# 732A54/TDDE31 BIG DATA ANALYTICS

## LAB EXERCISE 3: MACHINE LEARNING

JOSE M. PEÑA
IDA, LINKÖPING UNIVERSITY, SWEDEN

### INSTRUCTIONS

Hand in a lab report that includes the name and LiU-id for each group member. The report should include your code, results from the code execution, and written answers to the questions in the assignment. Comment each step in your program to provide a clear picture of your reasoning when solving the assignment.

### RESOURCES

Spark and Python. **Do not use Spark SQL or any other programming languages**.

### ASSIGNMENT

Implement in Spark (PySpark) a kernel model to predict the hourly temperatures for a date and place in Sweden. To do so, you should use the files `temperature-readings.csv` and `stations.csv` from previous labs. Specifically, the forecast should consist of the predicted temperatures from 4 am to 24 pm in an interval of 2 hours for a date and place in Sweden. Use a kernel that is the sum of three Gaussian kernels:

- The first to account for the distance from a station to the point of interest.
- The second to account for the distance between the day a temperature measurement was made and the day of interest.
- The third to account for the distance between the hour of the day a temperature measurement was made and the hour of interest.

Choose an appropriate smoothing coefficient or width for each of the three kernels above. You do not need to use cross-validation.

### QUESTIONS

- Show that your choice for the kernels' width is sensible, i.e. it gives more weight to closer points. Discuss why your definition of closeness is reasonable.
- Repeat the exercise using a kernel that is the product of the three Gaussian kernels above. Compare the results with those obtained for the additive kernel. If they differ, explain why.

### HELP

- Note that the file `temperature-readings.csv` may contain temperature measurements that are posterior to the day and hour of your forecast. You must filter such measurements out, i.e. they cannot be used to compute the forecast.
- Cache the data you will reuse by using `rdd.cache()`. Check the course slides.
- Avoid joining two RDDs. Instead, broadcast the smallest, if small enough. Check the course slides.
- My program takes 5-6 minutes (wallclock) on the whole `temperature-readings.csv`. However, you may want to use a sample when implementing and testing different settings. Then, do `rdd.sample(False, 0.1)` to obtain a sample without replacement of size 10 %. Note that the results you hand in should make use of the whole dataset, not a sample.

- Feel free to use the template below to solve the assignment.

```
from __future__ import division
from math import radians, cos, sin, asin, sqrt, exp
from datetime import datetime
from pyspark import SparkContext

sc = SparkContext(appName="lab_kernel")

def haversine(lon1, lat1, lon2, lat2):
    """
    Calculate the great circle distance between two points
    on the earth (specified in decimal degrees)
    """
    # convert decimal degrees to radians
    lon1, lat1, lon2, lat2 = map(radians, [lon1, lat1, lon2, lat2])
    # haversine formula
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2
    c = 2 * asin(sqrt(a))
    km = 6367 * c
    return km

h_distance = # Up to you
h_date = # Up to you
h_time = # Up to you
a = 58.4274 # Up to you
b = 14.826 # Up to you
date = "2013-07-04" # Up to you

stations = sc.textFile("data/stations.csv")
temps = sc.textFile("data/temps.csv")

# Your code here

for time in ["24:00:00", "22:00:00", "20:00:00", "18:00:00", "16:00:00", "14:00:00",
"12:00:00", "10:00:00", "08:00:00", "06:00:00", "04:00:00"]:

# Your code here
```