Machine Learning(10-601) Assignment#5

a) K=2.
no. of data points = n.
for each point in the given data set, we have an options to assign it to any of the given cluster. for K=2, it can either be cluster 1 or cluster 2. So, each points can be assigned in 2^n ways.
for each of the assignment (from 2^n ways), we need to calculate the objective function i.e $\sum \|x_j - c_i\|^2$, and see for which assignment the value is minimum. That will be the best assignment with minimum K-mean objective value.
The time complexity of all possible assignment = $O(2^n)$
time for calculating the K-mean objective = $O(n)$
Thus, running time of Brute-force algorithm will be upper bound by slowest running-time i.e $O(2^n)$

b), c), d), e) → Code submitted through autolab.

f) Script name = KvsKmean.m.
plot name = Experiment1-f.fig

g) Value of K, I would choose = 9 (nine)
Yes, it does agree with my intuitions as there are 9 clusters in the figure 1 shown in the assignment

Experiment #2

h) Average K-mean objective value:

Random initialization = 86.79

K-mean ++ = 77.96

i) If ~~one~~ gaussian ~~bands~~ has two centers, then it will distribute the density among data of that gaussians and some other cluster would be share two cluster which would not be optimal.

Applying K-mean ~~plus~~ ++, we will pick the next center which would be farthest from ~~all~~ the already assigned cluster.

so, if first center is choosen to be on point in one of the 9 gaussian function, then next would be a point which is farthest from it i.e the point in the cluster will be chosen such that the distance from this cluster. Likewise, next point nearest of two cluster is farthest. so, never will a point choosen will fall into ~~on~~ cluster with already assigned cluster point as distance between gaussian cluster is more than distance among the points in cluster.

① ⑥ ③

⑦ ⑨ ⑤ → one of the possible assigment through K mean ++.

④ ⑧ ②

Problem 2: Disagreement based Active learning

~~a) It's easy concept~~

a) True.
Since this is a relizable case, means the hypothesis / classifier exists in the concept class H. For any given number of sample, we should have a classifier that would classify the given queried data point. With the addition of more points, only the hypothesis will change to classify the additional data point correctly. So for any given t, we would have a hypothesis. Thus $H_t \neq \emptyset$.

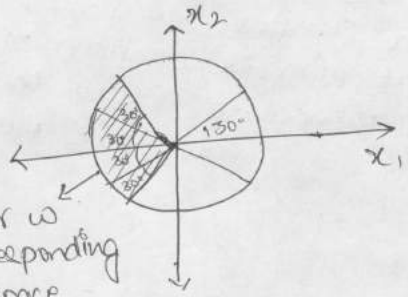
b) True.
While applying active learning, we request for the class of data points which are most informative or which lies closest to the classifier. Likewise, at each step, the classifier will change to adjust the new point. With all n points discovered/Queried, there will be one unique hypothesis that will classify the data (n-points)
This classifier would be same as that of ~~as~~ applying supervised techniques like SVM, perceptron, naive bayes etc.
It will be unique.

c) True.
If we pick any point in the region of disagreement, our version space will shrink. Because the hypothesis will change with the addition of any point from the region of disagreement.

An Example)

$$h_w(x) = \begin{cases} +1 & w^T x \geq 0 \\ -1 & w^T x < 0 \end{cases}$$

d)



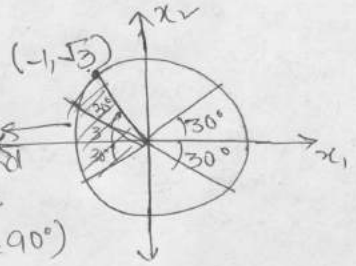
Set of vector w that is corresponding to version space.

(coverage $= \frac{2\pi}{3} = 120°$)

Angle Range: [from x_1 (+ve)]

$120° \leftrightarrow 240°$

e)



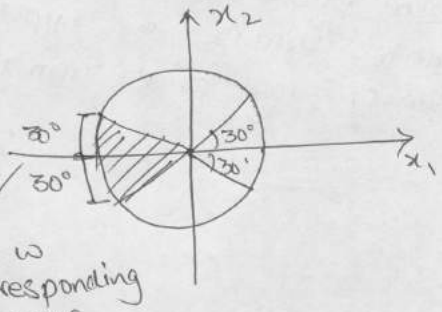
Set of vector w that is corresponding to version space

(coverage $= \frac{\pi}{2} = 90°$)

Angle Range:

$120° \leftrightarrow 210°$ (from the X-axis)

f)



Set of vector w that is corresponding to version space

(coverage $= \frac{\pi}{3} = 60°$)

Angle Range: (from the X-axis)

$150° \leftrightarrow 210°$

Problem 3: Parity function

$$h_S(x) = \left(\sum_{i \in S} x_i\right) \bmod 2$$

$$S \subseteq \{1, 2, \ldots n\}$$

$$H_{parity} = \{h_S : S \subseteq \{1, 2, 3, \ldots, n\}\}$$

a) VC dimensions is the cardinality of maximum number of points that can be shattered by the sample set of hypothesis. In this case of Hparity, the maximum of value of S can go upto n (upper bound), which means that total number of combination it can go upto is 2^n. With these n points, 2^{n+1} possible combinations of output (shattering) wouldn't be possible so the maximum dimensality or shattering can go upto n.

So, VC dimenson of Hparity $\leq n$. H is finite ie 2^n.

$VC\dim(H) \leq \log|H| \leq \log 2^n \leq n$

b) With $S=n$, x will have n bits, each of the bit can either take 0 or 1, thus we can say that the total number of combination of x can be 2^n with $S=n$.

With one flip of bit the hypothesis will also flip the output from 1 to 0 or 0 to 1.

Hparity ($h_S(x)$) will output 0/1 based on the number of digits as even or odd.

Likewise, with $S=n$ total no. of combination (shattering) possible can be 2^n. Thus, the VC dimensionality of Hparity is n.

Active vs passive learning

c) Starting with a single example, we would have a naive hypothesis, then we query another label of the point closest to the present hypothesis. Based on the class of the label, we say that how we need to shrink the version space and have more confidence bound.
Likewise querying the points from the region of disagreement or uncertainty would give us the confidence bound of better hypothesis.
With every addition of point query, we get better hypothesis and will shrink the version space.
So, after querying n points, we would have a hypothesis which would classify those points perfectly (reliagable case)
So, we can learn the true hypothesis using n queries

d) Passive learning, (Realizable Case)
As per PAC learning theory, number of sample needed to have generalization error of ϵ with probability $1-\delta$ is:

$$m = O\left(\frac{1}{\epsilon}\left[VC\dim(H)\log\frac{1}{\epsilon} + \log\left(\frac{1}{\delta}\right)\right]\right)$$

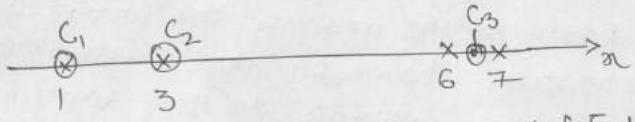
$VC\dim(H) = n$.

So,

$$\boxed{m = O\left(\frac{n}{\epsilon}\log\frac{1}{\epsilon} + \log\left(\frac{1}{\delta}\right)\right)}$$

Problem 4 : K-means on the real line

a)



Having cluster centers at $x = 1, 3$ and 6.5 would give us the optimal clustering with K-mean objective value as 0.5

$$\text{K-mean objective} = \sum_{j=1}^{n} \min_{i \in 1 \ldots k} \|x_j - c_i\|^2$$

$$= \|1-1\|^2 + |3-3|^2 + |6-6.5|^2 + |7-6.5|^2$$

$$= .25 + .25 = 0.5$$

b) Lloyd's method with random initialization might not give us optimal results as we got in part a. If we choose the initial points at the below location.

$c_1 = 6$, $c_2 = 7$ and $c_3 = 3$.

After convergence, c_3 will shift to 2 and have data points 1 and 3 under it. However, cluster points 6 & 7 will be assigned to c_1 & c_2 respectively and will remain like that. There will be no displacement in c_1 & c_2 as it has least distance assignment already w.r.t points assigned.

$$\text{k-mean objective value} = \sum_{j=1}^{n} \min_{i \in 1 \ldots k} \|x_j - c_i\|^2$$

$$= |6-6|^2 + |7-7|^2 + |1-2|^2 + |3-2|^2$$

$$= 1+1 = 2$$ (which is more than 0.5, we got in part a.

c)

Cluster are assigned based on the mean of the current assignment, then the assignment happens based on the nearest cluster.

In 1-D, the distance is the difference in the location of x-axis. So, for any cluster, the point assignment would be the points nearest to it in x-axis.

lets say if the points are not consecutive, x_i & x_{i+2} are C_j & x_{i+1} is C_{j+1}, then the k-means objective can further be reduced by switching x_{j+1} & x_{i+2} cluster center. So, any cluster center can be assigned only to points in consecutive manner.

d) Dynamic programming algorithm $\rightarrow O(Kn^2)$.

Algorithm:

(Bottom-up approach) $\rightarrow$ look up table

① Create a nxk table which will give the most optimal location of k^{th} cluster with i^{th} points taken into consideration.

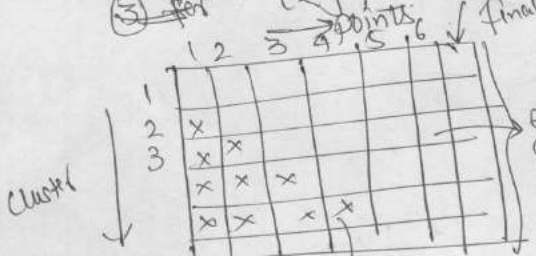
② for i^{th} point added to already i-1 points with $j-1^{th}$ cluster, there are n possible combination of assignment of j^{th} cluster to i points

j^{th} cluster assignments of points (1 points)

$(i-j-2)$ points $\rightarrow (i-j-1)$ . . .

for each of these points we have optimal solution for existing points with. so $O(n)$ for each i^{th} point & j cluster

③ for $i < j$



Time complexity for filling each value $= O(n)$

Total time complexity

$= O(nxK \times n)$

$= O(n^2K)$

Experiment#1, Part (f)

```
% Experiment#1, Part (f)
function [a]= kvskmean()

    obj=0;
    load kmeans_data;
    [a,b]= size(X);
    fprintf('size of x = %d %d \n',a,b);
    init = 'random';
    num_restarts=10;
    obj_val = size(20,1);

    for k=1:1:20
        [best_C, best_a, best_obj] = kmeans_cluster(X, k, init,
num_restarts);
        fprintf('k = %d best_obj = %d\n',k,best_obj);
        obj_val(k,1) = best_obj;
    end

    x = linspace(1,1,20);
    figure
    plot(obj_val);
    title('K vs K mean obj value');
    xlabel('K (number of clusters)');
    ylabel('K mean obj value');

end
```

Figure :