Linear Regression

Lab 3

Linear Regression

Regression

- Regression analysis is one of the most important fields in statistics and machine learning.
- There are many regression methods available. ***Linear regression*** is one of them.
- Regression searches for relationships between variables.

Linear Regression

- Linear regression is probably one of the most important and widely used regression techniques.

Linear Regression

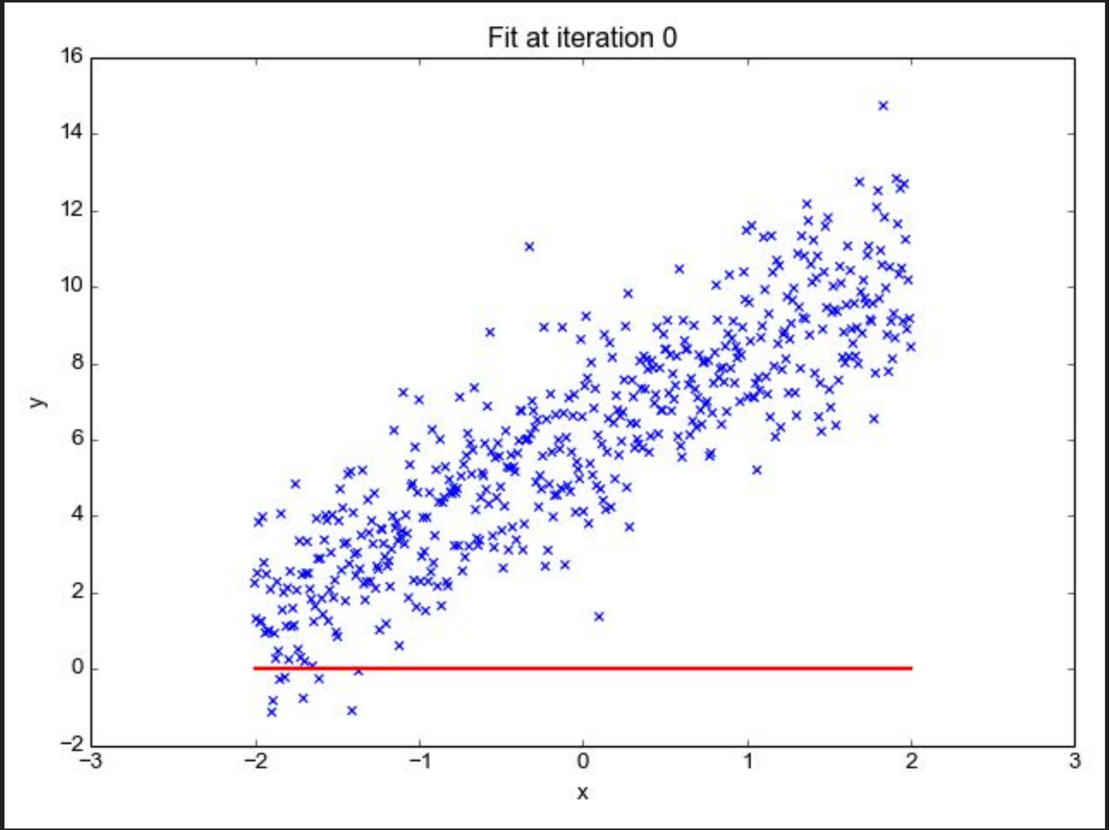
Problem Formulation

- When implementing linear regression of some dependent variable y on the set of independent variables $\mathbf{x} = (x_1, \ldots, x_r)$, where r is the number of predictors, you assume a linear relationship between y and $\mathbf{x}$:

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_r x_r + \varepsilon.$$

- This equation is the regression equation. $\beta_0, \beta_1, \ldots, \beta_r$ are the regression coefficients, and ε is the random error.

1D Linear Regression In Math!



1D Linear Regression In Math!

input : m data pairs, $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m)\}$ with $x_i \in \mathbb{R}$ and $y_i \in \mathbb{R}$. Here, y_i is the desired output for x_i.

output: w^* and b^*, the parameters of the linear model $\hat{y} = w^* x + b^*$

$$\bar{x} \leftarrow \frac{1}{m} \sum_{i=1}^{m} x_i;$$

$$\bar{y} \leftarrow \frac{1}{m} \sum_{i=1}^{m} y_i;$$

$$\hat{\sigma}_{xx}^2 \leftarrow \frac{1}{m} \sum_{i=1}^{m} (x_i - \bar{x})^2;$$

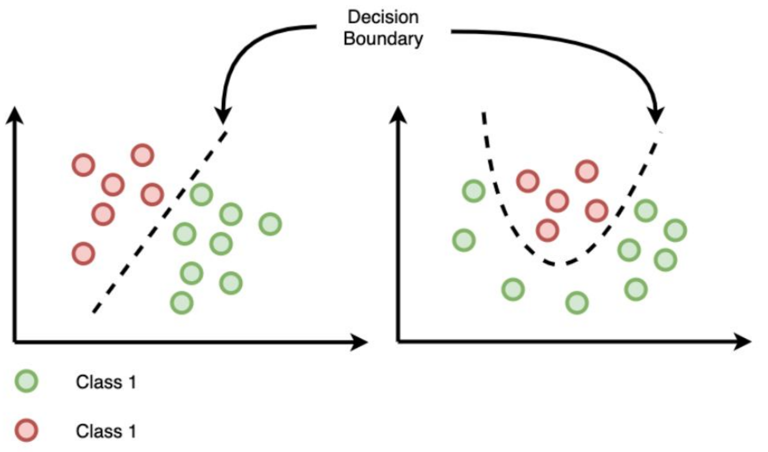
$$\hat{\sigma}_{xy}^2 \leftarrow \frac{1}{m} \sum_{i=1}^{m} (x_i - \bar{x})(y_i - \bar{y});$$

$$w^* \leftarrow \hat{\sigma}_{xy}^2 / \hat{\sigma}_{xx}^2;$$

$$b^* \leftarrow \bar{y} - \bar{x} w^*;$$

Linear Regression for Classification!

- In this lab we are using linear regression to determine a **decision boundary** for a binary classification problem.
- Although the baseline is to identify a binary decision boundary, the approach can be very well applied for scenarios with multiple classification classes or multi-class classification.



Decision Boundaries

Linear Regression for Classification - 1D case

- For a 1D input (and output) our model can be represented as:

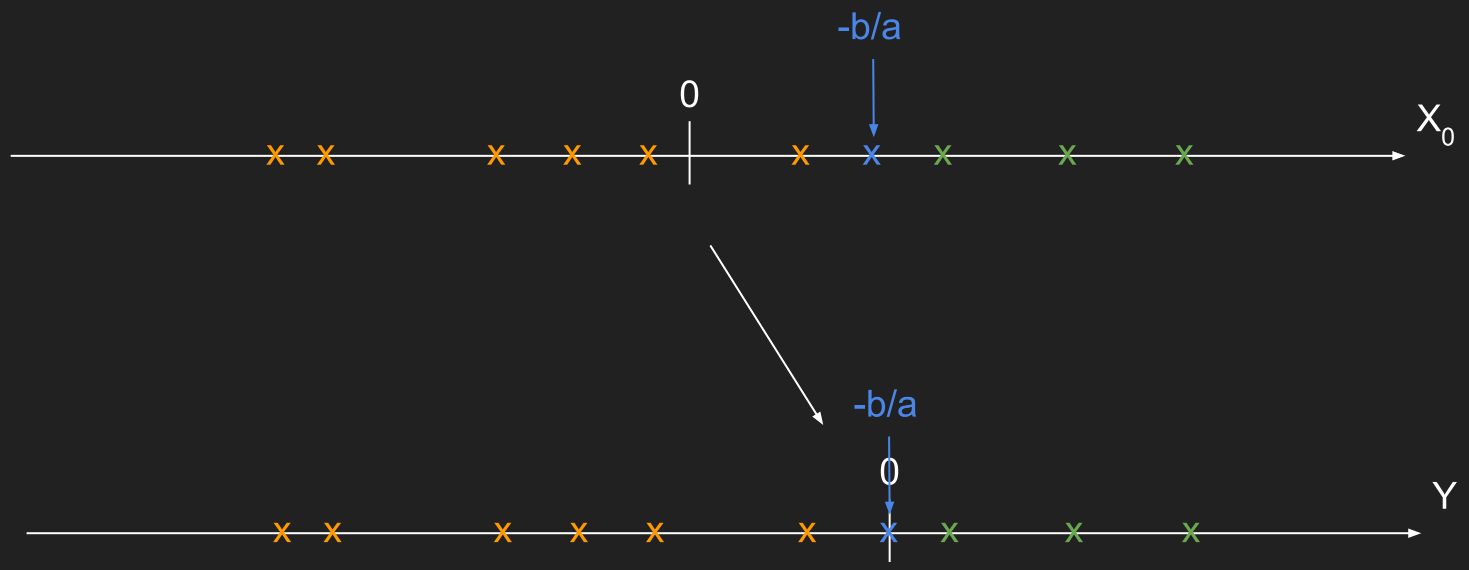
$$a\, x_0 + b = 0$$

- If we solve this for x we get:

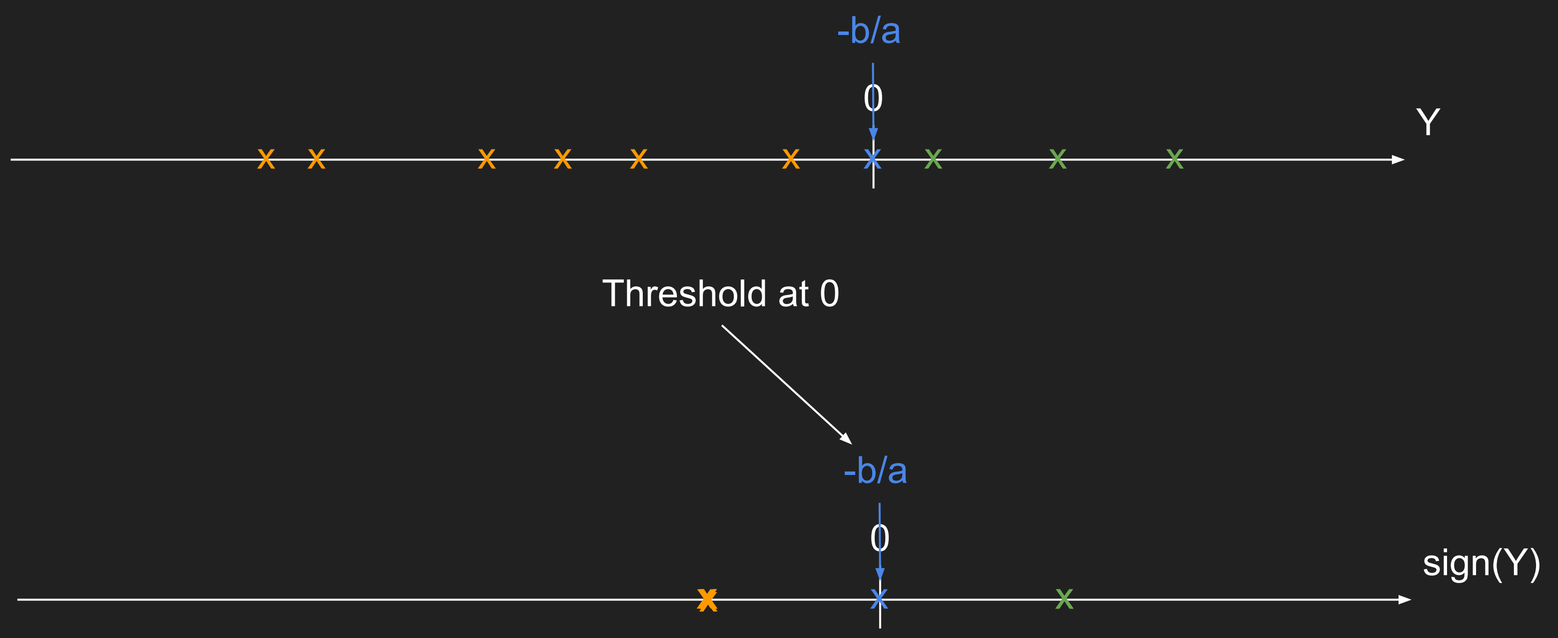
$$x_0 = -b/a$$

- This point will become the new “origin” after the transformation

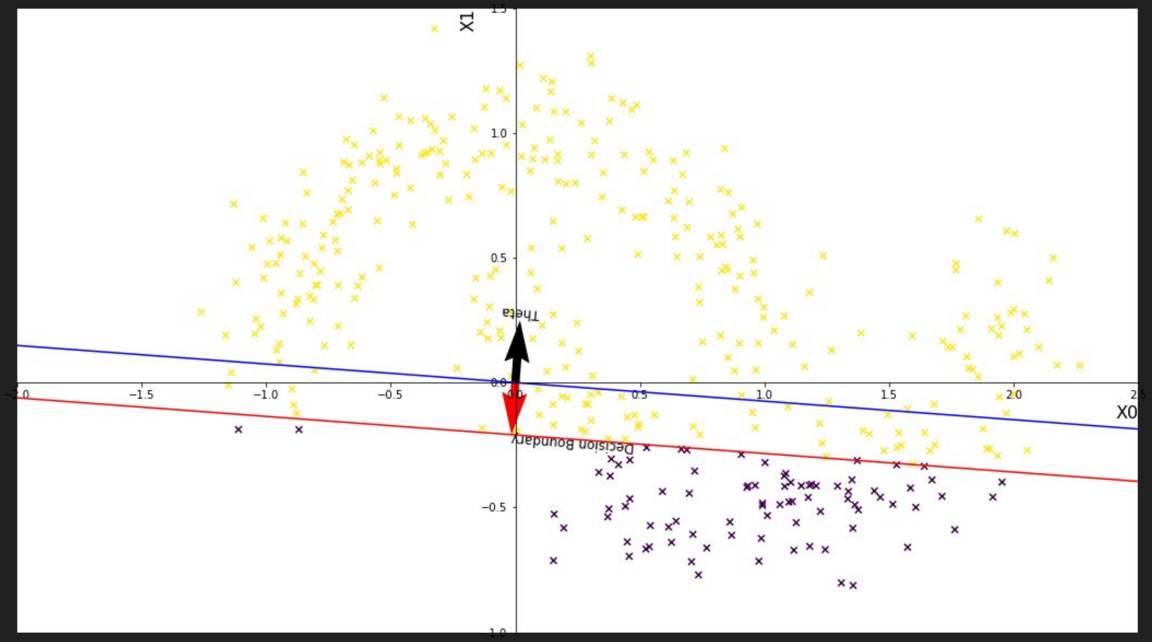
Linear Regression for Classification - 1D case



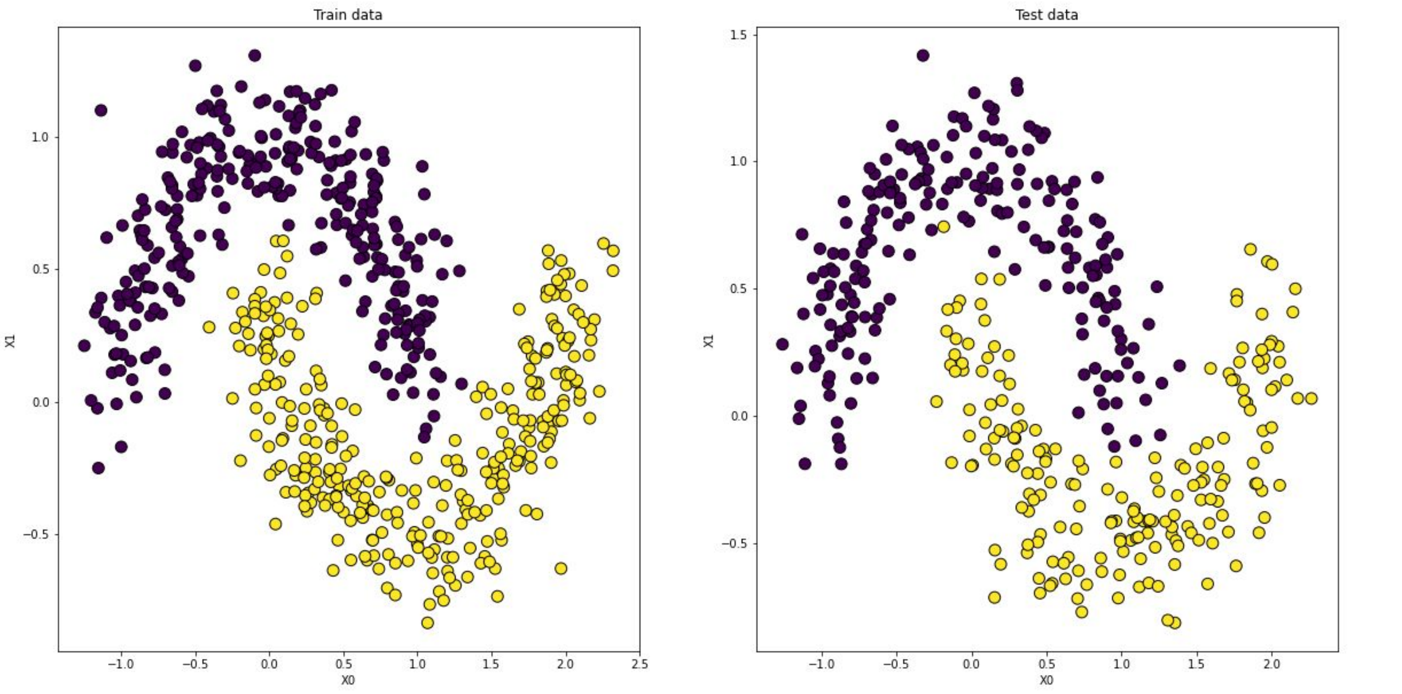
Linear Regression for Classification - 1D case



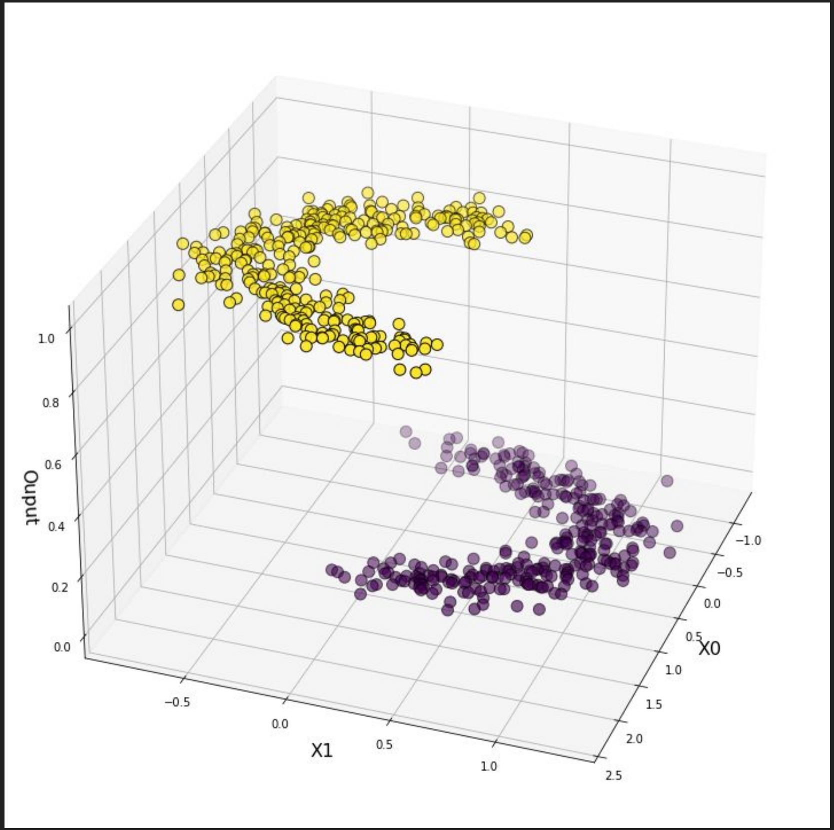
2D Linear Regression for Classification!



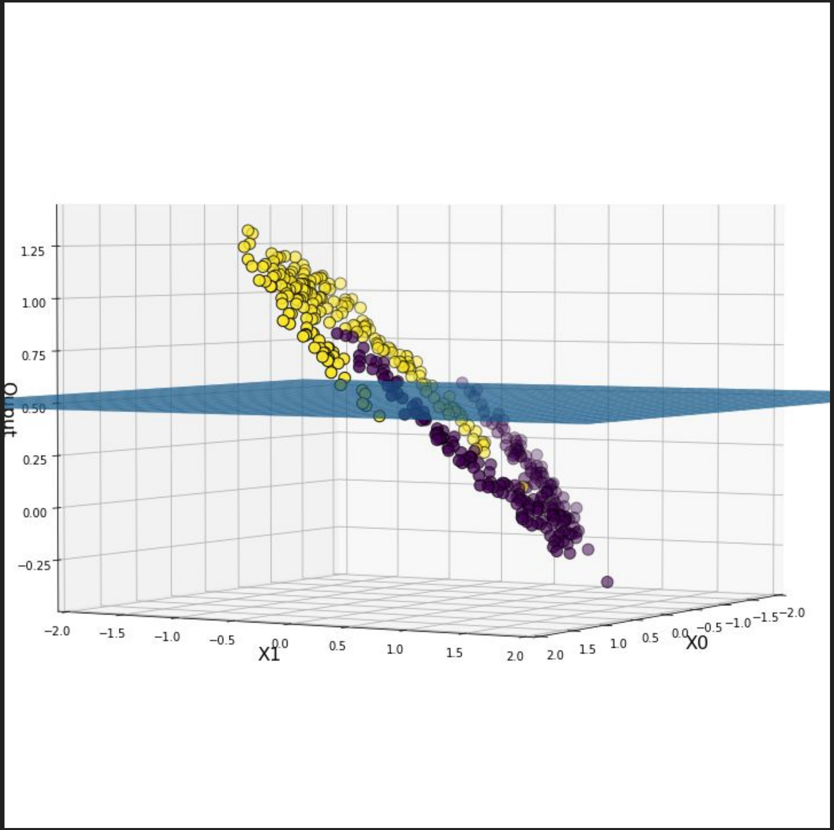
2D Linear Regression for Classification!



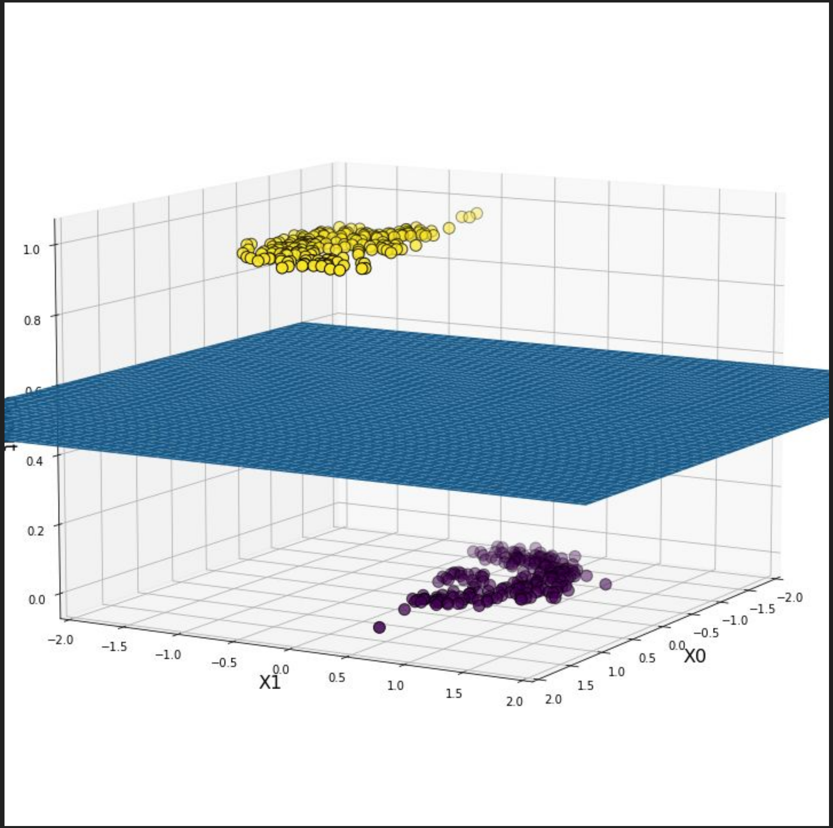
2D Linear Regression for Classification!



2D Linear Regression for Classification!



Linear Regression



Linear Regression model in closed form

- We recall that the parameters of a linear regression can be obtained in closed-form.
- The general linear expression is :

$$a\, x_1 + b\, x_2 + c = d_1$$

- For a dataset with n data points :

Datapoint 1 : $a\, x_1^{1} + b\, x_2^{1} + c = d_1$

Datapoint 2 : $a\, x_1^{2} + b\, x_2^{2} + c = d_2$

…………………..

Datapoint n : $a\, x_1^{n} + b\, x_2^{n} + c = d_n$

Linear Regression model in closed form

- We can write those general expressions in matrix form as (Assuming we have only 2 data points):

$$\begin{pmatrix} x_1^1 & x_2^1 & 1 \\ x_1^2 & x_2^2 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \end{pmatrix} = \begin{pmatrix} \mathbf{d_1} \\ \mathbf{d_2} \end{pmatrix}$$

- To find **a, b, c** values of general linear expressions, we use **pseudo inverse**.
  - Pseudo inverse of X (X^+) is $(X^T X)^{-1} X^T$
  - If X is MxN then X^+ is NxM

Linear Regression model in closed form

- The equation for our linear regression model takes the form:

$$\mathbf{X}\boldsymbol{\theta}^{\mathrm{T}} = \mathbf{Y}$$

- If $\boldsymbol{X} \in \mathbb{R}^{n \times m}$ denotes the matrix of input data (each column is a training sample) and $\boldsymbol{Y} \in \mathbb{R}^{m \times p}$ is the matrix of desired outputs, then the parameters of the model is obtained as:

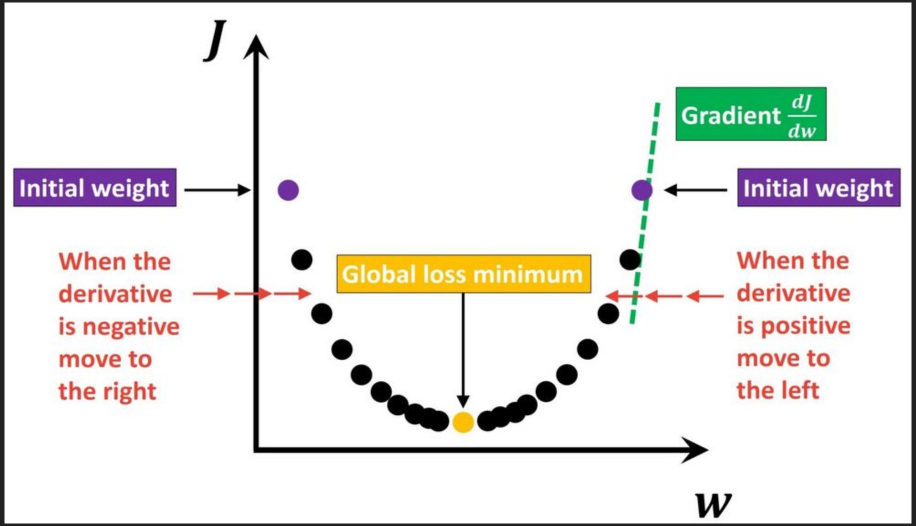
$$\boldsymbol{\theta}^{\mathrm{T}} = (\mathbf{X}^{\mathrm{T}}\,\mathbf{X})^{-1}\,\mathbf{X}^{\mathrm{T}}\,\mathbf{Y}$$

- Here $\boldsymbol{\theta}$ stands for the estimated parameter vector (shape MxP vector row by convention)

Linear Regression model with Gradient Descent

What is Gradient Descent ?

- Gradient descent is an optimization algorithm that's used when training a machine learning model.
- It's based on a convex function and tweaks its parameters iteratively to minimize a given function to its local minimum.



Linear Regression model with Gradient Descent

Loss/Cost Function

- Compute the loss with respect to the inputs and the parameters of the model.

$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m} \| \theta^\top \mathbf{x}_i - y_i \|^2$$

- Compute the gradient of the model with respect to its parameters θ.

$$\frac{\partial L}{\partial \theta} = \frac{1}{m}\sum_{i=1}^{m} 2(\theta^\top \mathbf{x}_i - y_i)\mathbf{x}_i$$

Linear Regression model with Gradient Descent

Learning Rate

- The learning rate is a hyperparameter that controls how much to change the model in response to the estimated loss each time the model weights are updated.
- Choosing the learning rate is challenging as a value too small may result in a long training process that could get stuck.
- Whereas a value too large may result in learning a sub-optimal set of weights too fast or an unstable training process.
- Typical learning rates for GD: 1 - 0.01
- Typical learning rates for SGD: 0.1 - 0.0001

Linear Regression model with Gradient Descent