Optimization Methods & Game Theory: applications

Antonio Frangioni

Department of Computer Science
University of Pisa
https://www.di.unipi.it/~frangio
mailto:frangio@di.unipi.it

Optimization Methods and Game Theory
Master in Artificial Intelligence and Data Engineering
University of Pisa

A.Y. 2022/23

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

Motivation I, reprise III: Clustering

Wrap up I, II, and III, References

Solutions

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

Motivation I, reprise III: Clustering

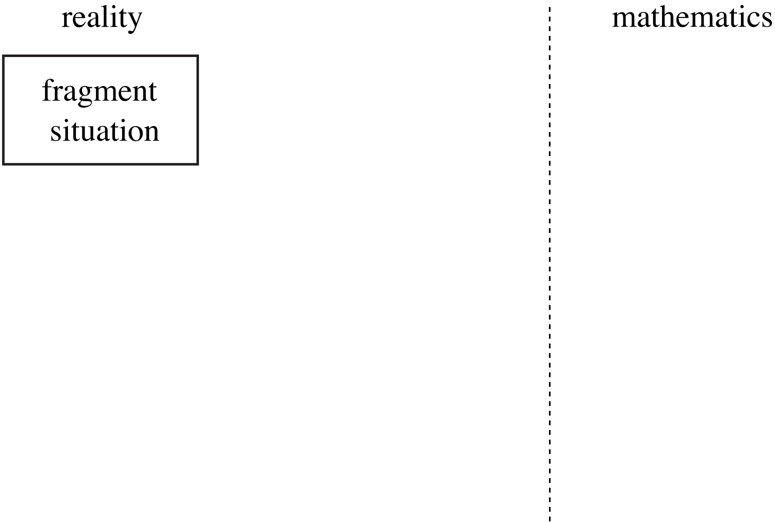
Wrap up I, II, and III, References

Solutions

Why this course: the view from the stratoshpere 1

- Huge amounts of data is generated and collected, but one has to make sense of it in order to use it: that's what data science is
- Take something big (data) and therefore unwieldy and produce something small and nimble that can be used in its stead ("actionable")
- That's a (mathematical) model
- Word comes from "modulus", diminutive from "modus" = "measure": "small measure", "measure in the small" (small is good)
- Known uses in architecture: proving beforehand that the real building won't collapse (e.g., Filippo Brunelleschi for the Cupola of the Cathedral of Florence)
- Countless many physical models afterwards (planes, cars, ...), but mathematics is cheaper than bricks / wood / iron ...
- Yet, mathematical problems can be difficult, too, for various reasons (and, of course, only truly viable after computers)
- And most of them remain (likely) difficult for quantum computers, too, `https://www.smbc-comics.com/comic/the-talk-3`

That’s how (Data) Science works, a.k.a., “the decision process”



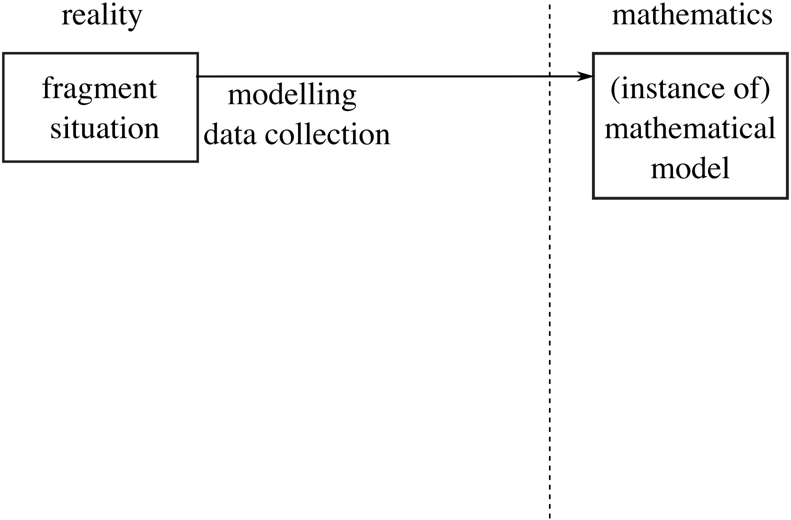
reality

mathematics

fragment situation

- The fundamental cycle of all science

That’s how (Data) Science works, a.k.a., “the decision process”



reality

mathematics

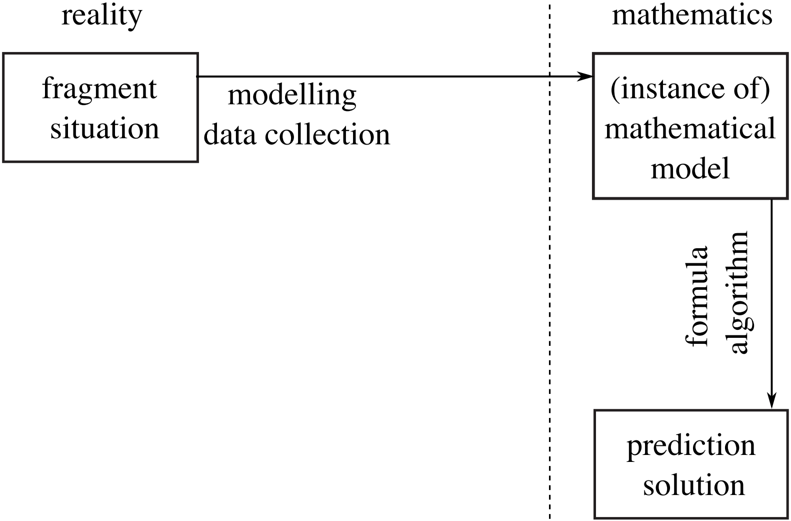
fragment situation

modelling data collection

(instance of) mathematical model

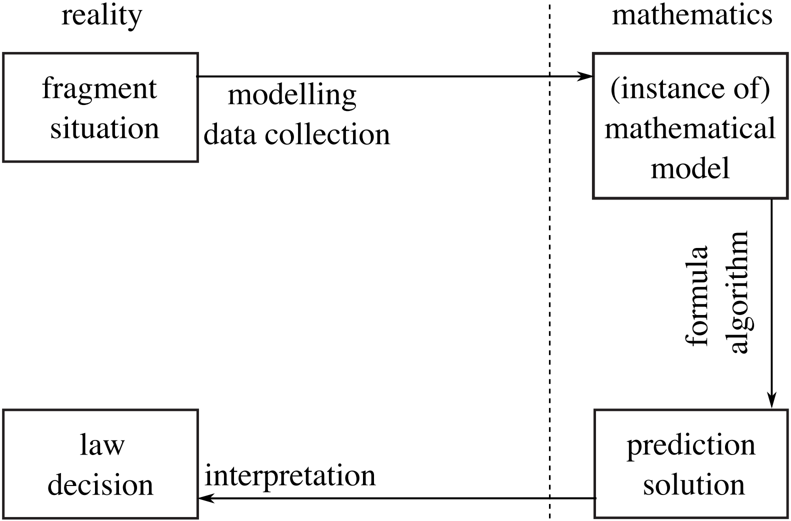
- The fundamental cycle of all science

That's how (Data) Science works, a.k.a., "the decision process"



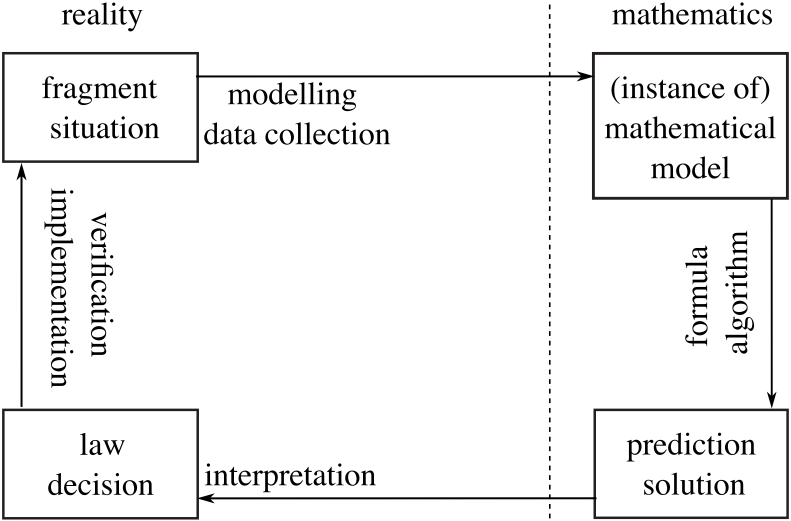
- The fundamental cycle of all science

That's how (Data) Science works, a.k.a., "the decision process"



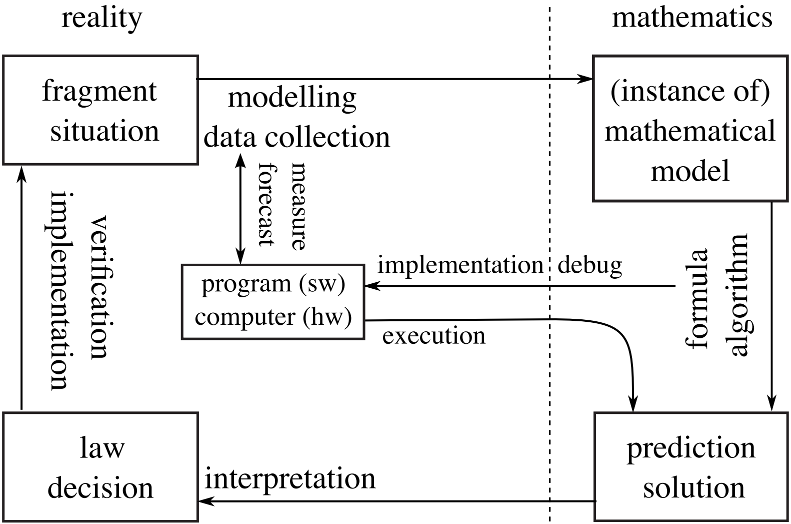
- The fundamental cycle of all science

That’s how (Data) Science works, a.k.a., “the decision process”



- The fundamental cycle of all science

That’s how (Data) Science works, a.k.a., “the decision process”



- The fundamental cycle of all science and its implementation

Choosing a mathematical model

- How a mathematical model should be:

Choosing a mathematical model

- How a mathematical model should be:
    1. accurate (describes well the process at hand)

Choosing a mathematical model

- How a mathematical model should be:
    1. accurate (describes well the process at hand)
    2. computationally inexpensive (gives answers rapidly)

Choosing a mathematical model

- How a mathematical model should be:
  1. accurate (describes well the process at hand)
  2. computationally inexpensive (gives answers rapidly)
  3. general (can be applied to many different processes)

Choosing a mathematical model

- How a mathematical model should be:
  1. accurate (describes well the process at hand)
  2. computationally inexpensive (gives answers rapidly)
  3. general (can be applied to many different processes)

  Typically impossible to have all three!

Choosing a mathematical model

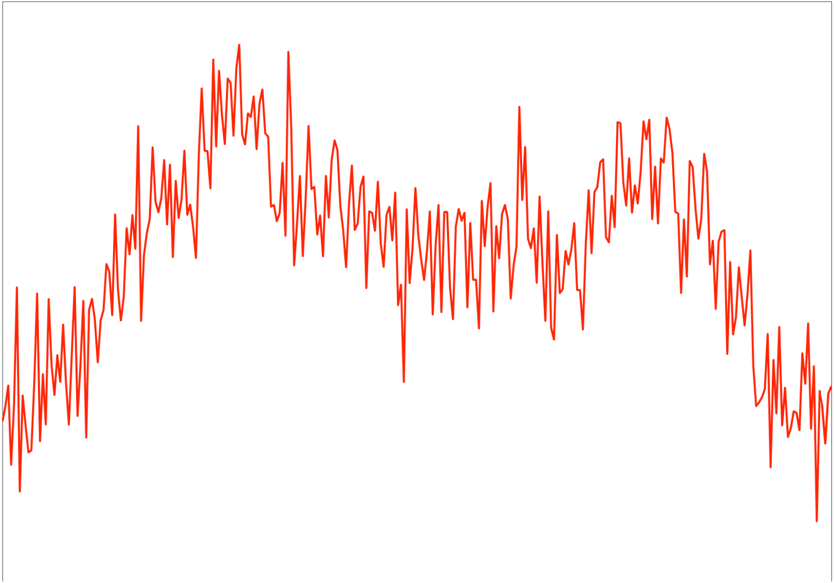
- How a mathematical model should be:
  1. accurate (describes well the process at hand)
  2. computationally inexpensive (gives answers rapidly)
  3. general (can be applied to many different processes)

  Typically impossible to have all three!
- Choice of the model crucial, trade-offs between efficiency and effectiveness
- Two fundamentally different model building approaches:
  1. analytic: model each component of the system separately + their interactions, (≈)accurate but hard to construct (need system access + technical knowledge)
  2. data-driven / synthetic: don't expect the model to closely match the underlying system, just to be simple and to (≈)accurately reproduce its observed behaviour
- All models are approximate (the map is not the world), for different reasons:
  - analytic models: flexible shape, (relatively) few "hand-chosen" parameters
  - synthetic models: rigid shape, (very) many automatically chosen parameters
- Fitting: find the parameters of the model that best represents the phenomenon, clearly some sort of optimization problem (often a computational bottleneck)
- But AI/ML $\gg$ fitting: fitting minimizes training error $\equiv$ empirical risk, but AI/ML aims at minimizing test error $\equiv$ risk $\equiv$ generalization error

Didactic example: sizing an Energy Community battery

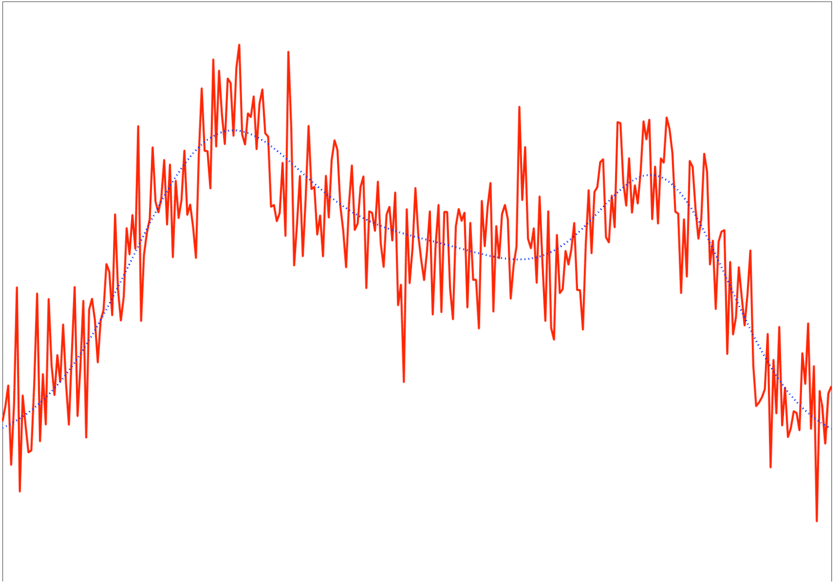
- Energy Community: pool of prosumers with both generation and consumption pooling resources to reduce (eliminate) reliance from the energy grid
- A serious application, don't just take my word for it: `https://ciress.it`, `https://autens.unipi.it`, `https://unescochair.unipi.it`
- Test problem: estimate daily energy production at 5-minutes resolution
- Too few measures of noisy process, large inherent random error, but underlying physical process "constant and smooth"
- Need to estimate the "constant" part / average of the process (yearly or multi-year planning, it's the long-term average that counts)
- Analytic model possible but requires a lot of information on the system ($\Longrightarrow$ has to be re-done if the system changes)
- Synthetic model just need to see the data, let's try that

Estimate an Energy Community Production, the data



- Noisy measurement

Estimate an Energy Community Production, the data



- Noisy measurement but (hopefully) simple/smooth underlying physical process

Polynomial interpolation

- Available set of observations $(x_0 , y_0), \dots , (x_m , y_m)$, $(m = 287)$
- Possibly reasonable (??) assumption: the dependence is polynomial, i.e.,

  $y = p_c(x) = c_0 + \sum_{i=1}^k c_i x^i$

  for fixed k and $k + 1$ real parameters $c = [c_0 , c_+ = [c_1 , \dots , c_n]]$
- This would imply that $y_h = p_c(x_h)$ for all h, which is not really true for any c
- Find the c for which it is less untrue $\equiv$ Linear Least Squares:

$$y = \begin{bmatrix} y_0 \\ \vdots \\ y_m \end{bmatrix} \; , \; X = \begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^k \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \dots & x_m^k \end{bmatrix} \quad , \quad \min_c \mathcal{L}(c) = \| \, y - Xc \, \|^2$$

- Minimize loss function $\| \, y - Xc \, \|^2$ = empirical risk: how much the model fails the predict the phenomenon on the available observations
- $\min\{ \, f(c) = \frac{1}{2} c^T Q c + q c \, \}$ with $Q = X^T X$ and $q = -y^T X$, simple
- In `Matlab` is just `c = y / X`, let's see this in practice

But how about k?

- LSS choses “optimal” c for fixed k, but how to choose k?
- Jointly optimizing over c and k is hard, have to try all (∞-ly many) values
- Even worse: how do you measure how good a given k is?
- If you knew the ground truth you could check, let’s see what would happen

But how about k?

- LSS choses “optimal” c for fixed k, but how to choose k?
- Jointly optimizing over c and k is hard, have to try all (∞-ly many) values
- Even worse: how do you measure how good a given k is?
- If you knew the ground truth you could check, let’s see what would happen
- $\exists\, k \;\equiv$ optimal model complexity: less $\Longrightarrow$ model too coarse to map the phenomenon, more $\Longrightarrow$ learning the noise, not the phenomenon
- Underfitting vs. overfitting, a.k.a. “the bias/variance dilemma”
- But we don’t know the ground truth, we need a proxy
- What we can measure: how good is the model at predicting data it has not seen $\Longrightarrow$ k-fold cross-validation (not the same k)
- Let’s see what this gives:

But how about k?

- LSS choses "optimal" c for fixed k, but how to choose k?
- Jointly optimizing over c and k is hard, have to try all (∞-ly many) values
- Even worse: how do you measure how good a given k is?
- If you knew the ground truth you could check, let's see what would happen
- $\exists\, k \equiv$ optimal model complexity: less $\Longrightarrow$ model too coarse to map the phenomenon, more $\Longrightarrow$ learning the noise, not the phenomenon
- Underfitting vs. overfitting, a.k.a. "the bias/variance dilemma"
- But we don't know the ground truth, we need a proxy
- What we can measure: how good is the model at predicting data it has not seen $\Longrightarrow$ k-fold cross-validation (not the same k)
- Let's see what this gives: roughly the same answer (by chance? or not?)

Wrap-up about synthetic models

- Available, hopefully large set of observations $(x_0 , y_0), \ldots , (x_m , y_m)$: x_h typically $\gg$ a single real (a long vector, a tree, . . .), often y_h too
- Sometimes y_h not even there (unsupervised learning)
- Want to learn from the data $\equiv$ construct a model
- Choose the form among one of the not too many different options, solve the fitting problem $\min_c \mathcal{L}(c)$ to find the optimal parameters
- Choose the hyperparameters to find the best bias/variance compromise (this often requires a grid search $\Longrightarrow$ costly)
- Very many other aspects to be considered: choice of the model (NN, SVR, RBF, DT, Bayesian, clustering, ARMA/ARIMA, . . .), regularization, feature selection, sparsification/pruning, explainability, fairness, . . .

Wrap-up about synthetic models

- Available, hopefully large set of observations $(\, x_0 \,,\, y_0 \,), \ldots, (\, x_m \,,\, y_m \,)$:
  x_h typically $\gg$ a single real (a long vector, a tree, ...), often y_h too

- Sometimes y_h not even there (unsupervised learning)

- Want to learn from the data $\equiv$ construct a model

- Choose the form among one of the not too many different options,
  solve the fitting problem $\min_c \mathcal{L}(\, c \,)$ to find the optimal parameters

- Choose the hyperparameters to find the best bias/variance compromise
  (this often requires a grid search $\Longrightarrow$ costly)

- Very many other aspects to be considered: choice of the model (NN, SVR, RBF, DT, Bayesian, clustering, ARMA/ARIMA, ...), regularization, feature selection, sparsification/pruning, explainability, fairness, ...

  of which we don't give a ↯ here

Wrap-up about synthetic models

- Available, hopefully large set of observations $(x_0 , y_0), \ldots, (x_m , y_m)$: x_h typically $\gg$ a single real (a long vector, a tree, . . .), often y_h too

- Sometimes y_h not even there (unsupervised learning)

- Want to learn from the data $\equiv$ construct a model

- Choose the form among one of the not too many different options, solve the fitting problem $\min_c \mathcal{L}(c)$ to find the optimal parameters

- Choose the hyperparameters to find the best bias/variance compromise (this often requires a grid search $\Longrightarrow$ costly)

- Very many other aspects to be considered: choice of the model (NN, SVR, RBF, DT, Bayesian, clustering, ARMA/ARIMA, . . .), regularization, feature selection, sparsification/pruning, explainability, fairness, . . .

  of which we don't give a ↯ here

  except insomuch as each of these things impact on optimization, which they do

- All that you will see in AI/ML/Data Engineering courses

What are we interested in here

- This course is about a different set of questions, such as:
  1. how do you solve a fitting problem? how are optimal solutions characterised?
  2. if I choose model XYZ, can the fitting problem be solved efficiently?
  3. if I choose model XYZ, which algorithms exist to solve the fitting problem?
  4. how the cost of solving the fitting problem changes when the characteristics of x and y (number, size, sparsity, ...) change?
  5. which algorithm is best to solve the fitting problem for model XYZ in case x and y have characteristics ABC?
- The optimization backbone of data engineering

What are we interested in here

- This course is about a different set of questions, such as:
  1. how do you solve a fitting problem? how are optimal solutions characterised?
  2. if I choose model XYZ, can the fitting problem be solved efficiently?
  3. if I choose model XYZ, which algorithms exist to solve the fitting problem?
  4. how the cost of solving the fitting problem changes when the characteristics of x and y (number, size, sparsity, ...) change?
  5. which algorithm is best to solve the fitting problem for model XYZ in case x and y have characteristics ABC?
- The optimization backbone of data engineering and much else beyond
- Typical of data engineering is to work with a small-ish set of models; here you will understand (a part of) why they have been chosen
- Typical of optimization is to let you build your model (within reason)
- In fact, so far data engineering $\equiv$ descriptive analytics $\equiv$ how things are
- But optimization $\equiv$ prescriptive analytics $\equiv$ how things should be
- (Sometimes) “the last step of data engineering”, let’s see an example

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

Motivation I, reprise III: Clustering

Wrap up I, II, and III, References

Solutions

Example, take II: optimal sizing of an Energy Community battery

- Why estimating the daily energy production at 5-minutes resolution?
  To estimate how large a battery you should buy

- Battery $\equiv$ selling the energy to the market when it's more convenient

- Price changes hourly with significant swings

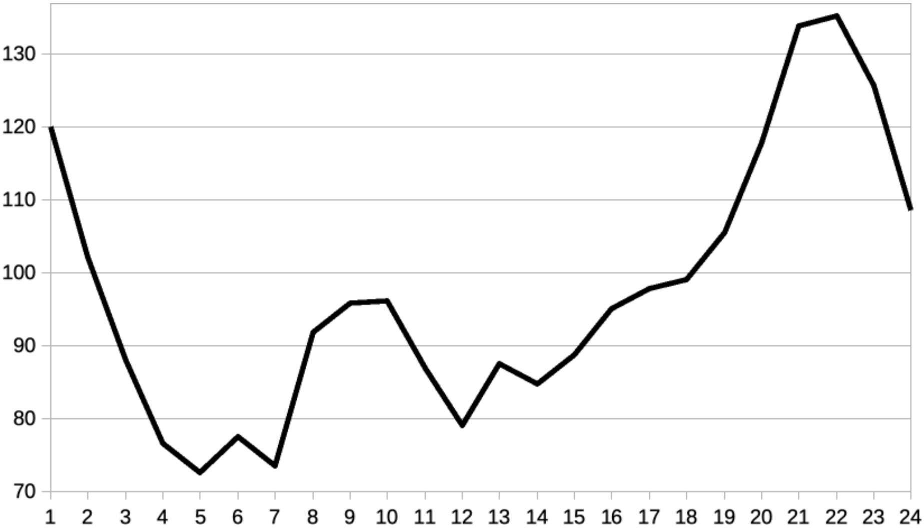
- But battery costly, have to precisely decide how it will be used
  $\Longrightarrow$ have to precisely estimate how much energy will be produced

- Necessary but not sufficient input for optimal scheduling problem
  considering technical constraints (max % discharge at every interval)

- Plus, battery only comes in discrete chunks

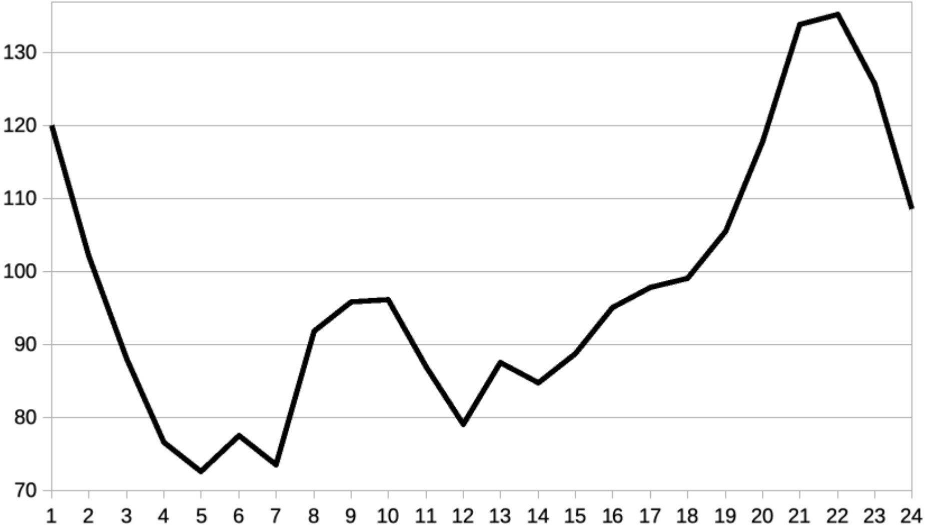
- All in all, optimal sizing requires optimal scheduling

Optimal sizing of an Energy Community battery, the data



- Real Italian € / MWh 08-08-2021

Optimal sizing of an Energy Community battery, the data



- Real Italian € / MWh 08-08-2021 . . . you don't want to see 08-08-2022

Optimal sizing of an Energy Community: analytic model I

- Data for the analytic model:
  1. $T = \{ 0 , 1 , \dots , 287 \}$ set of time periods, $T_{-} = T \setminus \{ 287 \}$
  2. for each $t \in T$, v_t = energy production and p_t = energy price
  3. for each battery module, capacity (60 MHw), ammortised cost (80 € / MWh), max % charge/discharge in one time period (10)
  4. energy loss for charging/discharging the battery (1%)

- Variables of the analytic model:
  1. b_t = amount of energy in the battery at the beginning of period t
  2. s_t / e_t: energy stored in / extracted from the battery within period t
  3. m_t: energy sold to the energy market within period t
  4. z: number of battery modules bought

- Problem is periodic: b_t must be equal at first and last period

Optimal sizing of an Energy Community: analytic model II

$$
\begin{aligned}
\max\ & \textstyle\sum_{t \in T} p_t m_t - 480z && && (1)\\
& 0 \leq b_t \leq 60z && t \in T && (2)\\
& b_t + 0.99 * s_t - e_t = b_{t+1} && t \in T_- && (3)\\
& b_{287} + 0.99 * s_{287} - e_{287} = b_0 && && (4)\\
& m_t = 0.99e_t - s_t + v_t && t \in T && (5)\\
& 0 \leq e_t \leq 6z && t \in T && (6)\\
& 0 \leq s_t \leq 6z && t \in T && (7)\\
& z \in \mathbb{N} && && (8)
\end{aligned}
$$

- May look complicated to the untrained eye, but in fact quite simple
- (2) battery capacity, (6)–(7) charging/discharging limit
- (3)–(4) battery energy flow, (5) defines m_t (may be projected away)
- (8) integrality constraint: bad, but only one variable
- Easy to write and solve in practice, let's see

Optimal sizing of an Energy Community: wrap up

- Synthetic / descriptive model is an input to the analytic / prescriptive model
- Better data leads to better decisions
- Don’t read too much in the example (artificial), but general concept holds

Optimal sizing of an Energy Community: wrap up

- Synthetic / descriptive model is an input to the analytic / prescriptive model
- Better data leads to better decisions
- Don't read too much in the example (artificial), but general concept holds
- Need a prescriptive model be analytic? In principle no
- Could use reinforcement learning to learn the optimal decision rule
- Very good for Go, Atari games, protein folding

Optimal sizing of an Energy Community: wrap up

- Synthetic / descriptive model is an input to the analytic / prescriptive model
- Better data leads to better decisions
- Don't read too much in the example (artificial), but general concept holds
- Need a prescriptive model be analytic? In principle no
- Could use reinforcement learning to learn the optimal decision rule
- Very good for Go, Atari games, protein folding . . . good and needed here?
- Data not there, so a simulator ("digital twin") needed $\Longrightarrow$ no less information on the system than the analytical model needs (likely more)
- Analytical MILP model easy to write and solve with established tools, guarantees optimal solutions, fast enough (in this case)
- Same tools can be used in zillions other cases

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

Motivation I, reprise III: Clustering

Wrap up I, II, and III, References

Solutions

An aside (not really): synthetic vs. analytic

- Are synthetic models better than analytic models?

An aside (not really): synthetic vs. analytic

- Are synthetic models better than analytic models?
  The question is ill-posed: better for what?
- No tool / model is unquestionably best

An aside (not really): synthetic vs. analytic

- Are synthetic models better than analytic models?
  The question is ill-posed: better for what?

- No tool / model is unquestionably best unless perhaps if $\mathcal{P} = \mathcal{NP}$, which nobody believes, and likely not even then

An aside (not really): synthetic vs. analytic

- Are synthetic models better than analytic models?
  The question is ill-posed: better for what?

- No tool / model is unquestionably best unless perhaps if $\mathcal{P} = \mathcal{NP}$, which nobody believes, and likely not even then

- Analytic and synthetic models are just very very different,
  the difference between reptilian brain and neocortex, intuition and proof

- Synthetic models $\equiv$ AI = Artificial Intuition $\approx$ subconscious:
  quickly takes hopefully good decisions, can fail, may not know why
  (which is OK in many cases, in particular if you can't wait longer)

- Analytic models $\equiv$ mathematical optimization $\approx$ logical brain:
  slowly carves its way through an optimal decision, and can prove it
  (which is OK if you really need to be sure and you can afford to wait)

An aside (not really): synthetic vs. analytic

- Are synthetic models better than analytic models?
  The question is ill-posed: better for what?

- No tool / model is unquestionably best unless perhaps if $\mathcal{P} = \mathcal{NP}$, which nobody believes, and likely not even then

- Analytic and synthetic models are just very very different,
  the difference between reptilian brain and neocortex, intuition and proof

- Synthetic models $\equiv$ AI = Artificial Intuition $\approx$ subconscious:
  quickly takes hopefully good decisions, can fail, may not know why
  (which is OK in many cases, in particular if you can't wait longer)

- Analytic models $\equiv$ mathematical optimization $\approx$ logical brain:
  slowly carves its way through an optimal decision, and can prove it
  (which is OK if you really need to be sure and you can afford to wait)

- Golden rule of all science/engineering: no tool is perfect for everything
  1. would you prove a theorem to decide how to run away from an hungry tiger?
  2. would you only rely on intuition to design a nuclear reactor?

Analytic models: advantages and disadvantages

- Analytic models have significant advantages:
  - can handle arbitrarily complex logical conditions
  - the answer is not influenced by prejudices
  - can formally prove the correctness of the answer

Analytic models: advantages and disadvantages

- Analytic models have significant advantages:
  - can handle arbitrarily complex logical conditions
  - the answer is not influenced by prejudices
  - can formally prove the correctness of the answer
- Analytic models have serious issues:
  - can only tackle fully assiomatized problems
  - correctness of the answer hinges on correctness of the model
  - models are very costly to solve (exactly)

Analytic models: advantages and disadvantages

- Analytic models have significant advantages:
    - can handle arbitrarily complex logical conditions
    - the answer is not influenced by prejudices
    - can formally prove the correctness of the answer
- Analytic models have serious issues:
    - can only tackle fully assiomatized problems
    - correctness of the answer hinges on correctness of the model
    - models are very costly to solve (exactly)
- Why analytic models are not everywhere?
    - the model of the world is just too complex to write
    - even if one could write it, solving it would be too slow
- Cannot use analytic models to govern a robot/car,
  can't take decisions quickly enough

Synthetic models: advantages and disadvantages

- Synthetic models have significant advantages
  - can take difficult to assiomatize decisions (is cat or not)
  - the model is automatically learnt, thus (hopefully) have to correctly represent the data
  - can be cheap to solve (with the right methods)

Synthetic models: advantages and disadvantages

- Synthetic models have significant advantages
  - can take difficult to assiomatize decisions (is cat or not)
  - the model is automatically learnt, thus (hopefully) have to correctly represent the data
  - can be cheap to solve (with the right methods)
- Synthetic models have serious issues:
  - require a lot of data to learn
  - can only learn what the data shows (the Black Swan is invisible)
  - (often) cannot explain why it took that decision
  - the answer is influenced by prejudices
  - it is hard to be sure it has learnt complex logical conditions

Synthetic models: advantages and disadvantages

- Synthetic models have significant advantages
  - can take difficult to assiomatize decisions (is cat or not)
  - the model is automatically learnt, thus (hopefully) have to correctly represent the data
  - can be cheap to solve (with the right methods)
- Synthetic models have serious issues:
  - require a lot of data to learn
  - can only learn what the data shows (the Black Swan is invisible)
  - (often) cannot explain why it took that decision
  - the answer is influenced by prejudices
  - it is hard to be sure it has learnt complex logical conditions
- Why synthetic models are not everywhere?
  - not obvious to have enough data ($\exists$ 70+ years, works only since 10)
  - can never 100% trust the decisions

The case for pacific co-existence between analytic and synthetic

- They tried to use analytic models to develop general intelligence

  https://en.wikipedia.org/wiki/Fifth_generation_computer

  but it didn't end up well: https://en.wikipedia.org/wiki/AI_winter
- Obvious in hindsight (which is 20/20): proving theorems is not how we work

The case for pacific co-existence between analytic and synthetic

- They tried to use analytic models to develop general intelligence
  `https://en.wikipedia.org/wiki/Fifth_generation_computer`
  but it didn't end up well: `https://en.wikipedia.org/wiki/AI_winter`
- Obvious in hindsight (which is 20/20): proving theorems is not how we work
- Could it still work? In principle yes, but (likely) $\mathcal{P} \neq \mathcal{NP}$: proving theorems quickly enough will never be possible in general (QC will not help)

The case for pacific co-existence between analytic and synthetic

- They tried to use analytic models to develop general intelligence

  https://en.wikipedia.org/wiki/Fifth_generation_computer

  but it didn't end up well: https://en.wikipedia.org/wiki/AI_winter
- Obvious in hindsight (which is 20/20): proving theorems is not how we work
- Could it still work? In principle yes, but (likely) $\mathcal{P} \neq \mathcal{NP}$: proving theorems quickly enough will never be possible in general (QC will not help)
- They are trying to replace all analytic models with synthetic ones, with only partial success, and anyway always without a guarantee

The case for pacific co-existence between analytic and synthetic

- They tried to use analytic models to develop general intelligence
  https://en.wikipedia.org/wiki/Fifth_generation_computer
  but it didn't end up well: https://en.wikipedia.org/wiki/AI_winter
- Obvious in hindsight (which is 20/20): proving theorems is not how we work
- Could it still work? In principle yes, but (likely) $\mathcal{P} \neq \mathcal{NP}$: proving theorems quickly enough will never be possible in general (QC will not help)
- They are trying to replace all analytic models with synthetic ones, with only partial success, and anyway always without a guarantee
- Could it still work? In principle yes, but I'd wager not since (likely) $\mathcal{P} \neq \mathcal{NP}$ $\Longrightarrow$ solving problems inherently requires exponentially many information, if AI could do that it would be with little information $\Longrightarrow \mathcal{P} = \mathcal{NP}$ (but then also analytic models would be "easy")
- In other words: even if AI reproduces our intelligence, it will not be good at proving theorems, since we are not (and nothing can be)

The case for pacific co-existence between analytic and synthetic

- They tried to use analytic models to develop general intelligence
  `https://en.wikipedia.org/wiki/Fifth_generation_computer`
  but it didn't end up well: `https://en.wikipedia.org/wiki/AI_winter`
- Obvious in hindsight (which is 20/20): proving theorems is not how we work
- Could it still work? In principle yes, but (likely) $\mathcal{P} \neq \mathcal{NP}$: proving theorems quickly enough will never be possible in general (QC will not help)
- They are trying to replace all analytic models with synthetic ones, with only partial success, and anyway always without a guarantee
- Could it still work? In principle yes, but I'd wager not since (likely) $\mathcal{P} \neq \mathcal{NP}$ $\Longrightarrow$ solving problems inherently requires exponentially many information, if AI could do that it would be with little information $\Longrightarrow$ $\mathcal{P} = \mathcal{NP}$ (but then also analytic models would be "easy")
- In other words: even if AI reproduces our intelligence, it will not be good at proving theorems, since we are not (and nothing can be)
- Wrap-up: do not expect AI to save you from theorems $\Longrightarrow$ this course

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

Motivation I, reprise III: Clustering

Wrap up I, II, and III, References

Solutions

(Artificial, Deep) Neural Network

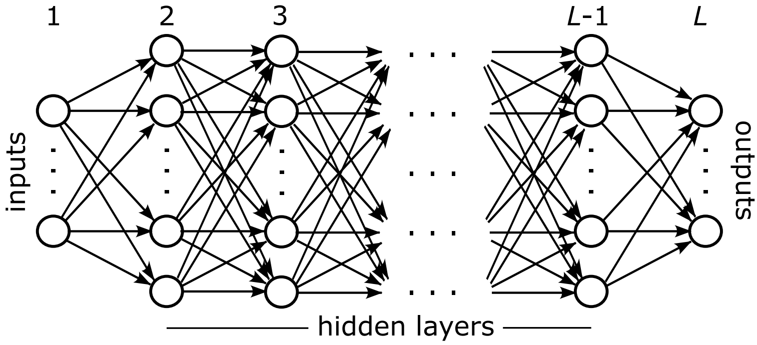
- Multivariate regression: $X = [\, x^i \in \mathbb{R}^h \,,\; y^i \in \mathbb{R}^k \,]_{i \in I}$ input / output samples construct $f : \mathbb{R}^h \to \mathbb{R}^k$ that "best" approximates $f(\, x^i \,) \approx y^i \;\forall\, i \in I$
- Usual caveats apply (bias/variance dilemma, . . .), not our focus here

(Artificial, Deep) Neural Network

- Multivariate regression: $X = [\, x^i \in \mathbb{R}^h \,,\, y^i \in \mathbb{R}^k \,]_{i \in I}$ input / output samples construct $f : \mathbb{R}^h \to \mathbb{R}^k$ that "best" approximates $f(\, x^i \,) \approx y^i \;\forall\, i \in I$

- Usual caveats apply (bias/variance dilemma, ...), not our focus here

- (A)NN = L-layered graph, $L > 2$: $l = 1$ input, $l = L$ output, others hidden complete here only for simplicity (convolutional NN, transformers, ...)



Neural Network notation

- $N(l)$ = nodes (neurons) of layer l, $n(l) = \#N(l)$, $n(1) = h$, $n(L) = k$
- Adjustable arc weights w^l_{qp} for $q \in N(l-1)$, $p \in N(l)$ plus bias w^l_p for $l < L$
- Notation: $W^l = [\, w^m_* \,]_{m<l}$ = all the weights at layers $< l$ (W^L = all them)
- Activation function $\sigma^l_p(\cdot)$ for $p \in N(l)$, typical examples (& derivatives)
  - Identity / linear: $\sigma(\, z \,) = z$, $\sigma'(\, z \,) = 1$
  - Sigmoid: $\sigma(\, z \,) = 1 \,/\, (\, 1 + e^{-z} \,)$, $\sigma'(\, z \,) = \sigma(\, z \,)(\, 1 - \sigma(\, z \,)\,)$
  - Hyperbolic tangent: $\sigma(\, z \,) = (\, e^z - e^{-z} \,) \,/\, (\, e^z + e^{-z} \,)$, $\sigma'(\, z \,) = 1 - \sigma(\, z \,)^2$
  - ReLU: $\sigma(\, z \,) = \max\{\, z \,,\, 0 \,\}$, $\sigma'(\, z \,) = 1$ if $z > 0$, 0 otherwise (!!)
- Recursive definition: $o^1_p(\, x \,) = x_p$, $p \in N(1)$,

  $$o^{l+1}_p(\, W^{l+1} \,;\, i \,) = \sigma^{l+1}_p\big(\, \textstyle\sum_{q \in N(l)} w^l_{qp} o^l_q(\, W^l \,;\, x \,) + w^l_p \,\big) \qquad p \in N(l+1)$$

- Simple enough to compute (given W^L), but by no means a simple function

Exercise: Write the pseudo-code for "Forward propagation", i.e., computing $\sigma^L_p(\, W^L \,;\, x \,) \; \forall \, p \in N(L)$

Neural Network; the fitting problem

- Final prediction function: $o^L(\, W^L\, ;\, x\,) = [\, o^L_p(\, W^L\, ;\, x\,)\,]_{p \in N(L)} : \mathbb{R}^n \to \mathbb{R}^k$,
  $n = n(2)(n(1)+1) + n(3)(n(2)+1) + \ldots + n(L)(n(L-1)+1)$ (large-scale)
- $\mathcal{L}(\, z\,)$ = loss function, typically MSE: $\mathcal{L}(\, z\,) = \frac{1}{2}\|\, z\,\|^2$ = simplest quadratic
- Highly nonlinear, very-large scale, nonconvex fitting problem:

  $$\min \left\{\, f(\, W^L\,) = \sum_{i \in I} \mathcal{L}(\, o^L(\, W^L\, ;\, x^i\,) - y^i\,) \;:\; W^L \in \mathbb{R}^n \,\right\}$$

  and even computing $f(\, W^L\,)$ costly as $\#S$ very large
- Typically regularised: $\min \left\{\, f(\, W^L\,) + \mathcal{R}(\, W^L\,) \;:\; W^L \in \mathbb{R}^n \,\right\}$
  $\mathcal{R}(\, W^L\,) = \lambda\|\, W^L\,\|^2$ (ridge) or $\mathcal{R}(\, W^L\,) = \lambda\|\, W^L\,\|_1$ (Lasso) or others . . .
  $\lambda > 0$ hyperparameter to be fixed somehow (grid search $\equiv$ exponential)
- Would be unfeasibly hard if global optima required
- Good news: 1) global optima not required (in fact, may even be bad),
  2) local optima typically of very good quality anyway (f not "adversarial")
- Of course, the gradient at least is needed

The gradient of a Neural Network

- . . .

The gradient of a Neural Network

- . . . . . .

The gradient of a Neural Network

- ... ...

The gradient of a Neural Network

- . . . . no, it's complicated to do efficiently
- Really, don't try this at home (save for $L = 2$, $L = 3$ tops): requires computational graphs $\Longrightarrow$ backpropagation
- A field of study in itself [1], obviously related to automatic differentiaion
- In fact, rough but functional (small-scale) solution: use any AD tool
- Nontrivial: distinguish variables (w, compute the derivative of) from parameters (X, y, change but not have to be differentiated)
- AD tools allow this, or rough solutions possible (cf. `Matlab` code)
- Enough for small prototypes, for large-scale
  - learn the theory (implementation not difficult for "normal" NN);
  - use any one of the many available established tools (`PyTorch` [6], `TensorFlow` [8], `scikit-learn` [7], `Fido` [5], . . .)
- Plenty of efficient and/or user-friendly ways, not our focus here
- Let's just see our optimization methods in action

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

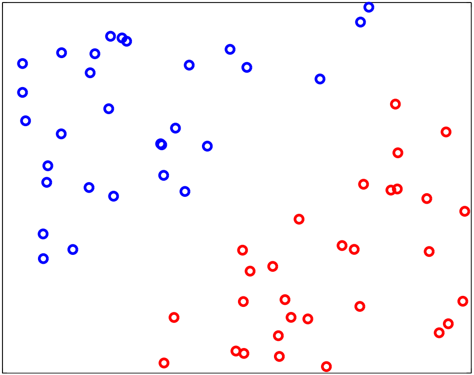
Motivation I, reprise III: Clustering

Wrap up I, II, and III, References

Solutions

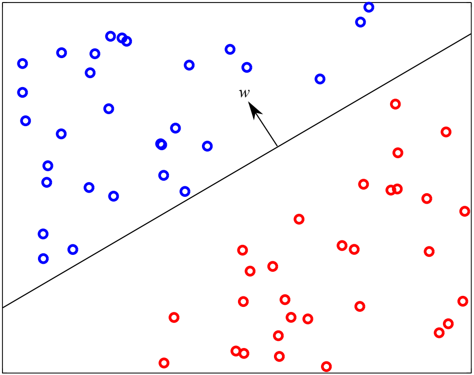
Support Vector Machine / Regression

- $X = [\, X^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, $y = [\, y^i \in \mathbb{R}^1 \,]_{i \in I}$ outputs, "explain" y from X
- Start simple: $y^i \in \{\, 1 \,,\, -1 \,\} \;\equiv$ classification



- $I = \{\, 1, \ldots, m \,\} = I^+ \cup I^-$
- $I^\pm = \{\, i \in I \,:\, y^i = \pm 1 \,\}$

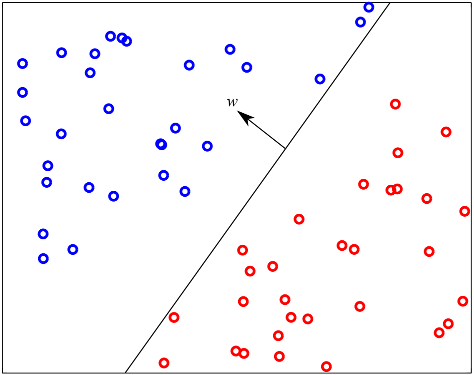
- $X = [\, X^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, $y = [\, y^i \in \mathbb{R}^1 \,]_{i \in I}$ outputs, "explain" y from X
- Start simple: $y^i \in \{\, 1 \,,\, -1 \,\} \;\equiv\;$ classification



- $I = \{\, 1, \ldots, m \,\} = I^+ \cup I^-$
- $I^\pm = \{\, i \in I \;:\; y^i = \pm 1 \,\}$
- (Affine) hyperplane $H(\, w \,,\, b \,) =$
  $= \{\, x \in \mathbb{R}^n \;:\; \langle\, w \,,\, x \,\rangle = b \,\}$
  $w =$ direction , $b =$ "bias"
- H separate I^+ from I^- $\equiv$
  $y^i = \;\; 1 \;\Longrightarrow\; \langle\, w \,,\, x^i \,\rangle > b$
  $y^i = -1 \;\Longrightarrow\; \langle\, w \,,\, x^i \,\rangle < b$

Support Vector Machine / Regression

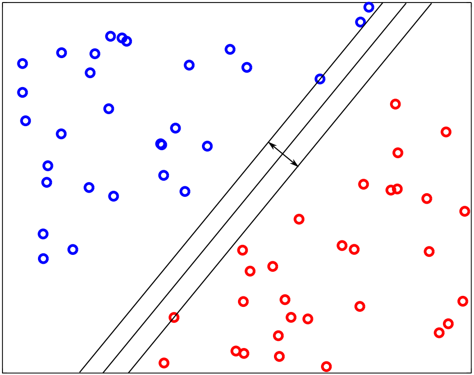
- $X = [\, X^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, $y = [\, y^i \in \mathbb{R}^1 \,]_{i \in I}$ outputs, "explain" y from X
- Start simple: $y^i \in \{\, 1 \,,\, -1 \,\} \;\equiv\;$ classification



- $I = \{\, 1, \ldots, m \,\} = I^+ \cup I^-$
- $I^\pm = \{\, i \in I \,:\, y^i = \pm 1 \,\}$
- (Affine) hyperplane $H(\, w \,,\, b \,) =$
  $= \{\, x \in \mathbb{R}^n \,:\, \langle\, w \,,\, x \,\rangle = b \,\}$
  $w =$ direction , $b =$ "bias"
- H separate I^+ from $I^- \;\equiv$
  $y^i = \;\; 1 \implies \langle\, w \,,\, x^i \,\rangle > b$
  $y^i = -1 \implies \langle\, w \,,\, x^i \,\rangle < b$

- But $\exists$ different H that separate I^+ from I^-, which one do we choose?

- $X = [\, X^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, $y = [\, y^i \in \mathbb{R}^1 \,]_{i \in I}$ outputs, "explain" y from X
- Start simple: $y^i \in \{\, 1 \,,\, -1 \,\} \;\equiv\;$ classification

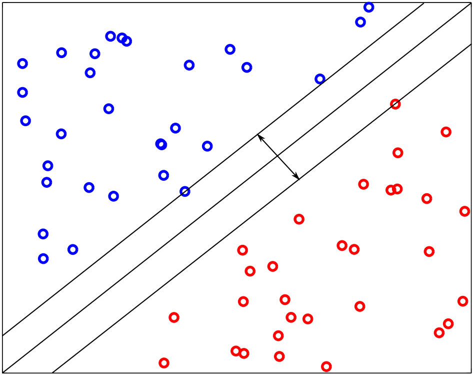


- $I = \{\, 1, \ldots, m \,\} = I^+ \cup I^-$
- $I^\pm = \{\, i \in I \;:\; y^i = \pm 1 \,\}$
- (Affine) hyperplane $H(\, w \,,\, b \,) =$
  $= \{\, x \in \mathbb{R}^n \;:\; \langle\, w \,,\, x \,\rangle = b \,\}$
  $w =$ direction , $b =$ "bias"
- H separate I^+ from $I^- \;\equiv$
  $y^i = \;\;1 \implies \langle\, w \,,\, x^i \,\rangle > b$
  $y^i = -1 \implies \langle\, w \,,\, x^i \,\rangle < b$

- But $\exists$ different H that separate I^+ from I^-, which one do we choose?
- Intuitively, the margin is important (and theory supports the intuition)

Support Vector Machine / Regression

- $X = [\, X^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, $y = [\, y^i \in \mathbb{R}^1 \,]_{i \in I}$ outputs, "explain" y from X
- Start simple: $y^i \in \{\, 1 \,,\, -1 \,\} \;\equiv\;$ classification



- $I = \{\, 1, \ldots, m \,\} = I^+ \cup I^-$
- $I^\pm = \{\, i \in I \,:\, y^i = \pm 1 \,\}$
- (Affine) hyperplane $H(\, w \,,\, b \,) =$
  $= \{\, x \in \mathbb{R}^n \,:\, \langle\, w \,,\, x \,\rangle = b \,\}$
  $w =$ direction , $b =$ "bias"
- H separate I^+ from $I^- \;\equiv$
  $y^i = \;\;\, 1 \implies \langle\, w \,,\, x^i \,\rangle > b$
  $y^i = -1 \implies \langle\, w \,,\, x^i \,\rangle < b$

- But $\exists$ different H that separate I^+ from I^-, which one do we choose?
- Intuitively, the margin is important (and theory supports the intuition)
- Larger margin $\implies$ more "robust" classification

Support Vector Machine / Regression (cont.d)

- $H(\, w \,,\, b \,)$ separate I^+ from I^- $\equiv$ $y^i(\, \langle\, w \,,\, x^i \,\rangle - b \,) \geq 1 \;\; \forall i \in I$ **(check)**
- Distance between $H(\, w \,,\, b \,)$ and $H(\, w \,,\, b' \,) = |\, b - b' \,| \,/\, \|\, w \,\|$ **(check)**

  $\Longrightarrow$ maximum margin separating hyperplane = solution of

  $$\min_{w,b} \{\, \|\, w \,\|^2 \;:\; y^i(\, \langle\, w \,,\, x^i \,\rangle - b \,) \geq 1 \quad i \in I \,\}$$

  (margin $= 2 \,/\, \|\, w \,\|$, "2" because quadratic objective) assuming any $\exists$

Support Vector Machine / Regression (cont.d)

- $H(\, w \,,\, b \,)$ separate I^+ from I^- $\equiv$ $y^i(\, \langle\, w \,,\, x^i \,\rangle - b \,) \geq 1 \;\; \forall i \in I$ **(check)**
- Distance between $H(\, w \,,\, b \,)$ and $H(\, w \,,\, b' \,) = |\, b - b' \,| \,/\, \|\, w \,\|$ **(check)**

  $\Longrightarrow$ maximum margin separating hyperplane = solution of

  $$\min_{w,b}\{\, \|\, w \,\|^2 \;:\; y^i(\, \langle\, w \,,\, x^i \,\rangle - b \,) \geq 1 \quad i \in I \,\}$$

  (margin $= 2 \,/\, \|\, w \,\|$, "2" because quadratic objective) assuming any $\exists$
- What if $\nexists$? Support Vector Machine

  $$\min_{w,b} \;\{\, \|\, w \,\|^2 + C[\, \mathcal{L}(\, w \,,\, b \,) = \sum_{i \in I} \max\{\, 1 - y^i(\, \langle\, w \,,\, x^i \,\rangle - b \,) \,,\, 0 \,\}\,] \,\}$$

  hyperparameter C weighs loss (of separation) against margin = regularization
- $\mathcal{L} \implies$ objective convex but nondifferentiable: $[\, \max\{\, \cdot \,,\, 0 \,\}\,]'(\, 0 \,) = ??$
- Extends to y^i arbitrary $\equiv$ Support Vector Regression

  $$\min_{w,b} \;\{\, \|\, w \,\|^2 + C[\, \mathcal{L}_\varepsilon(\, w \,,\, b \,) = \sum_{i \in I} \max\{\, |\, \langle\, w \,,\, x^i \,\rangle - b - y^i \,| - \varepsilon \,,\, 0 \,\}\,] \,\}$$

  hyperparameter ε controlling the "insensitivity tube", $\mathcal{L}_\varepsilon$ still nondifferentiable
- Linear constraints can be better than a nondifferentiable objective

Support Vector Machine / Regression as constrained problems

- Reformulation of SVM / SVR as a QP via "slack variables" ξ_i

$$\text{(SVM-P)}\quad \min_{w,b,\xi} \left\{ \tfrac{1}{2}\| w \|^2 + C \textstyle\sum_{i\in I} \xi_i \;:\; y^i(wx^i - b) \geq 1 - \xi_i \;,\; \xi_i \geq 0 \;\; i \in I \right\}$$

$$\begin{aligned}\text{(SVR-P)}\quad & \min_{w,b,\xi} \tfrac{1}{2}\| w \|^2 + C \textstyle\sum_{i\in I} \xi_i \\ & wx^i - b - y^i - \varepsilon \leq \xi_i \;,\; -wx^i + b + y^i - \varepsilon \leq \xi_i \;,\; \xi_i \geq 0 \;\; i \in I\end{aligned}$$

- Corresponding quadratic duals (**check**)

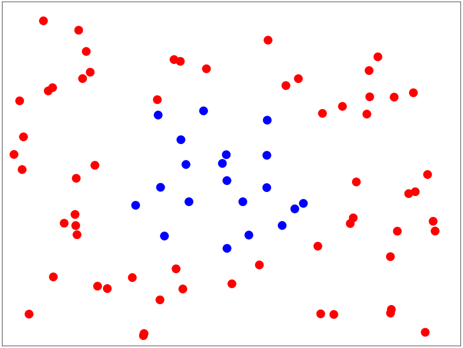
$$\begin{aligned}\text{(SVM-D)}\quad \max_\alpha\; & \textstyle\sum_{i\in I} \alpha_i - \frac{1}{2} \sum_{i\in I} \sum_{j\in I} \alpha_i y^i \langle x^i , x^j \rangle y^j \alpha_j \\ & \textstyle\sum_{i\in I} y^i \alpha_i = 0 \\ & 0 \leq \alpha_i \leq C \qquad\qquad i \in I\end{aligned}$$

$$\begin{aligned}\text{(SVR-D)}\quad \max_\alpha\; & \textstyle\sum_{i\in I} y^i \alpha_i - \varepsilon \sum_{i\in I} |\, \alpha_i \,| - \frac{1}{2} \sum_{i\in I} \sum_{j\in I} \alpha_i \langle x^i , x^j \rangle \alpha_j \\ & \textstyle\sum_{i\in I} \alpha_i = 0 \\ & -C \leq \alpha_i \leq C \qquad\qquad i \in I\end{aligned}$$

- Primal-dual relationships: $w^* = \sum_{i\in I} \alpha_i^* [\, y^i \,] x^i \implies$ classification / regression of new $\bar{x}$ with $\langle\, w^* , \bar{x} \,\rangle - b^* = \sum_{i\in I} \alpha_i^* [\, y^i \,] \langle\, \bar{x} , x^i \,\rangle - b^*$

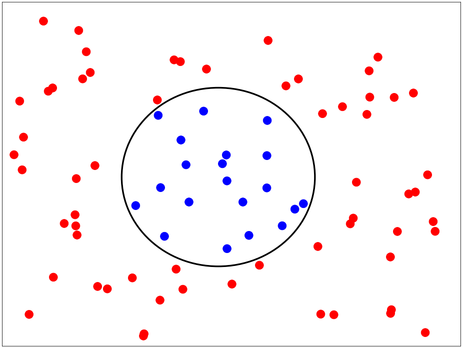
Exercise: prove how to compute w^*, b^* from α^*, discuss why "support vector"

SVM / SVR: the problem, and the solution



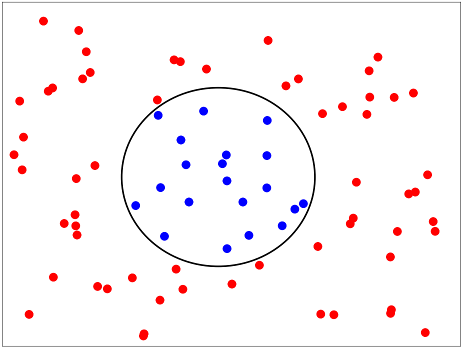
- (Approximate) linear separability

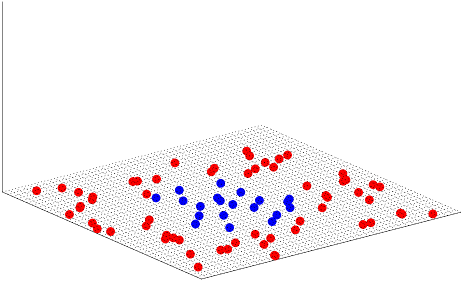
SVM / SVR: the problem, and the solution



- (Approximate) linear separability rare, (approximate) linear regression weak

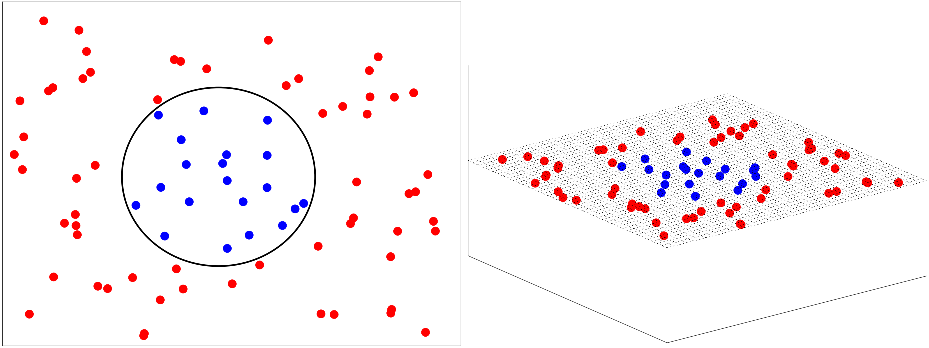
SVM / SVR: the problem, and the solution





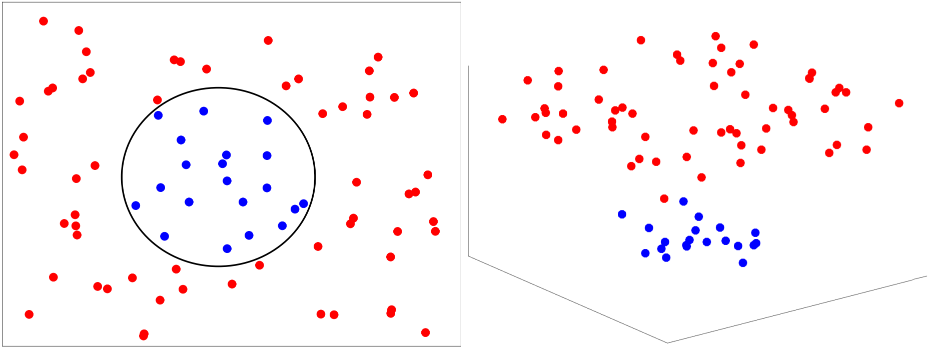
- (Approximate) linear separability rare, (approximate) linear regression weak
- Idea: embed in larger space

SVM / SVR: the problem, and the solution



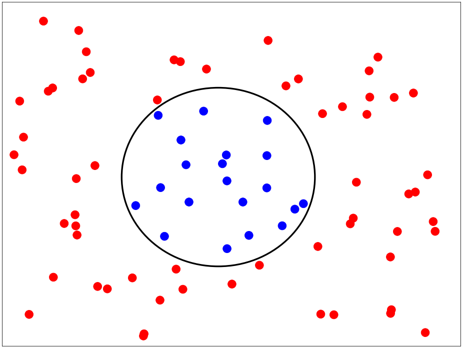
- (Approximate) linear separability rare, (approximate) linear regression weak
- Idea: embed in larger space

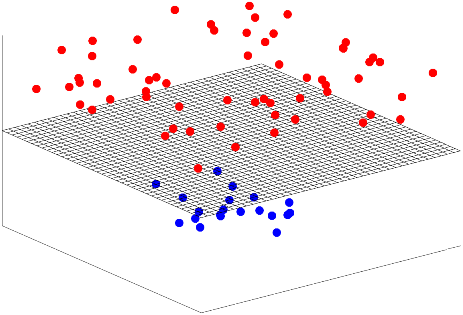
SVM / SVR: the problem, and the solution



- (Approximate) linear separability rare, (approximate) linear regression weak
- Idea: embed in larger space nonlinearly, then

SVM / SVR: the problem, and the solution

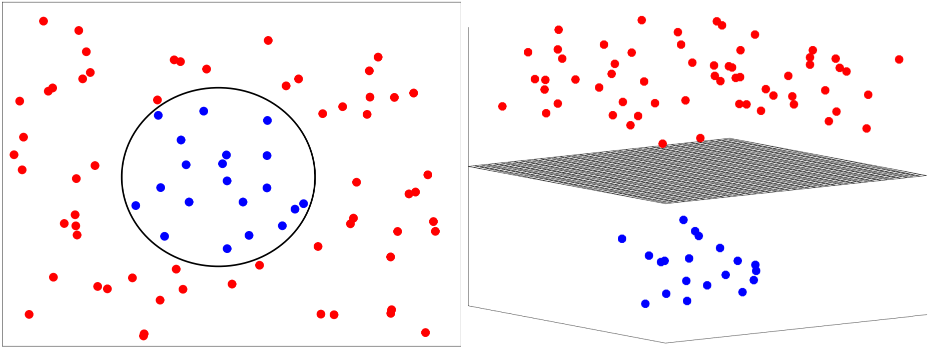




- (Approximate) linear separability rare, (approximate) linear regression weak

- Idea: embed in larger space nonlinearly, then linear function may work

SVM / SVR: the problem, and the solution



- (Approximate) linear separability rare, (approximate) linear regression weak
- Idea: embed in larger space nonlinearly, then linear function may work
- Doing this effectivey (how to embed) and efficiently nontrivial

Embedding SVM / SVR in a larger space

- $\phi : \mathbb{R}^h$ (input space) $\rightarrow \mathcal{F}$ feature space, $x^i \rightarrow \phi(x^i)$
- If $\mathcal{F} = \mathbb{R}^k$ for $k > h$, could just re-do (SVM-P) / (SVR-P) in $\mathbb{R}^k$

Embedding SVM / SVR in a larger space

- $\phi : \mathbb{R}^h$ (input space) $\rightarrow \mathcal{F}$ feature space, $x^i \rightarrow \phi(\, x^i \,)$
- If $\mathcal{F} = \mathbb{R}^k$ for $k > h$, could just re-do (SVM-P) / (SVR-P) in $\mathbb{R}^k$
- $k \gg h$ good (larger $\equiv$ easier to linearly fit / separate) and bad: fitting cost now scales with k rather than h
- Example: $w \rightarrow W = [\, Q \,,\, q \,]$ and $\langle\, w \,,\, x \,\rangle \rightarrow x^T Q x + q x$ ellipsoidal separation (not really, $Q \succeq 0$ not guaranteed)
- Linear??

Embedding SVM / SVR in a larger space

- $\phi : \mathbb{R}^h$ (input space) $\rightarrow \mathcal{F}$ feature space, $x^i \rightarrow \phi(x^i)$
- If $\mathcal{F} = \mathbb{R}^k$ for $k > h$, could just re-do (SVM-P) / (SVR-P) in $\mathbb{R}^k$
- $k \gg h$ good (larger $\equiv$ easier to linearly fit / separate) and bad: fitting cost now scales with k rather than h
- Example: $w \rightarrow W = [\, Q \,,\, q \,]$ and $\langle\, w \,,\, x \,\rangle \rightarrow x^T Q x + q x$ ellipsoidal separation (not really, $Q \succeq 0$ not guaranteed)
- Linear?? Indeed: $x \rightarrow F = [\, x x^T \,,\, x \,]$ and $x^T Q x + q x = \langle\, W \,,\, F \,\rangle$ nonlinearity in mapping ϕ, then linear once in $\mathcal{F}$
- A good thing: nonlinearity on the data (fixed), then problem "easy"

Embedding SVM / SVR in a larger space

- $\phi : \mathbb{R}^h$ (input space) → $\mathcal{F}$ feature space, $x^i \rightarrow \phi(x^i)$
- If $\mathcal{F} = \mathbb{R}^k$ for $k > h$, could just re-do (SVM-P) / (SVR-P) in $\mathbb{R}^k$
- $k \gg h$ good (larger $\equiv$ easier to linearly fit / separate) and bad: fitting cost now scales with k rather than h
- Example: $w \rightarrow W = [\, Q \,,\, q \,]$ and $\langle\, w \,,\, x \,\rangle \rightarrow x^T Qx + qx$ ellipsoidal separation (not really, $Q \succeq 0$ not guaranteed)
- Linear?? Indeed: $x \rightarrow F = [\, xx^T \,,\, x \,]$ and $x^T Qx + qx = \langle\, W \,,\, F \,\rangle$ nonlinearity in mapping ϕ, then linear once in $\mathcal{F}$
- A good thing: nonlinearity on the data (fixed), then problem "easy"
- Issue: $k \in O(h^2)$, cost grows significantly
- Even worse: $\phi(\cdot) \;\equiv\;$ terms of polynomial of degree > 2 (**check**)
- Even worse: one may want $\mathcal{F}$ to be ∞-dimensional

Why the dual is really useful in SVM / SVR: the kernel trick

- (SVM/R-D) require kernel function $\kappa(x^i , x^j) = \langle \phi(x^i) , \phi(x^j) \rangle \; \forall \, i , j$
- Classify / interpolate new $\bar{x}$ requires computing $\langle \phi(w^*) , \phi(\bar{x}) \rangle =$ $= \langle \sum_{i \in I} \alpha^*_i \phi(x^i) , \phi(\bar{x}) \rangle = \sum_{i \in I} \alpha^*_i \langle \phi(x^i) , \phi(\bar{x}) \rangle = \sum_{i \in I} \alpha^*_i \kappa(x^i , \bar{x})$

  whatever ∞-dimensional vector space $\mathcal{F}$ is (general properties of $\langle \cdot , \cdot \rangle$)
  $\Longrightarrow$ can use $\kappa(\cdot, \cdot)$ for everything, no need to ever compute $\phi(\cdot)$
- One κ computation for each support vector x^i s.t. $\alpha^*_i > 0$ (possibly $\ll | \, I \, |$)
- Incredibly clever kernel trick: very large $\mathcal{F}$ s.t. κ is efficient
- κ kernel function for some vector space $\mathcal{F} \iff \int \kappa(x , z) g(x) g(z) dx dz \geq 0$ $\forall \, g(\cdot)$ s.t. $\int g(x)^2 dx$ is finite (Mercier condition), e.g.
  - Polynomial Kernel (PK): $\kappa(x , z) = (\langle x , z \rangle + 1)^k$ (any k)
  - Gaussian Kernel (GK): $\kappa(x , z) = e^{- \| x - z \|^2 \, / \, (2 \sigma^2)}$ (any σ)
  - Sigmoid Kernel (SK): $\kappa(x , z) = \tanh(\sigma \langle x , z \rangle + \delta)$ (some σ, δ and X)
- Many specialised kernels for specific data (trees, graphs, strings, ...), SVR + GK approximates ∞-ly well any f ($\in C^0$, on $[x_- , x_+]$) if $\# X = \infty$

Exercise: discuss why, at least in one dimension, this is not surprising

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

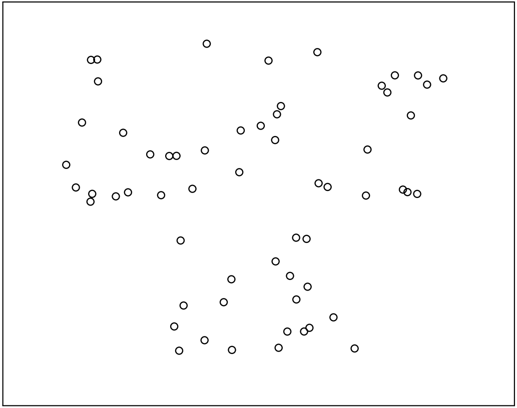
Motivation I, reprise III: Clustering

Wrap up I, II, and III, References

Solutions

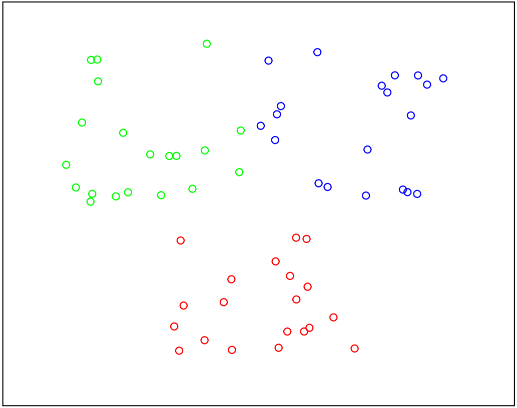
The clustering problem(s)

- $X = [\, x^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, no outputs available $\equiv$ each x^i “looks the same”



The clustering problem(s)

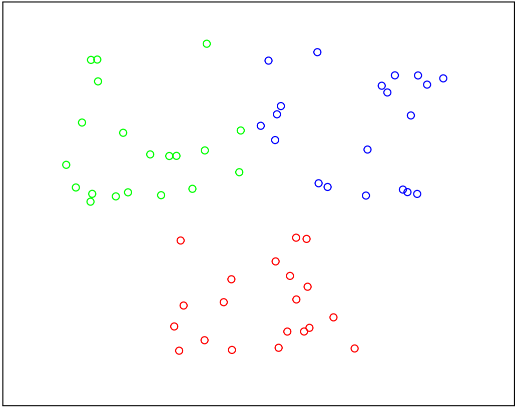
- $X = [x^i \in \mathbb{R}^h]_{i \in I}$ inputs, no outputs available $\equiv$ each x^i "looks the same"



- Given $k \in \mathbb{N}$ ($K = \{1, \ldots, k\}$), find $X = \bigcup_{p \in K} X^p \equiv$ partition of X in clusters s.t. X^i that are homogeneous (??) and well separated (??)

The clustering problem(s)

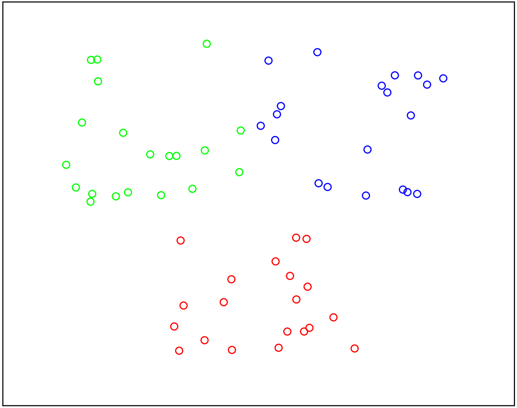
- $X = [\, x^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, no outputs available $\equiv$ each x^i "looks the same"



- Given $k \in \mathbb{N}$ ($K = \{\, 1 \,, \, \ldots \,, \, k \,\}$), find $X = \bigcup_{p \in K} X^p$ $\equiv$ partition of X in clusters s.t. X^i that are homogeneous (??) and well separated (??)
- Crucial problem in unsupervisioned ML: automatically figure out the labels from the data, ill-defined by definition (many $\neq$ ways to label the same stuff)

The clustering problem(s)

- $X = [\, x^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, no outputs available $\equiv$ each x^i “looks the same”

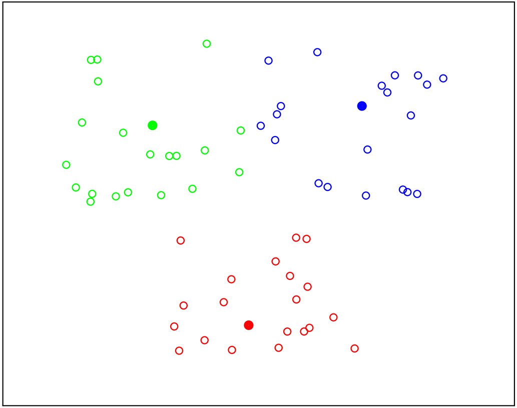


- Many $\neq$ possible variants
- Simplest:

- Given $k \in \mathbb{N}$ ($K = \{\, 1 \,, \ldots \,, k \,\}$), find $X = \bigcup_{p \in K} X^p \equiv$ partition of X in clusters s.t. X^i that are homogeneous (??) and well separated (??)
- Crucial problem in unsupervisioned ML: automatically figure out the labels from the data, ill-defined by definition (many $\neq$ ways to label the same stuff)

The clustering problem(s)

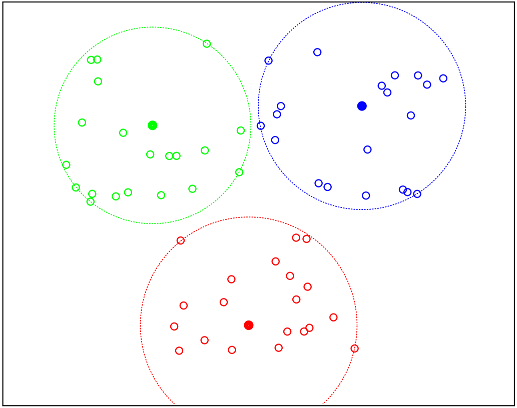
- $X = [\, x^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, no outputs available $\equiv$ each x^i "looks the same"



- Many $\neq$ possible variants
- Simplest: define k centroids c^p $\equiv$ "archetypes" of each $x^i \in X^p$
- Given $k \in \mathbb{N}$ ($K = \{\, 1 \,, \, \ldots \,, \, k \,\}$), find $X = \bigcup_{p \in K} X^p$ $\equiv$ partition of X in clusters s.t. X^i that are homogeneous (??) and well separated (??)
- Crucial problem in unsupervisioned ML: automatically figure out the labels from the data, ill-defined by definition (many $\neq$ ways to label the same stuff)

The clustering problem(s)

- $X = [\, x^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, no outputs available $\equiv$ each x^i "looks the same"



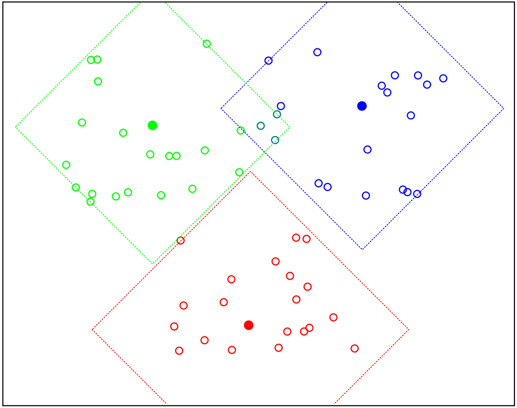
- Many $\neq$ possible variants
- Simplest: define k centroids c^p $\equiv$ "archetypes" of each $x^i \in X^p$
- $X^p = \{\, x^i \;:\;$ closer to c^p than to any other $c^q \,\}$
- Clusters (may) depend on the chosen norm $\equiv$ topology of $\mathbb{R}^h$
- Clusters in L_2

- Given $k \in \mathbb{N}$ ($K = \{\, 1 \,,\, \dots \,,\, k \,\}$), find $X = \bigcup_{p \in K} X^p$ $\equiv$ partition of X in clusters s.t. X^i that are homogeneous (??) and well separated (??)

- Crucial problem in unsupervisioned ML: automatically figure out the labels from the data, ill-defined by definition (many $\neq$ ways to label the same stuff)

The clustering problem(s)

- $X = [\, x^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, no outputs available $\equiv$ each x^i "looks the same"



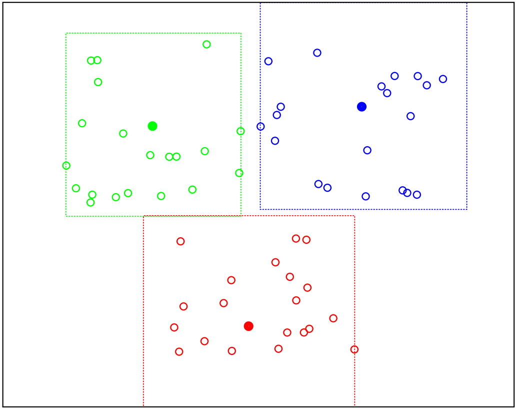
- Many $\neq$ possible variants
- Simplest: define k centroids c^p $\equiv$ "archetypes" of each $x^i \in X^p$
- $X^p = \{\, x^i \,:\,$ closer to c^p than to any other $c^q \,\}$
- Clusters (may) depend on the chosen norm $\equiv$ topology of $\mathbb{R}^h$
- Clusters in $L_2 \neq$ in L_1

- Given $k \in \mathbb{N}$ ($K = \{\, 1 \,,\, \dots \,,\, k \,\}$), find $X = \bigcup_{p \in K} X^p$ $\equiv$ partition of X in clusters s.t. X^i that are homogeneous (??) and well separated (??)

- Crucial problem in unsupervisioned ML: automatically figure out the labels from the data, ill-defined by definition (many $\neq$ ways to label the same stuff)

The clustering problem(s)

- $X = [\, x^i \in \mathbb{R}^h \,]_{i \in I}$ inputs, no outputs available $\equiv$ each x^i "looks the same"



- Many $\neq$ possible variants
- Simplest: define k centroids c^p $\equiv$ "archetypes" of each $x^i \in X^p$
- $X^p = \{\, x^i \,:\,$ closer to c^p than to any other $c^q \,\}$
- Clusters (may) depend on the chosen norm $\equiv$ topology of $\mathbb{R}^h$
- Clusters in $L_2 \neq$ in $L_1 \neq$ in L_∞

- Given $k \in \mathbb{N}$ ($K = \{\, 1 \,,\, \ldots \,,\, k \,\}$), find $X = \bigcup_{p \in K} X^p$ $\equiv$ partition of X in clusters s.t. X^i that are homogeneous (??) and well separated (??)

- Crucial problem in unsupervisioned ML: automatically figure out the labels from the data, ill-defined by definition (many $\neq$ ways to label the same stuff)

Optimization model(s), Euclidean norm

- $c = [\, c^p \,]_{p \in K} \in \mathbb{R}^{hk}$, nonconvex and nonsmooth unconstrained model
  $\min\{\, f(\, c\,) = \sum_{i \in I} \min_{p \in K} \|\, c^p - x^i \,\|_2^2 \,:\, c \in \mathbb{R}^{hk} \}$ (**check**)

- Reformulation I: nonconvex, smooth, combinatorial, constrained model

$$\begin{array}{lll} \min & \sum_{i \in I} \sum_{p \in K} z_{ip} \|\, c^p - x^i \,\|_2^2 & \\ & \sum_{p \in K} z_{ip} = 1 & i \in I \\ & z_{ip} \in \mathbb{N} \;\; [\equiv \;\{\, 0 \,,\, 1 \,\}\,] & p \in K \,,\; i \in I \end{array}$$

  z_{ip} "logical" variables: 1 if x^i "assigned" to cluster p, 0 otherwise

Exercise: prove the equivalence between the two formulations

- Two sources of nonconvexity: products zc in objective, integrality constraints

- But perfect structure for alternating minimization approaches:
  convex in z if c fixed (seen already, cf. exercise), convex in c if z fixed

- z fixed, $I(\, z \,,\, p \,) = \{\, i \in I \;:\; z_{pi} = 1 \,\} \implies (c^p)^* = \sum_{i \in I(\, z \,,\, p\,)} x^i \,/\, \# I(\, z \,,\, p \,)$
  optimal centroid $\equiv$ mean of the points in the cluster

Exercise: prove the previous statement

Immediately $\rightsquigarrow$ the k-means algorithm

```
procedure $c = k\text{-}means\,(\,X\,,\,c\,,\,\varepsilon\,)$   // note: $k$ implicit from size of $c$
 for( $v \leftarrow \infty$  ;  ; ) do
  foreach( $p \in K$ ) do $I(\,p\,) \leftarrow \emptyset$;
  foreach( $i \in I$ ) do $\bar{p} \leftarrow \text{argmin}\{\,\|\,c^p - x^i\,\|_2^2 \,:\, p \in K\,\}$; $I(\,\bar{p}\,) \leftarrow I(\,\bar{p}\,) \cup \{\,i\,\}$;
  foreach( $p \in K$ ) do $c^p \leftarrow \sum_{i \in I(\,p\,)} x^i \,/\, \# I(\,p\,)$;   // note: $I(\,p\,) = \emptyset$ happens
  $\bar{v} \leftarrow \sum_{p \in K} \sum_{i \in I(\,p\,)} \|\,c^p - x^i\,\|_2^2$;
  if( $v - \bar{v} \leq \varepsilon$ )  then break;  else $v \leftarrow \bar{v}$;
```

- Special case of (block) Gauss-Seidel approach: $f(\,x^1\,,\,x^2\,,\,\ldots\,,\,x^k\,)$, iteratively optimize over each individual (group of) variable(s) x^p keeping the other variables fixed $\Longrightarrow$ can work in parallel

- Convenient if f convex over each x^p individually but not jointly on all x

- Converges to stationary point if $f \in C^1$ and $k = 2$, even with constraints, also works for $k > 2$ under more stringent assumptions on f [4]

- In fact, *k-means* finitely terminates even if $\varepsilon = 0$ (**check**)

- Local approach to nonconvex problem $\Longrightarrow$ no guarantee of global optimality $\Longrightarrow$ initial centroids relevant issue in practice (attraction basin)

A (weird-ish) MIQP reformulation

- $c^p \in \mathrm{conv}(\, X\,) \implies \min\{\, x^i \; : \; i \in I\,\} = \underline{x} \leq c^p \leq \overline{x} = \max\{\, x^i \; : \; i \in I\,\}$
- Reformulation II: convex, quadratic, combinatorial, linearly constrained model

$$
\begin{array}{lll}
\min & \sum_{i \in I} \sum_{p \in K} \| \, v_{ip} \, \|_2^2 & \\
& (\overline{x} - x^i) z_{ip} \geq v_{ip} \geq (\underline{x} - x^i) z_{ip} & p \in K \; , \; i \in I \\
& c^p - x^i z_{ip} - \underline{x}(1 - z_{ip}) \geq v_{ip} \geq c^p - x^i z_{ip} - \overline{x}(1 - z_{ip}) & p \in K \; , \; i \in I \\
& \overline{x} \geq c^p \geq \underline{x} & p \in K \\
& \sum_{p \in K} z_{ip} = 1 & i \in I \\
& z_{ip} \in \{\, 0 \, , \, 1 \,\} & p \in K \; , \; i \in I
\end{array}
$$

- Weird tricks of the trade in integer optimization:
  1. $z_{ip} = 0 \implies 0 \geq v_{ip} \geq 0 \;\equiv\; v_{ip} = 0$, and
     $c^p - \underline{x} \geq 0 = v_{ip} \geq c^p - \overline{x}$ since $\overline{x} \geq c^p \geq \underline{x}$
  2. $z_{ip} = 1 \implies c^p - x^i \geq v_{ip} \geq c^p - x^i \;\equiv\; v_{ip} = c^p - x^i$, and
     $\overline{x} - x^i \geq c^p - x^i = v_{ip} \geq \underline{x} - x^i$ since $\overline{x} \geq c^p \geq \underline{x}$
- Very many auxiliary variables but all constraints linear (save integrality) $\implies$ can use an off-the-shelf, general-purpose MIQP solver for global optimality

Yet other (much weirder) reformulations

- General issue: strength of continuous relaxation $z_{ip} \in \mathbb{N} \rightarrow z_{ip} \in \mathbb{R}_+$
- Gives lower bound that drives the search in the integer variables space, but lower bound "weak" $\Longrightarrow$ search inefficient
- Reformulation III: convex, nonsmooth, combinatorial, linearly constrained model

$$\min \ \sum_{i \in I} \sum_{p \in K} \left(\| v_{ip} \|_2^2 \, / \, z_{ip} \right)$$

- Weirder tricks of the trade in (nonlinear) integer optimization: $\| v_{ip} \|_2^2 \, / \, z_{ip} \ \equiv \ \| v_{ip} \|_2^2$ for $z_{ip} \in \{ 0 \, , \, 1 \}$ (since 1. and $0^2 \, / \, 0 = 0$) [2] "Perspective Reformulation" not quadratic but still convex, better bound
- Can still use an off-the-shelf, general-purpose MIQP solver (with tricks [2])
- Reformulation IV: convex, conic, combinatorial, linearly constrained model

$$\begin{array}{ll} \min \ \sum_{i \in I} \sum_{p \in K} t_{ip} & \\ t_{ip} z_{ip} \geq \| v_{ip} \|_2^2 & p \in K \ , \ i \in I \end{array}$$

- $t \geq x^2 \, / \, z \ \equiv \ zt \geq x^2$ (if $z \geq 0$) rotated SOCP constraint [2] $\Longrightarrow$ even more auxiliary variables but "efficient" off-the-shelf solvers for MI-SOCP

Optimization model(s), L_1 norm

- Most arguments smoothly (pun intended) extend if $L_2 \rightarrow L_1$
- Formulation I: nonconvex, nonsmooth, combinatorial, linearly constrained

$$\begin{array}{lll} \min & \sum_{i \in I} \sum_{p \in K} z_{ip} \| \, c^p - x^i \, \|_1 & \\ & \sum_{p \in K} z_{ip} = 1 & i \in I \\ & z_{ip} \in \{ \, 0 \, , \, 1 \, \} & p \in K \; , \; i \in I \end{array}$$

Exercise: make the above formulation smooth (and larger, if needed)

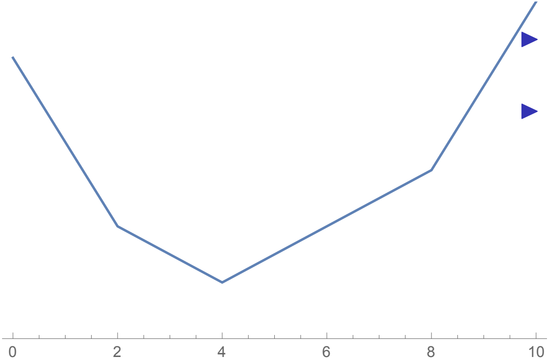
- Still convex in z if c fixed (exactly as before), still convex in c if z fixed
- Even better: c^* produced by easy closed formula as before (but a $\neq$ one)
- Computation decomposes along $p \in K$ and $j = 1, \ldots, h$ (**check**)
- $I(\, z \, , \, p \,) = \{ \, i \in I \; : \; z_{pi} = 1 \, \} \implies (c^p_j)^* =$ median$(\, \{ \, x^i_j \; : \; i \in I(\, z \, , \, p \,) \, \} \,)$
  optimal centroid $\equiv$ median of the points in the cluster [9]
- Proving it requires a bit more work as $f(\, c \,) = \sum_{i \in I(\, z \,)} | \, c - x^i \, |$ nonsmooth

What about the median now?

- W.l.o.g. $I(z) = \{1, \ldots, t\}$ with $x^1 \leq x^2 \leq \ldots \leq x^t$ (nondecreasing order)
- $f(c)$ is piecewise linear, convex, with t breakpoints

What about the median now?

- W.l.o.g. $I(z) = \{1, \ldots, t\}$ with $x^1 \leq x^2 \leq \ldots \leq x^t$ (nondecreasing order)
- $f(c)$ is piecewise linear, convex, with t breakpoints

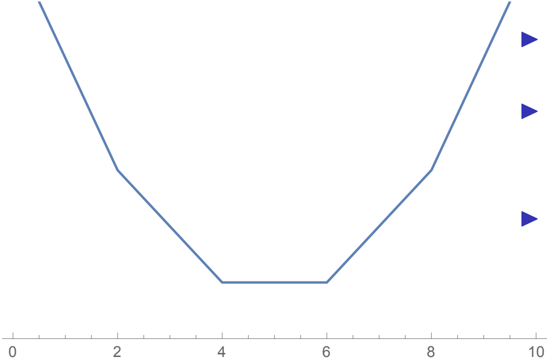


$f(c) = |c - 2| + |c - 4| + |c - 8|$

- Minimum depends on parity of t
- t odd: minimum is $x^{(t+1)/2}$
  $\equiv$ the breakpoint in the middle

What about the median now?

- W.l.o.g. $I(z) = \{1, \dots, t\}$ with $x^1 \leq x^2 \leq \dots \leq x^t$ (nondecreasing order)
- $f(c)$ is piecewise linear, convex, with t breakpoints



$f(c) = |c - 2| + |c - 4| + |c - 6| + |c - 8|$

- Minimum depends on parity of t
- t odd: minimum is $x^{(t+1)/2}$
  $\equiv$ the breakpoint in the middle
- t even: minimum is any point between $x^{t/2}$ and $x^{t/2+1}$
  $\equiv$ the middle breakpoints
  e.g. $(x^{t/2} + x^{t/2+1})/2$

Exercise: prove the previous statements

- Definition of median: the point dividing the population in half

Immediately $\rightsquigarrow$ the k-median algorithm

```
procedure $c = k\text{-}median\,(\, X \,,\, c \,,\, \varepsilon \,)$   // note: $k$ implicit from size of $c$
 for( $v \leftarrow \infty$ ; ; ) do
   foreach( $p \in K$ ) do $I(\, p\,) \leftarrow \emptyset$;
   foreach( $i \in I$ ) do $\bar{p} \leftarrow \text{argmin}\{\, \| \, c^p - x^i \, \|_1 \, : \, p \in K \,\}$; $I(\, \bar{p}\,) \leftarrow I(\, \bar{p}\,) \cup \{\, i \,\}$;
   foreach( $p \in K$ s.t. $I(\, p\,) \neq \emptyset$ ) do
     for( $j = 1$ ; $j \leq h$ ; $++j$ ) do $c^p_j \leftarrow \text{median}(\, \{\, x^i_j \, : \, i \in I(\, p\,) \,\} \,)$;
   $\bar{v} \leftarrow \sum_{p \in K} \sum_{i \in I(\, p\,)} \| \, c^p - x^i \, \|_1$;
   if( $v - \bar{v} \leq \varepsilon$ ) then break; else $v \leftarrow \bar{v}$;
```

- Again (block) Gauss-Seidel approach $\equiv$ local approach to nonconvex problem $\Longrightarrow$ no guarantee of global optimality $\Longrightarrow$ initial centroids

- Finitely converges ($\rightarrow$ stationary point less obvious to prove since $f \notin C^1$)

- Efficient parallel implementations possible, different clusters than L_2

- Similar weird-ish Reformulation II for exact solution

Exercise: develop Mixed-Integer Linear formulations for the problem

Clustering problem – application to Energy Community

- Clustering in epigraphical space $[\, x \,,\, f(\, x \,) \,]$:
  group close values of x that have $\approx$ the same $f(\, x \,)$

- Cluster $\equiv [\, \bar{x} \,,\, \bar{v} \,]$, all x close to $\bar{x}$ get $f(\, x \,) = \bar{v}$
  $\Longrightarrow$ a piecewise-constant approximation of f

- Not what clustering was developed for, don't take the results too seriously

- Is the exact clustering any better than k-means/medians?

- Let's just see how it goes

- Clustering in epigraphical space $[\, x\, ,\ f(\, x\,)\,]$:
  group close values of x that have $\approx$ the same $f(\, x\,)$
- Cluster $\equiv [\, \bar{x}\, ,\ \bar{v}\,]$, all x close to $\bar{x}$ get $f(\, x\,) = \bar{v}$
  $\Longrightarrow$ a piecewise-constant approximation of f
- Not what clustering was developed for, don't take the results too seriously
- Is the exact clustering any better than k-means/medians?
- Let's just see how it goes
- Not that great, but what did you expect? Not what clustering is for

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

Motivation I, reprise III: Clustering

Wrap up I, II, and III, References

Solutions

Wrap up I: what was this course about

- Mathematical optimization and its two (main) roles in Data Engineering:
  1. as a tool for developing mathematical models and solving fitting problems
  2. as prescriptive analytics, “the last step of the decision process”
- A strictly need-to-know review of the underlying mathematical concepts (calculus, linear algebra, numerical analysis, . . .)
- Some hands-on experience with the practicalities and pitfalls of implementations
- Focus on easy problems (linear, quadratic, conic, convex) or local optima, since Data Engineering problems are hard because large, not hard because hard, and besides the global optimal solution is not (necessarily) what you want
- Understanding if/why a problem is easy, a bit about what to do if it it not
- Data Engineering the main source of models/problems, not our real focus: Data Engineering is $\neq$ and $\gg$ than this, but you'll see plenty of that elsewhere

Wrap up II – what we learnt, technical side

- Models are important for algorithms, too (besides vice-versa)
- Models must be simple, but first- and second-order ones are!
- Want a better direction? Use a better model!
  If the world does not give you one, invent one yourself!
- Thank goodness you can go (much) faster than gradient,
  but there is only so much you can do with first-order methods
- Always keep it convex if possible, better if C^1, better still if C^2
- Duality an extremely useful tool, especially (but not only) in convex case
- Mind trade-offs: “fat” models $\rightsquigarrow$ fast convergence but high iteration cost
- If you don’t know it estimate it, but be ready to revise your estimate
- Best choices in theory not best in practice (worst-case $\neq$ average case)
- A lot of details need be considered, numerical aspects crucial

Wrap up III – what we learnt, “philosophical” side

- Dabble with math-based algorithms? Have to know (some) maths
- Learn simple things first: must know a Line Search to optimize in $\mathbb{R}^n$
- Algorithms can only get so far with nasty problems

  hence choose your problems (foes) wisely; AU/ML/DE most often does
- Always exploit all the structure of your problem
- There is no one-size-fits-all solution
- Your mileage may vary, so try, try, try!

Wrap up III – what we learnt, “philosophical” side

- Dabble with math-based algorithms? Have to know (some) maths
- Learn simple things first: must know a Line Search to optimize in $\mathbb{R}^n$
- Algorithms can only get so far with nasty problems

  hence choose your problems (foes) wisely; AU/ML/DE most often does
- Always exploit all the structure of your problem
- There is no one-size-fits-all solution
- Your mileage may vary, so try, try, try!

Lots of Fun!

[1] M. Collins *Computational Graphs, and Backpropagation*
Course notes for NLP, Columbia University
`https://www.cs.columbia.edu/~mcollins/ff2.pdf`

[2] A. Frangioni, C. Gentile “Perspective Cuts for a Class of Convex 0-1 Mixed Integer Programs” *Mathematical Programming* 106, 225—236, 2006

[3] A. Frangioni, C. Gentile “A Computational Comparison of Reformulations of the Perspective Relaxation: SOCP vs. Cutting Planes” *Operations Research Letters* 37, 206—210, 2009

[4] L. Grippo, M. Sciandrone “On the Convergence of the Block Nonlinear Gauss-Seidel Method Under Convex Constraints” *Operations Research Letters* 26, 127—136, 2000

[5] The Fido Project: `https://fidoproject.github.io`

[6] PyTorch: `https://pytorch.org`

[7] scikit-learn: `https://scikit-learn.org`

References II

[8] TensorFlow: https://www.tensorflow.org

[9] Wikipedia – Median https://en.wikipedia.org/wiki/Median

[10] Wikipedia – Universal Approximation Theorem https://en.wikipedia.org/wiki/Universal_approximation_theorem

Outline

Motivation I: Optimization Methods & Game Theory for AI/ML/DE

Motivation II: Data Engineering Techniques for Prescriptive Analytics

Synthetic vs. analytic models

Motivation I, reprise I: Neural Networks

Motivation I, reprise II: Support Vector Machines

Motivation I, reprise III: Clustering

Wrap up I, II, and III, References

Solutions

-
  ```
  procedure $\bar{o}^L = FP\,(\,w\,,\,x\,)$
    $\bar{o}^1 \leftarrow x$;
    for( $l = 1$ to $L - 1$ ) do
      foreach( $p \in N(l+1)$ ) do
        $\bar{o}^{l+1}_p = \sigma^{l+1}_p\big(\,\sum_{q \in N(l)} w^l_{qp}\bar{o}^l_q + w^l_p\,\big)$
  ```

  [back]

- $H(\,w\,,\,b\,)$ separates I^+ from $I^- \iff b^+ = \min\{\,\langle\,w\,,\,x^i\,\rangle\ :\ i \in I^+\,\} > b$ and $b^- = \max\{\,\langle\,w\,,\,x^i\,\rangle\ :\ i \in I^-\,\} < b$. Since $b^+ > b > b^- \implies b^+ > b^* = (b^+ + b^-)\,/\,2 > b^-$, also $H(\,w\,,\,b^*\,)$ separates I^+ from I^-. Clearly $H(\,w\,,\,b^*\,)$ is the hyperplane "in the middle" of the two "extreme" (a.k.a., "supporting") hyperplanes $H(\,w\,,\,b^+\,)$ and $H(\,w\,,\,b^-\,)$ that "touch" at least one x^i respectively with $y^i = 1$ and $y^i = -1$; in fact, $b^+ = b^* + \delta$ and $b^- = b^* - \delta$ for $\delta = (\,b^+ - b^-\,)\,/\,2 > 0$

  Thus, $I^+ = \{\,i \in I\ :\ \langle\,w\,,\,x^i\,\rangle \geq b^+ = b^* + \delta\,\}$: by scaling we can transform $(\,w\,,\,b^*\,)$ into $(\,\omega\,,\,\beta\,)$ s.t. $I^+ = \{\,i \in I\ :\ \langle\,\omega\,,\,x^i\,\rangle - \beta \geq 1\,\}$. Indeed, let $\gamma = 1\,/\,\delta > 0$, $\omega = \gamma w$ and $\beta = \gamma b^*$: $H(\,\omega\,,\,\beta\,) = H(\,w\,,\,b^*\,)$, thus it still separates I^+ from I^-. Since $\langle\,x^i\,,\,w\,\rangle \geq b^+\ \forall\,i \in I^+$ we have $\langle\,x^i\,,\,w\,\rangle - b^* \geq b^+ - b^* = \delta$; dividing by δ yields $\langle\,x^i\,,\,\omega\,\rangle - \beta \geq 1$.

Solutions II

Symmetrically, $\langle x^i , \omega \rangle - \beta \leq -1 \; \forall \, i \in I^-$; the two separate conditions can then be concisely written as $y^i(\langle \omega , x^i \rangle - \beta) \geq 1 \; \forall i \in I$ **[back]**

- For $H = H(w , b)$ and $H' = H(w , b')$, $d(H , H') =$ $= \min_{x,z}\{ \| x - z \| \, : \, x \in H , z \in H' \}$. H and H' are parallel, and it is geometrically obvious (cf. level sets of an affine function) that we can arbitrarily fix z s.t. $\langle w , z \rangle = b'$ and solve $\min_x\{ \| x - z \| \, : \, \langle w , x \rangle = b \}$. We will only see later on the optimality conditions for such a constrained problem which would allow to algebraically derive the solution; here we again rely on the geometric intuition that the solution must clearly be the point $x \in H$ on the half-line emanating from z along the direction $w \perp H'$, i.e., $x = z + \alpha w$ for some $\alpha > 0$. Left-multiplying by w^T gives $\langle w , x \rangle = \langle w , z \rangle + \alpha \langle w , w \rangle \implies b = b' + \alpha \| w \|^2 \equiv$ $\alpha = (b - b') / \| w \|^2$. Thus, $d(H , H') = \| x - z \| = \| \alpha w \| =$ $= [\, | b - b' | / \| w \|^2] \| w \| = | b - b' | / \| w \|^2$ **[back]**

- (SVM-P) is $\min\{\, Cu\xi + \|\, w\,\|^2 \,/\, 2 \,:\, \xi + YXw - yb \geq u \,,\, \xi \geq 0 \,\}$, where X is the matrix having the x^i as rows, y is the vector having the y^i as entries, $Y = \operatorname{diag}(\, y\,)$ and u is the all-1 vector. This is again a QP with "some linear-only variables" and whose "quadratic variables" have strictly convex $Q\,[= I]$: plugging the data of (SVM-P) in the right quadratic dual formula yields $\max\{\, \alpha u - \|\, v\,\|^2 \,/\, 2 \,:\, \alpha + s = Cu \,,\, \alpha y = 0 \,,\, \alpha YX - v = 0 \,,\, \alpha \geq 0 \,,\, s \geq 0 \,\}$ The dual variables s of the $\xi \geq 0$ constraints have no cost, i.e., they are slack variables and can be eliminated by changing the first constraints to $\alpha \leq Cu$. This yields the desired $\max\{\, \alpha u - \alpha^T Y(X^T X) Y\alpha \,/\, 2 \,:\, \alpha y = 0 \,,\, 0 \leq \alpha \leq Cu \,\}$ by just substituting away v. With the same notation, (SVR-P) is $\min\{\, Cu\xi + \|\, w\,\|^2 \,/\, 2 \,:\, \xi - Xw + bu \geq -y - \varepsilon u \,,\, \xi + Xw - bu \geq y - \varepsilon u \,,\, \xi \geq 0 \,\}$ and therefore its dual is
$$\begin{aligned} \max\ & \alpha^+(-y - \varepsilon u) + \alpha^-(y - \varepsilon u) - \|\, v\,\|^2 \,/\, 2 \\ & \alpha^+ + \alpha^- + s = Cu \\ & -\alpha^+ X + \alpha^- X - v = 0 \\ & \alpha^+ u - \alpha^- u = 0 \\ & \alpha^- \geq 0 \,,\, \alpha^+ \geq 0 \,,\, s \geq 0 \end{aligned}$$
with α^+ the multipliers of the first set of constraints, α^- those of the second, and s those of $\xi \geq 0$. Again, the slack variable s can be eliminated by making

the constraint a $\leq$ one. Then, the problem can be written in term of $\alpha = \alpha^+ - \alpha^-$ and $|\,\alpha\,| = \alpha^+ + \alpha^-$ (the latter being the element-wise absolute value vector), since in each optimal solution at least one between α_i^+ and α_i^- is 0 for each i. Indeed, if one had, say, $\alpha_i^+ > \alpha_i^- > 0$, doing $\alpha_i^+ \leftarrow \alpha_i^+ - \alpha_i^-$ and $\alpha_i^- \leftarrow 0$ the value of all terms $\alpha_i^+ - \alpha_i^-$ does not change while the value of all terms $\alpha_i^+ + \alpha_i^-$ decreases, hence the new solution is feasible (if the original one was) and it has a smaller objective value; thus the original solution could not be optimal. All in all, the dual can be rewritten $\max\{\, \alpha y - \varepsilon u|\,\theta\,| - \alpha^T(X^TX)\alpha\,/\,2 \,:\, -Cu \leq \alpha \leq Cu\,,\, \alpha u = 0\,\}$, which is not a QP but can easily be transformed into one by rewriting $\min\{\,|\,x\,|\,\}$ as $\min\{\, v \,:\, v \geq x\,,\, v \geq -x\,\}$. This requires one new variable for each α_i, and therefore yields a QP with as many variables as the original form (sans the s): however, in this form "half of the variables do not appear in the quadratic term", which is in general convenient [**back**]

- (KKT-G) on the variables w gives $[\,\nabla\|\cdot\|^2\,/\,2\,](\,w^*\,) - \alpha^* YW = 0$ for SVM and $[\,\nabla\|\cdot\|^2\,/\,2\,](\,w^*\,) - (\,\alpha^+ - \alpha^-\,)W = 0$ for SVR (recall that constraints need be written as $\leq$, which explains the change in sign to YW and W). Computing b^* requires using (KKT-CS): for any i s.t. $\alpha^*_i = 0$ the corresponding constraint $y^i(wx^i - b) \geq 1 - \xi_i$ in (SVM-P) must be satisfied as equality, and if also $\alpha^*_i < C$ then $\xi^*_i = 0$ (recall that s_i is the dual variable of $\alpha_i \leq C$), which gives $y^i(w^*x^i - b^*) = 1$ that allows to compute b^* once w^* is obtained out of α^* (if $0 < \alpha^*_i < C$ happens for multiple i it may be a good idea numerically to compute b^* multiple times and take the average). Alternatively, if the solver provides (as it should) dual variables, b is just the dual variable of the $\alpha y = 0$ constraint. Similarly for (SVR-P), $w^*x^i - b^* - y_i - \varepsilon = 0$ whenever $C > \alpha^*_i > 0$ ($\equiv C > \alpha^{+,*}_i > 0$ and $\alpha^{-,*}_i = 0$) and $-w^*x^i + b^* + y_i - \varepsilon = 0$ whenever $-C < \alpha^*_i < 0$ ($\equiv C > \alpha^{-,*}_i > 0$ and $\alpha^{+,*}_i = 0$); or fetch the dual variable of the $\alpha u = 0$ constraint from the solver. This discussion justifies the "support vector" moniker. Starting from SVR, the points x^i s.t. $0 < \alpha^*_i < C$ are those that are correctly classified ($\xi^*_i = 0$) and that "lie on the boundary" of the two parallel classifying hyperplanes, i.e., $w^*x^i - b^* = 1$ or $w^*x^i - b^* = -1$. These are called "supporting vectors" of the hyperplane, and surely at least one exists (at least one point of one class will be correctly classified, and there is no point in having them all strictly in the interior of the classification zone). Eliminating all other points would not change the optimal dual solution α^*, and therefore nor w^* and b^*. Thus, like in the Proximal Bundle case, the dual optimal solution provides information about which points are "important" for the current classification (depending on the current choice of C). A similar description holds for SVR: the support hyperplanes are those on the border of the "insensitivity zone" $[\, y^i - \varepsilon \, , \, y^i + \varepsilon \,]$, picture two lines parallel to the graph of the function to be interpolated, one lifted above by ε and one below by the same amount. Since they are "on the border" they are correctly interpolated ($\xi_i^* = 0$), and are the ones which characterise the predicted function in the sense that even if all the other ones are removed from the fitting problem, the function remains the same [**back**]

- Assume we want to classify / interpolate using a cubic polynomial: we can map $x = [\, x_i \,]_{i=1,\dots,h}$ onto the vector having all possible $h(h-1)(h-2) \,/\, 6$ triples $x_p x_q x_m$ plus all possible $h(h-1) \,/\, 2$ pairs $x_p x_q$ plus all individual entries x_p; thus, the corresponding w would have $O(\, h^3 \,)$ entries. In general, a polynomial of degree k would entail $O(\, h^k \,)$ entries [**back**]

- Each term $\kappa(\, x \,,\, x^i \,) = e^{-\|\, x - x^i \,\|^2 \,/\, (2\sigma^2)}$ is 1 for $x = x^i$, but it will vanish quickly (the more so the more σ is small) as x drifts away from x^i. Thus, any function $f(\, x\,)$ could in principle be replicated with arbitrarily high accuracy by, roughly speaking, having "uncountably ∞-ly many" terms $\kappa(\, x \,,\, x^i \,)$ in the sum, one for each $x^i \in \mathbb{R}$, with $\alpha^*_i = f(\, x^i \,) + b^*$, and an "infinitely small σ". Note that the constraint $\sum_{i \in I} \alpha^*_i = 0$ is satisfied by taking $b^* = -\int f(\, x\,)dx$, provided of course that the integral is finite, which is guaranteed to hold if $f \in C^0$ and restricted to a finite interval $[\, x_- \,,\, x_+ \,]$. Thus, over any such finite interval, an appropriately large (but finite) number of support "vectors" $x^i \in \mathbb{R}$ and an appropriately small (but finite) σ should reasonably be able to reproduce any continuous function f. Of course this comes at the cost of a "very large data set" and it is very likely to result in "overfitting", i.e., it is not a good solution in terms of the bias/variance dilemma. Furthermore, this does not imply that SVR with Gaussian kernel is a universal approximator in the strong sense envisioned by ML, unlike, e.g., Neural Networks [10] [**back**]

- Consider the trivial example where $h = 1$, $X = \{ 0 \}$, $k = 2$ and the Euclidean norm: $f(c_1 , c_2) = \min\{ c_1^2 , c_2^2 \}$. This function is clearly nonconvex and nonsmooth. To see this, picture the slice where, say, $c_2 = 1$, i.e., $f(c_1) = \min\{ c_1^2 , 1 \}$. The function in nonsmooth in $c_1 = 1$ as the left derivative is 1/2 while the right derivative is 0; by the same token it is not convex as the derivative is not increasing [**back**]

- Let's start with a general result: given $d \in \mathbb{R}^k$, $\min\{ d_p : p \in K\} =$ $= \min\{ \sum_{p \in K} d_p z_p : \sum_{p \in K} z_p = 1 , z_p \in \mathbb{N} \;\; p \in K \}$. That is, (for each i) the (independent) "combinatorial" problem of assigning the best value to the z_p corresponds to finding the index of the minimum element in the array d_p. Now, just take $d_p = [\| c^p - x \|_2^2]_{k \in K}$. In other words: the second formulation differs from the first because, once the c are fixed, it can arbitrarily assign each x^i to any cluster $p \in K$. But since it has to minimize the objective, once the c are fixed the z have to be chosen as the solution that (for each i independently) selects the minimum value. Thus, if c^* is the optimal solution of the first formulation, then (c^* , z^*) where z^* is chosen in that way is feasible in the second formulation and it has the same objective value (since all terms but the minimum one "disappear"). Obviously, the second formulation cannot have a solution (c^* , z^*) with a better objective than the optimal value of the first since c^* is feasible there and its objective value is not larger than that of (c^* , z^*) in the second formulation [**back**]

- Obviously, for fixed z the minimization problem decomposes along the clusters: each variable c^p can be optimized independently, hence we can drop the index p. The next crucial observation is that the problem also decomposes along the components of the vector: since it is $\min\{ \sum_{j=1}^{h} (c_j - x^i_j)^2 \}$, a component c_j has no "links" with any other component $c_{j'}$ for $j' \neq j$, and therefore each of the problems can be solved independently. Hence, we can also drop the index j and consider c and each x^i as simple real scalars. Solving the (convex) problem then just amounts at finding the (unique) root of the derivative, i.e., $\sum_{i \in I(z)} (2c - 2x^i) = 0 \; \equiv \; (2\#I(z))c - 2 \sum_{i \in I(z)} x^i = 0$, which immediately yields the announced result [**back**]

- This depends on the fundamental combinatorial nature of the underlying clustering problem: the possible different cluster are a finite (albeit exponential) number. At any iteration, c are selected to minimise the objective $f(\, z\, ,\, c\,)$ for the given z. Then, a new value z' is computed out of c, and $f(\, z'\, ,\, c\,) \leq f(\, z\, ,\, c\,)$, after which c' is computed out of z', and $f(\, z'\, ,\, c'\,) \leq f(\, z'\, ,\, c\,)$. If $z' = z$, then necessarily $c' = c$: thus, the value of f remains the same for two iterations and the algorithm stops (even if $\varepsilon = 0$). If this does not happen, then necessarily $z' \neq z$ and $f(\, z'\, ,\, c'\,) \leq f(\, z'\, ,\, c\,) < f(\, z\, ,\, c\,)$. This means that the (clusters implied by) z can never be repeated at any subsequent iteration, because if it were then also the value $f(\, z\, ,\, c\,)$ would repeat itself, which is not possible since it has strictly decreased and it can never increase. Thus, at each iteration a different z is produced (or the algorithm stops), and hence the total number of iterations is finite (albeit possibly exponential). This reasoning is akin to that that can be used to prove finite termination in the active-set method [**back**]

Solutions XI

- This can be easily done since $\| x \|_1 = \sum_i | x_i |$, and $| x | = \max\{ x \, , -x \}$. One then has to introduce (many) auxiliary variables w_{ipj} for $i \in I$, $p \in K$, and $j = 1, \ldots, h$, and the two constraints $w_{ipj} \geq c^p_j - x^i_j$, $w_{ipj} \geq -c^p_j + x^i_j$. Then, the objective function is replaced with $\sum_{i \in I} \sum_{p \in K} z_{ip}(\sum_{j=1,\ldots,h} w_{ipj})$. The constraints only imply that $w_{ipj} \geq \max\{ c^p_j - x^i_j \, , -c^p_j + x^i_j \}$, but when $z_{ip} = 1$, all the corresponding w_{ipj} have a positive coefficient and are minimised: thus, $w_{ipj} = \max\{ c^p_j - x^i_j \, , -c^p_j + x^i_j \}$ must necessarily hold in any optimal solution, as there are no other constraints but those two on the w_{ipj}. The same argument cannot be repeated when $z_{ip} = 0$, but on the other hand the value of the corresponding w_{ipj} is then irrelevant. All this proves that the two formulations are equivalent. The problem is now much larger but the added constraints are still linear, and the objective, while still being nonconvex, is now at least smooth [**back**]

- Exactly as before. Decomposability over p when z is fixed is trivial; then, since the objective is $\sum_{i \in I(z)} \sum_{j=1}^{h} |\, c_j - x_j^i \,|$, a component c_j has no "links" with any other component $c_{j'}$ for $j' \neq j$, and therefore each of the problems can be solved independently [**back**]

- Recall that the derivative of $|\, c - x^i \,|$ is -1 if $c < x^i$, $+1$ if $c > x^i$, and that the left derivative in $c = x^i$ is -1 while the right derivative is $+1$. We start in the odd t case by taking $\bar{t} = (t+1)/2$ and assuming that $x^{\bar{t}-1} < x^{\bar{t}} < x^{\bar{t}+1}$. Now, all the terms $|\, c - x^i \,|$ for $i = 1, \ldots, \bar{t} - 1$ provide a contribution of $+1$ to $f'(x^{\bar{t}})$, while those for $i = \bar{t} + 1, \ldots, t$ provide a contribution of -1. Since the two sets have the same cardinality, the two contributions cancel out: $f(x^{\bar{t}})$ has negative left derivative and positive right derivative as $|\, c - x^{\bar{t}} \,|$ does, which means that $x^{\bar{t}}$ is the unique minimum. The argument is similar in the even case if we assume that $x^{\bar{t}} < x^{\bar{t}+1}$, for $\bar{t} = t/2$. Then, for $x^{\bar{t}} < c < x^{\bar{t}+1}$ the contribution to the derivative of the $\bar{t} - 1$ terms $|\, c - x^i \,|$ with $i < \bar{t}$ cancels out with that of the $\bar{t} - 1$ terms with $i > \bar{t} + 1$ (note that this does not change if some of these x^i coincides with $x^{\bar{t}}$ or $x^{\bar{t}+1}$). Thus, the derivative is only determined by the two terms $|\, c - x^{\bar{t}} \,| + |\, c - x^{\bar{t}+1} \,|$, and it is plain to see that this is 0 for all $x^{\bar{t}} < c < x^{\bar{t}+1}$. Hence, the right derivative in $x^{\bar{t}}$ and the left derivative in $x^{\bar{t}+1}$ are both 0, which means that all $c \in [\, x^{\bar{t}} \,,\, x^{\bar{t}+1} \,]$ are optimal solutions; and they are the only ones, since the left derivative in $x^{\bar{t}}$ is negative and the right derivative in $x^{\bar{t}+1}$ is positive. The only remaining case is the one where there are multiple copies of the median value: that is, either t is odd and at least one of $x^{\bar{t}-1} = x^{\bar{t}}$, $x^{\bar{t}} = x^{\bar{t}+1}$ occurs (for $\bar{t} = (\, t + 1 \,) \,/\, 2$), or t is even and $x^{\bar{t}} = x^{\bar{t}+1}$ (for $\bar{t} = t \,/\, 2$). In either case we will prove that $x^{\bar{t}}$ is the unique minimum. Note that in the even t case $(\, x^{\bar{t}} + x^{\bar{t}+1} \,) \,/\, 2 = x^{\bar{t}}$, i.e., the formula of the even case still applies (which is why one should always use it). Let us denote as $t_0 > 1$ the number of i s.t. $x^i = x^{\bar{t}}$, as t_- those s.t. $x^i < x^{\bar{t}}$, and as t_+ those s.t. $x^i > x^{\bar{t}}$. The total contribution to $f'(\, x^{\bar{t}} \,)$ of the terms that have $x^i \neq x^{\bar{t}}$ is $t_- - t_+$. Also, the t_0 copies of the term $|\, c - x^{\bar{t}} \,|$ correspond to multiplying it by t_0, which means a left derivative of $-t_0$ and a right derivative of t_0. All in all, the left derivative in $x^{\bar{t}}$ is $t_- - t_+ - t_0$ and the right derivative is $t_- - t_+ + t_0$. For the first, we use the fact that $t_- < t \,/\, 2$ while $t_0 + t_+ > t \,/\, 2$ to obtain $t_- - t_+ - t_0 < t_- - t \,/\, 2 < 0$. Symmetrically, for the second we use $t_+ < t \,/\, 2$ while $t_0 + t_- > t \,/\, 2$ to obtain $t_- - t_+ + t_0 > t \,/\, 2 - t_+ > 0$. Hence, in $x^{\bar{t}}$ the left derivative is negative and the right derivative is positive, proving that once again $x^{\bar{t}}$ is the unique minimum of f **[back]**

Solutions XIV

- No, this last one is really left as exercise [**back**]