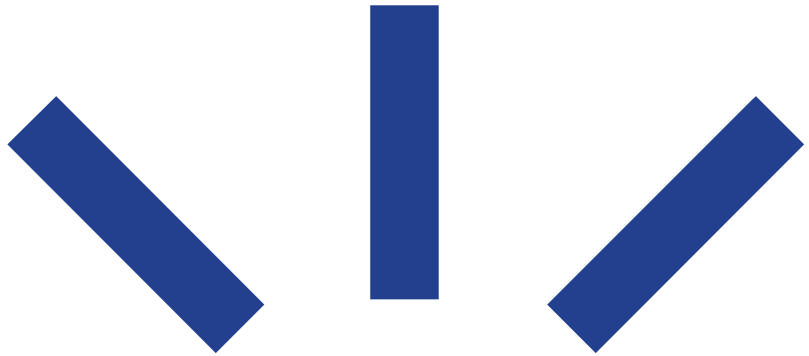
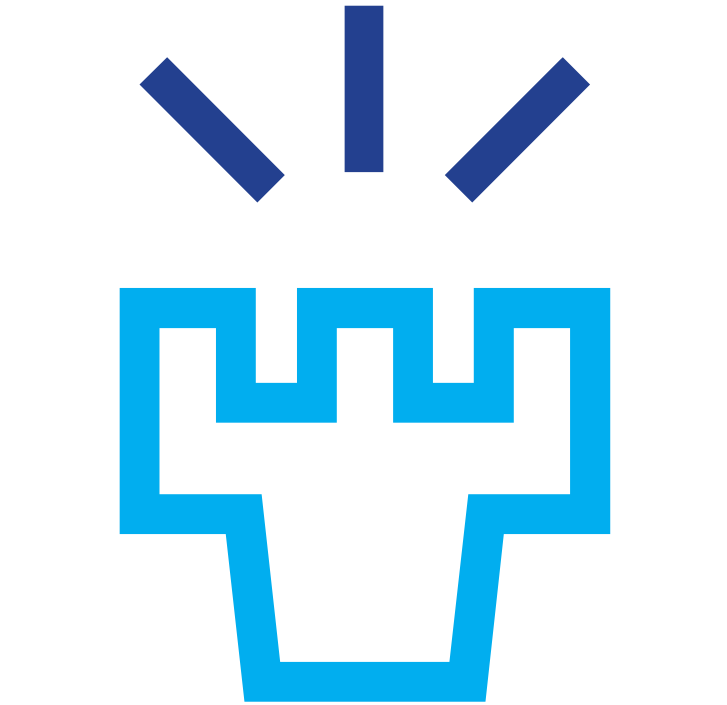


521156S

TOWARDS DATA MINING
Merging signals, data sheets



Topics

- Merging data sources
- Sampling
- Balancing

Merging Data

Motivation

- **Data coming from multiple sources**
  - Different sensors
  - Open data
  - Questionnaires
  - …
- **Data needs to be integrated to work together**
- **Problems?**

Motivation

- **2 Cases**

1. One phenomenon / event measured using several sensors
   - The aim of merging, add more columns (=variables) to data table
   - All sensors need to have the same sampling frequency
   - For instance, combining open weather data and traffic data
   - For instance, combining data from mobile phone sensors and smartwatch sensors
2. Two or more separate phenomenon / events measured using one or more sensors
   - The aim of merging, add more rows (=observations) to data table
   - Sensors need to have the same sampling frequency, units, similar data gathering protocol
   - For instance, two separate data gathering sessions merged to one data table

- **Following slides includes problems and challenges related to these cases**

Formats

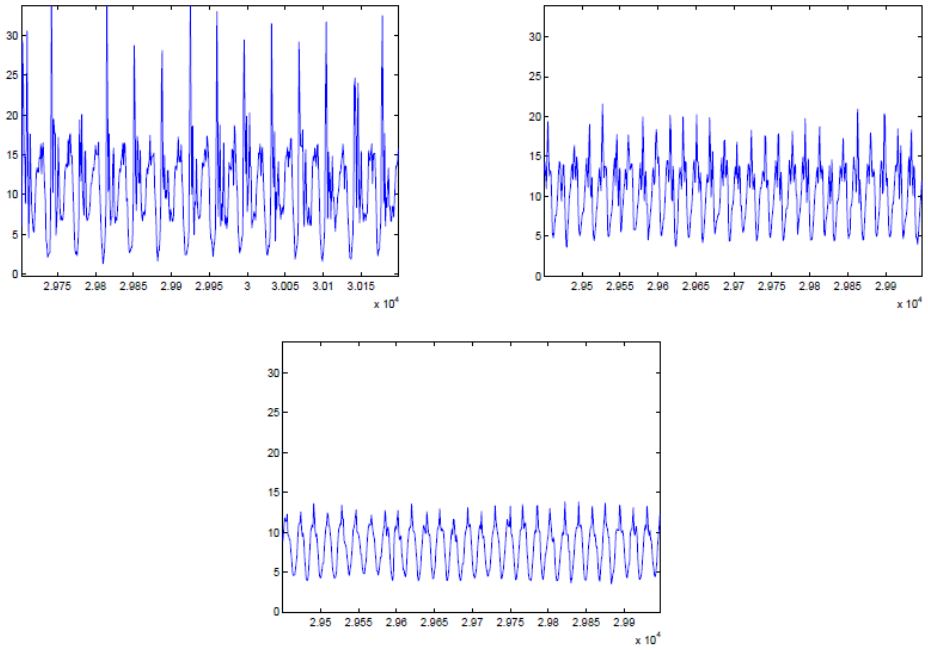
- **Data coming from different sensors may have different data format**
  - Csv, txt, …
  - Tabular separated, comma, …
  - Header, not header
- **Cultural differences**
  - Decimal comma or point
- **Different units**
  - Centimeter vs. inch
  - Gravity vs. m/s^2
- **Different timestamps**
  - Unix time stamp vs. date
  - Timestamps need to be syncronized

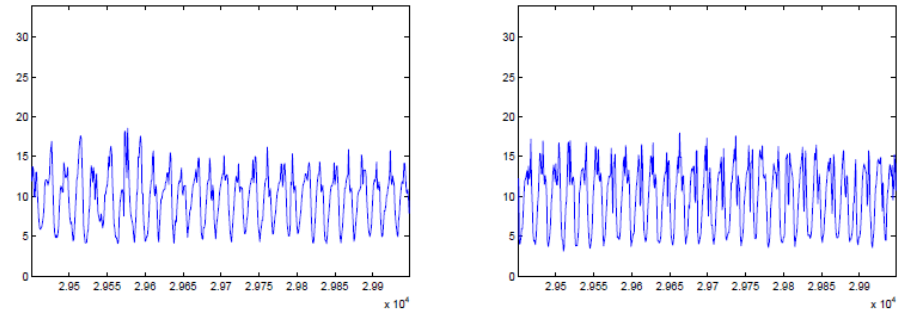
Data collection plan

- **Read the data gathering plan!**
- Same label, different meaning
  - Class label: walking = walking at flat surface
  - Class label: walking = walking at flat surface + walking at stairs
- Sensor placement
  - Different sensor placement causes different signal values
- Different conditions
  - Winter vs. summer
  - Flat surface vs. hills
  - Laboratory vs. outdoor
  - Controlled vs. non-controlled
- Differences in study subjects
  - Children vs. elderly
- Different sampling rates
- It is possible that data collection plans of two data sets are so different that they cannot be combined

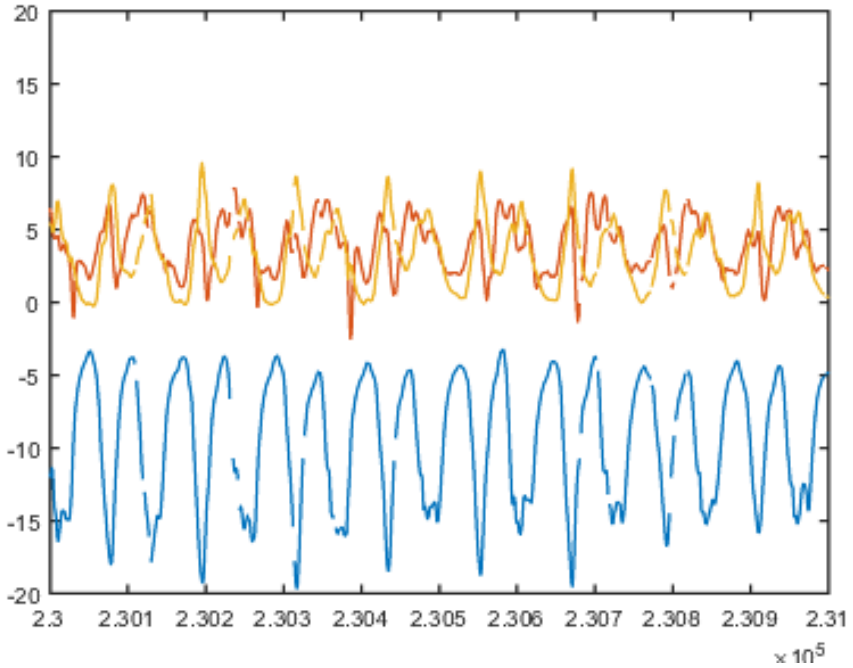
Sensor position

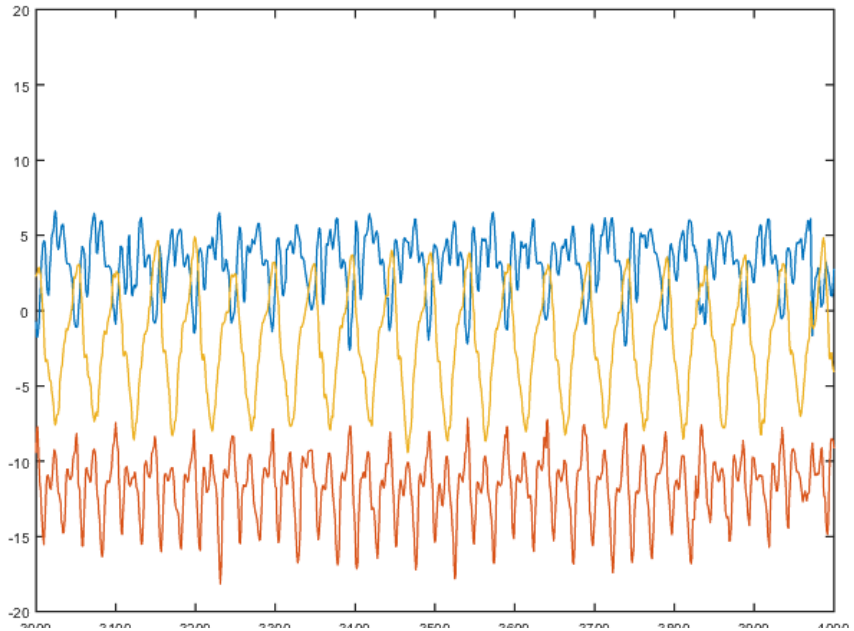
- Walking signal from different body positions

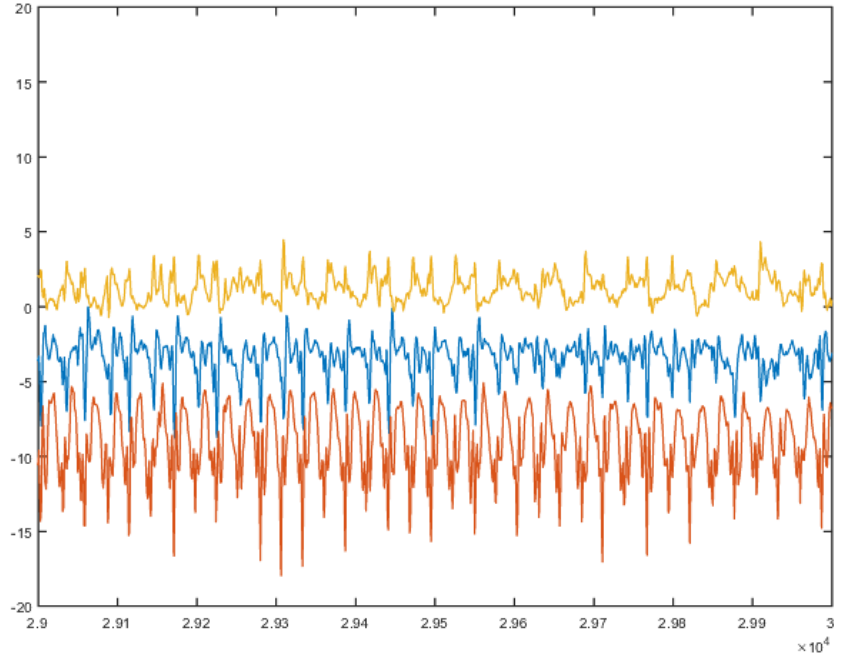




Different sensor

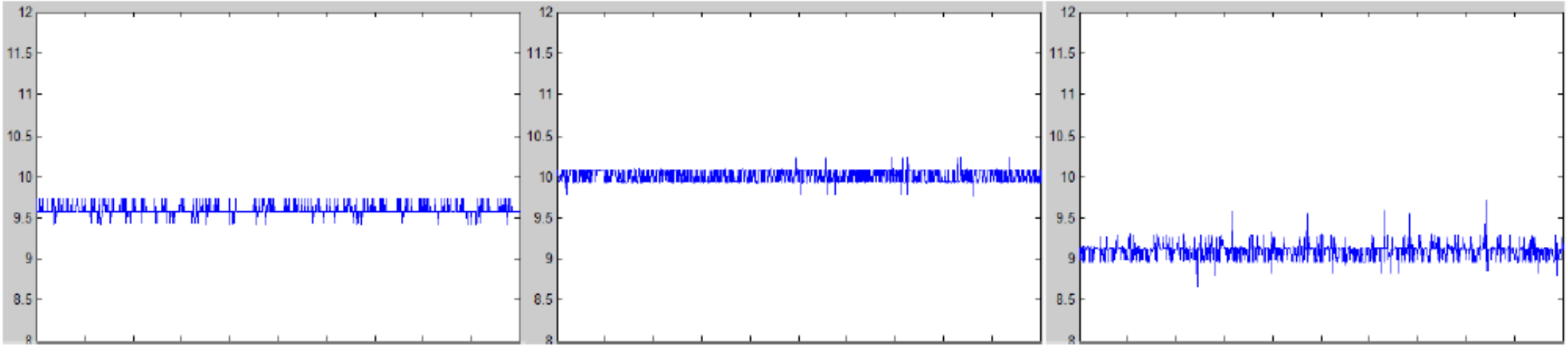






Differences between sensors

- **Sensors are not identical though they have the same brand and model**
  - Calibration differences



Different sampling rates

- **Measuring of different phenomenons require different sampling rate**
  - Voice normally measured using sampling rate 44100 Hz, but there is no point using the same sampling rate to measure for instance body temperature
- **Measurement units designed for different purposes have different sampling rates**
  - Trade-off between battery and sampling frequency
- **Problems arises when measurements from several sensors with different sampling rate are combined**

Downsampling

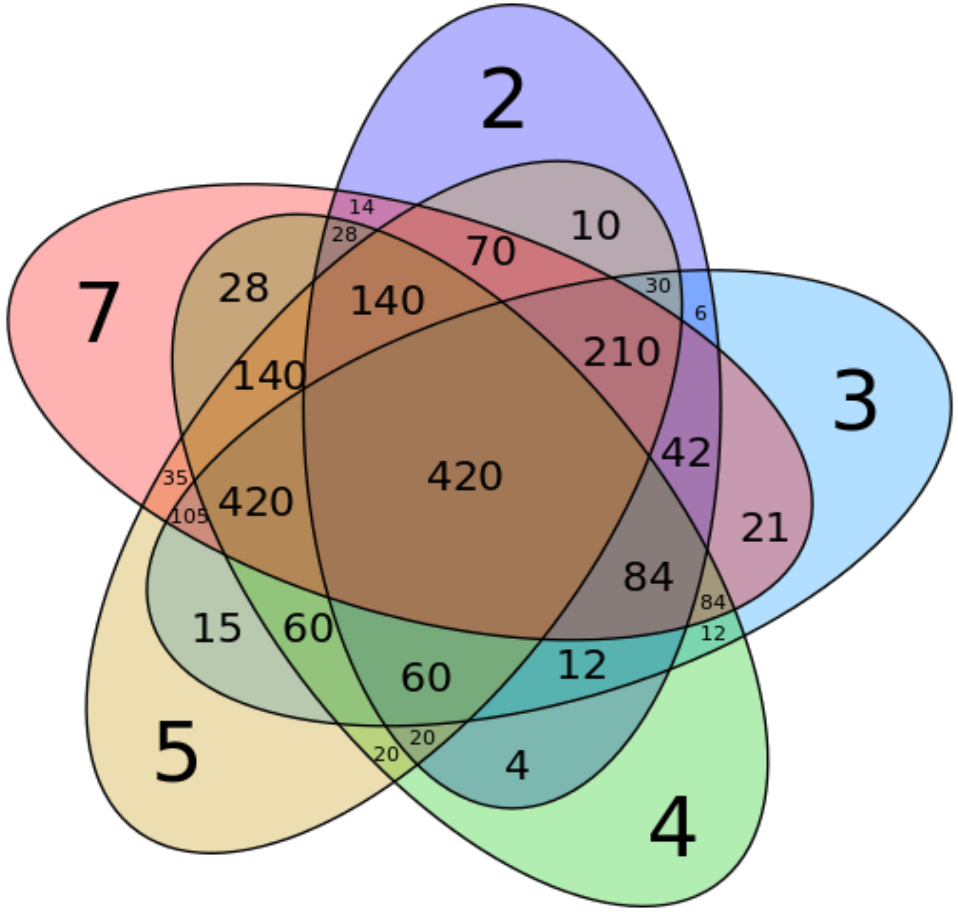
- **Data from multiple sources and multiple sampling rate**
  - Calculate the greatest common divider (GCD) X
  - X is the new sampling rate for all signals

Example:
Source 1: 100Hz
Source 2: 50Hz

GCD(100, 50)=50
-> 50Hz new sampling rate
Source1: take every other sample
Source 2: OK

- **Problem: amount of data decreases -> information will be lost**

Oversampling



- **Data from multiple sources and multiple sampling rate**
  - Calculate the least common multiple (LCM) X
  - X is the new sampling rate for all signals

Example:
Source 1: 100Hz
Source 2: 50Hz

LCM(100, 50)=100
-> 100Hz new sampling rate
Source1: OK
Source 2: Add new sample between two adjecent samples

- **Problem: The amount of data increases -> requires more time to analyze**


Sampling

How to deal with situation where there is not enough data or there is too much data?

- **Not enough data**
  - Reliable models cannot be trained
  - How to get more?
- **Too much data**
  - Not enough processing capacity
  - How to speed up calculations?
- **Biased data set**
  - Not enough data from all classes
  - Too much data from some classes

Too much data

- Simple random sample with replacement (SRSWR)
- Simple random sample without replacement (SRSWOR)
- Balanced sample

SRSWOR

- **Simple random sample without replacement (SRSWOR)**
  - D original data set, the size of the original data set N, D = {s_1, …, s_N}
  - The size of desired data set |D_new| = m < N
- **SRSWOR:** draw *m* samples from D
- Every sample of D can have only one copy in D_new as once sample s_i is drawn from D, it is not replaced
- The probability of drawing any sample from D is 1/*N* and all examples have equal chance to be sampled

SRSWR

- **Simple random sample with replacement (SRSWR)**
  - D original data set, the size of the original data set N, D = {s_1, …, s_N}
  - The size of desired data set |D_new| = m < N
- **SRSWR :** draw *m* samples from D
- Every sample of D can have multiple copies in D_new as once sample s_i is drawn from D, it is replaced
- All examples have equal chance to be sampled

Balanced sample

- **Balanced sample:** The target data set is forced to have a certain composition according to a predefined criterion
  - For instance, in the 2 class case we want that 90% of the samples are from class 1 and 10% from class 2
  - Sampling can be for instance based on SRSWOR or SRSWR
  - Especially good for imbalanced data sets

Not enough data

- Simple random sample with replacement (SRSWR)
- Noise injection
- Artificial data
- SMOTE

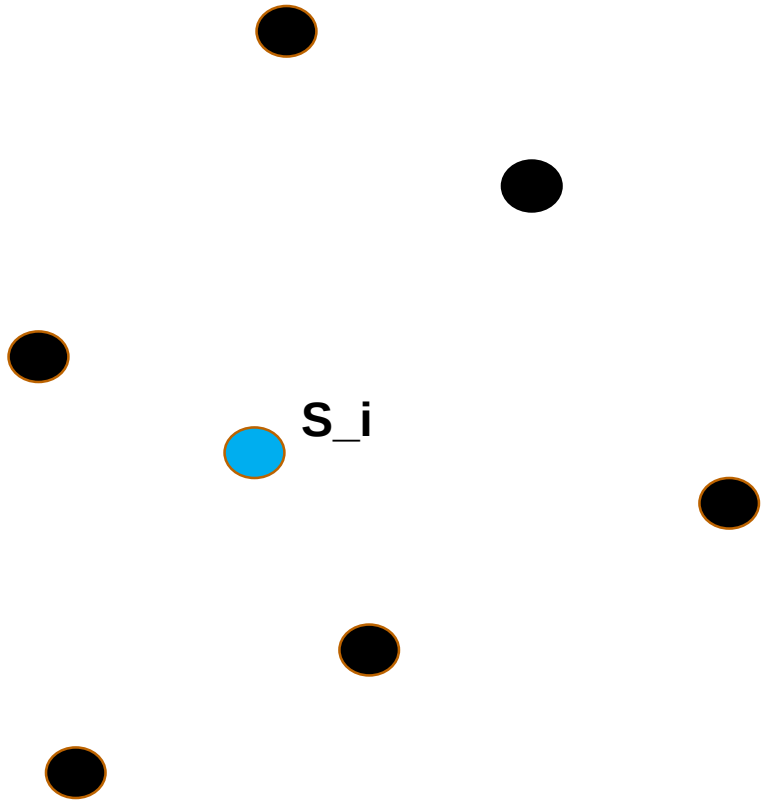
Artificial data

- **Artificial / synthetic data**
  - D original data set, the size of the original data set N, D = {s_1, …, s_N}
  - The size of desired data set |D_new| = m > N
- **Create population by describing original data set D using some parameters such as σ and μ**
  - Each class needs to be described using own parameters
  - Distribution needs to be known
- **New data set is created by drawing m samples from population**

Noise injection

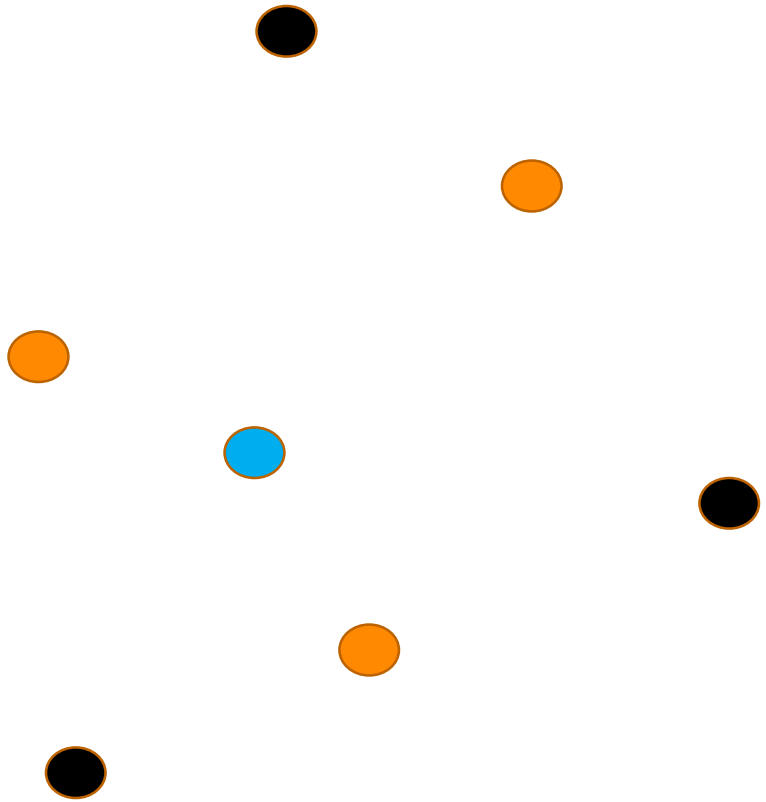
- **Noise injection**
  - D original data set, the size of the original data set N, D = {s_1, …, s_N}
  - The size of desired data set |D_new| = m > N
- **Idea is to expand the area cover by training data by injecting noise to data samples**
- **Make multiple copies of instances s_i, add white noise to copies**
  - The amount of noise depends from data set parametres such as σ and μ
- **In theory, enables training of more general models**

SMOTE



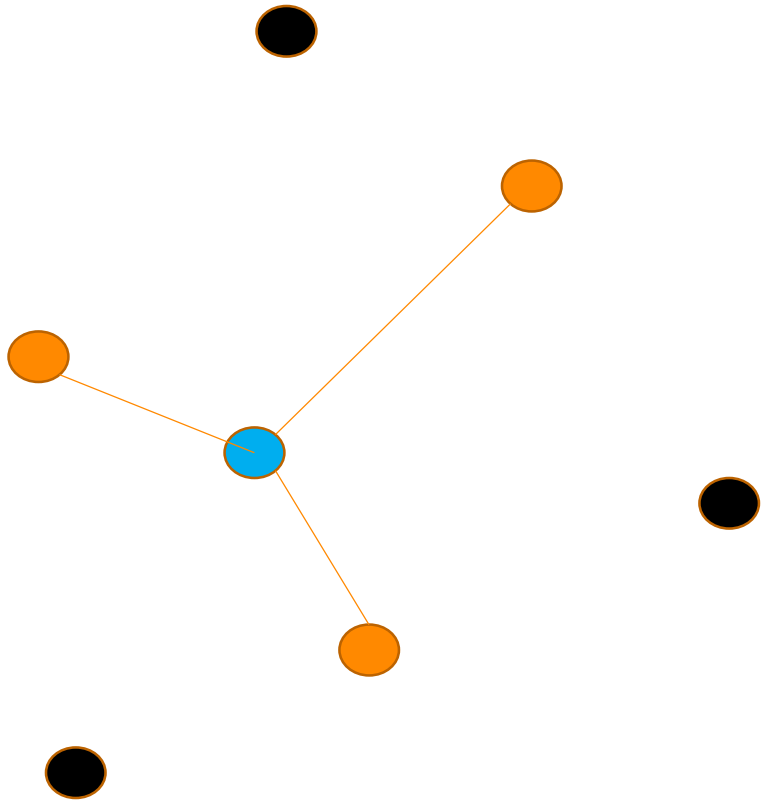
- **SMOTE**
  - The size of the original data set N, D = {s_1, …, s_N}
  - The size of desired data set |D_new| = m > N
- **Select data sample s_i**
  - Find k nearest neighbours (in feature space)
  - Take vector between k nearest neighbours and data sample s_i
  - Multiply vector by random number between 0 and 1
  - Add this to s_i
  - Add new sample to D_new

SMOTE



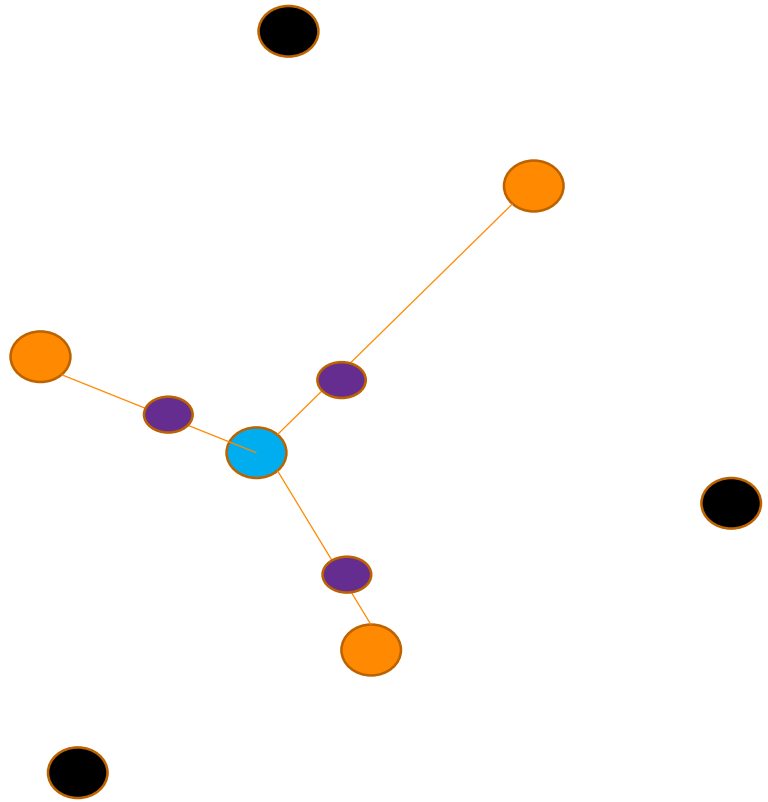
- **SMOTE**
  - The size of the original data set N, D = {s_1, …, s_N}
  - The size of desired data set |D_new| = m > N
- **Select data sample s_i**
  - Find k nearest neighbours (in feature space)
  - Take vector between k nearest neighbours and data sample s_i
  - Multiply vector by random number between 0 and 1
  - Add this to s_i
  - Add new sample to D_new

SMOTE



- **SMOTE**
  - The size of the original data set N, D = {s_1, …, s_N}
  - The size of desired data set |D_new| = m > N
- **Select data sample s_i**
  - Find k nearest neighbours (in feature space)
  - Take vector between k nearest neighbours and data sample s_i
  - Multiply vector by random number between 0 and 1
  - Add this to s_i
  - Add new sample to D_new

SMOTE



- **SMOTE**
  - The size of the original data set N, D = {s_1, …, s_N}
  - The size of desired data set |D_new| = m > N
- **Select data sample s_i**
  - Find k nearest neighbours (in feature space)
  - Take vector between k nearest neighbours and data sample s_i
  - Multiply vector by random number between 0 and 1
  - Add this to s_i
  - Add new sample to D_new

Biased data

– **Some classes have too much data and / or some classes do not have enough data**

- Apply explained methods class-wise
- SRSWOR or SRSWR for classes containing too many observations
- SRSWR, Smote of artificial data to classes that do not have enough observations

Balancing

Imbalanced data

- **Problem: Data set is imbalanced, not equal amount of data from each class**
  - Often it is not easy to get same amount of data from each class, the studied phenomenon can be rare
  - Example 1: 90% of the data from healthy persons and 10% from sick
  - Example 2: Fraudulent transactions, majority of transactions are non-fraud
  - Imbalance can lead to learning wrong things
- **Easy to obtain results that look good but in reality are not**
- **Imbalanced data sets require special methods**

Tactics with imbalanced data sets

- Collect more data
- Try different classification methods
- Give weights to observations
- Use sampling to generate more data to classes that do not have much observations
- Create artificial data
- Try new perspective
- Change performance metrics

Performance metrics

- **Performance matrix that work well with balanced data sets, do not necessarily work well with imbalanced data sets**
  - Results can be misleading
  - Good results do not mean that model is good
- **Often one performance metric is not enough**

Accuracy

- **Accuracy = Correctly classified instances / all instances**
- **The most commonly used performance metric**

Confusion matrix:

		Predicted	
		Positive	Negative
Correct	Positive	True positive (TP)	False negative (FN)
	Negative	False positive (FP)	True negative (TN)

Example:
2 class problem: 300 persons, 275 without cancer and 25 with cancer
Our model classified correctly 275 instances ->
acc = 275/300 = 91,7%

Accuracy paradox

- Accuracy can be misleading
- It may be desirable to select a model with a lower accuracy because it has a greater predictive power on the problem
  - Accuracy != prediction power
- Especially with imbalanced data, prediction power should be estimated using some other metric than accuracy

Confusion matrix

		Predicted	
		Positive	Negative
Correct	Positive	True positive (TP)	False negative (FN)
	Negative	False positive (FP)	True negative (TN)

Cancer example

- **Accuracy 91,7%**

		Predicted	
Correct		Positive	Negative
	Positive	20	5
	Negative	20	255

- **Accuracy 91,7%**

		Predicted	
Correct		Positive	Negative
	Positive	0	25
	Negative	0	275

- **Accuracy 75,0%**

		Predicted	
Correct		Positive	Negative
	Positive	25	0
	Negative	75	200

Balanced accuracy

- **Balanced accuracy = (TP/(TP+FN) + TN/(TN+FP))/2**

- **Balanced accuracy 86,4%**

		Predicted	
		Positive	Negative
Correct	Positive	20	5
	Negative	20	255

- **Balanced accuracy 50,0%**

		Predicted	
		Positive	Negative
Correct	Positive	0	25
	Negative	0	275

- **Balanced accuracy 86,4%**

		Predicted	
		Positive	Negative
Correct	Positive	25	0
	Negative	75	200

Precision

- Precision= TP/(TP+FP)

- Precision= 50,0%

		Predicted	
Correct		Positive	Negative
	Positive	20	5
	Negative	20	255

- Precision =0,0%

		Predicted	
Correct		Positive	Negative
	Positive	0	25
	Negative	0	275

- Precision =25,0%

		Predicted	
Correct		Positive	Negative
	Positive	25	0
	Negative	75	200

Recall aka sensitivity

- Recall = TP/(TP+FN)

- Recall =80,0%

		Predicted	
		Positive	Negative
Correct	Positive	20	5
	Negative	20	255

- Recall= 0,0%

		Predicted	
		Positive	Negative
Correct	Positive	0	25
	Negative	0	275

- Recall =100,0%

		Predicted	
		Positive	Negative
Correct	Positive	25	0
	Negative	75	200

Specificity

- Specificity = TN/(TN+FP)

- Specificity= 92,7%

		Predicted	
Correct		Positive	Negative
	Positive	20	5
	Negative	20	255

- Specificity =100,0%

		Predicted	
Correct		Positive	Negative
	Positive	0	25
	Negative	0	275

- Specificity =72,7%

		Predicted	
Correct		Positive	Negative
	Positive	25	0
	Negative	75	200

F1 score

- **F1 = 2*((precision*recall)/(precision+recall))**
  - Tends to measure the balance between precison and recall

- **F1 =61,5%**

		Predicted	
		Positive	Negative
Correct	Positive	20	5
	Negative	20	255

- **F1 =0,0%**

		Predicted	
		Positive	Negative
Correct	Positive	0	25
	Negative	0	275

- **F1 =40,0%**

		Predicted	
		Positive	Negative
Correct	Positive	25	0
	Negative	75	200

Cohen's kappa

- κ = (p_0 – p_e) / (1-p_e)
  - p_0 = p_11 + p_22
  - p_e = p_.1*p_1. + p_.2*p_2.
- p_x is percentage units, not number of observations

- κ = 57,1%

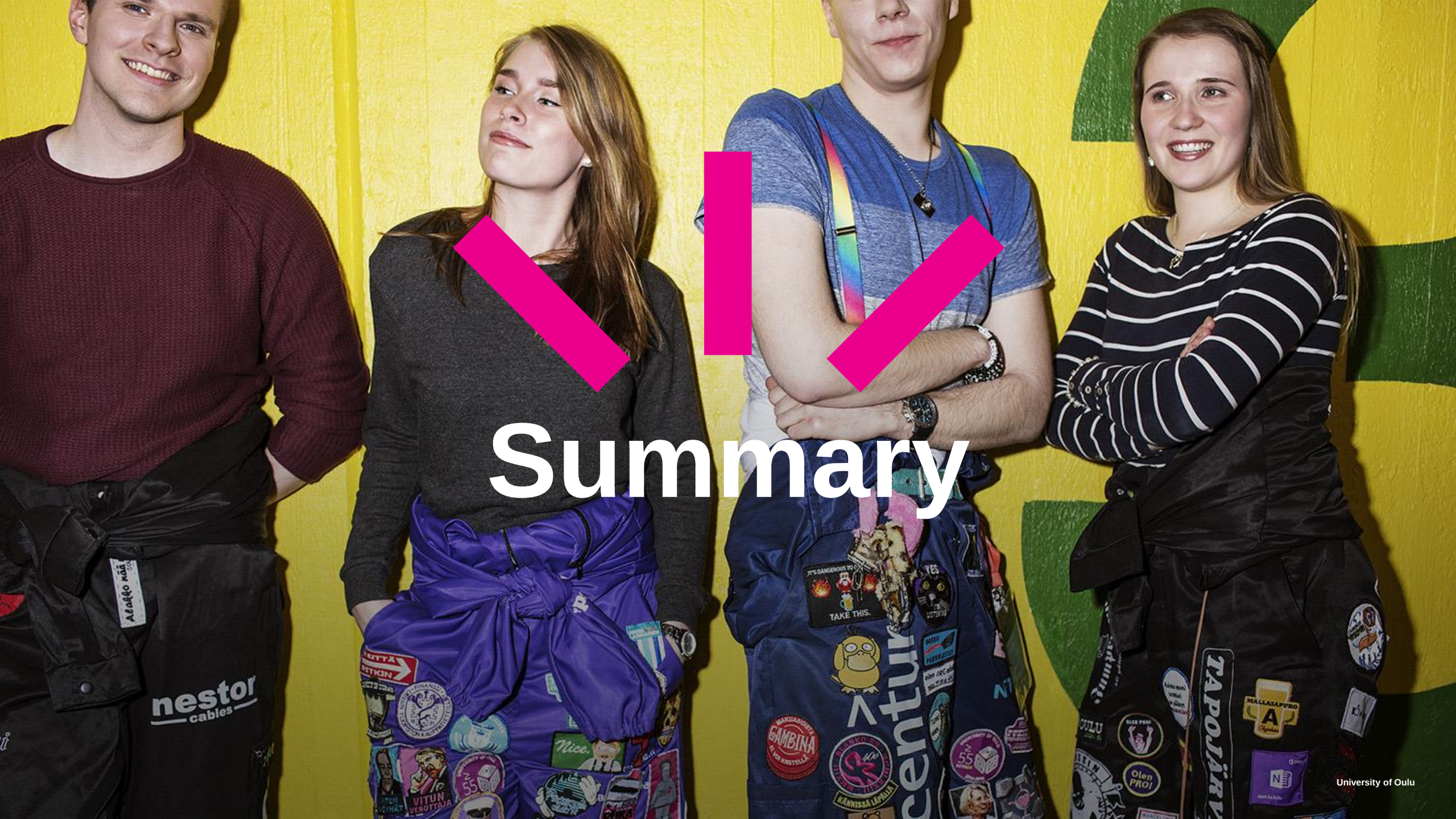
		Predicted	
Correct		Positive	Negative
	Positive	20, p_11=20/300	5
	Negative	20	255

- K = 0%

		Predicted	
Correct		Positive	Negative
	Positive	0	25
	Negative	0	275

- κ = 30,8%

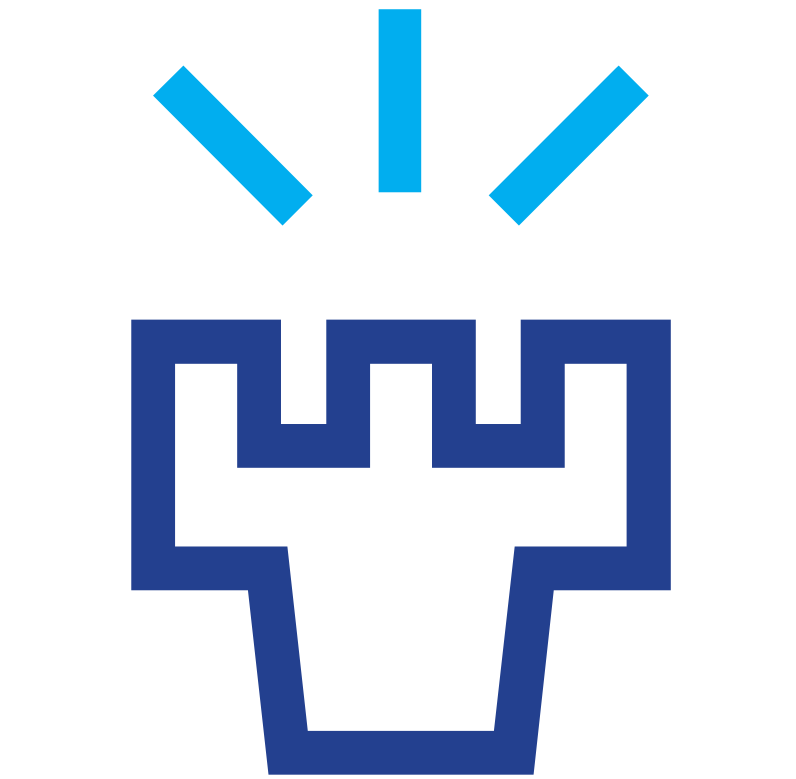
		Predicted	
Correct		Positive	Negative
	Positive	25	0
	Negative	75	200


nestor
cables
TAKE THIS.
GAMBINA
VITUN VEROTTAJA
Nice.
Olen PRO!
TAPOLIÄRI

Summary

Summary

- **When data from different sources are merged, make sure that you understand what data you have and where it is from**
  - Units
  - Frequences
- **If you think that data is not well documented, do not use it**
- **If you do not have enough data or you have too much data, try sampling or artificial data**
- **High accuracy does not mean that model recognition good**
  - Use also other performance metrics
- **Imbalanced data sets require different performance metrics than balanced**



UNIVERSITY
OF OULU