Mechatronic Engineering program:
Python for machine learning and data science

Regression

Ziemowit Dworakowski
AGH University of Krakow

1

Example 1: Used car retail (2023)

We have a **7-year old** Opel Astra, with **80 000 km mileage**.
We want to **sell it quickly** with as **high price** as possible.

Initial price

Too low — **We will not earn much**

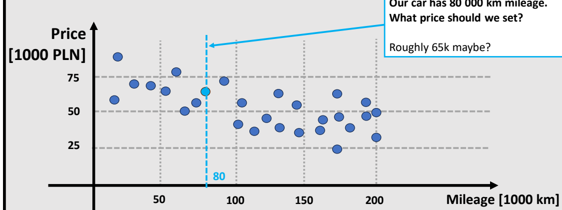
Too high — **Car won't sell quickly**

2

Example 1: Used car retail (2023)

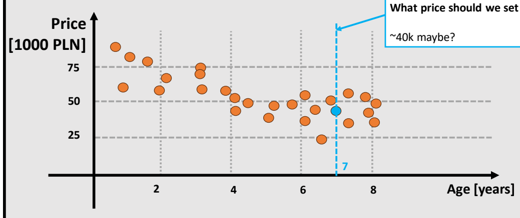
Our car has 80 000 km mileage.
What price should we set?

Roughly 65k maybe?



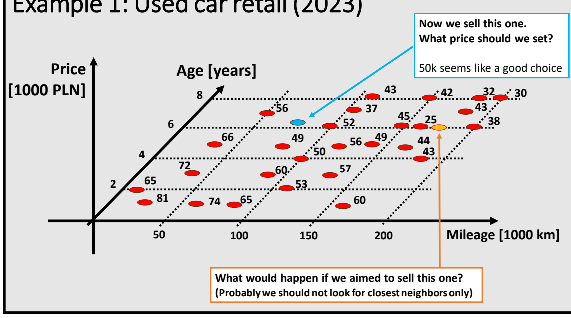
3

Example 1: Used car retail (2023)



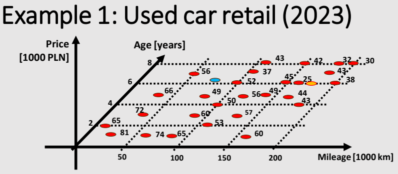
4

Example 1: Used car retail (2023)



5

Example 1: Used car retail (2023)



Of course we still don't see all the contributing factors.

- These cars can have different equipment versions (e.g. manual vs automatic gearbox)
- They can have different levels of general wear
- Prices can be affected by geographical factors

These could be included as further data dimensions

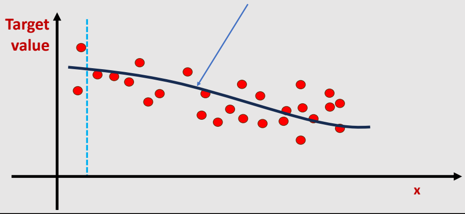
But then we should probably gather much more data to densely fill the feature space

6

Let's recap:

We want to have a model that predict
Target value based on features values.

$f(\boldsymbol{x})$ — *A vector of inputs (Features)*



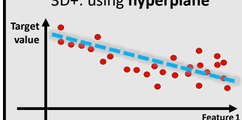
7

Let's recap:

We want to have a model that predict
Target value based on features values and model parameters.

$$f(\boldsymbol{w}, \boldsymbol{x})$$

Linear model
3D+: using **hyperplane**

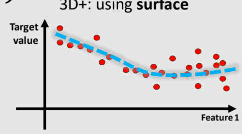


$f(\boldsymbol{w}, \boldsymbol{x}) = \boldsymbol{w}\boldsymbol{x} + b$

$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$

- Features
- Model parameters

Nonlinear model
3D+: using **surface**



We will still have a general $f(\boldsymbol{w}, \boldsymbol{x})$ form, but the relations will be nonlinear now and depending on the model.

8

Regression means finding a **model** that estimates relationship between one data variable and the others

$$y = f(\boldsymbol{w}, \boldsymbol{x})$$

Target value

a.k.a:
Dependent variable
Outcome
Response
(Label)

Model parameters

a.k.a:
Weights
Unknown parameters

(often also denoted as $\boldsymbol{\beta}$)

Features

a.k.a:
Independent variables
Explanatory variables
Inputs
Predictors

9

Regression means finding a **model** that estimates relationship between one data variable and the others

$$y = f(\boldsymbol{w}, \boldsymbol{x})$$

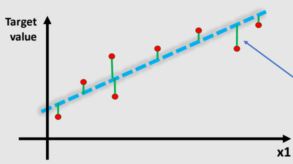
We want to minimize error between known targets $\boldsymbol{Y}$ and targets predicted by model for a known set of input data $\boldsymbol{X}$ by adjusting model parameters $\boldsymbol{w}$

For that we can use least squares minimization:

$$arg\,min_{\boldsymbol{w}} \sum_i (y_i - f(\boldsymbol{w}, \boldsymbol{x}_i))^2$$

10

Linear regression



We want to find such line that sum of squares of these green line segments is as small as possible

Model:

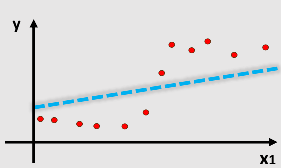
$$y = w_1 x + w_b$$

Model fitting:

$$arg\,min_w \sum_i (y_i - (w_1 x_i + w_b))^2$$

11

Locally weighted regression



A straight line does not really allow to model these data properly…

But what we actually use this line for, is to predict values for particular **x**

So maybe we could use a line model – with a small change of fitting method

12

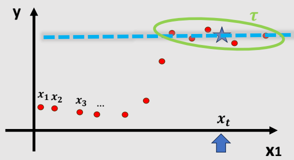
Locally weighted regression

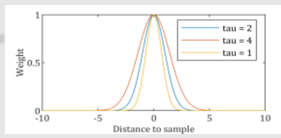
We want to use the same model: $y = w_1 x + w_b$

This time we will fit it separately for each point so that neighboring points will be more important

Given a point $\boldsymbol{x_t}$ assign weights $\boldsymbol{\alpha_i}$ for each data sample $\boldsymbol{x_i}$ where $\boldsymbol{\tau}$ serves as a „width" metaparameter

$$\alpha_i = e^{-\frac{(x_i - x_t)^2}{\tau}}$$



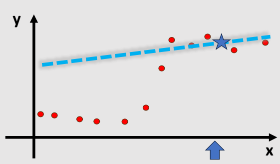


Lets answer what should be the model output (y) for this value of x

$$arg\ min_w \sum_i \alpha_i \cdot (y_i - (w_1 x_i + w_b))^2$$

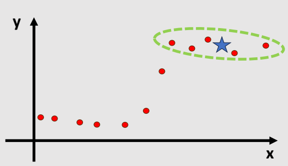
13

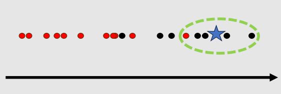
Locally weighted regression is nice, but requires calculation of a regression model each time we need an answer – **it is not viable if our model needs to be fast and memory efficient**



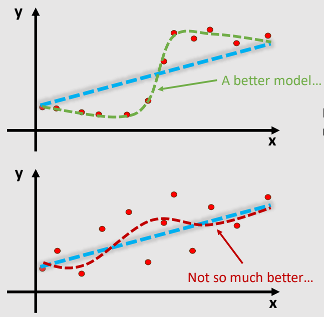
14

Do your remember a kNN classifier? This works using a similar idea! We are ignoring what happens in the entire feature space – we just assign a value or class based on the neighbors of the point of interest…





15



How do we know that we actually need a nonlinear model?

~~We could measure fitting error...~~

We just need to try a nonlinear one and check if it helps...

16

We want to have a model that predict
Target value based on features values and model parameters.

Linear model ← $f(\mathbf{w}, \boldsymbol{x})$ → **Nonlinear model**

3D+: using **hyperplane**

3D+: using **surface**

17

We want to have a model that predict
Target value based on features values and model parameters.

Linear model ← $f(\mathbf{w}, \boldsymbol{x})$ → **Nonlinear model**

3D+: using **hyperplane**

3D+: using **surface**

And now we just need to think how we want to introduce nonlinearities...

We will cover three examples:
- Polynomial model
- SVM
- Neural network

18

Polynomial model

$$f(\boldsymbol{w}, \boldsymbol{x}) = b + \boldsymbol{w}_1 \boldsymbol{x}^1 + \boldsymbol{w}_2 \boldsymbol{x}^2 + \boldsymbol{w}_3 \boldsymbol{x}^3 + \ \ldots$$

Linear model: $b + \boldsymbol{w}_1 \boldsymbol{x}^1$

Quadratic model: $b + \boldsymbol{w}_1 \boldsymbol{x}^1 + \boldsymbol{w}_2 \boldsymbol{x}^2$

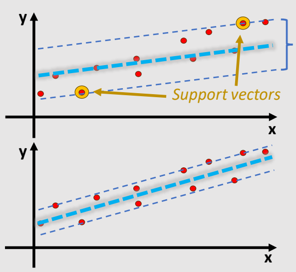
Note that these are vectors of weights, not just single numbers!

Polynomial models work just like linear models, with one exception – we actually need to set up **a metaparameter** – degree of the used polynomial

We do it usually by increasing the degree until the model stops improving significantly

19

Support Vector Machine (SVM)

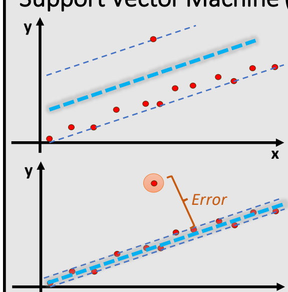


We fit the model so the width of the margin is minimal

(We are no longer interested in point-by-point errors)

20

Support Vector Machine (SVM)

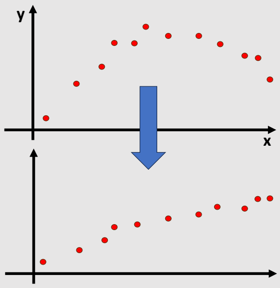


We fit the model so the width of the margin is minimal

We allow a few points to fall outside of margin (we add their errors to the cost function)

21

Support Vector Machine (SVM)



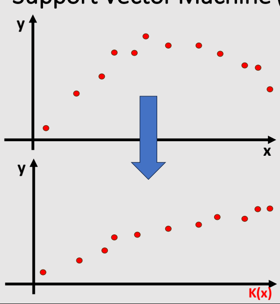
We fit the model so the width of the margin is minimal

We allow a few points to fall outside of margin (we add their errors to the cost function)

We change nonlinear regression into linear one by **morphing the feature space using kernels**

22

Support Vector Machine (SVM) : kernel trick



$$f(\boldsymbol{w}, \boldsymbol{x})$$

$$f(\boldsymbol{w}, K(\boldsymbol{x}))$$

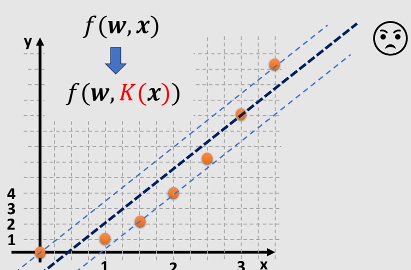
Kernel K maps data nonlinearly into space with higher number of dimensions

23

Support Vector Machine (SVM) : kernel trick

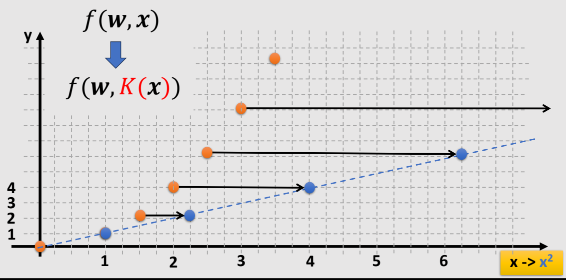
$$f(\boldsymbol{w}, \boldsymbol{x})$$

$$f(\boldsymbol{w}, K(\boldsymbol{x}))$$



24

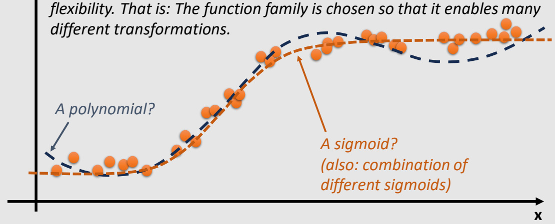
Support Vector Machine (SVM) : kernel trick



25

Support Vector Machine (SVM) : kernel trick

We are not hand-crafting a „proper" function!
We are just allowing the method to figure it out by allowing additional flexibility. That is: The function family is chosen so that it enables many different transformations.

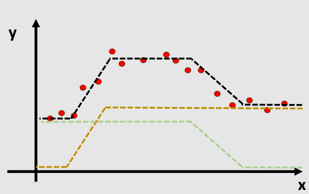


26

Artificial neural network

The idea here is to take a lot of simple nonlinear functions and add them together to get a more complex one

$$f(\boldsymbol{w}, \boldsymbol{x}) = f_1(\boldsymbol{w}_1, \boldsymbol{x}) + f_2(\boldsymbol{w}_2, \boldsymbol{x}) + \cdots + f_3(\boldsymbol{w}_3, \boldsymbol{x})$$



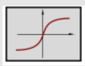
27

Artificial neural network

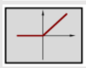
The idea here is to take a lot of simple nonlinear functions and add them together to get a more complex one

$$f(\boldsymbol{w},\boldsymbol{x}) = f_1(\boldsymbol{w}_1,\boldsymbol{x}) + f_2(\boldsymbol{w}_2,\boldsymbol{x}) + \cdots + f_3(\boldsymbol{w}_3,\boldsymbol{x})$$

$$\sigma(\boldsymbol{w}_1 \cdot \boldsymbol{x} + w_{b,1}) + \sigma(\boldsymbol{w}_2 \cdot \boldsymbol{x} + w_{b,2}) + \cdots + \sigma(\boldsymbol{w}_k \cdot \boldsymbol{x} + w_{b,2})$$



Sigmoid $\sigma(x) = \dfrac{e^x}{1+e^x}$



ReLU

… or many others

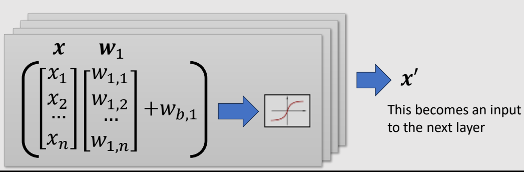
28

Artificial neural network

The idea here is to take a lot of simple nonlinear functions and add them together to get a more complex one

$$f(\boldsymbol{w},\boldsymbol{x}) = f_1(\boldsymbol{w}_1,\boldsymbol{x}) + f_2(\boldsymbol{w}_2,\boldsymbol{x}) + \cdots + f_3(\boldsymbol{w}_3,\boldsymbol{x})$$

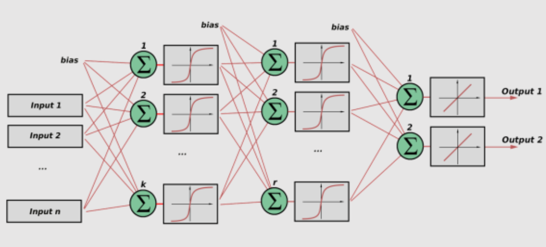
$$\sigma(\boldsymbol{w}_1 \cdot \boldsymbol{x} + w_{b,1}) + \sigma(\boldsymbol{w}_2 \cdot \boldsymbol{x} + w_{b,2}) + \cdots + \sigma(\boldsymbol{w}_k \cdot \boldsymbol{x} + w_{b,2})$$



This becomes an input to the next layer

29

Artificial neural network



30

Model summary

Weighted linear regression
Able to simplify nonlinear regression to linear one, slow, memory-consuming

Linear regression
Simple, straightforward, quick

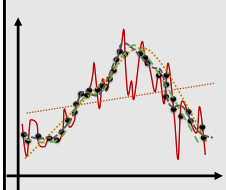
Support Vector Machine
Quick learning, shallow reasoning (requires good features), good scalability, adjustable for outliers

Polynomial fit
Simple, able to tackle simple nonlinearities in data relations

Artificial neural network
Adjustable for any problem (universal), complex configuration, slow training

31

How to set model complexity?



Model complexity (relative to the task)

- Model overfits unless training is stopped early
- Model performs well but contains unnecessary complexity
- „Sweet spot" – model complexity allows to model all relevant data characteristics and generalizes well
- Model is general enough (but not optimal) – General characteristics are modeled, nuances are lost
- Model is too simple

32

How to set model complexity?

We can start from the top down, reducing model complexity until overfitting stops

OR

We can pick complexity based on data: 10 independent samples per adjustable parameter

OR

We can start from the bottom up, increasing model complexity until validation error stops decreasing

Model complexity (relative to the task)

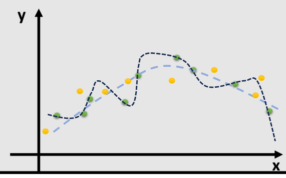
- Model overfits unless training is stopped early
- Model performs well but contains unnecessary complexity
- „Sweet spot" – model complexity allows to model all relevant data characteristics and generalizes well
- Model is general enough (but not optimal) – General characteristics are modeled, nuances are lost
- Model is too simple

33

Overfitting revisited

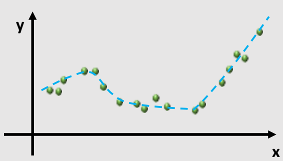
Overfitting means that the model memorizes training samples at the cost of generalization capabilities

We recognize it by looking at the error on an independent subset of data (either validation or testing subset). If it is significantly higher than for training subset – the model overfits.



34

Practical uses: **General interpretation of data**



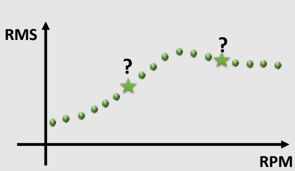
We want to infer a relationship between one (target) feature ***y*** and other features.

We use a labeled dataset and fit the model

From now on, we don't need to measure y any more, we can get it from other features

35

Practical uses: **Data imputation**

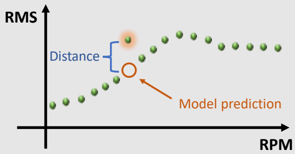


Assume, we have a dataset with some values missing

We can build a regression model to predict missing values

36

Practical uses: novelty detection



Assume, we recorded a measurement and want to check if it „feels OK" (if it is within a *normal range*)

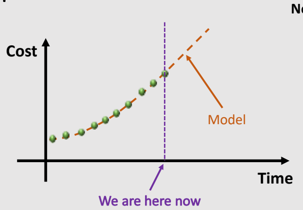
$$x = [x_1 + x_2 + \cdots + x_n]$$

We can fit a regression model to predict one of its elements given others

Now we can check how **distant** is this **prediction** from **reality**. If it is too far we mark data as **anomaly (novelty)**

37

Practical uses: prediction



We can fit the model to data as usual. Now we have a tool to predict future
(kind of)

1. This works **only for a short time** window (a few steps at most)
2. This does **not** allow to predict trend changes!

38

Things to remember:

1. Explain some real-life examples of regression (including new ones, not from the lecture!)
2. Explain generalized regression model y = f(w,x)
3. Explain simple regression methods: linear, weighted, polynomial
4. Explain differences between linear and nonlinear models
5. Explain what overfitting is and how to avoid it (in regression context)
6. Explain how SVM algorithm works (three main distinct features)
7. Explain a kernel trick
8. Explain what happens when model complexity increases for the same problem
9. Explain how to adjust model complexity for a problem
10. Show and describe MLP scheme
11. Explain particular regression uses: data imputation, novelty detection, prediction

39