graphical model and then follow the arrows (hint hint).

2.2 Practical

Now we will get our hands dirty, which for a machine learning scientist means including the data. Lets generate some data so that we know what we are looking to recover,

$$\mathbf{Y} = f_{\text{lin}}(f_{\text{non-lin}}(\mathbf{x})) \tag{11}$$
$$\mathbf{x} = [0, \dots, 4\pi] \tag{12}$$
$$|\mathbf{x}| = 100 \tag{13}$$
$$f_{\text{non-lin}}(x_i) = [x_i \sin(x_i), x_i \cos(x_i)] \tag{14}$$
$$f_{\text{lin}}(x') = \mathbf{A}^{\mathrm{T}}\mathbf{x}' \tag{15}$$
$$\mathbf{A} = \mathbb{R}^{10\times 2} \tag{16}$$
$$\mathbf{A}_{ij} \sim \mathcal{N}(0, 1). \tag{17}$$

The values in the linear mapping are draws 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 data-set $\mathbf{Y} \in \mathbb{R}^{100\times 10}$ which have been generated from a one dimensional *generating parameter* $\mathbf{x} \in \mathbb{R}^{1\times N}$. The aim is now to given only $\mathbf{Y}$ recover $\mathbf{x}$, i.e a single line. This is a very general and incredibly important task in machine learning, how to discover the parameters that have generated some observations and appears in many applications such as computer vision and computational biology just to name a few. It is important as many types of data are represented in very high-dimensional spaces which are very hard to interpret, providing the true generating parameters allows us to analyse the data and hopefully find the casual behavior in the data.

2.2.1 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 we can do using gradient descent. This is well implemented in `scipy.optimise`. Have a look at `SciPy Optimise` 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,

Code

```
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 21

Plot the representation that you have learned. Explain why it looks the way it does. Was this the result that you expected? Hint: Plot $\mathbf{X}$ as a two-dimensional representation.

Question 22

Draw a random two dimensional subspace (does not have to be an orthogonal basis) and plot the data. How is this result different compared to the subspace that you learnt? Provide a justification for the result.

2.2.2 Non-parametric Representation Learning

Advanced topic

This task is added for completion, and is intended for groups who are interested in getting a very good mark. However even if you do not submit this part have a look through as it is a very interesting example that shows the strength of non-parametric priors.

We will now look at non-parametric representation learning. We will use the same approach as in the regression task and use a Gaussian process prior over the mapping. These types of methods is a family of models referred to as *Gaussian Process Latent Variable Models* or GP-LVM for short. They where first introduced in [4] and have been applied to a large range of different applications [5, 6, 7]. They are also the underpinning technique of the first superhuman method for face recognition [8]. There is a great software package developed by Amazon and the Machine learning group at University of Sheffield called GPy [9]. These methods have been used extensively to real world problems and are among other things use by Ferraris F1 team to analyse data.

To perform this task you will need to compute the gradients of the marginal likelihood of the GP-regression model from Task 1.1.2. In log-space, for a single point $\mathbf{y}$ this becomes,

$$\mathcal{L} = -\frac{N}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{K}| - \frac{1}{2}\mathrm{tr}(\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y}),$$

where $|\cdot|$ is the determinant and tr is the trace of a matrix. In order to optimise the above with respect to $\mathbf{X}$ we need to compute gradients of $\mathcal{L}$ with respect to the latent variables. The simplest way to do this is to compute the gradients with respect to the kernel first and then use the chain rule to get the gradients of

$\mathbf{X}$. Now we can proceed much in the same way as in the linear case to find the latent representation. The objective function is non-convex, this together with a gradient based optimisation means that we will need to have a descent initialisation to our objective. One approach would be to initialise the latent locations using the linear approach that we just did. In the following lets do exactly that and use the PPCA solution to initialise the non-linear model.

In order to complete this task you need to,

- Derive the ML solution for the problem
- Learn a representation x and the kernel hyper-parameters directly from data
- Clearly show your results, generate data to support your argument
- Explain why and how this works

This model is much more powerful than the linear model that we did, therefore we should try to apply it to some more challenging data. On the repo I have provided a set of motion-capture sequences and some code to visualise this data.

3 The Evidence

Analysis

Compared the previous two sections, in this final part what we are even more than before looking for is your intuitions. The task outlined is designed to challenge your beliefs and your knowledge and what we want to see is how you reason. Therefore provide reasoning and discussion for the questions asked. Analyse the results and give justification.

So far we have looked at how we can derive models that can parametrise a distribution over a set of data, either in terms of a known variate as in the case of regression or in terms of latent variable for representation learning. In this task we will look at the remaining part of Bayes Rule, the evidence, and try to understand what this actually means and why it is important.

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.

Given a set of vectorial data $\mathbf{Y}$ we want to create a model of this data,

$$p(\mathbf{Y}|M_i, \boldsymbol{\theta}_i),$$

where M_i is a specific parametric model associated with parameters $\boldsymbol{\theta}_i$. In the previous task we wanted to infer the parameters from the observed data, *i.e.* $p(\boldsymbol{\theta}|\mathbf{Y}, M_i)$ using Bayes rule,

$$p(\boldsymbol{\theta}_i|\mathbf{Y}, M_i) = \frac{p(\mathbf{Y}|M_i, \boldsymbol{\theta}_i)p(\boldsymbol{\theta}_i)}{p(\mathbf{Y})}.$$

However, Bayes rule also allows us to do other interesting things. We can look at the

3.1 Theory

In the practical part of the task we will perform the experiments outlined in the excellent paper [11]. I recommend 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.

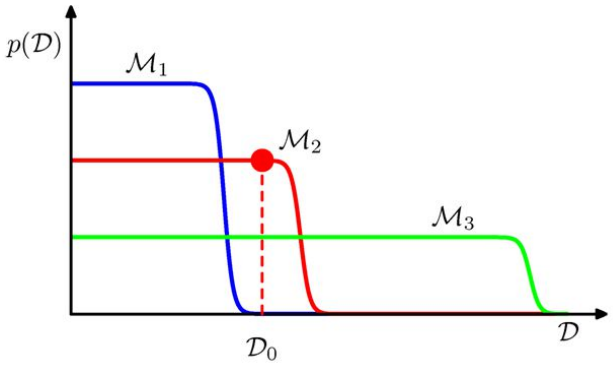

Figure 1: *The above figure (Figure 3.13 in [1]) shows the idea of model complexity and Occam's razor that was introduced in [10]. Occam's Razor tells us that we should always choose the "simplest" model that explains all of our data. In the figure the data domain is ordered on the* `x-axis` *by increasing 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 model* $\mathcal{M}_2$ *and* $\mathcal{M}_3$ *places probability mass over* $\mathcal{D}_0$ *but as* $\mathcal{M}_2$ *places more, due to it being less complex only modeling a part of the data domain, and should therefore be preferred according to Occam's Razor.*

3.1.1 Data

Consider a very simple data domain $\mathcal{D} = \{y^i\}_{i=1}^9$ where $y^i \in \{-1, 1\}$. This data is structured according to a grid whos 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.

3.1.2 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 that (I) can think of is something that simply takes all its probability mass and places it uniformly over the whole data space,

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

Question 23

What does this assumption actually imply? Make an argument for why,

1. this is the simplest possible model
2. this is the most complex model

The first model Eq.˜3.1.2 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 each y^i are independent we can factorise the model into 9 separate models,

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

where θ_i^j means the j:th element of the parameter vector for the t:th model. Each model can be expressed

using an exponential function which relates the vaule y^i to its location $\mathbf{x}^i$,

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

{Explain how the 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,

$$p(\mathcal{D}|M_2, \boldsymbol{\theta}_2) = \prod_{n=1}^{9} \frac{1}{1 + e^{-y^n (\theta_2^1 x_1^n + \theta_2^2 x_2^n)}} p(\mathcal{D}|M_3, \boldsymbol{\theta}_3) = \prod_{n=1}^{9} \frac{1}{1 + e^{-y^n (\theta_3^1 x_1^n + \theta_3^2 x_2^n + \theta_3^3)}},$$

Question 24

How have the choices we made above restricted the distribution of the model? What data sets are 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? Discuss and compare the models to each other?

3.1.3 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[1]. 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,

$$p(\mathcal{D}|M_i) = \int_{\forall \boldsymbol{\theta}} p(\mathcal{D}|M_i, \boldsymbol{\theta}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}. \tag{18}$$

Explain the process of marginalisation. Discuss its implications. The marginalisation above requires one more object that we haven't 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 very 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,

$$\begin{aligned} p(\boldsymbol{\theta}|M_i) &= \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ \boldsymbol{\mu} &= \mathbf{0} \\ \boldsymbol{\Sigma} &= \sigma^2 \boldsymbol{I} \\ \sigma^2 &= 10^3 \end{aligned} \tag{19}$$

Question 25

What does this choice of prior imply? How does the choice of the parameters of the prior $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ effect the model?

[1] In handwavy terms, don't think about things that will never happen.

Now when we have defined the prior $p(\boldsymbol{\theta})$ we just need to perform the marginalisation in Eq.˜18 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 naive Monte Carlo approach,

$$p(\mathcal{D}|M_i) \approx \frac{1}{S}\sum_{s=1}^{S} p(\mathcal{D}|M_i, \boldsymbol{\theta}^s), \tag{20}$$

$$\boldsymbol{\theta}^s \sim p(\boldsymbol{\theta}|M_i) \tag{21}$$

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.

3.2 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. When I wrote the code for this I found the following libraries very useful,

Code

```
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. I will only look at your plots and not your code so feel free to

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 data-sets such that you can easily compare the models. A good suggestion is the one provided in the appendix of the paper.

Question 26

For each model sum the evidence for the whole of $\mathcal{D}$ what numbers do you get? Explain these numbers for *all* the models and relate them to each other.

Question 27

Plot the evidence over the whole data set for each model. 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 28

Find `np.argmax` and `np.argmin` which element in $\mathcal{D}$ that is given most and least probability mass by each model. Plot the data-sets which are given the highest and lowest evidence for each model. Discuss these results, do they make sense?

Question 29

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 $\mu = [5, 5]^{\mathrm{T}}$?
- Redo evidence plot for these and explain the changes compared to using zero-mean.

4 Final thoughts

If you have made it this far and managed to do all the experiments outlined in this task you have done very well. As a last bit I recommend you to go back and read the abstract again, did performing this lab answer the main question stated there?

Question 30

Summarise the assignment in one paragraph, what have you learnt and what do you feel have been the prupose/message of performing this.

Good Luck!

References

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[2] K. B. Petersen and M. S. Pedersen. The Matrix Cookbook, November 2012. Version 20121115.

[3] Jan R Magnus and Heinz Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. Wiley, 1988.

[4] Neil D Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

[5] Wenbin Li, Fabio Viola, Jonathan Starck, Gabriel J. Brostow, and Neill D.F. Campbell. Roto++: Accelerating professional rotoscoping using shape manifolds. *ACM Transactions on Graphics (In proceeding of ACM SIGGRAPH'16)*, 35(4), 2016.

[6] Raquel Urtasun, David J Fleet, and P. Fua. 3D people tracking with Gaussian process dynamical models. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 1:238–245, 2006.

[7] Keith Grochow, Steven L Martin, Aaron Hertzmann, and Zoran Popović. Style-based inverse kinematics. *SIGGRAPH '04: SIGGRAPH 2004 Papers*, August 2004.

[8] Chaochao Lu and Xiaoou Tang. Surpassing Human-Level Face Verification Performance on LFW with GaussianFace. *arXiv.org*, April 2014.

[9] James Hensman, Nicolo Fusi, Ricardo Andrade, Nicolas Durrande, Alan Saul, Max Zwießele, and Neil D Lawrence. GPy: A Gaussian process framework in python. 2014.

[10] David J C Mackay. *Bayesian methods for adaptive models* . PhD thesis, California Institute of Technology, California Institute of Technology, December 1991.

[11] Iain Murray and Zoubin Ghahramani. A note on the evidence and Bayesian Occam's razor. Technical Report GCNU-TR 2005-003, August 2005.

5 Appendix

5.1 Computing the gradients Q19

In question 19 we will derive the gradient for Principled Component Analysis, in the formulation in the book the author derive a closed form solution for the Type-II Maximum likelihood, this is rather tricky so insted we will do a gradient based approach. Deriving these are still not a walk in the park if you are not comfortable with matrix derivatives. I will here go through a couple of tips on how to get you there.

The first thing that we need to do is to write up the objective function, this is going to consist of three terms which looks like this,

$$\mathcal{L}(\mathbf{W}) = \text{constant} + \log|\mathbf{C}(\mathbf{W})| + \sum_{i}^{N} \mathbf{y}_i^{\mathrm{T}} (\mathbf{C}(\mathbf{W}))^{-1} \mathbf{y}_i$$

Now we need to compute the derivatives of $\mathbf{C}(\mathbf{W})$ with respect to $\mathbf{W}$. To do this lets first rewrite everything on matrix form so that we can remove the sum from the objective function. First note that,

$$\sum_i \mathbf{x}_i \mathbf{x}_i = \text{tr}\left(\begin{bmatrix} \leftarrow & \mathbf{x}_1 & \rightarrow \\ \leftarrow & \mathbf{x}_2 & \rightarrow \\ & \vdots & \\ \leftarrow & \mathbf{x}_N & \rightarrow \end{bmatrix} \begin{bmatrix} \leftarrow & \mathbf{x}_1 & \rightarrow \\ \leftarrow & \mathbf{x}_2 & \rightarrow \\ & \vdots & \\ \leftarrow & \mathbf{x}_N & \rightarrow \end{bmatrix}^{\mathrm{T}} \right),$$

this means that we can rewrite the objective function as,

$$\mathcal{L}(\mathbf{W}) = \text{constant} + \log|\mathbf{C}(\mathbf{W})| + \text{tr}\left(\mathbf{Y}(\mathbf{C}(\mathbf{W}))^{-1}\mathbf{Y}^{\mathrm{T}}\right).$$

Now we have two terms we need to take derivatives of, both of these include the same matrix $\mathbf{C}$ in one form of the other so it will come in handy to take this derivative first,

$$\frac{\partial \mathbf{C}}{\partial \mathbf{W}_{ij}} = \frac{\partial \mathbf{W}\mathbf{W}^{\mathrm{T}}}{\partial \mathbf{W}_{ij}}.$$

We will now use the excellent Matrix Cookbook [2] to find the rules that we need to figure this one out. If you use the version from 2012 URL we can first use Eq. 37 to rewrite the the product,

$$\partial(\mathbf{X}\mathbf{Y}) = (\partial\mathbf{X})\mathbf{Y} + \mathbf{X}(\partial\mathbf{Y}).$$

We can then combine this with Eq. 32 which states,

$$\frac{\partial \mathbf{X}_{kl}}{\partial \mathbf{X}_{ij}} = \delta_{ik}\delta_{lj},$$

where δ_{ij} is the kronecker delta function,

$$\delta_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

This leads to the following derivative,

$$\frac{\partial \mathbf{W}\mathbf{W}^{\mathrm{T}}}{\partial \mathbf{W}_{ij}} = \mathbf{W}\frac{\partial \mathbf{W}^{\mathrm{T}}}{\partial \mathbf{W}_{ij}} + \frac{\partial \mathbf{W}}{\partial \mathbf{W}_{ij}}\mathbf{W}^{\mathrm{T}} = \mathbf{W}\mathbf{J}_{ij} + \mathbf{J}_{ji}\mathbf{W}^{\mathrm{T}}$$

where we $\mathbf{J}_{ij}$ is a matrix who has all zero entries except for $(\mathbf{J}_{ij})_{ij} = 1$.

Now lets strart by tackling the first term, the derivative of the log determinant.

$$\frac{\partial}{\partial \mathbf{W}_{i}j}\log|\mathbf{C}|$$

We will start by using Eq. 43 that states that,

$$\partial \log|\mathbf{X}| = \mathrm{tr}\left(\mathbf{X}^{-1}\partial\mathbf{X}\right).$$

We can now rewrite this as,

$$\frac{\partial}{\partial \mathbf{W}_{i}j}\log|\mathbf{C}| = \mathrm{tr}\left(\mathbf{C}^{-1}\frac{\partial \mathbf{C}}{\partial \mathbf{W}_{ij}}\right).$$

So now we have our first term and we can move on to the second. This term is a derivative of a trace we can now use Eq. 36 which states,

$$\partial\left(\mathrm{tr}(\mathbf{X})\right) = \mathrm{tr}\left(\partial\mathbf{X}\right).$$

This means that our second term becomes,

$$\frac{\partial}{\partial \mathbf{W}_{ij}}\mathrm{tr}\left(\mathbf{Y}(\mathbf{C})^{-1}\mathbf{Y}^{\mathrm{T}}\right) = \mathrm{tr}\left(\frac{\partial}{\partial \mathbf{W}_{ij}}\mathbf{Y}(\mathbf{C})^{-1}\mathbf{Y}^{\mathrm{T}}\right).$$

Now we want to break the quadratic form and we will do this by using the chain-rule to do the derivative,

$$\mathrm{tr}\left(\frac{\partial}{\partial \mathbf{W}_{ij}}\mathbf{Y}(\mathbf{C})^{-1}\mathbf{Y}^{\mathrm{T}}\right) = \mathrm{tr}\left(\frac{\partial}{\partial \mathbf{C}^{-1}}(\mathbf{Y}\mathbf{C}^{-1}\mathbf{Y}^{\mathrm{T}})\frac{\partial \mathbf{C}^{-1}}{\partial \mathbf{W}_{ij}}\right) = \mathrm{tr}\left((\mathbf{Y}\mathbf{Y}^{\mathrm{T}})^{\mathrm{T}}\frac{\partial \mathbf{C}^{-1}}{\partial \mathbf{W}_{ij}}\right).$$

Now we have isolated the derivative and we are nearly there. The last rule we will use is to use the derivative of a matrix inverse Eq. 40,

$$\partial\mathbf{X}^{-1} = -\mathbf{X}^{-1}(\partial\mathbf{X})\mathbf{X}^{-1}.$$

This means we can reach the derivative of the last term as,

$$\mathrm{tr}\left((\mathbf{Y}\mathbf{Y}^{\mathrm{T}})^{\mathrm{T}}\frac{\partial \mathbf{C}^{-1}}{\partial \mathbf{W}_{ij}}\right) = \mathrm{tr}\left(\mathbf{Y}^{\mathrm{T}}\mathbf{Y}\left(-\mathbf{C}^{-1}\frac{\partial \mathbf{C}}{\partial \mathbf{W}_{ij}}\mathbf{C}^{-1}\right)\right),$$

where we know the derivative of the inner-most term as we computed it first.

So now we have the full derivative, as we can see the dimensionality makes sense, we take the derivative of our objective function which is a scalar with another scalar and therefore expect a scalar back, as both of our terms are traces of matrices this makes sense. Now you just have to code this up and put this into the optimiser and see what solution comes out.