Introduction

Francisco Pereira

Filipe Rodrigues

DTU Management Engineering
Department of Management Engineering

Welcome!

Francisco Camara Pereira
Professor

DTU MANAGEMENT
DTU Management

Transportmodelling
Transport DTU
Transport

Technical University of Denmark
Bygningstorvet
Building 116, room 123A
2800 Kgs. Lyngby

Ph.	+45 45 25 14 96
E-mail	camara@dtu.dk
ORCID	0000-0001-5457-9909

- Office hours: Thursdays, 14:00-16:00

Filipe Rodrigues
Assistant Professor

DTU MANAGEMENT
DTU Management

Transportmodelling
Transport DTU
Transport

Technical University of Denmark
Bygningstorvet
Building 116, room 121A
2800 Kgs. Lyngby

Ph.	+45 45 25 65 30
E-mail	rodr@dtu.dk
ORCID	0000-0001-6979-6498

- Office hours: Tuesdays, 14:00-16:00

TAs

(a) Daniele Gammelli

(b) Sergio Garrido

(c) Mathias Tygesen

(d) Nikolaos Nakis

Outline

- Motivation
- A practical example
- Probabilistic programming
- Structure
  - Syllabus
  - Method
  - Requirements
  - Evaluation

Motivation - focus on the problem!

- Traditional machine learning
  - A very wide range of algorithms (k-Means, NN, SVM, PCA,...)
  - When faced with a new problem, map it onto one of the existing machine learning methods
  - **Focus is on algorithms**
- Model-based machine learning (MBML)
  - No need to learn about the huge range of traditional methods
  - When faced with a new problem, think about
    - the number and types of variables in the problem domain
    - which variables affect each other
    - what the effect of changing one variable is on another variable
    - encode that knowledge in the form of a model
  - **Focus is on the problem**
    - includes all the assumptions about the problem domain
    - highly tailored models for specific scenarios

MBML in a nutshell

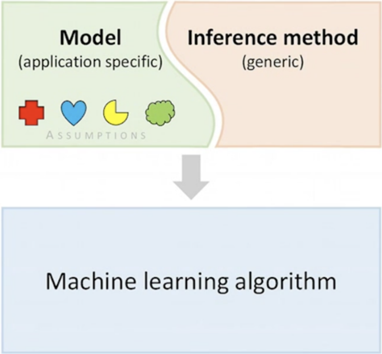


Figure: MBML big picture[1]

[1] In Christopher Bishop's Keynote, "Model-Based Machine Learning"
https://www.microsoft.com/en-us/research/video/keynote-model-based-machine-learning

Motivation - Why probabilistic models?

- **Uncertainty** in the real world
  - partial knowledge of state of the world
  - noisy observations
  - inherent stochasticity
- It is essential to account for uncertainty when building models of reality!
- **Probability theory** provides us with a consistent framework for quantifying and manipulating uncertainty

Motivation - A recent example...

Motivation - A recent example...

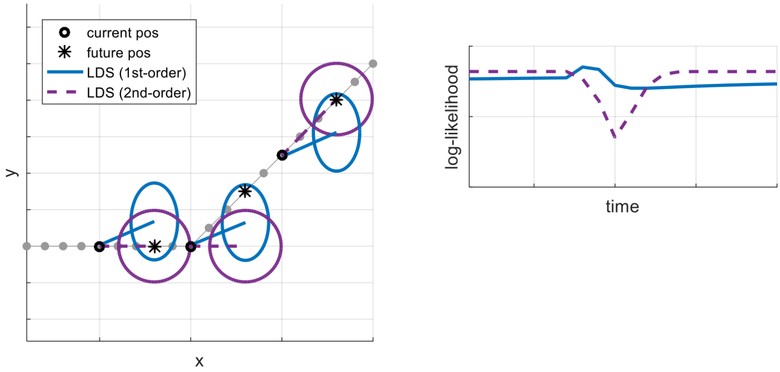
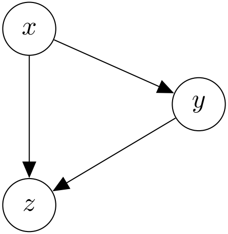


Figure: From Julian Kooij, TUDelft

Motivation - A recent example...

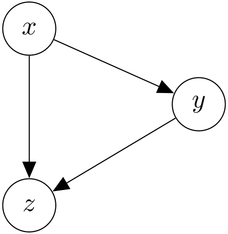
- Video

Motivation - Why probabilistic graphical models (PGMs)?



- **Nodes** - Represent random variables
- **Arrows** - Represent causal relationships

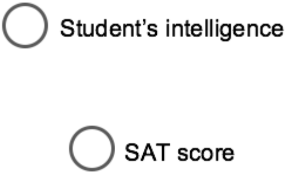
Motivation - Why probabilistic graphical models (PGMs)?



- Provide an **intuitive** and **compact** way of representing the structure of a probabilistic model
  - gives us insights about the properties of the model (e.g. relationships between variables, conditional independencies)
  - helps us communicate and design new models

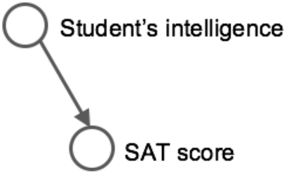
A practical example

- A recruiter looking for the most intelligent students



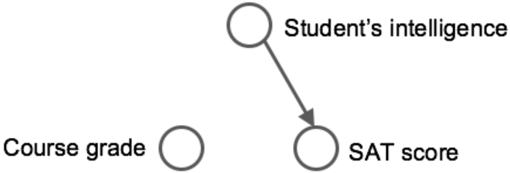
A practical example

- A recruiter looking for the most intelligent students



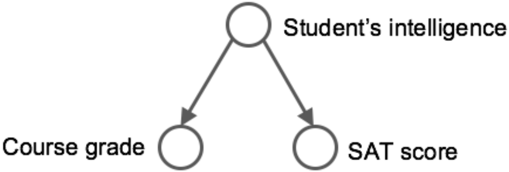
A practical example

- A recruiter looking for the most intelligent students



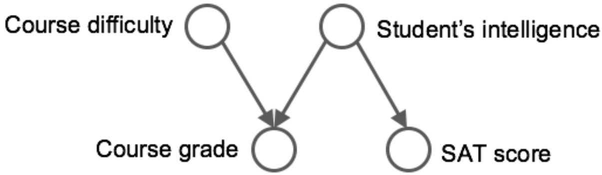
A practical example

- A recruiter looking for the most intelligent students



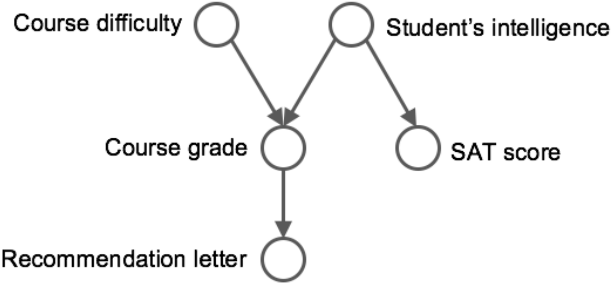
A practical example

- A recruiter looking for the most intelligent students



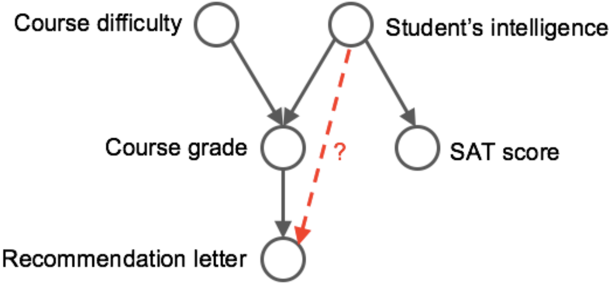
A practical example

- A recruiter looking for the most intelligent students



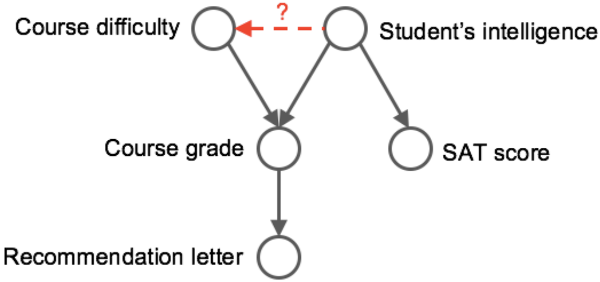
A practical example

- A recruiter looking for the most intelligent students



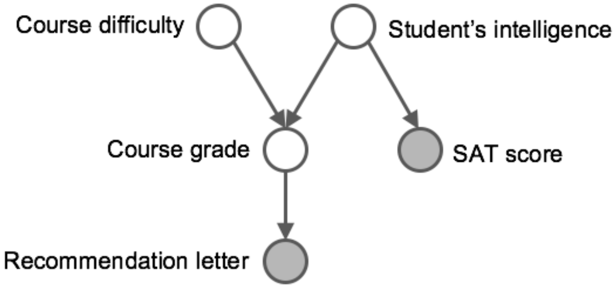
A practical example

- A recruiter looking for the most intelligent students



A practical example

- A recruiter looking for the most intelligent students



Playtime!

- Your first exercise!
- In pairs, build a graphical model that you think best represents this problem involving the following variables:
  - Climate change
  - GHG emissions
  - Energy demand
  - Earth's natural variability
  - Food supply
  - Population growth
  - Water quality
- Duration: 5 min.
- Exchange your graphical models between groups (and check if they are similar...)

A (familiar) example - Linear regression

- y - target (a.k.a. response or dependent variable)
- $\mathbf{x}$ - vector of explanatory variables
  (a.k.a. predictor or independent variables)
- Target y is assumed to be a function of $\mathbf{x}$

$$y = f(\mathbf{x})$$

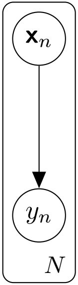
- Linear regression assumes f to be a **linear function** of $\mathbf{x}$

$$y = \boldsymbol{\beta}^T \mathbf{x} + \epsilon$$

- $\boldsymbol{\beta}$ is a vector of coefficients
- $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is an error term (accounts for observation noise)

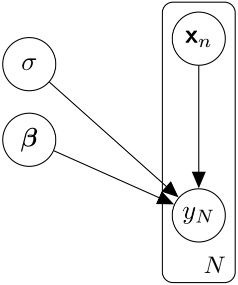
Linear regression

- Linear regression as a graphical model
  - We have a set of N observations of the targets y_n which depend on their corresponding explanatory variables $\mathbf{x}_n$



Linear regression

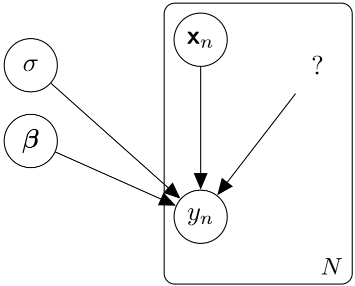
- Linear regression as a graphical model
  - We have a set of N observations of the targets y_n which depend on their corresponding explanatory variables $\mathbf{x}_n$



- We need to include parameters

Linear regression

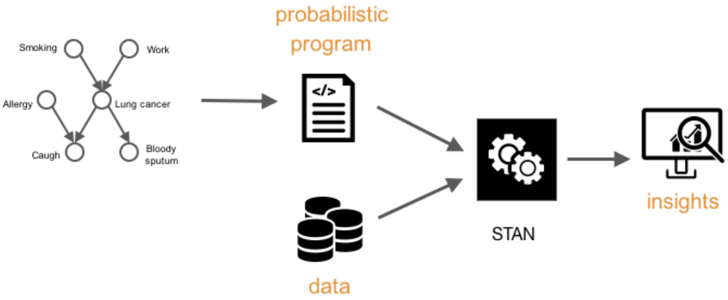
- Linear regression as a graphical model
  - We have a set of N observations of the targets y_n which depend on their corresponding explanatory variables $\mathbf{x}_n$



- In fact, we can even include other models (within our model)!

$$y = \boldsymbol{\beta_{lr}}^T \mathbf{x} + \boldsymbol{\beta_k} \textit{kernel\_model} + \boldsymbol{\beta_{dl}} \textit{deep\_learning\_model} \ldots$$

Probabilistic programming



- Popular probabilistic programming languages
  - **STAN** - we will use this throughout the course
  - **Pyro** - NEW! we will have Pyro versions of the notebooks (experimental)
  - PyMC3
  - Edward2

About STAN and Pyro

- **STAN** is a state-of-the-art platform for statistical modeling and high-performance statistical computation
- It allows for specification of a graphical model
- We will use STAN quite a bit in our course - you'll see it's pretty easy! :-)
- Check http://mc-stan.org

- **Pyro** is a universal probabilistic programming language written in Python and supported by PyTorch on the backend
- Pyro enables flexible and expressive deep probabilistic modeling
- Unifies the best of modern deep learning and Bayesian modeling
- Check https://pyro.ai

Example with Linear Regression

- Linear Regression with STAN:

```
model="""
data {
    int<lower=0> N; // data points
    vector[N] x;    // independent variable
    vector[N] y;    // depedent variables
}

parameters {
    real beta0;             // intercept
    real beta1;             // coefficient of x
    real<lower=0> sigma;    // observation noise parameter
}

model {
    y ~ normal(beta0 + beta1*x, sigma);
}
"""
```

- Linear Regression with Pyro:

```
def model(x, y):
    beta0 = pyro.sample("beta0", dist.Normal(0., 1.))   // intercept
    beta1 = pyro.sample("beta1", dist.Normal(0., 1.))   // coefficient of x
    sigma = pyro.sample("sigma", dist.Uniform(0., 10.)) // observation noise
    with pyro.plate("data", N):
        pyro.sample("obs", dist.Normal(beta0 + beta1*x, sigma), obs=y)
```

Example with Linear Regression

- Running inference in Python+STAN:

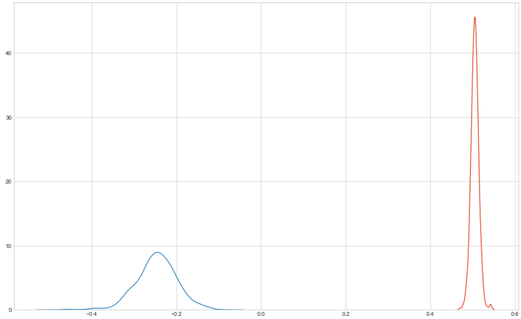
```
data_dict = {'N': N,
             'x': x,
             'y': y}

sm = pystan.StanModel(model_code=model)
fit = sm.sampling(data=data_dict, iter=1000, chains=4)
print(fit)
```

- Running inference in Pyro:

```
from pyro.infer import MCMC, NUTS
nuts_kernel = NUTS(model)
mcmc = MCMC(nuts_kernel, num_samples=1000, warmup_steps=200)
mcmc.run(x, y)
```

Example with Linear Regression

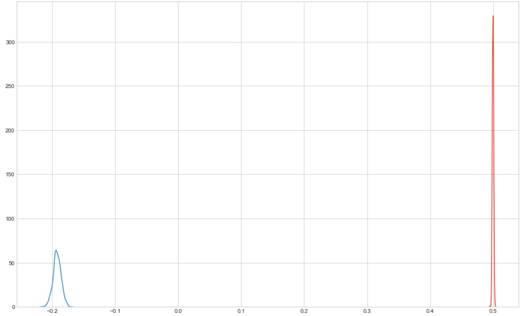
- Running inference... (N=10)



	mean
beta1	0.51
beta0	-0.25
sigma	0.1

Example with Linear Regression

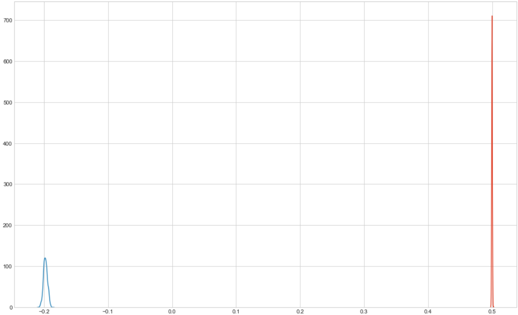
- Running inference... (N=200)



	mean
beta1	0.5
beta0	-0.19
sigma	0.1

Example with Linear Regression

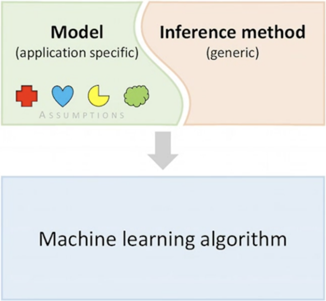
- Running inference... (N=1000)



	mean
beta1	0.5
beta0	-0.2
sigma	0.1

Course structure

- 3 main blocks:
  - Foundations of probabilistic graphical models (PGMs) and probabilistic programming - 4 lectures
  - Probabilistic modelling in various contexts (with practical examples of popular models) - 5 lectures
  - Bayesian inference (exact and approximate) - 3 lectures
- These are the key ingredients of MBML!



Syllabus: Lectures

1. Intro to MBML + probability and statistics review (F. Pereira)
2. PGM foundations 1 (F. Pereira)
3. PGM foundations 2 + Mixture models (F. Pereira)
4. Frequentist vs Bayesian + Prob. Programming (F. Rodrigues)
5. Regression models (F. Rodrigues)
6. Classification models (F. Rodrigues)
7. Temporal models (F. Rodrigues)
8. Topic models (F. Rodrigues)
9. Exact inference (F. Rodrigues)
10. Markov-chain Monte Carlo (F. Rodrigues)
11. Variational inference (F. Rodrigues)
12. Gaussian processes (F. Rodrigues)

What we expect you to get out of this course

- In-depth understanding of Probabilistic Graphical Models and the Bayesian framework
    - Easily understand new models (e.g. by relating them with others)
    - New unifying perspective to models that you already know
    - Exploit your domain knowledge to build new models
    - Break out of the suite of the standard ML algorithms
- Good knowledge of a broad class of probabilistic modelling approaches
    - Modelling ideas that you can "re-use"
    - Great starting point when tackling a new problem
- Experience with a state-of-the-art probabilistic programming language
    - Easily specify a model and run inference on it
    - Flexibility to try new ideas/variants
- Excitement about MBML and its application to solve real-world problems :-)

What we DON'T expect from you

- To know in detail every single topic mentioned
  - Curriculum is large, but we try to be reasonable!
  - Many lectures could be the topic of an entire course
  - Some topics we require that you know in depth
  - Other topics we will just cover at the conceptual level and provide pointers to where to learn more

Method

- Interleaved blocks of theory and practice
  - Theory: slides
  - Practice: Jupyter notebooks; Python and STAN/Pyro

Requirements

- Programming is important (preferably Python)
- Confident knowledge about basic statistics and probability
- Creativity and domain knowledge

Evaluation

- 2 Tests (approx. 1h duration; with aid), 25% each (March 20, May 12)
- Project, 50% (deadline: TBD)