TECHNISCHE UNIVERSITÄT BERLIN
Database Systems and Information Management Group
Prof. V. Markl, C. Boden, Dr. S. Schelter, and J. Soto

AIM-3 Scalable Data Science
Summer Semester 2016

Assignment 2

Visualization, Math Libraries, and Machine Learning Principles
Due Date: July 1, 2016 at 12:00 noon

1. Visualization in Tableau (Total: 10 pts.)

In this exercise, you will gain familiarity with Tableau. To get you started, have a look at their tutorials: http://www.tableau.com/learn/training. You should allocate a few hours to come up to speed with this tool. Next, you will want to select a dataset (e.g., identify a dataset from among those listed in Data Sources.pdf located in the Assignment 2 folder in GitLab or a pick a publicly available dataset based on a topic of interest to you). The dataset should be reasonably large (e.g., in the multi MB range). Once you have identified your dataset, analyze/explore the dataset, and prepare a few graphics. Be sure to incorporate an appropriate title, such as the question you aim to answer, corresponding labels to help better understand your data, and your conclusions/key findings depicted below (e.g., as a text box) your graphic. Also, be sure to document your data source, dataset size.

2. Dimensionality Reduction (Total: 5 pts.)

Complete exercise 11.1.4 on page 412 in the MMDS book by hand. Show all of your computations.

3. Math Libraries (Total: 5 pts.)

Numerical linear algebra libraries often offer specially designed implementations that are well-suited for either dense or sparse matrices. What does it mean for a matrix to be dense or sparse? Shed some light by providing a concrete definition. Also, how would you determine, which one to employ (i.e., use a dense matrix oriented solution vs. a sparse matrix oriented solution), in practice? Does it matter? Is there a difference? Be sure to cite your references when answering these questions.

4. Naive Bayes Classification in Apache Flink (Total: 10 pts.)

In this exercise, you will conduct a practical programming exercise using Apache Flink and employ the Naive Bayes algorithm to solve a practical classification problem.

A. Naive Bayes

Implement a Naive Bayes classifier using Apache Flink. Evaluate your implementation on a data set of postings from 20 newsgroups (check out `http://qwone.com/~jason/20Newsgroups/` for a description of the data). Clone the classification source code from the GitLab assignment repository and have a look at *de.tuberlin.dima.aim3.assignment2*. Be sure to adjust *de.tuberlin.dima.aim3.assignment2.Config* to point to the location of your local project. You will have to complete the three provided classes *Training*, *Classification* and *Evaluator*.

1. In *Training.java* you have to complete the two flink data flows that should compute the conditional counts of words per label (<label, word, count>) as well as the summed counts of all word counts per label (<label, counts>).
2. In *Classification.java* you have to implement the classification function.
3. In *Evaluator.java* you have to complete a function that computes the accuracy of your classifier on a provided test data set. Your classifier should produce an accuracy of well over 80%. If not you maybe forgot to introduce a smoothing parameter in your classifier.

We have also provided a test set without labels (*secrettest.dat*). Run your trained classifier and write the output to a file. It should contain the number, label, and probability assigned to each data point in a tab separated manner (e.g., [1]talk.politics.guns<tab>0.123). Please also upload this file together with your patch in ISIS.

B. Background Notes

1. **Naive Bayes Classifier: Theory**
   As discussed in class, a naive Bayes classifier models a joint distribution over a label Y and a set of observed random variables or features, $(F_1, F_2 \dots F_n)$ using the assumption that the full joint distribution can be factored as follows (features are assumed to be conditionally independent given the label):

$$P(F_1, F_2 \dots F_n) = P(Y) \prod_i P(F_i|Y)$$

   To classify a datum, we can find the most probable label given the feature values for each pixel, using Bayes theorem:

$$P(y|f_1, \dots, f_m) = \frac{P(f_1, \dots, f_m|y)P(y)}{P(f_1, \dots, f_m)} = \frac{P(y)\prod_{i=1}^{m} P(f_i|y)}{P(f_1, \dots, f_m)}$$

$$argmax_y P(y|f_1 \dots f_m) = argmax_y \frac{P(y)\prod_{i=1}^{m} P(f_i|y)}{P(f_1, \dots, f_m)} = argmax_y P(y) \prod_{i=1}^{m} P(f_i|y)$$

   Given that multiplying many probabilities can result in underflow, instead we will compute log probabilities which have the same argmax:

$$argmax_y \log P(y|f_1 \dots f_m) = argmax_y \left\{ \log P(y) + \sum_{i=1}^{m} \log P(f_i|y) \right\}$$

   To compute logarithms, use `Math.log()` function. Put simply, you will have to compute the most likely label given the data (text in the posting to be classified). This means you have to compute the probability for each label and then return the label which recieves the highest probability from your model for the document in question.

2. **Parameter Estimation**
   Our naive Bayes model has several parameters to estimate. One parameter is the prior distribution over labels (the names of the newsgroups), $P(Y)$. for simplicity we assume a uniform prior accross all classes. The term $\log P(y)$ can thus be ignored in the argmax computation. The other parameters to estimate are the conditional probabilities of our features given each label y: $P(F_i|Y = y)$, where is the conditional probability of a word w occurring in a posting of class(topic) y. We do this for each possible feature value (each word in the vocabulary).

$$\hat{P}(w|Y = y) = \frac{count(w, y)}{\sum_{w'} count(w', y)}$$

   where $count(w, y)$ is the number of times word w occured in the training examples of label y.

3. **Smoothing**
   Your current parameter estimates are unsmoothed, that is, you are using the empirical estimates for the parameters. These estimates are rarely adequate in real systems. Minimally, we need to make sure that no parameter ever receives an estimate of zero, but good smoothing can boost accuracy quite a bit by reducing overfitting. In this assignment, we

$$P(w|Y = y) = \frac{count(w, y) + k}{\sum_{w'} (count(w', y) + k)}$$

If $k = 0$, the probabilities are unsmoothed. As k grows larger, the probabilities are smoothed more and more.

5. Clustering in GraphLab (Total: 5 pts.)

Since we devoted several hours to GraphLab in class, we will will perform clustering using the K-means algorithm for a color reduction problem. Following the instructions in the README.md file, use the Python script image_to_pixel.py to transform the data/robocup.jpg image to a CSV file (consisting of the corresponding image pixel values) both of which are located in Assignment2/Clustering folder in our GitLab repository. Next, you will want to write a GraphLab program that invokes the K-means algorithm to perform the following tasks (`https://dato.com/products/create/docs/generated/graphlab.kmeans.KmeansModel.html`):

1. Reduce the number of colors in the image to 8, 4, and 2 then store the results into three corresponding CSV files.
2. Using the Python pixel_to_image.py script, transform your CSV files back to an image. Note: This will only work when your CSV file is correctly formatted.
3. Interpret your results. Do you see a difference in the image?

Be sure to include your GraphLab program and the three corresponding (reduced color) images.

General Instructions

The source code stub for exercise 4 is available in our GitLab repository. You will need to upload your results in ISIS as a zip archive (adhering to the structure and naming conventions listed below) by no later July 1, 2016 at 12:00 noon. On Friday, July 1, 2016 you will need to bring a stapled, printed copy of your homework assignment, including the key source code file(s), runs (i.e., the output), and hand-written solutions. Also, be aware that it is plausible you may be asked to meet with one of the instructors to run your codes. Lastly, be certain to work individually, you are free to drop some hints, but try your best to solve these problems on your own.

```
aim3-SS16-<name>.zip
├── author.txt (contains your name and matriculation number)
├── visualizations in Tableau
│   └── include your graphics with titles, legends, and conclusions as a PDF file
├── numerical library
│   └── documented explanation including references as a PDF file
├── dimensionality reduction
│   └── documented explanation including mathematical calculations as a PDF file
├── classification in Apache Flink
│   ├── key source code files (i.e., the files you updated)
│   ├── patch file
│   └── output of job for secrettest.dat
├── clustering in GraphLab
│   ├── the GraphLab program (e.g., as Jupyter Notebook) and the summary output
│   └── the three reduced-color images
└── documentation.pdf (this PDF file will contain your answers to exercises 1-3)
```