Software Engineer in the Machine Learning world

Tomasz Melcer, QuantUp

code::dive 2018, November 8th, 2018, Wrocław

About me

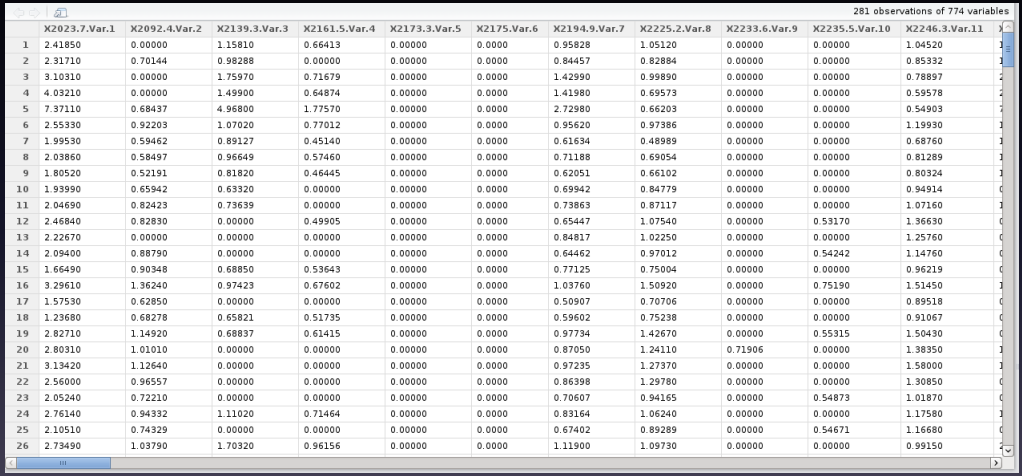
- QuantUp
  - Trainings/workshops, consulting in Machine Learning
- Wrocław University of Science and Technology
  - Bioinformatics, Digital Signal Processing and Machine Learning in glaucoma diagnostics
- DataX
  - Big Data in telecommunications
- Co-organizer of Data Science Wrocław
  - → https://www.meetup.com/Data-Science-Wroclaw/

QUANTUP

My first “real” job

QUANTUP

“Starter” problem: biomarkers identification

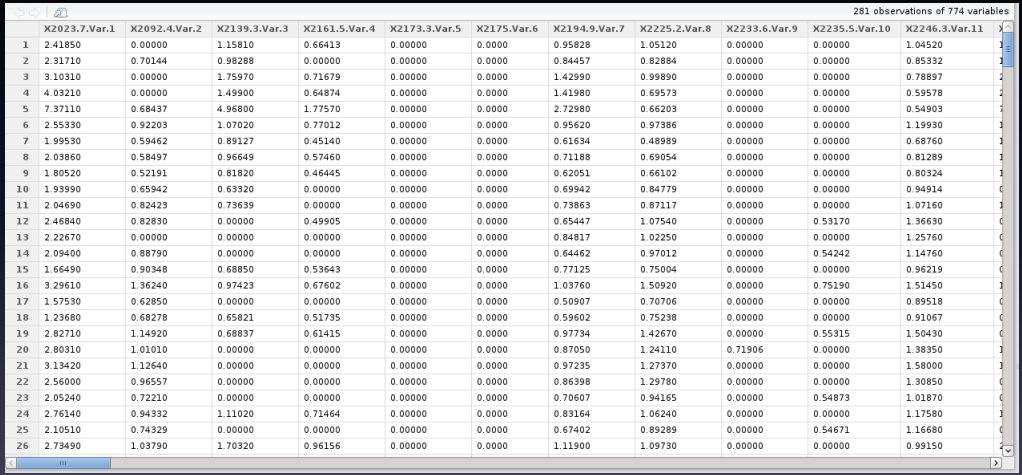


281 observations of 774 variables

	X2023.7.Var.1	X2092.4.Var.2	X2139.3.Var.3	X2161.5.Var.4	X2173.3.Var.5	X2175.Var.6	X2194.9.Var.7	X2225.2.Var.8	X2233.6.Var.9	X2235.5.Var.10	X2246.3.Var.11	X
1	2.41850	0.00000	1.15810	0.66413	0.00000	0.0000	0.95828	1.05120	0.00000	0.00000	1.04520	1
2	2.31710	0.70144	0.98288	0.00000	0.00000	0.0000	0.84457	0.82884	0.00000	0.00000	0.85332	1
3	3.10310	0.00000	1.75970	0.71679	0.00000	0.0000	1.42990	0.99890	0.00000	0.00000	0.78897	2
4	4.03210	0.00000	1.49900	0.64874	0.00000	0.0000	1.41980	0.69573	0.00000	0.00000	0.59578	2
5	7.37110	0.68437	4.96800	1.77570	0.00000	0.0000	2.72980	0.66203	0.00000	0.00000	0.54903	7
6	2.55330	0.92203	1.07020	0.77012	0.00000	0.0000	0.95620	0.97386	0.00000	0.00000	1.19930	1
7	1.99530	0.59462	0.89127	0.45140	0.00000	0.0000	0.61634	0.48989	0.00000	0.00000	0.68760	1
8	2.03860	0.58497	0.96649	0.57460	0.00000	0.0000	0.71188	0.69054	0.00000	0.00000	0.81289	1
9	1.80520	0.52191	0.81820	0.46445	0.00000	0.0000	0.62051	0.66102	0.00000	0.00000	0.80324	1
10	1.93990	0.65942	0.63320	0.00000	0.00000	0.0000	0.69942	0.84779	0.00000	0.00000	0.94914	0
11	2.04690	0.82423	0.73639	0.00000	0.00000	0.0000	0.73863	0.87117	0.00000	0.00000	1.07160	1
12	2.46840	0.82830	0.00000	0.49905	0.00000	0.0000	0.65447	1.07540	0.00000	0.53170	1.36630	0
13	2.22670	0.00000	0.00000	0.00000	0.00000	0.0000	0.84817	1.02250	0.00000	0.00000	1.25760	0
14	2.09400	0.88790	0.00000	0.00000	0.00000	0.0000	0.64462	0.97012	0.00000	0.54242	1.14760	0
15	1.66490	0.90348	0.68850	0.53643	0.00000	0.0000	0.77125	0.75004	0.00000	0.00000	0.96219	0
16	3.29610	1.36240	0.97423	0.67602	0.00000	0.0000	1.03760	1.50920	0.00000	0.75190	1.51450	1
17	1.57530	0.62850	0.00000	0.00000	0.00000	0.0000	0.50907	0.70706	0.00000	0.00000	0.89518	0
18	1.23680	0.68278	0.65821	0.51735	0.00000	0.0000	0.59602	0.75238	0.00000	0.00000	0.91067	0
19	2.82710	1.14920	0.68837	0.61415	0.00000	0.0000	0.97734	1.42670	0.00000	0.55315	1.50430	0
20	2.80310	1.01010	0.00000	0.00000	0.00000	0.0000	0.87050	1.24110	0.71906	0.00000	1.38350	1
21	3.13420	1.12640	0.00000	0.00000	0.00000	0.0000	0.97235	1.27370	0.00000	0.00000	1.58000	1
22	2.56000	0.96557	0.00000	0.00000	0.00000	0.0000	0.86398	1.29780	0.00000	0.00000	1.30850	0
23	2.05240	0.72210	0.00000	0.00000	0.00000	0.0000	0.70607	0.94165	0.00000	0.54873	1.01870	0
24	2.76140	0.94332	1.11020	0.71464	0.00000	0.0000	0.83164	1.06240	0.00000	0.00000	1.17580	1
25	2.10510	0.74329	0.00000	0.00000	0.00000	0.0000	0.67402	0.89289	0.00000	0.54671	1.16680	0
26	2.73490	1.03790	1.70320	0.96156	0.00000	0.0000	1.11900	1.09730	0.00000	0.00000	0.99150	2

QUANTUP

“Starter” problem: biomarkers identification



281 observations of 774 variables

	X2023.7.Var.1	X2092.4.Var.2	X2139.3.Var.3	X2161.5.Var.4	X2173.3.Var.5	X2175.Var.6	X2194.9.Var.7	X2225.2.Var.8	X2233.6.Var.9	X2235.5.Var.10	X2246.3.Var.11	X
1	2.41850	0.00000	1.15810	0.66413	0.00000	0.0000	0.95828	1.05120	0.00000	0.00000	1.04520	1
2	2.31710	0.70144	0.98288	0.00000	0.00000	0.0000	0.84457	0.82884	0.00000	0.00000	0.85332	1
3	3.10310	0.00000	1.75970	0.71679	0.00000	0.0000	1.42990	0.99890	0.00000	0.00000	0.78897	2
4	4.03210	0.00000	1.49900	0.64874	0.00000	0.0000	1.41980	0.69573	0.00000	0.00000	0.59578	2
5	7.37110	0.68437	4.96800	1.77570	0.00000	0.0000	2.72980	0.66203	0.00000	0.00000	0.54903	7
6	2.55330	0.92203	1.07020	0.77012	0.00000	0.0000	0.95620	0.97386	0.00000	0.00000	1.19930	1
7	1.99530	0.59462	0.89127	0.45140	0.00000	0.0000	0.61634	0.48989	0.00000	0.00000	0.68760	1
8	2.03860	0.58497	0.96649	0.57460	0.00000	0.0000	0.71188	0.69054	0.00000	0.00000	0.81289	1
9	1.80520	0.52191	0.81820	0.46445	0.00000	0.0000	0.62051	0.66102	0.00000	0.00000	0.80324	1
10	1.93990	0.65942	0.63320	0.00000	0.00000	0.0000	0.69942	0.84779	0.00000	0.00000	0.94914	0
11	2.04690	0.82423	0.73639	0.00000	0.00000	0.0000	0.73863	0.87117	0.00000	0.00000	1.07160	1
12	2.46840	0.82830	0.00000	0.49905	0.00000	0.0000	0.65447	1.07540	0.00000	0.53170	1.36630	0
13	2.22670	0.00000	0.00000	0.00000	0.00000	0.0000	0.84817	1.02250	0.00000	0.00000	1.25760	0
14	2.09400	0.88790	0.00000	0.00000	0.00000	0.0000	0.64462	0.97012	0.00000	0.54242	1.14760	0
15	1.66490	0.90348	0.68850	0.53643	0.00000	0.0000	0.77125	0.75004	0.00000	0.00000	0.96219	0
16	3.29610	1.36240	0.97423	0.67602	0.00000	0.0000	1.03760	1.50920	0.00000	0.75190	1.51450	1
17	1.57530	0.62850	0.00000	0.00000	0.00000	0.0000	0.50907	0.70706	0.00000	0.00000	0.89518	0
18	1.23680	0.68278	0.65821	0.51735	0.00000	0.0000	0.59602	0.75238	0.00000	0.00000	0.91067	0
19	2.82710	1.14920	0.68837	0.61415	0.00000	0.0000	0.97734	1.42670	0.00000	0.55315	1.50430	0
20	2.80310	1.01010	0.00000	0.00000	0.00000	0.0000	0.87050	1.24110	0.71906	0.00000	1.38350	1
21	3.13420	1.12640	0.00000	0.00000	0.00000	0.0000	0.97235	1.27370	0.00000	0.00000	1.58000	1
22	2.56000	0.96557	0.00000	0.00000	0.00000	0.0000	0.86398	1.29780	0.00000	0.00000	1.30850	0
23	2.05240	0.72210	0.00000	0.00000	0.00000	0.0000	0.70607	0.94165	0.00000	0.54873	1.01870	0
24	2.76140	0.94332	1.11020	0.71464	0.00000	0.0000	0.83164	1.06240	0.00000	0.00000	1.17580	1
25	2.10510	0.74329	0.00000	0.00000	0.00000	0.0000	0.67402	0.89289	0.00000	0.54671	1.16680	0
26	2.73490	1.03790	1.70320	0.96156	0.00000	0.0000	1.11900	1.09730	0.00000	0.00000	0.99150	2

My boss: *Please write some code to build cancer biomarker detectors.*

QUANTUP

Let’s start from something simpler

Which wine is likely to be considered good, based on chemical analysis?

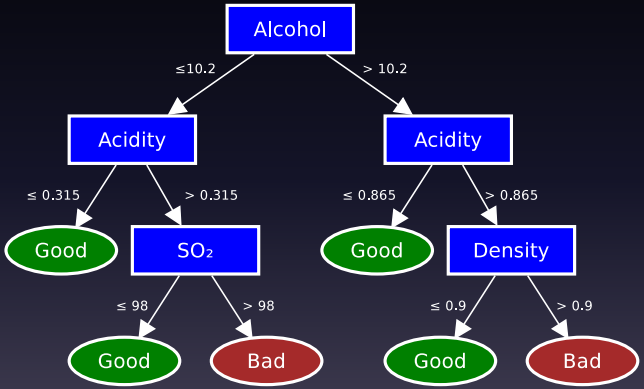
Is good?	Acidity	Sugar	Chlorides	SO_2	Density	Alcohol
Good	0.70	1.9	0.076	34	0.9978	9.4
Good	0.88	2.6	0.098	67	0.9968	9.8
Bad	0.28	1.9	0.075	60	0.9980	9.8
Good	0.62	1.5	0.080	119	0.99720	9.1
Bad	0.63	1.9	0.076	27	0.99670	9.5
Bad	0.70	1.9	0.074	19	0.99620	10.5
…	…	…	…	…	…	…

- Rows: “cases” (dataset: 1599 cases)
- Columns: “features” (dataset: 12 features)

A. Cerdeira, F. Almeida, T. Matos and J. Reis, *Viticulture Commission of the Vinho Verde Region*

QUANTUP

Decision trees



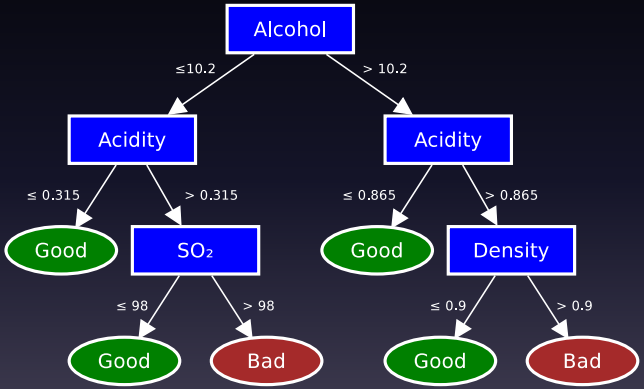
"Game of 20 questions"

Decision tree: a tree, where each node is:

- a question with two branches, or
- a decision.

QUANTUP

Decision trees



"Game of 20 questions"

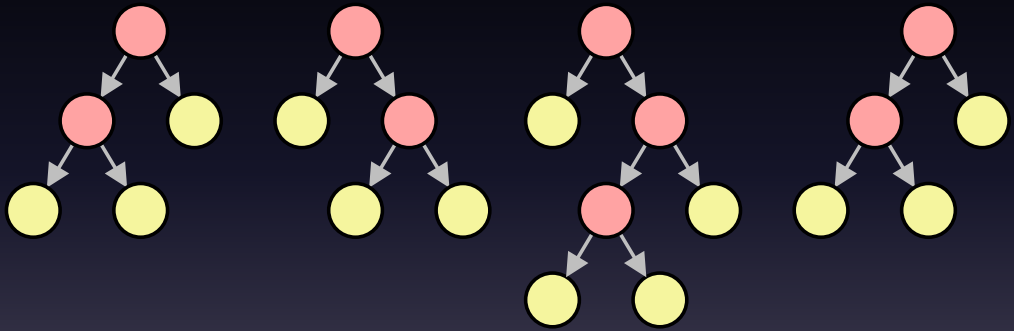
Decision tree: a tree, where each node is:

- a question with two branches, or
- a decision.

How to build a decision tree?

QUANTUP

Decision trees: Shape?



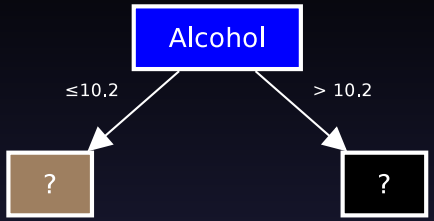
Exponentially many shapes. How to choose?

QUANTUP

Decision trees: Shape.

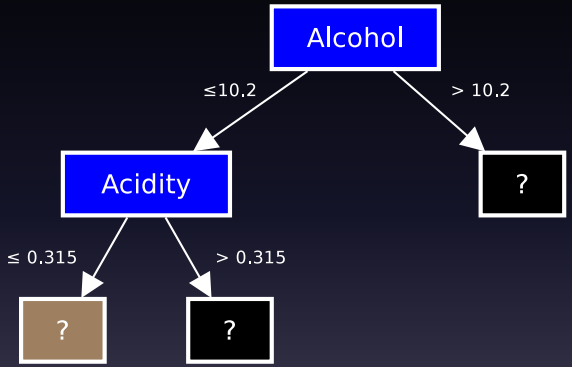
QUANTUP

Decision trees: Shape.



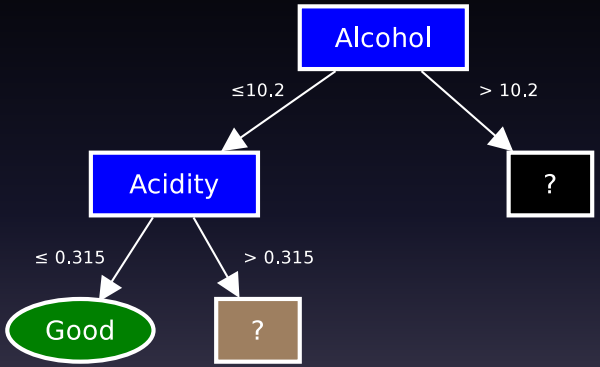
QUANTUP

Decision trees: Shape.



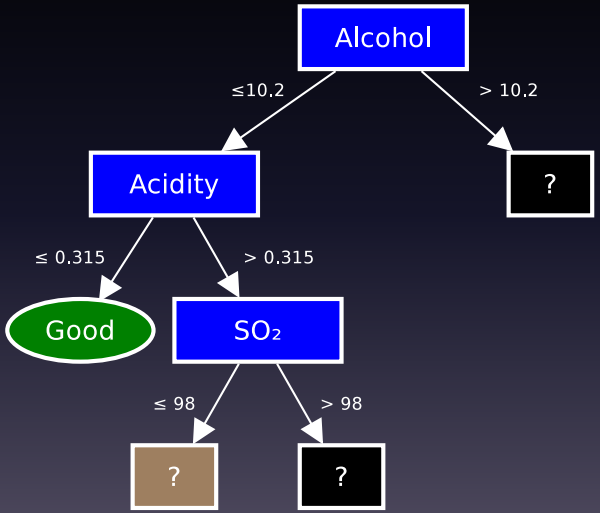
QUANTUP

Decision trees: Shape.



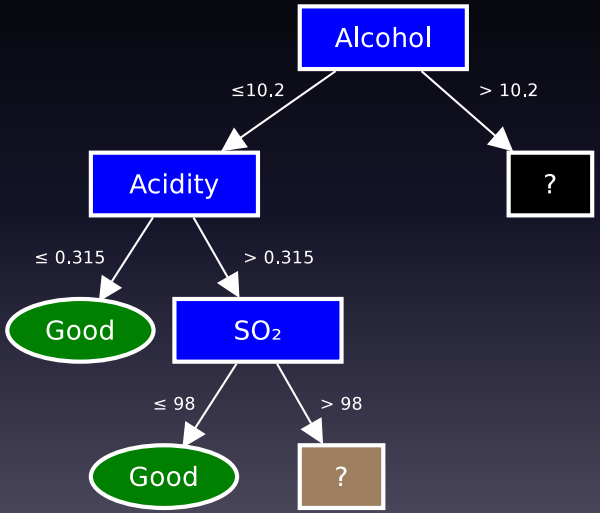
QUANTUP

Decision trees: Shape.



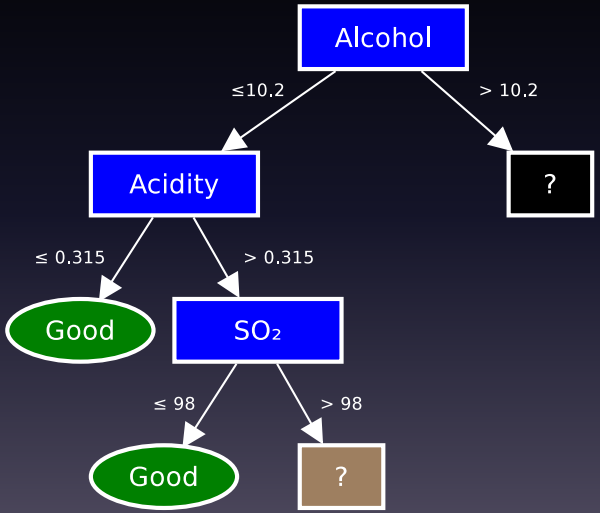
QUANTUP

Decision trees: Shape.



QUANTUP

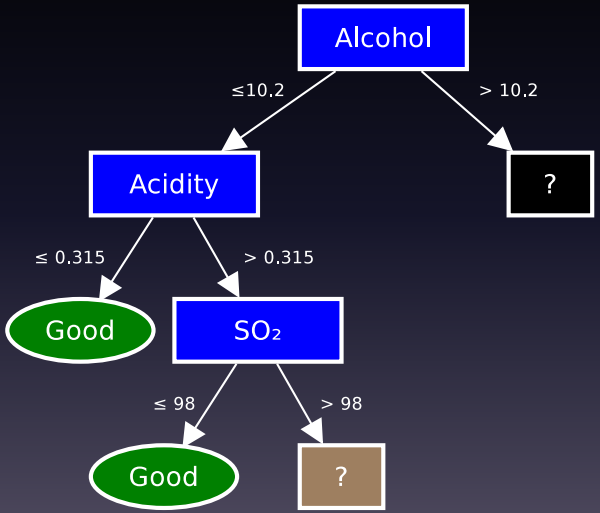
Decision trees: Shape.



Greedy strategy.
Does not lead to perfect decisions, but it's **fast** and **"good enough"**.

QUANTUP

Decision trees: Shape.



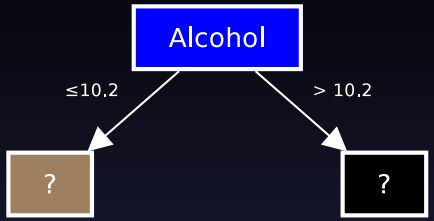
Greedy strategy.
Does not lead to perfect decisions, but it's **fast** and **"good enough"**.

A lot of machine learning algorithms prefer "good enough" to "theoretically sound"!

- *e.g. neural networks…*

QUANTUP

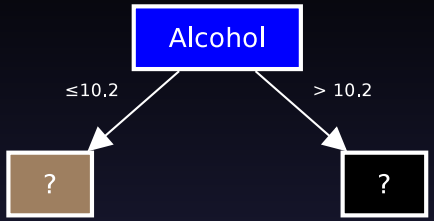
Decision trees: What questions?



What questions?

QUANTUP

Decision trees: What questions?

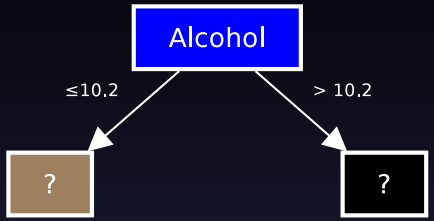


What questions?

…try **every** possible question and pick the “best” one. **Every** column (“feature”), **every** threshold value. If none good, make a leaf.

QUANTUP

Decision trees: What questions?



What questions?

…try **every** possible question and pick the "best" one. **Every** column ("feature"), **every** threshold value. If none good, make a leaf.

Fast enough (for small datasets), smart implementation $O(kn \log n)$.

When in doubt, use brute force
(Ken Thompson)

QUANTUP

I’ve got a prototype! Testing?

Ok, I wrote a prototype, how do I test the code?

- **simple synthetic data**

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Not always simple, e.g. for neural networks (Minsky and Papert, *Perceptron: an introduction to computational geometry*, 1969; Rumelhart et al., *Learning internal representations by error propagation*, 1986).

QUANTUP

I’ve got a prototype! Testing?

Ok, I wrote a prototype, how do I test the code?

- simple synthetic data
- **simple random data**

Depending on algorithm, not easily known what can be deduced.

QUANTUP

I’ve got a prototype! Testing?

Ok, I wrote a prototype, how do I test the code?

- simple synthetic data
- simple random data
- **real datasets, reproduce results from literature**

Time-consuming, not much business value. Great for libraries, though.

QUANTUP

I’ve got a prototype! Testing?

Ok, I wrote a prototype, how do I test the code?

- simple synthetic data
- simple random data
- real datasets, reproduce results from literature
- **compare to existing implementations**

Somewhat time-consuming.

QUANTUP

I’ve got a prototype! Testing?

Ok, I wrote a prototype, how do I test the code?

- simple synthetic data
- simple random data
- real datasets, reproduce results from literature
- compare to existing implementations

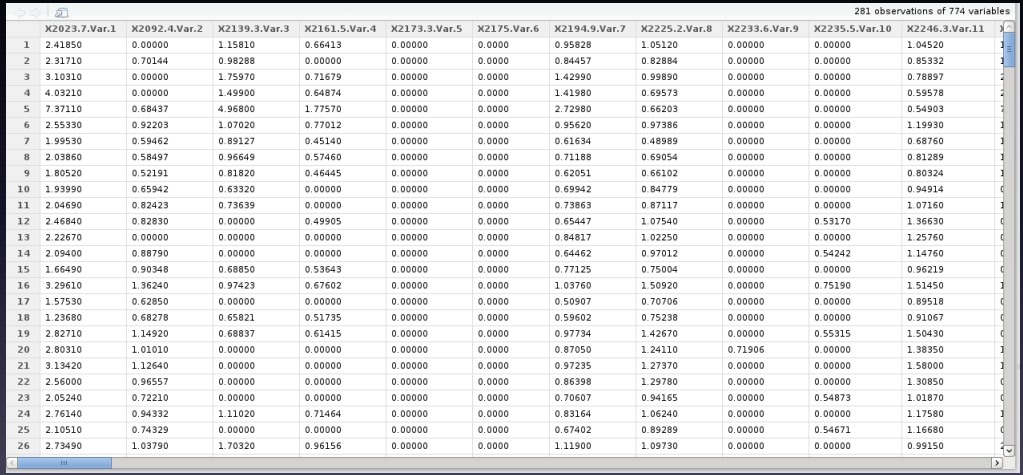
I wrote a prototype in Python, then wrote some synthetic tests. Then I wrote production code in C++ and coded unit tests that compared it on random and real datasets (“gold set”) with the prototype.

QUANTUP

The Pipeline

QUANTUP

Where do these numbers come from?

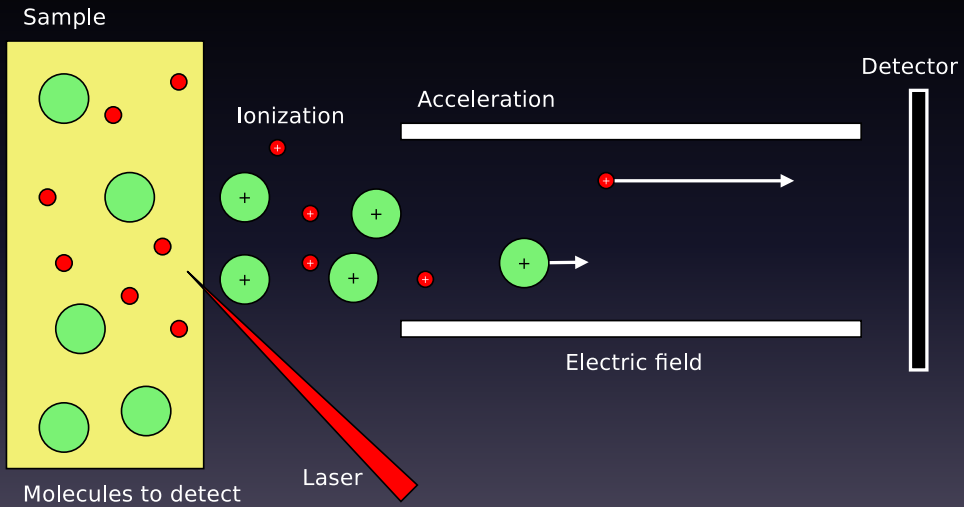


281 observations of 774 variables

	X2023.7.Var.1	X2092.4.Var.2	X2139.3.Var.3	X2161.5.Var.4	X2173.3.Var.5	X2175.Var.6	X2194.9.Var.7	X2225.2.Var.8	X2233.6.Var.9	X2235.5.Var.10	X2246.3.Var.11	X
1	2.41850	0.00000	1.15810	0.66413	0.00000	0.0000	0.95828	1.05120	0.00000	0.00000	1.04520	1
2	2.31710	0.70144	0.98288	0.00000	0.00000	0.0000	0.84457	0.82884	0.00000	0.00000	0.85332	1
3	3.10310	0.00000	1.75970	0.71679	0.00000	0.0000	1.42990	0.99890	0.00000	0.00000	0.78897	2
4	4.03210	0.00000	1.49900	0.64874	0.00000	0.0000	1.41980	0.69573	0.00000	0.00000	0.59578	2
5	7.37110	0.68437	4.96800	1.77570	0.00000	0.0000	2.72980	0.66203	0.00000	0.00000	0.54903	7
6	2.55330	0.92203	1.07020	0.77012	0.00000	0.0000	0.95620	0.97386	0.00000	0.00000	1.19930	1
7	1.99530	0.59462	0.89127	0.45140	0.00000	0.0000	0.61634	0.48989	0.00000	0.00000	0.68760	1
8	2.03860	0.58497	0.96649	0.57460	0.00000	0.0000	0.71188	0.69054	0.00000	0.00000	0.81289	1
9	1.80520	0.52191	0.81820	0.46445	0.00000	0.0000	0.62051	0.66102	0.00000	0.00000	0.80324	1
10	1.93990	0.65942	0.63320	0.00000	0.00000	0.0000	0.69942	0.84779	0.00000	0.00000	0.94914	0
11	2.04690	0.82423	0.73639	0.00000	0.00000	0.0000	0.73863	0.87117	0.00000	0.00000	1.07160	1
12	2.46840	0.82830	0.00000	0.49905	0.00000	0.0000	0.65447	1.07540	0.00000	0.53170	1.36630	0
13	2.22670	0.00000	0.00000	0.00000	0.00000	0.0000	0.84817	1.02250	0.00000	0.00000	1.25760	0
14	2.09400	0.88790	0.00000	0.00000	0.00000	0.0000	0.64462	0.97012	0.00000	0.54242	1.14760	0
15	1.66490	0.90348	0.68850	0.53643	0.00000	0.0000	0.77125	0.75004	0.00000	0.00000	0.96219	0
16	3.29610	1.36240	0.97423	0.67602	0.00000	0.0000	1.03760	1.50920	0.00000	0.75190	1.51450	1
17	1.57530	0.62850	0.00000	0.00000	0.00000	0.0000	0.50907	0.70706	0.00000	0.00000	0.89518	0
18	1.23680	0.68278	0.65821	0.51735	0.00000	0.0000	0.59602	0.75238	0.00000	0.00000	0.91067	0
19	2.82710	1.14920	0.68837	0.61415	0.00000	0.0000	0.97734	1.42670	0.00000	0.55315	1.50430	0
20	2.80310	1.01010	0.00000	0.00000	0.00000	0.0000	0.87050	1.24110	0.71906	0.00000	1.38350	1
21	3.13420	1.12640	0.00000	0.00000	0.00000	0.0000	0.97235	1.27370	0.00000	0.00000	1.58000	1
22	2.56000	0.96557	0.00000	0.00000	0.00000	0.0000	0.86398	1.29780	0.00000	0.00000	1.30850	0
23	2.05240	0.72210	0.00000	0.00000	0.00000	0.0000	0.70607	0.94165	0.00000	0.54873	1.01870	0
24	2.76140	0.94332	1.11020	0.71464	0.00000	0.0000	0.83164	1.06240	0.00000	0.00000	1.17580	1
25	2.10510	0.74329	0.00000	0.00000	0.00000	0.0000	0.67402	0.89289	0.00000	0.54671	1.16680	0
26	2.73490	1.03790	1.70320	0.96156	0.00000	0.0000	1.11900	1.09730	0.00000	0.00000	0.99150	2

QUANTUP

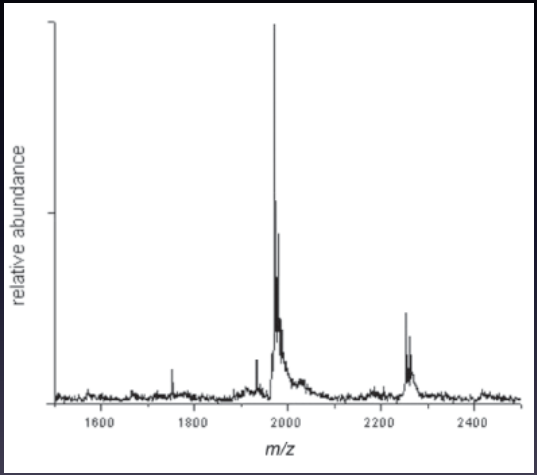
Original data: collection



Matrix-Assisted Laser Desorption/Ionization Time of Flight (MALDI-TOF)

QUANTUP

Original data

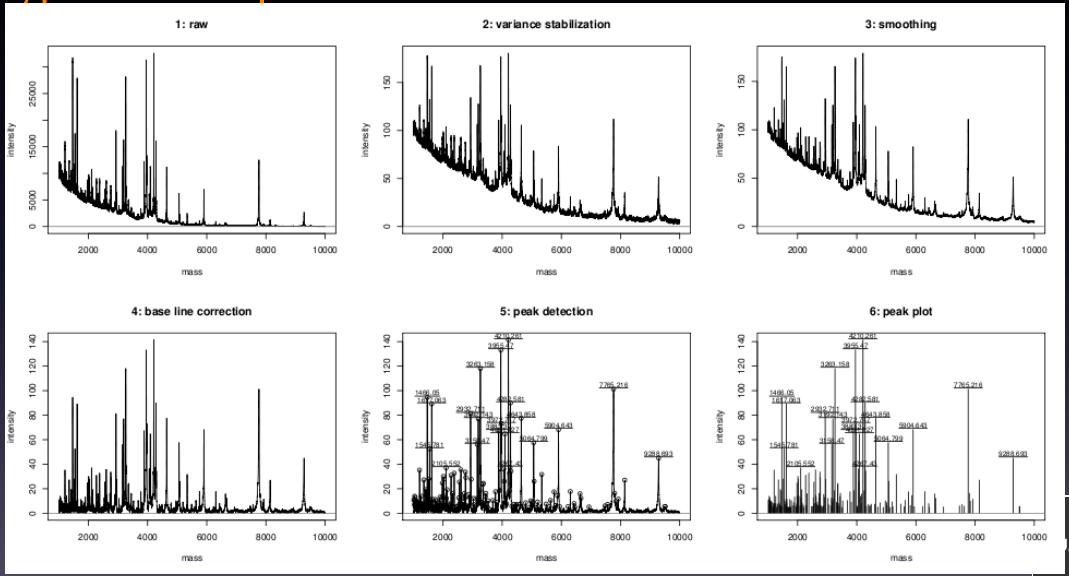


- Time series
- No clear information on molecules
- Noisy?
- Do we really have small amounts of molecules of every possible weight?
- Need to extract that data before applying machine learning

Zhang, Song, *Detection of retinoic acid receptor complex using mass spectrometry*, J. Braz. Chem. Soc. vol.18 no.3 São Paulo 2007

QUANTUP

Original data: problems



Sebastian Gibb, *MALDIquant: Quantitative Analysis of Mass Spectrometry Data*, 2018

The Pipeline

Steps:

- Running MALDI-TOF procedure
- Loading data
- Denoising
- Peak detection
- Binning across multiple spectra
- Peak identification
- Machine learning

QUANTUP

The Pipeline

Steps:

- Running MALDI-TOF procedure
- Loading data
- Denoising
- Peak detection
- Binning across multiple spectra
- Peak identification
- Machine learning

- *Each* step vital. *Each* step can be tuned.
- **Changing anything** (in the pipeline) **changes everything** (in the result), e.g.:
  - Less denoising → more peaks → bigger false-positive rate
  - More denoising → less peaks → bigger false-negative rate
- No easy tests for data correctness

QUANTUP

The Pipeline: summary

From the codebase:

- Around 90% of code was GUI, reporting, etc.
- Around 10% of code was domain-specific computations.
- Less than 1% implemented machine learning algorithms.
- Machine learning was the **easy** part.

QUANTUP

The Pipeline: summary

From the codebase:

- Around 90% of code was GUI, reporting, etc.
- Around 10% of code was domain-specific computations.
- Less than 1% implemented machine learning algorithms.
- Machine learning was the **easy** part.

From later experience:

- **This is true for most projects**

QUANTUP

The Pipeline: summary

From the codebase:

- Around 90% of code was GUI, reporting, etc.
- Around 10% of code was domain-specific computations.
- Less than 1% implemented machine learning algorithms.
- Machine learning was the **easy** part.

From later experience:

- **This is true for most projects**

- Unless ML model is the core business idea, most of effort elsewhere.
- Simple ML algorithms often good enough.

QUANTUP

Making sure things work

QUANTUP

Two components of a good model

- Does the code do what we think it does?
  - Unit tests, simple synthetic tests, etc.

QUANTUP

Two components of a good model

- Does the code do what we think it does?
  - Unit tests, simple synthetic tests, etc.
- Does the modelling approach work well on our data?
  - Validation schemes.

QUANTUP

Two components of a good model

- Does the code do what we think it does?
  - Unit tests, simple synthetic tests, etc.
- Does the modelling approach work well on our data?
  - Validation schemes.

Compare: UX design, project requirements, etc.

QUANTUP

The Black Box

A machine learning model is a black box.

- We don’t know **how** it works*.
- We don’t know **why** it works.

* Mostly.

QUANTUP

The Black Box

A machine learning model is a black box.

- We don’t know **how** it works*.
- We don’t know **why** it works.

* Mostly.

Why do we still use it?

- Because we need **some** solution, and…

QUANTUP

The Black Box

A machine learning model is a black box.

- We don’t know **how** it works*.
- We don’t know **why** it works.

* Mostly.

Why do we still use it?

- Because we need **some** solution, and…
- Because it works.

QUANTUP

The Black Box

A machine learning model is a black box.

- We don’t know **how** it works*.
- We don’t know **why** it works.

* Mostly.

Why do we still use it?

- Because we need **some** solution, and…
- Because it works.
- **Usually.**

QUANTUP

What could go wrong?

Where a perceptron had been trained to distinguish between - this was for military purposes - It could… it was looking at a scene of a forest in which there were camouflaged tanks in one picture and no camouflaged tanks in the other. And the perceptron - after a little training - got… made a 100% correct distinction between these two different sets of photographs. Then they were embarrassed a few hours later to discover that the two rolls of film had been developed differently. And so these pictures were just a little darker than all of these pictures and the perceptron was just measuring the total amount of light in the scene. But it was very clever of the perceptron to find some way of making the distinction.

Marvin Minsky, the founder of the AI Lab at MIT

QUANTUP

Generalization

Generalization is an ability of a model to be correct for data it has not seen during training.
We need to **measure** generalization.

QUANTUP

Generalization

Generalization is an ability of a model to be correct for data it has not seen during training.
We need to **measure** generalization.

Example: object detection.

- ImageNet: 10M pictures, 10k object categories
- Set aside 1M pictures.
- Train on 9M pictures.
- Check how well it works on 1M pictures set aside.

“Train/test split”

QUANTUP

Generalization

Generalization is an ability of a model to be correct for data it has not seen during training.
We need to **measure** generalization.

Example: object detection.

- ImageNet: 10M pictures, 10k object categories
- Set aside 1M pictures.
- Train on 9M pictures.
- Check how well it works on 1M pictures set aside.

“Train/test split”

Why not just test on the same dataset as train?

QUANTUP

Generalization

Generalization is an ability of a model to be correct for data it has not seen during training.
We need to **measure** generalization.

Example: object detection.

- ImageNet: 10M pictures, 10k object categories
- Set aside 1M pictures.
- Train on 9M pictures.
- Check how well it works on 1M pictures set aside.

“Train/test split”

Why not just test on the same dataset as train? **Memorization.**

QUANTUP

Unseen data

What does it mean “unseen data”?

QUANTUP

Unseen data

What does it mean “unseen data”?
Example: fashion recommendation model trained on Polish clothing.

- Will it work in Poland?
  - Test set: random items from the same dataset, set aside from training.

QUANTUP

Unseen data

What does it mean “unseen data”?
Example: fashion recommendation model trained on Polish clothing.

- Will it work in Poland?
  - Test set: random items from the same dataset, set aside from training.
- Will it work in Australia?
  - Test set: items collected in different place (different way?)

QUANTUP

Unseen data

What does it mean “unseen data”?
Example: fashion recommendation model trained on Polish clothing.

- Will it work in Poland?
  - Test set: random items from the same dataset, set aside from training.
- Will it work in Australia?
  - Test set: items collected in different place (different way?)
- Will it work in Poland… 20 years from now?
  - Test set: training/test split by date.

QUANTUP

Unseen data

What does it mean “unseen data”?
Example: fashion recommendation model trained on Polish clothing.

- Will it work in Poland?
  - Test set: random items from the same dataset, set aside from training.
- Will it work in Australia?
  - Test set: items collected in different place (different way?)
- Will it work in Poland… 20 years from now?
  - Test set: training/test split by date.

Other examples: almost **any** kind of human behavior.
Other domains: e.g. databases, optimization algorithms.

QUANTUP

Which model is better?

QUANTUP

Which algorithm is better?

- decision trees?
  - simple trees?
  - random forests?
  - gradient boosted trees?
- neural networks?
  - convolutional neural networks?
  - recurrent neural networks?
- linear regression?
  - with regularization?
- naïve Bayes?

QUANTUP

Which algorithm is better?

- decision trees?
  - simple trees?
  - random forests?
  - gradient boosted trees?
- neural networks?
  - convolutional neural networks?
  - recurrent neural networks?
- linear regression?
  - with regularization?
- naïve Bayes?

No free lunch!

- different **inductive bias**

QUANTUP

Which algorithm is better?

- decision trees?
  - simple trees?
  - random forests?
  - gradient boosted trees?
- neural networks?
  - convolutional neural networks?
  - recurrent neural networks?
- linear regression?
  - with regularization?
- naïve Bayes?

No free lunch!

- different **inductive bias**

Also,

- different properties, e.g. stability
- slower/faster
- less/more memory needed
- less/more data needed

etc.

QUANTUP

So how do I choose?

How do I choose between several black boxes?

QUANTUP

So how do I choose?

How do I choose between several black boxes?

Test them.

QUANTUP

So how do I choose?

How do I choose between several black boxes?

Test them.

Split data into train/test, train each algorithm on the training set, then test on the test set…

QUANTUP

So how do I choose?

How do I choose between several black boxes?

Test them.

~~Split data into train/test, train each algorithm on the training set, then test on the test set…~~

That would be too easy, right?

QUANTUP

So how do I choose?

How do I choose between several black boxes?
Test them.

~~Split data into train/test, train each algorithm on the training set, then test on the test set…~~
That would be too easy, right?

- We lose correctness of test set estimates (introduction of **bias**).

QUANTUP

So how do I choose?

How do I choose between several black boxes?
Test them.

~~Split data into train/test, train each algorithm on the training set, then test on the test set…~~
That would be too easy, right?

- We lose correctness of test set estimates (introduction of **bias**).

- We need another held-out dataset, **validation dataset**.
- Split the initial data into training, validation and test dataset. Train on training, select based on validation performance, then check results on the test set.

QUANTUP

So how do I choose?

How do I choose between several black boxes?
Test them.

~~Split data into train/test, train each algorithm on the training set, then test on the test set…~~
That would be too easy, right?

- We lose correctness of test set estimates (introduction of **bias**).

- We need another held-out dataset, **validation dataset**.
- Split the initial data into training, validation and test dataset. Train on training, select based on validation performance, then check results on the test set.

- The same story for **hyperparameters**.

QUANTUP

This sounds like a lot of work!

It does. Make it as simple as possible. Some standard practices:

- High-level libraries
- Modular design
- etc.

Some more specific to experimental work:

- Interactive environments (e.g. notebooks).
- Expressive languages with terse syntax (e.g. Python, R).
- Dedicated tools for monitoring experiments (e.g. `tensorboard`)
- Dedicated tools for managing experiments (e.g. `sacred`)
- ie. proper tooling.

QUANTUP

Sacred

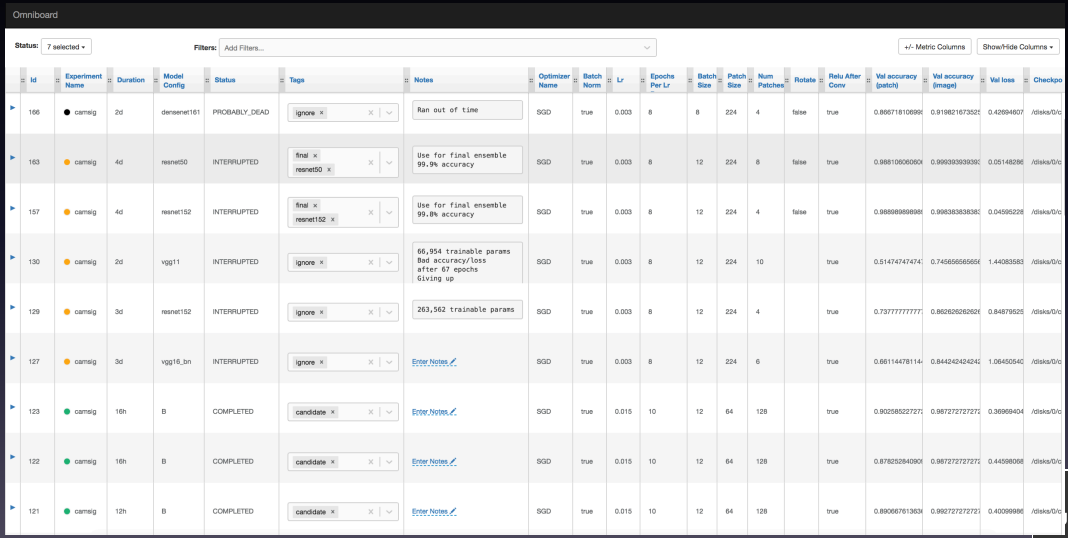
```
from numpy.random import permutation
from sklearn import svm, datasets
from sacred import Experiment
ex = Experiment('iris_rbf_svm')

@ex.config
def cfg():
  C = 1.0
  gamma = 0.7

@ex.automain
def run(C, gamma):
  iris = datasets.load_iris()
  per = permutation(iris.target.size)
  iris.data = iris.data[per]
  iris.target = iris.target[per]
  clf = svm.SVC(C, 'rbf', gamma=gamma)
  clf.fit(iris.data[:90],
          iris.target[:90])
  return clf.score(iris.data[90:],
                   iris.target[90:])
```

QUANTUP

Omniboard



Summary

Summary

- Approximations to make (some) things manageable
- Brute force when possible
- Change anything — changes everything
- Writing ML code, building ML models: it's *research*.
- Experiments, tests necessary for everything.

Contact

- During the conference!
- After the conference: tomasz@quantup.pl

QUANTUP