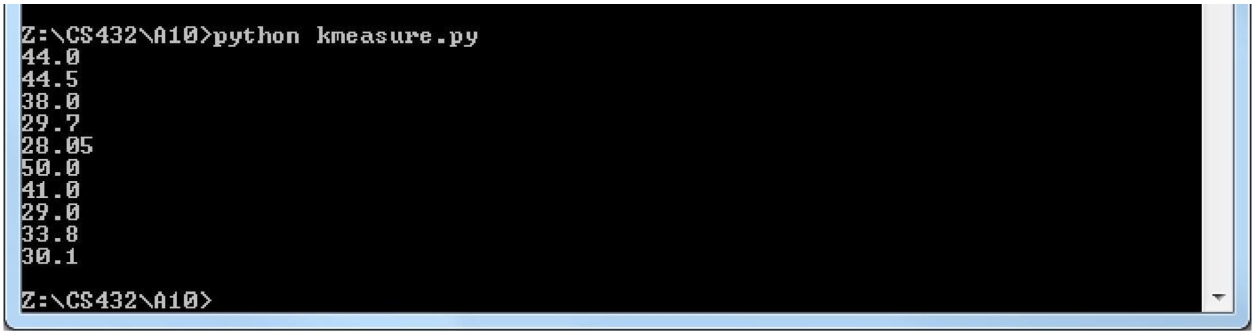
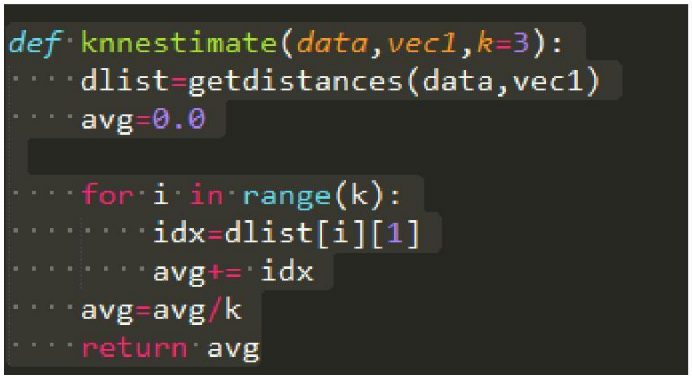
Question 1:



```
Z:\CS432\A10>python kmeasure.py
44.0
44.5
38.0
29.7
28.05
50.0
41.0
29.0
33.8
30.1

Z:\CS432\A10>
```



```
def knnestimate(data,vec1,k=3):
    dlist=getdistances(data,vec1)
    avg=0.0

    for i in range(k):
        idx=dlist[i][1]
        avg+= idx
    avg=avg/k
    return avg
```

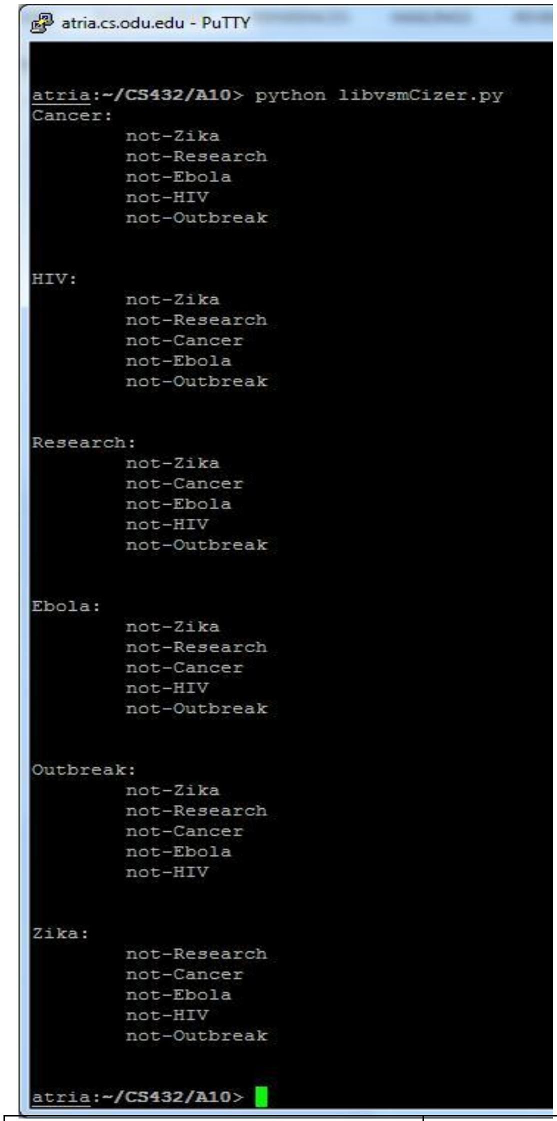
K-Estimate Computations:

44.0
44.5
38.0
29.7
28.05
50.0
41.0
29.0
33.8
30.1

Name	K = 1	K = 2	K = 5	K = 10	K = 20
`http://f-measure.blogspot.com/`	44	44.5	38.0	29.7	28.05
`http://ws-dl.blogspot.com/`	50.0	41.0	29.0	33.8	30.1

After obtaining the necessary functions from the Collective Intelligence Programming textbook, the computations were able to be done correctly.

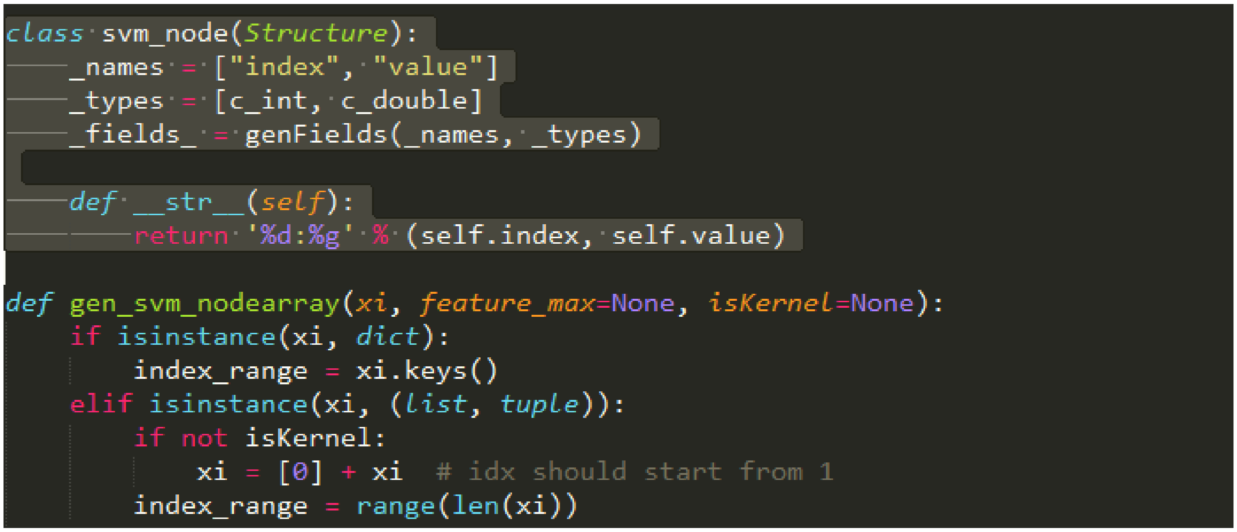
Question 2:



```
atria:~/CS432/A10> python libvsmCizer.py
Cancer:
        not-Zika
        not-Research
        not-Ebola
        not-HIV
        not-Outbreak


HIV:
        not-Zika
        not-Research
        not-Cancer
        not-Ebola
        not-Outbreak


Research:
        not-Zika
        not-Cancer
        not-Ebola
        not-HIV
        not-Outbreak


Ebola:
        not-Zika
        not-Research
        not-Cancer
        not-HIV
        not-Outbreak


Outbreak:
        not-Zika
        not-Research
        not-Cancer
        not-Ebola
        not-HIV


Zika:
        not-Research
        not-Cancer
        not-Ebola
        not-HIV
        not-Outbreak

atria:~/CS432/A10>
```

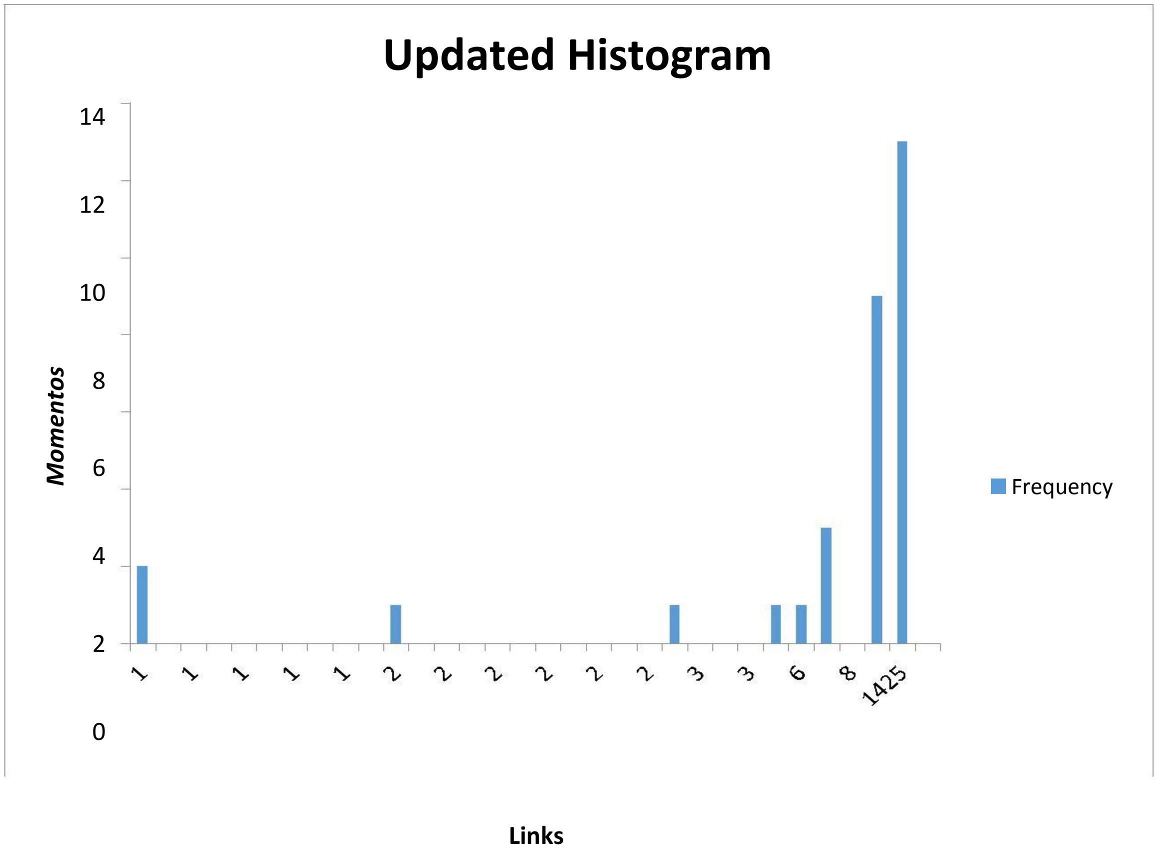
Category	Cross-Validation%	Correction%
Cancer	85.00%	66.00%
HIV	35.00%	42.00%
Research	68.00%	76.00%
Ebola	88.00%	66.00%
Outbreak	72.00%	46.00%
Zika	91.00%	95.00%

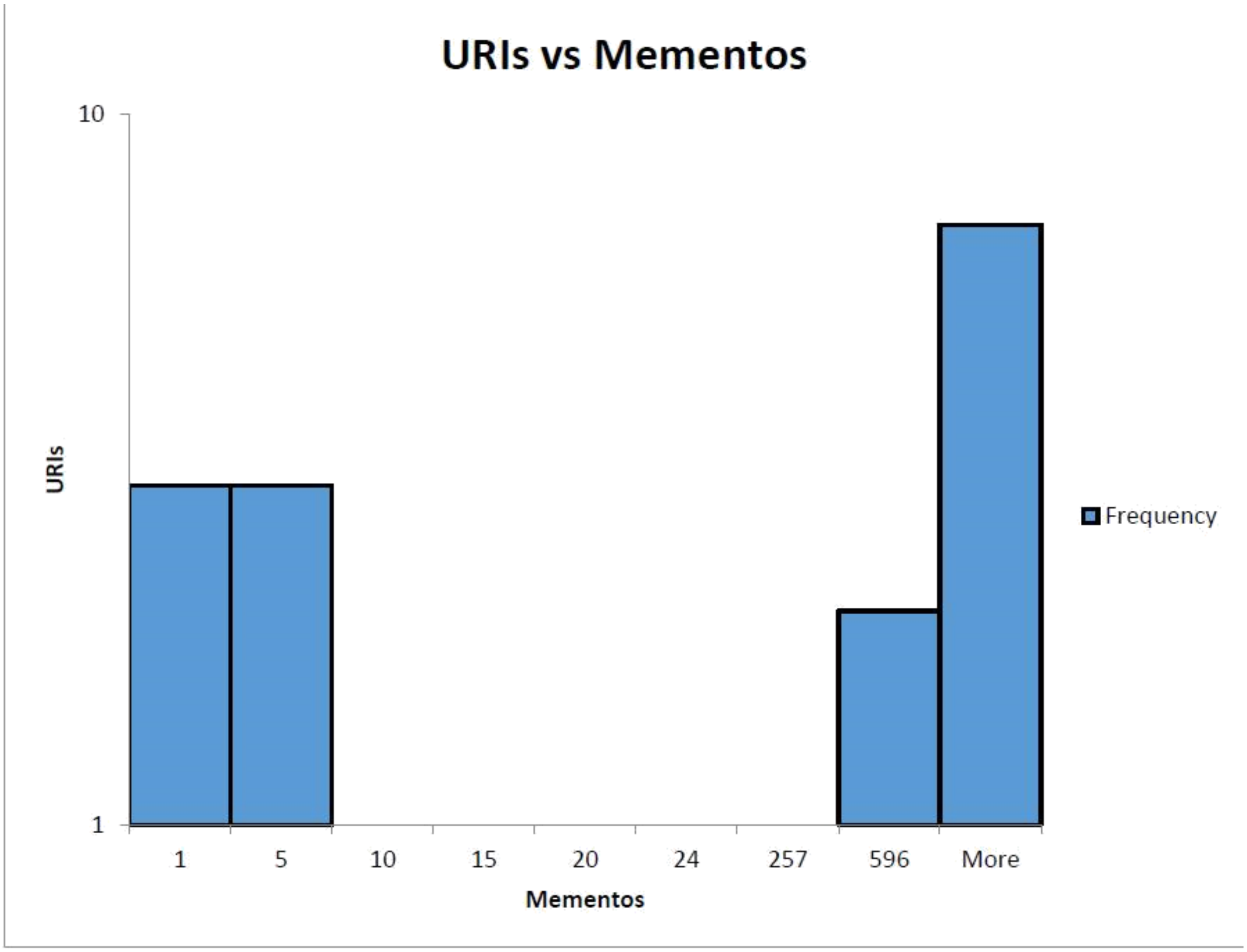
For question 2, I was able to figure out the LIBVSM scheme, however the 500 term vectors and 10-fold cross-validations was rather confusing. I researched on it for a while trying to figure out on how to implement it but no such luck up until Saturday when I decided to give it one last look. I found the necessary code form some of the LIBVSM tutorials online.



```python
class svm_node(Structure):
    _names = ["index", "value"]
    _types = [c_int, c_double]
    _fields_ = genFields(_names, _types)

    def __str__(self):
        return '%d:%g' % (self.index, self.value)

def gen_svm_nodearray(xi, feature_max=None, isKernel=None):
    if isinstance(xi, dict):
        index_range = xi.keys()
    elif isinstance(xi, (list, tuple)):
        if not isKernel:
            xi = [0] + xi  # idx should start from 1
        index_range = range(len(xi))
```

Question 3 (Extra Credit):





For the Extra Credit question 3, the first graph is an updated histogram compared to the old one. As it can be seen the graphs differ when it was produced in Assignment 2.

1	2
1	0
1	0
1	0
1	0
1	0
1	0
1	0
1	0
1	0
2	1
2	0
2	0
2	0
2	0
2	0

2	0
2	0
2	0
2	0
2	0
3	1
3	0
3	0
3	0
5	1
6	1
8	3
8	0
43	9
1425	13