Machine Learning 2019 Fall
HW2 Report

---

b05902105 余友竹

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

	PUBLIC	PRIVATE
Logistic Regression	0.85614	0.84915
Generative Model	0.84017	0.83871

Logistic Model 的performance明顯比較好，我認為有兩個決定性的因素影響Generative Model的performance：

1. 假設Data生成自某個高維的Gaussian Distribution

   在這個Task中，若使用經過one-hot encoding轉換的training data，代表我們想用一個106維的Gaussian Distribution估計原始的資料分布。

   但這高維的數據中，包含了許多經過one-hot encoding轉換的特徵值，我認為已經失去了數值大小的意義，粗略的使用Gaussian Distribution來估計可以方便做最佳化，卻有違常理。

2. Naive Bayes假設每筆機率都是獨立

   這在多數情況下都很不make sense。我們不知道我們採樣的data有沒有相依性，舉例，收集的data可能包含一個家庭的成員，若爸爸有錢，那極有可能他的小孩、老婆、爸媽、親戚，也有相當的經濟水平。

2. 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

我實作了兩種normalization的方式：

1. Normalization: $\frac{x_i - \mu_x}{\sigma_x}$
2. Rescaling: $\frac{x_i - \min(x)}{\max(x) - \min(x)}$

Logistic Regression

	PUBLIC	PRIVATE
Normalization	0.82960	0.82987
Rescaling	0.85479	0.84952

Generative Model

	PUBLIC	PRIVATE
Normalization	0.84336	0.84068
Rescaling	0.84017	0.83871

可以發現，在logistic regression model上，rescaling 的表現比normalization好。而generative model中，兩者表現相近。

我認為rescaling應該是比normalization合理的處理方式，因為資料中包含了經過one-hot encoding做的轉換，這類數據有個特性：非0即1，也就是說，這類數據是不含大小的特性的，因此坦討他距離平均值有多遠是沒有意義的。

在rescaling的轉換下，那些透過one-hot encoding轉換的特徵值，1者仍為1；0者仍為0，不會改變one-hot encoding的特性，比較合乎邏輯。

但同時資料中也有具數值大小意義的特徵，如age、fnlwgt, etc. 我認為，更合理的處理方式是：針對這類的數值做normalization，針對透過one-hot encoding轉換的數值做rescaling。

3. 請說明你實作的best model，其訓練方式和準確率為何?

先給上結果

	PUBLIC	PRIVATE
Best Model	0.86670	0.86205

我的best model使用了scikit-learn中的幾個套件

1. Logistic Regression
2. Random Forest Classifier
3. Gradient Boosting Classifier
4. Voting Classifier

sklearn中的logistic regression 的 performance其實就跟手刻差不多，差別只在sklearn中實作了多種不同的regularizer，而且支援平行處理，提升處理效率。

而我認為使用random forest在直覺上其實非常符合這類題目。我們可以想像在現實社會如何判斷一個人的年收入：可能會設下許多條件，並給上一個門檻，他是不是美國人？他的資產有多少？ ...等。

這其實就像random forest中，每顆decision tree在做的事。

而且我們可以透過篩選特徵、設定bootstrap的比例...等，來訓練出非常diverse的decision tree model，再讓這些decision tree進行投票，達成整個model的共識。

加上訓練時，每棵樹都可以個別進行訓練，不互相影響，因此他們非常適合做平行運算，訓練的過程相當快速，結果也相對準確。

實際上，單輸出random forest的結果就可以過strong baseline了，但為了更多樣化model的種類，我使用了跟random forest有點淵源的: gradient boosting model。

最後將這三種model一起丟進voting classifier裡做加權平均的動作，其實這步手刻不會差太多，大概的想法是讓三種model預測testing data的prabability，取log後做加權平均(weights: 1 : 3 : 3)(等同於對原本的probability做幾何平均)，使用套件的好處同樣是他可以支援平行運算。

補充：實作上不實用的幾種方法

1. 特徵篩選

   我嘗試了對原本的特徵進行篩選，選出比較重要的特徵進行其他操作

   篩選的方式是透過random forest訓練完找出feature importance，進行排序，篩出前50名

2. 特徵轉換

   配合上面的方法，我嘗試讓比較重要的特徵值互相相乘，2次方, 3次方, ... 10次方, etc.

   目的是將這些特徵投影到高維空間，讓我們可以做出非線性的決策超平面

3. iterative上述的步驟，篩出真正非常重要的前300名，進行其他相對複雜的訓練如fully connected neural network, etc.

但實際上能提升的效果非常有限，甚至跟沒做差不多，我認為有以下幾種可能

1. random forest本身就能預測出非線性的決策超平面

   random forest在收集不同decision tree的意見時，就像是收集不同線段的決策平面。

   由於每個決策超平面的方向都不盡相同，在voting時，若合成非常多的model，就能近似出一個非線性的決策超平面

2. overfitting training data

   這很好理解與解釋，當我們嘗試使用更高維度的決策超平面時，往往會有overfit的問題

   我認為解決辦法有幾種

   a. regularize

      嘗試給予決策係數較小的值，阻止最後的決策超平面過於扭曲、不切實際。

   b. SVM

      透過kernel trick將資料轉換到高維平面來做預測，嘗試預測出一條使得margin越大越好的、**中庸**的決策超平面。

      這本質上也是在對feature transform做regularize

但實際上，加上這幾種方法後仍然無法提升performance，由於繼續fine-tuning下去感覺淪於調參大賽，我沒有過於深究是哪方面出了問題，所以把這幾種方法歸類在不實用的方法。

4. 數學問題

1. optimal prior probability

令 $F = P_C(x_1 x_2 \cdots x_N)$ 表示以$C_1, \cdots, C_K$為參數定義的 likelihood function, C_{x_i} 表示x_i所屬的類別。其中，$i = 1, \cdots, N$。

根據獨立性假設：

$$\begin{aligned}F &= P_C(x_1 x_2 \cdots x_N) \\ &= P_C(x_1) P_C(x_2) \cdots P_C(x_N) \\ &= P(C_{x_1}) P(x_1 | C_{x_1}) \cdots P(C_{x_N}) P(x_N | C_{x_N}) \\ &= \prod_{i=1}^{N} P(C_{x_i}) P(x_i | C_{x_i})\end{aligned}$$

取log：

$$\begin{aligned}\log F &= \log \prod_{i=1}^{N} P(C_{x_i}) P(x_i | C_{x_i}) \\ &= \sum_{i=1}^{N} \log P(C_{x_i}) + \sum_{i=1}^{N} \log P(x_i | C_{x_i})\end{aligned}$$

$\because\ \forall\, x_i \in C_k,\ \ \log P(C_{x_i}) = \log P(C_k),\ i = 1, \cdots, N$

而 $\forall\, k = 1, \cdots, K,$ 共有 N_k 個資料屬於類別 C_k

因此對於所有k = 1, ..., K，可以如此表示

$$\sum_{x_i \in C_k} \log P(C_{x_i}) = N_k \cdot \log P(C_k)$$

所以可以將上式的$\sum_{i=1}^{N} \log P(C_{x_i})$ 改寫：

$$\begin{aligned}\sum_{i=1}^{N} \log P(C_{x_i}) &= \sum_{x \in C_1} \log P(C_x) + \cdots + \sum_{x \in C_K} \log P(C_x) \\ &= N_1 \cdot \log P(C_1) + \cdots + N_K \cdot \log P(C_K) \\ &= \sum_{k=1}^{K} N_k \cdot \log P(C_k)\end{aligned}$$

原式$\log F$可以改寫為：

$$\begin{aligned}\log F &= \sum_{i=1}^{N} \log P(C_{x_i}) + \sum_{i=1}^{N} \log P(x_i | C_{x_i}) \\ &= \sum_{k=1}^{K} N_k \cdot \log P(C_k) + \sum_{i=1}^{N} \log P(x_i | C_{x_i}) \\ &= \sum_{k=1}^{K} N_k \cdot \log \pi_k + \sum_{i=1}^{N} \log P(x_i | C_{x_i})\end{aligned}$$

令$f = \log F$，則 $\underset{\pi_1, \cdots, \pi_K}{\arg\max} f = \underset{\pi_1, \cdots, \pi_K}{\arg\max} F$

在 $\sum_{k=1}^{K} \pi_k = 1$ 的條件下，欲maximize f，可以使用lagrange multiplier：

令 $g=\sum_{k=1}^{K}\pi_k-1,\ \lambda\in\mathbb{R}^+$，設 $L(\pi_1,\ \cdots,\ \pi_K,\ \lambda)=f+\lambda(g-1)$

對L做偏微分 $\text{w.r.t.}\ \pi_k,\ k=1,\cdots,K$，並求解$\dfrac{\partial L}{\partial \pi_k}=0$

$$
\begin{aligned}
&\frac{\partial L}{\partial \pi_k}=\frac{\partial f}{\partial \pi_k}+\lambda\cdot\frac{\partial g}{\partial \pi_k}=0\\
\Rightarrow\ &\frac{N_k}{\pi_k}+\lambda=0\\
\Rightarrow\ &\pi_k=-\frac{N_k}{\lambda}
\end{aligned}
$$

帶回$\sum_{k=1}^{K}\pi_k=1$：

$$
\begin{aligned}
&\sum_{k=1}^{K}\pi_k=1\\
\Rightarrow\ &\sum_{k=1}^{K}-\frac{N_k}{\lambda}=1\\
\Rightarrow\ &\lambda=-N
\end{aligned}
$$

帶回$\pi_k=-\dfrac{N_k}{\lambda},\ k=1,\cdots,K$，可得：

$$
\pi_k=-\frac{N_k}{\lambda}=\frac{N_k}{N}
$$

2. The property of matrices partial derivative

$$
\frac{\partial}{\partial\sigma_{ij}}\log(\det\Sigma)=\frac{1}{\det\Sigma}\cdot\frac{\partial\det\Sigma}{\partial\sigma_{ij}}
$$

將$\det\Sigma$對第i列做展開，令$[\text{adj}(\Sigma)]_{ji}=C_{ij}$ 為 Σ 的 $(i,\ j)$ 餘因子 (cofactor)：

$$
\begin{aligned}
\frac{\partial}{\partial\sigma_{ij}}\log(\det\Sigma)&=\frac{1}{\det\Sigma}\cdot\frac{\partial\det\Sigma}{\partial\sigma_{ij}}\\
&=\frac{1}{\det\Sigma}\cdot\frac{\partial}{\partial\sigma_{ij}}\cdot(\sigma_{i1}C_{i1}+\cdots+\sigma_{ij}C_{ij}+\cdots+\sigma_{im}C_{im})\\
&=\frac{1}{\det\Sigma}\cdot C_{ij}=\frac{1}{\det\Sigma}\cdot[\text{adj}(\Sigma)]_{ji}=[\frac{1}{\det\Sigma}\cdot\text{adj}(\Sigma)]_{ji}\\
&=[\Sigma^{-1}]_{ji}=\mathbf{e_j}\ \Sigma^{-1}\mathbf{e_i}^T
\end{aligned}
$$

3. Maximum liklihood solution

當log likelihood function有最大值時，liklihood function有最大值。

根據problem 1的結果(參數沿用自problem 1)：

$$
\begin{aligned}
f &= \log P_C(x_1 \cdots x_n) \\
&= \sum_{k=1}^{K} N_k \log \pi_k + \sum_{n=1}^{N} \log P(x_n|C_{x_n}) \\
&= \sum_{k=1}^{K} N_k \log \pi_k + \sum_{k=1}^{K} \sum_{x \in C_k} \log P(x|C_k) \\
&= \sum_{k=1}^{K} N_k \log \pi_k + \sum_{k=1}^{K} \sum_{n=1}^{N} t_{nk} \log P(x_n|C_k)
\end{aligned}
$$

假設某個屬於C_k的x生成自以μ_k, Σ 為參數的Gaussian Distribution

則：

$$
P(x|C_k) = \frac{1}{\sqrt{(2\pi)^m \det \Sigma}} \exp(-\frac{1}{2}(\mu_k - x)^T \Sigma^{-1} (\mu_k - x))
$$

因此，$\log P(x|C_k)$等於：

$$
\log P(x|C_k) = -\frac{1}{2}(\mu_k - x)^T \Sigma^{-1} (\mu_k - x) - \frac{1}{2} \log \det \Sigma - \frac{m}{2} \log 2\pi
$$

而欲求最佳的$\mu_1, \cdots, \mu_K, \Sigma$ 使得f有最大值，我們首先將f對μ_k做偏微分，$k = 1, \cdots, K$

$$
\begin{aligned}
\frac{\partial f}{\partial \mu_k} &= \frac{\partial}{\partial \mu_k} \sum_{k=1}^{K} N_k \log \pi_k + \sum_{k=1}^{K} \sum_{n=1}^{N} t_{nk} \log P(x_n|C_k) \\
&= \frac{\partial}{\partial \mu_k} \sum_{k=1}^{K} \sum_{n=1}^{N} t_{nk} \log P(x_n|C_k) \\
&= \frac{\partial}{\partial \mu_k} \sum_{k=1}^{K} \sum_{n=1}^{N} t_{nk} (-\frac{1}{2}(\mu_k - x_n)^T \Sigma^{-1} (\mu_k - x_n) - \frac{1}{2} \log \det \Sigma - \frac{m}{2} \log 2\pi) \\
&= \frac{\partial}{\partial \mu_k} \sum_{n=1}^{N} t_{nk} (-\frac{1}{2}(\mu_k - x_n)^T \Sigma^{-1} (\mu_k - x_n) - \frac{1}{2} \log \det \Sigma - \frac{m}{2} \log 2\pi) \\
&= \sum_{n=1}^{N} t_{nk} \frac{\partial}{\partial \mu_k} (-\frac{1}{2}(\mu_k - x_n)^T \Sigma^{-1} (\mu_k - x_n) - \frac{1}{2} \log \det \Sigma - \frac{m}{2} \log 2\pi) \\
&= \sum_{n=1}^{N} t_{nk} (-\Sigma^{-1} (\mu_k - x_n)) \\
&= \Sigma^{-1} \sum_{n=1}^{N} (t_{nk} x_n - t_{nk} \mu_k) \\
&= \Sigma^{-1} (\sum_{n=1}^{N} t_{nk} x_n - \mu_k \sum_{n=1}^{N} t_{nk}) \\
&= \Sigma^{-1} (\sum_{n=1}^{N} t_{nk} x_n - \mu_k N_k)
\end{aligned}
$$

求解$\dfrac{\partial f}{\partial \mu_k} = 0$：

$$
\begin{aligned}
&\frac{\partial f}{\partial \mu_k} = 0 \\
\Rightarrow\ &\Sigma^{-1}(\sum_{n=1}^{N} t_{nk} x_n - \mu_k N_k) = 0 \\
\Rightarrow\ &\mu_k = \frac{1}{N_k}\sum_{n=1}^{N} t_{nk} x_n
\end{aligned}
$$

接著，根據problem 2的結論，$\frac{\partial}{\partial \sigma_{ij}}\log(\det \Sigma) = [\Sigma^{-1}]_{ji}$

我們可以一般化成，$\forall\ A \in \mathbb{R}^{n\times n}$：

$$
\frac{\partial}{\partial \Sigma^{-1}}\log(\det \Sigma) = (\Sigma^{-1})^T
$$

重新化簡：

$$
\begin{aligned}
\frac{\partial}{\partial \Sigma^{-1}}\log(\det \Sigma) &= \frac{\partial}{\partial \Sigma^{-1}}\log(\frac{1}{\det \Sigma^{-1}}) \\
&= -\frac{\partial}{\partial \Sigma^{-1}}\log(\det \Sigma^{-1}) \\
&= -((\Sigma^{-1})^{-1})^T = -\Sigma^T = -\Sigma
\end{aligned}
$$

將f對Σ^{-1}做偏微分：

$$
\begin{aligned}
\frac{\partial f}{\partial \Sigma^{-1}} &= \frac{\partial}{\partial \Sigma^{-1}}\sum_{k=1}^{K}\sum_{n=1}^{N} t_{nk}\log P(x_n|C_k) \\
&= \frac{\partial}{\partial \Sigma^{-1}}\sum_{k=1}^{K}\sum_{n=1}^{N} t_{nk}(-\frac{1}{2}(\mu_k - x_n)^T\Sigma^{-1}(\mu_k - x_n) - \frac{1}{2}\log\det\Sigma) \\
&= \sum_{k=1}^{K}\sum_{n=1}^{N} t_{nk}\frac{\partial}{\partial \Sigma^{-1}}(-\frac{1}{2}(\mu_k - x_n)^T\Sigma^{-1}(\mu_k - x_n) - \frac{1}{2}\log\det\Sigma) \\
&= \sum_{k=1}^{K}\sum_{n=1}^{N} t_{nk}(-\frac{1}{2}(\mu_k - x_n)(\mu_k - x_n)^T + \frac{1}{2}\Sigma) \\
&= \frac{1}{2}\sum_{k=1}^{K}(\sum_{n=1}^{N} t_{nk}\Sigma - \sum_{n=1}^{N} t_{nk}(\mu_k - x_n)(\mu_k - x_n)^T) \\
&= \frac{1}{2}\sum_{k=1}^{K}(N_k\Sigma - N_k S_k) \\
&= \frac{1}{2}(\sum_{k=1}^{K} N_k\Sigma - \sum_{k=1}^{K} N_k S_k) \\
&= \frac{1}{2}(N\Sigma - \sum_{k=1}^{K} N_k S_k)
\end{aligned}
$$

求解$\frac{\partial f}{\partial \Sigma^{-1}} = 0$：

$$
\begin{aligned}
&\frac{\partial f}{\partial \Sigma^{-1}} = 0 \\
\Rightarrow\ &\frac{1}{2}(N\Sigma - \sum_{k=1}^{K} N_k S_k) = 0 \\
\Rightarrow\ &N\Sigma = \sum_{k=1}^{K} N_k S_k \\
\Rightarrow\ &\Sigma = \sum_{k=1}^{K}\frac{N_k}{N} S_k
\end{aligned}
$$