**Linear Regression**

- Regression is **supervised learning**
- $y_i$ **-** winning time prediction
- $x_i$ **-** year of race
- **m -** rate of improvement
- **c -** winning time at year 0

**Overdetermined System -** When there are more equations than there are unknowns so you can't satisfy all of them

- When you have several points but can't fit a line through all of them

**Noise Model**

- Add an **error term ε** to each of the equations in the overdetermined system
  - $y_1 = mx_1 + c + \varepsilon_1$
  - $y_2 = mx_2 + c + \varepsilon_2$
  - $y_3 = mx_3 + c + \varepsilon_3$
- **Gauss** said that **ε** has to be modelled by a Gaussian distribution
- **Laplace** proved why that was a good idea using the **Central Limit Theorem**
  - Sum of Gaussian variables is also Gaussian
  - The noise can be modelled as a sum of Gaussian variables that we don't understand but are happening
  - The sum of those things is also going to be Gaussian

$$\sum_{i=1}^{n} y_i \sim \mathcal{N}\left(\sum_{i=1}^{n} \mu_i, \sum_{i=1}^{n} \sigma_i^2\right)$$

- **Gaussian** distribution of a **noise model** is modelled as **$\varepsilon \sim N(\mu, \sigma^2)$** **(Normal Probability Density)**
  - **~** - in maths means **approximately equal to**
  - **ε -** the noise of our model
  - **μ -** is replaced by our linear model (**mx + c**)
  - **$\sigma^2$ -** the noise that we use to represent our data?
    - To calculate the **covariance**, you can use $Cov(x,y) = \frac{\Sigma(x_i-\bar{x})(y_i-\bar{y})}{N}$

$$p(y|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)$$
$$\triangleq \mathcal{N}(y|\mu, \sigma^2)$$

  - **IID Assumption - Independent and Identically Distributed**
    - An assumption that a set of random variables are independent and identically distributed
      - They are statistically independent **and** each of them follows the same distribution

**Modelling a Linear Regression Model**

- You say that the **y's** can be described by a **Gaussian** distribution where the mean is given by the **function of your model** and the variance is given by the **variance of the noise assumed for the measurements**
  - **N(mx + c, ε)**
- This gives us a way to generate new observations by sampling from a Gaussian
- This means that the **mean** is a **stochastic function**
- **Regression Model**
  - $$y_i = f(x_i) + \epsilon_i$$ $$\epsilon_i \sim \mathcal{N}(0, \sigma^2).$$
  - **$f(x_i)$** is a **deterministic function**
  - By adding **$\varepsilon_i$** we are making **$y_i$** be a **stochastic function**
  - This means that **$y_i \sim N(f(x_i), \sigma^2)$**

**Data Point Likelihood**

- As **$y_i$** is modelled using a **Gaussian Distribution**, we can calculate the probability of a **$y_i$** given an **$x_i$** (determins.)

- $$p\left(y_i|x_i, m, c, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$
- **$\sigma^2$ -** noise variance
- This formula is called a **Likelihood Function**

- Usually for a distribution, we know the parameters for the distribution (mean and variance), and if you make a sum over all of the observations the answer will be 1.
- With **Likelihood functions**, we will have the **data (x's and y's)** but **not the parameters**
- The parameters are what we want to find to get a model
  - **m, c, $\sigma^2$** are **unknown parameters**
  - We try to find those parameters that best fit the likelihood function (the data)
  - This is done using an objective function
- We define two vectors (**x_** and **y_**)
- We use the IID assumption
  - **($y_1$, $x_1$)** and **($y_2$, $x_2$)** are independent and identically distributed
  - Therefore, the probability of **y_** can be written as a product of the probabilities of all **$y_i$**
    - $$p(\mathbf{y}) = \prod_{i=1}^{n} p(y_i)$$
      - Probability of a **joint distribution** can be calculated as the **product of the marginals**
    - $$p(\mathbf{y}|\mathbf{x}, m, c, \sigma^2) = \prod_{i=1}^{n} p(y_i|x_i, m, c, \sigma^2)$$
    - **Likelihood Function v**
      $$p(\mathbf{y}|\mathbf{x}, m, c, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - mx_i - c)^2}{2\sigma^2}\right)$$
      $$p(\mathbf{y}|\mathbf{x}, m, c, \sigma^2) = \frac{1}{\left(2\pi\sigma^2\right)^{\frac{n}{2}}} \exp\left(-\frac{\sum_{i=1}^{n} (y_i - mx_i - c)}{2\sigma^2}\right)$$
    - The bottom is the same as the above because we are simply doing the product of the left term n times and multiplying exponents is the same as the exponential of the sum of their arguments
      - exp(a) **x** exp(b) = exp(a **+** b)
- Given the data **y_** and **x_** we want to use the **likelihood function** to find the **m, c, $\sigma^2$** that maximise the probability of **y_** given the **x_**
- **Log Likelihood Function** (The Objective Function)

$$L(m, c, \sigma^2) = -\frac{n}{2}\log 2\pi - \frac{n}{2}\log\sigma^2 - \sum_{i=1}^{n}\frac{(y_i - mx_i - c)^2}{2\sigma^2}$$

- We prefer to use this over the likelihood function
  - It is a **monotonic function -** whatever you can say about the argument of the function, you can say about the entire function
    - *A **monotonic function** either nondecreasing or nonincreasing*
  - You don't have to deal with the exponential anymore, only with the argument of the exponential
  - Makes computation easier by converting products to sums which are easier to differentiate and keeps the numbers smaller (more precise)
- You can arrive at the **log likelihood function** using **log rules** (shown on right)

1: $\log_b(M \cdot N) = \log_b M + \log_b N$

2: $\log_b\left(\frac{M}{N}\right) = \log_b M - \log_b N$

3: $\log_b\left(M^k\right) = k \cdot \log_b M$

4: $\log_b(1) = 0$

5: $\log_b(b) = 1$

6: $\log_b\left(b^k\right) = k$

7: $b^{\log_b(k)} = k$

**Error Function**

- We use the **Negative Log Likelihood Function** as the **Objective Function**
  - We like to minimise the error function that's why it is the negative one
- We start with the objective function and we want to find an algorithm that will help us find the parameters of the model that maximise the **Log Likelihood (Objective) Function**

$$E(m, c, \sigma^2) = \frac{n}{2}\log\sigma^2 + \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - mx_i - c)^2$$

- We omit the log(2pi) term because it doesn't depend on the parameters
- The log likelihood function is similar in some ways to the sum of squares function
- **Error function** depends on **parameters** through the **prediction function**
  - **$E(m, c, \sigma^2)$ :** Error / Objective Function
  - **$f(x_i)$ :** Prediction function

**Learning is Optimisation**

- We minimise the cost function (through optimisation)
- **Finished learning when gradient = 0**
- **Coordinate Ascent -** Optimisation Algorithm
  - You can only move along the axis towards the minima (i.e. **x** or **y** axis)
  1. You randomly choose a **direction** to move in
  2. You then choose **how far** to go in that direction
  - This is repeated many times
  - **You need a model with several parameters and an objective function**
  - **You take the derivative of the objective function with respect to each parameter**
  - **You solve the optimisation problem for each of the variables one at a time**
  - E.g. solving m

**E(m, c) =** $\sum_{i=1}^{n}(y_i - mx_i - c)^2$

$$\frac{dE(m)}{dm} = -2\sum_{i=1}^{n}x_i(y_i - mx_i - c)$$

$$0 = -2\sum_{i=1}^{n}x_i(y_i - mx_i - c)$$

->

$$0 = -2\sum_{i=1}^{n}x_iy_i + 2\sum_{i=1}^{n}mx_i^2 + 2\sum_{i=1}^{n}cx_i$$

$$m = \frac{\sum_{i=1}^{n}(y_i - c)x_i}{\sum_{i=1}^{n}x_i^2}$$

- We use this expression to upgrade m
- To upgrade m we need to upgrade c

**E(m, c) =** $\sum_{i=1}^{n}(y_i - mx_i - c)^2$

$$\frac{dE(c)}{dc} = -2\sum_{i=1}^{n}(y_i - mx_i - c)$$

$$0 = -2\sum_{i=1}^{n}(y_i - mx_i - c)$$

->

$$0 = -2\sum_{i=1}^{n}y_i + 2\sum_{i=1}^{n}mx_i + 2nc$$

$$c = \frac{\sum_{i=1}^{n}(y_i - mx_i)}{n}$$

- But now c depends on m
- **Coordinate Ascent** **-** You update one parameter, and using the new value, update the other
  - And so on and so forth

- **Fixed Point Equations -** The equations used to calculate and therefore update the parameters

$$c^* = \frac{\sum_{i=1}^{n} (y_i - m^* x_i)}{n},$$

$$m^* = \frac{\sum_{i=1}^{n} x_i (y_i - c^*)}{\sum_{i=1}^{n} x_i^2},$$

$$\sigma^{2*} = \frac{\sum_{i=1}^{n} (y_i - m^* x_i - c^*)^2}{n}$$

  - When you satisfy **m** and **c**, you can compute $\boldsymbol{\sigma^2}$

**Recap**

- We started with a regression problem
- We use probabilistic interpretations of the regression problem via **noises** for an **overdetermined system**
- We can then assume that our observations follow a distribution
  - We now have a probabilistic interpretation of our data
- **Likelihood function** is a way to express an **objective function** which uses a data but doesn't know about the parameters
- We optimise the **Likelihood function** to update the parameters
- We can then use the model for prediction
- $\boldsymbol{\sigma^2}$ tells us how noisy is our data

**Multivariate Functions**

- **Linear Function -** $$f(\mathbf{x}_i) = \sum_{j=1}^{D} w_j x_{i,j} + c$$
- Let's assume that the input is **multi-dimensional**
- We need to change to **matrix (vector) notation** to deal with the cumbersomeness of multi-dimensional input
- $f(\mathbf{x}_i) = \mathbf{w}^\top \mathbf{x}_i + c$
- **This can be converted to** $f(\mathbf{x}_i) = \mathbf{w}^\top \mathbf{x}_i$
    - The **c** gets absorbed into **w** by adding an extra term $\mathbf{x_0}$ which is always **1**
- $\mathbf{a^T b}$ **=** inner product of the two matrices (sum of all of their products)
- By default, **b** and any other non-transposed matrix is a **column vector**
- We can now calculate the other functions using the **matrix form**
- **Likelihood of a single point**

$$p\left(y_i | x_i, \mathbf{w}, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(y_i - \mathbf{w}^\top \mathbf{x}_i\right)^2}{2\sigma^2}\right)$$

- **Log likelihood for the data set**

$$L(\mathbf{w}, \sigma^2) = -\frac{n}{2}\log\sigma^2 - \frac{n}{2}\log 2\pi - \frac{\sum_{i=1}^{n}\left(y_i - \mathbf{w}^\top \mathbf{x}_i\right)^2}{2\sigma^2}$$

- **Error Function**

$$E(\mathbf{w}, \sigma^2) = \frac{n}{2}\log\sigma^2 + \frac{\sum_{i=1}^{n}\left(y_i - \mathbf{w}^\top \mathbf{x}_i\right)^2}{2\sigma^2}$$
$$= \frac{n}{2}\log\sigma^2 + \frac{1}{2\sigma^2}\sum_{i=1}^{n} y_i^2 - \frac{1}{\sigma^2}\mathbf{w}^\top \sum_{i=1}^{n} \mathbf{x}_i y_i +$$

$$E(\mathbf{w}, \sigma^2) = \frac{n}{2}\log\sigma^2 + \frac{\sum_{i=1}^{n}\left(y_i - \mathbf{w}^\top \mathbf{x}_i\right)^2}{2\sigma^2}.$$
$$= \frac{n}{2}\log\sigma^2 + \frac{1}{2\sigma^2}\sum_{i=1}^{n} y_i^2 - \frac{1}{\sigma^2}\sum_{i=1}^{n} y_i \mathbf{w}^\top \mathbf{x}_i +$$
$$= \frac{n}{2}\log\sigma^2 + \frac{1}{2\sigma^2}\sum_{i=1}^{n} y_i^2 - \frac{1}{\sigma^2}\mathbf{w}^\top \sum_{i=1}^{n} \mathbf{x}_i y_i +$$

**Multivariate Derivatives**

- $\frac{d\mathbf{w}^\top \mathbf{a}}{d\mathbf{w}} = \mathbf{a}$ $\quad\quad$ $\frac{d\mathbf{w}^\top \mathbf{A}\mathbf{w}}{d\mathbf{w}} = \left(\mathbf{A} + \mathbf{A}^\top\right)\mathbf{w}$ $\quad\quad$ if $\mathbf{A}$ is symmetric (*i.e.* $\mathbf{A} = \mathbf{A}^\top$) $\frac{d\mathbf{w}^\top \mathbf{A}\mathbf{w}}{d\mathbf{w}} = 2\mathbf{A}\mathbf{w}.$
- So we have the error function with the expanded brackets

  $$E(\mathbf{w},\ \sigma^2) = \frac{n}{2}\log\sigma^2 + \frac{1}{2\sigma^2}\sum_{i=1}^{n} y_i^2 - \frac{1}{\sigma^2}\mathbf{w}^\top\sum_{i=1}^{n}\mathbf{x}_i y_i + \frac{1}{2\sigma^2}\mathbf{w}^\top\left[\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^\top\right]\mathbf{w}.$$
- We can differentiate with respect to **w**
  - $$\frac{\partial E\left(\mathbf{w}, \sigma^2\right)}{\partial \mathbf{w}} = -\frac{1}{\sigma^2}\sum_{i=1}^{n}\mathbf{x}_i y_i + \frac{1}{\sigma^2}\left[\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^\top\right]\mathbf{w}$$
- This leads us to the **fixed point update functions**
  - $$\mathbf{w}^* = \left[\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^\top\right]^{-1}\sum_{i=1}^{n}\mathbf{x}_i y_i$$
- Using **matrix notation**, we can convert

  $$\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^\top = \mathbf{X}^\top\mathbf{X} \quad\quad \sum_{i=1}^{n}\mathbf{x}_i y_i = \mathbf{X}^\top\mathbf{y}$$
- This gives us
  - $$\mathbf{w}^* = \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{y}$$
  - This equation is called the **normal equation**
  - The dimensionality of **w** is going to be the size of the input $\mathbf{x_i}$ + 1 (for the **c intercept**)
- We can also find the equation for $\boldsymbol{\sigma^2}$
  - $$\sigma^{2*} = \frac{\sum_{i=1}^{n}\left(y_i - \mathbf{w}^{*\top}\mathbf{x}_i\right)^2}{n}$$

**Summary**

- You put the **x vectors** from the dataset into matrix **X_**
- You put your **y** values into a vector **y_**
- You get an expression for **w*** and you get an estimation of that
- To make new predictions, make an inner product between **x_** and **w** $\mathbf{x\_}^\top\mathbf{w}$