Exercises

- Different to what announced in the first lecture, exercise results are NOT relevant for earning a bonus for the first exam (if passed)
- ONLY the projects must be successfully completed and presented to earn the bonus
- The exercises will help you to familiarize with the content in depth, implement them in python and get to know the relevant libraries
- Completing the exercises will help to prepare for the exam and for your project

Exercises

You will need

- Python 3.x installed
- Python libraries such as

  jupyter, numpy, scipy,
  pandas, scikit-learn,
  pyod, matplotlib,
  seaborn, plotly

- Sometimes pen and paper for hand-written exercises

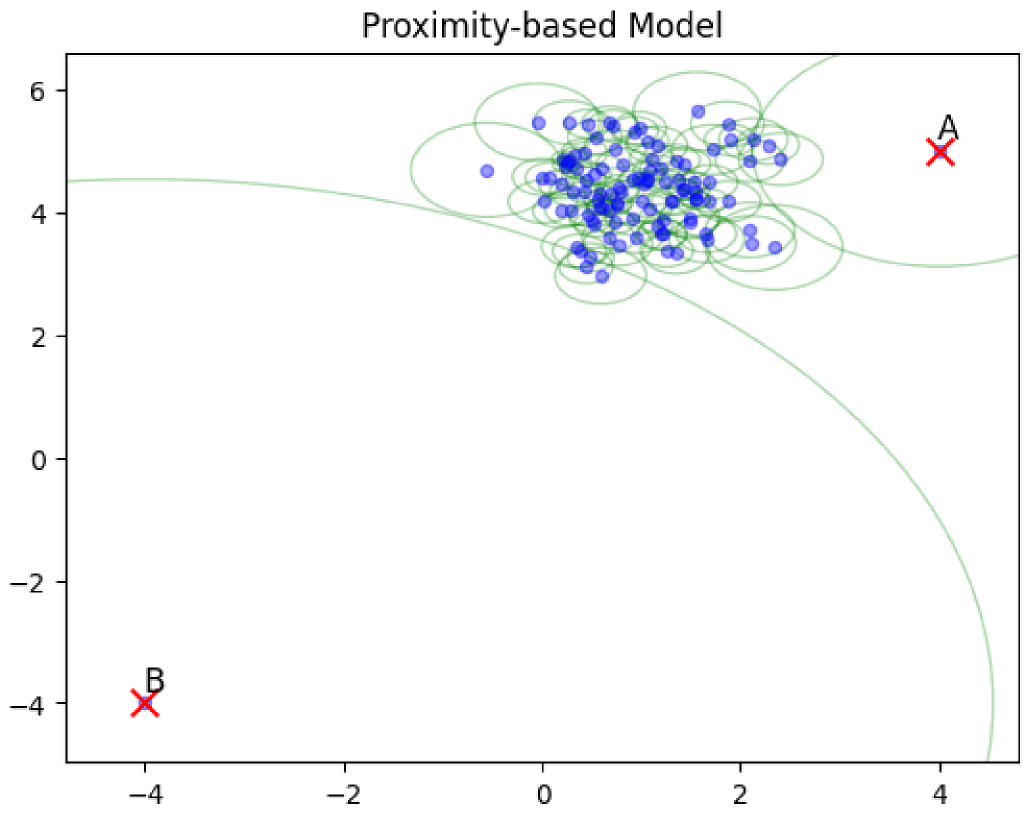
The Data Model is Everything (cont.)

Example 3

Question:

How would you model the following examples, and can you make up a criterion on the “outlierness”?

Density- / proximity-based models, distance to *n* neighbors



The Data Model is Everything (cont.)

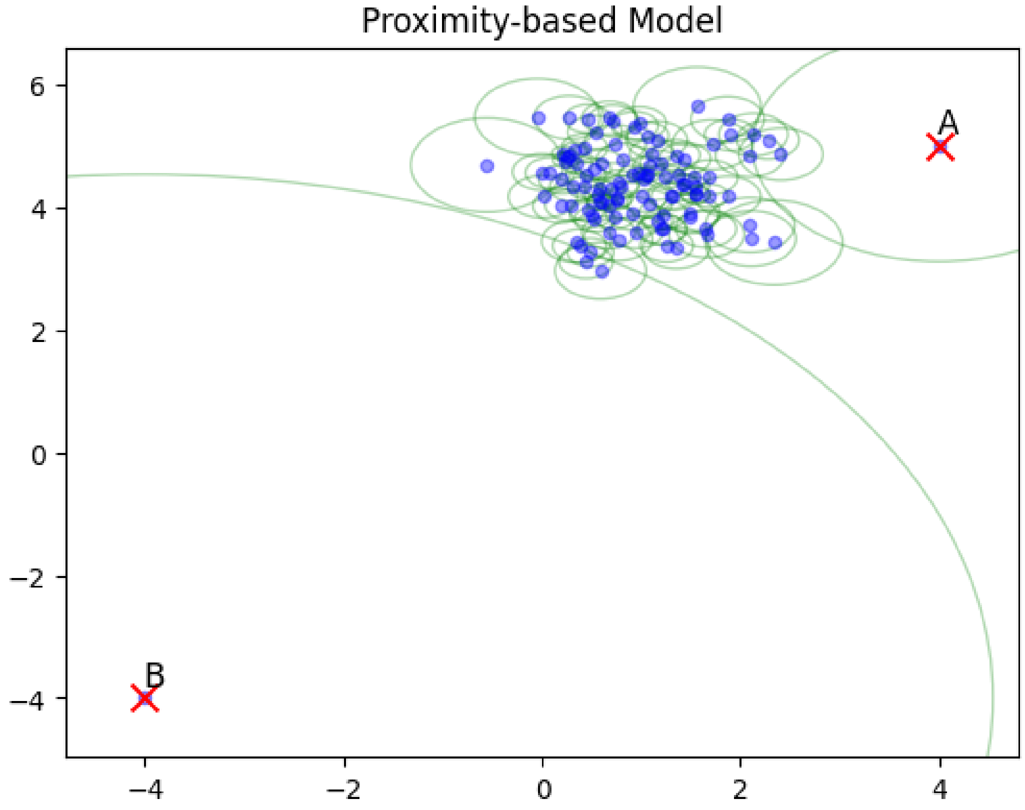
Naive approach:

- Define distance metric e.g., Euclidean

$$d(\boldsymbol{a}, \boldsymbol{b}) = \|\boldsymbol{a} - \boldsymbol{b}\|_2 = \sqrt{\sum_i^d (a_i - b_i)^2}$$

`numpy.linalg.norm`

- For each datapoint go through the dataset
- Create a list of all neighbors sorted by distance
- Get the k-th entry from each list to get the k-nearest neigbor distance



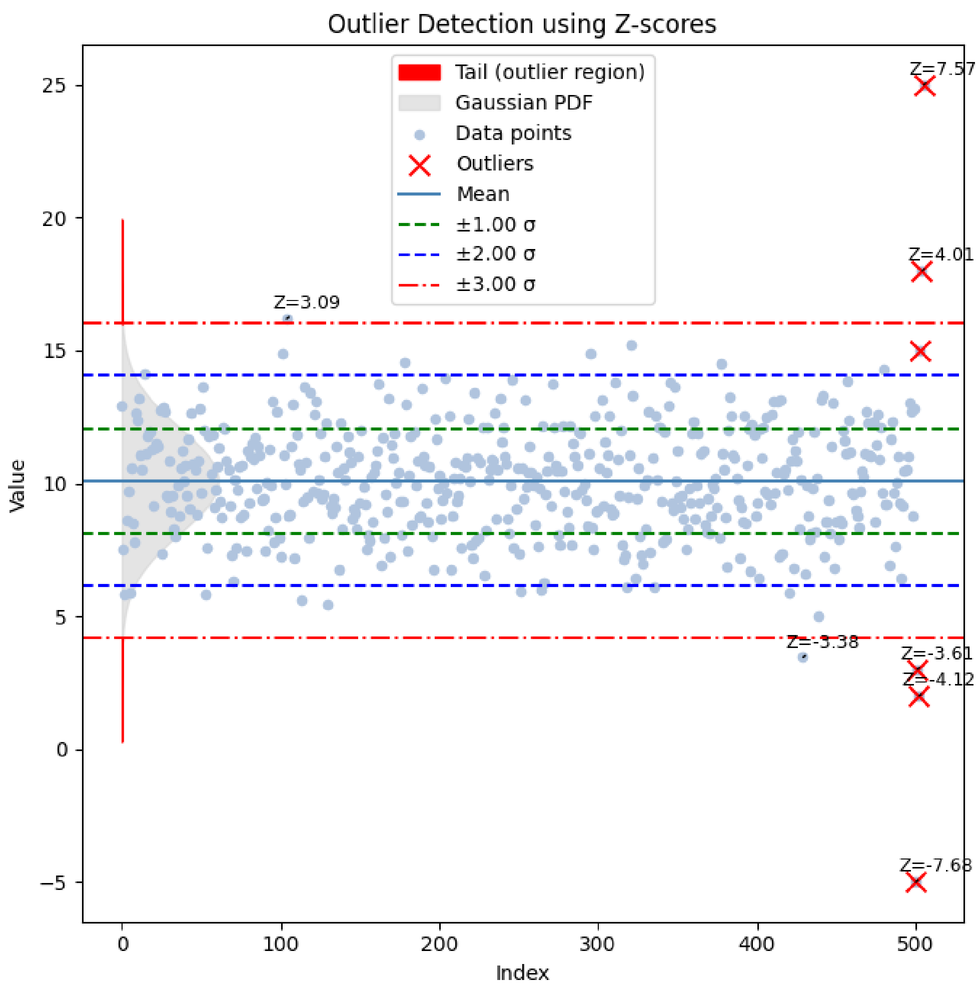
Overview of Anomaly Detection Methods

Z-value Test for Outlier Detection

- Simple model for outlier detection
- One-dimensional data $X_i, \ldots, X_N$ with mean μ and standard deviation σ
- Z-value for a data point X_i:

$$z_i = \frac{|x_i - \mu|}{\sigma}$$

- Z-value denotes the number of standard deviations to mean
- **Implicit assumption: data follows normal distribution**



Z-value Test for Outlier Detection

- “3σ rule-of-thumb”: $z_i \geq 3$ as decision criterion for anomalies:

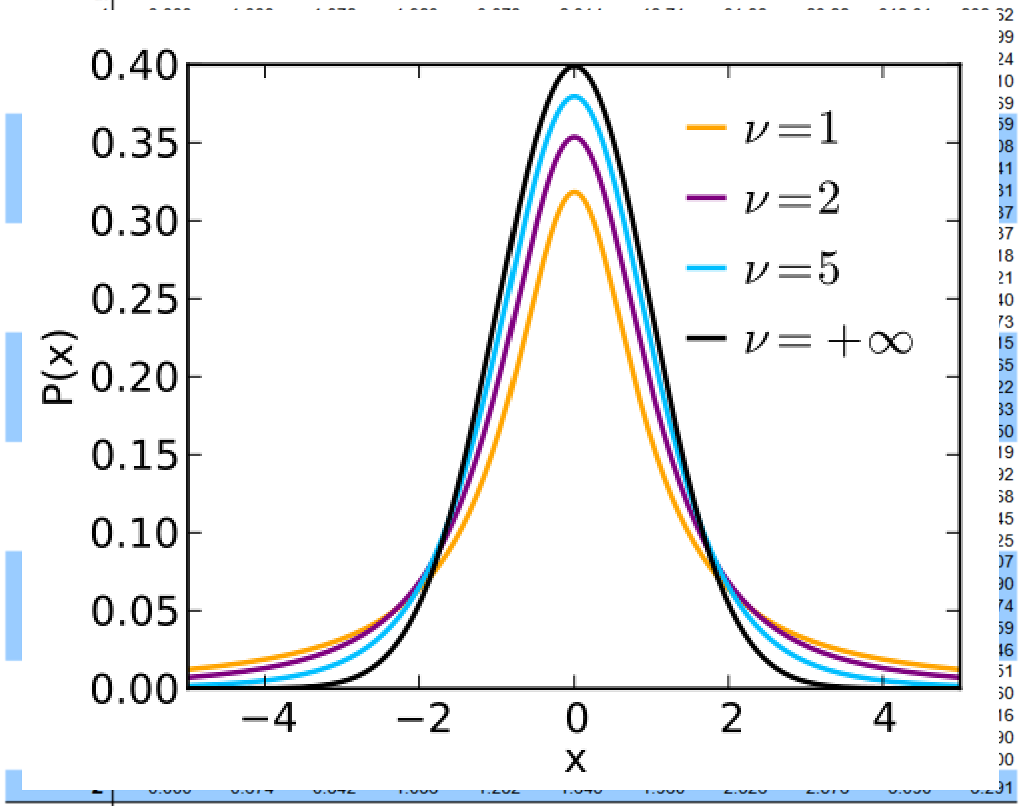
$$P(z < 3) = 0.9973$$

- Typically: μ and σ not explicitly known
  - For enough data (n > 30) assumption of normality
  - For few data interpretation by *Student's t-distribution* and the (absolute of the) *t-value*

$$t_i = \left| \frac{x_i - \mu}{\sigma / \sqrt{n}} \right|$$

for sample size n

unbiased $\sigma = \sqrt{\frac{\sum_i (x_i - \mu)^2}{n-1}}$



t Table

cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	0.50	0.25	0.20	0.15	0.10	0.05	0.025	0.01	0.005	0.001	0.0005
two-tails	1.00	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01	0.002	0.001
df											
	0%	50%	60%	70%	80%	90%	95%	98%	99%	99.8%	99.9%
	Confidence Level										

https://www.sjsu.edu/faculty/gerstman/StatPrimer/t-table.pdf

Example: Z-value Test

Given 100 samples S with estimated

mean: 7.13

std.dev: 1.86

including $x = \{1, 2, 13, 14\} \subset S$.

1. Calculate the Z-values for these points.
2. Decide, which will be labeled as outliers according to the "3σ rule-of-thumb".

Calculate: $z = |x-\mu| / \sigma$

Data point (1):

z-score = 6.13 / 1.86 =

= 3.2957

Data point (2):

z-score = 5.13 / 1.86

= 2.7581

Data point (13):

z-score = 5.87 / 1.86

= 3.1559

Data point (14):

z-score = 6.87 / 1.86

= 3.6935

Outliers: 1, 13, 14 z>=3

Example: Z-value Test

Given 100 samples S with estimated

mean: 7.13

std.dev: 1.86

including $x = \{1, 2, 13, 14\} \subset S$.

1. Calculate the Z-values for these points.
2. Decide, which will be labeled as outliers according to the "3σ rule-of-thumb".

```python
mean = 7.13
std = 1.86
p = [1, 2, 13, 14]

z_val = [abs(x - mean) / std for x in p]
print("z-values", z_val)

# > z-values [3.2956989247311825, 2.758064516129032,
3.155913978494623 5, 3.693548387096774]

out = [p[i] for i, z in enumerate(z_val) if abs(z) > 3]
print("Outliers:", out)

# > Outliers: [1, 13, 14]
```

Example: Z-value Test

Given the following data points:

3, 8, 6, 15, 13, 7

1. Calculate the mean and sample standard deviation.
2. Compute the t-values for each data point.
3. Identify if any points are outliers for a tail probability mass of 0.05.

***t* Table**

cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	0.50	0.25	0.20	0.15	0.10	0.05	0.025	0.01	0.005	0.001	0.0005
two-tails	1.00	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01	0.002	0.001
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	0.000	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	0.000	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	0.000	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.144	4.587

Example: Z-value Test

Given the following data points:

3, 8, 6, 15, 13, 7

1. Calculate the mean and sample standard deviation.
2. Compute the t-values for each data point.
3. Identify if any points are outliers for a tail probability mass of 0.05.

```python
import numpy as np
data = [3, 8, 6, 15, 13, 7]
mean = np.mean(data)
print("Mean:", mean)

# > Mean: 8.666666666666666

std = np.std(data, ddof=1)
print("Sample Standard Deviation:", std)

# > Sample Standard Deviation: 4.501851470969102

n = len(data)
t_val = [(x - mean) / (std / np.sqrt(n)) for x in data]
print("t-values:", t_val)

# > t-values: [-3.083274062967549, -
0.3627381250500547, -1.4509525002200228,
3.446012188022554, 2.357797812857538, -
0.9068453126375141]

t_critical = 2.571 # df = 5 and alpha = 0.05
outliers = [data[i] for i, t in enumerate(t_val) if
abs(t) > t_critical]
print("Outliers:", outliers)

# > Outliers: [3, 15]
```

Probabilistic and Statistical Models

- Data is modeled as closed-form probability distribution
- The parameters of this model are learned
- Key assumption: choice of data distribution
- The likelihood fit of a data point to a generative model is the outlier score

Example:

- Gaussian PDF:

$$g(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

- „Fit“ the parameters μ, σ to the data x_i

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \,, \sigma = \sqrt{\frac{\sum_i (x_i - \mu)^2}{n-1}}$$

Probabilistic and Statistical Models (cont.)

Example (cont.):

Approach:

1. Choose Gaussian (Normal) Distribution Probability Density Function
2. "Fit" parameters to dataset
3. Calculate the likelihood fit for a data point
4. Convert the likelihood fit to an outlier score (e.g. negative log likelihood)

Example:

1. Gaussian PDF:
$$g(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$
2. „Fit" the parameters μ, σ to the data x_i
$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \, , \sigma = \sqrt{\frac{\sum_i (x_i-\mu)^2}{n-1}}$$
3. Likelihood for $\hat{x}$: $g(\hat{x};\mu,\sigma)$
4. $\mathrm{NLL}(\hat{x}) = -\log(g(\hat{x};\mu,\sigma))$

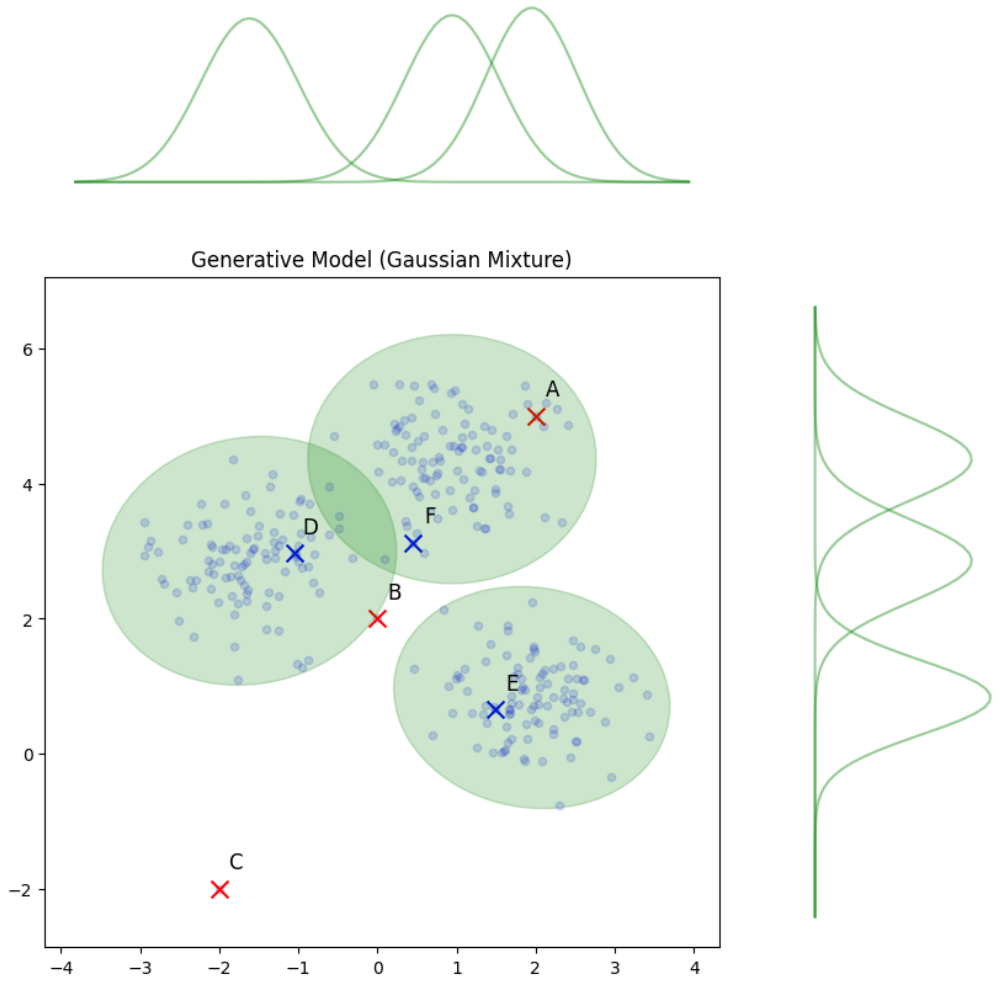
Probabilistic and Statistical Models (cont.)

Gaussian Mixture Models

- Probabilistic model
- Assumption: Data is generated from a mixture of Gaussians
- Data is described as combination of K Gaussians

$$p(x; \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^{K} \pi_k g(x; \mu_k, \Sigma_k)$$

- Parameters are learned via EM algorithm



Reynolds, Douglas A. "Gaussian mixture models." Encyclopedia of biometrics 741.659-663 (2009).

Gaussian Mixture Modelling Example:
http://localhost:8888/notebooks/AD02-S94-GMM_Example.ipynb