1 Linear Regression: Assembling the Design Matrix

Lets say we have 5 training samples, each with a predictor (x_i) and target (y_i). The predictors are the mass of different chemicals and the targets are the time taken by each chemical to react to an agent. These training samples are summarised below:

Mass (x)	20	55	100	180	220
Time (y)	3	5	8	15	25

Table 1: Training set.

The training samples can be visualised in the plot below:

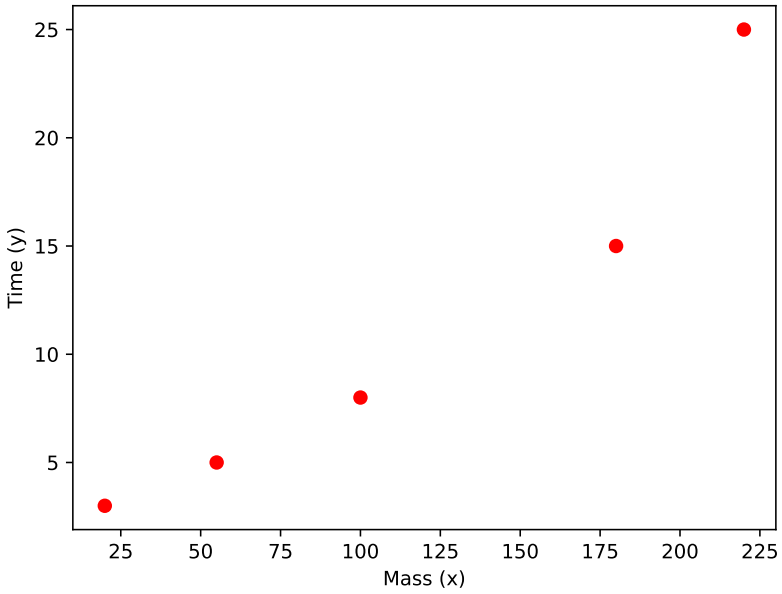

Figure 1: Training samples.

We want to train a linear regression model (using the OLS approach) that captures the relationship between our predictors (mass of a chemical) and our targets (time taken to react) based on this training set such that given the mass of some new chemicals unseen in the training set (i.e. given a validation set), we can predict their reaction times. We start with assuming our linear regression function can be represented as a simple degree = 1 polynomial (i.e. a straight line). If we want to capture the relationship between mass ($\mathbf{x}$) and time ($\mathbf{y}$) using a degree = 1 polynomial, then the unknown parameters (referred to as the intercept ω_0 and slope ω_1) we want to estimate for our regression function are those that give us the best possible approximation of a line that fits these training samples. As you can tell from Figure 1 these training samples are not collinear (i.e. are not perfectly aligned in a straight line). IF all the points were collinear then the intercept (ω_0) and slope (ω_1) that we estimate for our regression function would satisfy the following system of 5 linear equations for each of the 5 training samples:

$$y_1 = \omega_0 + \omega_1 x_1 \tag{1a}$$

$$y_2 = \omega_0 + \omega_1 x_2 \tag{1b}$$

$$y_3 = \omega_0 + \omega_1 x_3 \tag{1c}$$

$$y_4 = \omega_0 + \omega_1 x_4 \tag{1d}$$

$$y_5 = \omega_0 + \omega_1 x_5 \tag{1e}$$

Since the training points are not collinear what we actually have for some as yet unknown values of ω_0 and ω_1 are approximations/predictions of time, denoted $\hat{y}_i$, for each chemical's mass (or predictor) x_i. Meaning this can be expressed as a system of linear equations as well as:

$$\hat{y}_1 = \omega_0 + \omega_1 x_1 = [\omega_0 \;\; \omega_1] \times \begin{bmatrix} 1 \\ x_1 \end{bmatrix} \tag{2a}$$

$$\hat{y}_2 = \omega_0 + \omega_1 x_2 = [\omega_0 \;\; \omega_1] \times \begin{bmatrix} 1 \\ x_2 \end{bmatrix} \tag{2b}$$

$$\hat{y}_3 = \omega_0 + \omega_1 x_3 = [\omega_0 \;\; \omega_1] \times \begin{bmatrix} 1 \\ x_3 \end{bmatrix} \tag{2c}$$

$$\hat{y}_4 = \omega_0 + \omega_1 x_4 = [\omega_0 \;\; \omega_1] \times \begin{bmatrix} 1 \\ x_4 \end{bmatrix} \tag{2d}$$

$$\hat{y}_5 = \omega_0 + \omega_1 x_5 = [\omega_0 \;\; \omega_1] \times \begin{bmatrix} 1 \\ x_5 \end{bmatrix} \tag{2e}$$

This has been expressed on the right hand side in vectorial form as the product of two vectors, namely, a vector of the unknown parameters/weights $\boldsymbol{\omega}^T = [\omega_0, \omega_1]$ and a *basis function vector* denoted as $\boldsymbol{\phi} = \begin{bmatrix} \phi_0(x_i) \\ \phi_1(x_i) \end{bmatrix}$. (Note: We write transpose of $\boldsymbol{\omega}$ as we are using column orientation of vectors. Hence for row orientation you have a transpose of the vector.)

Here, notice that $\phi_0(x_i)$ is always a constant equal to 1 as it corresponds to the basis function for the intercept ω_0. Now based on equations 2a - 2e, notice that the left hand-side of all equations can be combined into a single vector $\hat{\mathbf{y}}^T = [\hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4, \hat{y}_5]$ (which represents the vector of predicted targets for all 5 training samples). Now, on the right-hand side, the weights vector $\boldsymbol{\omega}^T$ is the same across all samples but the basis function vectors are different. So to combine them in matrix-vector form, we need to multiply the weights vector $\boldsymbol{\omega}^T$ by a matrix that concatenates all of the basis function vectors for all 5 training samples. This matrix is referred to as the design matrix.

Exercise 1.1: Assemble the design matrix for these five training samples keeping the algebraic notation of x's, y's and ω's.

Exercise 1.2: Once the design matrix ($\mathbf{D}$) is assembled, multiply $\mathbf{D}$ by the weights vector $\boldsymbol{\omega}^T = [\omega_0, \omega_1]$ and show that this product results in the vector of predicted targets $\hat{\mathbf{y}}^T = [\hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4, \hat{y}_5]$.

Exercise 1.3: Next substitute the x's in the design matrix D with the values for the predictors (i.e. mass) for each training sample presented in Table 1 and write out the design matrix in numeric form.

Exercise 1.4: Now that we know how to express a linear regression function as either a system of linear equations or as a matrix-vector product, express how the equations in 2a-2e would change if you chose a degree = 2 polynomial as your linear regression function instead. And, write out the corresponding design matrix $\mathbf{D}$ first in algebraic form (as in Ex. 1.1) and then in numeric form (as in Ex. 1.3) using the values for the predictors shown in Table 1.

2 Estimating the weights/unknown parameters of Linear Regression model using OLS:

Let's consider again a simple linear regression model based on a degree = 1 polynomial. As discussed on page 1, the provided training samples are not collinear, meaning that no matter what values we estimate for regression weights ω_0, ω_1, the resulting line will not fit all the training points perfectly. In other words, we will always have *residual errors* between the model's predictions ($\hat{\mathbf{y}}$) and the real known targets ($\mathbf{y}$) of the training samples. So, in order to get the best linear regression model based on the provided training data, we want to estimate a set of regression weights that *minimise* these residual errors. Or put another way, we can get the OLS solution to the unknown weights of our linear regression model by *minimising* the sum of squared residuals error/cost function $\mathcal{R}(\boldsymbol{\omega})$ with respect to our unknown weights. Estimation of the weights can be done using iterative numerical optimisation techniques such as gradient descent, or solved analytically. The analytical approach leads to a solution wherein the best set of weights for the linear regression model is given by calculating the pseudoinverse of the design matrix ($\mathbf{D}$) and multiplying this with the vector of the real known targets ($\mathbf{y}$). This is derived by calculating the derivatives of the sum of squared residuals function with respect to the unknown weights and organising the resulting system of equations in matrix-vector form. You will not be asked to show this derivation in the exam.

Given that we know that the OLS solution to our unknown weights $\boldsymbol{\omega}$ is given as the matrix-vector product of $\mathbf{D}^\dagger$ and $\mathbf{y}$, where the symbol † represents the pseudoinverse, answer the following questions: (Note - depending on how you have assembled the design matrix, i.e. with the basis function vectors as columns or as rows, you may need to transpose $\mathbf{D}^\dagger$ before multiplying with $\mathbf{y}$).

Note: The questions below cannot be asked in the exam as you will not have access to a PC to compute the pseudoinverse. But they are important to answer to ensure you have understood the concepts covered correctly.

Exercise 2.1: Using the design matrix you assembled in Ex. 1.3 calculate the pseudoinverse of the matrix ($\mathbf{D}^\dagger$) using the numpy python function 'np.linalg.pinv' (np is just the abbreviation if you "import numpy as np" in your code). Next use the calculated $\mathbf{D}^\dagger$ to estimate the OLS solution for your regression weights $\boldsymbol{\omega}$ using the values provided for the targets ($\mathbf{y}$) in your training set (shown in Table 1). Report the estimated weights.

Exercise 2.2: Repeat the process from Ex. 2.1, this time for estimating the weights of a degree = 2 polynomial based on the corresponding design matrix you assembled in Ex. 1.4. Report the estimated weights.

Exercise 2.3: Now that you have estimated the regression weights for both your degree 1 and 2 polynomial linear regression functions based on the training set, make predictions using both regression models using the unseen validation set provided in Table 2. I.e. using the mass of the

new chemicals (predictors $\mathbf{x}_{val}$) make predictions for their reaction times. Compare the predictions made by both regression models and report which model performs best and comment on why you think one model performs better than the other. *Hint: Refer back to your solution to Ex. 1.2.*

Mass (x_{val})	15	40	120	200	240
Time (y_{val})	1	4	10	22	30

Table 2: Validation set.

Exercise 2.4: Using the regression weights estimated in Ex 2.1, predict target values for predictors ($\mathbf{x}$) sampled uniformly in the interval $[15, 250]$ (you can use the 'np.linspace' numpy function for this) and plot the resulting line using the sampled values for $\mathbf{x}$ and the target values predicted ($\hat{\mathbf{y}}$) by your linear regression model. And overlay the scatter plot of training sample points on the same plot (similar to how they are presented in Figure 1). *Hint: Refer to the linear regression notebook and to the use of the 'np.polyval' numpy function to make predictions for all uniformly sampled predictors.*

Exercise 2.5: Using the regression weights estimated in Ex 2.2, predict target values for predictors ($\mathbf{x}$) sampled uniformly in the interval $[15, 250]$ and plot the resulting curve using the sampled values for $\mathbf{x}$ and the target values predicted ($\hat{\mathbf{y}}$) by your linear regression model. And overlay the scatter plot of training sample points on the same plot (similar to how they are presented in Figure 1).