

Data Mining & Pattern Recognition

113444a

**Versuch 1 und 2:
Allgemeine Data Mining Prozessschritte**

Protokoll von:

Ruben Müller (rm034)

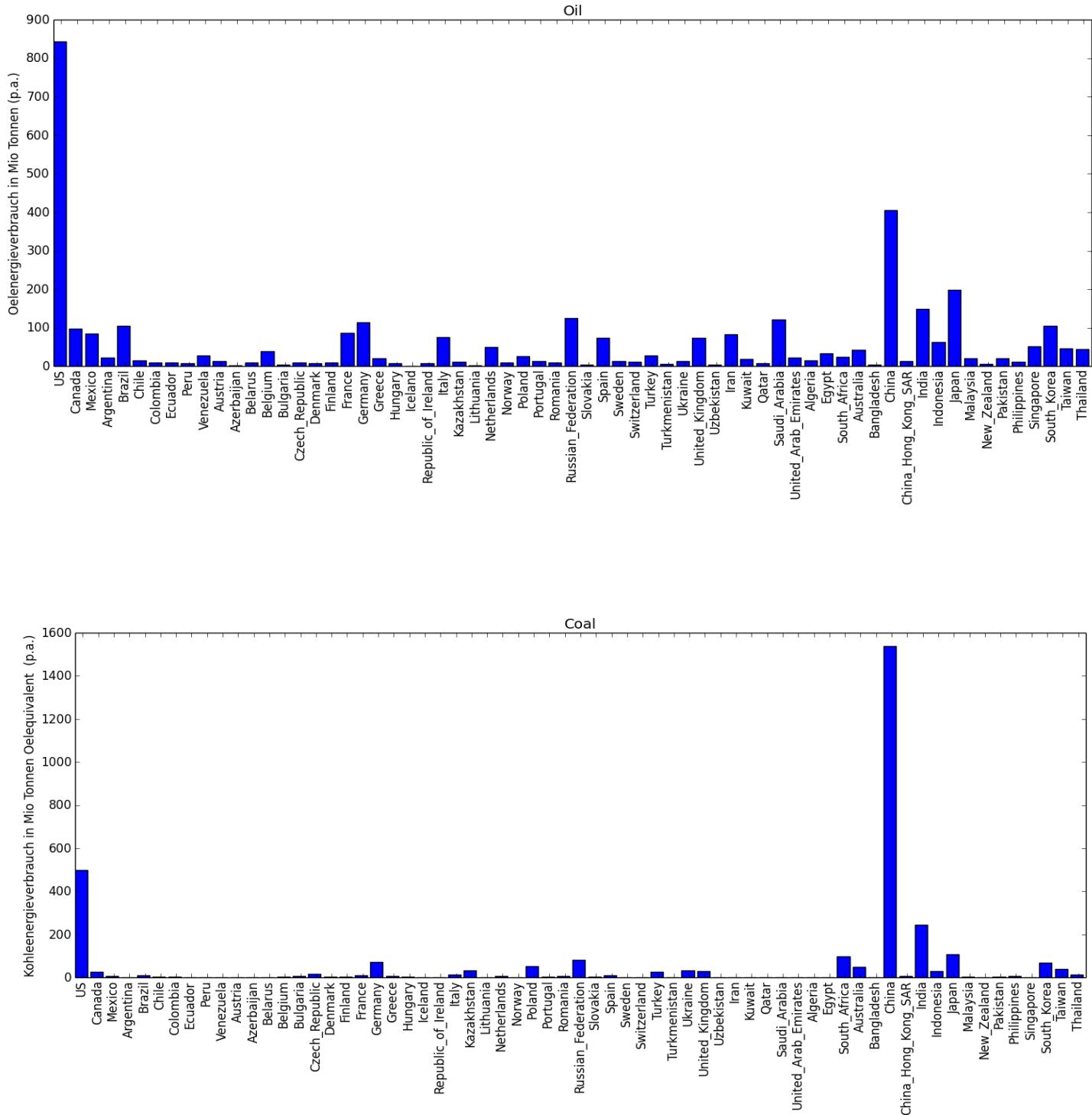
Lisa Otter (lo008)

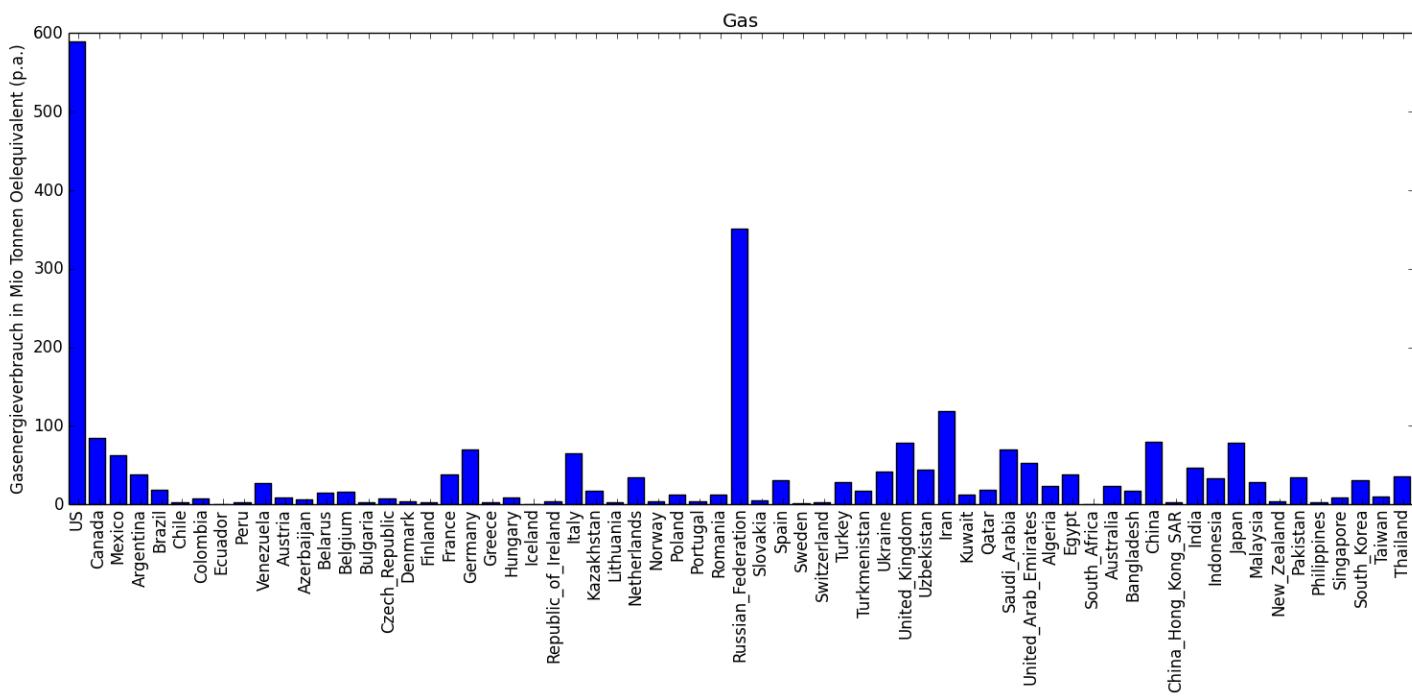
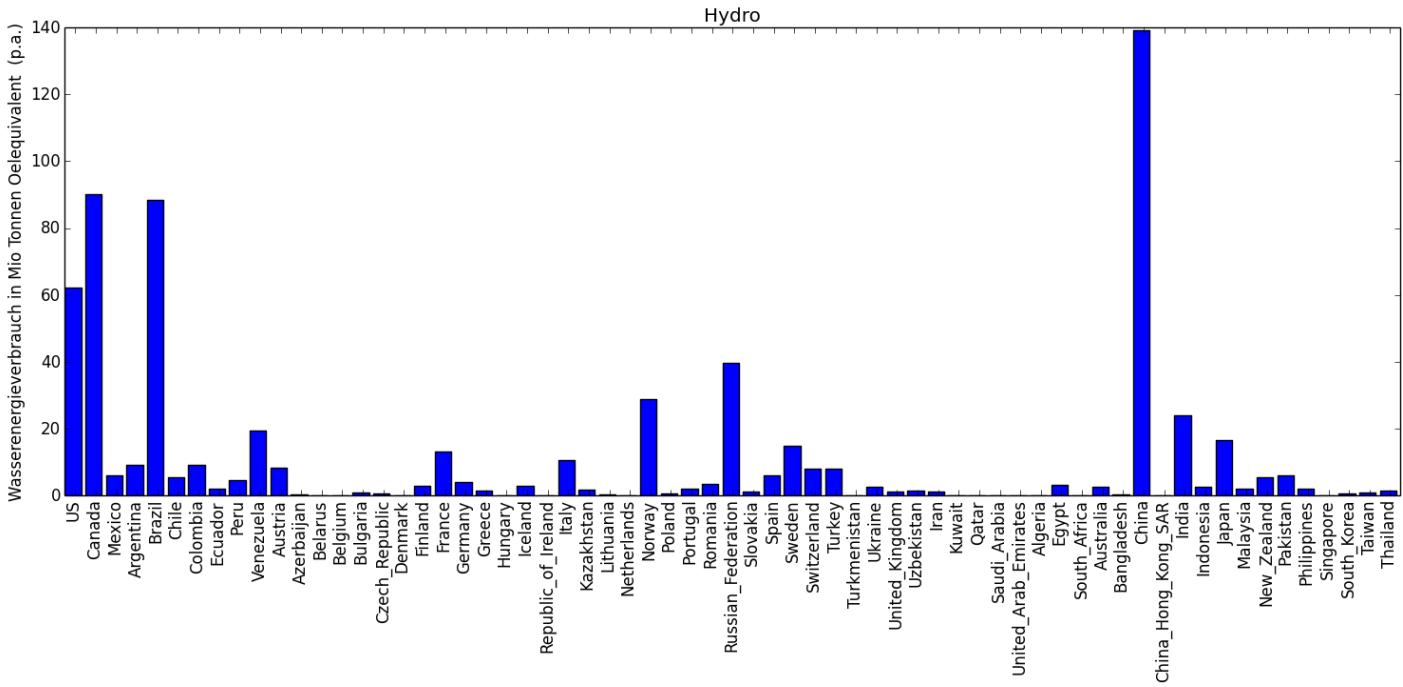
Manuela Schuler (ms334)

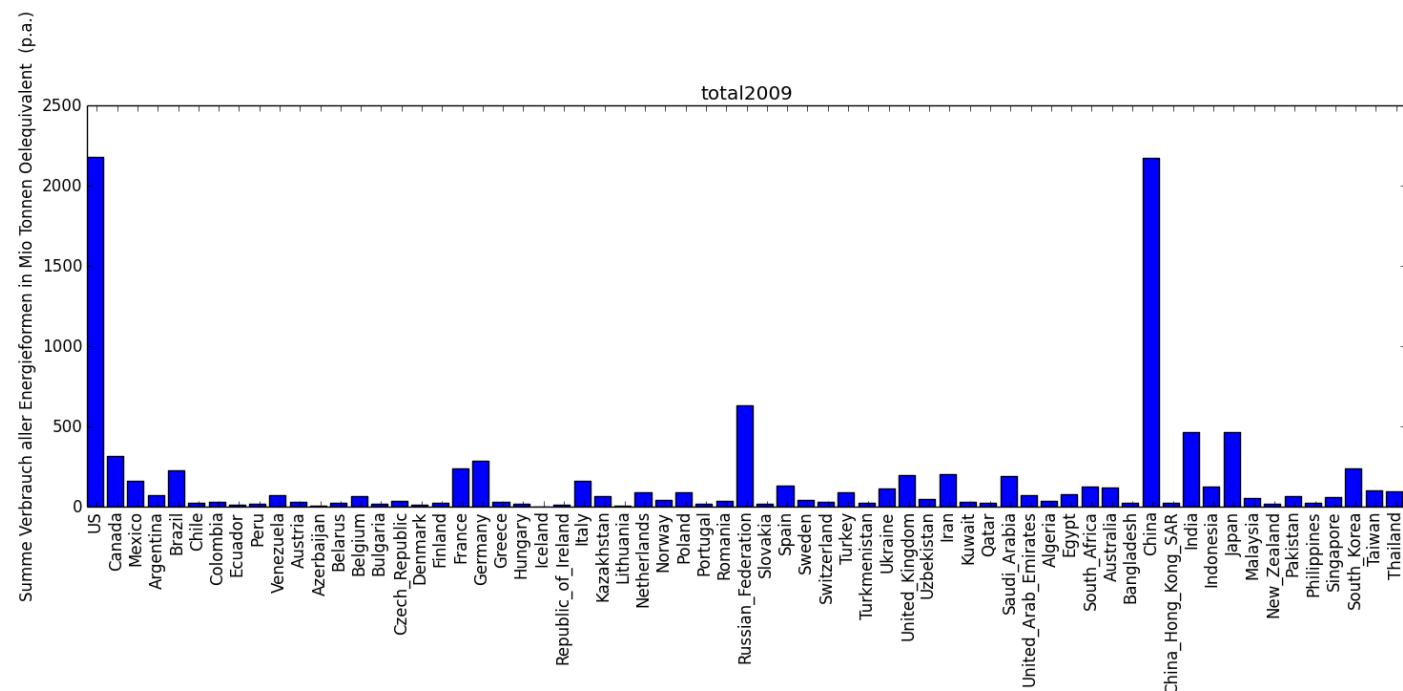
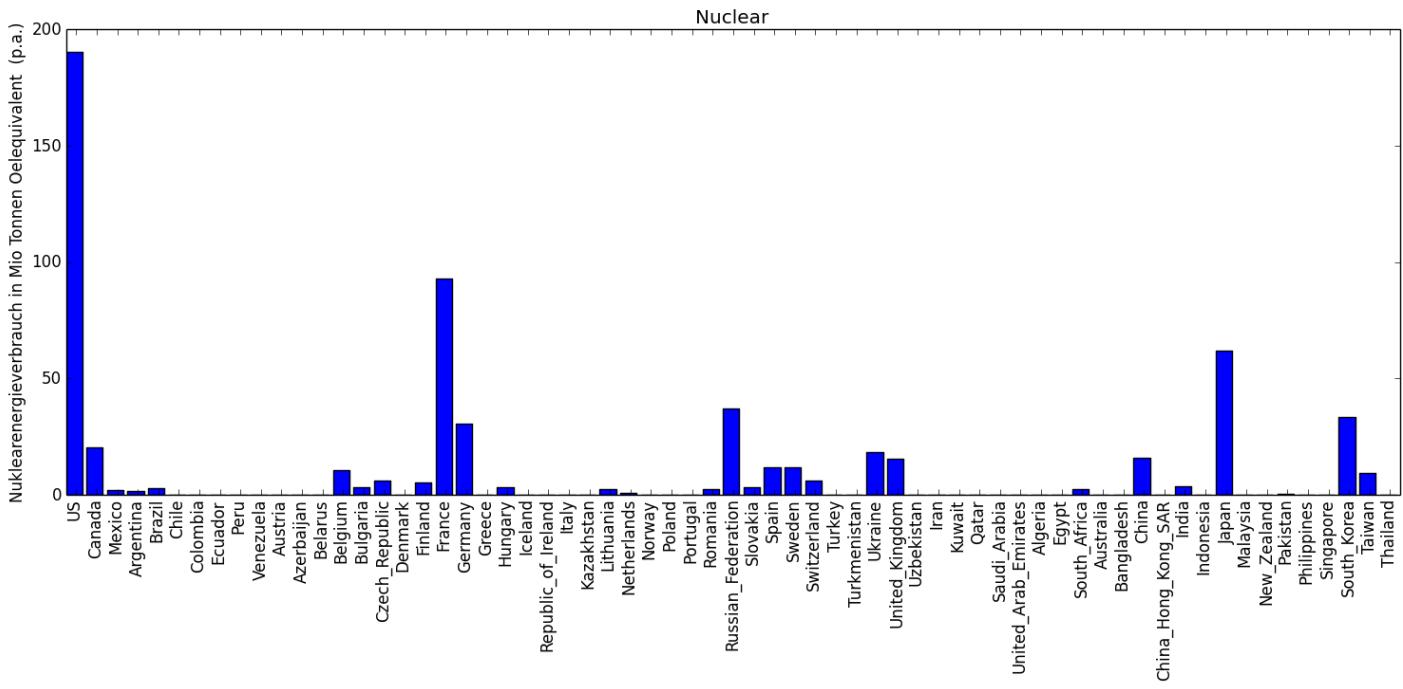
Versuch 1: Energieverbrauch und CO2-Emission

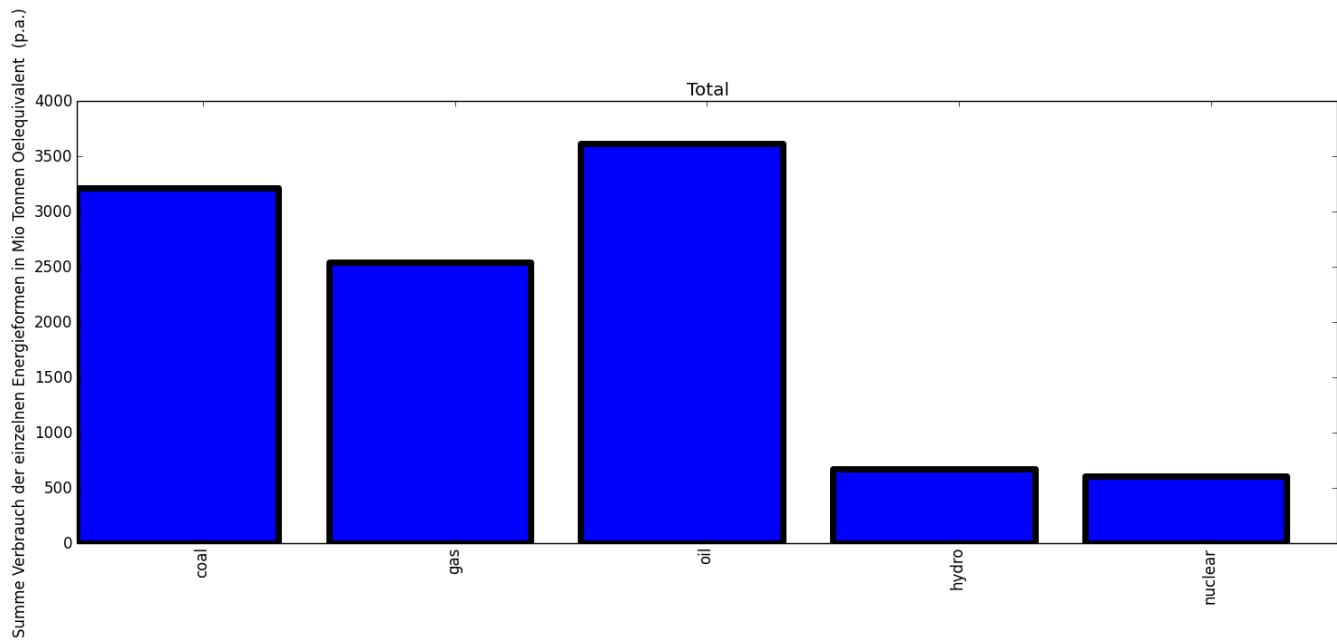
2.1 Datenverwaltung und Statistik

2.1.2 Einlesen der Daten, Hinzufügen der GPS Koordinaten, Abspeichern in neuer Datei









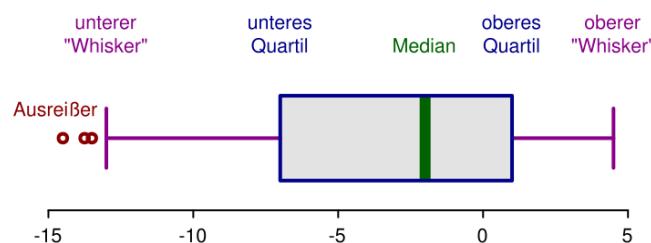
Ausgehend von der implementierten Visualisierung des Energieverbrauchs der Länder: Nennen Sie die 3 Ihrer Meinung nach interessantesten Beobachtungen.

1. Da die USA nur 23,5% der Bevölkerung von China hat, brauchen sie verhältnismäßig viel Energie, sogar noch mehr als China.
317.238.626 Usa Einwohner mit Gesamtverbrauch: 2182 Mio Tonnen pro Jahr
1.349.585.838 China Einwohner mit Gesamtverbrauch: 2177 Mio Tonnen pro Jahr
2. Kohle wird von sehr wenigen Ländern benutzt, trotzdem ist der Gesamtverbrauch an Kohle der zweithöchste.
3. Öl wird von jedem Land verbraucht, Gas von fast jedem (außer Island, Südafrika)
4. Der Nuklearenergieverbrauch ist geringer als der Hydroenergieverbrauch. Intuitiv hätten wir erwartet, dass der Nuklearenergieverbrauch höher als der Hydroenergieverbrauch ist.

2.1.3 Statistik der Daten

1. Erklären Sie sämtliche Elemente eines Boxplot (allgemein):

Ein Boxplot soll schnell einen Eindruck darüber vermitteln, in welchem Bereich die Daten liegen und wie sie sich über diesen Bereich verteilen.



- Innerhalb der Box liegt der Hauptanteil der Daten
- Median: Mittelwert für Verteilungen in der Statistik, der am häufigsten vorkommende Wert bzw. der Wert in der Mitte der geordneten Daten
- unterer Whisker: kleinster Datenwert der geordneten Daten (Nicht-Außreißer)
- oberer Whisker: größter Datenwert der geordneten Daten (Nicht-Außreißer)
- Die Whiskergrenzen richten sich nach dem Interquartilsabstand
- Außreißer: alle Werte außerhalb dieser Whisker-Grenzen sind Ausreißer.
- unteres Quartil: 25% der geordneten Daten liegen unterhalb dieses Kennwerts
- oberes Quartil: 75% der geordneten Daten liegen unterhalb dieses Kennwerts

Die Boxplot-Methode

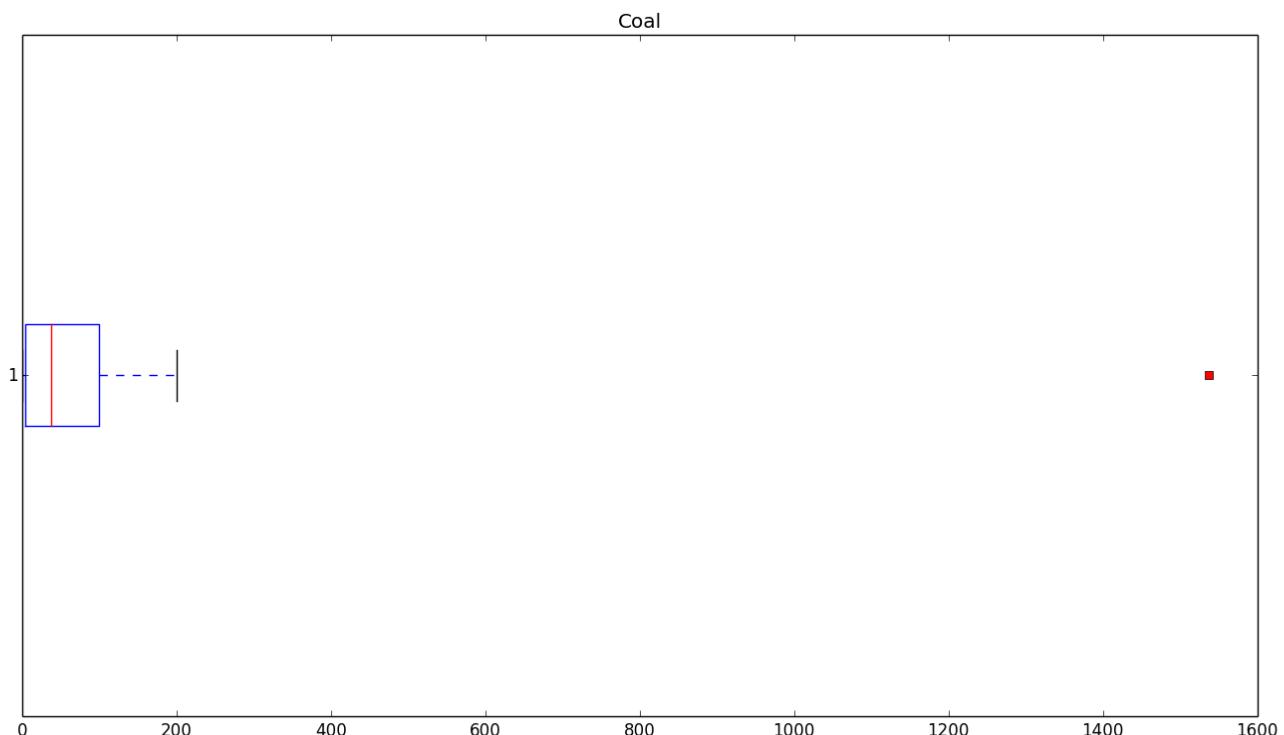
boxplot(data,0,'rs',0) :

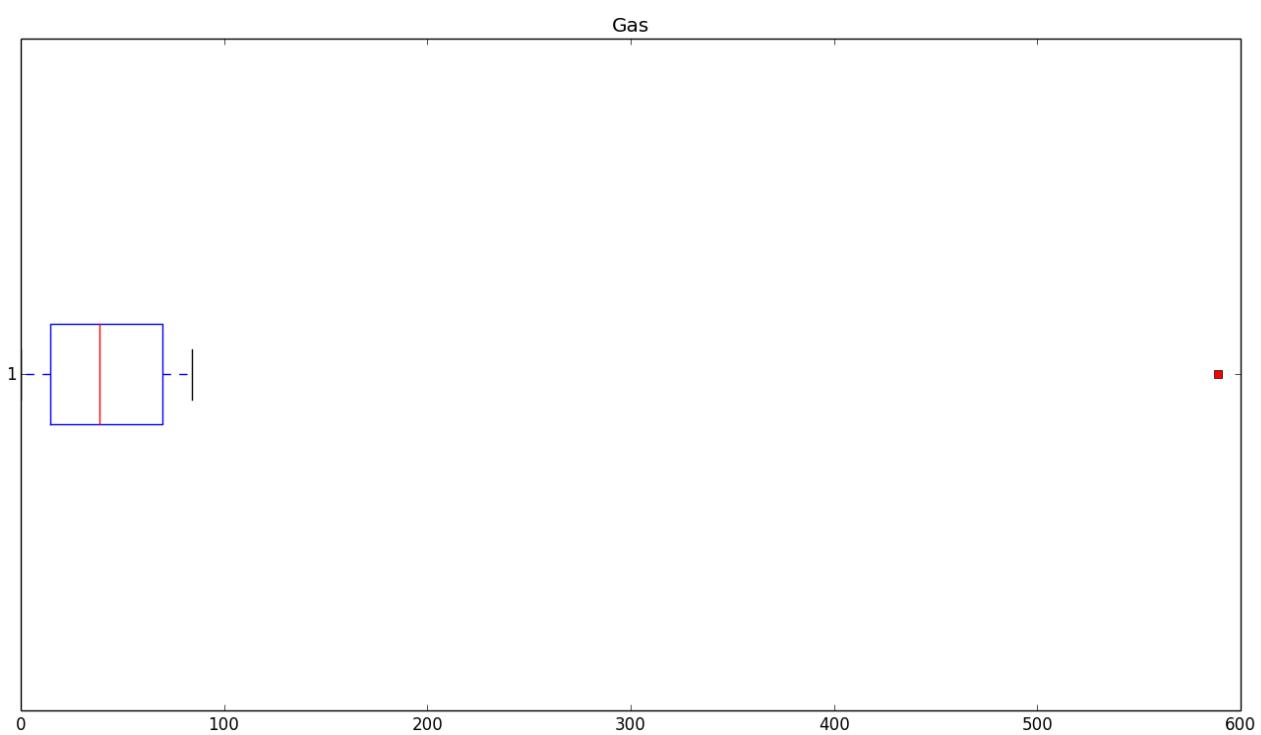
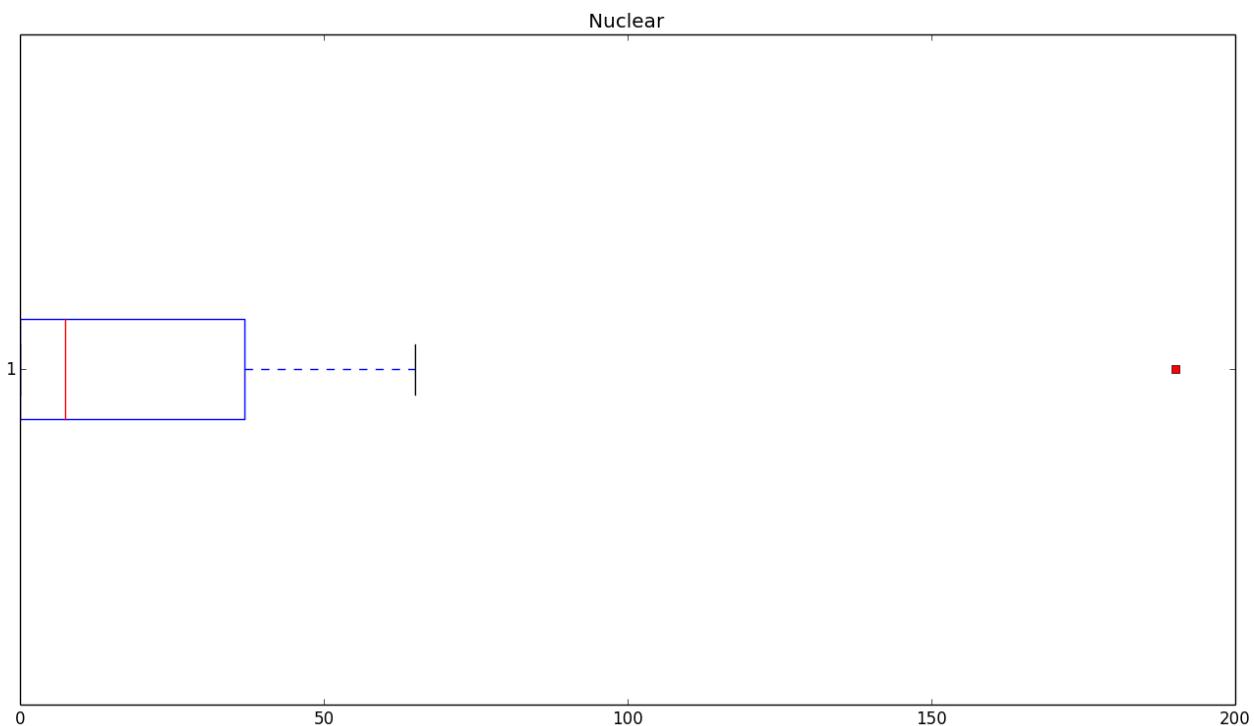
data:

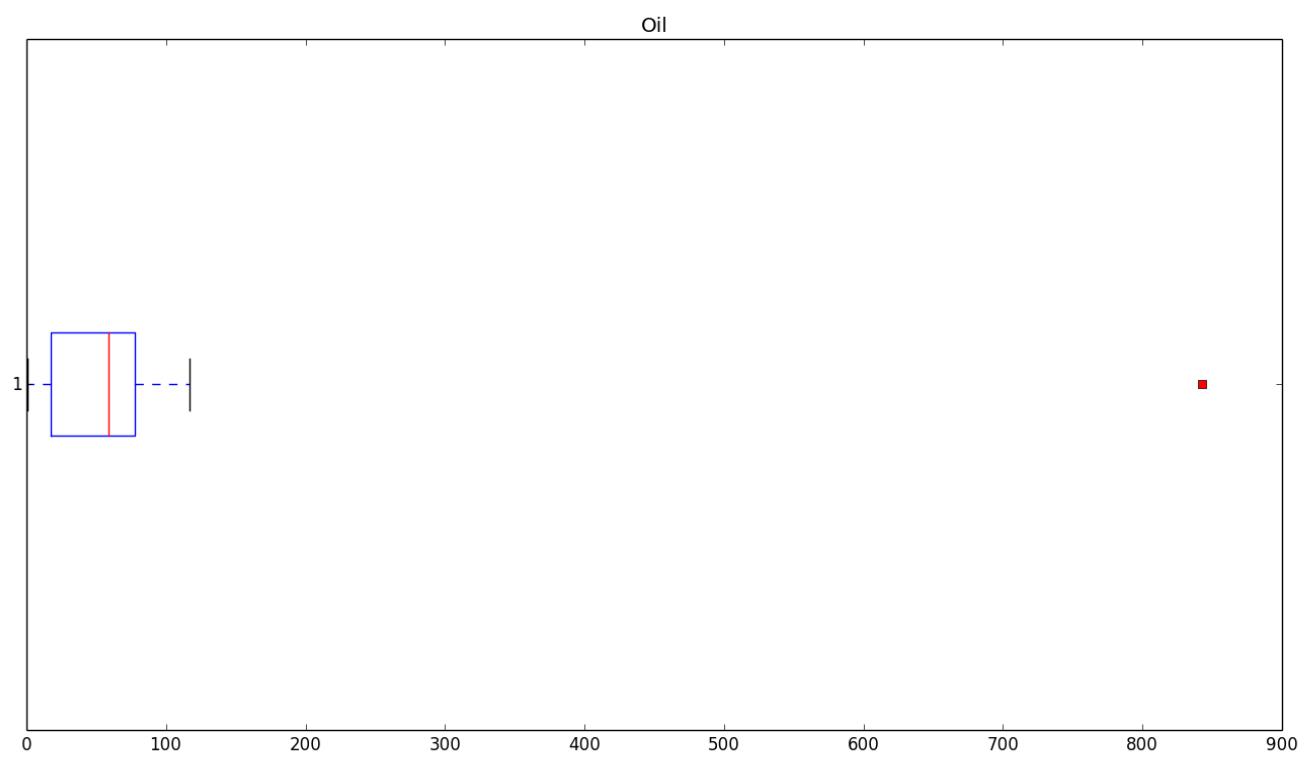
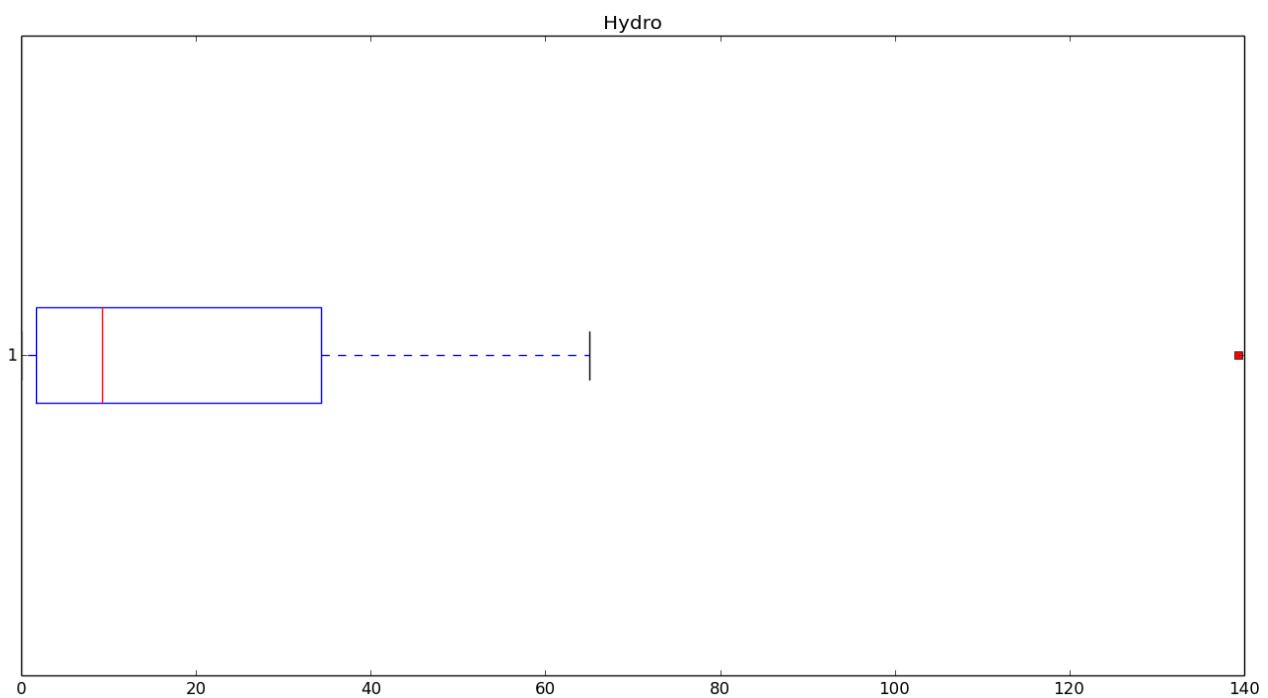
(0 oder 1): bei 1 wird ein notched box plot angezeigt

'rs': Darstellungsweise der Ausreißer, in diesem Fall als rotes Viereck

(0 oder 1): 0 steht für horizontal und 1 steht für vertikal







2. Diskutieren Sie die im Boxplot angezeigte Statistik der Energieverbrauchdaten:

- Im Boxplot von Kohle und Nuklear liegt das untere Quartil bei 0. Dies bedeutet, dass im Vergleich zu den anderen Energieformen mehr Länder gar nichts dieser beiden Energieformen verbrauchen.
- Quantitativ wird wenig Hydroenergie verbraucht, da einerseits bereits der obere Whisker bei ca. 70 liegt., gleichzeitig aber auch fast 50% der Länder relativ wenig verbrauchen. Der große Abstand zwischen Median und oberem Whisker bedeutet eine starke Streuung in diesem Bereich.
- Bei Nuklearenergie ist ebenfalls wie bei der Hydroenergie eine hohe Streuung im oberen Bereich zwischen Median und oberem Whisker.
- Der Verbrauch von Gas ist relativ gleichverteilt.
- Der Gesamtverbrauch von Kohle, Gas und Öl liegt jeweils höher als der Verbrauch von Hydro und Nuklear. Dies zeigt der obere Whisker an, der bei den Erstgenannten höher liegt.

2.2 Anwendung von Verfahren des unüberwachten Lernens auf Energieverbrauchsdaten

2.2.1 Hierarchisches Clustering

1. Was wird beim Standardisieren gemacht? Welcher Effekt könnte ohne Standardisieren beim Clustering eintreten (insbesondere wenn die euklidische Metrik verwendet wird)?

Beim Standardisieren wird die Varianz jedes Merkmals auf 1 gesetzt, d.h. alle Elemente des jeweiligen Merkmals werden dadurch skaliert. Dies kann zusätzlich mit oder ohne Verschiebung um den Mittelwert geschehen (mean_shift = True/False). Ohne Standardisierung haben die Merkmale abhängig von ihrer Varianz einen unterschiedlich großen Einfluss auf das Clustering-Ergebnis. Merkmale mit einer höheren Varianz hätten einen größeren Einfluss auf die Distanzmatrix, dies ist jedoch ein unerwünschter Effekt. zB.: Merkmale "Alter" (relativ geringe Varianz) und "Einkommen" (hohe Varianz) Da die euklidische Metrik die absolute Distanz zwischen zwei Merkmalsvektoren berechnet, wäre der Einfluss dadurch umso größer.

2. Erklären Sie die beim hierarchischen Clustering einstellbaren Parameter linkage-method und metric. Welche Metrik ist Ihrer Meinung nach für diese Anwendung geeignet? Warum?

Die linkage-Method gibt an, die die Distanz zwischen zwei Clustern berechnet wird und entscheidet damit darüber, welche zwei Cluster jeweils zu einem neuen Cluster zusammengefasst werden: Konkret gibt die Linkage-Methode an, zwischen welchen Punkten von Cluster A und Cluster B jeweils die Distanz berechnet werden soll. Es gibt Single-Linkage ("kürzester Weg": Abstand zwischen Cluster A und B ist minimaler Abstand zwischen einer Instanz in Cluster A und einer Instanz in Cluster B), Complete-Linkage ("weitester Weg", Abstand zwischen Cluster A und B ist maximaler Abstand zwischen einer

Instanz in Cluster A und einer Instanz in Cluster B) und Average-Linkage (Berechnung aller möglichen Distanzen und Mittelwert daraus berechnen) sowie Centroid-Linkage (Distanz zwischen den Clustermittelpunkten).

Die Metrik gibt das Distanzmaß an und damit die Formel, nach der die Distanz zwischen zwei Punkten berechnet wird. Es gibt z.B. die Euklidische Distanz, Jaccard, Tanimoto, Pearson, Correlation, MSE, MAD usw.:

Es handelt sich ausschließlich um skalierte numerische Werte, welche nicht Mittelwertfrei sind.

In Frage kommen daher die euklidische Distanz, sowie Correlation und Pearson (hier wird der Mittelwert jeweils abgezogen, Pearson ist Maß für die lineare Korrelation zwischen zwei Vektoren, falls Mittelwerte a und b = 0 ist Pearson == Cosinus-Korrelation)

3. Welches Land ist bezüglich des Verbrauchs der hier betrachteten Energiequellen Deutschland am ähnlichsten, wenn für die linkage-method average und die Metrik correlation konguriert wird?

Belgien ist Deutschland am ähnlichsten, wenn als linkage-methode "average" und als metrik "correlation" benutzt wird.

4. Charakterisieren Sie die 4 Cluster. Was ist typisch für die jeweiligen Cluster?

Cluster0:

Hauptsächlicher Energieverbrauch von Öl, Kohle, etwas Gas, wenig Nuklear und so gut wie keine Hydro

Cluster1:

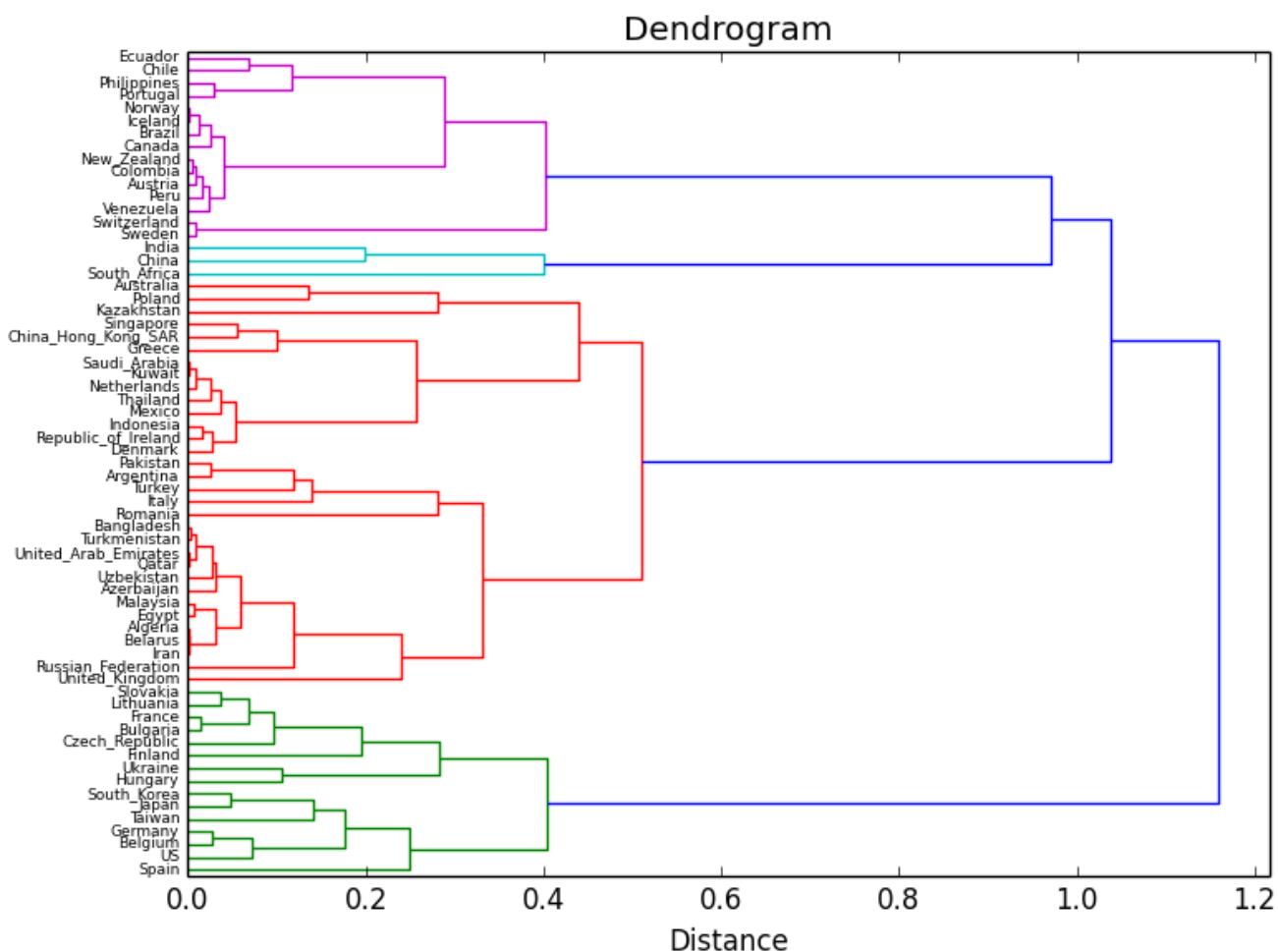
Hauptsächlicher Energieverbrauch von Öl, Gas, etwas Kohle, so gut wie kein Nuklear und Nydro

Cluster2:

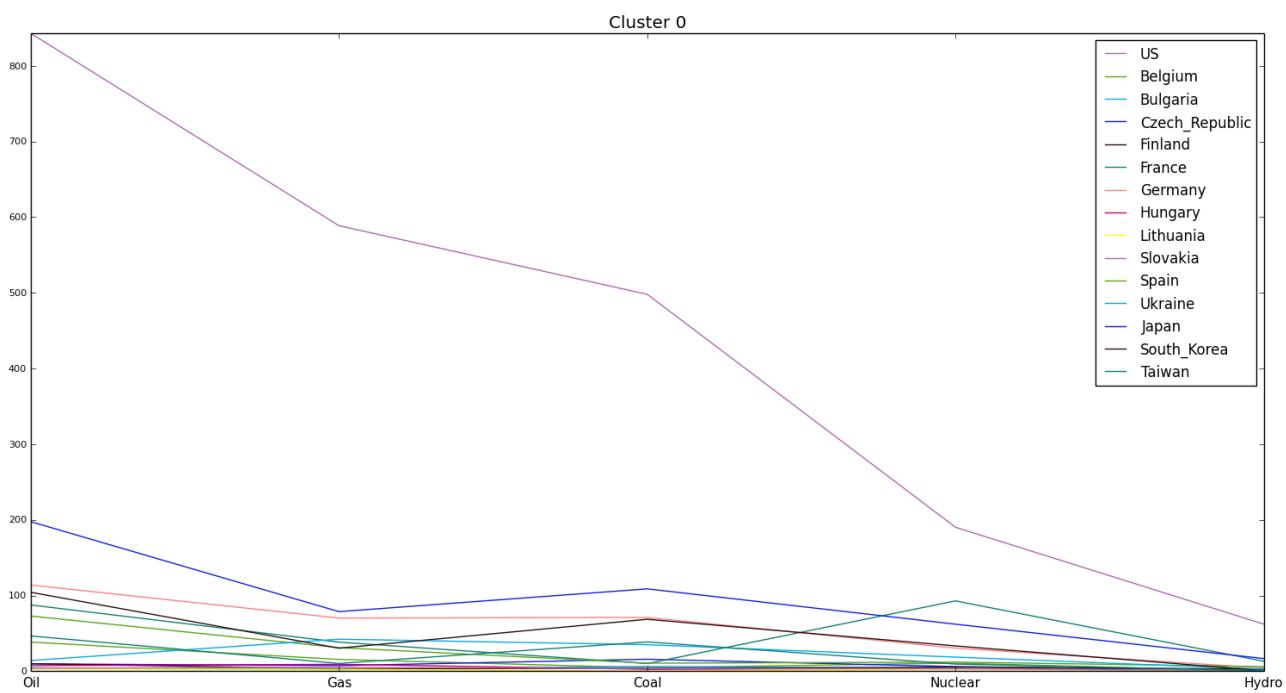
Hauptsächlicher Energieverbrauch von Kohle, etwas Öl, so gut wie kein Nuklear, Gas und Hydro

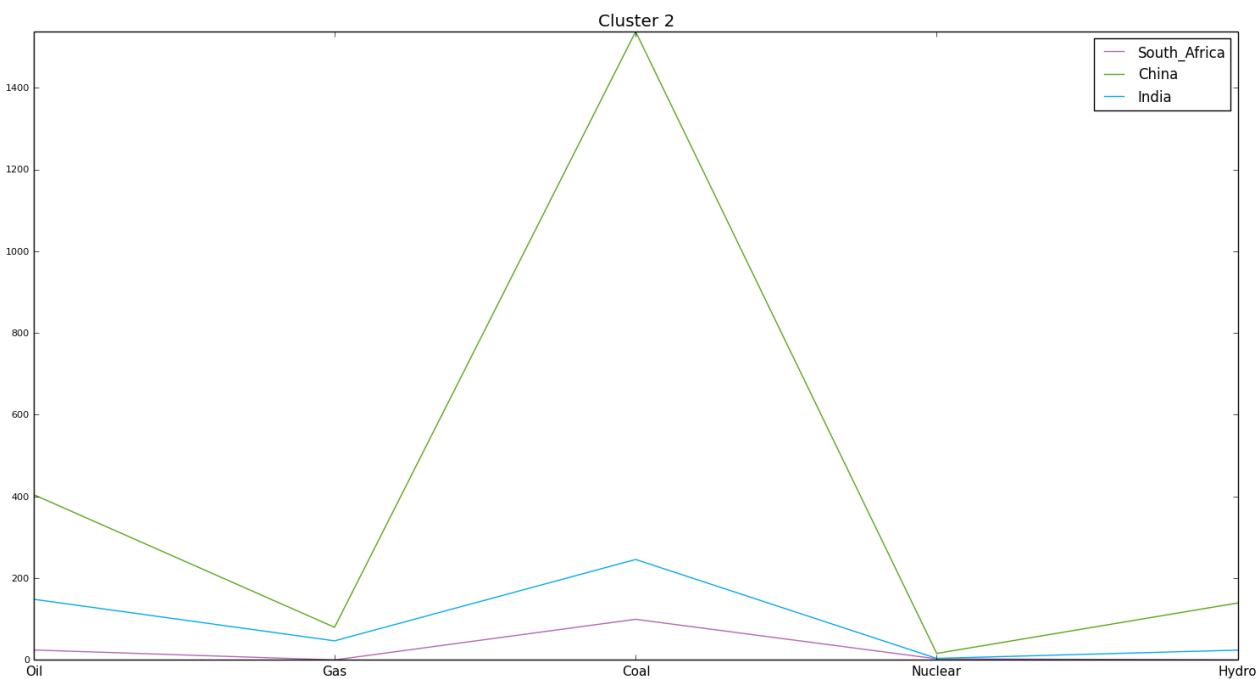
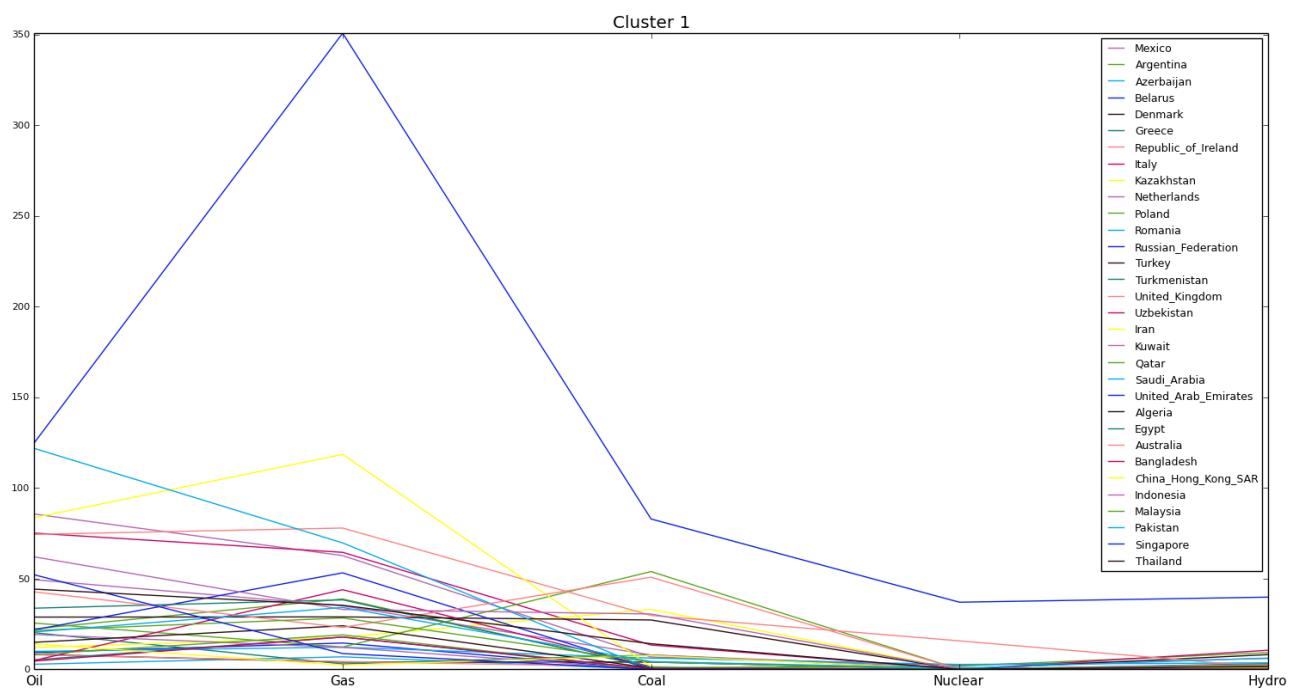
Cluster3:

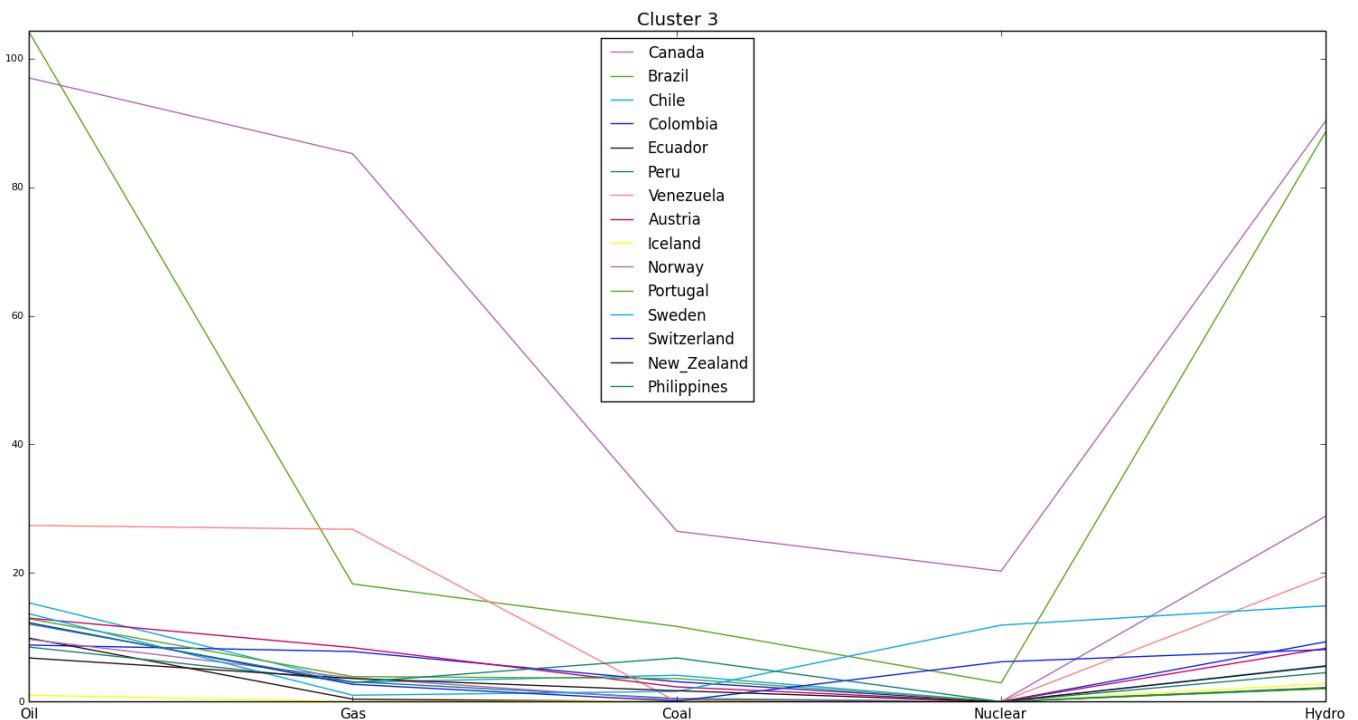
Hauptsächlicher Energieverbrauch von Öl, Hydro, wenig Gas, Kohle, so gut wie kein Nuklear



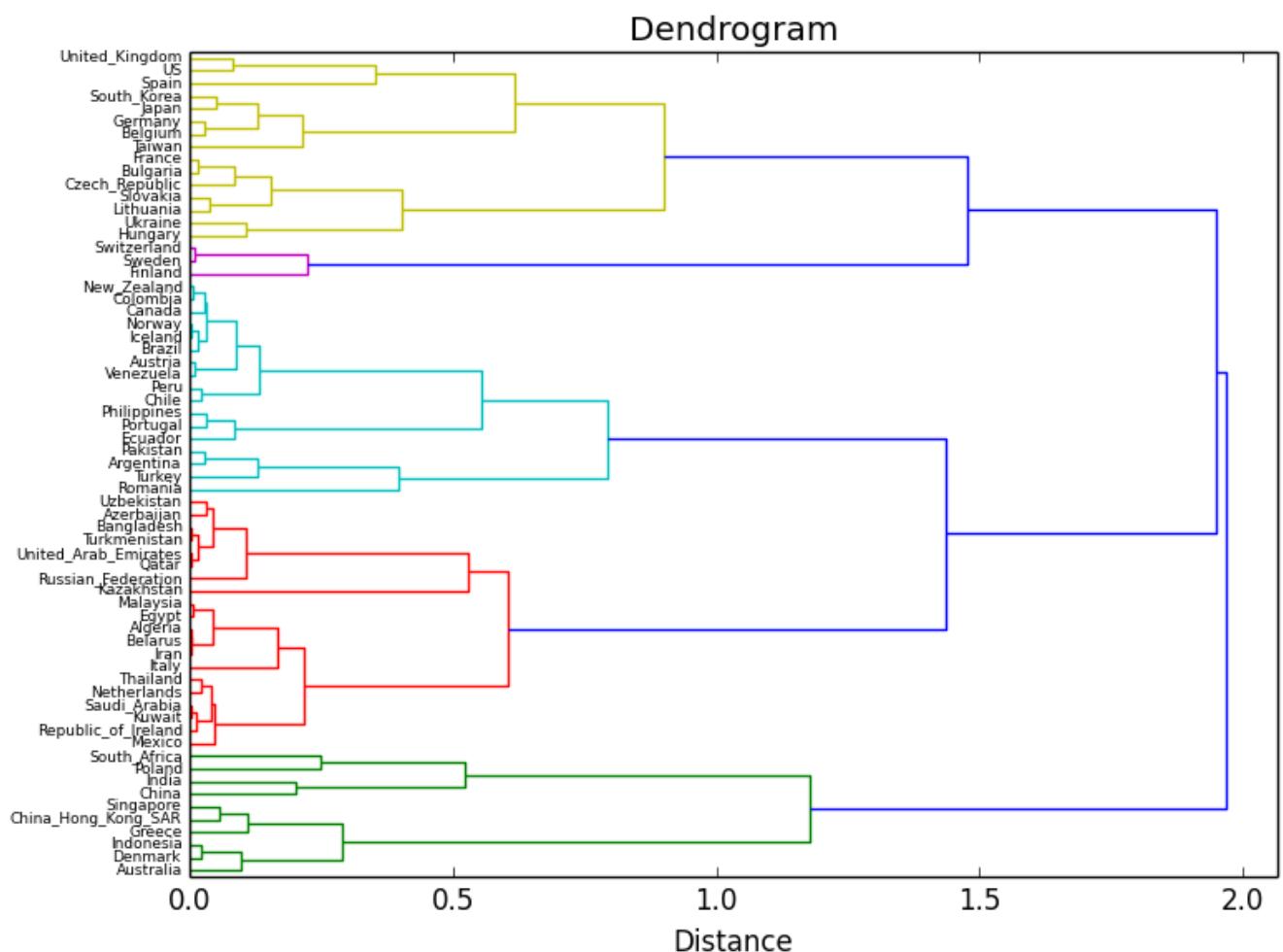
Individual Clusters:

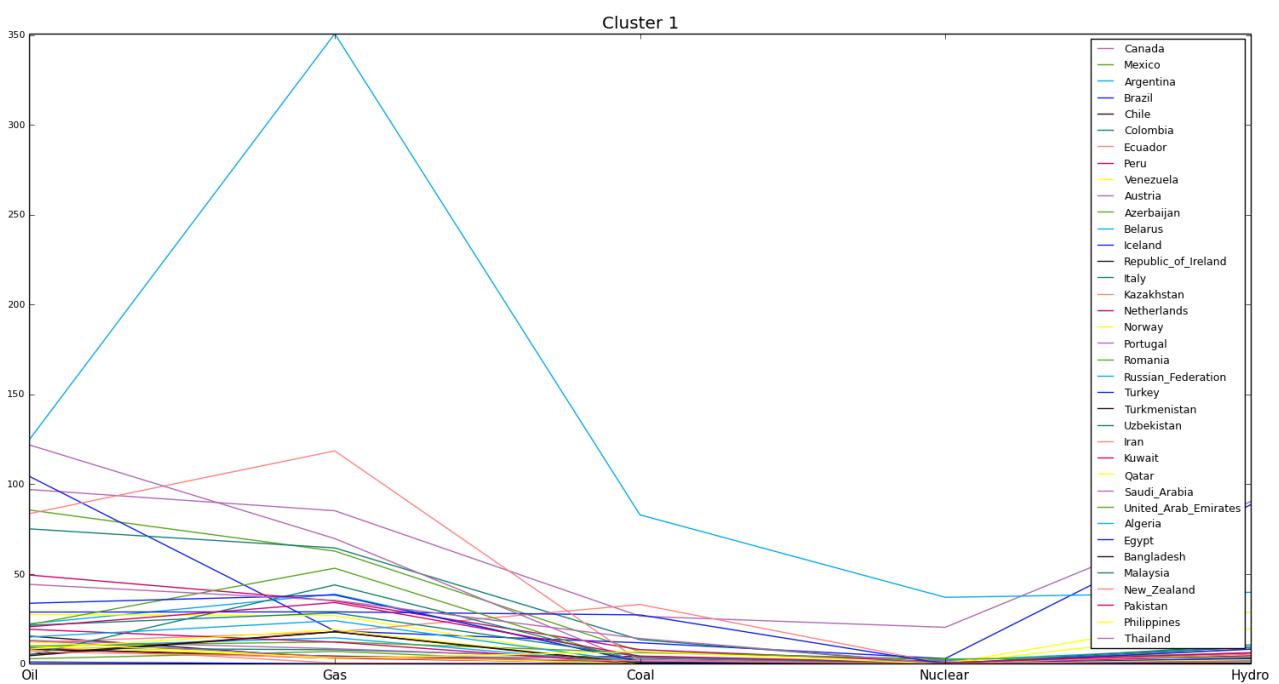
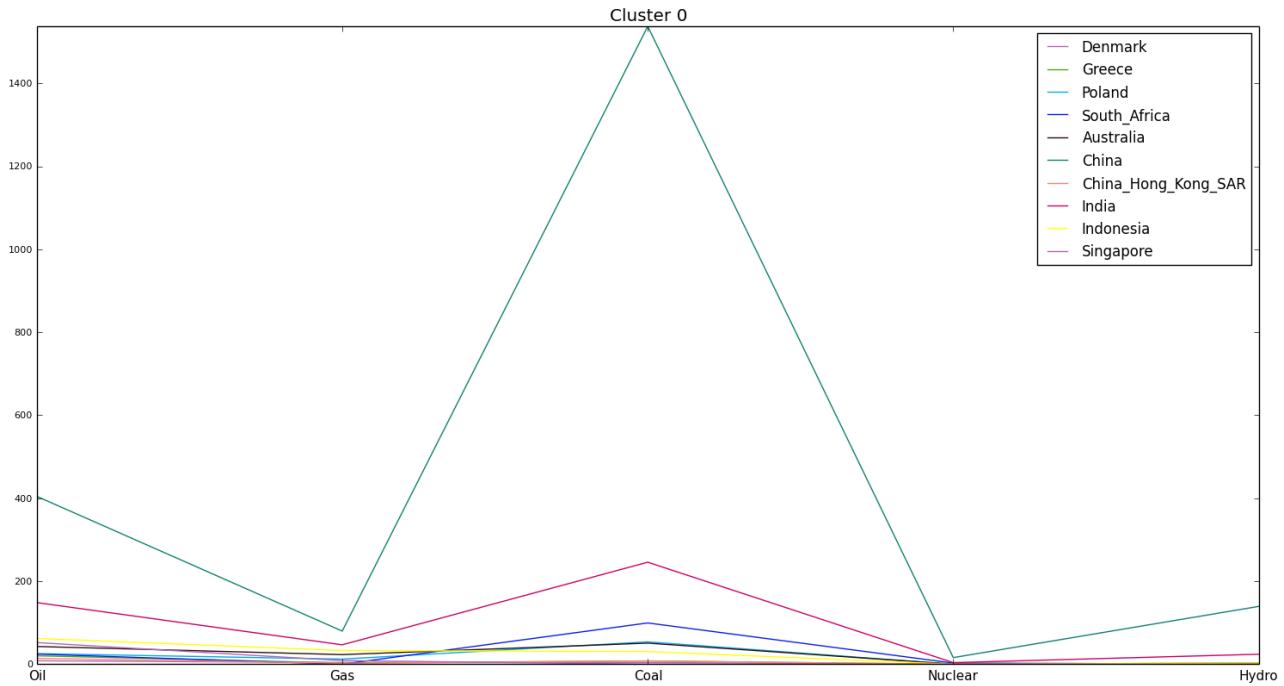


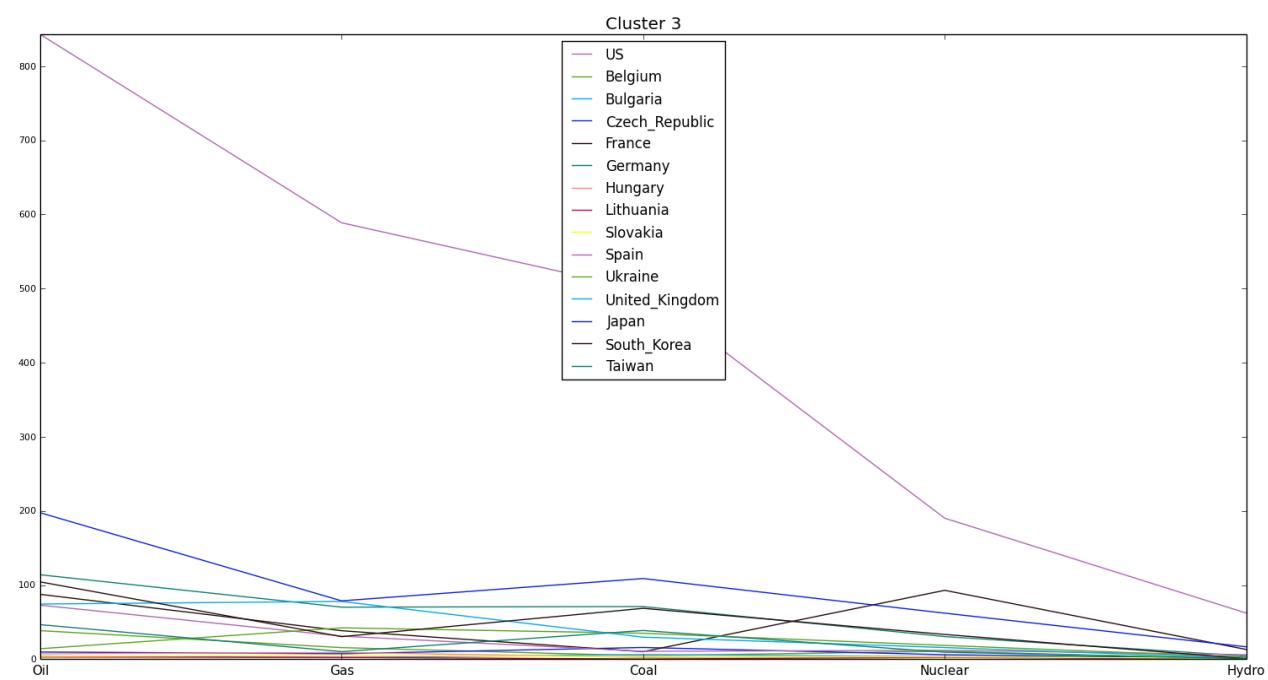
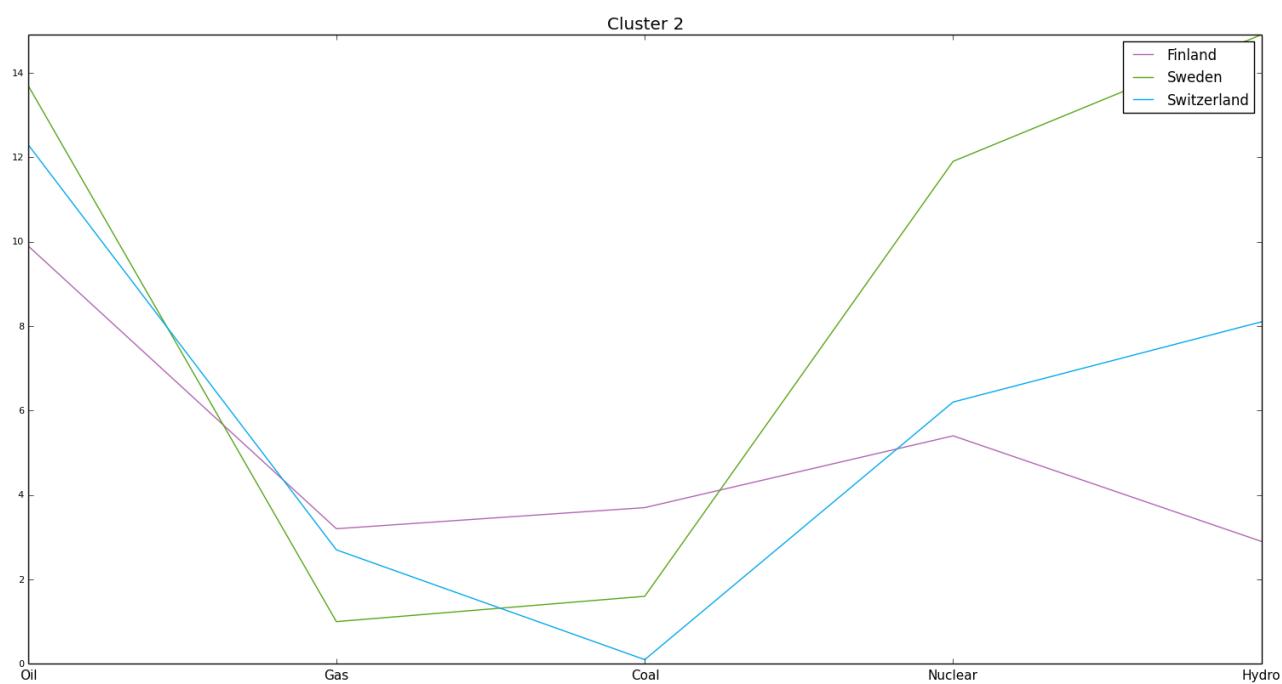




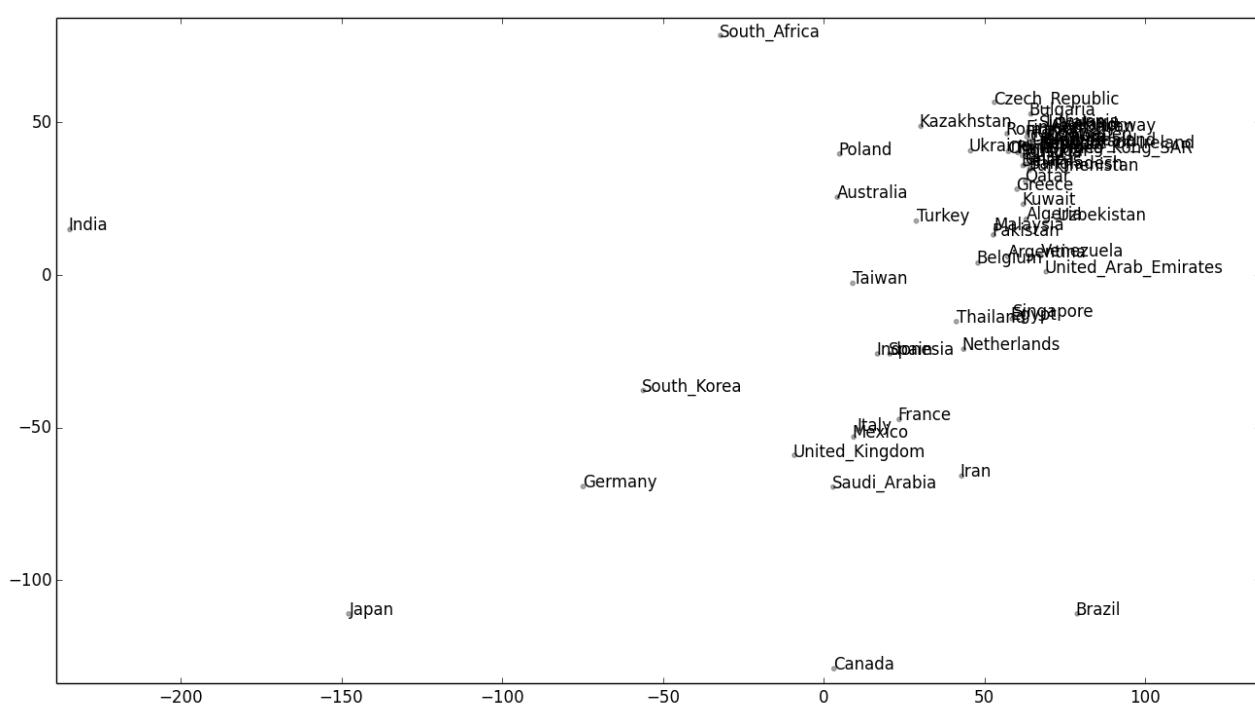
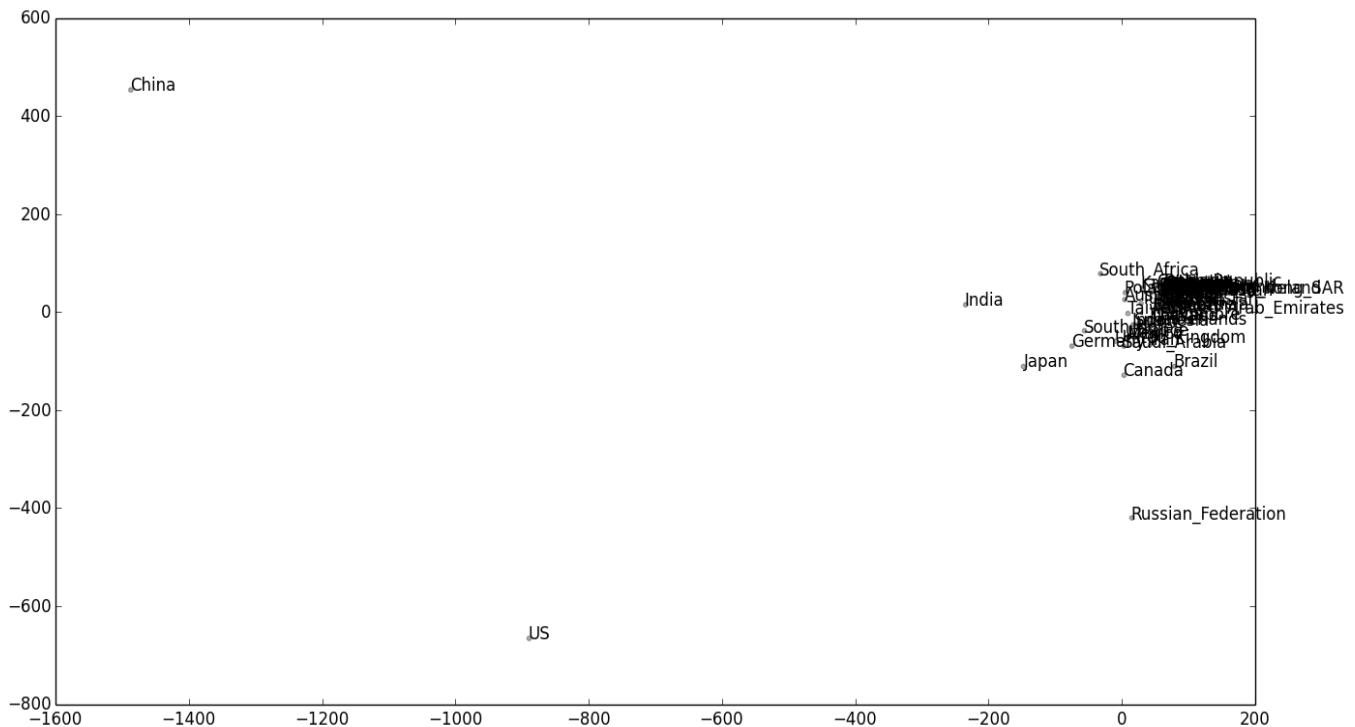
Zusätzlich haben wir es auch mit der Linkage-Methode ausprobiert. In diesem Fall würden die Diagramme folgendermaßen aussehen







2.2.2 Dimensionalitätsreduktion



1. Welches Land ist nach dieser Darstellung Deutschland am ähnlichen?

Südkorea ist Deutschland, gemäß der Isomap, am ähnlichen.

2. Warum entspricht die hier dargestellte Ähnlichkeit nicht der im oben erzeugten Dendrogramm?

Das Dendrogram berücksichtigt alle Merkmale (5 Ressourcen), wohingegen bei der Isomap eine Dimensionsreduktion auf die zwei aussagekräftigsten (höchste Varianz) Ressourcen vorgenommen wird.

2.3 Überwachtes Lernen: Schätzung der CO2-Emmission

2.3.1 Feature Selection

1. Welche 3 Merkmale haben den stärksten Einfluss auf das Ausgabemerkmal CO2-Emmission? Wie groß sind die vom Programm ausgegebenen Scores?

Öl, Kohle und Hydro haben den größten Einfluss auf das Ausgabemerkmal CO2-Emission.

Energieform	Score
Öl:	220.01
Gas:	46
Nuclear:	34.57
Kohle:	378.27
Hydro:	79.05

2.3.2 Regression mit Epsilon-SVR

1. Optimieren Sie die SVR-Parameter C und Epsilon so dass der Score in der Kreuzvalidierung minimal wird. Welche Werte für C und Epsilon liefern das beste Ergebnis?

Für C = egal und epsilon = 0.1 ist der Score minimal.

2. Für das SVR-Objekt können die Koeffizienten der linearen Abbildung, welche durch die trainierte SVR realisiert wird, ausgegeben werden: meineSVR.coef_. Notieren Sie diese Koeffizienten für die beste SVR.

####Koeffizienten####

[-3.07583949 -2.34385976 -3.95681056 -0.01631254 0.01185543]]

####MAD####

0.409842569042741078355618355999

3. Welchen Aufschluss geben diese Koeffizienten über den Einfluss der einzelnen Eingangsmerkmale auf das Ausgangsmerkmal?

Um so höher der Wert der Koeffizienten, desto mehr beeinflussen sie das Ausgabemerkmals. Die letzten beiden Merkmale, Hydro und Nuclear, weisen höhere Koeffizienten auf. Im Allgemeinen bewirkt eine hohe Varianz innerhalb eines Merkmals eine bessere Trennbarkeit der Daten. Ein Blick auf unsere Boxplot-Grafiken bestätigt diese hohe Varianz der beiden Merkmale.

4. Wie groß ist die mittlere absolute Differenz zwischen Soll- und Ist-Ausgabe für die beste SVR?

Diskutieren Sie dieses Ergebnis.

Der MAD beträgt ca. 0.41.

Das bedeutet, dass die Schätzung nur ca 0.41% neben dem Sollwert liegt. Der Grund dafür ist, dass wir nur die Trainingsdaten voraussagen. Wir haben das Modell nicht auf unbekannte Testdaten angewandt, was eine qualitativ hochwertigere Aussage zuließe.

2.4 Visualisierung des Clusterings in Google Maps



Auffälligkeiten zur Grafik:

- Geographische Nähe innerhalb eines Clusters
- China, Indien und Südafrika liegen zwar geographisch nicht nah beieinander, gehören aber zu den Schwellenländern
- Mittelmeer-Raum und der Orient sowie Südost Asien bilden ein riesiges Cluster
- Industrieländer liegen auch in einem Cluster

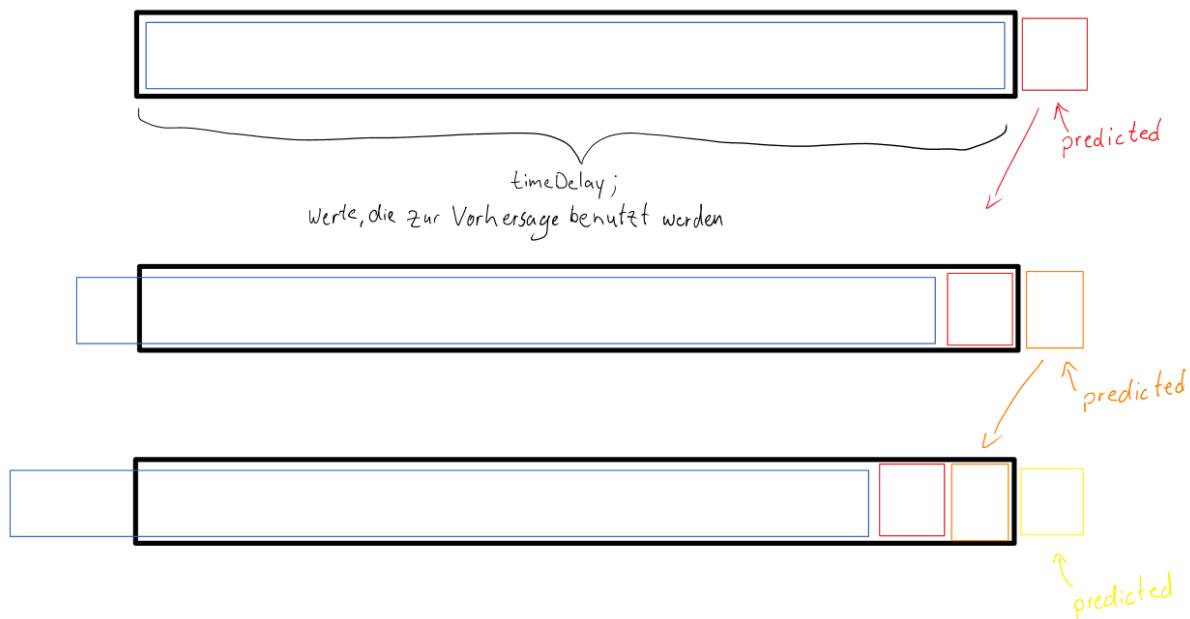
Versuch 3: Durchführung Teil 2: Vorhersage und Clustering auf Finanzdaten

3.1 Zeitreihenschätzung: Vorhersage des Aktienkurses

3.1.1 Datenbeschaffung

3.1.2. Kursvorhersage mit SVR

1. Überlegen Sie sich genau, wie die Datenvektoren des Vorhersagezeitraums (also die Vektoren, die der Methode `prediction()` des trainierten SVR-Modells übergeben werden), aufgebaut werden müssen.



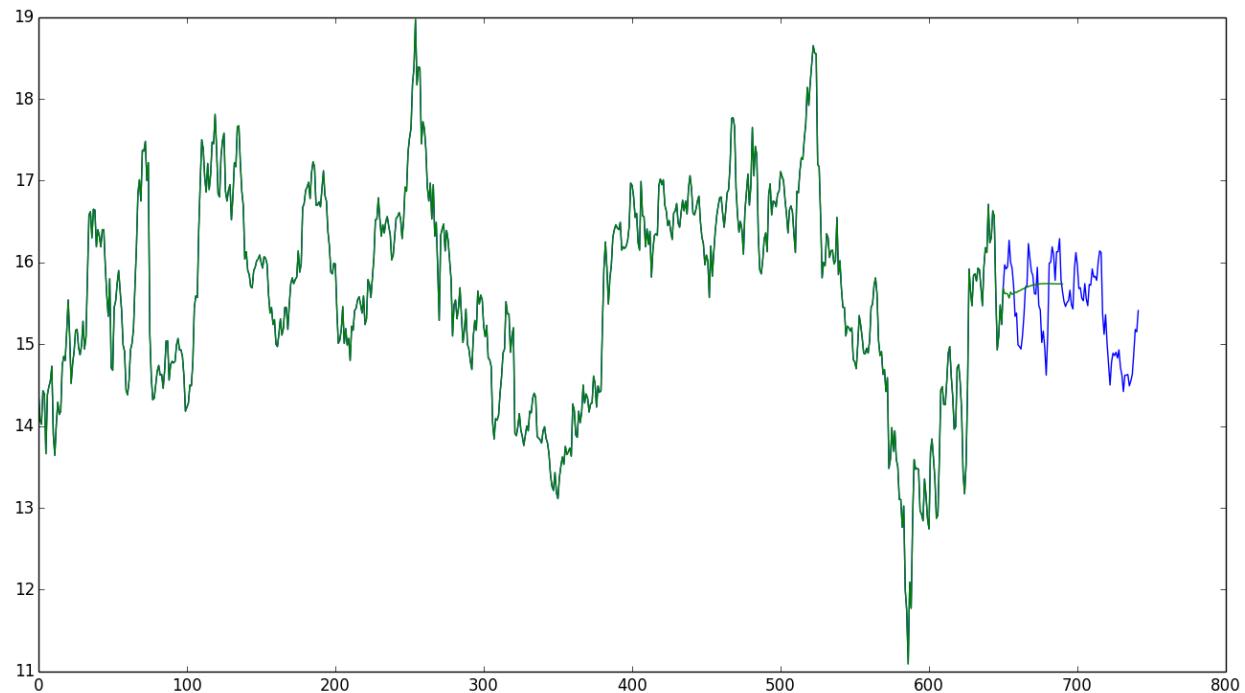
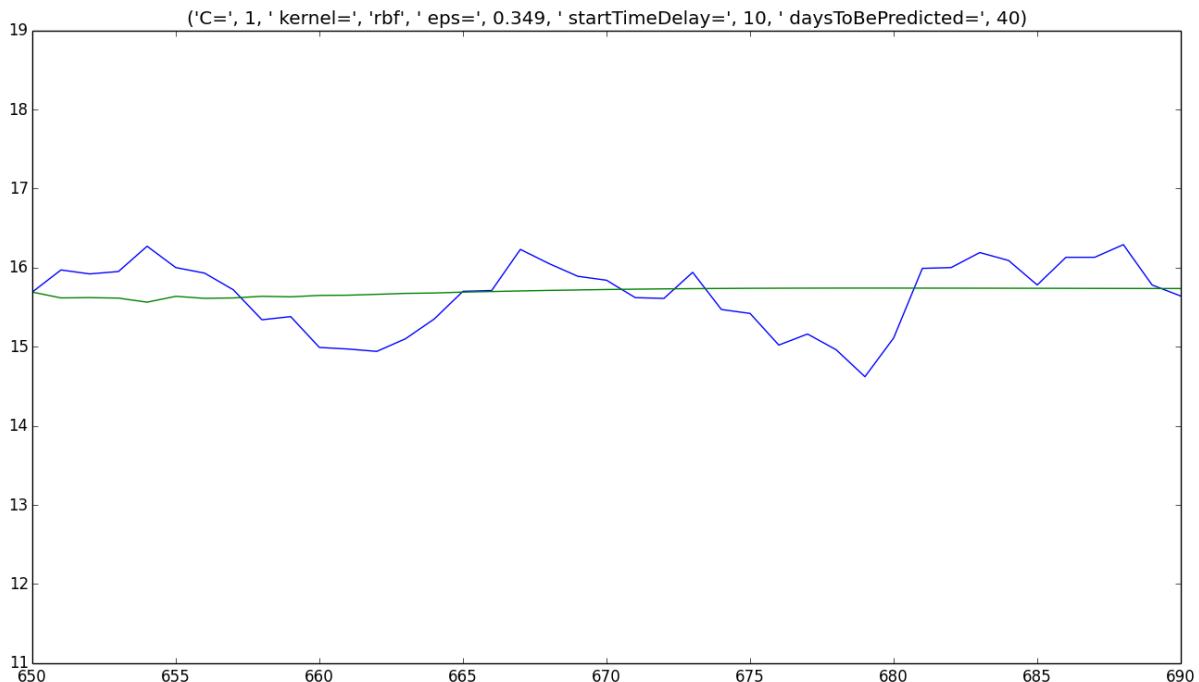
2. Für welche Werte von

- Time Delay
- SVR-Parameter C
- SVR-Parameter epsilon

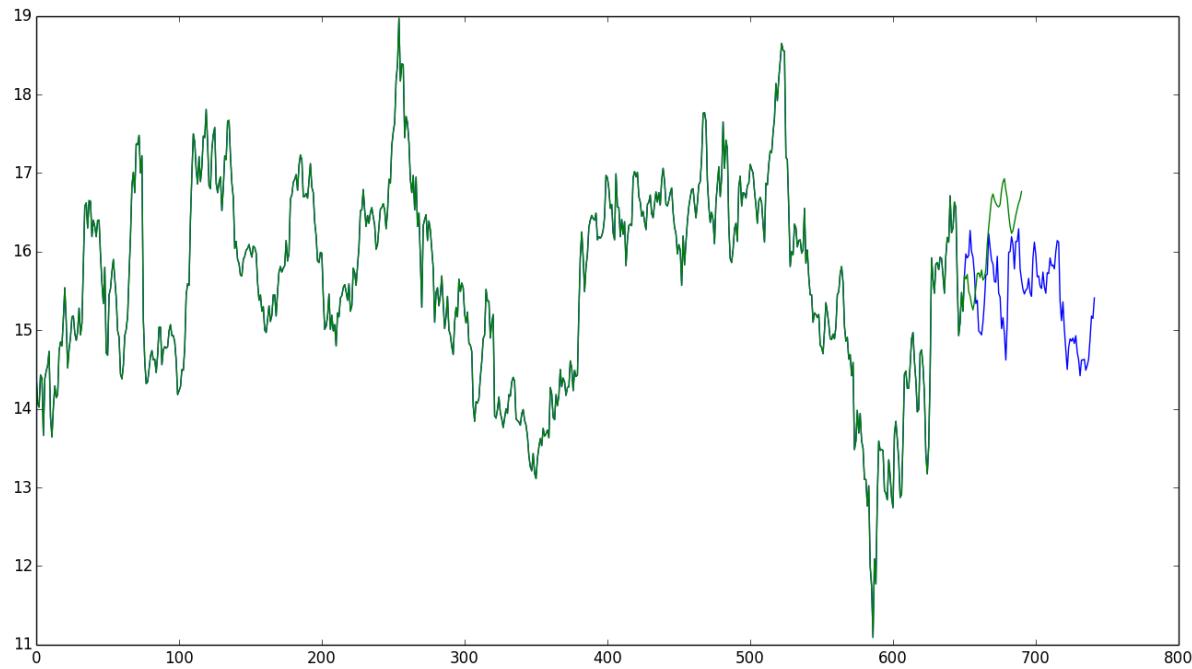
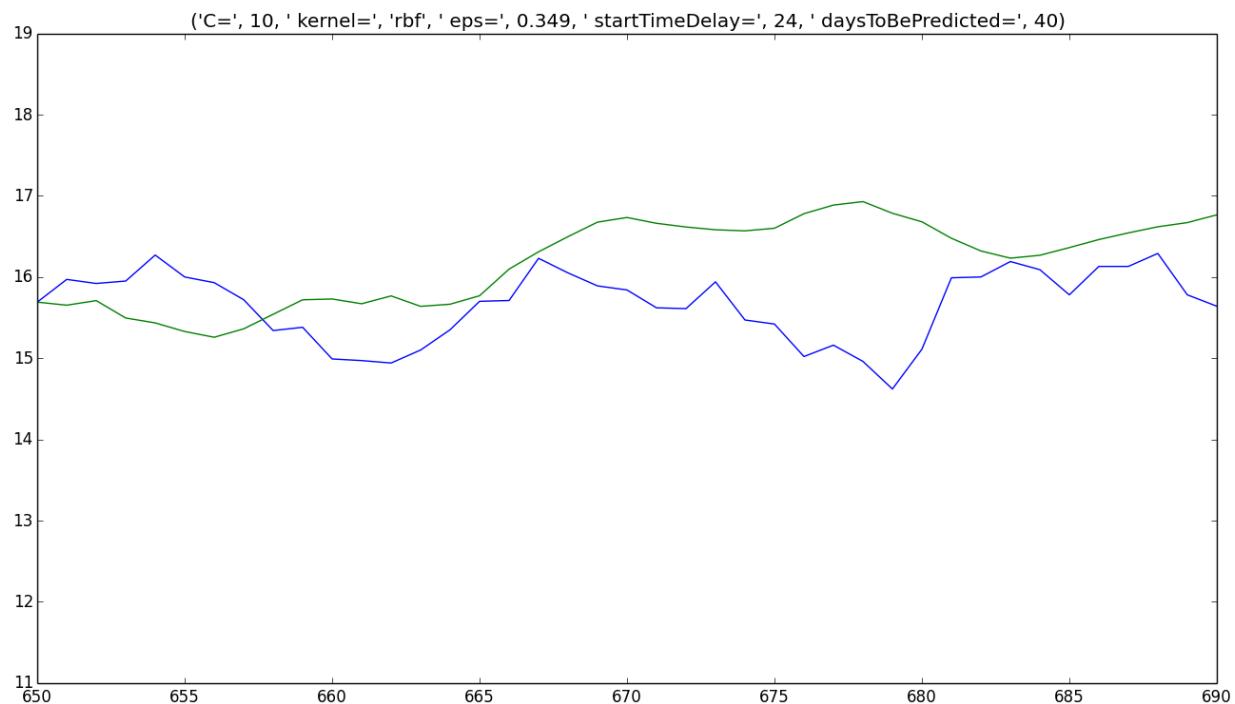
erreichen Sie die beste Vorhersage? Wie groß ist in diesem Fall der MAE?

Für Time Delay = 10, C = 1 und epsilon = 0,349 erreicht man die beste Vorhersage.

Der MAE = 0,37



Vorhersage mit noch nicht optimierten Werten C, eps, startTimeDelay:



Bewertung des Ergebnisses:

Durch ein iteratives Verfahren konnten jeweils die optimalen Werte der Parameter startTimeDelay, C und epsilon bezüglich des MAE gefunden werden, d.h. es wurde pro Parametereinstellung trainiert, der MAE berechnet und der Wert gefunden, an dem der MAE minimal ist.

Die dargestellte Kurve bildet die Schwankungen des Kursverlaufs nicht gut ab, was jedoch an der Wahl des MAE als Fehlermaß liegt. Der MAE ist im Bereich der vorhergesagten Werte durch die Wahl passender Parameter zwar minimal (über den gesamten Zeitraum in etwa eine Linie durch den Mittelpunkt), aber die Schwankungen werden nicht berücksichtigt. Die Schwankungen selbst resultieren jedoch bereits aus unvorhersehbaren Ereignissen. Durch das Finden der minimalen Kostenfunktion über den MAE als Fehlermaß kann zwar den eigentlichen Kursverlauf nicht abbilden kann. Aufgrund der Unvorhersehbarkeit kann dies jedoch gar nie erreicht werden. Die Grafik zeigt den langfristigen Trend der Aktie.

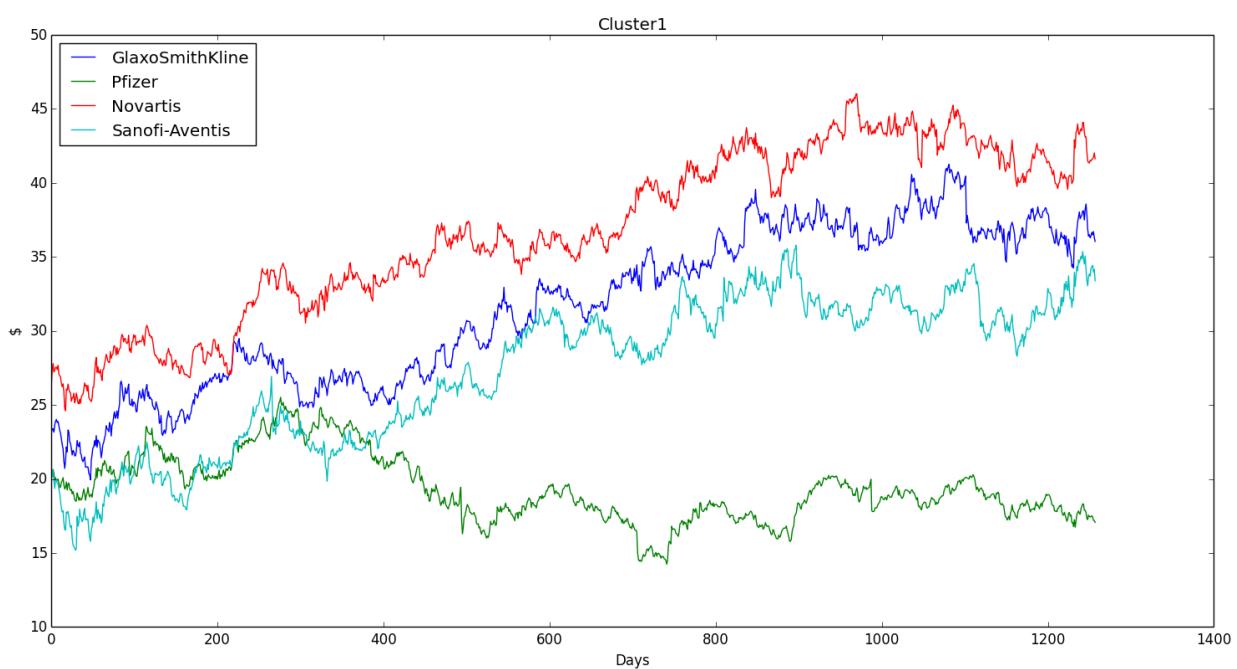
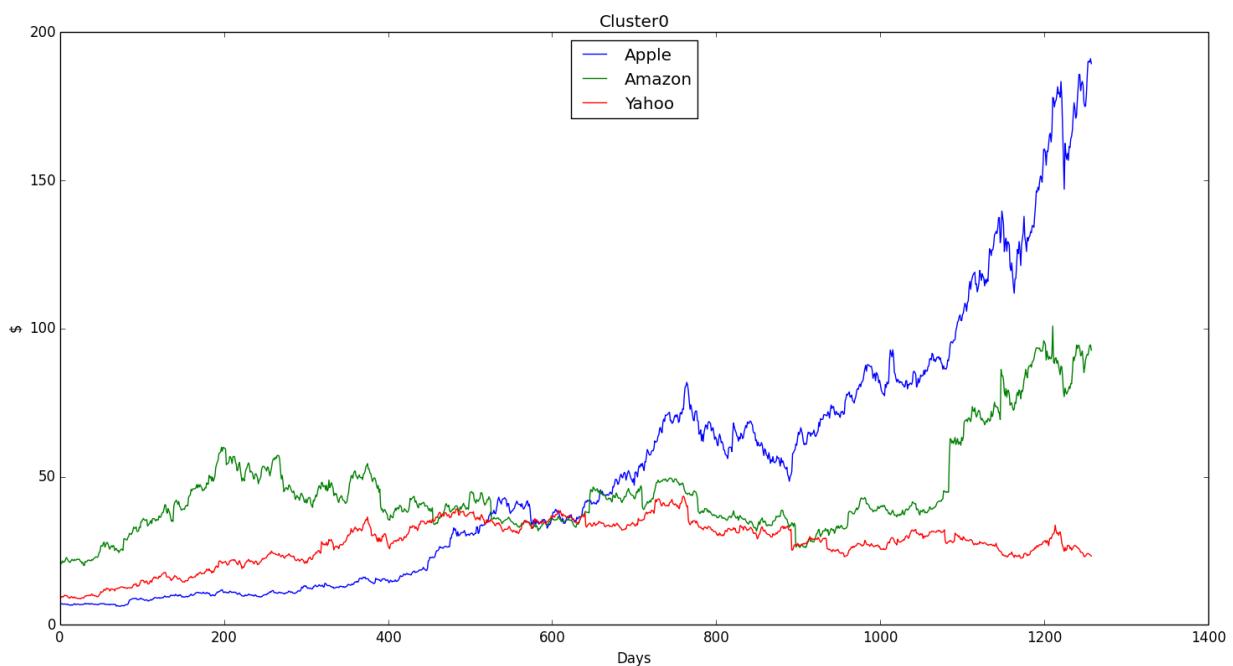
Im Fall der optimierten Parameter bringt der RBF-Kernel keinen echten Mehrwert gegenüber einem linearen Kernel, da ein linearer Kernel einen linearen Verlauf besitzt und die Grafik genau einen solchen linearen Verlauf für einen minimalen MAE über den gesamten Bereich zeigt. Würde man den MAE nicht über den gesamten Prediction-Zeitraum berechnen, sondern in kleinen Intervallen, so könnte eine Annäherung an den Kursverlauf erreicht werden, jedoch müsste hierfür für jeden Zwischenzeitraum extra mit speziell für den jeweiligen Zwischenzeitraum optimierten Parametern trainiert werden und das ist wiederum nicht praktikabel.

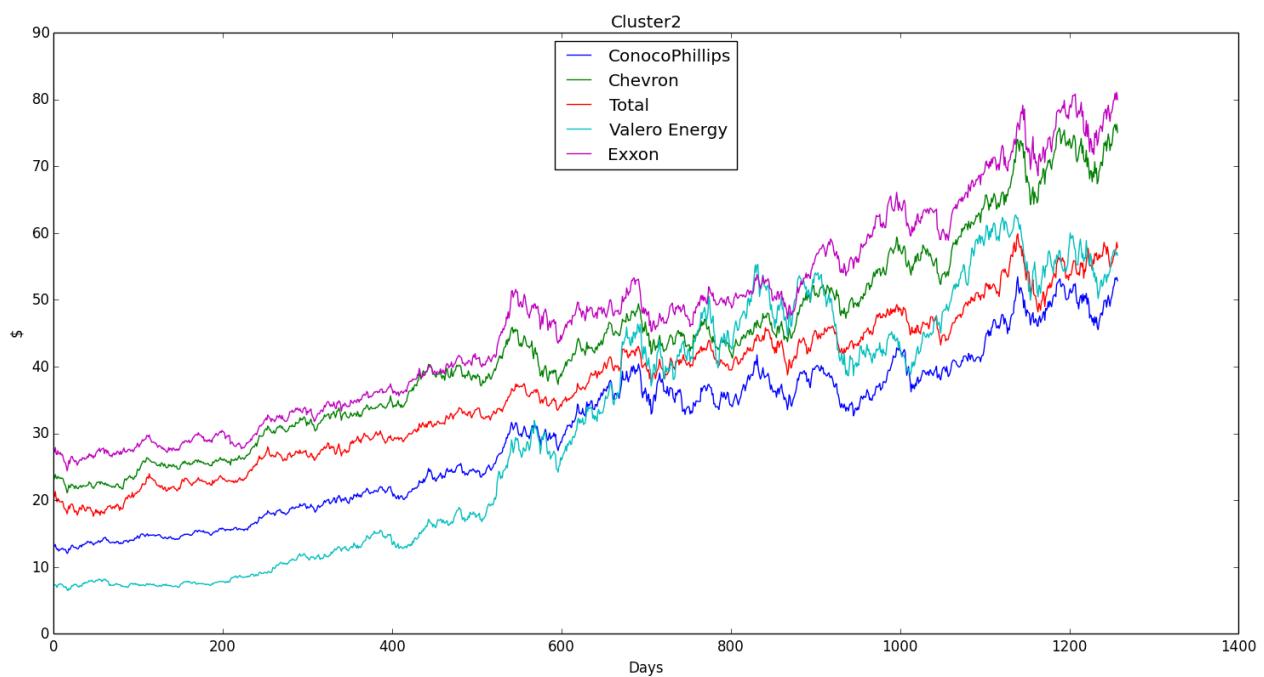
3.2 Clustering der Aktienkursverläufe

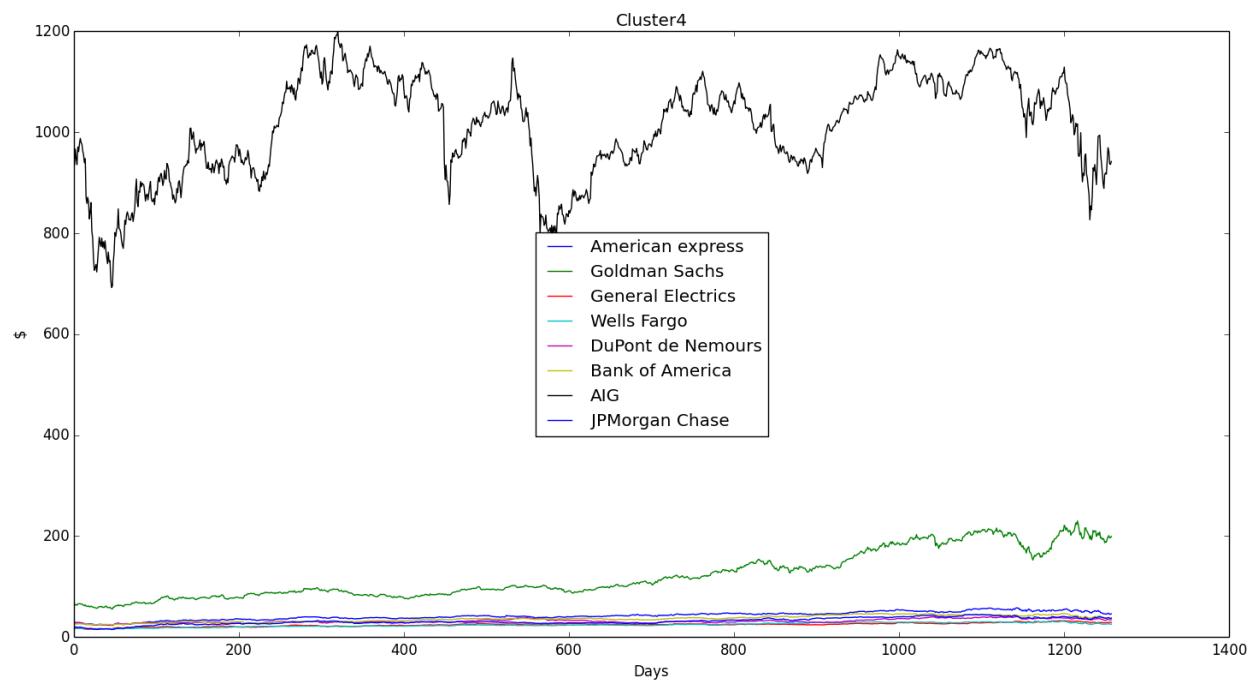
1. Analysieren Sie die Clusterzuweisungen. Gehören die Unternehmen, welche in einem Cluster zusammenfallen, irgendwie zusammen?

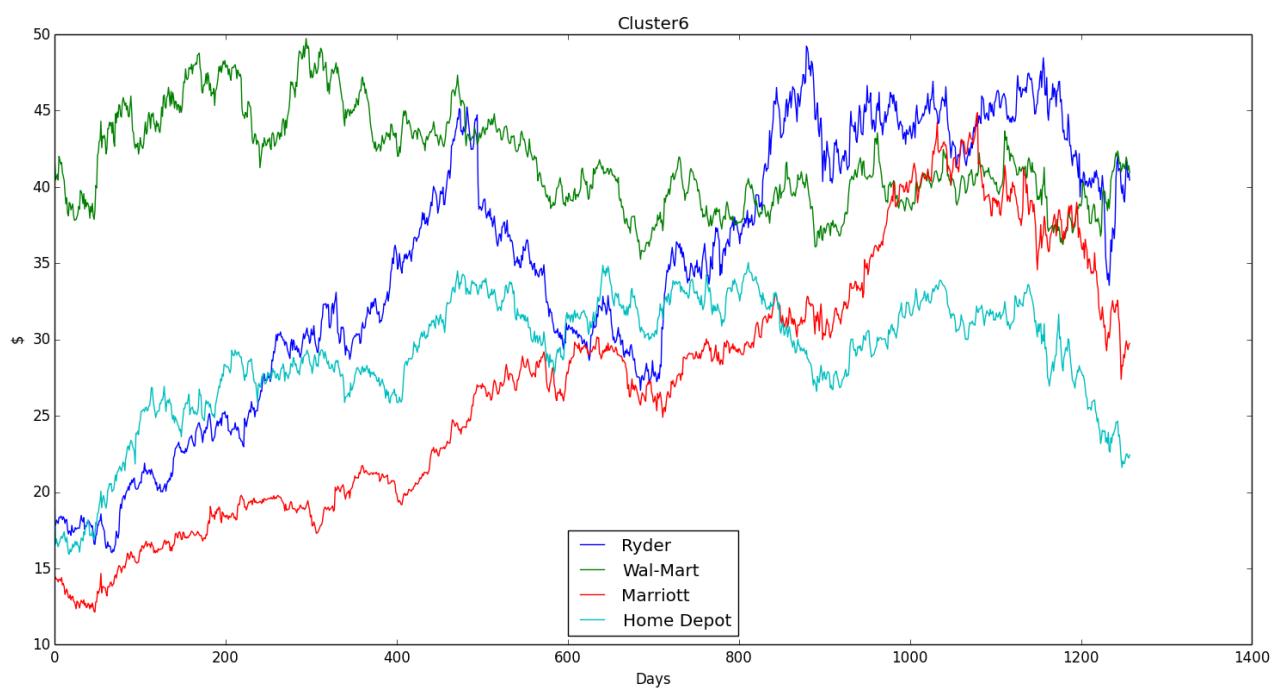
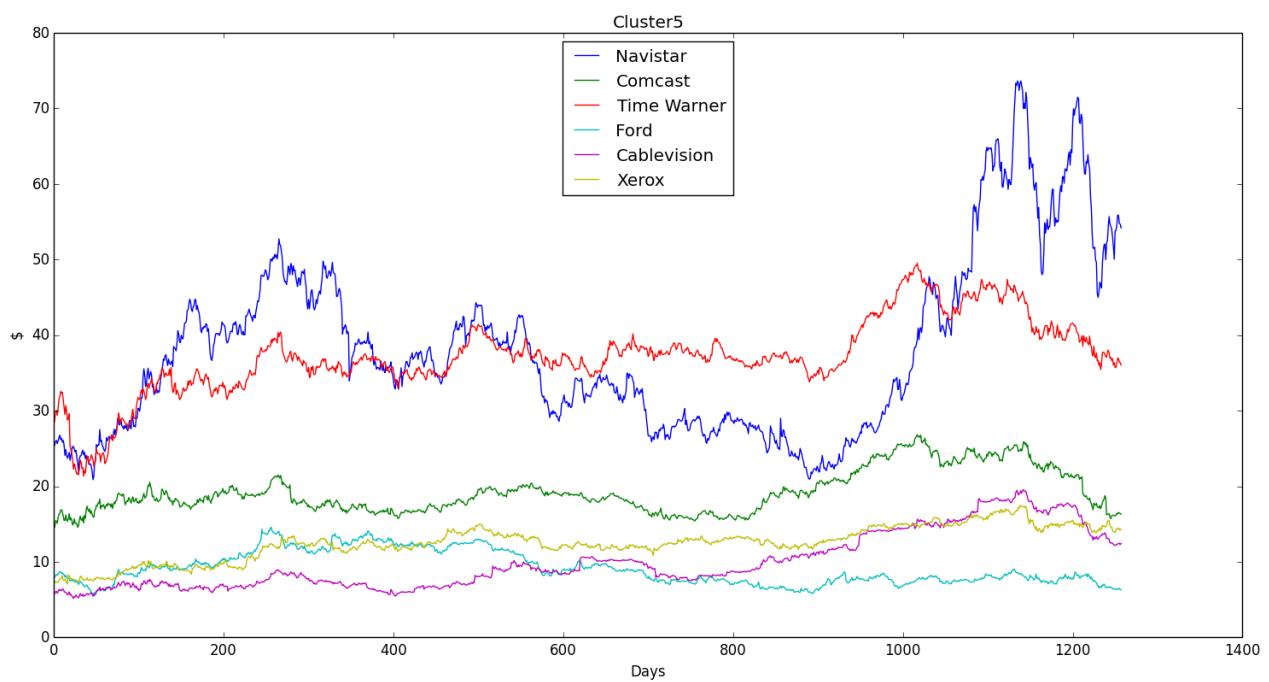
Ja, die Unternehmen, die in einem Cluster zusammenfallen, gehören überwiegend zu einer Branche:

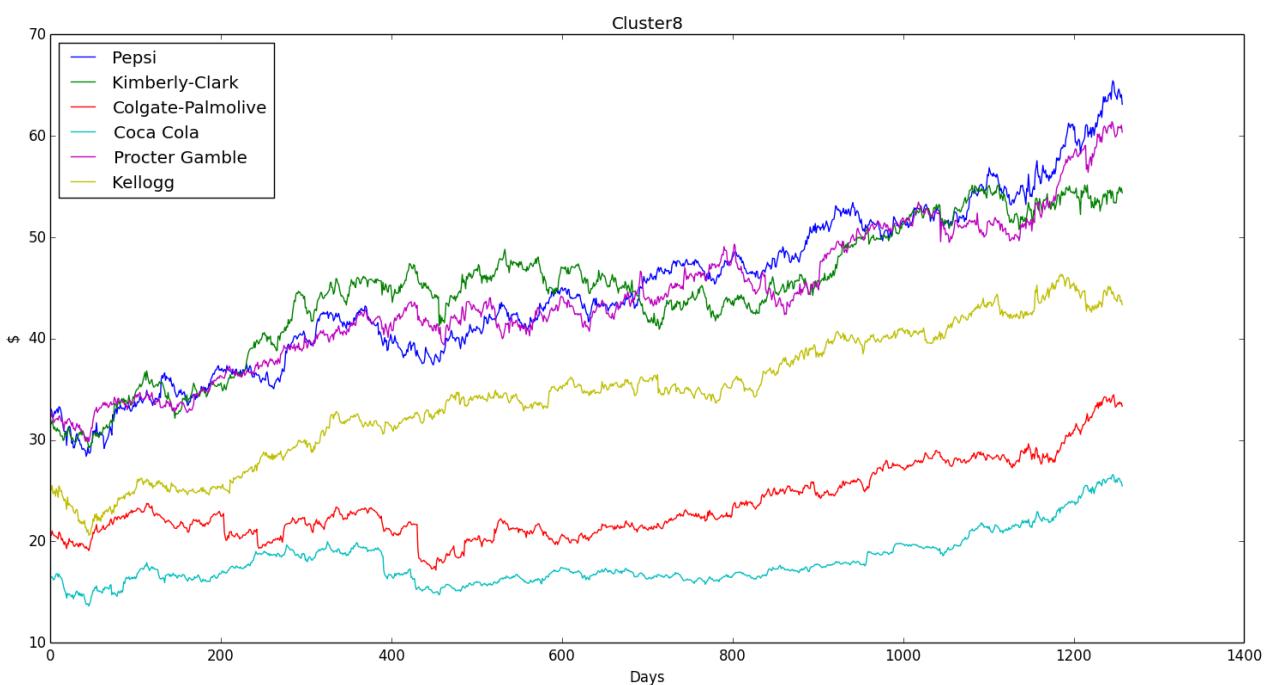
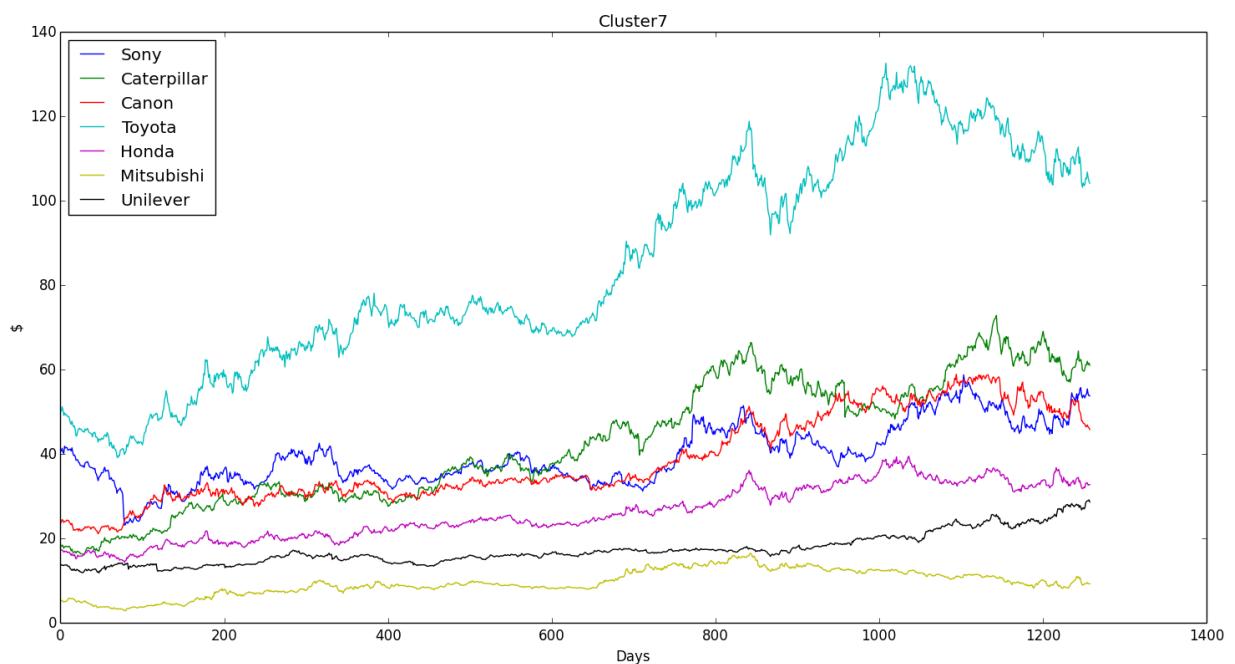
Cluster 0:	Internetunternehmen
Cluster 1:	Pharmazieunternehmen
Cluster 2:	Energieversorger
Cluster 3:	Einzelhandelsunternehmen
Cluster 4:	Finanz-/Versicherungsunternehmen
Cluster 5:	Medienunternehmen und Automobilhersteller
Cluster 6:	Einkaufsketten
Cluster 7:	Hersteller von Fahrzeugen, Verbrauchsgüter
Cluster 8:	Hersteller von Verbrauchsgüter
Cluster 9:	Software-Unternehmen
Cluster 10:	Rüstungskonzerne

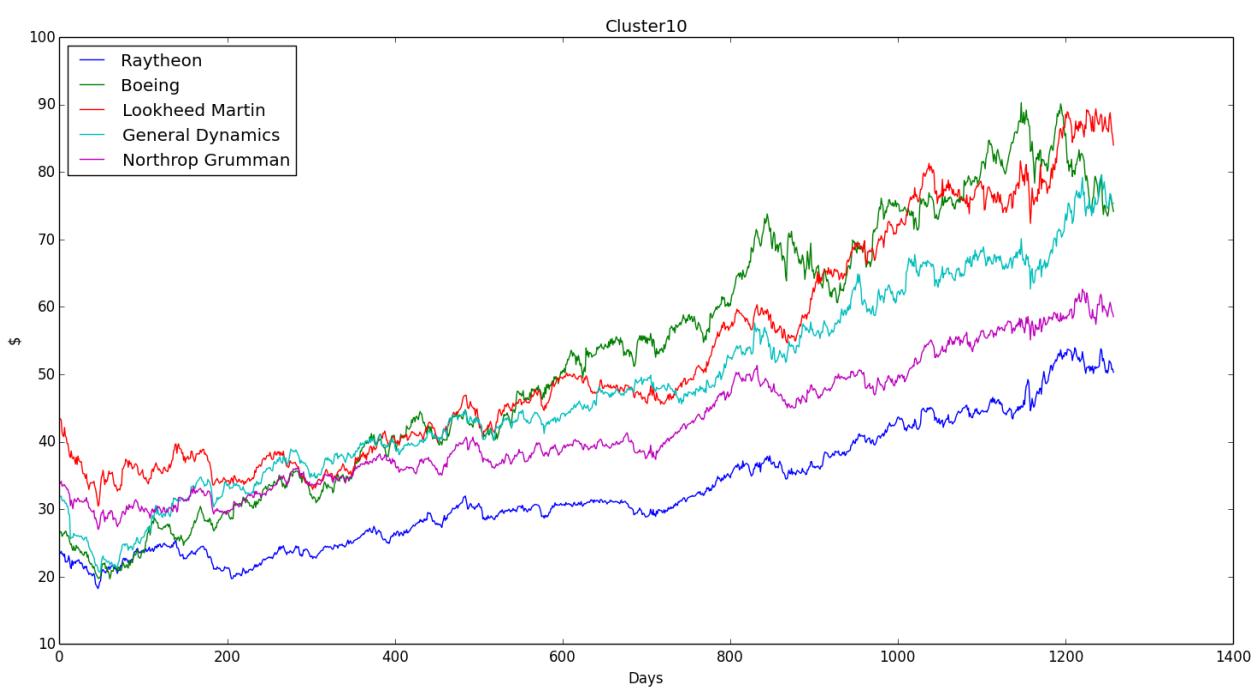
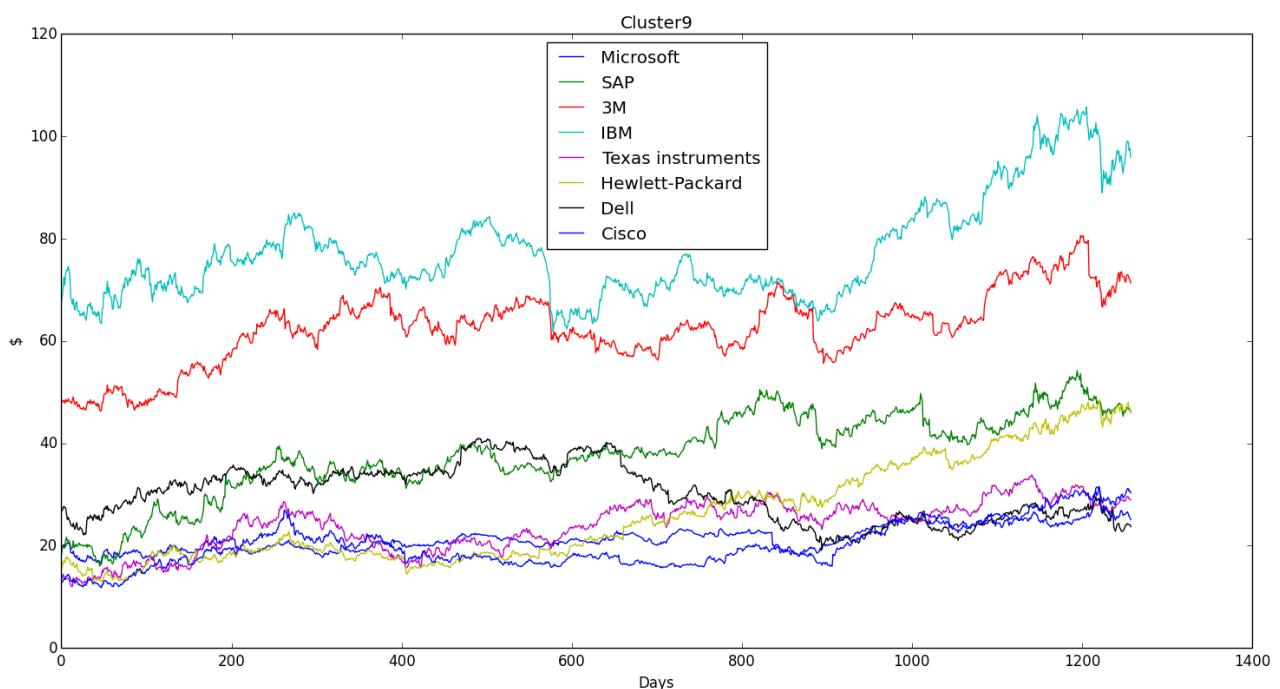












Anhang

Aus der in Kapitel 1.3 beschriebenen Theorie

1. Erklären Sie den Sinn der Transformation innerhalb der Data Mining Prozesskette.

Die Transformation von vorverarbeiteten Daten dient dazu, dass Eingabeattribute gemäß ihres Einflusses auf das Ausgabeattribut bewertet werden und durch eine Auswahl der wichtigsten Features eine Komplexitätsreduktion (z.B. durch eine Dimensionsreduktion durch Isomap oder PCA) durchgeführt wird, wodurch auch eine bessere Modellbildung ermöglicht wird.

2. Worin besteht der Unterschied zwischen überwachtem und unüberwachtem Lernen.

Der Unterschied zwischen den beiden Arten des Lernens besteht darin, dass beim überwachten Lernen jedem Datensatz sowohl in der Trainings- als auch in der Testphase ein bekannter Ausgabewert (Sollausgabe) zugeordnet wird, welche z.B. in Form eines Klassenlabels (Klassifikation) oder eines numerischen Werts (Regression) vorliegen kann. Der Lernprozess geschieht in einer iterativen Form von Prediktion, Vergleich zwischen Ist- und Sollausgabe und einer anschließenden Gewichteanpassung der Zuordnungsfunktion. Beim unüberwachten Lernen ist dagegen keine Sollausgabe bekannt. Es werden Klassengrenzen gefunden, indem eine Gruppierung ähnlicher Datensätze (Clustering) vorgenommen wird oder Assoziationen zwischen Merkmalen/Features gefunden werden.

3. Beschreiben Sie die Unterschiede zwischen Klassifikation, Regression und Clustering. Nennen Sie für diese 3 verschiedenen Verfahren je ein Anwendungsbeispiel.

Klassifikation ist ein überwachtes Lernverfahren, das jedem Inputdatensatz einen diskreten Wert (Klasse) zuordnet. Anwendung: Objekterkennung. Regression ist ebenfalls überwachtes Lernen, jedoch wird statt eines diskreten Wertes jedem Datensatz ein numerischer Wert zugeordnet. Anwendung: Analyse von Aktienkursen. Clustering gehört zu den unüberwachten Lernverfahren, d.h. es ist keine Sollausgabe pro Datensatz bekannt. Eine Trennung von Daten in sogenannte Cluster wird über die Berechnung von Distanzen zwischen den Datensätzen erreicht. Einander ähnliche Datensätze weisen eine geringe Distanz auf und werden dem selben Cluster zugeordnet wohingegen einander unähnliche Datensätze verschiedenen Cluster zugeordnet werden. Anwendung: Kundenanalyse bei Onlineshops.

Python Allgemein:

1. Was ist eine Python List-Comprehension?

Wenn man Elementen aus einer bestehenden Liste eine neue Liste erstellen möchte, Python unterstützt eine sehr viel flexiblere Syntax, die für gerade diesen Zweck geschaffen wurde: die sogenannten List Comprehensions. Die folgende List Comprehension erzeugt aus einer Liste mit ganzen Zahlen eine neue Liste, die die Quadrate dieser Zahlen enthält:

```
>>> lst = [1,2,3,4,5,6,7,8,9]
>>> lst2 = [x**2 for x in lst]
```

```
>>> lst2
[1, 4, 9, 16, 25, 36, 49, 64, 81]
```

2. Wie importiert man Daten aus einem Textfile?

```
file = open("filename.txt", "r")
fileContent= file.read()           #fileContent is a string
file.close()
```

3. Wie speichert man Daten aus Python in ein Textfile?

```
fileOut= open("fileName2.txt","w")
fileOut.write("this is some text")
fileOut.close()
```

4. Wie hängt man an eine Python-Liste die Elemente einer zweiten Liste an?

s.append(x) hängt x an das Ende der Liste an,
 >>> s.append(x)
 >>> s
 [5, 3, 4, 8, 2, 1, 3, 3, 4, 2, [1, 2, 3]]

s.extend(x) hängt die Elemente von x an das Ende der Liste an,
 >>> s.extend(x)
 >>> s
 [5, 3, 4, 8, 2, 1, 3, 3, 4, 2, 1, 2, 3]

Numpy:

1. Nennen Sie zwei verschiedene Möglichkeiten ein Numpy-Array zu erzeugen.

```
import numpy as np
a = np.array([1,2,3,4])
x = np.array(range(24))
```

oder mit matrix
 a = np.matrix('1 2 3 4')

2. Wie legt man ein (3;4)-Array mit ausschließlich 0-Einträgen an?

```
import numpy as np
a1 = np.zeros( (3,4) ) #(3,4) - Tuple
```

3. Wie ruft man die Anzahl der Dimensionen, die Anzahl der Elemente pro Dimension und den Datentyp der Arrayelemente ab?

a = np.arange(15).reshape(3, 5) #reshape legt das Array in einer bestimmten Form an:
 als

3x5 Matrix

```
print a.shape      #Anzahl Elemente pro Dimension
print a.ndim       #Dimension
print a.dtype.name #Datentyp der Elemente
print a.itemsize   #Größe des Datentyps
```

4. Wie wandelt man ein (3 ;4)-Array in ein (2;6)-Array um?

```
a1 = np.zeros( (3,4) )
a1 =a1.reshape((2,6))
```

5. Wie transponiert man ein zweidimensionales Array?

```
a1.transpose()
```

6. Wie multipliziert man zwei Arrays elementweise?

```
np.multiply(A,B)
```

7. Wie führt man eine Matrixmultiplikation zweier zweidimensionaler Arrays A und B aus? Welche Bedingungen müssen A und B erfüllen, damit überhaupt eine Matrixmultiplikation durchgeführt werden kann?

```
A[m,n]
B[m,n]
Bedingung: An = Bm !
Man nützt die Matrix-Klasse:
A = np.matrix('1.0 2.0; 3.0 4.0')
print A
B = np.matrix('5.0 7.0')
B= B.T
print B
print A*B
#oder
print np.dot(A,B)
```

8. Wie greift man auf das Element (2;3) in einem Array A zu? Wie greift man auf die erste Spalte, wie auf die erste Zeile dieses Arrays zu?

```
a[2,3]
a[:1] # erste Zeile
a[4:] # Zeile 1-4
a[:,0] #erste Spalte
```

9. Wie berechnet man die Quadratwurzel aller Elemente eines Arrays?

```
a = np.sqrt(a)
```

10. Wie legt man eine flache Kopie, wie eine tiefe Kopie eines Arrays an?

```
deep copy: d = a.copy()
flache Kopie: c = a.view()
```

Pandas:**1. Wie wird ein Numpy-Array in einen Pandas-Dataframe geschrieben? Wie legt man dabei die Spaltenbezeichnungen und einen Index an?**

```
indices = ["here","are","your","indices","!"]
pd.DataFrame(nparray, index=indices,columns=list('A'))
```

2. Wie kann auf einzelne Spalten, wie auf einzelne Zeilen eines Pandas Dataframes zugegriffen werden?

Spalten:

df.A

df['A']

Zeilen:

df[0:1]

df.loc[indices[0]]

3. Wie können Pandas Dataframes sortiert werden?

df.sort(columns='A')

4. Wie kann zu einem bestehenden Dataframe eine neue Spalte hinzugefügt werden?

df['f']=[0,0,0,0,0]

5. Wie werden Daten aus einem .csv File in einen Pandas Dataframe geschrieben?

pd.read_csv('foo.csv')

6. Wie wird ein Pandas Dataframe in einem .csv File abgelegt?

df.to_csv('foo.csv')

Matplotlib:

1. Wie erzeugt man mit Matplotlib einen Plot, wie er in Abbildung 4 dargestellt ist?

```
import numpy as np
import matplotlib.pyplot as plt
t = np.arange(0.0, 7.0, 0.1)
s = np.sin(0.32*np.pi*t)
line, = plt.plot(t, s, 'o')
plt.setp(line, color='r')
```

```
plt.ylabel('sin(x)')
plt.xlabel('x')
plt.title('Sinusfunktion')
plt.grid(True)
plt.ylim(-1,1)
plt.show()
```

2. Wie kann man mehrere Graphen in einen Plot eintragen?

Durch die Funktion subplot(numrows, numcols, fignum)

Beispiel:

```
plt.figure(1)      # ein Schaubild
plt.subplot(211)   # 2 Zeilen(Diagramme), 1 Spalte ,1. Diagramm
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k')
```

```
plt.subplot(212)   # das ist das zweite Diagramm
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
```

```
plt.show()
```

3. Wie erzeugt man mit Matplotlib ein Bild, das 12 Subplots in 3 Zeilen und 4 Spalten geordnet enthält

```
import numpy as np
import matplotlib.pyplot as plt
def f(t):
    return np.exp(-t) * np.cos(2*np.pi*t)
t1 = np.arange(0.0, 5.0, 0.1)
t2 = np.arange(0.0, 5.0, 0.02)
```

```
plt.figure(1)
plt.subplot(341)
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k')
plt.subplot(342)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.subplot(343)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.subplot(344)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.subplot(345)
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k')
plt.subplot(346)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.subplot(347)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.subplot(348)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.subplot(349)
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k')
plt.subplot(3,4,10)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.subplot(3,4,11)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.subplot(3,4,12)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.show()
```

4. Wie erzeugt man mit Matplotlib ein Histogramm?

```
import numpy as np
import matplotlib.pyplot as plt
mu, sigma = 100, 15
x = mu + sigma * np.random.randn(10000) # the histogram of the data

n, bins, patches = plt.hist(x, 50, normed=1, facecolor='g', alpha=0.75)
# 50 = Anzahl der Linien; alpha= zeigt Transparenz an; normed=
```

```
plt.xlabel('Smarts')
plt.ylabel('Probability')
```

```

plt.title('Histogram of IQ')
plt.text(60, .025, r'$\mu=100,\ \sigma=15$')
plt.axis([40, 160, 0, 0.03])
plt.grid(True)
plt.show()

```

5. Wie erzeugt man mit Matplotlib einen Boxplot?

```

import numpy as np
import matplotlib.pyplot as plt
from pylab import *
spread= rand(50) * 100
center = ones(25) * 50
flier_high = rand(10) * 100
flier_low = rand(10) * -100
data =concatenate((spread, center, flier_high, flier_low), 0)
boxplot(data,0,'rs',0)      # data;  notched box plot (0 oder 1); 'darstellungsweise der
                           # Ausreißer', hier rote Vierecke; horizontal = 0 oder vertikal= 1
plt.show()

```

Scipy:

1. Geben Sie kurz die Schritte an, die für die Durchführung eines hierarchischen Clustering mit Scipy notwendig sind.

```

#Berechnung des hierarchischen Clustering anhand der Distanzmatrix 'distMatrix'
hierclust = hierarchy.linkage((distMatrix,method="complete"))
#Anzahl Cluster
n_clusters = 15
#Einteilung der Instanzen in die mit .linkage gefundenen Cluster
clustlabels = hierarchy.fcluster(hierclust, t=n_clusters, criterion="maxclust")
clustlabels = clustlabels -1
#Ausgabe aller Instanzen pro Cluster
for cl in range(n_clusters):
    print '-' * 20 + 'Cluster' + str(cl) + '-' * 20
    ind = find(clustlabels == cl)
    for a in ind:
        print fileids[a]

```

Scikit Learn:

1. Sklearn stellt u.a. eine umfassende Bibliothek von Klassen für das überwachte Lernen bereit. Mit welchem Methodenaufruf werden diese Klassen trainiert? Mit welchem Methodenaufruf können die trainierten Modelle auf neue Eingabedaten angewandt werden um denentsprechenden Ausgabewert zu berechnen?

```

classifier = svm.SVC(gamma=0.001)
classifier.fit(digits.data[:n_samples/2], digits.target[:n_samples/2]) # Training
predicted = classifier.predict(digits.data[n_samples/2:]) # Test

```

2. Mit welchem Leistungsmaß kann die Qualität eines Regressionsmodells bewertet werden? Wie wird dieses Leistungsmaß mit Sklearn berechnet?

- Durch Mean Square Error (MSE) oder durch Mean Absolute Difference (MAD)
- MSE und MAD kann zum einen mit dem Modul metrics aus sklearn und zum anderen mit Standard Numpy Funktionen berechnet werden kann (Beide Möglichkeiten führen zum gleichen Ergebnis):

```

mse1=1.0/numInstances*np.sum((predictedOutput-targets)**2)    # mit Numpy
mse2=metrics.mean_squared_error(predictedOutput,targets);      # mit sklearn

mad1=1.0/numInstances*np.sum(np.abs(predictedOutput-targets))
mad2=1.0/numInstances*metrics.pairwise.manhattan_distances(predictedOutput,targets)
print "Mean Squared Error 1 %2.3f" % (mse1)
print "Mean Squared Error 2 %2.3f" % (mse2)
print "Mean Absolute Difference 1 %2.3f" % (mad1)
print "Mean Absolute Difference 2 %2.3f" % (mad2)

```

Ausgabe::

```

Mean Squared Error 1 17.506
Mean Squared Error 2 17.506
Mean Absolute Difference 1 3.439
Mean Absolute Difference 2: 3.439
→ Im Mittel liegt also die Schätzung 3.439 % neben dem Sollwert.

```

3. Mit welchem Leistungsmaß kann die Qualität eines Klassifikationsmodells bewertet werden? Wie wird dieses Leistungsmaß mit Sklearn berechnet?

```

print "Confusion matrix:\n%s\n" % metrics.confusion_matrix(expected, predicted)
print "Classification report \n%s" % (metrics.classification_report(expected, predicted))

```

Ausgabe:

Confusion matrix:

```

[[87 0 0 0 1 0 0 0 0 0]
 [ 0 88 1 0 0 0 0 0 1 1]
 [ 0 0 85 1 0 0 0 0 0 0]
 [ 0 0 0 79 0 3 0 4 5 0]
 [ 0 0 0 0 88 0 0 0 0 4]
 [ 0 0 0 0 0 88 1 0 0 2]
 [ 0 1 0 0 0 0 90 0 0 0]
 [ 0 0 0 0 0 1 0 88 0 0]
 [ 0 0 0 0 0 0 0 0 88 0]
 [ 0 0 0 1 0 1 0 0 0 90]]

```

Classification report

	precision	recall	f1-score	support
0	1.00	0.99	0.99	88
1	0.99	0.97	0.98	91
2	0.99	0.99	0.99	86
3	0.98	0.87	0.92	91

4	0.99	0.96	0.97	92
5	0.95	0.97	0.96	91
6	0.99	0.99	0.99	91
7	0.96	0.99	0.97	89
8	0.94	1.00	0.97	88
9	0.93	0.98	0.95	92
avg / total	0.97	0.97	0.97	899

In der Confusion Matrix gibt der Eintrag in Zeile i, Spalte j an, wie häufig ein Element der Klasse i, als Bild der Klasse j erkannt wurde. Der Classification Report gibt für alle Klassen die Parameter Precision, Recall und F1-Score aus. (werden aus der Confusion Matrix berechnet)

- Precision: Welcher Anteil der als Klasse i erkannten Elemente gehört tatsächlich zur Klasse i.
- Recall: Welcher Anteil der Klasse i Bilder wurde als Klasse i-Bild erkannt.
- F1-Score = $(\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$

4. Was versteht man unter x-facher Kreuzvalidierung und wie wird diese mit Sklearn durchgeführt?

Die Unterteilung in Trainings- und Testdaten ist willkürlich (-> Führt eine andere Unterteilung zu anderen Performancewerten? ja!)

- Durch x-facher Kreuzvalidierung kann dieses Problem teilweise kompensiert werden
- gesamte Datenmenge in x disjunkte Partitionen unterteilt (typische Werte für x liegen im Bereich 4-10)
- Dann wird in x Iterationen jeweils eine neue Partition als Testdatensatz und die restlichen x-1 Partitionen als Trainingsdatensatz gewählt
- In jeder Iteration wird mit den x-1 Trainingspartitionen ein Modell trainiert und mit der verbleibenden Testpartition das Modell getestet
- Die endgültige Performance ergibt sich dann als Durchschnitt über die Performancewerte aller x Partitionen