**HOMEWORK ASSIGNMENT #2**

**B555- MACHINE LEARNING**

**FNU ANIRUDH**

**(aanirudh@umail.iu.edu)**

**Question 1**

**Solution:-**

This situation can be modeled as having data set D = { xi }9i=1 , where each xi = number of accidents in the plant in day i, for 1≤ i ≤9.

We know that i= 79

1. **Maximum Likelihood is given by**

λML = argmaxλ {p(D|λ)}

Where probability of single observation xi givenλ is λxi e-λ

p(xi|λ)= \_\_\_\_\_\_\_

xi !

Since we assume a poisson distribution for number of accidents. From this it follows that the likelihood for a general data set D with n observation is

p(D|λ)= by independence of xi

λxi e-λ

= \_\_\_\_\_\_\_ by assumption of poisson distribution

xi !

λ i e-nλ

= \_\_\_\_\_\_\_\_\_\_\_

This shows that the likelihood function is l (λ)= λ i e-nλ

\_\_\_\_\_\_\_\_\_\_\_

To find λ that maximizes the likelihood, we will first take a logarithm to simplify the calculation, then find its first derivative with respect to λ, and finally equate it with zero to find the maximum. Specifically, we express the log-likelihood ɭ ɭ (D,λ)= ln p(D|λ) as

ɭ ɭ (D, λ) = ln λ ∑i=1 n xi – nλ - ∑i=1 n ln( xi !)

and proceed with the first derivative as

ɗ ɭ ɭ (D, λ)

\_\_\_\_\_\_\_\_ = 1/λ ∑ i=1 n  xi – n

ɗ λ

= 0

Hence we can say that n= 1/λ ∑ i=1 n  xi

Or

1

λML  = \_\_\_\_\_ ∑ i=1 n xi

n

Since xi ϵ N for all i and λ2 >0 . Hence we ignore the degenerate case where xi= 0 for all i

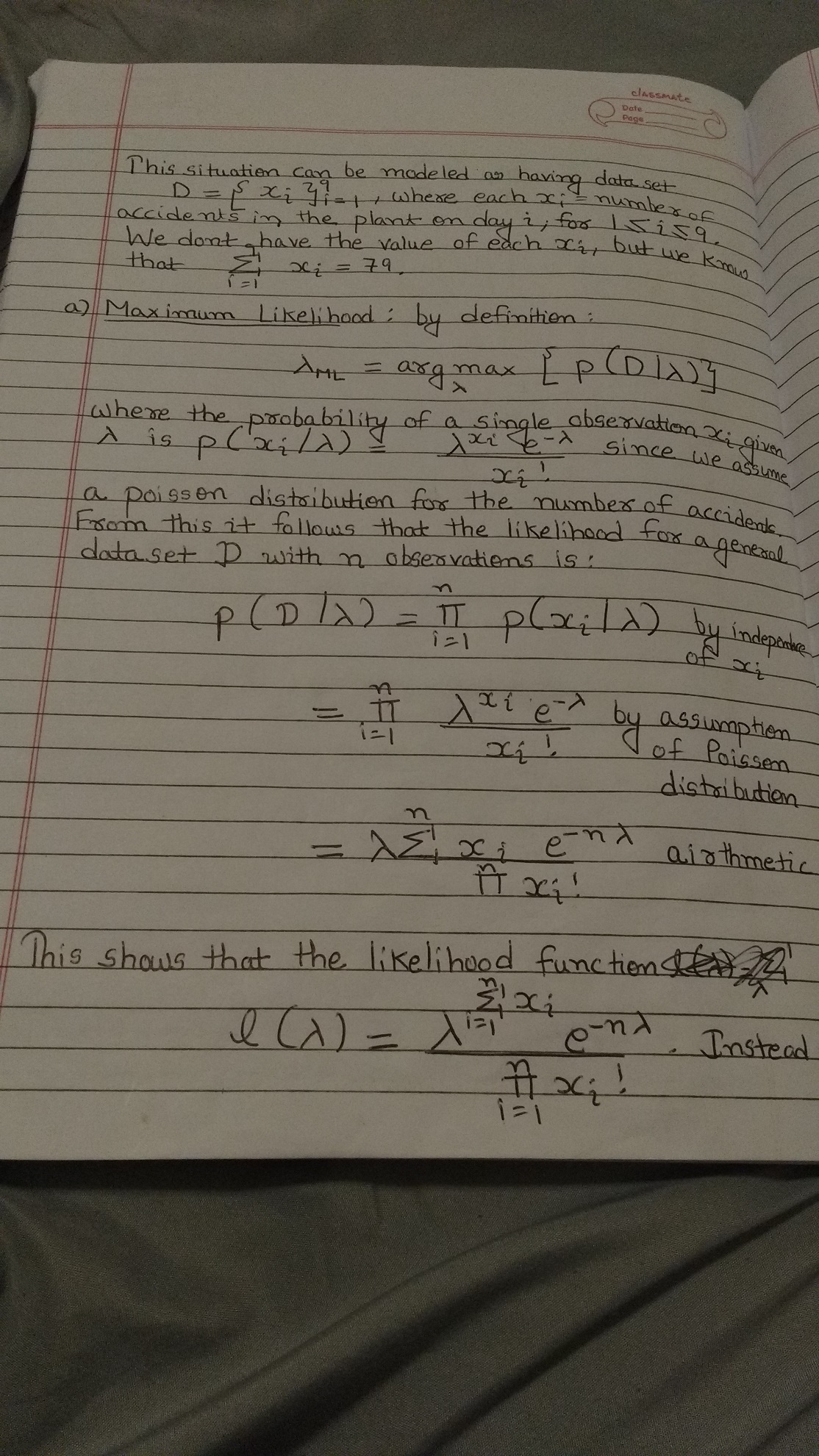
Given that n=9 and ∑ i=1 9 xi = 79,

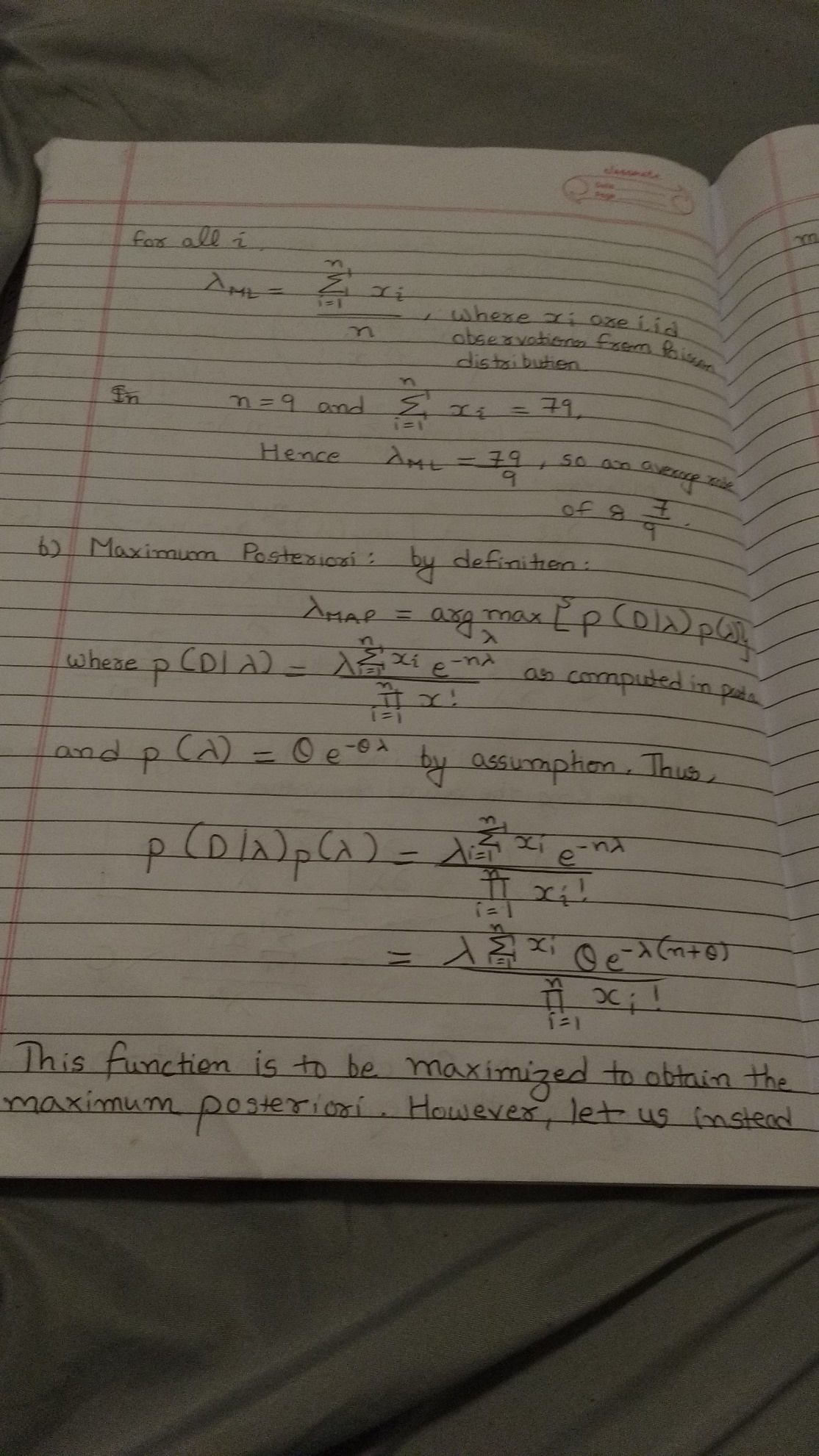
Hence,

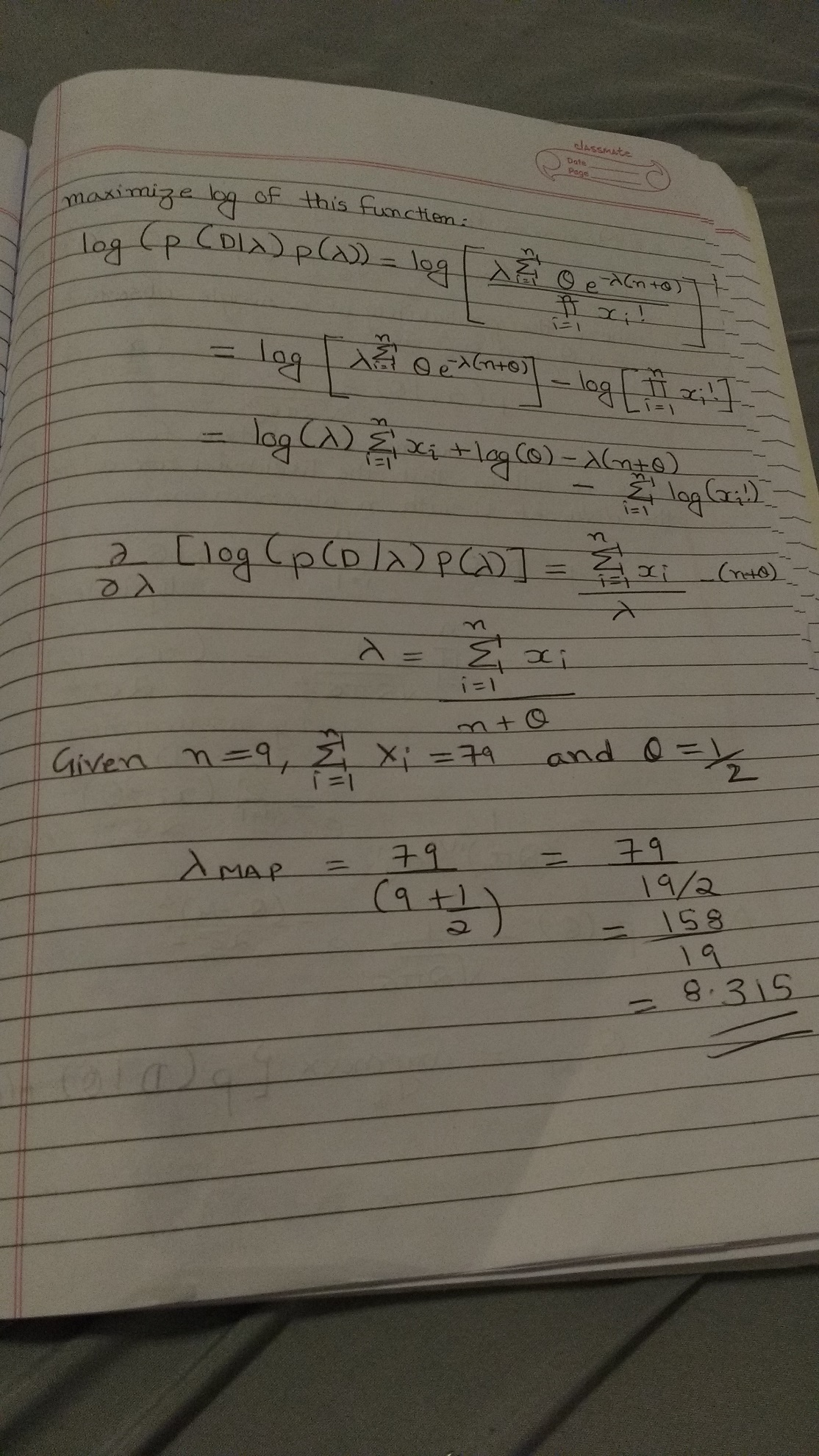
79

λML  = \_\_\_\_\_ = 8.7777

9







**Maximum Posteriori by definition is**

λMAP = argmaxλ {p(D|λ) p(λ)}

Where

λ i e-nλ

p(D|λ) = \_\_\_\_\_\_\_\_\_\_\_ from part a

And p(λ) = ϴ e -ϴλ

p(D|λ) p(λ) = i e-nλ

\_\_\_\_\_\_\_\_\_\_\_

λ ∑i=1 n xi ϴ e-λ (n+ϴ)

= \_\_\_\_\_\_\_\_\_\_\_\_\_

We can maximize this function by taking logarithm

[λ ∑i=1 n xi ϴ e-λ (n+ϴ)]

ln {p(D|λ) p(λ)}=ln \_\_\_\_\_\_\_\_\_\_\_\_\_

= ln [λ ∑i=1 n xi ϴ e-λ (n+ϴ)] – ln

= ln (λ ) ∑i=1 n xi + ln (ϴ) – λ (n+ϴ) - ∑i=1 n ln (xi

Differentiating and equating to zero we get

ɗ ln [p(D|λ) p(λ)] = ∑i=1 n xi

\_\_\_ \_\_\_\_\_\_ - (n+ϴ) = 0

ɗλ λ

∑i=1 n xi

λ = \_\_\_\_\_\_

n+ϴ

Since xi ϵ N for all i and λ2 >0 . Hence we ignore the degenerate case where xi= 0 for all i.

∑i=1 n xi

λMAP = \_\_\_\_\_\_

n+ϴ

given that n=9 and ∑i=1 n xi = 79 and ϴ= ½

Hence

79 79 x 2 158

λMAP = \_\_\_\_\_\_\_\_ = \_\_\_\_\_\_\_ = \_\_\_\_\_\_\_\_\_

(9+1/2) 19 19

= 8.315

1. We know that

∑i=1 n xi

λ = \_\_\_\_\_\_

n+ϴ

λ (n + ϴ) = ∑i=1 n xi

Given that ∑i=1 n xi < 4n

n + ϴ < 4n

Or

ϴ < 4 - λ

**2.**

1. **Maximum Posteriori:**

ϴMAP = argmaxϴ {p(D|ϴ) p(ϴ)}

Where the probability of a single observation xi given ϴ and σ02  is

1

P (xi |ϴ)= \_\_\_\_\_\_ e –( xi - ϴ) 2 / 2 σ02

\_\_\_\_\_\_\_\_

√ 2π σ02

Since we assume a normal distribution for Xi, From this it follows that the likelihood for a general data set D with n observation is

p(D|ϴ)= by independence of xi

1

= nΠi=1 \_\_\_\_\_\_ e –( xi - ϴ) 2 / 2σ02 (By Assumption of normal distribution)

\_\_\_\_\_\_\_\_

√ 2π σ02

1 e – (n∑ i=1 ( xi - ϴ) 2 / 2 σ02 )

= \_\_\_\_\_\_\_\_\_\_

(2π)n/2 σ0n

1

Also p(ϴ)= \_\_\_\_\_ e –( ϴ - ϻ) 2 / 2 σ2

ϴMAP = argmaxϴ {p(D|ϴ) p(ϴ)}

1 e – (n∑ i=1 ( xi - ϴ) 2 / 2 σ02 ) 1

=argmaxϴ \_\_\_\_\_\_\_\_  \_\_\_\_\_ e –( ϴ - ϻ) 2 / 2 σ2

(2π)n/2 σ0n

1 e – (n∑ i=1 ( xi - ϴ) 2 / 2 σ02 ) – (ϴ - ϻ)2 / 2 σ2

=argmaxϴ \_\_\_\_\_\_\_\_

(2π) (n+1)/2 σ0n σ

As usual, let us take the log

ln (p(D|ϴ) p(ϴ)) = ln (1/ (2π) (n+1)/2 σ0n σ) - n∑ i=1 ( xi - ϴ) 2 / 2 σ02 - (ϴ - ϻ)2 / 2 σ2

Maximize by differentiating

ɗ [ln (p(D|ϴ) p(ϴ)] ɗ

\_\_\_\_ = \_\_ [ln (1/ (2π) (n+1)/2 σ0n σ) - n∑ i=1 ( xi - ϴ) 2 / 2 σ02 - (ϴ - ϻ)2 / 2 σ2]

ɗϴ ɗϴ

n∑i=1 (xi - ϴ) (ϴ - ϻ)

= \_\_\_\_\_\_\_\_\_\_ \_\_ \_\_\_\_\_\_\_\_

σ02  σ2

(n∑i=1 xi)- nϴ ϴ - ϻ

= \_\_\_\_\_\_\_\_\_\_ \_\_ \_\_\_\_\_\_\_

σ02 σ2

σ2 (n∑i=1 xi) - σ2 nϴ - σ02 ϴ - σ02 ϻ

= \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

σ2 σ02

Equating Left hand side to zero, we get

0 = σ2 (n∑i=1 xi) - σ2 nϴ - σ02 ϴ - σ02 ϻ

0 = -ϴ (σ2 n + σ02) + σ2 (n∑i=1 xi) + σ02 ϻ

σ2 (n∑i=1 xi) + σ02 ϻ

ϴ = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

σ2 n + σ02

ɗ2  ɗ (n∑i=1 xi) - nϴ ϴ - ϻ

\_\_\_ [ ln p(D|ϴ) p(ϴ))] = \_\_\_\_\_ [ \_\_\_\_\_\_\_\_\_\_\_\_\_ \_ \_\_\_\_\_\_ ]

ɗϴ2 ɗϴ σ02 σ2

n 1

= - \_\_\_ \_\_ \_\_\_\_ < 0

σ02 σ2

Since n>0 and σ02 σ2 >0

σ2 (n∑i=1 xi) - σ02 ϻ

ϴMAP = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

σ2n - σ02

ϻ n∑i=1 xi

\_\_ - \_\_\_\_\_

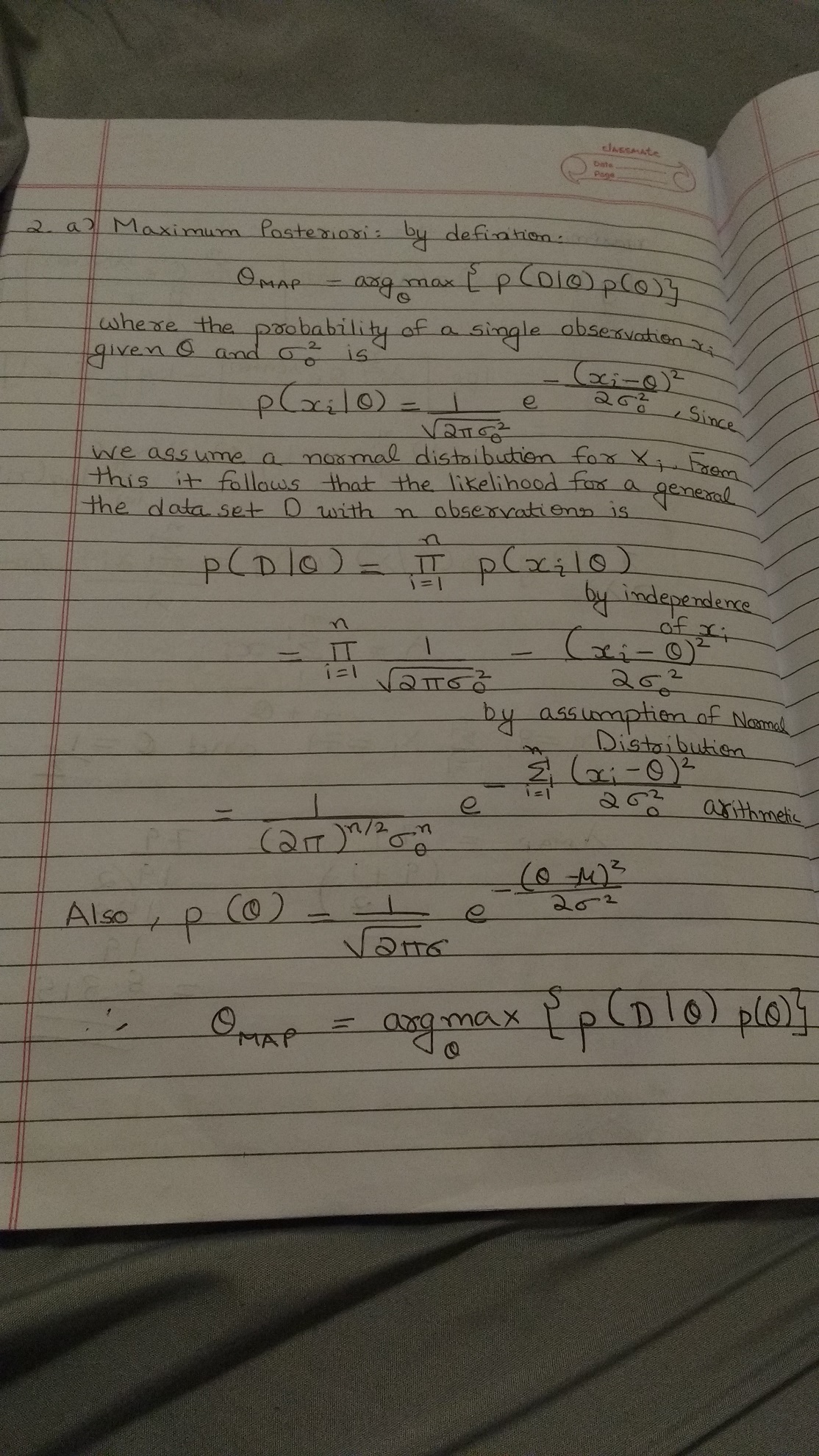
σ2 σ02

ϴMAP = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

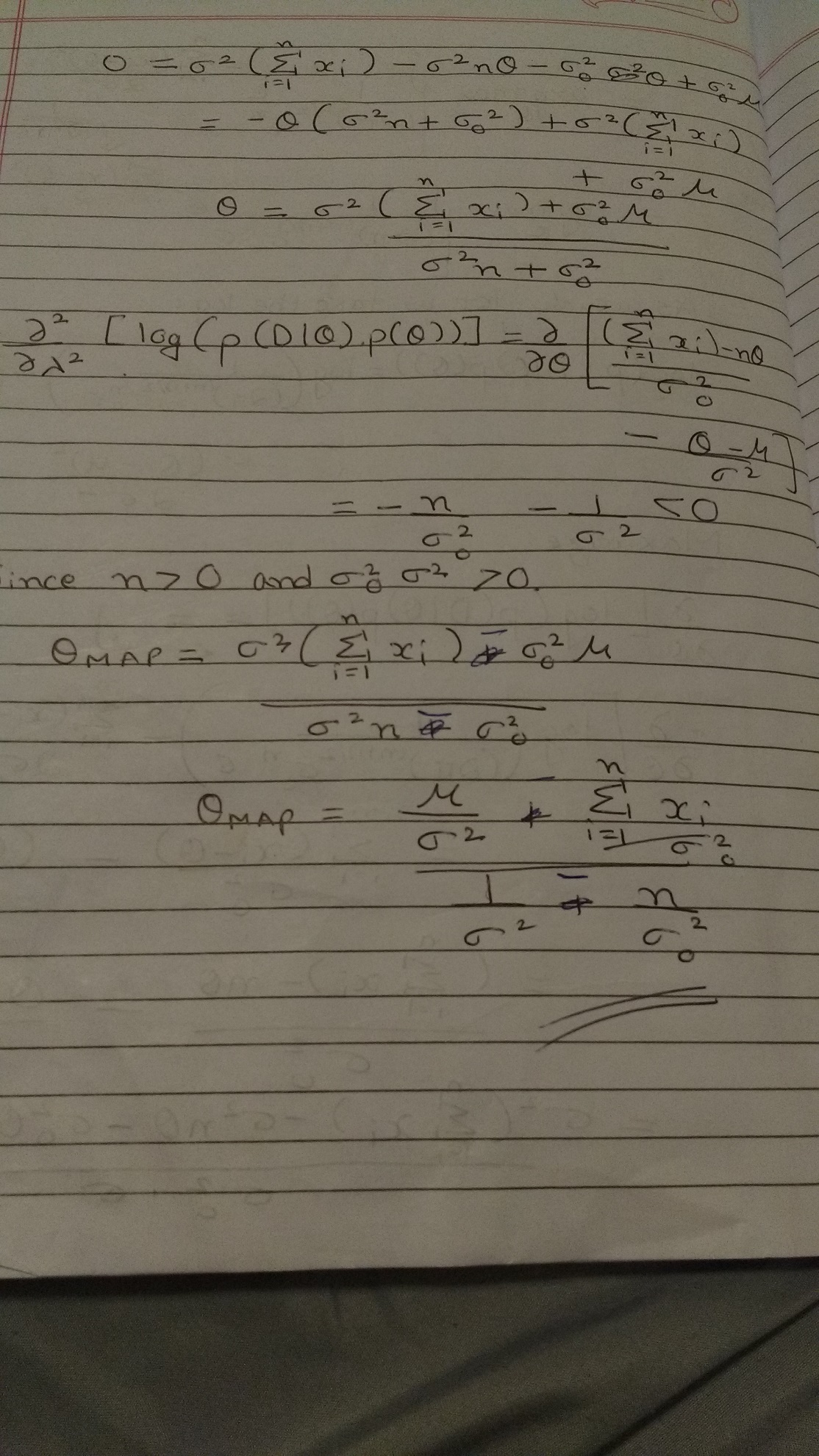
1. n

\_\_\_\_ - \_\_\_\_\_

σ2 σ02







**b)**

1

p(x)= \_\_\_ exp (- |x - ϻ |/b)

2b

We know that

ϴMAP = argmaxϴ {p(D|ϴ) p(ϴ)}

1

P(ϴ)= \_\_ exp (- |ϴ - ϻ | /b )

2b

solving for MAP, by taking log and maximizing we get

Since μ = 0

As ϴ cannot be definite, we cannot have closed form solution hence we can solve this by using gradient descent.

**c)**

This is similar to part a) of the question except need to do in matrix, hence

ɗ -1 ɗ

\_\_ = \_\_ \_\_\_ [ σ2 ϴTϴ + ∑ ( XT X - XTϴ) – (ϴT X + ϴTϴ)]

ɗϴ 2 ɗϴ

= (-1/2) [ 2 σ2 ϴ + ∑ (0 – Xi- Xi + 2ϴ)]

= (-1/2) [2 σ2 ϴ + 2 ∑ Xi + 2nϴ]

= ∑ Xi – (n + σ2) ϴ

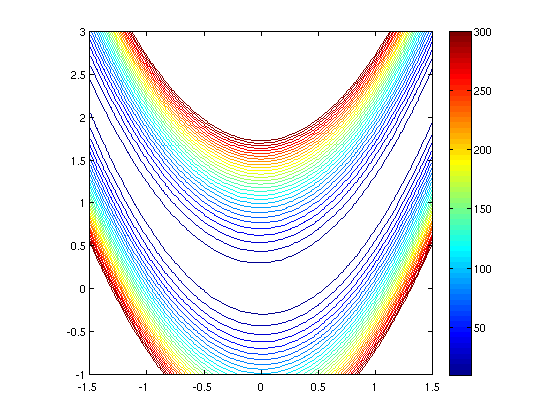
∑ Xi

ϴMAP = \_\_\_\_\_\_\_\_

(n + σ2)

**3.**

Contour plot to expression is given below and is called as rosenbrock function.



**Gradient Descent**

**Case i) (1.2, 1.2)**

1. Result too large is displayed on execution if start with learning rate =1 hence we need to reduce our learning rate. (failed to work even when I tried 0.5,0.25)
2. At learning rate=0.0005, I see that f (x) value kept on increasing.
3. At learning rate=0.00025, Gradient descent becomes stuck and follows zig zag pattern.
4. At learning rate=0.0001 and after 337734 iterations I can conclude that minimum value for the function is (1, 1).

**Case ii) (-1.2, 1)**

1. With starting point as (-1.2, 1) and learning rate =0.0001 Gradient descent takes more number of iterations to converge to minimum value (1, 1).

**Newton’ s Method**

**Case i) (1.2, 1.2)**

Current function value: 0.000000

Iterations: 12

Function evaluations: 16

Gradient evaluations: 27

Hessian evaluations: 12

[ 0.99999998 0.99999997]

It took 12 iterations to converge to minimum value (1, 1) when our starting point is (1.2, 1.2) using Hessian in newton’s method.

**Case ii) (-1.2, 1)**

Current function value: 0.000000

Iterations: 85

Function evaluations: 107

Gradient evaluations: 191

Hessian evaluations: 85

[ 1. 1.]

It took 85 iterations to converge to minimum value (1, 1) when our starting point is (-1.2, 1) using Hessian in newton’s method.

**Note: -** Please refer attached programs to verify veracity of my solution.

**4.**

1. Once features are increased to 17 or greater, we get an error related to overfitting “**line 90, in \_raise\_linalgerror\_singular raise LinAlgError("Singular matrix")**

**numpy.linalg.linalg.LinAlgError: Singular matrix** “

This occurs when our (X XT)-1 is non-invertible or singular matrix which may be caused by identical, similar, or linearly dependent features. Determinant value for singular matrix is 0.

From section 4.5.1 in notes

This problem can be solved by singular value decomposition which can be solved by using X = UΣVT.

**b)**

In order to implement Ridge regression, we need to add λ and ||w|| in our algorithm.

After implementing ridge regression and after careful comparison I observe following

* Ridge regression provides better solution compared to FS Linear regression.
* Accuracy for Ridge regression is lower compared to FS Linear regression.
* Accuracy decreases after implementing regularization and choosing high value of lambda.
* Extremely high values of lambda can increase accuracy.

**c)**

There are different ways to do feature selection:-

1. **Filter method**:- perform feature selection as a preprocessing step, independently of the learning algorithm used for model construction. An example of such a mechanism is variable ranking, using e.g. correlation coefficients between each feature and the dependent variable. Another filter approach selects features based on a linear mode and then constructs a non-linear model using the selected features.
2. **Wrapper method:-** are characterized as being a subset selection approach. The main idea of wrapper methods is to assess subsets of variables according to their usefulness to a given learning algorithm. Here, the learning algorithm is treated as a black box, and the best subset of features is determined according to the performance of the particular algorithm applied to build a regression mode.
3. **Embedded methods: -** incorporate feature selection as part of the training process, i.e., feature selection is done when building the predictive model. Such mechanisms usually involve changes in the objective function of the applied learning algorithm and therefore are commonly associated to a specific predictor. Examples of embedded methods are decision trees.

1. **I implemented stochastic gradient descent in the program:-**

self.weights = np.zeros(rows)

        for i in range(columns):

         self.weights = self.weights - np.dot(alpha,(np.dot(Xless.T,np.dot(Xless, self.weights) - ytrain)))

         print self.weights



In stochastic approximation, we typically approximate the gradient with one sample.

**e)**

Both ordinary least square and logistic regression are derived from Generalized linear models.

In its simplest form, a linear model specifies the (linear) relationship between a dependent (or response) variable *Y*, and a set of predictor variables, the *X*'s, so that

Y = b0 + b1X1 + b2X2 + ... + bkXk

In this equation *b0* is the regression coefficient for the intercept and the *bi* values are the regression coefficients (for variables 1 through *k*) computed from the data.

**List of References**

1. Pattern Recognition and Machine Learning :- Christopher M. Bishop
2. Machine Learning Notes:- Predrag Radivojac and Martha White
3. <http://www.onmyphd.com/?p=gradient.descent>
4. <http://code.activestate.com/recipes/576762-newton-raphson-root-finding/>
5. <http://people.duke.edu/~ccc14/sta-663/BlackBoxOptimization.html>
6. <http://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html>
7. <http://www.dcs.uchile.cl/images/dcs/publicaciones/jmirandap/Internacional/A%20Hybrid%20Forecasting%20Methodology%20using%20Feature%20Selection%20and%20Support%20Vector%20Regression.pdf>
8. Parts of question 4 discussed with Rohit Nair, Sohail Jain.