COM4509 and COM6509 Revision Worksheet (First half of module)

To help with the first part of the module we have prepared this exercise sheet. It covers *some* of the key topics you should ensure you spend time revising. For a few of the questions we have provided an answer, indicated by a (Q). We won't be providing answers to all these questions but you should be able to find all the answers in the lecture slides. We have highlighted some questions with [$\star$] or [$\star\star\star$]. These aren't just the ones that are most important, but are the ones that people are most likely to get wrong and we think need extra attention.

1 Basics

Check you understand these terms - can you write down a definition for each?

1. Supervised or Unsupervised Learning.
2. Regression and Classification. (Q) Give an example of a (1) regression and (2) classification problem.
3. Training set, Validation set, Test set [$\star$]
4. Leave-one-out cross validation.
5. Generalisation [$\star$]
6. Extrapolation vs Interpolation. (Q) Give examples of these.
7. Objective Function (Q).
8. Circular Analysis (Q) Define it and give an example. [$\star\star\star$]

Solution:

- (Q2) Give an example of a (1) regression and (2) classification problem. Regression: Predicting the average global temperature in 2050. Classification: Identifying the species of an Isopod from an image. We usually consider a problem to be regression if the data is ordinal even if it isn't continuous: E.g. predict number of students at the lab this week.
- (Q6) Extrapolation vs Interpolation. Give examples of these. Extrapolation - weather forecasting; Interpolation - predicting a student's grade from their attendance, given the rest of the class's attendance and grades.
- (Q7) The objective is typically the function to be minimised or maximised, typically for parameter estimation. Typically it might consist of a measure of how badly a model is predicting a training set, and one would wish to minimise this function. It might include a regularisation term. [Side Note: 'cost function' and 'loss function' are used to sometimes mean the same or related things and as these terms are often used interchangeably by many authors, don't get stressed about trying to discern a distinction between them!]

- (Q8) Circular Analysis. Define: Edited from wikipedia: 'the selection of the details [such as parameters] of a data analysis using the data that is being [used for testing].' Example. There are 152 parameters involved in the preprocessing of fMRI data, if we adjust these all to get the 'best' answer for our test data, then we will be reporting a much more significant result than is true.

2 Entropy

1. Write down the equation that defines what the expectation is of a function of a discrete random variable. [$\star$]
2. (Q) Write down the equation that defines the entropy of a discrete random variable, X. [$\star$]
3. What base do you need to use for your log to get an answer in bits?
4. What is the entropy of a fair dice roll, in bits?
5. What is the entropy of an unfair coin toss (with a 90% chance of heads?).
6. (Q) Define conditional entropy in words.
7. (Q) We know the conditional probability of rain if it's cloudy is 30%, and is only 5% if not cloudy. If there is a 80% chance of it being cloudy, what is the conditional entropy of rain/not-rain given cloud state? [$\star$]
8. (Q) Define information gain.
9. Using the example in Q7, what is the marginal probability of rain?
10. What is the entropy of rain/no-rain? Confirm this is not-less-than the conditional entropy you computed previously.

Solution:

- (Q2) Write down the equation that defines the entropy of a discrete random variable, X. (Q).

$$H[X] = E[-\log p(X)] = -\sum_{i=1}^{N} p(x_i) \log p(x_i),$$

  where x_i are the possible values of X.

- (Q6) Define conditional entropy: I'll let you look at the slides for the equation, but in words: the conditional entropy of X given Y is the expectation (over Y) of the entropy of X given the values of Y. So one finds the entropy of X—Y for each possible value of Y, and then finds the sum, weighted by the probability of each value of Y.

- (Q7) Cloud question: We compute the entropy of rain/no-rain given it's cloudy $-(0.3 \log_2 0.3 + 0.7 \log_2 0.7) = 0.88$ and the entropy of rain/no-rain given it's not cloudy $-(0.05 \log_2 0.05 + 0.95 \log_2 0.95) = 0.29$. We find the expectation then: $0.8 \times 0.88 + 0.2 \times 0.29 = 0.76$ bits.

- (Q8) Information gain: It is the expected reduction in entropy given new information $H(X) - H(X|Y)$.

3 End-to-End ML

1. How does one compute: (a) the root mean squared error, (b) the mean absolute error? [$\star$] (c) the negative log-predictive density (NLPD) [$\star$].
2. (Q) Draw the confusion matrix for these results: My machine learning algorithm labels images of Bees, Wasps and Flies.
   - Of the ten true bees, it labels seven correctly, two as wasps and one as a fly.
   - Of the five true wasps, it labels all of them correctly.
   - of the eight flies, it labels four as flies and four as wasps.
3. A dataset of traffic speeds in different conditions are collected; with each datapoint taken every minute. A *random* cross-validation approach is used to see if the traffic speed could be predicted from the weather conditions – it was found to do really well. (a) What problem is there with this result? (b) How might you address this? (Q) [$\star\star\star$]

Solution:

- (Q2) Draw the confusion matrix

		Predicted	Class	
		Bee	Wasp	Fly
True	Bee	7	2	1
Class	Wasp	0	5	0
	Fly	0	4	4

- (Q3) Neighbouring minutes are likely to have similar weather *and* similar traffic speeds, even if these two variables are largely independent – thus if we train on a random subset and test on the remaining, we are likely to do well, just because of the correlation between neighbouring times. We should instead ensure that there isn't such a correlation - for example by testing on data from other days. An additional mistake is that, if one uses (for example) test points all from the same day – the results are highly correlated, these aren't independent samples, so the resulting accuracy score won't really give a good insight into the accuracy of the method. Ideally you would have test points from many different days and weather conditions.

4 Decision Trees

1. (Q) Explain the general idea of how we use 'purity' to build the decision tree?
2. (Q) We want to determine whether a wind turbine needs remedial maintenance on its blades (i.e. its 'state' or whether it will fail). We have data from 16 turbines (10 functioning, 6 failed). An example feature is whether there are additional vibrations. 8 of the turbines have additional vibrations. Of these 5 are failed, and three are functioning. Of those 8 without additional vibrations, 1 has failed and 7 are functioning. Compute using this, the information gain this feature provides. [$\star$]
3. (Q) What method can we use to help avoid over-fitting in decision trees?

Solution:

- (Q1) Explain the general idea of how we use 'purity' to build the decision tree? When building a decision tree, at each node we want to pick a feature that splits the data in such a way that the two sets of data (on each branch) are each as pure as possible. I.e. the labels are as unmixed as possible (this applies to both categorical labels, where one would want them to ideally be of the same class down each branch) and of continous labels (where one would want them to have as little variance as possible on each branch).

- (Q2) We want to determine whether a wind turbine needs remedial maintenance on its blades (i.e. its 'state': whether it will fail). We have data from 16 turbines (10 functioning, 6 failed). An example feature is whether there are vibrations. Of those 8 turbines with vibrations: 5 have failed, and three are functioning. Of those 8 without vibrations, 1 has failed and 7 are functioning. Compute using this information, the information gain that the 'vibration' feature provides. If the height of turbine feature provides 0.1 bits of information gain, which feature would you use to split the data?

  $$H[\text{state}] = -(10/16 \times \log_2(10/16) + 6/16 \times \log_2(6/16)) = 0.95$$

  H[state|vibration] $= -(8/16 \times (5/8 \log_2 5/8 + 3/8 \log_2 3/8) +$
  $8/16 \times (1/8 \log_2 1/8 + 7/8 \log_2 7/8)) = 0.75$

  So the information gain is about 0.2 bits. As this is more than the 0.1 bits from the turbine height feature, we would use the vibration feature to initially split the data.

- (Q3) What method can we use to help avoid overfitting in decision trees? One possible answer is: Pruning (pre or post-).

5 Ensemble Methods

1. (Q) A neural network predicts river flow given rainfall data from some rain gauges. There is training data from 50 days. Explain how you can use bagging (bootstrap aggregation) to compute a 95% confidence interval for the river flow given today's gauge measurements.[$\star$]
2. (Q) Explain the random forest algorithm step-by-step.[$\star\star\star$]

Solution:

- (Q1) The key is that we must *resample with replacement the training data.* So we sample 50 days WITH REPLACMENT from the training data (this means some items will be in the new set more than once, while others will be missing). We then train the neural network using this sample, and make a prediction using the model on today's measurements. We then repeat this process many times. We are left then with a distribution of answers. To find the 95% CI we find the 2.5% and 97.5%-percentile results (e.g. if there were 10000 resamplings, we could order the results and use the 250th and 9750th predictions).
  Importantly this does not give any indication of the accuracy of the underlying model - if the model is wrong or bias, or the training data doesn't represent today's measurements well, then the prediction is likely to lie outside the confidence interval.

- (Q2) (1) Multiple bootstrap samples are generated from the dataset *with replacement* to build a new training set. (2) A decision tree is built with each of these training sets. (3) at each split a small number of the features are used, this is called *subspace sampling.* (4) The many trees built in this way are each used to make a prediction – if classifying we uses the most popular predicted class, if regressing we use the average of the tree predictions.

6 Linear Regression

1. (Q) If $\boldsymbol{y}$ is a column vector of N continuous training data labels; X is an $N \times D$ design matrix of regressors; and $\boldsymbol{w}$ is a column vector of D parameters, write for the linear regression model the sum-squared error (SSE) in vector/matrix notation.
2. (Q) Write, maybe using a summation sign and the 'abs' function $|\cdot|$, an expression for the mean absolute error for the above. Notice this is the mean rather than the sum. [1][$\star$]
3. (Q) Differentiate (a) the SSE cost function[$\star$] and (b) the MAE wrt $\boldsymbol{w}$[$\star\star\star$].
4. Show that if our likelihood is based on the assumption that independent and identically distributed Gaussian noise is added to our linear model to generate the training values in $\boldsymbol{y}$, then value of $\boldsymbol{w}$ that maximises the likelihood is the same value of $\boldsymbol{w}$ that minimises the sum-squared-error.[$\star$]
5. (Q) We are modelling the height of a tree over time. We wish to build a design (regressor) matrix using a 3rd order polynomial basis. We have five training observations at times 2,3,4,6 and 8 years. What does the design matrix look like?
6. (Q) What would you add to the cost function to incorporate L_2 reguarlisation?
7. What would you add for L_1 regularisation?[$\star$]
8. Explain standard gradient descent. What is the learning rate parameter for? What happens if it is too large? Or too small?

Solution:

- (Q1) Write for the linear regression model the sum-squared error in vector and matrix notation. We start by writing the prediction. This is $X\boldsymbol{w}$. The error will be $\boldsymbol{y} - X\boldsymbol{w}$. The sum of the squares of the errors is $(\boldsymbol{y} - X\boldsymbol{w})^\top(\boldsymbol{y} - X\boldsymbol{w})$.
- (Q2) The error for one training point, i, is: $y_i - [X]_{i:}\boldsymbol{w}$. So the absolute error is $|y_i - [X]_{i:}\boldsymbol{w}|$. So the mean absolute error, is $\frac{1}{N}\sum_{i=1}^{N}\Big|y_i - [X]_{i:}\boldsymbol{w}\Big|$.
- (Q3)
  (a) An easy way to do this is to multiply out the brackets, $(\boldsymbol{y} - X\boldsymbol{w})^\top(\boldsymbol{y} - X\boldsymbol{w}) = \boldsymbol{y}^\top\boldsymbol{y} - \boldsymbol{y}^\top X\boldsymbol{w} - (X\boldsymbol{w})^\top\boldsymbol{y} + (X\boldsymbol{w})^\top X\boldsymbol{w}$. Then differentiate each term wrt $\boldsymbol{w}$:

  $$0 - X^\top\boldsymbol{y} - X^\top\boldsymbol{y} + 2X^\top X\boldsymbol{w}.$$

  (b) The gradient of the MAE is a bit more tricky to write. We want to compute $\frac{d}{d\boldsymbol{w}}\frac{1}{N}\sum_{i=1}^{N}\Big|y_i - [X]_{i:}\boldsymbol{w}\Big|$. Which is equal to

  $$\frac{1}{N}\sum_{i=1}^{N}\frac{d}{d\boldsymbol{w}}\Big|y_i - [X]_{i:}\boldsymbol{w}\Big|.$$

  [edit: Sorry, I made a mistake in a previous version and accidentally left out a minus sign.] If $y_i - [X]_{i:}\boldsymbol{w}$ is POSITIVE then $\Big|y_i - [X]_{i:}\boldsymbol{w}\Big| = y_i - [X]_{i:}\boldsymbol{w}$, and the gradient wrt $\boldsymbol{w}$ is simply $-[X]_{i:}$.

  If $y_i - [X]_{i:}\boldsymbol{w}$ is NEGATIVE then $\Big|y_i - [X]_{i:}\boldsymbol{w}\Big| = -(y_i - [X]_{i:}\boldsymbol{w})$, and the gradient wrt $\boldsymbol{w}$ is $+[X]_{i:}$.

[1] As a side note, some libraries and tools use the sum, others the mean, so I've switched between these to get you used to both - sorry if that's also quite confusing!

We can express this using the 'sgn(·)' function. This returns a +1 if the argument is positive, and -1 if it is negative. So we can combine these two situations and write the gradient of the term as,

$$-[X]_{i:}\ \mathrm{sgn}\big(y_i - [X]_{i:}\boldsymbol{w}\big).$$

So the overall gradient is,

$$-\frac{1}{N}\sum_{i=1}^{N}[X]_{i:}\ \mathrm{sgn}\big(y_i - [X]_{i:}\boldsymbol{w}\big).$$

- (Q5) The design matrix is:
$$\begin{bmatrix} 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \\ 1 & 6 & 36 & 216 \\ 1 & 8 & 64 & 512 \end{bmatrix}$$
- (Q6) For L_2 (ridge) regression: Add a term $\lambda \boldsymbol{w}^\top \boldsymbol{w}$ (sum of the square of your parameters) to your current cost. This penalises large parameter values.

7 Gaussian Processes

1. (Q) The exponential kernel is
$$k(x, x') = e^{-|x-x'|/l}.$$
Assume we have zero-noise in our observations, and two training data points, at $x_1 = 1$, $y_1 = 2$ and at $x_2 = 3$, $y_2 = 4$. And assume zero-mean. A kernel lengthscale of 10. Our test point is at $x_* = 2$.
(a) Compute the covariance between the test point and the training points, i.e. $\boldsymbol{k}_{*f}$.
(b) Compute the covariance matrix between training points, i.e. K_{ff}.
(c) With no observation noise, the posterior mean is $\boldsymbol{k}_{*f} K_{ff}^{-1} \boldsymbol{y}$ and the variance is $k_{**} - \boldsymbol{k}_{*f} K_{ff}^{-1} \boldsymbol{k}_{f*}$. Compute the posterior mean and variance of the posterior at the test point. [note that I will give you these two expressions in the exam if needed, similarly I won't expect you to invert a big matrix! However, during revision it makes sense you are able to do these steps!][$\star\star\star$]
2. Try the above again, but using the exponentiated quadratic kernel.
3. Getting a bit more of an intuition about the effect of the noise variance and lengthscale would be useful. Maybe return to Lab 10 (Gaussian processes) and look at the answers to question 5, seeing how different values change the fit.
4. (Q) Define Gaussian process.
5. (Q) How might we handle our uncertainty in the lengthscale? [$\star\star\star$]
6. Explain why each of these might have issues if modelled by a 'standard' Gaussian processes (a) the choice of sock colour of a student over time (b) rainfall (c) a student's grades over time.
7. What would increasing the lengthscale do to the covariance between different locations when using the EQ kernel [$\star\star\star$].

Solution:

- (Q1) (a) $\boldsymbol{k}_{*k}$ can be computed by entering the test point at the two training points into the kernel function, so;

$$\boldsymbol{k}_{*f} = \begin{bmatrix} k(1,2) & k(3,2) \end{bmatrix} = \begin{bmatrix} e^{-1/10} & e^{-1/10} \end{bmatrix} \approx \begin{bmatrix} 0.90 & 0.90 \end{bmatrix}$$

  (b) Similarly,

$$K_{ff} = \begin{bmatrix} k(1,1) & k(1,3) \\ k(3,1) & k(3,3) \end{bmatrix} = \begin{bmatrix} e^{0} & e^{-2/10} \\ e^{-2/10} & e^{0} \end{bmatrix} \approx \begin{bmatrix} 1 & 0.82 \\ 0.82 & 1 \end{bmatrix}.$$

  (c) With no observation noise, the posterior mean is $\boldsymbol{k}_{*f} K_{ff}^{-1} \boldsymbol{y}$. The posterior mean is then,

$$y_* \approx \begin{bmatrix} 0.90 & 0.90 \end{bmatrix} \left(\begin{bmatrix} 1 & 0.82 \\ 0.82 & 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} 2 \\ 4 \end{bmatrix} \approx 2.99$$

  This makes sense. The mean is about halfway between the two training point values, and given the long lengthscale the posterior will be roughly a straight line between them.

  The variance is going to be $k_{**} - k_{*f} K_{ff}^{-1} k_{f*}$. Noting that k_{**} equals $e^0 = 1$, the posterior variance is 0.099. So the posterior standard deviation is 0.32. So the possible functions that exist have still a wide spread, this is because the exponential kernel leads to very rough functions.

- (Q4) Adapted from wikipedia: A 'Gaussian process is a stochastic process (a collection of random variables indexed by time or space), such that every finite collection of those random variables has a multivariate normal distribution'. To put it less formally, it is a stochastic process such that if one takes samples from any two points they will form a Gaussian distribution when plotted against each other.

- (Q5) The uncertainty in the lengthscale can be incorporated into our final predictions by modelling the uncertainty in a Bayesian manner. Specifically, we can place a 'hyperprior' on the value of the lengthscale[2] then 'integrate out' the lengthscale. We write this hyperprior as $p(\theta)$ - This is the distribution of possible values of our hyperparameters, prior to seeing data.

  Let's break down how we might compute a posterior prediction integrating out the hyperparameters:

  - First, we write down what we want to compute. We want the posterior prediction of the function at our test points, given our training data X, y: $p(f_*|X, y)$, but not conditioned on a particular value of hyperparameters.
  - Second we note that we can compute a prediction for given values of hyperparameters (θ - e.g. the lengthscale): $p(f_*|X, y, \theta)$. This is what we learnt to do before in lectures, and it gives us a Gaussian distribution at our test points.
  - Third: We can also compute the *marginal likelihood* – this is how likely our data is, given the model and hyperparameters: $p(y|X, \theta)$. This can also be computed in closed form (I won't cover how for this module).

  We can put this together:

$$p(f_*|X, y) = \frac{\int p(f_*|X, y, \theta) p(y|X, \theta) p(\theta) d\theta}{\int p(y|X, \theta) p(\theta) d\theta}$$

  ☐ Note: To check this is true, it is worth computing each part. First note the contents of the numerator's integral can be written as $p(f_*, y, \theta|X)$. So integrating over θ gives $p(f_*, y|X)$. The contents of the denominator's integral can be written as $p(y, \theta|X)$; so its integral is $p(y|X)$. We finally note that $p(f_*, y|X)/p(y|X) = p(f_*|X, y)$.

[2] For those interested an inverse gamma might be a good choice.

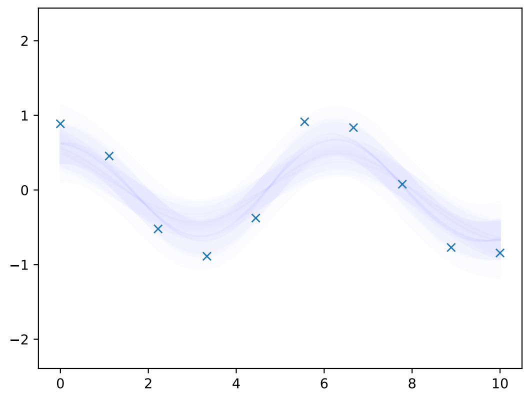
So we've an expression that in principle gives us the right answer, but it can't be computed in closed form.

There are lots of tricks to compute approximations to this. One fairly simple approach is to draw samples from the hyperprior $p(\theta)$, then compute the posterior predictive distribution $p(f_*|X, y, \theta)$ and the marginal likelihood, $p(y|X, \theta)$ for each sampled value; then find the sum of the predictive distributions, 'weighted' by the associated marginal likelihoods. The approximate denominator can be computed then at the same time.

I've skimmed over some details, as I want you just to have the general idea of how you might handle the uncertainty in the hyperparameters in a Bayesian fashion.

☐ **Summary**: We can compute predictions for specific values of our lengthscale in closed form. To incorporate the uncertainty in the lengthscale, we sample from a prior distribution for the lengthscale, then compute the prediction and the marginal likelihood for each, and then average the predictions together weighted by the marginal likelihoods.

Figure shows an example of sampling a series of different lengthscales (from an inverse gamma prior) and drawing the posterior predictions, weighted by the normalised marginal likelihoods. Unrealistically I didn't also sample to select the noise-parameter, but this hopefully still gives the idea: there are two plausible lengthscales (short or long).



8 Kernel Trick

1. Review the kernel trick slides, in particular think about the shape of $XX^\top$ and $X^\top X$ and which is easier to invert.

2. Think about what the calculations are in computing each term in $XX^\top$.