Assignment 1

Machine Learning, Summer term 2015, Tobias Lang and Ulrike von Luxburg

To be discussed in exercise groups on April 13-15

Exercise 1 (Data Exploration, 10 points (1+2+2+2+3))
We start the exercises for machine learning with a basic data exploration task. We analyze the data-set `vaccination.csv`, a simple *artificial* data-set on vaccination of children. A description of the data is provided in the file `vaccination.readme.txt`.

(a) Read the data into your Matlab workspace, for example like this:

```
data = readtable('vaccination.csv', 'Delimiter', ',');
```

Determine the numbers of boys/girls, age groups and olderSiblings. Visualize these numbers with bar plots.

(b) We are interested in the **marginal probabilities** of individual values in our data. More technically, we are interested in $P(A = a)$, where a is a specific value of a random variable A. The random variables correspond to the fields / column names in the data-set, for example, $A = gender$ and $a = 1$ (where 1 denotes "male"). We use short-hand $P(a)$ for $P(A = a)$.

$P(a)$ can be estimated from the data using relative frequencies as follows:

$$\hat{P}(a) = \frac{\text{rows with } a}{\text{all rows}}$$

$\hat{P}(a)$ denotes the empirical estimator of $P(a)$ according to the data.

Calculate the empirical probabilities

- to have a vaccination against disease X,
- to live on the country side,
- to have at least one older sibling.

(c) **Preprocessing** variables can help to better understand the data. A common preprocessing step is to discretize continuous variables. For example, the variable *height* can be transformed into a binary variable $isTallerThan1Meter$.

Calculate the following empirical probabilities:

- What is the probability to be taller than 1 meter?
- What is the probability to be heavier than 40 kg?

Another preprocessing step is the combination of variables. Calculate a variable $diseaseYZ$ which denotes whether a child has had either disease Y or Z or both of them. What is $\hat{P}(diseaseYZ)$?

(d) **Conditional probabilities** relate two or more variables. $P(a \mid b)$ measures the probability of a given that we know b. For example, $P(diseaseX = 1 \mid vacX = 0)$ quantifies the probability that someone has had disease X given that he/she was not vaccinated against X.

$P(a \mid b)$ can be estimated using relative frequencies as follows:

$$\hat{P}(a \mid b) = \frac{\text{rows with } a \text{ and } b}{\text{rows with } b} .$$

Calculate the following probabilities:

- $\hat{P}(diseaseX \mid vacX = 0/1)$[1]
- $\hat{P}(vacX \mid diseaseX = 0/1)$
- $\hat{P}(diseaseY \mid age = 1/2/3/4)$
- $\hat{P}(vacX \mid age = 1/2/3/4)$
- $\hat{P}(knowsToRideABike \mid vacX = 0/1)$

Visualize $\hat{P}(diseaseY \mid age = 1/2/3/4)$ and $\hat{P}(vacX \mid age = 1/2/3/4)$ as line plots with age on the x-axis.

What can you conclude from your results?

(e) Finally, we take a closer look at the effects of vaccination. Calculate $\hat{P}(diseaseYZ \mid vacX = 0/1)$ and compare it to $\hat{P}(diseaseX \mid vacX = 0/1)$. What do you conclude from these results?

Now, condition additionally on age and calculate $\hat{P}(diseaseYZ \mid vacX = 0/1, age = 1/2/3/4)$.

How sure are you that your estimates for $P(diseaseYZ \mid vacX = 0/1, age = 1/2/3/4)$ are accurate? What does this depend on?

Plot $\hat{P}(diseaseYZ = 1 \mid vacX = 0, age = 1/2/3/4)$ and $\hat{P}(diseaseYZ = 1 \mid vacX = 1, age = 1/2/3/4)$ as two lines in one figure with age on the x-axis and the probability on the y-axis. What do you conclude from your plot?

Remark 1: The effects in (e) due to the confounding variable *age* are similar to what is known as Simpson paradox. See here: `http://en.wikipedia.org/wiki/Simpson%27s_paradox`.

Remark 2: This artificial data-set was inspired by the KiGGS data-set (`http://www.kiggs-studie.de/english/survey/kiggs-baseline-study.html`). Some people have used this data-set for problematic data analyses to make obscure claims about putative side-effects of vaccination. For an example in German see here: `http://www.efi-online.de/wp-content/uploads/2014/01/UngeimpfteGesuender.pdf`.

Exercise 2 (kNN classifier, 4+4+2 points) In this exercise, you will implement a kNN classifier in Matlab. A good practice for writing codes is to test it on datasets generated by simple rules. You can check every step of your code using these easy-to-visualize datasets. So first we generate a toy dataset:

```
n1 = 20; train_data_class1 = rand(n1,2);
n2 = 20; train_data_class2 = rand(n2,2) + ones(n2,2)*[1 0; 0 0];
train_data = [train_data_class1 ; train_data_class2];
train_label(1:n1) = 1;
train_label(n1+1:n1+n2) = 2;
```

(a) Prepare the dataset and the classifier:

- Describe the data for class 1 and 2 in words.
- Plot your dataset to see it visually

```
figure (1); clf; hold all; axis equal;
plot(train_data(1:n1,1), train_data(1:n1,2), 'r*');
plot(train_data(n1+1:n1+n2,1), train_data(n1+1:n1+n2,2), 'bo');
```

- Generate 100 test points for each class (`test_data`) and the labels of the test points (`test_label`).
- Write a Matlab function `knnClassify` that gets the training data `train_data,train_label`, the test data `test_data` and `k` as its input, and returns the predicted labels for the test data (helpful Matlab command: `sort`). Save it as `knnClassify.m`:

[1] $\hat{P}(a \mid b = 0/1)$ is shorthand for $\hat{P}(a = 1 \mid b = 0)$ *and* $\hat{P}(a = 1 \mid b = 1)$.

```
function pred = knnClassify(train_data, train_label, test_data, k);
```

(b) Test the classifier:

- Write a Matlab function `loss01` that gets as input a prediction `y_pred` and correct labels `y`. The function should return the average error (empirical risk with respect to the 0-1 loss) of this prediction:

  ```
  function err = loss01(y_pred,y);
  ```

- Test the classifier with different values `k=1,3,5,7,10,15,20` and store their training and test errors.

  ```
  k_values=[1, 3, 5, 7, 10, 15, 20];
  for i = 1:length(k_values)
  predTrain = ...
  errTrain(i) = ...
  ...
  end
  ```

- Plot the training and the test errors. Do results change between different runs? Why?

  ```
  figure(2); hold all;
  plot(k_values,errTrain,'r*:');
  plot(k_values,errTest, 'b.-');
  ```

- Plot your prediction (`predTest`) for the best k (the one with the smallest test error) in order to see which points are missclassified:

  ```
  figure(3); clf; hold all; axis equal;
  pred_class1 = find(predTest==1);
  pred_class2 = find(predTest==2);
  plot(test_data(pred_class1,1),test_data(pred_class1,2),'r*');
  plot(test_data(pred_class2,1),test_data(pred_class2,2),'bo');
  plot([1 1],[0 1],'k');
  ```

(c) Evaluate the performance of your classifier in different datasets:

- **More training examples:** Now increase the size of your training data to 100 examples in class 1 and 100 examples in class 2. How does the performance of kNN classifier change?
- **Unbalanced classes:** Test your classifier for unbalanced class sizes: 200 examples in class 1 and 40 examples in class 2. How does the performance of kNN classifier change?