Project phase 2
Data Analysis with Spark

Due date:

24 April 2018 23:59

The project of the course TDT4305 consists of two phases. This document describes the second phase which focuses on creating a data analysis application on a large dataset. As in the previous phase, you will work with the Apache Spark framework[1] and you can choose freely which of the Spark-compatible languages you prefer: Python, Scala or Java. You can work in groups of max. two people.

1 Objective

The aim of the second phase is to create a Spark application that will estimate the location for a tweet text in terms of place. The Twitter dataset in Phase 1 will be used for training (to calculate which words are used in which places). Then, given a new tweet text, its place should be estimated according to this training data.

1.1 Data

A link to the dataset is available on Blackboard, under the "Prosjekt\Project" tab. It is the same dataset with the Phase 1.

1.2 Notes before you start

- Try to find appropriate Spark RDD functions with respect to the tasks you are trying to accomplish.
- Consider distributed execution and use Spark RDD APIs whereever appropriate.
- Use only data/columns needed for each task to make your solution efficient.
- Words in the tweet texts are separated by space (' '). All text analyses should be case-insensitive, so it is suggested to convert the tweet texts to lowercase.

[1] http://spark.apache.org/

1.3 Motivation

Tweets can have geographical information in terms of various data types, such as country, place or latitude-longitude. This information in tweets can be useful for making location-based event detection, targeted advertisement, etc. However, among millions of tweets posted per day, only 1-3% of them includes such geographical information. Therefore, location estimation methods are developed to estimate tweet locations according to their texts.

1.4 Location Estimation of Tweets

In this second phase of the project, you are expected to estimate the place for a given tweet text, according to the texts and places of previous tweets in a training file. The basic idea of text-based location estimation can be exemplified as follows: If we observe that most of the tweets mentioning 'NTNU' are posted from Trondheim, the location for a new tweet mentioning 'NTNU' is also very likely to be Trondheim. You are expected to implement a Naive Bayes classifier in Spark to make this estimation.

Your program will take three parameters:

1. tweets file for training (geotweets.tsv or another file with the same format),
2. input file that contains a single line with tweet text to estimate location,
3. output file to write the estimation result.

For the given tweet text in the input file, your Spark program is expected to calculate probabilities for each possible place (the places that are listed in the training file) and select the place with the the highest probability. How to calculate probabilities for each place is described below.

1.4.1 Naive Bayes Classifier

The calculation of probabilities is a common measure used in the basic classification methods. A Naive Bayes classifier practically calculates the probability of a new object to be in one of the classes defined before. In our case, the classes are the places in the training file. Tweets posted from these places in training data will be used to calculate place probabilities for the tweet in the input file.

The calculation of probabilities is performed as follows: Assume you are given a tweet with n words $(w_1, w_2, ...w_n)$ in the input file. This represents the tweet text for which you will estimate the location. For each distinct place c in the training set, you are expected to calculate the probability $p(c)$ using equation 1.

$$p(c) \simeq \frac{|T_c|}{|T|} \times \frac{|T_{c,w_1}|}{|T_c|} \times \frac{|T_{c,w_2}|}{|T_c|} ... \times \frac{|T_{c,w_n}|}{|T_c|} \tag{1}$$

In this equation,
$|T|$ is the total number of tweets in the training set,
$|T_c|$ is the number of tweets in the training set that have been posted from place c,
$|T_{c,w_i}|$ is the number of tweets in the training set with word w_i and posted from place c.
As mentioned before, your goal is to calculate the probabilities $p(c)$ for every distinct place in the training dataset for the tweet text $(w_1, w_2, ...w_n)$ in the input file. The place

with the highest non-zero probability $p(c)$ should be the result of the estimation. You are expected to write this place name and its probability into an output file. The format of the output file should be TSV with the most probable place together with its probability (place<TAB>probability).

Note1: If more than one place have the highest probability, you should write all these place names into the output file. (place1<TAB>place2<TAB>... <TAB>placeX<TAB>probability).

Note2: If no place is found with a probability greater than zero, do not return anything in the output file.

1.4.2 Example

Example place-tweet texts with 5 tweets in a training file:
1. `NewYork` ... *empire state building*
2. `NewYork` ... *big empire state*
3. `NewYork` ... *new york state*
4. `London` ... *big ben*
5. `London` ... *big state building*

Input tweet text: *state*	Expected result in the output
$p(\texttt{NewYork}) = \frac{3}{5} \times \frac{3}{3} = 0.6$ $p(\texttt{London}) = \frac{2}{5} \times \frac{1}{2} = 0.2$	NewYork<TAB>0.6

Input tweet text: *state building*	Expected result in the output file
$p(\texttt{NewYork}) = \frac{3}{5} \times \frac{3}{3} \times \frac{1}{3} = 0.2$ $p(\texttt{London}) = \frac{2}{5} \times \frac{1}{2} \times \frac{1}{2} = 0.1$	NewYork<TAB>0.2

Input tweet text: *state building building building*	Expected result in the output file
$p(\texttt{NewYork}) = \frac{3}{5} \times \frac{3}{3} \times \frac{1}{3} \times \frac{1}{3} \times \frac{1}{3} = 0.022$ $p(\texttt{London}) = \frac{2}{5} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = 0.025$	London<TAB>0.025

Input tweet text: *empire building*	Expected result in the output file
$p(\texttt{NewYork}) = \frac{3}{5} \times \frac{2}{3} \times \frac{1}{3} = 0.133$ $p(\texttt{London}) = \frac{2}{5} \times \frac{0}{2} \times \frac{1}{2} = 0$	NewYork<TAB>0.133

Input tweet text: *hello*	Expected result in the output file
$p(\texttt{NewYork}) = \frac{3}{5} \times \frac{0}{3} = 0$ $p(\texttt{London}) = \frac{2}{5} \times \frac{0}{2} = 0$	

1.4.3 Program

Create a standalone program called **classify** that accepts three parameters with the following flags:

- -training <full path of the training file>
- -input <full path of the input file>
- -output <full path of the output file>

Example:

classify.py -training ./data/geotweets.tsv -input ./data/input1.txt -output ./data/output1.tsv

The training file will have the same tab separated column structure as in geotweets.tsv. That means, the 5th column will be the place name, and 11th column will contain the tweet text. You will not need other columns in your calculations. Input file will contain a single line of tweet text (words separated by space). Output filename is the path to the file to write the result.

1.5 Delivery

You should deliver to Blackboard a ZIP file containing:

- a short report (PDF) describing your code
- source codes of your script/program with the described name (.py/.scala/ .java)
- for scala and java coding, include also a *RUNNABLE* jar with compilation and spark submit parameters

In your report, briefly describe how your programm works and which Spark functions you used in your solution and why. You can include examples to help you describe better. You can work in groups of max. two people, providing both names in your report and both should deliver the project on Blackboard.

There will be slots for presentation (and questioning) about the full project on April 12-13 and April 23-24.