Introduction to Supervised Learning

Supervised Learning (COMP0078)

Carlo Ciliberto

AI Centre, Department of Computer Science
University College London

Course Information

1. When
   - Thursday 11-12am [Office Hours] (On request)
   - Thursday 4-5 pm [Q+A with TA] Delving deeper on a topic + some exercises.
   - Fridays 11am-2pm [Flipped Classroom]

2. TAs:
   - Antonin Schrab
   - Ming Liang Ang
3. Instructors:
   - Carlo Ciliberto (me)
   - John Shawe-Taylor

Flipped Classroom Structure

1. Watch video lectures at your pace during the week.

Flipped Classroom Structure

1. Watch video lectures at your pace during the week.
2. Submit questions:
    - To this form to be discussed next Friday.

Flipped Classroom Structure

1. Watch video lectures at your pace during the week.
2. Submit questions:
    - To this form to be discussed next Friday.
    - But, if you are **really stuck** (or we haven't answered in class)
        - Questions that could benefit everyone: **Moodle forum**
        - Otherwise: `sl-support@cs.ucl.ac.uk`

Flipped Classroom Structure

1. Watch video lectures at your pace during the week.
2. Submit questions:
   - To this form to be discussed next Friday.
   - But, if you are **really stuck** (or we haven't answered in class)
     - Questions that could benefit everyone: **Moodle forum**
     - Otherwise: `sl-support@cs.ucl.ac.uk`
3. Thursday: TA Session
   - TAs will tackle some problems in class.
   - Oportunity to interact / ask questions.

Flipped Classroom Structure

1. Watch video lectures at your pace during the week.
2. Submit questions:
   - To this form to be discussed next Friday.
   - But, if you are **really stuck** (or we haven't answered in class)
     - Questions that could benefit everyone: **Moodle forum**
     - Otherwise: `sl-support@cs.ucl.ac.uk`
3. Thursday: TA Session
   - TAs will tackle some problems in class.
   - Oportunity to interact / ask questions.
4. Friday Class
   - Q&A: I will ask you questions / answer your submitted questions.

     Note. We are $150+$, please send your questions in advance.

Flipped Classroom Structure

1. Watch video lectures at your pace during the week.
2. Submit questions:
   - To this form to be discussed next Friday.
   - But, if you are **really stuck** (or we haven't answered in class)
     - Questions that could benefit everyone: **Moodle forum**
     - Otherwise: `sl-support@cs.ucl.ac.uk`
3. Thursday: TA Session
   - TAs will tackle some problems in class.
   - Oportunity to interact / ask questions.
4. Friday Class
   - Q&A: I will ask you questions / answer your submitted questions.
     Note. We are $150+$, please send your questions in advance.
   - I will propose a problem (either theoretical or practical) and will be available until the end of the class to ask questions on tackling them.

Flipped Classroom Structure

1. Watch video lectures at your pace during the week.
2. Submit questions:
   - To this form to be discussed next Friday.
   - But, if you are **really stuck** (or we haven't answered in class)
     - Questions that could benefit everyone: **Moodle forum**
     - Otherwise: `sl-support@cs.ucl.ac.uk`
3. Thursday: TA Session
   - TAs will tackle some problems in class.
   - Oportunity to interact / ask questions.
4. Friday Class
   - Q&A: I will ask you questions / answer your submitted questions.
     Note. We are $150+$, please send your questions in advance.
   - I will propose a problem (either theoretical or practical) and will be available until the end of the class to ask questions on tackling them.
   - [Not mandatory] If suitable, I will ask you to send me (anonymously or not as you prefer) some results (e.g. plots) to discuss them live.

Assessment

- **Assessments Breakdown:**
  - Coursework 1 (20%)
  - Coursework 2 (20%)
  - Exam (60%)

- **Coursework:**
  - ~ 1 month to complete.
  - Both theory and practice (= code implementation)
  - Penalty on late submissions.

- **Pass marks:**
  - MSc ML/CSML/DSML: must obtain weighted marks above 50%
  - MENG and other degrees: check with your programme coordinator.

.

Prerequisites

Today we will get a flavor of the prerequisite we will assume:

- Calculus (real-valued functions, limits, derivatives, Taylor series, integrals,...)
- Elements of probability theory (random variables, expectation, variance, conditional probabilities, Bayes rule,...)
- Fundamentals of linear algebra (vectors, angles, matrices, eigenvectors/eigenvalues,...),
- A bit of optimization theory (convex functions, Lagrange multipliers)

Antonin will give a Math "boothcamp" on the first TA session (Next Thursday).

Readings / References

Useful References

- *Learning theory from first principles.*, F. Bach, MIT press, (2024).
- *Machine learning: a probabilistic perspective*, Murphy, K.P., MIT press (2012).
- *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, T. Hastie, R. Tibshirani, & J. Friedman, Springer (2009)
- *Understanding Machine Learning from Theory to Algorithms*, S. Shalev-Shwartz and S. Ben-David, Cambridge University Press (2014)
- *Convex Optimization*, S. Boyd and L. Vandenberghe (2004)
- *Kernel Methods for Pattern Analysis*, J. Shawe-Taylor, and N. Cristianini, Cambridge University Press (2004)
- *Pattern Recognition and Machine Learning*, C. Bishop, Springer (2006)

Today's Plan: A Supervised Learning Feast

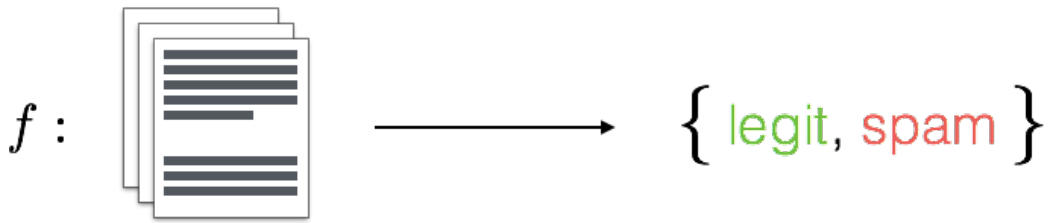
- Amuse Bouche: Supervised Learning - how and why?
- Antipasto: SPAM
- Primo: Least Squares
- Secondo: Generalization
- Dessert: No-free lunch theorem
- Bill

Amuse-bouche

Supervised Learning to the Rescue!

In many situations we **cannot** hand-code the solution of a problem. . .

- Too many "rules",
- Too many corner cases,
- Not a clear definition of the problem itself!

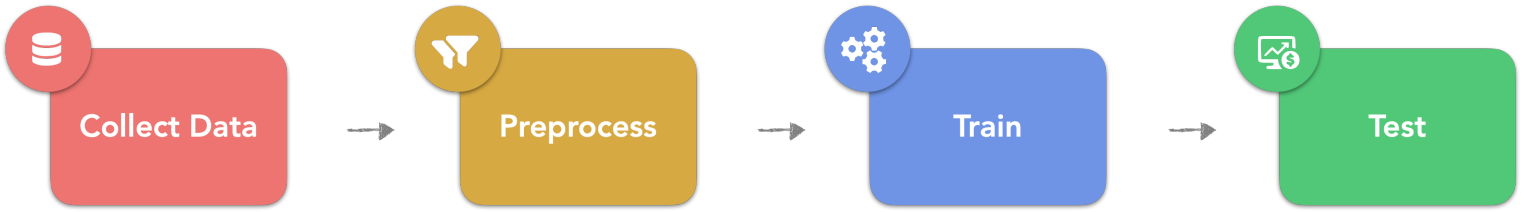


Supervised Learning answers this questions with a different approach:

- By learning the rules,
- without explicit pre-programming,
- from examples.

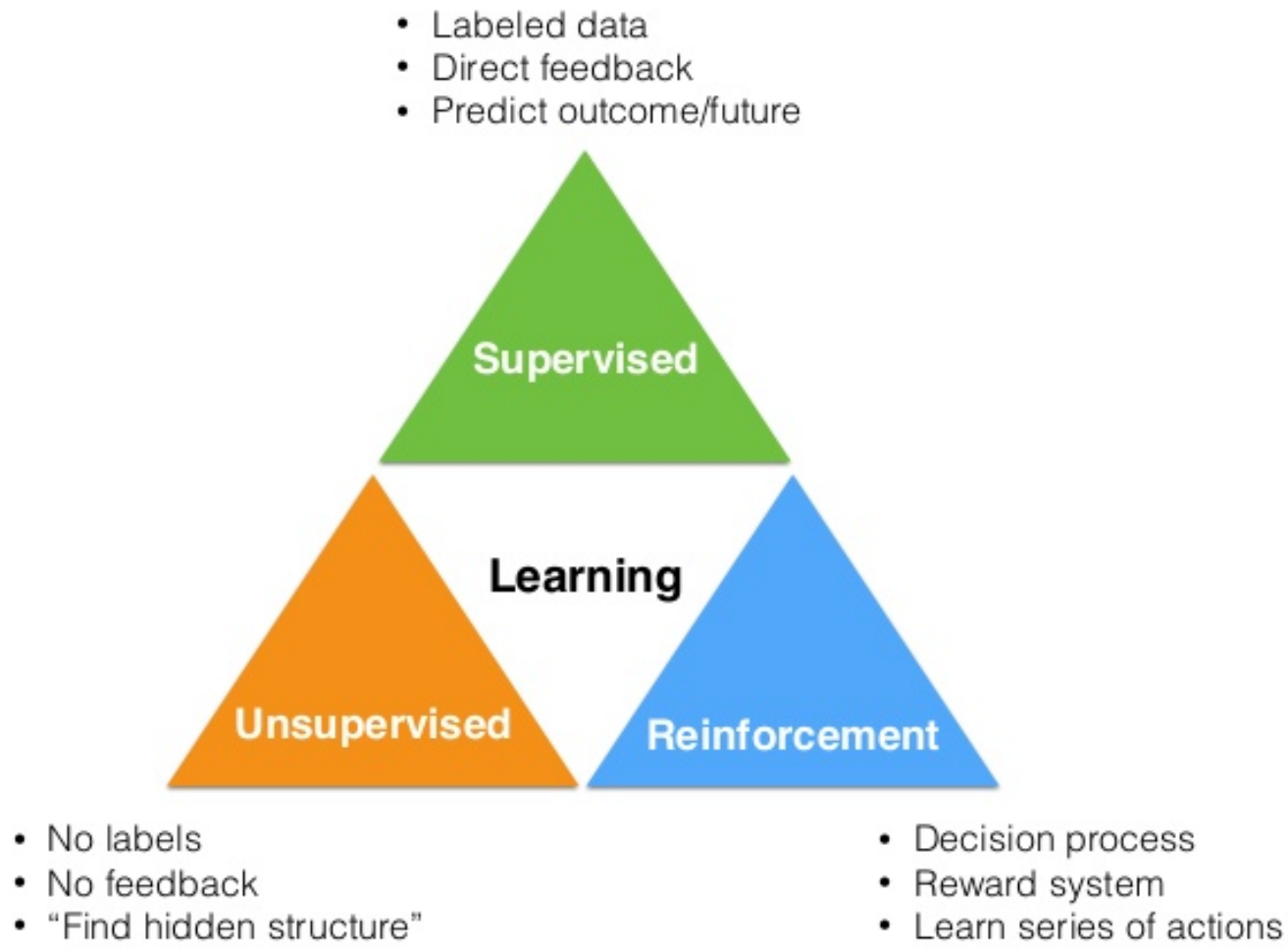
Supervised Learning Pipeline

We can summarize the **prototypical** learning pipeline as



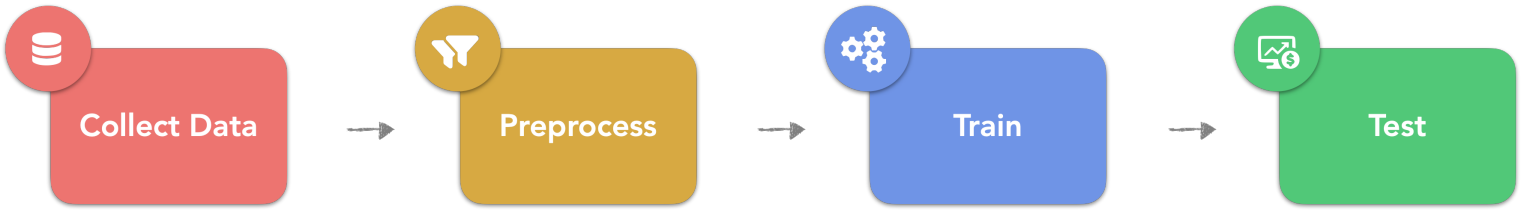
- Collect Data: to be **representative** of your problem.
- Pre-process it: to be easily **"digestible"** by your machine.
- Train a model: where the **learning** actually happens.
- Profit: Test/Apply the model to **new data!**

Taxonomy of Learning Paradigms



Supervised Learning - Practical Questions

When considering the SL pipeline, several practical questions arise:



- How to collect the data (so that it's useful for the task)?
- How do we "feed" the data to a machine (representation)?
- How can we design a training algorithm?
- How can we determine whether we trust a trained model?
- ...

Supervised Learning - Further Complications

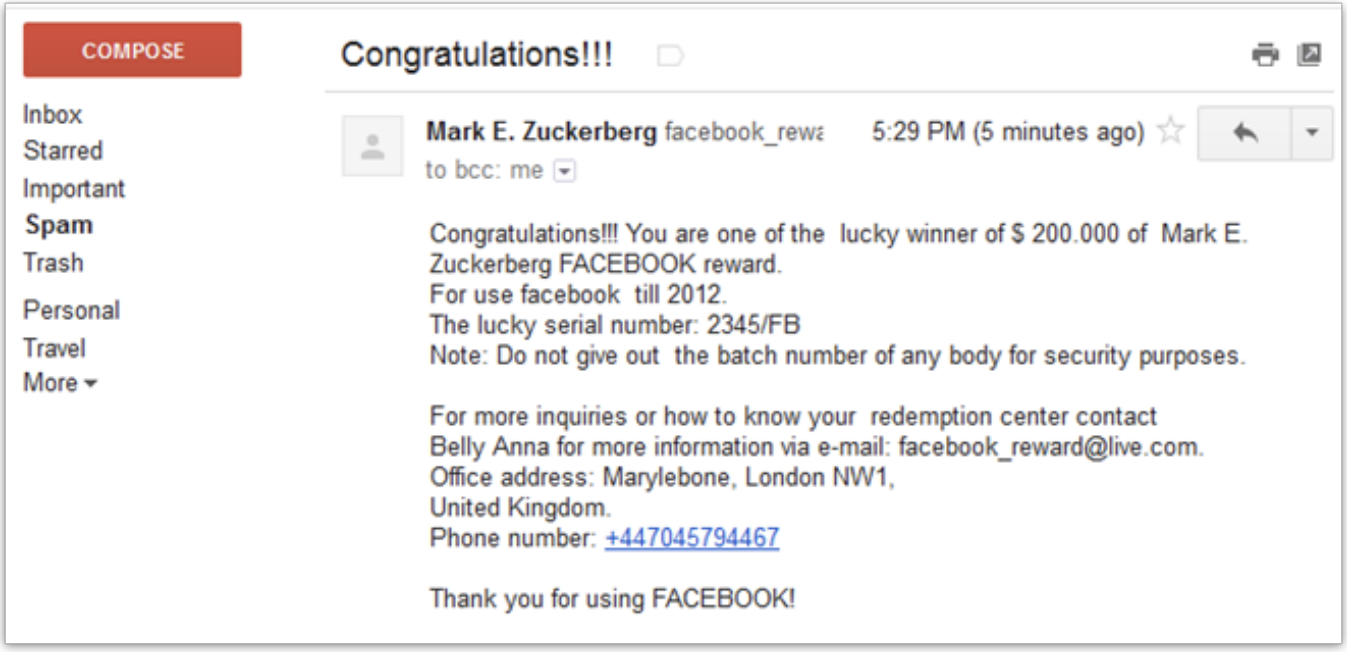
In theory, theory and practice are the same.
In practice, they are not.
– Albert Einstein

- Domain shift
- Noise (both in the input AND the output)
- Non-unique relationship between $x \mapsto y$, namely $y \sim \mathbb{P}(\cdot|x)$
- High-dimensional input with LOTs of irrelevant features.
- Need to incorporate prior knowledge (e.g. constraints).
- ...

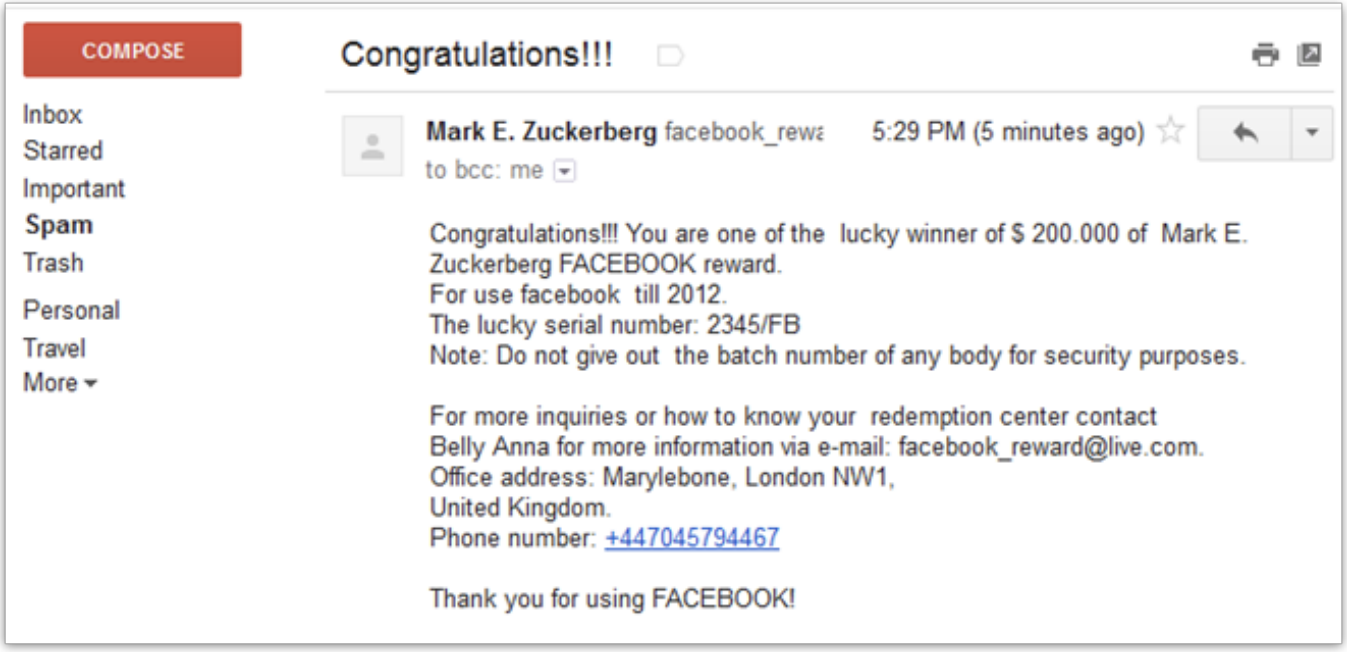
Rather than thinking in the abstract, let's address a concrete problem...


Antipasto

SPAM


COMPOSE
Inbox
Starred
Important
Spam
Trash
Personal
Travel
More
Congratulations!!!
Mark E. Zuckerberg facebook_rewa
5:29 PM (5 minutes ago)
to bcc: me
Congratulations!!! You are one of the lucky winner of $ 200.000 of Mark E. Zuckerberg FACEBOOK reward.
For use facebook till 2012.
The lucky serial number: 2345/FB
Note: Do not give out the batch number of any body for security purposes.
For more inquiries or how to know your redemption center contact
Belly Anna for more information via e-mail: facebook_reward@live.com.
Office address: Marylebone, London NW1,
United Kingdom.
Phone number: +447045794467
Thank you for using FACEBOOK!

SPAM



- How do we design a good spam detector?
- What kind of features make an e-mail. . . spam?

Supervised Learning - Practical Questions

When considering the SL pipeline, several practical questions arise:

- **How to collect the data (so that it's useful for the task)?**
- How do we "feed" the data to a machine (representation)?
- How can we design a training algorithm?
- How can we determine whether we trust a trained model?
- ...

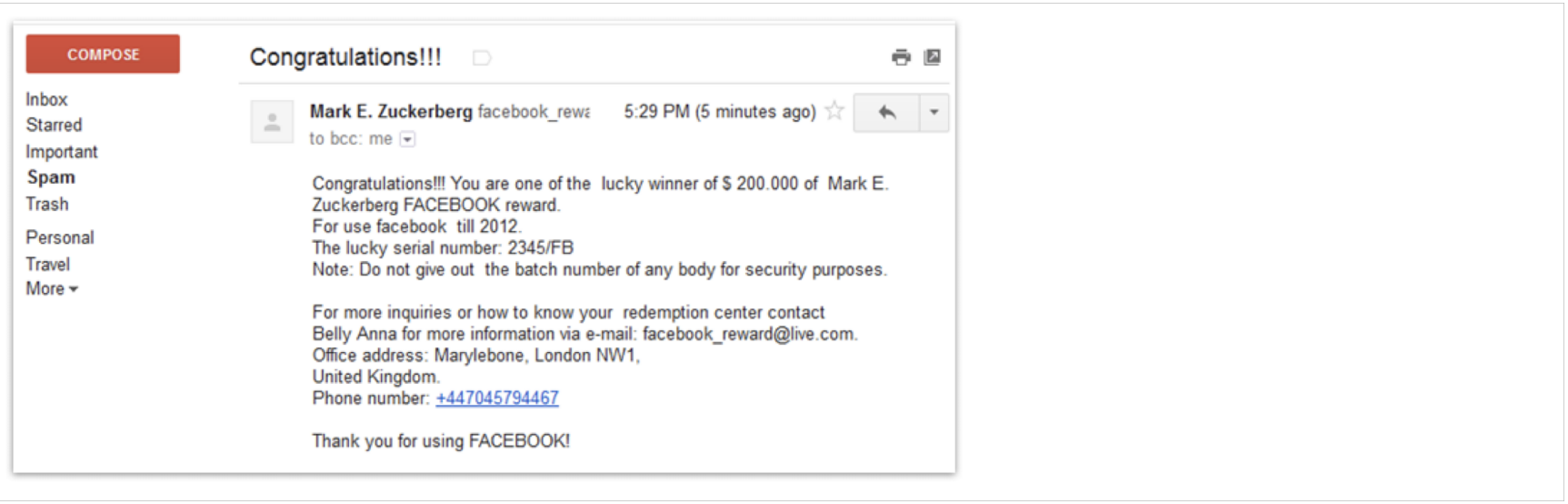
- Get many examples of spam and try to find a "rule"...

Supervised Learning - Practical Questions

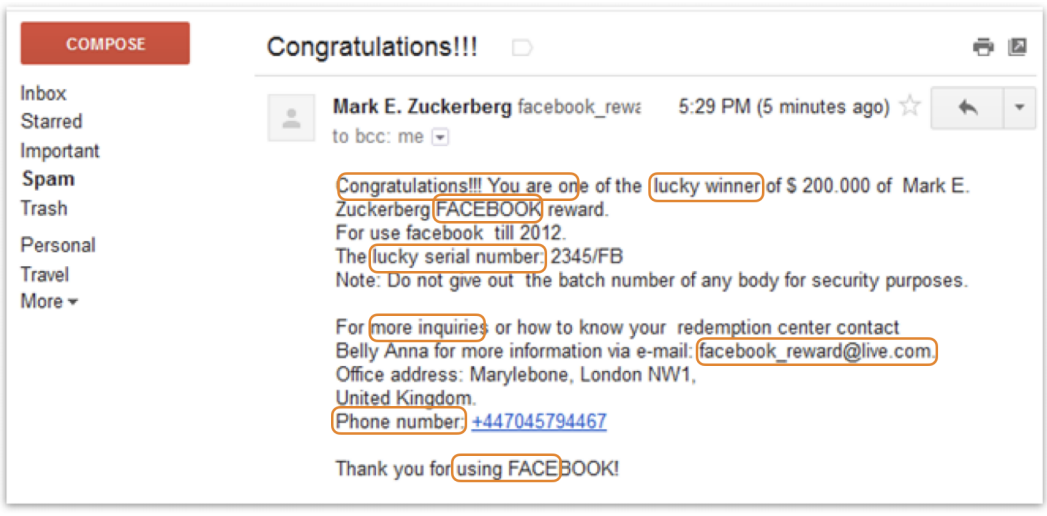
When considering the SL pipeline, several practical questions arise:

- How to collect the data (so that it's useful for the task)?
- **How do we "feed" the data to a machine (representation)?**
- How can we design a training algorithm?
- How can we determine whether we trust a trained model?
- ...

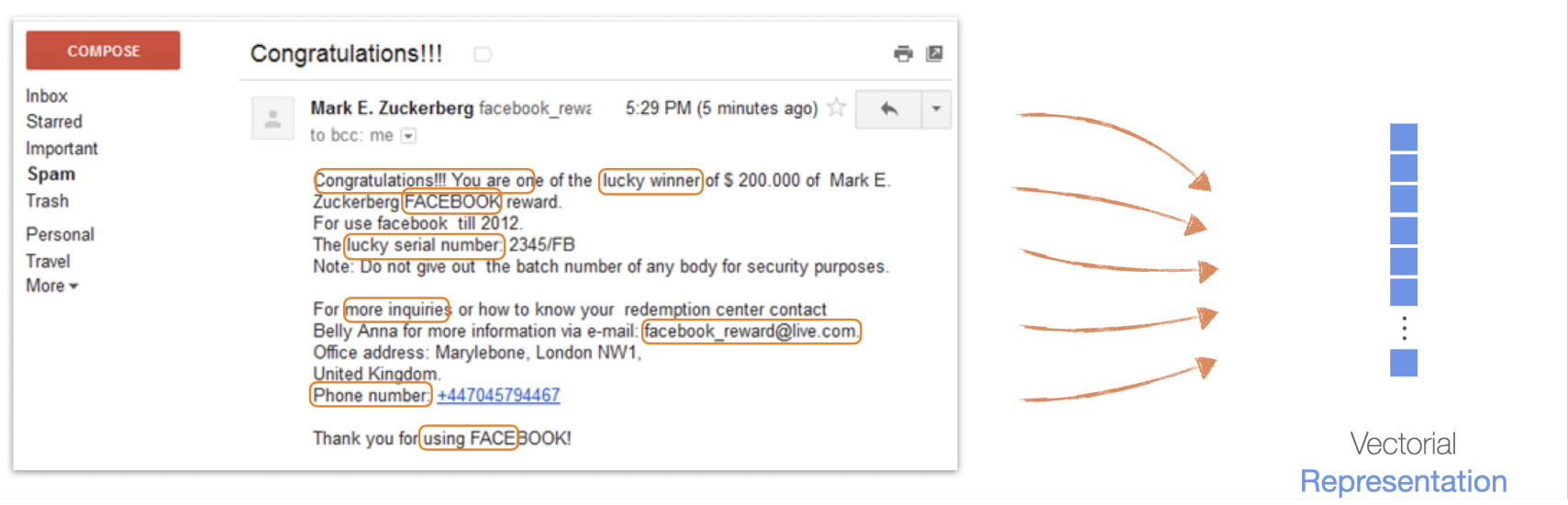
Feature Extraction



Feature Extraction



Feature Extraction

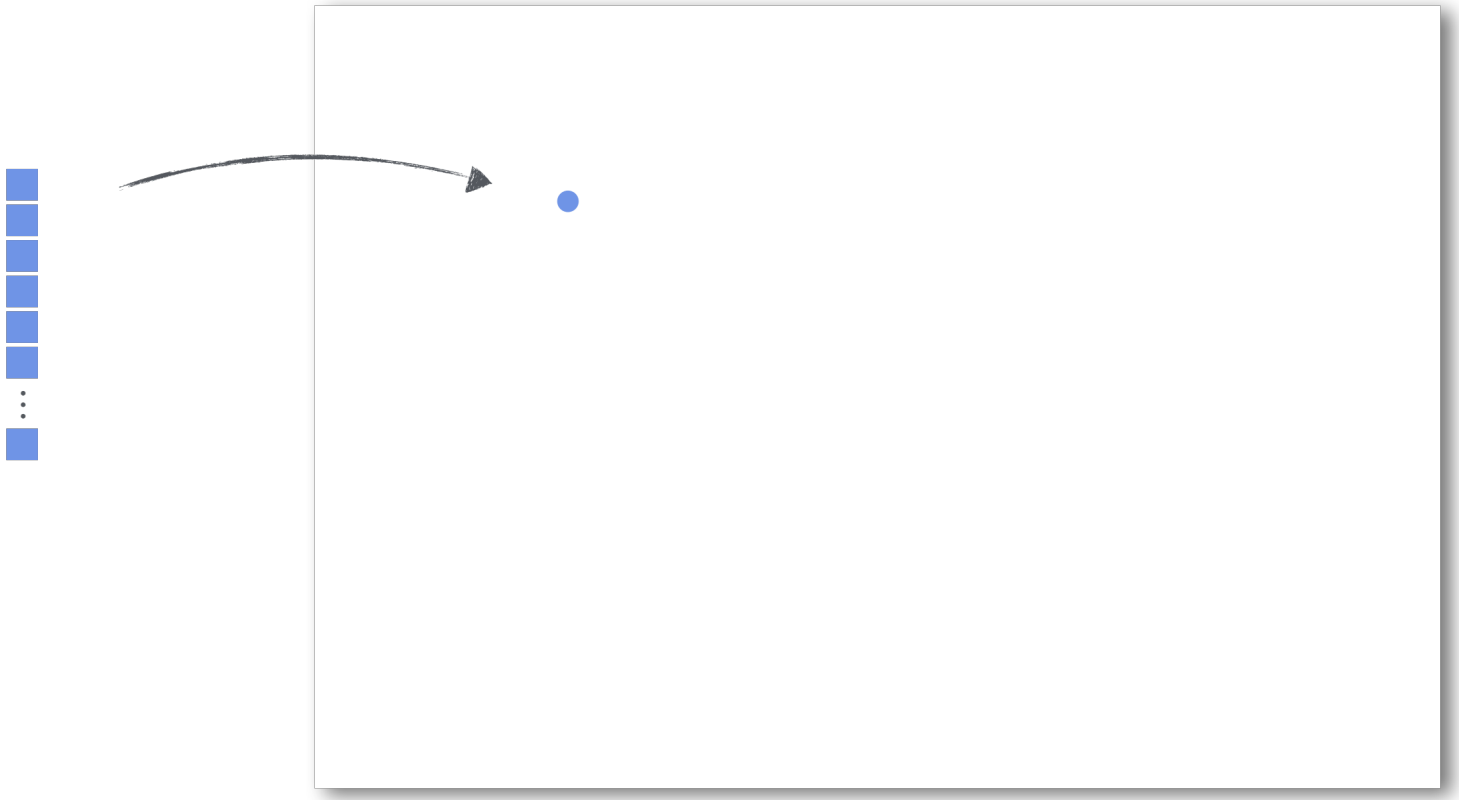


Supervised Learning - Practical Questions

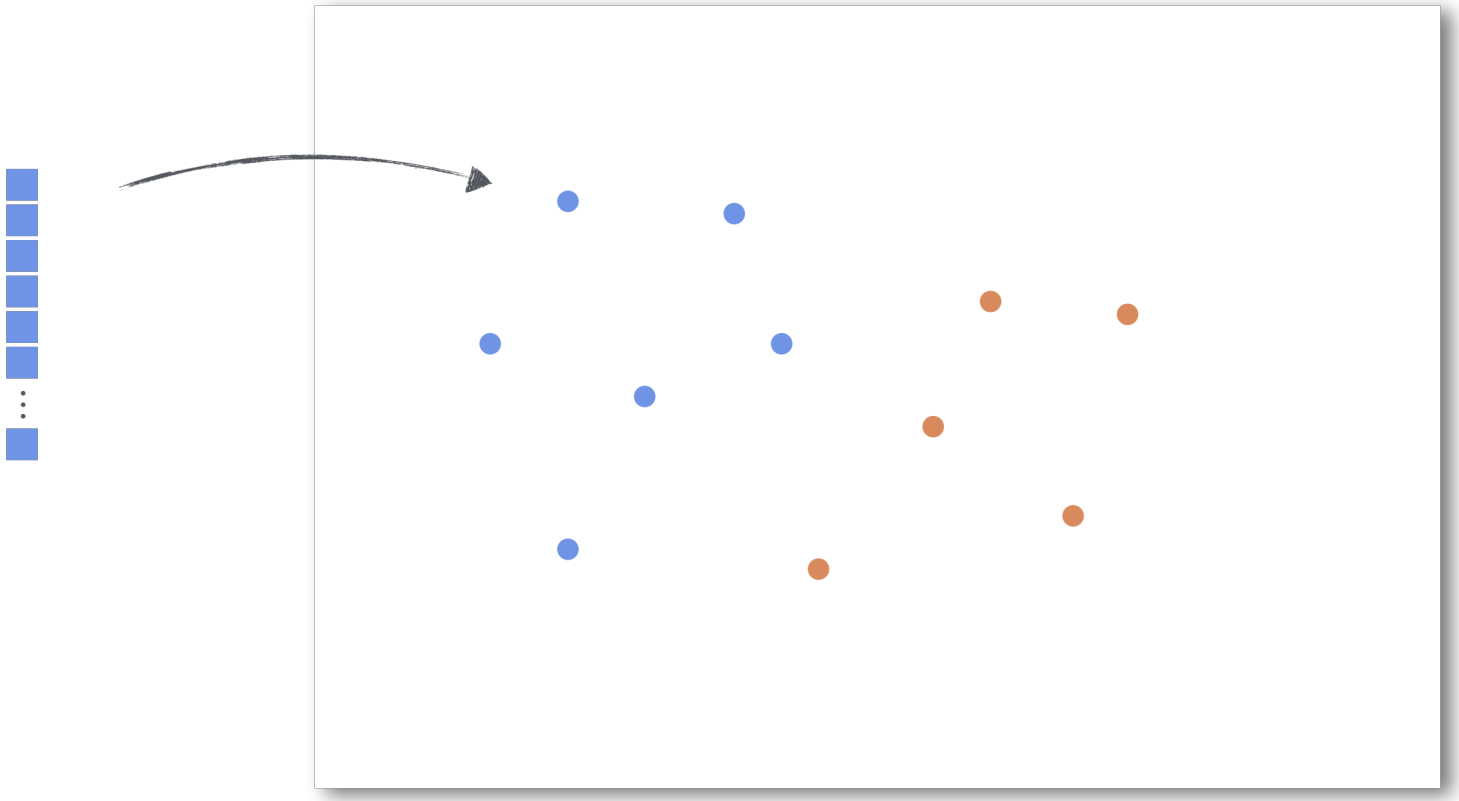
When considering the SL pipeline, several practical questions arise:

- How to collect the data (so that it's useful for the task)?
- How do we "feed" the data to a machine (representation)?
- **How can we design a training algorithm?**
- How can we determine whether we trust a trained model?
- ...

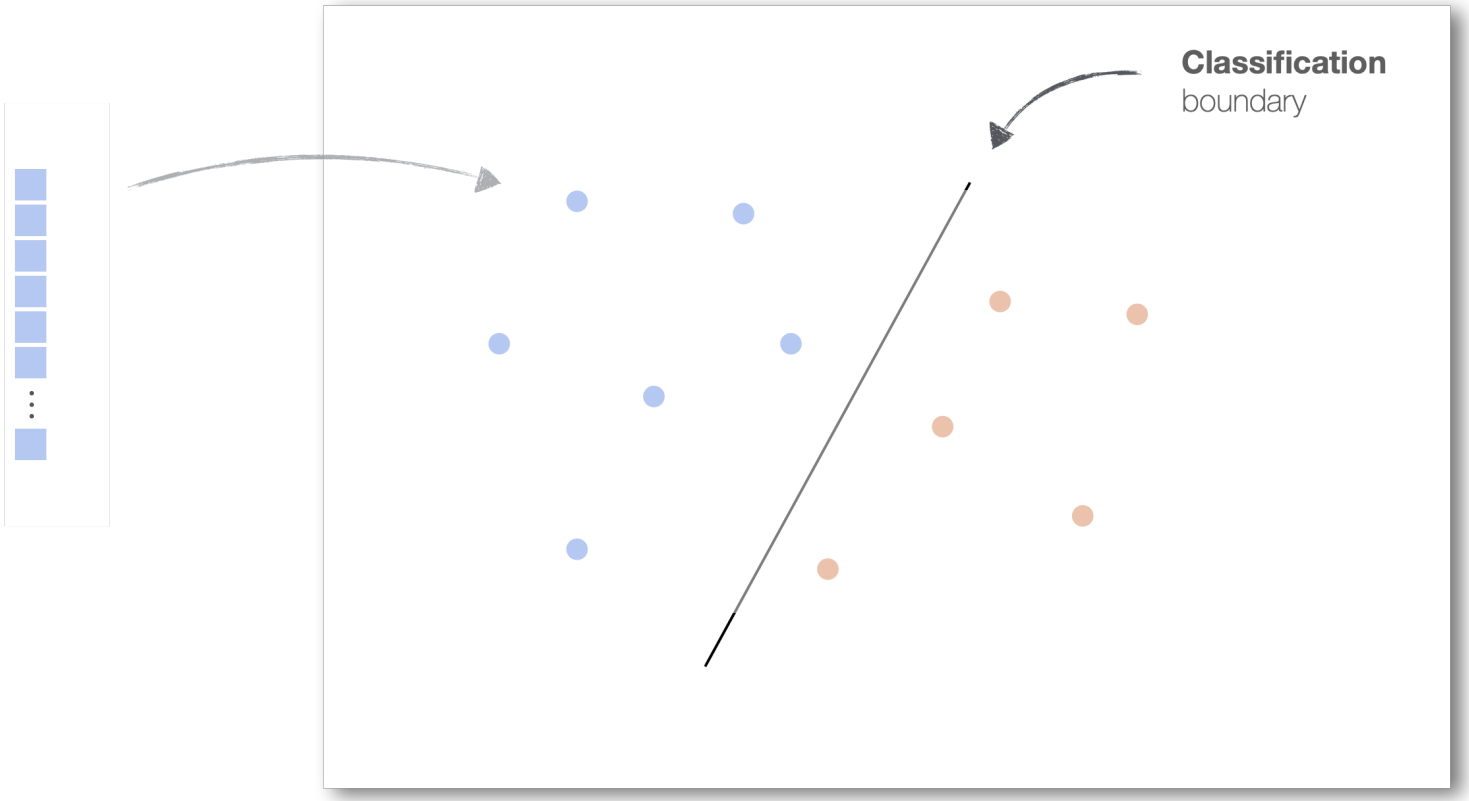
Geometry to the Rescue!



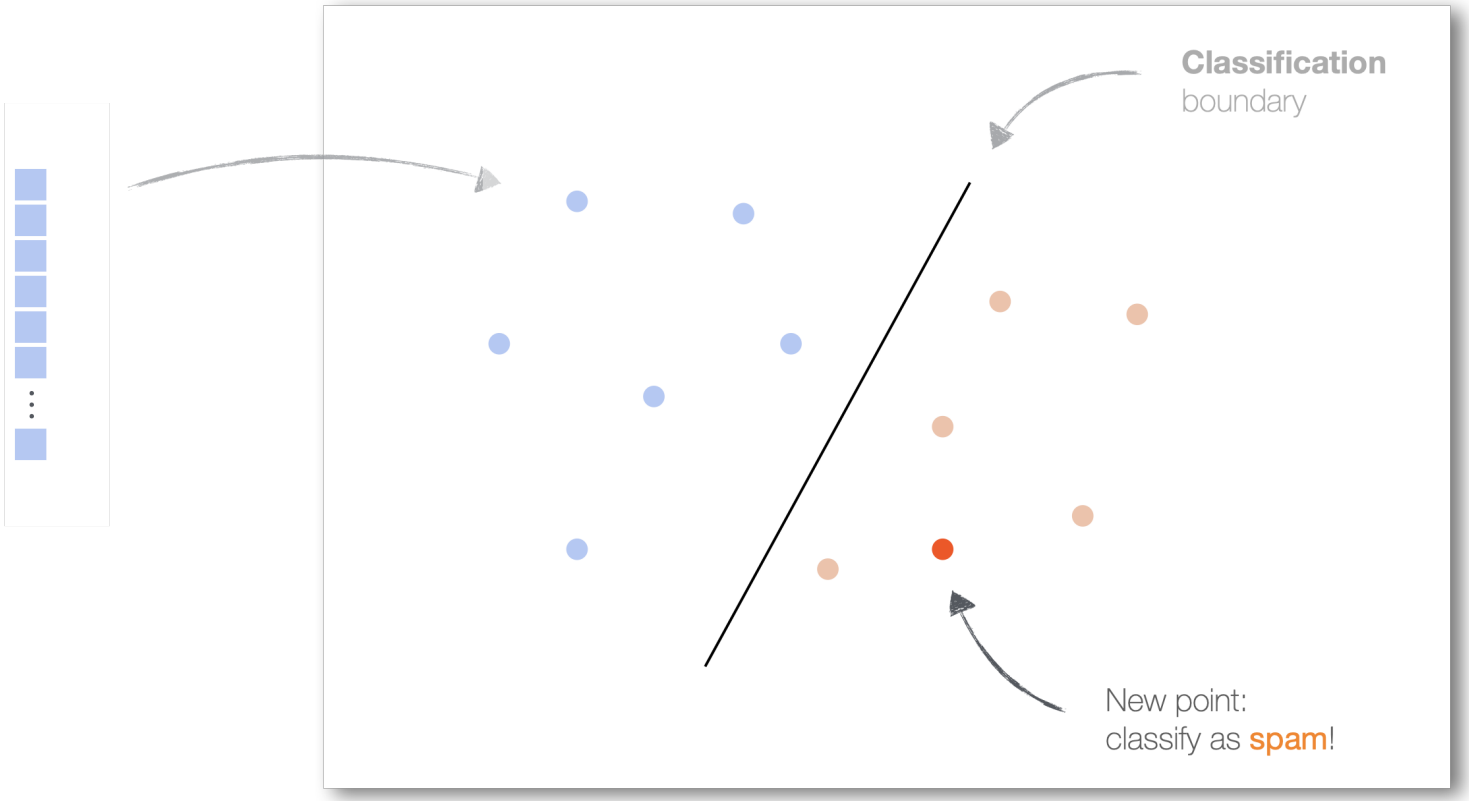
Geometry to the Rescue!



Geometry to the Rescue!



Geometry to the Rescue!

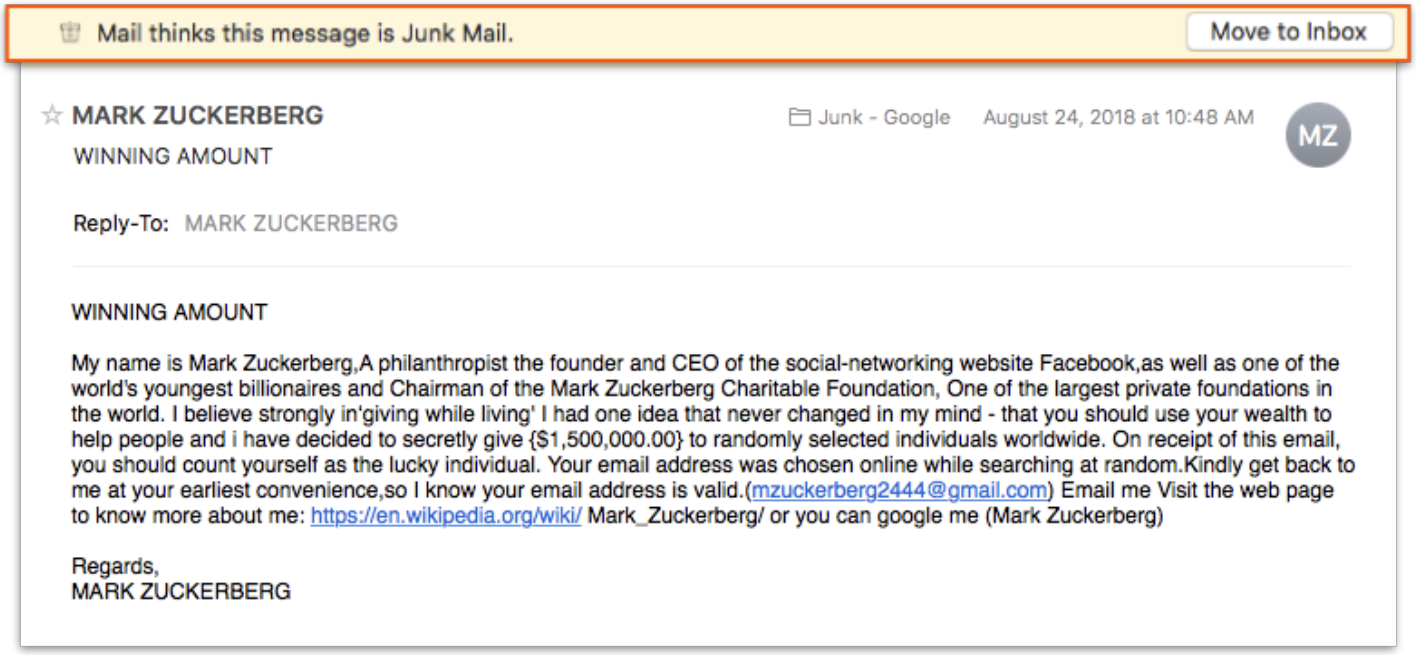


Supervised Learning - Practical Questions

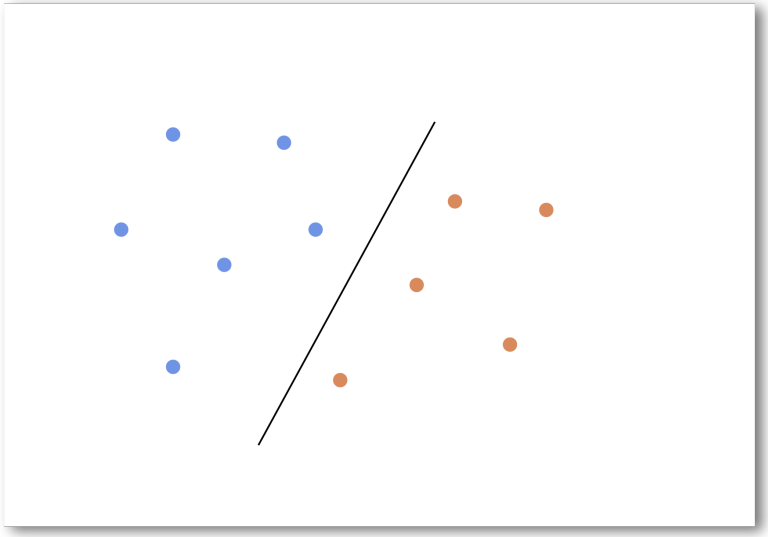
When considering the SL pipeline, several practical questions arise:

- How to collect the data (so that it's useful for the task)?
- How do we "feed" the data to a machine (representation)?
- How can we design a training algorithm?
- **How can we determine whether we trust a trained model?**
- ...

Finally. . .


Mail thinks this message is Junk Mail.
Move to Inbox
MARK ZUCKERBERG
Junk - Google
August 24, 2018 at 10:48 AM
MZ
WINNING AMOUNT
Reply-To: MARK ZUCKERBERG
WINNING AMOUNT
My name is Mark Zuckerberg,A philanthropist the founder and CEO of the social-networking website Facebook,as well as one of the world's youngest billionaires and Chairman of the Mark Zuckerberg Charitable Foundation, One of the largest private foundations in the world. I believe strongly in'giving while living' I had one idea that never changed in my mind - that you should use your wealth to help people and i have decided to secretly give {$1,500,000.00} to randomly selected individuals worldwide. On receipt of this email, you should count yourself as the lucky individual. Your email address was chosen online while searching at random.Kindly get back to me at your earliest convenience,so I know your email address is valid.(mzuckerberg2444@gmail.com) Email me Visit the web page to know more about me: https://en.wikipedia.org/wiki/ Mark_Zuckerberg/ or you can google me (Mark Zuckerberg)
Regards,
MARK ZUCKERBERG

Classification: where do you draw the line?



Classification: where do you draw the line?

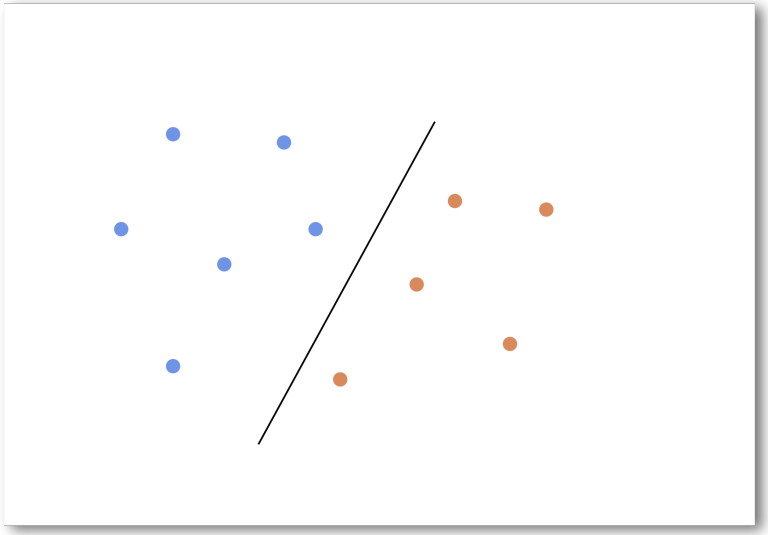
Setting

- Dataset $(x_i, y_i)_{i=1}^n$
- Model $f : \mathcal{X} \to \{-1, 1\}$
- Objective

$$L(f) = \sum_{i=1}^{n} \text{error}(f(x_i), y_i)$$

- **Goal:** find the "best" model

$$f = \underset{f \in \mathcal{F}}{\text{argmin}}\, L(f)$$



How do we choose the space $\mathcal{F}$ of candidate models?

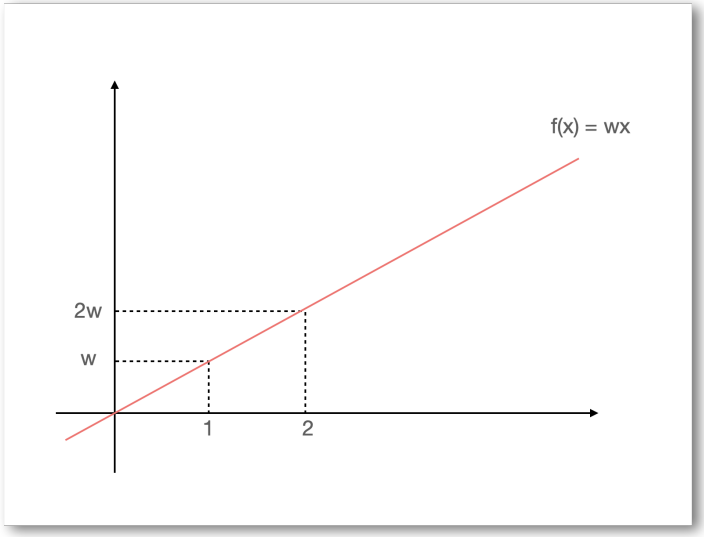
Linear Models and squared Loss

Let's start simple...

Linear Model.[1] for any $x \in \mathcal{X} = \mathbb{R}^d$

$$f(x) = w^\top x = \sum_{i=1}^{d} w_i x_i.$$

f is **parametrized** by[2] $w \in \mathbb{R}^d$.
(We can also denote $f = f_w$)



[1] Note $f(x) \in \mathbb{R}$ but we can always get "back" to $\mathcal{Y} = \{-1, 1\}$ by taking $\text{sign}(f(x))$.
[2] We will always assume vectors to be "column" vectors.

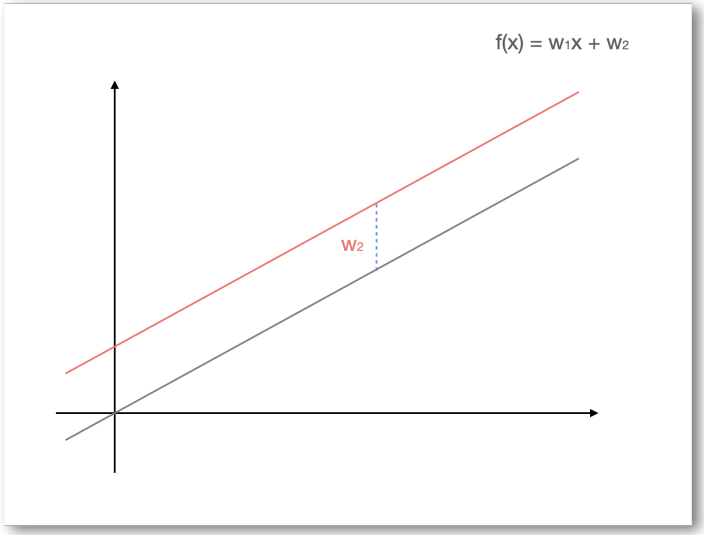
Linear Models and squared Loss

Let's start simple...

Linear Model.[1] for any $x \in \mathcal{X} = \mathbb{R}^d$

$$f(x) = w^\top x = \sum_{i=1}^{d} w_i x_i.$$

f is **parametrized** by[2] $w \in \mathbb{R}^d$.
(adding bias $x \leftarrow [x, 1]$)



[1] Note $f(x) \in \mathbb{R}$ but we can always get "back" to $\mathcal{Y} = \{-1, 1\}$ by taking $\text{sign}(f(x))$.
[2] We will always assume vectors to be "column" vectors.

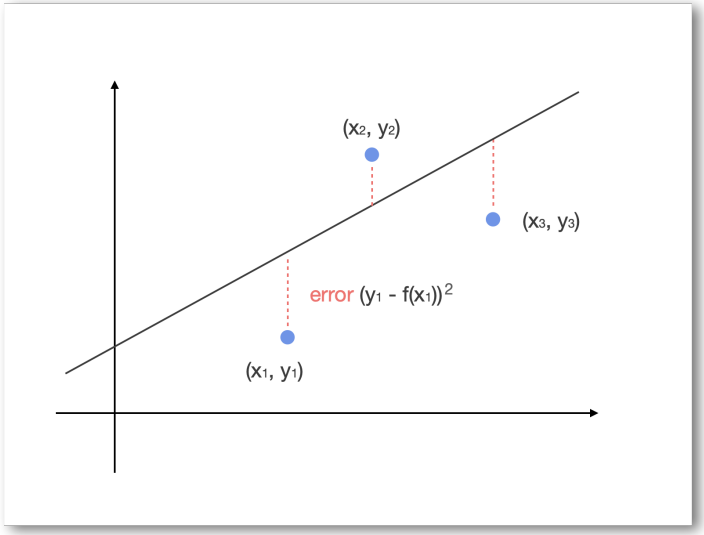
Linear Models and squared Loss

Let's start simple...

Linear Model.[1] for any $x \in \mathcal{X} = \mathbb{R}^d$

$$f(x) = w^\top x = \sum_{i=1}^{d} w_i x_i.$$

f is **parametrized** by[2] $w \in \mathbb{R}^d$.
(adding bias: $x \leftarrow (x, 1)$)



squared Loss. Errors measured as

$$\mathbf{error}(f(x), y) = (f(x) - y)^2$$

[1] Note $f(x) \in \mathbb{R}$ but we can always get "back" to $\mathcal{Y} = \{-1, 1\}$ by taking $\text{sign}(f(x))$.
[2] We will always assume vectors to be "column" vectors.

An Optimization Problem

So everything boils down to the following problem

$$\hat{w} = \underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \ \frac{1}{n} \sum_{i=1}^{n} (w^\top x_i - y_i)^2$$

- How can we find such a $\hat{w}$?
- Is it unique? If not, how to choose?


Primo

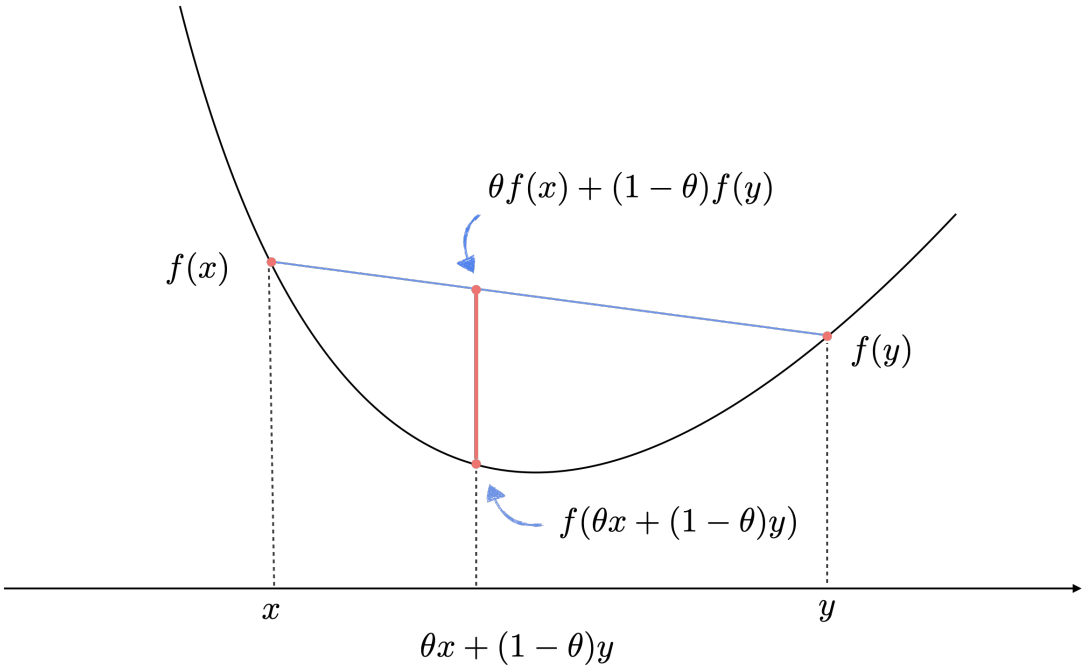
Prerequisites (Again)

We will use this example to have an idea of this module's prerequisites:

- Linear Algebra: vectors, linear functions, matrix products, norms, SVD, spectra, solving linear systems, ...
- Calculus: convex functions, derivatives, first order optimality conditions, integrals, ...
- Probability theory: distributions, expectation, variance, Bayes rule, ...

Convex Functions

The squared loss is a **convex function**.



A function f is convex if $\forall x, y$ and $\theta \in [0, 1]$

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y)$$

Convex Function - Useful Properties

- Any local minimizer is a global minimizer
- Any local descent direction is a global descent direction.
- **First Order Condition.** If f is *differentiable*...

  a point x_* is a minimizer $\Leftrightarrow \nabla f(x_*) = 0$
- ...

We'll get back to these properties in more detail in the future.

Convex Function - Useful Properties

- Any local minimizer is a global minimizer
- Any local descent direction is a global descent direction.
- **First Order Condition.** If f is *differentiable*...

  a point x_* is a minimizer $\Leftrightarrow \nabla f(x_*) = 0$
- ...

We'll get back to these properties in more detail in the future.

The squared loss is convex and differentiable, hence it is sufficient to find which parameter set its derivative to zero!

Goal. Find a $\hat{w}$ such that $\nabla \left(\frac{1}{n} \sum_{i=1}^{n} (\hat{w}^\top x_i - y_i)^2 \right) = 0$

Solving for the squared Loss

While we could do that by computing the derivative with respect to direction in $w = (w_1, \ldots, w_d)$, it is often more practical to use **matrix calculus**.

First. Let's write our objective in compact matrix notation as

$$\frac{1}{n}\sum_{i=1}^{n}(w^\top x_i - y_i)^2 = \frac{1}{n}\|Xw - y\|^2$$

- $X \in \mathbb{R}^{n\times d}$ matrix with i-th row corresponding to x_i.
- $y \in \mathbb{R}^d$ is the vector whose entries are the outputs y_i.
- Xw is a matrix-vector product (yielding a vector in $\mathbb{R}^n$).
- $\|v\| = \sqrt{\sum_{i=1}^{n} v_i^2}$ is the Euclidean norm of a vector $v \in \mathbb{R}^n$.

Matrix Calculus

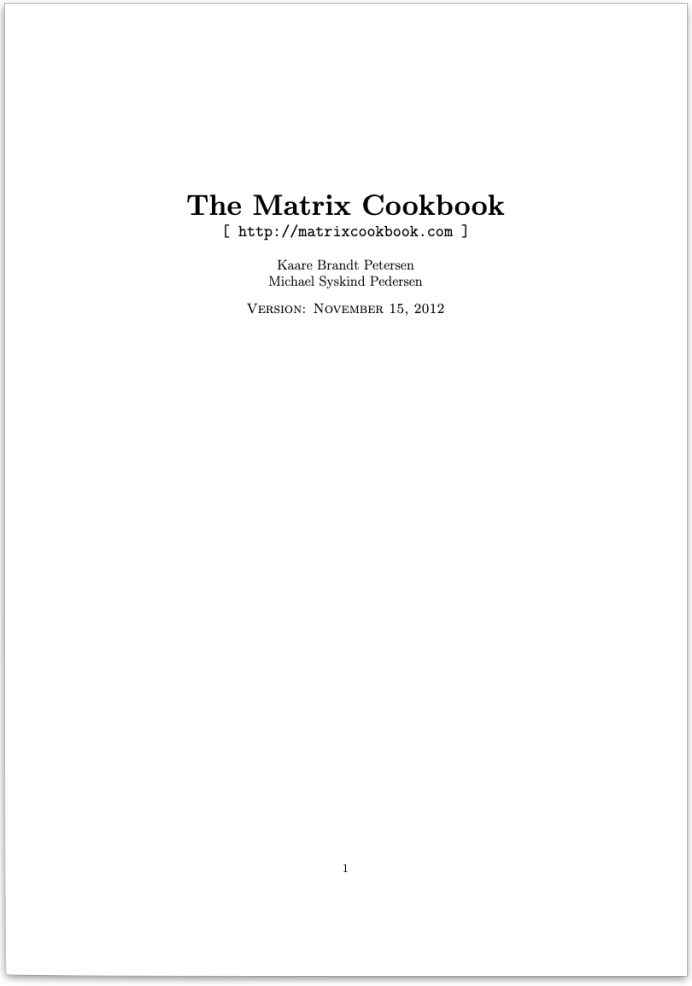
Analogously to scalar calculus, we have useful rules that make ours lives easier when differentiating wrt vector/matrix variables.

For example

- $\nabla(v^\top w) = v$
- $\nabla \|w\|^2 = 2w$
- $\nabla[f(w)^\top g(w)] = (\nabla f(w))^\top f + f(w)^\top (\nabla g(w))$
- $\nabla_w f(g(w)) = \nabla_z f(z)|_{z=g(w)} \nabla_w g(w)$
- ...

In particular

$$\nabla \|Xw - y\|^2 = 2X^\top Xw - 2X^\top y$$


The Matrix Cookbook
[http://matrixcookbook.com]
Kaare Brandt Petersen
Michael Syskind Pedersen
Version: November 15, 2012
1

The matrix cookbook

Least Squares: Solving a Linear System

The equation $\nabla \left\| Xw - y \right\|^2 = 2X^\top Xw - 2X^\top y = 0$ is solved by

$$\hat{w} = (X^\top X)^{-1} X^\top y$$

Least Squares: Solving a Linear System

The equation $\nabla \left\| Xw - y \right\|^2 = 2X^\top Xw - 2X^\top y = 0$ is solved by

$$\hat{w} = (X^\top X)^{-1} X^\top y$$

But...

- Does such a $\hat{w}$ **always** exist?
- Is it the **unique** solution?

In practice, better add a bit of "regularization"... (why?)

(Tikhonov) Regularization

We can introduce a "regularization" parameter $\lambda > 0$

$$\hat{w}_\lambda = (X^\top X + n\lambda I)^{-1} X^\top y$$

Why/when is this a good idea?

Interpretation (one of many). $\hat{w}_\lambda$ is the **unique** minimizer of

$$\frac{1}{n}\sum_{i=1}^{n}(w^\top x_i - y_i)^2 + \lambda \|w\|^2$$

Also, known as the Tikhonov regularization/ridge regression.

Least Squares Solved

We solved the learning problem, yay!

Given a new input x we can predit $\hat{y} = \hat{w}_\lambda^\top x$... **Now what?**

We can go back to our initial list of questions:

- How to collect the data (so that it's useful for the task)?
- How do we "feed" the data to a machine (representation)?
- How can we design a training algorithm?
- **How can we determine whether we trust a trained model?**
- ...

Of course, we can answer this question on a case-by-case basis but...

- **Can we say something about the expected behavior of our algorithm, in general?**
- **If yes, can such information help us improve it / design better algorithms?**


Secondo

Formulating our Real Objective

We have found a minimizer for

$$\frac{1}{n}\sum_{i=1}^{n}\text{error}(f(x_i), y_i).$$

Suppose we found a function $\hat{f}$ (e.g. with the method just described) that achives 0 error, **would you be satisfied?**

Formulating our Real Objective

We have found a minimizer for

$$\frac{1}{n}\sum_{i=1}^{n}\text{error}(f(x_i), y_i).$$

Suppose we found a function $\hat{f}$ (e.g. with the method just described) that achives 0 error, **would you be satisfied?**

Unlikely. What we really care is about **test** error (not training).

How do we define test error? Tricky.

Formulating our Real Objective

We have found a minimizer for

$$\frac{1}{n}\sum_{i=1}^{n}\text{error}(f(x_i), y_i).$$

Suppose we found a function $\hat{f}$ (e.g. with the method just described) that achives 0 error, **would you be satisfied?**

Unlikely. What we really care is about **test** error (not training).

How do we define test error? Tricky.

Consider a simplified (maybe unrealistic in some applications) scenario...

Expected Risk

Assume pairs (x, y) sampled according to an (unknown) distribution. We define the **expected risk** of a function $f : \mathcal{X} \to \mathcal{Y}$ as

$$\mathcal{E}(f) = \mathbb{E}[\ell(f(x), y)]$$

- $\mathbb{E}$ denotes **expectation** wrt (x, y),
- ℓ denotes a **loss** (more concise than error.

Goal (informal). Find a minimizer[1] for $\mathcal{E}(\cdot)$, given a finite dataset $S = (x_i, y_i)^n$ of independent and identically distributed (i.i.d.) samples.

[1] well, approximately at least – see next slide.

Excess Risk

Let us try to be more formal. How can we define the learning problem?

Ingredients

- An unknown data distribution and corresponding risk $\mathcal{E}(\cdot)$
- A dataset $S = (x_i, y_i)_{i=1}^n$ of i.i.d. samples
- A learning algorithm $A : S \mapsto f_S$

Goal. we want to minimize the **Excess Risk**

$$\mathcal{E}(f_S) - \inf_f \mathcal{E}(f)$$

Excess Risk

Let us try to be more formal. How can we define the learning problem?

Ingredients

- An unknown data distribution and corresponding risk $\mathcal{E}(\cdot)$
- A dataset $S = (x_i, y_i)_{i=1}^n$ of i.i.d. samples
- A learning algorithm $A : S \mapsto f_S$

Goal. we want to minimize the **Excess Risk**

$$\mathcal{E}(f_S) - \inf_f \mathcal{E}(f) \ \to 0$$

as $n \to +\infty$

Excess Risk

Let us try to be more formal. How can we define the learning problem?

Ingredients

- An unknown data distribution and corresponding risk $\mathcal{E}(\cdot)$
- A dataset $S = (x_i, y_i)_{i=1}^n$ of i.i.d. samples
- A learning algorithm $A : S \mapsto f_S$

Goal. we want to minimize the **Excess Risk**

$$\mathbb{E}_S[\mathcal{E}(f_S) - \inf_f \mathcal{E}(f)] \to 0$$

as $n \to +\infty$ (in e.g. expectation with respect to the sample S)

Prerequisites - Probability Theory

Notation. We will use measure-theoretic notation.

- $x \sim \rho$ denoting a point sampled according to a distribution ρ on $\mathcal{X}$.
- Expectation $\mathbb{E}[f(x)] = \int f(x)\ d\rho(x)$. For example

$$\mathcal{E}(f) = \mathbb{E}[\ell(f(x), y)] = \int \ell(f(x), y)\ d\rho(x, y)$$

- Variance $\mathbb{V}[x] = \mathbb{E}[(x - \mathbb{E}[x])^2]$
- Probability of an event A

$$\mathbb{P}(A) = \int \mathbf{1}_A(x)\ d\rho(x)$$

(!) We will not use many tools from probability theory in this module. **But** you should be at least a bit comfortable with manipulating with expectations / integrals (essentially, calculus prereqs).

Prerequisites - Probability Theory (Some more)

We will also make use of the notion of conditional probability.

For any two events (sets) $A, B \subset \mathcal{X}$

- **Definition (set-based).**

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(B \cap A)}{\mathbb{P}(A)}.$$

- **Indepdendent events.** $\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B)$
- **Bayes rule.**

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(B \cap A)}{\mathbb{P}(A)} = \frac{\mathbb{P}(A|B)\mathbb{P}(B)}{\mathbb{P}(A)}$$

We will always assume that the data distribution ρ over $\mathcal{X} \times \mathcal{Y}$ admits:

- a marginal distribution $\rho_{\mathcal{X}}(\cdot)$ over $\mathcal{X}$.
- a conditional distribution $\rho(\cdot|x)$ over $\mathcal{Y}$ for any $x \in \mathcal{X}$.

Bayes Estimator

The ideal minimizer of the expected risk is the (a?) **Bayes Estimator**

$$f_* = \operatorname*{argmin}_{f:\mathcal{X}\to\mathcal{Y}} \mathcal{E}(f)$$

Recalling the definition, if we knew ρ we could minimize $\mathcal{E}(f)$ pointwise...

$$f_*(x) = \operatorname*{argmin}_{z\in\mathcal{Y}} \int \ell(z,y)\, d\rho(y|x)$$

In other words, the Bayes estimator is the *pointwise minimizer* $f_*(x)$ of the **conditional expectation**

$$\mathbb{E}_{y\sim\rho(y|x)}[\ell(z,y)]$$

Note. While existence of the Bayes estimator is not guaranteed, most loss functions used in machine learning can be shown to admit one.

Bayes Risk for the squared Loss

Is the Bayes Risk useful even if ρ is **unknown**?

Bayes Risk for the squared Loss

Is the Bayes Risk useful even if ρ is **unknown**?

Yes! It can provide us some information on the underlying problem.

For example in the case of the squared loss

$$f_*(x) = \operatorname*{argmin}_{z \in \mathcal{Y}} \int (z - y)^2 \, d\rho(y|x),$$

is pointwise minimized by

Bayes Risk for the squared Loss

Is the Bayes Risk useful even if ρ is **unknown**?

Yes! It can provide us some information on the underlying problem.

For example in the case of the squared loss

$$f_*(x) = \operatorname*{argmin}_{z \in \mathcal{Y}} \int (z - y)^2 \, d\rho(y|x),$$

is pointwise minimized by

$$f_*(x) = \int y \, d\rho(y|x),$$

namely the **conditional expectation** $\mathbb{E}[y|x]$ of y given x (according to the unknown relation $\rho(y|x)$).

Excess Risk and the Squared Loss

Plugging this in the excess risk, yields

$$\mathcal{E}(f_S) - \mathcal{E}(f_*) = \mathbb{E}_x[(f_S(x) - f_*(x))^2] = \|f_S - f_*\|^2_{L^2(\mathcal{X},\rho_\mathcal{X})}$$

Namely the ($\rho_\mathcal{X}$-weighted) point-wise distance between f_S and f_*.

Exercise: prove the equality above

Hints:

- Write the expected risk $\mathcal{E}(f)$ explicitly for the squared loss.
- Use the fact that $\mathbb{E}_{x,y}[f(x,y)] = \mathbb{E}_x\left[\mathbb{E}_{y|x}[f(x,y)]\right]$
- "Open up" the squares.
- Recall that the Bayes risk is $f_*(x) = \mathbb{E}_{y|x}[y]$

Expected Excess Risk and the Squared Loss

We can then take the **expectation** with respect to a random dataset S

$$\mathbb{E}_S[\mathcal{E}(f_S) - \mathcal{E}(f_*)] = \mathbb{E}_x \mathbb{E}_S[(f_S(x) - f_*(x))^2]$$

(!) Note. Here $\mathbb{E}_S \mathbb{E}_x = \mathbb{E}_x \mathbb{E}_S$ since S and x are[2] **independent**.

[2] Assumption. In later classes, we will see how we could drop it.

Expected Excess Risk and the Squared Loss

We can then take the **expectation** with respect to a random dataset S

$$\mathbb{E}_S[\mathcal{E}(f_S) - \mathcal{E}(f_*)] = \mathbb{E}_x \mathbb{E}_S[(f_S(x) - f_*(x))^2]$$

(!) Note. Here $\mathbb{E}_S \mathbb{E}_x = \mathbb{E}_x \mathbb{E}_S$ since S and x are[2] **independent**.

Let $\bar{f}_n : \mathcal{X} \to \mathcal{Y}$ denote the "average" model

$$\bar{f}_n(x) = \mathbb{E}_S[f_S(x)]$$

when **trained on n points**.

[2] Assumption. In later classes, we will see how we could drop it.

Expected Excess Risk and the Squared Loss

The expected excess risk for the squared loss can then be decomposed as

$$\mathbb{E}_S[(f_S(x) - f_*(x))^2] = (f_*(x) - \bar{f}_n(x))^2 + \mathbb{E}_S[(f_S(x) - \bar{f}_n(x))^2]$$

Exercise: prove the equality above

Hints:

- Proceed left-to-right (i.e. work on the right hand side).
- "Open up" the squares.
- Recall that $\bar{f}_n(x) = \mathbb{E}_S[f_S(x)]$

Bias and Variance

Putting everything back together...

$$\mathbb{E}_S[\mathcal{E}(f_S) - \mathcal{E}(f_*)] = \mathbb{E}_x[(f_*(x) - \bar{f}_n(x))^2] + \mathbb{E}_x[\mathbb{E}_S[(f_S(x) - \bar{f}_n(x))^2]]$$

- **Bias.** Average "distance" from the ground-truth.

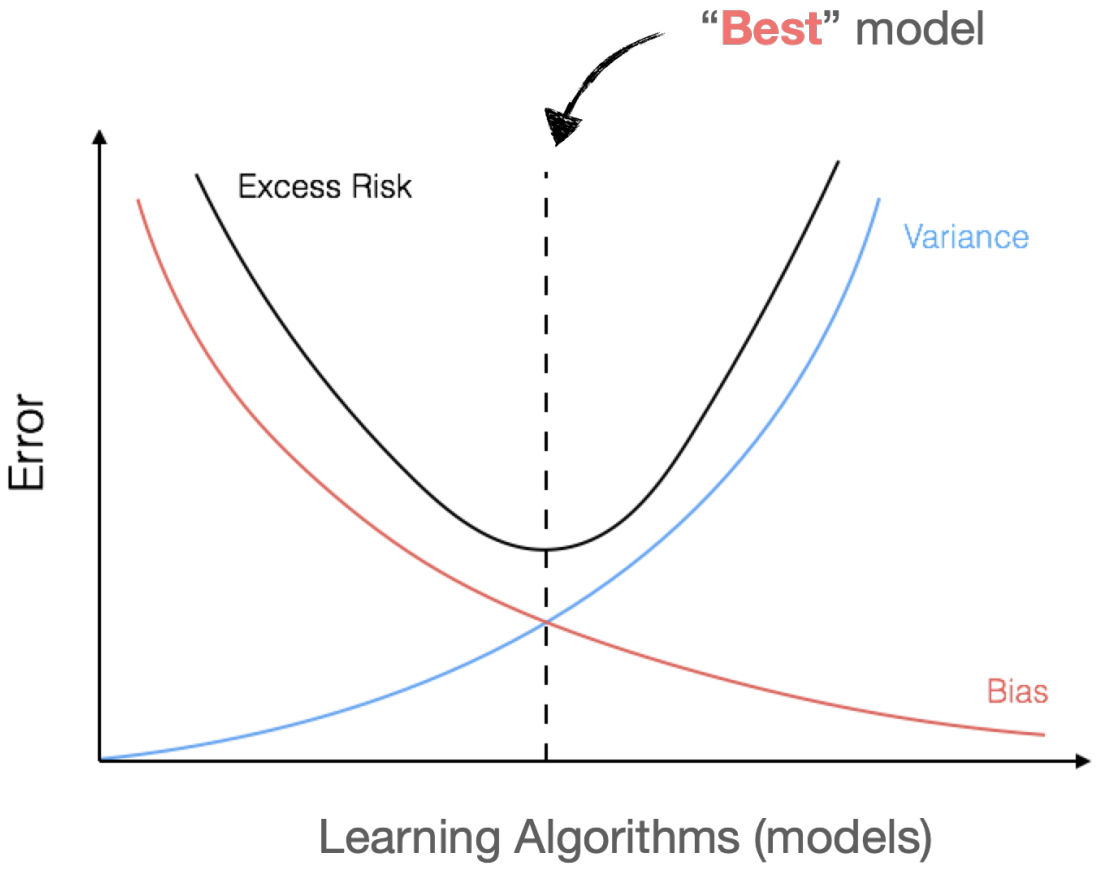
$$\mathbb{E}_x[(f_*(x) - \bar{f}_n(x))^2] = \|f_* - \bar{f}_n\|^2_{L^2(\mathcal{X},\rho_\mathcal{X})}$$

- **Variance.** The actual variance output of models as trained on different datasets

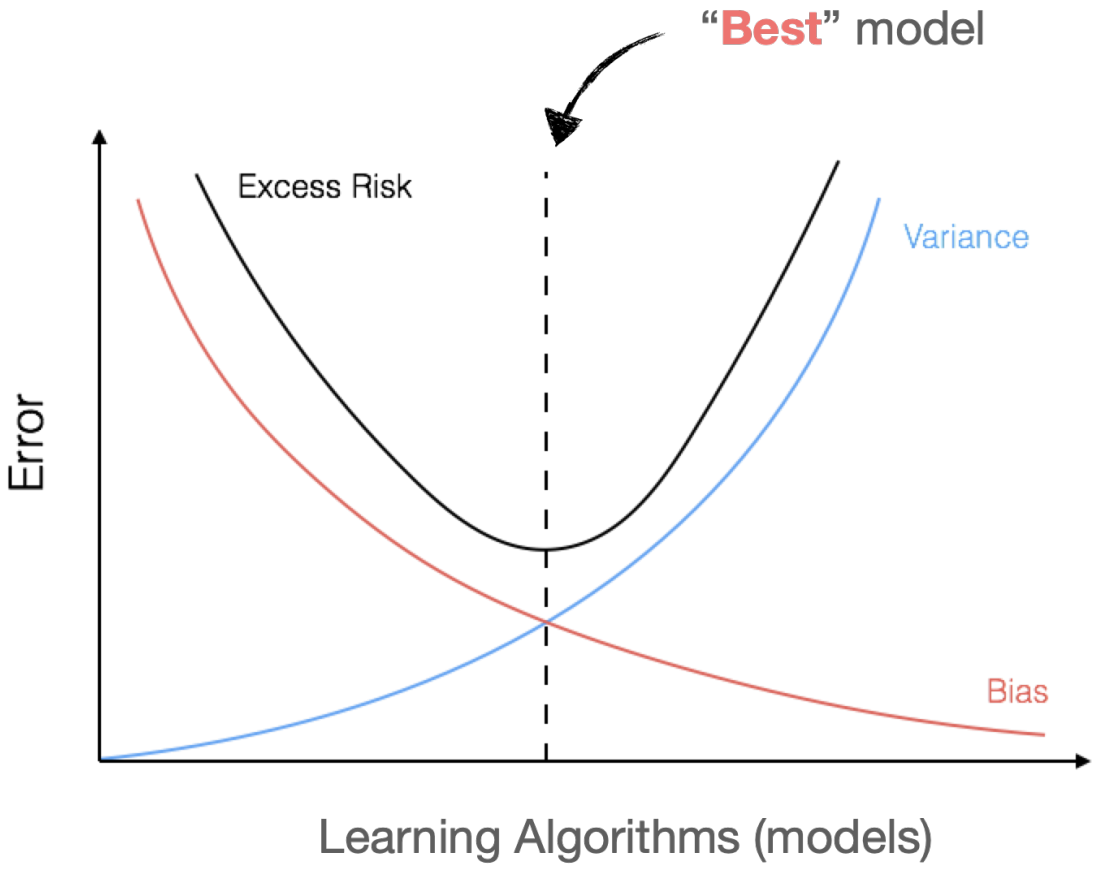
$$\mathbb{E}_x[\mathbb{E}_S[(f_S(x) - \bar{f}_n(x))^2]] = \mathbb{E}_x[\mathbb{V}[f_S(x)]]$$

How do these two term vary as the number n of points increases?

Bias-Variance Tradeoff



Bias-Variance Tradeoff



Ideally, we would like to choose the algorithm that achieves always the best trade-off between **bias** and **variance**.

A "Reasonable" Question

Can we design an algorithm that **always** achieves the best bias-variance?

In other words...

Can we "Solve" Supervised Learning?

A "Reasonable" Question

Can we design an algorithm that **always** achieves the best bias-variance?

In other words...

Can we "Solve" Supervised Learning?

No. Not really...


Dessert

A No Free Lunch Theorem

Note. For the classification loss $\ell(z, y) = \mathbf{1}_{z \neq y}$, the risk corresponds to

$$\mathcal{E}(f) = \mathbb{E}_{x,y}[\mathbf{1}_{f(x) \neq y)}] = \mathbb{P}(f(x) \neq y)$$

A No Free Lunch Theorem

Note. For the classification loss $\ell(z, y) = \mathbf{1}_{z \neq y}$, the risk corresponds to

$$\mathcal{E}(f) = \mathbb{E}_{x,y}[\mathbf{1}_{f(x) \neq y)}] = \mathbb{P}(f(x) \neq y)$$

Theorem

Let A be a learning algorithm for binary classification. Let $n < |\mathcal{X}|/2$, then there exists a distribution ρ over $\mathcal{X} \times \mathcal{Y}$ such that,

1. There exists a Bayes estimator $f_* : \mathcal{X} \to \mathcal{Y}$ (with $\mathbb{P}(f_*(x) \neq y) = 0$).
2. Let $A(S) = f_S$ be the model trained on a dataset S of n independent samples. Then

$$\mathbb{P}_S\Big(\mathbb{P}_{(x,y)}[f_S(x) \neq y] \geq 1/8\Big) \geq 1/7$$

A No Free Lunch Theorem

Note. For the classification loss $\ell(z, y) = \mathbf{1}_{z \neq y}$, the risk corresponds to

$$\mathcal{E}(f) = \mathbb{E}_{x,y}[\mathbf{1}_{f(x) \neq y)}] = \mathbb{P}(f(x) \neq y)$$

Theorem

Let A be a learning algorithm for binary classification. Let $n < |\mathcal{X}|/2$, then there exists a distribution ρ over $\mathcal{X} \times \mathcal{Y}$ such that,

1. There exists a Bayes estimator $f_* : \mathcal{X} \to \mathcal{Y}$ (with $\mathbb{P}(f_*(x) \neq y) = 0$).
2. Let $A(S) = f_S$ be the model trained on a dataset S of n independent samples. Then

$$\mathbb{P}_S\Big(\mathbb{P}_{(x,y)}[f_S(x) \neq y] \geq 1/8\Big) \geq 1/7$$

Proof? Exercise!

A No Free Lunch Theorem

Note. For the classification loss $\ell(z, y) = \mathbf{1}_{z \neq y}$, the risk corresponds to

$$\mathcal{E}(f) = \mathbb{E}_{x,y}[\mathbf{1}_{f(x) \neq y)}] = \mathbb{P}(f(x) \neq y)$$

Theorem

Let A be a learning algorithm for binary classification. Let $n < |\mathcal{X}|/2$, then there exists a distribution ρ over $\mathcal{X} \times \mathcal{Y}$ such that,

1. There exists a Bayes estimator $f_* : \mathcal{X} \to \mathcal{Y}$ (with $\mathbb{P}(f_*(x) \neq y) = 0$).
2. Let $A(S) = f_S$ be the model trained on a dataset S of n independent samples. Then

$$\mathbb{P}_S\Big(\mathbb{P}_{(x,y)}[f_S(x) \neq y] \geq 1/8\Big) \geq 1/7$$

Proof? Exercise!

(No, seriously)

Implication of the NFT

Is the NFT really such a pessimistic message?

Implication of the NFT

Is the NFT really such a pessimistic message?

Maybe not? NFT proofs typically work by creating a hard learning problem tailored to each individual algorithm.

Implication of the NFT

Is the NFT really such a pessimistic message?

Maybe not? NFT proofs typically work by creating a hard learning problem tailored to each individual algorithm.

Perhaps, a more interesting question would be whether learning problems "in the wild" are hard or not for the algorithm we have in mind.

But, can we find it out?

Implication of the NFT

Is the NFT really such a pessimistic message?

Maybe not? NFT proofs typically work by creating a hard learning problem tailored to each individual algorithm.

Perhaps, a more interesting question would be whether learning problems "in the wild" are hard or not for the algorithm we have in mind.

But, can we find it out?

Probably not. However... we could identify which problems are easy / hard for a specific learning algorithm. Or better yet, design algorithms that are good for specific learning problems (and inevitably bad for learning problems that we are not interested in)

This is why Supervised Learning theory is useful!

The Bill


BCE ECB EZB EKT EKP
50
53.28
Thanks for Shoppi
Card Number
Date 21.10.
*** Credit

What we have talked about

- Supervised Learning motivations and examples.
- The prototypical learning pipeline.
- Example: Spam, Squared Loss and Linear Models.
- A bit of math to solve our problems.
- Learning Theory: the iceberg under the tip.
- No Free Lunch, Bias-variance and why there is still hope...


Coffee
(On the House)

Course Outline

From practice...

- Week 1: Introduction
- Week 2: Kernels and Regularization
- Week 3: Support Vector Machines
- Week 4: Convex Optimization - First Order Methods
- Week 5: Tree-based methods

...to theory (and back)

- Week 6: Statistical Learning Theory (Foundations)
- Week 7: Statistical Learning Theory (Rademacher Complexity)
- Week 8: Online Learning (Part 1)
- Week 9: Online Learning (Part 2)
- Week 10: Advanced Topics and Loose Ends

But this is for the next time.

Now, a well deserved...


NAP