

PCA, KPCA and LDA FOR CLASSIFICATION

LAB 9

PCA clustering theory:

The main idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of many variables correlated with each other, **while retaining the variation** present in the dataset. Importantly, the **dataset** on which PCA technique is to be used **must be scaled**. The results are also sensitive to the relative scaling. It can be thought of as a projection method where data with m-columns (features) is projected into a subspace with m or fewer columns, whilst retaining the essence of the original data.

The **output** of PCA are these principal components, the number of which is less than or equal to the number of original variables. Less, in case when we wish to discard or reduce the dimensions in our dataset. PCA operates by diagonalizing the covariance matrix,

$$C = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i \mathbf{x}_i^T$$

KPCA clustering theory:

Kernel principal component analysis (kernel PCA) is an extension of principal component analysis (PCA) using techniques of **kernel methods**. As a result, we can convert linearly non-separable data into linearly separable data by taking the code to the higher dimension, by applying the kernel trick, and then applying PCA in that dimension.

LDA theory:

Linear discriminant analysis (LDA) is a method used to find a **linear combination of features** that characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier. LDA **performs well** only on dataset that has a near normal or normal

distribution. LDA reduces the given feature set into feature set containing C-1 features, where C is the number of different classes.

PART 1: Lab implementation

Implementation of PCA, KPCA from the book Sebastian Rashka helped to understand the **mean vector, scatter matrix, covariance matrix, eigenvectors** and corresponding eigen values. We used the **sklearn.decomposition** library for sklearnPCA. We were able to visually see how a 3 dimensional dataset was **transformed** to a 2 dimension dataset. Next, KPCA was implemented on the two moons dataset using the kernel='rbf'. The Kernel PCA was also able to separate the concentric circle dataset. Finally the **Swiss Roll** dataset was explored and it was reduced to lower dimensions using the first 2 principal components and Locally Linear Embedding.

PART 2: ABSENTEEISM FROM WORK DatasetIMPLEMENTATION 1

The dataset had 21 features, but it was non linearly separable data, and was hence used for classification. First, classification for 6 number of pets owned by the employees was done using the **MLPClassifier**. The confusion matrix showed that while for **4 of 6 classes** classification was mostly correct, for 2 classes it was wrong. To improve it, we applied PCA and KPCA as **pre-processing techniques** and then applied MLPClassifier to it. For PCA, I kept the **n_components** to 0.95 which meant that it would take those many features till the time that the total variance became greater than 95% of the data. This helped us to reduce the 21 features to 12 features. While there was no major improvement when using PCA, there was a significant improvement in performance when using **KPCA and LDA**. With default kernel='linear'

for KPCA, it helped us to transform the data such that **Class 6** of the Pet class became separable. **LDA** reduces the given feature set into feature set containing C-1 features, where **C** is the **number of different classes**. In this case it converted the feature set into fewer features, and was able to make even the class 7 separable, and provided the best classification accuracy for the other 3 classes. The **solver='lsqr'** gave the best performance for LDA.

PART 3: FOREST COVER TYPE dataset

Since we were required to take a non-linearly separable or a clumpy dataset, Forest Cover Type was chosen. Even in this dataset, the algorithm used was the **MLPClassifier**. Even with max_iter of MLPClassifier set to a high value of 300, the converged weight did not give us ideal results. The confusion matrix showed that there were atleast 50 points misclassified for each class type. Taking **PCA** with **n_components = 0.96**, MLPClassifier was applied again, after using Standard Scaler. **With PCA**, there was again, only minor change in the accuracy of the classification. With **KPCA** however, the improvement was much more (Since the outputs are confusion matrices, I haven't given the percentage accuracy). For KPCA, we transformed the data into higher dimension using kernel trick to get classes more separable and then applied PCA. With the '**svd**' kernel of KPCA we improved the accuracy of classification, to a good extent – definitely better than PCA. We then applied LDA. However, in this case the **LDA performed badly** as the **dataset** did not have a **normal or near normal distribution**. Thus, for the Forest Cover Type dataset, KPCA was the optimal solution.

Comparison of PCA, KPCA and LDA for regression with SVR:

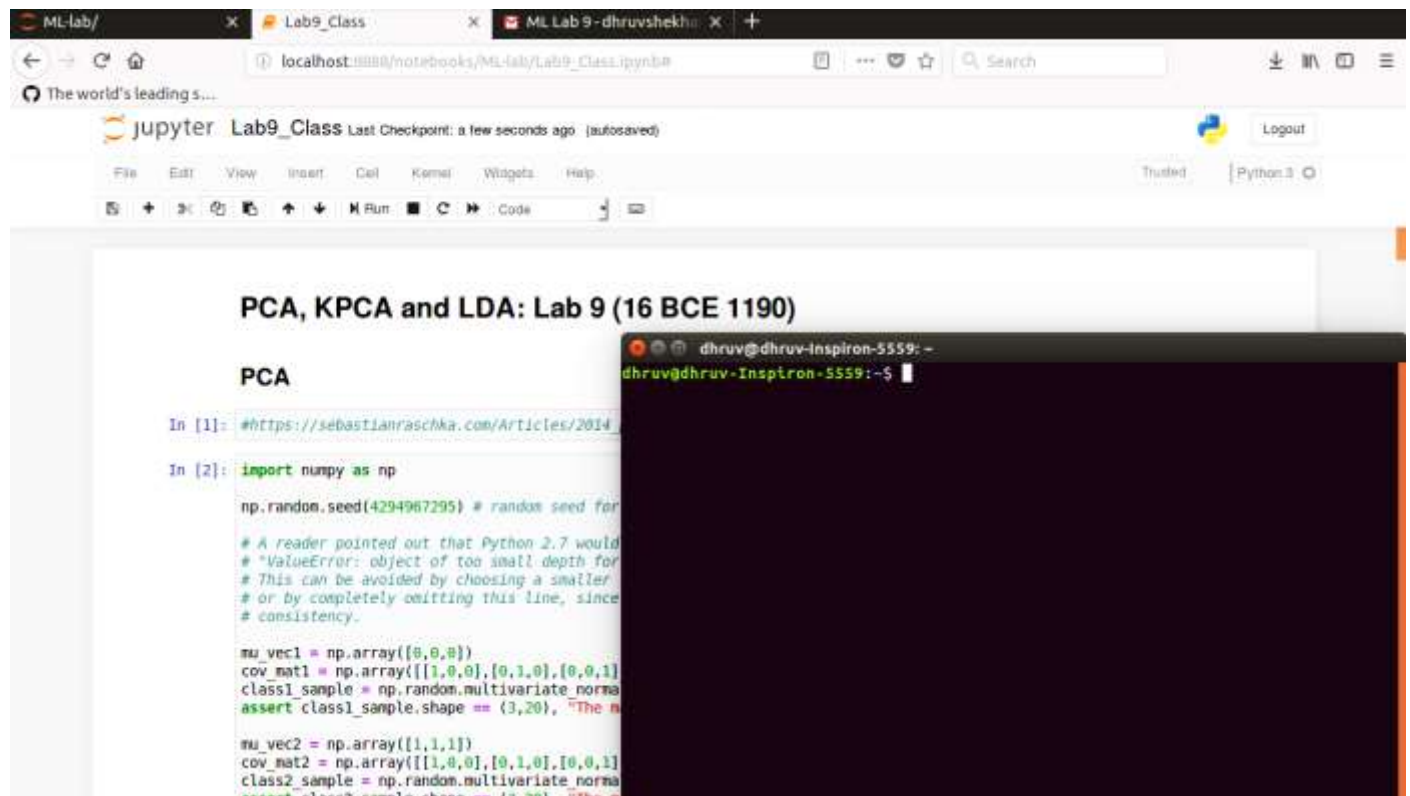
We used dimension reduction techniques on both linearly and non-linearly separable dataset. We

saw that in case of linearly separable data, the decomposition techniques **increase the accuracy** by small amount 2-4% but for non-linear separable dataset there might be a huge boost as in case of house pricing dataset.

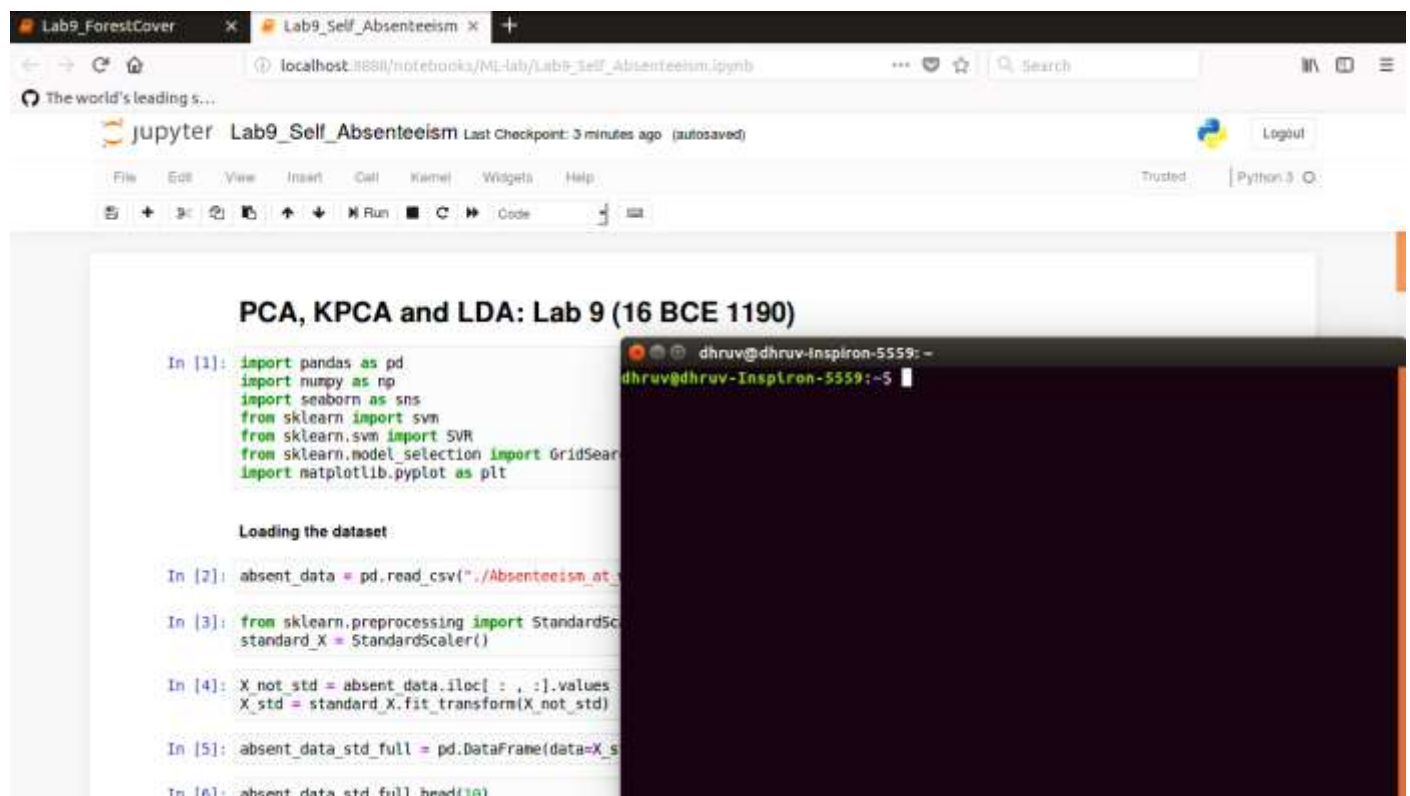
Also we noticed that **KPCA performs better than PCA**. This may be because PCA does not care about separability of classes, but only on capturing information, while KPCA first projects them to higher dimension for better separability and then apply PCA. **LDA reduces the given feature set** into feature set containing C-1 features, where C is the number of different classes. The final feature set is the linear combination of original feature set.

SCREENSHOTS

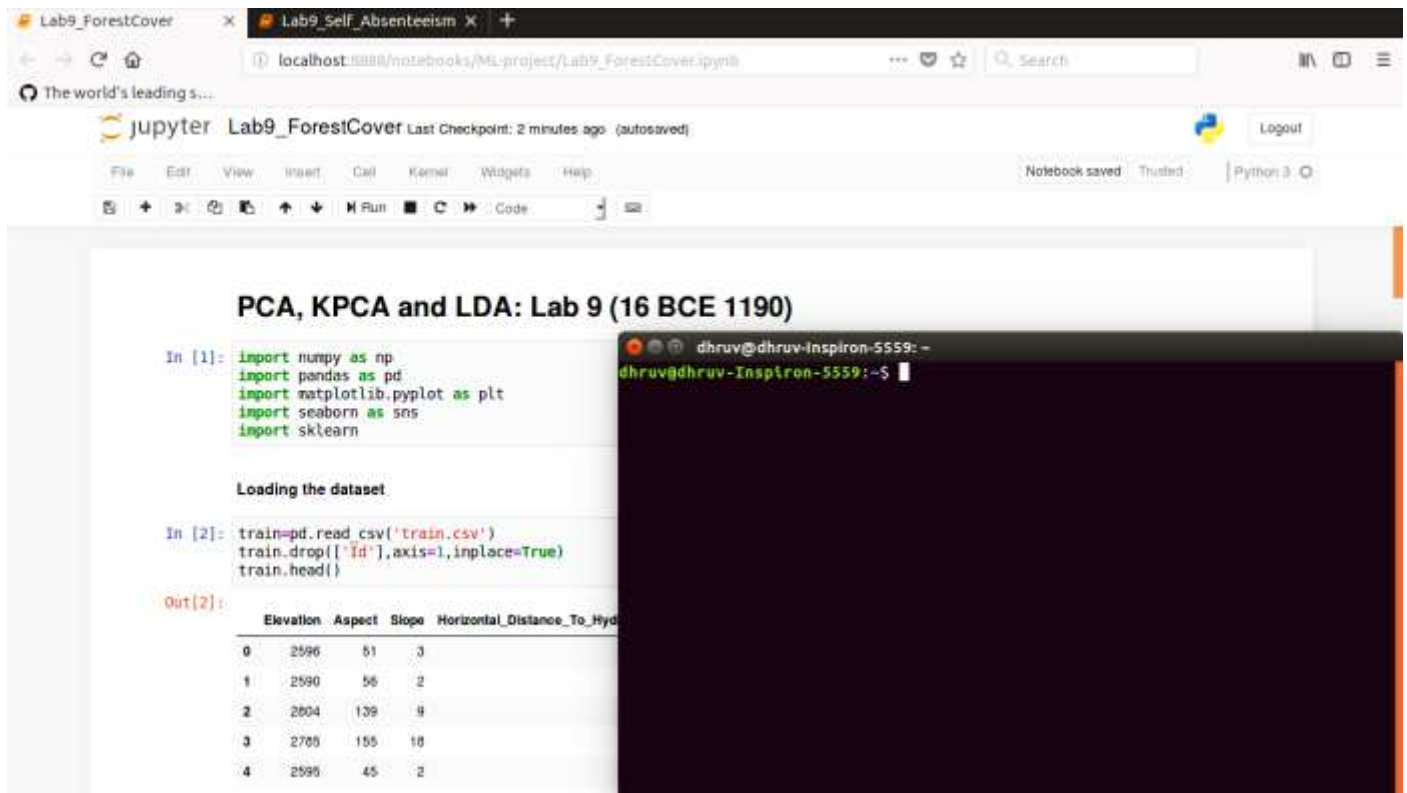
Lab implementation



Chosen dataset 1 – Absenteeism at work



Chosen dataset 2 – Forest cover type



The screenshot shows a Jupyter Notebook interface with two tabs: 'Lab9_ForestCover' and 'Lab9_Self_Absenteeism'. The active tab is 'Lab9_ForestCover', which displays the following content:

PCA, KPCA and LDA: Lab 9 (16 BCE 1190)

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
```

Loading the dataset

```
In [2]: train=pd.read_csv('train.csv')
train.drop(['Id'],axis=1,inplace=True)
train.head()
```

Out[2]:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydro
0	2596	51	3	
1	2590	56	2	
2	2604	139	9	
3	2705	155	10	
4	2595	45	2	

In the foreground, a terminal window is open, showing the user 'dhruv' at the 'dhruv-Inspiron-5559' machine. The prompt is 'dhruv@dhruv-Inspiron-5559:~\$'.