University of Vienna
Faculty of Computer Science
Prof. Claudia Plant
Prof. Torsten Möller
Prof. Moritz Grosse-Wentrup
Dr. Thomas Torsney-Weir

Foundations of Data Analysis : Lab Exercise A4

SoSe 2019

General Remarks:

- The deadline for the submission is at 09:45 a.m. on 13.06.2019. Please upload your solutions on Moodle. No deadline extension is possible.
- For answering the questions, please use Python and utilize the provided Jupyter Notebook templates.
- Upload your solutions as a zip archive with the following naming scheme **matrikelnumber_A4.zip**. The archive should contain your code as well as a report with your assumptions, results and a description how to compile and run your code.
- Mark external sources you are using in the code and report
- If you have problems do not hesitate to contact the tutors or post a question on Moodle.

Task -1 Dimensionality Reduction, SVD and PCA (30 P)

1) SVD Image Compression Exercise (10P)

The singular value decomposition (SVD) of a matrix helps to compress its representation, so that only essential information remains. Hence, we want to reduce the dimensionality of the image without distorting it too much. Reducing the dimensionality of data can be tricky because you do not want to eliminate too much information, so that you would not be able to recognize the image any more. We want you to explore how dimensionality reduction works using an example of image compression. As an input use the provided image ('faculty.png'). You should explore the effects of SVD with the gray value matrix of the image. The template *task_1.ipynb* helps you to retrieve the matrix (*numpy-array*) that describes the image and to plot the image using *matplotlib*.

(a) Conduct a SVD using the *numpy.linalg.svd* command. Retrieve the singular values, the left singular vectors and the right singular vectors. Reconstruct and plot the original image from the computed factorization.

(b) Describe the properties of the individual components i.e. inspect the properties of the vectors/matrices including the original matrix.

(c) Inspect the vector containing the singular values. The original matrix can be compressed using a number of singular values that you choose. What happens to your matrix A when you choose a very high or very low number of singular values? If you intended to reduce the dimensionality of your data, what would be the number of singular values you would choose and why?

b) Comparing PCA to SVD (20P)

Please go to the UCI machine learning repository (a great resource to find datasets). Download the *Breast Tissue* dataset from this link: `http://archive.ics.uci.edu/ml/datasets/Breast+Tissue`

In this dataset the type of various tissues is being predicted based on 9 variables.

(a) Convert the dataset into a text file (e.g. .csv) and load it using *numpy.loadtxt*. Do not use attributes that cannot be considered independent variables. What would you do if you had a mix of numerical and categorical variables?

(b) Find the correlation matrix using *np.corrcoef*, inspect it and try to predict which variables will form principal components. This will also help you in the next step. There are no right answers because it is a complex problem, which is why we use PCA. However, you will be able to predict some trends that will be reflected in the PCA.

(c) Scale your data to unit variance but do not center it. Conduct a PCA through calculating the eigenvalues and vectors of the covariance matrix of the scaled data as well as of the original data using the *numpy.linalg.eigh* command. How many components would you choose for either PCA? Elaborate your answer with a plot and explain how much variance of the original features is explained by the PCs, by computing the variance that is explained by each PC. Why would it be sensible to scale the data in our case before applying PCA and what are problems arising when conducting PCA on the unscaled data?

(d) Now additionally center your data and conduct a SVD on the standardized data. Use it to retrieve the eigenvalues of the covariance matrix. What connection between SVD and PCA can you observe?

Task -2 Clustering (30 P)

In this task you try different clustering algorithms and evaluate them using Normalized Mutual Information (NMI) and the Rand index. Go to the UCI Machine learning repository[1] and download the Wine dataset. For this task, use the *task_2.ipynb* template.

(a) Import and plot the data and if necessary preprocess it using *sklearn.preprocessing*.

(b) Try three different clustering techniques already implemented in *scikit-learn*. Please state which algorithms you have used and what parameters you have chosen and why you have chosen them.

(c) Evaluate each clustering technique using Normalized Mutual Information[2] as well as the (adjusted) Rand Index. For the evaluation, use the *sklearn.metrics* package.

(d) Briefly discuss your results.

[1] `https://archive.ics.uci.edu/ml/datasets/wine`
[2] `https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html`

Task -3 Apriori Algorithm for Recommender Systems (40P)

Task Definition:
In this programming assignment, you are required to implement the Apriori algorithm and apply it to mine frequent itemsets for movie recommendation.

Input: The provided input file (movies.txt) is based on the MovieLens dataset[3]. It contains movie preferences of 9871 users. Each line in the file corresponds to a user and represents a list of movies the user has rated with five stars. An example:

Coco;Avengers: Infinity War - Part I

In the example above, the corresponding user has rated the movies “Coco” and “Avengers: Infinity War - Part I” with five stars.

Output: You need to implement the Apriori algorithm and use it to mine sets of movies that are frequent in the input data. After implementing the Apriori algorithm, please set the relative minimum support to 0.05 and run it on the 9871 lists of preferred movies. In other words, you need to extract all the itemsets that have an absolute support larger than 493.

(a) Output all the length-1 frequent movies with their absolute supports into a text file named “oneItems.txt” and place it in the root of your zip file. Every line corresponds to exactly one frequent movie and should be in the following format:

Support:movie

For example, suppose a movie (e.g. Black Panther) has an absolute support 1583, then the line corresponding to this frequent item set in “oneItems.txt” should be:

1583:Black Panther

(b) Please write all the frequent itemsets along with their absolute supports into a text file named “patterns.txt” and place it in the root of your zip file. Every line corresponds to exactly one frequent itemset and should be in the following format:

support:movie_1;movie_2;movie_3;...

For example, suppose an itemset (Black Panther; The Shape of Water) has an absolute support 538, then the line corresponding to this frequent itemset in “patterns.txt” should be:

538:Black Panther;The Shape of Water

(c) Imagine you should recommend a movie to a user. You know that the user has rated the movies “The Shape of Water” and “Three Billboards Outside Ebbing, Missouri“ with five stars. Based on the result of the apriori algorithm, give a movie recommendation for this user by maximizing the confidence that the user will rate your recommendation with five stars. Explain your choice.

[3] `https://grouplens.org/datasets/movielens/`