NEURAL NETWORKS, DEEP LEARNING AND BIO-INSPIRED COMPUTING

MATHS FOR DEEP LEARNING

Nathan Elazar

LECTURE OUTLINE

- Data Structures
- Linear Algebra
- Vector Calculus
- Probability and Distributions
- Machine Learning

DATA STRUCTURES

- Used to represent a collection or group of objects.
- Different types support different operations.

TYPES OF DATA STRUCTURE

- Set
  - An unordered collection of objects, denoted with {}.
  - e.g. $S = \{a, b, c\}$
  - Can be queried for membership with ∈, e.g. $a \in S = True$

- List (or Tuple)
  - An ordered collection of objects, denoted with [] (or ())
  - e.g. $L = [a, b, c], L = (a, b, c)$
  - Can be indexed to select a specific element, e.g. $L_1 = a$

TYPES OF DATA STRUCTURE

- Function
  - A function maps element from one set A to another set B, denoted as $f : A \rightarrow B$.
  - A function is a set of lists of length 2, where the first element of the list is an element of A and the second is an element of B.
  - A function can be applied to an element of A to get the corresponding element of B, e.g. f(a)=b

COMMONLY USED SETS

- Naturals
  - Denoted with $\mathbb{N}$
  - Set that contains all non-negative whole numbers
  - $\mathbb{N} = \{0, 1, 2, \dots\}$

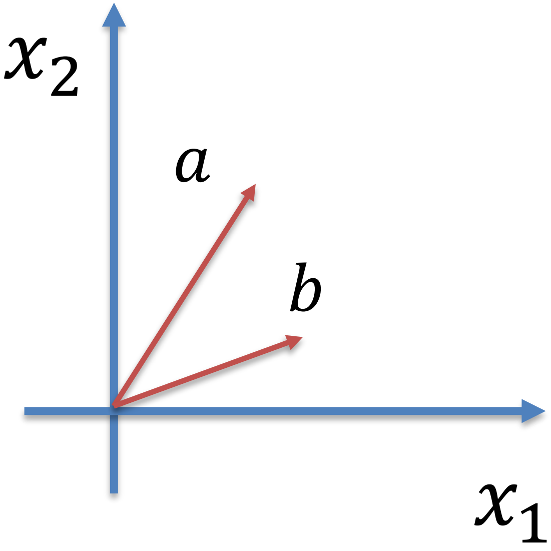
- Reals
  - Denoted with $\mathbb{R}$
  - Set that contains all continuous numbers
  - $\mathbb{R} = \{0, 0.3, \sqrt{2}, -\pi, \dots\}$

- Lists of $n \in \mathbb{N}$ real numbers
  - Denoted with $\mathbb{R}^n$
  - Set that contains all lists of length n whose elements are all real numbers
  - $\mathbb{R}^2 = \{[0.2, 0.4], [2, 1],\ [\pi, e], \dots\}$

LINEAR ALGEBRA

- A vector space is a set equipped with scaling and addition operations.
  - Elements of the set are called vectors.
  - Elements that can be scalar multiplied are scalars.

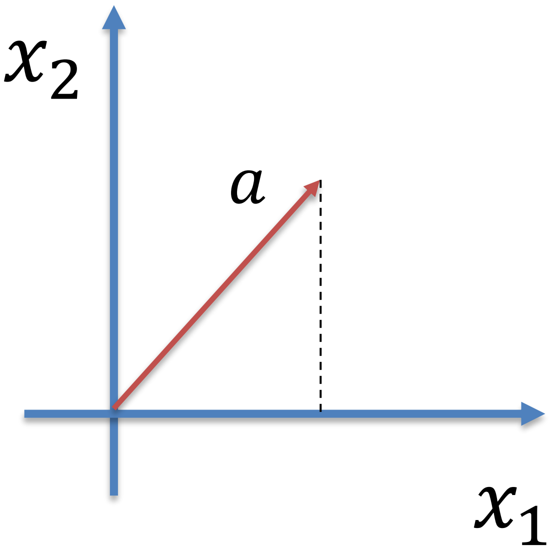
- We will be using $\mathbb{R}^n$ equipped with component-wise scaling and component-wise addition
  - $x, y \in \mathbb{R}^n \Rightarrow (x + y) \in \mathbb{R}^n, (x + y)_i = x_i + y_i$
  - $x \in \mathbb{R}^n, c \in \mathbb{R} \Rightarrow (cx) \in \mathbb{R}^n, (cx)_i = cx_i$



NORMS

- Norms measure the length of a vector
  - Roughly, how far away from the origin is it.
  - A norm is a function $f : \mathbb{R}^n \to \mathbb{R}$.
  - The returned number is always non-negative, smaller number means closer to the origin.

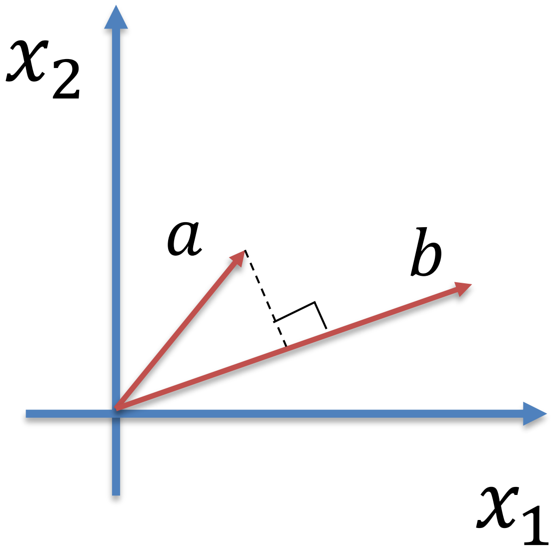
- L_1 (or Manhattan) Norm
  - $||x||_1 = \sum_{i=1}^{n} |x_i|$
- L_2 (or Euclidean) Norm
  - $||x||_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$
- Distance is norm of difference
  - $dist(x, y) = ||x - y||$



INNER PRODUCTS

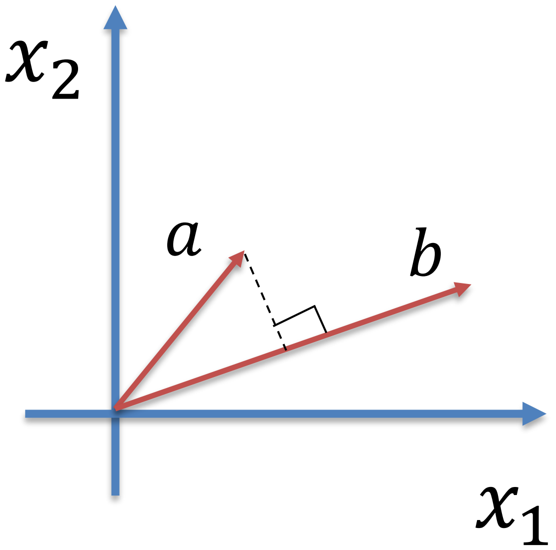
- Inner products measure the similarity between two vectors
  - An inner product is a function f : $\mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$
  - The larger the returned value, the more similar the two input vectors are.

- Standard inner product (dot product)
  - $\langle x, y \rangle = \sum_{i=1}^{n} x_i \, y_i$



ANGLES

- Inner products are used to measure angles
  - $dot\_angle(x, y) = \frac{\langle x,y \rangle}{||x|| \times ||y||} = \left\langle \frac{x}{||x||}, \frac{y}{||y||} \right\rangle$
  - Applying $\cos^{-1}$ converts the result into radians.



MATRICES

- An $m \times n$ matrix is a list of m vectors of length n.
  - Set of $m \times n$ matrices is denoted $\mathbb{R}^{m \times n}$
  - e.g. $\mathrm{A} = \begin{bmatrix} 0.2 & -\pi & 1.2 \\ -2.1 & -0.8 & 0.6 \end{bmatrix} \in \mathbb{R}^{2 \times 3}$
  - Matrices require two indexes to locate a value, the first specifies the row and the second specifies the column, e.g. $A_{1,2} = -\pi$
  - If we want to index an entire row or column we use the symbol :, e.g. $A_{:,1} = \begin{bmatrix} 0.2 \\ -2.1 \end{bmatrix}$

MATRIX MULTIPLICATION

- Given a matrix $A \in \mathbb{R}^{m\times k}$ and a matrix $B \in \mathbb{R}^{k\times n}$, then the product $\mathrm{C} = AB$ exists and $C \in \mathbb{R}^{m\times n}$ with $C_{i,j} = \langle A_{i,:}, B_{:,j}\rangle$
  - e.g. $\begin{bmatrix} 3 & -1 & 0 \\ 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 2 & 0 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} -2 & -3 \\ -3 & 0 \end{bmatrix}$
  - Set of $m \times n$ matrices is denoted $\mathbb{R}^{m\times n}$

MATRIX TRANSPOSE

- The transpose of a matrix $A \in \mathbb{R}^{n \times m}$ is a matrix $A^\top \in \mathbb{R}^{m \times n}$ with $A^\top_{i,j} = A_{j,i}$.
- e.g. $\begin{bmatrix} 1 & 2 \\ 3 & 3 \end{bmatrix}^\top = \begin{bmatrix} 1 & 3 \\ 2 & 3 \end{bmatrix}$

TENSORS

- Generalization of scalar, vector, matrix to higher dimensions.
- A d-dimensional tensor is an array of scalars that is indexed by d numbers.
  - e.g. 1-d tensor is a vector, 3-d tensor is a cube of numbers, etc.

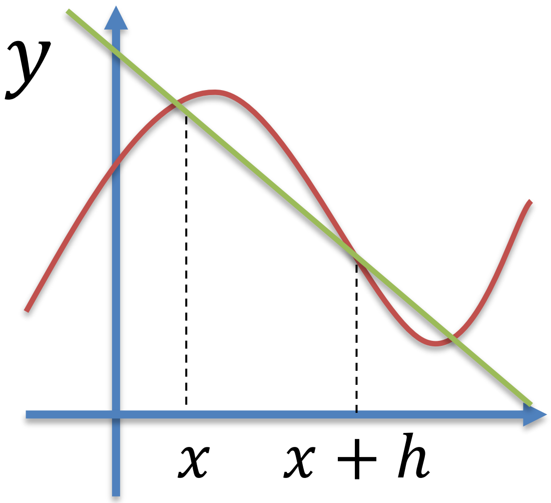
TENSOR OPERATIONS

- Concatenation
  - Stack two tensors together.
  - e.g. $concatenate(\begin{bmatrix}1\\3\end{bmatrix}, \begin{bmatrix}4\\-1\end{bmatrix}) = \begin{bmatrix}1\\3\\4\\-1\end{bmatrix}$
- Reshape
  - Change the shape of a tensor.
  - e.g. $reshape\left(\begin{bmatrix}1\\3\\4\\-1\end{bmatrix}, [2, 2]\right) = \begin{bmatrix}1 & 3\\4 & -1\end{bmatrix}$

DIFFERENTIATION

- The derivative of a function $f : \mathbb{R} \rightarrow \mathbb{R}$ is a function $\frac{df}{dx} : \mathbb{R} \rightarrow \mathbb{R}$
  - $\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h)-f(x)}{h}$

- The derivative evaluated at x gives the slope of the line tangent to f at x.



VECTOR GRADIENTS

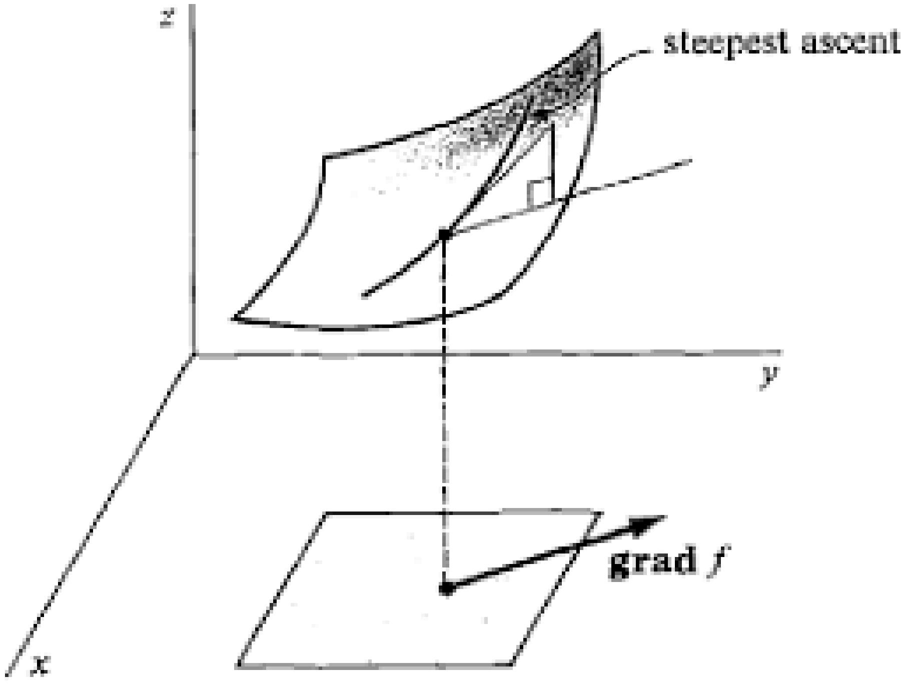
- The gradient of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a function $\frac{df}{dx} : \mathbb{R}^n \rightarrow \mathbb{R}^{1\times n}$. Also denoted ∇f.

- $\frac{df}{dx} = [\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}] \in \mathbb{R}^{1\times n}$

- $\frac{\partial f}{\partial x_i} = \lim_{h\rightarrow 0} \frac{f(x_1, x_2, \ldots, x_i + h, \ldots, x_n) - f(x)}{h}$

VECTOR GRADIENTS

- The gradient evaluated at a point x gives a vector which points in the direction of steepest ascent. The length of the vector gives the rate of change.



THE JACOBIAN

- The Jacobian of a function $f : \mathbb{R}^n \to \mathbb{R}^m$ is a function $\frac{df}{dx} : \mathbb{R}^n \to \mathbb{R}^{m \times n}$
- $\frac{df}{dx} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$

RULES FOR DIFFERENTIATION

Product rule

$$\frac{d}{dx}\big(f(x)g(x)\big) = \frac{df}{dx}g(x) + f(x)\frac{dg}{dx}$$

Sum rule

$$\frac{d}{dx}\big(f(x) + g(x)\big) = \frac{df}{dx} + \frac{dg}{dx}$$

Chain rule

$$\frac{d}{dx}f(g(x)) = \frac{df}{dg}\frac{dg}{dx}$$

USEFUL IDENTITIES

- For $W \in \mathbb{R}^{m\times n}, x \in \mathbb{R}^n$

$$\frac{d}{dx}Wx = W$$

$$\frac{d}{dW}Wx = \left.\begin{bmatrix} x^\top \\ \vdots \\ x^\top \end{bmatrix}\right\} m\ rows$$

- For a, b, x $\in \mathbb{R}$,

$$\frac{d}{dx}ax^b = abx^{b-1}$$

PROBABILITY DISTRIBUTIONS

- A probability distribution over a set Ω is a function $p: \Omega \to \mathbb{R}$ such that
  - $p(x) \geq 0$
  - $\sum_{x \in \Omega} p(x) = 1$ if Ω is discrete or else $\int_{x \in \Omega} p(x) dx = 1$

- We denote a value randomly sampled from a distribution as $X \sim p$. This means that X is a random variable – the chance that it has any particular value is given by $P(X = x) = p(x)$.

JOINT DISTRIBUTIONS

- Often it is useful to define a joint distribution over multiple variables, $P(X = x, Y = y)$.
- The joint distribution defines how likely a pair of values (x, y) is to be sampled.
  - From which it is possible to work out conditional probabilities $P(x|y)$, which measures how likely a value x is given that the value of y is known.

RULES FOR JOINT DISTRIBUTIONS

- Sum rule

$$P(x) = \begin{cases} \sum_{y} P(x,y) & if\ y\ is\ discrete \\ \int_{y} P(x,y)dy & if\ y\ is\ continuous \end{cases}$$

- Product rule

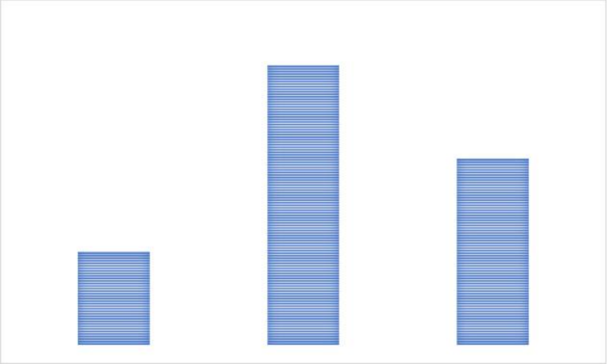
$$P(x,y) = P(x|y)P(y) = P(y|x)P(x)$$

- Bayes rule

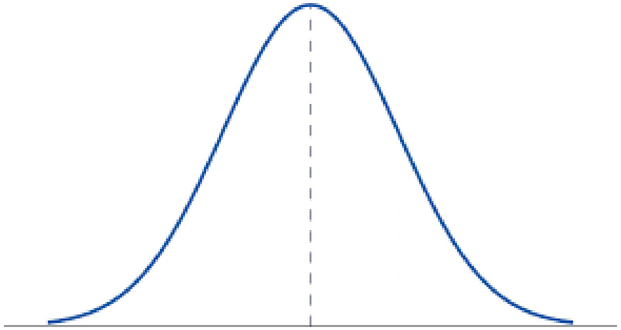
$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

COMMON DISTRIBUTIONS

- Categorical
  - A distribution over a list K of length M
  - Parameterised by a vector $k \in \mathbb{R}^M$.
  - $P\big(K_i \sim Cat(k)\big) = k_i$



- Normal
  - A distribution over $\mathbb{R}$.
  - Parameterised by a mean μ and a variance σ^2.
  - $P\big(x \sim \mathbb{N}(\mu, \sigma^2)\big) = \frac{1}{\sigma\sqrt{(2\pi)}} e^{-\left(\frac{x-\mu}{\sigma}\right)^2}$



EXPECTATIONS

- Expectation measures the average outcome when sampling from a distribution many times.
- $\mathbb{E}_{x\sim p} f(x) = \begin{cases} \sum_x f(x)P(x \sim p) & if\ x\ is\ discrete \\ \int_x f(x)P(x \sim p)dx & if\ x\ is\ continuous \end{cases}$

EMPIRICAL ESTIMATION

- Given a set of samples $\{x_i\}_{i=1}^{N}$ drawn from a (unknown) normal distribution, the mean and variance can be estimated as
  - $\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$
  - $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$

NORMALIZATION

- Normal distributions have the special property that $P(\mathrm{X} \sim \mathbb{N}(\mu, \sigma^2) = x) = P(X \sim \mathbb{N}(0,1) = \frac{x-\mu}{\sigma})$.

- This means that samples from any normal distribution can be converted into samples from the standard normal distribution simply by subtracting the mean and dividing by the standard deviation.

CORRELATION

- Given two sets of samples drawn from two normal distributions $x_i \sim \mathbb{N}(\mu_1, \sigma_1^2), y_i \sim \mathbb{N}(\mu_2, \sigma_2^2)$, their correlation is given by

$$\left\langle \frac{x-\mu_1}{\sigma_1}, \frac{y-\mu_2}{\sigma_2} \right\rangle = \sum_i \frac{(x_i-\mu_1)(y_i-\mu_2)}{\sqrt{\sum_i (x_i-\mu_1)^2} \sqrt{\sum_i (y_i-\mu_2)^2}}$$

- Measures how similar the distributions are – ignoring their scale and location.

SUPERVISED MACHINE LEARNING

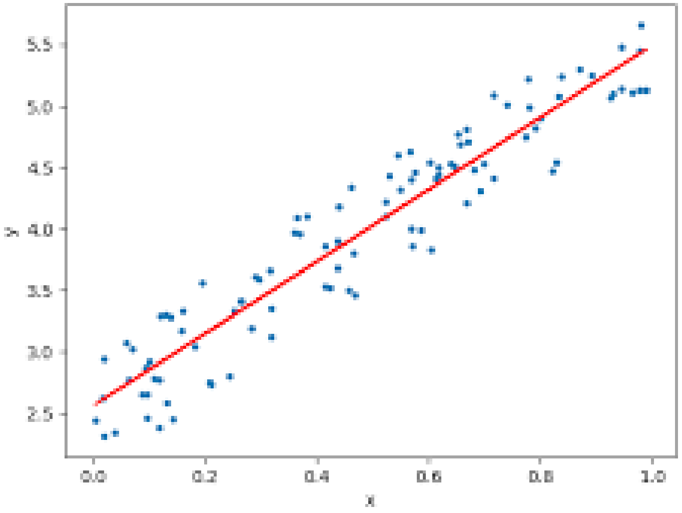
- Given a dataset $D = \{(x_i, t_i)\}_{i=1}^{N}$ where
  - $x_i \in \mathbb{R}^n, t_i \in \mathbb{R}^m$
  - $(x_i, t_i) \sim p$ (unknown)

- And a loss function $L: \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$
  - Measures how close predictions are to targets, lower is better.
  - e.g. $L(f(x), t) = ||f(x) - t||_2$

- Aim to find a function (model) $f: \mathbb{R}^n \to \mathbb{R}^m$ which minimises $\mathbb{E}_{(x,t)\sim p} L(f(x), t)$
  - This function should make predictions which are close to the true targets, even for points not in the training set.

LINEAR REGRESSION

- Search over linear models only.
  - $f(x) = Wx + b$,
    with $W \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^{m}$.
  - Specifying a parametric form of the model makes it easy to search for good models (e.g. with gradient descent).



GRADIENT DESCENT

- $L(W) = \sum_{i=1}^{N} \frac{1}{2} ||Wx_i - t_i||_2$

- $\frac{dL(W)}{dW} = \sum_{i=1}^{N} (Wx_i - t_i)\, x^\top$

LOSS FUNCTIONS AND DISTRIBUTIONS

- For commonly used loss functions, minimizing the loss is equivalent to maximising the probability of the data given some distribution.

- This means your choice of loss function will implicitly enforce assumption about the data
  - Always beware of what assumptions you are making!

MEAN SQUARED ERROR AND NORMAL DISTRIBUTIONS

Assume that data points are independent of each other and sampled from a normal distribution parameterised by our model $P(t|x) = \mathbb{N}(f(x), \sigma^2)$, then

$$\operatorname*{argmax}_f P(D) = \operatorname*{argmax}_f \prod_{i=1}^{N} P(t_i|x_i)$$

$$= \operatorname*{argmax}_f \prod_{i=1}^{N} \frac{1}{\sigma\sqrt{(2\pi)}} e^{-\left(\frac{||f(x_i)-t_i||_2}{\sigma}\right)^2}$$

$$= \operatorname*{argmin}_f \sum_{i=1}^{N} ||f(x_i) - t_i||_2$$

CLASSIFICATION

- What if we wish to assign a label from a discrete list K to each data point, instead of a continuous value?
- Represent target labels as one-hot vectors
  - $K_1 = [1,0, \dots, 0]$
  - $K_2 = [0,1, \dots, 0]$
  - $\dots$
  - $K_m = [0,0, \dots, 1]$

CROSS-ENTROPY LOSS

Assume that the label is sampled from a categorical distribution parameterised by our model $P(\mathrm{t} \mid x) = Cat(f(x))$, then

$$\begin{aligned}\underset{f}{\operatorname{argmax}}\, P(D) &= \underset{f}{\operatorname{argmax}} \prod_{i=1}^{N} P(t_i|x_i) \\ &= \underset{f}{\operatorname{argmax}} \prod_{i=1}^{N} \prod_{j=1}^{M} f(x_i)_j^{t_{i,j}} \\ &= \underset{f}{\operatorname{argmin}} \sum_{i=1}^{N} -\log(f(x_i)_j)\end{aligned}$$

Note that this step only works if our model outputs a categorical distribution

Usually we minimize this for classification

CLASSIFICATION

- Our model needs to output a categorical distribution over labels
  - Outputs a vector of length m.
  - Each component is non-negative, and they sum to one.
    - e.g. apply sigmoid or softmax to model output.

- We then assign the label with the highest probability under our model as the predicted class.

EVALUATION

- In order to know how well a model is going to perform on new data, we need to evaluate it on previously unseen data.
  - Usually, split the dataset into two parts, one is used exclusively for testing, and the other for training.

- There are many evaluation criteria to consider
  - Loss
  - Accuracy
  - F1-score
  - AUC
  - Computation time
  - etc.

CONFUSION MATRIX

		True Class	
		1	0
Predicted Class	1	23 (TP)	48 (FP)
	0	12 (FN)	166 (TN)

- True Positive (TP): The model predicted positive and was correct.
- True Negative (TP): The model predicted negative and was correct.
- False Positive (TP): The model predicted positive and was incorrect.
- False Negative (TP): The model predicted negative and was incorrect.

- Depending on your goals, some of these mistakes are worse than others!