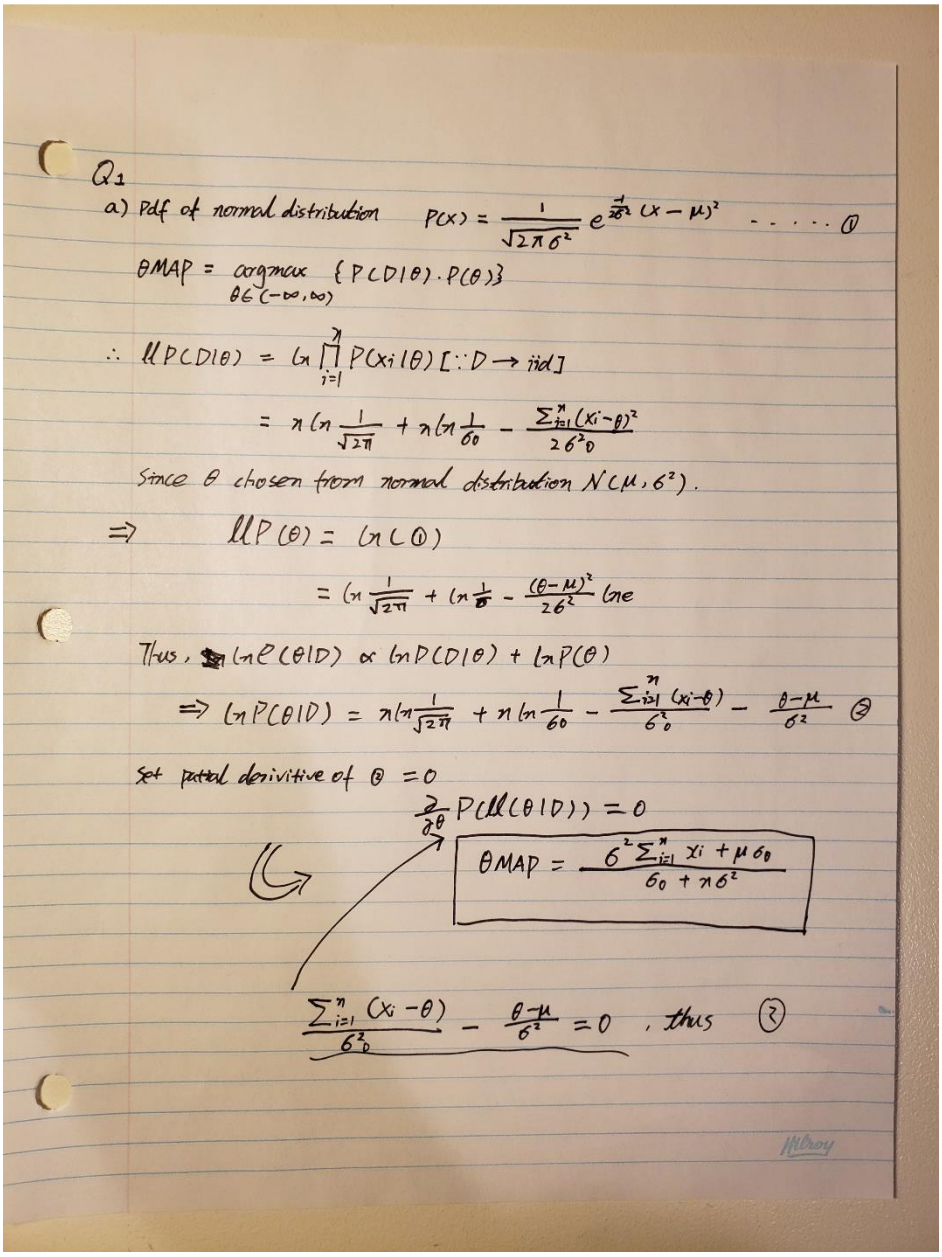
Question1:



Q1

a) Pdf of normal distribution $P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{1}{2\sigma^2}(x-\mu)^2}$ ..... ①

$\theta MAP = \underset{\theta \in (-\infty, \infty)}{\operatorname{argmax}} \{P(D|\theta) \cdot P(\theta)\}$

$\therefore \ln P(D|\theta) = \ln \prod_{i=1}^{n} P(x_i|\theta)$ [$\because D \rightarrow$ iid]

$= n \ln \frac{1}{\sqrt{2\pi}} + n \ln \frac{1}{\sigma_0} - \frac{\sum_{i=1}^{n}(x_i-\theta)^2}{2\sigma_0^2}$

Since θ chosen from normal distribution $N(\mu, \sigma^2)$.

$\Rightarrow \ln P(\theta) = \ln$ ①

$= \ln \frac{1}{\sqrt{2\pi}} + \ln \frac{1}{\sigma} - \frac{(\theta-\mu)^2}{2\sigma^2} \ln e$

Thus, $\ln P(\theta|D) \propto \ln P(D|\theta) + \ln P(\theta)$

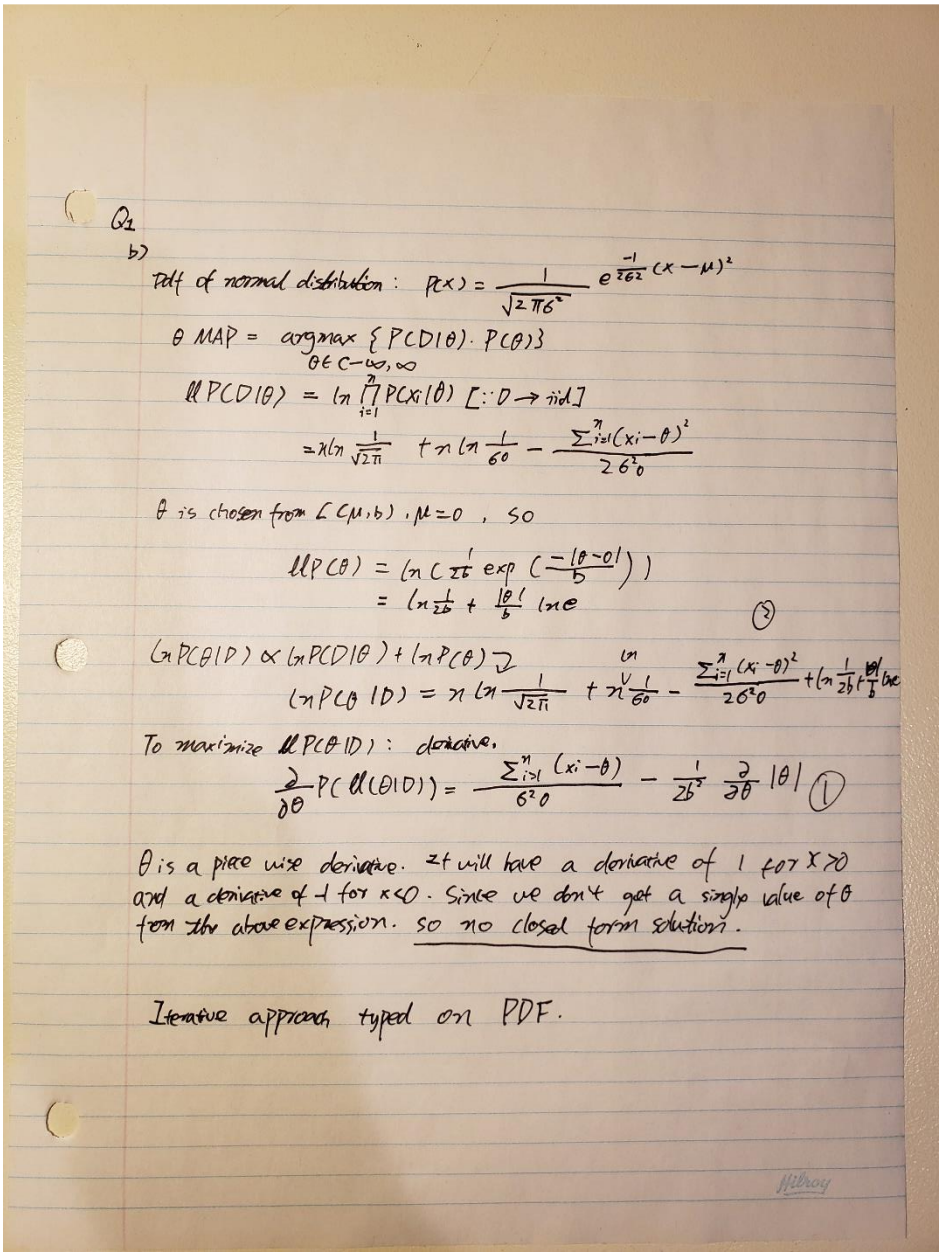
$\Rightarrow \ln P(\theta|D) = n \ln \frac{1}{\sqrt{2\pi}} + n \ln \frac{1}{\sigma_0} - \frac{\sum_{i=1}^{n}(x_i-\theta)}{\sigma_0^2} - \frac{\theta-\mu}{\sigma^2}$ ②

Set partial derivitive of $\theta = 0$

$\frac{\partial}{\partial\theta} P(\ln(\theta|D)) = 0$

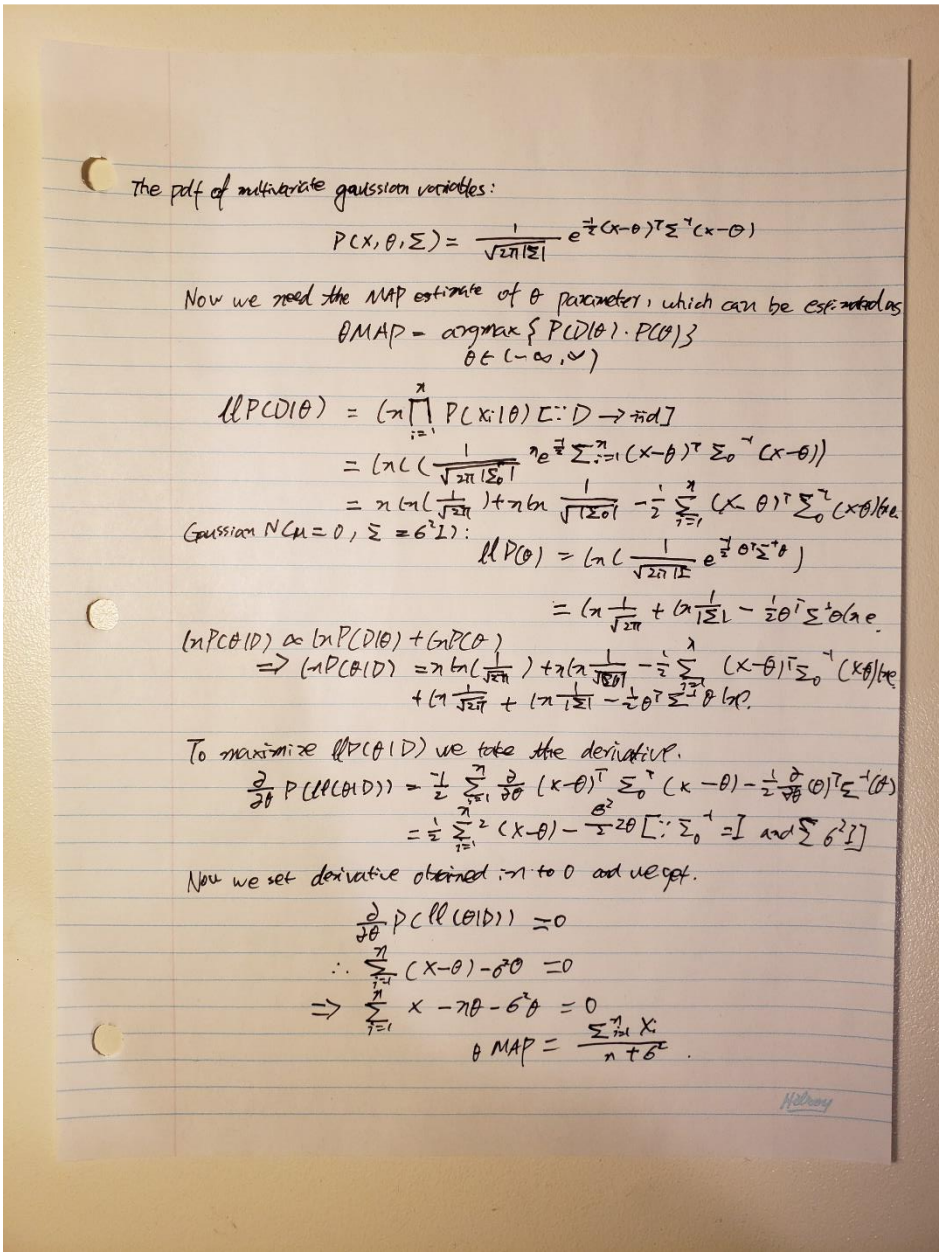
$\theta MAP = \frac{\sigma^2 \sum_{i=1}^{n} x_i + \mu\sigma_0}{\sigma_0 + n\sigma^2}$

$\frac{\sum_{i=1}^{n}(x_i-\theta)}{\sigma_0^2} - \frac{\theta-\mu}{\sigma^2} = 0$, thus ③



b) Iterative approach typed on PDF. By using a gradient descent approach we can solve this problem. We can start with some random value for theta >1 and take the derivative of theta in Equation1. After that, we can get the gradient of the MAP function. Our goal is to maximize the function. To achieve that we need to keep decreasing theta by small weight and see if the value of the function in equation 2 increases. Once the gradient for this function approaching 0 and the function reaches it's maxing value. We repeat this process till the time two consecutive theta values become almost equal and the gradient of the function becomes 0. Similarly, we can repeat the same process for a

value of theta<0 and repeat the same process until the time the function converges. So, I think there might be 2 values of theta for which the function converges.



Question2:

a) By increasing the number of selected features to 385, we raised an error of "numpy.linalg.LinAlgError: Singular matrix". The error might be due to the issue that we used the formula $\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y}$ in our implementation. However, some matrices from the dataset might have a determinant of zero because of identicalness, similarity,

or they are linearly dependent features. This implies that it is a singular matrix; thus, the inverse of the matrix does not exist.
To remedy this problem, we can use singular value decomposition, or just adding the regularization weight decay.

c) The model trained by Ridge Regression have better result when comparing to the FSLinearRegression from (a). We had average error for FSLR(385features) around 11.53 +-0.37 and average error for RLR(lambda = 0.01) around 8.51 +- 0.19. The reason for RLR error is lower than FSLR error might be the idea of Ridge Regression regularized the model, so the model generally performs better with the test data (help deal with overfitting). Another assumption might be the default lambda of FSLR given code= 0.5(to avoid singular matrix error) is just way too big.

f) Stochastic gradient descent (SGD) typically reaches convergence much faster than batch gradient descent since it updates weight more frequently. Unlike the batch gradient descent, which computes the gradient using the whole dataset, because the SGD, also known as incremental gradient descent, tries to find minimums or maximums by iteration from a single randomly picked training example, the error is typically noisier than in gradient descent.

In terms of the number of times, the entire training set is processed. SGD processed the entire training set for epochs times and epochs=1000 for my implementation, according to Algorithm 3 from Note section 6.3. BGD processed the entire training set for epochs * maxIteration times and epochs=1000, maxIteration = 10e5 for my implementation, according to Algorithm 2 and Algorithm 1 from Note section 6.3.

Above is my runtimes result for the SGD and BGD. Therefore error versus epochs: SGD(9.27/1000) and BGD (8.93/1000). For error versus runtime: SGD(9.27/68.81) and BGD (8.93/384.90).