Übungen des Datenmanagement-Praktikums

Die Lösungen der Aufgaben sind online im Lernraum-Kurs (https://lernraum.fh-luebeck.de) zur Lehrveranstaltung per Upload einzureichen. Bei der Bezeichnung der Upload-Dateien bitte folgendes Muster einhalten:

DM_<Vorname>_<Nachname>_<Übungsnr>.<Dateierweiterung>

Zum Beispiel **DM_Thomas_Müller_3.ipynb** für die Lösung des Teams von Thomas Müller zu Übung 3 im Format eines Jupyter-Notebooks. Eine Abgabe pro Team durch einen beliebigen Teilnehmer ist ausreichend. Die sinnvollen Dateierweiterungen können zwischen den Übungen variieren (.xsd, .ipynb, .pdf, .zip, …).

Übung 1: XML, XML-Schema und XPath

Teil A: XML-Schema und XML-Dokument

Entwickle ein XML-Schema und ein zugehöriges, valides XML-Dokument für dein gewähltes Anwendungsszenario. Die im Szenario genannten Aspekte sollen sich in deinem XML-Dokument und XML-Schema grundsätzlich wiederfinden. Sinnvolle Änderungen, Erweiterungen oder Vereinfachungen sind erwünscht!

Nutze die Möglichkeiten, die XML-Schema bietet:

- Datentypen mit eingeschränkten Wertebereichen
- Default-Werte
- Definierte Reihenfolgen von Elementen, optionale und alternative Elemente
- Referenzielle Integrität
- Wiederverwendung von Typen

Trage in das XML-Dokument sinnvolle Beispieldaten ein. Für jeden Typ soll es min. 3 Beispiel-Datensätze geben. Prüfe die Validität des XML-Dokuments gegen das XML-Schema.

Lade das XML-Dokument (.xml) und das XML-Schema (.xsd) fristgerecht in den Lernraum hoch.

Teil B: Erstellen von XPath-Ausdrücken

Für jedes Anwendungszenario sollen XPath-Ausdrücke für die Fragestellungen auf der fogenden Seite formuliert werden. Erstelle die XPath-Ausdrücke und teste sie mit einem geeigneten XML-Editor, z.B. *XMLSpy* oder *Editix*. Wenn im Einzelfall kein XPath-Ausdruck zu finden ist, entwerfe alternativ eine XQuery-Anfrage.

Lade eine einfache Textdatei (.txt) mit den XPath-Ausdrücken und dem zugehörigen XML-Dokument, das zum Testen verwendet wurde, in den Lernraum hoch.

Anwendungsszenario Schauspielagentur

- Die durchschnittliche Dauer der Drehzeiten pro Schauspieler.
- Die Namen (als Text) aller Schauspieler, deren Geburtsdatum im Dezember ist.
- Die IDs der Schauspieler, deren Einsatzmöglichkeiten sich auf „Daily-Soaps“ beschränken und die nach 1990 geboren wurden.
- Alle Filme (als Text), in denen mindestens ein Schauspieler mitwirkt, dessen Gage unterhalb von 5.000 € liegt oder die Drehzeit mindestens 30 Min. beträgt.
- Die Summe der Gagen aller Schauspieler, die bereits in Filmen mitgewirkt haben, die dem Genre „Science-Fiction“ zuzuordnen sind.

Anwendungsszenario Baumarktkette

- Die Summe der Umsätze aller Baumärkte, deren Sortiment „Holzschrauben“ enthält.
- Alle Kunden (als Text), deren E-Mailadressen syntaktisch nicht korrekt sind.
- Alle Artikel der Baumärkte, die weniger als 10 € kosten. Es sollen nur Baumarkt-Standorte berücksichtigt werden, die auf „…burg“ enden.
- Alle Retouren-IDs aus dem Zeitraum vom 01.10.2018 bis 31.03.2019
- Die durchschnittlichen Ausgaben der Firmenkunden je Baumarkt im Jahr 2019.

Anwendungsszenario Teehersteller

- Die Summe der Verkaufsmengen aller Teesorten, die kein „a“ im Namen enthalten.
- Alle Kunden, die die Teesorte „Karamell“ bewertet haben und im Bewertungstext das Wort „gut“ geschrieben haben.
- Das Durchschnittsalter aller Kunden, die sich die Teesorte „Tiramisu“ gewünscht haben.
- Die Namen aller Kunden (als Text), die mindestens drei verschiedene Teesorten bestellt haben und in Dresden wohnen.
- Die IDs der Teesorten, die 2019 noch nicht einmal bestellt worden sind, 2018 eine Verkaufsmenge unter 5 erzielten und in den dazugehörigen Bewertungen das Wort „schlecht“ beinhalten.

Anwendungsszenario Hochschule

- Alle Namen der Lehrenden (als Text), deren Titel „Prof. Dr.“ beinhaltet
- Die Anzahl und die Namen der Studierenden, die das Modul „Datenbanken“ belegt haben und deren Note dort besser als 2,7 war.
- Das durchschnittliche Alter der Studierenden, die im Studiengang „BWL“ eingeschrieben sind und mindestens einmal in „Mathematik“ durchgefallen sind.
- Alle IDs der Evaluationen aus dem Modul „Programmierung“, die Wörter „toll“ oder „super“ enthalten.
- Alle Studiengänge (als Text), die das Modul „Mathematik“ anbieten und deren Durchschnittsnote bezüglich dieses Moduls besser als 3,0 ist.

Anwendungsszenario Fahrschule

- Die durchschnittliche Anzahl von Prüfungen pro Fahrschüler.
- Alle Fahrlehrer, die min. einen Termin haben, bei dem ein Fahrzeug mit Automatikgetriebe zum Einsatz kommt.
- Die Summe aller Theoriestunden (Termine ohne Fahrzeug) von Fahrschülern, die im Jahr 2000 geboren sind.
- Das Modell des zuletzt hinzugefügten Fahrzeugs für Führerscheinklasse B – unter der Annahme, dass neue Fahrzeuge immer am Ende angehängt werden.
- Die Namen derjenigen Fahrlehrer (als Text), deren IBAN-Angabe syntaktisch nicht korrekt ist.

Anwendungsszenario Fitnessstudio

- Der prozentuale Anteil aller Verträge mit einem Monatsbeitrag unter 20 Euro.
- Die Namen aller ausgebuchten Kurse.
- Die Anzahl der Termine, die vom Trainer Meier betreut werden und ein Gerät benötigen.
- Die Nummern aller Kurse, für die es keinen Trainer gibt, der sie durchführen kann.
- Die Namen (als Text) aller Kunden, deren Geburtsdatum im Juni oder Juli ist.

Anwendungsszenario Ferienwohnungsvermittlung

- Die durchschnittliche Aufenthaltsdauer in Tagen von Gästen aus Hamburg.
- Alle Ferienwohnungen, die über min. eine Sauna verfügen.
- Alle Namen von Gästen (als Text), die min. eine Ferienwohnung gebucht haben, die sich an ihrem Wohnort befindet.
- Den Gesamtpreis der Buchung mit der Nummer 1 – unter der Annahme, dass das Von-Datum der Buchung genau dem Von-Datum eines Saisonpreis-Eintrags entspricht und die Buchung sich auch nicht in die nächste Saison erstreckt.
- Die Namen derjenigen Gäste, deren IBAN-Angabe syntaktisch nicht korrekt ist.

Anwendungsszenario Gebrauchtwagenplattform

- Die Preise (als Text) aller Fahrzeuge, für die es min. 5 Finanzierungsangebote mit einer Kreditsumme von jeweils über 10.000 Euro gibt.
- Die Modellbezeichnungen derjenigen Fahrzeuge, die im Schnitt mehr als 20.000 km pro Jahr gefahren wurden.
- Für alle Fahrzeuge des Herstellers VW jeweils den ersten Termin, der für eine Probefahrt vereinbart wurde – unter der Annahme, dass Termine in der Reihenfolge ihrer Vereinbarung abgelegt werden.
- Die Nachnamen aller Anwender, die jeweils mehr als 3 Rezensionen verfasst und bekommen haben.
- Die IDs aller Termine, die im Januar oder Februar liegen ist.

Übung 2: REST Web Services und JSON

Entwickle eine REST-API für dein gewähltes Anwendungsszenario, die in der Programmiersprache Java implementiert ist. Als Orientierung kann das unter Verwendung von JAX-RS/Jersey entwickelte Beispiel dienen, das in der Vorlesung vorgestellt worden ist. Es darf alternativ auch Spring Data eingesetzt werden. Der Zustand der Ressourcen soll server-seitig in einer Datenbank (z.B. mittels JPA oder Firebase) gespeichert werden.

Für min. 2 Datentypen des Anwendungsszenarios (= Klassen des Datenmodells) sollen die relevanten HTTP-Methoden (GET, POST, PUT, DELETE) in der API implementiert werden. Dabei soll min. 1 Typ auch ein Attribut aufweisen, das eine Menge eines weiteren eigenen komplexen Typs darstellt. Dieses Attribut soll eingebettet ausgegeben, gespeichert und gelöscht werden können.

Beispiel: *Eine Bestellung enthält Bestellpositionen als Attribut mit Listen-Typ. Die Listen-Elemente sind wiederum vom komplexen Typ Bestellposition.*

```
{
     "orderId": 100,
     "orderDate": "2016-12-31",
     "orderPositions": [
          {"productId": 10, "quantity": 3, "unitPrice": 100},
          {"productId": 20, "quantity": 1, "unitPrice": 250}
     ]
}
```

Teste die API mit einem REST-Client wie Postman (http://getpostman.com/) oder Insomnia (http://insomnia.rest/) und exportiere ein Skript, über das beispielhafte HTTP-Requests an die API gesendet werden können. Alternativ kann die API auch über JUnit-Testfälle getestet werden.

Lade eine Anwendung, die die REST-API implementiert, in einem Zip-Archiv in den Lernraum hoch. Das Zip-Archiv soll außerdem die Testfälle zur API enthalten.

Übung 3: Regression

In dieser Übung sollen Regressionsmodelle auf Basis verschiedener Beispieldaten entwickelt werden. Die grundsätzliche Aufgabenstellung ist für alle Datensätze gleich. Bitte wähle die Datenbasis zur Regression in Abhängigkeit deines Anwendungsszenarios aus den vorherigen Übungen:

Anwendungsszenario	**Datenbasis zur Regression**
Fahrschule, Gepäckmanager	*Water Usage of Production Plant* http://www.statsci.org/data/general/water.html
Fitnessstudio, Videothek	*Effect of Punishment Regimes on Crime Rates* http://www.statsci.org/data/general/uscrime.html
Music-Online-Store, Gebrauchtwagenplattform	*American Football Punters* http://www.statsci.org/data/general/punting.html
Ferienwohnungsvermittlung	*Mass and Physical Measurements for Male Subjects* http://www.statsci.org/data/oz/physical.html

Die Aufgaben sind in einem Jupyter-Notebook mit den Python-Bibliotheken Pandas und Scikit-Learn zu bearbeiten. Eine Teilaufgabe ist handschriftlich auf Papier zu lösen.

Aufgabenstellung

a. Importiere die Daten mittels der oben genannten URL in ein Pandas-DataFrame.
b. Auf Basis der Trainingsdaten soll ein Zielmerkmal Y aus den anderen Merkmalen X vorhergesagt werden. Das Zielmerkmal kann der folgenden Tabelle entnommen werden.

Daten zur Regression	**Zielmerkmal Y**	**Anzahl der Merkmale in X**
Water Usage of Production Plant	*Water*	4
Effect of Punishment Regimes on Crime Rates	*Crime*	15
American Football Punters	*Distance*	5 (exklusive *Hang*)
Mass and Physical Measurements for Male Subjects	*Mass*	10

c. Visualisiere den linearen Zusammenhang zwischen Y und jedem einzelnen Merkmal in X in einem Streuungsdiagramm über die Methode `seaborn.pairplot`.
d. Bestimme die Korrelationskoeffizienten zwischen Y und jedem Merkmal in X über die Methode `pandas.DataFrame.corr` und interpretiere die Zusammenhänge.
e. Wähle die ersten 3 Datensätze aus und berechne für diese den Korrelationskoeffizienten zwischen Y und dem ersten Merkmal in X handschriftlich. Prüfe, ob sich der gleiche Wert im Jupyter-Notebook ergibt.
f. Bestimme das Bestimmtheitsmaß für ein multivariates lineares Regressionsmodell, das alle Merkmale in X zur Bestimmung von Y berücksichtigt. Verwende dazu die Methoden `sklearn.linear_model.LinearRegression.fit/predict` und `sklearn.metrics.r2_score`.
g. Wähle das Merkmal aus X aus, über das sich Y am besten anhand der vorliegenden Trainingsdaten vorhersagen lässt. Visualisiere für das gewählte Merkmal einfache polynomiale Regressionsmodelle für die Polynome 1. bis 3. Ordnung über die Methode `seaborn.regplot` und vergleiche deren Bestimmtheitsmaße. Beurteile das Underfitting und Overfitting der Modelle.

Die nicht bearbeiteten Datensätze eignen sich zur späteren Klausurvorbereitung.

Lade die Lösungen zu den Aufgaben in einem Zip-Archiv, das sowohl das Jupyter-Notebook (.ipynb) als auch den handschriftlichen Teil (.pdf oder .jpg) enthält, in den Lernraum hoch.

Übung 4: Clusteranalyse und Klassifikationsbäume

Teil 1: Grundlagen der Clusteranalyse (Sample Size = 7)

Diese Aufgabe ist handschriftlich auf Papier zu bearbeiten, wobei die Lösung gerne in Python rechnerisch geprüft werden kann.

Bitte wähle eine Datenbasis in Abhängigkeit deines Anwendungsszenarios aus den vorherigen Übungen. Auf der folgenden Seite sind für jede Datenbasis jeweils 7 Datensätze mit 2 Merkmalen, eine unvollständige Distanzmatrix und die initialen Clusterzentren für das k-Means-Verfahren gegeben.

Anwendungsszenario	**Datenbasis für die Clusteranalyse**
Fahrschule, Gepäckmanager	A
Fitnessstudio, Videothek	B
Music-Online-Store, Gebrauchtwagenplattform	C
Ferienwohnungsvermittlung	D

Aufgabenstellung

a. Vervollständige die Distanzmatrix. Das Distanzmaß ist der euklidische Abstand.
b. Zeichne die Datensätze in ein Streuungsdiagramm ein.
c. Führe ein agglomeratives Clustering mit *Single Linkage*, *Complete Linkage* und *Centroid Linkage* durch und erstelle jeweils das vollständige Dendogram. Vergleiche die Unterschiede bei der Clusterbildung.
d. Führe ein partitionierendes Clustering nach dem k-Means-Verfahren durch. Verwende dazu die gegebenen initialen Clusterzentren. Skizziere die Verschiebung der Clusterzentren in einem Streuungsdiagramm.

7 Datensätze mit 2 Merkmalen, unvollständige Distanzmatrix, initiale Clusterzentren

	Daten	Distanzmatrix	Clusterzentren
A	**0 1 2 3 4 5 6** **X** 12 6 4 1 0 3 6 **Y** 2 1 10 6 3 0 12	**0 1 2 3 4 5 6** **0** - 6.1 11.3 11.7 ? 9.2 11.7 **1** - 9.2 7.1 6.3 ? 11.0 **2** - ? 8.1 10.0 2.8 **3** - 3.2 6.3 ? **4** - ? 10.8 **5** - 12.4 **6** -	(1,1) und (3,3)
B	**0 1 2 3 4 5 6** **X** 7 0 1 8 5 10 12 **Y** 2 9 12 1 7 5 0	**0 1 2 3 4 5 6** **0** - 9.9 11.7 1.4 ? 4.2 5.4 **1** - 3.2 11.3 5.4 ? 15.0 **2** - ? 6.4 11.4 16.3 **3** - 6.7 4.5 ? **4** - ? 9.9 **5** - 5.4 **6** -	(8,3) und (10,4)
C	**0 1 2 3 4 5 6** **X** 1 0 12 3 6 3 9 **Y** 7 12 12 0 0 4 10	**0 1 2 3 4 5 6** **0** - 5.1 12.1 7.3 ? 3.6 8.5 **1** - 12.0 12.4 13.4 ? 9.2 **2** - ? 13.4 12.0 3.6 **3** - 3.0 4.0 ? **4** - ? 10.4 **5** - 8.5 **6** -	(1,1) und (6,3)
D	**0 1 2 3 4 5 6** **X** 12 7 8 4 0 7 4 **Y** 2 0 1 8 5 5 12	**0 1 2 3 4 5 6** **0** - 5.4 4.1 10.0 ? 5.8 12.8 **1** - 1.4 8.5 8.6 ? 12.4 **2** - ? 8.9 4.1 11.7 **3** - 5.0 4.2 ? **4** - ? 8.1 **5** - 7.6 **6** -	(10,4) und (12,3)

Teil 2: Partitionierende Clusteranalyse und Klassifikationsbäume (Sample Size = 1000)

Diese Aufgabe ist in einem Jupyter-Notebook mit Python und Scikit-Learn zu bearbeiten.

Aufgabenstellung

a. Generiere 1000 künstliche Datensätze (*samples*) mit 5 metrischen Merkmalen (*features*) über die Scikit-Learn-Methode `make_blobs`. Jeder Datensatz soll einer von 4 Zielklassen (*centers*) zugeordnet sein.

```
make_blobs(n_samples=1000, n_features=5, centers=4, cluster_std=3, random_state=RS)
```

Zwecks Vergleichbarkeit der Ergebnisse soll der Zufallszustand (*random state*) in Abhängigkeit deines Anwendungsszenarios aus den vorherigen Übungen gewählt werden:

Anwendungsszenario	**Random State RS**
Fahrschule, Gepäckmanager	0
Fitnessstudio, Videothek	1
Music-Online-Store, Gebrauchtwagenplattform	2
Ferienwohnungsvermittlung	3

Stelle dir vor, dass die 5 erzeugten Merkmale messbaren Eigenschaften von 1000 Kunden entsprechen, die in 4 Gruppen wie z.B. Bonitäts- oder Prioritätsklassen segmentiert werden sollen.

b. Visualisiere die Abhängigkeiten zwischen diesen 5 Merkmalen über die Methode `seaborn.pairplot`. Verwende den Parameter `hue` der Methode um die Zielklassen in verschiedenen Farben darzustellen.

- Partitionierendes Clustering -

c. Führe ein partitionierendes Clustering nach dem k-Means-Verfahren mittels `sklearn.cluster.KMeans` durch. Setze dabei den Zufallszustand wie oben angegeben.
d. Berechne die Genauigkeit des Clustering als Anteil der korrekt zugeordneten Datensätze.
e. Gebe die Konfusionsmatrix (`sklearn.metrics.confusion_matrix`) für das Clustering mittels der Methode `seaborn.heatmap` aus und interpretiere das Ergebnis.

- Klassifikationsbaum -

f. Teile die Datensätze in Trainingsdaten (80%) und Testdaten (20%) mittels der Methode `sklearn.model_selection.train_test_split`. Setze dabei den Zufallszustand wie oben angegeben.
g. Trainiere einen Klassifikationsbaum mittels `sklearn.tree.DecisionTreeClassifier` anhand der Trainingsdaten. Das Unreinheitsmaß soll der Gini-Index (`criterion="gini"`) sein. Die minimale Verbesserung der Unreinheit für einen Split soll 0,05 (`min_impurity_decrease = 0.05`) betragen.
h. Visualisiere den resultierenden Klassifikationsbaum mittels der Methode `sklearn.tree.export_graphviz` (mit Parameter `filled=True`) und interpretiere das Ergebnis.
i. Rechne die Unreinheit gemäß Gini-Index des ersten Knotens nach.
j. Berechne die Verbesserung der Unreinheit durch den ersten Split.
k. Teste die Genauigkeit des Klassifikationsbaums anhand der Testdaten. Gebe die Konfusionsmatrix für den Klassifikationsbaum mittels der Methode `seaborn.heatmap` aus.

- Random-Forest-Klassifikator -

l. Trainiere einen Random-Forest-Klassifikator, der aus 20 Bäumen (`n_estimators`) bestehen soll, mittels `sklearn.ensemble.RandomForestClassifier`. Setze dabei den Zufallszustand wie oben angegeben.
m. Teste den Random-Forest-Klassifikator anhand der Testdaten und gebe wiederum die Genauigkeit und die Konfusionsmatrix aus. Vergleiche die Ergebnisse mit den vorherigen Genauigkeiten und Konfusionsmatrizen.
n. Erzeuge wie folgt einen Bericht der Klassifikation und erkläre die Begriffe der Ausgabe.

```
from sklearn.metrics import classification_report
print( classification_report(y_true = … , y_pred = … ) )
```

Lade die Lösungen zu den Aufgaben in einem Zip-Archiv, das sowohl die Jupyter-Notebooks (.ipynb) als auch die handschriftlichen Teile (.pdf oder .jpg) enthält, in den Lernraum hoch.

Übung 5: Datenanalyse mit Spark

Dokumentation zu Google Dataproc: https://cloud.google.com/dataproc/

Dokumentation zu Spark SQL und DataFrames: https://spark.apache.org/docs/latest/sql-programming-guide.html

Aufgabenstellung

1. Installiere einen Cluster zur verteilten Datenanalyse über den Dienst Dataproc auf der Google Cloud Platform .
2. Lade die folgenden Beispiel-Dateien eines Webshops in das HDFS des Clusters: https://console.cloud.google.com/storage/browser/oncampus/webshop/. Die Datei `iw_customer.txt` kann z.B. über die URL `gs://oncampus/webshop/iw_customer.txt` eingelesen werden.
3. Beantworte folgende Fragestellungen bzw. befolge die Anweisungen bezüglich dieses Webshops mittels Spark SQL/DataFrames in einem Jupyter-Notebook.
    - Wie viele Kunden-Datensätze gibt es in der Datei `iw_customer.txt`? Zeige die ersten 5 Kunden.
    - Entferne folgende Attribute aus dem DataFrame: 'owner', 'firstname', 'lastname', 'street', 'eMail'.
    - Entferne alle Kunden, die vor 1900 geboren worden sind, aus dem DataFrame.
    - Nenne das Attribut 'birthdate' in 'birthyear' um und speichere entsprechend nur noch das Geburtsjahr im DataFrame.
    - Ersetze im Attribut 'salutation' alle Werte 'Frau' durch 'weiblich' sowie 'Herr' durch 'männlich' und nenne das Attribute in 'gender' um.
    - Ergänze ein Attribut 'state', in dem das Bundesland zur PLZ gespeichert sein soll (s. `plz_mapping.txt`).
    - Persistiere das bisher konstruierte DataFrame als Tabelle 'customers'. Die Tabelle wird per Default im Hive-Metastore unter `/user/hive/warehouse/customers` abgelegt.
    - Leider wird in dem Webshop i.d.R. als Gast ohne Kundenkonto bestellt. Daher existieren viele Kunden-Dupletten. Alle Datensätze mit gleichem Wert im Attribut 'riskId' sollen als ein eindeutiger Kunde in einem neuen DataFrame 'uniqueCustomers' zusammengeführt werden.
    - Erstelle ein Kreisdiagramm, dass die Verteilung der Kunden je Geschlecht visualisiert.
    - Erstelle ein Balkendiagramm, dass die Verteilung der Kunden je Bundesland und je Geschlecht visualisiert. Die Verteilung soll absteigend sortiert nach Häufigkeit der Kunden je Bundesland sein.
    - Lese die Umsätze und Retouren für weitere dispositive Fragestellungen ein.
    - Gruppiere die Umsätze des Jahres 2011 je Monat und stelle sie in einem Balkendiagramm dar.
    - Wie viele Bestellungen (nicht Bestellpositionen) führt ein eindeutiger Kunde durchschnittlich durch?
    - Was ist die durchschnittliche Anzahl an Bestellpositionen und der durchschnittliche Gesamtbetrag einer Bestellung?
    - Ermittle die Anzahl der Bestellpositionen und die Anzahl der retournierten Positionen je Bundesland. Bestimme anschließend die Retourenquote (= Anzahl an Retourenpositionen / Anzahl an Bestellpositionen)

Lade das Jupyter-Notebook-Skript (.ipynb) in den Lernraum hoch.

Google Compute Engine

Der Dienst zum für virtuelle Maschinen (VMs) innerhalb der Google Cloud Platform heißt *Google Compute Engine (GCE)*. Die VMs können alternativ zum folgenden Vorgehen auf über da Web UI unter https://console.cloud.google.com/compute/ konfiguriert und gesteuert werden. Es ist aber einfacher die VMs über Skripte/Kommandos zu steuern, da sich das Starten und das Beenden mehrerer VMs auf diese Weise automatisieren lässt.

- Herunterladen und installieren des GCloud SDK von https://cloud.google.com/sdk/
- GCloud initialisieren,
  https://cloud.google.com/compute/docs/gcloud-compute/

```
gcloud init
```

- ggf. Firewall-Regel anlegen, wenn Ports ins Internet freigegeben werden sollen
  https://cloud.google.com/sdk/gcloud/reference/compute/firewall-rules/create

```
gcloud compute firewall-rules create jupyter-ports
  --allow tcp:8888 --target-tags=allow-jupyter-ports
```

- GCE-Instanzen erzeugen und starten,
  https://cloud.google.com/sdk/gcloud/reference/compute/instances/create

```
gcloud compute instances create node1 node2 node3
  --machine-type=n1-standard-1
  --image-project=ubuntu-os-cloud --image-family=ubuntu-1804-lts
  --boot-disk-size=10GB --boot-disk-type=pd-ssd
  --tags=allow-jupyter-ports
  --scopes=compute-rw
```

- GCE-Instanzen beenden, wenn nicht weiter benötigt

```
gcloud compute instances stop node1 node2 node3
```

Google Dataproc

- Der Dienst *Google Dataproc* (https://cloud.google.com/dataproc/) ermöglicht es, mehrere GCE-Instanzen zu starten, auf denen bereits Hadoop, Spark und ggf. Jupyter vorinstalliert ist.
  https://cloud.google.com/sdk/gcloud/reference/dataproc/clusters/create
  https://github.com/GoogleCloudPlatform/dataproc-initialization-actions/tree/master/jupyter

```
gcloud dataproc clusters create my-dataproc-cluster
  --metadata "JUPYTER_PORT=8124,JUPYTER_CONDA_PACKAGES=numpy:pandas:scikit-learn:matplotlib"
  --initialization-actions gs://dataproc-initialization-actions/jupyter/jupyter.sh
  --properties spark:spark.executorEnv.PYTHONHASHSEED=0,spark:spark.yarn.am.memory=1024m
  --worker-machine-type=n1-standard-4 --num-workers=3
```

- SSH-Tunnel zum internen Netzwerk der GCE-Instanzen aufbauen,
  https://cloud.google.com/dataproc/docs/concepts/accessing/cluster-web-interfaces

```
gcloud compute ssh my-dataproc-cluster-m --zone=us-west1-a -- -D 5000
```

Hadoop YARN:	http://my-dataproc-cluster-m:8088
HDFS NameNode:	http://my-dataproc-cluster-m:9870
Jupyter:	http://my-dataproc-cluster-m:8124

Folgend findet sich eine Beschreibung der Input-Dateien unter https://console.cloud.google.com/storage/browser/oncampus/webshop/.

iw_customer		Kunden
Spaltenname	**Datentyp**	**Inhalt**
owner	varchar	Shop-/Mandatenkennung
customerNo	varchar	Kundennummer
salutation	varchar	Anrede
firstname	varchar	Vorname (anonymisiert)
surname	varchar	Nachname (anonymisiert)
postcode	varchar	Postleitzahl
city	varchar	Wohnort
street	varchar	Straße (anonymisiert)
eMail	varchar	E-Mail (anonymisiert)
newsletter	varchar	Newsletter (1 = Ja, 0 = Nein)
birthdate	datetime	Geburtsdatum
riskID	varchar	ID der Bonitätsprüfung (dient der Identifikation eines Kunden)
credit	numeric	Höhe des zulässigen Kredits
credidLimit	varchar	1 = hat Kredit, 2 = kein Kredit

iw_sales		Bestellungen
Spaltenname	**Datentyp**	**Inhalt**
owner	varchar	Shop-/Mandatenkennung
line_No	numeric	Zeilennummer der Rechnung
orderNo	varchar	Bestellnummer
customerNo	varchar	Kundennummer
type	numeric	2 = Artikel, 1 = Versand
IWAN	numeric	Eindeutige Artikelnummer wie EAN (5000 = Versand)
quantity	numeric	Anzahl der Artikel
amount	money	Nettopreis des Artikels
vat_amount	money	Preis inkl. MwSt.
line_amount	money	Summe der Zeile inkl. MwSt.
VATpercent	varchar	Mehrwertsteuersatz
bill_customerNo	varchar	Kundennummer des Rechnungsempfängers
orderDate	datetime	Bestelldatum
postingDate	datetime	Verarbeitungsdatum

iw_article		Artikel
Spaltenname	**Datentyp**	**Inhalt**
owner	varchar	Shop-/Mandatenkennung
IWAN	numeric	Eindeutige Artikelnummer wie EAN
article_No	varchar	Interne Artikelnummer
description	varchar	Artikelbeschreibung
unitPrice	money	Stückpreis
deftime	datetime	Zeitstempel Artikel gelistet
modtime	datetime	Zeitstempel Artikel zuletzt bearbeitet
seasonCode	varchar	Saison-Code
productGroup	varchar	Produktgruppen-Code
colorCode	varchar	Farb-Code
colorDescription	varchar	Farbbeschreibung
size	varchar	Größe
articleOnline	varchar	1 = online, 0 = offline

iw_payment		Buchungen
Spaltenname	**Datentyp**	**Inhalt**
owner	varchar	Shop-/Mandatenkennung
orderNo	varchar	Bestellnummer
customerNo	varchar	Kundennummer
outstandingAmount	money	Offener Betrag (Rechnungsbeträge > 0, Zahlungen und Retouren < 0)
postingDate	datetime	Bearbeitungs-, Lieferdatum
dueDate	datetime	Zahlungsziel
closedAccountDate	datetime	Zeitstempel Konto geschlossen
openAccount	varchar	0 = geschlossen, 1 = offen
dunningLevel	varchar	Mahnstufen 1 bis 4 (0 = keine Mahnstufe)

iw_return_header		Retouren
Spaltenname	**Datentyp**	**Inhalt**
owner	varchar	Shop-/Mandatenkennung
returnNo	varchar	Retourennummer
orderNo	varchar	Bestellnummer
paymentCode	varchar	Zahlungsart
returnType	varchar	Retourentyp
shippingAgent	varchar	Versender
customerNo	varchar	Kundennummer
bill_customerNo	varchar	Kundennummer des Rechnungsempfängers
shipmentDate	datetime	Versanddatum
postingDate	datetime	Bearbeitungsdatum

iw_return_line		Retouren-Positionen
Spaltenname	**Datentyp**	**Inhalt**
owner	varchar	Shop-/Mandatenkennung
returnNo	varchar	Retourennummer
customerNo	varchar	Kundennummer
bill_customerNo	varchar	Kundennummer des Rechnungsempfängers
quantity	numeric	Artikelanzahl
unitPrice	money	Stückpreis
IWAN	varchar	Eindeutige Artikelnummer wie EAN
type	numeric	2 = Artikel, 1 = Versand
returnReason	varchar	Retourengrund
productGroup	varchar	Produktgruppe
vat_line_amount	money	Summe der Zeile inkl. MwSt.
line_amount	money	Summe der Zeile ohne MwSt.
shipmentDate	datetime	Versanddatum
postingDate	datetime	Bearbeitungsdatum

iw_code_reason		Codes
Spaltenname	**Datentyp**	**Inhalt**
owner	varchar	Shop-/Mandatenkennung
Type	varchar	returnReason/returnType/payment
Code	varchar	Code
Reason	varchar	Klartext