MIDS Machine Learning at Scale

MidTerm Exam

4:00PM - 6:00PM(CT) October 19, 2016
Midterm

MIDS Machine Learning at Scale

Please insert your contact information here

Insert you name here : Jason Sanchez
Insert you email here : jason.sanchez@ischool.berkeley.edu
Insert your UC Berkeley ID here: 26989981

In [1]:
```
import numpy as np
from __future__ import division

%reload_ext autoreload
%autoreload 2
```

Exam Instructions

1. : Please insert Name and Email address in the first cell of this notebook
2. : Please acknowledge receipt of exam by sending a quick email reply to the instructor
3. : Review the submission form first to scope it out (it will take a 5-10 minutes to input your answers and other information into this form):

    - Exam Submission form (http://goo.gl/forms/ggNYfRXz0t)
4. : Please keep all your work and responses in ONE (1) notebook only (and submit via the submission form)
5. : Please make sure that the NBViewer link for your Submission notebook works
6. : Please submit your solutions and notebook via the following form:

    - Exam Submission form (http://goo.gl/forms/ggNYfRXz0t)
7. : For the midterm you will need access to MrJob and Jupyter on your local machines or on AltaScale/AWS to complete some of the questions (like fill in the code to do X).
8. : As for question types:

    - Knowledge test Programmatic/doodle (take photos; embed the photos in your notebook)
    - All programmatic questions can be run locally on your laptop (using MrJob only) or on the cluster
9. : This is an open book exam meaning you can consult webpages and textbooks, class notes, slides etc. but you can not discuss with each other or any other person/group. If any collusion, then this will result in a zero grade and will be grounds for dismissal from the entire program. Please complete this exam by yourself within the time limit.

Exam questions begins here

===Map-Reduce===

MT1. Which of the following statememts about map-reduce are true?

(I) If you only have 1 computer with 1 computing core, then map-reduce is unlikely to help
(II) If we run map-reduce using N single-core computers, then it is likely to get at least an N-Fold speedup compared to using 1 computer
(III) Because of network latency and other overhead associated with map-reduce, if we run map-reduce using N computers, then we will get less than N-Fold speedup compared to using 1 computer
(IV) When using map-reduce for learning a naive Bayes classifier for SPAM classification, we usually use a single machine that accumulates the partial class and word stats from each of the map machines, in order to compute the final model.

Please select one from the following that is most correct:

- (a) I, II, III, IV
- (b) I, III, IV
- (c) I, III
- (d) I,II, III

C

```
In [ ]:
```

===Order inversion===

MT2. normalized product co-occurrence

Suppose you wish to write a MapReduce job that creates normalized product co-occurrence (i.e., pairs of products that have been purchased together) data form a large transaction file of shopping baskets. In addition, we want the relative frequency of coocurring products. Given this scenario, to ensure that all (potentially many) reducers receive appropriate normalization factors (denominators)for a product in the most effcient order in their input streams (so as to minimize memory overhead on the reducer side), the mapper should emit/yield records according to which pattern for the product occurence totals:

(a) emit (*,product) count
(b) There is no need to use order inversion here
(c) emit (product,*) count
(d) None of the above

A

```
In [ ]:
```

===Map-Reduce===

MT3. What is the input to the Reduce function in MRJob? Select the most correct choice.

(a) An arbitrarily sized list of key/value pairs.
(b) One key and a list of some values associated with that key
(c) One key and a list of all values associated with that key.
(d) None of the above

C

(Although it is not a list, but a generator)

```
In [ ]:
```

===Bayesian document classification===

MT4. When building a Bayesian document classifier, Laplace smoothing serves what purpose?

(a) It allows you to use your training data as your validation data.
(b) It prevents zero-products in the posterior distribution.
(c) It accounts for words that were missed by regular expressions.
(d) None of the above

B

```
In [ ]:
```

MT5. Big Data

Big data is defined as the voluminous amount of structured, unstructured or semi-structured data that has huge potential for mining but is so large that it cannot be processed nor stored using traditional (single computer) computing and storage systems. Big data is characterized by its high velocity, volume and variety that requires cost effective and innovative methods for information processing to draw meaningful business insights. More than the volume of the data – it is the nature of the data that defines whether it is considered as Big Data or not. What do the four V's of Big Data denote? Here is a potential simple explanation for each of the four critical features of big data (some or all of which is correct):

Statements

- (I) Volume –Scale of data
- (II) Velocity – Batch processing of data offline
- (III)Variety – Different forms of data
- (IV) Veracity –Uncertainty of data

Which combination of the above statements is correct. Select a single correct response from the following :

- (a) I, II, III, IV
- (b) I, III, IV
- (c) I, III
- (d) I,II, III

B

```
In [ ]:
```

MT6. Combiners can be integral to the successful utilization of the Hadoop shuffle.

Using combiners result in what?

- (I) minimization of reducer workload
- (II) minimization of disk storage for mapper results
- (III) minimization of network traffic
- (IV) none of the above

Select most correct option (i.e., select one option only) from the following:

- (a) I
- (b) I, II and III
- (c) II and III
- (d) IV

B (uncertain)

```
In [ ]:
```

Pairwise similarity using K-L divergence

In probability theory and information theory, the Kullback–Leibler divergence (also information divergence, information gain, relative entropy, KLIC, or KL divergence) is a non-symmetric measure of the difference between two probability distributions P and Q. Specifically, the Kullback–Leibler divergence of Q from P, denoted DKL(P\||Q), is a measure of the information lost when Q is used to approximate P:

For discrete probability distributions P and Q, the Kullback–Leibler divergence of Q from P is defined to be

```
+ KLDistance(P, Q) = Sum_over_item_i (P(i) log (P(i) / Q(i))
```

In the extreme cases, the KL Divergence is 1 when P and Q are maximally different and is 0 when the two distributions are exactly the same (follow the same distribution).

For more information on K-L Divergence see:

```
+ [K-L Divergence](https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_di
vergence)
```

For the next three question we will use an MRjob class for calculating pairwise similarity using K-L Divergence as the similarity measure:

- Job 1: create inverted index (assume just two objects)
- Job 2: calculate/accumulate the similarity of each pair of objects using K-L Divergence

Using the following cells then fill in the code for the first reducer to calculate the K-L divergence of objects (letter documents) in line1 and line2, i.e., KLD(Line1||line2).

Here we ignore characters which are not alphabetical. And all alphabetical characters are lower-cased in the first mapper.

Using the MRJob Class below calculate the KL divergence of the following two string objects.

In [2]:
```
%%writefile kltext.txt
1.Data Science is an interdisciplinary field about processes and systems
 to extract knowledge or insights from large volumes of data in various
 forms (data in various forms, data in various forms, data in various fo
rms), either structured or unstructured,[1][2] which is a continuation o
f some of the data analysis fields such as statistics, data mining and p
redictive analytics, as well as Knowledge Discovery in Databases.
2.Machine learning is a subfield of computer science[1] that evolved fro
m the study of pattern recognition and computational learning theory in
 artificial intelligence.[1] Machine learning explores the study and con
struction of algorithms that can learn from and make predictions on dat
a.[2] Such algorithms operate by building a model from example inputs in
 order to make data-driven predictions or decisions,[3]:2 rather than fo
llowing strictly static program instructions.
```

```
Writing kltext.txt
```

MRjob class for calculating pairwise similarity using K-L Divergence as the similarity measure

Job 1: create inverted index (assume just two objects)

Job 2: calculate the similarity of each pair of objects

In [3]:
```
import numpy as np
np.log(3)
```

Out[3]:
```
1.0986122886681098
```

In [4]:
```
!cat kltext.txt
```

```
1.Data Science is an interdisciplinary field about processes and system
s to extract knowledge or insights from large volumes of data in variou
s forms (data in various forms, data in various forms, data in various
 forms), either structured or unstructured,[1][2] which is a continuati
on of some of the data analysis fields such as statistics, data mining
 and predictive analytics, as well as Knowledge Discovery in Databases.
2.Machine learning is a subfield of computer science[1] that evolved fr
om the study of pattern recognition and computational learning theory i
n artificial intelligence.[1] Machine learning explores the study and c
onstruction of algorithms that can learn from and make predictions on d
ata.[2] Such algorithms operate by building a model from example inputs
 in order to make data-driven predictions or decisions,[3]:2 rather tha
n following strictly static program instructions.
```

In [23]:

```
%%writefile kldivergence.py
# coding: utf-8

from __future__ import division
from mrjob.job import MRJob
from mrjob.step import MRStep
import re
import numpy as np

class kldivergence(MRJob):
    # process each string character by character
    # the relative frequency of each character emitting Pr(character|st
r)
    # for input record 1.abcbe
    # emit "a"    [1, 0.2]
    # emit "b"    [1, 0.4] etc...
    def mapper1(self, _, line):
        index = int(line.split('.',1)[0])
        letter_list = re.sub(r"[^A-Za-z]+", '', line).lower()
        count = {}
        for l in letter_list:
            if count.has_key(l):
                count[l] += 1
            else:
                count[l] = 1
        for key in count:
            yield key, [index, count[key]*1.0/len(letter_list)]

    # on a component i calculate (e.g., "b")
    # Kullback–Leibler divergence of Q from P is defined as (P(i) log (P
(i) / Q(i))
    def reducer1(self, key, values):
        p = 0
        q = 0
        for v in values:
            if v[0] == 1:  #String 1
                p = v[1]
            else:          # String 2
                q = v[1]

            if p and q:
                yield (None, p*np.log(p/q))

    #Aggegate components
    def reducer2(self, key, values):
        kl_sum = 0
        for value in values:
            kl_sum = kl_sum + value
        yield "KLDivergence", kl_sum

    def steps(self):
        mr_steps = [self.mr(mapper=self.mapper1,
                        reducer=self.reducer1),

                    self.mr(reducer=self.reducer2)]
#         mr_steps = [MRStep(mapper=self.mapper1, reducer=self.reducer
1)]
```

```
        return mr_steps

if __name__ == '__main__':
    kldivergence.run()
```

```
Overwriting kldivergence.py
```

In [24]:
```
%reload_ext autoreload
%autoreload 2
from mrjob.job import MRJob
from kldivergence import kldivergence

#dont forget to save kltext.txt (see earlier cell)
mr_job = kldivergence(args=['kltext.txt'])
with mr_job.make_runner() as runner:
    runner.run()
    # stream_output: get access of the output
    for line in runner.stream_output():
        print mr_job.parse_output_line(line)
```

```
(u'KLDivergence', 0.08088278445318145)
```

Questions:

MT7. Which number below is the closest to the result you get for KLD(Line1||line2)?

(a) 0.7
(b) 0.5
(c) 0.2
(d) 0.1

D

In [ ]:
```
```

MT8. Which of the following letters are missing from these character vectors?

(a) p and t
(b) k and q
(c) j and q
(d) j and f

```
In [28]: words = """
1.Data Science is an interdisciplinary field about processes and systems
 to extract knowledge or insights from large volumes of data in various
 forms (data in various forms, data in various forms, data in various fo
rms), either structured or unstructured,[1][2] which is a continuation o
f some of the data analysis fields such as statistics, data mining and p
redictive analytics, as well as Knowledge Discovery in Databases.
2.Machine learning is a subfield of computer science[1] that evolved fro
m the study of pattern recognition and computational learning theory in
 artificial intelligence.[1] Machine learning explores the study and con
struction of algorithms that can learn from and make predictions on dat
a.[2] Such algorithms operate by building a model from example inputs in
 order to make data-driven predictions or decisions,[3]:2 rather than fo
llowing strictly static program instructions."""

for char in ['p', 'k', 'f', 'q', 'j']:
    if char not in words:
        print char
```

```
q
j
```

C

```
In [ ]:
```

```
In [29]:
```

```
%%writefile kldivergence_smooth.py
from __future__ import division
from mrjob.job import MRJob
import re
import numpy as np
class kldivergence_smooth(MRJob):

    # process each string character by character
    # the relative frequency of each character emitting Pr(character|st
r)
    # for input record 1.abcbe
    # emit "a"    [1, (1+1)/(5+24)]
    # emit "b"    [1, (2+1)/(5+24) etc...
    def mapper1(self, _, line):
        index = int(line.split('.',1)[0])
        letter_list = re.sub(r"[^A-Za-z]+", '', line).lower()
        count = {}

        # (ni+1)/(n+24)

        for l in letter_list:
            if count.has_key(l):
                count[l] += 1
            else:
                count[l] = 1

        for letter in ['q', 'j']:
            if letter not in letter_list:
                count[letter] = 0

        for key in count:
            yield key, [index, (1+count[key]*1.0)/(24+len(letter_list))]


    def reducer1(self, key, values):
        p = 0
        q = 0
        for v in values:
            if v[0] == 1:
                p = v[1]
            else:
                q = v[1]

        yield (None, p*np.log(p/q))

    # Aggregate components
    def reducer2(self, key, values):
        kl_sum = 0
        for value in values:
            kl_sum = kl_sum + value
        yield "KLDivergence", kl_sum

    def steps(self):
        return [self.mr(mapper=self.mapper1,
                        reducer=self.reducer1),
                self.mr(reducer=self.reducer2)
                ]

if __name__ == '__main__':
    kldivergence_smooth.run()
```

```
Writing kldivergence_smooth.py
```

In [31]:
```
%reload_ext autoreload
%autoreload 2

from kldivergence_smooth import kldivergence_smooth
mr_job = kldivergence_smooth(args=['kltext.txt'])
with mr_job.make_runner() as runner:
    runner.run()
    # stream_output: get access of the output
    for line in runner.stream_output():
        print mr_job.parse_output_line(line)
```

```
(u'KLDivergence', 0.06791349183751216)
```

MT9. The KL divergence on multinomials is defined only when they have nonzero entries.

For zero entries, we have to smooth distributions. Suppose we smooth in this way:

(ni+1)/(n+24)

where ni is the count for letter i and n is the total count of all letters.
After smoothing, which number below is the closest to the result you get for KLD(Line1||line2)??

(a) 0.08
(b) 0.71
(c) 0.02
(d) 0.11

A

In [ ]:

MT10. Block size, and mapper tasks

Given ten (10) files in the input directory for a Hadoop Streaming job (MRjob or just Hadoop) with the following filesizes (in megabytes): 1, 2,3,4,5,6,7,8,9,10; and a block size of 5M (NOTE: normally we should set the blocksize to 1 GigB using modern computers). How many map tasks will result from processing the data in the input directory? Select the closest number from the following list.

(a) 1 map task
(b) 14
(c) 12
(d) None of the above

B

In [ ]:

MT11. Aggregation

Given a purchase transaction log file where each purchase transaction contains the customer identifier, item purchased and much more information about the transaction. Which of the following statements are true about a MapReduce job that performs an “aggregation” such as get the number of transaction per customer.

Statements

- (I) A mapper only job will not suffice, as each map tast only gets to see a subset of the data (e.g., one block). As such a mapper only job will only produce intermediate tallys for each customer.
- (II) A reducer only job will suffice and is most efficient computationally
- (III) If the developer provides a Mapper and Reducer it can potentially be more efficient than option II
- (IV) A reducer only job with a custom partitioner will suffice.

Select the most correct option from the following:

- (a) I, II, III, IV
- (b) II, IV
- (c) III, IV
- (d) III

C

In [ ]:

MT12. Naive Bayes

Which of the following statements are true regarding Naive Bayes?

Statements

- (I) Naive Bayes is a machine learning algorithm that can be used for classifcation problems only
- (II) Multinomial Naive Bayes is a flavour of Naive Bayes for discrete input variables and can be combined with Laplace smoothing to avoid zero predictions for class posterior probabilities when attribute value combinations show up during classification but were not present during training.
- (III) Naive Bayes can be used for continous valued input variables. In this case, one can use Gaussian distributions to model the class conditional probability distributions Pr(X|Class).
- (IV) Naive Bayes can model continous target variables directly.

Please select the single most correct combination from the following:

- (a) I, II, III, IV
- (b) I, II, III
- (c) I, III, IV
- (d) I, II

B

```
In [ ]:
```

MT13. Naive Bayes SPAM model

Given the following document dataset for a Two-Class problem: ham and spam. Use MRJob (please include your code) to build a muiltnomial Naive Bayes classifier. Please use Laplace Smoothing with a hyperparameter of 1. Please use words only (a-z) as features. Please lowercase all words.

# Training Data # Record format # Class docID:"doc contents string" ham d1: “good.” ham d2: “very good.” spam d3: “bad.” spam d4: “very bad.” spam d5: “very bad, very BAD.”

```
In [37]: %%writefile spam.txt
0002.2001-05-25.SA_and_HP       0       0       good
0002.2001-05-25.SA_and_HP       0       0       very good
0002.2001-05-25.SA_and_HP       1       0       bad
0002.2001-05-25.SA_and_HP       1       0       very bad
0002.2001-05-25.SA_and_HP       1       0       very bad, very BAD
```

```
Overwriting spam.txt
```

```
In [39]: %%writefile spam_test.txt
0002.2001-05-25.SA_and_HP       1       0       good? bad! very Bad!
```

```
Overwriting spam_test.txt
```

```
In [40]:
```

```
%%writefile NaiveBayes.py

import sys
import re
from mrjob.job import MRJob
from mrjob.step import MRStep
from mrjob.protocol import TextProtocol, TextValueProtocol

# Prevents broken pipe errors from using ... | head
from signal import signal, SIGPIPE, SIG_DFL
signal(SIGPIPE,SIG_DFL)

def sum_hs(counts):
    h_total, s_total = 0, 0
    for h, s in counts:
        h_total += h
        s_total += s
    return (h_total, s_total)


class NaiveBayes(MRJob):
    MRJob.OUTPUT_PROTOCOL = TextValueProtocol

    def mapper(self, _, lines):
        _, spam, subject, email = lines.split("\t")
        words = re.findall(r'[a-z]+', (email.lower()+" "+subject.lower
()))

        if spam == "1":
            h, s = 0, 1
        else:
            h, s = 1, 0
        yield "***Total Emails", (h, s)

        for word in words:
            yield word, (h, s)
            yield "***Total Words", (h, s)

    def combiner(self, key, count):
        yield key, sum_hs(count)

    def reducer_init(self):
        self.total_ham = 0
        self.total_spam = 0

    def reducer(self, key, count):
        ham, spam = sum_hs(count)
        if key.startswith("***"):
            if "Words" in key:
                self.total_ham, self.total_spam = ham, spam
            elif "Emails" in key:
                total = ham + spam
                yield "_", "***Priors\t%.10f\t%.10f" % (ham/total, spam/
total)
        else:
            pg_ham, pg_spam = ham/self.total_ham, spam/self.total_spam
            yield "_", "%s\t%.10f\t%.10f" % (key, pg_ham, pg_spam)
```

```
if __name__ == "__main__":
    NaiveBayes.run()
```

```
Overwriting NaiveBayes.py
```

In [41]:
```
!cat spam.txt | python NaiveBayes.py --jobconf mapred.reduce.tasks=1 -q
| head
```

```
***Priors       0.0000000000    0.0000000000
bad     0.0000000000    0.0000000000
good    0.0000000000    0.0000000000
very    0.0000000000    0.0000000000
```

QUESTION

Having learnt the Naive Bayes text classification model for this problem using the training data and classified the test data (d6) please indicate which of the following is true:

Statements

- (I) P(very|ham) = 0.33
- (II) P(good|ham) = 0.50
- (I) Posterior Probability P(ham| d6) is approximately 24%
- (IV) Class of d6 is ham

Please select the single most correct combination of these statements from the following:

- (a) I, II, III, IV
- (b) I, II, III
- (c) I, III, IV
- (d) I, II

C (wild guess)

In [ ]:
```

```

MT14. Is there a map input format (for Hadoop or MRJob)?

(a) Yes, but only in Hadoop 0.22+.
(b) Yes, in Hadoop there is a default expectation that each record is delimited by an end of line charcacter and that key is the first token delimited by a tab character and that the value-part is everything after the tab character.
(c) No, when MRJob INPUT_PROTOCOL = RawValueProtocol. In this case input is processed in format agnostic way thereby avoiding any type of parsing errors. The value is treated as a str, the key is read in as None.
(d) Both b and c are correct answers.

D

```
In [ ]:
```

MT15. What happens if mapper output does not match reducer input?

(a) Hadoop API will convert the data to the type that is needed by the reducer.
(b) Data input/output inconsistency cannot occur. A preliminary validation check is executed prior to the full execution of the job to ensure there is consistency.
(c) The java compiler will report an error during compilation but the job will complete with exceptions.
(d) A real-time exception will be thrown and map-reduce job will fail.

D

```
In [ ]:
```

MT16. Why would a developer create a map-reduce without the reduce step?

(a) Developers should design Map-Reduce jobs without reducers only if no reduce slots are available on the cluster.
(b) Developers should never design Map-Reduce jobs without reducers. An error will occur upon compile.
(c) There is a CPU intensive step that occurs between the map and reduce steps. Disabling the reduce step speeds up data processing.
(d) It is not possible to create a map-reduce job without at least one reduce step. A developer may decide to limit to one reducer for debugging purposes.

C

```
In [ ]:
```

===Gradient descent===

MT17. Which of the following are true statements with respect to gradient descent for machine learning, where alpha is the learning rate. Select all that apply

- (I) To make gradient descent converge, we must slowly decrease alpha over time and use a combiner in the context of Hadoop.
- (II) Gradient descent is guaranteed to find the global minimum for any unconstrained convex objective function J() regardless of using a combiner or not in the context of Hadoop
- (III) Gradient descent can converge even if alpha is kept fixed. (But alpha cannot be too large, or else it may fail to converge.) Combiners will help speed up the process.
- (IV) For the specific choice of cost function J() used in linear regression, there is no local optima (other than the global optimum).

Select a single correct response from the following:

- (a) I, II, III, IV
- (b) I, III, IV
- (c) II, III
- (d) II,III, IV

D

```
In [ ]:
```

===Weighted K-means===

Write a MapReduce job in MRJob to do the training at scale of a weighted K-means algorithm.

You can write your own code or you can use most of the code from the following notebook:

- http://nbviewer.jupyter.org/urls/dl.dropbox.com/s/oppgyfqxphlh69g/MrJobKmeans_Corrected.ipynb (http://nbviewer.jupyter.org/urls/dl.dropbox.com/s/oppgyfqxphlh69g/MrJobKmeans_Corrected.ipynb)

Weight each example as follows using the inverse vector length (Euclidean norm):

weight(X)= 1/||X||,

where ||X|| = SQRT(X.X)= SQRT(X1^2 + X2^2)

Here X is vector made up of two component X1 and X2.

Using the following data to answer the following TWO questions:

- https://www.dropbox.com/s/ai1uc3q2ucverly/Kmeandata.csv?dl=0 (https://www.dropbox.com/s/ai1uc3q2ucverly/Kmeandata.csv?dl=0)

In [87]:
```
def inverse_vector_length(x1, x2):
    norm = (x1**2 + x2**2)**.5
    return 1.0/norm

inverse_vector_length(1, 5)
```

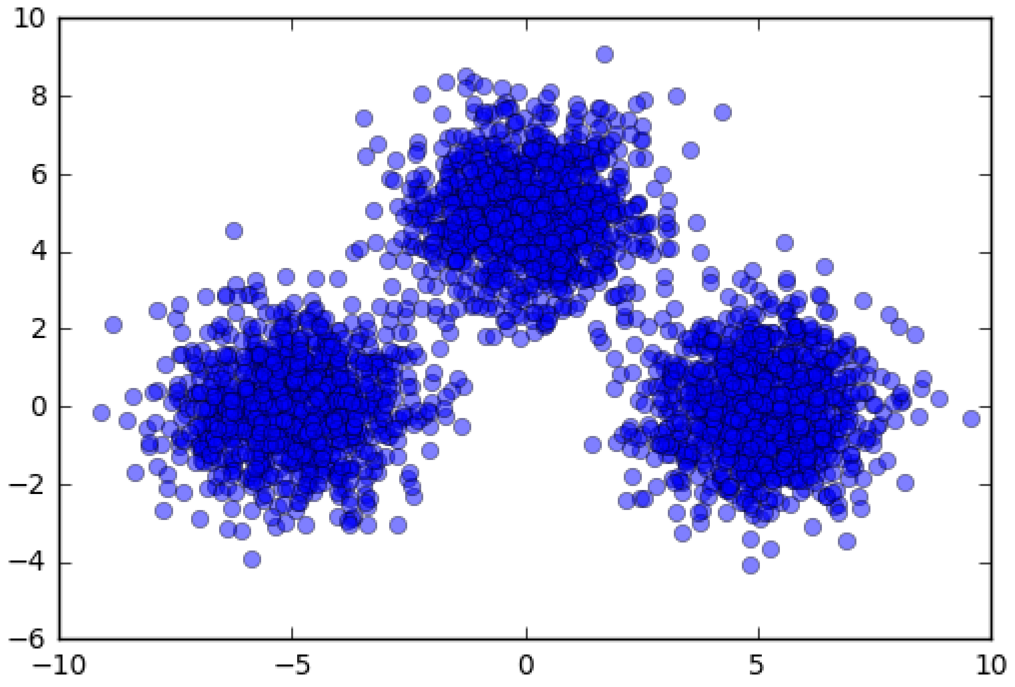
Out[87]:
```
0.19611613513818404
```

In [ ]:
```
0 --> .2
```

In [68]:
```
%matplotlib inline
import numpy as np
import pylab
import pandas as pd
```

In [56]:
```
data = pd.read_csv("Kmeandata.csv", header=None)
```

In [63]:
```
pylab.plot(data[0], data[1], 'o', linewidth=0, alpha=.5);
```



In [65]:

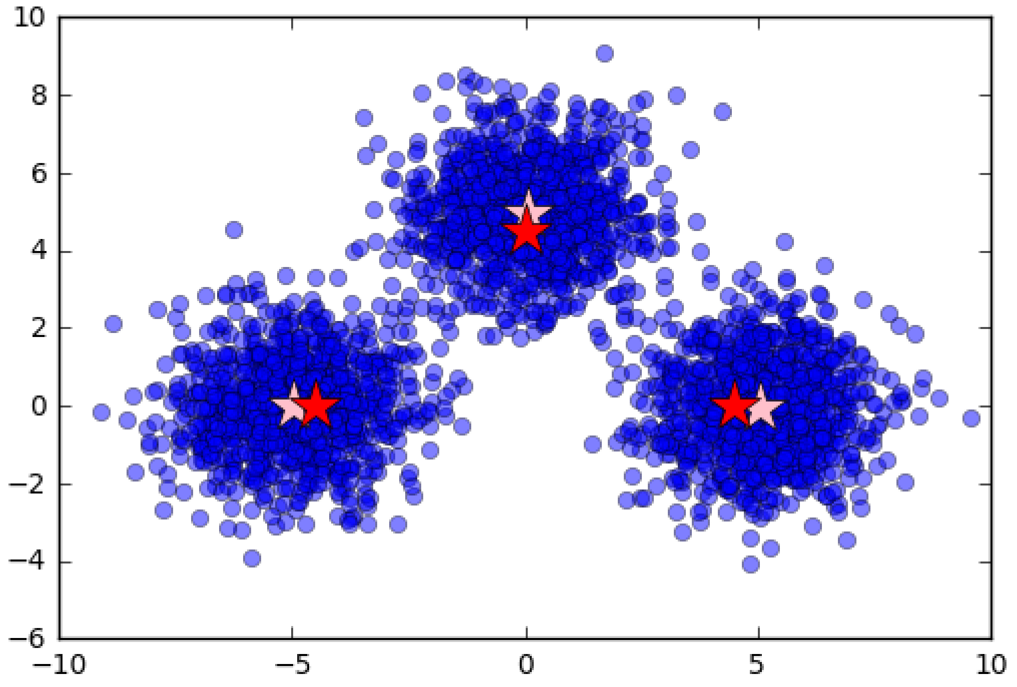
```
%%writefile Kmeans.py
from numpy import argmin, array, random
from mrjob.job import MRJob
from mrjob.step import MRStep
from itertools import chain
import os

#Calculate find the nearest centroid for data point
def MinDist(datapoint, centroid_points):
    datapoint = array(datapoint)
    centroid_points = array(centroid_points)
    diff = datapoint - centroid_points
    diffsq = diff*diff
    # Get the nearest centroid for each instance
    minidx = argmin(list(diffsq.sum(axis = 1)))
    return minidx

#Check whether centroids converge
def stop_criterion(centroid_points_old, centroid_points_new,T):
    oldvalue = list(chain(*centroid_points_old))
    newvalue = list(chain(*centroid_points_new))
    Diff = [abs(x-y) for x, y in zip(oldvalue, newvalue)]
    Flag = True
    for i in Diff:
        if(i>T):
            Flag = False
            break
    return Flag

class MRKmeans(MRJob):
    centroid_points=[]
    k=3
    def steps(self):
        return [
            MRStep(mapper_init = self.mapper_init, mapper=self.mapper,co
mbiner = self.combiner,reducer=self.reducer)
               ]
    #load centroids info from file
    def mapper_init(self):
#         print "Current path:", os.path.dirname(os.path.realpath(__file
__))

        self.centroid_points = [map(float,s.split('\n')[0].split(',')) f
or s in open("Centroids.txt").readlines()]
        #open('Centroids.txt', 'w').close()
#         print "Centroids: ", self.centroid_points

    #load data and output the nearest centroid index and data point
    def mapper(self, _, line):
        D = (map(float,line.split(',')))
        yield int(MinDist(D, self.centroid_points)), (D[0],D[1],1)

    #Combine sum of data points locally
    def combiner(self, idx, inputdata):
        sumx = sumy = num = 0
        for x,y,n in inputdata:
            num = num + n
            sumx = sumx + x
            sumy = sumy + y
        yield idx,(sumx,sumy,num)

    #Aggregate sum for each cluster and then calculate the new centroids
    def reducer(self, idx, inputdata):
        centroids = []
        num = [0]*self.k
        for i in range(self.k):
            centroids.append([0,0])
        for x, y, n in inputdata:
            num[idx] = num[idx] + n
            centroids[idx][0] = centroids[idx][0] + x
            centroids[idx][1] = centroids[idx][1] + y
        centroids[idx][0] = centroids[idx][0]/num[idx]
        centroids[idx][1] = centroids[idx][1]/num[idx]

        yield idx,(centroids[idx][0],centroids[idx][1])

if __name__ == '__main__':
    MRKmeans.run()
```

```
Overwriting Kmeans.py
```

In [66]:
```
%reload_ext autoreload
%autoreload 2
from numpy import random
from Kmeans import MRKmeans, stop_criterion
mr_job = MRKmeans(args=['Kmeandata.csv', '--file=Centroids.txt'])

#Geneate initial centroids
centroid_points = []
k = 3
for i in range(k):
    centroid_points.append([random.uniform(-3,3),random.uniform(-3,3)])
with open('Centroids.txt', 'w+') as f:
        f.writelines(','.join(str(j) for j in i) + '\n' for i in centroi
d_points)

# Update centroids iteratively
i = 0
while(1):
    # save previous centoids to check convergency
    centroid_points_old = centroid_points[:]
    print "iteration"+str(i)+":"
    with mr_job.make_runner() as runner:
        runner.run()
        # stream_output: get access of the output
        for line in runner.stream_output():
            key,value =  mr_job.parse_output_line(line)
            print key, value
            centroid_points[key] = value

        # Update the centroids for the next iteration
        with open('Centroids.txt', 'w') as f:
            f.writelines(','.join(str(j) for j in i) + '\n' for i in cen
troid_points)

    print "\n"
    i = i + 1
    if(stop_criterion(centroid_points_old,centroid_points,0.01)):
        break
print "Centroids\n"
print centroid_points
```

```
iteration0:
0 [5.029576650922463, -0.0029051016453652692]
1 [-5.056032748447057, -0.030107844551825306]
2 [-0.022086029177989085, 4.90174519728454]


iteration1:
0 [5.0402327160888465, -0.026294229978289455]
1 [-4.988208799410262, -0.0016052213546992817]
2 [0.05043492418549814, 4.985354277214307]


iteration2:
0 [5.0402327160888465, -0.026294229978289455]
1 [-4.98580568889943, 0.0009376094363626959]
2 [0.053065423788147964, 4.987793423944292]


Centroids

[[5.0402327160888465, -0.026294229978289455], [-4.98580568889943, 0.000
9376094363626959], [0.053065423788147964, 4.987793423944292]]
```

In [82]:
```
pylab.plot(data[0], data[1], 'o', linewidth=0, alpha=.5);
for point in centroid_points:
    pylab.plot(point[0], point[1], '*',color='pink',markersize=20)

for point in [(-4.5,0.0), (4.5,0.0), (0.0,4.5)]:
    pylab.plot(point[0], point[1], '*',color='red',markersize=20)
pylab.show()
```



MT18. Which result below is the closest to the centroids you got after running your weighted K-means code for K=3 for 10 iterations?

(old11-12)

- (a) (-4.0,0.0), (4.0,0.0), (6.0,6.0)
- (b) (-4.5,0.0), (4.5,0.0), (0.0,4.5)
- (c) (-5.5,0.0), (0.0,0.0), (3.0,3.0)
- (d) (-4.5,0.0), (-4.0,0.0), (0.0,4.5)

B

In [ ]:
```
```

MT19. Using the result of the previous question, which number below is the closest to the average weighted distance between each example and its assigned (closest) centroid?

The average weighted distance is defined as sum over i (weighted_distance_i) / sum over i (weight_i)

- (a) 2.5
- (b) 1.5
- (c) 0.5
- (d) 4.0

C

```
In [ ]:
```

END of Exam