HW5 ECE498 Spring 2020

Netid :- GOWTHAM4

Name:- **Gowtham Kuntumalla**

Problem1 :- → Jupyter Notebook has been submitted online.

→ It contains codes for SVM, DecisionTree, Randomforest classifier and 5-fold cross validation results (sklearn)

1) (Mean Accuracy (μ), S.D Accuracy (σ))

SVM		DT		RF	
C=0.1	(97.77%, 0.00204)	d=3	(97.74%, 0.00288)	N=5	(97.73%, 0.00184)
C=1	(97.871%, 0.0023)	d=4	(97.89%, 0.00222)	N=11	(97.782%, 0.00217)
C=10	(97.88%, 0.00235)	d=6	(97.73%, 0.002898)	N=13	(97.79%, 0.00175)

(mean Precision (μ), S.D Precision (σ))

SVM		DT		RF	
C=0.1	(94.28%, 0.01996)	d=3	(89.74%, 0.0371)	N=5	(93.66%, 0.01756)
C=1	(94.23%, 0.0200)	d=4	(91.91%, 0.0296)	N=11	(93.39%, 0.0235)
C=10	(94.051%, 0.02081)	d=6	(90.54%, 0.03978)	N=13	(93.23%, 0.0247)

(mean Recall, s.D Recall)

SVM (μ)		DT (σ)		RF	
C=0.1	(80.66%, 0.0376)	d=3	(85.29%, 0.0303)	N=5	(80.72%, 0.0354)
C=1	(81.88%, 0.0396)	d=4	(84.62%, 0.0333)	N=11	(81.69%, 0.0378)
C=10	(82.12%, 0.0391)	d=6	(84.26%, 0.0311)	N=13	(81.99%, 0.0426)

2) Label imbalances cause accuracy to be high. Even if most are predicted to of class 0 (not pulsar), accuracy will be > 90%. Recall, Precision take TP into account.

See Jupyter Notebook

3) Best classifier :- SVM C=10, we used F1_score weighted → takes care of label imbalance

---

Problem 2 :- Activation function $g(x) = \frac{1}{1+e^{-x}}$ (sigmoid function)

$g'(x) = g(x)(1-g(x))$

1) $z_1 = w_1 \cdot x_1 + b_1$ (vector dot products)

$z_4 = w_4 \cdot a_1 + w_5 \cdot a_2 + w_6 \cdot a_3 + b_4$

$a_1 = \frac{1}{1+e^{-z_1}}$

$a_4 = \frac{1}{1+e^{-z_4}}$

Output $= \hat{y} = a_4$

$a_4 = \frac{1}{1+e^{-z_4}} = \frac{1}{1+e^{-\sum_{i,j} w_i \cdot a_j + b_4}}$

$\Rightarrow \hat{y} = a_4 = \frac{1}{1+e^{-[w_4 \cdot (\frac{1}{1+e^{-w_1 x_1 + b_1}}) + w_5 \cdot (\frac{1}{1+e^{-w_2 x_1 + b_2}}) + w_6 \cdot (\frac{1}{1+e^{-w_3 x_1 + b_3}}) + b_4]}}$

Note :- $g(x) = \frac{1}{1+e^{-x}}$:- Sigmoid function

True label ↑ Predicted label ↑

2) Loss, $L_i = (y_i - \hat{y}_i)^2$

$$\frac{\partial L_i}{\partial b_4} = 2(y_i - \hat{y}_i) \frac{\partial \hat{y}_i}{\partial b_4}$$

$$\hat{y}_i = a_4 = g(z_4)$$

$$\Rightarrow \frac{\partial \hat{y}_i}{\partial b_4} = g(z_4)(1-g(z_4)) \frac{\partial z_4}{\partial b_4}, \quad \frac{\partial z_4}{\partial b_4} = 1$$

$$\Rightarrow \boxed{\frac{\partial L_i}{\partial b_4} = 2(y_i - \hat{y}_i)\, g(z_4)(1-g(z_4))}$$

Similarly

$$\frac{\partial L_i}{\partial w_4} = \frac{\partial \hat{y}_i}{\partial w_4} 2(y_i - \hat{y}_i)$$

also, $\frac{\partial \hat{y}_i}{\partial w_4} = g(z_4)(1-g(z_4)) \frac{\partial z_4}{\partial w_4}, \quad \frac{\partial z_4}{\partial w_4} = a_1$

$$\Rightarrow \boxed{\frac{\partial L_i}{\partial w_4} = 2(y_i - \hat{y}_i)\, g(z_4)(1-g(z_4))\, a_1}$$

$$\frac{\partial L_i}{\partial b_1} = \frac{\partial L_i}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_4} \frac{\partial z_4}{\partial a_4} \frac{\partial a_4}{\partial z_1} \frac{\partial z_1}{\partial b_1}$$

$$\Rightarrow \boxed{\frac{\partial L_i}{\partial b_1} = 2(y_i - \hat{y}_i)\left[g(z_4)(1-g(z_4))\right] w_4 \left[g(z_1)(1-g(z_1))\right]}$$

$$\frac{\partial L_i}{\partial w_1} = \frac{\partial L_i}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_4} \frac{\partial z_4}{\partial a_1} \frac{\partial a_1}{\partial z_1} \frac{\partial z_1}{\partial w_1}$$

$$\Rightarrow \boxed{\frac{\partial L_i}{\partial w_1} = 2(y_i - \hat{y}_i)\left[g(z_4)(1-g(z_4))\right] w_4 \left[g(z_1)(1-g(z_1))\right] x_1}$$

3) Gradient descent step :-

$$w_1' = w_1 - \eta \frac{\partial L_i}{\partial w_1} \quad \text{(Single } x_i\text{)}$$

$$\Rightarrow \boxed{w_1' = w_1 - \eta \, \% \, L}$$

4) Gradient descent multiple samples (n)

$$L = \frac{1}{n} \sum_{i=1}^{n} L_i = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$\frac{\partial L}{\partial w_1} = \sum_{i=1}^{n} \frac{\partial L_i}{\partial w_1}$$

$$\Rightarrow \boxed{w_1' = w_1 - \frac{\eta}{n} \sum_{i=1}^{n} \frac{\partial L_i}{\partial w_1}} \rightarrow \text{given by } q2.2.$$

5) Sigmoid vs ReLU Activation functions

Ref:- Stackexchange, 1262 Wikipedia entries

Advantages of ReLU over Sigmoid

1) Gradient is easier to calculate and is 1 when $a > 1$; if $ReLU(a) = max(0, a)$

2) Sparsity, when $a \leq 0$ the resulting computation is sparse because function outputs zero.

Sparsity functions better in many cases than dense representations (common in sigmoid)

Advantages of sigmoid over ReLU

Sigmoid

1) Activation produces output in the range $(0, 1)$ → easy to store. Doesn't blow up output of neurons.

2) It is better than ReLU in some cases where large number of neurons die (ex:- multiple -ve values in input)