學號：R07921001 系級：電機所二 姓名：李尚倫

1. (2%) 請比較實作的 generative model 及 logistic regression 的準確率，何者較佳？請解釋為何有這種情況？

以baseline model的結果來看generative model的訓練結果：

Size of training set: 54256

Size of testing set: 27622

Dimension of data: 510

Training accuracy: 0.8725302270716603

Top 10 related data categories：

Retail trade 8.3359375

Not in universe -6.703125

34 -6.33203125

37 -5.80859375

Different county same state -5.65625

Other service -5.42578125

Abroad 4.625

Finance insurance and real estate 4.0

Same county 4.0

Other Rel 18+ never marr RP of subfamily 3.9375

以baseline model的結果來看logistic regression model的訓練結果(iteration=200)：

Training loss: 0.26462566695822753

Development loss: 0.2850628360679869

Training accuracy: 0.8856850296948597

Development accuracy: 0.879100626612606

Top 10 related data categories：

Not in universe -2.19705786183989

Spouse of householder -1.6574094289726329

Other Rel <18 never married RP of subfamily -1.4264126521048024

Child 18+ ever marr Not in a subfamily -1.3003326945908804

Other Rel <18 ever marr RP of subfamily -1.1759633817911175

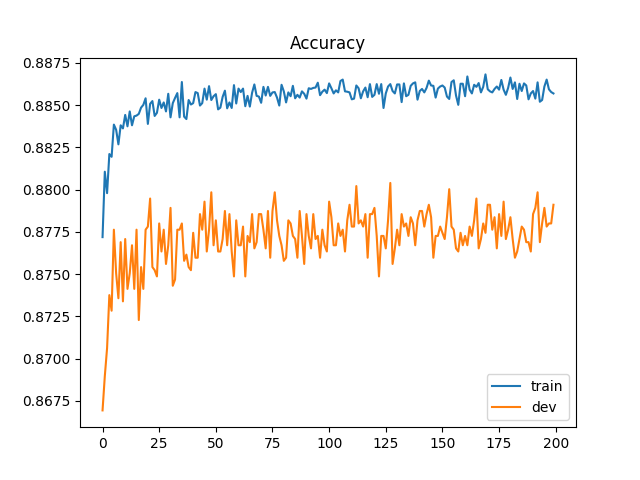
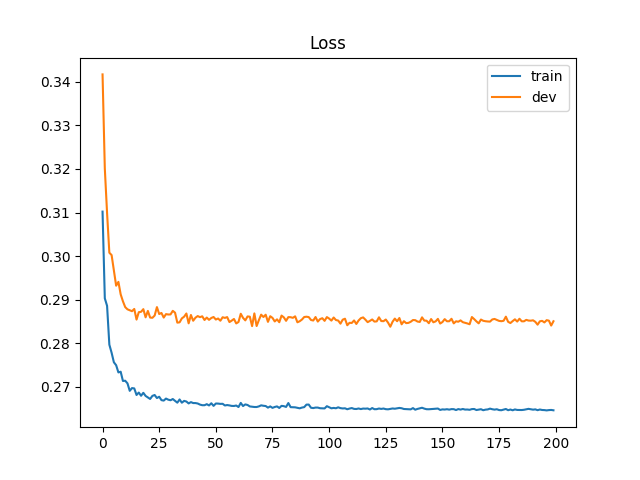
Unemployed full-time 1.1158406568740264

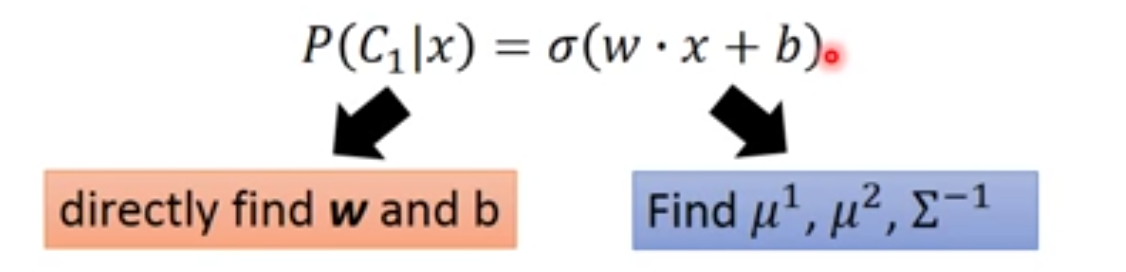
Italy -1.1016297598508888

Vietnam -1.0827250388575935

1 0.8058595497596417

Same county -0.7573031321591495





兩者都假設擁有一樣的covariance，用的是同一個機率的model(擁有相同的function candidates)，但discriminate是靠data和regression的方式直接求得w和b，而generative則是基於Gaussian distribution的假設，和假設每個個dimension相互independent(沒有correlation)等等，在不同的情況下者兩種求解方式各有利弊，而就這次的題目來說，是discriminate model(logistic regression)有較好的performance，因為資料中有許多是具有相關性的，也有很多不一定會符合Gaussian distribution，所以用logistic regression做出來的結果會更貼近於真實的反映一點。

因此最後選用的是logistic model來當baseline model：

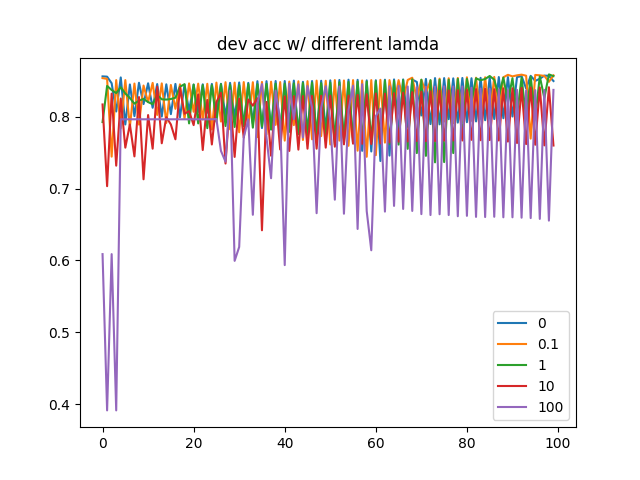
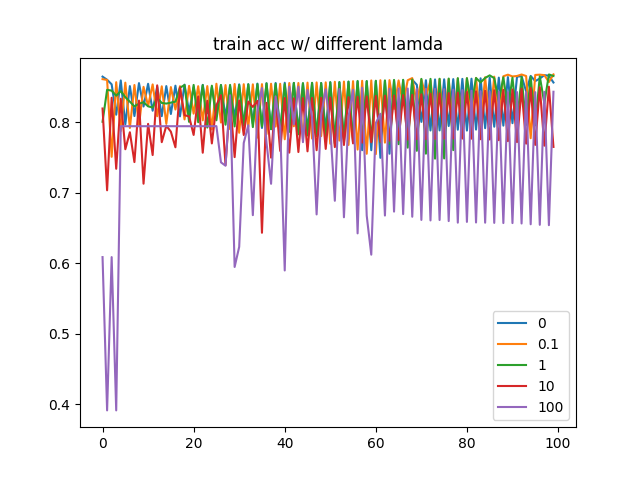
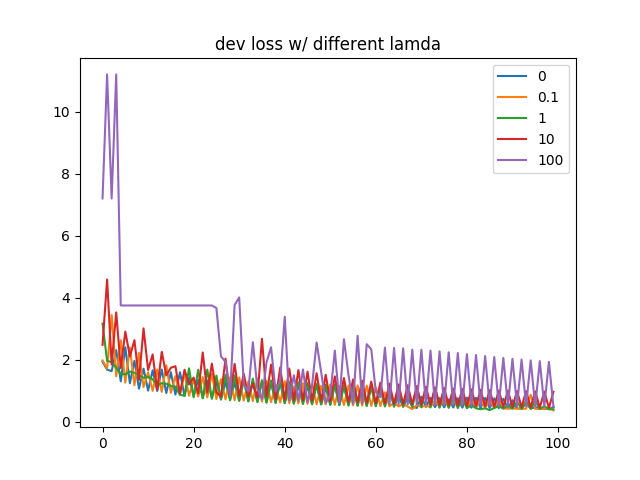
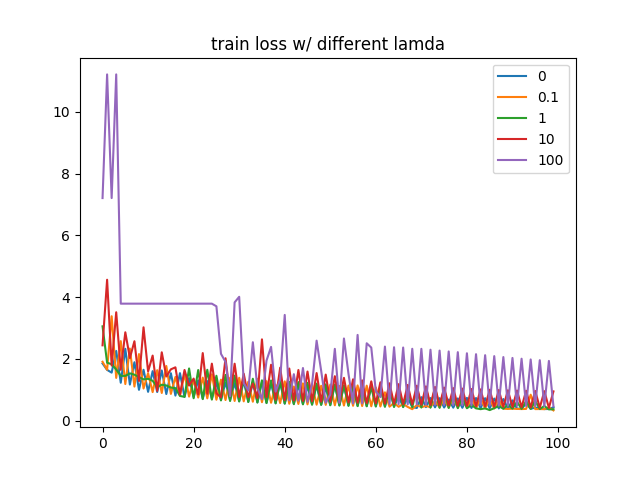
The best in 200iter, lr=0.2, bch=8, epoch=127

Development loss: 0.28380508846919755

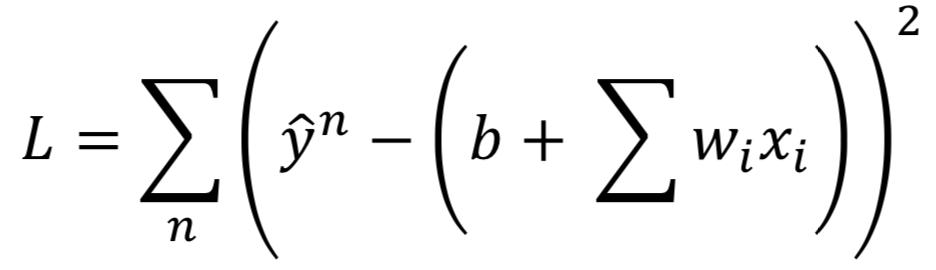
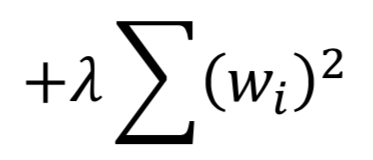
Development accuracy: 0.8803907113896056

Testing accuracy(public): 0.88987 > 0.88617 = simple baseline

2. (2%) 請實作 logistic regression 的正規化 (regularization)，並討論其對於你的模型準確率的影響。接著嘗試對正規項使用不同的權重 (lambda)，並討論其影響。(有官 regularization 請參考 [https://goo.gl/SSWGhf p.35](https://goo.gl/SSWGhf%20p.35))



由上圖可以看到，我考慮了五種不同的lamda(包含=0也就是不做regularization)，加入太大的lamda做regularization會讓weight的平滑度被考慮的權重比training error大很多，可能會使model收斂上困難，loss產生震盪，而lamda太小則都考慮training error和沒加入差不多，而在這個model 中，考慮平滑度沒有太顯著的幫助收斂效果，因此最後strong baseline的model中也未採用。

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

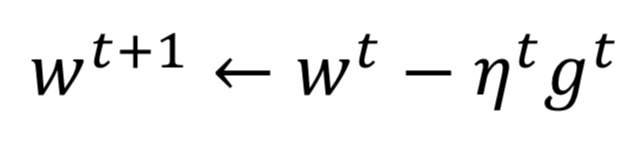
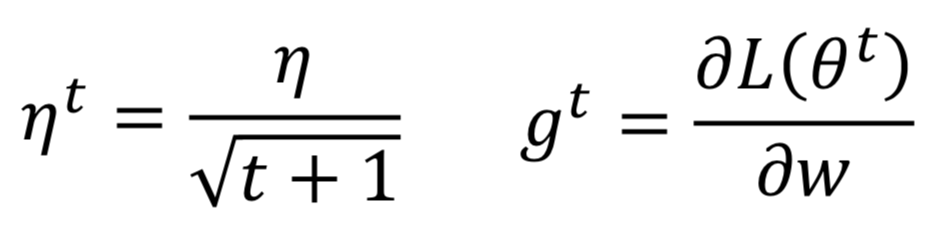
* Batch size, learning rate, iteration

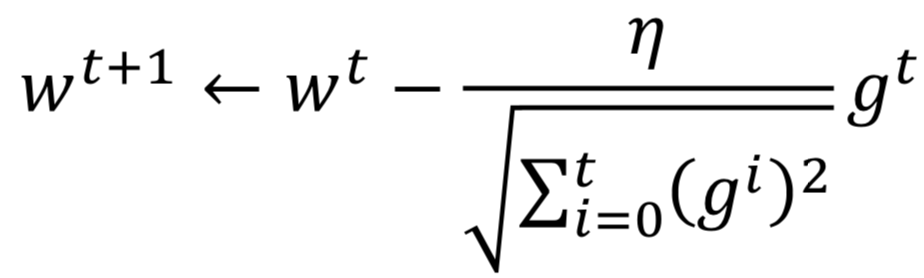
經過測試，batch size取1024可以使一個epoch算快一點且loss和取size=8差不多。Learning rate的部分也是經過測試後0.2可以在適當的時間下降到比較低的loss，iteration的部分統一給3000個iteration進行training，但取其中最小loss的weight為model來做prediction。

* Regularization in regression

經過前面第二題的實驗，最後採不進行regularization 的方式(lamda=0)。

* Vanilla gradient v.s. adagrad

Vanilla gradient:  

Adagrad: 

參考作業一實作adagrad和本次作業使用的vanilla gradient比較，adagrad具有更好的收斂能力，能在比較少的epoch數內找到最小的loss，故採用adagrad。

* Filtering data

借鏡第一次作業的經驗，在data normalized後把大於正負三個標準差的data濾掉變成等於正或負三個標準差，來減少與mean值相差太遠的資料，在作業一中即為雜訊，但本次作業中資料來源應為問卷，並無雜訊或過大的無意義偏差，因此加入後並未改善，最後未採用此方法。

* Normalizing data

根據第四題實驗，將data normalized可以更穩定地趨近最小的loss，故採用。

* Carefully selecting features

根據直接觀察dataset和用baseline model取出的前十相關項目，此dataset中有許多無謂且多維的feature，例如：region/state of previous residence, country of birth father/mother/self，這些資料因為one hot encoding會變成57維和129維的資料，雖然能使model變複雜，但太瑣碎，不是有意義的增加複雜度，還會因為增大維度導致計算和收斂變慢，拿掉不會有太大影響。另外或是資料分布不均衡的項目，例如：reason for unemployment裡面大部分都是not in universe，不看這項會比較好。

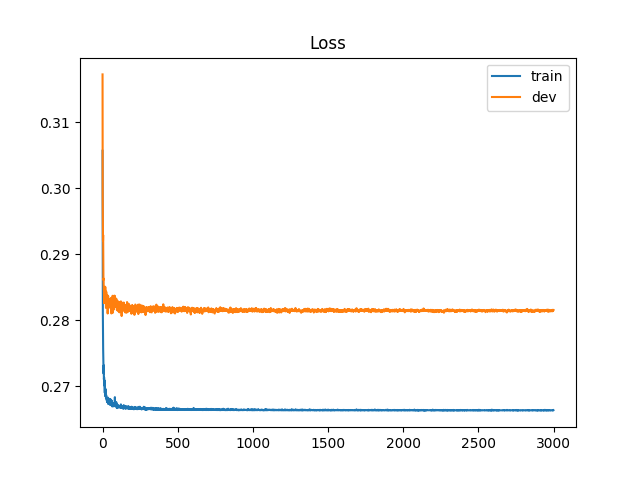
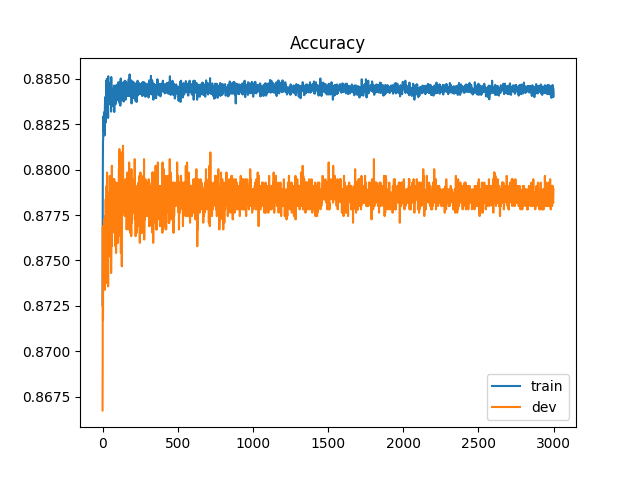
* Increasing dimension of features

由於根據上述改進後，僅能拿到比strong高一點點的正確率：

Dimension = 318

Epoch 127, Dev Loss 0.2806616055019925, Dev Acc: 0.8809436048654626

Testing accuracy(public): 0.89052

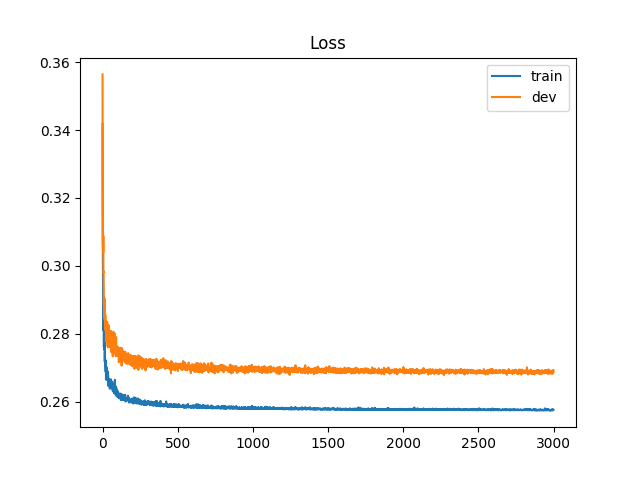
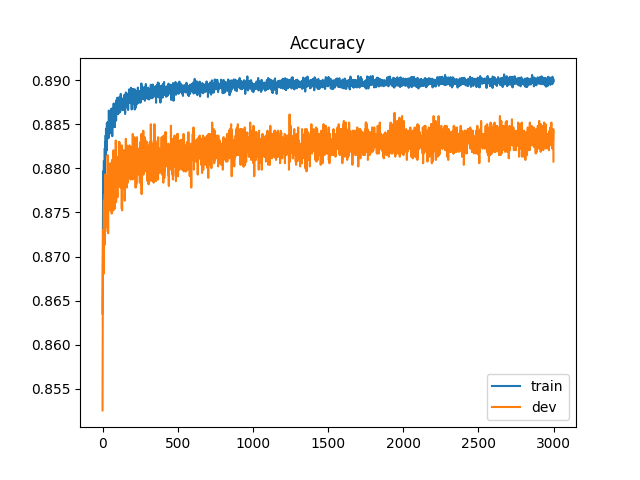


因此決定增加model的複查度(維度)，加入每個分類的平方項和根號項：

Dimension = 954

Epoch 2271, Dev Loss 0.26787734418288583, Dev Acc: 0.8846295613711759

Testing accuracy(public): 0.89406

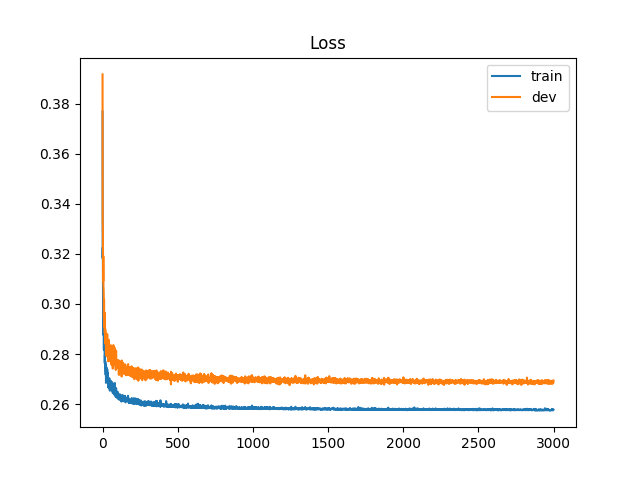
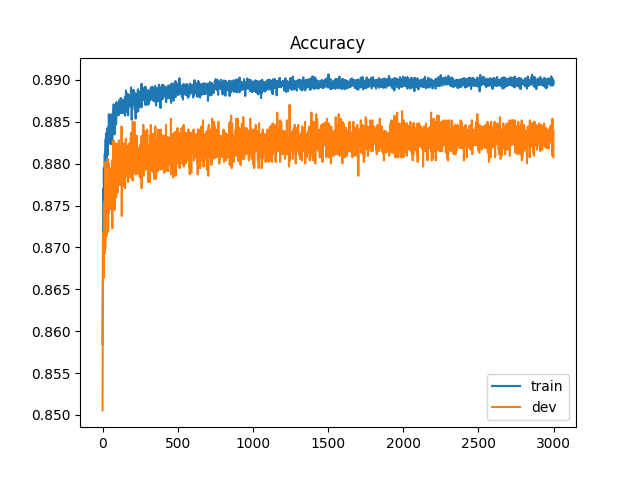


增加為度後model的準確率有明顯的改善，因此決定再增加維度，補上原本資料的一次項，變成1\*根號項+2\*一次項+1\*二次項，有些許的改善為最終定案：

Dimension = 1272

Epoch 1244, Dev Loss 0.26756170678263397, Dev Acc: 0.8870254330998895

Testing accuracy(public): 0.89457 > 0.89052 = strong baseline



4. (1%) 請實作輸入特徵標準化 (feature normalization)，並比較是否應用此技巧，會對於你的模型有何影響。

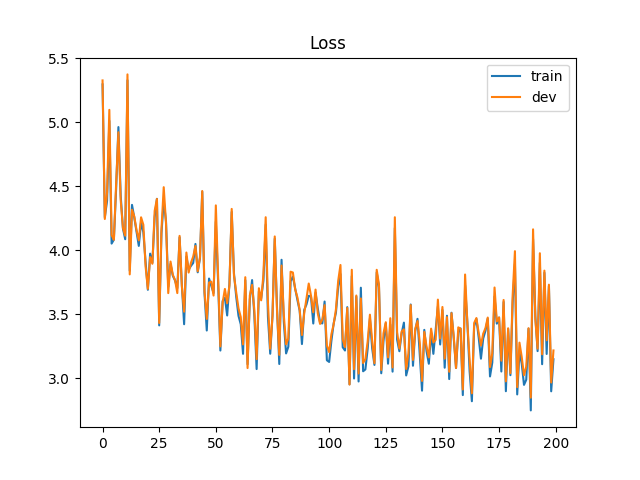
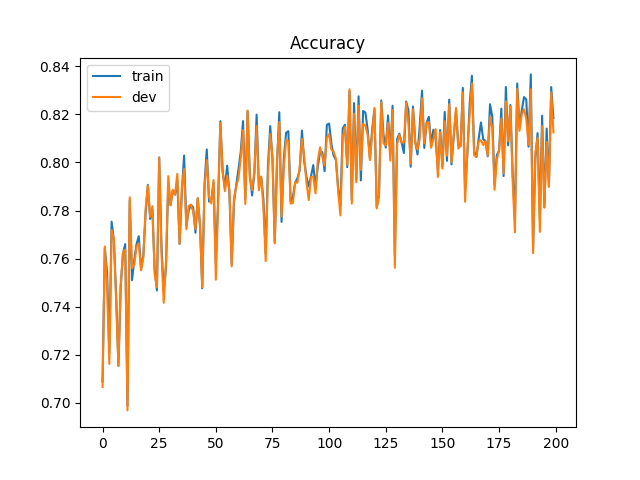
Without normalized:

Training loss: 3.149125348988123

Development loss: 3.2175856963260223

Training accuracy: 0.8183903338111816

Development accuracy: 0.8125691116844821



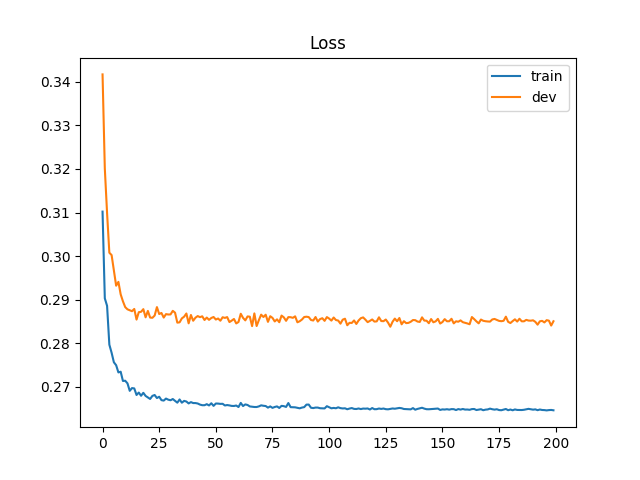
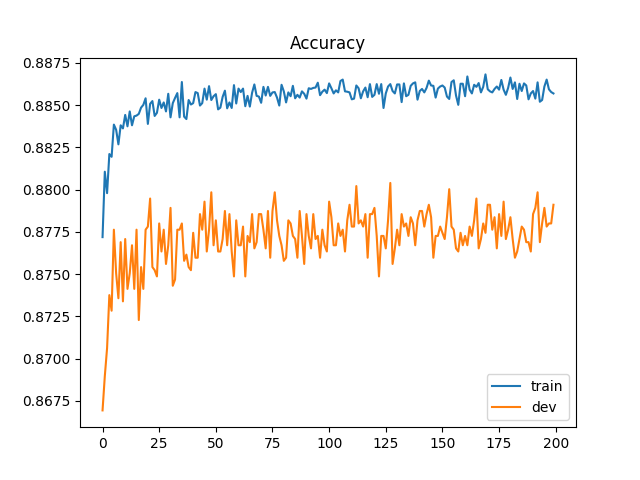
Normalized:

Training loss: 0.26462566695822753

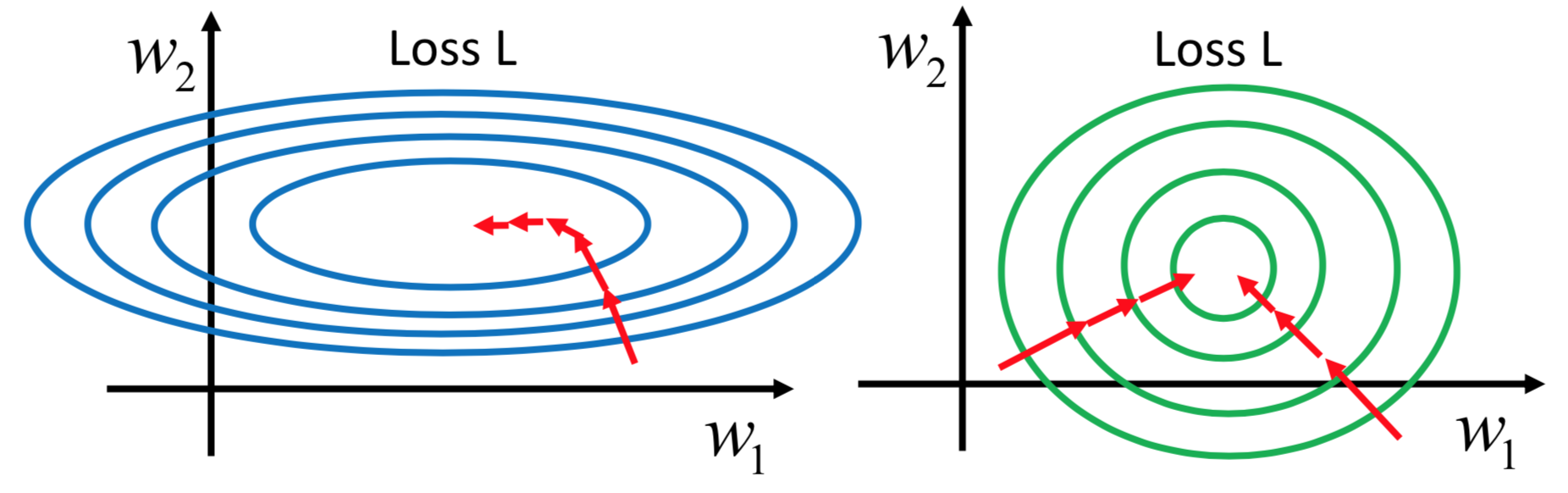
Development loss: 0.2850628360679869

Training accuracy: 0.8856850296948597

Development accuracy: 0.879100626612606



由上面的數據可以看到，少了data的normalization會讓model的收斂變困難非常多，較難找到loss的極值，而加入normalization會make different features have the same scaling



如上圖，因為我們的feature有scalar的也有one hot encoded的，因此做normalization後loss會好收斂很多。