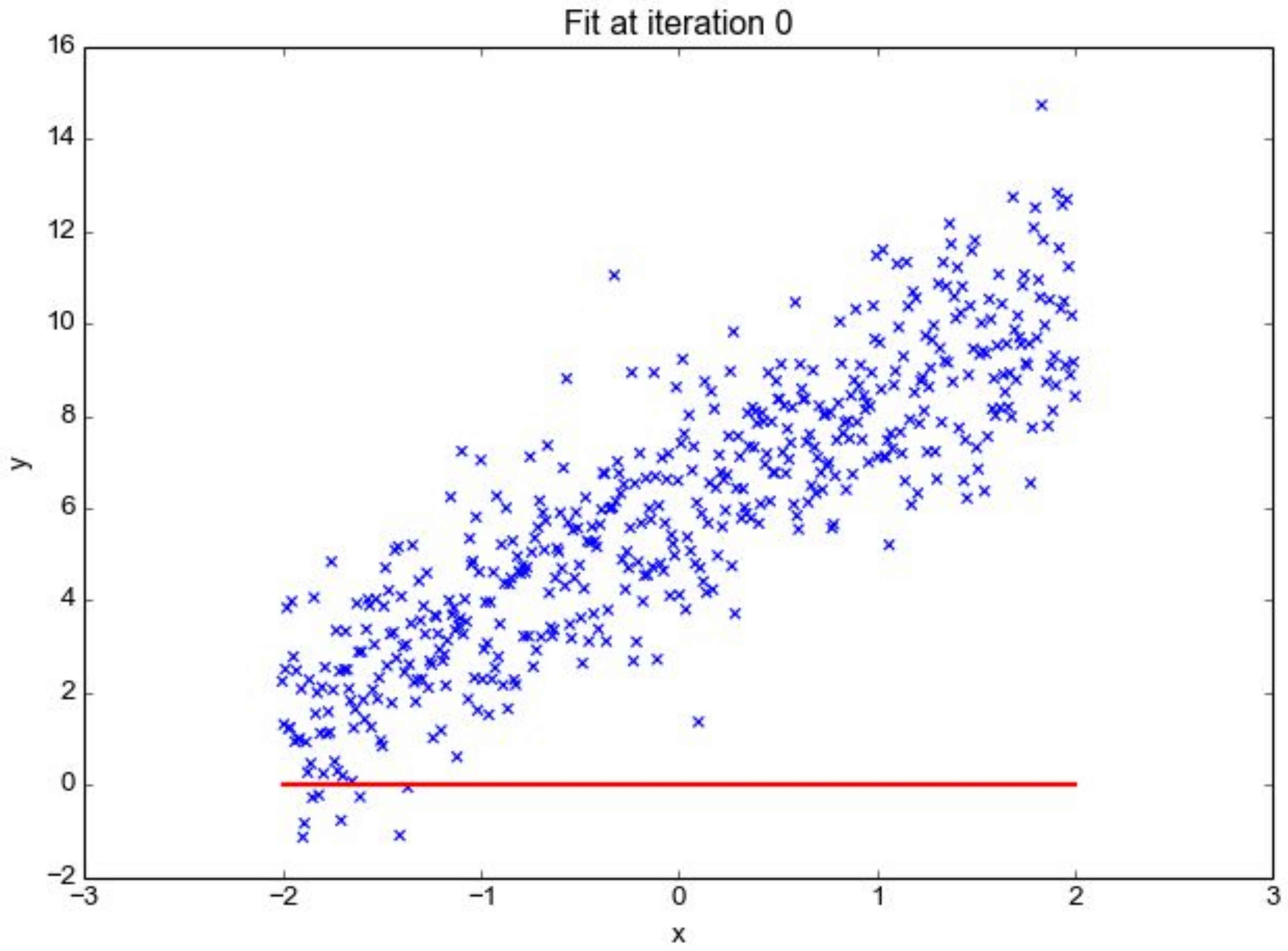
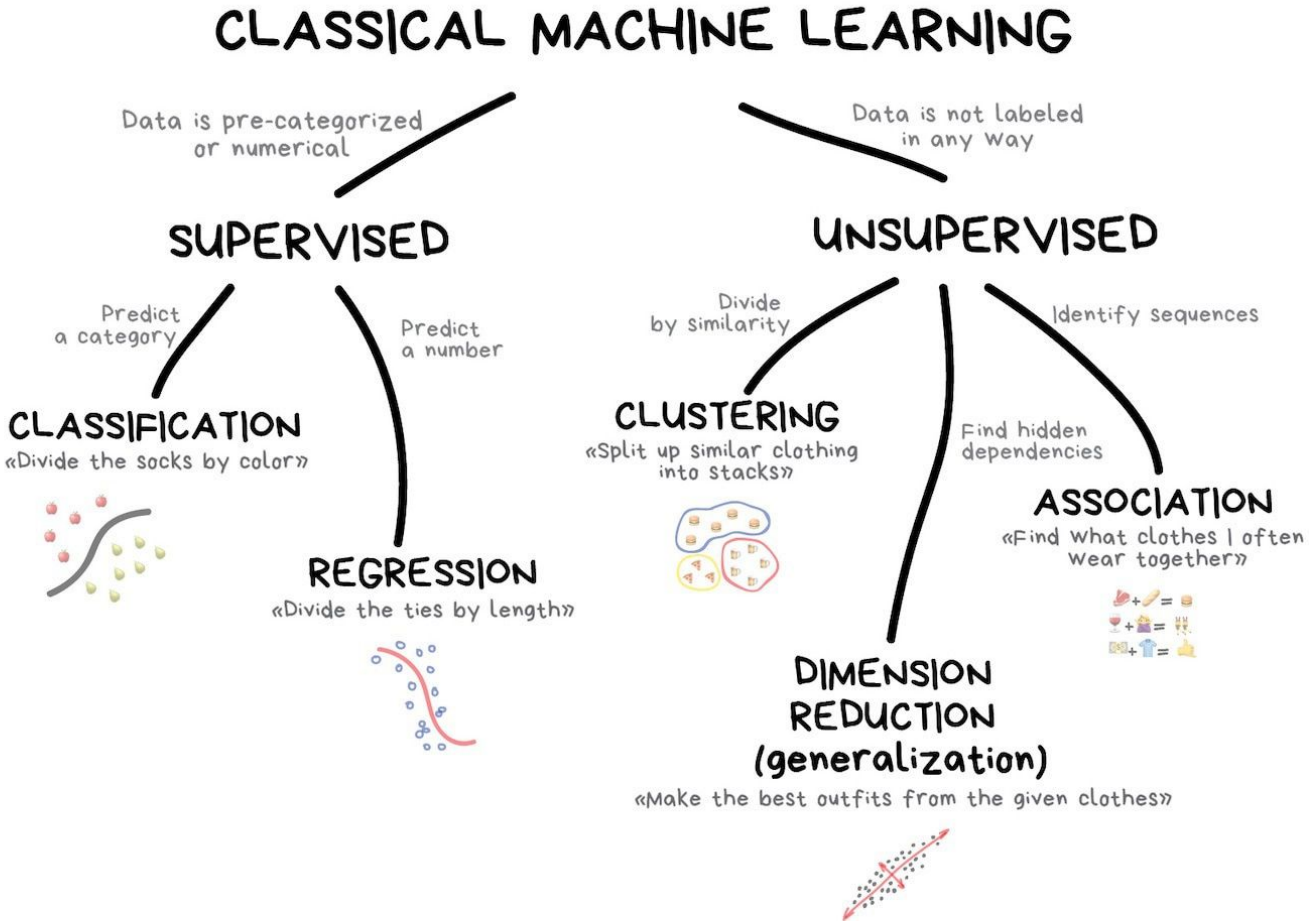


ML#5 Supervised Learning

Providing universal access to AI education and practice

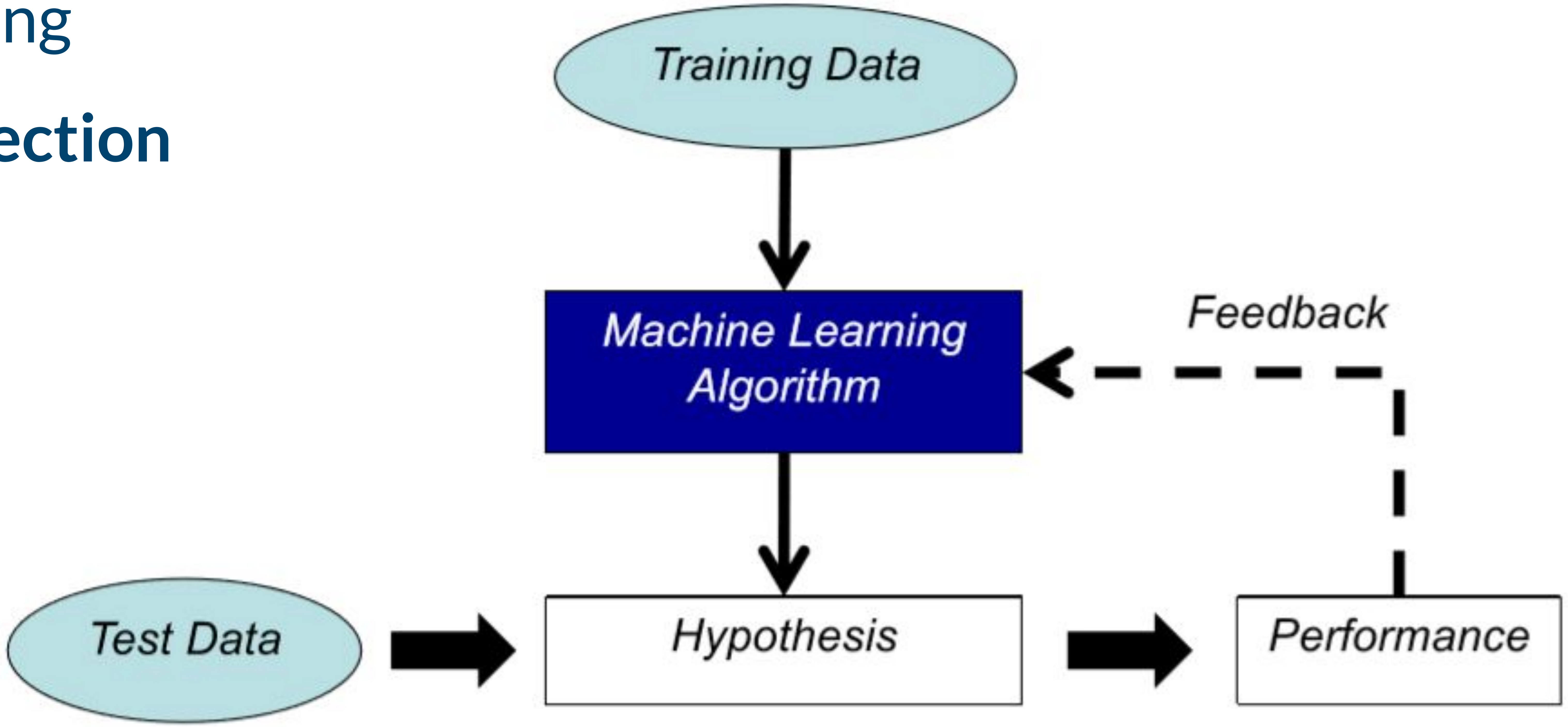
Important things to remember:

Machine Learning **Overview**


CLASSICAL MACHINE LEARNING
Data is pre-categorized or numerical
Data is not labeled in any way
SUPERVISED
UNSUPERVISED
Predict a category
Predict a number
Divide by similarity
Identify sequences
CLASSIFICATION
«Divide the socks by color»
REGRESSION
«Divide the ties by length»
CLUSTERING
«Split up similar clothing into stacks»
Find hidden dependencies
ASSOCIATION
«Find what clothes I often wear together»
DIMENSION REDUCTION (generalization)
«Make the best outfits from the given clothes»

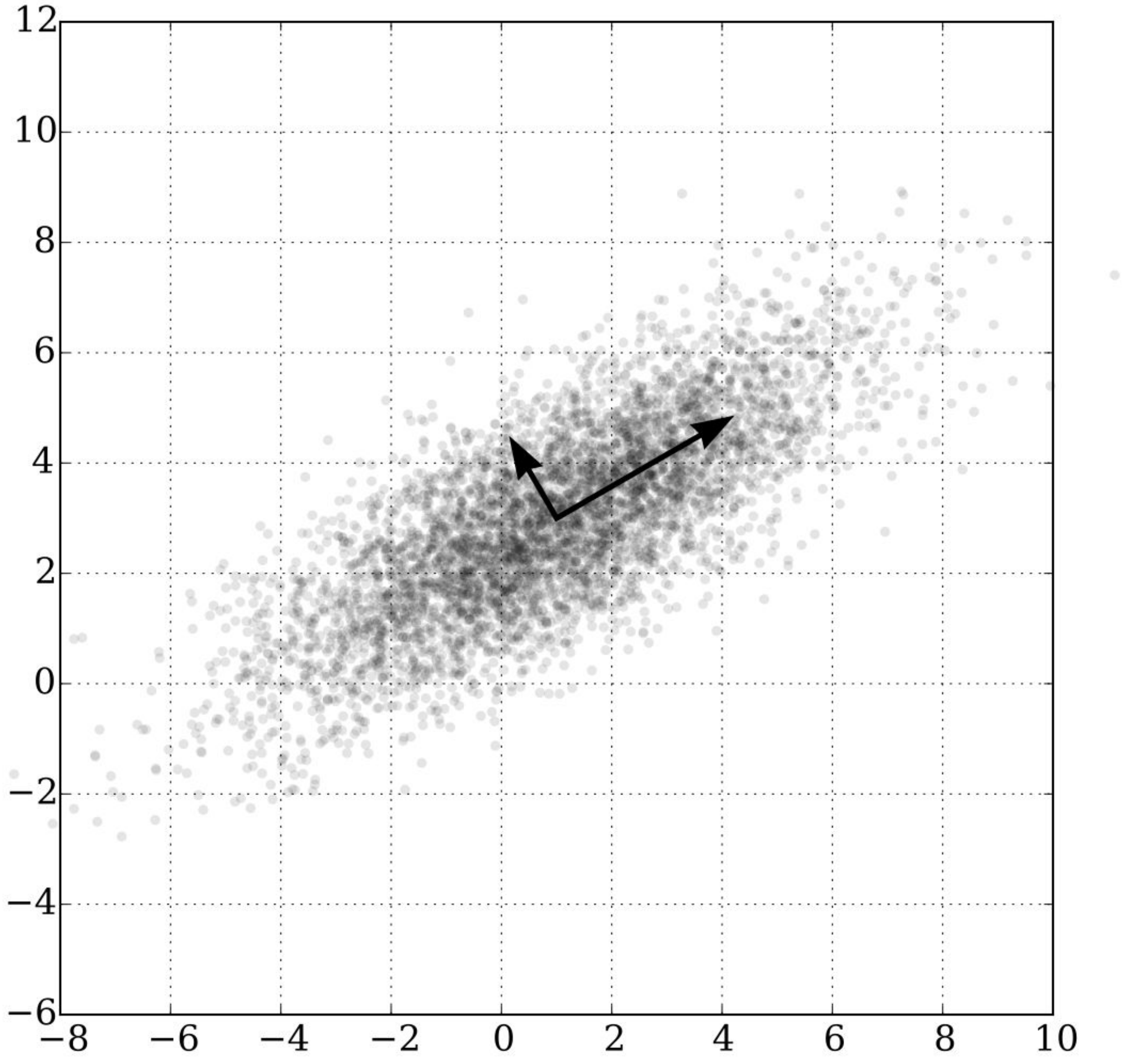
To **review**: ML Process

1. **Data collection** and **Preparation**
2. **Feature Selection** → Ticket price in Titanic
3. **Algorithm** Choice
4. **Split** between Test / Training
5. **Model** and **Parameter selection**
6. **Training Model** with Data
7. Evaluation with **Test set**



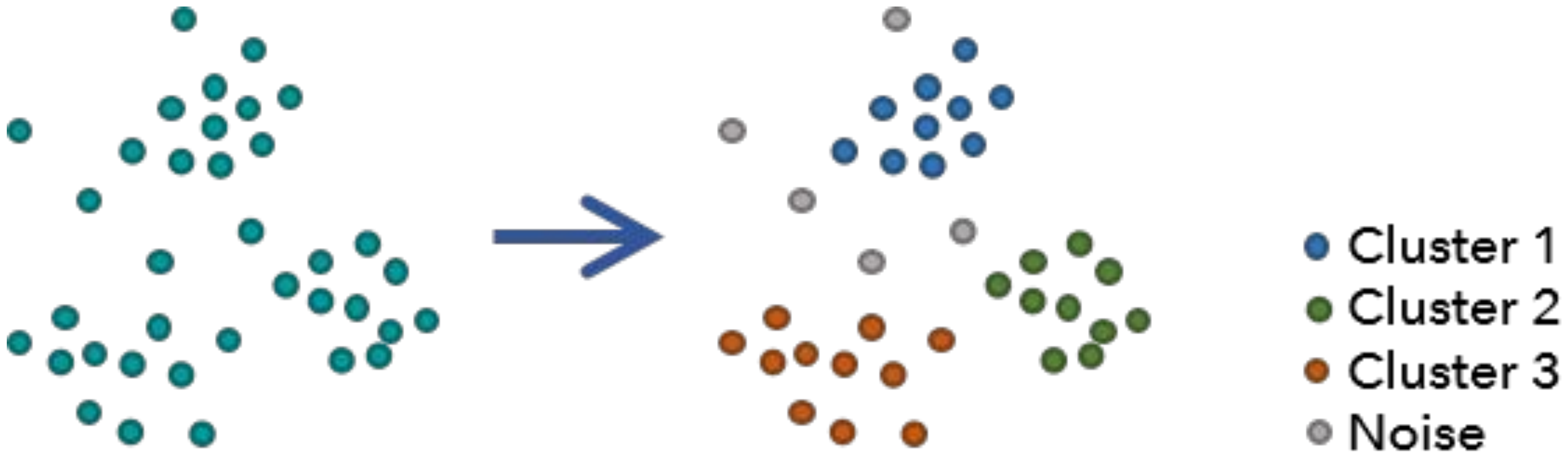
Dimensionality Reduction, Overview

- **Goal:** reducing the number of variables under consideration by obtaining a set of principal variables.
- **How does it work?:** Transforming the data in the high-dimensional space to a space in fewer dimensions.
  - **PCA** (Not considering labels, **unsupervised**)
  - **LDA** (Considering labels, **supervised**)
- **Usage:** Avoiding curse of dimensionality or overfitting due to strong correlations



Clustering

- **Formal Definition:** It is the task of grouping a set of objects in such a way that **objects** in the **same group** (called a cluster) are **more similar** (in some sense or another) to each other than to those in **other groups** (clusters).
- Raw Data → **Clustering Algorithm** → Clusters of data



Clustering: What is similarity?

- **Hard to define!**
- But we know it when we see it.
- We can actually **compute the distance between clusters / images**… etc

Picking the right **Distance Measure**

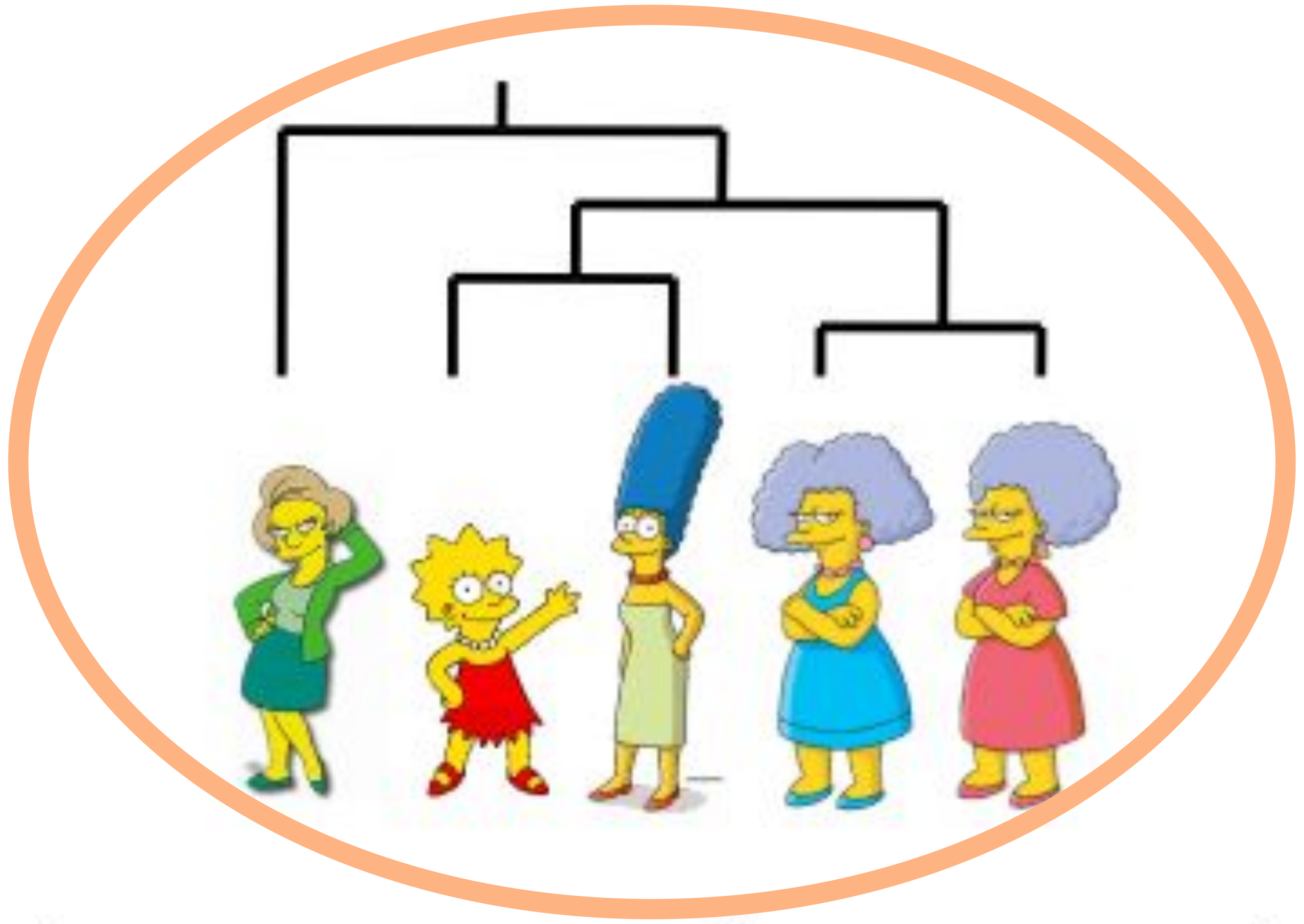
- Actually, there are many more distance measures… In the case of clustering, this parameter is important since it has a **strong influence in the results**. The most common one is the **Euclidean.**

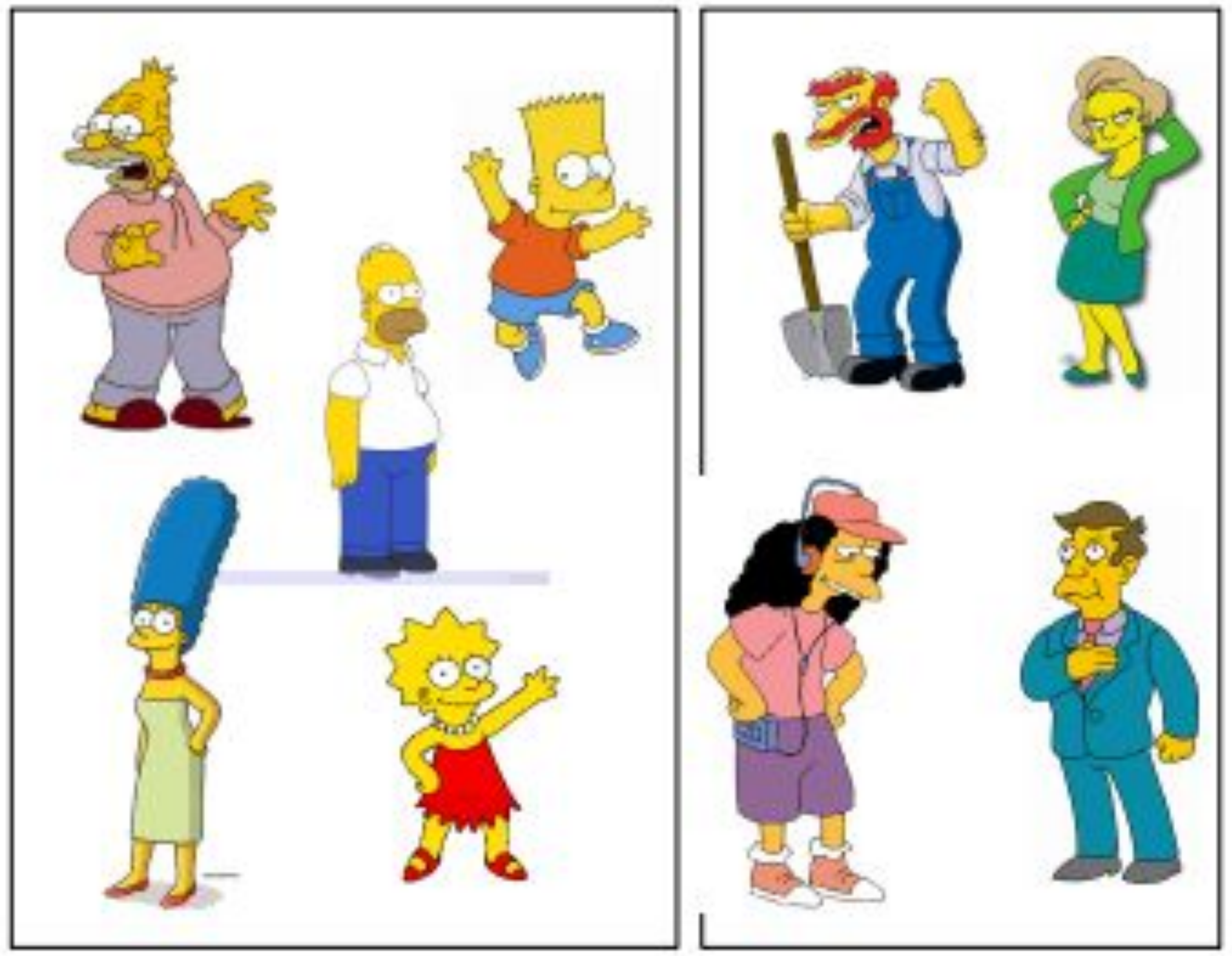
  Correlation based distance considers two objects together if their features are **highly correlated**, even if their values are far apart by Euclidean distance measures.

- Because of this, it is extremely **important to scale the data.** We would like to be able to compare between values.

Types of Clustering **Algorithms**

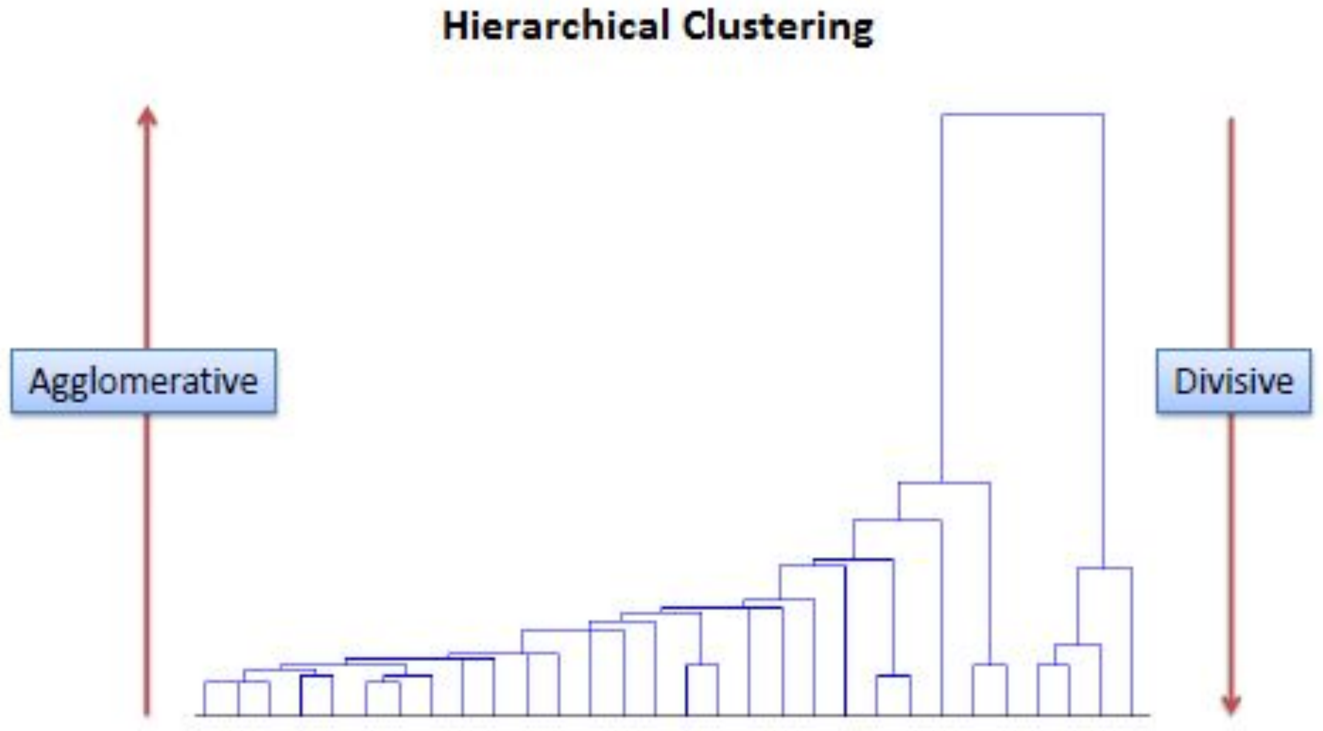
- **Hierarchical algorithms**
  - Examples are organized as a binary tree
  - No explicit division in groups
    - Bottom-up
    - Top-down
- **Partitional algorithms**
  - Usually start with a random (partial) partitioning
  - Refine it iteratively:
    - K-means clustering
    - Mixture-model based clustering





Hierarchical Clustering

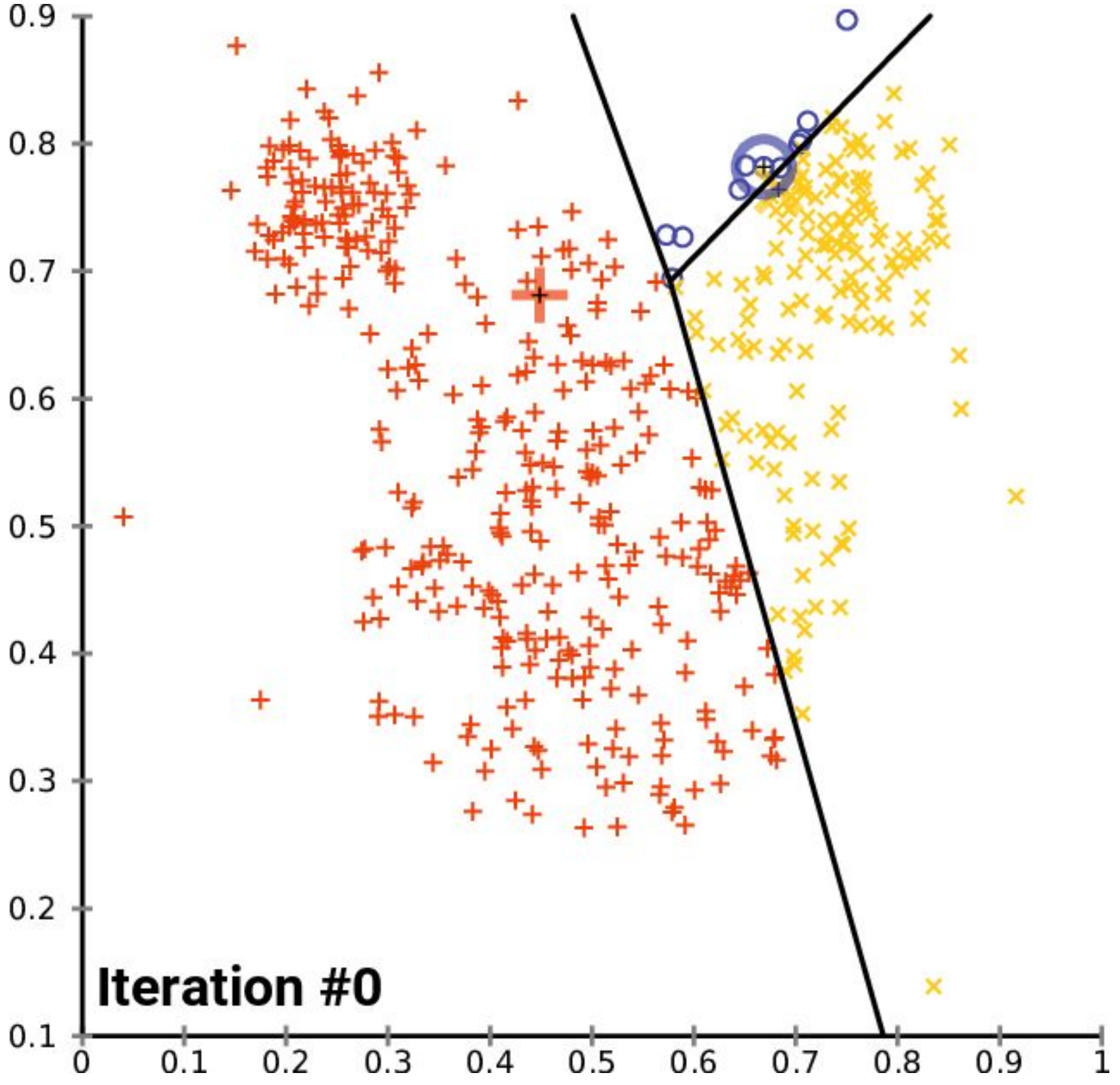
- **Agglomerative (bottom-up)**
  - Methods start with each example in its own cluster
  - Iteratively combine them to form larger and larger clusters
- **Divisive (partitional, top-down)**
  - Methods start with all the examples in a single cluster
  - Consider all the possible way to divide the cluster into two. Choose the best division
  - Recursively operate on

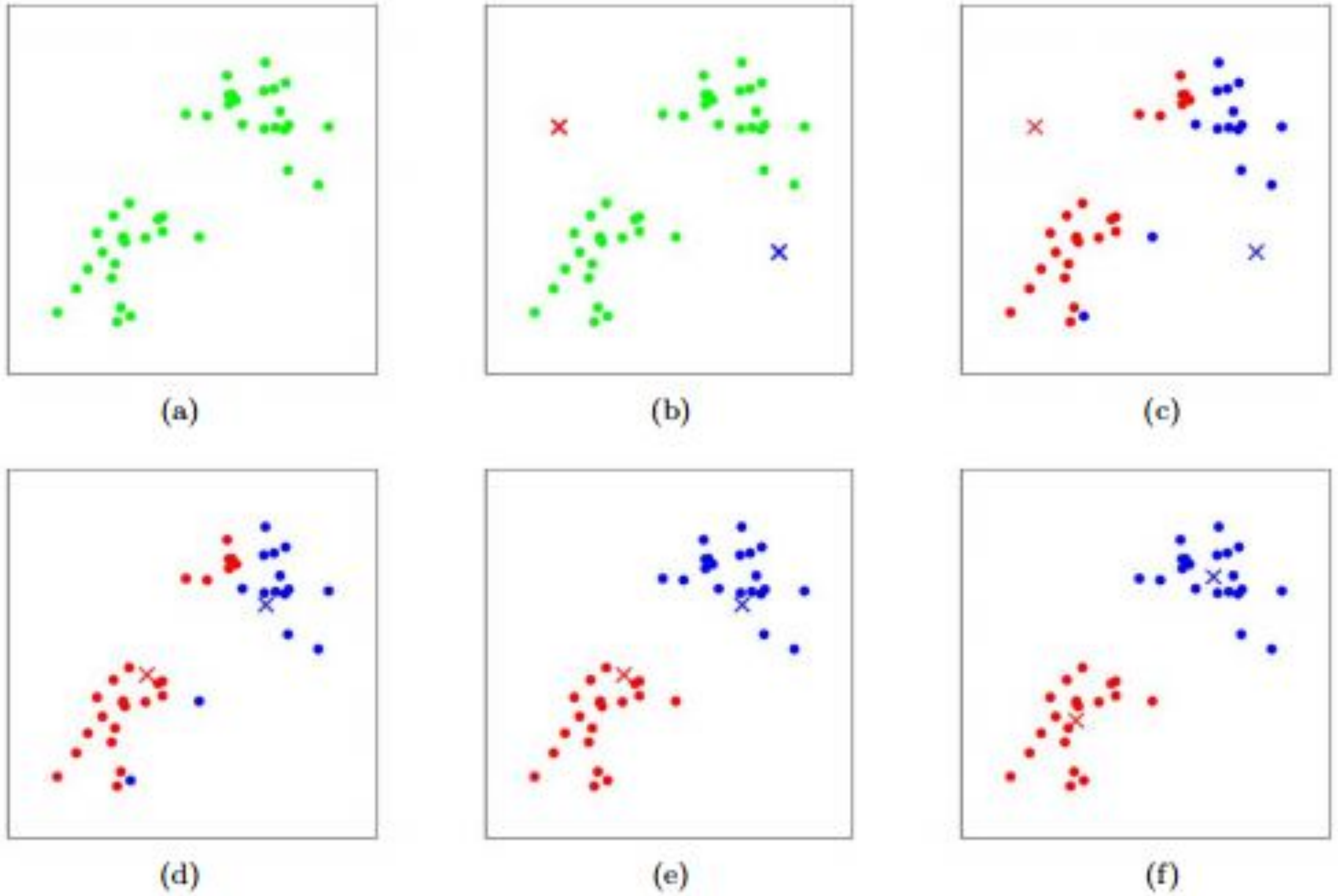


Clustering: Partitional Algorithms

- **Method:** construct a partition of **n** objects into a set of **k** clusters
- **Given:** a set of objects (training set) and typically must provide the number of desired clusters, **K.**
- **Basic process:**
  - Randomly choose K instances as **seeds**, one per cluster
  - Form initial clusters based on these seeds
  - Iterate , repeatedly reallocating instances to different clusters to improve the overall clustering
  - Stop when clustering converges or after a fixed number of iterations

Clustering: K-Means Example

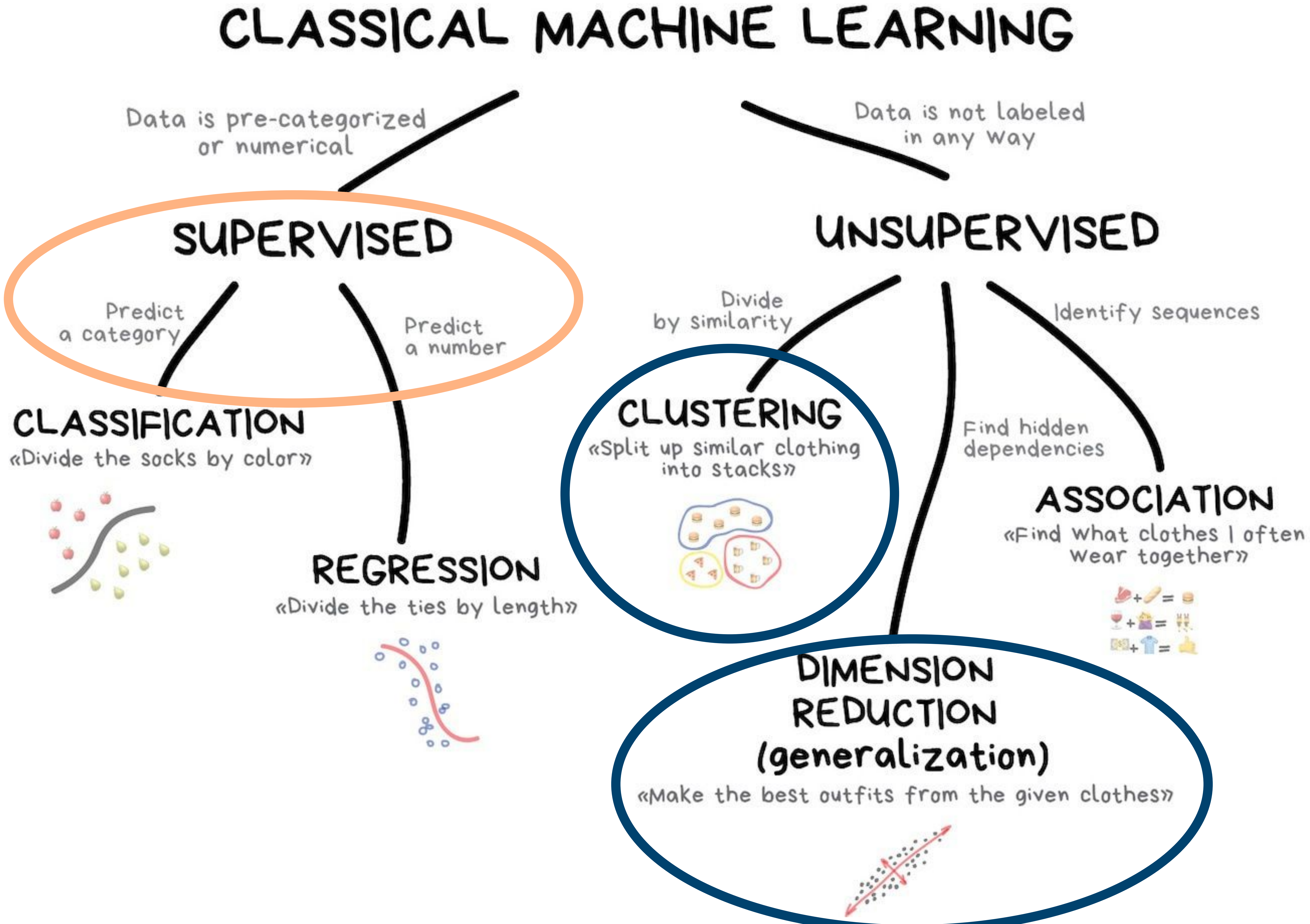




Comments on K-Means Method

- **Strengths:**
  - Relatively Efficient (computationally)
  - Often terminates at a local optimum
- **Weakness:**
  - Applicable only when mean is defined… What about categorical data?
  - Need to specify k, the number of clusters, in advance
  - Unable to handle noisy data and outliers
  - Not suitable is clusters have non-convex shapes
- **Seed Choice:** Results can vary based on Random seed Choice
  - Some seeds can result in poor convergence. Important to **start** several times.

What we will do **today**


CLASSICAL MACHINE LEARNING
Data is pre-categorized or numerical
Data is not labeled in any way
SUPERVISED
UNSUPERVISED
Predict a category
Predict a number
Divide by similarity
Identify sequences
Find hidden dependencies
CLASSIFICATION
«Divide the socks by color»
REGRESSION
«Divide the ties by length»
CLUSTERING
«Split up similar clothing into stacks»
ASSOCIATION
«Find what clothes I often wear together»
DIMENSION REDUCTION (generalization)
«Make the best outfits from the given clothes»

What is **Machine Learning?**

Improve the performance of a software system, based on previous experience:

- **prediction: supervised learning** given (x, y) pairs, find a mapping from a new x to a new y, e.g., regression, classification
- **understanding: unsupervised learning** given a set of x, find something interesting or useful about their structure, e.g. density estimation, clustering, dimensionality reduction
- **control:** reinforcement learning given an external system upon which you can exert control action a and receive percepts, p, a reward signal r indicating good performance, find a mapping from P→A that maximizes some long-term measure of r

What is Supervised Learning

- Supervised learning is where you have input variables (x) and an output variable **(y)** and you use an algorithm to learn the mapping function from the input to the output. **y = f(x)**
- The process of an algorithm learning from the training dataset can be thought of as a **teacher supervising** the **learning process**.
- We know the **correct answers**, the algorithm iteratively makes predictions on the **training data** and is corrected by the teacher. **Learning stops** when the algorithm achieves an

Generalization in Supervised Learning

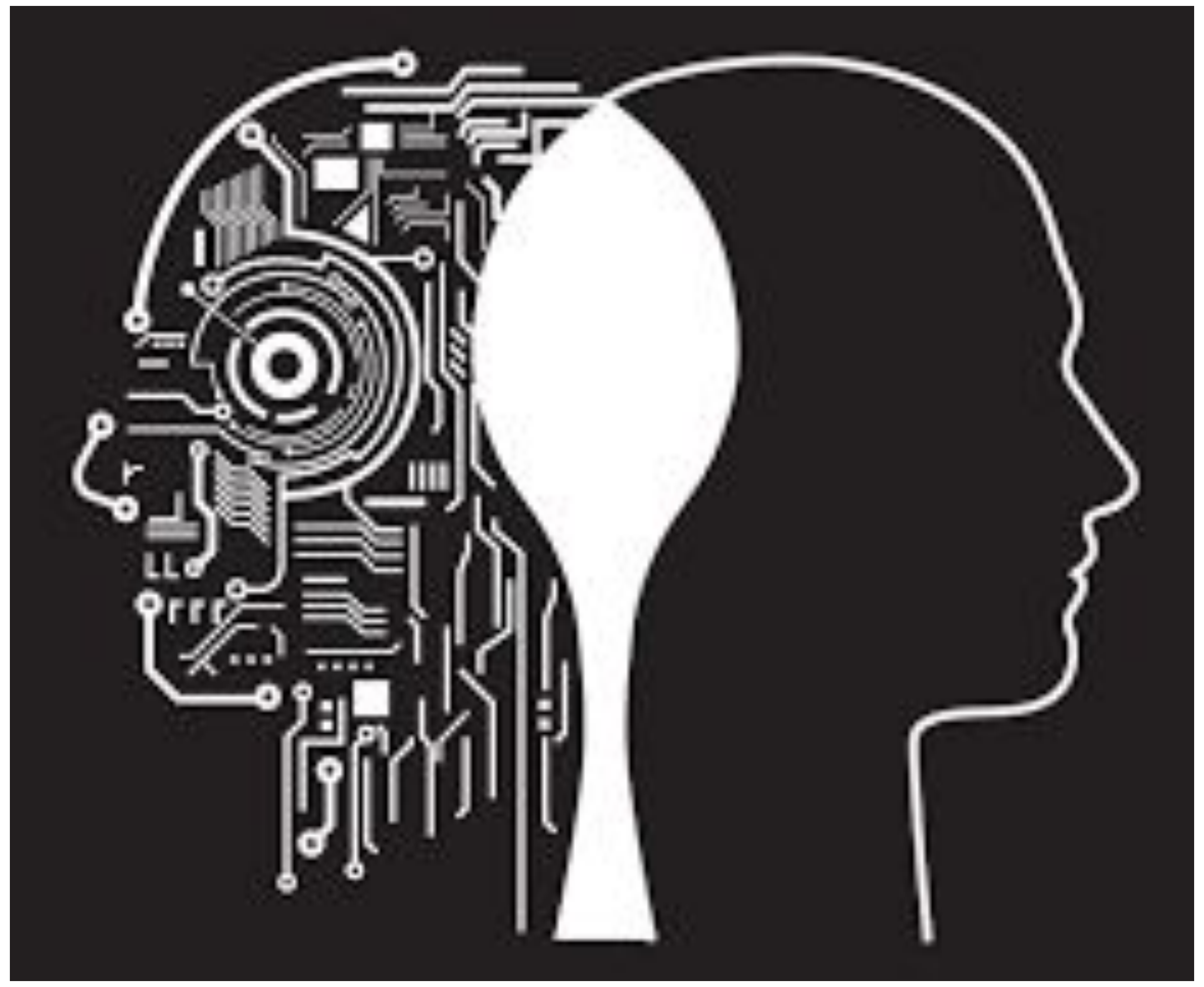
We want to find a model that will perform the best on future examples

- We don’t know what the future data will be
- We have **some past data**
- We hope that future data will **remember past data** in a way that while let us use the past data to construct a model that will perform well in the future
- *It will still most likely not work for the stock market...*

Human input needed

Humans have to do a **lot of work to do**, up front, to set up a machine learning problem.

- **What** do you want to predict?
- What kind of **data** can you get?
- What is the **relative costs** of different types of errors?
- How does the **available data** relate to **future data**?
- What **model** should we use?
- Etc...



A practical example

Imagine that this class took place in university and not a bootcamp and we were grading your course.

If you know **information** (students record) about people who passed **this course** (data x), and **their performance** (label y), then, **given your own record** you may ask the system to p**redict your grade** based on the previous experience and the power of the learning algorithm.

Actually, I built this for my university a while back and they were able to predict 4 years graduation success based on first year grades.

A practical example: ML Grades

Av. math grade	Grade in ML
5.0	4.2
6.2	5.9
7.4	8.1
⋮	⋮

Supervised learning:
Given the "right" answer for each example in de data.

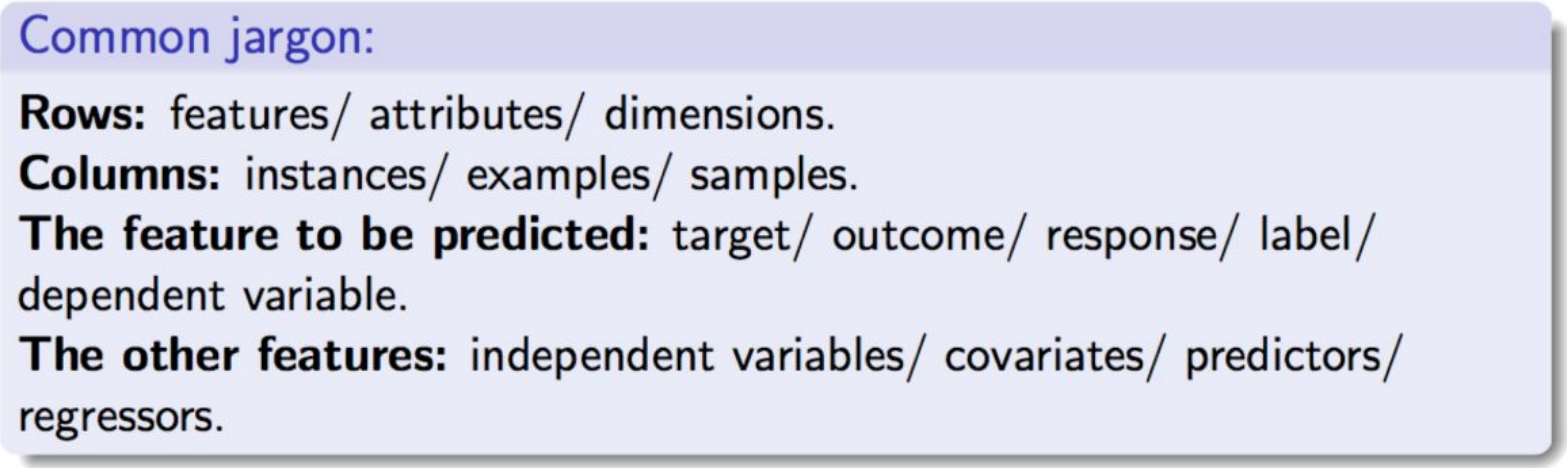
Regression problem:
Predict a real-value output.

Notation

Input (features): $\mathbf{x} \in \mathbf{R}^d$ (student's record)
Output (labels): $y \in \mathbf{R}$ (actual grade in ML)
Data: Examples of inputs and output pairs: $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$

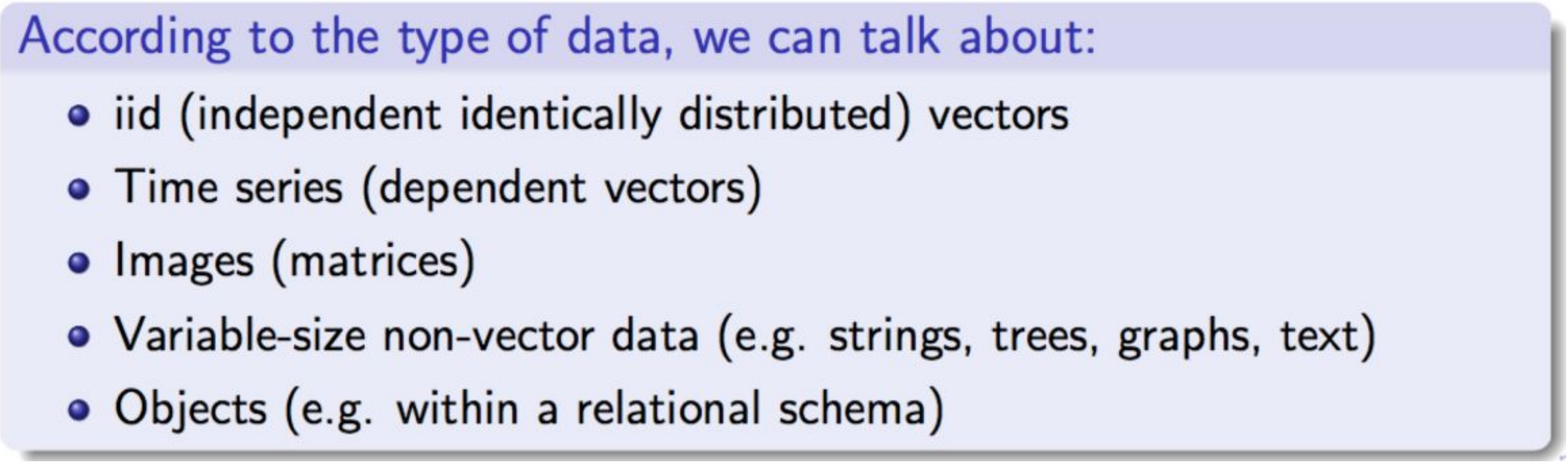
A practical example: ML Grades

We will usually use column-wise notation for each sample.



Common jargon:

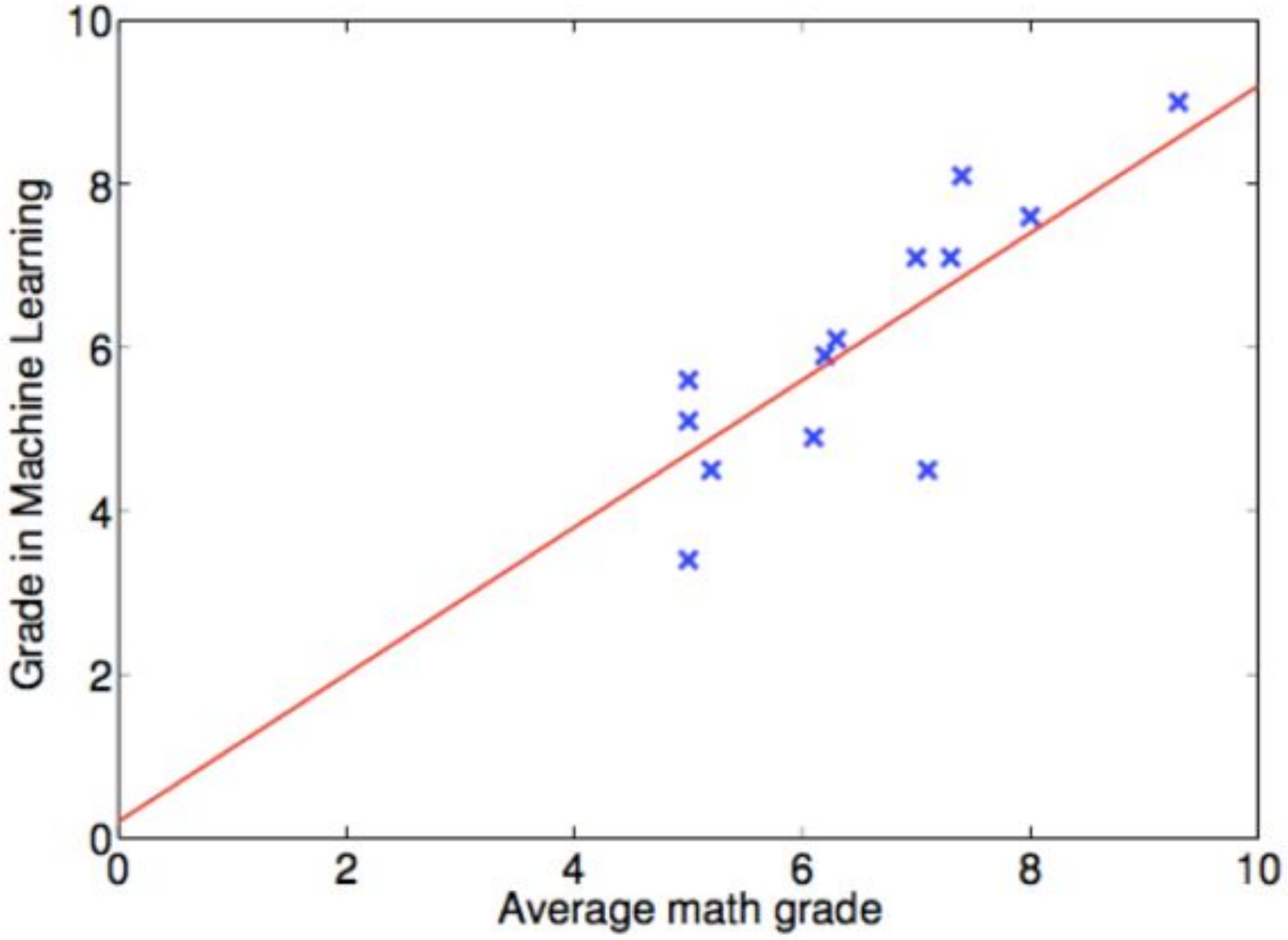
Rows: features/ attributes/ dimensions.
Columns: instances/ examples/ samples.
The feature to be predicted: target/ outcome/ response/ label/ dependent variable.
The other features: independent variables/ covariates/ predictors/ regressors.

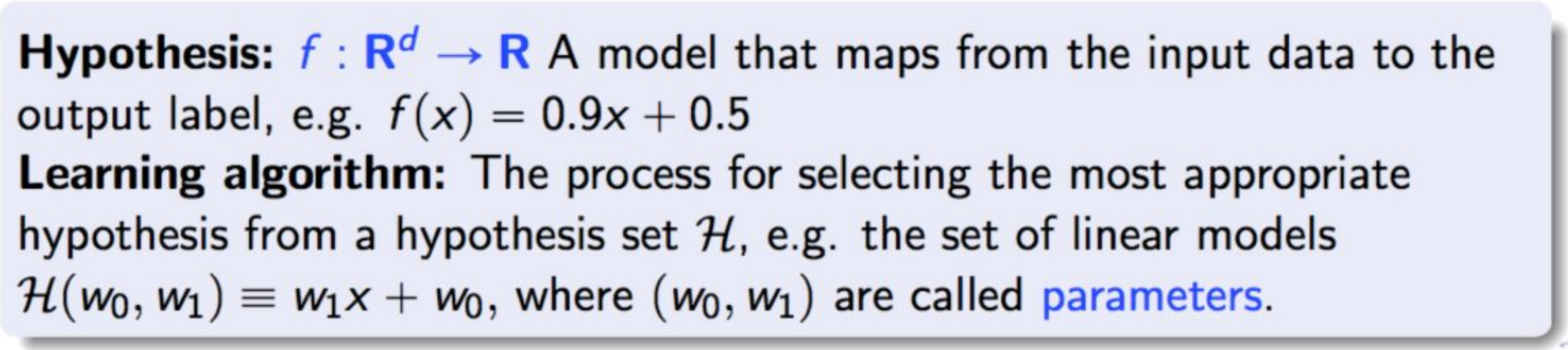


According to the type of data, we can talk about:

- iid (independent identically distributed) vectors
- Time series (dependent vectors)
- Images (matrices)
- Variable-size non-vector data (e.g. strings, trees, graphs, text)
- Objects (e.g. within a relational schema)

A practical example: ML Grades

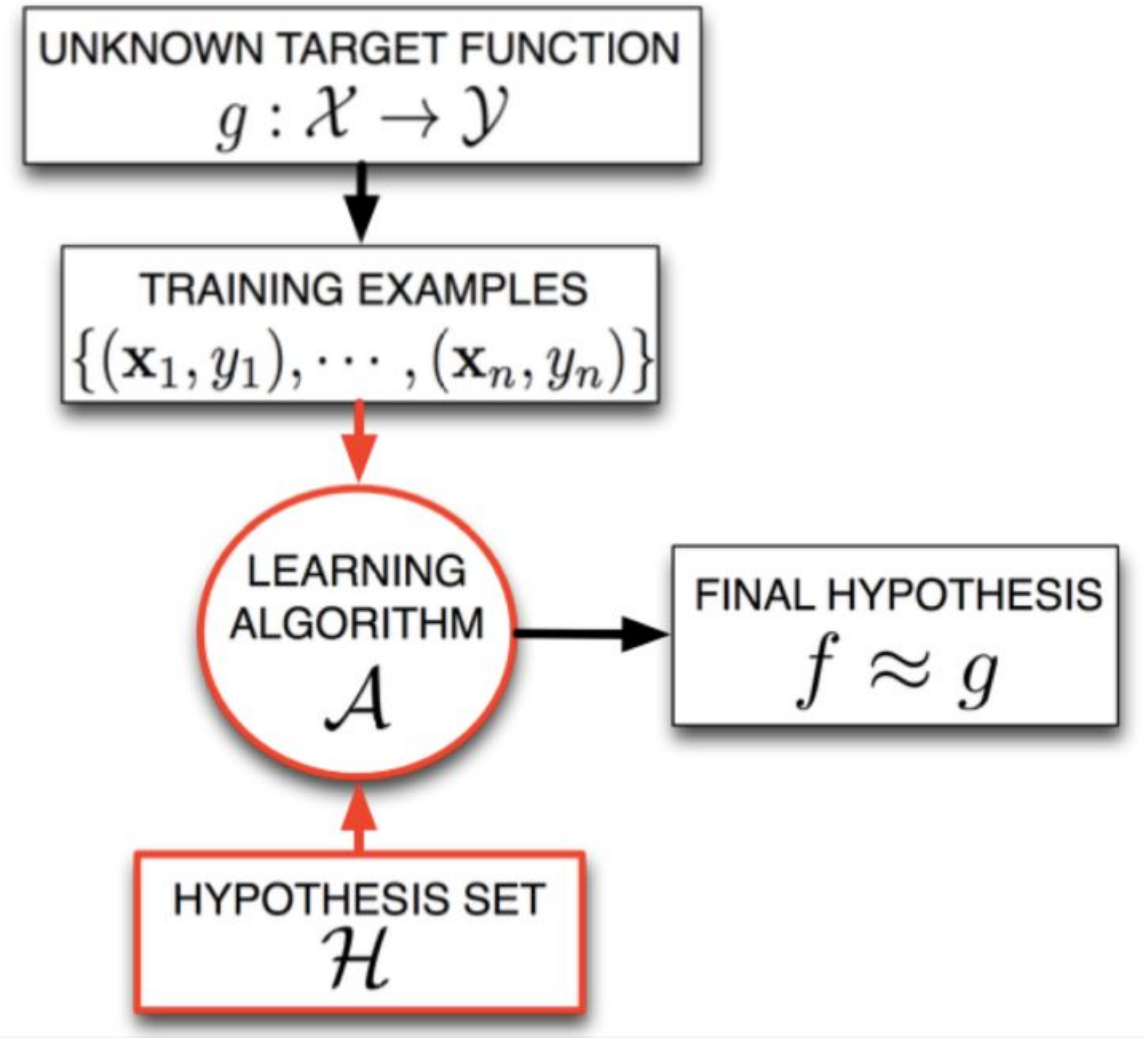


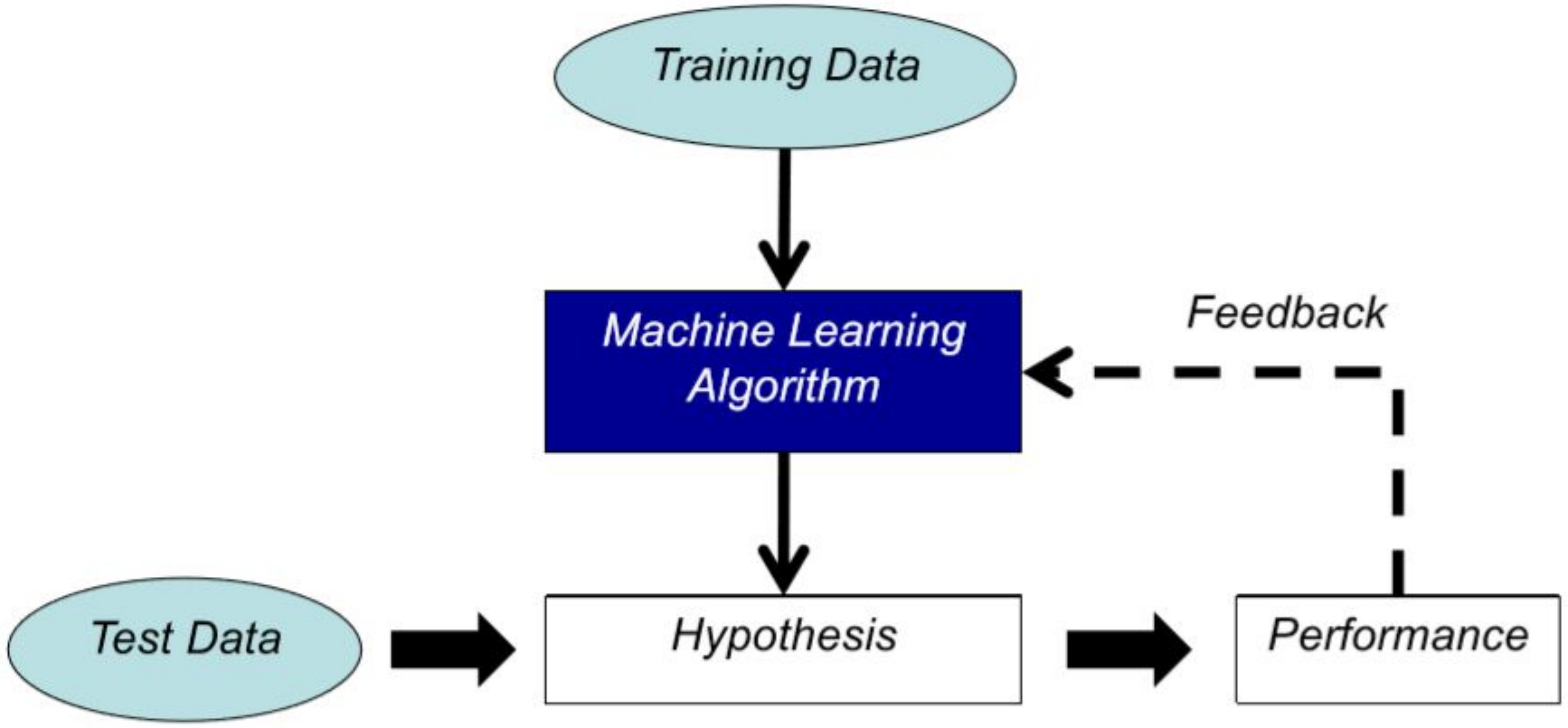


Hypothesis: $f : \mathbf{R}^d \rightarrow \mathbf{R}$ A model that maps from the input data to the output label, e.g. $f(x) = 0.9x + 0.5$

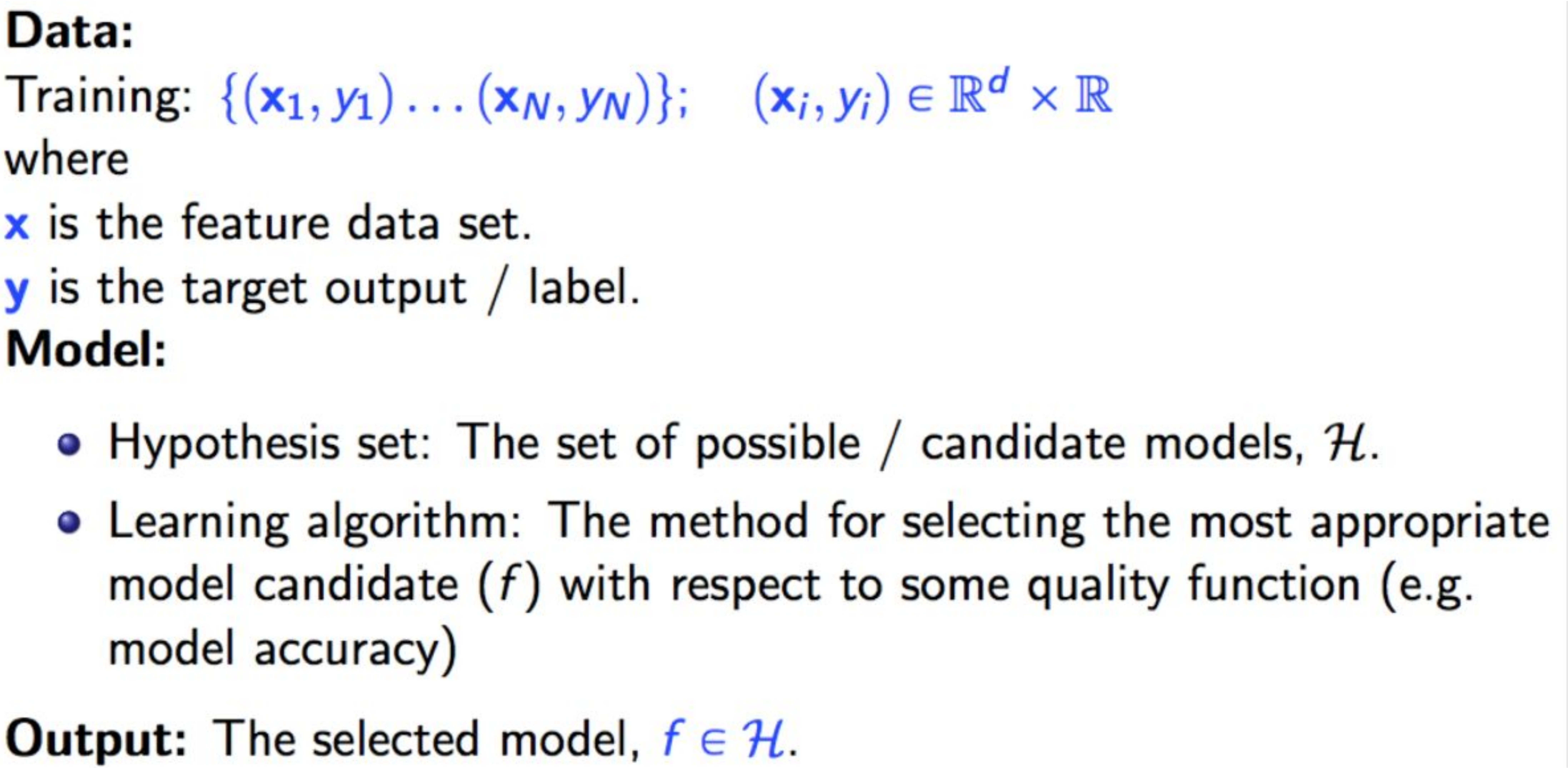
Learning algorithm: The process for selecting the most appropriate hypothesis from a hypothesis set $\mathcal{H}$, e.g. the set of linear models $\mathcal{H}(w_0, w_1) \equiv w_1 x + w_0$, where (w_0, w_1) are called parameters.

A practical example: ML Grades





Elements of a supervised problem



Data:

Training: $\{(\mathbf{x}_1, y_1) \ldots (\mathbf{x}_N, y_N)\}; \quad (\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$

where

$\mathbf{x}$ is the feature data set.

$\mathbf{y}$ is the target output / label.

Model:

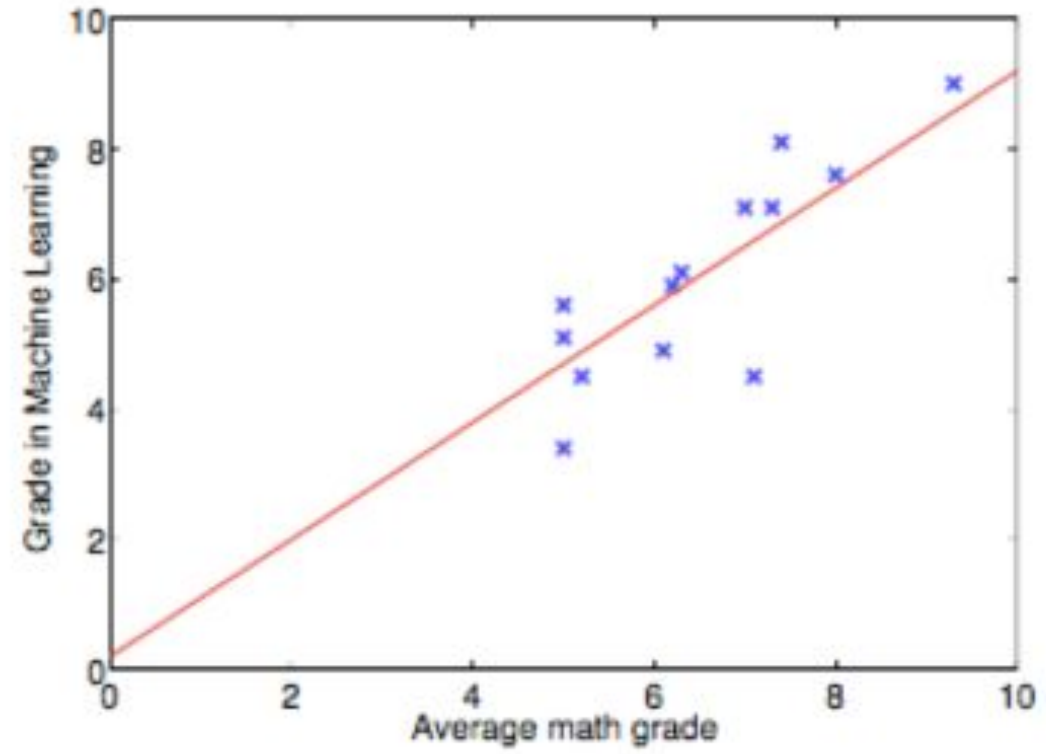
- Hypothesis set: The set of possible / candidate models, $\mathcal{H}$.
- Learning algorithm: The method for selecting the most appropriate model candidate (f) with respect to some quality function (e.g. model accuracy)

Output: The selected model, $f \in \mathcal{H}$.

A Basic model: Linear Regression

Univariate linear regression hypotheses set

$\mathcal{H}(w_0, w_1) \equiv w_1 x + w_0.$



Learning process idea:

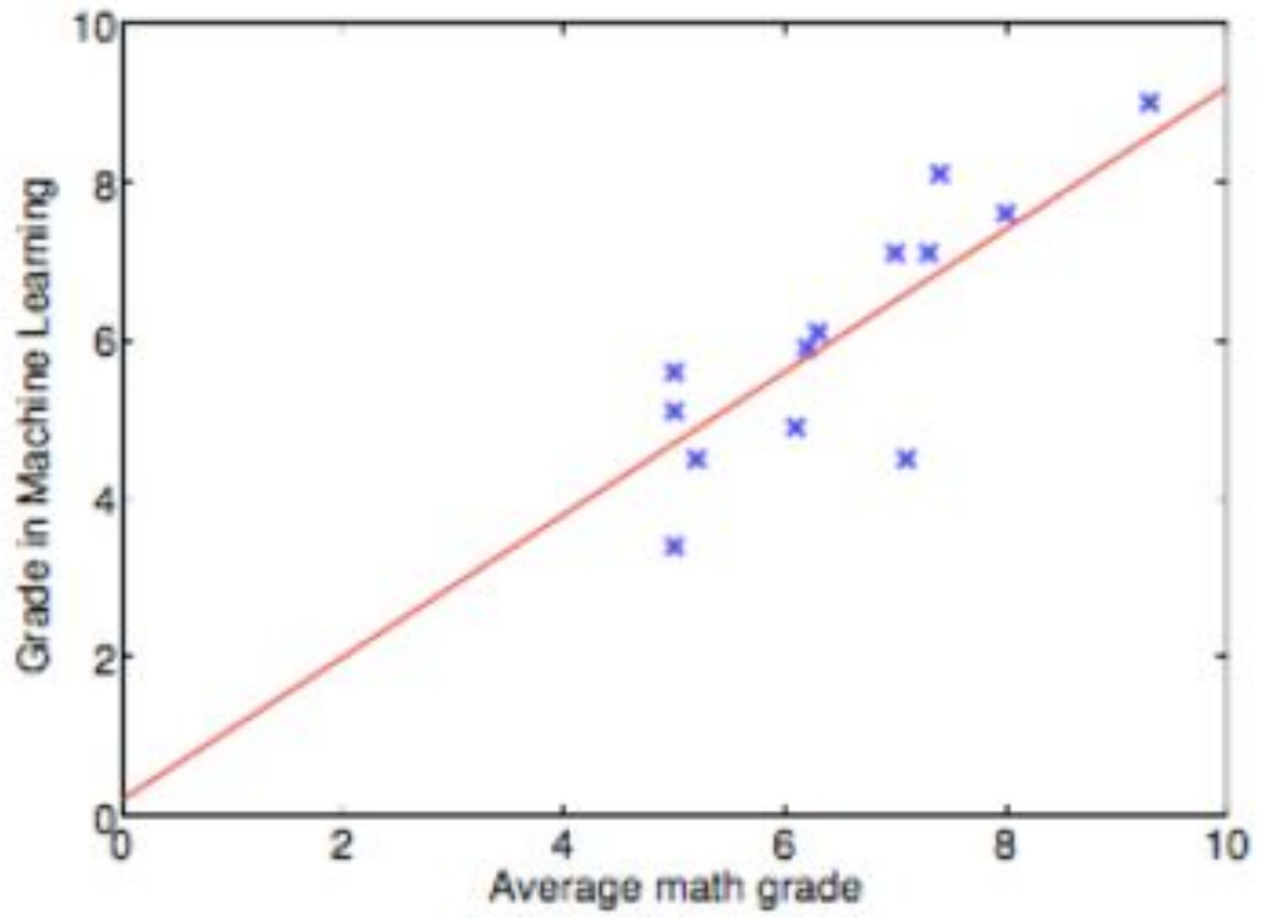
We want to find the model defined by the parameters (w_0, w_1) so that our prediction $f(x_i; w_0, w_1)$ on sample x_i is as close as possible to y_i. This is, the "distance" between $f(x_i; w_0, w_1)$ and y_i is minimum for all elements in the training set.

$$\underset{w_0, w_1}{\text{minimize}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i; w_0, w_1) - y_i)^2$$

A Basic model: Linear Regression

Univariate linear regression hypotheses set

$\mathcal{H}(w_0, w_1) \equiv w_1 x + w_0.$



The last ingredient:

The cost function/ loss function/ error measure:

$$\underset{w_0, w_1}{\text{minimize}}\ J(w_0, w_1)$$

Adding it all together

Hypothesis:

$$f(x, w) = w_1 x + w_0$$

Parameters:

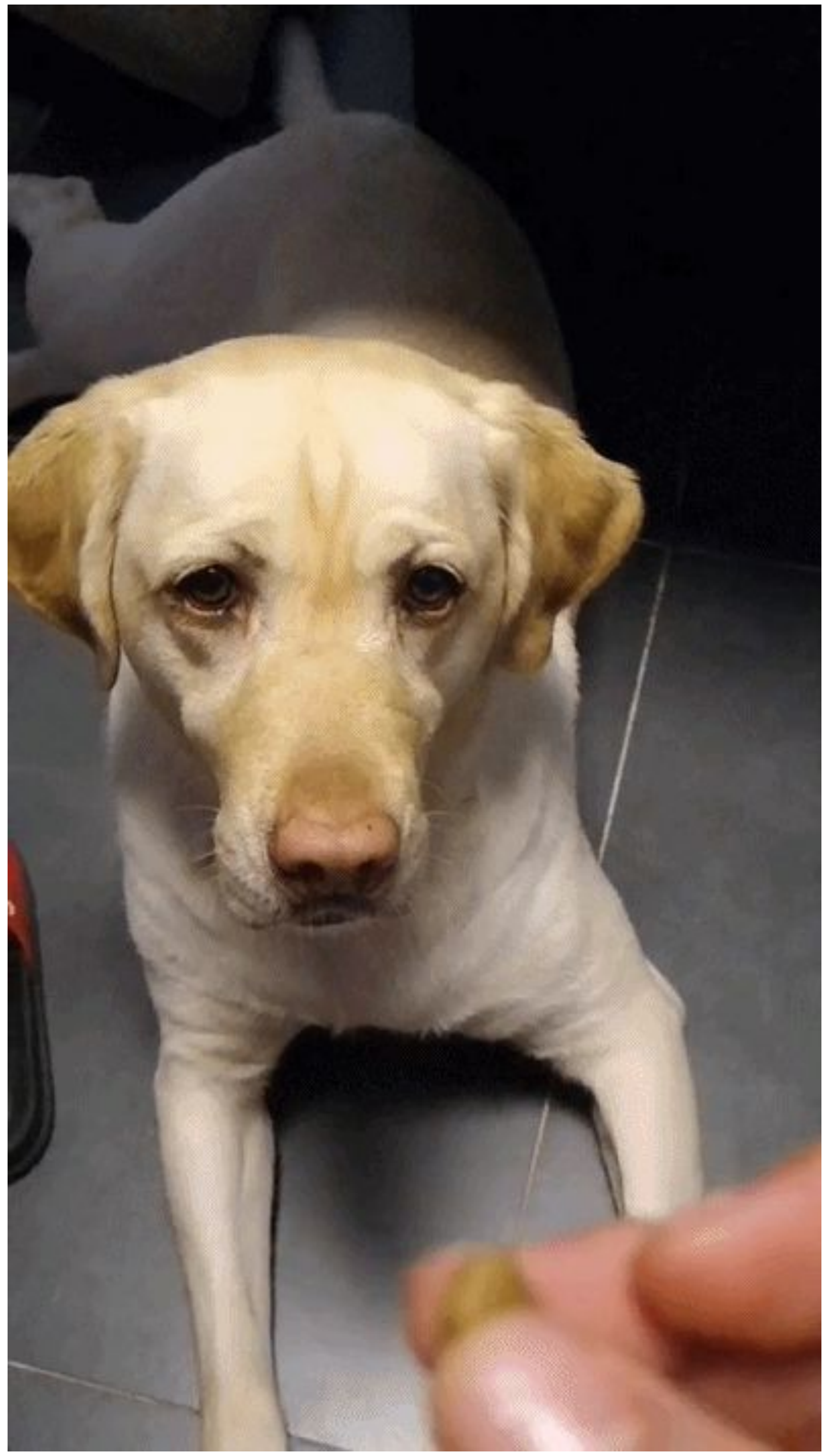
$$w = (w_0, w_1)^T$$

Cost function:

$$\mathcal{J}(w_0, w_1) = \frac{1}{N} \sum_{i=1}^{N} (f(x_i; w_0, w_1) - y_i)^2$$

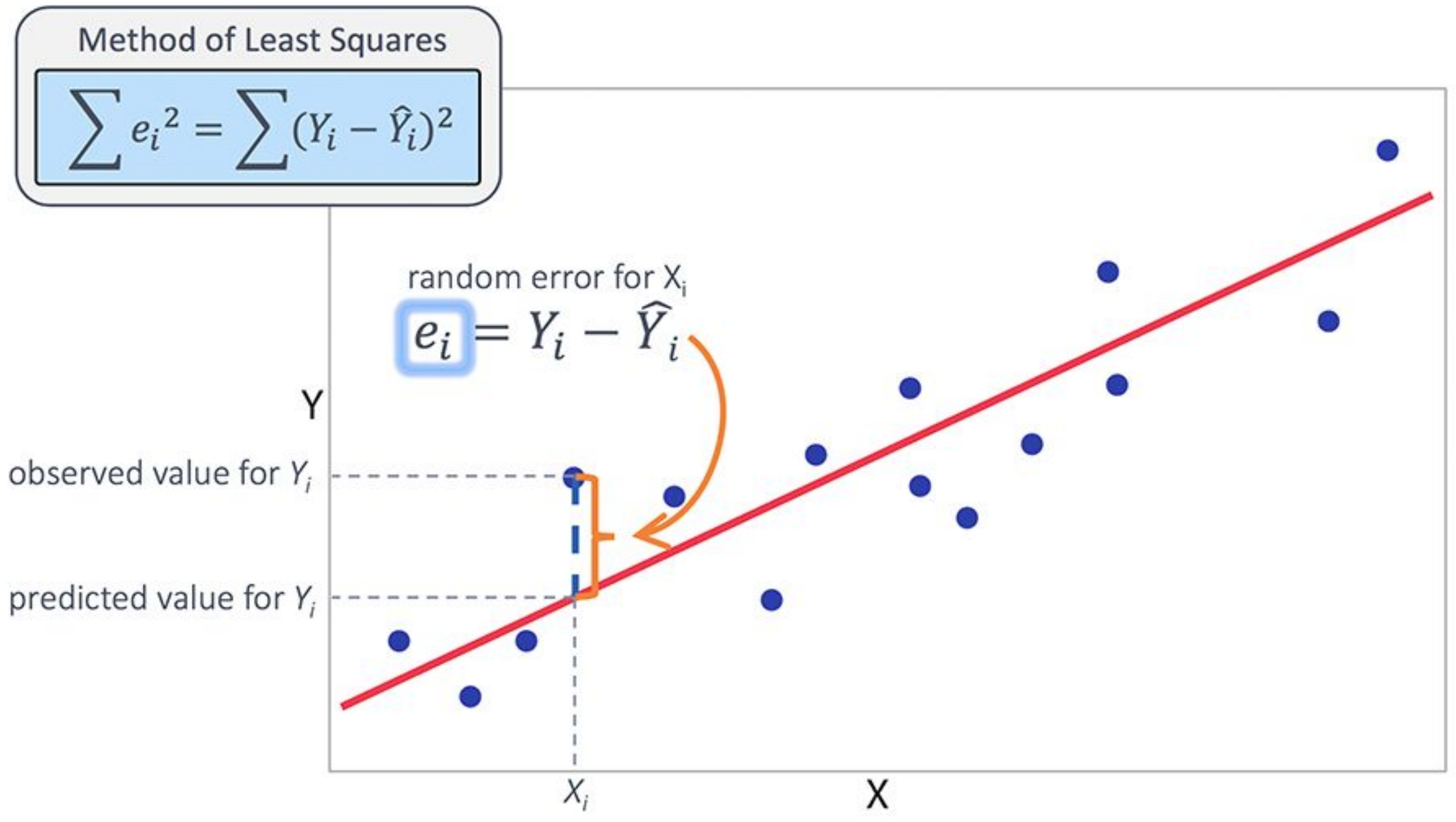
Goal:

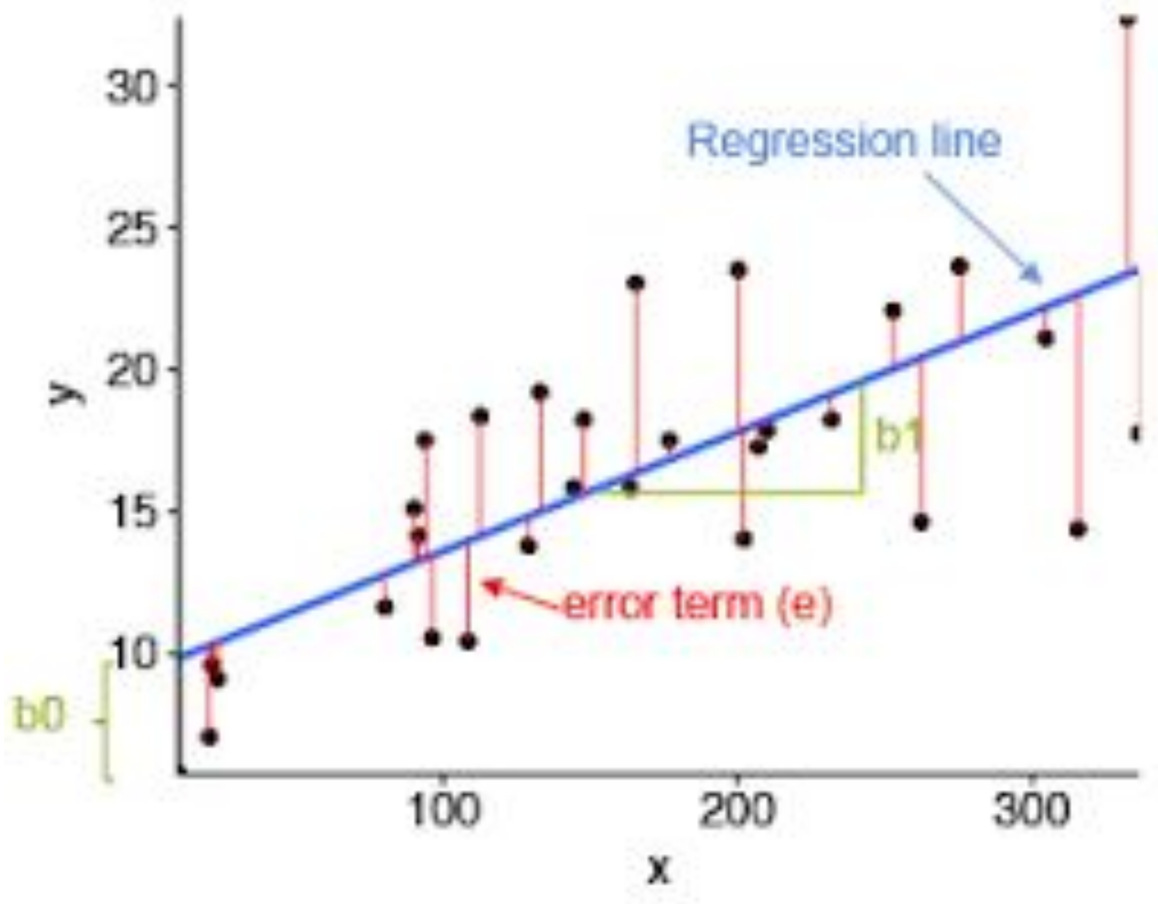
$$\underset{w_0, w_1}{minimize}\ \mathcal{J}(w_0, w_1)$$

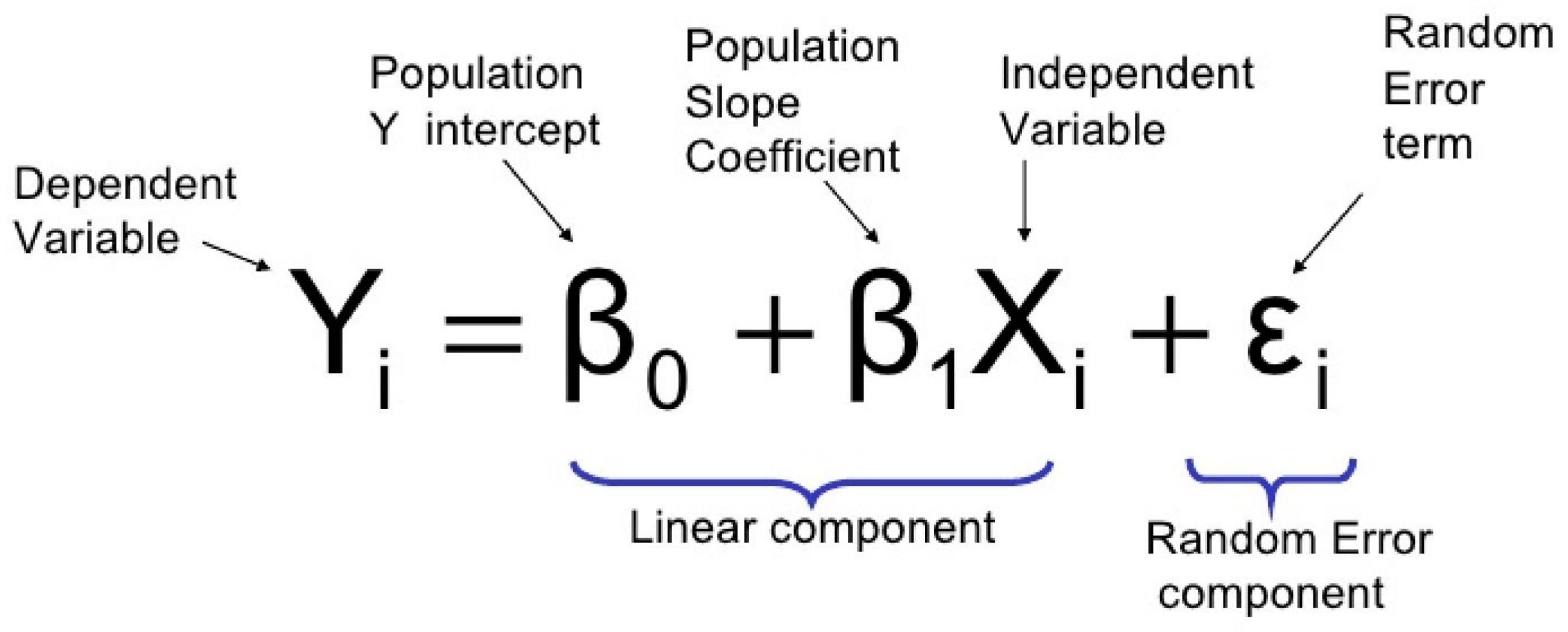


Which we have seen before!

Attempts to minimize some measure of error (**loss function**) such as **mean squared error**:







Dependent Variable

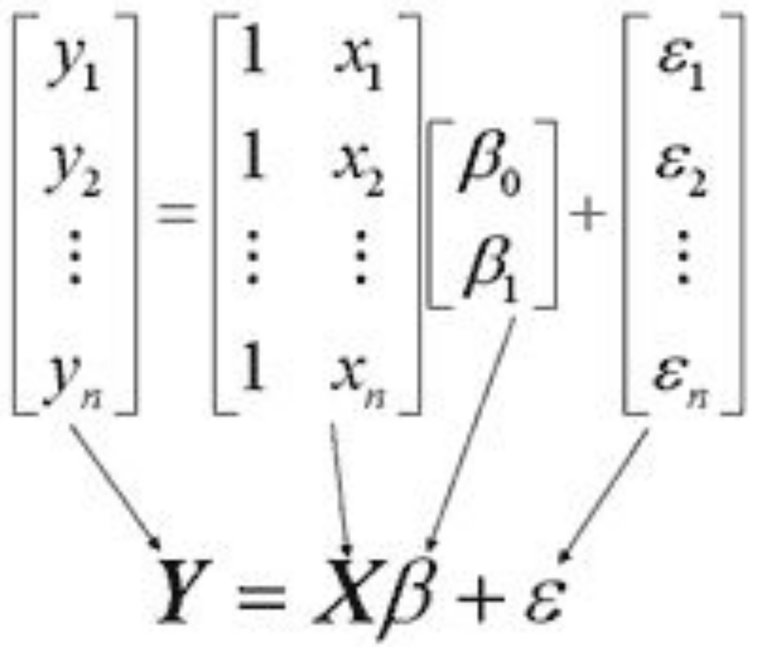
Population Y intercept

Population Slope Coefficient

Independent Variable

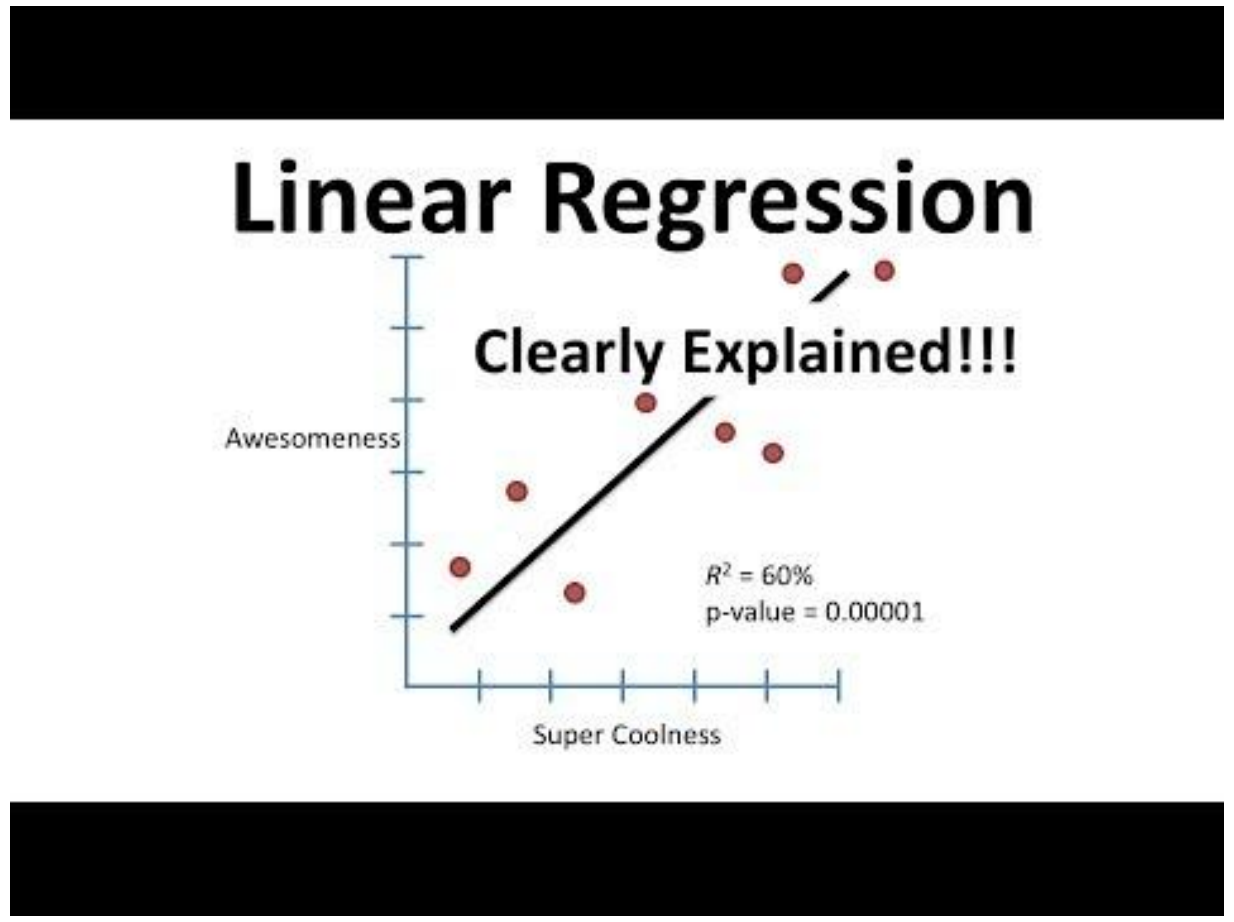
Random Error term

$$\underbrace{Y_i = \beta_0 + \beta_1 X_i}_{\text{Linear component}} + \underbrace{\varepsilon_i}_{\text{Random Error component}}$$



$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

$$Y = X\beta + \varepsilon$$



And if you remember...

I promised I would explain better :)

Linear Regression: Slope and Intercept

A simple linear regression model defines the relationship between a dependent variable and a single independent predictor variable using a line defined by an equation in the following form:

$$\hat{y} = a + bx + \varepsilon$$

- **ŷ** = Target variable, dependent variable.
- **x** = Feature, independent variable.
- a = Intercept, y value when x is 0.
- b = Slope, raise of the y value for every unit of x.
- **ε** = Error term, residuals

Linear Regression: Least Squares Estimation

$$\hat{y} = a + bx + \varepsilon$$

Though the proof is beyond the scope of this course, it can be shown using calculus that the value of b that results in the minimum squared error is:

$$b = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$

The horizontal bar appearing over the **x̄** and **ȳ** terms indicates the **mean value of x or y**. The solution for **a** depends on the value of **b**. It can be obtained using the following formula: **$a = \bar{y} - b\bar{x}$**

Linear Regression: Least Squares Estimation

$$b = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$

If we break this equation apart into its component pieces, we can simplify it a bit.

The **numerator** involves taking the **sum of each data point's deviation** from the **mean x value** multiplied by that **point's deviation** away from the **mean y value**. This is similar to the **covariance function** for x and y, denoted as Cov(x, y).

$$\underline{\mathrm{Cov}(x, y)} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{n}$$

$$\underline{\mathrm{Var}(x)} = \frac{\sum(x_i - \bar{x})^2}{n}$$

Linear Regression: Least Squares Estimation

$$b = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$

If we break this equation apart into its component pieces, we can simplify it a bit.

The denominator for b should also look familiar; it is very **similar** to the **variance** of **x**, which is denoted as **Var(x)**.

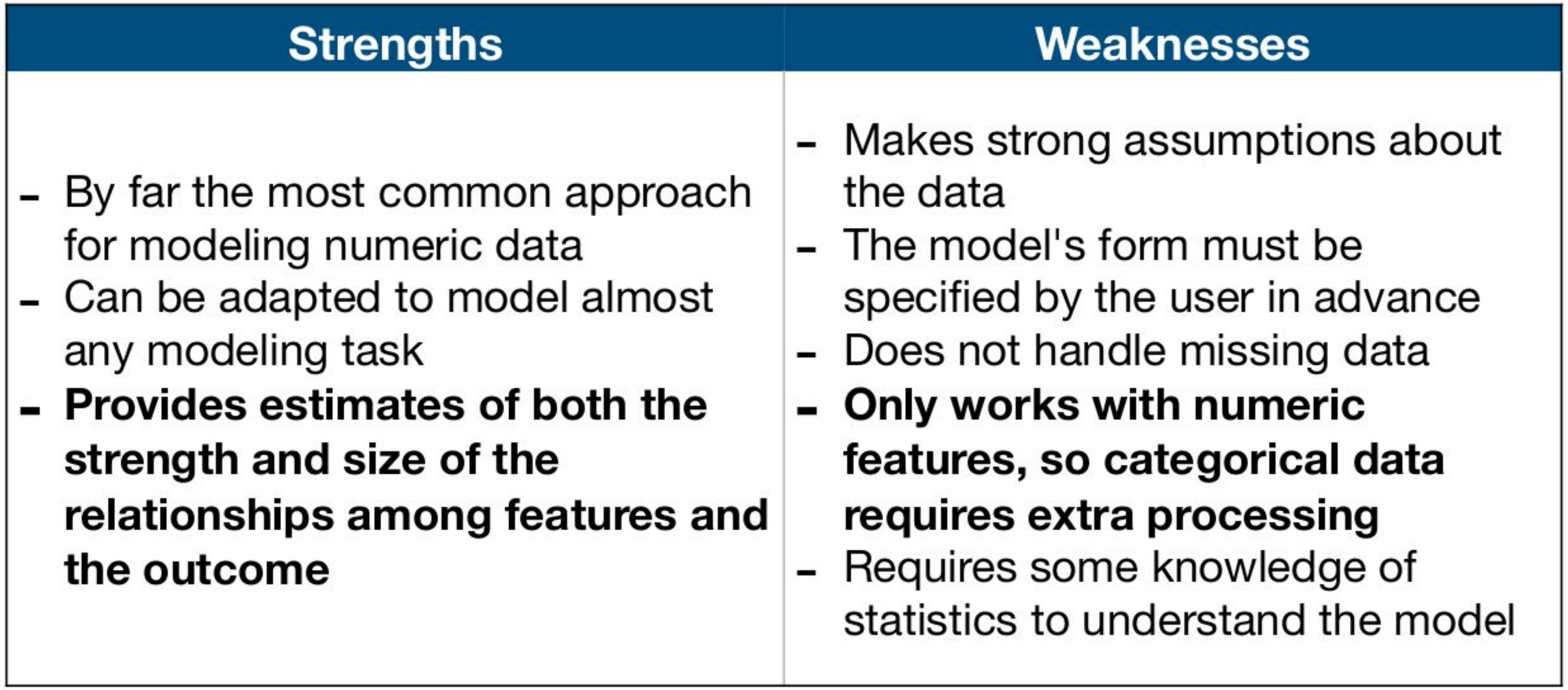
$$\underline{\mathrm{Cov}(x, y)} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{n}$$

$$\underline{\mathrm{Var}(x)} = \frac{\sum(x_i - \bar{x})^2}{n}$$

Multiple Linear Regression

Most real-world analyses have more than one independent variable.

Therefore, it is likely that you will be using multiple linear regression for most numeric prediction tasks.



Strengths	Weaknesses
- By far the most common approach for modeling numeric data - Can be adapted to model almost any modeling task - **Provides estimates of both the strength and size of the relationships among features and the outcome**	- Makes strong assumptions about the data - The model's form must be specified by the user in advance - Does not handle missing data - **Only works with numeric features, so categorical data requires extra processing** - Requires some knowledge of statistics to understand the model

Multiple Linear Regression

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_i x_i + \varepsilon$$

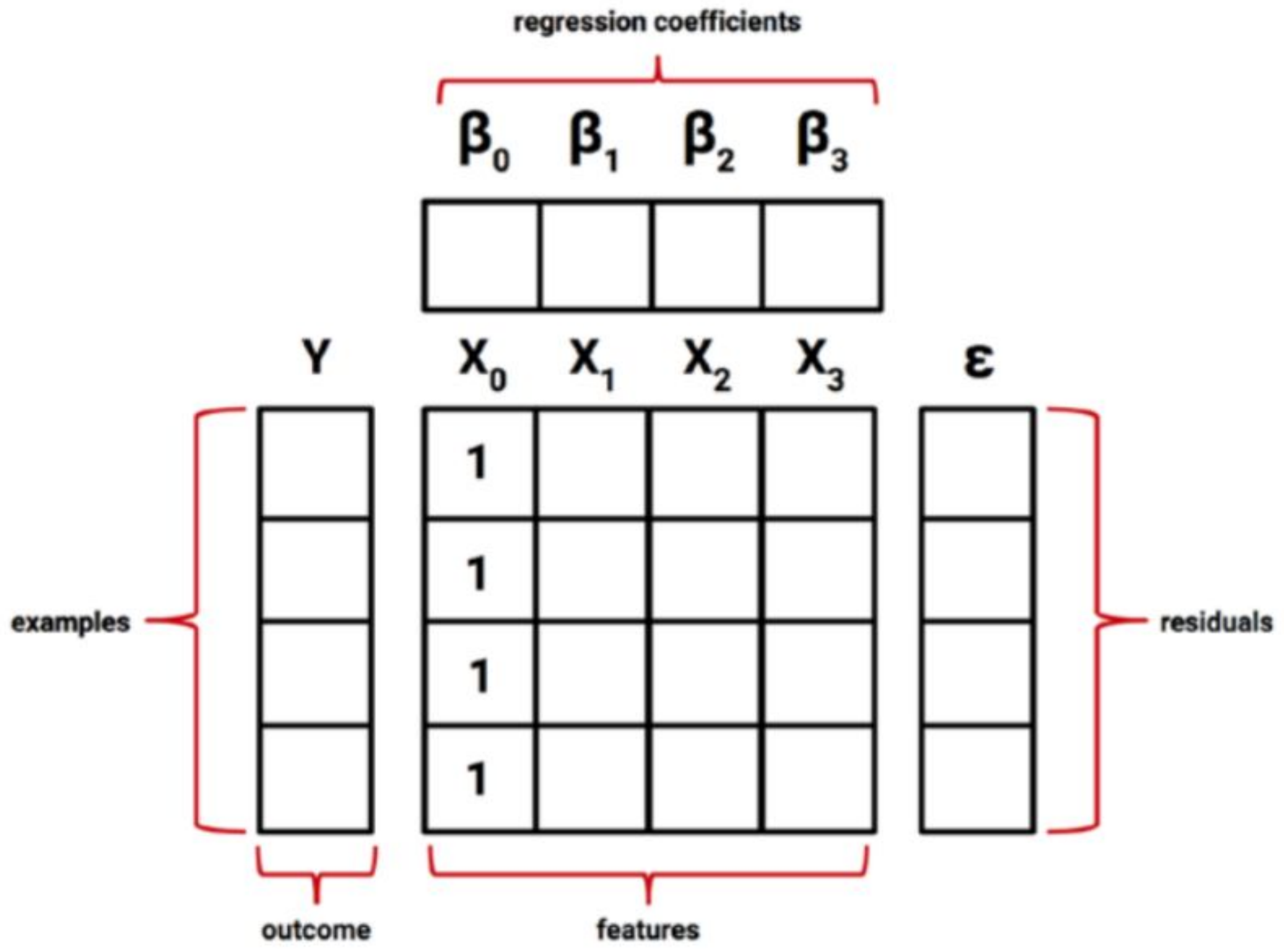
Let's consider for a moment the interpretation of the estimated regression parameters.

You will note that in the preceding equation, a **coefficient is provided for each feature. This allows each feature to have a separate estimated effect on the value of y.**

Multiple Linear Regression

In order to estimate the values of the regression parameters, each observed value of the dependent variable y must be related to the observed values of the independent x variables using the regression equation in the previous form. The following figure illustrates this structure:

$$\hat{y} = b_0x_0 + b_1x_1 + b_2x_2 + b_3x_3 + \varepsilon$$



Logistic Regression

Classification problem

Suppose that we are trying to predict the medical condition of a patient in the emergency room on the basis of her symptoms. In this simplified example, there are three possible diagnoses: **stroke, drug overdose,** and **epileptic seizure.** We could consider encoding these values as a quantitative response variable, Y , as follows:

$$Y = \begin{cases} 1 & \text{if } \texttt{stroke}; \\ 2 & \text{if } \texttt{drug overdose}; \\ 3 & \text{if } \texttt{epileptic seizure}. \end{cases}$$

Why not Linear regression?

- We want a Qualitative response instead of a Quantitative response
- Estimate class probabilities, not a numeric result

Why not Linear regression?

Using Linear Regression, least squares could be used to fit a linear regression model to predict Y on the basis of a set of predictors X1, . . ., Xp.

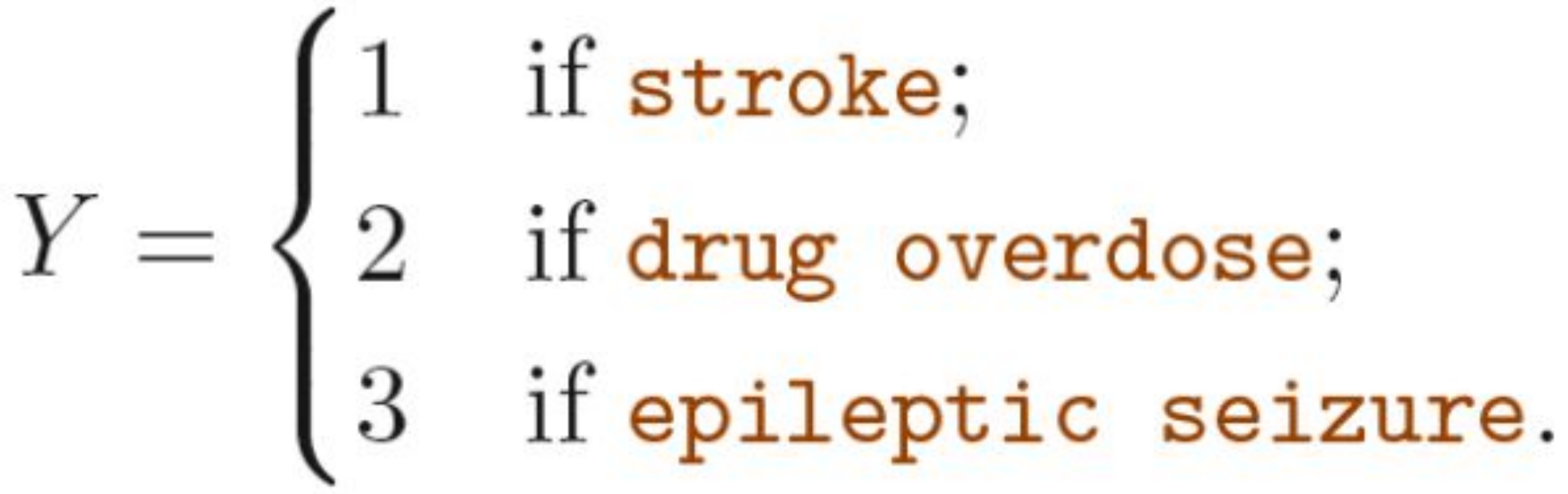
Unfortunately, this coding implies an **ordering on the outcomes**, putting drug overdose in between stroke and epileptic seizure, and insisting that the difference between stroke and drug overdose is the same as the difference between drug overdose and epileptic seizure

$$Y = \begin{cases} 1 & \text{if } \texttt{stroke}; \\ 2 & \text{if } \texttt{drug overdose}; \\ 3 & \text{if } \texttt{epileptic seizure}. \end{cases}$$

Why not Linear regression?

Using Linear Regression, least squares could be used to fit a linear regression model to predict Y on the basis of a set of predictors X1, . . ., Xp.

Unfortunately, this coding implies an **ordering on the outcomes**, putting drug overdose in between stroke and epileptic seizure, and insisting that the difference between stroke and drug overdose is the same as the difference between drug overdose and epileptic seizure



$$Y = \begin{cases} 1 & \text{if } \texttt{stroke}; \\ 2 & \text{if } \texttt{drug overdose}; \\ 3 & \text{if } \texttt{epileptic seizure}. \end{cases}$$

Example: Probability of Default

We have a data set containing information on ten thousand customers. The aim here is to predict which customers will default on their credit card debt.

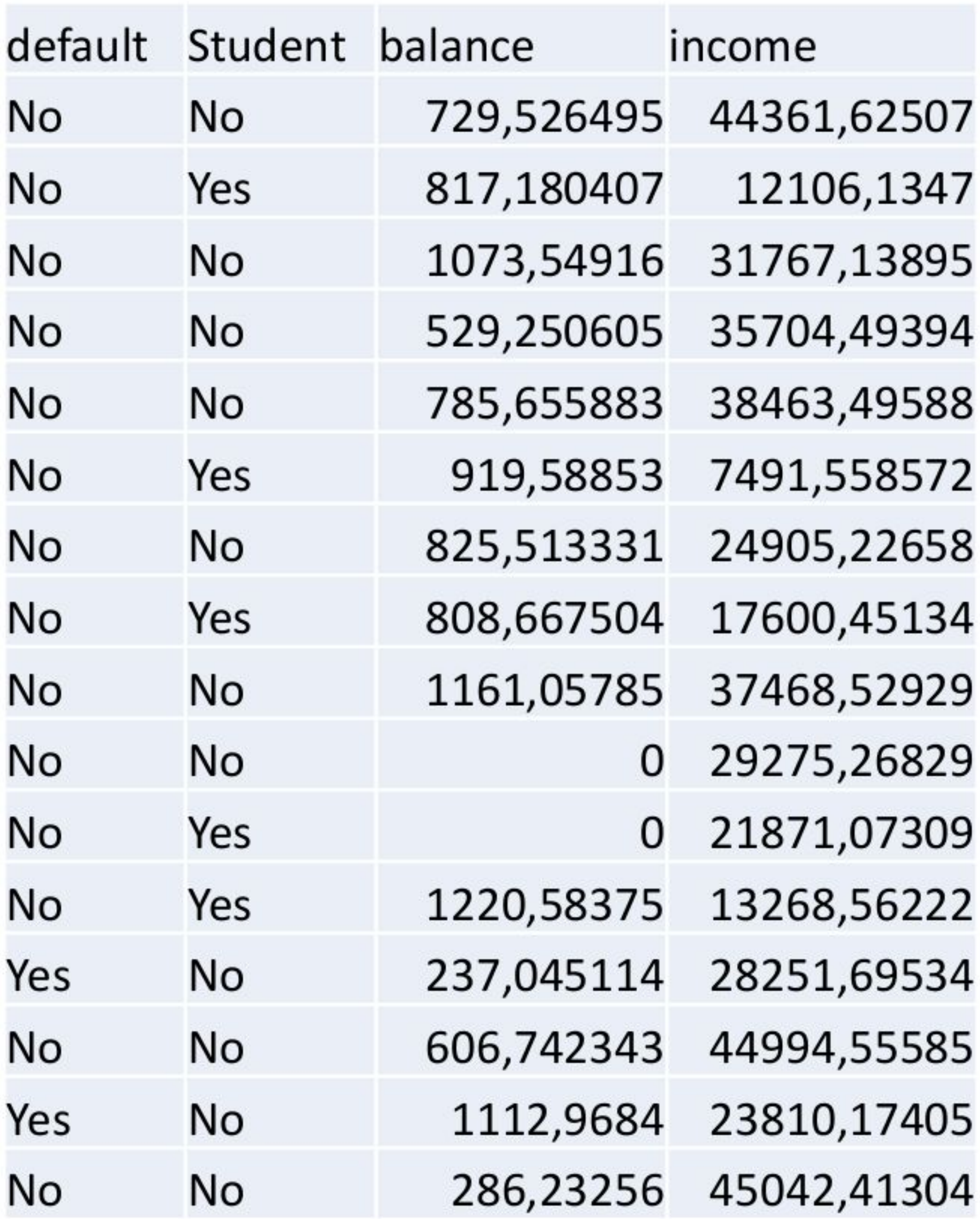
There are 4 variables:

default: A factor with levels No and Yes indicating whether the customer defaulted on their debt.

student: A factor with levels No and Yes indicated whether the customer is a student.

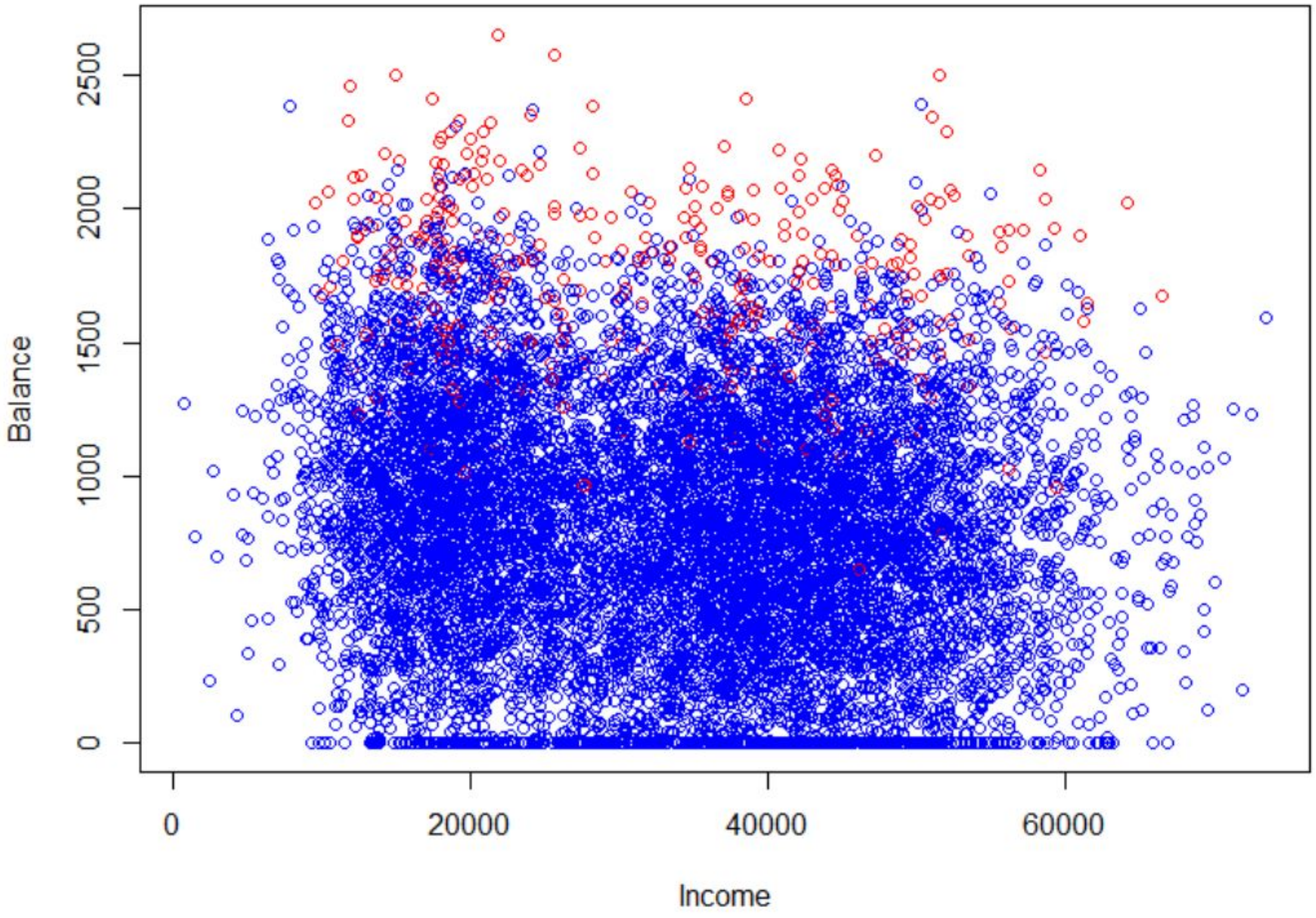
balance: The average balance that the customer has remaining on their credit card after making their monthly payment.

income: Income of the customer

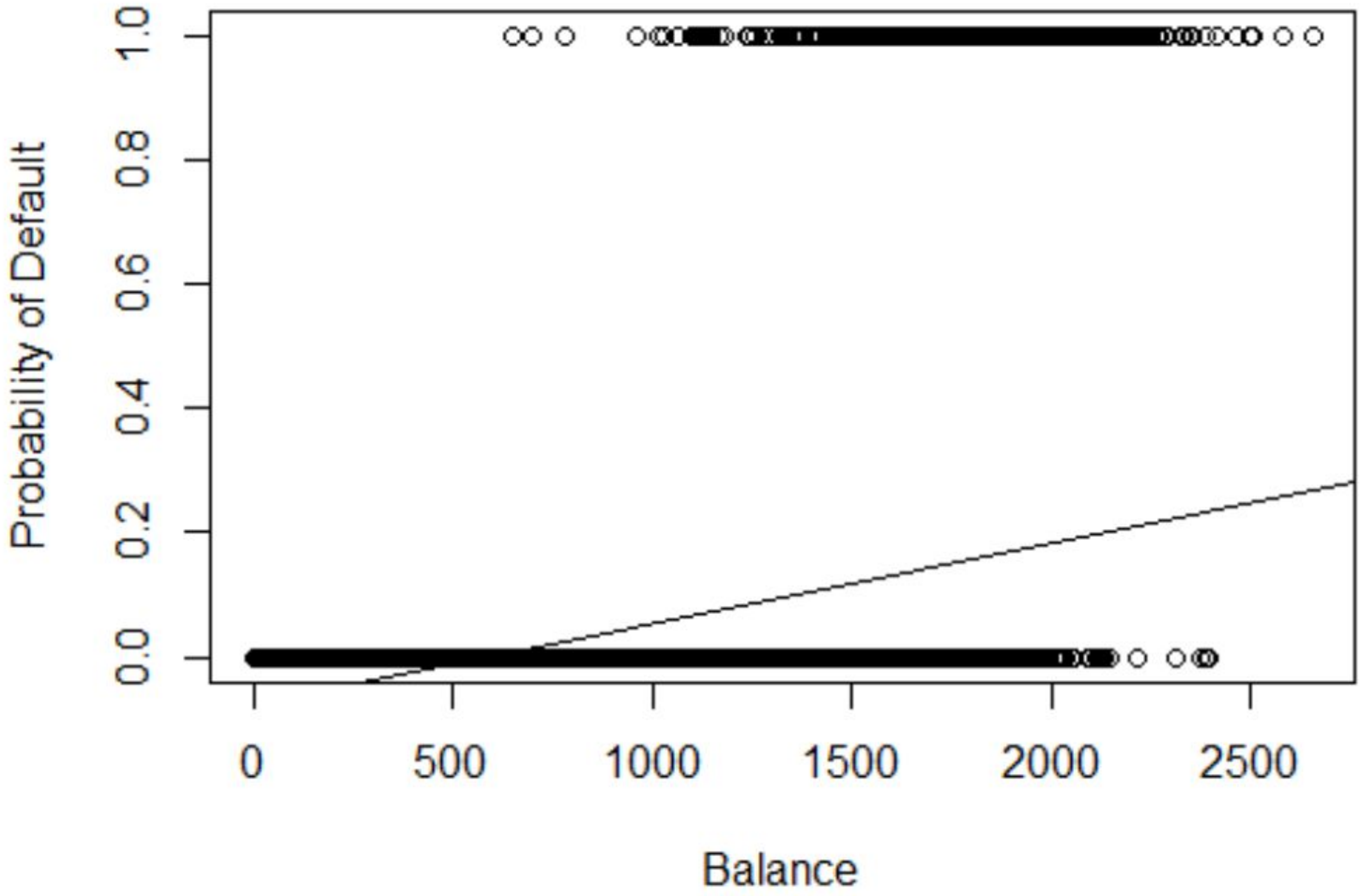


default	Student	balance	income
No	No	729,526495	44361,62507
No	Yes	817,180407	12106,1347
No	No	1073,54916	31767,13895
No	No	529,250605	35704,49394
No	No	785,655883	38463,49588
No	Yes	919,58853	7491,558572
No	No	825,513331	24905,22658
No	Yes	808,667504	17600,45134
No	No	1161,05785	37468,52929
No	No	0	29275,26829
No	Yes	0	21871,07309
No	Yes	1220,58375	13268,56222
Yes	No	237,045114	28251,69534
No	No	606,742343	44994,55585
Yes	No	1112,9684	23810,17405
No	No	286,23256	45042,41304

Example: Probability of Default



Example: Probability of Default using LR



If we build a linear regression model on the default, we get negative probabilities of default.

Logistic Regression

Instead of modelling the target variable ‘Probability of Default’ directly, we’re interested in estimating the probabilities that X belongs to each category (Yes, No) of Default.

Let’s revise the formula of linear regression to understand how logistic regression works:

$$p(X) = \beta_0 + \beta_1 X.$$

And also have a look at the logistic function, which takes any input and the output will always be between 0 and 1, where e ~ 2.71828 is a mathematical constant (Euler’s number).

$$\boldsymbol{\sigma}(x) = \frac{e^x}{e^x + 1}$$

Logistic Regression

Logistic Regression uses the form,

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}.$$

And with a bit of rearrangement we get,

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0+\beta_1 X}.$$

This value is known as odds, and can take up any value from 0 to ∞. Value of odds very close to 0 and ∞ indicate very low and very high probabilities of default, respectively.

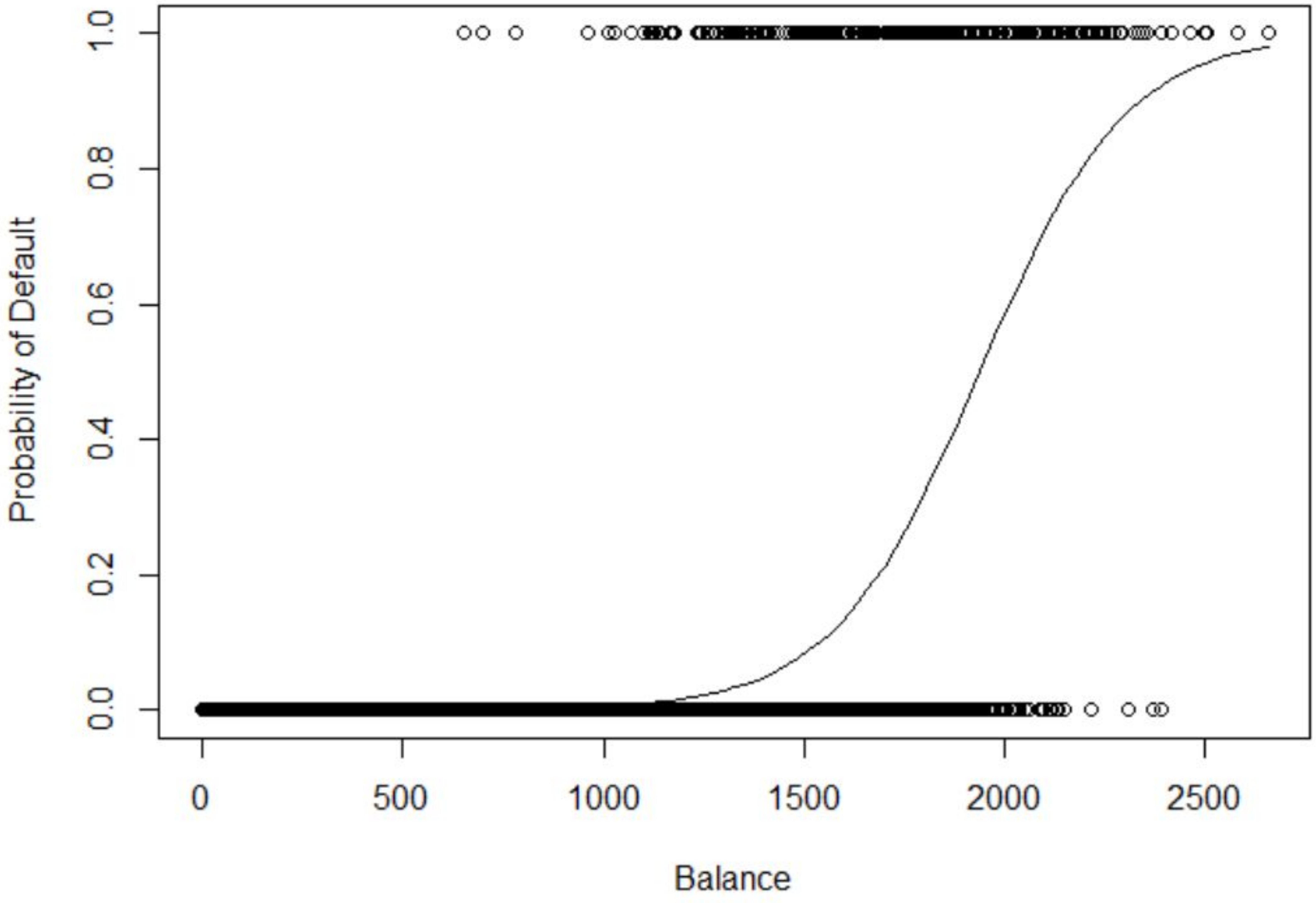
Logistic Regression

Taking log on both sides of the odds function gives us the **log odds** or the logit transformation of p(X).

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X.$$

Therefore we see that the logistic regression model has a logit that is linear in X. Logistic regression ensures that our estimate for p(X) lies between 0 and 1.

Logistic Regression



Logistic Regression Coefficients

Unlike linear regression where we use least squares estimation to estimate the regression coefficients, in logistic regression we use maximum likelihood to estimate the parameters.

$$\ell(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'})).$$

This **likelihood** gives the probability of the observed zeros and ones in the data. **We pick β0 and β1 to maximize the likelihood of the observed data.**

Predicting with Logistic Regression

We can predict the target variable values with a logistic regression model using this formula:

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}.$$

Logistic Regression **Predictions**

What is our estimated probability of default for someone with a balance of $1000?

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0+\hat{\beta}_1 X}}{1+e^{\hat{\beta}_0+\hat{\beta}_1 X}} = \frac{e^{-10.6513+0.0055\times 1,000}}{1+e^{-10.6513+0.0055\times 1,000}} = 0.00576$$

What about with a balance of $2000?

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0+\hat{\beta}_1 X}}{1+e^{\hat{\beta}_0+\hat{\beta}_1 X}} = \frac{e^{-10.6513+0.0055\times 2000}}{1+e^{-10.6513+0.0055\times 2000}} = 0.586$$

Logistic Regression **Predictions**

With student as a predictor, let's predict the probability of default if the person is a student.

$$\widehat{\Pr}(\texttt{default=Yes}|\texttt{student=Yes}) = \frac{e^{-3.5041+0.4049\times 1}}{1+e^{-3.5041+0.4049\times 1}} = 0.0431$$

And if the person is not a student?

$$\widehat{\Pr}(\texttt{default=Yes}|\texttt{student=No}) = \frac{e^{-3.5041+0.4049\times 0}}{1+e^{-3.5041+0.4049\times 0}} = 0.0292$$

Multinomial Logistic Regression

Logistic Regression is easily generalized to more than two classes.

$$\Pr(Y = k|X) = \frac{e^{\beta_{0k}+\beta_{1k}X_1+\ldots+\beta_{pk}X_p}}{\sum_{\ell=1}^{K} e^{\beta_{0\ell}+\beta_{1\ell}X_1+\ldots+\beta_{p\ell}X_p}}$$

Here there is a linear function for each class, but in this case it is better to figure out which **“independent” variables** are a **linear combination of the others.**

We can then model the distribution of X in each of the classes separately and then use **Bayes Theorem to obtain Pr(Y|X)**

Multinomial Logistic Regression

Now let's consider the problem of predicting a binary response using multiple predictors.

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

Where X = (X1 , X2 , ..., XP) are p predictors

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

More on Evaluation Methods later...

Today’s exercises

- You have been hired in a new company. You are making way above the average spanish salary, only have to work 4 days a week from home or the office -to which you can bring your pet- and the meals are provided for, together with childcare, and a gym membership.
- There is only one problem. Your new boss doesn’t trust you. After all, he learned Machine Learning through a 5 year PhD…
- He asks that you code for a Linear Regression using Numpy with a dataset that he prepared manually. **Are you up for the challenge?**

Today’s exercises

- Libraries you can use: **numpy**, **pandas** and **matplotlib**
- The dataset is here
- If you complete this, your boss offers you to double down your income if you can also do one the following:
  - Implement PCA from scratch on the iris dataset
  - Review a couple of additional notebooks in PCA
  - Implement logistic regression from scratch with a dataset of your choice that contains categorical variables

Yesterday’s feedback

- Jan teacher rating decreases when he’s slept for 3 hours
  - *I’m literally ashamed to watch the lecture.*
- We have 4 teachers lined up for August, so this won’t happen again ;)
- Why are there no exercises on hierarchical clusters?
  - Updated on github, now live.
- Getting an overview of the week at the start of the week
  - Makes sense, adding it in the next slide.
- *“yesterday during the explanation I got lost so I couldn't follow well what Jan was explaining and later I had to revise all”*
  - This should **not happen**. We can do a PCA Saturday next weekend (not this one)

What we will do this week

- **Monday: Clustering**
  - Hierarchical Algorithms
  - Partitional Algorithms
- **Tuesday**
  - Linear Regression
  - Logistic Regression
- **Wednesday**
  - Gradient descent and learning
  - Evaluation methods
  - Knn and Naïve Bayes

What we will do this week

- **Thursday (Alessia)**
  - Decision Trees
  - Ensemble Methods
  - Random Forest
- **Friday (Alessia)**
  - SVMachines
  - Data preprocessing
  - Whatever else we're missing on ML that is essential
- **Saturday**
  - Day off as promised