Introduction

Machine Learning Theory (MLT)

Edinburgh

Rik Sarkar

Machine Learning Theory – Course info

- Course code: INFR11202/INFR11224, Shorthand: MLT
  - Web page: https://opencourse.inf.ed.ac.uk/mlt
- Lecturer : Rik Sarkar (rik.sarkar@ed.ac.uk)
- Schedule
  - Tuesdays 16:10 – 18:00 (2.35 | Lecture Theatre - Edinburgh Futures Institute (EFI))
- Three Tutorial sessions
  - Weeks 5, 7, 9 (may change)
- 1 Coursework 30% (written analysis, proofs) (Released Feb 14, Due march 12)
- 1 Exam (April/May) : 70%

Resources

- Book: Understanding Machine Learning: From Theory to Algorithms. Shai Shalev-Shwartz and Shai Ben-David
  - Available in library, Book retailers, free pdf Online

- Other notes and papers to be given out as we go.
- Exercises given in tutorials and notes.
- Piazza forum active (linked from Learn)
  - For communication and discussion
  - We will try to help, but not immediately. Try yourselves first!
- Sample Exam: Previous years' exams available online
- Forms of feedback:
  - Coursework.
  - Tutorials: Attempt tutorial exercises. Attend tutorials. Ask questions.
  - Exercises in notes: Some exercises available in notes. Attempt them and check solutions.

Today

- Discussion of Machine learning and Theory
- Overview what fundamental topics/problems we will study in this course
  - Models and neural networks
  - Optimisation and loss landscapes
  - Learning theory
  - Privacy
  - Fairness
  - Explainablility
- Analysis of a simple 1 parameter problem

What is machine learning

- What is learning?
- When is machine learning useful?

Learning is useful when

- Available data is small compared to possible inputs/questions
  - If answers to all relevant questions are available, then it is just a matter of memorization!
- Data possibly contains noise
- We have some idea (hypothesis class) of what the learned model could be

- Ideally the smaller quantity of data we learn from, the better
  - But what is the definition of “small”? How little data is sufficient?

Why theory

- This course is not about new types of models or applications
- We are interested in mathematics of ML
  - Understand the behavior of ML models
  - Gain better understanding of their strengths and weaknesses – where they work where they do not. What is understood/not understood
- Do better ML in the future
  - Accuracy, generalization
  - Privacy
  - Fairness
  - Explainability
  - Other desirable properties….

Have you taken an ML course before?

- Raise your hand if you have **never** taken an ML course

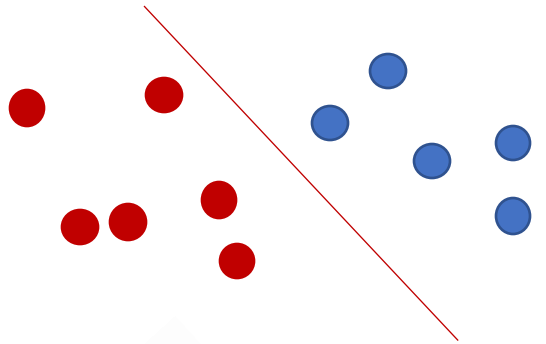
Who/What the course is for

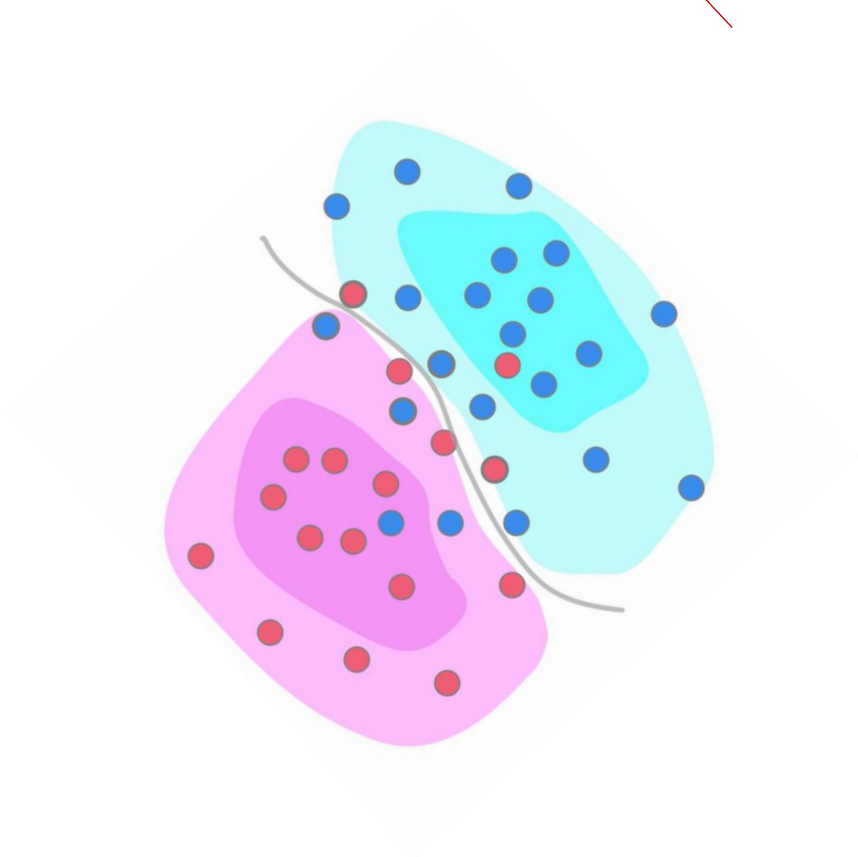
- Prepare you to study more advanced ML
  - Reading latest papers in cutting edge ML is not easy.
  - This course will help you understand such material

- The course is suitable for two types of students
  - You have learned various ML models and algorithms in other courses and would like to have a unified view and understand them at a deeper level
  - You have studied maths/stats and would like to know how to think about ML
- If neither description works for you, talk to me about suitability

Classification

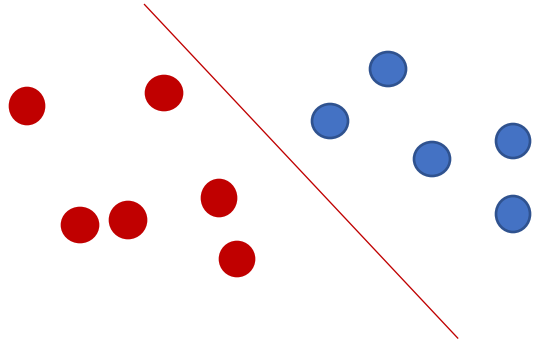
- Could be simple
  - Linear separation
  - Red: $y \leq mx + c$
- Or more complex
  - Red $y \leq ax^3 + bx^2 + cx + d$

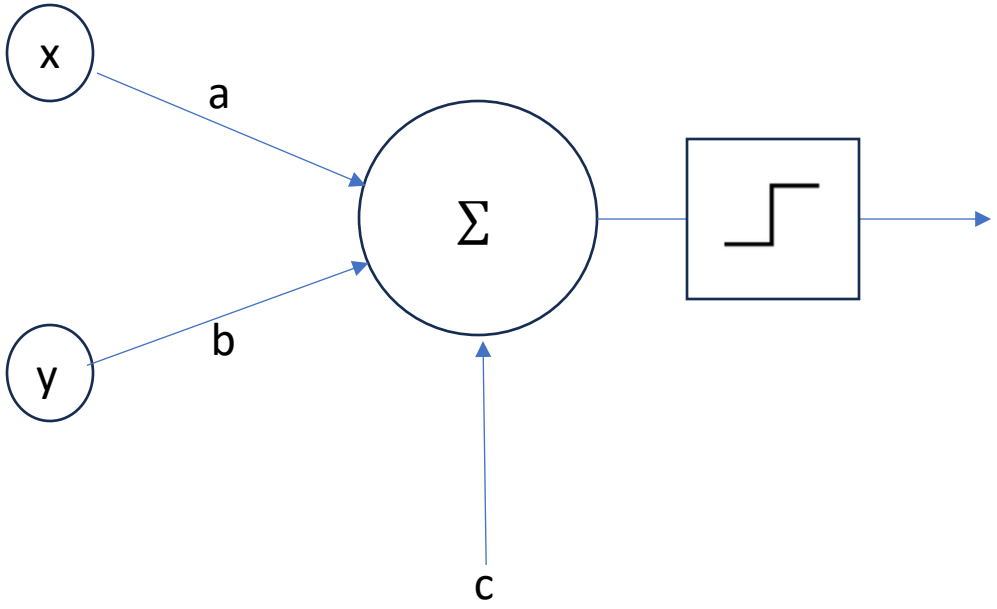




Perceptrons

- Straight line separators
  - $ax + by + c \geq 0$
- Can be drawn diagrammatically
- Sometimes the summation sign is omitted and activation function put in place

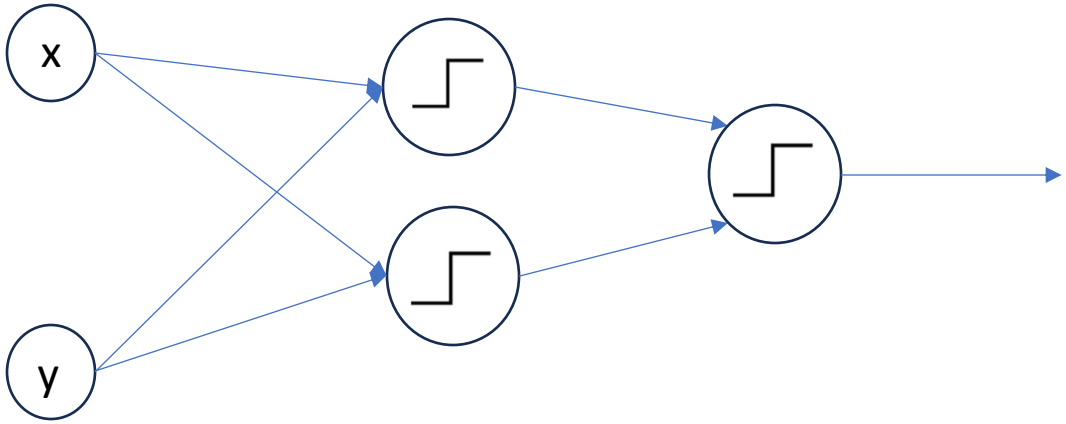


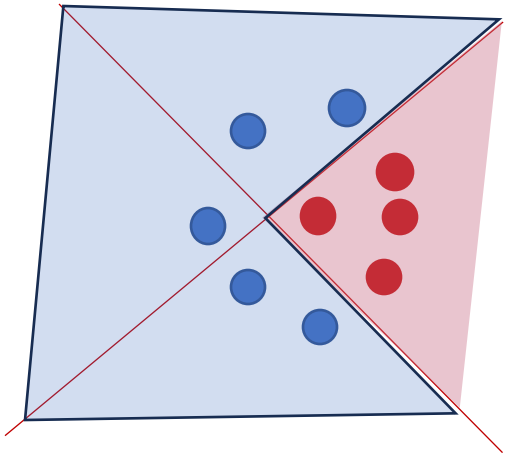


What do we get with more perceptrons?

- Find out what weights would be needed to achieve something like this

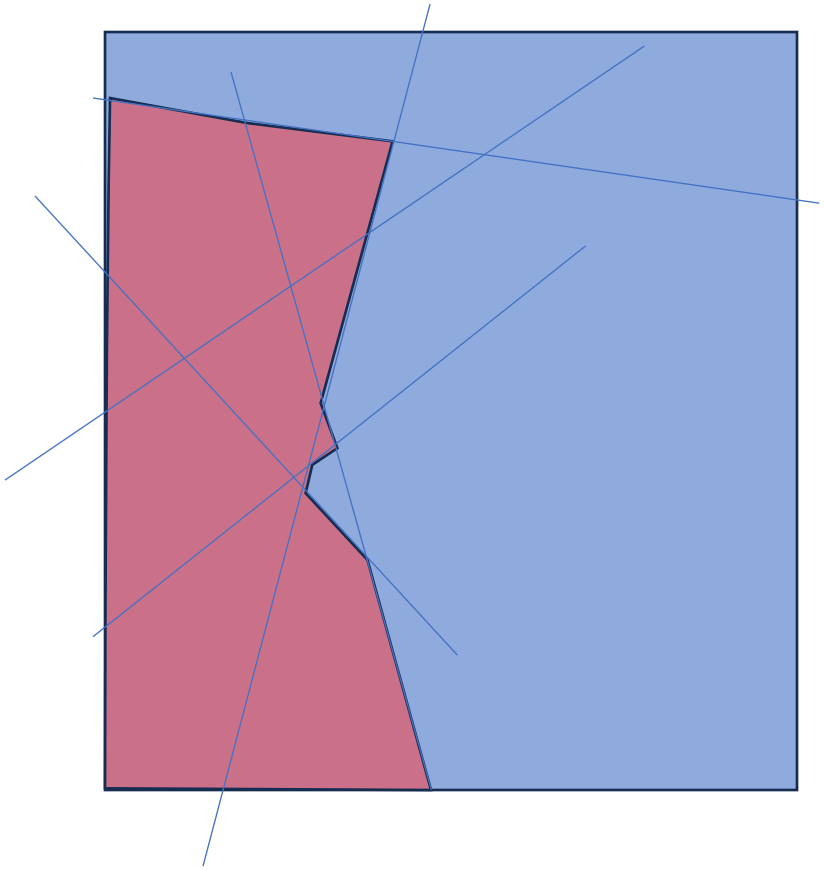
- Exercise:
  - Supposing the centre is at the origin and separator lines are at 45degree, work out what weights will achieve this





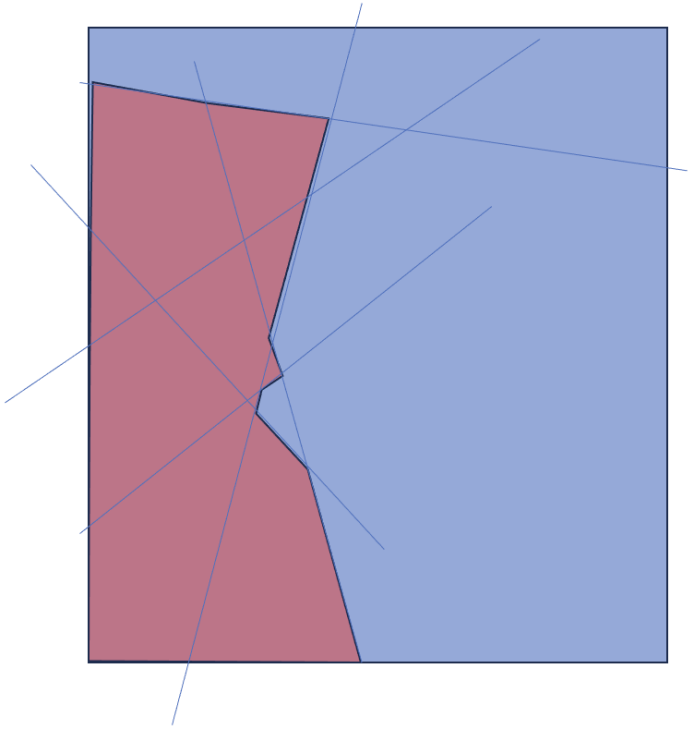
With more complicated networks:

- We can achieve more complicated decision boundaries



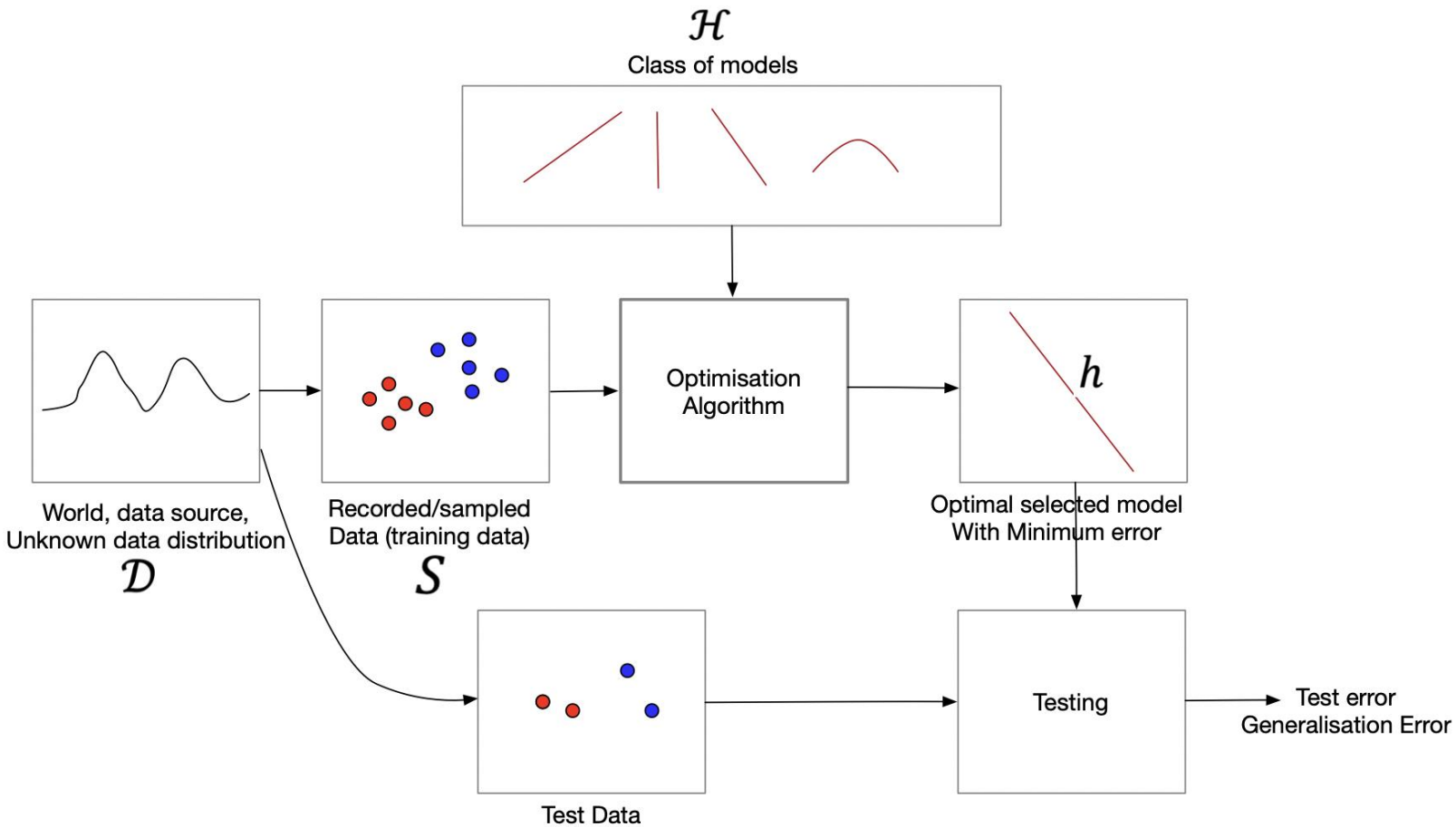
Fundamental topics

- What exactly can neural networks achieve?
- Why are large deep networks so powerful?
- Do they really need to be “large”?
- Why do neural networks overfit?
- How should we think about the different layers of NNs?



The machine learning pipeline

- Assume there is a distribution $\mathcal{D}$ from which data is drawn
  - S is a sample of m data points used as training data
  - Written as $S \in \mathcal{D}^m$
- $\mathcal{H}$ is a hypothesis class: a set of possible models
- $h \in \mathcal{H}$ is a model E.g. a model selected by an algorithm
- An optimization algorithm $\mathcal{A}$ takes in $S, \mathcal{H}$ and produces a model h that it thinks has lowest errors on S
- h is tested on test data also taken from $\mathcal{D}$ to estimate generalization of h


$\mathcal{H}$
Class of models
Optimisation Algorithm
h
Optimal selected model
With Minimum error
World, data source,
Unknown data distribution
$\mathcal{D}$
Recorded/sampled
Data (training data)
S
Test Data
Testing
Test error
Generalisation Error

Question:

- Can the model class be all possible models?

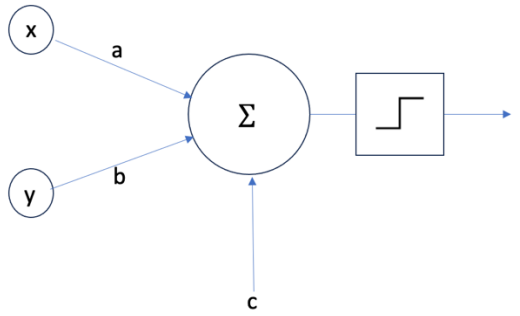
Models as vector

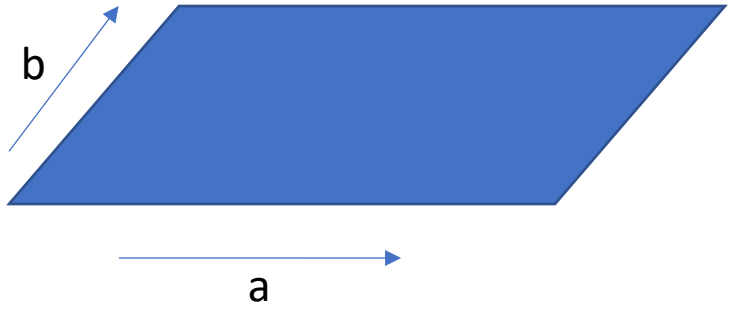
- For a known class of models, we can represent them by a vector of numbers (parameters):
  - Neural networks: A vector of edge weights
  - Regression: Coefficients of polynomials
- Observe: The vector of numbers does not say the type (class) of model. That is up to us.

- Usually, this vector is written as a weight vector $\boldsymbol{w} = (w_1, w_2, w_3, \dots)$
- The size or dimension of is the size of the model
  - Larger models are likely to be more *complex.*

The *space* of models

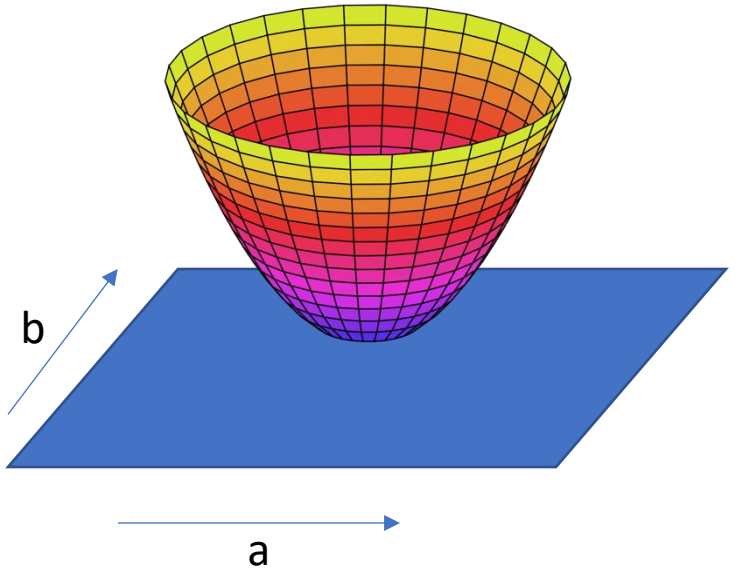
- If each model is a vector
- We can imagine a vector space or Euclidean space
  - Where each point is a model
  - The dimension of the space (number of weights, coordinates) is the dimension of $\mathcal{H}$
- Optimisation Algorithm: Search over all possible (a, b) to find the best model



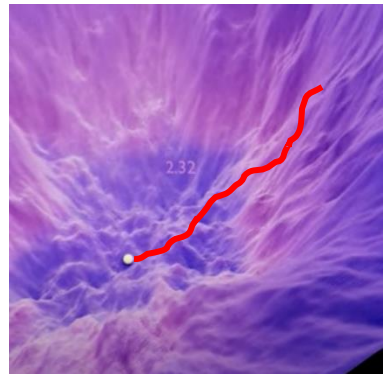


Loss functions and optimization algorithms

- A *loss function* is an estimate of how bad a model (a, b) is (how much error it will have)
  - Cross entropy loss is a common one for deep learning
- In the a,b space, we are looking for the model with the lowest loss



- We will study stochastic gradient descent algorithm used to train neural networks and its loss landscape
- Finding low loss is harder when the loss landscape is more complex



Fundamental topics

- What do loss landscapes look like?
- How do they affect the training of complex models?
- Is it always best to get to the minimum loss?
- How are loss landscapes affected by individual data points
- How does that affect the SGD training

Learning theory and Empirical risk minimization (ERM)

- Given a dataset S of size m,
- The empirical loss of hypothesis h is defined as
  - The average loss over all data points

$$L_S(h) \stackrel{\text{def}}{=} \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m}$$

- This is called the empirical risk or empirical error or empirical loss
- ERM is finding h with minimum $L_S(h)$

Observation 1

- Finding the true minimum-loss L_{min}
- may be difficult
  - E.g. Searching in an infinite model class is not easy
  - And we do not know what the min loss is

- What we can hope is to ensure that loss is not much higher than minimum
- That is, loss is approximately minimum
- It is not higher than $L_h \leq (1+\epsilon)L_{min}$
- Note: we do not know L_{min}

Observation 2

- Our eventual goal is a good classifier for $\mathcal{D}$
- But we work on S : a random sample of $\mathcal{D}$
- We cannot guarantee that S is a good representative of true $\mathcal{D}$
- But, with enough samples, we are *likely* to get close
  - But not for sure. It is still probabilistic

- So, with enough data:
  - Probably, we can approximate the minimum loss model!

PAC learning

- Probably Approximately Correct learning
  - If true min-loss is L_{min} and L_h is loss of h, then
  - PAC learnability means we can get:
  - $\Pr[(L_h - L_{min}) \leq \epsilon] \geq 1 - \delta$
- (Hopefully for small ϵ, δ)
- With a reasonably small amount of data

Fundamental questions

- When is PAC learning provably possible?
- How much training data do we need?
- What does the data size depend on?
- What are the causes of overfitting?

Privacy

- The problem: We need data for ML
  - Data comes from people
- It may reveal sensitive information about people
  - The data itself may get leaked
  - The ML model, or decisions made by it may reveal information
- Simple example: A company releases average salary every week. When you join the company, someone can guess your salary comparing with previous month's average.
  - how can you prevent that?
- What is the machine learning analog?

Privacy preserving machine learning and differential privacy

- The study of these small changes to functions and models
  - Understanding the effect of each tiny data point

- A different perspective in ML/AI
  - How can we reveal some facts and hide others?
  - What are the limits of this tradeoff?

- Differential privacy works by adding small calculate amounts of noise to models. Precise Bayesian definitions and properties

Fundamental topics

- What is *privacy*? How can we measure it?
- How can we get privacy in machine learning?
- What does privacy cost us?
  - Trying to make models private makes the less accurate.
  - What is the tradeoff?
- How does privacy relate to other aspects of ML?

Fairness

- ML can be unfair
- E.g.
  - Most employees in a company belong to a certain community
  - The CV scanning software learns that bias
  - Even if the community is not stated explicitly (e.g. correlations with name, address etc)
- Data is likely to be biased toward the majority
- Anyone can be a minority with suitable combination of parameters (e.g. race, religion, gender, age, advanced degree..)

Fairness

- Fairness study is important for better ML
- E.g. a bank software refuses loan to a “minority” due to fairness failure
  - Bad for the person
  - Bad for the society in the long term that a deserving person did not get a loan
  - Bad for the bank as they miss a good investment

Fairness

- Suppose we are recruiting based on an exam score
- Is it fair to admit on exam score?

Fairness

- Suppose we are recruiting based on an exam score
- Is it fair to admit on exam score?

- Suppose there is the majority group A and minority group B.
- The minority group is disadvantaged in some way
  - What are examples of such disadvantages?

- How can we adjust for such imbalances?

Fairness

- We will study
  - Precise mathematical definitions
  - See that everything we want may not be achievable
- Fairness is a complex topic
  - What is fair from one perspective is unfair from others
  - Many possible metrics of fairness
  - Affected by other properties like generalization, stability, privacy etc…

Fundamental quesitons

- What is fairness? In what ways can we quantify it?
- What are the limits of fairness?
- How do we test if a dataset is fair? If a model is fair?

Explainability

- Neural networks or any kind of large models are hard to understand
- That is, they may produce good results, but we do not understand why they work and how they are producing the result

- Question: Why do we want to understand?

Why explainability?

- Models are not perfect – explanation lets us analyse and cross check its decision
- Essential in certain applications (e.g. medical diagnosis)
- Can help us gain better understanding and recogonise patters that we can’t see otherwise – e.g. in scientific data
- Help improve the models

Explainability

- Giving scores to features
  - For a particular output
  - For general accuracy of the model
    - Which features contribute more to accuracy?
- Attaching value to data
  - Which data points contribute more to the accuracy?
- We will study:
  - Shapley Value from Economics – assigns Value to different items
  - Other techniques

Fundamental questions

- What is a simple model that would have produced the same result?
- Why did the model produce a particular result?
  - E.g. a model predicts rain tomorrow
- Was a specific feature(s) were important?
- Did certain training data points play a role?
- Will a different type of model work better?

A simple classifier (see book chapters 1 & 2)

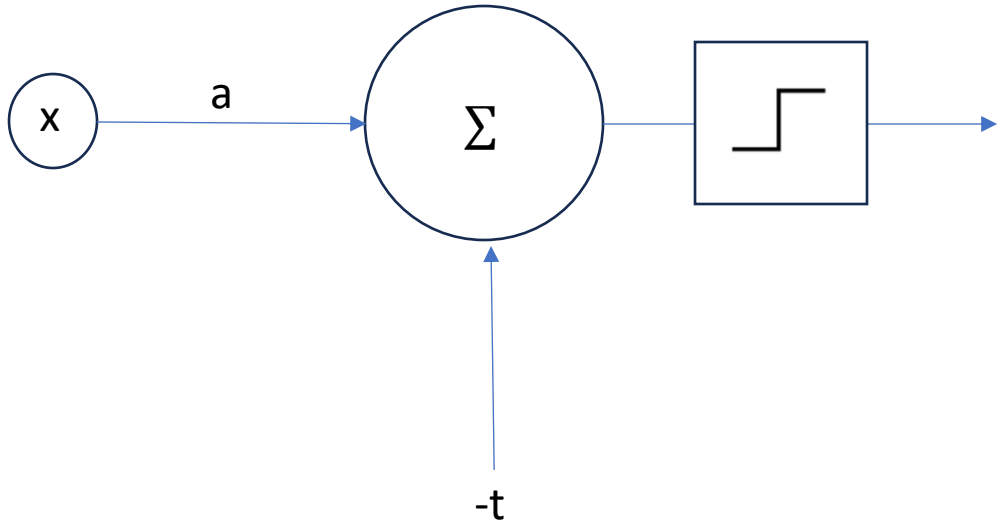
- A supermarket has asked us to build a model classify ripe papayas
- Green is unripe, yellow is ripe
- A sensor reads the colour
- And returns a value in [0,1]

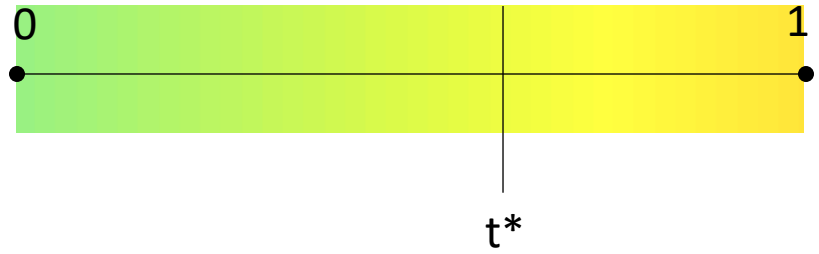
- Assume the supermarket sends us a random sample of labelled readings
- There is a color threshold t* of ripe papayas but we don’t know it.

We are using a simple 1-perceptron network

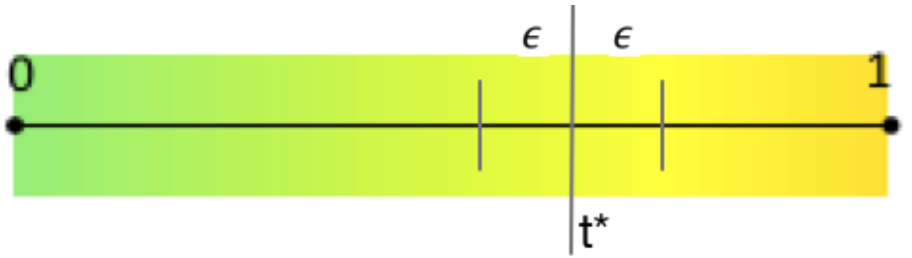
- Let us set a = 1
- We just have to find a t
  - This will determine the classification

- We want to select a threshold c that is at distance at most ϵ from t*
  - That is: $|t - t^*| \leq \epsilon$





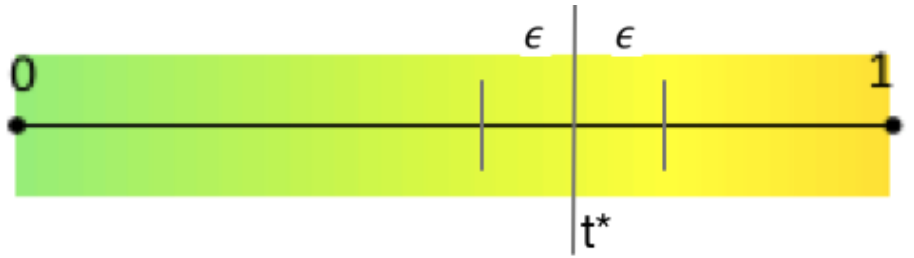
- We want to select a threshold c that is at distance at most ϵ from t*
  - That is: $|t - t^*| \leq \epsilon$
- Algorithm
  - Draw enough samples
    - So that there are samples in ϵ intervals to the left and right of t*
  - Take the highest “unripe” label and lowest “ripe” label.
  - Select any point between these two



Sample count

- If we take one sample, what is the probability that it is in the left interval of size ϵ?

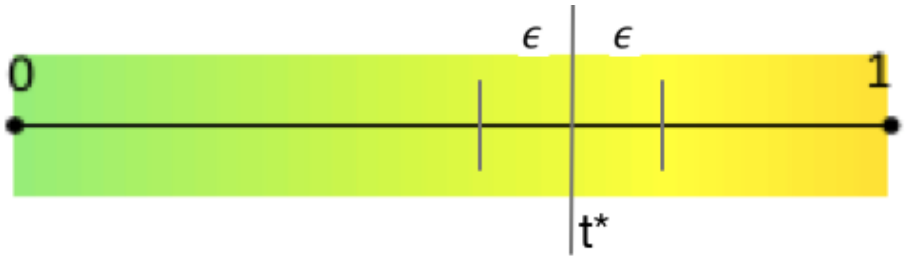
- How many random samples do we need to ensure that there is a sample on the left interval with probability at least $1 - \delta$?



Final sample count

- We need sample size of $m \geq \frac{1}{\epsilon} \ln \frac{1}{2\delta}$

- To ensure that there is a sample in each of the two intervals with probability at least $1 - 2\delta$



- Homework: Write this out nicely.
  - The inequality you need is $(1-p)^{\frac{1}{p}} \leq \frac{1}{e}$

Homework

- Read lecture notes (to be uploaded)
- Read chapters 1 and 2 of the book: Understanding Machine Learning