Machine Learning

Cristopher G. S. Freitas

Who am I?

Cristopher Gabriel de Sousa Freitas

Computer Engineer grad @ UFAL, Oct, 2018.
Master's Student @ UFAL since Nov, 2018,
Computational Modeling of Knowledge.

Research Interests:
Networks, Information Theory and Machine Learning.

Main publication:
Freitas, C.G.S., Aquino, A.L.L., Ramos, H.S., Frery, A.C., Rosso, O.A..
A detailed characterization of complex networks using Information Theory.
Sci Rep **9,** 16689 (2019) doi:10.1038/s41598-019-53167-5

Other publications:
Santos E.G., Freitas C.G.S., Aquino, A.L.L..
A Deep Learning approach for Energy Disaggregation considering Embedded Devices. SBESC 2019.

Machine Learning

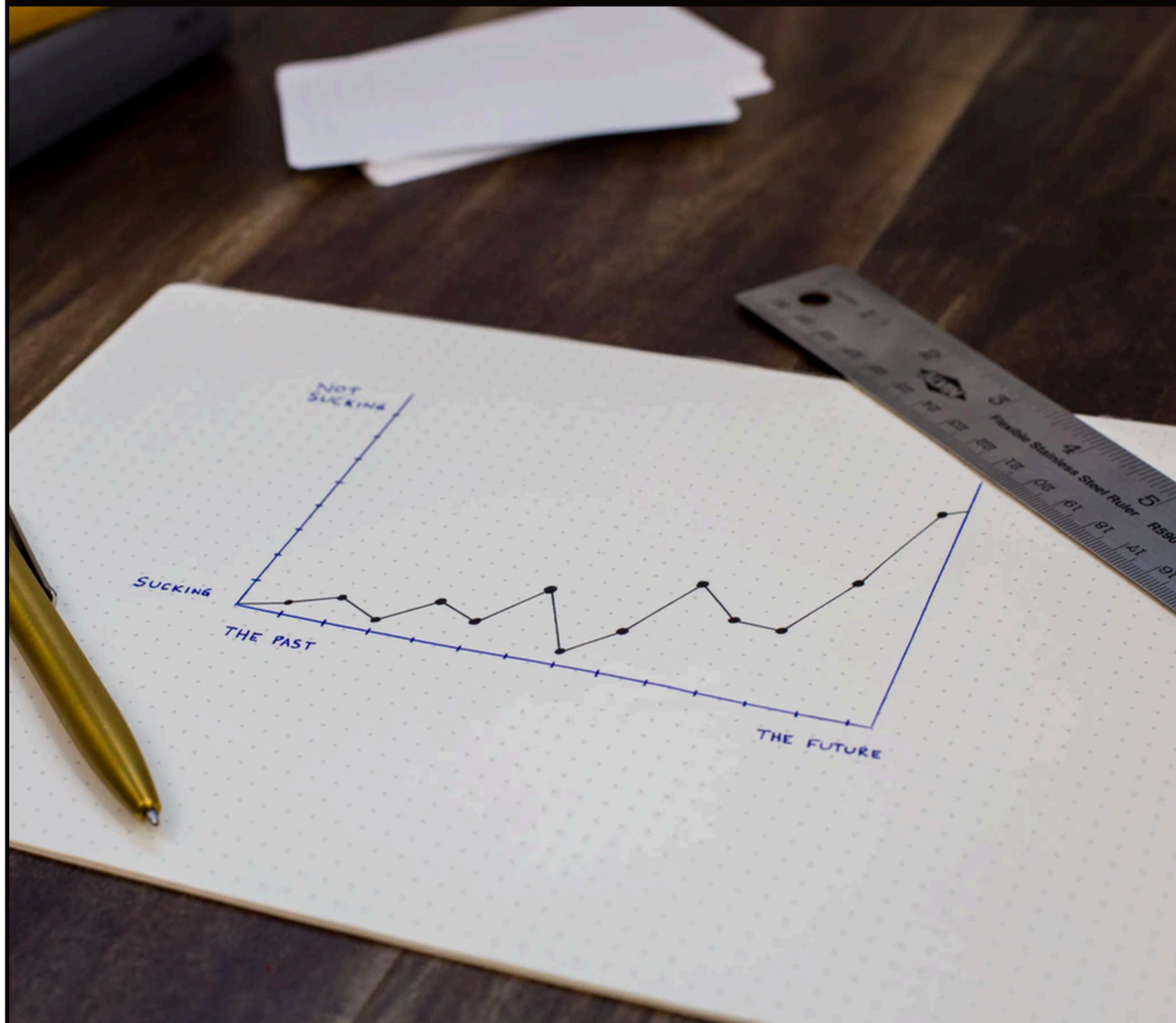
- What ML is about?
  - It is about inferring knowledge from *observations* or *experiences*, subject to the physical laws of the world.
  - It is also about using this knowledge to guide observation and **hypothesis testing**.
  - ML is about creating new knowledge, using the present knowledge, to solve a large diversity of novel problems.
- Important to distinguish:
  - Associative knowledge
  - Explanatory knowledge

First picture ever of a black hole unveiled.

This slide was inspired in the presentation of Prof. Nando De Freitas at Khipu, 2019

Systems and Models

- What is a ***system***?
  - A system is a group of interacting or interrelated entities that form a unified whole.
  - A system is delineated by its spatial and temporal boundaries, surrounded and influenced by its environment, described by its structure and purpose and expressed in its functioning.
- What is a ***model***?
  - It can be defined as a representation of a system using **general rules** and **concepts**.



A linear model is one of the simplest ML models we know.

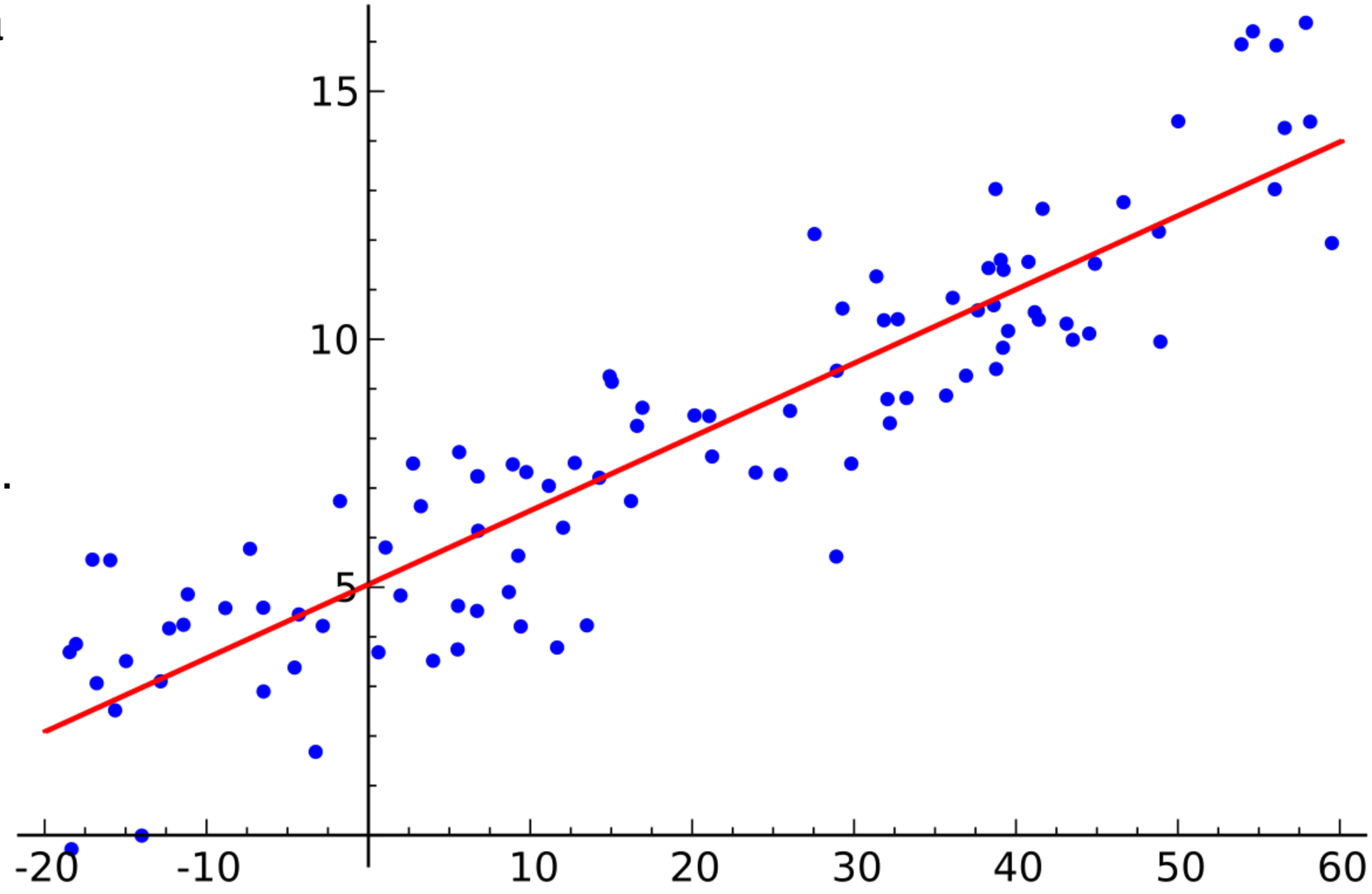
We represent this model through a linear equation:

$$y = ax + b$$

Or, more generally:

$$\mathbf{Y} = \mathbf{XB} + \mathbf{U}$$

We can model a system in several ways.



Learning

- So long as our machine learning models «cheat» by relying only on superficial statistical regularities, they remain vulnerable to out-of-distribution examples.
- Humans generalize better than other animals thanks to a more accurate internal model of the **underlying causal relationships**.
- To predict future situations (e.g., the effect of planned actions) far from anything seen before while involving known concepts, an essential component of reasoning, intelligence and science

This slide was inspired in the presentation of Prof.Yoshua Bengio at Khipu, 2019

Linear Algebra

A brief introduction

Scalars: A scalar is just a single number. Usually, scalars take a lowercase variable names. (e.g. Let $s \in \mathbb{R}$ be the slope of the line.)

Vectors: A vector is an array of numbers. Typically, we give vectors lowercase names in bold typeface, such as $\boldsymbol{x}$. The first element of $\boldsymbol{x}$ is x_1, the second element is x_2, and so on. We can think of vectors as identifying points in space, with each element giving the coordinate along a different axis. If $\boldsymbol{x} \in \mathbb{R}^n$, we consider

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Matrices: A matrix is a 2-D array of numbers, so each element is identified by two indices instead of just one. We usually give matrices uppercase variable names with bold typeface, such as $\boldsymbol{A}$. If a real-valued matrix $\boldsymbol{A}$ has a height of m and a width of n, then we say that $\boldsymbol{A} \in \mathbb{R}^{m \times n}$.

We usually identify the elements of a matrix using its name in *italic but not bold font*, and the indices are listed with separating commas. For example, $A_{1,1}$ is the upper left entry of $\boldsymbol{A}$ and $A_{m,n}$ is the bottom right entry.

$$\boldsymbol{A} = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix}$$

Tensor: In some cases we will need an array with more than two axes. We denote a tensor as $\mathbf{A}$, and we identify its elements by writing $A_{i,j,k}$.

Important operations on matrices:

Transpose: The transpose of a matrix is the mirror image of the matrix across a diagonal line, called the **main diagonal**. We denote the transpose of a matrix $\boldsymbol{A}$ as $\boldsymbol{A}^\top$, and it is defined such that

$$(\boldsymbol{A}^\top)_{i,j} = A_{j,i}$$

Addition:

We can add matrices to each other, as long as they have the same shape: $\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{B}$.

We can also add a scalar to a matrix or multiply a matrix by a scalar, just by performing that operation on each element of a matrix: $\boldsymbol{D} = a \cdot \boldsymbol{B} + c$.

In the context of deep learning, we allow the addition of a matrix and a vector, yielding another matrix $\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{b},$ where $C_{i,j} = A_{i,j} + b_j$. This implicit copying of $\boldsymbol{b}$ to many locations is called **broadcasting**.

Important operations on matrices:

Matrix product: $\boldsymbol{C = AB}$.

For this product to be defined, $\boldsymbol{A}$ must have the same number of columns as $\boldsymbol{B}$ has rows.

Element-wise product: $\boldsymbol{A \odot B}$ is the product of the individual elements.

Dot product…

Identity…

Inverse…

Important operations on matrices:

“Many mathematical objects can be understood better by breaking into constituent part, or finding some properties of them that are universal, not caused by the way we choose to represent them.”

For examle, integers can be decomposed into prime factors.

Matrices can be decomposed as **eigenvectors** and **eigenvalues**.

An eigenvector of a square matrix $\boldsymbol{A}$ is a nonzero vector $\mathbf{v}$ such that multiplication by $\boldsymbol{A}$ alters only the scale of $\boldsymbol{v}$**:**

$$\boldsymbol{A}\boldsymbol{v} = \lambda\boldsymbol{v}$$

The scalar λ is known as the eigenvalue corresponding to this eigenvector $\boldsymbol{v}$.

One simple algorithm can be derived using only the basic knowledge of Linear Algebra: **Principal Components Analysis (PCA)**.

Suppose we have a collection of m points $\{\boldsymbol{x}^{(i)}, \ldots, \boldsymbol{x}^{(n)}\}$ in $\mathbb{R}^n$ and we want to apply **lossy** compression to these points.

Lossy compression means we want to store these points with less memory, allowing to lose some precision.

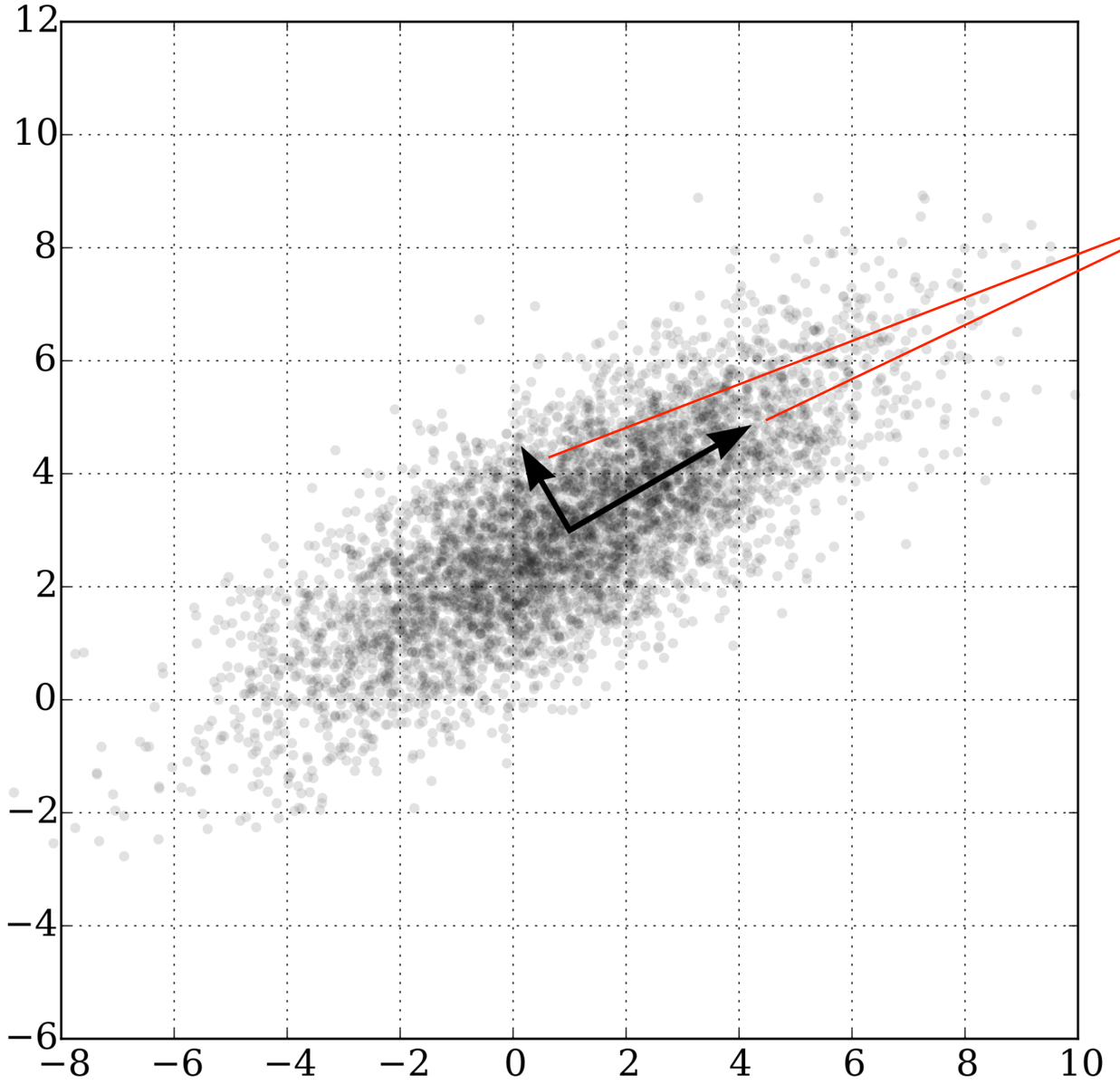
One way we can encode these points is to represent a lower-dimensional version of them:

For each point $\boldsymbol{x}^{(i)} \in \mathbb{R}^n$, we will find a corresponding code vector $\boldsymbol{c}^{(i)} \in \mathbb{R}^l$. If l is smaller than n, storing the code points will take less memory than storing the original data.

We will want to find some encoding function that produces the code for an input, $f(\boldsymbol{x}) = \boldsymbol{c}$, and a decoding function that produces the reconstructed input given its code, $\boldsymbol{x} \approx g(f(\boldsymbol{x}))$.

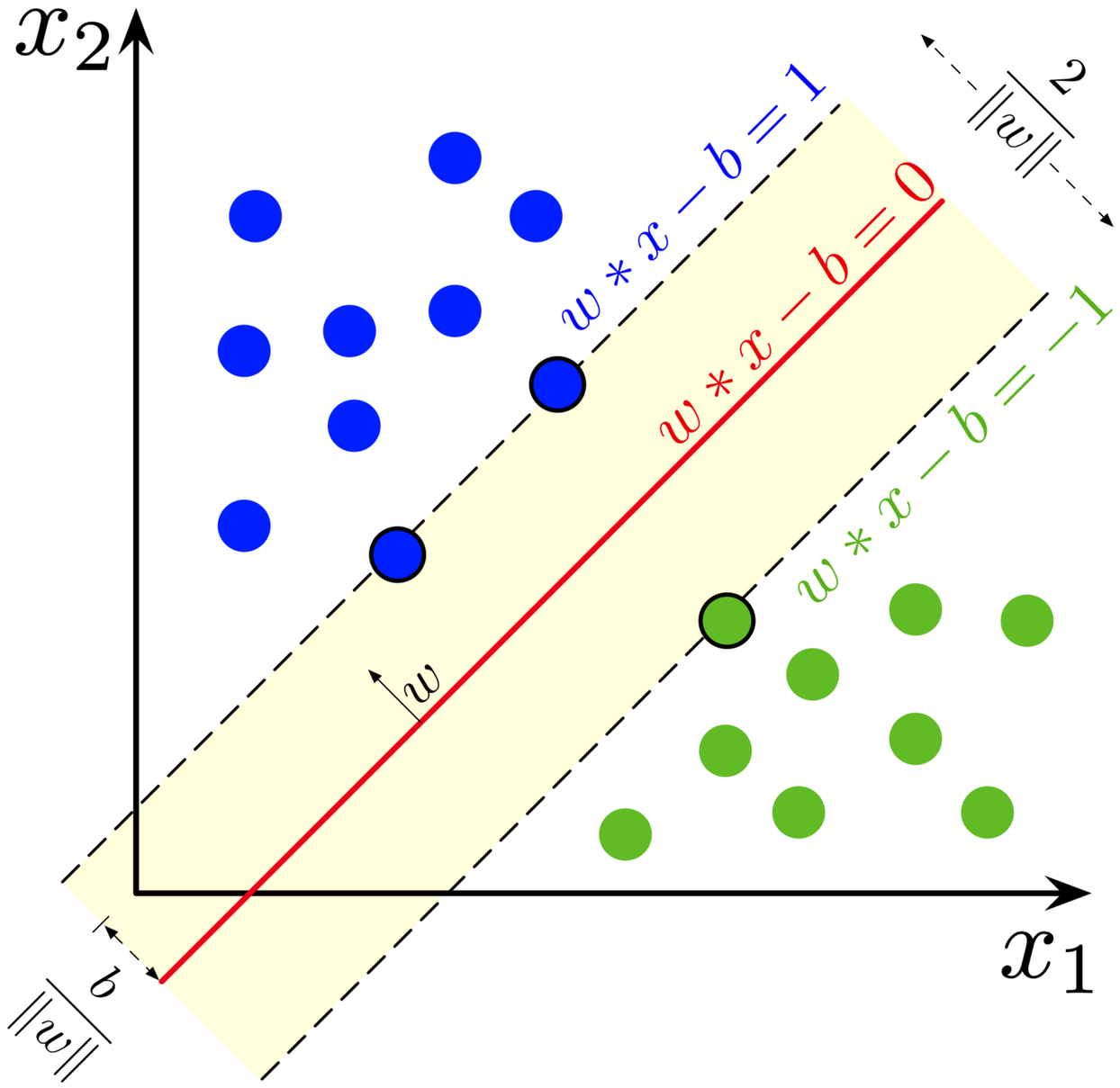
PCA is defined by our choice of the decoding function.

PCA finds the eigenvectors with highest eigenvalues.
In other words, it finds the components that explains the largest variance of the data.



These are eigenvectors!

Another simple ML technique that relies only on Linear Algebra is **SVM (Support Vector Machine)**.



Probability and Information Theory

Reviewing concepts

Probability Theory is a mathematical framework for representing uncertain statements.

A **random variable** is a variable that can take on different values randomly.
We write a random variable as $\mathbf{x}$, and one of its values as $\boldsymbol{x}$.

On its own, a random variable is just a description of the possible states of the underlying system.

A **probability distribution** is a description of how likely a random variable or set of random variables is to take on each of its possible states.

We may describe a probability distribution over discrete variables as a **probability mass function** (PMF). We typically denote probability mass functions with a capital P.

When working with continuous random variables, we describe a probability distribution using a **probability density function** (PDF) rather than a PMF. We denote a PDF as a function p.

Information Theory is a branch of applied mathematics that revolves around quantifying how much information is present in a signal.

The basic intuition behind information theory is that learning that an unlikely event has occurred is more informative than learning that a likely event has occurred.

Likely events have **low information content**, and in the extreme case, events that are guaranteed to happen should have no information content whatsoever.

Less likely events should have **higher information content**.

Independents events should have additive information. For example, finding out that a tossed coin has come up as heads twice should convey twice as much information as finding out that a tossed coin has come up as heads once.

To satisfy all three of these properties, we defined the **self-information** of an event $\mathbf{x} = \boldsymbol{x}$ to be

$$I(\boldsymbol{x}) = -\log P(\boldsymbol{x}) .$$

We can quantify the amount of uncertainty in an entire probability distribution using the **Shannon Entropy,**

$$H(\mathbf{x}) = -\mathbb{E}_{\mathbf{x}\sim P}[I(\boldsymbol{x})] = -\sum_i P(\boldsymbol{x_i}) \log P(\boldsymbol{x_i})$$

We can also denote as $H(P)$. When $\mathbf{x}$ is continuous, the Shannon Entropy is known as **differential entropy**.

If we have two separate probability distribution $P(\boldsymbol{x})$ and $Q(\boldsymbol{x})$ over the same random variable $\mathbf{x}$, we can measure how different these two distributions are using the *Kullback-Leibler (KL)* divergence:

$$D_{KL}(P \parallel Q) = \mathbb{E}_{\mathbf{x}\sim P}\left[log\frac{P(\boldsymbol{x})}{Q(\boldsymbol{x})}\right] = \mathbb{E}_{\mathbf{x}\sim P}[\log P(\boldsymbol{x}) - \log Q(\boldsymbol{x})]$$

The KL divergence is 0 if and only if $P(\boldsymbol{x})$ and $Q(\boldsymbol{x})$ are the same distribution for discrete variables.

A KL divergence is not a true distance measure because is not symmetric:
$D_{KL}(P \parallel Q) \neq D_{KL}(Q \parallel P)$ for some P and Q.

A quantity that is closely related to the KL divergence is the **cross-entropy**

$$H(P, Q) = H(P) + D_{KL}(P \parallel Q) = -\mathbb{E}_{\mathbf{x} \sim P} \log Q(\boldsymbol{x})$$

Available Projects

1. Titanic: Machine Learning from Disaster

2. Convolutional Networks

3. Text Classification

4. Reinforcement Learning

Requirements:

1. ***Solve the problem***
2. Create a standard presentation with:

Introduction - Methods - Results and Discussion

Max duration: 5 mins