# Data Science and Machine Learning in Python

Stephan Weyers



Fachhochschule Dortmund
University of Applied Sciences and Arts

# Topics covered in the online lectures

Fachhochschule Dortmund
University of Applied Sciences and Arts

**Part 1: Data Science**

| | Date | Topics covered |
|---|---|---|
| 1 | Apr $13^{th}$ | Course introduction<br>Data Science motivation<br>How to use Jupyter Notebook<br>Python types and lists<br>Loops, if/else, functions |
| 2 | Apr $20^{th}$ | Python tuples, lists, dictionaries<br>Functions<br>Numpy basics, operations<br>Image processing |
| 3 | Apr $27^{th}$ | Pandas Series, DataFrame<br>Pandas basic operations<br>Import/export files |
| 4 | May $4^{th}$ | Principles of data visualization<br>Data cleaning and preparation<br>Join, combine and reshape data |
| 5 | May $11^{th}$ | Volkswohl Bund dataset<br>Data visualization in Python<br>How to write Data Science reports<br>Data aggregation and grouping |

**Part 2: Machine Learning**

| | Date | Topics covered |
|---|---|---|
| 6 | Jun $1^{st}$ | Introduction to supervised learning<br>Classification and regression<br>scikit-learn<br>k-Nearest Neighbors<br>Linear regression (ridge and lasso) |
| 7 | Jun $8^{th}$ | Linear classification models<br>Decision trees<br>Random forests and gradient boosting |
| 8 | Jun $15^{th}$ | Kernel support vector machines<br>Neural networks |
| 9 | Jun $22^{nd}$ | Introduction to unsupervised learning<br>Preprocessing and scaling<br>Dimensionality reduction<br>Principal component analysis |
| 10 | Jun $29^{th}$ | k-means clustering<br>Hierarchical clustering<br>DBSCAN |
| 11 | Jul $6^{th}$ | Representing data<br>Engineering features<br>Model evaluation and improvement<br>Text data analysis |

# Deadlines for Submission and Distribution of Grading

Fachhochschule Dortmund
University of Applied Sciences and Arts

| Student task | Deliverables | Deadline | Work | Share of grade |
|---|---|---|---|---|
| W01 Assignment | Code and results | Apr 26th | Team A | 5.0% |
| W02 Case Study | Code / presentation slides | May 22nd | Team B | 18.0% |
| W02 Case Study | Peer review* | May 31st | Individual | 2.0% |
| W03 Assignment | Code and results | May 29th | Team B | 5.0% |
| W04 Assignment | Code and results | Jun 12th | Team C | 10.0% |
| W05 Assignment | Code and results | Jun 28th | Team D | 7.0% |
| W06 Assignment | Code and results | Jul 8th | Team D | 13.0% |
| W07 Case Study | Code / presentation slides | Jul 17th | Team D | 22.0% |
| W07 Case Study | Peer review* | Jul 31st | Individual | 3.0% |
| DataCamp 1 | Finish course | May 9th | Individual | 2.5% |
| DataCamp 2 | Finish course | May 30th | Individual | 2.5% |
| DataCamp 3 | Finish course | Jun 20th | Individual | 2.5% |
| DataCamp 4 | Finish course | Jul 11th | Individual | 2.5% |

* Peer review is mandatory. Quality of peer review itself is graded. Not providing peer review at all would result in high point deduction

# Agenda for online lecture 11

Fachhochschule Dortmund
University of Applied Sciences and Arts

| Session | Topic | Mode | Materials used | Minutes | End |
|---|---|---|---|---|---|
| 14:30-16:00 | Organizational questions | Q&A | | 10 | 14:40 |
| | OCEAN Big Five | Lecture / Q&A | Lecture 10d notebook | 10 | 14:50 |
| | Text clusters – introduction | Lecture / Q&A | Lecture 11a notebook | 5 | 14:55 |
| | Find and label text clusters | Team work in break-out rooms | Lecture 11a notebook | 25 | 15:20 |
| | Text cluster results | Discussion in main room | Lecture 11a notebook | 10 | 15:30 |
| | Model tuning / evaluation | Lecture / Q&A | Lecture slides | 15 | 15:45 |
| | Course evaluation | Individual work | Evaluation form | 10 | 15:55 |
| 16:10-17:40 | Example churn – intro | Lecture / Q&A | Lecture 11b notebook | 20 | 16:30 |
| | Example churn – Exercise | Joint discussion in main room | Lecture 11b notebook | 40 | 17:10 |
| | Course evaluation | Team work in break-out rooms | 2 stars + 2 wishes | 25 | 17:30 |
| | Farewell | | | 5 | 17:40 |
| 17:50-19:20 | Optional | Optional | Optional | | |

# OCEAN Big Five – DataSet

Fachhochschule Dortmund
University of Applied Sciences and Arts

Dataset Source: kaggle (https://www.kaggle.com/tunguz/big-five-personality-test)

Attribute Information:

The scale was labeled 1=Disagree, 3=Neutral, 5=Agree

E - Surgency or Extraversion

EXT1 I am the life of the party. EXT2 I don't talk a lot. EXT3 I feel comfortable around people. EXT4 I keep in the background. EXT5 I start conversations. EXT6 I have little to say. EXT7 I talk to a lot of different people at parties. EXT8 I don't like to draw attention to myself. EXT9 I don't mind being the center of attention. EXT10 I am quiet around strangers.

N - Emotional Stability or (not) Neuroticism

EST1 I get stressed out easily. EST2 I am relaxed most of the time. EST3 I worry about things. EST4 I seldom feel blue. EST5 I am easily disturbed. EST6 I get upset easily. EST7 I change my mood a lot. EST8 I have frequent mood swings. EST9 I get irritated easily. EST10 I often feel blue.

A - Agreeableness

AGR1 I feel little concern for others. AGR2 I am interested in people. AGR3 I insult people. AGR4 I sympathize with others' feelings. AGR5 I am not interested in other people's problems. AGR6 I have a soft heart. AGR7 I am not really interested in others. AGR8 I take time out for others. AGR9 I feel others' emotions. AGR10 I make people feel at ease.

C - Conscientiousness

CSN1 I am always prepared. CSN2 I leave my belongings around. CSN3 I pay attention to details. CSN4 I make a mess of things. CSN5 I get chores done right away. CSN6 I often forget to put things back in their proper place. CSN7 I like order. CSN8 I shirk my duties. CSN9 I follow a schedule. CSN10 I am exacting in my work.

O - Openness to experience or Intellect or Imagination

OPN1 I have a rich vocabulary. OPN2 I have difficulty understanding abstract ideas. OPN3 I have a vivid imagination. OPN4 I am not interested in abstract ideas. OPN5 I have excellent ideas. OPN6 I do not have a good imagination. OPN7 I am quick to understand things. OPN8 I use difficult words. OPN9 I spend time reflecting on things. OPN10 I am full of ideas.

# OCEAN Big Five – DataSet

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df = pd.read_csv("Lecture_09c_OCEAN_data.csv",sep="\t")
```

```
df
```

| | EXT1 | EXT2 | EXT3 | EXT4 | EXT5 | EXT6 | EXT7 | EXT8 | EXT9 | EXT10 | ... | OPN1 | OPN2 | OPN3 | OPN4 | OPN5 | OPN6 | OPN7 | OPN8 | OPN9 | OPN1( |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 4.0 | 1.0 | 5.0 | 2.0 | 5.0 | 1.0 | 5.0 | 2.0 | 4.0 | 1.0 | ... | 5.0 | 1.0 | 4.0 | 1.0 | 4.0 | 1.0 | 5.0 | 3.0 | 4.0 | 5.( |
| **1** | 3.0 | 5.0 | 3.0 | 4.0 | 3.0 | 3.0 | 2.0 | 5.0 | 1.0 | 5.0 | ... | 1.0 | 2.0 | 4.0 | 2.0 | 3.0 | 1.0 | 4.0 | 2.0 | 5.0 | 3.( |
| **2** | 2.0 | 3.0 | 4.0 | 4.0 | 3.0 | 2.0 | 1.0 | 3.0 | 2.0 | 5.0 | ... | 5.0 | 1.0 | 2.0 | 1.0 | 4.0 | 2.0 | 5.0 | 3.0 | 4.0 | 4.( |
| **3** | 2.0 | 2.0 | 2.0 | 3.0 | 4.0 | 2.0 | 2.0 | 4.0 | 1.0 | 4.0 | ... | 4.0 | 2.0 | 5.0 | 2.0 | 3.0 | 1.0 | 4.0 | 4.0 | 3.0 | 3.( |
| **4** | 3.0 | 3.0 | 3.0 | 3.0 | 5.0 | 3.0 | 3.0 | 5.0 | 3.0 | 4.0 | ... | 5.0 | 1.0 | 5.0 | 1.0 | 5.0 | 1.0 | 5.0 | 3.0 | 5.0 | 5.( |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | .. |
| **874429** | 4.0 | 2.0 | 4.0 | 3.0 | 4.0 | 3.0 | 3.0 | 3.0 | 3.0 | 3.0 | ... | 2.0 | 2.0 | 4.0 | 3.0 | 4.0 | 2.0 | 4.0 | 2.0 | 2.0 | 4.( |
| **874430** | 4.0 | 3.0 | 4.0 | 3.0 | 3.0 | 3.0 | 4.0 | 4.0 | 3.0 | 3.0 | ... | 4.0 | 1.0 | 5.0 | 1.0 | 5.0 | 1.0 | 3.0 | 4.0 | 5.0 | 4.( |
| **874431** | 4.0 | 2.0 | 4.0 | 3.0 | 5.0 | 1.0 | 4.0 | 2.0 | 4.0 | 4.0 | ... | 5.0 | 1.0 | 5.0 | 1.0 | 4.0 | 1.0 | 5.0 | 5.0 | 4.0 | 5.( |
| **874432** | 2.0 | 4.0 | 3.0 | 4.0 | 2.0 | 2.0 | 1.0 | 4.0 | 2.0 | 4.0 | ... | 5.0 | 2.0 | 4.0 | 2.0 | 3.0 | 2.0 | 4.0 | 5.0 | 5.0 | 3.( |
| **874433** | 4.0 | 2.0 | 4.0 | 2.0 | 4.0 | 1.0 | 4.0 | 2.0 | 4.0 | 4.0 | ... | 5.0 | 1.0 | 5.0 | 1.0 | 3.0 | 1.0 | 5.0 | 4.0 | 5.0 | 5.( |

874434 rows × 50 columns

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(df)
df_scaled = scaler.transform(df)
df = pd.DataFrame(df_scaled,columns=df.columns)
directions = [1,-1,1,-1,1,-1,1,-1,1,-1,-1,1,-1,1,-1,-1,-1,-1,-1,-1,-1,1,-1,1,-1,1,-1,1,1,1,1,-1,1,-1,1,-1,1,-1,1,1,1,-1,1,-1,
for j in range(50):
    df.iloc[:,j] = directions[j] * df.iloc[:,j]
```

# OCEAN Big Five – kmeans



```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=5,random_state=10)
kmeans.fit(df)
```

```
KMeans(n_clusters=5, random_state=10)
```

```
print(df.loc[:,"EXT1"].groupby(kmeans.labels_).count())
p_mean = df.groupby(kmeans.labels_).mean().transpose()
import seaborn as sns
plt.figure(figsize = (10,20))
ax = sns.heatmap(p_mean,yticklabels=df.columns.values,annot=True, fmt=".2f",cmap="PiYG")
# Observation: The five types of variables really mostly appear to be "packages"
```

```
0    178189
1    222885
2    181026
3    166348
4    125986
Name: EXT1, dtype: int64
```

# OCEAN Big Five – kmeans



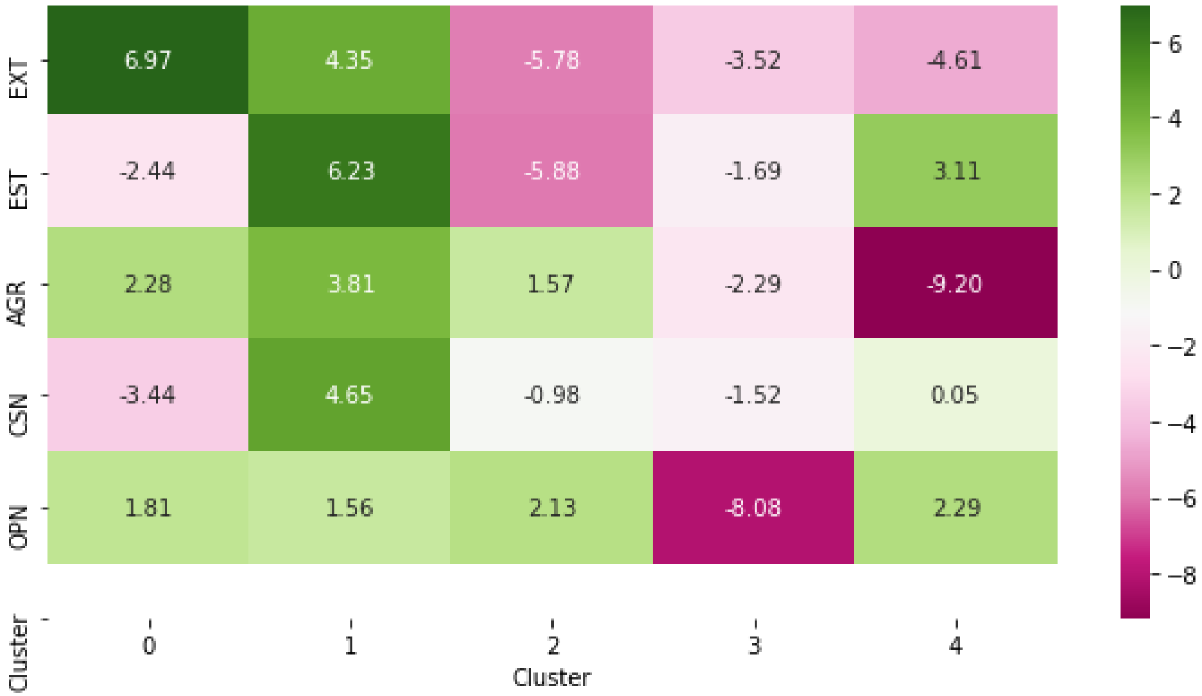
| | | | | | |
|---|---|---|---|---|---|
| EXT1 | 0.76 | 0.37 | -0.59 | -0.29 | -0.49 |
| EXT2 | 0.79 | 0.37 | -0.52 | -0.34 | -0.56 |
| EXT3 | 0.57 | 0.68 | -0.69 | -0.31 | -0.60 |
| EXT4 | 0.72 | 0.47 | -0.70 | -0.29 | -0.46 |
| EXT5 | 0.72 | 0.55 | -0.57 | -0.41 | -0.63 |
| EXT6 | 0.61 | 0.42 | -0.31 | -0.62 | -0.33 |
| EXT7 | 0.76 | 0.49 | -0.63 | -0.35 | -0.57 |
| EXT8 | 0.68 | 0.22 | -0.53 | -0.22 | -0.29 |
| EXT9 | 0.71 | 0.29 | -0.55 | -0.37 | -0.25 |
| EXT10 | 0.65 | 0.50 | -0.67 | -0.32 | -0.42 |
| EST1 | -0.22 | 0.63 | -0.67 | -0.24 | 0.48 |
| EST2 | -0.09 | 0.48 | -0.56 | -0.14 | 0.29 |
| EST3 | -0.21 | 0.49 | -0.63 | -0.11 | 0.48 |
| EST4 | -0.15 | 0.45 | -0.52 | 0.01 | 0.14 |
| EST5 | -0.22 | 0.55 | -0.43 | -0.27 | 0.31 |
| EST6 | -0.30 | 0.68 | -0.62 | -0.25 | 0.45 |
| EST7 | -0.43 | 0.73 | -0.57 | -0.17 | 0.37 |
| EST8 | -0.40 | 0.75 | -0.61 | -0.18 | 0.36 |
| EST9 | -0.28 | 0.73 | -0.49 | -0.22 | 0.10 |
| EST10 | -0.13 | 0.75 | -0.78 | -0.11 | 0.13 |
| AGR1 | 0.13 | 0.31 | 0.25 | -0.28 | -0.73 |
| AGR2 | 0.48 | 0.47 | -0.01 | -0.39 | -0.97 |
| AGR3 | -0.29 | 0.45 | 0.06 | -0.02 | -0.45 |
| AGR4 | 0.24 | 0.34 | 0.37 | -0.19 | -1.23 |
| AGR5 | 0.29 | 0.37 | 0.20 | -0.23 | -1.05 |
| AGR6 | 0.17 | 0.15 | 0.37 | -0.01 | -1.03 |
| AGR7 | 0.42 | 0.51 | 0.02 | -0.34 | -1.06 |
| AGR8 | 0.22 | 0.39 | 0.14 | -0.23 | -0.89 |
| AGR9 | 0.30 | 0.30 | 0.35 | -0.23 | -1.15 |
| AGR10 | 0.32 | 0.52 | -0.17 | -0.37 | -0.63 |

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| AGR1 | 0.13 | 0.31 | 0.25 | -0.28 | -0.73 |
| AGR2 | 0.48 | 0.47 | -0.01 | -0.39 | -0.97 |
| AGR3 | -0.29 | 0.45 | 0.06 | -0.02 | -0.45 |
| AGR4 | 0.24 | 0.34 | 0.37 | -0.19 | -1.23 |
| AGR5 | 0.29 | 0.37 | 0.20 | -0.23 | -1.05 |
| AGR6 | 0.17 | 0.15 | 0.37 | -0.01 | -1.03 |
| AGR7 | 0.42 | 0.51 | 0.02 | -0.34 | -1.06 |
| AGR8 | 0.22 | 0.39 | 0.14 | -0.23 | -0.89 |
| AGR9 | 0.30 | 0.30 | 0.35 | -0.23 | -1.15 |
| AGR10 | 0.32 | 0.52 | -0.17 | -0.37 | -0.63 |
| CSN1 | -0.36 | 0.50 | -0.11 | -0.22 | 0.07 |
| CSN2 | -0.51 | 0.42 | -0.15 | 0.10 | 0.06 |
| CSN3 | -0.20 | 0.32 | 0.17 | -0.43 | 0.04 |
| CSN4 | -0.45 | 0.70 | -0.38 | -0.14 | 0.13 |
| CSN5 | -0.37 | 0.54 | -0.21 | -0.00 | -0.13 |
| CSN6 | -0.48 | 0.54 | -0.19 | -0.04 | 0.06 |
| CSN7 | -0.33 | 0.29 | 0.08 | -0.13 | -0.00 |
| CSN8 | -0.29 | 0.57 | -0.18 | -0.21 | -0.06 |
| CSN9 | -0.28 | 0.43 | -0.06 | -0.05 | -0.20 |
| CSN10 | -0.17 | 0.36 | 0.04 | -0.41 | 0.08 |
| OPN1 | 0.14 | 0.14 | 0.22 | -0.85 | 0.34 |
| OPN2 | 0.09 | 0.26 | 0.14 | -0.87 | 0.37 |
| OPN3 | 0.30 | 0.00 | 0.39 | -0.80 | 0.07 |
| OPN4 | 0.14 | 0.16 | 0.24 | -0.76 | 0.19 |
| OPN5 | 0.27 | 0.31 | 0.01 | -0.91 | 0.28 |
| OPN6 | 0.24 | 0.18 | 0.24 | -0.83 | 0.10 |
| OPN7 | 0.05 | 0.32 | 0.02 | -0.77 | 0.34 |
| OPN8 | 0.19 | -0.04 | 0.25 | -0.70 | 0.38 |
| OPN9 | 0.05 | 0.01 | 0.43 | -0.55 | 0.02 |
| OPN10 | 0.34 | 0.23 | 0.20 | -1.05 | 0.20 |

# OCEAN Big Five – kmeans



```
df2 = pd.DataFrame()
df2["EXT"] = df.loc[:,"EXT1":"EXT10"].sum(axis=1)
df2["EST"] = df.loc[:,"EST1":"EST10"].sum(axis=1)
df2["AGR"] = df.loc[:,"AGR1":"AGR10"].sum(axis=1)
df2["CSN"] = df.loc[:,"CSN1":"CSN10"].sum(axis=1)
df2["OPN"] = df.loc[:,"OPN1":"OPN10"].sum(axis=1)
df2["Cluster"] = kmeans.labels_
p_mean = df2.groupby("Cluster").mean().transpose()
import seaborn as sns
plt.figure(figsize = (10,5))
ax = sns.heatmap(p_mean,yticklabels=df2.columns.values,annot=True, fmt=".2f",cmap="PiYG")
```



# OCEAN Big Five – kmeans

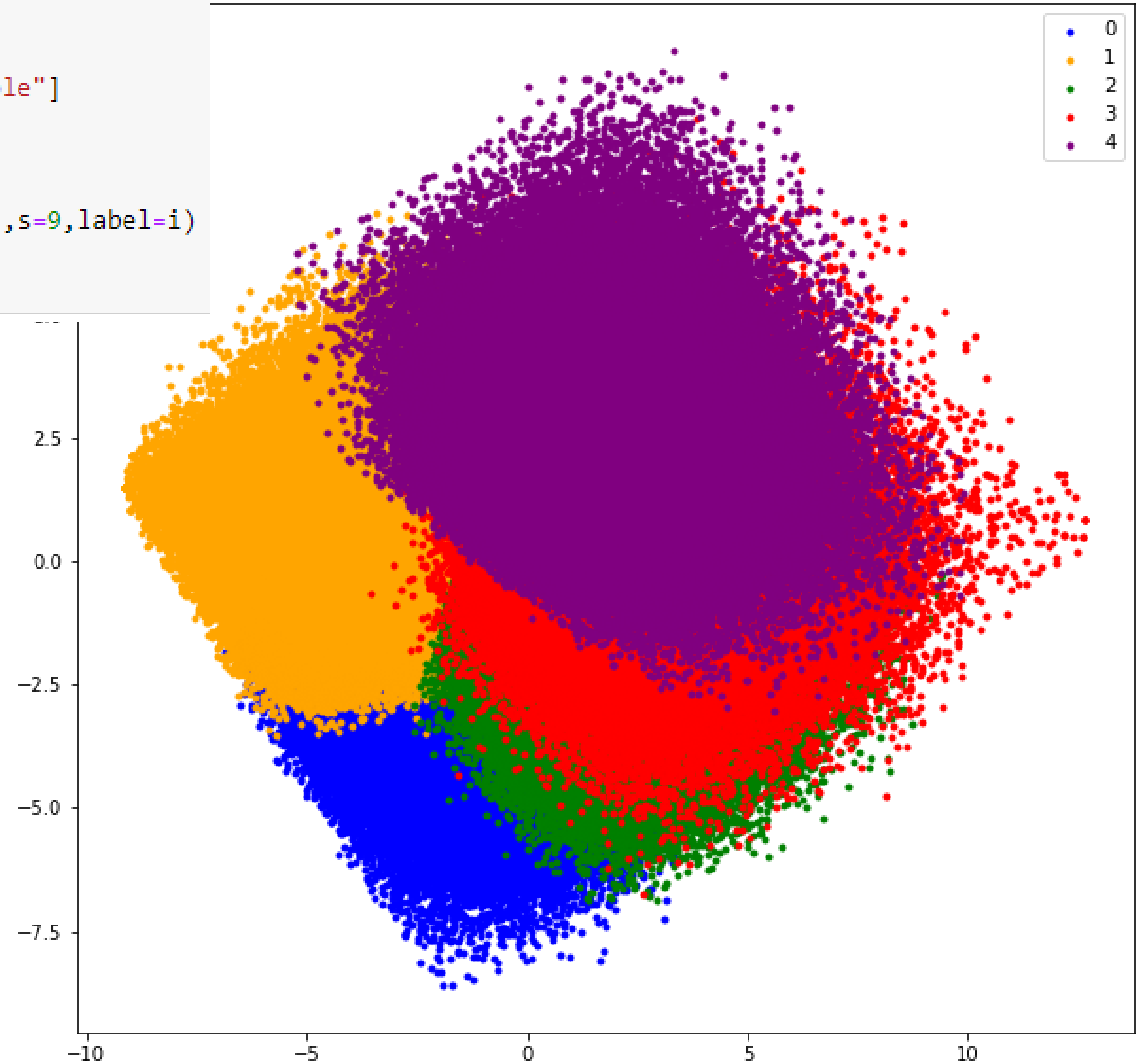


# OCEAN Big Five – PCA and kmeans



```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca.fit(df)
projected = pca.transform(df)

colors = ["blue","orange","green","red","purple"]
plt.figure(figsize=(10,10))
for i in range(5):
    Z = projected[kmeans.labels_ == i]
    plt.scatter(Z[:,0],Z[:,1],color=colors[i],s=9,label=i)
plt.legend(loc="upper right")
plt.show()
```



# OCEAN Big Five – Hierarchical Clustering



```
from sklearn.cluster import AgglomerativeClustering
agg = AgglomerativeClustering(n_clusters=5)
agg.fit(df)
```

```
---------------------------------------------------------------------------
MemoryError                               Traceback (most recent call last)

MemoryError: Unable to allocate 2.78 TiB for an array with shape (382316972961,) and data type float64
```

# Text Cluster – DataSet

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
sentences[12]
```

```
'crude oil prices back above $50 cold weather across parts of the united states and much of europe has pushed us crude oil p
rices above $50 a barrel for the first time in almost three months.  freezing temperatures and heavy snowfall have increased
demand for heating fuel in the us  where stocks are low. fresh falls in the value of the dollar helped carry prices above th
e $50 mark for the first time since november. a barrel of us crude oil closed up $2.80 to $51.15 in new york on tuesday. ope
c members said on tuesday that it saw no reason to cut its output.  although below last year s peak of $55.67 a barrel  whic
h was reached in october  prices are now well above 2004 s average of $41.48.  brent crude also rose in london trading  addi
ng $1.89 to $48.62 at the close. much of western europe and the north east of america has been shivering under unseasonably
low temperatures in recent days. the decline in the us dollar to a five-week low against the euro has also served to inflate
prices.  the dollar moved sharply overnight and oil is following it   said chris furness  senior market strategist at 4cast.
if the dollar continues to weaken  oil will be obviously higher.   several opec members said a cut in production was unlikel
y  citing rising prices and strong demand for oil from asia.  i agree that we do not need to cut supply if the prices are as
much as this   fathi bin shatwan  libya s oil minister  told reuters.  i do not think we need to cut unless the prices are f
alling below $35 a barrel   he added. opec closely watches global stocks to ensure that there is not an excessive supply in
the market. the arrival of spring in the northern hemisphere will focus attention on stockpiles of us crude and gasoline  wh
ich are up to 9% higher than at this time last year. heavy stockpiles could help force prices lower when demand eases.\n'
```

```
sentences[1215]
```

```
'wru proposes season overhaul the welsh rugby union wants to restructure the northern hemisphere season into four separate b
locks.  the season would start with the celtic league in october  followed by the heineken cup in february and march  and th
e six nations moved to april and may. after a nine week break  the wru then proposes a two-month period of away and home int
ernational matches. wru chairman david pickering said the structure would end problems of player availability for club and c
ountry. he added:  we feel sure that spectator interest would respond to the impetus of high intensity rugby being played co
ntinuously rather than the fragmented timetable currently in operation.  equally  we suspect that the sponsors would prefer
the sustained interest in a continuous tournament and hopefully  the broadcasters would also enjoy increased exposure.  movi
ng the six nations from its traditional february beginning should also ensure better weather conditions and  stimulate great
er interest in the games and generally provide increased skills and competition and attract greater spectator viewing   pick
ering argued. the plan will be put before the international rugby board next month  where four other plans drawn up by indep
endent consultants for a global integrated season will also be discussed. pickering added:  it s very early days and there a
re a number of caveats associated with it - not least the revenue from the broadcasters  which is extremely important.  we v
e got a good plan and one which should be judged on its merits.\n'
```

# Text Cluster – Preprocessing



```python
## full_remove takes a string x and a list of characters removal_list
## returns x with all the characters in removal_list replaced by ' '
def full_remove(x, removal_list):
    for w in removal_list:
        x = x.replace(w, ' ')
    return x
## Remove digits
digits = [str(x) for x in range(10)]
digit_less = [full_remove(x, digits) for x in sentences]
## Remove punctuation
punc_less = [full_remove(x, list(string.punctuation)) for x in digit_less]
## Make everything lower-case
sents_lower = [x.lower() for x in punc_less]
## Define our stop words
stop_set = set(['the', 'a', 'an', 'i', 'he', 'she', 'they', 'to', 'of', 'it', 'from',
               "and", "in", "is", "for", "that", "on", "was", "be", "with",
               "as", "has", "have", "at", "are", "but", "will", "by",
               "this", "which", 'then', 'him', 'going', 'any', 'while', 'before',
                'because', 'should', 'many', 'three', 'very', 'made', 'such', 'get',
                'told', 'being', 'just', 'best', 'no', 'into', 'some', 'what',
                'so', 'now', 'two', 'when', 'over', 'could', 'if', 'you', 'all', 'than',
                'or', 'can', 'about', 'there', 'one', 'us', 'new', 'who', 'also', 'were',
                'its', 'their', 'been', 'had', 'would', 'his', 'not', 'we','said','after',
                'out','more','up','first','last','like','make','only','do','other','her',
                'year','years','them','against','back','next','bbc','well','set','number',
                'take','most', 'way', 'added', 'may', 'says', 'my', 'our', 'off', 'good',
                'how', 'down', 'still', 'those', 'much', 'uk', 'england', 'go', 'since', 'say'])
## Remove stop words
sents_split = [x.split() for x in sents_lower]
sents_processed = [" ".join(list(filter(lambda a: a not in stop_set, x))) for x in sents_split]
```

# Text Cluster – After Preprocessing

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
n = 12
print(sentences[n])
print(sents_processed[n])
```

```
crude oil prices back above $50 cold weather across parts of the united states and much of europe has pushed us crude oil pr
ices above $50 a barrel for the first time in almost three months.  freezing temperatures and heavy snowfall have increased
demand for heating fuel in the us  where stocks are low. fresh falls in the value of the dollar helped carry prices above th
e $50 mark for the first time since november. a barrel of us crude oil closed up $2.80 to $51.15 in new york on tuesday. ope
c members said on tuesday that it saw no reason to cut its output.  although below last year s peak of $55.67 a barrel  whic
h was reached in october  prices are now well above 2004 s average of $41.48.  brent crude also rose in london trading  addi
ng $1.89 to $48.62 at the close. much of western europe and the north east of america has been shivering under unseasonably
low temperatures in recent days. the decline in the us dollar to a five-week low against the euro has also served to inflate
prices.  the dollar moved sharply overnight and oil is following it   said chris furness  senior market strategist at 4cast.
if the dollar continues to weaken  oil will be obviously higher.   several opec members said a cut in production was unlikel
y  citing rising prices and strong demand for oil from asia.  i agree that we do not need to cut supply if the prices are as
much as this   fathi bin shatwan  libya s oil minister  told reuters.  i do not think we need to cut unless the prices are f
alling below $35 a barrel   he added. opec closely watches global stocks to ensure that there is not an excessive supply in
the market. the arrival of spring in the northern hemisphere will focus attention on stockpiles of us crude and gasoline  wh
ich are up to 9% higher than at this time last year. heavy stockpiles could help force prices lower when demand eases.

crude oil prices above cold weather across parts united states europe pushed crude oil prices above barrel time almost month
s freezing temperatures heavy snowfall increased demand heating fuel where stocks low fresh falls value dollar helped carry
prices above mark time november barrel crude oil closed york tuesday opec members tuesday saw reason cut output although bel
ow s peak barrel reached october prices above s average brent crude rose london trading adding close western europe north ea
st america shivering under unseasonably low temperatures recent days decline dollar five week low euro served inflate prices
dollar moved sharply overnight oil following chris furness senior market strategist cast dollar continues weaken oil obvious
ly higher several opec members cut production unlikely citing rising prices strong demand oil asia agree need cut supply pri
ces fathi bin shatwan libya s oil minister reuters think need cut unless prices falling below barrel opec closely watches gl
obal stocks ensure excessive supply market arrival spring northern hemisphere focus attention stockpiles crude gasoline high
er time heavy stockpiles help force prices lower demand eases
```

# Text Cluster – Bag of Words Representation



```
## Transform to bag of words representation.
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(min_df = 20, max_df=100000, max_features = None)
data_features = vectorizer.fit_transform(sents_processed)
X = data_features.toarray()
vocabulary = vectorizer.get_feature_names()
df = pd.DataFrame(X,columns=vectorizer.get_feature_names())
X.shape, df.shape
```

```
((2225, 3258), (2225, 3258))
```

```
# Show top 100 words that appear most often
df.sum(axis=0).sort_values(ascending=False).head(50).index
```

```
Index(['mr', 'people', 'time', 'world', 'government', 'bn', 'film', 'game',
       'music', 'labour', 'market', 'company', 'home', 'election', 'party',
       'games', 'win', 'work', 'firm', 'second', 'top', 'blair', 'show', 'won',
       'think', 'week', 'use', 'million', 'part', 'play', 'technology',
       'minister', 'high', 'public', 'want', 'between', 'under', 'mobile',
       'see', 'british', 'did', 'five', 'country', 'used', 'european', 'tv',
       'players', 'through', 'news', 'end'],
      dtype='object')
```

# Text Cluster – Alternative Representations



**term frequency–inverse document frequency (tf–idf):** Give high weight terms that appear often in a particular document, but not in many documents in the corpus

**n-grams:**

```
bards_words:
['The fool doth think he is wise,',
 'but the wise man knows himself to be a fool']

Vocabulary:
['be fool', 'but the', 'doth think', 'fool doth', 'he is', 'himself to',
 'is wise', 'knows himself', 'man knows', 'the fool', 'the wise',
 'think he', 'to be', 'wise man']
```

**Lemmatization and Stemming:**

```
compare_normalization(u"Our meeting today was worse than yesterday, "
                       "I'm scared of meeting the clients tomorrow.")

Lemmatization:
['our', 'meeting', 'today', 'be', 'bad', 'than', 'yesterday', ',', 'i', 'be',
 'scared', 'of', 'meet', 'the', 'client', 'tomorrow', '.']
Stemming:
['our', 'meet', 'today', 'wa', 'wors', 'than', 'yesterday', ',', 'i', "'m",
 'scare', 'of', 'meet', 'the', 'client', 'tomorrow', '.']
```



Source: Müller, A. C., Guido, S. (2016) Introduction to Machine Learning with Python : A Guide for Data Scientists, O'Reilly Media

# Text Cluster – kmeans



```
# Perform kmeans clustering
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4,random_state=20)
kmeans.fit(X)
# Show number of cases per cluster
df["CLUSTER"] = kmeans.labels_
df.iloc[:,-2:].groupby("CLUSTER").count().transpose()
```

| CLUSTER | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| zealand | 5 | 1942 | 277 | 1 |

```
# Display top 50 words appearing most often in cluster n
# as well as the m-th sentence in cluster n
n = 2
m = 1
print(df[df["CLUSTER"]==n].drop("CLUSTER",axis=1).sum(axis=0).sort_values(ascending=False).head(50).index)
i = df[df["CLUSTER"]==n].index[m]
sentences[i]
```

```
Index(['mr', 'labour', 'election', 'blair', 'party', 'government', 'people',
       'brown', 'minister', 'howard', 'prime', 'tory', 'tax', 'public',
       'plans', 'leader', 'chancellor', 'tories', 'britain', 'campaign',
       'secretary', 'home', 'general', 'bn', 'tony', 'time', 'lib', 'michael',
       'country', 'under', 'between', 'world', 'budget', 'spokesman',
       'kennedy', 'liberal', 'saying', 'lord', 'part', 'mps', 'eu', 'think',
       'vote', 'issue', 'british', 'conservative', 'want', 'claims',
       'spending', 'did'],
      dtype='object')
```

```
'howard hits back at mongrel jibe michael howard has said a claim by peter hain that the tory leader is acting like an  atta
ck mongrel  shows labour is  rattled  by the opposition.  in an upbeat speech to his party s spring conference in brighton
he said labour s campaigning tactics proved the tories were hitting home. mr hain made the claim about tory tactics in the a
nti-terror bill debate.  something tells me that someone  somewhere out there is just a little bit rattled   mr howard said.
mr hain  leader of the commons  told bbc radio four s today programme that mr howard s stance on the government s anti-terro
rism legislation was putting the country at risk. he then accused the tory leader of behaving like an  attack mongrel  and
playing opposition for opposition sake .  mr howard told his party that labour would  do anything  say anything  claim anyth
```

# Text Cluster – Hierarchical Clustering



```
# Perform hierarchical clustering
from sklearn.cluster import AgglomerativeClustering
agg = AgglomerativeClustering(n_clusters=4,linkage="complete")
agg.fit(X)
# Show number of cases per cluster
df["CLUSTER"] = agg.labels_
df.iloc[:,-2:].groupby("CLUSTER").count().transpose()
```

| CLUSTER | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| **zealand** | 2 | 1 | 2221 | 1 |

```
# Display top 50 words appearing most often in cluster n
# as well as the m-th sentence in cluster n
n = 1
m = 0
print(df[df["CLUSTER"]==n].drop("CLUSTER",axis=1).sum(axis=0).sort_values(ascending=False).head(50).index)
i = df[df["CLUSTER"]==n].index[m]
sentences[i]
```

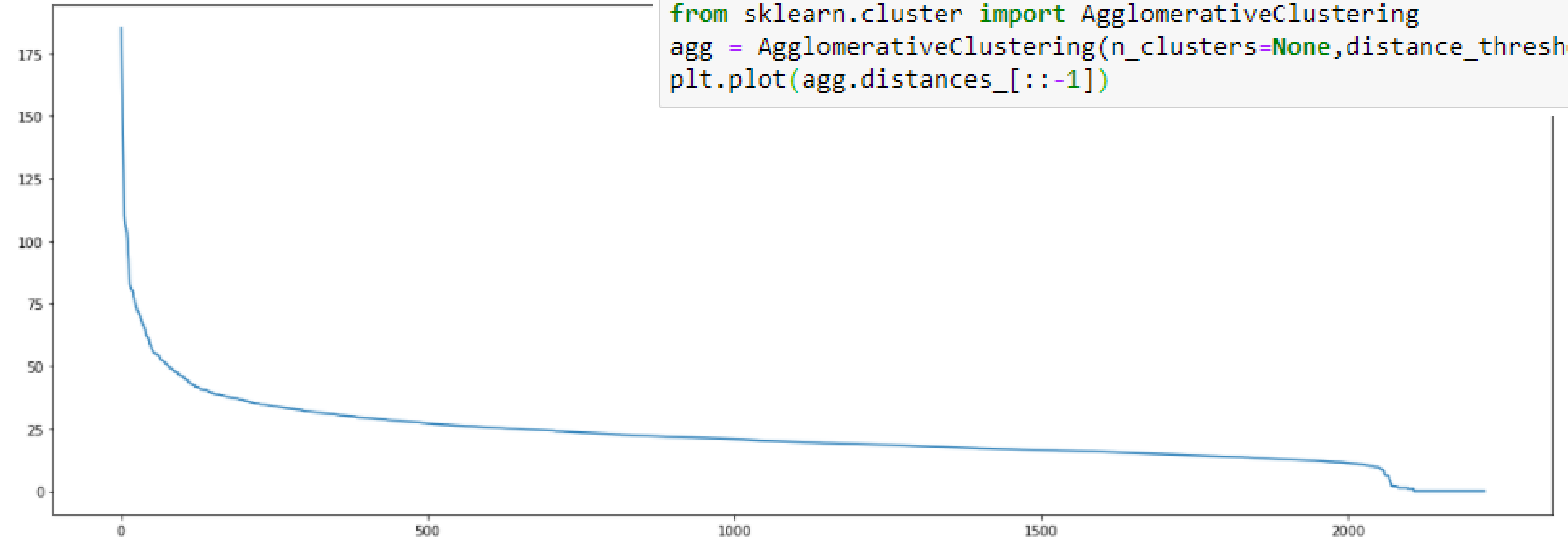
```
Index(['people', 'time', 'mr', 'world', 'film', 'game', 'bn', 'music',
       'market', 'company', 'government', 'games', 'firm', 'second', 'top',
       'win', 'show', 'won', 'million', 'technology', 'mobile', 'play', 'use',
       'players', 'tv', 'high', 'home', 'work', 'week', 'sales', 'five', 'end',
       'half', 'used', 'part', 'however', 'through', 'net', 'four', 'want',
       'growth', 'six', 'european', 'both', 'group', 'think', 'see', 'old',
       'between', 'did'],
      dtype='object')
```

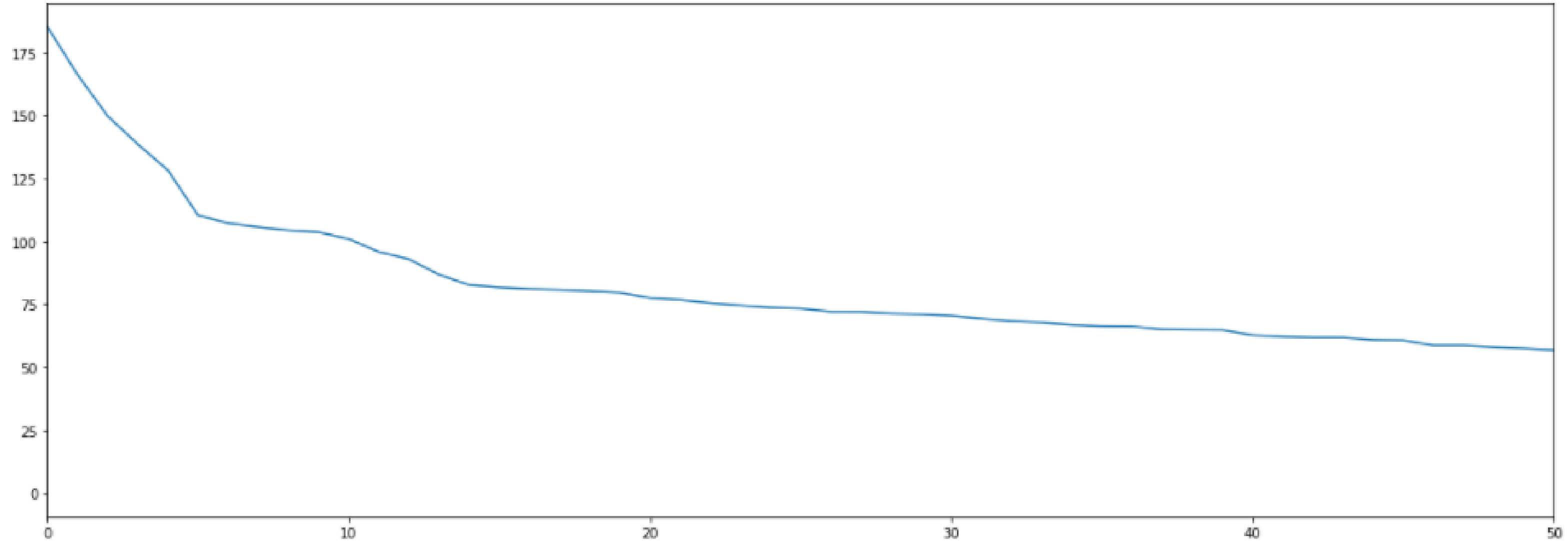
```
'tv future in the hands of viewers with home theatre systems  plasma high-definition tvs  and digital video recorders moving
into the living room  the way people watch tv will be radically different in five years  time.  that is according to an expe
rt panel which gathered at the annual consumer electronics show in las vegas to discuss how these new technologies will impa
ct one of our favourite pastimes. with the us leading the trend  programmes and other content will be delivered to viewers v
ia home networks  through cable  satellite  telecoms companies  and broadband service providers to front rooms and portable
devices.  one of the most talked-about technologies of ces has been digital and personal video recorders (dvr and pvr). thes
e set-top boxes  like the us s tivo and the uk s sky+ system  allow people to record  store  play  pause and forward wind tv
programmes when they want.  essentially  the technology allows for much more personalised tv. they are also being built-in t
```

# Text Cluster – Elbow graph



```
from sklearn.cluster import AgglomerativeClustering
agg = AgglomerativeClustering(n_clusters=None,distance_threshold=0).fit(X)
plt.plot(agg.distances_[::-1])
```



# Text Cluster – Exercise

Fachhochschule Dortmund
University of Applied Sciences and Arts

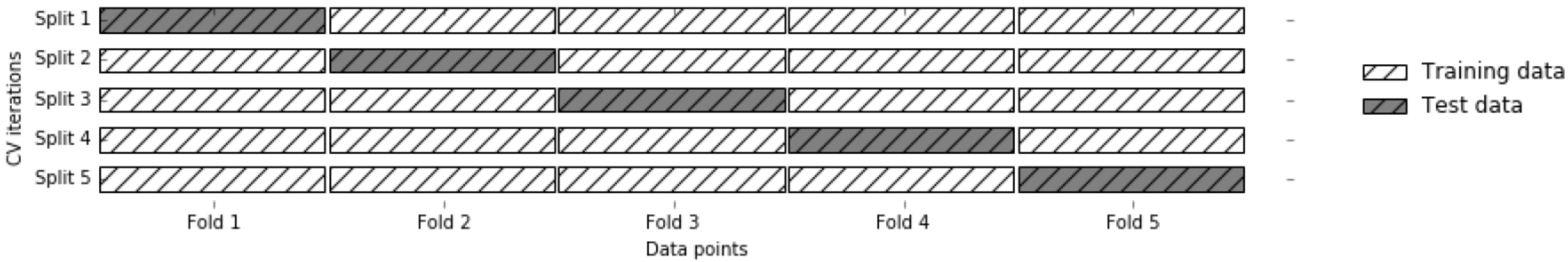
Find a good cluster solution and give appropriate names to the resulting clusters!

- Change the number of clusters in kmeans and hierarchical clustering
- Change the linkage method
- Potentially change the stopwords or the parameters in CountVectorizer
- Explore the clusters by looking at the top 50 words in each cluster and sample sentences

How many distinct meaningful clusters do you find?

# Cross Validation



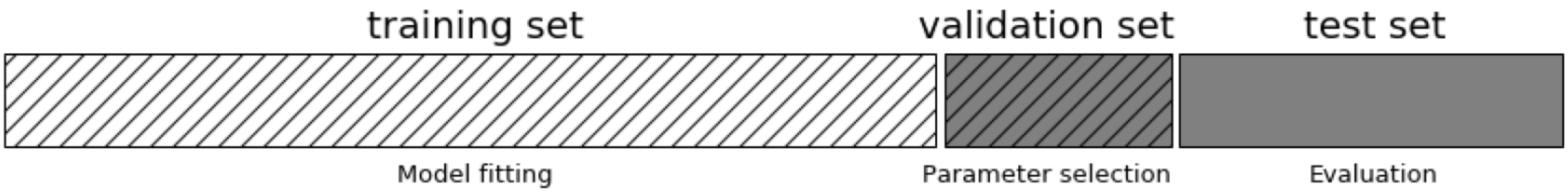





Source: Müller, A. C., Guido, S. (2016) Introduction to Machine Learning with Python : A Guide for Data Scientists, O'Reilly Media

# Parameter Selection








Source: Müller, A. C., Guido, S. (2016) Introduction to Machine Learning with Python : A Guide for Data Scientists, O'Reilly Media

# Construction of New „Smart“ Variables



## Examples

**Scaling / Rescaling**

- Standardize (mean 0, variance 1)
- Min/Max scaling
- Logarithm, square root, inverse,…

**Ratios**

- Sales above/below average
- Average sales per customer
- Purchases per view
- Share of sales in category X

**Categorization**

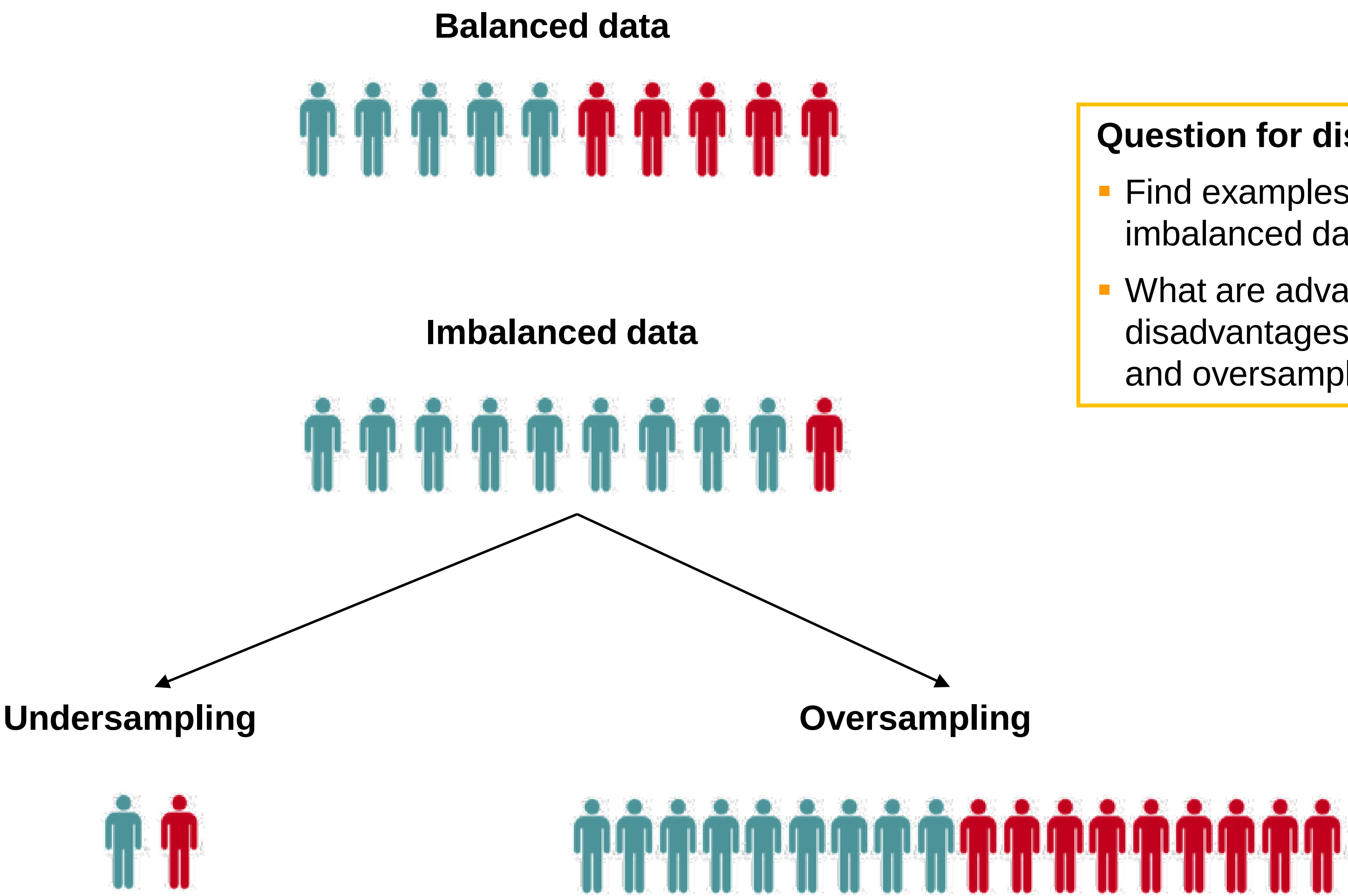
- Binning of metric variables
- Transform zip code to area (East, West, South, North)
- Combine two categorical variables (e.g. area with income group)

**Segmentation / dimensionality reduction**

- Result of cluster analysis (k-means, hierarchical clustering,…) as new variable
- New variables through principal component analysis, factor analysis,…

# Balanced vs. Imbalanced Data






**Question for discussion**

- Find examples for balanced and imbalanced datasets.
- What are advantages and disadvantages of undersampling and oversampling?



Source: Prof. Dr. Michael Bücker – Data Mining

# Change probability threshold



predict_proba

[0.78 , 0.22]

[0.99 , 0.01]

[0.41 , 0.59]

[0.02 , 0.98]

[0.67 , 0.33]

[0.11 , 0.89]

# Precision and Recall



**Confusion matrix**

| | | Predicted churn | |
|---|---|---|---|
| | | No | Yes |
| Actual churn | No | True Negative | False Positive |
| | Yes | False Negative | True Positive |

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\mathrm{F}_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

# Example Churn – Dataset



**3333 rows (customers)**

**21 variables:**

- State: customer residence, e.g. OH or NJ
- Account Length: number of days this account has been active
- Area Code: 3-digit area code of customer's phone number
- Phone: remaining seven-digit phone number
- Int'l Plan: customer has international calling plan: yes/no
- VMail Plan: customer has voice mail feature: yes/no
- VMail Message: average number of voice mail messages per month
- Day Mins: total number of calling minutes used during the day
- Day Calls: total number of calls placed during the day
- Day Charge: billed cost of daytime calls
- Eve Mins, Eve Calls, Eve Charge: same for evening calls
- Night Mins, Night Calls, Night Charge: same for night calls
- Intl Mins, Intl Calls, Intl Charge: same for international calls
- CustServ Calls: number of calls placed to Customer Service
- Churn: customer left the service: true/false

# Example Churn – Imbalanced Data



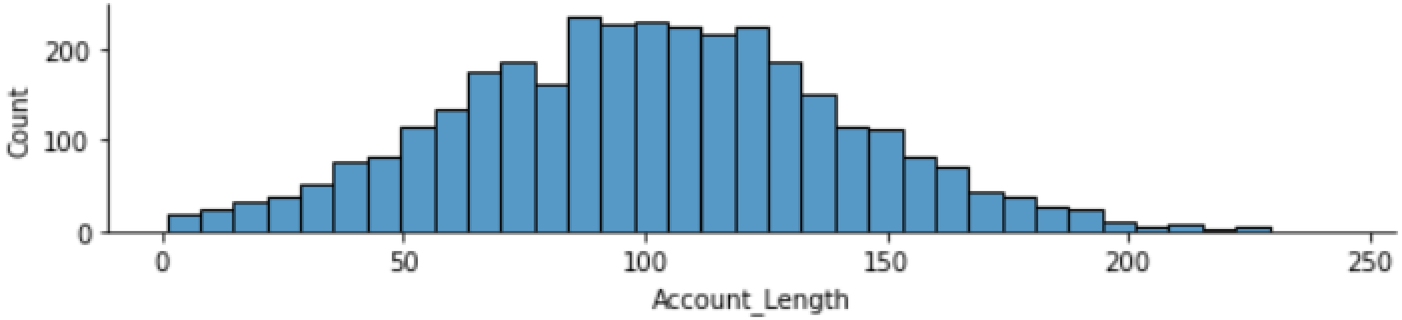
```python
print(telco['Churn'].value_counts())
```

```
no      2850
yes      483
Name: Churn, dtype: int64
```

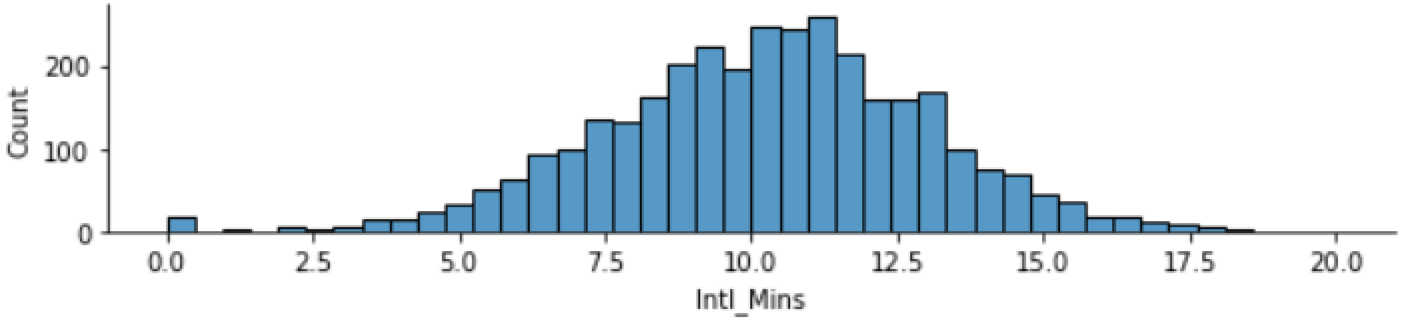
# Example Churn – Distributions

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
sns.displot(data=telco, x='Account_Length',kind="hist",height=2,aspect=4)
plt.show()
```



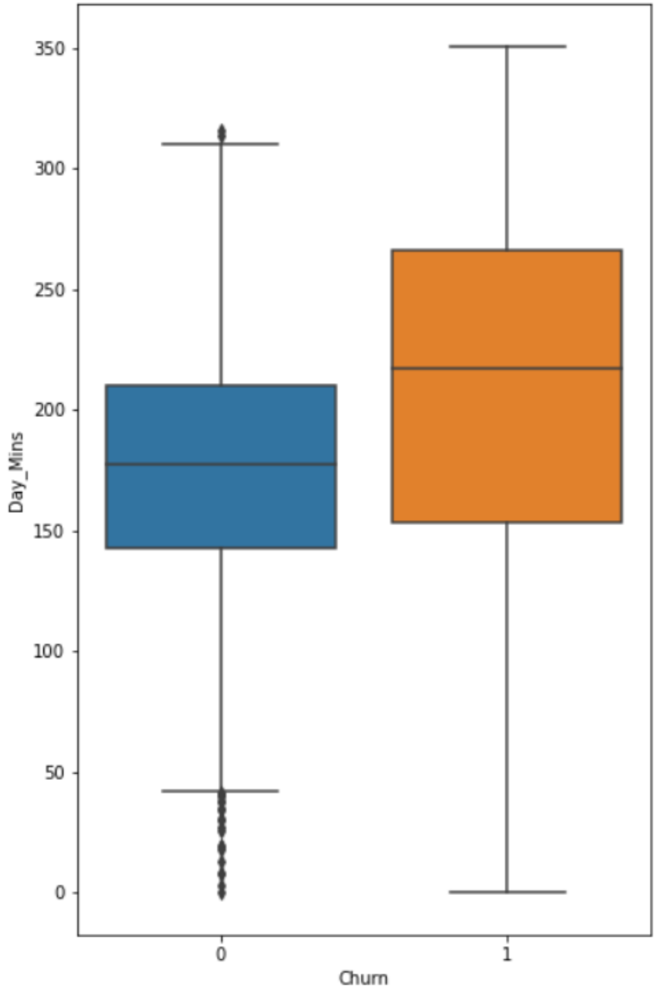

```
sns.displot(data=telco, x='Intl_Mins',kind="hist",height=2,aspect=4)
plt.show()
```



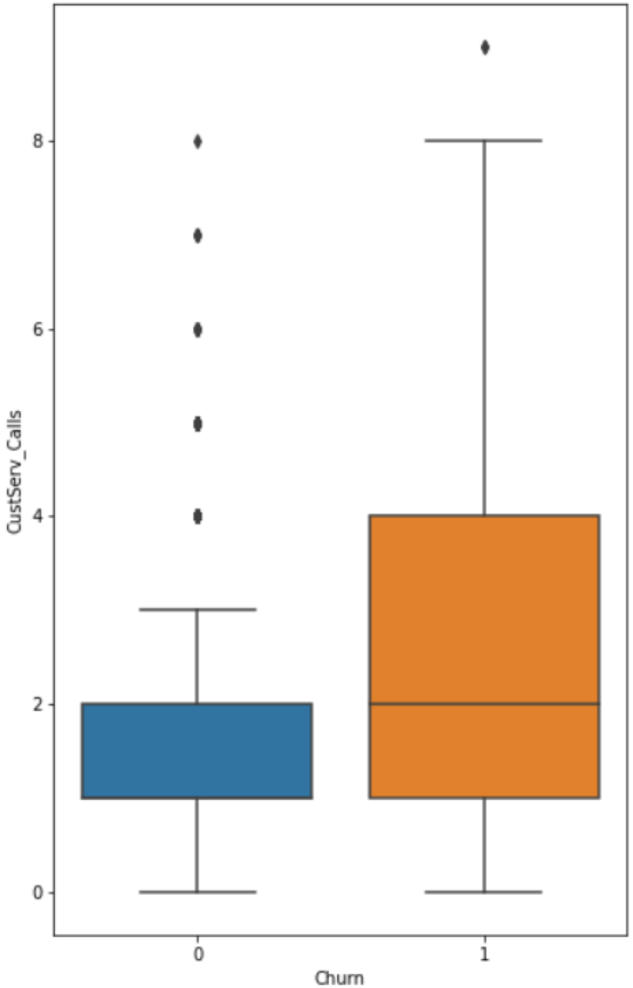
# Example Churn – Boxplots



```
plt.figure(figsize=(6, 10))
sns.boxplot(x = 'Churn', y = 'Day_Mins', data = telco)
plt.show()
```




```
plt.figure(figsize=(6, 10))
sns.boxplot(x = 'Churn', y = 'CustServ_Calls', data = telco)
plt.show()
```



# Example Churn – Categorical to Numbers

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
telco.dtypes
```

```
Account_Length        int64
Vmail_Message         int64
Day_Mins            float64
Eve_Mins            float64
Night_Mins          float64
Intl_Mins           float64
CustServ_Calls        int64
Churn                object
Intl_Plan            object
Vmail_Plan           object
Day_Calls             int64
Day_Charge          float64
Eve_Calls             int64
Eve_Charge          float64
Night_Calls           int64
Night_Charge        float64
Intl_Calls            int64
Intl_Charge         float64
State                object
Area_Code             int64
Phone                object
dtype: object
```

```
telco["Intl_Plan"].value_counts()
```

```
no     3010
yes     323
Name: Intl_Plan, dtype: int64
```

```
telco["Intl_Plan"].replace({"yes":1, "no":0},inplace=True)
telco["Churn"].replace({"yes":1, "no":0},inplace=True)
telco["Vmail_Plan"].replace({"yes":1, "no":0},inplace=True)
```

# Example Churn – Categorical to Dummies



```python
df = pd.get_dummies(telco["State"])
telco.drop("State",axis=1,inplace=True)
df
```

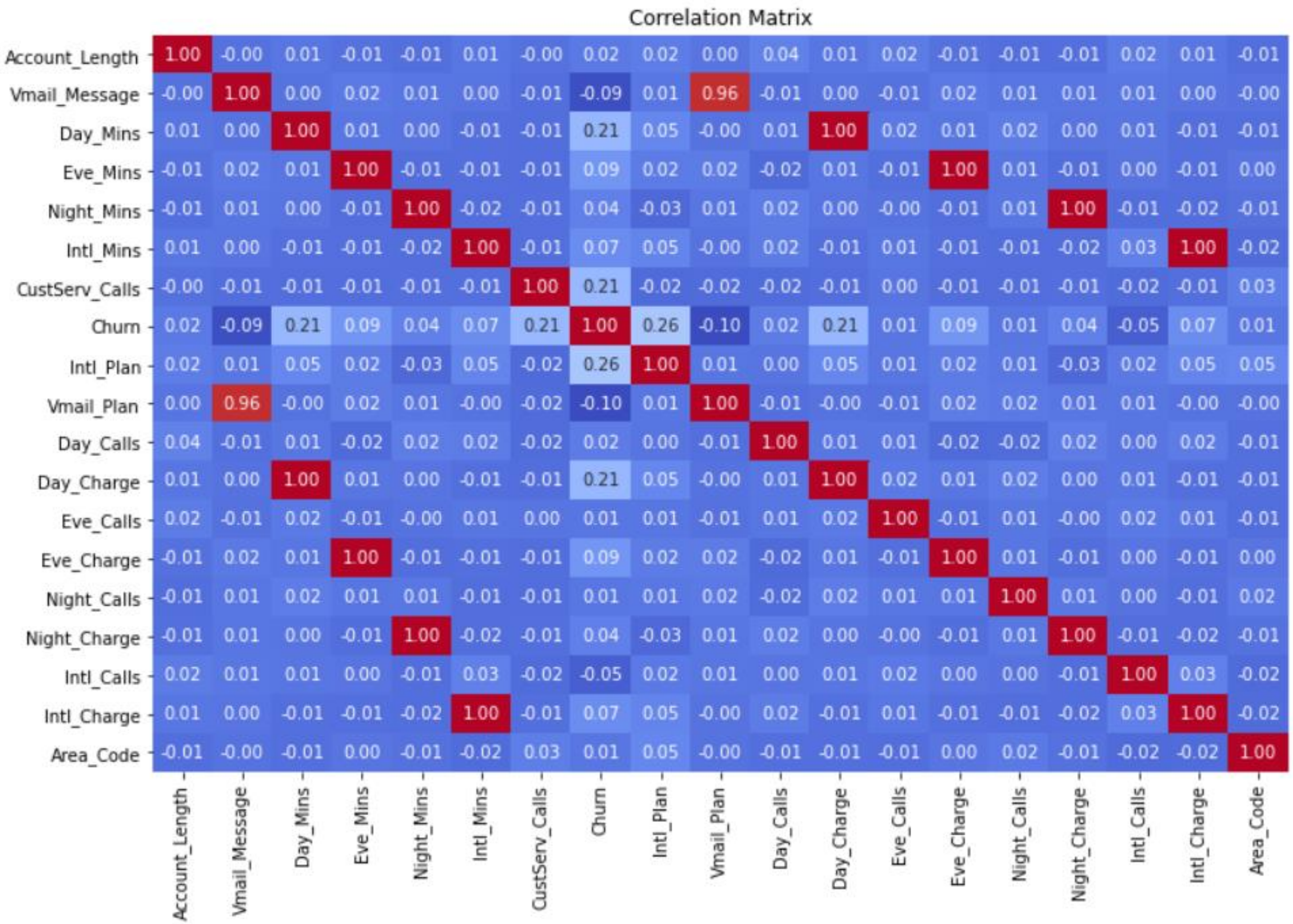
| | AK | AL | AR | AZ | CA | CO | CT | DC | DE | FL | ... | SD | TN | TX | UT | VA | VT | WA | WI | WV | WY |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **3** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **4** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **3328** | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **3329** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| **3330** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **3331** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **3332** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

3333 rows × 51 columns

```python
telco = pd.merge(telco,df,left_index=True,right_index=True,how="inner")
```

# Example Churn – Correlations

```
plt.figure(figsize=(12, 8))
sns.heatmap(cbar=False,annot=True,fmt=".2f",data=telco.drop("State",axis=1).corr(),cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```



# Example Churn – Redundant Variables



```
telco["Vmail_Message"].groupby(telco["Vmail_Plan"]).describe()
```

| Vmail_Plan | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **0** | 2411.0 | 0.000000 | 0.000000 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **1** | 922.0 | 29.277657 | 7.559027 | 4.0 | 24.0 | 29.0 | 34.0 | 51.0 |

```
telco["Day_Mins"] / telco["Day_Charge"]
```

```
0       5.881961
1       5.882781
2       5.882069
3       5.882122
4       5.882145
          ...
3328    5.883239
3329    5.881904
3330    5.881588
3331    5.881706
3332    5.882058
Length: 3333, dtype: float64
```

# Example Churn – New Variables



```
telco["Total_Charge"] = telco["Day_Charge"] + telco["Eve_Charge"] + telco["Night_Charge"] + telco["Intl_Charge"]
telco["Total_Mins"] = telco["Day_Mins"] + telco["Eve_Mins"] + telco["Night_Mins"] + telco["Intl_Mins"]
telco["Avg_Rate"] = telco["Total_Charge"] / telco["Total_Mins"]
telco["Avg_Rate"].describe()
```

```
count    3333.000000
mean        0.100354
std         0.008440
min         0.066950
25%         0.094893
50%         0.100385
75%         0.106056
max         0.129791
Name: Avg_Rate, dtype: float64
```

# Example Churn – Dataset after preprocessing

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
telco.drop("Phone",axis=1,inplace=True)
```

```
print(telco.columns)
```

```
Index(['Account_Length', 'Vmail_Message', 'Day_Mins', 'Eve_Mins', 'Night_Mins',
       'Intl_Mins', 'CustServ_Calls', 'Churn', 'Intl_Plan', 'Vmail_Plan',
       'Day_Calls', 'Day_Charge', 'Eve_Calls', 'Eve_Charge', 'Night_Calls',
       'Night_Charge', 'Intl_Calls', 'Intl_Charge', 'Area_Code',
       'Total_Charge', 'Total_Mins', 'Avg_Rate', 'AK', 'AL', 'AR', 'AZ', 'CA',
       'CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA', 'ID', 'IL', 'IN', 'KS',
       'KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS', 'MT', 'NC', 'ND',
       'NE', 'NH', 'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA', 'RI', 'SC',
       'SD', 'TN', 'TX', 'UT', 'VA', 'VT', 'WA', 'WI', 'WV', 'WY'],
      dtype='object')
```

# Example Churn – First Quick Classification

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
X = telco.drop("Churn",axis=1)
y = np.array(telco["Churn"])
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, random_state=30)
```

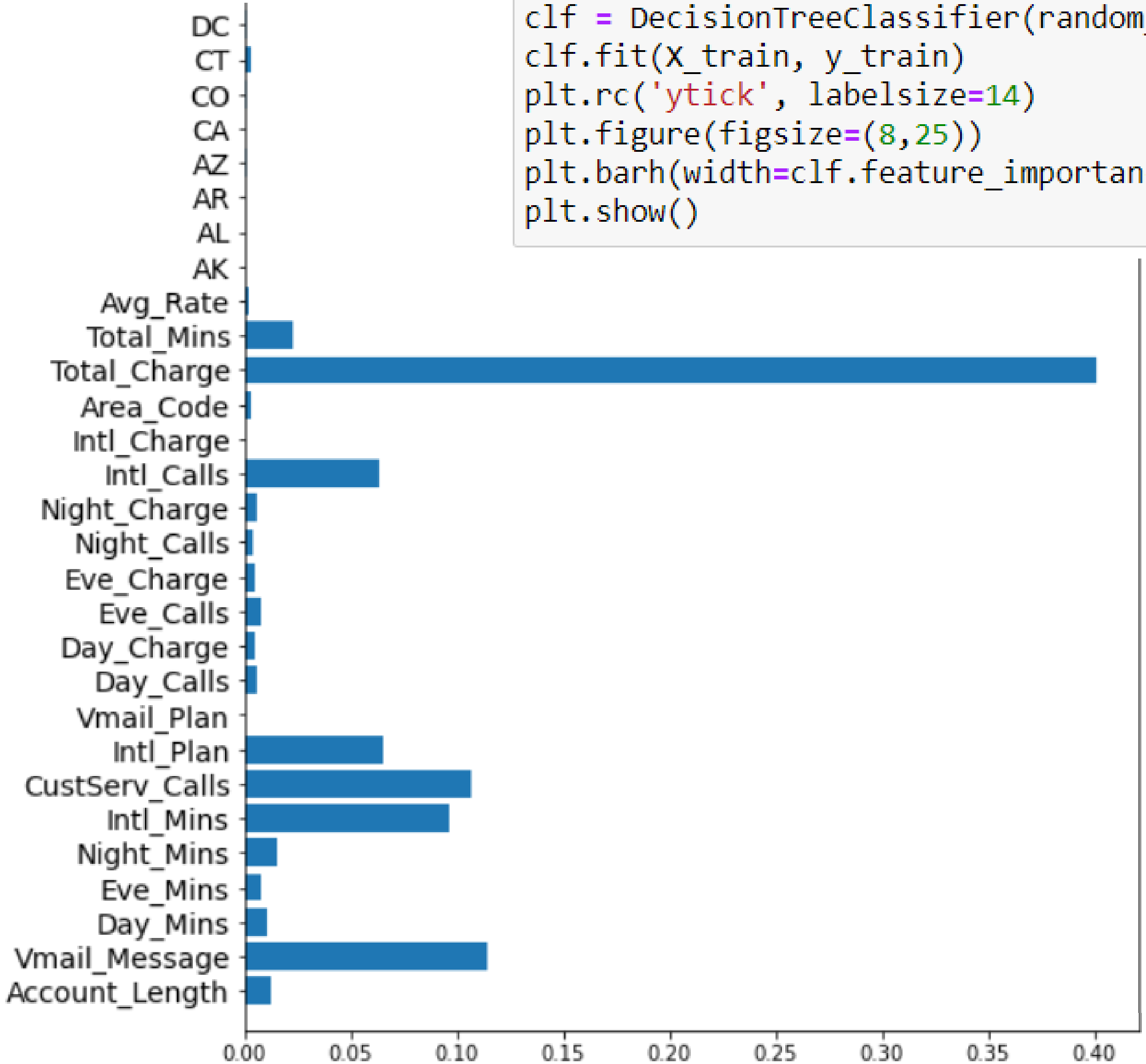
```
from sklearn.tree import DecisionTreeClassifier
tree = DecisionTreeClassifier(random_state=40)
tree.fit(X_train, y_train)
print("Accuracy on training set: {:.3f}".format(tree.score(X_train, y_train)))
print("Accuracy on test set: {:.3f}".format(tree.score(X_test, y_test)))
y_pred = tree.predict(X_test)
from sklearn.metrics import confusion_matrix, precision_score, recall_score
print("Precision on test set: {:.3f}".format(precision_score(y_test, y_pred)))
print("Recall on test set: {:.3f}".format(recall_score(y_test, y_pred)))
confusion_matrix(y_test, y_pred)
```

```
Accuracy on training set: 1.000
Accuracy on test set: 0.950
Precision on test set: 0.793
Recall on test set: 0.884

array([[685,  28],
       [ 14, 107]], dtype=int64)
```

# Example Churn – First Quick Classification

```
clf = DecisionTreeClassifier(random_state=40)
clf.fit(X_train, y_train)
plt.rc('ytick', labelsize=14)
plt.figure(figsize=(8,25))
plt.barh(width=clf.feature_importances_,y=X.columns.values)
plt.show()
```



# Example Churn – Tuning Logistic Regression (1/7)

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
used_features = ['Account_Length', 'Vmail_Message', 'Day_Mins', 'Eve_Mins', 'Night_Mins',
        'Intl_Mins', 'CustServ_Calls', 'Intl_Plan', 'Vmail_Plan', 'Day_Calls', 'Day_Charge',
        'Eve_Calls', 'Eve_Charge', 'Night_Calls', 'Night_Charge', 'Intl_Calls', 'Intl_Charge',
        'Area_Code', 'AK', 'AL', 'AR', 'AZ', 'CA', 'CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI',
        'IA', 'ID', 'IL', 'IN', 'KS','KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS',
        'MT', 'NC', 'ND', 'NE', 'NH', 'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA', 'RI',
        'SC', 'SD', 'TN', 'TX', 'UT', 'VA', 'VT', 'WA', 'WI', 'WV', 'WY']
```

```
l_copies = [0]
l_threshold = [0.5]
l_penalty = ["l2"]
l_C = [1,10,0.1]
l_tol = [10**(-4)]
l_max_iter = [10000]

from sklearn.linear_model import LogisticRegression
```

```
Data not scaled

Best F1 and best parameters:
0.32499999999999996 (0, 0.5, 'l2', 1, 0.0001, 10000)
```

| | Training | Validation | Test |
|---|---|---|---|
| **Accuracy** | 0.87 | 0.865 | 0.892 |
| **Precision** | 0.639 | 0.591 | 0.75 |
| **Recall** | 0.245 | 0.224 | 0.375 |
| **F1** | 0.354 | 0.325 | 0.5 |
| **Confusion matrix Row 1** | [1837, 44] | [666, 18] | [279, 6] |
| **Confusion matrix Row 2** | [241, 78] | [90, 26] | [30, 18] |

# Example Churn – Tuning Logistic Regression (2/7)



```
used_features = ['Account_Length', 'Vmail_Message', 'Day_Mins', 'Eve_Mins', 'Night_Mins',
        'Intl_Mins', 'CustServ_Calls', 'Intl_Plan', 'Vmail_Plan', 'Day_Calls', 'Day_Charge',
        'Eve_Calls', 'Eve_Charge', 'Night_Calls', 'Night_Charge', 'Intl_Calls', 'Intl_Charge',
        'Area_Code', 'AK', 'AL', 'AR', 'AZ', 'CA', 'CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI',
        'IA', 'ID', 'IL', 'IN', 'KS','KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS',
        'MT', 'NC', 'ND', 'NE', 'NH', 'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA', 'RI',
        'SC', 'SD', 'TN', 'TX', 'UT', 'VA', 'VT', 'WA', 'WI', 'WV', 'WY']
```

```
l_copies = [0]
l_threshold = [0.5]
l_penalty = ["l2"]
l_C = [1,10,0.1]
l_tol = [10**(-4)]
l_max_iter = [10000]

from sklearn.linear_model import LogisticRegression
```

```
Data scaled

Best F1 and best parameters:
0.3312101910828026 (0, 0.5, 'l2', 1, 0.0001, 10000)
```

| | Training | Validation | Test |
|---|---|---|---|
| **Accuracy** | 0.862 | 0.869 | 0.874 |
| **Precision** | 0.577 | 0.634 | 0.688 |
| **Recall** | 0.188 | 0.224 | 0.229 |
| **F1** | 0.284 | 0.331 | 0.344 |
| **Confusion matrix Row 1** | [1837, 44] | [669, 15] | [280, 5] |
| **Confusion matrix Row 2** | [259, 60] | [90, 26] | [37, 11] |

# Example Churn – Tuning Logistic Regression (3/7)

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
used_features = ['Account_Length', 'Vmail_Message', 'Day_Mins', 'Eve_Mins', 'Night_Mins',
        'Intl_Mins', 'CustServ_Calls', 'Intl_Plan', 'Vmail_Plan', 'Day_Calls', 'Day_Charge',
        'Eve_Calls', 'Eve_Charge', 'Night_Calls', 'Night_Charge', 'Intl_Calls', 'Intl_Charge']
```

```
l_copies = [0]
l_threshold = [0.5]
l_penalty = ["l2"]
l_C = [1,10,0.1]
l_tol = [10**(-4)]
l_max_iter = [10000]

from sklearn.linear_model import LogisticRegression
```

```
Data scaled

Best F1 and best parameters:
0.3312101910828026 (0, 0.5, 'l2', 1, 0.0001, 10000)
```

| | Training | Validation | Test |
|---|---|---|---|
| **Accuracy** | 0.862 | 0.869 | 0.874 |
| **Precision** | 0.577 | 0.634 | 0.688 |
| **Recall** | 0.188 | 0.224 | 0.229 |
| **F1** | 0.284 | 0.331 | 0.344 |
| **Confusion matrix Row 1** | [1837, 44] | [669, 15] | [280, 5] |
| **Confusion matrix Row 2** | [259, 60] | [90, 26] | [37, 11] |

# Example Churn – Tuning Logistic Regression (4/7)

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
used_features = ['Account_Length', 'Vmail_Message', 'Day_Mins', 'Eve_Mins', 'Night_Mins',
         'Intl_Mins', 'CustServ_Calls', 'Intl_Plan', 'Day_Calls', 'Eve_Calls','Night_Calls',
         'Intl_Calls']
```

```
l_copies = [0]
l_threshold = [0.5]
l_penalty = ["l2"]
l_C = [1,10,0.1]
l_tol = [10**(-4)]
l_max_iter = [10000]

from sklearn.linear_model import LogisticRegression
```

```
Data scaled

Best F1 and best parameters:
0.3312101910828026 (0, 0.5, 'l2', 1, 0.0001, 10000)
```

| | Training | Validation | Test |
|---|---|---|---|
| **Accuracy** | 0.862 | 0.869 | 0.874 |
| **Precision** | 0.577 | 0.634 | 0.688 |
| **Recall** | 0.188 | 0.224 | 0.229 |
| **F1** | 0.284 | 0.331 | 0.344 |
| **Confusion matrix Row 1** | [1837, 44] | [669, 15] | [280, 5] |
| **Confusion matrix Row 2** | [259, 60] | [90, 26] | [37, 11] |

# Example Churn – Tuning Logistic Regression (5/7)

```
used_features = ['Account_Length', 'Vmail_Message', 'Day_Mins', 'Eve_Mins', 'Night_Mins',
         'Intl_Mins', 'CustServ_Calls', 'Intl_Plan', 'Day_Calls', 'Eve_Calls','Night_Calls',
         'Intl_Calls', 'Total_Charge', 'Avg_Rate']
```

```
l_copies = [0]
l_threshold = [0.5]
l_penalty = ["l2"]
l_C = [100,10,1000,1]
l_tol = [10**(-4)]
l_max_iter = [10000]

from sklearn.linear_model import LogisticRegression
```

```
Data scaled

Best F1 and best parameters:
0.4311377245508982 (0, 0.5, 'l2', 100, 0.0001, 10000)
```

| | Training | Validation | Test |
|---|---|---|---|
| Accuracy | 0.871 | 0.881 | 0.898 |
| Precision | 0.628 | 0.706 | 0.75 |
| Recall | 0.27 | 0.31 | 0.438 |
| F1 | 0.377 | 0.431 | 0.553 |
| Confusion matrix Row 1 | [1830, 51] | [669, 15] | [278, 7] |
| Confusion matrix Row 2 | [233, 86] | [80, 36] | [27, 21] |

# Example Churn – Tuning Logistic Regression (6/7)

Fachhochschule Dortmund
University of Applied Sciences and Arts

```
used_features = ['Account_Length', 'Vmail_Message', 'Day_Mins', 'Eve_Mins', 'Night_Mins',
         'Intl_Mins', 'CustServ_Calls', 'Intl_Plan', 'Day_Calls', 'Eve_Calls','Night_Calls',
         'Intl_Calls', 'Total_Charge', 'Avg_Rate']
```

```
l_copies = [0]
l_threshold = [0.5,0.4,0.3,0.2,0.1]
l_penalty = ["l2"]
l_C = [100,10,1000,1]
l_tol = [10**(-4)]
l_max_iter = [10000]

from sklearn.linear_model import LogisticRegression
```

```
Data scaled

Best F1 and best parameters:
0.5409252669039145 (0, 0.2, 'l2', 10, 0.0001, 10000)
```

| | Training | Validation | Test |
|---|---|---|---|
| **Accuracy** | 0.827 | 0.839 | 0.85 |
| **Precision** | 0.438 | 0.461 | 0.486 |
| **Recall** | 0.68 | 0.655 | 0.75 |
| **F1** | 0.533 | 0.541 | 0.59 |
| **Confusion matrix Row 1** | [1603, 278] | [595, 89] | [247, 38] |
| **Confusion matrix Row 2** | [102, 217] | [40, 76] | [12, 36] |

# Example Churn – Tuning Logistic Regression (7/7)

Fachhochschule Dortmund
University of Applied Sciences and Arts

```python
used_features = ['Account_Length', 'Vmail_Message', 'Day_Mins', 'Eve_Mins', 'Night_Mins',
         'Intl_Mins', 'CustServ_Calls', 'Intl_Plan', 'Day_Calls', 'Eve_Calls','Night_Calls',
         'Intl_Calls', 'Total_Charge', 'Avg_Rate']
```

```python
l_copies = [0,1,2,3,4]
l_threshold = [0.5,0.4,0.3,0.2,0.1]
l_penalty = ["l2"]
l_C = [100,10,1000,1]
l_tol = [10**(-4)]
l_max_iter = [10000]

from sklearn.linear_model import LogisticRegression
```

```
Data scaled

Best F1 and best parameters:
0.5625 (1, 0.4, 'l2', 10, 0.0001, 10000)
```

| | Training | Validation | Test |
|---|---|---|---|
| **Accuracy** | 0.815 | 0.86 | 0.862 |
| **Precision** | 0.646 | 0.514 | 0.515 |
| **Recall** | 0.592 | 0.621 | 0.708 |
| **F1** | 0.618 | 0.562 | 0.596 |
| **Confusion matrix Row 1** | [1674, 207] | [616, 68] | [253, 32] |
| **Confusion matrix Row 2** | [260, 378] | [44, 72] | [14, 34] |

# Example Churn – Exercise



(1) Go through the notebook and try to roughly understand the code behind the previous slides

(2) Adapt the last cells of the workbook and tune the parameters for

  a) Kernelized Support Vectors Machines

  b) Random Forest

  c) Gradient Boosted Trees

(3) If you still have time: Create new cells in the notebook for another model (e.g. neural network, k-nearest neigbors, decision tree) and tune appropriate model parameters

(4) Which model and which parameter setting results in the best F1-score on the validation set? What is the corresponding F1-score on the test set? Which model / setting would you recommend to use for predictions on unknown data?

# Course evaluation

Fachhochschule Dortmund
University of Applied Sciences and Arts

Discuss the course. What went well, what can be approved?

Align on the top 2 stars and wishes within your group, i.e. present

- 2 things that worked very well (due to the course design, but could also be best practices that you applied as team or individuals)
- 2 concrete suggestions how the course design can be improved (please be very concrete and specific)

Focus on the 2 most important ones in each category. Sort out and rank all of your ideas