Data Mining
Homework Assignment #4

Dmytro Fishman, Anna Leontjeva and Jaak Vilo

March 7, 2014

You are free to use any programming language you are comfortable with. Hints in R are optional. Check appendix for more hints at the end of the document.

Task 1

Choose your favorite visualization tool, play around with it and report few fancy pictures produced by you in this tool, e.g. ggplot2 (in R), Excel (hope you manage to find it yourself),Google Code Playground, GNU plot, matplotlib (for Python).

Task 2

Use the data file: iris.data.txt. Take a look at the description of the dataset.

- For each variable calculate mean, standard deviation, median, minimum, maximum and mode. Describe what type of data you have (e.g. continuous or discrete). Are the distributions of variables symmetric, positively or negatively skewed?
- Plot 4 boxplots: for each continuous variable with respect to the class (x-axis is the subclass of the flower: Iris-setosa, Iris-versicolor, Iris-virginica and y-axis is a continuous variable). What do these figures mean and how to interpret them?
- Check, whether there are some outliers in the data. In spite of the outlier detection being a subjective exercise, we will use two very basic ways to detect them. The first one uses the notion of the interquartile range (IQR). It is defined as the difference between upper quartile(UQ, 75%) and lower quartile(LQ, 25%).
  `Hint`: in R these are 3rd Qu. and 1st Qu. in the output of the command `summary(`*mydata*`)` or use function `quantile(`*myfeature*`)`.

The outliers are considered those that are outside the range:

$$[LQ - k \times IQR, UQ + k \times IQR],$$

where k is some non-negative constant. In our case let $k = 1.5$.
The second method uses the notion of mean and standard deviation:

$$\frac{|x - mean(x)|}{sd.dev(x)} > 3$$

Use two methods to detect outliers. Do they agree?

- Compare outliers on previously plotted boxplots with the outliers detected by you. Do they agree?

Task 3

Use the iris dataset to discretize continuous variables. For each of the variables apply equal-width and equal-depth binning. Firstly, partition the data into two intervals and look at the contingency tables with respect to the subclass of the flower (e.g. `table`(*iris$class*,*iris$discretized_sepal_length*)). Play with a different number of intervals (at least with 3 and 4). What do you observe?

Task 4

Often, datasets you work with have missing values. It is important to understand whether the values are missing completely at random or missing data are systematic in some way. In this task you are provided with two files called *iris_missing_1.txt* and *iris_missing_2.txt*. Investigate, which case of missing data you are dealing with in both cases. Accordingly to your discoveries, make an imputation of missing values. Describe, how you do it and compare with the initial dataset. How do your descriptive statistics differ? Measure the mean squared error (MSE) you made with your imputations for each variable ($MSE = 1/n \sum_{i=1} n(imputed\_value_i - real\_value_i)^2$).

Task 5

Implement a density estimation function using a triangular kernel. Use that to plot the density. Compare to histogram and a smooth kernel using the same iris data.

Task 6 (2pt)

In the Task 2 we experimented with two types of the binning. There is another one, which is based on the notion of entropy. Use the following link as the reference and implement the method of entropy-based binning. Apply on the same dataset. How the results differ from the results in the task 2?

Additional help

- Note, that you may have problems with copy-paste from the pdf to the editor, I recommend you to type the commands yourself.

- The iris dataset has no header in the source file. Don't forget to load it properly: `iris = read.table("path", header = FALSE, sep =',')`

- However, it is more convenient to refer to a particular variable by name, thus, assign names to your data:
  `names(iris) = c("sepal_length","sepal_width", "petal_length","petal_width","class")`

- Check, what type of data you are dealing with by typing `class(iris)`. It should be data.frame.

- In R you can request the columns or rows. For example, `iris[,1]` will print the first column, while `iris[1,]` gives you the first row. Alternatively, you can ask for a particular column by name: `iris$sepal_length`.

- Calculation of descriptive statistics is very easy – `summary(iris)` will do all the dirty work for you. You can ask it separately as well, using `mean`(*mydata$myvariable*) or `sd`(*mydata$myvariable*). Surprisingly, the mode is an exception. Function mode will print you something like the storage mode of an object. It is not what you need.

- If you are also eager to learn visualization in R using library `ggplot2`, take a look at the documentation. Use of packages in R is easy. You need to install package only once (`install.packages(''ggplot2''))` and load every time you want to use it: `library(ggplot2)`. Now a simple example of boxplot would be:

  ```
  ggplot(iris, aes(x=sepal_length,
      fill=class))+geom_histogram()
  ```

- for the binning task I recommend you to dive into the help of the function `cut`. Gentle reminder: ?cut

- If you want to create a new feature, let's say the sum of two existing ones, type:

  ```
  iris$meaningless_sum = iris$sepal_length + iris$sepal_width
  ```

- the most important function for the missing values is `is.na`(*mydata$myfeature*). It will produce a logical variable, which will be TRUE for indices with missing data:

```
is.na(iris_missing$sepal_width)
 [1] FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE
    ...
```

  and the following command will give you the subset of the data with missing values in the variable sepal width:

```
subset(iris_missing, is.na(iris_missing$sepal_width)==TRUE)
```

- Don't forget to go through the slides. For example, some version of kernel function is already there for you.