Istanbul Technical University

BLG527E Machine Learning(HW5)
Aydin Ayanzadeh
Department of Computer Engineering
Email:ayanzadeh17@itu.edu.tr
student ID: 504161503

January 10, 2018

Q 1	Q 2	Q 3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20

Note: I write some of my answers in the paper and take the photo from that at the end of question that has written by word processing.
Q2)

$$\begin{aligned} P(X_i|X_{i-1}) &= 0.8 \\ P(\sim X_i|X_{i-1}) &= 0.2 \\ P(X_i|\sim X_{i-1}) &= 0.1 \\ P(\sim X_i|\sim X_{i-1}) &= 0.9 \\ P(X_1) &= 0.2 \end{aligned} \tag{1}$$

$$P(X_3|X_1, \sim X_4) = \alpha\pi(X_3)\lambda(X_3) \tag{2}$$

$$\begin{aligned} \pi(X_3) &= \Sigma_{X_2} P(X_3|X_2)\pi(X_2) \\ &= P(X_3|X_2)\pi(X_2) + P(X_3|\sim X_2)\pi(\sim X_2) \\ &= P(X_3|X_2)P(X_2|X_1)P(X_1) \\ &+ P(X_3|\sim X_2)P(\sim X_2|X_1)P(X_1) \\ &= 0.8 \times 0.8 \times 0.2 + 0.1 \times 0.2 \times 0.2 \\ &= 0.132 \end{aligned} \tag{3}$$

$$\begin{aligned} \lambda(X_3) &= P(\sim X_4|X_3)\lambda(\sim X_4) \\ &= 0.2 \end{aligned} \tag{4}$$

Now we have:

$$\begin{aligned} P(X_3|X_1, \sim X_4) &= 0.132 \times 0.2\alpha \\ &= 0.0264\alpha \end{aligned} \tag{5}$$

In order to find the α we use the following summation.

$$P(X_3|X_1, \sim X_4) + P(\sim X_3|X_1, \sim X_4) = 1 \tag{6}$$

$$P(\sim X_3|X_1, \sim X_4) = \alpha\pi(\sim X_3)\lambda(\sim X_3) \tag{7}$$

$$\begin{aligned} \pi(\sim X_3) &= \Sigma_{X_2} P(\sim X_3|X_2)\pi(X_2) \\ &= P(\sim X_3|X_2)\pi(X_2) + P(\sim X_3| \sim X_2)\pi(\sim X_2) \\ &= P(\sim X_3|X_2)P(X_2|X_1)P(X_1) \\ &+ P(\sim X_3| \sim X_2)P(\sim X_2|X_1)P(X_1) \\ &= 0.2 \times 0.8 \times 0.2 + 0.9 \times 0.2 \times 0.2 \\ &= 0.068 \end{aligned} \tag{8}$$

$$\begin{aligned} \lambda(\sim X_3) &= P(\sim X_4| \sim X_3)\lambda(\sim X_4) \\ &= 0.9 \\ \\ P(\sim X_3|X_1, \sim X_4) &= 0.068 \times 0.9\alpha \\ &= 0.0612\alpha \end{aligned} \tag{9}$$

We replace the results in equation 18 to find the α.

$$\begin{aligned} 0.0264\alpha + 0.0612\alpha &= 1 \\ 0.0876\alpha &= 1 \\ \alpha &= 11.415 \end{aligned} \tag{10}$$

$$\begin{aligned} P(X_3|X_1, \sim X_4) &= 0.132 \times 0.2\alpha \\ &= 0.0264 \times 11.415 \\ &= 0.301559 \end{aligned} \tag{11}$$

Q3 a)

$$\begin{aligned} H_0 &: \mu = 0 \\ H_1 &: \mu \neq 0 \end{aligned} \tag{12}$$

$$\begin{aligned} m &= \frac{\Sigma_{i=1}^{K} a}{K} \\ S^2 &= \frac{\Sigma_{i=1}^{K} (a - m)^2}{K - 1} \\ m &= -0.001 \\ S^2 &= 0.000943 \end{aligned} \tag{13}$$

Under the null hypothesis that $\mu = 0$, we use the following t-Distributed statistic with K-1 degree of freedom.

$$\begin{aligned} \frac{\sqrt{K}(m)}{\sqrt{S^2}} &\sim t_{K-1} \\ \frac{\sqrt{10}(-0.001)}{\sqrt{0.000943}} &\sim t_{K-1} \\ -0.10298 &\sim t_{K-1} \end{aligned} \tag{14}$$

Q4 b) We use the following null hypothesis and to accept or reject that the classification algorithms have the same expected error rates as significance level $\alpha = 0.05$.

$$\begin{gathered} H_0 : \mu_{MLP} = \mu_{KNN} = \mu_{NB} \\ H_1 : \mu_r \neq \mu_s, \text{ for at least one pair (r, s)} \end{gathered} \tag{15}$$

We calculate the mean of error rates for each class.

$$\begin{aligned} m_{MLP} &= \Sigma_{i=1}^{10} \frac{X_{iMLP}}{10} \\ &= 0.033 \\ m_{KNN} &= \Sigma_{i=1}^{10} \frac{X_{iKNN}}{10} \\ &= 0.034 \\ m_{NB} &= \Sigma_{i=1}^{10} \frac{X_{iNB}}{10} \\ &= 0.048 \end{aligned} \tag{16}$$

$$\begin{aligned} m &= \Sigma_{j=1}^{3} \frac{m_j}{L} \\ &= \frac{m_{MLP} + m_{KNN} + m_{NB}}{3} \\ m &= 0.0383 \\ S^2 &= \Sigma_{j=1}^{3} \frac{(m_j - m)^2}{L-1} \\ &= 7.0335 \times 10^{-5} \end{aligned} \tag{17}$$

$$\begin{aligned} SS_b &= \Sigma_{j=1}^{3} (m_j - m)^2 \\ &= 0.001407 \end{aligned} \tag{18}$$

$$\begin{aligned} SS_w &= \Sigma_{j=1}^{3} \Sigma_{i=1}^{10} (X_{ij} - m_j)^2 \\ &= 7(-0.033)^2 + 2(0.13 - 0.033)^2 + (0.07 - 0.033)^2 \\ &= 6(-0.034)^2 + 3(0.07 - 0.034)^2 + (0.13 - 0.034)^2 \\ &= 5(0.07 - 0.048)^2 + 4(-0.048)^2 + (0.13 - 0.048)^2 \\ &= 0.06621 \end{aligned} \tag{19}$$

$$\begin{aligned} F_0 &= \frac{SS_b/(L-1)}{SS_w/(L(K-1))} \\ &= \frac{0.001407/2}{0.06621/27} \\ &= 0.28688 \end{aligned} \tag{20}$$

Q19 The *Deep Nueáral Network* sWhat are the differences between a deep neural network and multilayer perceptron?

Deep-learning networks are distinguished from the more commonplace single-hidden-layer neural networks by their depth; that is, the number of node layers through which data passes in a multistep process of pattern recognition.

Traditional machine learning relies on shallow nets, composed of one input and one output layer, and at most one hidden layer in between. More than three layers (including input and output) qualifies as deep learning. So deep is a strictly defined, technical term that means more than one hidden layer.

In deep-learning networks, each layer of nodes trains on a distinct set of features based on the previous layers output. The further you advance into the neural net, the more complex the features your nodes can recognize, since they aggregate and recombine features from the previous layer.

CNN[1] are very similar to ordinary Neural Networks ,they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer and all the tips/tricks we developed for learning regular Neural Networks still apply.

So what does change? ConvNet architectures make the explicit assumption that the inputs are images, which allows us to encode certain properties into the architecture. These then make the forward function more efficient to implement and vastly reduce the amount of parameters in the network. There are several architectures in the field of Convolutional Networks that have a name. The most common are:LeNet,AlexNet,ZF Net,GoogLeNet,VGGNet, ResNet and DenseNet.

Dropout is a regularization technique where, while you're updating a layer of your neural net, you randomly don't update, or "dropout," half of the layer. That is, while updating your neural net layer, you update each node with probability 1/2, and leave it unchanged with probability 1/2. This helps prevent the net from relying on one node in the layer too much. I haven't read it in detail, but this is one of the earlier papers on dropout and looks like it motivates it pretty well

Q15)a *Adaptive learning Rate:* In gradient Descent, the learning factor η determines the magnitude of change to be made in the parameter. It solve the problem of learning rate for solving the specific learning problem.

[1] Convolutional Neural Networks

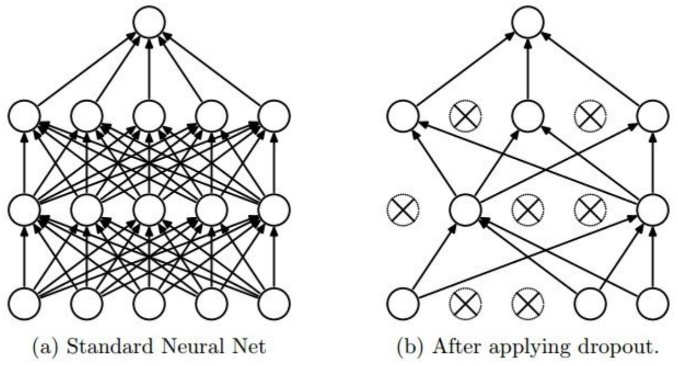


Figure 1: Dropout Neural Net Model.Left: A standard neural net with 2 hidden layers.Right:An example of a thinned net produced by applying dropout to the network on the left.Crossed units have been dropped

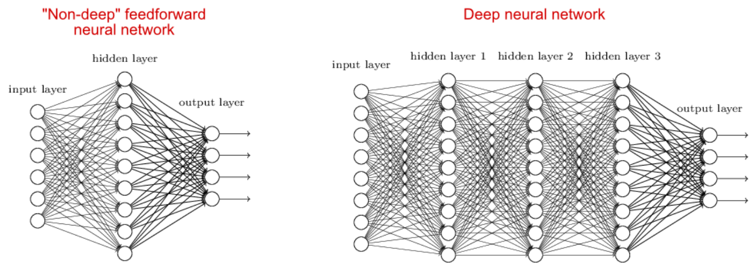


Figure 2: The histograms of generated datasets .

*Mpmentum:*In neural networks, we use gradient descent optimization algorithm to minimize the error function to reach a global minima.
So you are guaranteed to find the global optimum because there are no local minimum where your optimization can get stuck. However in real the error surface is more complex, may comprise of several local minimal.In this case, you can easily get stuck in a local minima and the algorithm may think you reach the global minima leading to sub-optimal results. To avoid this situation, we use a momentum term in the objective function, which is a value between 0 and 1 that increases the size of the steps taken towards the minimum by trying to jump from a local minima. If the momentum term is large then the learning rate should be kept smaller. A large value of momentum also means that the convergence will happen fast. But if both the momentum and learning rate are kept at large values, then you might skip the minimum with a huge step. A small value of momentum cannot reliably avoid local minima, and can also slow down the training of the system. Momentum also helps in smoothing out the variations, if the gradient keeps changing direction. A right value of momentum can be either learned by hit and trial or through cross-validation.

*L2 regularization:*we regularize our function to avoid overfitting in our MLP.

Q15)b I attach the answer of this question at the end of the page (I write it in the papert.)

Q17a) Bagging draws K bootstrap samples giving each training instance equal weight, trains K classifiers and takes the average/majority (regression/classification) of those K classifiers as the output. In Adaboost, the bootstrap sample for classifier i (i=1..K) is drawn based on the performance of previous classifiers, an instance which has been misclassified has more probability of being included in the tth bootstrap sample, weighted average of classifiers, based on how they perform on the whole dataset, is taken to produce the final classifier.

b)

8a) PCA projects inputs into a lower dimensional space, i.e. it is a feature projection technique. Each dimension in the lower dimensional space is a linear combination of the original inputs. The weights for the linear combination for the first dimension is the eigenvector corresponding to the largest eigenvalue, the second set of weights is the eigenvector corresponding to the next largest eigenvalue,etc. PCA tries to project the inputs according to the directions in which there is maximum variance.

but, Backward feature selection is a feature selection method. It first trains d classifiers, leaving out a single feature at a time. It chooses the feature whose validation error is the minimum as the feature to leave out. This process is repeated until enough number of features or validation error is reached

2-By feature selection you reducing number of dimension by throwing out irrelevant information. By PCA, you reducing number of dimension by transforming to artificial set.

8c) Lda is dimensionality reduction that maximize and seprate the datasets distance, but pca wants to maximize the the variance of datasets.

PCA: is provided for unsupervised learning but LDA is dimensonality reduction techniques for supervised learning. Q18)a dataset generated from a polynomial of degree of 4.

Q14 Hinge loss and Squared error in SVM: hinge loss also penalizes instances in the margin even though they may be on the correct side, and the loss increases linearly as the instance moves away on the wrong side.
SVM minimizes the error function: E = p * train + norm(W) We can change the value of p, smaller p means less emphasis on training error and more emphasis on small w, hence simpler models. Small w also means large margin we can use different kernel functions. For example linear kernel is less complicated than the quadratic. therfore these error function can help us to generalized our classifier better.
In Mlp

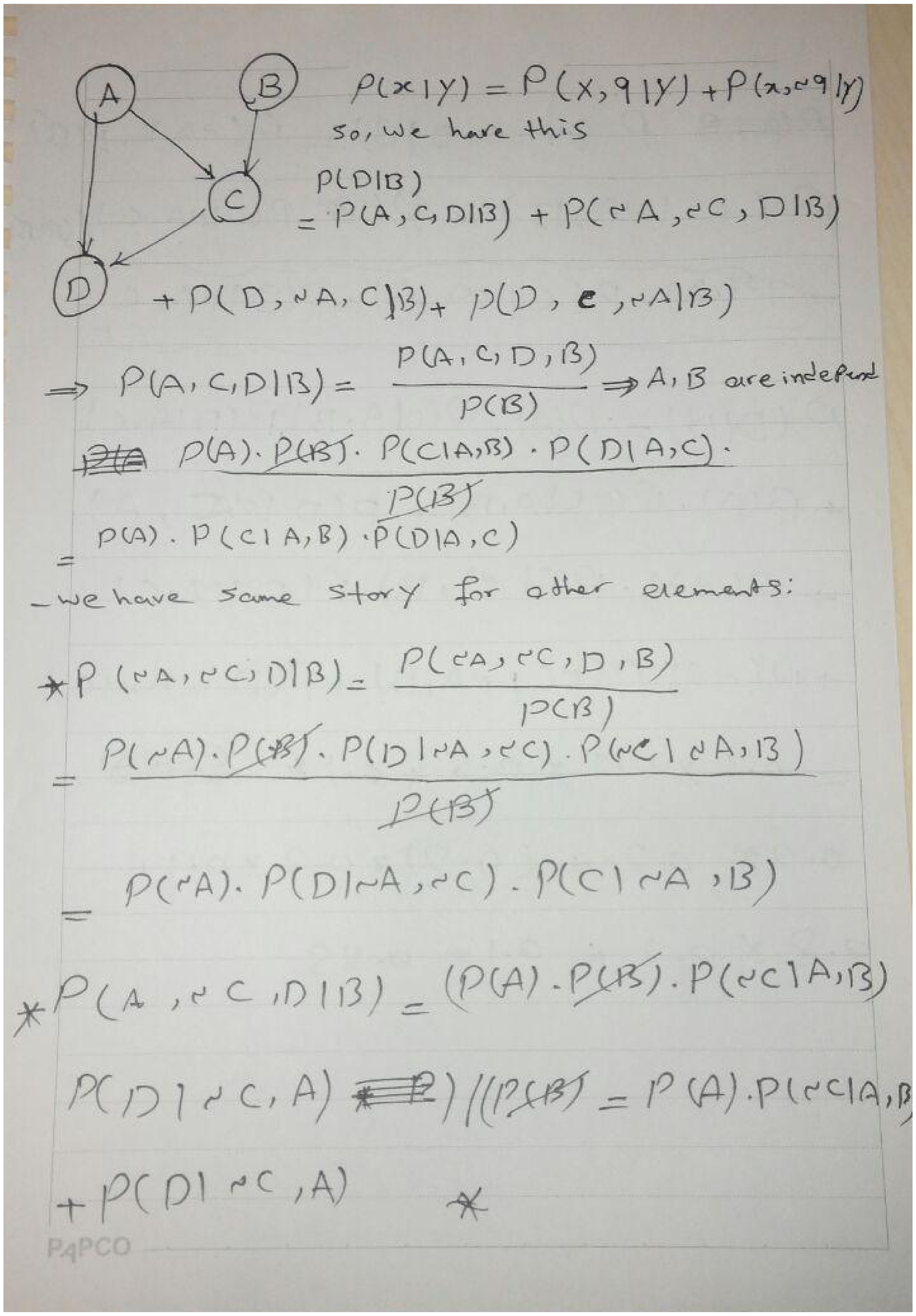
18c) The least complex classifier that can be obtained is the least mean classifier, which is a linear classifier. It just computes the distance between the input vector and the class means and selects the class whose mean is closest. This corresponds to assuming that both

Status	Bias	variance
Polynomial of degree 1	H	L
Polynomial of degree 4	L	L
Polynomial of degree 10	L	H
Polynomial of degree 10 trained with regularization	L	L

class covariance matrices are the same and can be written as 2 I where I is the 2 dxd identity matrix. The number parameters that need to be estimated is 2d + 1, d . parameters for each of the mean vectors and 1 parameter for 2 2 The most complex classifier is a quadratic classifier. It corresponds to assuming that both classes have arbitrary covariance matrices. The number of parameters that need to be estimated is 2d + d(d+1). Again d parameters for each of the mean vectors and 0.5 d(d+1) parameters for each of the class covariance matrices.

References

[1] Srivastava, Nitish, et al. ”Dropout: a simple way to prevent neural networks from overfitting.” Journal of machine learning research 15.1 (2014): 1929-1958.



$$\begin{tikzcd} A \arrow[dr] \arrow[dd] & & B \arrow[dl] \\ & C \arrow[dl] & \\ D & & \end{tikzcd}$$

$P(x|y) = P(x,q|y) + P(x,\sim q|y)$

so, we have this

$P(D|B)$

$= P(A,C,D|B) + P(\sim A,\sim C,D|B)$

$+ P(D,\sim A,C|B) + P(D,C,\sim A|B)$

$\Rightarrow P(A,C,D|B) = \frac{P(A,C,D,B)}{P(B)} \Rightarrow$ A, B are independ

$= \frac{P(A)\cdot\cancel{P(B)}\cdot P(C|A,B)\cdot P(D|A,C)\cdot}{\cancel{P(B)}}$

$= P(A)\cdot P(C|A,B)\cdot P(D|A,C)$

- we have same story for other elements:

$\star\, P(\sim A,\sim C,D|B) = \frac{P(\sim A,\sim C,D,B)}{P(B)}$

$= \frac{P(\sim A)\cdot\cancel{P(B)}\cdot P(D|\sim A,\sim C)\cdot P(\sim C|\sim A,B)}{\cancel{P(B)}}$

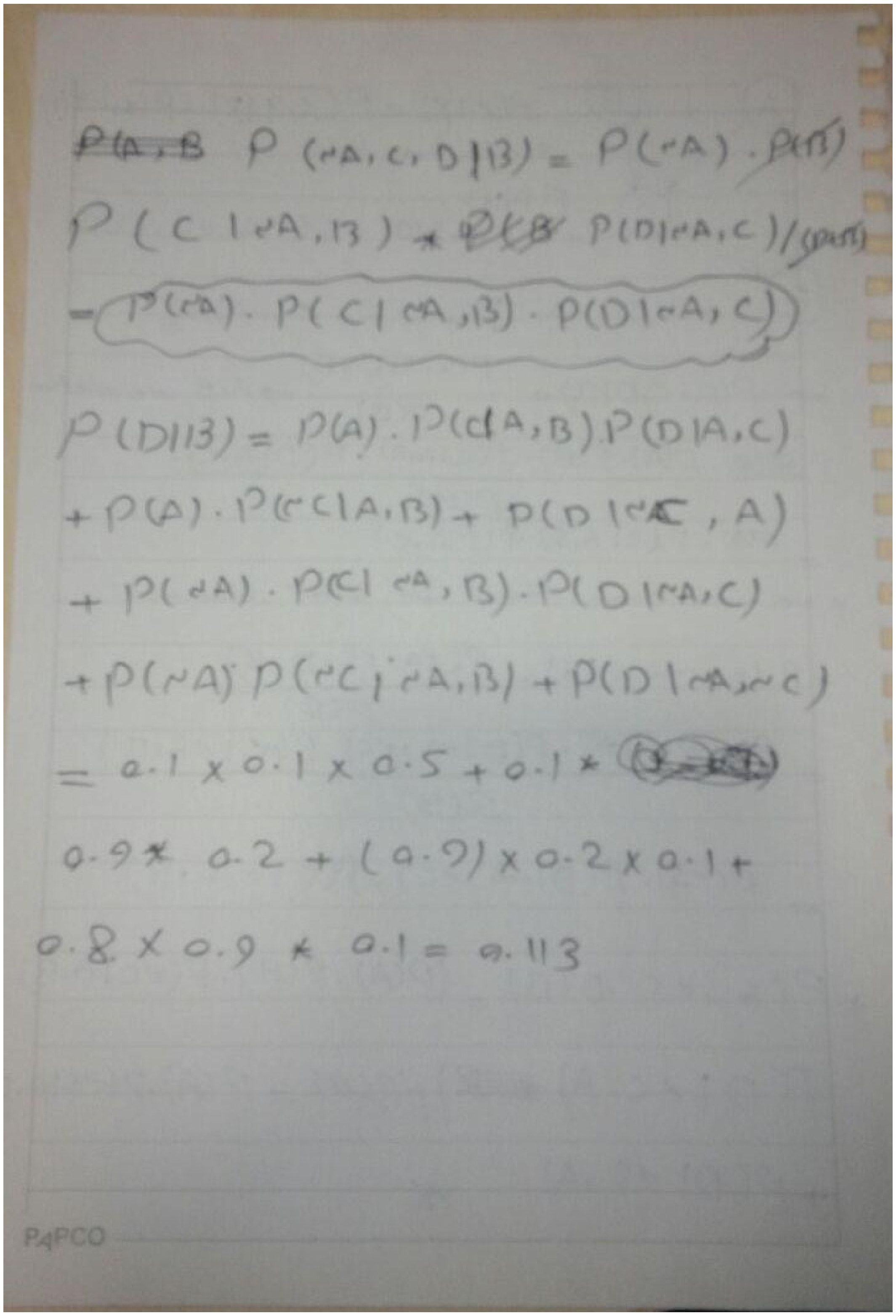
$= P(\sim A)\cdot P(D|\sim A,\sim C)\cdot P(C|\sim A,B)$

$\star\, P(A,\sim C,D|B) = (P(A)\cdot\cancel{P(B)}\cdot P(\sim C|A,B)$

$P(D|\sim C,A))/(\cancel{P(B)} = P(A)\cdot P(\sim C|A,B$

$+ P(D|\sim C,A)$ $\star$

Figure 3: Question 4



~~P(A, B~~ $P(\sim A, C, D \mid B) = P(\sim A) \cdot$ ~~P(B)~~

$P(C \mid \sim A, B) *$ ~~P(B~~ $P(D \mid \sim A, C)$ ~~/(P(B))~~

$= \boxed{P(\sim A) \cdot P(C \mid \sim A, B) \cdot P(D \mid \sim A, C)}$

$P(D \mid B) = P(A) \cdot P(C \mid A, B) \cdot P(D \mid A, C)$

$+ P(A) \cdot P(\sim C \mid A, B) + P(D \mid \sim C, A)$

$+ P(\sim A) \cdot P(C \mid \sim A, B) \cdot P(D \mid \sim A, C)$

$+ P(\sim A) \cdot P(\sim C \mid \sim A, B) + P(D \mid \sim A, \sim C)$

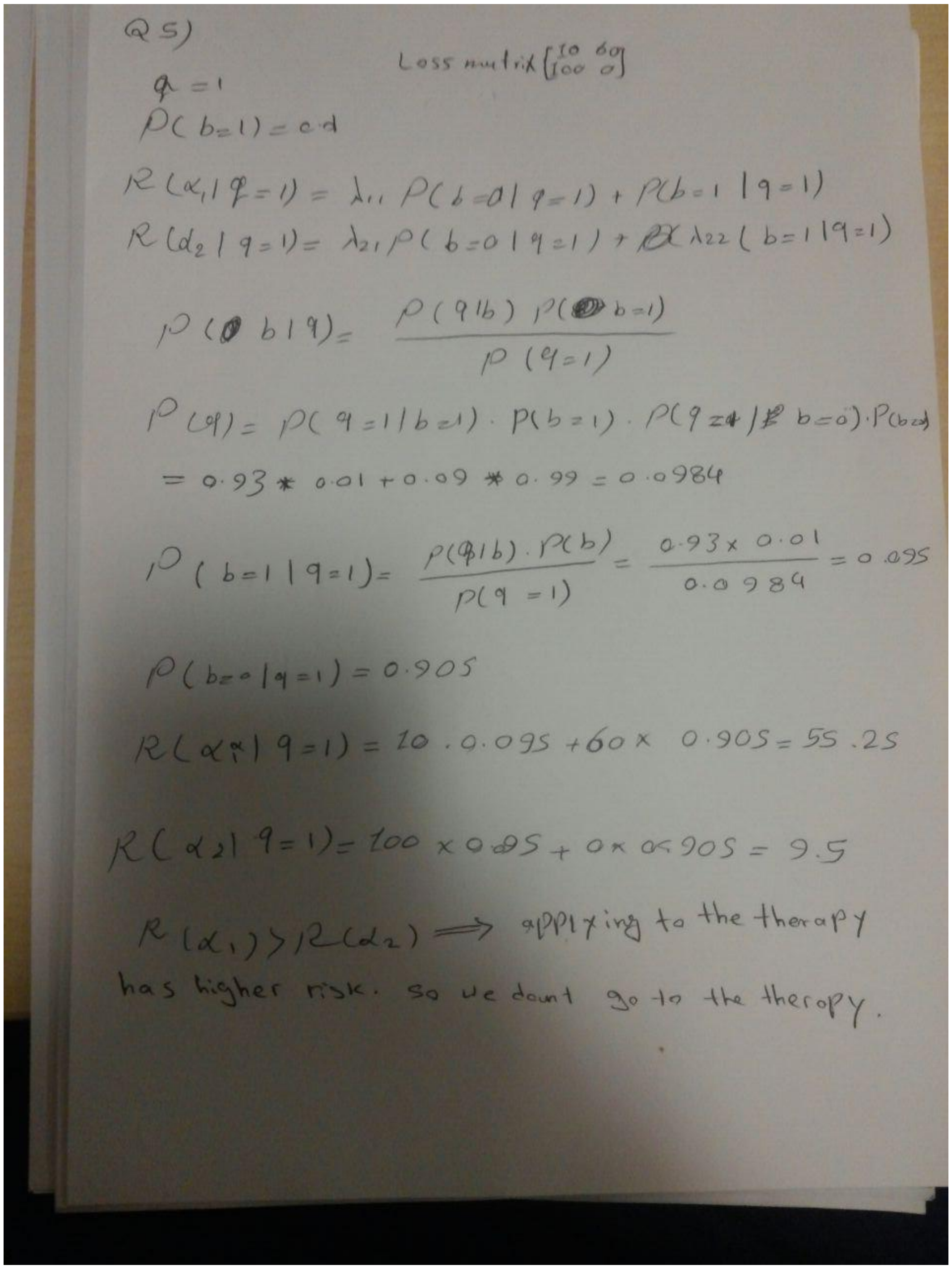
$= 0.1 \times 0.1 \times 0.5 + 0.1 *$ [illegible]

$0.9 * 0.2 + (0.9) \times 0.2 \times 0.1 +$

$0.8 \times 0.9 * 0.1 = 0.113$

PAPCO

Figure 4: Question 4



Q5)

Loss matrix $\begin{bmatrix} 10 & 60 \\ 100 & 0 \end{bmatrix}$

$q = 1$

$P(b=1) = 0.1$

$R(\alpha_1 | q=1) = \lambda_{11} P(b=0 | q=1) + P(b=1 | q=1)$

$R(d_2 | q=1) = \lambda_{21} P(b=0 | q=1) + \lambda_{22} (b=1 | q=1)$

$$P(b | q) = \frac{P(q|b)\, P(b=1)}{P(q=1)}$$

$P(q) = P(q=1 | b=1) \cdot P(b=1) \cdot P(q=1 | b=0) \cdot P(b=0)$

$= 0.93 * 0.01 + 0.09 * 0.99 = 0.0984$

$$P(b=1 | q=1) = \frac{P(q|b) \cdot P(b)}{P(q=1)} = \frac{0.93 \times 0.01}{0.0984} = 0.095$$

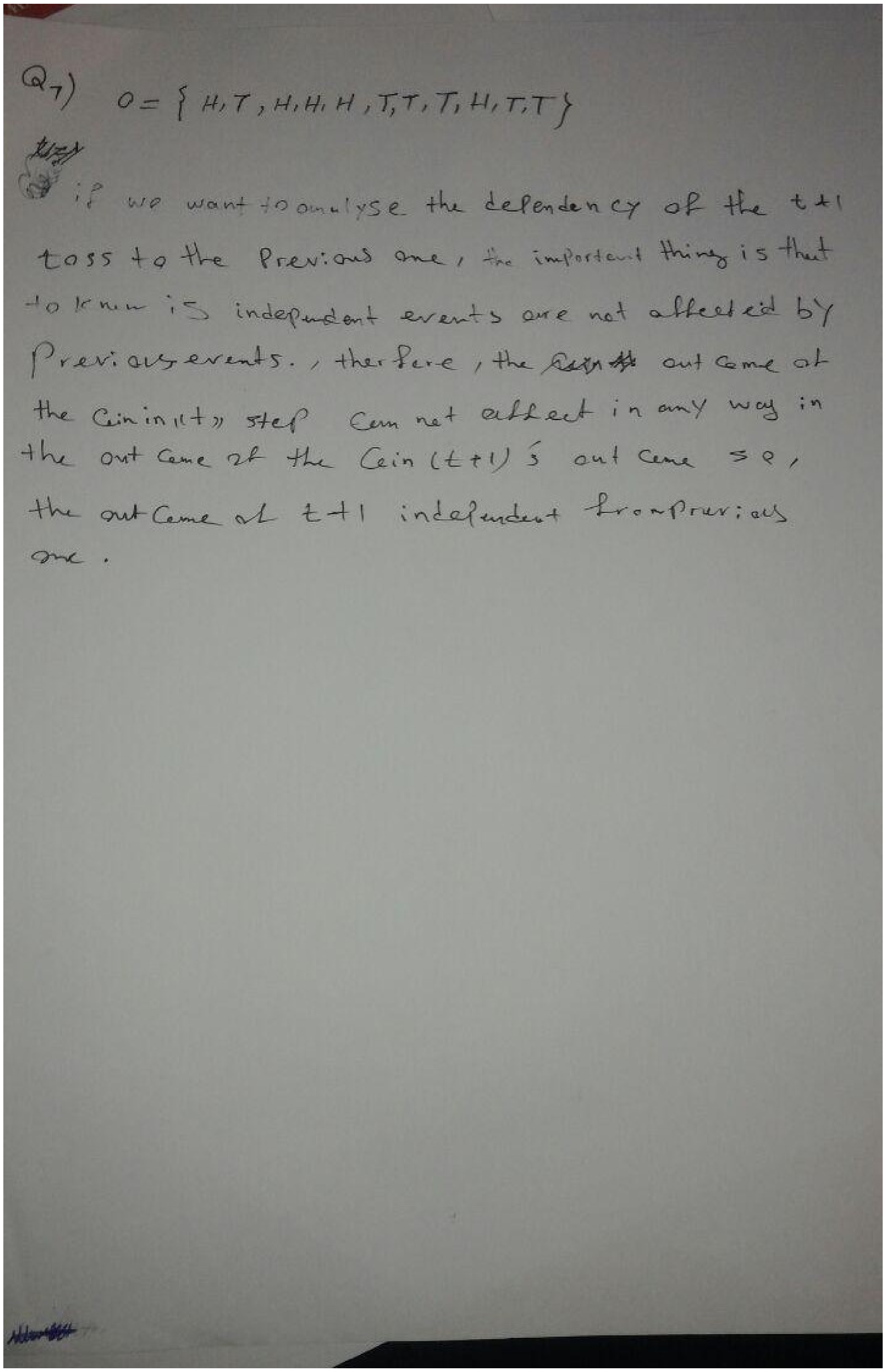
$P(b=0 | q=1) = 0.905$

$R(\alpha_1 | q=1) = 10 \cdot 0.095 + 60 \times 0.905 = 55.25$

$R(\alpha_2 | q=1) = 100 \times 0.095 + 0 \times 0.905 = 9.5$

$R(\alpha_1) > R(\alpha_2) \Longrightarrow$ applying to the therapy has higher risk. so we dont go to the theropy.

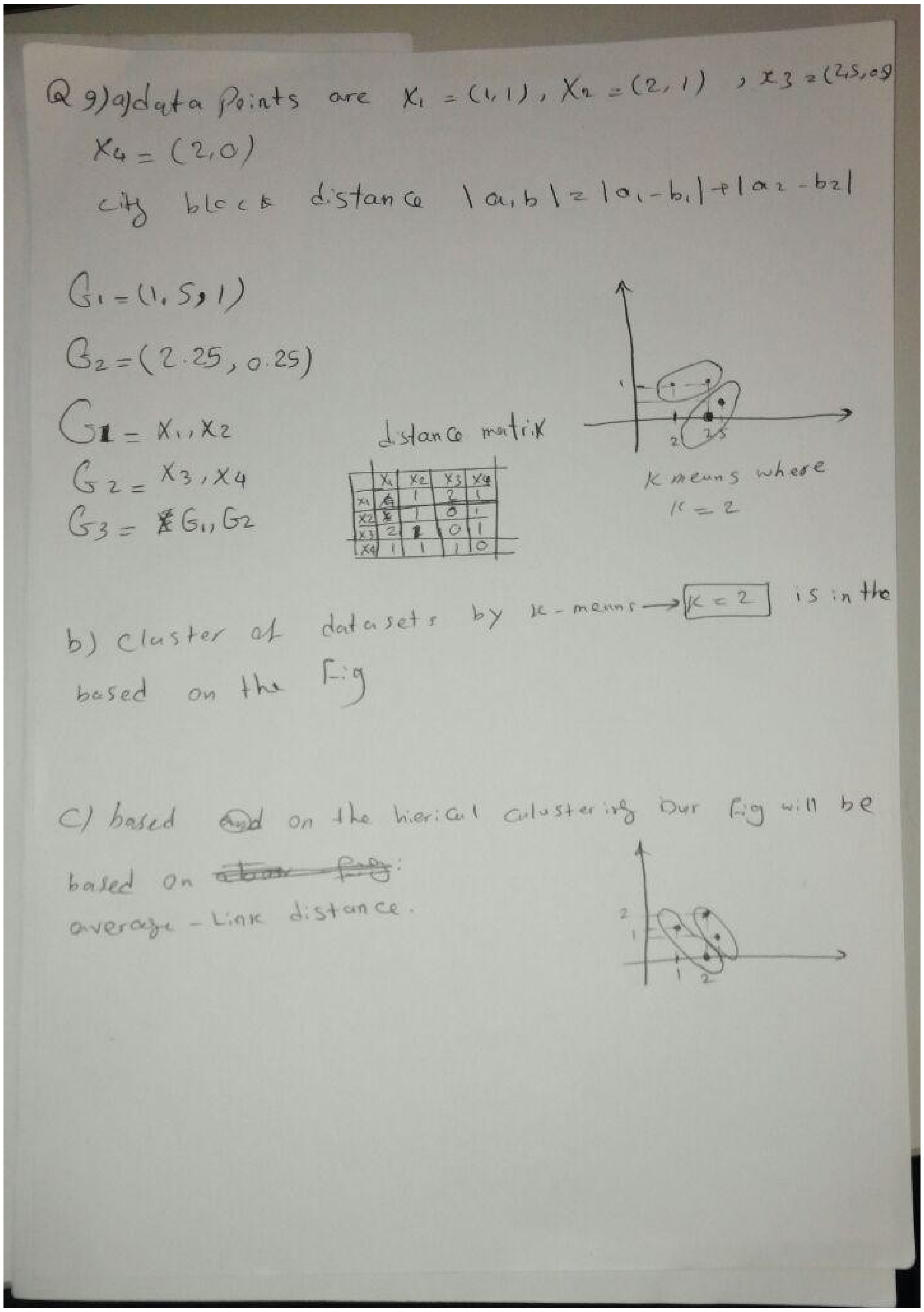
Figure 5: Question 5



Q_7) $O = \{H, T, H, H, H, T, T, T, H, T, T\}$

if we want to analyse the dependency of the $t+1$ toss to the previous one, the important thing is that to know is independent events are not affected by previous events., therfore, the ~~coin~~ out come of the coin in "t" step can not affect in any way in the out come of the coin $(t+1)$'s out come so, the out come of $t+1$ independent from previous one.

Figure 6: Question 7



Q 9) a) data Points are $X_1 = (1,1)$, $X_2 = (2,1)$, $x_3 = (2.5, 0.5)$

$X_4 = (2,0)$

city block distance $|a, b| = |a_1 - b_1| + |a_2 - b_2|$

$G_1 = (1.5, 1)$

$G_2 = (2.25, 0.25)$

$G_1 = X_1, X_2$

$G_2 = X_3, X_4$

$G_3 = \cancel{X} G_1, G_2$

distance matrix

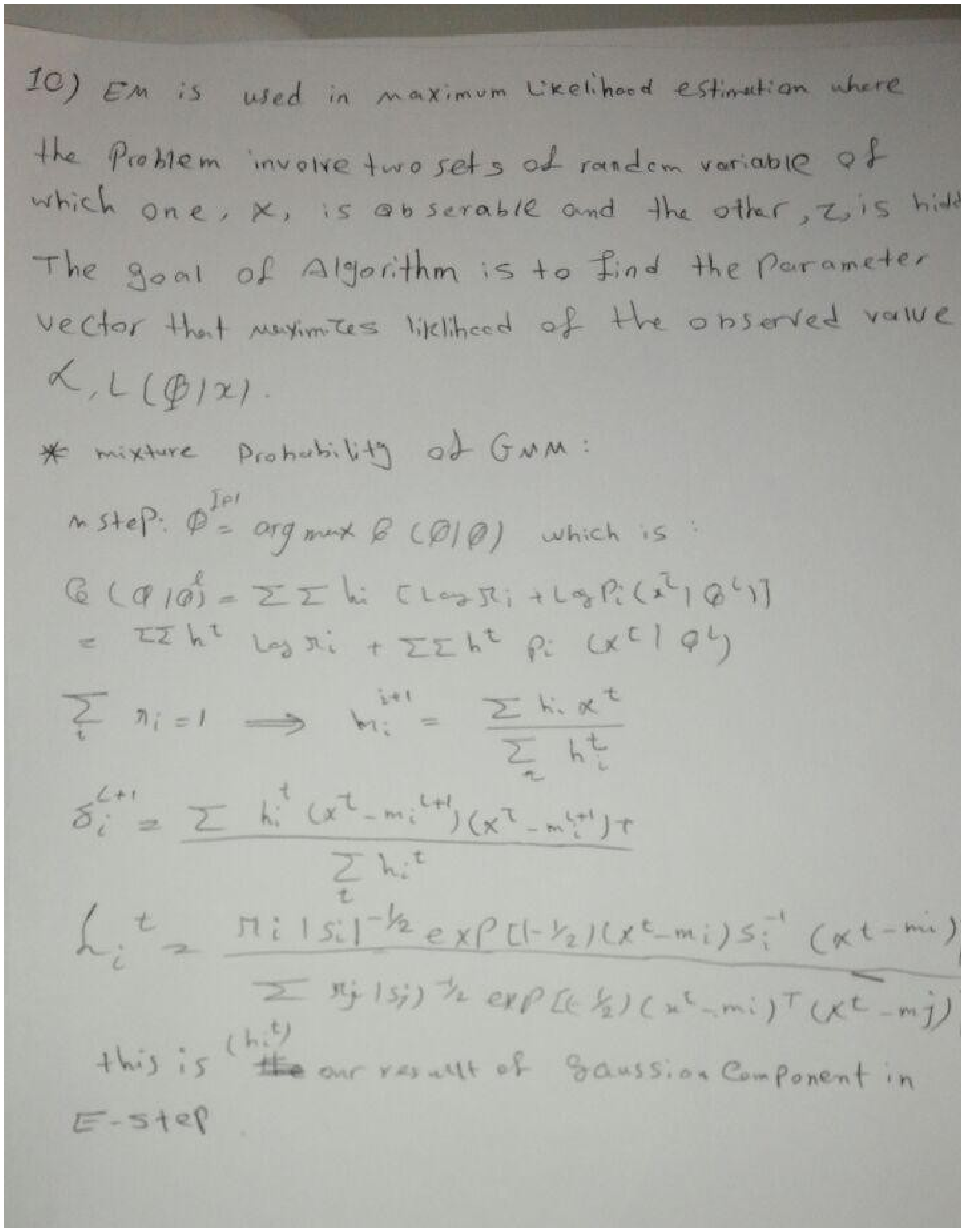
	X1	X2	X3	X4
X1	[illegible]	1	2	1
X2	[illegible]	1	0	1
X3	2	[illegible]	0	1
X4	1	1	1	0

k means where $k = 2$

b) cluster of datasets by k - means $\rightarrow \boxed{k = 2}$ is in the based on the Fig

c) based ~~and~~ on the hierical culustering our fig will be based on ~~above fig~~:
average - Link distance.

Figure 7: question9



10) EM is used in maximum Likelihood estimation where the Problem involve two sets of random variable of which one, x, is absERable and the other, z, is hidd

The goal of Algorithm is to find the Parameter vector that maximizes likelihood of the observed value α, $L(\phi|x)$.

* mixture Probability of GMM:

m step: $\phi^{l+1} = \arg\max \mathcal{Q}(\phi|\phi)$ which is :

$$\mathcal{Q}(\phi|\phi^{l}) = \sum\sum h_i \,[\log \pi_i + \log P_i(x^t|\phi^l)]$$
$$= \sum\sum h^t \log \pi_i + \sum\sum h^t P_i\,(x^t|\phi^l)$$

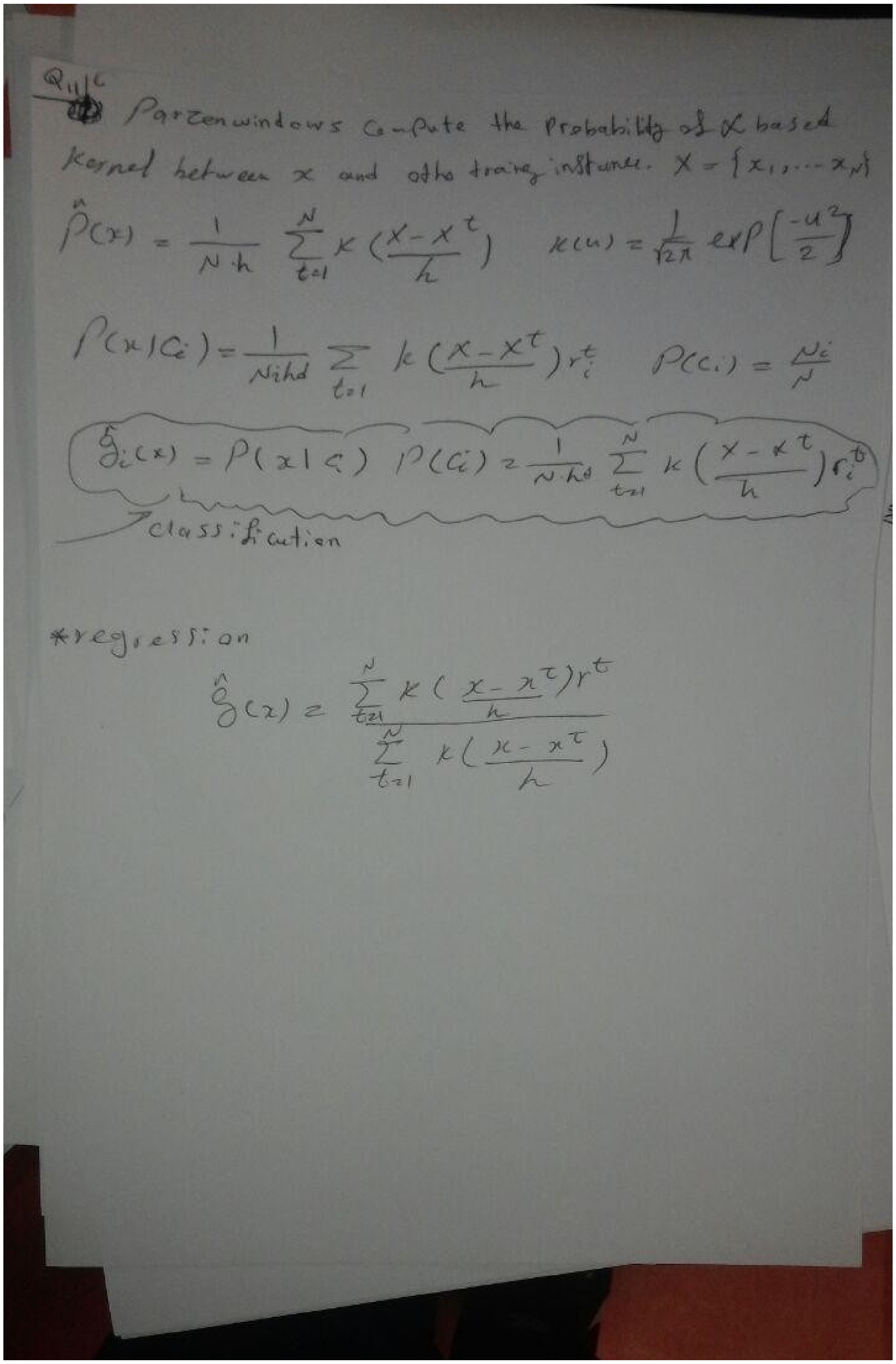
$$\sum_i \pi_i = 1 \implies m_i^{l+1} = \frac{\sum h_i x^t}{\sum_t h_i^t}$$

$$S_i^{l+1} = \frac{\sum h_i^t (x^t - m_i^{l+1})(x^t - m_i^{l+1})^T}{\sum_t h_i^t}$$

$$h_i^t = \frac{\pi_i |S_i|^{-1/2} \exp[(-1/2)(x^t - m_i) S_i^{-1} (x^t - m_i)}{\sum \pi_j |S_j|^{-1/2} \exp[(-1/2)(x^t - m_i)^T (x^t - m_j)}$$

this is ~~the~~ (h_i^t) our result of gaussian Component in E-step

Figure 8: question10



Q11 c

Parzenwindows Compute the Probability of x based kernel between x and othe traing instance. $X = \{x_1, \dots x_N\}$

$$\hat{P}(x) = \frac{1}{N \cdot h} \sum_{t=1}^{N} K\left(\frac{x - x^t}{h}\right) \qquad K(u) = \frac{1}{\sqrt{2\pi}} \exp\left[\frac{-u^2}{2}\right]$$

$$P(x|C_i) = \frac{1}{N_i h d} \sum_{t=1} k\left(\frac{x - x^t}{h}\right) r_i^t \qquad P(C_i) = \frac{N_i}{N}$$

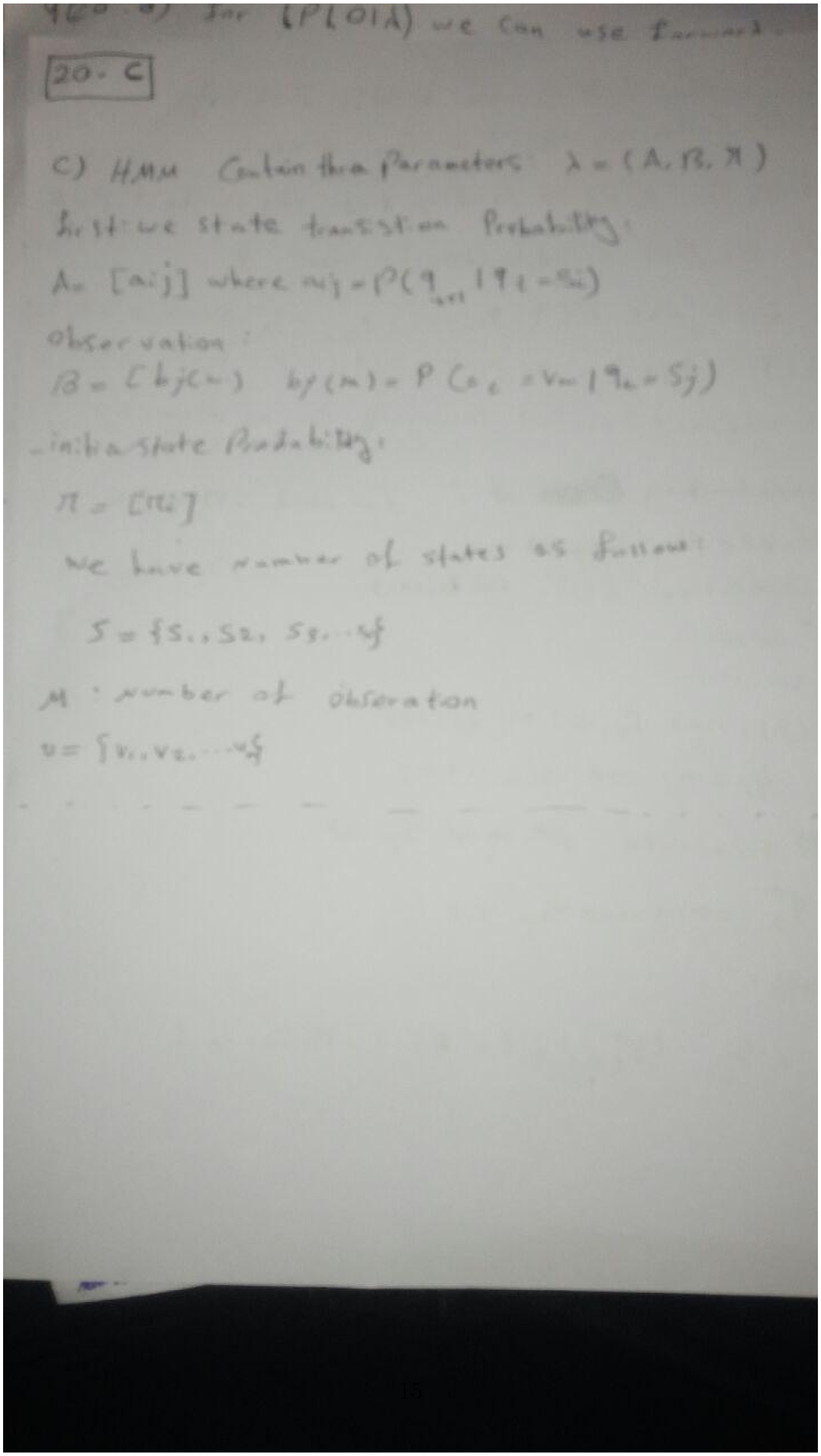
$$\hat{g}_i(x) = P(x|C_i)\, P(C_i) = \frac{1}{N \cdot h d} \sum_{t=1}^{N} k\left(\frac{x - x^t}{h}\right) r_i^t$$

classification

*regression

$$\hat{g}(x) = \frac{\sum_{t=1}^{N} K\left(\frac{x - x^t}{h}\right) r^t}{\sum_{t=1}^{N} K\left(\frac{x - x^t}{h}\right)}$$

Figure 9: question11



[illegible] for $(P(O|\lambda))$ we can use forward

$\boxed{20 \cdot C}$

C) HMM Contain three Parameters $\lambda = (A, B, \pi)$

first we state transistion Probability:

$A = [a_{ij}]$ where $a_{ij} = P(q_{t+1} | q_t = S_i)$

observation:

$B = [b_j(m)]$ $b_j(m) = P(o_t = v_m | q_t = S_j)$

- initia state Probability:

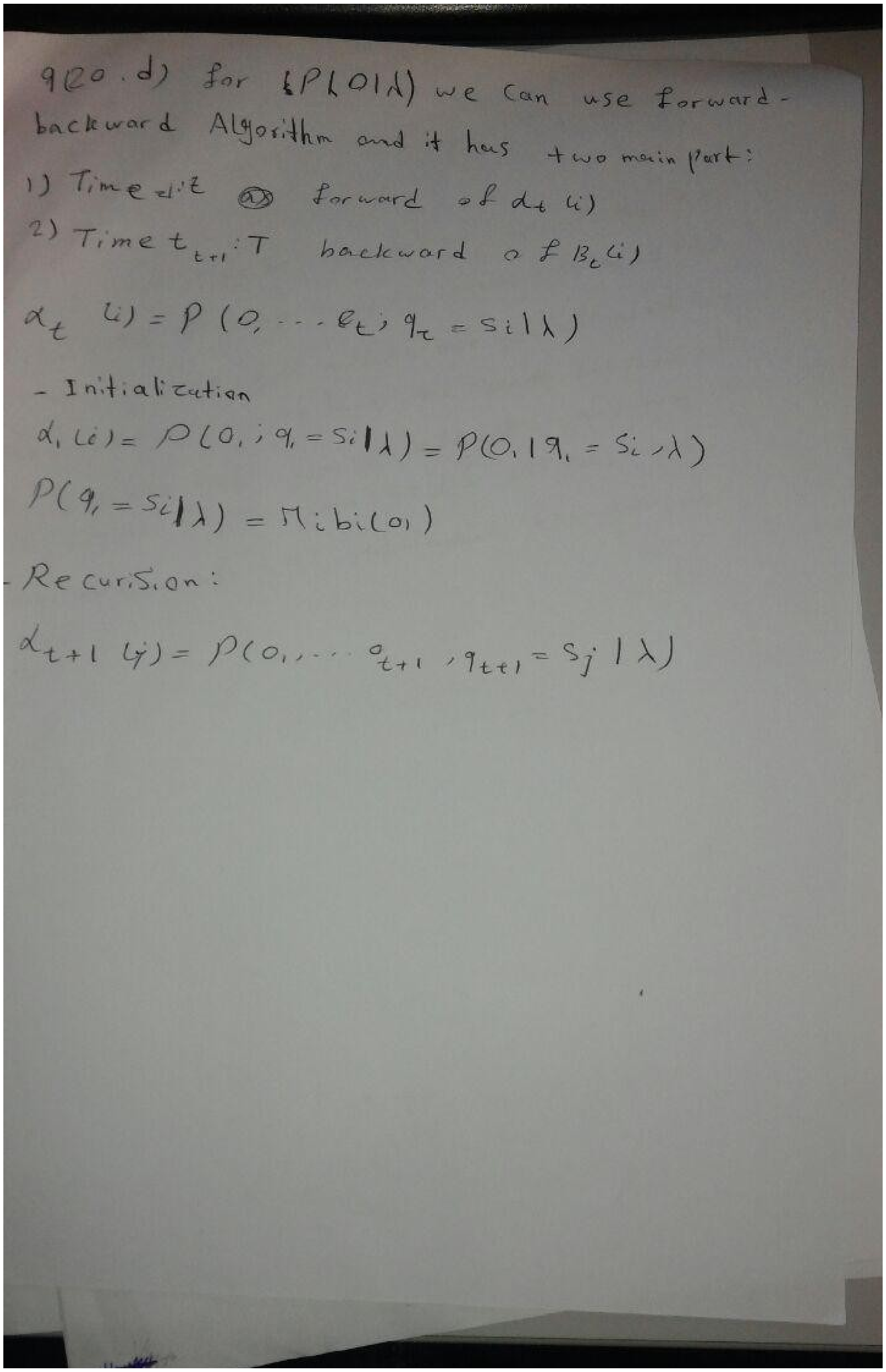
$\pi = [\pi_i]$

we have number of states as follow:

$S = \{S_1, S_2, S_3, \dots N\}$

M: number of observation

$v = \{v_1, v_2, \dots v_M\}$



q20.d) for $P(O|\lambda)$ we can use forward-backward Algorithm and it has two main part:

1) Time $1:t$ (a) forward of $\alpha_t(i)$

2) Time $t_{t+1}:T$ backward of $\beta_t(i)$

$$\alpha_t(i) = P(O_1 \ldots O_t, q_t = S_i | \lambda)$$

- Initialization

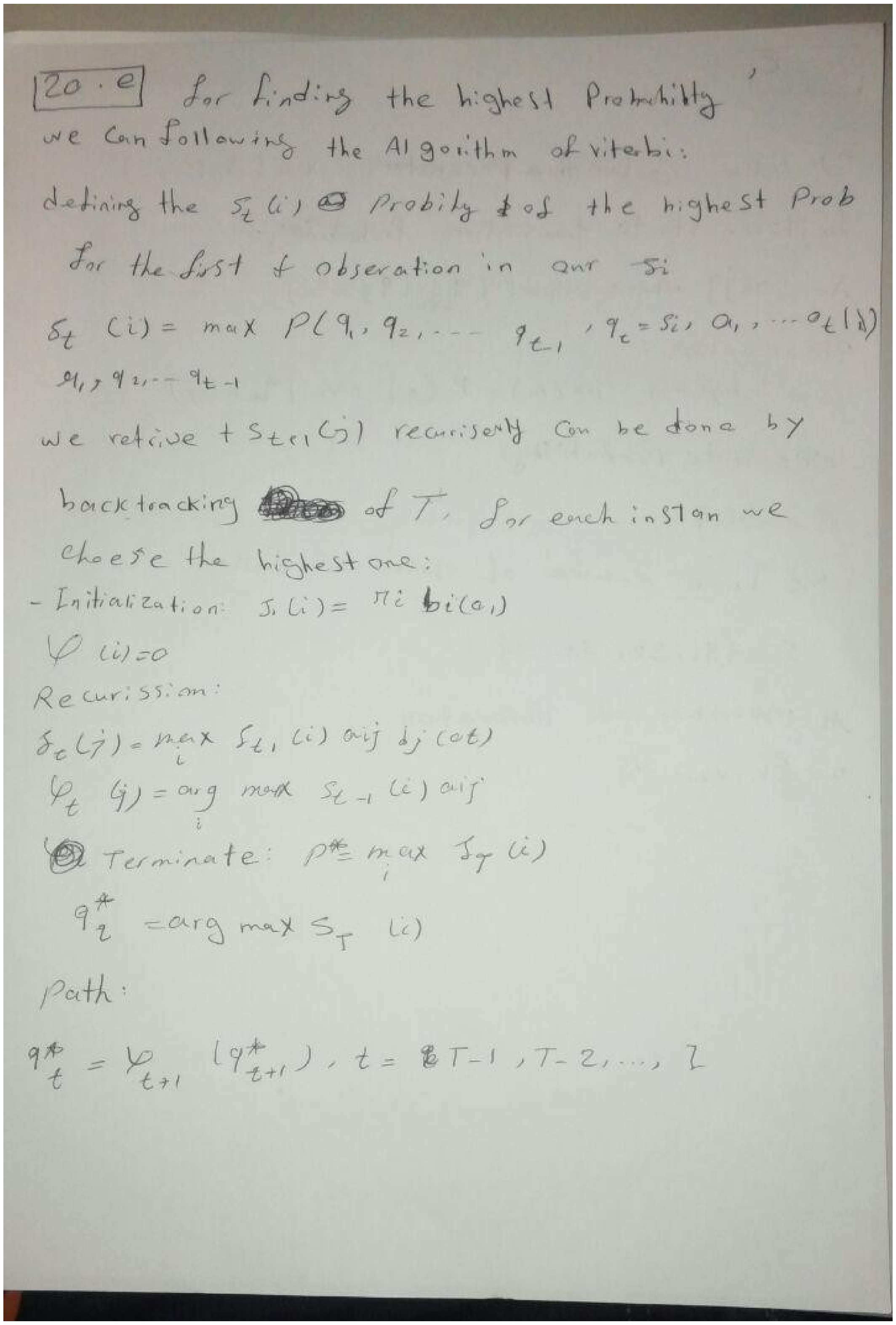
$$\alpha_1(i) = P(O_1, q_1 = S_i | \lambda) = P(O_1 | q_1 = S_i, \lambda)$$

$$P(q_1 = S_i | \lambda) = \pi_i b_i(O_1)$$

- Recursion:

$$\alpha_{t+1}(j) = P(O_1, \ldots O_{t+1}, q_{t+1} = S_j | \lambda)$$

Figure 11: question20d



$\boxed{20.e}$ for finding the highest Probability, we can following the Algorithm of viterbi:

defining the $\delta_t(i)$ Probility of the highest Prob for the first t observation in our S_i

$$\delta_t(i) = \max_{q_1, q_2, \ldots q_{t-1}} P(q_1, q_2, \ldots q_{t-1}, q_t = S_i, O_1, \ldots O_t \mid \lambda)$$

we retrive $\delta_{t+1}(j)$ recurisevly can be done by backtracking of T. for each instan we choose the highest one:

- Initialization: $\delta_1(i) = \pi_i b_i(O_1)$

$\psi(i) = 0$

Recurission:

$\delta_t(j) = \max_i \delta_{t-1}(i)\, a_{ij}\, b_j(O_t)$

$\psi_t(j) = \arg\max_i \delta_{t-1}(i)\, a_{ij}$

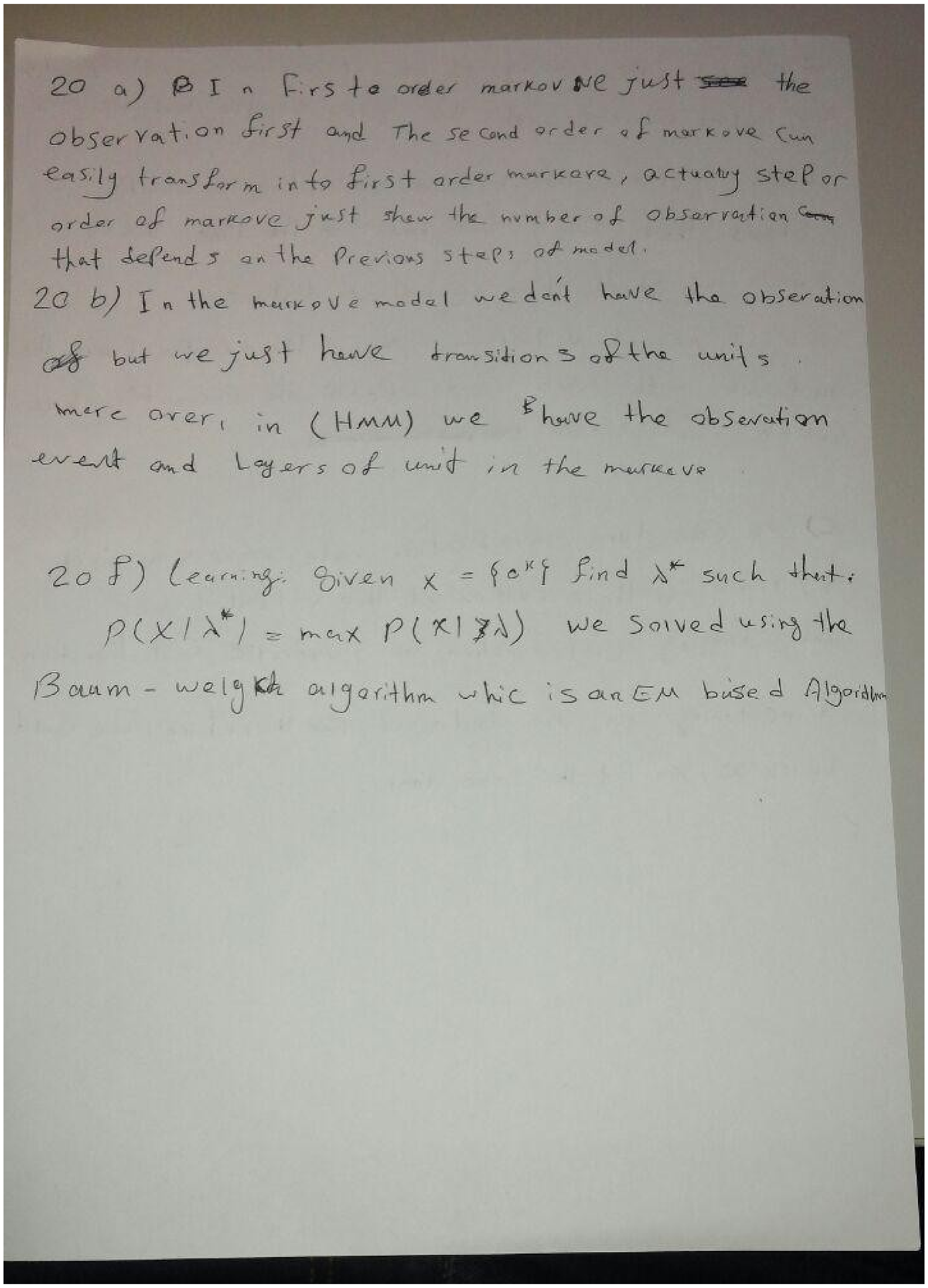
Terminate: $p^* = \max_i \delta_T(i)$

$q^*_T = \arg\max \delta_T(i)$

Path:

$q^*_t = \psi_{t+1}(q^*_{t+1})$, $t = T-1, T-2, \ldots, 1$

Figure 12: question20e



20 a) ~~B~~ In First order markov we just ~~see~~ the observation first and The second order of markove can easily transform into first order markove, actually step or order of markove just show the number of observation ~~can~~ that depends on the Previous steps of model.

20 b) In the markove model we don't have the obseration ~~of~~ but we just have transitions of the units.

more over, in (HMM) we have the observation event and Layers of unit in the markove

20 f) learning: given $X = \{o^k\}$ find λ^* such that:

$P(X|\lambda^*) = \max P(X|\lambda)$ we solved using the Baum - welgh algorithm whic is an EM based Algorithm

Figure 13: question20a,b,f