COMP9417 - Machine Learning
Tutorial: Regression I

In this tutorial we will explore the theoretical foundations of linear regression. We first work through linear regression in 1 dimension (univariate linear regression), and then extend the analysis to multivariate linear regression. If you find yourself unsure about any of the math covered in this tutorial, we strongly recommend reading the corresponding material in the text (freely available online):

Mathematics for Machine Learning by Marc Peter Deisenroth, A. Aldo Faisal and Cheng Soon Ong.

We will refer to this as the MML book throughout the course.

Question 1. (Univariate Least Squares)

A *univariate linear regression model* is a linear equation $y = w_0 + w_1 x$. Learning such a model requires fitting it to a sample of training data, $(x_1, y_1), \ldots, (x_n, y_n)$, so as to minimize the loss (usually Mean Squared Error (MSE)), $\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(y_i - (w_0 + w_1 x_i))^2$. To find the best parameters w_0 and w_1 that minimize this error function we need to find the error *gradients* $\frac{\partial \mathcal{L}}{\partial w_0}$ and $\frac{\partial \mathcal{L}}{\partial w_1}$. So we need to derive these expressions by taking partial derivatives, set them to zero, and solve for w_0 and w_1.

(a) Derive the least-squares estimates (minimizers of the MSE loss function) for the univariate linear regression model.

(b) Show that the centroid of the data, i.e. the point $(\overline{x}, \overline{y})$ is always on the least squares regression line.

(c) To make sure you understand the process, try to solve the following loss function for linear regression with a version of "$L2$" regularization, in which we add a penalty that penalizes the size of w_1. Let $\lambda > 0$ and consider the regularised loss

$$\mathcal{L}(w_0, w_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - (w_0 + w_1 x_i))^2 + \lambda w_1^2$$

Question 2. (Multivariate Least Squares)

In the previous question, we found the least squares solution for the univariate (single feature) problem. We now generalise this for the case when we have p features. Let $x_1, x_2, \ldots, x_n$ be n feature vectors (e.g. corresponding to n instances) in $\mathbb{R}^p$, that is:

$$x_i = \begin{bmatrix} x_{i0} \\ x_{i1} \\ \vdots \\ x_{ip-1} \end{bmatrix}$$

We stack these feature vectors into a single matrix, $X \in \mathbb{R}^{n \times p}$, called the *design* matrix. The convention is to stack the feature vectors so that each row of X corresponds to a particular instance, that is:

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix} = \begin{bmatrix} x_{10} & x_{11} & \cdots & x_{1,p-1} \\ x_{20} & x_{21} & \cdots & x_{2,p-1} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n0} & x_{n1} & \cdots & x_{n,p-1} \end{bmatrix}$$

where the superscript T denotes the transpose operation. Note that it is standard to take the first element of the feature vectors to be 1 to account for the bias term, so we will assume $x_{i0} = 1$ for all $i = 1, \ldots, n$. Analagously to the previous question, the goal is to learn a weight vector $w \in \mathbb{R}^p$, and make predictions:

$$\hat{y}_i = w^T x_i = w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_{p-1} x_{i,p-1},$$

where $\hat{y}_i$ denotes the i-th predicted value. To solve for the optimal weights in w, we can use the same procedure as before and use the MSE:

$$\begin{aligned} \mathcal{L}(w_0, w_1, \ldots, w_{p-1}) &= \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \frac{1}{n} \sum_{i=1}^{n} (y_i - (w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_{p-1} x_{i,p-1}))^2. \end{aligned}$$

One approach to solve this would be to take derivatives with respect to each of the p weights and solve the resulting equations, but this would be **extremely** tedious. The more efficient way to solve this problem is to appeal to matrix notation. We can write the above loss as:

$$\mathcal{L}(w) = \frac{1}{n} \|y - Xw\|_2^2,$$

where $\| \cdot \|_2$ is the Euclidean norm. For the remainder of this question, we will assume that X is a full-rank matrix, which means that we are able to compute the inverse of $X^T X$.

(a) Show that $\mathcal{L}(w)$ has a critical point:

$$\hat{w} = (X^T X)^{-1} X^T y,$$

note: critical point here means that $(\frac{d\mathcal{L}(w)}{w})(\hat{w}) = 0$, i.e. the gradient evaluated at the critical point $\hat{w}$ is zero).

Hint 1: if u is a vector in $\mathbb{R}^n$, and v is a fixed vector in $\mathbb{R}^n$, then $\frac{\partial v^T u}{\partial u} = v$.

Hint 2: if A is a fixed $n \times n$ matrix, and if $f = z^T A z$, then $\frac{\partial u^T A u}{\partial u} = Au + A^T u$.

(b) The condition $(\frac{d\mathcal{L}(w)}{w})(\hat{w}) = 0$ is necessary but not sufficient to show that $\hat{w}$ is a (global) minimizer of $\mathcal{L}$, since this point could be a local minimum or a saddle point. Show that the critical point in part (a) is indeed a global minimizer of $\mathcal{L}$.

Hint 1: $\mathcal{L}(w)$ is a function of $w \in \mathbb{R}^p$, and so its Hessian, H, is the $p \times p$ matrix of second order partial derivatives, that is, the (k, l)-th element of H is

$$H_{kl} = \frac{\partial \mathcal{L}(w)}{\partial w_k \partial w_l}.$$

We will often write $H = \nabla_w^2 \mathcal{L}(w)$, where ∇ is the gradient operator, and ∇^2 means taking the gradient twice. Note that the Hessian plays the role of second derivative for multivariate functions.

Hint 2: a function is convex if its Hessian is positive semi-definite, which means that for any vector u,

$$u^T H u \geq 0.$$

Note also that this condition means that for any choice of u, the product term will always be non-negative.

Hint 3: Any critical point of a convex function is a global minimum.

(c) In the next parts, we will use the formula derived above to verify our solution in the univariate case. We assume that $p = 2$, so that we have a two dimensional feature vector (one term for the intercept, and another for our feature). Write down the following values:

$$x_i, \quad y, \quad w, \quad X, \quad X^T X, \quad (X^T X)^{-1}, \quad X^T y.$$

(d) Compute the least-squares estimate for the $p = 2$ case using the results from the previous part.

(e) Consider the following problem: we have inputs $x_1, \ldots, x_5 = 3, 6, 7, 8, 11$ and outputs $y_1, \ldots, y_5 = 13, 8, 11, 2, 6$. Compute the least-squares solution and plot the results both by hand and using python. Finally, use the sklearn implementation to check your results.

(f) **Advanced:** Some of you may have been concerned by our assumption that X is a full-rank matrix, which is an assumption made to ensure that $X^T X$ is an invertible matrix. If $X^T X$ was not invertible, then we would not have been able to compute the least squares solution. So what happens if X is not full-rank? Does least squares fail? The answer is no, we can use something called the pseudo-inverse, also referred to as the Moore-Penrose Inverse. This is outside the scope of this course, and the interested reader can refer to the following notes, or chapter 2 in MML. At a very high level, the pseudo-inverse is a matrix that acts like the inverse for non-invertible matrices. In NumPy, we can easily compute the pseudo inverse using the command 'np.linalg.pinv'.

(g) Discuss the idea of a *feature map*. How would you use a feature map in the context of least squares regression?

(h) The mean-squared error (MSE) is

$$\text{MSE}(w) = \frac{1}{n} \|y - Xw\|_2^2,$$

whereas the sum of squared errors (SSE) (also referred to as the residual sum of squares (RSS)) is

$$\text{SSE}(w) = \|y - Xw\|_2^2.$$

Are the following statements True or False. Explain why.

(i) $\arg\min_{w \in \mathbb{R}^p} \text{MSE}(w) = \arg\min_{w \in \mathbb{R}^p} \text{SSE}(w)$

(ii) $\min_{w \in \mathbb{R}^p} \text{MSE}(w) = \min_{w \in \mathbb{R}^p} \text{SSE}(w)$

Notation: recall that $\min_x g(x)$ is the smallest value of $g(x)$, whereas $\arg\min_x g(x)$ is the value of x that minimizes $g(x)$. So $\min_x (x-2)^2 = 0$ but $\arg\min_x (x-2)^2 = 2$.

Question 3. (Population Versus Sample Parameters)

(a) What is the difference between a population and a sample?

(b) What is a population parameter? How can we estimate it?