DATASCI W261: Machine Learning at Scale

MrJob class for Kmeans

If you want to change the code, please edit Kmeans.py directly

In [4]:
```
!which python
```

```
/w261/venv/bin/python
```

In [17]:
```
!pwd
```

```
/w261/coursework/Untitled Folder
```

In [14]:

```
%%writefile Kmeans.py
#/w261/venv/bin/python
from numpy import argmin, array, random
from mrjob.job import MRJob
from mrjob.step import MRStep
from itertools import chain

#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**2

    distances = (diffsq.sum(axis = 1))**0.5
    # Get the nearest centroid for each instance
    min_idx = argmin(distances)
    return min_idx

#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):
        self.centroid_points = [map(float,s.split('\n')[0].split(',')) f
or s in open('/w261/coursework/Untitled Folder/Centroids.txt').readlines
()]
        #open('Centroids.txt', 'w').close()
    #load data and output the nearest centroid index and data point
    def mapper(self, _, line):
        D = (map(float,line.split(',')))
        idx = MinDist(D,self.centroid_points)
        yield int(idx), (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 int(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
        distances = 0
        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]
        with open('/w261/coursework/Untitled Folder/Centroids.txt', 'a')
 as f:
            f.writelines(str(centroids[idx][0]) + ',' + str(centroids[id
x][1]) + '\n')
        yield idx,(centroids[idx][0],centroids[idx][1])

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

```
Overwriting Kmeans.py
```

Driver:

Generate random initial centroids

New Centroids = initial centroids

While(1):

- Cacluate new centroids
- stop if new centroids close to old centroids
- Updates centroids

In [15]:
```
!pwd
```

```
/w261/coursework/Untitled Folder
```

In [16]:
```
from numpy import random, array
from Kmeans import MRKmeans, stop_criterion
mr_job = MRKmeans(args=['Kmeandata.csv'])

#Geneate initial centroids
centroid_points = [[0,0],[6,3],[3,6]]
k = 3
with open('/w261/coursework/Untitled Folder/Centroids.txt', 'w+') as f:
        f.writelines(','.join(str(j) for j in i) + '\n' for i in centroi
d_points)

# Update centroids iteratively
for i in range(3):
    # save previous centoids to check convergency
    centroid_points_old = centroid_points[:]
    print "iteration"+str(i+1)+":"
    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
    print "\n"
    i = i + 1
print "Centroids\n"
print centroid_points
```

```
iteration1:
```

```
IOErrorTraceback (most recent call last)
<ipython-input-16-5de47a6acc7c> in <module>()
     15     print "iteration"+str(i+1)+":"
     16     with mr_job.make_runner() as runner:
---> 17         runner.run()
     18         # stream_output: get access of the output
     19         for line in runner.stream_output():

/w261/venv/lib/python2.7/site-packages/mrjob/runner.pyc in run(self)
    471             raise AssertionError("Job already ran!")
    472
--> 473         self._run()
    474         self._ran_job = True
    475

/w261/venv/lib/python2.7/site-packages/mrjob/sim.pyc in _run(self)
    170             self._counters.append({})
    171
--> 172             self._invoke_step(step_num, 'mapper')
    173
    174             if 'reducer' in step:

/w261/venv/lib/python2.7/site-packages/mrjob/sim.pyc in _invoke_step(se
lf, step_num, step_type)
    257
    258             self._run_step(step_num, step_type, input_path, out
put_path,
--> 259                            working_dir, env)
    260
    261             self._prev_outfiles.append(output_path)

/w261/venv/lib/python2.7/site-packages/mrjob/inline.pyc in _run_step(se
lf, step_num, step_type, input_path, output_path, working_dir, env, chi
ld_stdin)
    155                     child_instance.sandbox(stdin=child_stdin,
    156                                            stdout=child_stdout)
--> 157                     child_instance.execute()
    158
    159             if has_combiner:

/w261/venv/lib/python2.7/site-packages/mrjob/job.pyc in execute(self)
    437
    438         elif self.options.run_mapper:
--> 439             self.run_mapper(self.options.step_num)
    440
    441         elif self.options.run_combiner:

/w261/venv/lib/python2.7/site-packages/mrjob/job.pyc in run_mapper(sel
f, step_num)
    497
    498         if mapper_init:
--> 499             for out_key, out_value in mapper_init() or ():
    500                 write_line(out_key, out_value)
    501

/w261/coursework/Untitled Folder/Kmeans.py in mapper_init(self)
```

```
     38                 ]
     39     #load centroids info from file
---> 40     def mapper_init(self):
     41         self.centroid_points = [map(float,s.split('\n')[0].spli
t(',')) for s in open('/w261/coursework/Untitled
Folder/Centroids.txt').readlines()]
     42         #open('Centroids.txt', 'w').close()

IOError: [Errno 2] No such file or directory: 'Centroids.txt'
```

In [ ]: