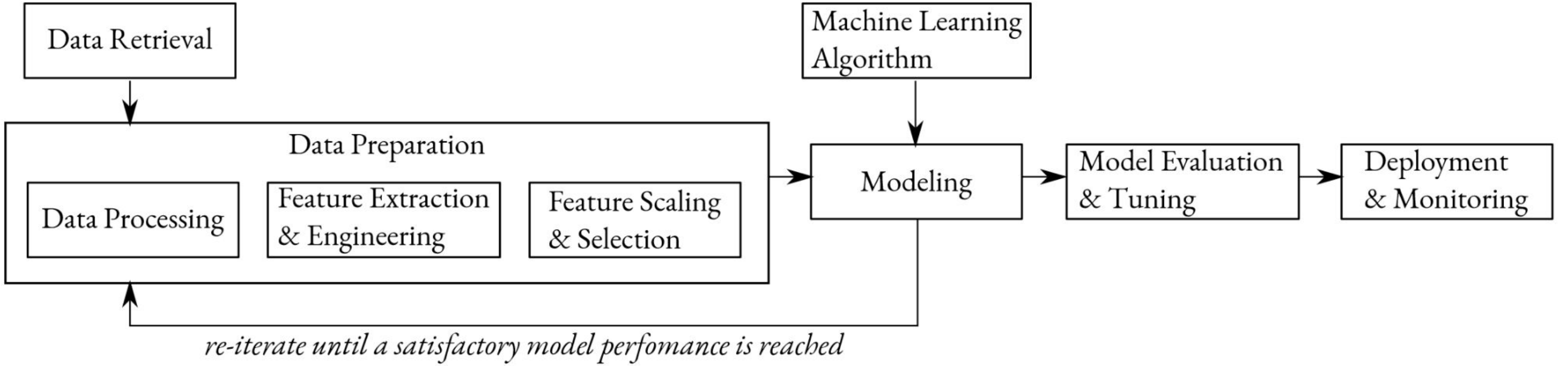
Central European Institute of Technology
BRNO | CZECH REPUBLIC

Data Science Practicum

(Lecture 8, 6.11.)

Denisa Šrámková

Data Preparation



Exercise:

https://github.com/simecek/dspracticum2023/blob/main/lesson08/ds_practicum_ex_astronauts_pandas.ipynb

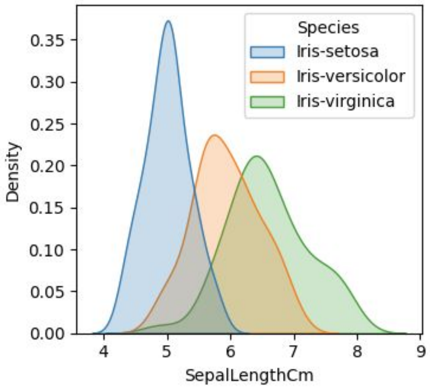
Data Preparation - lecture outline

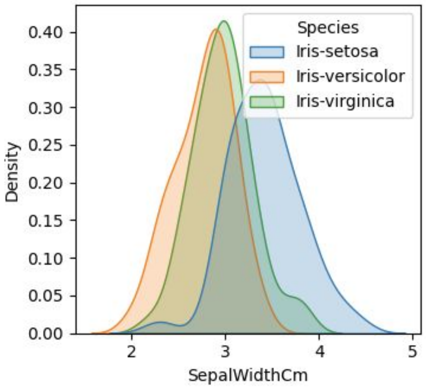
0. Data exploration
1. Data cleaning
2. Feature manipulation
3. Feature selection
4. Dataset sampling
5. Dimensionality reduction
6. Dataset splitting

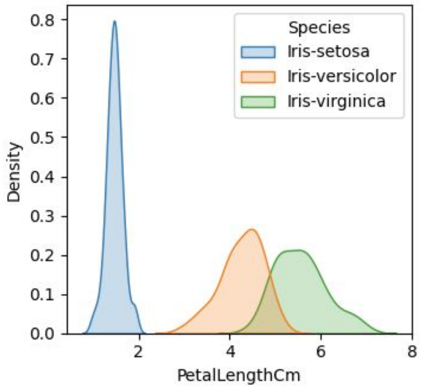
Exercise:
https://github.com/simecek/dspracticum2023/blob/main/lesson08/ds_practicum_ex_astronauts_pandas.ipynb

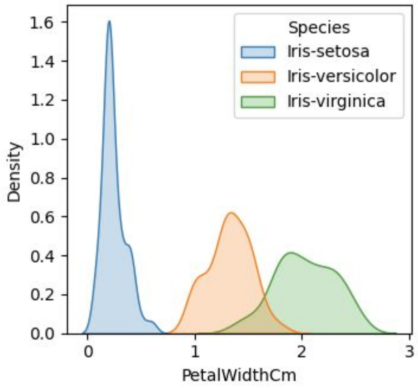
0. Data exploration

- get to know your data (statistical properties of the features)
- make visualization to get more insights









Feature types

Numerical:

Type	Description	Example
interval	Values are arranged in order, and differences between them are meaningful, but there is no inherent starting point, and ration are meaningless.	
ration	The values are an extension of interval values - an inherent zero starting point is included.	

Feature types

Numerical:

Type	Description	Example
interval	Values are arranged in order, and differences between them are meaningful, but there is no inherent starting point, and ration are meaningless.	temperature in Celsius, calendar dates 
ration	The values are an extension of interval values - an inherent zero starting point is included.	

Feature types

Numerical:

Type	Description	Example
interval	Values are arranged in order, and differences between them are meaningful, but there is no inherent starting point, and ration are meaningless.	temperature in Celsius, calendar dates 
ration	The values are an extension of interval values - an inherent zero starting point is included.	age, RGB 

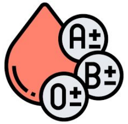
Feature types

Categorical:

Type	Description	Example
nominal	Values consist of explicit categories, but their ordering does not make sense.	
ordinal	Values are arranged in some order, but the difference between values can’t be determined, or is meaningless.	

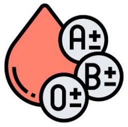
Feature types

Categorical:

Type	Description	Example
nominal	Values consist of explicit categories, but their ordering does not make sense.	blood types, colors 
ordinal	Values are arranged in some order, but the difference between values can’t be determined, or is meaningless.	

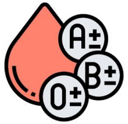
Feature types

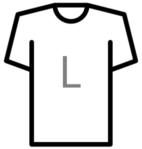
Categorical:

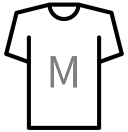
Type	Description	Example
nominal	Values consist of explicit categories, but their ordering does not make sense.	blood types, colors 
ordinal	Values are arranged in some order, but the difference between values can't be determined, or is meaningless.	clothes sizes, rating 

Feature types

Categorical:

Type	Description	Example
nominal	Values consist of explicit categories, but their ordering does not make sense.	blood types, colors 
ordinal	Values are arranged in some order, but the difference between values can’t be determined, or is meaningless.	clothes sizes, rating 

XS < S < M < L , however what should 

 - 

 equal to?

1. Data cleaning

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	None	30	10
poppy	red	60	6	666
iris	purple	60	3	15

. . .

1. Data cleaning

A) Missing values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	None	30	10
poppy	red	60	6	666
iris	purple	60	3	15

1. Data cleaning

A) Missing values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	None	30	10
poppy	red	60	6	666
iris	purple	60	3	15

Solutions:

1. Leave as it is

1. Data cleaning

A) Missing values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
~~rose~~	~~red~~	~~None~~	~~30~~	~~10~~
poppy	red	60	6	666
iris	purple	60	3	15

Solutions:

1. Leave as it is,   2. Eliminate entire row

1. Data cleaning

A) Missing values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	None	30	10
poppy	red	60	6	666
iris	purple	60	3	15

Solutions:

1. Leave as it is, 2. Eliminate entire row (column)

1. Data cleaning

avg(60,60,60) = 60

A) Missing values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	None	30	10
poppy	red	60	6	666
iris	purple	60	3	15

Solutions:

1. Leave as it is, 2. Eliminate entire row (column), 3. Fill it in (interpolation)

1. Data cleaning

A) Missing values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	60	30	10
poppy	red	60	6	666
iris	purple	60	3	15

Solutions:

1. Leave as it is, 2. Eliminate entire row (column), 3. Fill it in (interpolation)

1. Data cleaning

B) Inconsistent values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	60	30	10
poppy	red	60	6	666
iris	purple	60	3	15

1. Data cleaning

B) Inconsistent values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	60	30	10
~~poppy~~	~~red~~	~~60~~	~~6~~	~~666~~
iris	purple	60	3	15

1. Data cleaning

birthdate
24.3.1997
26

age

B) Inconsistent values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	60	30	10
~~poppy~~	~~red~~	~~60~~	~~6~~	~~666~~
iris	purple	60	3	15

1. Data cleaning

C) Duplicate values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	60	30	10
iris	purple	60	3	15

1. Data cleaning

C) Duplicate values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	60	30	10
~~iris~~	~~purple~~	~~60~~	~~3~~	~~15~~

1. Data cleaning (switch to exercise)

C) Duplicate values:

flower_name	*petal_colour*	*max_stem_length_cm*	*num_petals*	*lifespan_years*
iris	purple	60	3	15
rose	red	60	30	10
~~iris~~	~~purple~~	~~60~~	~~3~~	~~15~~

2. Feature manipulation

Categorical features encoding: for nominal features

2. Feature manipulation

Categorical features encoding: for nominal features

One-Hot encoding

sample_id	*hobby*
111	hiking
112	knitting
113	hiking
114	reading books

2. Feature manipulation

Categorical features encoding: for nominal features

One-Hot encoding

sample_id	*hobby*
111	hiking
112	knitting
113	hiking
114	reading books

→

sample_id	*hiking*	*knitting*	*book_reading*
111	1	0	0
112	0	1	0
113	1	0	0
114	0	0	1

2. Feature manipulation

Categorical features encoding: for nominal features

Binning

2. Feature manipulation

Categorical features encoding: for nominal features

Binning

id	*city_of_birth*
21	New York
22	Prague
23	Brno
24	Funchal
25	Svit
66	Toronto

2. Feature manipulation

Categorical features encoding: for nominal features

Binning

id	*city_of_birth*
21	New York
22	Prague
23	Brno
24	Funchal
25	Svit
66	Toronto

id	*city_of_birth*
21	USA
22	Czech Republic
23	Czech Republic
24	Madeira
25	Slovakia
66	Canada

2. Feature manipulation

Categorical features encoding: for nominal features

Binning

id	*city_of_birth*
21	New York
22	Prague
23	Brno
24	Funchal
25	Svit
66	Toronto

or →

id	*city_of_birth*
21	South America
22	Europe
23	Europe
24	Europe
25	Europe
66	South America

2. Feature manipulation

Categorical features encoding: for nominal features

Binning

id	*city_of_birth*
21	New York
22	Prague
23	Brno
24	Funchal
25	Svit
66	Toronto

or

id	*city_of_birth*
21	South America
22	Europe
23	Europe
24	Europe
25	Europe
66	South America

One-Hot Encoding

id	*south_america*	*europe*
21	1	0
22	0	1
23	0	1
24	0	1
25	0	1
66	1	0

2. Feature manipulation

Categorical features encoding: for ordinal features

2. Feature manipulation

Categorical features encoding: for ordinal features

Simply mapping to a set of integers

id	*rating*
42	excellent
43	good
44	neutral
45	not well
46	terrible

→

id	*rating*
42	5
43	4
44	3
45	2
46	1

2. Feature manipulation *(scaling)*

Numerical features encoding:

2. Feature manipulation *(scaling)*

Numerical features encoding:

Normalization (Min-Max scaling)

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

=> rescales to range (0, 1)

2. Feature manipulation *(scaling)*

Numerical features encoding:

Normalization (Min-Max scaling)

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

=> rescales to range (0, 1)

Standardization

$$X' = \frac{X - \mu}{\sigma}$$

mean of feature values

standard deviation

2. Feature manipulation *(scaling)*

Numerical features encoding:

Normalization (Min-Max scaling)

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

=> rescales to range (0, 1)

Standardization

mean of feature values

$$X' = \frac{X - \mu}{\sigma}$$

standard deviation

3. Feature selection

Multicollinearity = problem, when some features are strongly dependent on each other.

3. Feature selection

Multicollinearity = problem, when some features are strongly dependent on each other.

Identifying such features:

	0	1	2	3	4	5	6	7	8	9
0	1	0.347533	0.398948	0.455743	0.0729144	-0.233402	-0.731222	0.477978	-0.442621	0.0151847
1	0.347533	1	-0.284056	0.571003	-0.285483	0.38248	-0.362842	0.642578	0.252556	0.190047
2	0.398948	-0.284056	1	-0.523649	0.152937	-0.139176	-0.0928948	0.0162655	-0.434016	-0.383585
3	0.455743	0.571003	-0.523649	1	-0.225343	-0.227577	-0.481548	0.473286	0.279258	0.44665
4	0.0729144	-0.285483	0.152937	-0.225343	1	-0.104438	-0.147477	-0.523283	-0.614603	-0.189916
5	-0.233402	0.38248	-0.139176	-0.227577	-0.104438	1	-0.0302517	0.41764	0.205851	0.0950844
6	-0.731222	-0.362842	-0.0928948	-0.481548	-0.147477	-0.0302517	1	-0.49444	0.381407	-0.353652
7	0.477978	0.642578	0.0162655	0.473286	-0.523283	0.41764	-0.49444	1	0.375873	0.417863
8	-0.442621	0.252556	-0.434016	0.279258	-0.614603	0.205851	0.381407	0.375873	1	0.150421
9	0.0151847	0.190047	-0.383585	0.44665	-0.189916	0.0950844	-0.353652	0.417863	0.150421	1

3. Feature selection (switch to exercise)

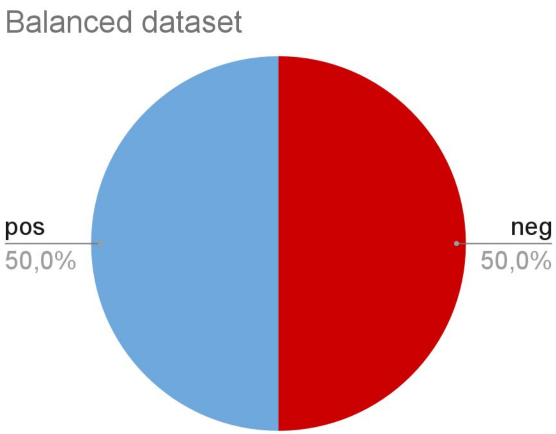
Multicollinearity = problem, when some features are strongly dependent on each other.

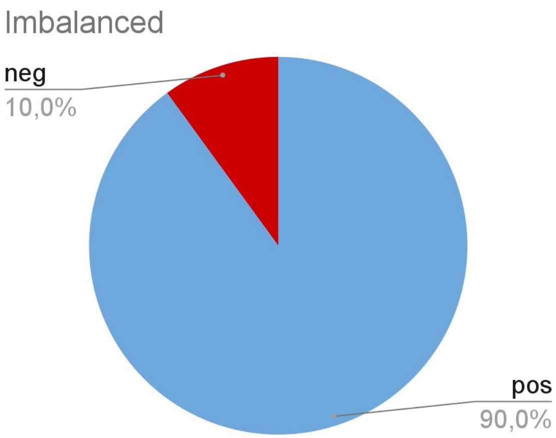
Identifying such features:

	0	1	2	3	4	5	6	7	8	9
0	1	0.347533	0.398948	0.455743	0.0729144	-0.233402	-0.731222	0.477978	-0.442621	0.0151847
1	0.347533	1	-0.284056	0.571003	-0.285483	0.38248	-0.362842	0.642578	0.252556	0.190047
2	0.398948	-0.284056	1	-0.523649	0.152937	-0.139176	-0.0928948	0.0162655	-0.434016	-0.383585
3	0.455743	0.571003	-0.523649	1	-0.225343	-0.227577	-0.481548	0.473286	0.279258	0.44665
4	0.0729144	-0.285483	0.152937	-0.225343	1	-0.104438	-0.147477	-0.523283	-0.614603	-0.189916
5	-0.233402	0.38248	-0.139176	-0.227577	-0.104438	1	-0.0302517	0.41764	0.205851	0.0950844
6	-0.731222	-0.362842	-0.0928948	-0.481548	-0.147477	-0.0302517	1	-0.49444	0.381407	-0.353652
7	0.477978	0.642578	0.0162655	0.473286	-0.523283	0.41764	-0.49444	1	0.375873	0.417863
8	-0.442621	0.252556	-0.434016	0.279258	-0.614603	0.205851	0.381407	0.375873	1	0.150421
9	0.0151847	0.190047	-0.383585	0.44665	-0.189916	0.0950844	-0.353652	0.417863	0.150421	1

4. Dataset sampling

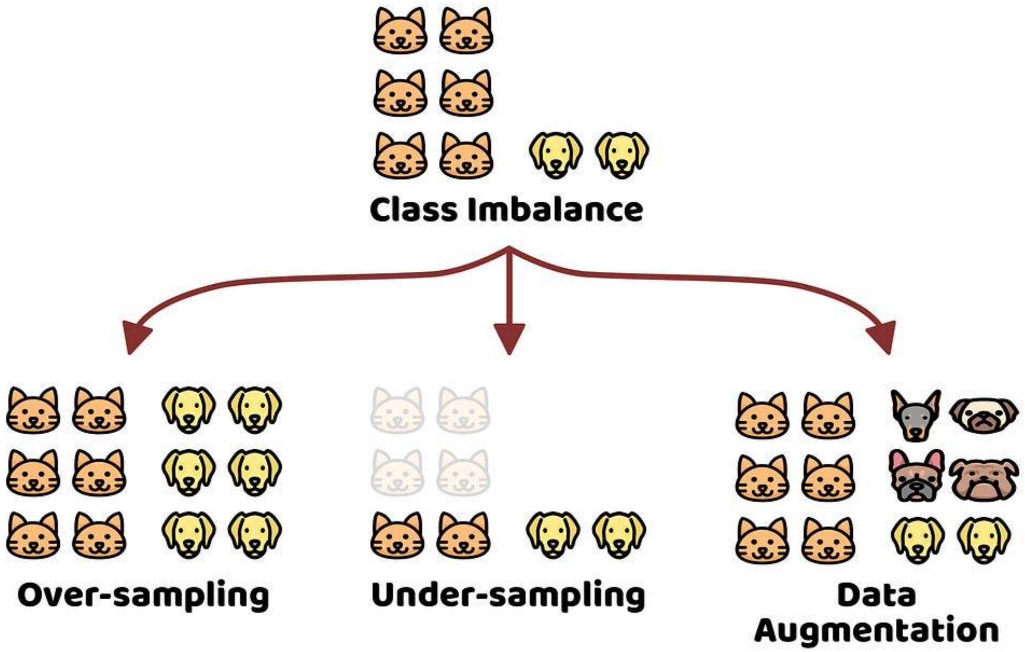
Unbalanced dataset:





4. Dataset sampling

Unbalanced dataset:

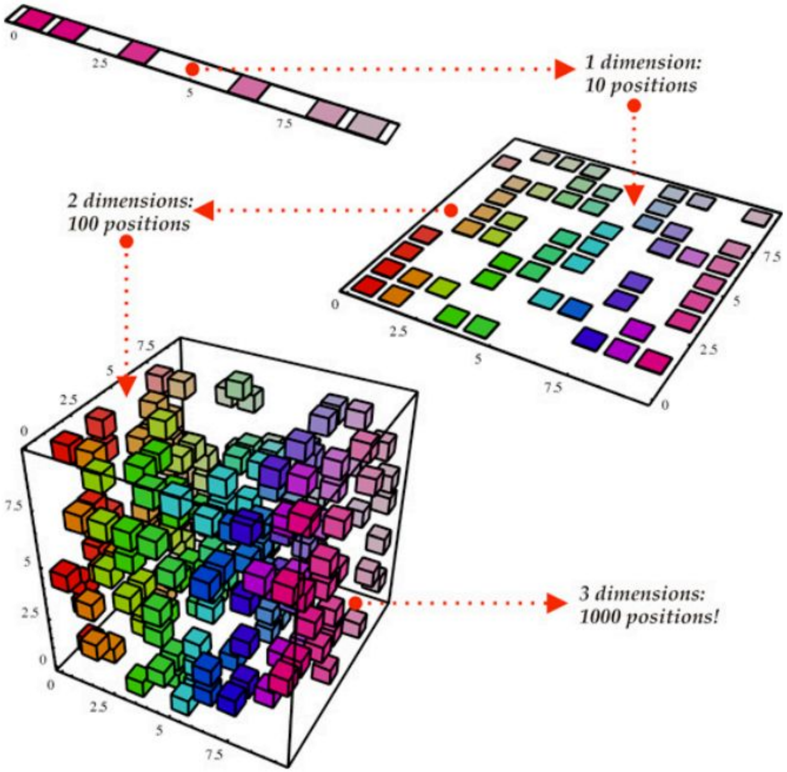


https://towardsdatascience.com/class-imbalance-random-sampling-and-data-augmentation-with-imbalanced-learn-63f3a92ef04a

5. Dimensionality reduction

PCA:

https://www.youtube.com/watch?v=HMOI_lkzW08



6. Dataset splitting

Train:

Validation:

Test:

6. Dataset splitting

Train: used to build the model

Validation:

Test:

6. Dataset splitting

Train: used to build the model

Validation: to improve hyperparameters

Test:

6. Dataset splitting

Train: used to build the model

Validation: to improve hyperparameters

Test: to test the hypothesis of the model (not used until the model is trained and its hyperparameters are decided)

6. Dataset splitting

Train: used to build the model

Validation: to improve hyperparameters

Test: to test the hypothesis of the model (not used until the model is trained and its hyperparameters are decided)

Split ration 80% train : 20% test

- validation size depends on number of hyperparameters (taken from train)

6. Dataset splitting (switch to exercise)

Train: used to build the model

Validation: to improve hyperparameters

Test: to test the hypothesis of the model (not used until the model is trained and its hyperparameters are decided)

Split ration 80% train : 20% test

- validation size depends on number of hyperparameters (taken from train)

Homework

1) Choose some dataset from Kaggle and try to explore it - get as much insights into the dataset as possible (write a report about it, visualize some statistical properties of its features, answer interesting questions, …)
2) Send the link to your solution on GitHub through https://forms.gle/weiXmtYqJ3hfsuJB6