CT 4101 Machine Learning Assignment 1 Report

Name – Tapan Auti **Class –** MSc Computer Science (Artificial Intelligence) -1MAI1 **ID –** 20231499

Package Selection

For the assignment I researched about many different open-source packages available, the major thought process was not to select the best one out there but to select the one that best suits the given dataset. After a thorough reviewing of the same, I decided to use Scikit-Learn for the assignment.

Scikit-Learn is built on top of matplotlib, NumPy, and SciPy libraries and it is mainly used for data mining and analysis, but the main thing was that it is very simple and easy to use along with a lot of machine learning models already featured. Compared to Scikit-Learn, there were other packages which were more efficient and had lot of features but to use such heavy packages for such a small classification problem was not justifiable.

Scikit-Learn is very accessible and simple and for beginners like me it is very easy to understand, also it is good to use when the data set is comparatively very small.

It has various features to support machine learning like feature extraction, cross-validation along with dimension reduction etc. We can say that it is very versatile in nature.

Preprocessing

The data set given were in text format so before giving data to algorithm I converted the data into csv format and added label to the data. All this was done with help of pandas library.

The algorithms that I used for the dataset are K Nearest Neighbor and Support Vector Machine. For both these algorithms the predictor requires data to have significantly different ranges so scaling the data will help in mitigating the different features, also if the data is in measured format, scaling will reduce the similar measured features.

For the assignment I am using the standard scaler from preprocessing to standardize the data, this will transform all data to a mean of 0 with standard deviation of 1.

Before preprocessing my data was like below: -

[[45.30530973 0.45954818 1.91727273 4.22769231 16.67 12.56894737 11.04 62.17857143] [43.88938053 0.54897701 3.18636364 4.28923077 16.73 14.974 13.44 63.03285714] [41.58849558 0.54284696 1.56818182 4.34461538 16.48 11.84878947 14.04 63.46857143] [44.55309735 0.48030092 1.87181818 4.42461539 18.59 13.87963158 12.48 63.53142857]

After preprocessing it looked like: -

[[1.40333663e+00 1.02068221e+00 6.56823465e-02 9.18706025e-01 -6.27193497e-01 1.06677522e+00 -9.93259943e-02 -8.50307003e-01] [8.61501491e-01 1.68563412e+00 1.49423824e+00 1.16536521e+00 -5.73097512e-01 1.77287347e+00 7.48328741e-01 -7.04006595e-01] [-1.89806144e-02 1.64005387e+00 -3.27272857e-01 1.38735849e+00 -7.98497447e-01 8.55344408e-01 9.60242424e-01 -6.29388493e-01] [1.11548672e+00 1.17499008e+00 1.45163048e-02 1.70801543e+00 1.10387800e+00 1.45157837e+00 4.09266847e-01 -6.18623915e-01]

Algorithms Used

K Nearest Neighbors -

It assumes that similar things exist in proximity, based on this KNN choses the data points and calculates the distance between the nearest data points and forms a new datapoint and then matches it to the training data to compare the proximity, based on this it recognizes similarities.

Implementation steps: -

STEP 1 – Load the data from the set into training and testing

STEP 2 - Choose value of k to set nearest datapoints.

STEP 3 – For each example calculate distance between sample and current.

STEP 4 - Based on the distance sort them in ascending order

STEP 5 – Pick the first k records and get their labels and return the mode.

In this way a KNN algorithm works and finds the most suitable class a value belongs based on the data points.

Support Vector Machine -

SVM is fast and performs very well with limited amount of data.

SVM takes labeled data and plots it in a hyperplane and based on the outputs separates the data points and classifies the similar data.

We can say that SVM is a representation of different classes in a multidimensional hyperplane which first generates the hyperplane that segregates the classes and then chooses the plane that separates the classes correctly. This uses a subset of training points so is in turn memory efficient.

Predictions

KNN: -

(k =10 throughout)

For training set an accuracy of 97 was achieved for a data of 124 examples for which the confusion matrix is as follows:

Confusion Matrix [[42 0 0]

[3 41 0]

[0 1 37]]

And for testing set an accuracy of 96.67 was achieved for a data of 30 examples whose confusion matrix is as follows:

	true ale	true lager	true stout	class precision
pred. ale	10	1	0	90.91%
pred. lager	0	9	0	100.00%
pred. stout	0	0	10	100.00%
class recall	100.00%	90.00%	100.00%	

From the above table we can clearly state that only one example was predicted wrong the model prediction was precise.

The classification report is as below:

	Precision	recall	f1-score	support
ale	1.00	0.90	0.95	10
lager	0.91	1.00	0.95	10
stout	1.00	1.00	1.00	10

SVM: -

As the dataset is polynomial nature the kernel was set to ‘poly’ and the degree was taken as 3.

For the training set an accuracy of 99 was achieved for data of 124 examples for which the confusion matrix is as below:

Confusion Matrix [[42 0 0]

[0 44 0]

[1 0 37]]

And for testing set an accuracy of 93.3 was achieved for data of 30 examples whose confusion matrix is as below.

	true ale	true lager	true stout	class precision
pred. ale	10	1	0	90.91%
pred. lager	0	9	1	90.00%
pred. stout	0	0	9	100.00%
class recall	100.00%	90.00%	90.00%	

From the above data we can state that the precision was less in lager and stout class.

The classification report is as below:

	Precision	recall	f1-score	support
ale	1.00	0.80	0.89	10
lager	0.91	1.00	0.95	10
stout	0.91	1.00	0.95	10

Conclusion

From the results we can conclude that KNN gives better accuracy than SVM.

As the problem was of low dimension KNN proved to be better, also to get the proper output we need to tune SVM based on the cost C etc which was not done in this case, which also might have affected the accuracy of SVM.

References:

1. https://scikit-learn.org/
2. https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/
3. https://medium.com/
4. https://www.researchgate.net/
5. https://www.upgrad.com/blog/scikit-learn-in-python/
6. https://opensource.com/article/18/5/top-8-open-source-ai-technologies-machine-learning
7. https://www.dataquest.io/blog/sci-kit-learn-tutorial/
8. https://www.activestate.com/blog/top-10-python-machine-learning-packages/
9. https://www.geeksforgeeks.org/data-preprocessing-machine-learning-python/
10. https://www.datacamp.com/community/tutorials/preprocessing-in-data-science-part-1-centering-scaling-and-knn
11. https://analyticsindiamag.com/7-types-classification-algorithms/
12. https://www.tutorialspoint.com/machine_learning_with_python
13. https://vas3k.com/blog/machine_learning/
14. https://www.udemy.com/course/machinelearning