Fragen zu Versuch 1 (Energy Data)

Aus der in Kapitel 1.3 beschriebenen Theorie

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

Ziel der Transformation ist es, vorverarbeitete Daten so zu transformieren, dass nicht nur die Datenmenge reduziert wird, sondern auch Mustererkennung oder Modellbestimmung vereinfacht wird.
Neben der Reduktion der Attribute gehört auch das Normalisieren der Attribute dazu.

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

Bei *überwachtem Lernen* wird nach jeder Iteration geprüft, ob die Eingangsdaten und die erwarteten Ausgangsdaten (Sollausgabe) übereinstimmen und das Lernen kann bewertet werden.
Beim *unüberwachten Lernen* gibt es nur Eingangsdaten, aber keine Sollausgabe.

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

Klassifikation wird beim überwachten Lernen eingesetzt und zwar wird ein diskretes Ausgabeattribut zurückgegeben. *Bsp: Bereiche für Typische Klassen.*

Regression wird beim überwachten Lernen eingesetzt und liefert zu jedem möglichen Ausgabeattribut einen numerischen Funktionswert zurück. Die Trennung erfolgt durch eine mathematische Funktion.
Bsp: Bei der Gesichtserkennung kann von Wahrscheinlichkeiten ausgegangen werden, welche Person auf dem Bild ist.

Clustering wird beim unüberwachten Lernen eingesetzt und liefert Gruppen von Eingabewerten zurück, die sich im Bezug auf ihre Attribute relativ ähnlich sind.
Bsp: Bildsegmentierung für Video-Codierungen

Python Allgemein:

1. Was ist eine Python List-Comprehension?

Mit List Comprehension kann man auf verkürzte Weise Lambda Funktionen einsetzen, um Listen zu erzeugen.

```
squares = [ (x*x) for x in range(2,10,2) ]
points = [ (x,y) for x in range(1,3) for y in range(2,4) ]
```

Ergebnis:

```
squares = [4, 16, 36, 64]
points = [(1, 2), (1, 3), (2, 2), (2, 3)]
```

Achtung: points hält einen Liste mit N x N Verknüpfungen der Liste x und der Liste y in Tupeln

2. Wie importiert man Daten aus einem Textfile?

```
f = open('rumspielen.py', 'r')
daten = f.read()
f.close()
```

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

```
f2 = open('textspeichern.txt', 'w')
f2.write("hallo")
f2.close()
```

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

```
arr1 = [1, 2, 3]
arr2 = [4, 5, 6]
arr1.extend(arr2)
```

Ergebnis:

```
arr1 =[1, 2, 3, 4, 5, 6]
```

Achtung:

- `arr1.append(arr2)` liefert `[1, 2, 3, [4, 5, 6]` zurück.
- `arr1 + arr2` liefert `[1, 2, 3, 4, 5, 6]` zurück, verändert aber arr1 nicht.

Numpy:

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

erzeugen.

```
import numpy as np
arr1 = np.array( [1, 17, 23] )
arr2 = np.arange(5)
```

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

```
import numpy as np
arr3 = np.zeros( (3, 4) )
```

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

- Anzahl Dimensionen: `arr1.ndim`
- Anzahl Elemente pro Dimension: `arr1.shape`
- Datentyp der Arrayelemente: `arr1.dtype`

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

```
import numpy as np
arr1 = np.floor(10*np.random.random((3,4)))
arr1.reshape(2,6)
```

5. Wie transponiert man ein zweidimensionales Array?

```
arr1.transpose()
```

6. Wie multipliziert man zwei Arrays elementweise?

```
arr1 * arr2
```

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 muss gleich viele Zeilen wie B Spalten haben, und umgekehrt.

```
arr1=np.floor(10*np.random.random( (2,3) ))
arr2=np.floor(10*np.random.random( (3,2) ))
arrayA.dot(arrayB)
```

8. Wie greift man auf das Element (2,3) in einem (4,4)-Array A zu? Wie greift man auf die erste Spalte, wie auf die erste Zeile dieses Arrays zu?

Funktioniert nur für numpy Arrays, Python Arrays sind eindimensional.

- Element(2,3): `arr[1,2]`
- Erste Spalte: `arr[:,0]`
- erste Zeile: `arr[0,:]`

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

Hierfür werden *Universal Functions* verwendet.
`np.sqrt( np.array( [1, 2, 4] ) )` liefert `array([ 1, 1.41421356, 2])`

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

```
a = np.array( [1, 2, 3] )
shallowCopy = a.view()
deepCopy = a.copy()
```

Pandas:

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

Eingabe

```
numpyArray = np.array([1,2],[3,4],[5,6])
df = pd.DataFrame(numpyArray, index=range(3), columns=list(['Clumn A'], ['Column B'
])
```

Ausgabe

```
   Column A  Column B
0         1         2
1         3         4
2         5         6
```

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

Spalten

```
df['A'] # column name
```

Zeilen

```
df[0:1] # slice of row
```

3. Wie können Pandas Dataframes sortiert werden?

```
df.sort(columns='A') # sort by column A
df.sort_index(axis=0, ascending=False) # sort by axis 0=y-axis 1=x-axis
```

flo: möglichweise bei df.sort auch zeilenweise?

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

```
df['Column Z'] = pd.Series([10,20,30],index=range(3))
```

Series ist dabei ein passendes Objekt zum Dataframe.

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

```
df.read_csv('source.csv')
```

Laut dokumentation erzeugt der default von read_csv bereits einen Dataframe mit einfach numerierten Zeilen.

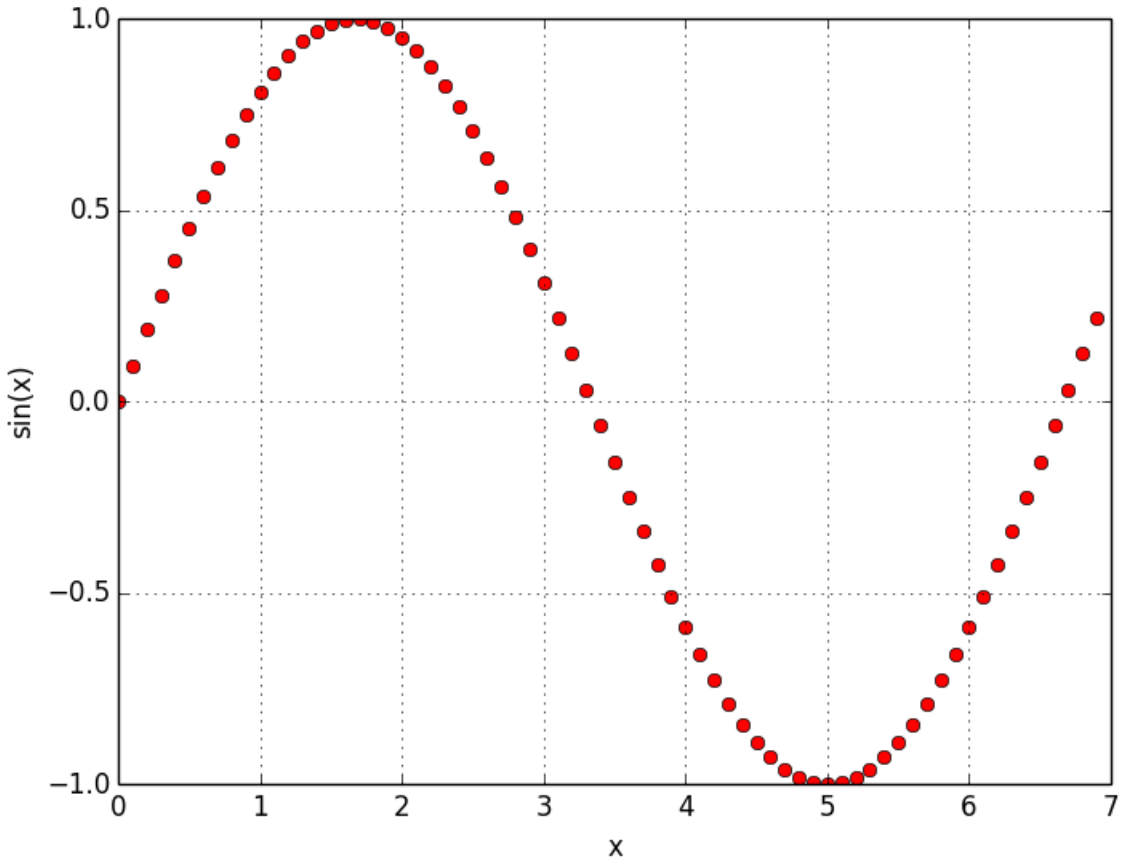
6. Wie wird ein Pandas Dataframe in einem .csv File abgelegt?

```
df.to_csv('target.csv')
```

Matplotlib:

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

```
range = np.arange(0.0,7.0,0.1) # range 0-7 with 0.1 intervals
plt.plot(range, np.sin(1*range), 'ro') # x=range, y=np.sin(1*range), ro=red dotted
plt.grid(True)
plt.title('Sinusfunktion')
plt.xlabel('x')
plt.ylabel('sin(x)')
plt.show()
```



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

Option 1 In einen Plot mehrere Funktionen eintragen:

```
plt.plot(range, np.sin(1*range), 'ro', range, range**2, 'ro')
```

Option 2 Mehrere Plots hintereinander generieren:

```
plt.plot(range, np.sin(1*range), 'ro')
plt.plot(range, range**2, 'ro')
```

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

Die Funktion plt.subplot kann 3 Parameter aufnehmen:

- numrows – Gesamtanzahl Zeilen
- numcols – Gesamtanzahl Spalten
- fignum – Position der Figur

Bsp.:

```
plt.subplot(4,3,1)
plt.subplot(4,3,2)
```

```
plt.subplot(4,3,3)
plt.subplot(4,3,4)
...
plt.subplot(4,3,10)
plt.subplot(4,3,11)
plt.subplot(4,3,12)
```

Achtung: unter jeden Subplot kommt noch der Plot selbst.

4. Wie erzeugt man mit Matplotlib ein Histogram?

Unter Verwendung der plt.hist Funktion.

```
plt.hist(x, 50, normed=1, facecolor='g', alpha=0.75)
```

- x – Array aus Werten
- 50 – Anzahl der Balken
- normed – Normalisierung True/False
- facecolor – Farbe der Balken ('g' = green)
- alpha – alpha Channel der Balkenfarbe (0->invisible)

5. Wie erzeugt man mit Matplotlib einen Boxplot?

```
x1 = np.random.normal(0,1,50) # generate random numbers
plt.boxplot([x1],vert=False) # plot x1 and set boxplot vertical
plt.show()
```

Boxplot kann noch mit weiteren Parametern angepasst werden.

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

```
1. Datenset als Liste (auch mehrdimensional) speichern.
2. Eine Distanzfunktion passend zum gestellten Problem finden.
3. Werte hierarchisch Clustern.
4. (optional) Dendogram erstellen.
```

Bsp. :

```
import scipy.cluster.hierarchy as sch
import scipy.spatial.distance as ssd
import matplotlib.pyplot as plt
from numpy.random import rand

plt.subplot(111)

# get random numbers and multiply one half by 2 for simulating groups
X = rand(10,100)
```

```
X[0:5,:] *= 2

# do pairwise distance calculation -> n!/(k!*(n-k)!)
Y = ssd.pdist(X)

# use linkage for Hierarchical Clustering
Z = sch.linkage(Y)

# Create Dendogram from Clustered Data
sch.dendogram(Z)

plt.show()
```

Mehr Informationen und ein Beispiel gibts bei stack overflow (http://stackoverflow.com/questions/21638130/tutorial-for-scipy-cluster-hierarchy) .

Scikit Learn:

Sklearn stellt u.a. eine umfassende Bibliothek von Klassen für das überwachte Lernen bereit.

Mit welchem Methodenaufruf werden diese Klassen trainiert?

```
regressor=linear_model.LinearRegression()
regressor.fit(features,targets)
```

Mit welchem Methodenaufruf können die trainierten Modelle auf neue Eingabedaten angewandt werden um den entsprechenden Ausgabewert zu berechnen?

```
predictedOutput = regressor.predict(features)
```

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

Mean Square Error *(Vorteil: normiert)*
`mse = metrics.mean_squared_error(predictedOutput, targets)`
und **Mean Absolute Difference**
`mad = 1.0 / numInstances * metrics.pairwise.manhattan_distances(predictedOutput,targets)`

Ohne scikit-learn:

- `mse2 = 1.0 / numInstances * np.sum( (predictedOutput-targets)**2 )`
- `mad2 = 1.0 / numInstances * np.sum(np.abs(predictedOutput-targets))`

Mit welchem Leistungsmaß kann die Qualität eines Klassifikationsmodells bewertet werden?

Indem die Präzision der erwarteten Ergebnisse berechnet wird (welcher Prozentsatz wurde richtig erkannt).

Wie wird dieses Leistungsmaß mit Sklearn berechnet?

`metrics.confusion_matrix` und `metrics.classification_report`

Beispiel aus tutorial zur Vorlesung (http://www.hdm-stuttgart.de/~maucher/Python/SklearnIntro/html/dataminingSklearn.html#beispiel-1-erkennung-von-handgeschriebenen-dezimalzahlen) :

```
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
expected = digits.target[n_samples/2:]                                 # Sollwerte
der Testdaten

print "-"*60
print "Classsifier : %s\n" %  (classifier)
print "Confusion matrix:\n%s\n" % metrics.confusion_matrix(expected, predicted)
print "Classification report \n%s" % (metrics.classification_report(expected, predi
cted))
```

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

Ist besonders sinnvoll, falls die Menge an Testdaten gering ist.
Was ist es: Es werden alle Daten in (disjunkte) Partitionen unterteilt, und bei jedem Durchlauf fungiert eine Partition als Testdatensatz für die Überprüfung nach dem Lernen, die restlichen Partitionen werden zum Training verwendet.
Dadurch erhält man gute Ergebnisse und kann das Modell mit verschiedenen Werten testen.