Assignment Report - Machine Learning

Btech - 5th Sem

Aditya Aggarwal - IIT2019210

10-10-2021

Assignment on Dimensionality Reduction with several techniques

Google Colab Link for Codebase

CLICK HERE

Problem 1

Method 1: From Iris flower data set take only two features (sepal width and petal length). Project the labelled data on a 2D graph. Try to classify this data by drawing linear boundaries intuitively.

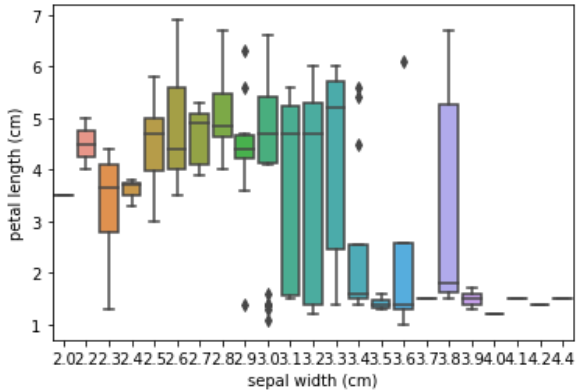
Method 2 : Now take all the features and apply MDA on this Iris flower data set and plot the labelled data on 2D graph.

Part 3 : Also show the plots of above two methods and write in your own words why method2 works better than method1

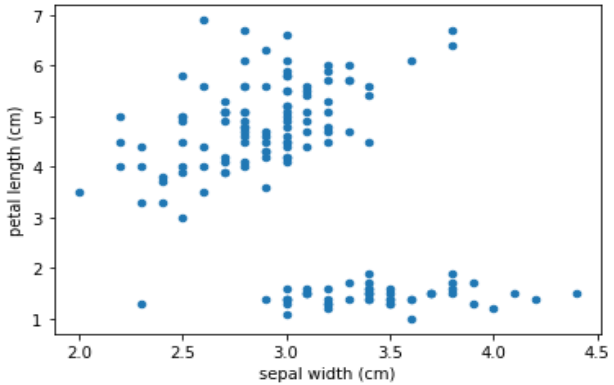
Approach for Method 1

1. Load Iris Dataset from sklearn library.
2. Plot Box graph and Scatter Matrix(with x as sepal width and y as petal length) that shows that there are outliers in the data and data is highly coupled.

```
In [7]: # 3. I am doing this by using box plot
        sns.boxplot(data=data, x="sepal width (cm)", y="petal length (cm)")
Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x7fafc5d9b2d0>
```

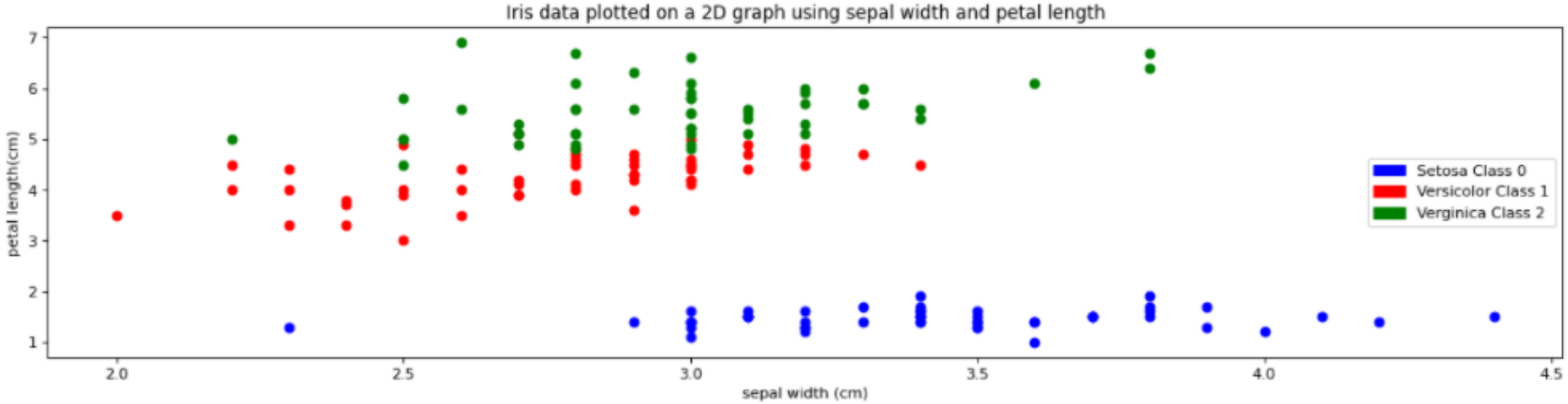



```
In [9]: # 5. I am doing this by using scatter matrix
        data.plot.scatter(x="sepal width (cm)",y="petal length (cm)")
Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0x7fafbd1c29d0>
```



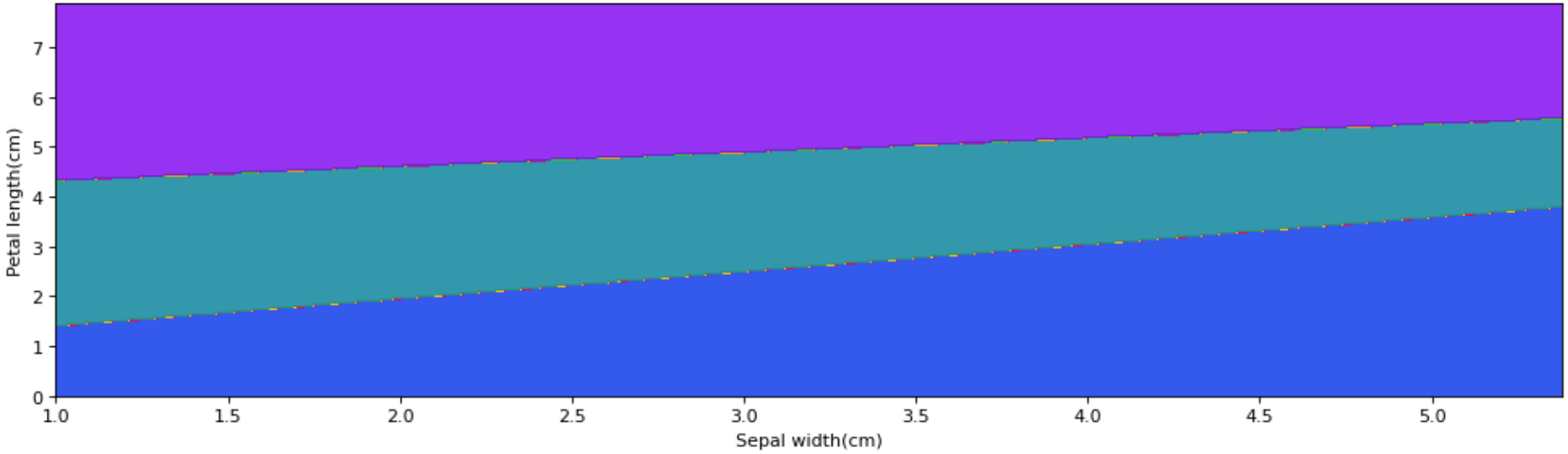
Approach for Method 1 (cont.)

3. Extract the sepal width and petal length from the data in X and target column in y.
4. Now, plot the 2d graph by using scatter matrix with X[0] and X[1] and y. The output will be same as below.



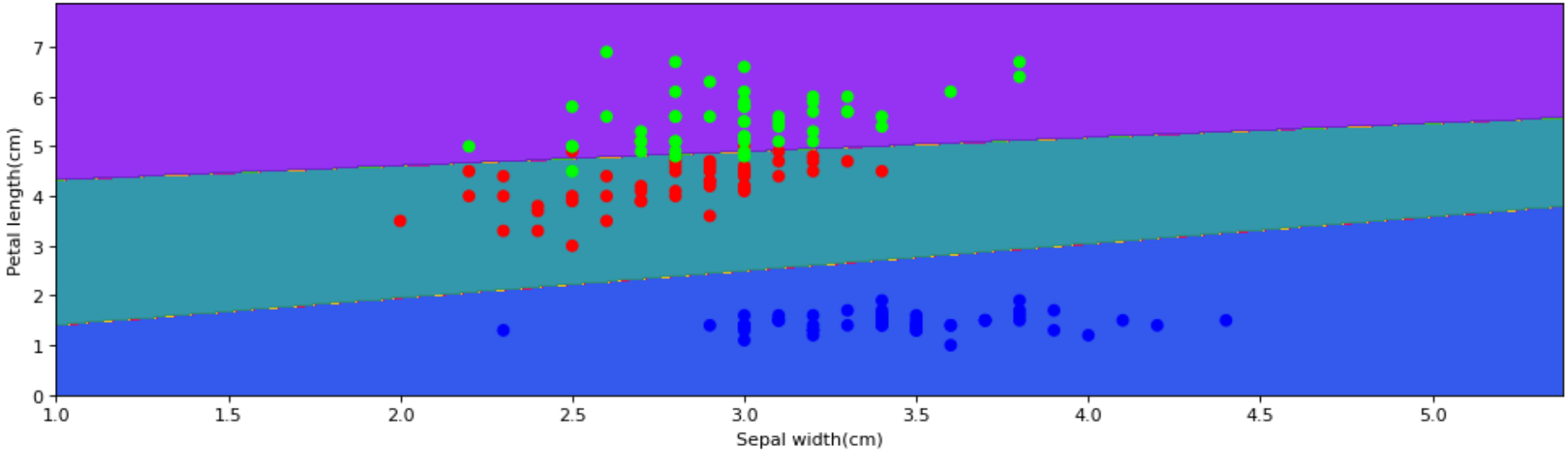
Approach for Method 1 (cont.)

5. Now, to create linear boundaries in the data, I am using SVM with a linear kernel to do that. For that we have to create SVM model with linear kernel.
6. We have the data points as in previous slide and we will plot the boundary using SVM model. The following plot will be seen after plotting the linear boundary.



Approach for Method 1 (cont.)

Observation: After plotting both the points and linear boundary, I can observe there that class 2 is almost separated from class 0 and class 1 (>95%) but the class 0 and class 1 have some points which goes on the wrong side. This is because of the outliers that we saw in the box graph.



Approach for Method 2

1. Load Iris Dataset from sklearn library.
2. This time consider all the feature columns and store them into X and target column into y.
3. Now, split the data into training and testing with train-test ratio as 66%.
4. We have to apply MDA model on iris dataset. So, to do that we will apply LDA on multiple features and it becomes MDA.
5. For that, create a new LDA model and train the model with training data. Now, check its accuracy. I have attached the screenshot below for the same.

```
Actual test output values are : [0.0, 0.0, 0.0, 2.0, 1.0, 1.0, 2.0, 2.0, 1.0, 2.0, 0.0, 2.0, 1.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0,
1.0, 2.0, 0.0, 0.0, 0.0, 2.0, 2.0, 1.0, 2.0, 0.0, 1.0, 2.0, 1.0, 2.0, 2.0, 2.0, 2.0, 1.0, 2.0, 1.0, 2.0, 2.0, 2.0, 0.0, 1.0, 2.
0, 1.0, 1.0, 1.0, 1.0, 1.0]
Predicted test output values are : [0.0, 0.0, 0.0, 2.0, 1.0, 1.0, 2.0, 2.0, 1.0, 2.0, 0.0, 2.0, 1.0, 1.0, 0.0, 1.0, 0.0, 0.0,
0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 2.0, 2.0, 2.0, 2.0, 0.0, 1.0, 2.0, 1.0, 2.0, 2.0, 2.0, 2.0, 1.0, 2.0, 1.0, 2.0, 2.0, 2.0, 0.0, 1.
0, 2.0, 1.0, 1.0, 1.0, 1.0, 1.0]
```

We can see almost all the classes predicted by the LDA model are in sync with (similar) with the actual class.

```
# Now, I am predicting the accuracy score of the model.
accuracy_score(y_test, lda.predict(X_test))
```

```
0.96
```

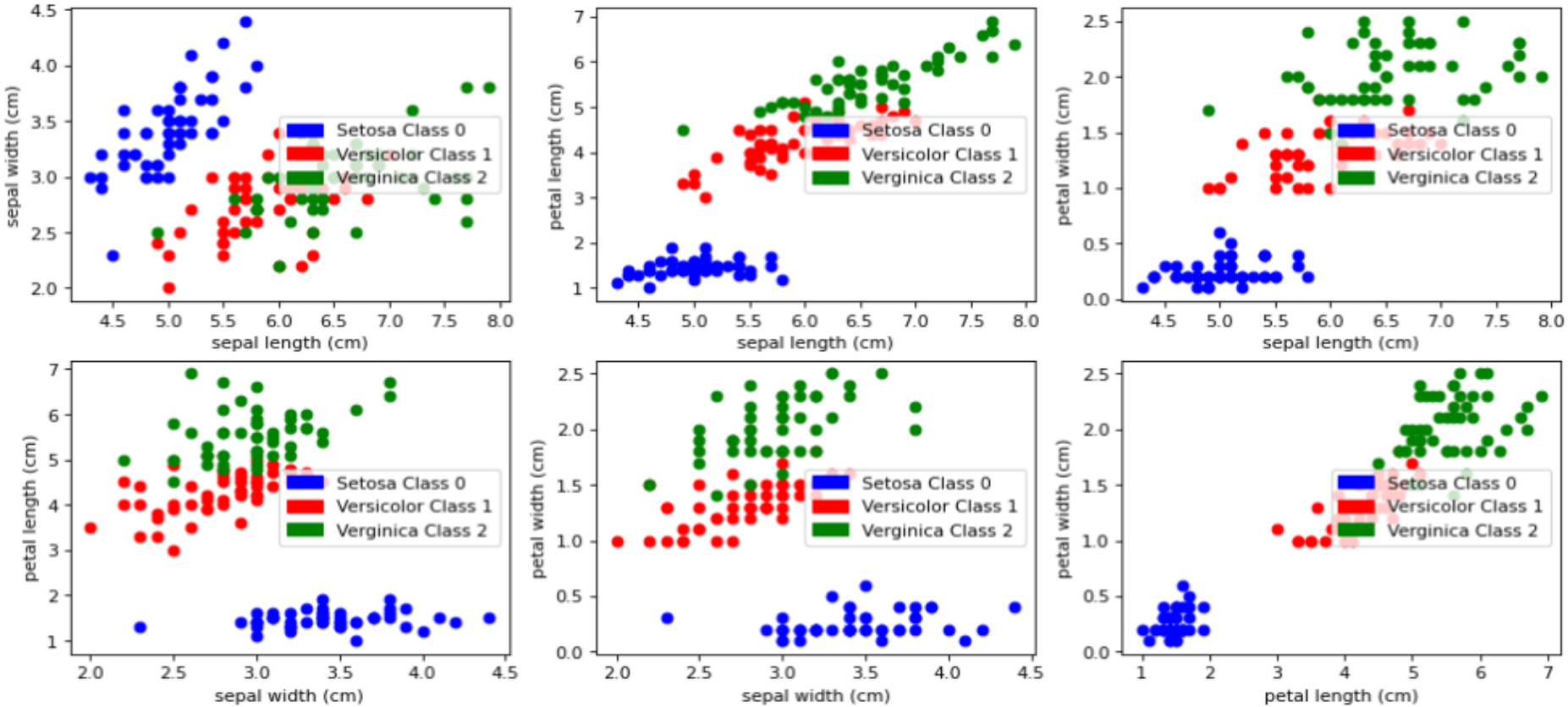
Approach for Method 2 (cont.)

Summary for this part: We got 96% accuracy which is good. Even I would not get this much accuracy in the method 1 by using SVM model.

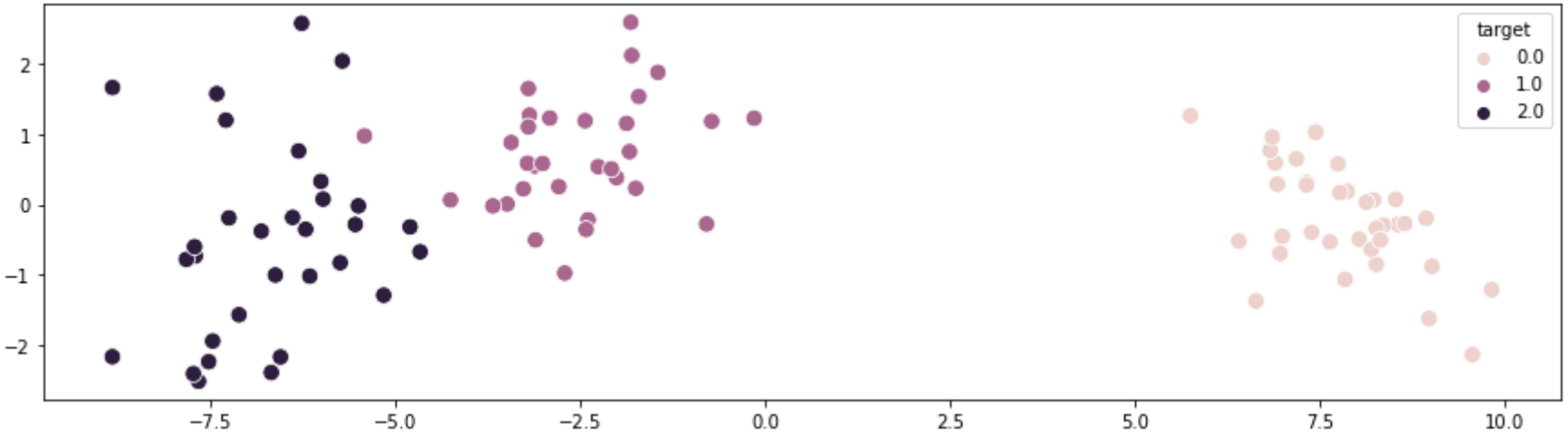
Next part of method 2 includes plotting the 4 features on a 2D graph which is continued as below.

6. Now, we have to plot the data on a 2D graph. I have done this with 2 different ways.

7. In first way, I have drawn 6 subplots for 6 different relations as on below figure.



8. In second way, I have reduced dimensions by LDA into 2 and then plotted. This was done by taking n_components=2 inside LDA model and this reduced 4 features into 2. Now, I had 2 features so, I used one of them for x axis and another for y axis and plotted similarly as in method 1 with 2 features. Here, one can also observer that there is no x and y label because they are reduced dimensions.



Part 3 - Method 2 will work better than Method 1

The reason that method2 will work more better than method1 is that because of the current accuracy. Accuracy is more in method2 as compared to method1. Moreover, for the incoming data we can't say that where the data will go. For example, say if class 1 data is going into (class 2 and class 0) and getting merged with them then it will become very difficult to separate the data points using a line. Rather we need a polynomial kernel to separate them and due to the linearity is allowed, underfitting problem will come for the upcoming data which is not the case with the method2 where we are using LDA to predict and reducing the dimensions. The next problem that can arise in method1 is that if more classes comes in it, then it will become more difficult to separate the classes from each other and underfitting will come for sure in method1 whereas we are always reducing dimensions to 2 in method2 and it will predict more accurately then method1.

Problem 2

From MNIST Fashion data take 100 data points belonging to classes Sneaker, Pullover and Ankle boot.

Part 1: Project these data points on a 2D plane using data reduction techniques PCA, T-SNE and MDA.

Part 2 : Take 50 data points belonging to classes Sneaker, Pullover and Ankle boot from the test data set and compare the performance of these techniques(PCA, T-SNE, MDA) in terms of accuracy.

Part 3 : According to your view what is the best dimensionality reduction technique for solving this problem and justify your answer.

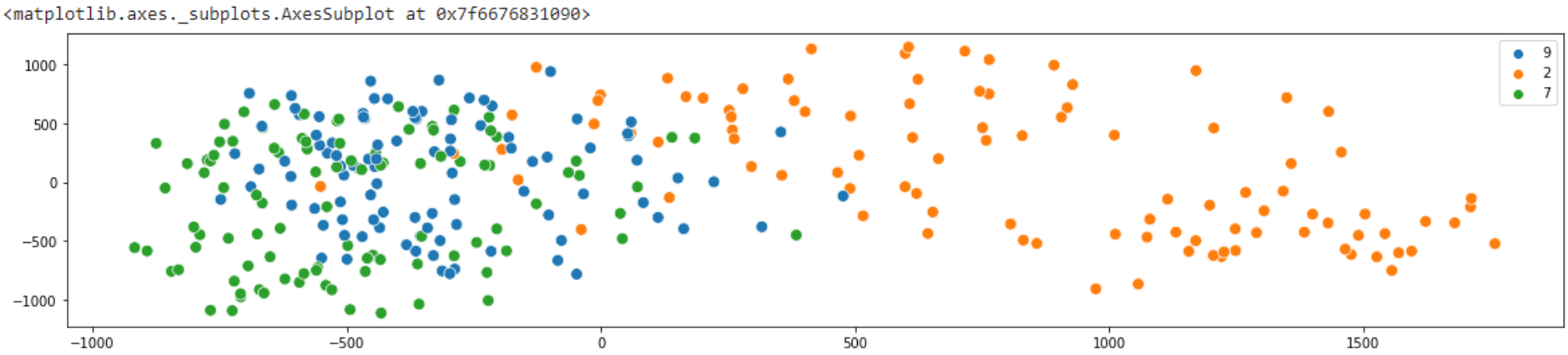
Approach for Part 1

1. Load Mnist Fashion Dataset from fetch_openml library.
2. We need to filter out the classes other than Sneaker, Pullover and Ankle Boot. Moreover, also after doing this we have to take only 300 data rows.
3. After doing this for the part 1, we need to make the model with n_components as 2 and then call the fit_transform method to get the reduced dimentions.

Note: Benefit of doing this is that the original dataset has more than 750 features and its difficult to plot with so many features. To make it easy we are reducing the features into by 3 different dimensionality reduction models. These are PCA, TSNE and LDA. In this part, I have plotted the graph after dimensionality reduction to 2.

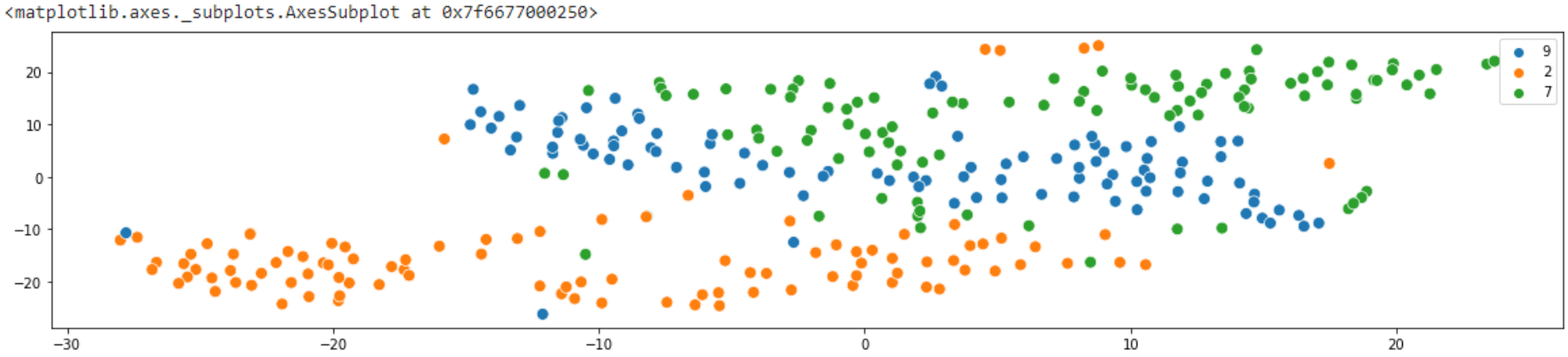
Approach for Part 1 (cont.)

4. Principal Component Analysis or PCA. We can observe that data is scattered for label number 2. However, for label 7 and 9, the data is hindered as compared to the label 2.



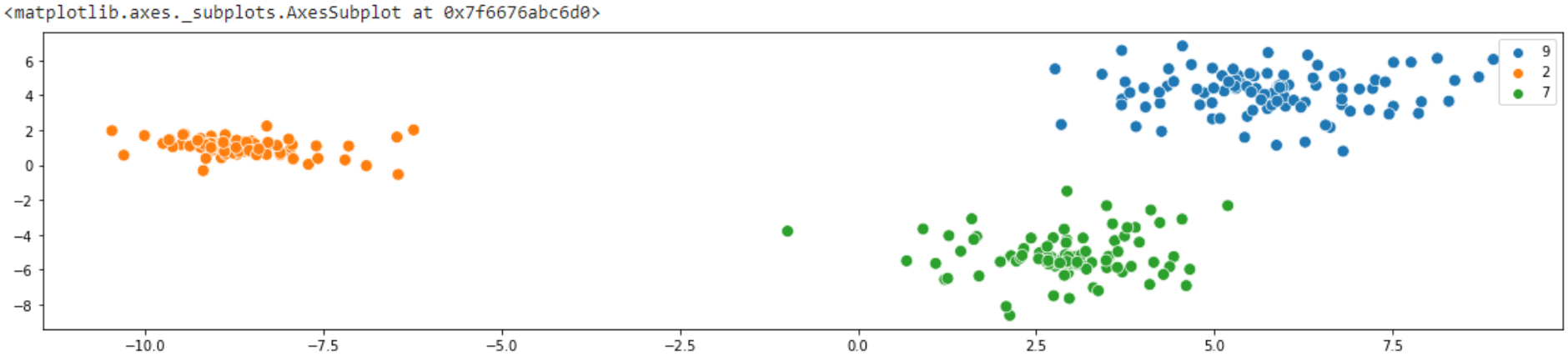
Approach for Part 1 (cont.)

5. T-Distributed Stochastic Neighbour Embedding or (TSNE). We can observe that data is scattered for all the label numbers that is label 2, label 7 and label 9. We can directly say PCA will be more accurate then TSNE



Approach for Part 1 (cont.)

6. Linear Discriminant Analysis or LDA. We can observe that data is hindered for all the label numbers that is label 2, label 7 and label 9. We can directly say LDA will be more accurate then TSNE and can also be accurate then PCA.



Approach for Part 2

1. We have already filtered out the classes other than Sneaker, Pullover and Ankle Boot and also taken only 100 data rows.
2. We need to split the data into training and testing with test_size as 50%.
3. For this part we will be creating the 3 models for dimensionality reduction and then use Logistic Regression Classifier to predict the accuracy scores.

4. The classification model with highest accuracy after performing 3 different dimensionality reduction techniques will be compared and the one with highest accuracy, will taken into consideration and its dimensionality reduction technique will be best out of the three.

Approach for Part 2 (cont.)

5. After performing PCA technique to reduce dimensions, accuracy is 70%

```
# Creating the logistic classifier
logisitcModel=LogisticRegression()

#Fitting the features got after reducing dimentions and y as y_training data
logisitcModel.fit(reducedDim, y_train)

# Predicting the output for testing data
predicted_values=logisitcModel.predict(pcaModel.fit_transform(X_test))

# Printing the actual Y values and prediceted Y values by the Logistic Regression model after dimentionality reduction by PCA
print("Actual test output values are :", list(y_test))
print("Predicted test output values are :", list(predicted_values))
```

```
Actual test output values are : ['9', '7', '7', '2', '2', '7', '7', '9', '7', '7', '2', '2', '9', '9', '7', '2', '7', '2', '2', '2', '7', '2', '7', '7', '7', '9'
Predicted test output values are : ['7', '7', '7', '2', '2', '7', '7', '9', '7', '7', '2', '2', '7', '9', '7', '2', '9', '2', '2', '2', '9', '2', '7', '9', '7',
```

+ Code + Text

```
print('Accuracy of logistic regression classifier after dimentiionality reduction by PCA :', logisitcModel.score(pcaModel.fit_transform(X_test), y_test))
```

```
Accuracy of logistic regression classifier after dimentiionality reduction by PCA : 0.7066666666666667
```

Approach for Part 2 (cont.)

6. After performing TSNE technique to reduce dimensions, accuracy is 42%

```
# Creating the logistic classifier
logisitcModel=LogisticRegression()

#Fitting the features got after reducing dimentions and y as y_training data
logisitcModel.fit(reducedDim, y_train)

# Predicting the output for testing data
predicted_values=logisitcModel.predict(tsneModel.fit_transform(X_test))

# Printing the actual Y values and prediceted Y values by the Logistic Regression model after dimentionality reduction by TSNE
print("Actual test output values are :", list(y_test))
print("Predicted test output values are :", list(predicted_values))
```

```
Actual test output values are : ['9', '7', '7', '2', '2', '7', '7', '9', '7', '7', '2', '2', '9', '9', '7', '2', '7', '2', '2', '2', '7', '2', '7', '7', '7', '9',
Predicted test output values are : ['7', '7', '7', '9', '9', '7', '7', '7', '7', '7', '9', '2', '9', '7', '9', '9', '9', '2', '2', '9', '2', '2', '7', '7', '7', '
```

```
print('Accuracy of logistic regression classifier after dimentiionality reduction by TSNE :', logisitcModel.score(tsneModel.fit_transform(X_test), y_test))
```

```
Accuracy of logistic regression classifier after dimentiionality reduction by TSNE : 0.4266666666666667
```

Approach for Part 2 (cont.)

7. After performing LDA technique to reduce dimensions, accuracy is 91%

```
# Creating the logistic classifier
logisitcModel=LogisticRegression()

#Fitting the features got after reducing dimentions and y as y_training data
logisitcModel.fit(reducedDim, y_train)

# Predicting the output for testing data
predicted_values=logisitcModel.predict(LDAModel.fit_transform(X_test, y_test))

# Printing the actual Y values and prediceted Y values by the Logistic Regression model after dimentionality reduction by LDA
print("Actual test output values are :", list(y_test))
print("Predicted test output values are :", list(predicted_values))
```

```
Actual test output values are : ['9', '7', '7', '2', '2', '7', '7', '9', '7', '7', '2', '2', '9', '9', '7', '2', '7', '2', '2', '2', '7', '2', '7', '7', '7', '9'
Predicted test output values are : ['9', '7', '7', '2', '2', '7', '7', '9', '7', '9', '2', '2', '9', '9', '7', '2', '7', '2', '2', '2', '7', '2', '7', '7', '7',
```

```
print('Accuracy of logistic regression classifier after dimentiionality reduction by LDA :', logisitcModel.score(LDAModel.fit_transform(X_test, y_test), y_test))
```

```
Accuracy of logistic regression classifier after dimentiionality reduction by LDA : 0.9133333333333333
```

Part 2 Conclusion

So, we are getting 91% accuracy by first reducing dimensions using LDA and then applying logistic regression classifier which is the highest accuracy that I got and this means LDA is more good than PCA dimensionality reduction.
Basically these three(PCA, TSNE and LDA) are used to reduce the dimensions. The dataset given to us has more than 750 features and I reduce these 750+ features into 2 features by using these three models one by one and then used a Logistic Regression Classifier to predict the accuracy of the model. The comparison in terms of accuracy I got is : accuracy(LDA)>accuracy(PCA)>>accuracy(TSNE).
Accuracy with LDA = 91%
Accuracy with PCA = 70%
Accuracy with TSNE = 42%
So, I will be concluding that TSNE has shown bad performance and LDA and PCA both have shown a good performance and comparing them LDA is giving more accuracy then PCA.

Part 3 – Which of the PCA, LDA and TSNE is best?

Some of the observations were made in part 2 where I concluded that LDA is more good than PCA and PCA & LDA are more good than TSNE. But this conclusion was done on the basis of accuracy score. We can see that there is small difference(15%-20%) in accuracy scores of LDA and PCA so, we need a other reason on which we can compare them. But, there is a tremendous amount of difference of accuracies in TSNE and LDA. It's nearly 2 times the accuracy of TSNE. So, I can directly conclude that I will not be taking TSNE to reduce dimensions for this question. We still need to compare between PCA and LDA. I will compare them on the basis of graph they have after dimensionality reduction.

Part 3 (cont.)

The main reason that why LDA has more accuracy then PCA is that the data is more hindered in LDA(or we can say that data is more close in LDA). It can be easily observed that there are 3 groups in both of the LDA and PCA but in LDA, the three groups are not scattered much and are coming within them and in PCA, the points are scattered throughout the graph. In LDA, there is no interference of any group and in PCA, some of the classes have values interfering with the other class. This will lead to a good accuracy in LDA since classes can be distinguished easily. At this moment, the points in PCA are not much scattered due to which accuracy is coming high. But when new points will be added we can easily say points will remain to be intact(or which its own group) in LDA and will get more scattered with PCA. Talking about the TSNE, the same problem comes there that in TSNE, the points are even more scattered than PCA due to which it has very low accuracy.

Part 3 (cont.) - Conclusion

So, making this observation is also important and I will go for LDA because in future also it will give a good accuracy because its points will not be much scattered as compared to PCA and TSNE. PCA's accuracy will decrease at a high rate when data points will be increases that is not the case with LDA.
So, according to me, LDA is the best dimensionality reduction techniques(out of PCA, LDA and TSNE) for solving this problem. I have answered this by seeing the accuracy, scattering of the graph and hinderness of the graphs.

THANKS