Report

1.

假設w中有$w_1, w_2 \dots w_k$等 d 個參數，$\nabla \mathrm{E_{in}}(w)$就是把$\mathrm{E_{in}}(w)$分別對$w_1, w_2 \dots w_d$進行偏微分，再合併起來成一個 d 維的向量。

$\mathrm{err}(\mathrm{w^T x, y}) = \begin{cases} (1 - yw^T x)^2 \;\; if \;\; yw^T x > 1 \\ 0 \;\; if \;\; yw^T x \leq 1 \end{cases}$，$\mathrm{E_{in}}(w) = \frac{1}{N}\sum_{n=1}^{N} \mathrm{err}(\mathrm{w^T x_n, y_n})$，所以對$\mathrm{E_{in}}(w)$進行偏微相當於對$\frac{1}{N}\sum_{n=1}^{N} \mathrm{err}(\mathrm{w^T x_n, y_n})$用 k 個 w 進行偏微，也就是說，

$\nabla \mathrm{err}(\mathrm{w^T x, y}) = [-2y_n x_i max(1 - yw^T x, 0) \;\; for \;\; i \;\; in \;\; range(d)]$

代入可得:

$$\nabla \mathrm{E_{in}}(w) = \frac{1}{N}\sum_{n=1}^{N} [-2y_n x_i max(1 - yw^T x, 0) \;\; for \;\; i \;\; in \;\; range(d)]$$

$$= [\frac{1}{N}\sum_{n=1}^{N} -2y_n x_i max(1 - yw^T x, 0) \;\; for \;\; i \;\; in \;\; range(d)]$$

2.

根據 wiki 的寫法，$\mathrm{p_u(x_n)} = \frac{\mathrm{e}^{-\frac{1}{2}(x_n - \mu)*(x_n - \mu)}}{\sqrt{(2\pi)^d}}$。$\mathrm{u^*} = \arg\max \prod_{n=1}^{N} p_u(x_n)$ ，為了$\mathrm{u^*}$可以使$\prod_{n=1}^{N} p_u(x_n)$最大，所以要微分等於 0。但因為長相很難看，所以可以先取 ln 再微分。

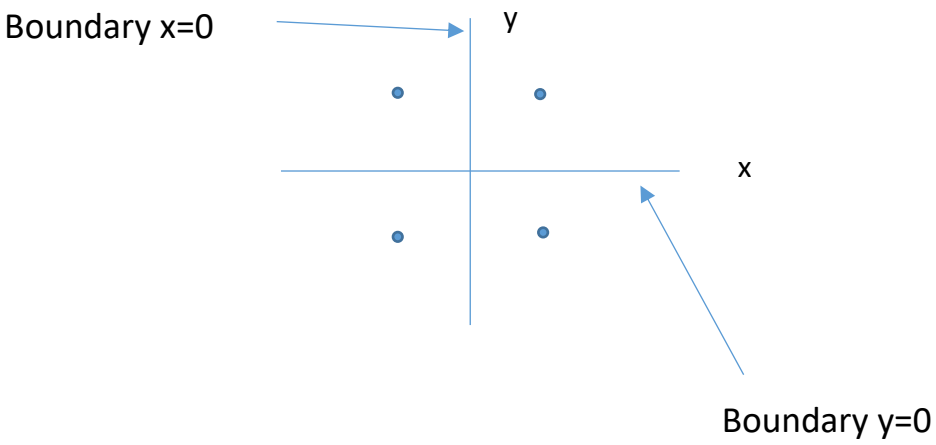
$$\ln(\prod_{n=1}^{N} p_u(x_n)) = \ln\left(\prod_{n=1}^{N} \frac{\mathrm{e}^{-\frac{1}{2}(x_n - u)*(x_n - u)}}{\sqrt{(2\pi)^d}}\right) =$$

$\sum_{n=1}^{N} \ln(\frac{e^{-\frac{1}{2}(x_n-u)*(x_n-u)}}{\sqrt{(2\pi)^d}}) = \sum_{n=1}^{N} \ln(\frac{e^{-\frac{1}{2}(x_n-u)*(x_n-u)}}{\sqrt{(2\pi)^d}}) = \sum_{n=1}^{N} \left(-\frac{1}{2}(x_n - u) * (x_n - u) - 跟\ u\ 無關的常數\right)$

所以，$\frac{d \ln(\prod_{n=1}^{N} p_u(x_n))}{du} = \frac{d \sum_{n=1}^{N} \left(-\frac{1}{2}(x_n-u)*(x_n-u) - 跟\ u\ 無關的常數\right)}{du} = \frac{d \sum_{n=1}^{N} \left(-\frac{1}{2}(x_n-u)*(x_n-u)\right)}{du} = \sum_{n=1}^{N} \left((x_n - u)\right) \rightarrow u = \frac{1}{N} \sum_{n=1}^{N} x_n$

3.

非常合理的構造了$-x_1 x_2$。所以設定$w = (0,0,0,0,1,0)$就可以得到$y_n = sign(w^T z_n) = sign(-x_1 x_2)$，把$x_1 = 0,1, x_2 = 0,1$代進去就可以發現這符合答案。classification boundary 是$x_1 = 0$ or $x_2 = 0$



4.

$$\epsilon_t = \frac{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n)}{\sum_{n=1}^{N} w_n^t} \rightarrow d_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$$

$$= \sqrt{\frac{1 - \frac{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n)}{\sum_{n=1}^{N} w_n^t}}{\frac{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n)}{\sum_{n=1}^{N} w_n^t}}}$$

$$= \sqrt{\frac{\sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n)}{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n)}}$$

這裡，我定義$\sim\delta(g_t(x_n), y_n)$就是 0 if $\delta(g_t(x_n), y_n) = 1$，是 1 if $\delta(g_t(x_n), y_n) = 0$

所以，

$$\frac{\sum_{n=1}^{N} w_n^{t+1} * \delta(g_t(x_n), y_n)}{\sum_{n=1}^{N} w_n^{t+1}}$$

$$= \frac{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n) * d_t}{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n) * d_t + \sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n)/d_t}$$

$$= \frac{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n) * d_t^2}{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n) * d_t^2 + \sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n)}$$

$$= \frac{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n) * \frac{\sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n)}{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n)}}{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n) * \frac{\sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n)}{\sum_{n=1}^{N} w_n^t * \delta(g_t(x_n), y_n)} + \sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n)}$$

$$= \frac{\sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n)}{\sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n) + \sum_{n=1}^{N} w_n^t * \sim\delta(g_t(x_n), y_n)} = 0.5$$

1.

我的結果如下:

Logistic Regression Accuracy: 1.0
Decision Tree Classifier Accuracy: 0.8888888888888888
Random Forest Classifier Accuracy: 0.9333333333333333
Linear Regression MSE: 39.663748810565984
Decision Tree Regressor MSE: 28.341570306709183
Random Forest Regressor MSE: 26.92599736842105

可以看出來在 iris 的 dataset 上，linear model 表現最好。我 iteration 設 200000。Random forest 比 Decision tree 好是正常的。Boston 的部分我 iteration 設 2000，因為大於 2000 會有 overfit 的問題。這部分 random forest 是最好的。

2.

在 logistic regression 的部分，iteration 設 20000，用 normalization 跑，accuracy 是 1。用 standardization 跑，accuracy 是 0.93。這 2 個步驟都是為了處理資料，讓訓練過程誤差或計算不會被某個參數大幅度影響，而且 gradient descent 的速度會比較快。在這題中，好處就如上面所示，但壞處就是會讓一些應該明顯的特徵變得不明顯。

3.

Learning rate and iteration	logistic	linear

0.01 and 1000	0.6666666666666666	43.42393109996493
0.5 and 1000	0.9555555555555556	63.60109876772848
0.01 and 200000	1.0	63.363705859664655
0.5 and 200000	0.9777777777777777	63.3626480412708

可以看到 logistic 的部分 learning rate 跟 iteration 越大效果越好。但 linear 的部分如果 learning rate 跟 iteration 太大就會有 overfit 的問題。我測 linear 最好大概是在 learning rate 跟 iteration 是 0.01 跟 2000 左右，mse 可以降到 39。

4.

Num of trees and max_depth	Classification	Regression
100 and 5	0.9333333333333333	26.92599736842105
10 and 5	0.9555555555555556	26.18026315789474
100 and 20	0.9333333333333333	27.007740131578945
10 and 20	0.9555555555555556	27.70394736842105

正常來說，max_depth 越深，model 就越複雜，越容易 overfit。而 num of trees 越多，generalize 的效果就越好。透過以上數據可以看到，max_depth=20 的 mse 比 max_depth=5 再高一些，有 overfit 的問題。而在 Classification 上沒有明顯差別。而 Num of trees 對數據影響不大，大概是因為這個數量的 trees 就已經很足夠了。

5.

如果結果跟給定的參數有很明確的線性關係就可以用 linear model，像 iris，缺點是如果沒有明確的線性關係的話 linear moel 表

現就不太好，用 nonlinear model 會更好。如果用邏輯判斷比較好就會用 decision tree 或 random forest，通常後者會比前者更 generalize 一點，缺點是如果不太能用邏輯判斷的話 decision tree 就不太好。Model 複雜度通常越高表現越好，但不太 generalize，而且不太好理解參數的意義。複雜度低一點表現就不會太好，但會比較 generalize，也比較容易理解參數的意義。像 linear model 的複雜度就比較低，參數很好理解就是重要性。但如果有加 sigmoid 等比較複雜的 model 就不太好解釋計算的結果了