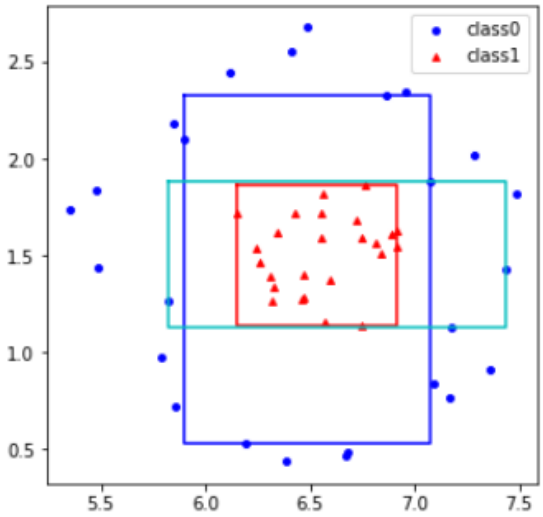
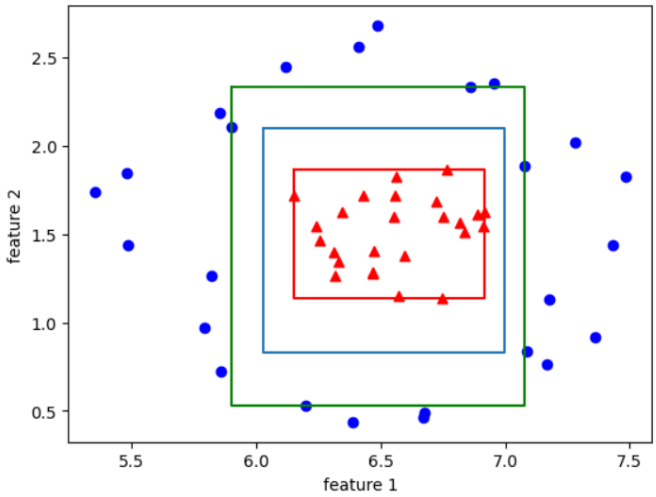
Machine Learning Homework 2

1. (a)



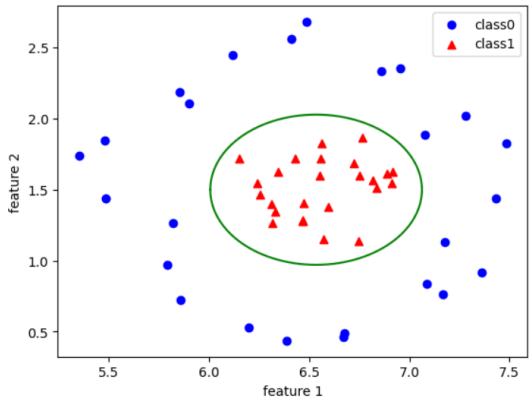



```
(a) (a < feature1 < b) AND (c < feature2 < d) :
a = 6.026296259216194
b = 6.998079203533234
c = 0.8335293241420705
d = 2.096388054631785
```

左圖：找到內層紅框(S)之後，往外延伸找外層藍框(G1、G2) 顏色較深的藍框(G1)是先將紅框(S)的左右邊界固定，往上下找第一個會碰到的點，再將這兩個點當作新的上下邊界，往左右再找第一個會碰到的點；而顏色較淺的青藍色框(G2)是將上下跟左右的順序調換來找。接下來較 G1 跟 G2 的範圍大小，選擇比範圍較大的那個(即 G1)，因為在找 most general hypothesis 時會想要找到比較大的範圍能包含所有可能的 class1 極盡可能沒有 class2。

右圖：將 G 和 S 相加取平均就得出青藍色框，取平均的目的是希望這個 hypothesis 能夠比較有彈性，而平均是在 specific 和 general 之間，因此錯誤的機率會相比單純採用 specific 或 general 來的小，因此選用這個方法。

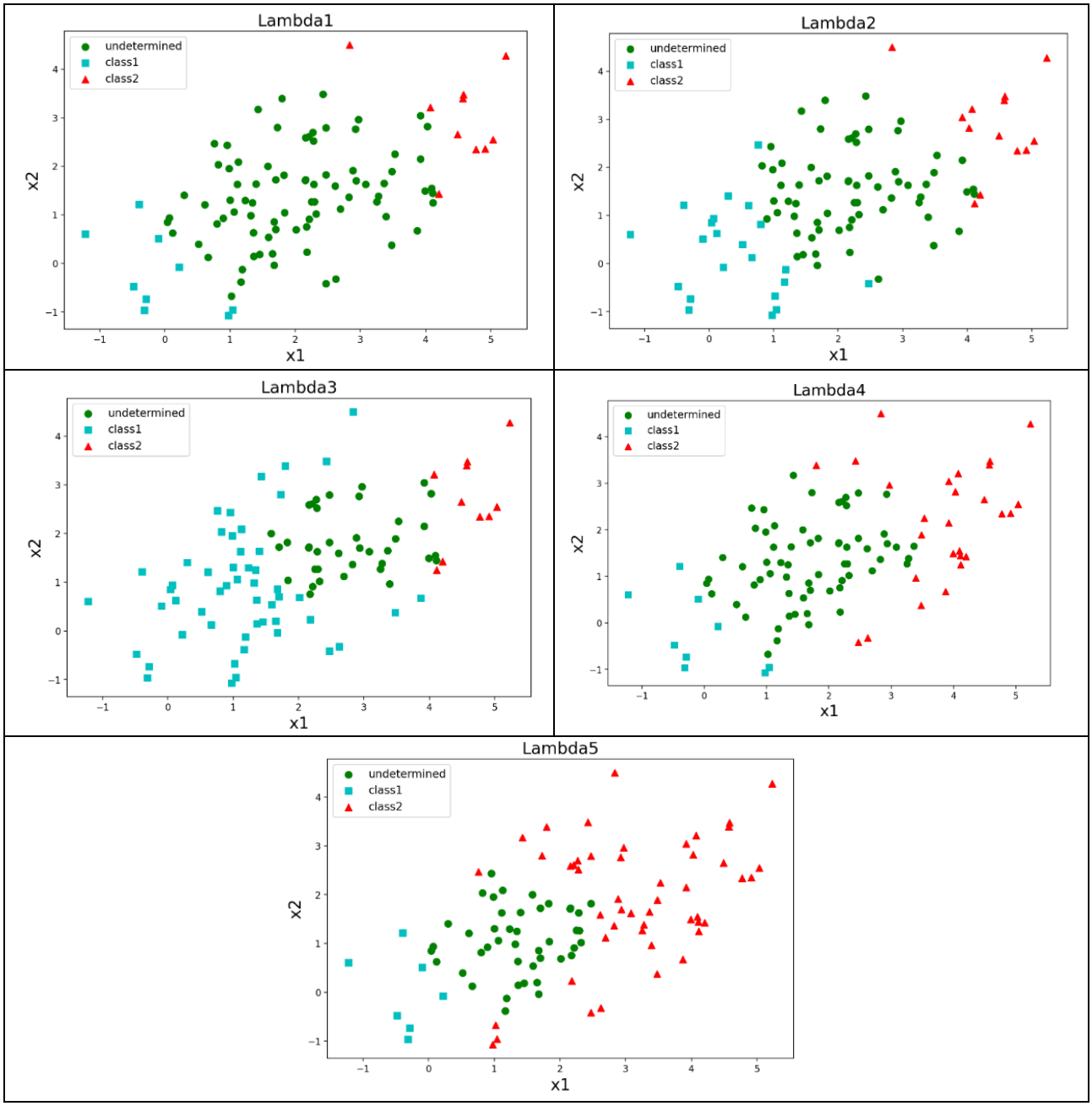
(b)




```
(b) (feature1 - a)^2 + (feature2 - b)^2 = r^2 :
a = 6.534277706375207
b = 1.4990416859683662
r = 0.5283724831686845
```

用(a)小題中紅框的中心點作為圓心，對角線長度作為直徑畫圓。因為橢圓形較難控制涵蓋點的範圍，此法較為簡易，而根據結果來看，此分類方法還算有效，故選擇此方法作為分類依據。

2.


Lambda1
undetermined
class1
class2
x1
x2
Lambda2
undetermined
class1
class2
x1
x2
Lambda3
undetermined
class1
class2
x1
x2
Lambda4
undetermined
class1
class2
x1
x2
Lambda5
undetermined
class1
class2
x1
x2

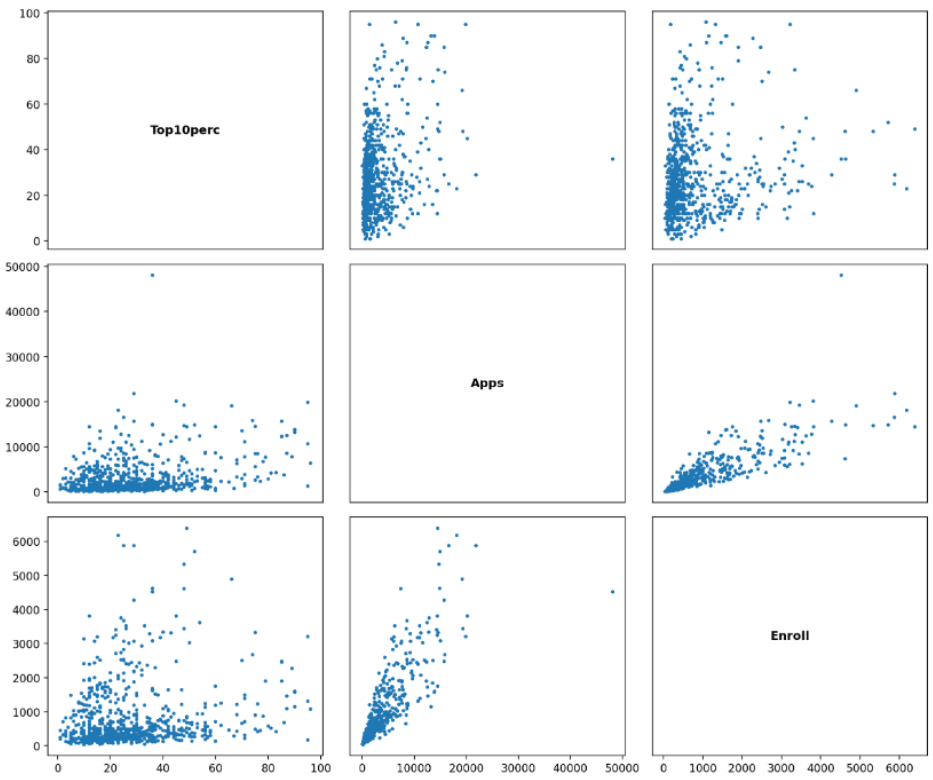
3.

(a)

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal
count	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000
mean	3001.638353	2018.804376	779.972973	27.558559	55.796654	3699.907336	855.298584	10440.669241	4357.526384	549.380952	1340.642214
std	3870.201484	2451.113971	929.176190	17.640364	19.804778	4850.420531	1522.431887	4023.016484	1096.696416	165.105360	677.071454
min	81.000000	72.000000	35.000000	1.000000	9.000000	139.000000	1.000000	2340.000000	1780.000000	96.000000	250.000000
25%	776.000000	604.000000	242.000000	15.000000	41.000000	992.000000	95.000000	7320.000000	3597.000000	470.000000	850.000000
50%	1558.000000	1110.000000	434.000000	23.000000	54.000000	1707.000000	353.000000	9990.000000	4200.000000	500.000000	1200.000000
75%	3624.000000	2424.000000	902.000000	35.000000	69.000000	4005.000000	967.000000	12925.000000	5050.000000	600.000000	1700.000000
max	48094.000000	26330.000000	6392.000000	96.000000	100.000000	31643.000000	21836.000000	21700.000000	8124.000000	2340.000000	6800.000000

從這些數據我們能看到每個 feature 的 mean、variance 等資訊，其中比較能夠從這些數據來分析結果的資訊大概只有 mean，比如申請人數和錄取人數間的差異：平均而言，有 3001 人申請，但只有 2018 人被接受。這可能表示整體來看，大學的錄取率約為 2/3；或者是高中成績排名前 10%和前 25%的學生百分比：平均而言，27.56%的學生屬於前 10%，55.8%的學生屬於前 25%，這可能表明這些學校在學術上具有一定的選擇性。但只根據數據也很難去理解他的分佈，因此接下來才會希望透過圖表方便我們觀察。

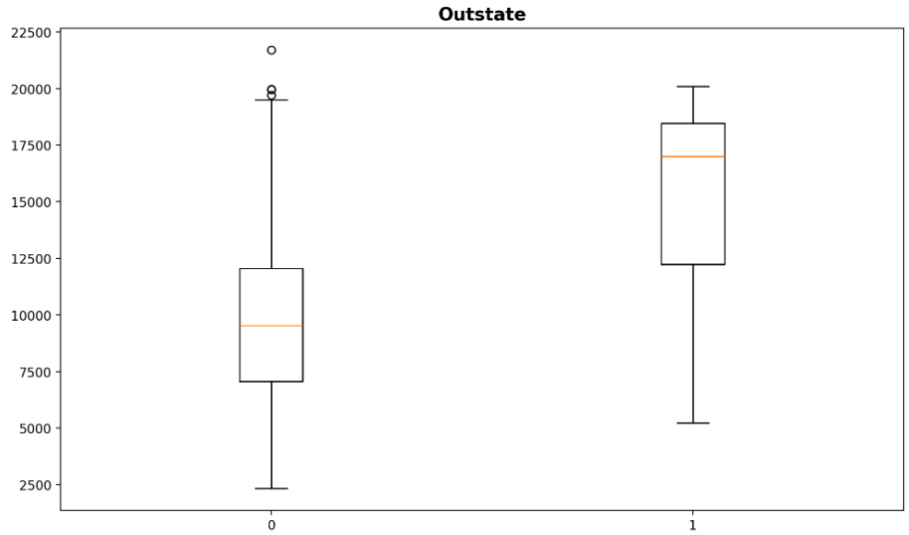
(b)



從這些 scatter plot 中我們可以清楚看到兩個 feature 之間的分佈情形與關聯性，也可以觀察到極端值得存在。而這些 feature 的對應關係我們可以觀察到比如申請人數和註冊人數的關係：隨著申請人數的增加，註冊人數也有增加的趨勢，因為申請人數越多，理論上有更多的學生有機會註冊。而和 Top10%有關的集中在左下，因為 Top10%的學生也只佔全體 10%，因此它的比例集中在比較低的地方也很合理。但從 scatter plot 我們只觀察到 feature 之間的關係。

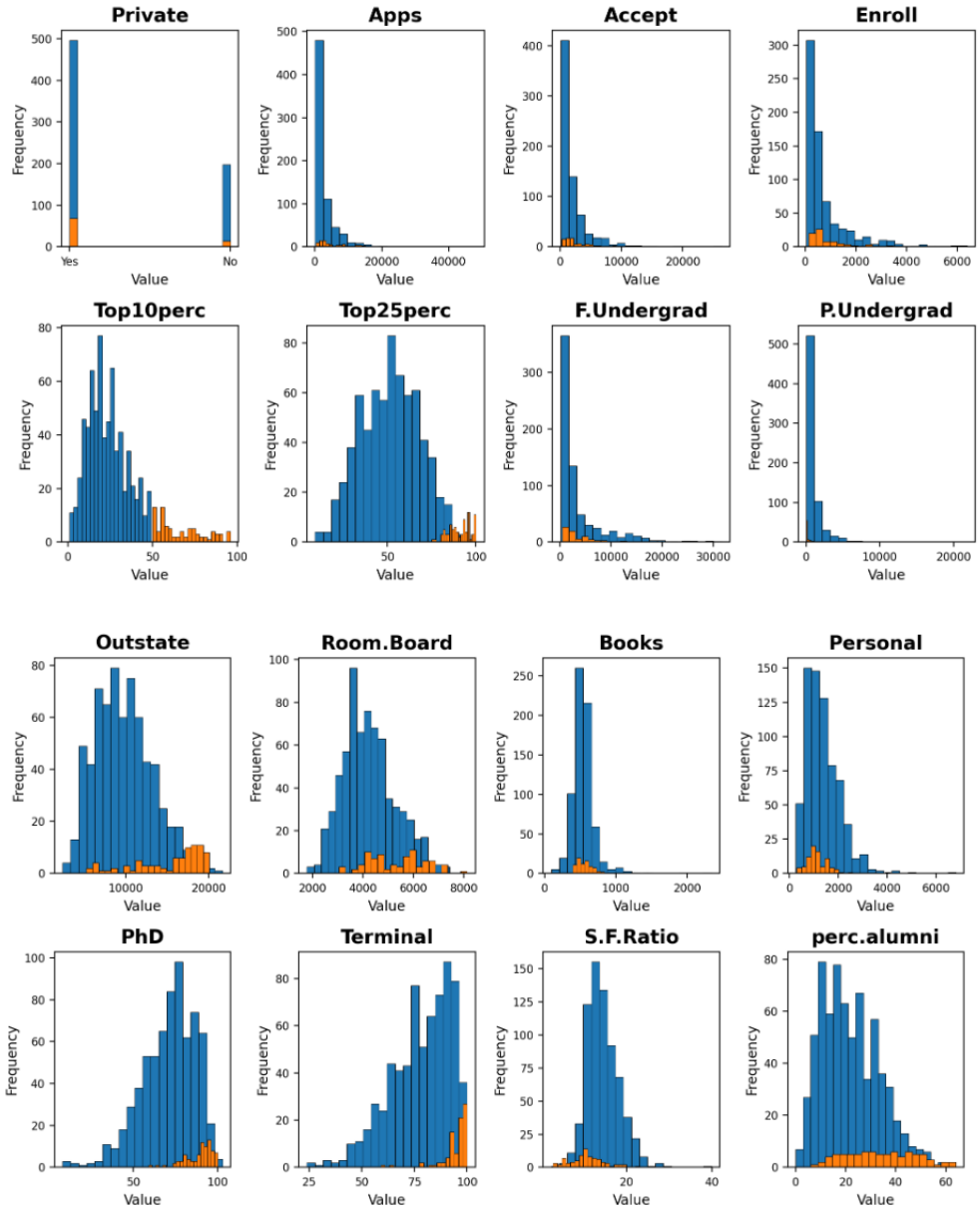
(c)

```
Total elite university : 83
```



從 box plot 我們可以觀察到 data 的中位數、四分位數及極端值。而以下幾點是我的觀察：Elite 學校的外州學費在整體上似乎比非 Elite 學校要高。這可以通過兩個 box 的中位數來看出，其中 Elite 學校的中位數高於非 Elite 學校。還有在兩組中都有極端值，但在非 Elite 學校中，這些極端值似乎特別高，遠高於普通範圍。Box plot 展示了 Elite 和非 Elite 學校在學費上的顯著差異，並且即使在同一個分組內，學校之間的學費也可能有很大的差異。

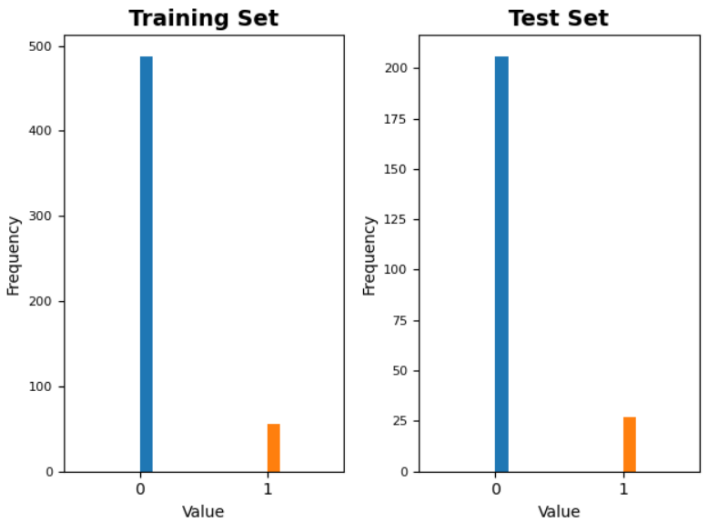
(d)



接下來我透過 histogram 去觀察 data 的分佈情形，也透過 histogram 去選定我想要用於 model 的 feature。在這些圖表中我想要挑選分佈比較像 normal distribution 的 feature，因為接下來使用的 model 是基於 normal distribution，還有能從 mean 分出差異的圖，以及 variance 比較小的圖。首先像是 Top25%、Outstate、PhD 這些我認為就是屬於比較好的 feature，因為以上都大致符合我上面所說的條件，而舉個反例 perc.alumni 我認為就不算好的 feature，因為它的 variance 比較大，因此不適合做為分辨 Elite 的 feature。

最後我選擇 Top25%、Outstate、PhD 以及 Books 作為我的 feature，後面會再討論 Books 這個 feature 的影響。

(e)



(f)

```
Accuracy of Normal distribution:  0.94
Confusion matrix of Normal distribution:
 [[ 22   5]
 [  9 197]]
```

```python
# (f)
select_feature = ["Top25perc", "Outstate", "Books", "PhD", "Elite"]
mu             = train[select_feature].groupby("Elite").mean()
sd             = train[select_feature].groupby("Elite").std()
count          = train[select_feature].groupby("Elite").count().iloc[:,0]. rename('counts')

print("Mean:\n")
display(mu)
print("Standard deivation:\n")
display(sd)
print("Counts:\n")
display(count)
```

首先我使用 Top25%、Outstate、PhD 以及 Books 這四個 feature 套進 Naïve Bayes Classifier 之後，得出的準確率約為 94%，從 confusion matrix 也可以看到 False Positives，也就是模型錯誤預測為 Elite 類別的數量小於 False Negatives，也就是模型錯誤預測為非 Elite 類別的數量。代表被誤判成非 Elite 的情況比較多。


```
Accuracy of Normal distribution:  0.94
Confusion matrix of Normal distribution:
 [[ 22   5]
 [  9 197]]
```

```python
# (f)
select_feature = ["Top25perc", "Outstate", "PhD", "Elite"]
mu             = train[select_feature].groupby("Elite").mean()
sd             = train[select_feature].groupby("Elite").std()
count          = train[select_feature].groupby("Elite").count().iloc[:,0]. rename('counts')

print("Mean:\n")
display(mu)
print("Standard deivation:\n")
display(sd)
print("Counts:\n")
display(count)
```


接下來我想討論 Books 這個 feature 對結果的影響。我將 Books 這個 feature 移除之後發現最後得出來的結果和它存在時是相同的，代表說 Books 這個 feature 存在與否其實對結果沒有影響，因此我們也能看出其實當 feature 選的足夠好的話其實 model 就能很準確地預測了，feature 的多寡並不是直接和準確度做關聯。

(g)

```
Accuracy of Log-Normal distribution:  0.884
Confusion matrix of Log-Normal distribution:
 [[  0  27]
 [  0 206]]
```

因為在上題中 normal distribution 的準確度滿高的，因此在這題我使用了與 normal distribution 有關的 lognormal distribution 去做，但結果卻跟 normal distribution 相差甚遠。

判斷結果顯示全部的學校都是 class 0，原因可能是因為 lognormal 比較適用於極端事件較多的數據集。因為 lognormal distribution 是由多個隨機變量相乘的結果，所以相乘後會產生出很多接近 0 或是非常大的數字。因此，可能是因為我選擇的 feature 中有很多偏小的隨機變量，造成相乘後的結果都接近 0。