學號：b05602052 系級：電機四 姓名：舒泓諭

1. (0.5%) 請比較你實作的 generative model、logistic regression 的準確率，何者較佳?

	Public	Private
generative model	0.84606	0.84399
logistic regression	0.85380	0.84964

從以上可以看出 logistic regression 不論在 public score，還是 private score，都來的比 generative model 來的更高。符合老師在上課提到大部分的狀況下 logistic regression 都會來的比 generative model 更好。然而這取決於 training data，是否符合當初 generative 的機率分佈。如果 generative 的 training data 符合當初假設的機率分布，那麼在這個條件下，generative 效果通常會比 logistic regression 來得更好。

2. (0.5%) 請實作特徵標準化(feature normalization)並討論其對於你的模型準確率的影響

logistic regression:

	Public	Private
Non-feature normalization	0.84815	0.84277
feature normalization	0.85380	0.84964

Generative model:

	Public	Private
Non-feature normalization	0.84373	0.84289
feature normalization	0.84606	0.84399

有做 feature normalization 的話不論是 logistic regression 還是 Generative model，public score 和 private score 都會上升。然而如果不做 feature normalization 的話 Private score 連 simple baseline 都過不了，可見 feature normalization 的重要性。

	Change of Public	Change of Private
logistic regression	5.65*e-3	6.87*e-3
Generative model	2.33*e-3	1.1*e-3

由上表可以發現 feature normalization，對於 logistic regression 的影響會比 Generative model 來的更高，主要原因在於 logistic regression，再找 w , b 值時有進行 gradient descent，然而 Generative model 只是純粹從機率分佈去推估 w , b。所以可以看出符合老師在上課提到的 feature normalization，可以有效解決當 feature range 不一樣時，gradient 會前後 oscillate 的問題。

3. (1%) 請說明你實作的 best model，其訓練方式和準確率為何?

Data preprocessing:

因為經過 one hot encoding 之後，大部分的 feature，都變為 0 和 1，只有 age,fnlwgt, capital_gain,capital_loss,hours_per_week，的數值不是離散的，所以只有在這幾個 feature 進行 normalization。

Model training:

本次使用的 model 只疊了一層 sklearn GradientBoostingClassifier，訓練參數如下：

	parameter
Loss function	logistic regression
learning_rate	0.1
n_estimators(epoch)	100
max_depth	3
min_impurity_split (early stopping)	1*e-7

Predict outcome:

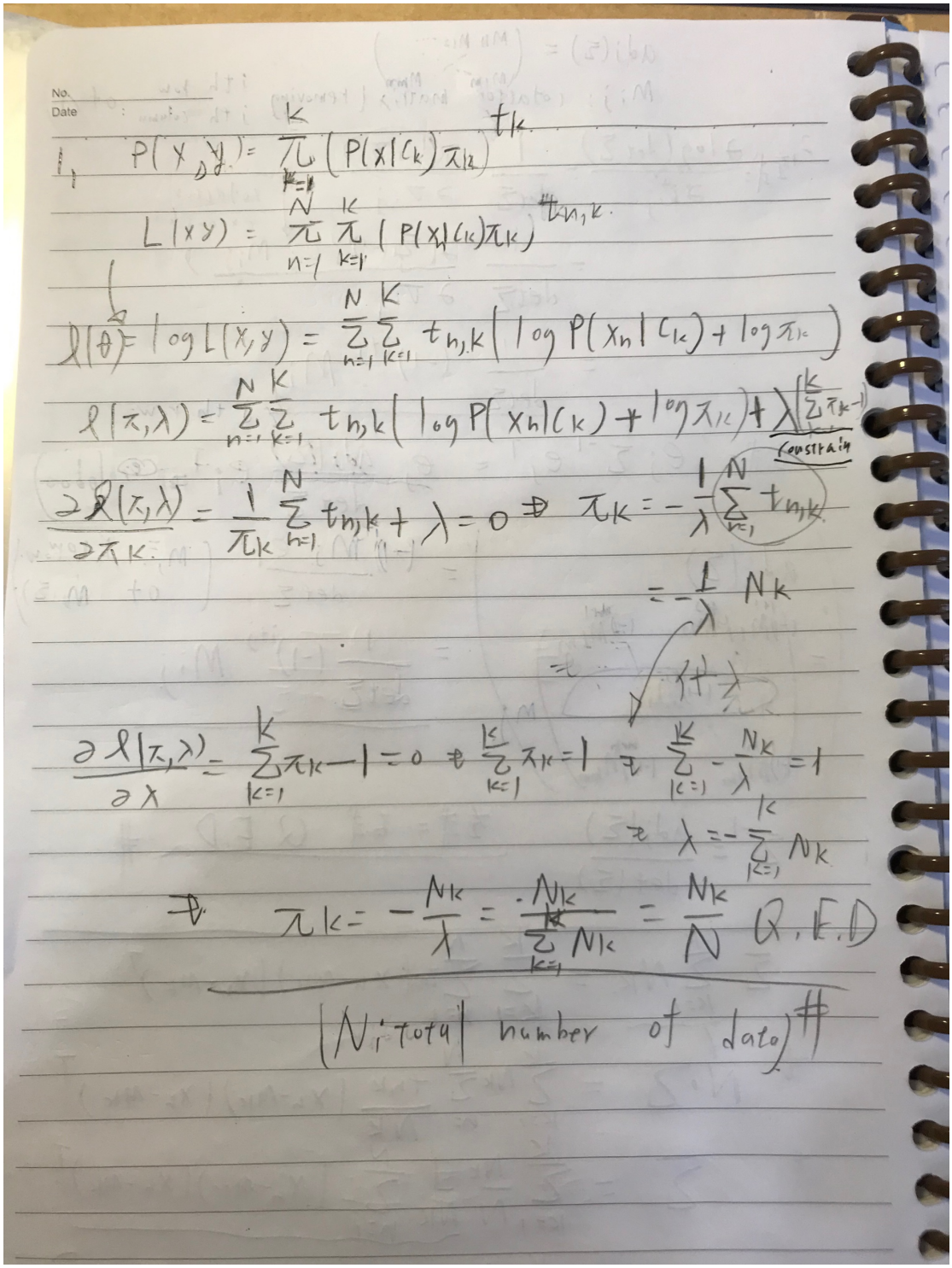
	Training accuracy	Public	Private
generative model	0.841927	0.84606	0.84399
logistic regression	0.849973	0.85380	0.84964
gradient boosting	**0.86689**	**0.86977**	**0.86549**

原本的兩個 **model** 在同樣的 **data preprocessing** 的方法下，不管怎麼樣都過不了 **strong baseline**，然而使用 GradientBoosting 的方式後，一下子就過了 strong baseline。

此外，如果從這三種方法進行比較我們可以發現，從 Training accuracy 來看，**gradient boosting**，在 training data fit 的比起另外兩種方式來得更好。然而從 public score 和 private score 來看，**gradient boosting** 也表現的比另外兩種方式來的更好，所以沒有發現 **overfitting** 的現象。

手寫作業：

(1)


No.
Date
1, $P(X,y) = \prod_{k=1}^{K} (P(x|C_k)\pi_k)^{t_k}$
$L(xy) = \prod_{n=1}^{N} \prod_{k=1}^{K} (P(x_n|C_k)\pi_k)^{t_{n,k}}$
$l(\theta) = \log L(x,y) = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{n,k} (\log P(x_n|C_k) + \log \pi_k)$
$l(\pi,\lambda) = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{n,k} (\log P(x_n|C_k) + \log \pi_k) + \lambda (\sum_{k=1}^{K} \pi_k - 1)$
constrain
$\frac{\partial l(\pi,\lambda)}{\partial \pi_k} = \frac{1}{\pi_k} \sum_{n=1}^{N} t_{n,k} + \lambda = 0 \Rightarrow \pi_k = -\frac{1}{\lambda} \sum_{n=1}^{N} t_{n,k}$
$= -\frac{1}{\lambda} N_k$
$\frac{\partial l(\pi,\lambda)}{\partial \lambda} = \sum_{k=1}^{K} \pi_k - 1 = 0 \Rightarrow \sum_{k=1}^{K} \pi_k = 1 \Rightarrow \sum_{k=1}^{K} -\frac{N_k}{\lambda} = 1$
$\Rightarrow \lambda = -\sum_{k=1}^{K} N_k$
$\therefore \pi_k = -\frac{N_k}{\lambda} = \frac{N_k}{\sum_{k=1}^{K} N_k} = \frac{N_k}{N}$ Q.E.D
(N: total number of data) #

(2)

$adj(\Sigma) = \begin{pmatrix} M_{11} & M_{12} & \cdots \\ & & \\ M_{1m} & & M_{mm} \end{pmatrix}$

M_{ij}: cofactor matrix (removing ith No. row, jth Date column of Σ)

左式 $= \dfrac{\partial \log(\det\Sigma)}{\partial v_{ij}} = \dfrac{1}{\det\Sigma} \dfrac{\partial(\det\Sigma)}{\partial v_{ij}}$ → cofactor expansion

$= \dfrac{1}{\det\Sigma} \dfrac{\partial(\sum_j (-1)^{i+j} v_{ij} M_{ij})}{\partial v_{ij}}$

$= \dfrac{1}{\det\Sigma} (-1)^{i+j} M_{ij}$

右式 $= e_j \Sigma^{-1} e_j^T = e_j \dfrac{adj(\Sigma)}{\det\Sigma} e_i^T$ — jth row.

$= \dfrac{(-1)^{i+j} M_{j\cdot} e_i^T}{\det\Sigma}$ ($M_{j\cdot}$: jth row of $adj(\Sigma)$)

$adj(\Sigma) = \begin{pmatrix} (-1)^{1+1}M_{11} & (-1)^{1+2}M_{12} & \cdots & (-1)^{1+m}M_{1m} \\ & & (-1)^{i+j}M_{ij} & \\ (-1)^{1+m}M_{1m} & & & (-1)^{m+m}M_{mm} \end{pmatrix}$ $M_{j\cdot}$

$= \dfrac{1}{\det\Sigma}(-1)^{i+j} M_{ij}$

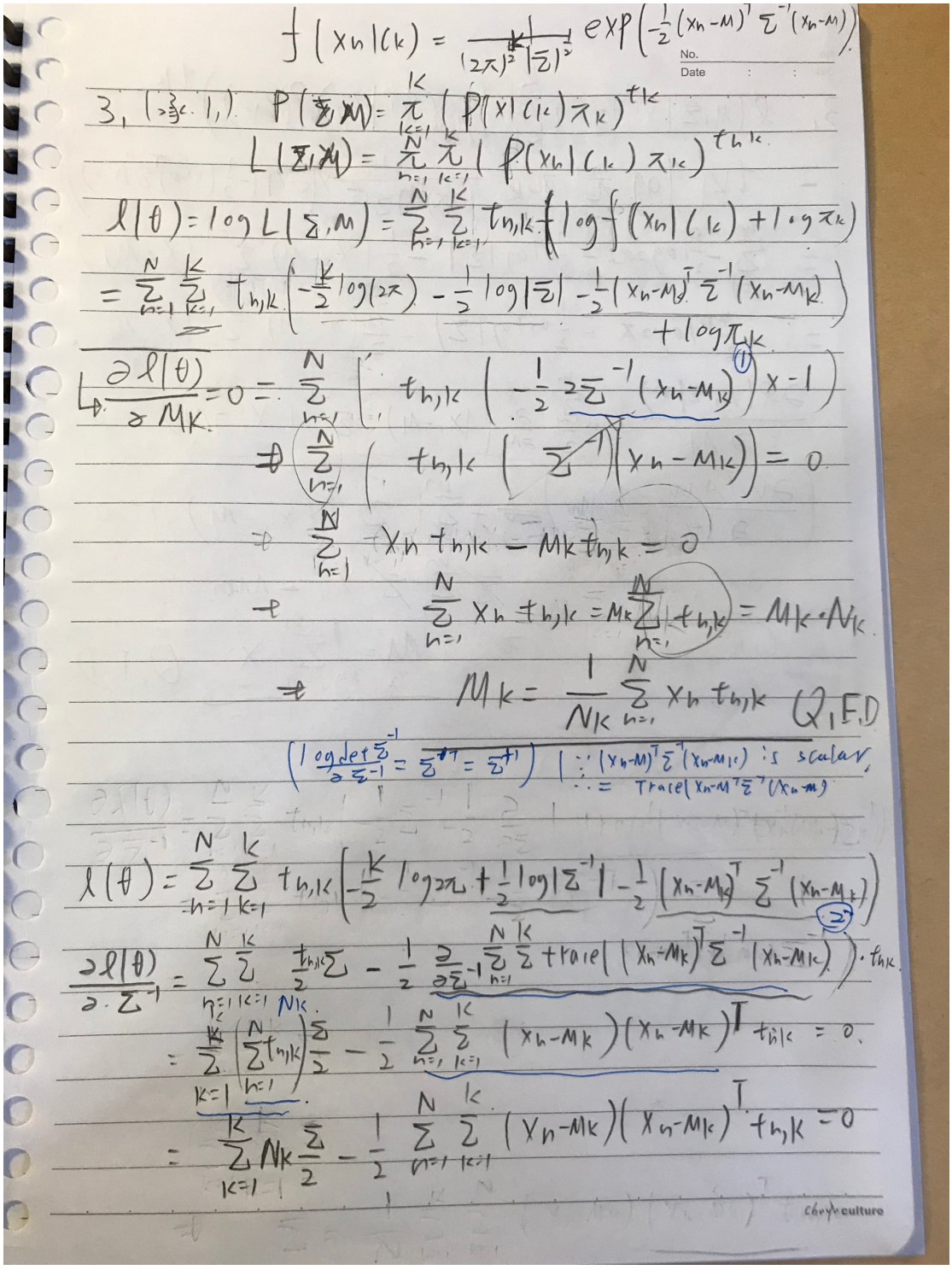
$\Sigma^{-1} = \dfrac{adj(\Sigma)}{\det(\Sigma)}$

左式 = 右式 Q.E.D #

$\sum_{k=1}^{K} N_k = \sum_{k=1}^{K} \sum_{n=1}^{N} t_{nk}(x_n - \mu_k)(x_n - \mu_k)^T$

$\therefore N_k = \sum_{n=1}^{N} t_{nk}(x_n - \mu_k)(x_n - \mu_k)^T$

(3)



$$\sum \sum_{k=1}^{K} N_k = \sum_{k=1}^{K} \sum_{n=1}^{N} t_{nk} (x_n - \mu_k)(x_n - \mu_k)^T$$

$$\therefore N \cdot \Sigma = \sum_{k=1}^{K} N_k \sum_{n=1}^{N} \frac{t_{nk}}{N_k} (x_n - \mu_k)(x_n - \mu_k)^T$$

$$\therefore \Sigma = \sum_{k=1}^{K} \frac{N_k}{N} \frac{1}{N_k} \sum_{n=1}^{N} \cdot (x_n - \mu_k)(x_n - \mu_k)^T$$

$$= \sum_{k=1}^{K} \frac{N_k}{N} \cdot S_k \quad \# \quad Q.E.D.$$

chuyu culture

No.

Date : :

$$\left(\frac{\partial w^T A w}{\partial w} = 2Aw. \right) \quad ①$$

$$\left(\frac{\partial (x_n - \mu_k)^T \Sigma^{-1} (x_n - \mu_k))}{\partial \mu_k} = -2\Sigma^{-1}(x_n - \mu_k) \right)$$

$\because x^T A x$ is scalar

$$\therefore \frac{\partial}{\partial A} x^T A x = \frac{\partial}{\partial A} tr(x^T x A) = (x x^T)^T = x x^T \quad ②$$

$$\frac{\partial}{\partial \Sigma^{-1}} trace (x_n - \mu_k)^T \Sigma^{-1} (x_n - \mu_k) = (x_n - \mu_k)(x_n - \mu_k)^T$$