Mechatronic Engineering program

Basics of AI and Deep Learning:
5: Practical data processing

Ziemowit Dworakowski
AGH University of Krakow

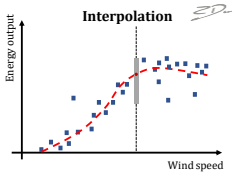
1

General data processing and model testing loop

2

Difference between classic and AI-based data processing

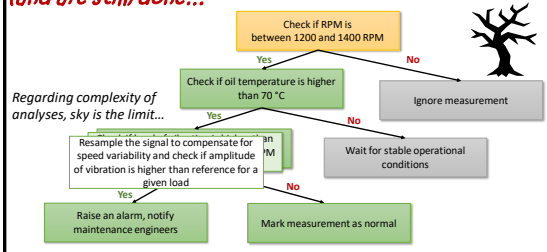
Classical



3

How things were traditionally (and are still) done...

Decision tree



Regarding complexity of analyses, sky is the limit...

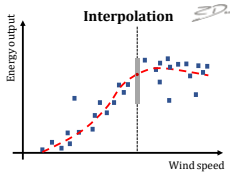
The simplest way of decision automation is just to replace particular steps with results of machine learning models or adjust the tree in autonomous fashion (e.g. by changing thresholds)

4

Difference between classic and AI-based data processing

Classical

Interpolation



Advantages:

- We can control all the variables and explain why result is as it is
- Mostly we use decision trees in the process

Limitations:

- Do we really need to spend time on repetitive tasks?
- How can we automate expert's actions?
- What to do, if there are more than 3 data dimensions?

5

Machine learning

- Provides definition of a task, guides data processing, assembles training datasets, assists in model training

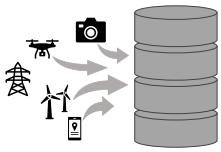
Lets Focus on that for now

Deep learning

- Provides definition of a task, assists in model training

6

Sources of data



It does not matter whether you work with images, Fourier spectra, waveforms, etc. In the end, you need to design a way of mapping all that into features…

We gather:
- Vectors of variables (features)
- 1D signals (e.g. time-domain)
- 2D signals (e.g. Images)
- Descriptions

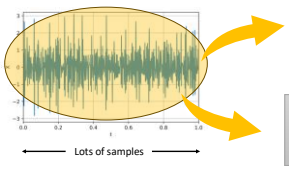
Feature extraction

Features are the only thing that can be used in typical decision systems

7

Feature extraction

lets use one of these two approaches



Overproduce features (calculate much more than we need) and then **select** some that provide good results

This looks like an optimization problem…

Think about how we would „manually" recognize classes and mimic features that allow for that

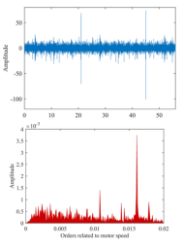
And this require actual domain knowledge

Either way we'll often be doing lots of advanced signal processing to extract information that is necessary…

8

Feature extraction

Lets do a few examples:
time domain vibration signal for machine state recognition



RMS contains information about the energy stored in vibration

Kurtosis of signal refer to how „peaky" it is

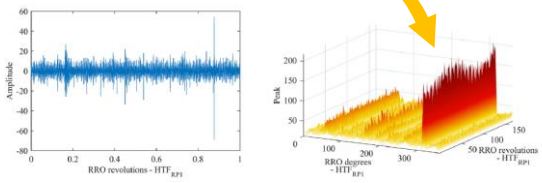
Calculating frequency or order spectrum allows calculating **energy in bands** (which may refer to particular damage types)

If we look for particular types of damage, we can craft features dedicated to that

9

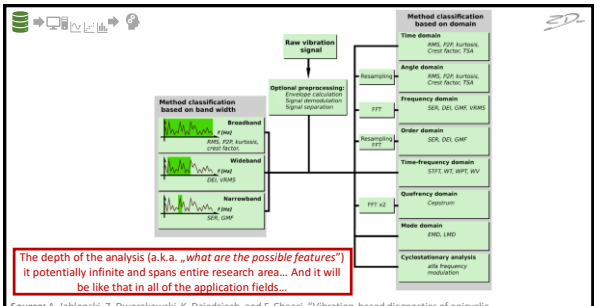
Feature extraction

This ridge refers to damage of particular tooth of a particular element of a planetary gearbox



Source: K. Dziedziech, A. Jablonski, and Z. Dworakowski, "Epicyclic gearbox fault detection by Instantaneous Circular Pitch Cycle Map," *Mech. Syst. Signal Process.*, vol. 121, pp. 600–614, 2019

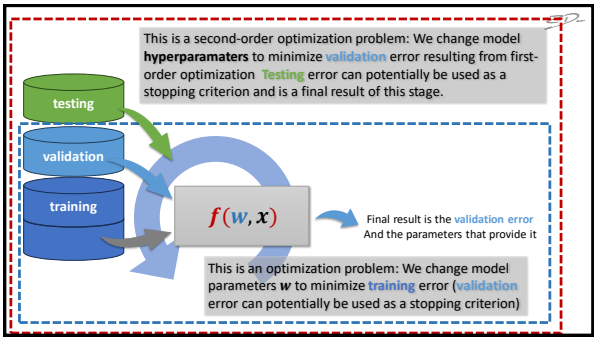
10



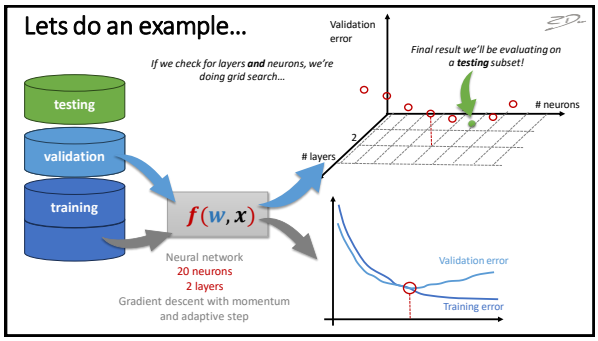
The depth of the analysis (a.k.a. „*what are the possible features*") it potentially infinite and spans entire research area… And it will be like that in all of the application fields…

Source: A. Jablonski, Z. Dworakowski, K. Dziedziech, and F. Chaari, "Vibration-based diagnostics of epicyclic gearboxes – From classical to soft-computing methods," *Measurement*, vol. 147, p. 106811, 2019

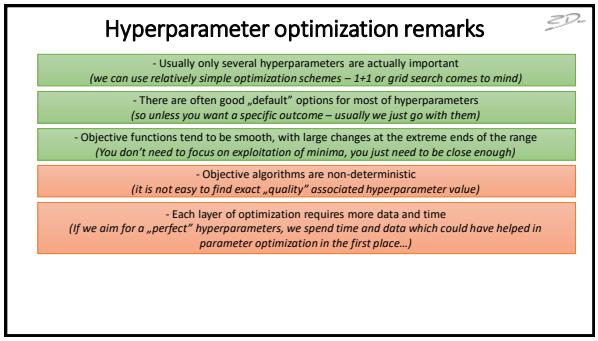
11



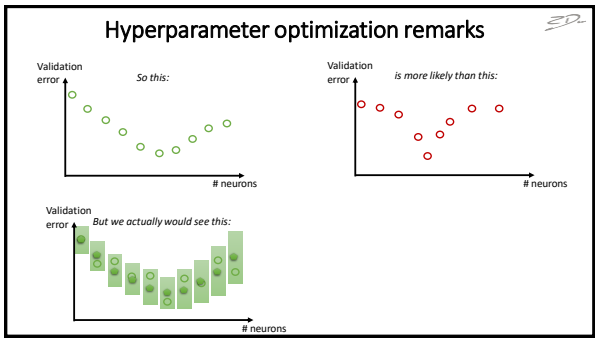
12


Lets do an example...
Validation error
Final result we'll be evaluating on a testing subset!
If we check for layers and neurons, we're doing grid search...
neurons
layers
testing
validation
training
$f(w, x)$
Validation error
Training error
Neural network
20 neurons
2 layers
Gradient descent with momentum and adaptive step

13


Hyperparameter optimization remarks
- Usually only several hyperparameters are actually important
(we can use relatively simple optimization schemes – 1+1 or grid search comes to mind)
- There are often good „default" options for most of hyperparameters
(so unless you want a specific outcome – usually we just go with them)
- Objective functions tend to be smooth, with large changes at the extreme ends of the range
(You don't need to focus on exploitation of minima, you just need to be close enough)
- Objective algorithms are non-deterministic
(it is not easy to find exact „quality" associated hyperparameter value)
- Each layer of optimization requires more data and time
(If we aim for a „perfect" hyperparameters, we spend time and data which could have helped in parameter optimization in the first place...)

14


Hyperparameter optimization remarks
Validation error
So this:
neurons
Validation error
is more likely than this:
neurons
Validation error
But we actually would see this:
neurons

15

So how do we actually do it in practice?

Approach 1: Make sure only that the model is complex enough to grasp data and go with „defaults" – or copy hyperparameter values from a similar setup
(Saves time and data)

Approach 2: Pick a reasonable starting point (default?) and then slightly change one hyperparameter at a time checking if it helps in a statistically significant way – until we run out of time (a 1+1 approach)
(We are allocating time and data resources as they are needed -
but we can end up in a local minimum or stagnate)

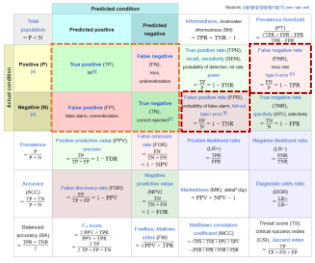
Approach 3: Decide which hyperparameters will influence the result the most – and do full optimization routine only for those (using grid search)
(We risk doing lots of computations that will end in conclusion that it does not matter much…)

Approach 4: Do a one-dimensional optimization for all hyperparameters separately
(Easy optimization but high risk of local minima here.
Hyperparameters often influence each other to a large extent)

16

And finally – model evaluation!

So far, we've just counted errors (in classification) and MSE (in regression).
But this is far from complete…



In classification, apart from **accuracy** („general correct classifications") two most important measures are percentage probability of observing errors of type „positive" and „negative" for each class:
False positive rate and **False negative rate**

How often is object from a different class assigned this class label?

How often is object from this class assigned a wrong label?

17

Confusion matrix

	Actual A (100)	Actual B (200)	Actual C (100)	Actual D (20)
Predicted A	100	10	30	0
Predicted B	0	150	10	0
Predicted C	0	0	60	0
Predicted D	0	40	0	20

We can use confusion matrix for:
- Estimation of probability of correct classification

- Estimation of probability of class presence

- Experiment planning (which classes require more samples)

- Model tuning (which classes require higher accuracy)

18

What about regression?

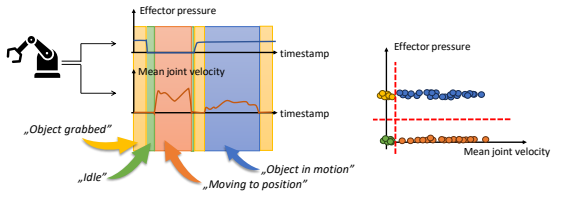
MAE (Mean absolute error) ← *Easy interpretation (same unit)*

MSE (Mean squared error) ← *Focuses on „important errors"*

RMSE (Root mean squared error) ← *Focuses on „important errors" but is also easy to interpret*

19

Application for machine operational state recognition



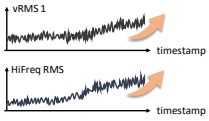
We can of course have much more operational variables (rendering the feature space to be highly multidimensional). But still,

This is a classification task – which we know how to solve!

20

Application for prediction (e.g. RUL)

Imagine a rotating machinery, for which we have the following measurements:



Increase in vRMS and high-frequency vibration suggests damaged bearings…

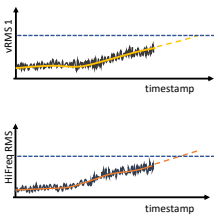
So they will break…

Someday…

But when?

21

Application for prediction (e.g. RUL)



We can measure at which level of features bearing usually breaks – and then check when this level will be breached based on our model of damage growth

It is a regression task which we know how to do!

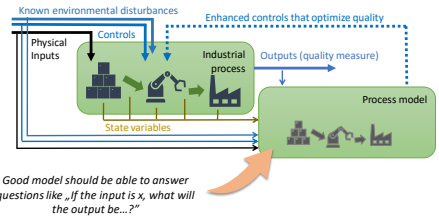
Or better:

We can gather a database connecting feature levels with Remaining Useful Life (RUL) of a bearing and then do regression based on our current features

Which we again know how to do!

22

Application for process modeling

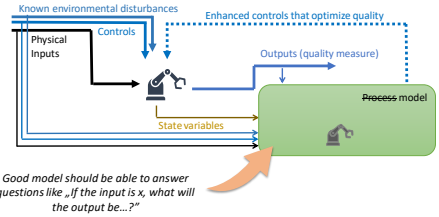


Good model should be able to answer questions like „If the input is x, what will the output be…?"

And this is a regression task – which we know how to solve!

23

Application for identification



Good model should be able to answer questions like „If the input is x, what will the output be…?"

And this is a regression task – which we know how to solve!

24

One-class classification (novelty detection)

Sometimes we have a dataset with lots of examples belonging to just one class and none or almost none to the other classes:

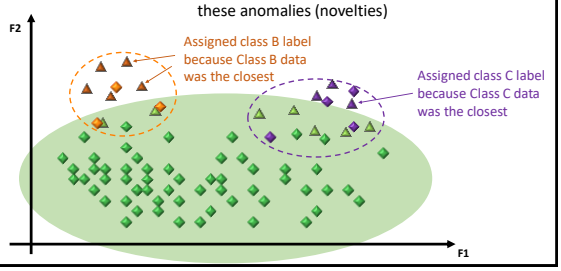
- „**Healthy** people"
- „**Normal** behavior in subway"
- „**Normal** operation state of an assembly line
- „**Typical** weather conditions in September"

Then, our goal might be to learn this **normal** range of data, to detect any **anomalies (outliers, novelties)**

In Novelty Detection we usually don't know what to expect (there is possibly Infinite set of „norm breaches")

25

1. Learn the normal range
2. Detect what breaches normal range
3. If you have other class data, use it to classify only these anomalies (novelties)



26

One-class classification (novelty detection)

How to learn „normal range"?

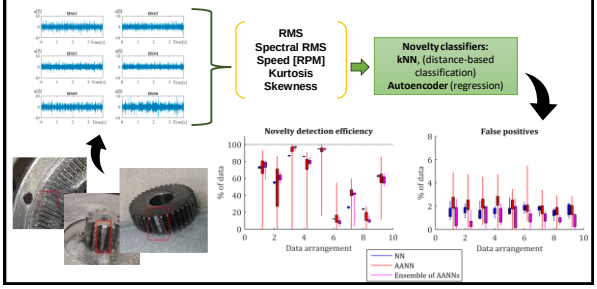
1. Estimate typical distances between samples within training set, detect anything that breaches it > e.g. **kNN**

2. Estimate underlying probability density for data – e.g. by fitting multidimensional gaussians into training data -> e.g. **GMM**

3. Train a system to derive some of the features from others – in normal range (for known data) it should work very good, for novel samples it should produce large errors > e.g. **Multidimensional regression**

27

Novelty detection application example



28

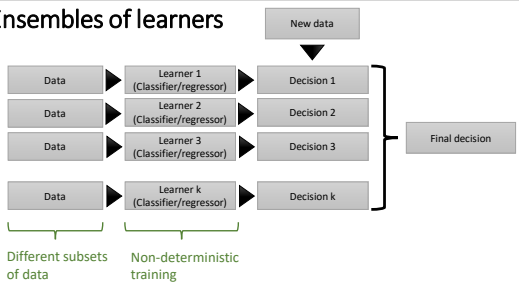
(Because they operate based on randomly selected training dataset. Had the dataset been different, the results would also change!)

Classifiers and regressors should **always** be treated as non-deterministic.

How to be reliable in such a context?
(How to ensure, that we are not just lucky or unlucky in how data are drawn?)

29

Ensembles of learners



The more **diverse** and **accurate** the learners are, the better

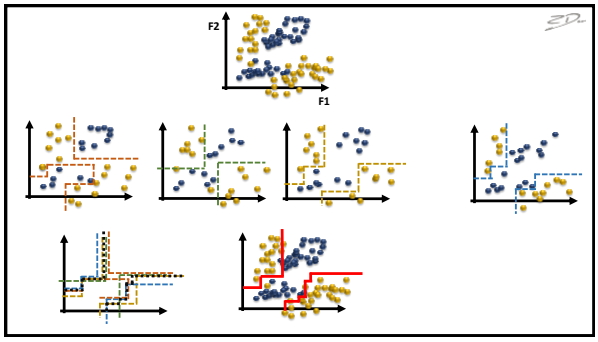
30

Random forest

1. Do **Bagging** (Bootstrap Aggregating) by selecting repeatedly subsets of data using drawing with replacement
2. If dataset consists of many features, sample them also (let different subsets use also subsets of features!)
3. Train simple decision trees based on these subsets (each tree uses different subset)
4. Average responses from many trees

The more **diverse** and **accurate** the learners are, the better

31



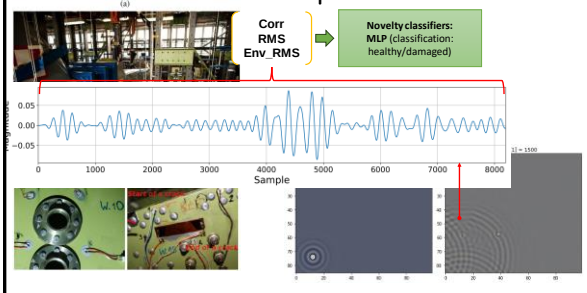
32

Ensembles of learners

In general: Ensembles provide **increased reliability** at the cost of
- **lack of direct control over classification proces,**
- **almost no structural optimization possibilities,** and
- **a lot of required memory and time**

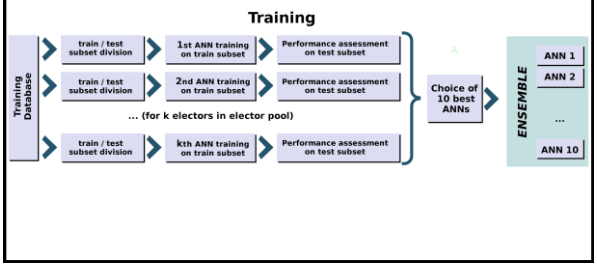
33

Ensembles of learners: Example



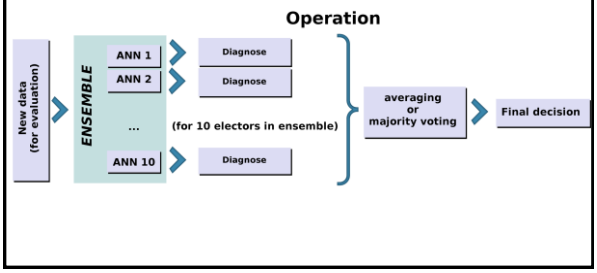
34

Ensembles of learners: Example



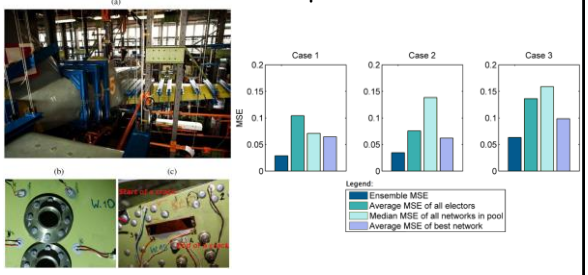
35

Ensembles of learners: Example



36

Ensembles of learners: Example



37

Things to remember:

1. How does a Decision Tree algorithm works and what are its properties?
2. What are the typical engineering sources of data and how can they be processed to extract features?
3. Show scheme of classifier training indicating role of data subsets and highlight first-order and second-order optimization goals
4. What is typical for a hyperparameter optimization problem and what are the approaches to solve it?
5. What are the quality metrics used in classification and regression? Draw and explain confusion matrix.
6. Explain different practical applications for machine learning models. Show examples for identification, state recognition, prediction, process modeling and novelty detection
7. How does ensemble approach work (on the example of random forest algorithm), what are its advantages and disadvantages?

38