姓名: 吳耿暉

學號: 0556171

1. ML approach/MAP approach/Bayesian approach

這個部分的論討會集中在ML approach上面, 將ML approach設計完整之後後面的部分基本上就會直接使用相同參數去跑。

A_1. ML approach

- Model:

在model的設計上, 所有的basis為了簡單起見, 使用能夠有close form的Gaussian Distribution來當作basis, 另外嘗試了4種不同的model設計方式。

1. 先用一組小的training data, 在每個data point的X, Y上面都放上一個Gaussian Distribution,這種方式只是為了想先做出overfitting的結果, 進而初步確認design matrix和basis function的設計上"可能"沒有錯誤。此方法參考自Stanford CS231n課程當中的notes(http://cs231n.github.io/neural-networks-3/), 雖然這部份是在談訓練neuron network的前置作業, 但我認為在這部分的方法是共通的, 節錄原文如下：

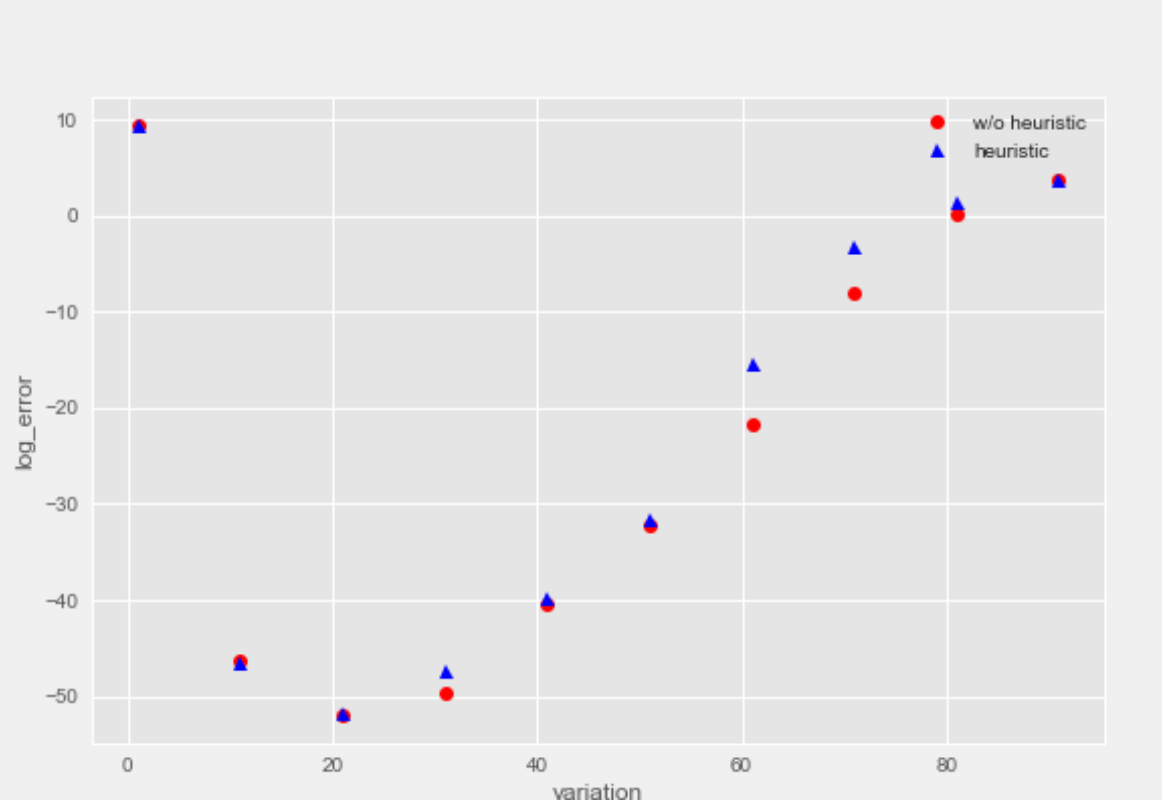
> "…Overfit a tiny subset of data. Lastly and most importantly, before training on the full dataset try to train on a tiny portion (e.g. 20 examples) of your data and make sure you can achieve zero cost. For this experiment it's also best to set regularization to zero, otherwise this can prevent you from getting zero cost."

此部份我認為特別適合於ML approach, 因為我們只考慮了data本身, 因此如果設計正確那麼應該會得到training data的error極低而validation data的error(可能)極高的效果, 而在實際測試上的確獲得如此的結果, 在每個點的variance為(5,5), 亂數選取400個training data和100個validation data下, 得到的training error=2.10359347506e-25, 而validation error則為14403.7896313。

2. 將地圖平均分割

如同作業描述當中所使用的方法, 對於地圖做分割後, 每個分割點上都放上一個Gaussian Distribution, 在這邊首先要先挑選幾個參數, grid的大小, variance的選取, 但是對於這其中的組合由於計算資源不足，沒辦法全部測試, 因此我選擇了先固定grid大小選擇最好的variance, 接下來再由固定這個variance去選擇比較好的grid大小，此處variance原本使用了一點heuristic的設計,在地圖上觀察發現左邊X=1~200, Y=1~1082的長方形區域大部分都為海洋, 因此在這邊的grid使用較高的variance(兩倍於陸地的variance), 因為判斷海洋應該是比較平滑的區域, 而陸地就採用相對較小的variance。在此設定下, 使用亂數選取1000筆資料, grid size為30之下, 實驗結果如圖一, 發現實際上這個heuristic沒什麼用處。

另外有嘗試在陸地上使用X較小而Y較大的長方形grid的分割方式, 並且將陸地的variance改成沿Y軸大於沿X軸的方式，也是一樣看地圖覺得沿Y軸似乎有比較高的變異程度才試著做這種方法，但是隨便試驗了一下也沒有顯著改善error，如果做PCA可能可以看出這方面的關係，但由於時間關係這方法就沒有繼續深入。

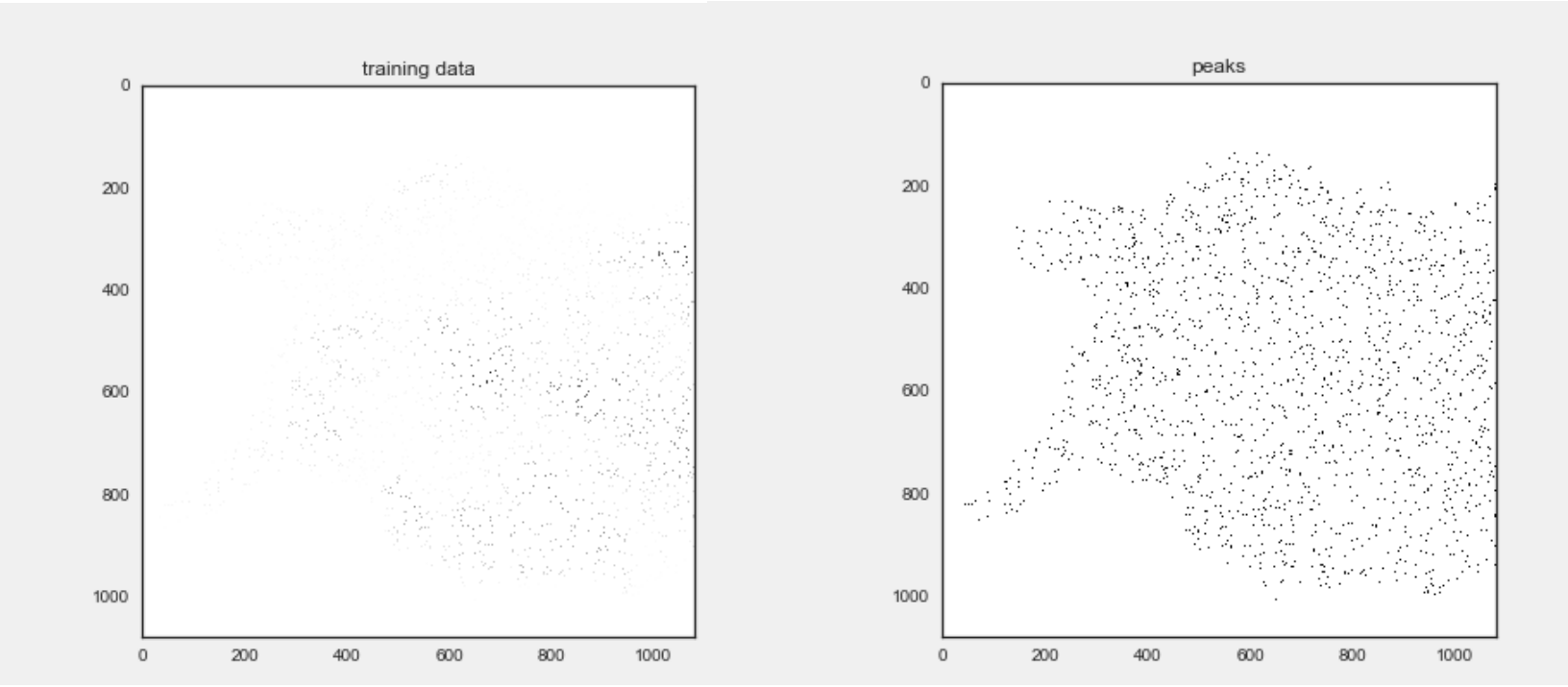


圖一：比較有無使用heuristic的方法

3. 找出地圖的peak

理想的model應該是所有的最高點剛好都是Gaussian Distribution的mean所在位置，因此試著利用peak detection的方式找出所有的相對高點，首先先將資料轉換為1082x1082之類似image的形式，不知道的點就填0, 其他填高度，接著通通把值map到0和255之間, 接著就是利用peak detection, (作法參考自http://stackoverflow.com/questions/3684484/peak-detection-in-a-2d-array)以8-nearest的定義方法找出所有的相對高點，但很快就發現一個很大的問題，一般來說圖像不會有那麼多的未知點， 因此相對的高點並不會太多，但是在這次的資料當中太多未知資料，如果使用peak detection將會產

生非常多的相對高點(32000筆training data最後跑出18XXX個peak, 如圖二), 在連續度不足的資料上，這顯然是個用處不大的model。



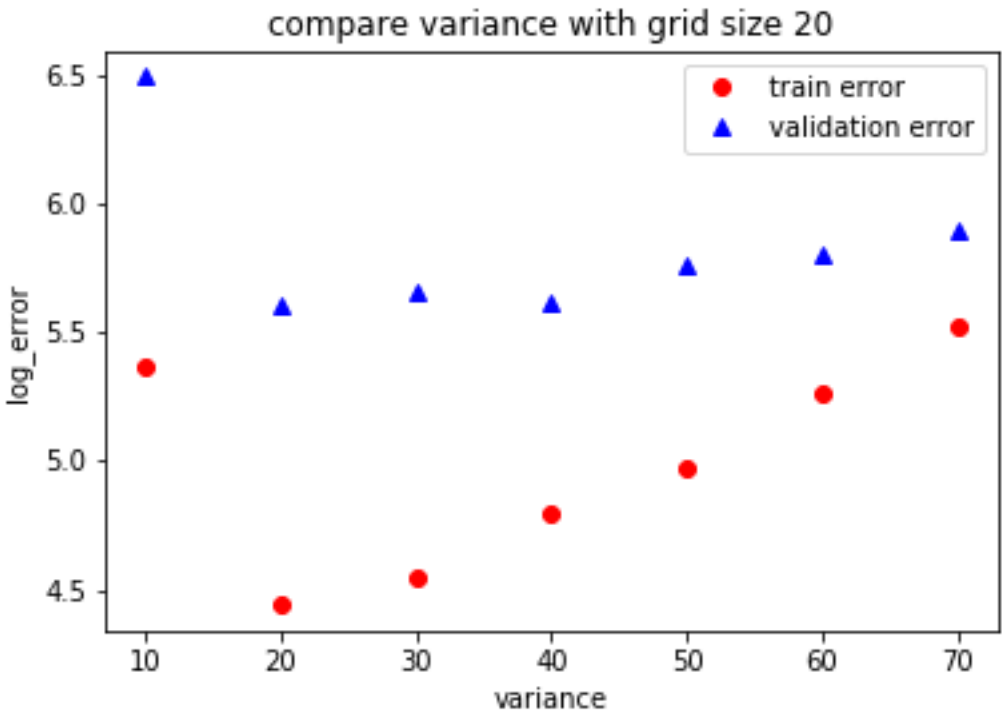
圖二：比較training data和peaks的分佈, 過多的peak對於運算上完全無法負擔

4. 結合2和3的混合型

雖然3的結果並不成功, 但是其中的思路對我還是蠻有參考價值的, model的設計上如同法2一般是對整個區域做切割, 但是將3的peak detection簡化為只取整個區域的最大值, 將這個最大值的座標上面放Gaussian Distribution。只取最大值的原因是因為, 如果取到第二甚至第三大的值, 有可能這些值的座標距離最大值的座標很近, 相當於我們使用幾乎重複的分佈來去近似某一小塊grid, 雖然可以利用距離來判斷, 但是太多feature也會造成計算

上的負擔, 因此就只取最大值。另外如果這塊grid的最大值是0, 那這塊grid就只在正中間放一個分佈, 原本是隨機去放, 但是實際上在正中間的效果較好, 這也蠻符合直覺的。另外在法2當中失敗的heuristic這邊成功敗部復活, 發現固定grid最大值是0的區域, 選擇其variance為70~80的效果能夠讓performance有所提升, 因此後面會使用這個setting。

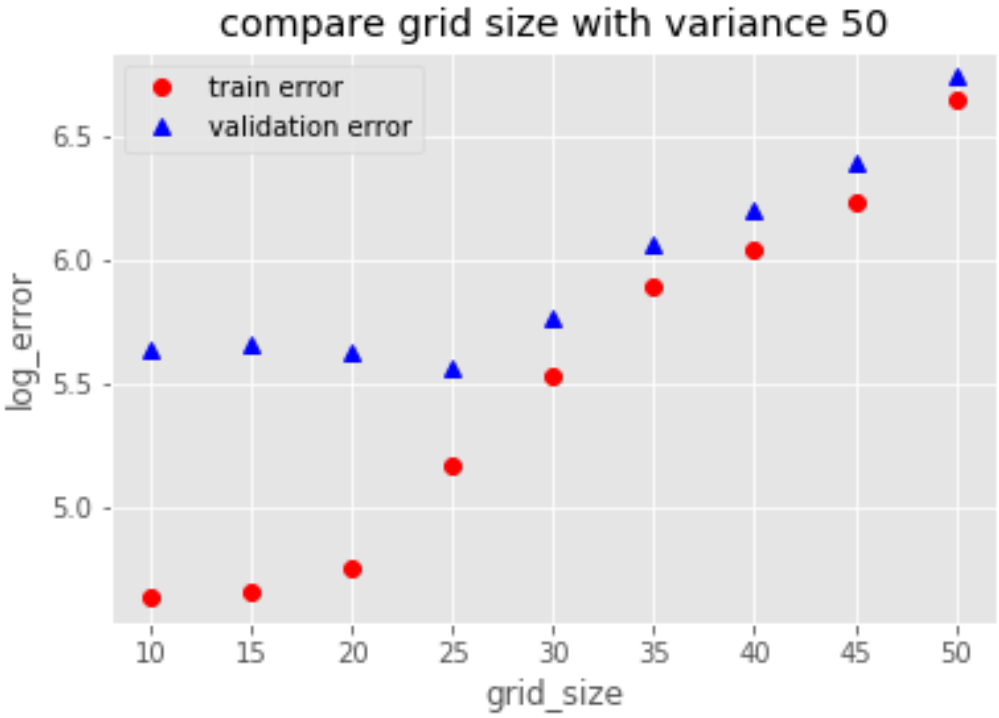
首先先看高度非0的區域variance取多少較好, 實驗結果如下圖三, 此為grid size為20的結果：



圖三：不同variance下的error比較

由上圖可發現在此設定下最佳的variance為20, 因此選擇variance為20, 接著再回頭去看grid大小的比較如下圖四, 可以看到在grid小於25時validation set的error開始上升, 因此選擇grid大小為25, 在調這些參數時, 這些結果會隨著資料取的不同和資料量大小的不同抑

或是是否有用heuristic上都有蠻大的相關性, 而且調整grid size時最佳的variance也會有所變化, 目前沒有想到除暴力法外可以同時考慮這幾點的方式, 只能先假定這個參數的設定還不錯(因為電腦不夠暴力)。在此總結一下ML approach最後的參數設定:grid size=25, 可能為陸地之variance=(20,20), 可能為海洋之variance=(70,70), 最終將資料切為32000之training和8000之validation之error分別為203.04701和 230.87634。



圖四：不同grid size的比較

A_2. MAP approach

由於feature basis都為高斯分佈, 因此這部分直接用Ridge Regression來求即可, 比較麻煩的是由於資料量多於feature數目不少, 因此若選擇使用QR分解的方式求的話複雜度

會顯著高於利用反矩陣方式求得的W_MAP, 因此為了將大部分的資料都納入計算, 我還是決定使用普通的反矩陣求法, 如此一來可以利用cross-validation挑選lambda(更正: 後來在grid切越來越小時, 會發生數值問題導致error整個爆掉, 因此還是必須要用QR分解來計算W_MAP, 並且改為使用cross-validation挑選beta, 但是在variance的部分簡單起見就沒有再去分為20/70, 而是統一為20) 後面再對這部分討論。

補充: 在經過最後面的cross validation步驟後, 發現在這個問題的model設計上, 使用了regularization項反而造成了整體performance下降, 在此重新使用beta=1,000,000的model來求testing data的prediction, 得到training /validation error分別為356.493 / 454.177。

A_3. Bayesian approach

此處為了計算方便, 將m_0設為0, 如此m_N的計算就變得很單純, 最後training/ validation error分別為698.589和734.165, 可以看出performance並沒有很好, 推斷是由於prior過於簡化造成, 這邊可以看出如果prior設計的不好的話, 那麼使用bayesian approach可能不但造成運算量增加, 效果更可能變更差, 但m_0可以利用cross validation重新設計, 因此應該非常有改善的空間。

D_1. Overfitting with W_ML

造成overfitting有幾種可能, 上述A_1.1有兩種概念:

1. 資料量過少, 導致fit的結果的generalization很差, 造成training error非常低但是validation error卻極高的結果。

2. 用每個data point都放高斯的方式去設計model, 等於我們手動去添加了一個重要度極高的feature叫做”人腦判斷”(參考自台大林軒田教授的課程內容), 這也會造成overfitting的結果, 即model的設計沒辦法反映整體的狀況。

以上兩種問題都可以利用加入更多的data point來一定程度的解決, 類似regularization此種對過大的weight做penalty的方式也是一種解決方案。

D_2. Underfitting with W_ML

Underfitting的問題較為單純, 通常就是model太過簡單造成的, 在圖四中可以看到, 當grid size選取越來越大時, 等同於我們的model變得越來越簡單, 因為我們嘗試利用越來越少的feature來fit training data, 造成不止training error, validation error也會逐漸上升。

2. Cross Validation for beta of MAP

這邊採用5-fold cross validation, 並且挑選beta由100到190, 間隔200, 步驟如下:

```
for each lambda
    for each fold
        計算W_MAP得到training/validation error
    若平均的validation error較好, 更新最佳的W_MAP/ beta
return best beta
```

之所以選擇這個range來測試是因為前面嘗試幾個較小的beta結果都很差, 後來beta漸漸調大時才開始讓error下降, 這可能是因為原本的model設計的就有不

錯的generalization, beta取太小等於lambda增加, 表示懲罰了較大的weight, 反而讓對於validation set的generalization變得更差。

由於最後才發現一個bug沒修, 導致最後我只用了12000筆資料來進行cross-validaion, 對於每個beta, 其average validation error如下:

	1	2	3	4	5	average
100	965	885	849	877	886	892
500	764	671	670	688	709	700
900	717	622	633	646	672	658
1500	681	585	605	614	646	625
1900	665	570	593	601	538	593

最終選定beta為1900, training / validation error分別為 550.945 / 630.772, 這邊的validation set並不包含於做cross validation所用到的data, 而是原始的40000筆資料中獨立切出8000筆的結果, 在這邊觀察到實際上這幾種beta的選擇仍然沒有讓error收斂, 還在緩步降低當中, 因此應該要繼續嘗試選用更高的beta, 在beta逐漸升高的過程中就會接近ML approach的結果(lambda趨近於0), 但是是否能夠比ML approach好則很難說, 如果資料是從母體中完全隨機抽取的, 那麼這40000筆資料可能已經足以反映整體的分佈, 如此ML approach的結果可能已經非常接近這個model的極限。

3. 結論:

在上課的時候, 以為ML/MAP/Bayesian approach有一種循序漸進的感覺, 在準確度上我也有此認知, 但是在做完這次的作業之後會發現, 當model設計還ok並且資料量足夠時, 有可能MAP/Bayesian approach並不一定能夠得到更好的結果, 後者可能比前面考慮到更多的東西, 但同時也多了更多的假設, 如果這些假設不良就會導致不太好的結果, 因此根據數據的分佈, 模型的複雜度以及對於一些prior的前提知識都對選擇何種approach有很重要的影響, 在訊息量極少的狀況之下, 利用後面兩種approach才能夠避免ML approach可能造成的overfitting。