TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

Actuarial Data Science

Maschinelles Lernen in der Versicherungen

Institute for
Finanz- und Versicherungsmathematik TU Wien
under the guidance of
Associate Prof. Dipl.-Ing. Dr.techn. Stefan Gerhold

Vladimir Maric
Matriculation number: 11741615
Wien, 17. December 2021

Table of contents:

1. Introduction to the idea of machine learning in insurance

I. Cancellation forecast

In the risk management departments of insurance companies, actuaries deal with all the risks to which the respective company is exposed and try to model and quantify them, so they carry out calculations in which they project financial figures and payment flows of the insurance company into the future as models over a few or many years.
In addition to many other things, suitable cancellation forecasts are required for the projection calculations. There are so-called classic cancellation models for this, but the question is also suitable for the use of machine learning methods.
Incidentally, sales are also interested in good cancellation forecasts. To prevent cancellation, they want to find out which customers are more likely to terminate their contract in the near future.

II. Stock consolidation

In the projection calculations mentioned above, we differentiate between deterministic and stochastic projections. With a deterministic projection, the entire company model is extrapolated once over the projection period. In a stochastic projection, several (usually many) capital market paths are used and the company model is extrapolated for each individual capital market path (Monte Carlo simulation). The extensive results must then be appropriately aggregated.
In addition to the use cases detailed above, there are numerous other demanding tasks in the insurance sector in which data science methods are successfully used. Examples are: Modelling of biometric risks, motor vehicle Telematix, the reduction of the catalog of questions for risk assessment in occupational disability insurance, the automated performance assessment for non-life insurance, etc.

III. Data mining crash course

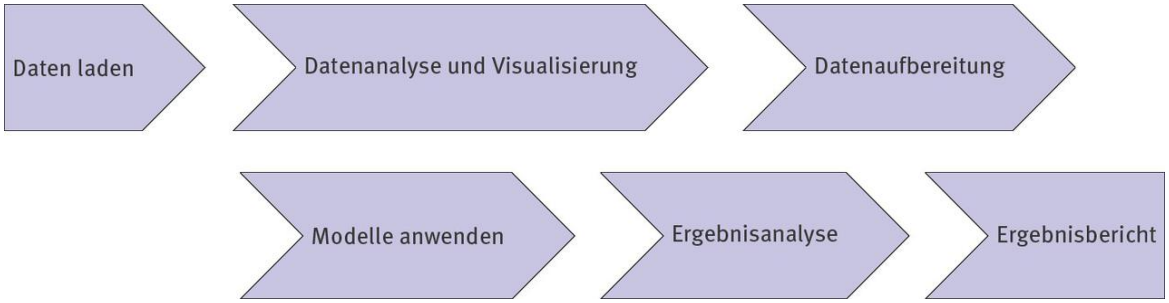
The term data science is often used to describe the discipline that is dedicated to extracting knowledge from data. This encompasses the entire

process of handling data, starting with the collection and recording, preparation and storage, processing and evaluation through to data mining. Actuarial Data Science deals with the collection, recording, processing and evaluation of insurance-specific data under an actuarial question. The data is usually evaluated using machine learning methods ie. artificial intelligence (building a machine that can independently process tasks or solve problems). Weak artificial intelligence is AI in the technical sense, i.e. machines that appear intelligent because they are able to learn and adapt their strategies to changing circumstances. In the context of asset-liability management, the actuary needs the expected probability of cancellation for his life insurance portfolio for his projection calculations. The actuary would therefore like to write a simple program that calculates the probability of cancellation for a contract in his portfolio.

Example. Let us assume that the actuary models the probability of a cancellation for a contract depending on the actuarial age x of the insured person at the time of cancellation, the time (date) t_0 of the conclusion of the contract, the product P and the insured amount V. And he also has considered a formula that appears plausible to him from an actuarial point of view and that fits his observations. He derived the formula from historical data from his inventory for the past ten years, specifically from the data tuples (x, t_0, P, V) of the canceled contracts. For example, he has observed that younger policyholders are more likely to cancel, that the probability of cancellation decreases with the expiry of the contract and increases again in the years shortly before it expires. In addition, the likelihood of cancellation increases as the sum insured gets bigger and is significantly higher in certain products than in others.

IV. Tools and resources (data); Python and R

Software solutions that can be used for tasks in the field of actuarial data science.



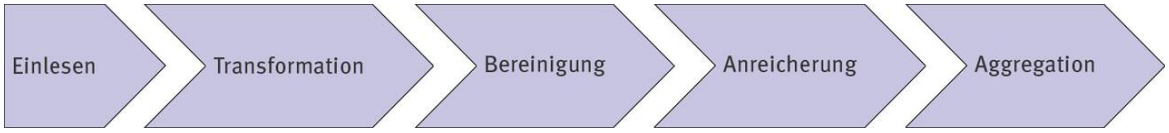
Together with WEKA and KNIME, Python and R are open source, more approachable programs that are getting bigger and bigger. Python is a rising force in many areas in the last 10 years, such as programming, web development as well as statistics, insurance calculations, data plotting etc. That's why in the next examples, we will focus on how they could be done and represented in Python, rather than other similar programs.

V. Reproducible research

The term reproducible research refers to the idea that scientific results should be documented in such a way that their deduction is fully transparent. This requires a detailed description of the methods used to obtain the data and making the full dataset and the code to calculate the results easily accessible. This is the essential part of open science.
To make any research project computationally reproducible, general practice involves all data and files being clearly separated, labelled, and documented. All operations should be fully documented and automated as much as practicable, avoiding manual intervention where feasible. The workflow should be designed as a sequence of smaller steps that are combined so that the intermediate outputs from one step directly feed as inputs into the next step. - Tables, Figures, Graphs etc.

2. Information technologies

In this chapter we discuss the basics of data processing from a technical perspective.



I. Working with data and various data sorts

Permanent storage of digital documents takes place in the form of files on electronic media such as hard drives. Depending on the specific work

situation (is team access required?), confidentiality requirements or availability in general, different variants are used, for example:
- the storage of files on the local hard drive of a workstation computer,
- the storage of files on central network drives in the company,
- the use of collaboration platforms such as Microsoft's SharePoint,
- the storage in a cloud offer such as Google Docs, Google Drive or Microsoft OneDrive,
- Cloud Object Storage like Amazons S3 (especially for pictures etc.),
- Third-party or self-hosted collaboration platforms such as Git.
Here you usually have to make compromises in individual cases, as each variant has specific advantages and disadvantages.
In contrast to text files, the bytes in a binary file do not necessarily represent characters. As a consequence, we state that binary storage is usually more compact and can be processed more efficiently, but requires precise knowledge of the structure and can therefore usually only be read by (the correct version) of the correct software.

II. Modelling information into useful input

In many cases, input data for a data analysis are already available in tabular form or they are transferred to it in a preprocessing step. Next to the tabular input, Information can be stored in Excel not-tabular shape, textually or in some sort of pseudo form.

III.JSON, Web-APIs, Websites

If you obtain data directly from the Internet, you will find not only the known file formats but also other formats with html and **json** that have been specially developed for the web. JSON (Java Script Object Notation) is probably the most important of the formats mentioned. This is due, among other things, to the following properties:
- the format is easy to process by machine,
- JSON is human-readable,
- JSON is more compact than many other formats (e.g. as XML),
- JSON is part of the JavaScript specification, which is probably the most widely used programming language at the moment.

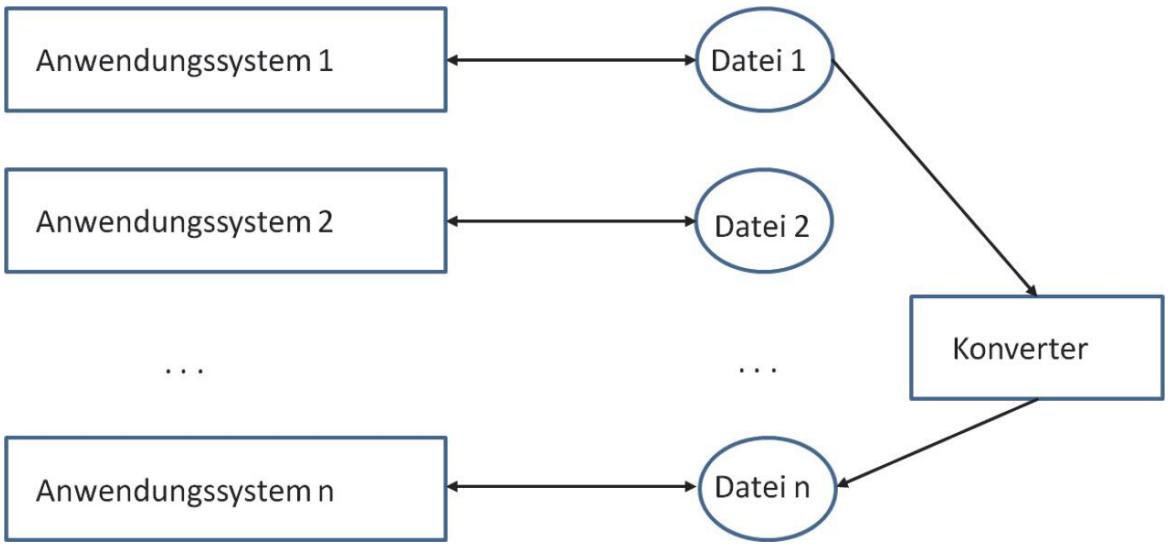
The term **Web-API** stands for an interface provided by a server, which can be addressed via HTTP requests from other software systems (in this context then called clients).
S3 is an offer from the Object Storage category in the Amazon Cloud AWS (Amazon Web Services). The concept of object storage makes it possible to store any binary data (objects) together with a limited amount of metadata (a set of key-value pairs) under one key. In S3, the objects are also stored in a bucket (a kind of folder at the highest level).

IV. Data banks

The relational database systems are the most important data storage system in insurance companies and almost all operational systems use them for data storage.

The benefits of (relational) databases

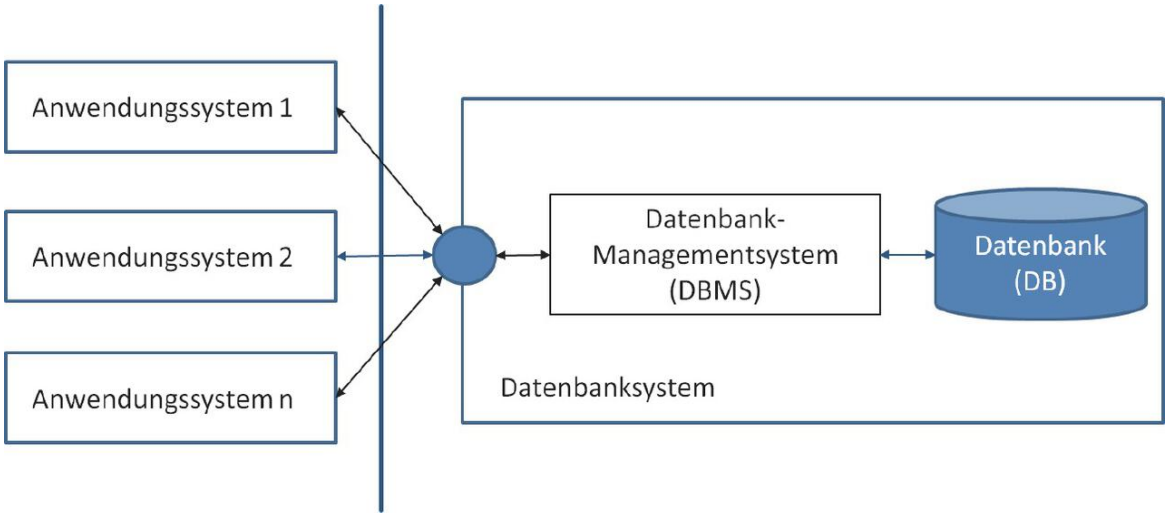
The application-specific management of data by one application was the rule for a long time, which made it possible to achieve a high level of efficiency in routine operation. The application writes and reads all required data exclusively directly in the file system.



The exclusive use of data also has disadvantages:
- Data that is required by several application systems may be saved multiple times (redundant data storage),
- Necessity of converter programs in order to be able to exchange data between different applications,

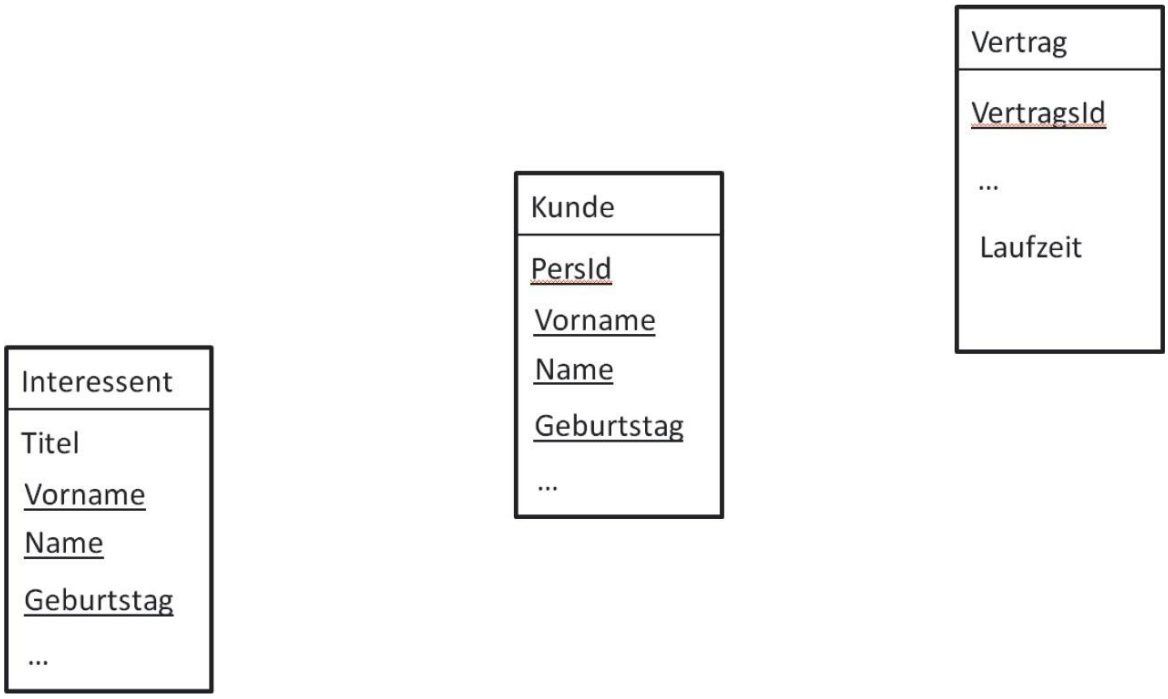
- Changes in the data structure force reprogramming of the applications (depending on the physical data structure),
- Exclusive use by an application, i.e. no parallel access
- Each application system ensures the integrity of its data itself (generally no general mechanism for data consistency / access protection).

The latter point naturally also implies that the corresponding nontrivial mechanisms of data management must be designed and created from scratch for each application. To counter these problems, the concept of centralised data storage, so-called databases (DB), was developed in the 1960s. A database is a collection of structured data, between which there are logical relationships.



Objects and object sorts

We understand an entity to be a concrete object about which information is stored. Objects can be people (e.g. customers), objects (buildings, cars,...) Or non-material things such as an employment relationship. An entity type is a uniquely named class of objects about which structurally identical information is stored and which is basically processed in the same way.



Example of bank transfer

100 EUR are to be transferred from account X to account Y, whereby it is assumed as a secondary condition that account X has more than 100 EUR. If the system now first assigns the 100 EUR to Y and then determines that X does not have the necessary coverage, you will have a problem, because there could be a power outage, which in this case could have unintended economic effects after startup.

Even if X has coverage, it must be ensured that both processes only take place together (or fail together). External factors such as network failure or server crash (too little memory, etc.) should also be taken into account here. If an error occurs, a ROLLBACK is triggered, which reverses all actions of the transactions that have not yet been completed.

The successful completion of a transaction is the COMMIT (provides the database confirmation regarding the changes made). The encapsulation of actions in transactions alone does not, however, cover all possible problems. Under the keyword transaction isolation you can find out about the details of the guarantees that an interesting database system offers. Other means that may be necessary to guarantee consistency are so-called locks, which temporarily block individual lines or entire tables for other connections. Accesses with set locks are then exclusive for a transaction and so certain conflicts can be avoided.

V. Decomposed data systems

The relational database model (also called SQL databases) is not the only one imaginable. However, it was not until shortly after the turn of the millennium that several database systems, each of which differed from relational databases in certain respects, reached a level of quality in a short period of time that made them appear to be real alternatives from then on. Some of these products then became known as No-SQL databases. The main advantages of those are higher performance and scalability. Many developments in the area of so-called No-SQL databases have their origin in the goal of providing distributed databases or database management systems. There are a number of reasons why you want to run databases as distributed systems:

- Size: The amount of data can simply be too large for a single computer, you have to distribute the data on different computers or storage media.
- Availability: You want to protect yourself against (even brief) system failures by using several so-called (hot) standby systems.
- Security: Protection against data loss through redundant storage.
- Performance: If the query load is high, distributing the queries can help to reduce the response times. The accesses are distributed to different servers (load balancing), so that a higher degree of parallelism can be achieved in processing.

VI.CAP theorem

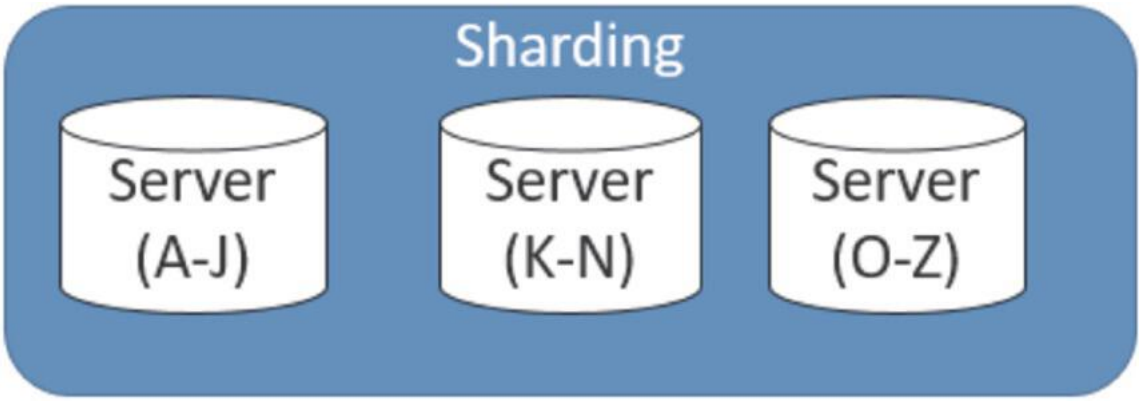
Complete solution of the problem of connecting the distributed databases with the guarantees of consistency of the relational model cannot be achieved. According to Brewer's CAP theorem, it is not possible to guarantee the properties of consistency, availability and partition tolerance of a distributed database management system at the same time. It is assumed here that data are stored at different locations in a networked system and that data is stored redundantly (data repetition).

The terms used in the CAP theorem can now be interpreted as follows:

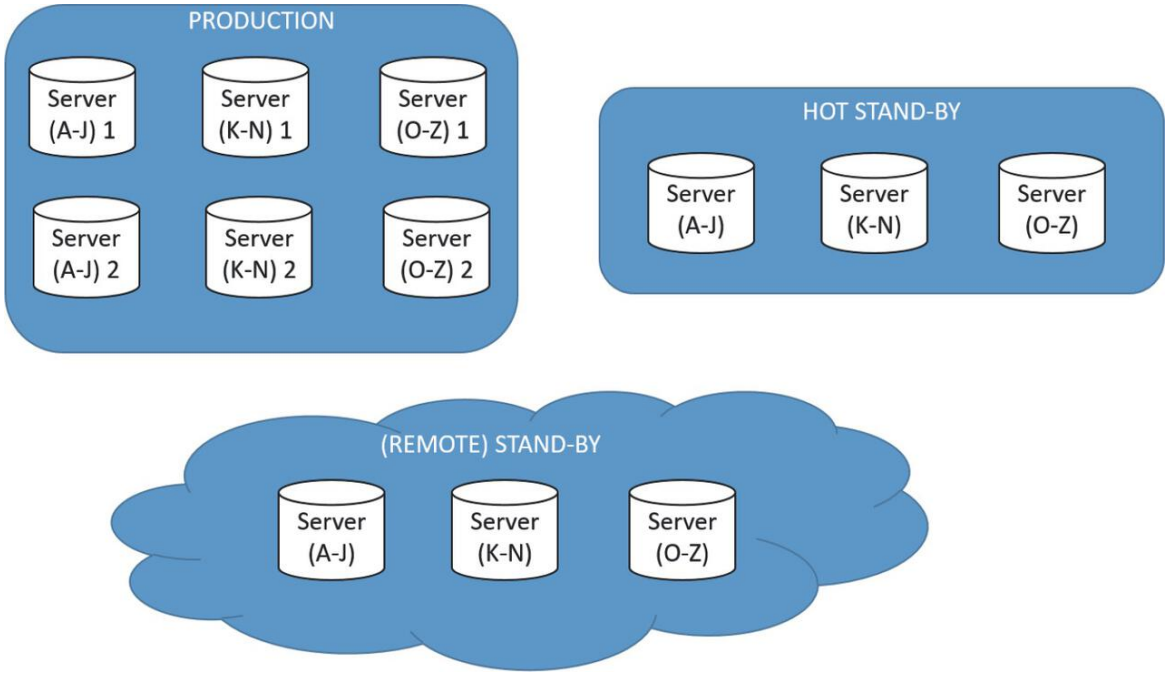
- According to P, the system should work even if communication within the system is disturbed.
- According to A (every part) of the system should process transactions, even if communication within is disturbed.

- According to C, the answers from different parts of the system should be consistent.
If communication within the system is disturbed, the delivery of consistent responses and the simultaneous processing of write operations cannot be agreed with one another. However, certain scenarios such as the sharding (horizontal partition) and replication scenario can be implemented:

The entire data is stored redundantly here. If write processes are carried out in parallel on all instances, then all instances can react to read requests through suitable load balancing and thus increase the read performance (but each request goes entirely to one instance).



Depending on the value of a key, data is stored in a separate instance (no redundancy). Each data record is only available once in this form.



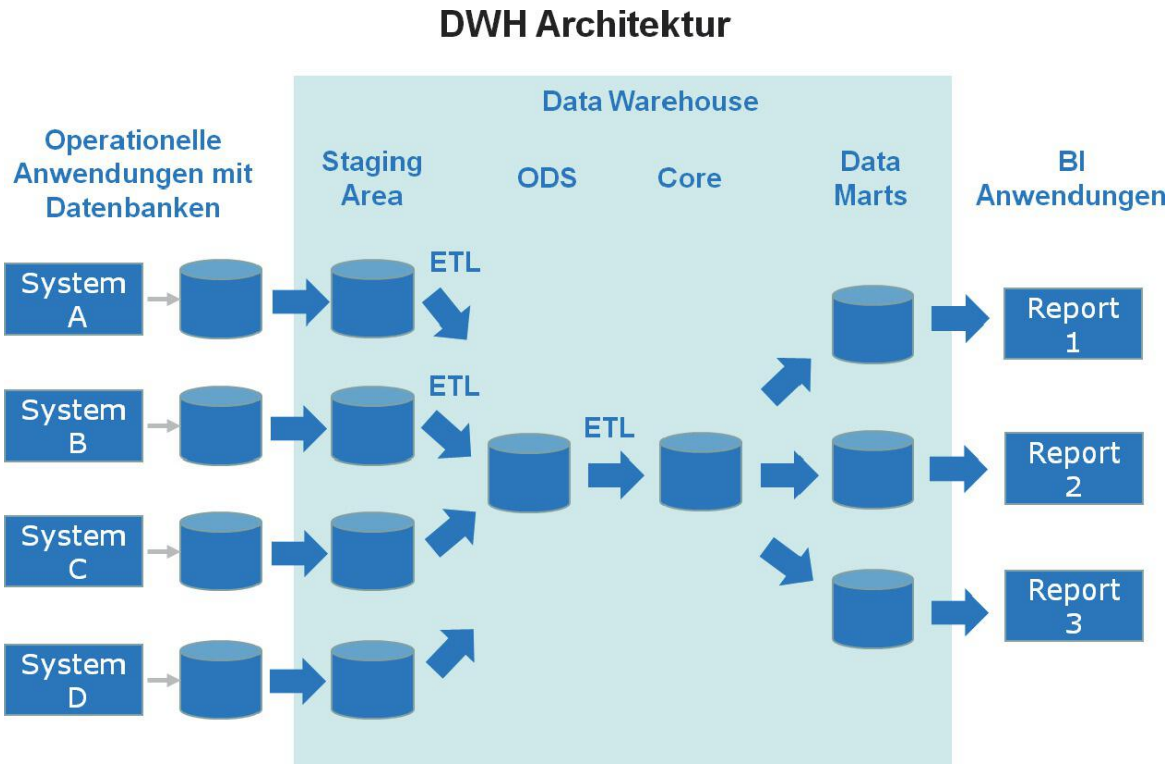
A possible configuration of a productive distributed database. The database is divided into six subsystems, which hold the data in three separate units, which in turn consist of two units that are kept synchronised in order to be able to process read accesses quickly. The productive system is supplemented by two backup systems: A hot standby system is always kept in sync with the productive database and can replace the productive system at short notice in the event of a failure. The remote standby server is located at a different location and the spatial separation ensures additional data security.

VII.Data warehouses

Operational data storage. Insurance companies are nowadays supported in their operational areas of application by a large number of operational information systems. The most prominent example are inventory management systems. Such an operational system naturally not only consists of software that provides the user input and the business logic, but also of one or more databases for storing the required data. These databases are optimised for fast and error-free access by the operational systems and are sometimes characterised by the following properties:

- high access frequency,
- dynamic, transaction-related updating of data,
- no redundancy of data storage and referential integrity.

Dispositive data storage. In order to support analytical and planning management tasks or company controlling, dispositive information systems are required. Dispositive information processing is based on data extracted from operational application systems and external sources. Typically the target of this extraction is a data warehouse (DWH). In addition to the DWH, another dispositive IT system has recently been established: the data lake. With the help of a data lake, very large amounts of structured and unstructured data can be stored and processed. It is therefore an ideal starting point for data science use cases, although there may be increased initial effort for data preparation.



The staging area initially contains a copy of the operational data. This is required because the data warehouse should not access the productive databases directly (this could, for example, slow down operational applications).
The operational databases often contain many data fields that are of no interest for dispositive applications. The Operational Data Store (ODS) only stores data that is relevant for reports and other applications in the warehouse. While the data was distributed across different databases in the operational layer, it is now brought together in a common database with a standardised schema. In order to make the data usable for analyses, it is also cleaned up and validated, incorrect entries and duplicates are removed.

VIII.Parallel data processing

Scalability is the next major factor in improving our work on data. It denotes the ability to increase the performance of a system by adding components. A distinction is made here between vertical and horizontal scaling. Vertical scaling (scale up) is expanding the available computing power through faster CPUs, more main and mass storage, faster network connections, etc. The width of the expansion described below is potentially unlimited. A major advantage of vertical scaling, on the other hand, is that basically no changes to the algorithms and programs are required.

Horizontal scaling (scale out) means scaling by adding further computers or nodes. In principle, any number of further units can be added to the system, but changes to the algorithms are generally required so that the additional resources made available can actually be used (parallelism).

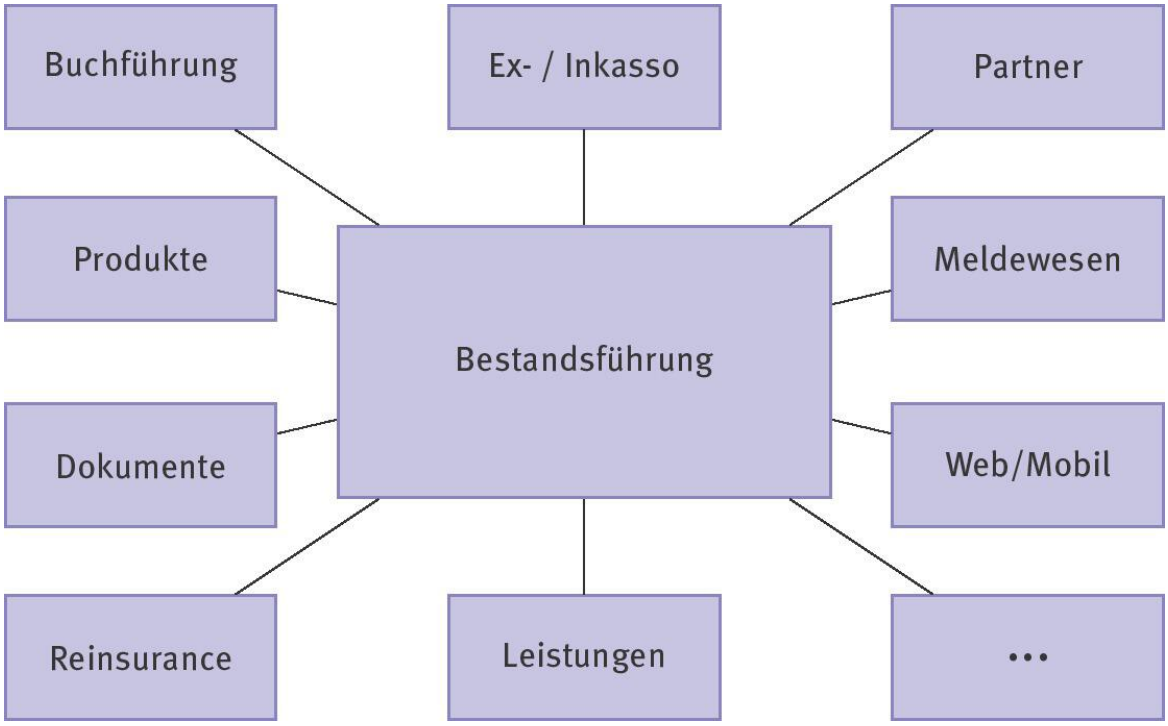
In addition to the aspects of the technical implementation of scalability, a distinction is made between the following conceptual types of scalability:

- Load scalability: It concerns the scalability in relation to the load on the system. A good example of this are websites or web servers, which even during peak loads, i.e. high-frequency demand,
should always be responsive.
- Spatial scalability: This is about the use of resources with increasing requirements, especially when it comes to data storage. The use of compression methods in data storage can, for example, result in the storage requirement growing only sub-linearly with the number of stored data records.
- Temporal-spatial scalability: This concerns the temporal consumption of resources with an increasing number of relevant objects. An example of this are search engines that scale well in terms of time and space,
when a search index is used. However, they scale poorly in a simple, linear search.
- Structural scalability: This is about structure-related limits in the growth of a system. A counterexample would be a system with hard implemented limits (e.g. possible number of entries in a directory <1 million), a positive example would be a system without such fundamental (structural) limits that grows with the possibilities of the hardware.

IX. Information processing for insurance companies

To apply data science in the insurance environment, it is not enough just to know the mathematical methods, the regulatory environment and some exciting use cases, but also a basic understanding of where and how which data is required within a business process of insurance company.

Business processes in insurance companies. Accordingly, there are strategic management and leadership processes such as strategy definition, controlling, risk management and internal auditing, operational core processes such as sales, product development, inventory management, claims management and supporting processes such as procurement and accounting.



Typical system landscape of an insurance company. The inventory management is prominent in the middle, surrounded by various edge systems.

Many of these systems are still operated today on three different platforms:

- the IBM mainframes (also called host or mainframe),
- various servers,
- the single-user PCs.

X. Recapitulation through Python example

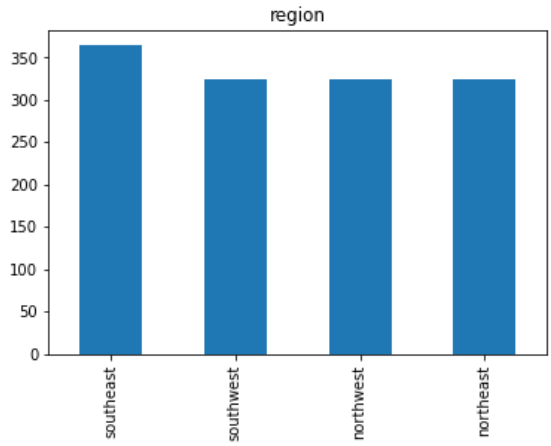
We are going to finish off the first half of the work with a Python example, through which we will show in praxis how some of the mentioned ideas are used, as well as how the computer programs made for this specific tasks do their job.

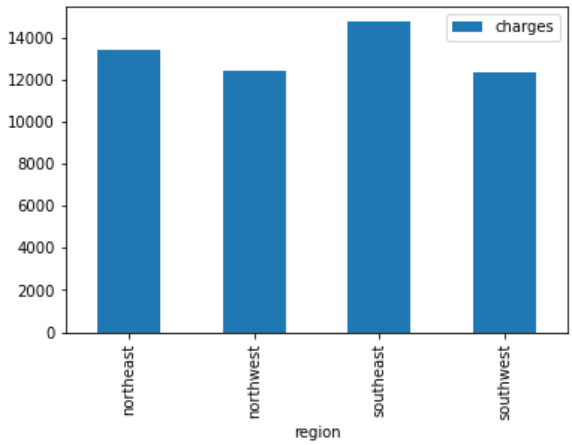
Health Care Cost Analysis/Prediction: Consider the data frame of over a thousand people with informations about their age, sex, bmi, region etc. and their health care charges. Now visualise those charges with respect to the informations which affect it, and make a prediction for the future health care costs considering the variables from above.

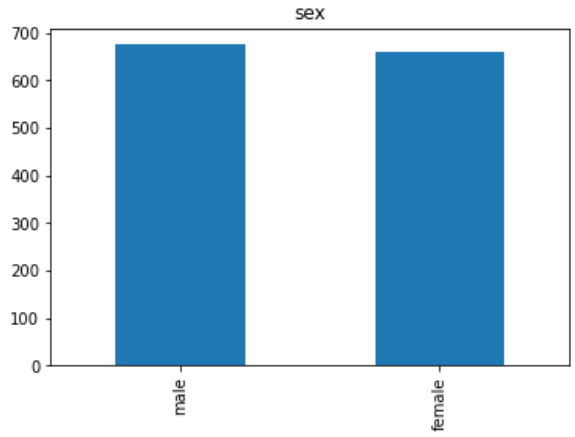
Index	age	sex	bmi	children	smoker	region	charges	bmi_int
0	19	0	27.9	0	1	3	16884.9	27
1	18	1	33.77	1	0	2	1725.55	33
2	28	1	33	3	0	2	4449.46	33
3	33	1	22.705	0	0	1	21984.5	22
4	32	1	28.88	0	0	1	3866.86	28
5	31	0	25.74	0	0	2	3756.62	25
6	46	0	33.44	1	0	2	8240.59	33
7	37	0	27.74	3	0	1	7281.51	27
8	37	1	29.83	2	0	0	6406.41	29
9	60	0	25.84	0	0	1	28923.1	25
10	25	1	26.22	0	0	0	2721.32	26
11	62	0	26.29	0	1	2	27808.7	26
12	23	1	34.4	0	0	3	1826.84	34
13	56	0	39.82	0	0	2	11090.7	39
14	27	1	42.13	0	1	2	39611.8	42
15	19	1	24.6	1	0	3	1837.24	24
16	52	0	30.78	1	0	0	10797.3	30
17	23	1	23.845	0	0	0	2395.17	23
18	56	1	40.3	0	0	3	10602.4	40
19	30	1	35.3	0	1	3	36837.5	35
20	60	0	36.005	0	0	0	13228.8	36
21	30	0	32.4	1	0	3	4149.74	32
22	18	1	34.1	0	0	2	1137.01	34
23	34	0	31.92	1	1	0	37701.9	31
24	37	1	28.025	2	0	1	6203.9	28
25	59	0	27.72	3	0	2	14001.1	27

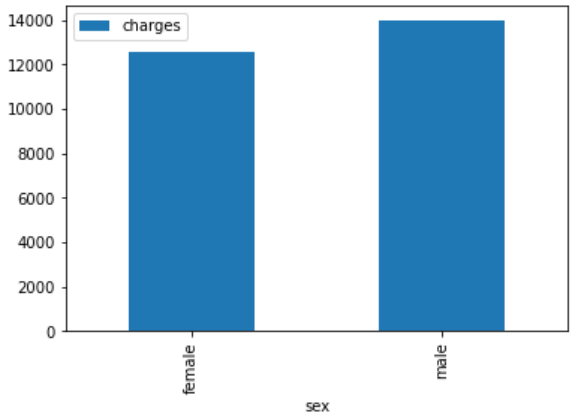
After importing the data frame, we can visualise the data with Python's plotting packages:

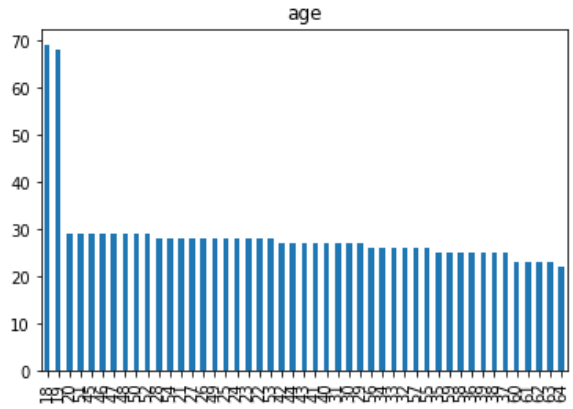
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Dec 14 16:11:04 2021

@author: vladimirmaric
"""

import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
import numpy as np
import sklearn.metrics
import warnings


warnings.filterwarnings('ignore')

#load data
df = pd.read_csv('/Users/vladimirmaric/Desktop/Seminar/Python Code/insurance.csv')
df = df.dropna()

#general information
df.describe()
df.corr()

df['bmi_int'] = df['bmi'].apply(lambda x: int(x))
variables = ['sex','smoker','region','age','bmi_int','children']

# data distribution analysys
print('Data distribution analysys')
for v in variables:
    df = df.sort_values(by=[v])
    df[v].value_counts().plot(kind = 'bar')
    plt.title(v)
    plt.show()


#average cost analysys
print('Mean cost analysys:')
for v in variables:
    group_df = df.groupby(pd.Grouper(key=v)).mean()
    group_df = group_df.sort_index()
    group_df.plot(y = ['charges'],kind = 'bar')
    plt.show()
```

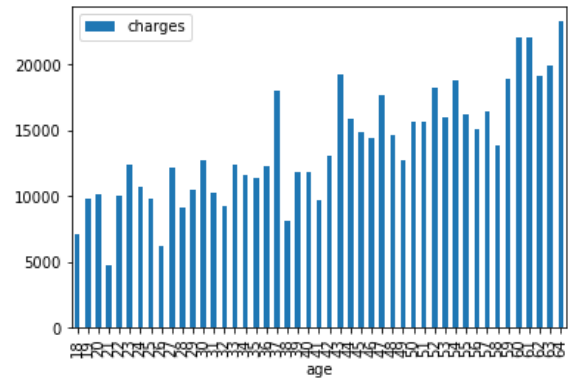


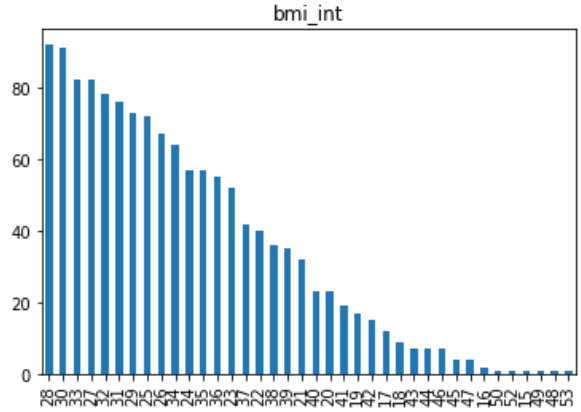


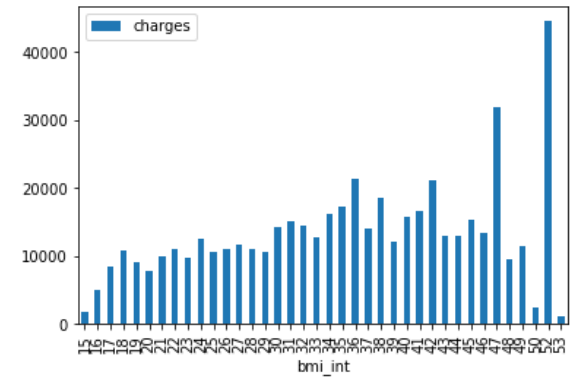

sex
male
female

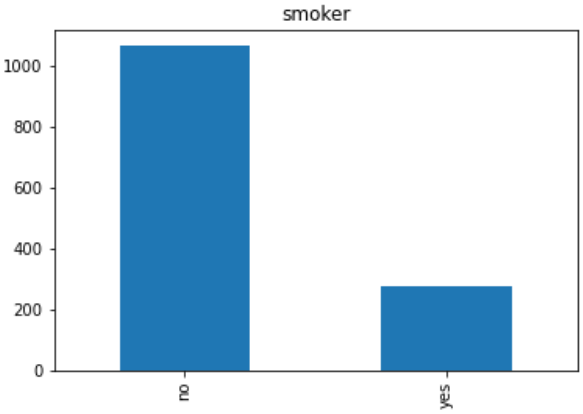

charges
female
male
sex

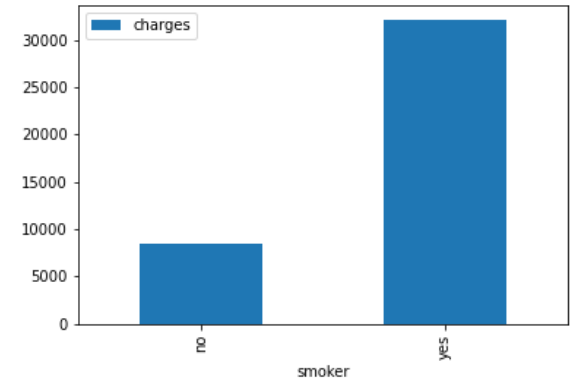

age

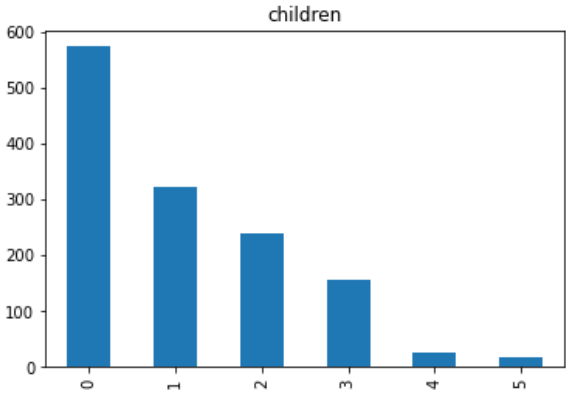

charges
age

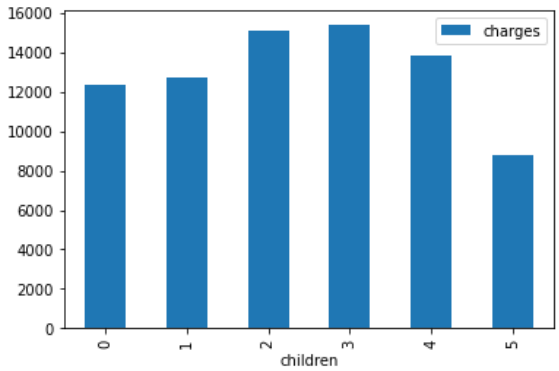

bmi_int


charges
bmi_int


smoker
no
yes

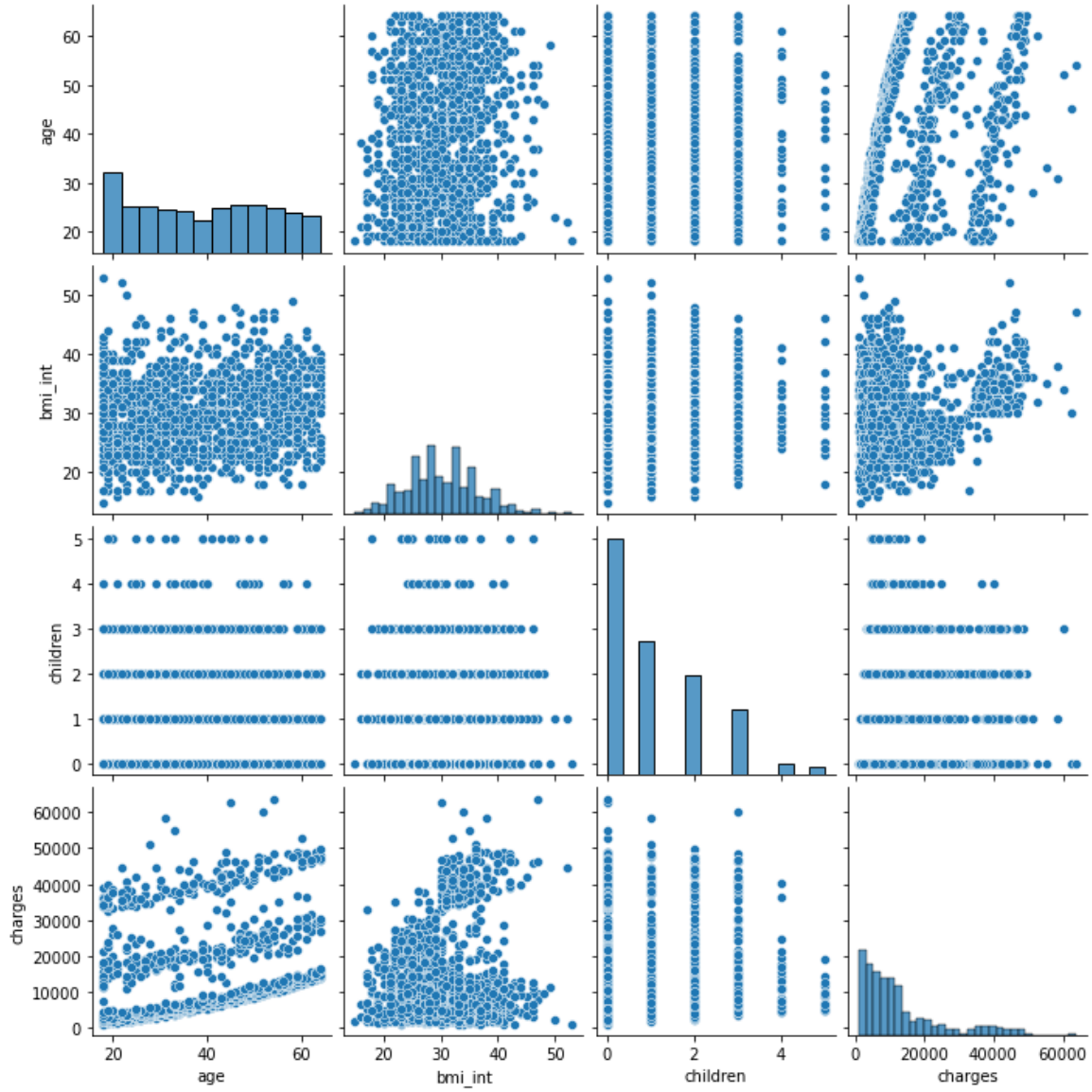

charges
no
yes
smoker





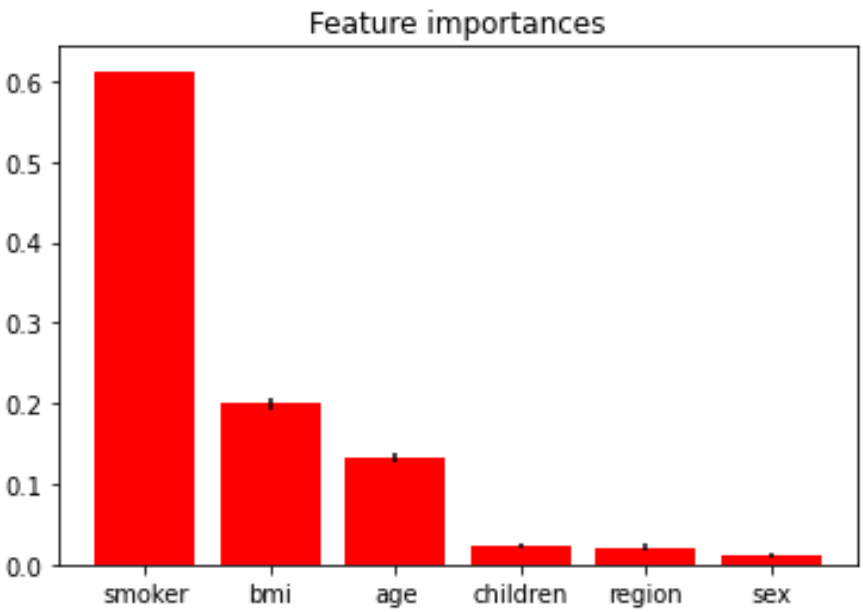
Let's compare our variables and see if there's correlation (statistical association between variables) or causation (change in one variable causes a change in another variable) between them:

```
#variables pairplot
print('Variables pairplot:')
variables = ['sex','smoker','region','age','bmi_int','children','charges']
sns_plot = sns.pairplot(df[variables])
plt.show()

print('Model training and evaluating\n\n')
#transform categorical data
le_sex = LabelEncoder()
le_smoker = LabelEncoder()
le_region = LabelEncoder()

df['sex'] = le_sex.fit_transform(df['sex'])
df['smoker'] = le_smoker.fit_transform(df['smoker'])
df['region'] = le_region.fit_transform(df['region'])

variables = ['sex','smoker','region','age','bmi','children']

X = df[variables]
sc = StandardScaler()
X = sc.fit_transform(X)
Y = df['charges']
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)

#train model
regressor = ExtraTreesRegressor(n_estimators = 200)
regressor.fit(X_train,y_train)

#prediction and evaluation
y_train_pred = regressor.predict(X_train)
y_test_pred = regressor.predict(X_test)

print('ExtraTreesRegressor evaluating result:')
print("Train MAE: ", sklearn.metrics.mean_absolute_error(y_train, y_train_pred))
print("Train RMSE: ", np.sqrt(sklearn.metrics.mean_squared_error(y_train, y_train_pred)))
print("Test MAE: ", sklearn.metrics.mean_absolute_error(y_test, y_test_pred))
print("Test RMSE: ", np.sqrt(sklearn.metrics.mean_squared_error(y_test, y_test_pred)))
```



Finally, we calculate the importance of each aspect in forming of the health care price, and are making our predictions about future users (with Python's regressor.predict command) based on given parameters:

```
print('Feature importance ranking\n\n')
importances = regressor.feature_importances_
std = np.std([tree.feature_importances_ for tree in regressor.estimators_],axis=0)
indices = np.argsort(importances)[::-1]

importance_list = []
for f in range(X.shape[1]):
    variable = variables[indices[f]]
    importance_list.append(variable)
    print("%d.%s(%f)" % (f + 1, variable, importances[indices[f]]))

# Plot the feature importances of the forest
plt.figure()
plt.title("Feature importances")
plt.bar(importance_list, importances[indices],
       color="r", yerr=std[indices], align="center")
plt.show()
```




```
print('Predicting on new data\n\n')

billy = ['male','yes','southeast',25,30.5,2]
print('Billy - ',str(billy))

billy[0] = le_sex.transform([billy[0]])[0]
billy[1] = le_smoker.transform([billy[1]])[0]
billy[2] = le_region.transform([billy[2]])[0]

X = sc.transform([billy])

cost_for_billy = regressor.predict(X)[0]
print('Cost for Billy = ',cost_for_billy,'\n\n')


dennis = ['female','no','southeast',45,19,0]
print('Dennis - ',str(dennis))

dennis[0] = le_sex.transform([dennis[0]])[0]
dennis[1] = le_smoker.transform([dennis[1]])[0]
dennis[2] = le_region.transform([dennis[2]])[0]

X = sc.transform([dennis])

cost_for_dennis = regressor.predict(X)[0]

print('Cost for Dennis = ',cost_for_dennis)
```

```
Predicting on new data


Billy -  ['male', 'yes', 'southeast', 25, 30.5, 2]
Cost for Billy =  31746.65618799997


Dennis -  ['female', 'no', 'southeast', 45, 19, 0]
Cost for Dennis =  8518.815572750005
```

XI. Predict command

Since one of our topic's key points is the idea of machine learning, I would find it superficial not to discuss at least about the intuition behind Python's "*regressor.predict*" command. Primarily, the function accepts only a single argument which is usually the data to be tested. Then, through the method of regression (which will be assessed in the following chapter), function calculates the importance of each parameter from the data set ie. draws a conclusion, by comparing given sets of values, on how much every parameter affected the price. Finally, all of the parameters are multiplied with the coefficients (weights) that the program assigned to them, regarding the "feature importance" of every one of them. When all of them get "summed up" (through a pre-defined formula) we get an expected value for newly inputed parameters (of in our case health insurance cost).

3. Mathematical procedures

In this chapter we address various mathematical aspects of machine learning. This of course includes an insight into the theoretical basics of the most common algorithms for supervised learning, as well as dissection of the ideas such as cluster method, dimension reduction and model evaluation, among others.

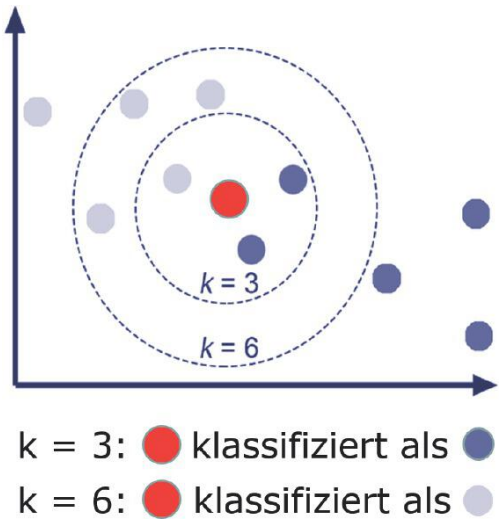
I. Preparation and dealing with missing data

When preparing and visualising the data, the goal is to convert the data into a more understandable format (as seen in the example above). However, since information tend to get corrupt more often than not, ways of dealing with missing data must be considered as well. Firstly, let's make the distinction between the following situations:

1. Missing Completely at Random (MCAR): There is no systematic connection between the absence and the other data.
2. Missing at Random (MAR): This is the case when the probability of the missing in certain groups of observations is constant.

3. Missing Not at Random (MNAR): This is the case if neither of the other two apply.

Depending on the situation, some solutions could be more effective than others. Some simpler solutions could be to ignore whole rows or columns if there is missing data, or the replacement of missing entries with the mean value, the median or the most frequent value. Now let's inspect one of the used methods, the **k-Nearest-Neighbors (kNN)** method: It is the idea of replacing the missing values with neighbouring values. As usual, you have to define the number of neighbouring points to be considered and the distance to be used. It should also be considered whether the different dimensions of the data set should be normalised beforehand, i.e. brought to a common scale.



II. Classification and regression method

In this section we discuss some important methods of supervised machine learning. It is a matter of predicting the value of a target variable $\mathbf{y}$ using a number of other features (also called predictors or covariables). A distinction is made between the regression problems and the classification problems in which the target variable y only has a finite number of values, i.e. ($y \in \{C_1, \ldots, C_n\}$). Typical examples of this would be the indicators cancellation / no cancellation or spam / no spam.

III. Types of variables

For the algorithms it is usually decisive which types of observations are available. On a first level, a distinction can be made between numerical and categorical variables. The class of numerical variables can be further subdivided into discrete (i.e. the values can be counted or counted infinitely) and continuous variables. The continuous variables can be further subdivided into interval-scaled variables, for which differences in measurements have a meaning (this is the case, for example, with body temperature in ° Celsius), and ratio-scaled variables. The latter allow the formation of meaningful quotients and have a "meaningful zero point". This is the case, for example, with the heart rate.

For the categorical variables, a distinction can be made between ordinally scaled and nominally scaled cases. There is no linear order for nominally scaled variables. They can be numerically coded, but the specific value is irrelevant. An example is the gender of a person, which can be coded with the values 0 and 1, but just as well with M or F.

IV. Multiple linear regression

Linear regression is a linear approach for modelling the relationship between a scalar response and one or more explanatory variables (also known as dependent and independent variables). The case of one explanatory variable is called simple linear regression; for more than one, the process is called multiple linear regression. For regression problems we consider p explanatory variables $X_1, \ldots, X_p$ and a continuous (target) variable Y. A connection between $X = (X_1, \ldots, X_p)$ and Y in general form is established by an (unknown) function

$$f : Y = f(x) + \varepsilon.$$

Here, the term ε represents so-called disturbance variables, about which further assumptions are usually made. The task now is to find the best possible estimate of the unknown function f, i.e. to achieve as precise a separation as possible between the systematic component f and the error ε.

The observations of the explanatory variables are denoted by x_{ij}, which means the j-th variable of the i-th observation. These can be combined to form the so-called *design matrix* $X \in \mathbb{R}^{n\times(p+1)}$:

$$\begin{pmatrix} 1 & x_{11} & \dots & x_{1p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \dots & x_{np} \end{pmatrix}.$$

The row vectors of X are also denoted by x_i, i.e.

$$x_i = \begin{pmatrix} x_{i1} \\ \vdots \\ x_{ip} \end{pmatrix}.$$

Notation: The columns of X are denoted by $\mathbf{x_1}, \dots, \mathbf{x_{p+1}}$. Random variables are noted with capital letters, e.g. X or Y. The expected value of a random variable is denoted by E, the variance by Var and the covariance by Cov. Estimators are given a roof, e.g. $\hat{f}$ for an estimator for f. The transpose of a vector is given a superscript T, $\mathbf{x_i^T}$.

Generalised linear models (GLM) are particularly widespread in risk assessment in the financial sector. They can be applied to supervised learning problems. Typical representatives of GLMs are linear, logistic or Poisson regression. Let's inspect the first one, which we define as follows: the model

$$Y = \mathbf{X}\beta + \varepsilon$$

is called (classical) **linear regression** problem if:

1. The mean disturbances are 0, i.e. $E(\varepsilon) = 0$.
2. The mean variance of the disturbance variables remains constant, i.e. $Var(\varepsilon_i) = \sigma^2$ (homoscedastic error).
3. In addition, it is assumed that the disturbance variables are uncorrelated, i.e. $Cov(\varepsilon_i, \varepsilon_j) = 0$ for $i \neq j$.

Both assumptions result in a covariance matrix of the form $Cov(\varepsilon) = E(\varepsilon\varepsilon^T) = \sigma^2 I$. An additional condition for the classical normal

regression is that the errors are normally distributed, i.e. $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$. The following properties result directly from these assumptions:

- $E(Y_i) = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}$,
- $Var(Y_i) = \sigma^2$,
- $Cov(Y_i, Y_j) = 0 \ \ (i \neq j)$.

In matrix notation we get $E(\mathbf{Y}) = \mathbf{X}\beta$ and $Cov(Y) = \sigma^2 I$, from which for normally distributed ϵ: $\mathbf{Y} \sim \mathcal{N}(\mathbf{X}\beta, \sigma^2 I)$ follows. The expected value is therefore dependent on the covariables, but the variance is not. For the design matrix X the columns are linearly independent.
The estimator for the model coefficients is then

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y},$$

and the variance parameter can be estimated by

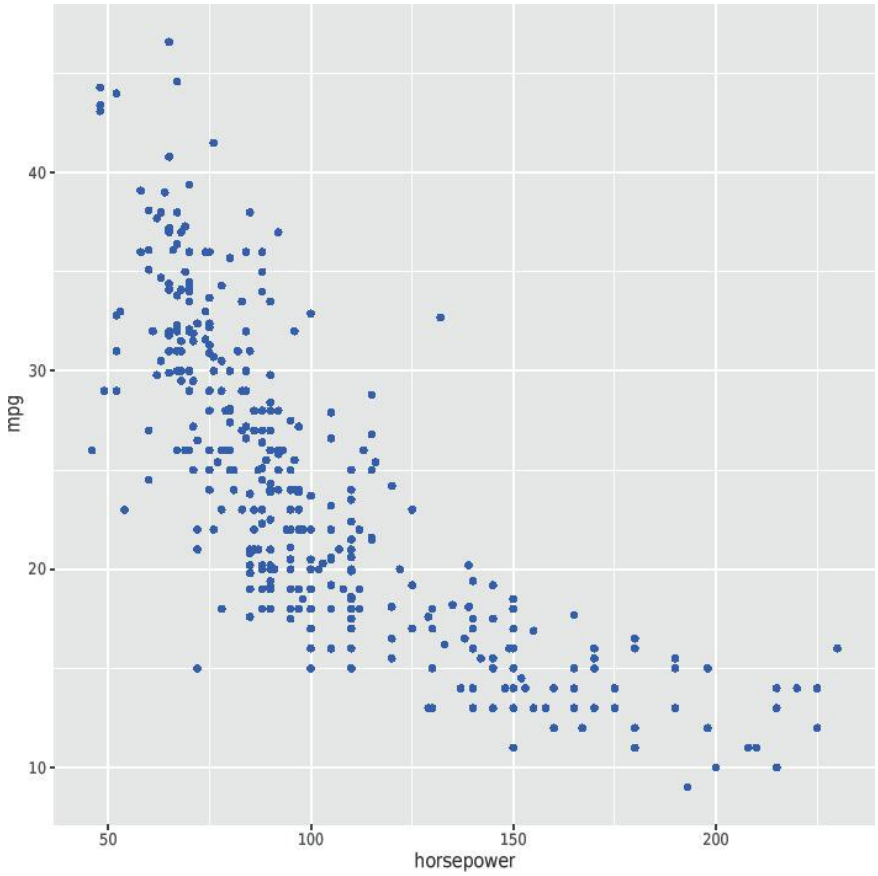
$$\hat{\sigma}^2 = \frac{1}{n-(p+1)} \sum_{i=1}^{n} (y_i - (\mathbf{X}\hat{\beta})_i)^2.$$

V. Simple linear regression example:

Let's observe the data set which contains information about the fuel consumption (mpg: miles per gallon) of various automobiles:

1	Mpg	stetig
2	Cylinders	Diskret
3	Displacement	stetig
4	Horsepower	stetig
5	Weight	stetig
6	Acceleration	stetig
7	Model year	Diskret
8	Origin	Diskret
9	Car name	Diskret

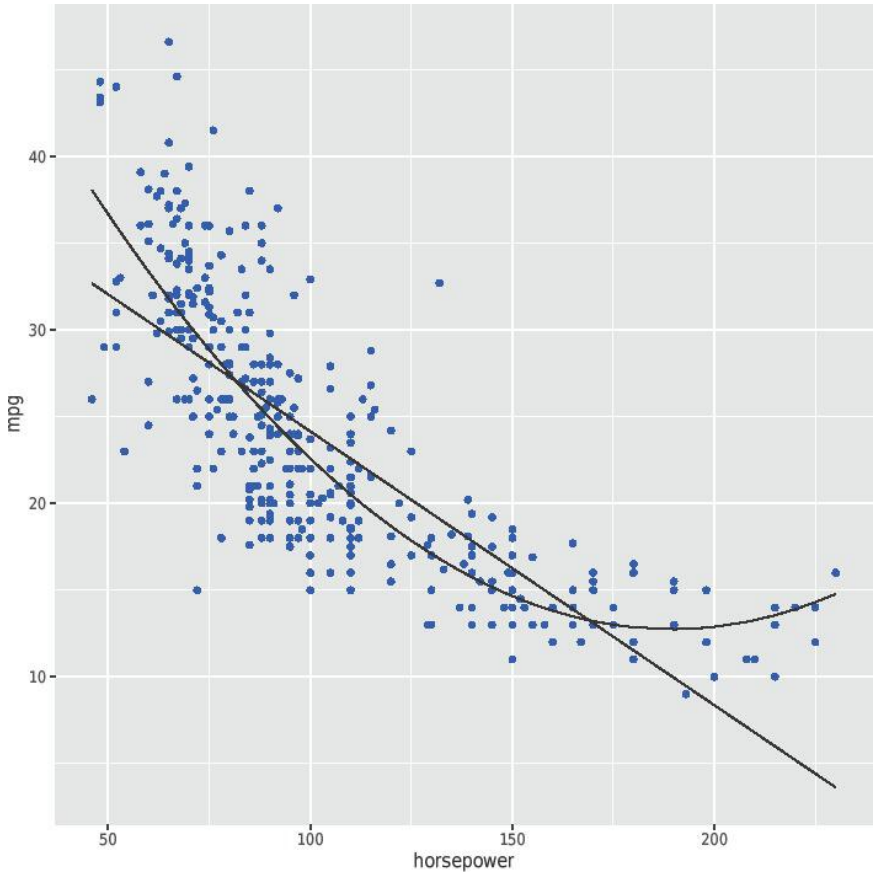
First, the relationship between the variables horsepower and mpg should be examined. We create a graphical representation of the two variables using the ggplot package and get:



Monotonically falling trend can be seen between the variables horsepower and mpg. This should be taken up in a linear model of the type $mpg = \beta_0 + \beta_1 \cdot horsepower + \varepsilon$ The absolute term calculated here and the value for the regression coefficient of the variable horsepower are

(intercept) horsepower
39.9359 **-0.1578**.

In the next step, the data points should be visualised together with the fitted regression line:



Result of the (simple) linear regression and the polynomial regression.

When looking at the regression line and the data points, it can be seen that the linear model is not necessarily suitable because the data contains a visible curvature. The explanatory variable of the model (horsepower) is significant, so that this model can serve as a starting point for more complex models.

VI.Other methods

Next to classification and regression method, there are others as well. Those which deal with the *reduction of data* are **Cluster method and Dimensionality reduction method**.

Cluster method is initially about partitioning methods that break down a set of objects (data points) into a set of groups (clusters) of similar objects. In other words, Clustering methods are used to identify groups of similar objects in a multivariate data sets collected from fields such as marketing,

bio-medical and geo-spatial. They are different types of clustering methods, including:

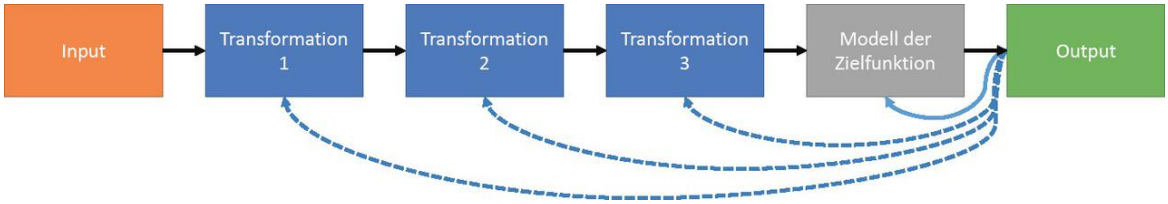
- Partitioning methods - subdividing the data sets into a set of k groups, where k is the number of groups pre-specified by the analyst. The most popular is the K-means clustering (MacQueen 1967), in which, each cluster is represented by the centre or means of the data points belonging to the cluster. The K-means method is sensitive to outliers.
- Hierarchical clustering - alternative approach to partitioning clustering for identifying groups in the dataset. It does not require to pre-specify the number of clusters to be generated.
- Fuzzy clustering - also known as soft method. Standard clustering approaches produce partitions (K-means, PAM), in which each observation belongs to only one cluster. This is known as hard clustering. In *Fuzzy clustering*, items can be a member of more than one cluster. Each item has a set of membership coefficients corresponding to the degree of being in a given cluster.
- Density-based clustering - partitioning method that has been introduced in Ester et al. (1996). It can find out clusters of different shapes and sizes from data containing noise and outliers (Ester et al. 1996). The basic idea behind density-based clustering approach is derived from a human intuitive clustering method.
- Model-based clustering - the data are viewed as coming from a distribution that is mixture of two or more clusters. It finds best fit of models to data and estimates the number of clusters.

On the other hand, **dimensionality reduction method** refers to techniques for reducing the number of input variables in training data. It is a data preparation technique performed on data prior to modelling. There are many techniques that can be used for dimensionality reduction.
The most common are so-called *feature selection techniques* that use scoring or statistical methods to select which features to keep and which features to delete. Two main classes of feature selection techniques include wrapper methods and filter methods. Wrapper methods, as the name suggests, wrap a machine learning model, fitting and evaluating the model with different subsets of input features and selecting the subset the results in the best model performance. Filter methods use scoring methods, like

correlation between the feature and the target variable, to select a subset of input features that are most predictive.
Techniques from linear algebra can be used for dimensionality reduction. Specifically, *matrix factorisation methods* can be used to reduce a dataset matrix into its constituent parts. Examples include the eigendecomposition and singular value decomposition. The parts can then be ranked and a subset of those parts that best captures the salient (noteworthy) structure of the matrix that can be used to represent the dataset can be selected.
Techniques from high-dimensionality statistics can also be used for dimensionality reduction. These techniques are sometimes referred to as "*manifold learning*" and are used to create a low-dimensional projection of high-dimensional data, often for the purposes of data visualisation. The projection is designed to both create a low-dimensional representation of the dataset whilst best preserving the salient structure or relationships in the data.

VII.Model evaluation

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future. First, here is a brief example: In the case of simple problems, the assumption is caught that (small) changes to the preprocessing should not have a major influence on the results of the actual inference steps. This is not obviously the case with complex data science pipelines, especially when the transformations consist of dimensionality reduction, clusterings and other aggregations. In fact, the opposite is often the case: If pseudo-random numbers are used in preprocessing, even if only to determine the initial conditions of a k-means run, it is in principle conceivable that the change in the transformation step will affect the evaluation of the objective function. Of course, this means, on the one hand, that some parameters are unnecessarily viewed as volatile, and on the other hand, especially in the case of complex pipelines, it is often not intuitively clear which changes will affect how.



There are two methods of evaluating models in data science, Hold-Out and Cross-Validation. To avoid overfitting, both methods use a test set (not seen by the model) to evaluate model performance.

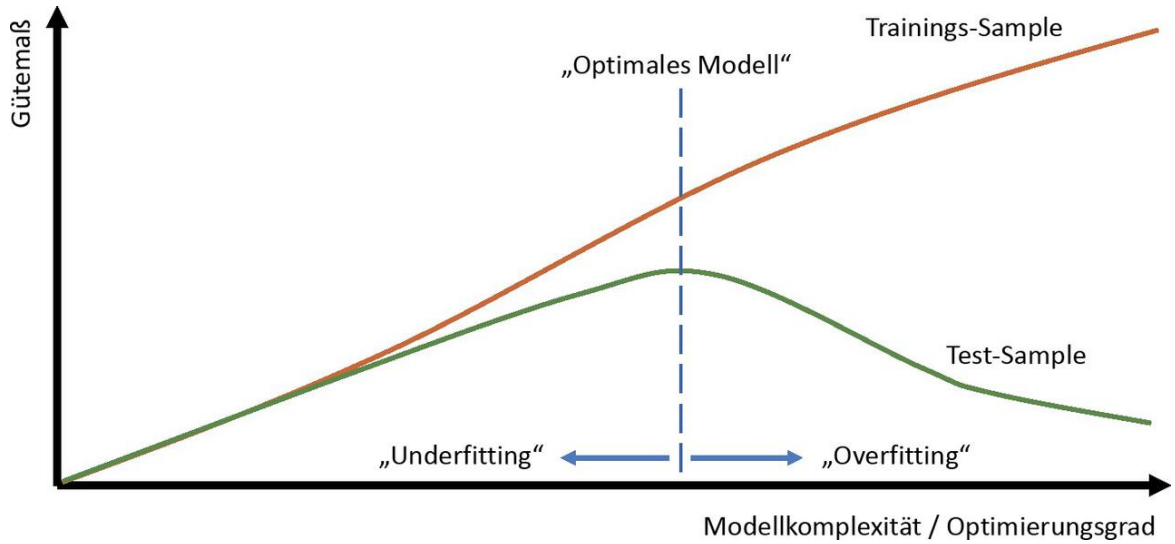
Hold-Out

In this method, the mostly large dataset is randomly divided to three subsets:

- *Training set* is a subset of the dataset used to build predictive models (training model is a process in which ML algorithm is fed with sufficient training data to learn from).
- *Validation set* is a subset of the dataset used to assess the performance of model built in the training phase. It provides a test platform for fine tuning model's parameters and selecting the best-performing model. Not all modelling algorithms need a validation set.
- *Test set* or unseen examples is a subset of the dataset to assess the likely future performance of a model. If a model fit to the training set much better than it fits the test set, overfitting is probably the cause.

Cross-Validation

When only a limited amount of data is available, to achieve an unbiased estimate of the model performance we use k-fold cross-validation. In k-fold cross-validation, we divide the data into k subsets of equal size. We build models k times, each time leaving out one of the subsets from training and use it as the test set. If k equals the sample size, this is called "leave-one-out". Evaluating model performance with the data used for training is not acceptable in data science because it can easily generate overoptimistic and overfitted models, as depicted below:



Explainability, reproducibility and a solid understanding of the limits of the generalisability of the solution are extremely helpful in the assessment.

4. Social and ethical questions, conclusion

Data scientists have a greater responsibility than just penetrating the mathematical, statistical or information technology content. This applies in particular to actuaries who, in accordance with their professional rules, are also obliged to the company and the insured persons they care for. For this reason, we will finish of with the effects of data science on society and on the social framework within which current data science applications are implemented.
Three key aspects of the field in question are:

1. *Artificial intelligence*
2. *Privacy*
3. *Ethical questions.*

As AI takes bigger decision-making role, problems that it has to solve become more difficult and less straight forward. That’s when the various questions arise, while the answers on those can often be polar opposites. For example, if our AI machine was making mistakes of some sort, the logical solution would be to provide it with more information, which would lead the

AI to the “correct solution”. However, the big question is if we have the rights to give certain informations away? And how far we can go with the use of personal data, for the good of science and prospect, until it is considered a privacy breach? These contradictions are the soil of the many concerns with AI, which we can overcome only with sticking to the following:
- Respect for human autonomy: People who interact with AI systems must be able to retain their full and effective self-determination about themselves and be able to participate in the democratic process.
- Loss prevention: AI systems should neither cause nor aggravate damage or otherwise impair humans.
- Fairness: ensuring an even and fair distribution of benefits and costs.
- Explainability: Processes must be transparent, the capabilities and purpose of AI systems must be openly communicated, and decisions must be explainable to those affected.
- Technical robustness and security. AI systems have to be reliable, secure and resistant to attacks.
- Protection of privacy and data quality management. Ensuring privacy and the right to informational self-determination.

To conclude, the central principle of insurance (like any financial service) is trust. The customer relies on receiving the agreed service, even if the time for it is still a long way off. Only then is he ready to pay the price today. Without strictly following the rules from above, AI cannot build much needed trust in the society, and without trust, there is no way forward.

References:

1. Martin Seehafer, Stefan Nörtemann, Jonas Offtermatt, Fabian Transchel, Axel Kiermaier, René Külheim und Wiltrud Weidner: Actuarial Data Science, Maschinelles Lernen in der Versicherung.
2. https://www.kaggle.com/flagma/health-care-cost-analysys-prediction-python/notebook
3. https://en.wikipedia.org/wiki/Simple_linear_regression
4. https://machinelearningmastery.com/dimensionality-reduction-for-machine-learning/
5. https://www.datanovia.com/en/blog/types-of-clustering-methods-overview-and-quick-start-r-code/
6. https://www.weforum.org/agenda/2016/10/top-10-ethical-issues-in-artificial-intelligence/