Portfolio Assessment I

Data Classification (Part 2)

Report Prepared By: Hardik Rathod (18070609)

1. Introduction

In this report, we are going to extend the part 1 by dividing the class label dataset into training, validation and test set. We will classify the datapoint by logistic regression method and talk about the cost function, optimisation algorithm and regularisation term. We also try to predict the label from features data and learn about the accuracy rate, learning curve and PCA affect the performance of learning.

2. Description

Task 1 : Splitting Data

First most thing I had done is to Divide the data in features and label. Here we are going to predict class in further task so class will be label as Label_values and other parameter of data will be the features as target_values.

Using train_test_split from sklearn.model_selection Dividing the data set into a Training set (I) and a test set in 80:20 which is 20% of 1599 data points which will be 1279 in training and 320 in the test. Further, the training set (II) is divided into training and validation set in 25:75 which is 20% of whole data, After this validation set contain 320 and validation will contain 959 data points.

Task 2 : PCA Analysis

Principal Component Analysis (PCA) is a method to reduce set of observation by orthogonal transformation with possibly correlated variables into a set of linear uncorrelated variable.In simple terms it reduce the sets of feature in N number of principal component which define the true value of all features.

First I scale the training set by StandardScaler().fit_transform(X_train) which is used from from sklearn.preprocessing import StandardScaler library to scaler_train_set variable.
then, I calculated the PCA from from sklearn.decomposition import PCA giving argument of n_components = 2. So , by these I got the PC1 and PC2 which describe the all the features value.

Secondly, I find out the variance and ratio of variance.Variance ratio defines the amount of data contain in the particular principal component.

Task 3 : Do a classification using the logistic regression model

A. Cost Function:

Given a data(X,Y), X being a matrix of values with m examples and n features and Y being a vector with m examples. The objective is to train the model to predict which class the future values belong to. Primarily, we create a weight matrix with random initialisation. Then we multiply it by features.

$$a = w_0 + w_1 x_1 + w_2 x_2 + \ldots w_n x_n$$

We then pass the output obtained from Eq 1. to a link function.

$$\hat{yi} = 1/(1 + e^{-a})$$

This is followed by calculating the cost for that iteration whose formula is

Logistic regression cost function

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m} \underbrace{\boxed{y^{(i)}\log h_\theta(x^{(i)})}}_{0 \quad \text{If actual y=0}} + \overbrace{\boxed{(1-y^{(i)})\log(1-h_\theta(x^{(i)}))]}}^{0 \quad \text{If actual y=1}}\right.$$

$$\boxed{\begin{array}{l} J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)}) \\ \mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1-h_\theta(x)) & \text{if } y = 0 \end{cases} \\ \text{Note: } y = 0 \text{ or } 1 \text{ always} \end{array}}$$

Optimisation and Regularisation of Logistic Classification:

SGD Classifier and Regressor are used to optimise the logistic regression model by adding a loss = "log" to a SDG function.

Penalised logistic regression imposes a penalty to the logistic model for having too many variables. This results in shrinking the coefficients of the less contributive variables toward zero. This is also known as **regularisation**. Defaults to 'l2' which is the standard regulariser for linear SVM models. 'l1' and 'elastic-net' might bring sparsity to the model (feature selection) not achievable with 'l2'.

```
optimize = SGDRegressor(max_iter=100, tol=1e-3, eta0=0.1 ,penalty = "l2")
optimize.fit(scaled_trnX, y_train)

# print the coefficients
print('The intercept =',optimize.intercept_)
print('The trained coefficients are:',optimize.coef_)

# predict for the test set
y_pred_SDG = optimize.predict(scaled_tstX)
```

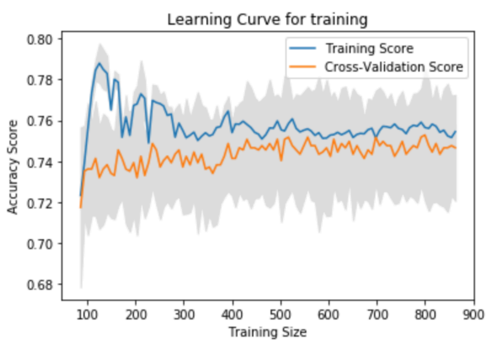
B. ([num1, num2]=misPatterns(predictions, labels))

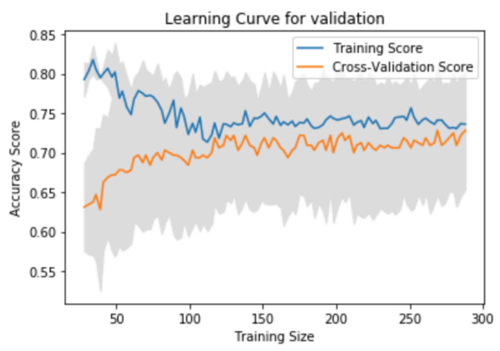
```
def misPatterns(y_pred,y_test):
    count=0;num1=0;num2=0;
    for i in range(y_test.shape[0]):
        if(y_pred[i]!=y_test[i]):
            if(y_pred[i]==2):
                num1+=1
                print('predected  ',y_pred[i],'  Label',y_test[i],'num1')
            if(y_pred[i]==1):
                num2+=1
                print('predected  ',y_pred[i],'  Label',y_test[i],'num2')
        else:
            print('predected  ',y_pred[i],'  Label',y_test[i],'True')

        count+= 1
    return [num1,num2]
```

Answer of misPatterns Function is [42, 48]

Task 4: Investigate how the size of the training dataset affects the model performance on the test set

A. Accuracy rate perform on the validation and training set(II)





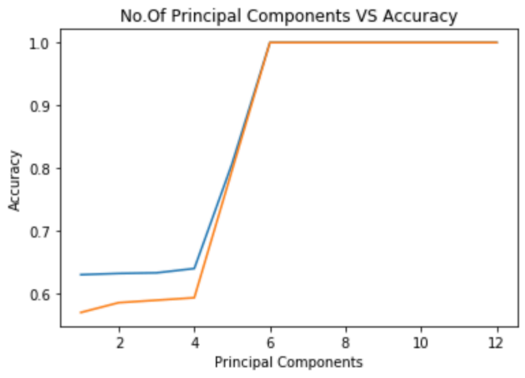
B. The Best training dataset size we are getting is at 400 ie 33.33% of validation set. At these point we get the stable cross validation as well as consistence training score

C. Accuracy at 25% of validation and 75% of training set(II) give us test_set_accuracy: 0.7146464646464646

Task 5: Investigate how the number of features extracted from PCA affects the model performance on the test set

Feature selection is a process where you automatically select those features in your data that contribute most to the prediction variable or output in which you are interested.
Having irrelevant features in your data can decrease the accuracy of many models, especially linear algorithms like linear and logistic regression.
Three benefits of performing feature selection before modelling your data are:

- **Reduces Overfitting**: Less redundant data means less opportunity to make decisions based on noise.
- **Improves Accuracy**: Less misleading data means modelling accuracy improves.
- **Reduces Training Time**: Less data means that algorithms train faster.



By visualising graph, we can say that 6 minimum component is required to get best accuracy

Conclusion :

Hence, It can be concluded that the some principal component with high variance cover most of the data so accuracy don’t depend on the number of components but also the selection of component.The set data for test and train are determined with the learning curve. These analysis are made with considering the regression logistic functions which generates the analysis of red wine quality. The overall performance which has been used to get attained will be cent percent regarding the accuracy.

Reference :

1. Sun, Yi. (2019). Units. [online] Herts.instructure.com. Available at: https://herts.instructure.com/courses/51019/modules [Accessed 18 Dec. 2019].
2. GitHub. (2019). laxmimerit/Learning-Curve-Machine-Learning-in-Python-KGP-Talkie. [online] Available at: https://github.com/laxmimerit/Learning-Curve-Machine-Learning-in-Python-KGP-Talkie/blob/master/Learning%20Curve%20Machine%20Learning%20in%20Python%20KGP%20Talkie.ipynb [Accessed 18 Dec. 2019].
3. Scikit-learn.org. (2019). sklearn.preprocessing.StandardScaler — scikit-learn 0.22 documentation. [online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html [Accessed 18 Dec. 2019].
4. VanderPlas, J. (2019). Python Data Science Handbook | Python Data Science Handbook. [online] Jakevdp.github.io. Available at: https://jakevdp.github.io/PythonDataScienceHandbook/ [Accessed 18 Dec. 2019].