Machine Learning HW#3

0510822 陳紀愷

Problem 1. Cross Entropy, KL divergence and Logistic Regression

證明 $\mathrm{H}\big(p(x),q(x)\big)=D_{KL}(p(x)||q(x))+\mathrm{H}\big(p(x)\big)$

由於

$$\mathrm{H}\big(p(x),q(x)\big)=\sum_{x}p(x)\log\frac{1}{q(x)}$$

$$=-\sum_{x}p(x)\log q(x)$$

加上

$$D_{KL}(p(x)||q(x))=\sum_{x}p(x)\log\frac{p(x)}{q(x)}$$

可以藉此推導出

$$D_{KL}(p(x)||q(x))=\sum_{x}p(x)\log\frac{p(x)}{q(x)}$$

$$=\sum_{x}(p(x)\log p(x)-p(x)\log q(x))$$

$$=-\mathrm{H}\big(p(x)\big)-\sum_{x}p(x)\log q(x)$$

$$=-\mathrm{H}\big(p(x)\big)+\mathrm{H}\big(p(x),q(x)\big)$$

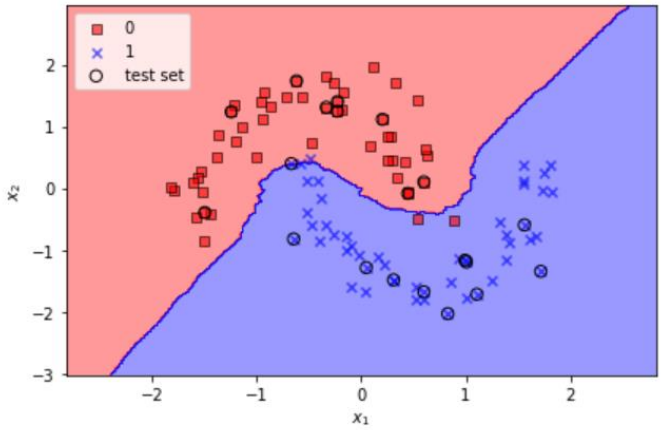
移項以後可得證

$$\mathrm{H}\big(p(x),q(x)\big)=D_{KL}(p(x)||q(x))+\mathrm{H}\big(p(x)\big)$$

Problem 2: Python code Exercise

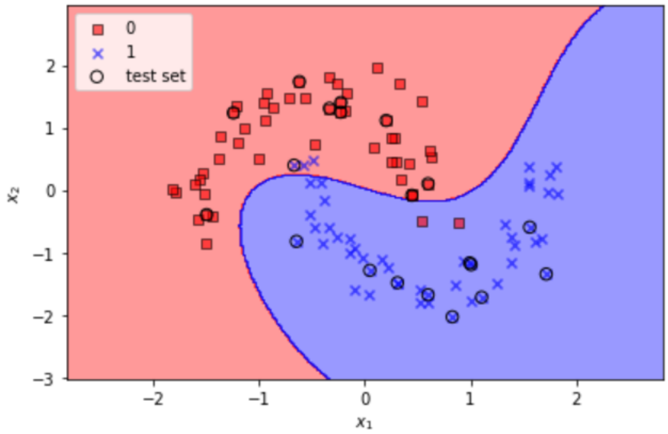
(a) KNN classification Please implement a KNN classifier in scikit-learn using a **Euclidean distance metric** where K = 11.

```
[KNN]
Misclassified samples: 1
Accuracy: 0.95
```



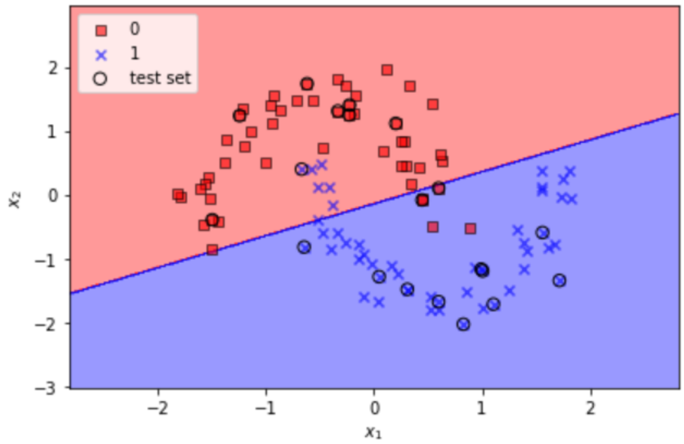
(b) SVM classifier Please implement a SVM classifier in scikit-learn using **'rbf'** kernel where random state = 0, γ = 0.2, and C = 10.0.

```
[SVM]
Misclassified samples: 1
Accuracy: 0.95
```



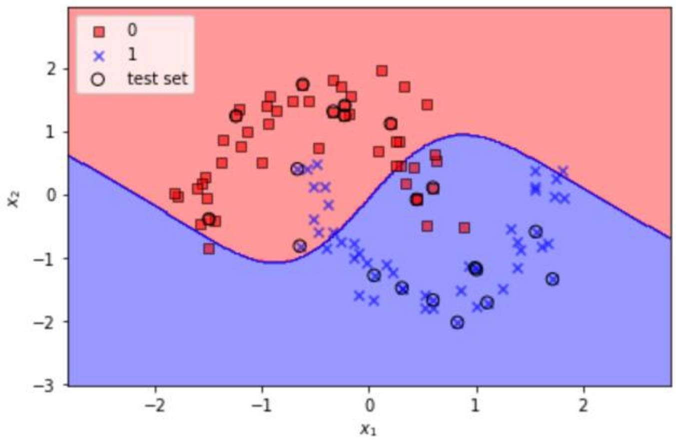
(c) SVM classifier Please implement a SVM classifier in scikit-learn using **'linear'** kernel where C = 1000.0 and random state = 0.

```
[SVM]
Misclassified samples: 3
Accuracy: 0.85
```



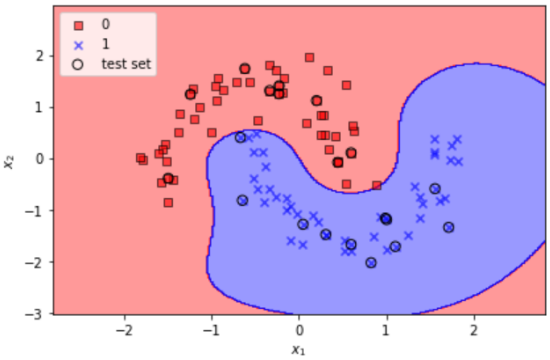
(d) SVM classifier Please implement a SVM classifier in scikit-learn using **'sigmoid'** kernel.

```
[SVM]
Misclassified samples: 4
Accuracy: 0.80
```

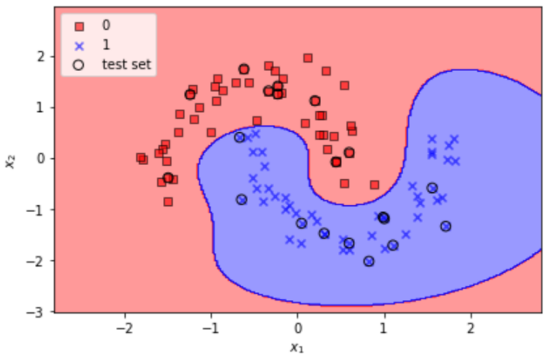


(e) Given C ∈ {0.1,1.0,10.0,100.0,1000.0,10000.0} and γ ∈ {0.00001,0.0001,0.001,0.01,0.1,1.0}, please find the best combination of (C,γ) for default SVM classifier with random state = 0 in scikit-learn.

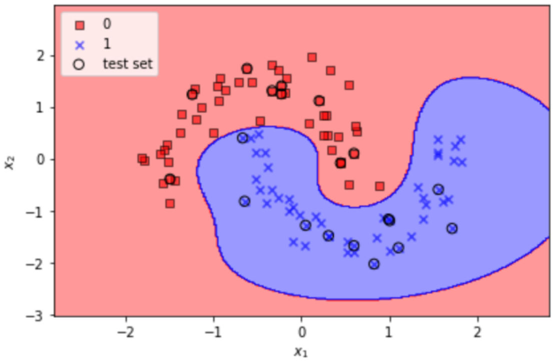
```
------------------------------------------------------------
C = 1.000000 gamma = 1.000000
Misclassified samples: 0
Accuracy: 1.00
```



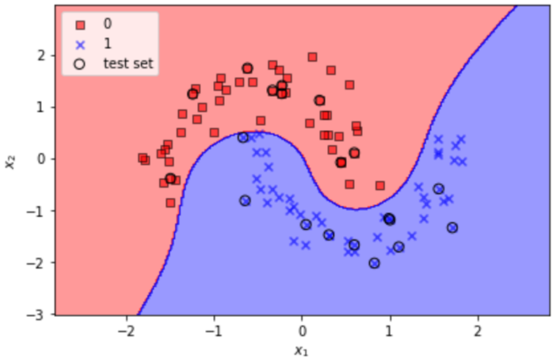

```
------------------------------------------------------------
C = 10.000000 gamma = 1.000000
Misclassified samples: 0
Accuracy: 1.00
```



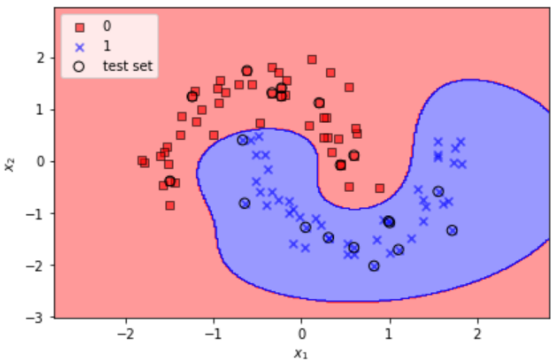

```
------------------------------------------------------------
C = 100.000000 gamma = 1.000000
Misclassified samples: 0
Accuracy: 1.00
```



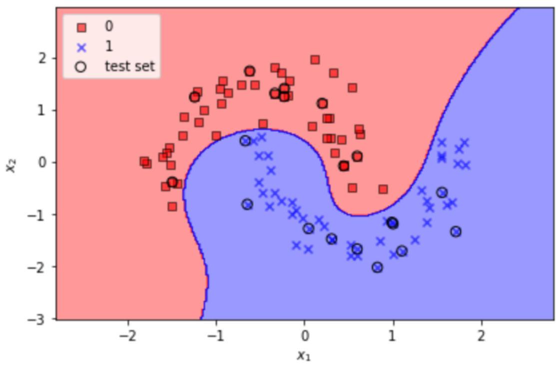

```
------------------------------------------------------------
C = 1000.000000 gamma = 0.100000
Misclassified samples: 0
Accuracy: 1.00
```



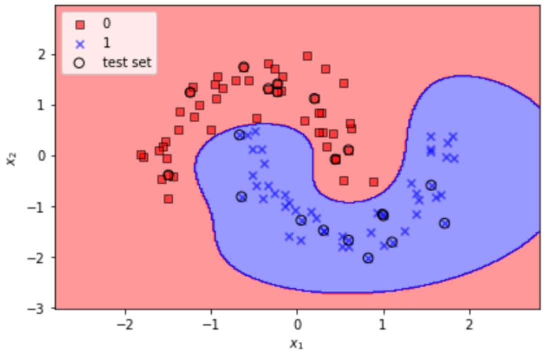

```
------------------------------------------------------------
C = 1000.000000 gamma = 1.000000
Misclassified samples: 0
Accuracy: 1.00
```




```
------------------------------------------------------------
C = 10000.000000 gamma = 0.100000
Misclassified samples: 0
Accuracy: 1.00
```



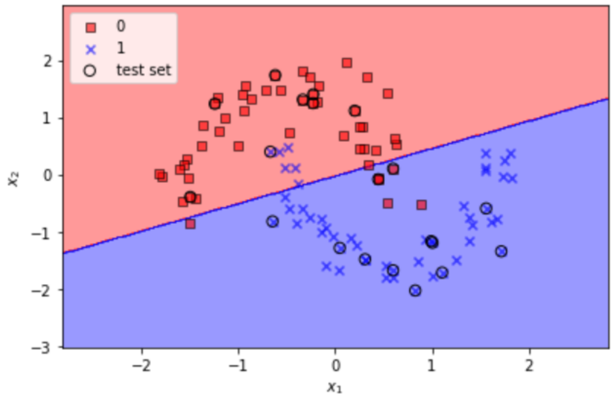
```
--------------------------------------------------------
C = 10000.000000 gamma = 1.000000
Misclassified samples: 0
Accuracy: 1.00
```



當{C, γ}={1,1},{10,1},{100,1},{1000,1},{10000,1},{1000,0.1},{10000,0.1},時，會有最佳解

(f) Logistic Regression Please implement Logistic Regression in scikit-learn where C = 1000.0, random state = 0, and solver = ”**liblinear**”.

```
[LogisticRegression]
Misclassified samples: 3
Accuracy: 0.85
```



Problem 3: Loss functions comparison

(a) Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\|y_i - \hat{y_i}\|_2^2$$

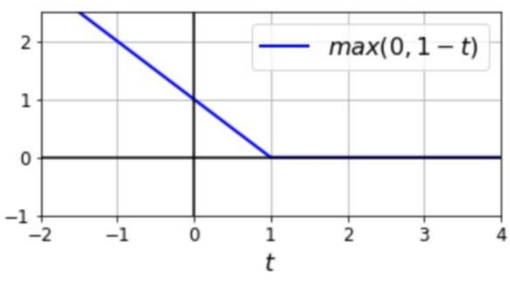
由於 MSE 是把實際值和預測值兩者相減以後取平方和，所以如果有異常的值時，他會被平方倍的放大，這也會造成誤差變的很大，因此魯棒性較差。但是在另一方面而言，平方反而可以使損失的梯度較為靈敏，在損失的大的時候梯度大，反之則梯度小，使得在梯度學習時可以較準確的求出解。

(b) Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\|y_i - \hat{y_i}\|$$

不同於 MSE,MAE 直接把實際值和預測值兩者相減以後的絕對值相加，所以在異常值造成的影響較小，因此魯棒性也較好。但相對來說 MAE 對損失的梯度也較不靈敏，因此在做梯度學習時需要謹慎的調整學習率，否則沒辦法迅速的求出解。

(c) Hinge Loss for SVM



以右圖為例，Hinge Loss 的定義是在當 t>=1 時結果為 0，而 t<1 時會隨著距離會有線性增加的值，

在 SVM 中則用他來表示預測結果和實際的差距，當預測出來的結果θTx>=1 時 x 的預測結果應該是 y=1，與真正答案相符，此時損失就是 0，如果不一樣的話則會像圖一樣，差距越多會造成越大的 loss。

(d) Cross Entropy Loss

交叉熵 loss 通常用在分類的訓練中，它的優點是他的 loss 很接近線性變化，因此受到異常點的影響較小，而且他是連續可微分的，因此較好使用推導。

Problem 4: Kernel Ridge Regression and Soft Margin SVM

利用 Kernel Ridge Regression 做分類，也就是 Least-Squares SVM，Least-Squares SVM 和 Soft Margin SVM 相比起來他們做創造出來的邊界會非常相近，只是 Kernel Ridge Regression 的 Support vector 比較多，所以運算速度較慢