Data Analysis Methods 2016
Exam

Aasa Feragen, Christian Igel

This is the exam for Data Analysis Methods 2016. It will be made available to all students on Absalon on Monday April 4th, and it should be submitted **through Digital Exam** by Sunday April 17th.

Guidelines for the exam

- You should submit your report as a PDF named `firstname.lastname.pdf`, where your code should be attached **as an appendix**, not scattered throughout the report.
- Your code will be inspected manually, not by the code checker, for grading your exam. Therefore, your code should be attached at the end of your pdf report as an appendix. You can still use the code checker as a tool to check that your implementations are correct.
- You are allowed to reuse your own (sometimes code-checker-checked code from the mandatory assignments throughout the exam assignment, for instance for k-NN, cross validation, hypothesis testing etc.
- In the code appendix, please also include a please add a `read.me` file where you briefly explain how to run your code to genereate the numbers that appear in your report.
- **Please include the line**
  `np.random.seed(0)`
  at the beginning of all your code to remove the randomization effect on your answers.
- **The DAM exam must be completed and written individually.** In contrast to the assignments throughout the course, you **should not** discuss your solution of the exam with your fellow students
- If you use code from the textbook, from the course material, or developed with anyone throughout the course, **you must reference the source of your code as a comment at the beginning of your code**. Example:

  ```
  # This code is based on code from the course textbook ''Data Science from
  Scratch'' by Joel Grus
  # This code is based on examples from Lecture 10 by Aasa Feragen
  # This code is based on my solution from Assignment 3 which I discussed with
  Martha Stewart
  ```

  If you do not properly cite where your code comes from, and your code is found to be identical to that of another student, then **you risk being expelled from the university** for cheating.
- You are allowed to use numpy functions such as mean, std, sum. You are, however, not allowed to use built-in functions for the specific learning algorithms (hypothesis test, regression, PCA, clustering, classification) unless this is specifically stated in the exercise. If you are in doubt, ask in the Absalon forum.
- Some exercises or sub-tasks are labeled as **bonus**. This means that it is possible to reach 100% without doing these exercises, and that any points that you gain from the bonus exercises will make up for potential lost points elsewhere in the exam.

1 Statistics

Here, we will study how the quality of red wine varies with its alcohol level, using the red wine dataset described in Appendix A.1. For the exercises in Part 1 you should **only use the training dataset** and the corresponding class labels, which divides the dataset into good and bad wines.

Exercise 1 (Data visualization)**.**

a) Make a box plot of the alcohol level in the two groups consisting of good and bad wines. Put both box plots in a single plot and remember to describe it properly (title, labels, etc). Give a short explanation of how to read a box plot. What can you conclude from the plot?
b) Make a combined histogram over the alcohol levels of the wines in each of the two groups *good* and *bad* wines. What do you see?

Deliverables. Figures with box plot and histogram, accompanied by a couple of lines of discussion.

Exercise 2 (Hypothesis testing)**.**

a) Please write a script that performs a *two-sample t-test* to investigate whether the alcohol level really is different in good and bad wines.
b) Which null hypothesis do you propose for the task?
c) Use a significance level of $\alpha = 0.05$, and return your computed degrees of freedom, the p-value, a binary response indicating acceptance or rejection of the null hypothesis. Report your results and discuss your accept/reject decision.

You may use the `t.cdf` function from the `scipy.stats` library to estimate your p-value; however, you should *not* use a built-in function for hypothesis testing.

Deliverables. a) Your code attached as an appendix, b) your null hypothesis and c) the value of the t-statistic and of the degrees of freedom ν, the returned p-value, whether or not you rejected the hypothesis, and a short discussion of the result.

2 Classification

Exercise 3 (Logistic regression)**.** In this exercise, you will use logistic regression and k-NN to predict whether or not a wine is good, using all the features from the red wine dataset.

a) Write a script that trains a logistic regression classifier and makes predictions with the trained model. Here you are allowed to use either a built-in function for logistic regression, or your own implementation.

b) **Bonus exercise.** Implement your own logistic regression classifier.

c) Train the logistic regression on the red wine training set and run it on the test set. Threshold the returned probabilities at 0.5 to obtain a binary classification, and report the accuracy in % on both the training and test sets. You can use either your own logistic regression implementation or a built-in implementation.

d) Why should you not make any conclusions based on your training set accuracies?

Deliverables. a) Please include your code as an appendix, b) please include your code as an appendix, c) please return the accuracies in % on both training and test set, d) a couple of sentences of discussion.

Exercise 4 (Decision trees)**.** Let us consider classification with input space $\mathbb{R}^d$. As discussed back in Assignment 2, normalizing each component to zero mean and variance one (measured on the training set) is a common preprocessing step, which can remove undesired biases due to different scaling (e.g., when using nearest neighbour classification or logistic regression).

How does normalization to zero mean and variance one (using an affine linear mapping) influence the training process and the classification accuracy of a CART or Random Forest?

Deliverables. A couple of lines of discussion.

Exercise 5 (Multi-class classification)**.** In this exercise, we consider a famous multi-class classification problem from the UCI benchmark repository [1]. The data was provided by J. A. Blackard [2].

The goal of this problem is to predict the forest cover type from cartographic variables. There are seven different cover types (e.g., spruce or different pines) and 54 input variables. We consider a small subset of the data, 3750 training and 1250 test patterns. See the appendix A.2 for a detailed description of the data.

a) Apply a random forest classifier to the data. You may use a built-in implementation such as from `scikit-learn`. Use $B = 100$ trees in the ensemble. Do not use pruning, all leave nodes have to be pure (i.e., no limits on tree depth and number of samples per leaf).
   Report the accuracy (one minus the average 0-1 loss) on the training and test set. Briefly describe the software you used to obtain the results.
   If you use `sklearn.ensemble.RandomForestClassifier` set the random seed to 42 (i.e., pass the option `random_state=42`).
b) Implement k-NN classification and run it on the data with $k = 1, 3, \ldots, 9$. Report a table of accuracies in % for all these k on both the training and test sets.
c) Write a script that performs 5-fold cross-validation on the training set to find a good value for $k \in \{1, 3, 5, 7, 9\}$. Which value of k gives the best results? Report the accuracy (one minus the average 0-1 loss) for this k on the (full) training and test set (as computed in exercise b).

Deliverables. a) Short description (few lines) of how you proceeded, especially which software you used; short description of the hyperparameters (ensemble size B, the number m of features/variables considered when looking for the best split; impurity measure) you used for building the random forest; accuracy on the training and test data; source code as attachement. b) Please include your code as an appendix. Please report a table of classification accuracies in % for all the k on both sets. c) Short description (few lines) of how you proceeded; best value for k found by cross-validation, accuracy for this k on training and test data. Please include your code as an appendix.

3 Clustering

In this task you should cluster the Iris dataset using k-means clustering, using the commonly used Euclidean distance as distance measure. See Appendix A.3 for more details about the dataset.

Exercise 6. a) Implement k-means clustering and perform clustering on the dataset $X = \mathbf{x}_1, \ldots, \mathbf{x}_n$, where each $\mathbf{x}_i$ represents the measurements of a single flower in the form of a 4-dimensional vector. Use a $k > 1$ of your choice and k randomly chosen data points (from the data set) as initial cluster centers.

b) Write a script that performs PCA on the dataset and projects every data point onto the two first principal components (PCs). Recall that projection onto a line is given by dot product with the unit vector spanning the line.
Hint: Recall that you have to center the data before you project in onto the PCs.

c) Visualize the projected data points from b) in a 2-D scatter plot. How many clusters do you see?

d) Repeat the plot, giving different colors to the clusters found in a). Include the corresponding final cluster centers, visualized with a different marker but the same color as the cluster. What do you see?

Deliverables. a) Please include your code in the appendix, b) please include your code in the appendix, c) a plot and a one-liner, d) a plot and a short description.

Exercise 7. For the first four steps of your k-means algorithm, plot your dataset with color-coded cluster memberships and cluster centers just like in Exercise 6 c). What do you see?

Deliverables. The four plots and a short description.

Exercise 8 (**Bonus exercise.** Selecting k)**.** The choice of k is often based on prior knowledge about data. Without this knowledge k-means clustering is performed for different k in practice. Now, however, one needs to evaluate the performance for each k.

a) Implement a function `eval_clustering.py` based on the supplied template found in Absalon, which does the following:

1. Runs your k-means clustering for $k \in \{1, \ldots, 10\}$ on a given dataset with N data points $\mathbf{x}_i$ of dimension d, and saves the value of the k-means objective function

$$E(k) = \sum_{j=1}^{k} \sum_{\mathbf{x}_i \in C_j} ||\mathbf{x}_i - \mu_j||^2 \ ,$$

after termination for every k.

2. Creates a random benchmark dataset of same size by sampling uniformly N values for each of the d dimensions from the interval of the minimum to the maximum of the corresponding dimension

3. Runs your k-means clustering for $k \in \{1, \ldots, 10\}$ on the benchmark dataset and save the final objective error $E^{rand}(k)$ for every k.

4. Computes the gap statistic (as a function of k)

$$G(k) = \log E^{rand}(k) - \log E(k)$$

5. Repeats step (b), (c) and (d) 10 times and compute the average gap statistic for each k.
6. Returns the average gap statistic for $k = 1, \dots, 10$.

Please include your code in the appendix.

b) Plot the average gap statistic for the Iris dataset as a function of k. Argue that the maximum of the gap statistic is a reasonable choice for the number of clusters. What value of k has the best performance?

A The data material

A.1 Red wine

The following description is based on `https://archive.ics.uci.edu/ml/datasets/Wine+Quality`.

This datasets is related to red variants of the Portuguese "Vinho Verde" wine. The input dataset includes the following attribute information (based on physicochemical tests): 1 - fixed acidity 2 - volatile acidity 3 - citric acid 4 - residual sugar 5 - chlorides 6 - free sulfur dioxide 7 - total sulfur dioxide 8 - density 9 - pH 10 - sulphates 11 - alcohol

The classification labels are extracted from a quality score between 0 and 10 (thresholding at 5) based on sensory data. For more details, consult [3].

A.2 Forest cover types

The following detailed description is taken from `https://archive.ics.uci.edu/ml/datasets/Covertype`:

> The actual forest cover type for a given observation (30 x 30 meter cell) was determined from US Forest Service (USFS) Region 2 Resource Information System (RIS) data. Independent variables were derived from data originally obtained from US Geological Survey (USGS) and USFS data. Data is in raw form (not scaled) and contains binary (0 or 1) columns of data for qualitative independent variables (wilderness areas and soil types).
>
> This study area includes four wilderness areas located in the Roosevelt National Forest of northern Colorado. These areas represent forests with minimal human-caused disturbances, so that existing forest cover types are more a result of ecological processes rather than forest management practices.
>
> Some background information for these four wilderness areas: Neota (area 2) probably has the highest mean elevational value of the 4 wilderness areas. Rawah (area 1) and Comanche Peak (area 3) would have a lower mean elevational value, while Cache la Poudre (area 4) would have the lowest mean elevational value.
>
> As for primary major tree species in these areas, Neota would have spruce/fir (type 1), while Rawah and Comanche Peak would probably have lodgepole pine

Figure 1: Iris setosa, Iris versicolor and Iris virginica. Photos by Radomil Binek, Danielle Langlois and Frank Mayfield distributed under a CC license.

> (type 2) as their primary species, followed by spruce/fir and aspen (type 5). Cache la Poudre would tend to have Ponderosa pine (type 3), Douglas-fir (type 6), and cottonwood/willow (type 4).
>
> The Rawah and Comanche Peak areas would tend to be more typical of the overall dataset than either the Neota or Cache la Poudre, due to their assortment of tree species and range of predictive variable values (elevation, etc.) Cache la Poudre would probably be more unique than the others, due to its relatively low elevation range and species composition.

The input variables are elevation in meters; aspect in degrees azimuth; slope in degrees; horizontal and vertical distances to the nearest surface water features; horizontal distance to nearest roadway; Hillshade index at 9am, noon, and 3pm (summer solstice); horizontal distance to nearest wildfire ignition points; a binary indicator for the absence or presence of wilderness area designation; and the soil type designation (encoded in 40 binary columns).

The training and test data are in the files `covtype_train.csv` and `covtype_test.csv`, respectively. The last column indicates the label.

A.3 The Iris Dataset

In this task you are supposed to cluster the data using k-means clustering. You should use the commonly used euclidean distance as distance measure.

On Absalon, you will find the file `Irisdata.txt`, which consists of measurements made of 50 flowers from each of 3 different *Iris* species (see Figure 1: Iris setosa, Iris Versicolor and Iris virginica). You are given four features: Sepal length in cm, sepal width in cm, petal length in cm and petal width in cm.

References

[1] A. Asuncion and D. J. Newman. *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2007. `http://www.ics.uci.edu/\~mlearn/MLRepository.html`.

[2] J. A. Blackard. *Comparison of Neural Networks and Discriminant Analysis in Predicting Forest Cover Types*. PhD thesis, Department of Forest Sciences, Colorado State University, Fort Collins, Colorado, USA, 1998.

[3] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009.