# MACHINE LEARNING-Assignment1

尹健璋
*Department of Statistics*
*National Cheng Kung University*
Tainan, Taiwan
R26111052@gs.ncku.edu.tw

## I. INTRODUCTION

本次作業包含手刻Linear Classifier、K-NN Classifier、Naïve Decision Tree和Decision Tree with Pruning四種模型的演算法並透過cross validation來選出最佳模型以及實現一種演算法可以在Linear Classifier和Decision Tree中定義特徵重要性並與SHAP套件做比較。

## II. DATA

本次作業的data來自kaggle公開資料集 https://www.kaggle.com/datasets/ifteshanajnin/carinsuranceclaimprediction-classification?select=train.csv：此為汽車保險索賠的資料集其中包含 58592 筆資料、44 個變項，其中"is_claim"這個變項將作為預測目標，它包含"0"及"1"兩種類別，由於本次資料數很多要執行所有分類器演算法以及其他演算法需要花費很多時間，因此本次作業僅從資料集中抽取25000筆資料來使用，抽取方式為了防止隨機抽取會使抽取到的資料的預測目標類別比例不同，因此隨機抽取時，已取出不放回的方式依照原本預測目標各類別的數量乘以25000/59582並四捨五入至整數的數量去抽取。

預處理部分先將"police_id"刪除，在Linear classifier中，有兩種類別的類別變型特徵會使用one-hot encoding轉換成連續型，而兩個類別以上的類別型特徵則使用frequential encoding做轉換並與連續型特徵一同做標準化；而在手刻的Decision Tree中，則會把所有連續變數轉換為兩類別的類別變數，方法為觀察切在不同值的cross entropy並選出cross entropy最小的切點來對連續型特徵進行分類，而在決策樹的連續型特徵轉換為類別型特徵會在訓練模型時進行。

## III. CLASSIFICATION TASK

### A. Linear Classifier

首先將用來訓練的預測目標$y_i$類別從(0, 1)轉換為(-1, 1)並隨機設定起始權重以及偏誤並建立一個線性分類器：$\hat{y} = \boldsymbol{w^T x} + b$ 其中的 $\boldsymbol{x}$ 為特徵向量、$\hat{y}$為預測結果，若$\boldsymbol{w^T x} + b$ 大於 0 則$\hat{y}$為 1，若$\boldsymbol{w^T x} + b$ 小於等於 0 則$\hat{y}$為-1，接下來將隨機排序後的訓練資料逐一代入預測並判斷其預測結果與正確的結果比較，若預測結果乘上正確結果大於 0，則代入下一筆訓練資料去判斷，若預測結果乘上正確結果小於等於 0，則進入梯度下降法的更新過程，梯度下降法的算法如下：$w_i = w_i + learning\ rate \times x_i \times y_i$ 和$b_i = b_i + y_i$, 對於所有的 $i = 1, \dots, n$, $n$為特徵數，重複上述步驟直到所有訓練資料都訓練過，其中可以設定最大迭代次數（max_iter），設定多少次數，就重複上述步驟幾次，直至分類器於迭代次數內已能將所有訓練資料完全分類或是達到設定的迭代次數就停止訓練，在本次作業中最大迭代次數設定為 100 次。

### B. K-NN Classifier

K-NN 分類器原理不難且並不像其他分類器一樣需要去訓練模型預測，只需將每筆測試資料與所有訓練資料計算距離並依照距離由近到遠來排序，再依照設定的 K 值來決定要選取多少筆最近的訓練資料來預測該樣本資料的預測結果，預測方式為最近的 K 筆訓練資料的已知預測目標多寡來判斷，若 0 較多則測試資料歸類為 0，若 1 較多則歸類為 1，以此方式來預測所有測試資料，其中的距離計算方式在本次作業中以三種距離算法來呈現，分為曼哈頓距離（Manhattan distance）、歐氏距離（Euclidean distance）以及明氏距離（Minkowski distance），明氏距離公式：$D = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}$, 其中 $p = 1$ 為曼哈頓距離, $p = 2$ 為歐氏距離，在本次作業中 K 值的設定為 101。

### C. Naïve Decision Tree

在本次作業中，以二元分類樹來實現決策樹演算法且在訓練中尋找節點與該節點的切分資料點同時進行，再評估節點是否分支是以是否完全分類（purity）來決定，而評估節點優劣以及切分資料點的方法是以 cross entropy 來做判定，執行流程如下：從根節點開始先透過 purity 來評估訓練資料的預測目標是否皆為同一類別，若為同一類別則不分類，若有不同類別則使用 cross entropy 去觀察各特徵，找到切分後 cross entropy 最小的特徵來當作節點並以他的分類結果來繼續分類，而每個節點都同樣先觀察是否完全分類再去做切分以及選擇特徵，直到所有訓練資料完全分類為止。

### D. Decision Tree with Pruning

而剪枝部分則是以預剪枝的方式來處理，以上述的決策樹算法去設定最大深度，也就是說依照設定的最大深度決策樹的分類層數會受到限制，那最大深度要如何設定才會比較好呢？在本次作業中，依照 k-fold cross validation 去觀察不同深度下的平均 error，比較設定不同最大深度結果，找 error 最小的來當作此資料集的最佳深度設定，本次作業中設定的最大深度為 5。

*Result:*

四種分類器對測試資料的預測結果

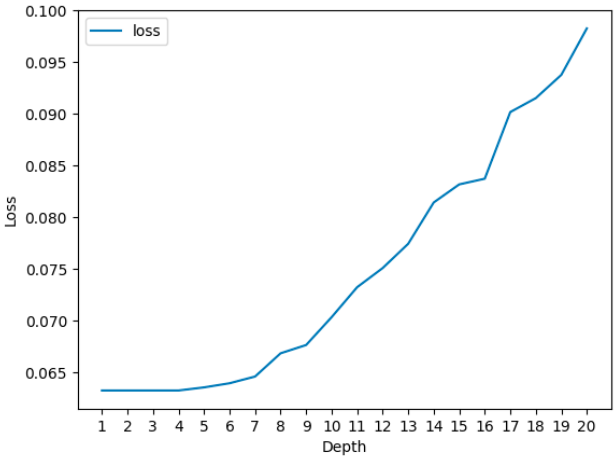
| | *Linear classifier* | *K-NN(p=1)* | *Decision tree* | *Pruning* |
|---|---|---|---|---|
| *accuracy* | *0.9324* | *0.933* | *0.8434* | *0.9102* |

比較手刻的四種分類器預測結果，發現 K-NN 的預測結果相較於其他分類器模型還要來的好，個人認為主要是因為這筆資料的預測目標類別不平衡且還要考慮到模型複雜度，因此才有這種差異，再來是 K-NN 的部分去觀察不同距離公式下的預測結果。

不同距離公式的 K-NN 預測結果

| | 歐氏距離 | 曼哈頓距離 | 明氏距離 |
|---|---|---|---|
| ACCURACY | 0.933 | 0.933 | 0.933 |

不同距離公式卻有相同預測結果，初步推論原因為資料不平衡所導致，因為 K-NN 演算法與其他分類器不同，其演算法並沒有訓練的過程，而是依照訓練資料去推測每筆測試資料，因此 K-NN 相較於其他演算法會更加依賴訓練資料，最後是為何決策樹剪枝設定最大深度為 5，原因是因為透過 5-FOLD CROSS VALIDATION 去計算不同深度的平均誤差並找出能控制誤差不要太大又能減少計算量的深度，用折線圖表示結果如下：




透過這張圖能發現在深度大於 5 後，誤差開始有明顯的提升，因此設定最大深度到 5 相較其他深度來說是更好的選擇。

## IV. FEATURE ENGINEERING

*A. Implement an algorithm that can determine the "feature importance" for both linear classifiers and decision trees. Explain the rationale behind your chosen algorithm.*

關於特徵重要性，在本次作業中使用排列重要性算法，其原理為將模型訓練好後，帶入原始預測資料進行預測並計算準確率，再來將其中一個特徵隨機打亂並重新預測重複此過程 N 次會得到 N 個準確率，再來將收集到的 N 個準確率取平均後與原始測試資料的準確率做比較求絕對誤差，在對每個特徵都執行一次後做最大最小標準化，再去比較各特徵的誤差值由大到小做排序，誤差值越大則越重要，從中去觀察哪些特徵是重要的，本次作業針對線性分類器與決策樹去做觀察，結果如下：

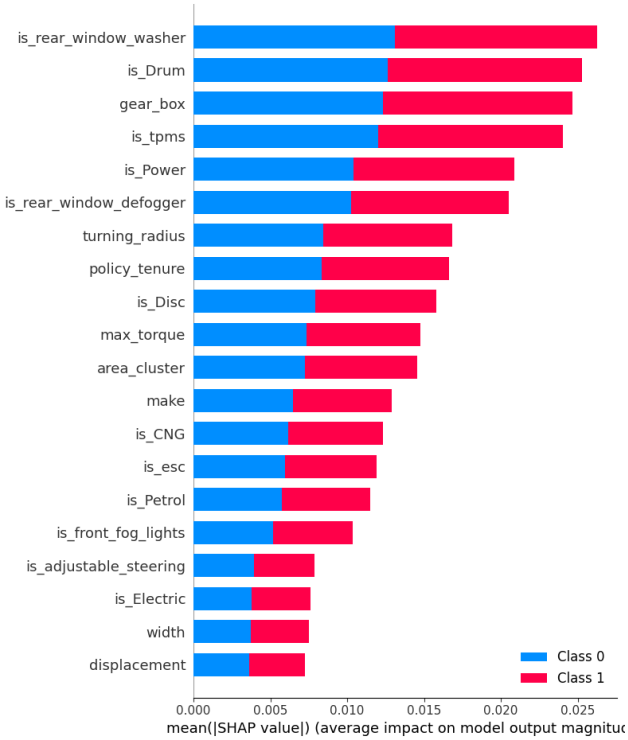
TABLE III.特徵重要性（僅列出排序前 10 個特徵）

| Linear classifier | | Decision tree | |
|---|---|---|---|
| Feature | Error | Feature | Error |
| is_Drum | 1 | policy_tenure | 1 |
| is_CNG | 0.7446 | age_of_car | 0.9133 |
| is_Petrol | 0.3373 | is_adjustable_steering | 0.6608 |
| is_Electric | 0.1174 | age_of_policyholder | 0.4829 |
| population_density | 0.0450 | gross_weight | 0.4530 |
| make | 0.0151 | is_Automatic | 0.3127 |
| gear_box | 0.0140 | ncap_rating | 0.1874 |
| age_of_policyholder | 0.0140 | population_density | 0.0938 |
| age_of_car | 0.0139 | is_Petrol | 0.0727 |
| gross_weight | 0.0137 | cylinder | 0.0612 |

從上表結果中可以發現不同分類器模型會有不同的重要特徵結果，也就是說模型所看重的特徵不同。

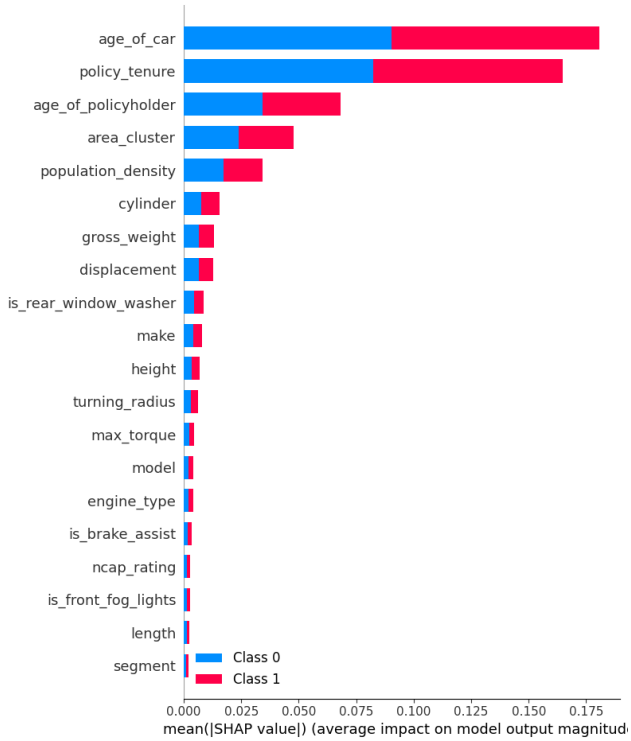
*B. UtilizeSHAP(https://shap.readthedocs.io/en/latest/) you're your implemented algorithm to assess feature importance. Compare your findings with those obtained using SHAP.*

再來是使用 SHAP 套件來觀察特徵重要性，本次作業中僅有線性分類器有成功套用此套件，決策樹部分由於無法套用在手刻的決策樹演算法，所以改由手刻決策樹演算法搭配上述的排列重要性方法去和 sklearn 的決策樹模型搭配 SHAP 套件的特徵重要性做比較，首先是線性分類器的部分，下圖是手刻的線性分器搭配 SHAP 套件所繪出的 bar chart：




比較套件與上表的排列重要性方法，發現兩種找尋特徵重要性方法差異很大，前十個特徵中僅有兩個特徵皆有被兩種方法選中。

再來是決策樹的部分，使用 sklearn 套件決策樹模型搭配 SHAP 套件去找尋特徵重要性，結果如下：




比較手刻決策樹搭配排列重要性方法與sklearn的決策樹搭配SHAP套件結果，發現雖然排序不同，但有四個特徵皆被兩種方法選中，代表說這四個變數對於用這筆資料來訓練的決策樹分類器來說是較重要的，不過由於模型不同，因此這樣的結果可能不夠可靠。

*C. It is known that sometimes the original feature set may not be effective. Designing new features based on the original set is crucial for model performance. Based on your observations and experience, propose an algorithm that can derive new features to enhance model accuracy.*

關於增加新特徵使模型性能提升意指準確率提升，在本次作業中的作法為將資料特徵依照類別型與連續型分成兩部分，針對類別變數透過cramr's V係數去觀察類別特徵之間的相關性並由大到小做排序，而連續特徵部分則是依照pearson correlation來計算並排序，隨後建立兩個新特徵，一個是相關性較高的類別特徵相乘作為其交互作用項，另一個是相關性較高的連續特徵相乘作為其交互作用項，比較線性分類器與決策樹透過5-fold cross validation去觀察新特徵集預測結果是否有比原始特徵集好，以表格呈現如下：

V. 原特徵集與新特徵集預測結果比較

| | LINEAR CLASSIFIER | | DECISION TREE | |
|---|---|---|---|---|
| | ORIGINAL | NEW | ORIGINAL | NEW |
| MEAN ERROR | 0.0632 | 0.0632 | 0.1256 | 0.1253 |

依照線性分類器來看新特徵集與原始特徵集沒有差異，但以決策樹來說新特徵集的平均誤差較低確實有比較好。

## V. CROSS VALIDATION

*Based on **Problem 1**, use k-fold cross-validation to verify the stability of each classifier. Note that cross-validation could adopt any existing package. Answer the questions below:*

*A. Use k=3,5,10, and make some discussions of your observation.*

*Ans:*

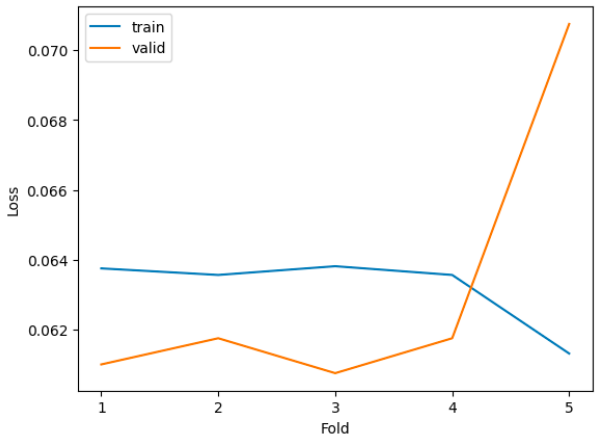
將各分類器在不同的 fold 數下的預測結果如下：

TABLE VI. 各分類器在不同 fold 數下的平均誤差

| | K=3 | K=5 | K=10 |
|---|---|---|---|
| Linear classifier | 0.063206 | 0.063200 | 0.063200 |
| K-NN | 0.063206 | 0.063200 | 0.063200 |
| Decision tree | 0.125213 | 0.125600 | 0.123000 |
| Pruning | 0.063306 | 0.063600 | 0.063500 |

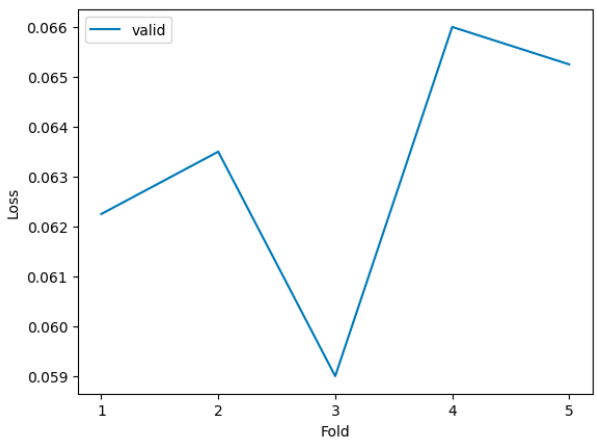
以上表的結果來看，除了決策樹外，其他模型的表現都蠻穩定的，在不同折數下平均誤差結果差異不大，但若是要觀察模型穩定性，則需要觀察各模型在每個 fold 訓練以及驗證的 loss，以下附上在 5-fold cross validation 下各模型的 training/validation loss 折線圖：

FigureI. Linear classifier



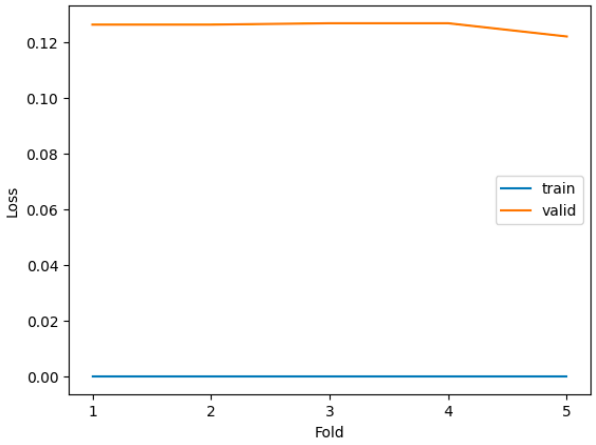

從此圖可以看出線性分類器雖然預測結果不錯，但是模型不穩定。

FigureII. K-NN classifier



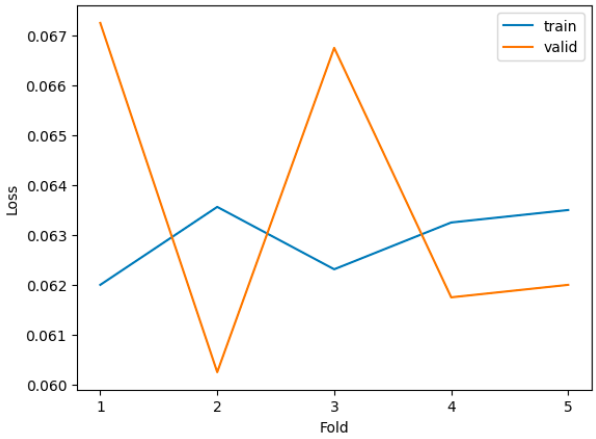
由於 K-NN 並無需訓練，因此無訓練誤差，但可以明顯看到 K-NN 也是相當不穩定。

FigureIII. Decision tree




而未剪枝決策樹雖然平均誤差是四格模型中最高的，但是以穩定性來說，是四個模型之中最好的。

FigureIV. Decision tree with pruning




剪枝後的決策樹變得較不穩定，但是相對也降低平均誤差，可以解釋說捨棄一些穩定性，換取較好的模型性能。

B. *Now you have a test dataset you have partitioned from train.csv. Please design an algorithm that can merge/aggregate the predicted results from k classifiers in k-fold cross-validation. Compare the performance and complexity of the cross-validation with Problem 1.*

*Ans:*

此題選擇以未剪枝的決策樹搭配 5-fold cross validation 合併預測結果方法來做比較。

TABLE VII. 預測結果比較

| | accuracy | complexity |
|---|---|---|
| Problem1 | 0.8434 | 66.126 |
| 5-fold cv | 0.8858 | 243.924 |

首先先介紹本次作業所使用的方法，方法為將在每次 fold 中，使用分割出來的訓練集去訓練模型並去預測測試集，接著將每個 fold 的預測結果依照投票機制整合，舉例來說假如第一筆測試資料在 5-fold 的預測結果為 1, 1, 0, 1, 0，則因為 1 最多，所以該筆測試資料的預測結果為 1，依照這個方式對所有測試資料做整合，計算出準確率有 0.8858、運行時間花費 243.924423 秒，相比將所有訓練資料去做訓練後的模型預測出來的準確率有 0.8434、運行時間為 66.12634 秒，以預測結果來說比 problem1 結果還要來得好，但在複雜度上需要付出的成本也會相對提高，也就是說透過整合 k-fold 的預測結果雖然預測的結果比較好但同時也會使複雜度提升。

C. *How do we know the performance of one model is really better than another one? Please compare the result in 5-fold cross-validation and the result of Problem 1 to justify which is "REALLY" better. Also show me why.*

*Ans:*

由於模型穩定性不好，因此要比較模型好壞只依訓練單一模型來評估可能比較沒有那麼可靠，因此我認為透過 k-fold cross validation 去評估哪個模型比較好會比較合適；5-fold cross validation 跟 problem 1 比較結果，依照上表（TABLE VII）結果來看，發現交叉驗證預測結果比 problem1 好，畢竟透過訓練多個模型同時對測試資料做預測再投票決定相較於僅由一個模型來決定會更加合適。