CSE 574

Assignment 3 Report
Submitted by

Group 76

Paritosh Kumar Velalam (pvelalam) (50295537)

Sampreeth Boddi Reddy (sboddire) (50298591)

Abhinay Reddy Alluri (aalluri) (50288936)

LOGISTIC REGRESSION:

Algorithm	Accuracy		
	Training data	Validation data	Test Data
Logistic Regression	92.756%	91.42%	92.02%

Table 1.1

Logistic Regression considers all the Data points to construct the hyperplane separating the classes. This works good for linearly separable Data.

As Data given to us is not linearly separable as it has high dimensions (high number of input features) so Logistic regression does not provide high accuracy on our Data sets.

From the results of training and test accuracies, we can infer that the model is somewhat over fitted to training data. Hence the accuracy on test data is lower than training data.

MULTICLASS-LOGISTIC-REGRESSION:

Algorithm	Accuracy		
	Training data	Validation data	Test Data
MULTICLASS-LOGISTIC-REGRESSION	93.326%	92.45%	92.49000000000001%

Table 1.2

Multiclass Logistic Regression accuracy is higher than Binary Logistic regression because in multiclass logistic regression we perform the classification by dividing the data into individual classes class1, class2 and so on and compare the individual class probabilities. Where as in binary logistic regression we are computing class1 vs rest classes, class2 vs rest classes, and so on.

SVM:

SVM vs LOGISTIC REGRESSION:

Kernel	Training data	Validation data	Test Data
SVM Linear Kernel	97.286%	93.64%	93.78%
Logistic Regression	92.756%	91.42%	92.02%

Table 1.3

logistic regression considers all the points in the dataset for finding a hyperplane that separates our data, Whereas, SVM works with the samples which are close to each other and close to the margin and tries to provide the maximum possible margin.

As we can see from Above table that SVM performs better than LOGISTIC REGRESSION on all the 3 Data Sets.

SVM performs better due to following reasons:

- 1.SVM selects the hyperplane with maximum margin which provides good efficiency while classing the data, whereas LOGISTIC REGRESSION selects any arbitrary hyperplane
- 2. SVM works well with large number of input features in our case around 716 so accuracy is more whereas LOGISTIC REGRESSION works well with the small number of input features.
- 3. LOGISTIC REGRESSION works well when there is low confidence in the Data (it does not assume data is enough to give a final decision). In our data each pixel represents some feature of the output so SVM performs better
- 4. Logistic loss diverges faster than hinge loss. So, in general, it will be more sensitive to outliers.
- 5. Logistic loss does not go to zero even if the point is classified sufficiently confidently. This might lead to minor degradation in accuracy

SVM WITH DIFEERENT KERNALS:

Kernel		Accuracy			
		Training data	Validation data	Test Data	Time Taken
SVM	Linear Kernel	97.286%	93.64%	93.78%	1401.0746126174927 seconds
	RBF, gamma=1	100.0%	15.479999999999999%	17.14%	25652.32101249695 seconds
	RBF, default gamma	94.294%	94.02000000000001%	94.42%	2724.88703417778 seconds

Table 1.4

In General, we select the kernel based on below facts:

- Use linear kernel when number of features is larger than number of observations.
- Use Gaussian kernel when number of observations is larger than number of features.
- If number of observations are more Gaussian kernel takes more time as observed in the above table

SVM gives higher accuracy when using a Gaussian kernel, that maps the input features into an infinite dimensional space that is (squared exponential kernel defines a function space that is a lot larger than that of the linear kernel)

INFLUENCE OF GAMMA:

The hyper parameter γ controls the tradeoff between error due to bias and variance in your model

the gamma parameter is the inverse of the standard deviation of the RBF kernel (Gaussian function), which is used as similarity measure between two points.

If gamma is too large, the radius of the area of influence of the support vectors only includes the support vector itself and no amount of regularization with C will be able to prevent overfitting.

If gamma is very small, the model is too constrained and cannot capture the complexity or “shape” of the data. The region of influence of any selected support vector would include the whole training set.

As shown in above table gamma=1 shows that when gamma is too high, we have overfitting and very low accuracy on both validation and test set, despite getting 100% of accuracy on the training set.

When gamma is 0 (actually close to 0 - SVC sets it to 1 / number of features), we have much better results on validation and test set.

INFLUENCE OF C:

Algorithm		Accuracy			
		Training Data	Validation data	Test Data	Time Taken
SVM	RBF, C=1	94.294%	94.02000000000001%	94.42%	1472.6988787651062 seconds
	RBF, C=10	97.13199999999999%	96.17999999999999%	96.1%	885.3572208881378 seconds
	RBF, C=20	97.952%	96.89999999999999%	96.67%	781.6771605014801 seconds
	RBF, C=30	98.372%	97.1%	97.04%	772.0698752403259 seconds
	RBF, C=40	98.706%	97.23%	97.19%	764.4114787578583 seconds
	RBF, C=50	99.002%	97.31%	97.19%	727.0168123245239 seconds
	RBF, C=60	99.196%	97.38%	97.16%	1020.8625028133392 seconds
	RBF, C=70	99.33999999999999%	97.36%	97.26%	1290.6616926193237 seconds
	RBF, C=80	99.438%	97.39%	97.33000000000001%	1287.7770221233368 seconds
	RBF, C=90	99.542%	97.36%	97.34%	1282.2570791244507 seconds
	RBF, C=100	99.612%	97.41%	97.39999999999999%	1282.8138377666473 seconds

Table 1.5

C controls the cost of misclassification on the training data.

Our goal in SVM is to find a hyperplane that would give maximum margin between input points from two classes. There is a tradeoff between "less margin / no mistakes" and "more margin, few mistakes".

- **Small C makes the cost of misclassification low (**A small C gives you higher bias and lower variance.**).**

- **Large C makes the cost of misclassification high (thus making our Model potentially over fit,** a large C gives you low bias and high variance**)**

Graph:

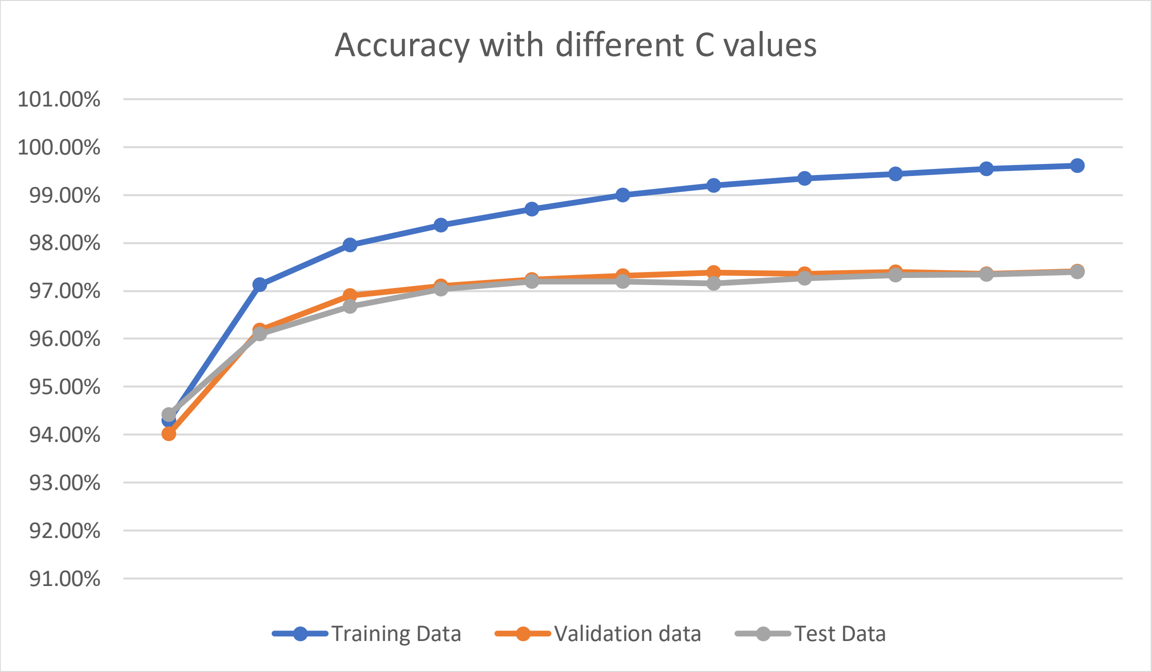

Fig:1.1

When C is low, the weight of each error term is low as well: therefore, a larger error value is accepted during the training phase. Consequently, a larger margin hyperplane is created in the expense of having more samples misclassified.

Conversely, when C increases, the weight of each error term increases as well, so lower error values will be accepted. As a result, a smaller margin hyperplane will be built, but less number of points will be misclassified, so the accuracy of data classification increases.

Based on this explanation, there is a high risk of overfitting for larger values of C we can see how the accuracy on the training set increases considerably when C grows, while the accuracy on validation and test set are same almost as shown in **Fig:1.1**.
So based on the requirement (tradeoff between "less margin / no mistakes) we choose C.