資料探勘期末個人報告

組員：羅天宏

一、簡介

語言：

python 2.7 / 3.6

工具：

numpy

scipy

scikit-learn

matplotlib

分析：

Classification

使用方法：

SVM

評估方法：

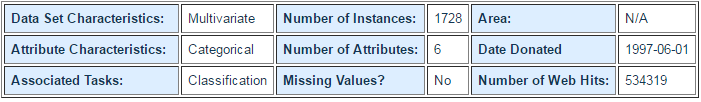
Mean Square Error

Accuracy Score

Confusion Matrix

Classification Report(Precision、Recall、F1-Score、Support)

二、資料前處理：



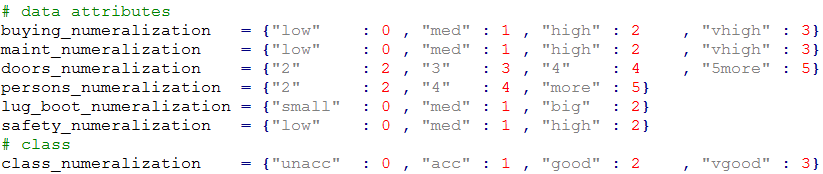
屬性值：

|  |  |
| --- | --- |
| buying | v-high, high, med, low |
| maint | v-high, high, med, low |
| doors | 2, 3, 4, 5-more |
| persons | 2, 4, more |
| lug\_boot | small, med, big |
| safety | low, med, high |

類別分佈：

|  |  |  |
| --- | --- | --- |
| class | N | N[%] |
| unacc | 1210 | 70.023 % |
| acc | 384 | 22.222 % |
| good | 69 | 3.993 % |
| v-good | 65 | 3.762 % |

數值化：



i.e.

Key = 原來資料格式

Value = 轉換後的資料格式

Random shuffle：

Seed = 5000

資料分割：

Training Set:40%

Testing Set:60%

設計實驗方式：

1.調整Training set比例，觀察不同Kernel下的表現結果

2.調整前處理方式，觀察對於SVM的分類影響

3.增加Noise，觀察SVM的Robustness

4. 2和3的條件組合情況

預設執行結果：

參數設定：

C=1.0,

cache\_size=200,

class\_weight=None,

coef0=0.0,

decision\_function\_shape='ovo',

degree=3,

gamma='auto',

kernel='rbf',

max\_iter=-1,

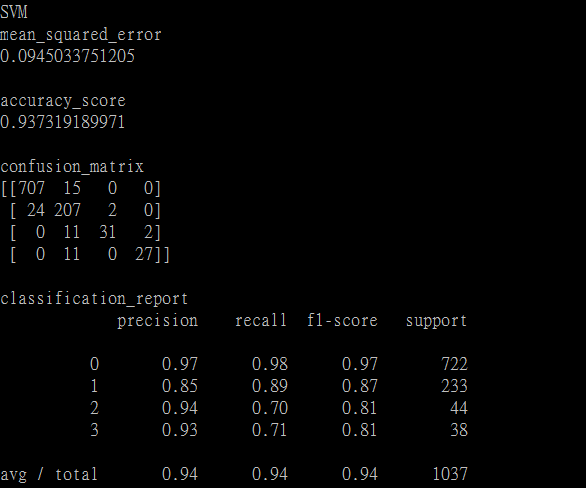
probability=False,

random\_state=False,

shrinking=True,

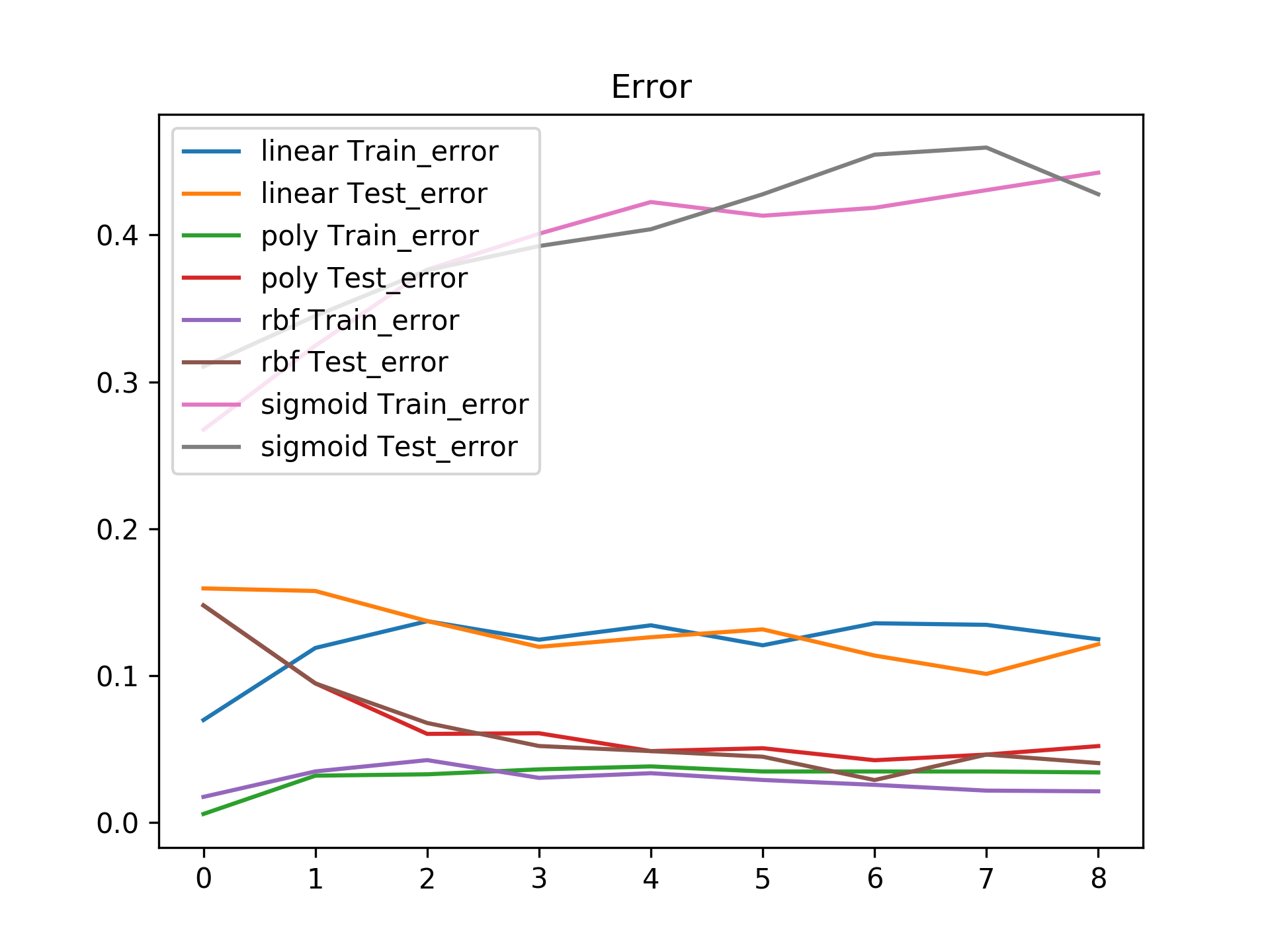
tol=0.001,

verbose=False



實驗設計

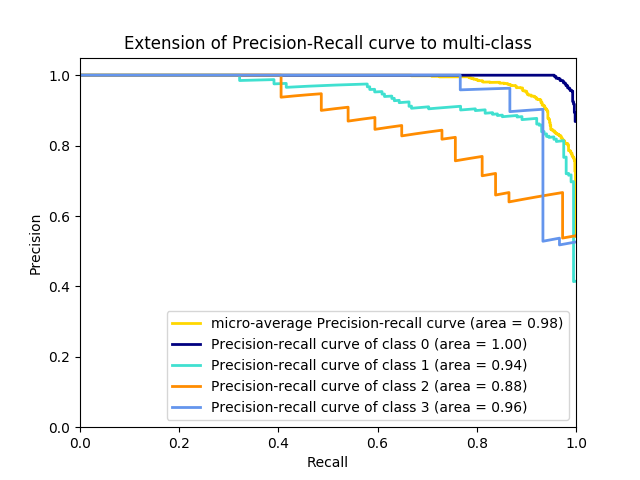
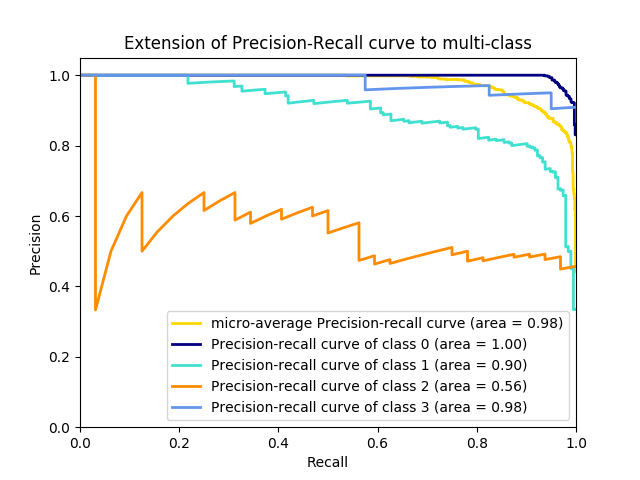
1.調整Training set比例，觀察不同Kernel下的表現結果

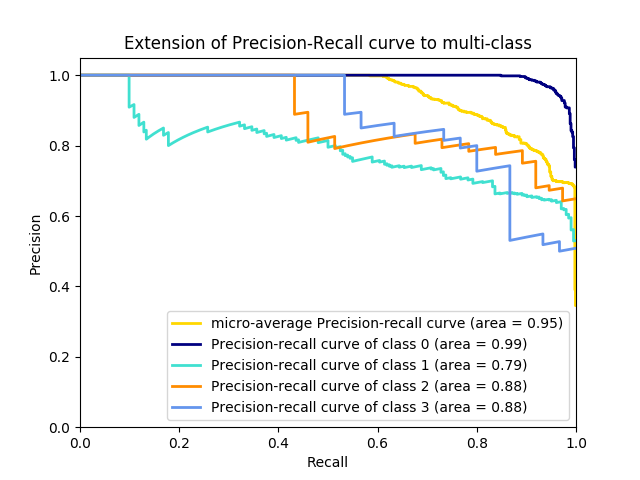
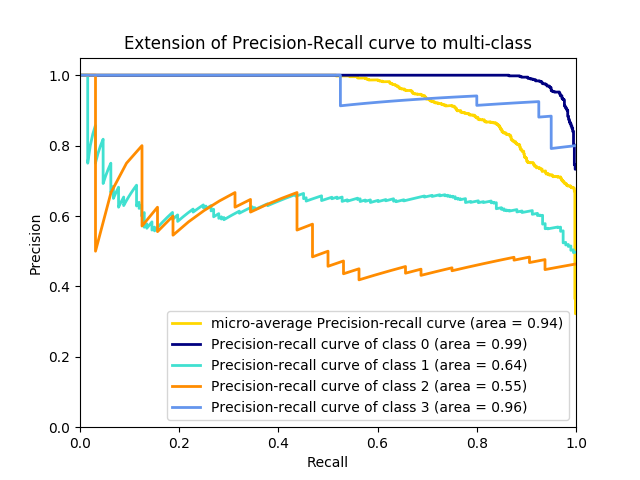


　　設計實驗的方式，主要是以使用不同的Kernel function來做SVM，而系統內建的Kernel function為’linear’、'poly'、'sigmoid'、’rbf’，其餘參數沒有更動太多。經過一輪的測試之後，最低的accuracy為0.7，kernel function為sigmoid，而目前實驗呈現的0.93是最好的結果，以kernel function為rbf。以confusion matrix來看，資料上學習最好的比例為Training set:0.5、Testing set 0.5。因此後續的實驗皆採用Training Set：0.5、Testing Set，Kernel : RBF來做。

2.調整前處理方式，觀察對於SVM的分類影響

由左至右，由上至下，分別，未調整、Shuffle、Normalization、Shuffle + Normalization





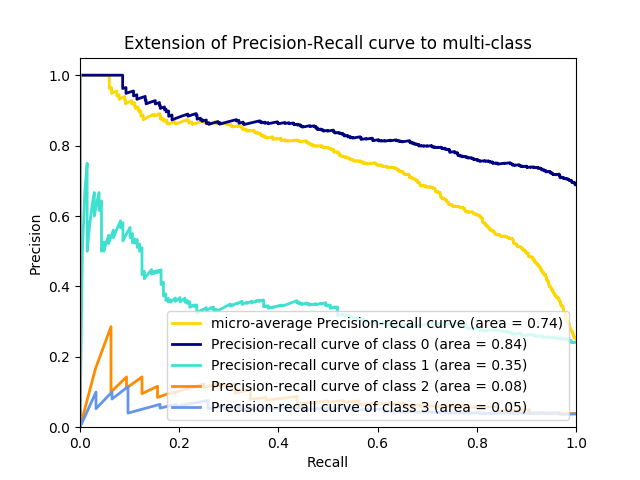
在前處理的部分，主要設計兩種實驗的方式，加入shuffle、normalization，從上圖中可以觀察到有經過shuffle後各class的AUC面積大於原先未經過shuffle。可以認為，原先較少分布的class 2和3在經過shuffle後，比較能被放進Training Set裡面，被SVM學到特徵，而原本未經過shuffle的圖則是在class 2底下表現不佳，應是原先資料的分布裡面，Trainging Set並沒有太多class 2的data。

在Normalization的部分，我們可以觀察到class 1 和class 2的表現皆變得比較差，這裡的解釋為原來資料的featrue設計是有階層式的關係，所以當我們將每個Attribute的範圍都mapping到0和1之間時，反而會降低學習的效果。

最後是shuffle+normalization，與單單只做nomalization比較起來，AUC的表現較好，但與只做shuffle比較起來，AUC的表現又變得比較差。因此在這兩個前處理的方式下，我得到的結論是，Shuffle會讓原來不對稱的分布變得較均勻一些，Normallization雖然會讓每個資料的比重變得一樣，但卻會失去原來階層式的比重關係，反而會讓效果變差。

3.增加Noise，觀察SVM的Robustness

Noise

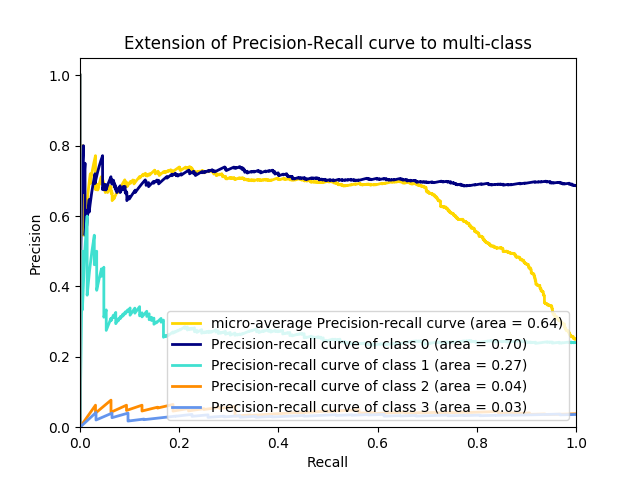
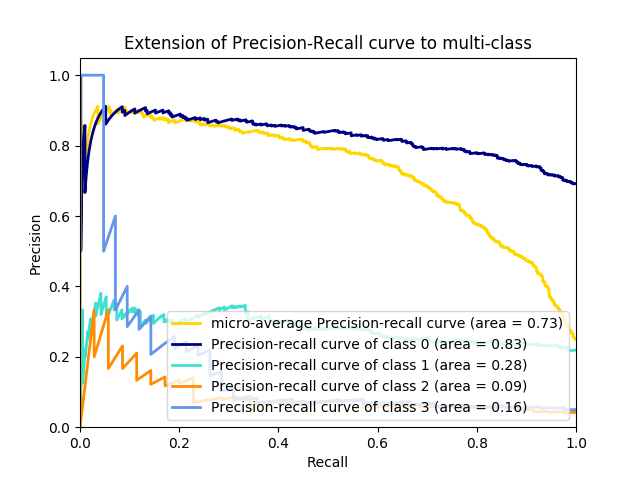


這個實驗主要是觀察加入Noise後，SVM對於錯誤資料的Robustness。因此實驗設計是在毒入資料後，在原先的資料裡面插入200個錯誤資料，觀察SVM的AUC變化。從實驗的結果來看，SVM下降的正確率很多。

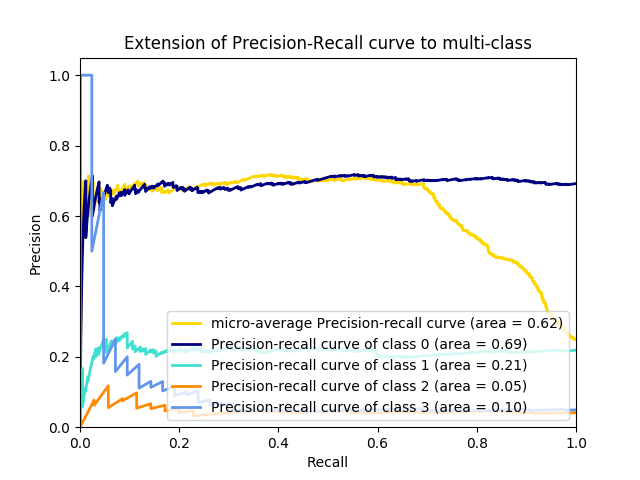
有趣的是，對於低頻(數量很多)的class 0，我加入了最多錯誤的資料，然而卻沒有下降太多。而資料量本身比較少的class 1、2、3，影響AUC的情況就變得非常明顯。結論是對於錯誤的資料很敏感，因此SVM並不具有很好的Robustness。

4. 2和3的條件組合情況

由左至右，由上至下，分別為Shuffle + Noise、Normalization + Noise



Shuffle + Normalization + Noise



從實驗結果可以看出，不論是哪一種情況，AUC的表現都非常差，但我觀察到還是有些不同，shuffle的表現仍然比Normalization的實驗狀況好。