```
import heapq as hq
import numpy as np
import time
import math


#create the covariance matrix
covar = np.zeros ((100,100))
np.fill_diagonal (covar, 1)

#and the mean vector
mean = np.zeros (100)


#create 3000 data points
all_data = np.random.multivariate_normal (mean, covar, 3000)

#now create the 20 outliers
for i in range (1, 20):
  mean.fill (i)
  outlier_data = np.random.multivariate_normal (mean, covar, i)
  all_data = np.concatenate ((all_data, outlier_data))



#k for kNN detection
k = 10

#the number of outliers to return
m = 5


#start the timer
start_time = time.time()

#the priority queue of outliers
outliers = []

#YOUR CODE HERE!
minpriorO = []
for i, vec1 in enumerate(all_data):
  maxpriorQ = []
  for j, vec2 in enumerate(all_data):
    if i != j:
      dist = np.linalg.norm(vec1 - vec2)
      #print("exec")
      hq.heappush(maxpriorQ, -dist) #The negative sign makes the heap maxpriority
      if len(maxpriorQ) > k:
        hq.heappop(maxpriorQ)
  hq.heappush(minpriorO, (-min(maxpriorQ), i))
  if len(minpriorO) > m:
    hq.heappop(minpriorO)


print("--- %s seconds ---" % (time.time() - start_time))

#print the outliers...
for outlier in minpriorO:
  print (outlier)
```

```
    --- 58.873881101608276 seconds ---
    (21.326433042228828, 3001)
    (23.66702461570138, 3003)
    (21.33061772535937, 3002)
    (24.31778944358807, 3005)
    (24.2815644018392, 3004)
```

```
np.random.shuffle(all_data)


#start the timer
start_time = time.time()

#the priority queue of outliers
outliers = []

#YOUR CODE HERE!
minpriorO = []
for i, vec1 in enumerate(all_data):
  maxpriorQ = []
  for j, vec2 in enumerate(all_data):
    if i != j:
      dist = np.linalg.norm(vec1 - vec2)
      #print("exec")
      hq.heappush(maxpriorQ, -dist) #The negative sign makes the heap maxpriority
      if len(maxpriorQ) > k:
        hq.heappop(maxpriorQ)

    if len(maxpriorQ) == k:
      if len(minpriorO) == m and -maxpriorQ[0] < minpriorO[0][0]:
        break

  hq.heappush(minpriorO, (-maxpriorQ[0], i))
  if len(minpriorO) > m:
    hq.heappop(minpriorO)



print("--- %s seconds ---" % (time.time() - start_time))

#print the outliers...
for outlier in minpriorO:
  print (outlier)
```

```
--- 2.30839467048645 seconds ---
(21.326433042228828, 1759)
(21.33061772535937, 568)
(24.2815644018392, 2370)
(24.31778944358807, 1908)
(23.66702461570138, 1481)
```