Lecture #5: Machine Learning Basics

COMP3608: Intelligent Systems

Inzamam Rahaman

The phenomena of human intelligence

- One of the most intriguing and defining characteristics of human intelligence is our ability to learn!
- If our goal is to build AI tools that are useful to us, it would be useful to build agents capable of learning
- What is learning?

What is Learning?

- Broadly speaking, we use experience to construct mental **models** to encode understanding
- Use mental models to reason about situations that we **have not yet experienced**
- Introspect mental models to gain **insight** into what we experienced
- Use mental models to help **make decisions** about which actions to taken
- Is different from memorising data

What is Machine Learning?

- The usage of algorithms to computationally and in an automated fashion derive models from data such that we can glean non-trivial insight, make predictions about future events and unseen scenarios, and make decisions
- Recall our definition of a rational agent
  - Uses knowledge of state
  - To make decisions
  - To optimise performance measure

Types of Machine Learning

- Learning is broad. Likewise, there are many flavours of machine learning with different use cases!
- Three major categories:
  - Supervised Learning
  - Unsupervised Learning
  - Reinforcement Learning
- Others exist, but these are typically grey areas around these main three, e.g. weakly supervised learning, semi-supervised learning, etc..

Why is it when something happens, it is always you three?
Believe me, Professor, I've been asking myself the same question for 6 years.

Types of Machine Learning

- Will focus on some supervised learning
- In the space of unsupervised learning, will only be able to cover autoencoders
- Won’t have time to cover reinforcement learning at all
- But should know definitions and sub-types of problems
  - Can’t search for a solution unless you know the problem you are trying to solve!

On Data and Features

- Most machine learning algorithms assume that your data is numerically encoded in some way
- For the most part, we store our encodings for some data as a tensor of some sort
    - We will work with vectors and matrices here
- Each component of tensor represents some aspect of characteristic of our data
    - Each characteristic is termed a feature
    - Example, we can measure the width and length of sepals and petals of flowers. Each measurement has a numeric value, that we can then encapsulate in a consistent way as feature vectors

Types of data

- Different types of data:
  - Quantitative: regular numerical data that can be real numbers, e.g. height and weight
  - Ordinal: integer data where ordering of numbers is important, e.g. preference scores in surveys
  - Categorical: label data (usually represented as an integer) where order does not matter and numerical association is usually “arbitrary” and to the user’s discretion, e.g. dog (0) or cat (1)
    - If we have more than 2 categories, we can either label each an arbitrary integer or use one-hot encoding

One Hot Encoding

- Suppose that we have n categories. We let every category have an associated integer between 0 and $n-1$ and represent vector as n-length bit vector with all zeros except in the one component associated with the category.

- Example, suppose we have three colours of flowers: pink, purple, and yellow. Let pink = 0, purple = 1, yellow = 2. Our flower is purple, so we encode this as $[0,1,0]^T$ and we append to the rest of the feature vector!

Feature Vector Example

- Suppose that we have several flowers, and we record the attributes average sepal width, average sepal length, average petal width, average petal length, and colour (one of pink, purple, or yellow).
    - Sepal Width = 2.0 cm
    - Sepal Length = 3.1 cm
    - Petal Width = 4.5 cm
    - Petal Length = 6.7 cm
    - Color = pink
    - How to encode this information into a feature vector consumable by an ML algorithm?

Feature Vector Example

- We can encode it as $[2.0, 3.1, 4.5, 6.7, 1, 0, 0]^T$
    - Sepal Width = 2.0 cm
    - Sepal Length = 3.1 cm
    - Petal Width = 4.5 cm
    - Petal Length = 6.7 cm
    - Color = pink

Supervised Learning

- Given a set of input-output pairs.
  - $\mathscr{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_1, y_1)\}$ where $\forall i, x_i \in \mathscr{X}$ and $\forall i, y_i \in \mathscr{Y}$
  - Assume that there exists some function $f : \mathscr{X} \rightarrow \mathscr{Y}$. f can be some "process" or relationship, i.e. the relationship between height, weight, average weekly caloric intake, and Haemoglobin A1C (HbA1C)
  - We want to approximate f

Supervised Learning

- $x_1, x_2, \ldots, x_n$ are our independent or manipulated variables
- $y_1, y_2, \ldots, y_n$ are our dependent or responding variables
- Want to use independent variables' values to predict dependent variable values!
- For those of you, who did O'level Physics, think about recording some data from an experiment. You plotted the components you had control over (your x-axis) against what you measured (your y-axis), and then drew the best-fit line
- In essence, supervised learning is about trying to draw best-fit lines

Supervised Learning

- Two main subtypes:
  - Regression: dependent variable is continuous or discrete and infinite (e.g. integers). E.g. predicting house prices from square footage, number of bedrooms, number of bathrooms, etc..
  - Classification: dependent variable is discrete and finite. We also assume that the dependent variable is categorical. E.g. building a spam classifier, building a tool to recognise emotions in facial expressions
    - Ordinal Regression: can be used when the dependent variable is ordinal, E.g. determining the if some someone will strongly disagree, disagree, agree, or strongly agree with an opinion given demographic information

Hypothesis Space in Supervised Learning

- Recall that we want to find a function that approximates the relationship between independent and dependent variables well
- But space of possible functions is infinite and very varied
- Need to make simplifying assumptions of what sort of functions would work well enough
- Usually, we make assumptions about general structure of functions, e.g. linear, polynomial, etc…

Hypothesis Space in Machine Learning

- For example, we say that we are only considering linear functions. Hence, our hypothesis space is set of all linear functions from our input space ($\mathbb{R}^n$) to output space ($\mathbb{R}$).

- Every linear function over some $\mathbb{R}^n$ is of the form $w^T x + c$. Here x is our function's input. w and c are parameters of our linear function. We can formalise this by saying $f(x; w, c) = w^T x + c$

- Need to make decisions about values for w and c.

- Hence, we reduce the problem of finding a good linear function to approximate our relationship to finding good values of w and c

Empirical Risk Minimisation (ERM)

- How to choose good parameter values?
- Assume that our data is a representative sample
- Need to be able to measure how badly we are performing on average across data points as proxy for how badly we will perform in the wild
- This function that computes this acts as a loss function that we want to minimise!

Empirical Risk Minimisation

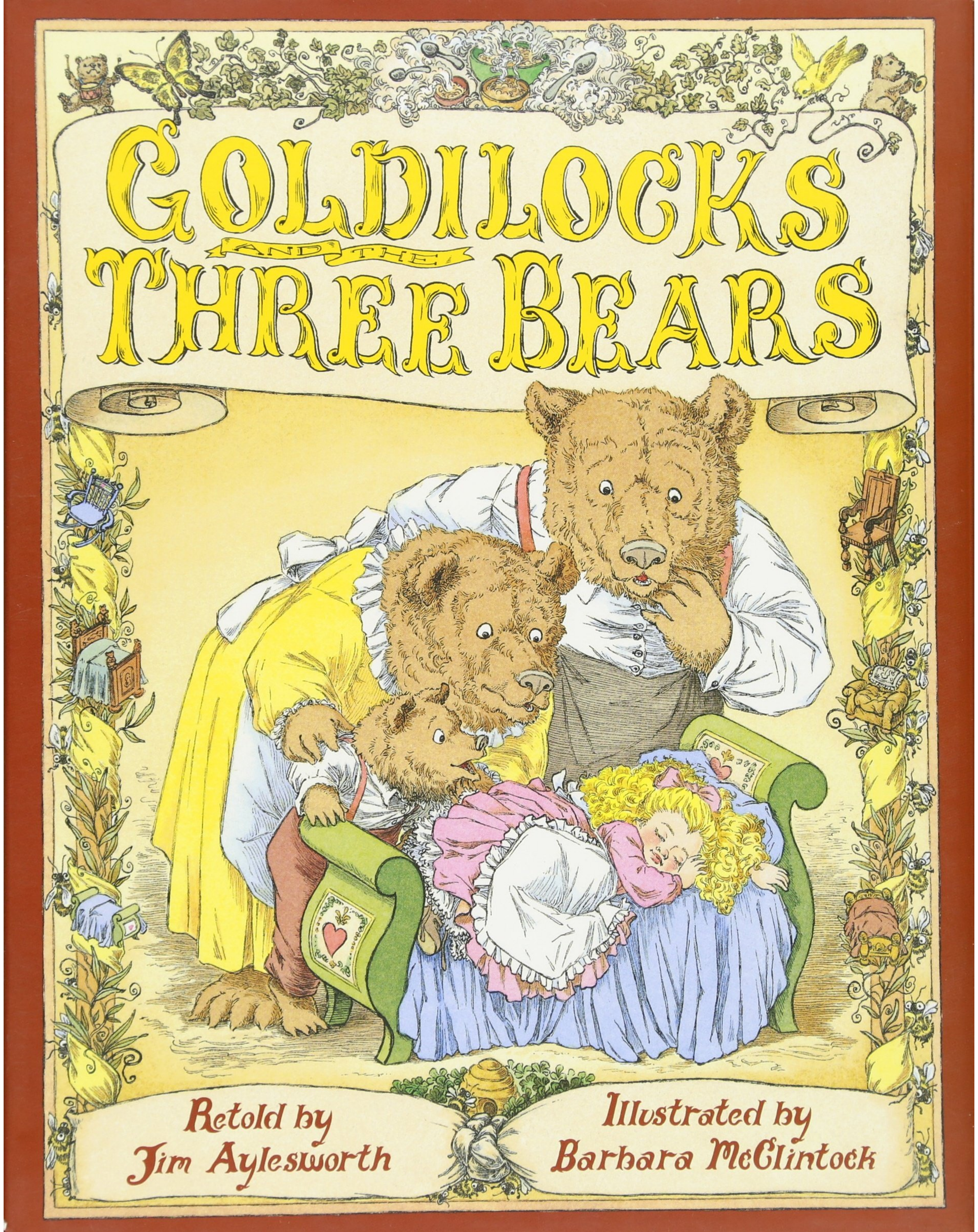
- Suppose that our hypothesis space is a set of functions parameterised by some parameter vector $\Theta = \left[\theta_1, \theta_2, \ldots \theta_m\right]$
- Let the prediction of a function f on input x_i be $f(x_i; \Theta) = \hat{y}_i$
- We have a loss function $\ell(\hat{y}, y)$ that measures how bad an approximation $\hat{y}$ is when compared to the ground truth of y
- ERM allows us to formalise learning as an optimisation problem of the form

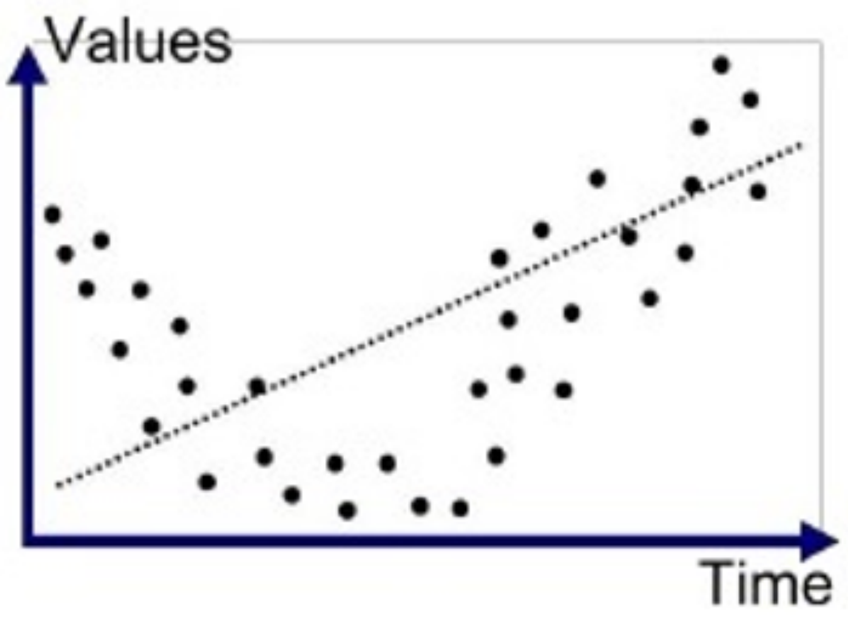
$$\min_{\Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i; \Theta), y)$$

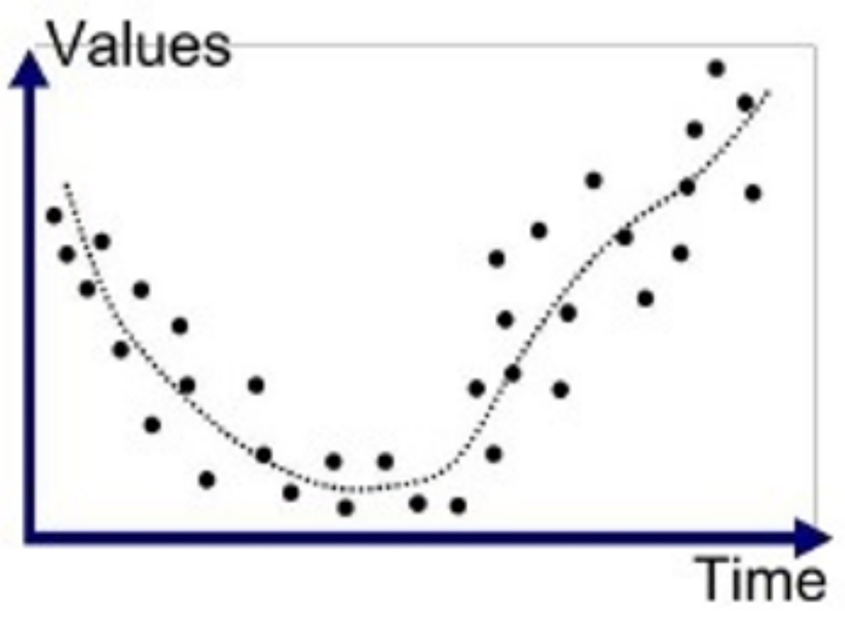
- We want to find a Θ that minimises how badly we perform on average

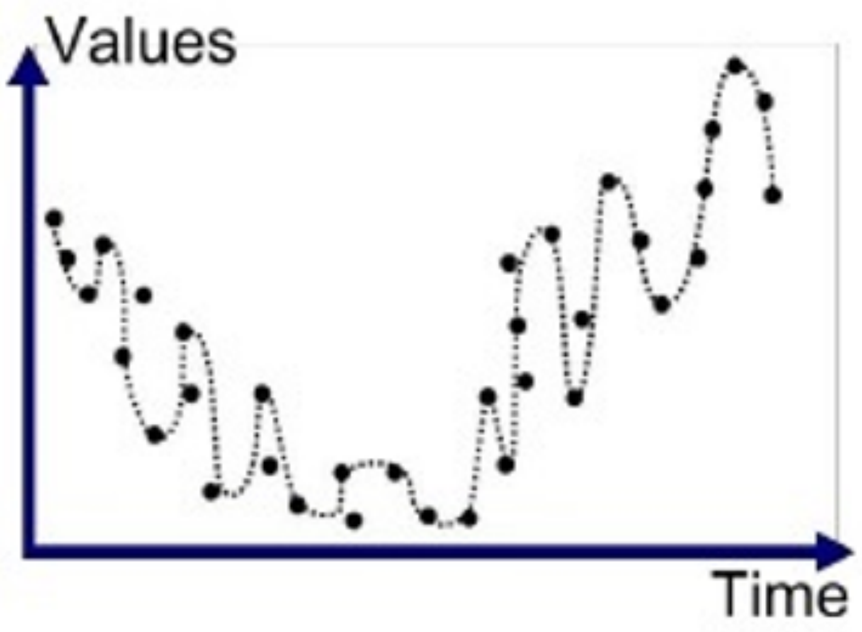
Underfitting and Overfitting

- So we used a technique to find a good set of parameters according to our hypothesis space
- How do we know our hypothesis space was suitable or not?
- Our model needs to be able to learn the relationships
- But don’t want our model to memorise relationships (think memorising selection sort without understanding how it works)


GOLDILOCKS
AND THE
THREE BEARS
Retold by
Jim Aylesworth
Illustrated by
Barbara McClintock





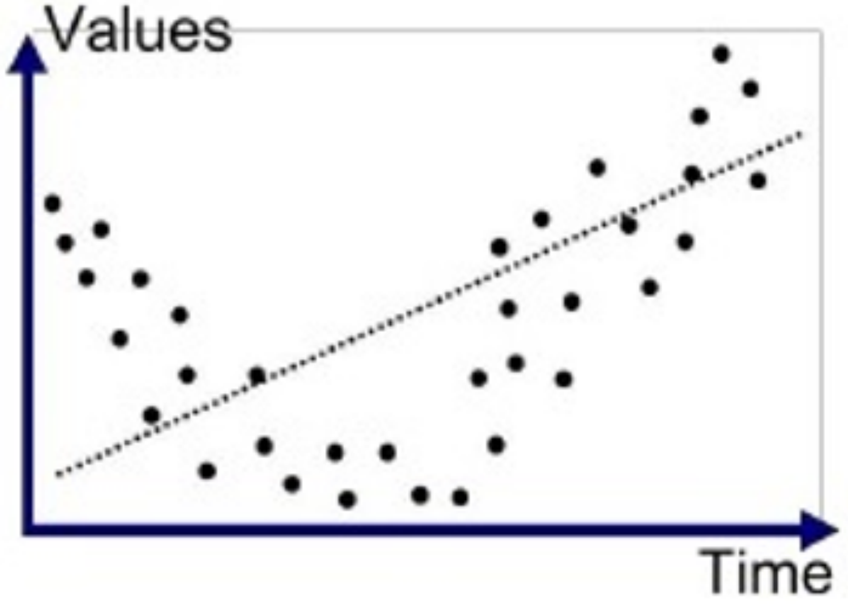


Underfitted

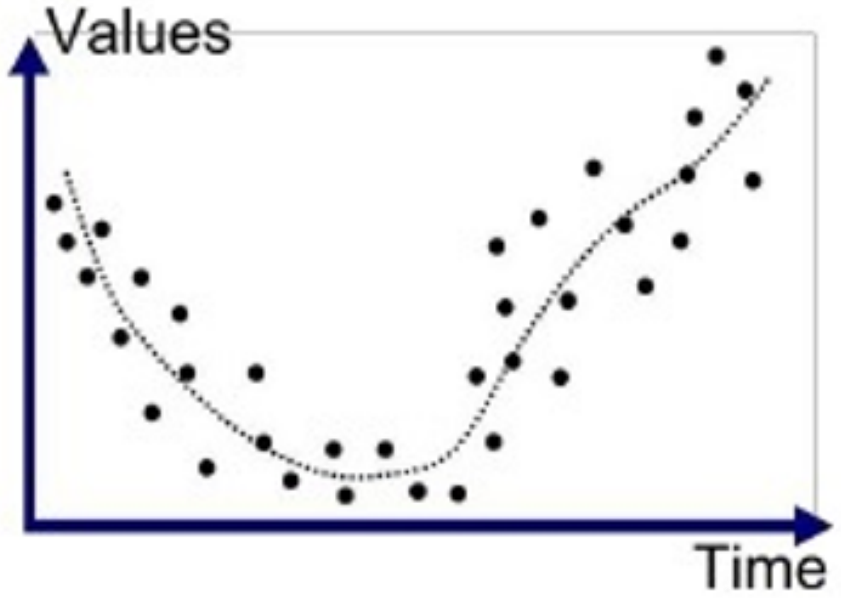
Good Fit/Robust

Overfitted

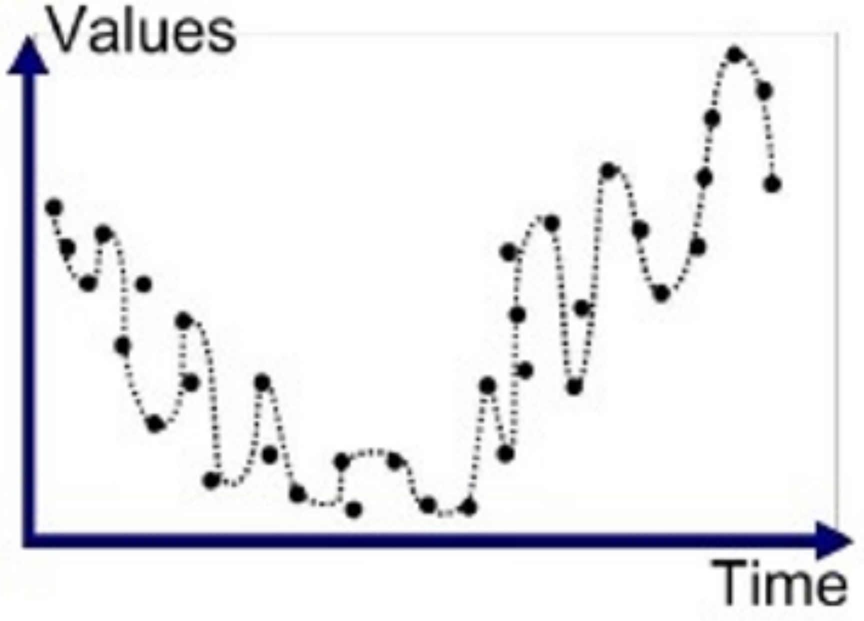
Model lacks
Flexibility to capture
relationship



Underfitted

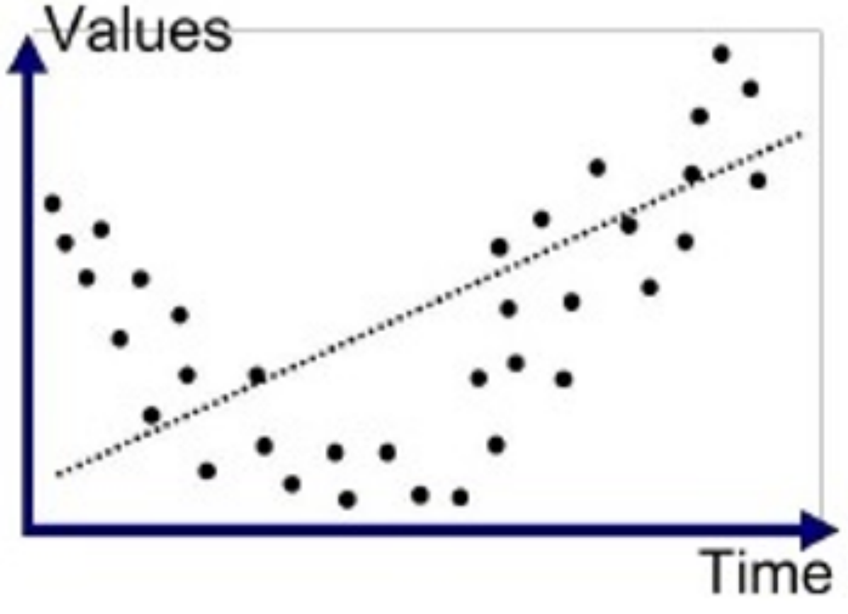


Good Fit/Robust

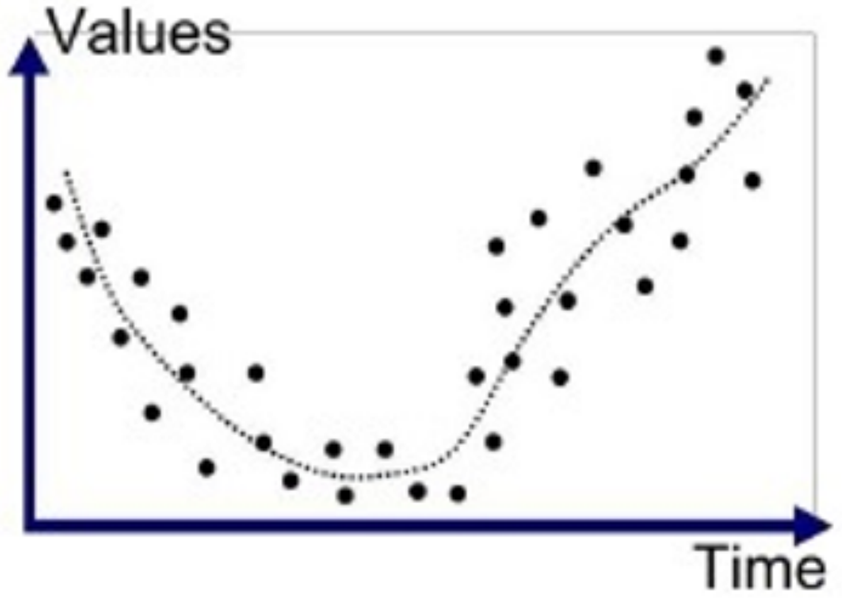


Overfitted

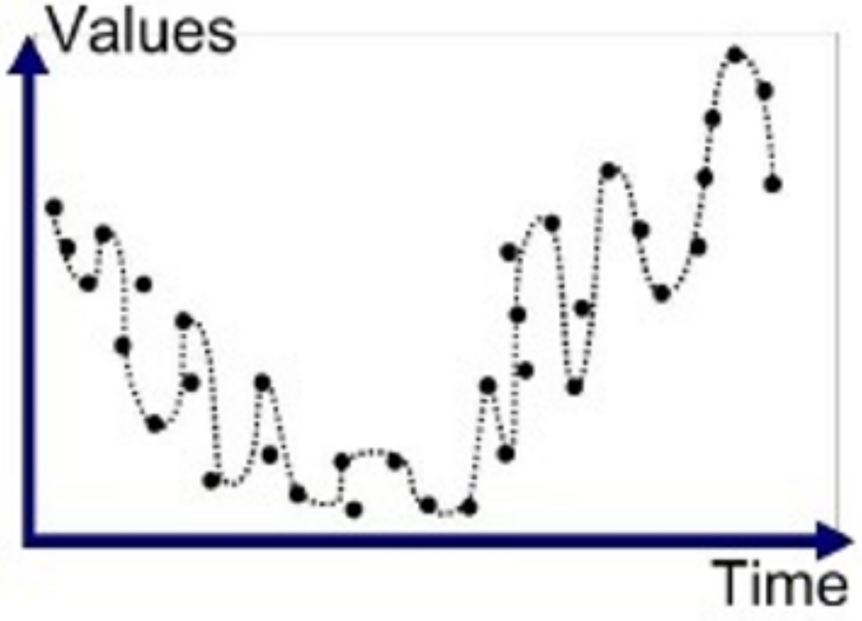
Model has too much Flexibility, so memorises data



Underfitted



Good Fit/Robust

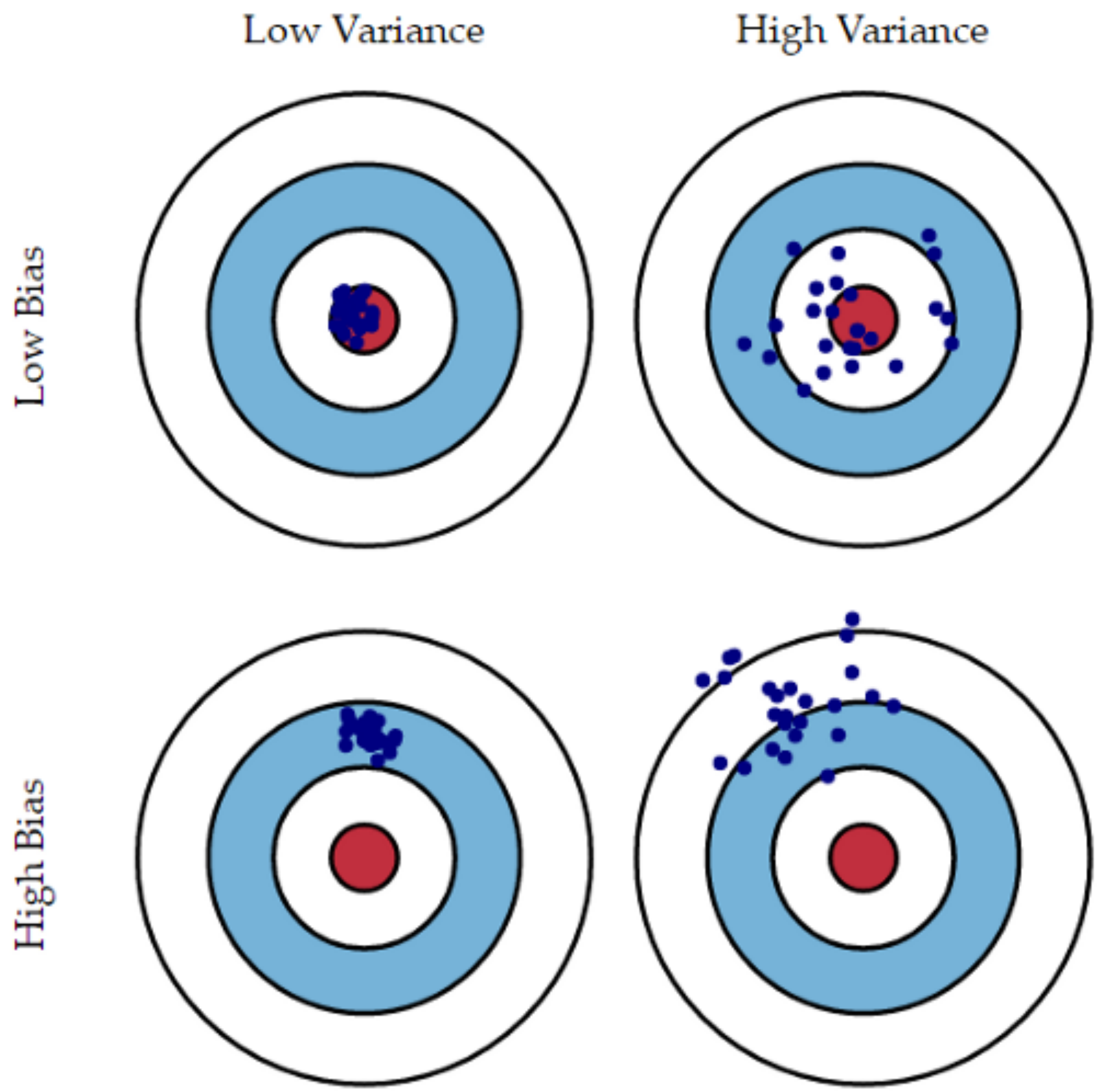


Overfitted

Just Right! :D

Variance-Bias Tradeoff

- Underfitting and overfitting are related to two statistical properties of our model - the model’s variance and the model’s bias
- Won’t belabour formalism here, but important to get intuition of these concepts
- Think of variance as representing how “sensitive” the model is to perturbations in the input
- Think of bias as representing how consistent the model is in learning incorrect concepts
- A model with fewer parameters have less variance but more bias and vice versa

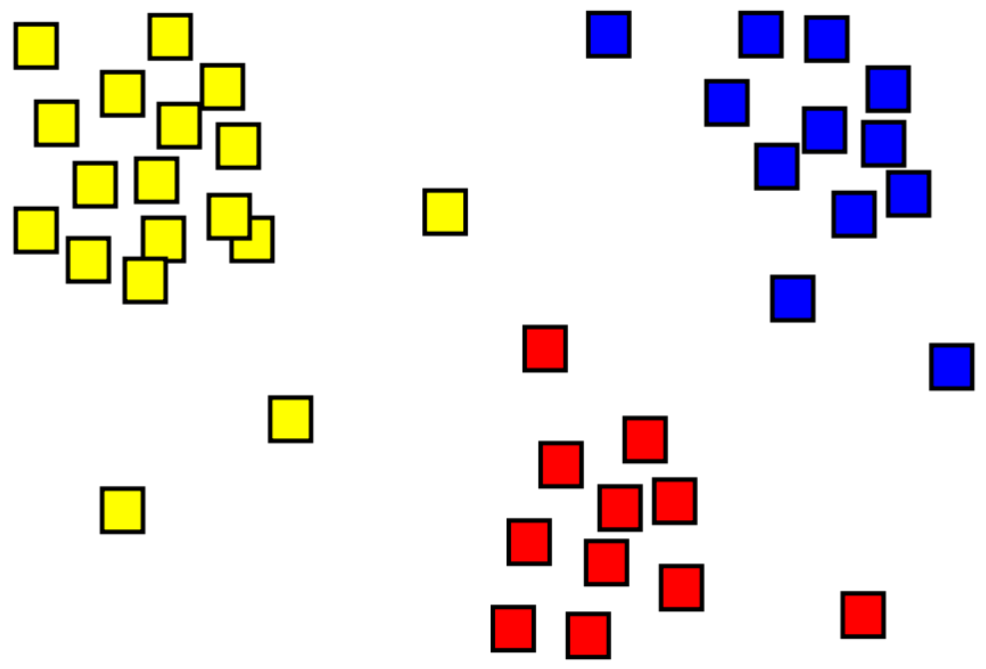

Low Variance
High Variance
Low Bias
High Bias

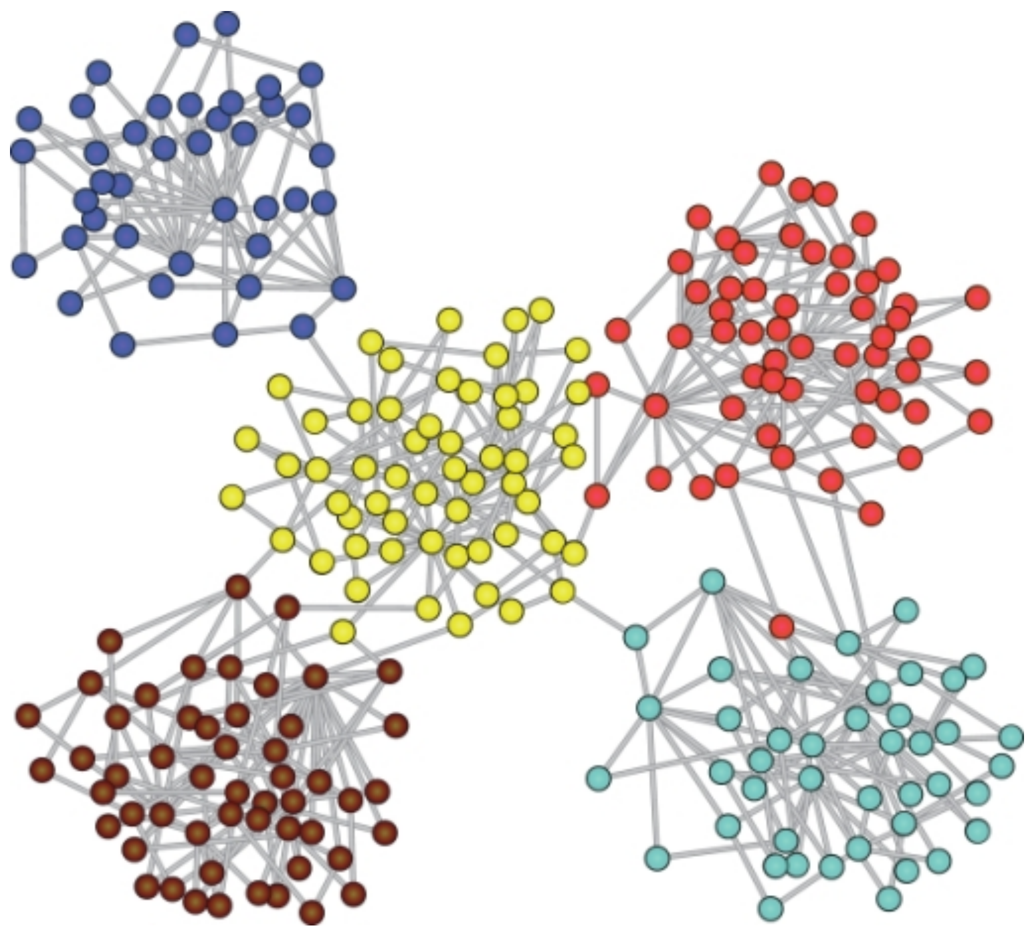
Unsupervised Learning

- Given data points without dependent variables or labels
- Want to gain insight into data or preprocess for other machine learning algorithms
- Sometimes also called knowledge discovery or pattern mining
- Several types of unsupervised learning

Clustering

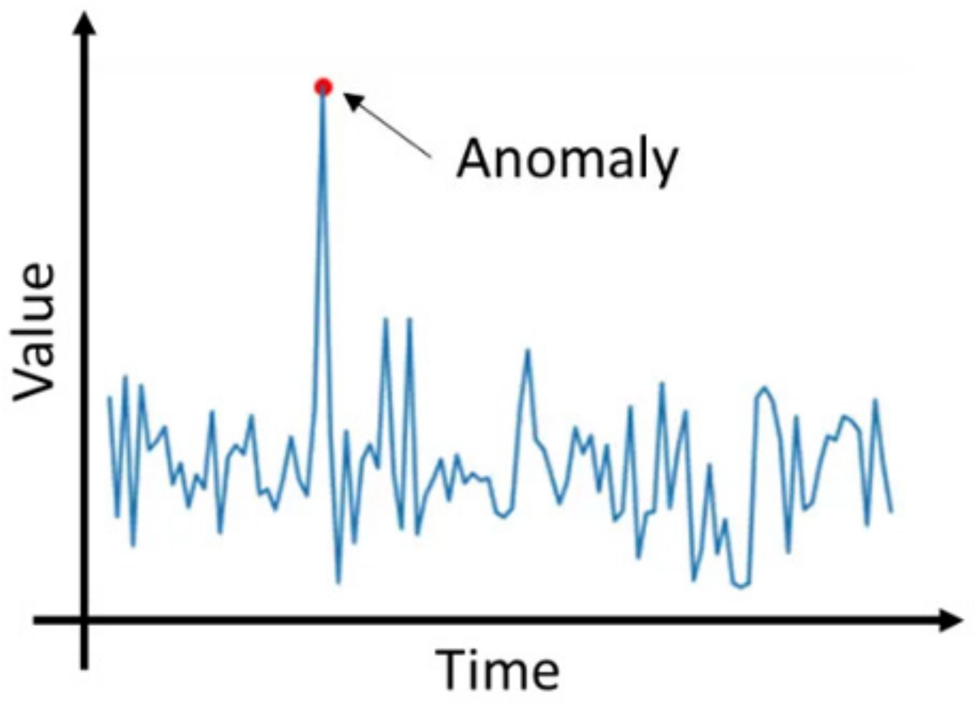
- Discover groups within data with similar characteristics
- Used in fields such as marketing, sociology, and biology
- Graph specific variant called community detection
- Techniques: K-Means, Louvain Modularity, DBSCAN, HDBSCAN





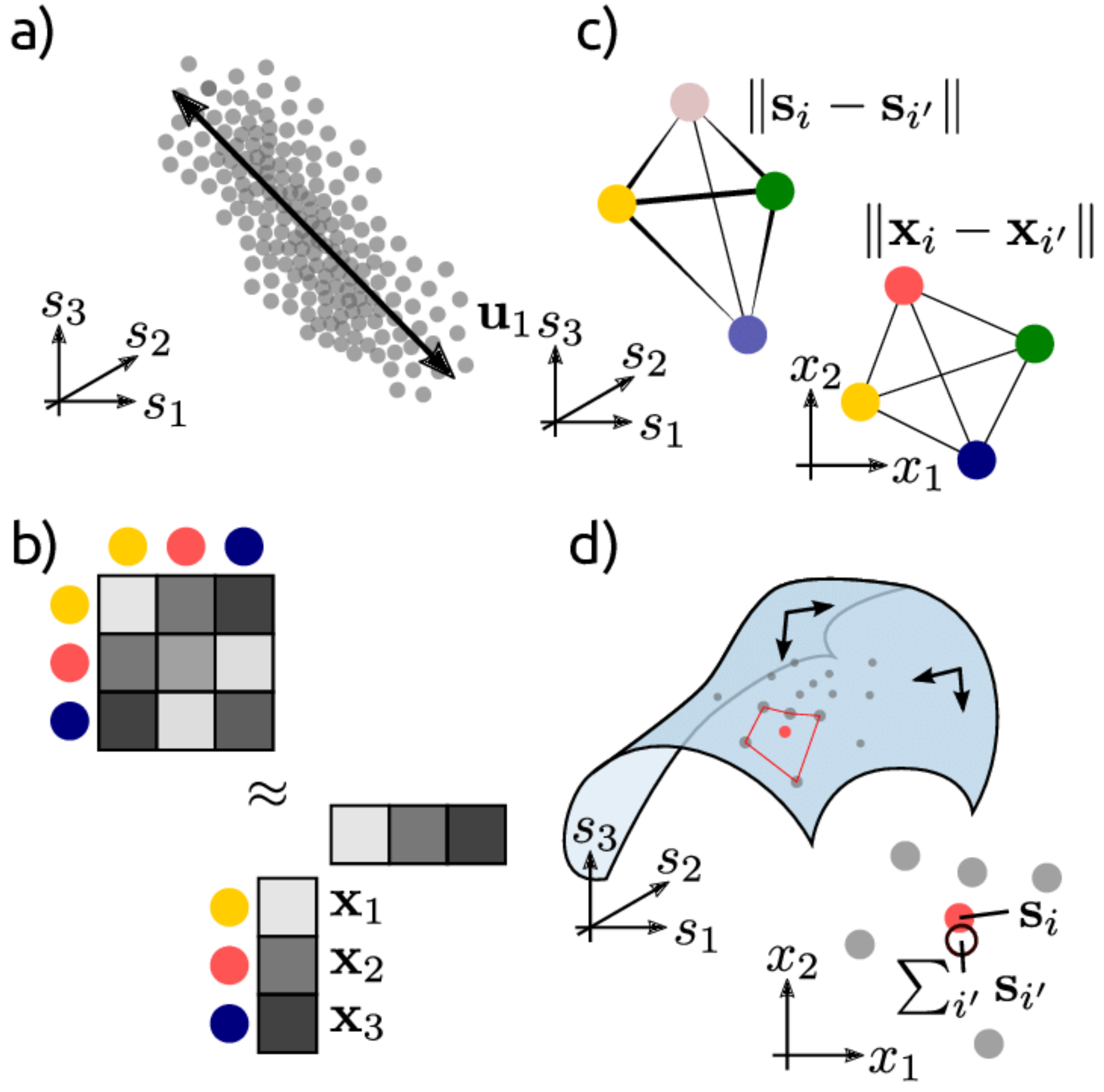
Anomaly Detection

- Detect novel or unexpected events or data points
- Used in medical data analysis and tasks such as credit card fraud detection and intrusion detection
- Can also be used in any sort of signal processing task
- Sometimes uses clustering as preprocessing step
- Examples: clustering-based, local outlier factor



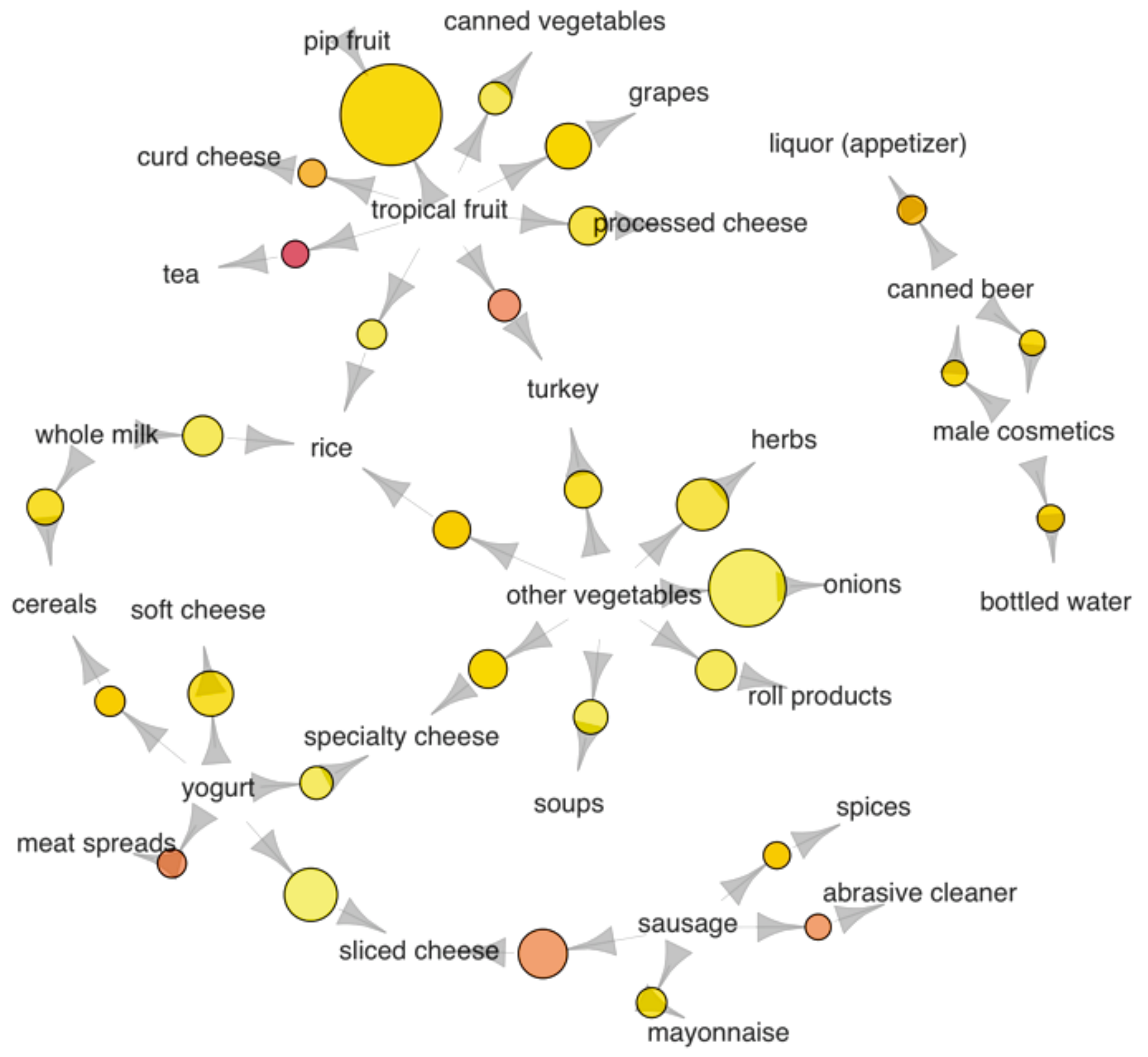
Dimensionality Reduction and Feature Learning

- Given some data, compress data into lower dimensional space
- Or learn projection of data into another space (called the latent feature space)
- Useful for data visualisation
- Or as preprocessing step
- Examples: PCA, TSNE, UMAP, Word2Vec, BERT, Node2Vec, StEM, Autoencoders
- Many use matrix factorisations or self-supervision



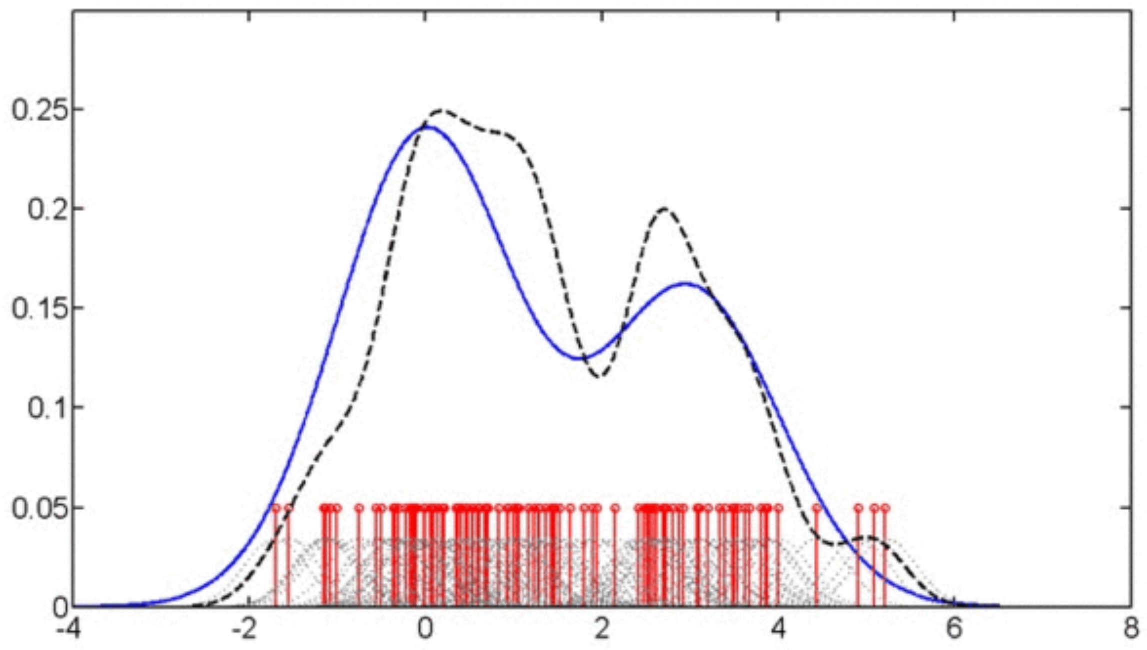
Association Rule Mining

- Find common associations between items
- Used in fields such as recommendation systems, market basket analysis, and insurance
- Examples: APriori Algorithm



Density Estimation

- Given data, try to fit data non-parametrically
- Can be used as preprocessing step to anomaly detection or to some supervised learning techniques



Reinforcement Learning

- Agents can interact with environment
  - Receives state s_t
  - Executes action a_t
  - Transitions to state s_{t+1} based on a_t and earns reward r_t
- Such as circumstance can be described by something called a Markov Decision Process (MDP)
- We want to come up with policy function π such that we earn the most rewards at the end
- Examples of use cases: Playing Atari Games, teaching robots how to walk, and protein folding

Markov Decision Process

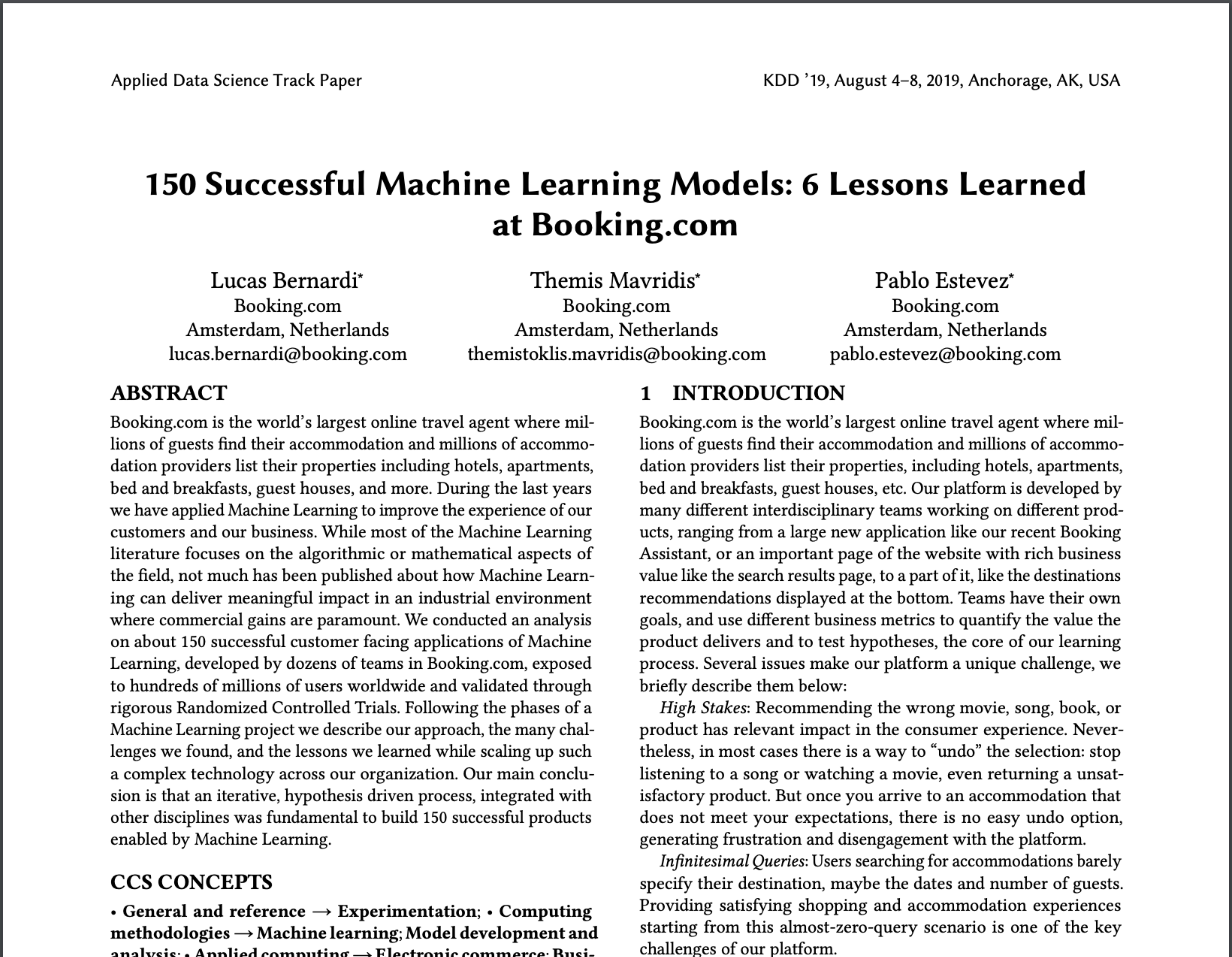
- Five tuple
  - $< S, A, \gamma, R, T >$
  - S - set of states
  - A - set of actions
  - γ - discounting factor - controls importance of long-term against short term outcomes
  - R - reward function $R : (S \times A \times S) \rightarrow \mathbb{R}$, .i.e. $R(s, a, s')$
    - Going from s, to s' by using a gives us reward $R(s, a, s')$
  - T - transition function, $T(s' | s, a)$ gives probability of entering state s' from s by taking action a

Reinforcement Learning

- Reinforcement Learning tries to learn policy function $\pi : S \rightarrow A$ that maximises reward at the end
- Examples of techniques: Q-Learning, Deep Q-Learning
- Interestingly enough, the genetic algorithm and related techniques have been show to outperform gradient based optimisation for learning π

Machine Learning Experiments

- Important to adopt proper, scientifically robust methodology to conduct machine learning
- Good experimental design is important
- Can’t always prove things formally :’(
- So, good experiments is sometimes all that we have
- Interesting research done at level of methodology!



150 Successful Machine Learning Models: 6 Lessons Learned at Booking.com

Lucas Bernardi*
Booking.com
Amsterdam, Netherlands
lucas.bernardi@booking.com

Themis Mavridis*
Booking.com
Amsterdam, Netherlands
themistoklis.mavridis@booking.com

Pablo Estevez*
Booking.com
Amsterdam, Netherlands
pablo.estevez@booking.com

ABSTRACT

Booking.com is the world’s largest online travel agent where millions of guests find their accommodation and millions of accommodation providers list their properties including hotels, apartments, bed and breakfasts, guest houses, and more. During the last years we have applied Machine Learning to improve the experience of our customers and our business. While most of the Machine Learning literature focuses on the algorithmic or mathematical aspects of the field, not much has been published about how Machine Learning can deliver meaningful impact in an industrial environment where commercial gains are paramount. We conducted an analysis on about 150 successful customer facing applications of Machine Learning, developed by dozens of teams in Booking.com, exposed to hundreds of millions of users worldwide and validated through rigorous Randomized Controlled Trials. Following the phases of a Machine Learning project we describe our approach, the many challenges we found, and the lessons we learned while scaling up such a complex technology across our organization. Our main conclusion is that an iterative, hypothesis driven process, integrated with other disciplines was fundamental to build 150 successful products enabled by Machine Learning.

CCS CONCEPTS

• **General and reference** → **Experimentation**; • **Computing methodologies** → **Machine learning**; **Model development and analysis**; • **Applied computing** → **Electronic commerce**; **Busi-**

1 INTRODUCTION

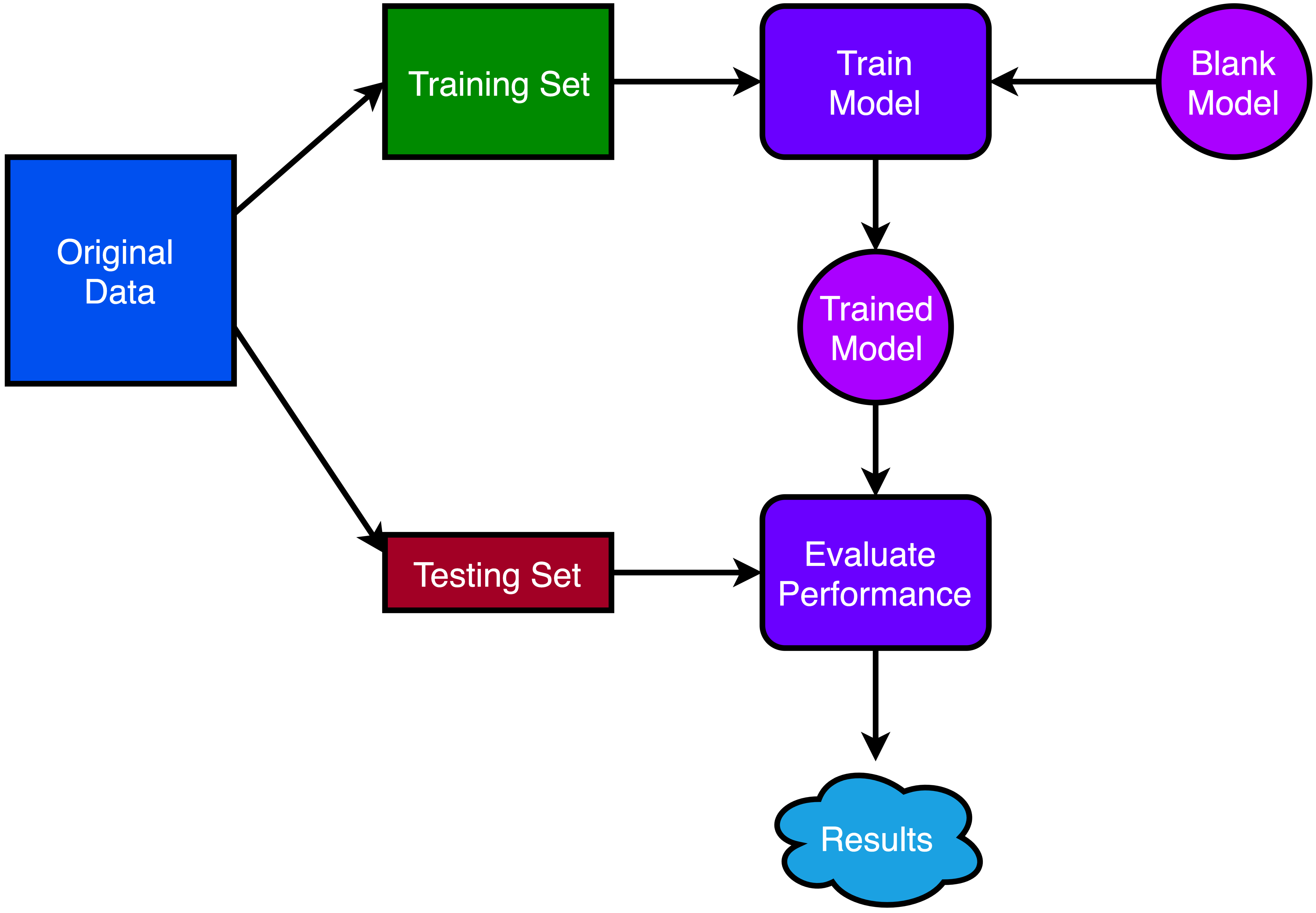
Booking.com is the world’s largest online travel agent where millions of guests find their accommodation and millions of accommodation providers list their properties, including hotels, apartments, bed and breakfasts, guest houses, etc. Our platform is developed by many different interdisciplinary teams working on different products, ranging from a large new application like our recent Booking Assistant, or an important page of the website with rich business value like the search results page, to a part of it, like the destinations recommendations displayed at the bottom. Teams have their own goals, and use different business metrics to quantify the value the product delivers and to test hypotheses, the core of our learning process. Several issues make our platform a unique challenge, we briefly describe them below:

High Stakes: Recommending the wrong movie, song, book, or product has relevant impact in the consumer experience. Nevertheless, in most cases there is a way to “undo” the selection: stop listening to a song or watching a movie, even returning a unsatisfactory product. But once you arrive to an accommodation that does not meet your expectations, there is no easy undo option, generating frustration and disengagement with the platform.

Infinitesimal Queries: Users searching for accommodations barely specify their destination, maybe the dates and number of guests. Providing satisfying shopping and accommodation experiences starting from this almost-zero-query scenario is one of the key challenges of our platform.

Training Set and Testing Set

- Remember that our models are supposed to work for a population of scenarios
- But only have samples (we hope are representative)
- How can we conclude that our supervised learning models work well?


Original Data
Training Set
Testing Set
Train Model
Blank Model
Trained Model
Evaluate Performance
Results

Evaluating Performance

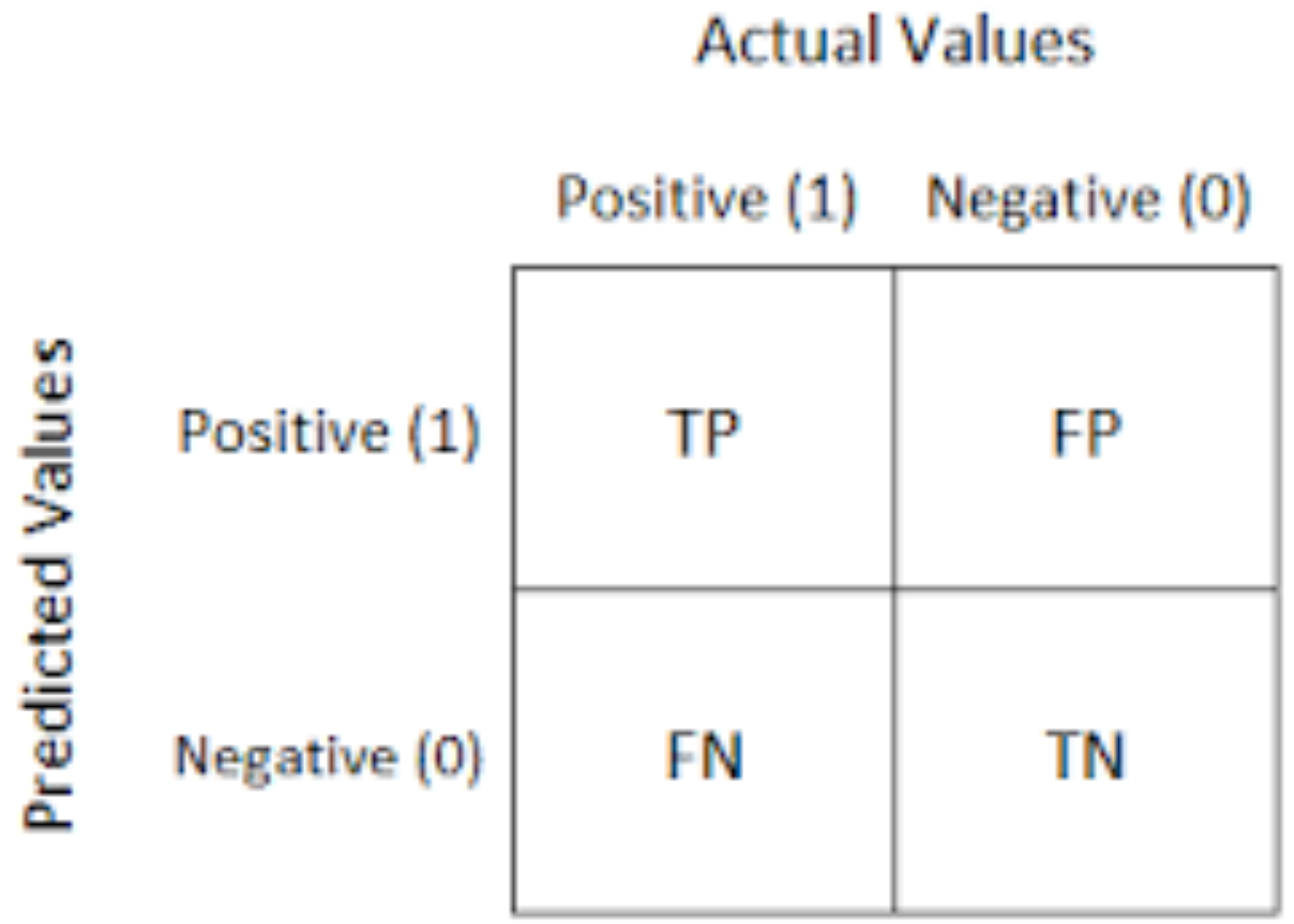
- Need to use appropriate way to measure model performance on testing set
- The appropriate metric depends on the problem type and characteristics
- Note the metrics and the loss function used are not necessarily the same!

Metrics - Regression

- MSE - Mean Square Error
  - $$\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$
- MAE - Mean Absolute Error
  - $$\frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$
- MAPE - Mean Absolute Percentage Error
  - $$\frac{1}{n}\sum_{i=1}^{n}\left(\frac{|\hat{y}_i - y_i|}{y_i}\times 100\right)$$

Metrics - Classification

- Many (binary) classification metrics are based around confusion matrices



		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

To Be Continued Next Week

- Next Week Will look at how to compute metrics for classification problems!
- Will look at the weaknesses of typical training set-testing set methodology and how we can overcome it
- Simply Hypothesis Testing for Machine Learning
    - Student-t test
    - Wilcox-Rank Signed Test

Enjoy your long-weekend :D