

# Guided Project Report

## LDA Classification

Name: Shruti Verma  
Course: AI and ML  
(Batch 4)  
Duration: 10 months

Problem Statement: Build a machine learning model to learn a classifier to test the LDA performance for dimensionality reduction.

### Prerequisites

What things you need to install the software and how to install them:

Python 3.8 or higher versions This setup requires that your machine has latest version of python. The following url <https://www.python.org/downloads/> can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: <https://www.pythoncentral.io/add-python-to-path-python-is-not-recognized-as-an-internal-or-external-command/>. Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic.

Second and easier option is to download anaconda and use its anaconda prompt to run the commands. To install anaconda check this url <https://www.anaconda.com/download/> You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy if you have chosen to install python 3.8 then run below commands in command prompt/terminal to install these packages `pip install -U scikit-learn` `pip install numpy` `pip install scipy` if you have chosen to install anaconda then run below commands in anaconda prompt to install these packages `conda install -c scikit-learn` `conda install -c anaconda numpy` `conda install -c anaconda scipy`

### Dataset used

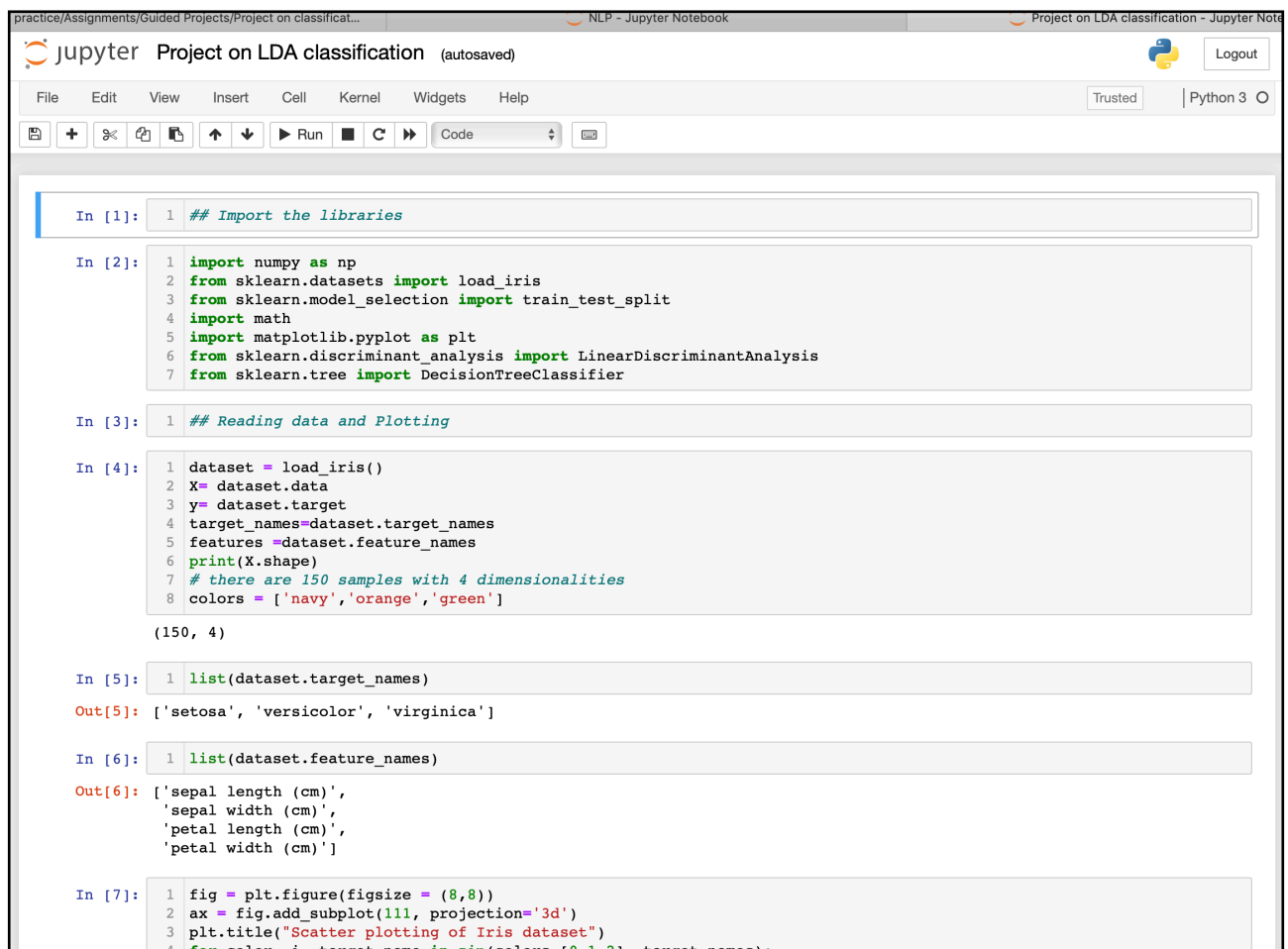
The data source is Iris dataset provided in the scikit-learn library. This data sets consists of 3 different types of irises' (Setosa, Versicolour, and Virginica) petal and sepal length, stored in a 150x4 numpy.ndarray. The data set contains total 150 samples and there are four dimensionalities.

# Method used for detection

## Linear Discriminant Analysis

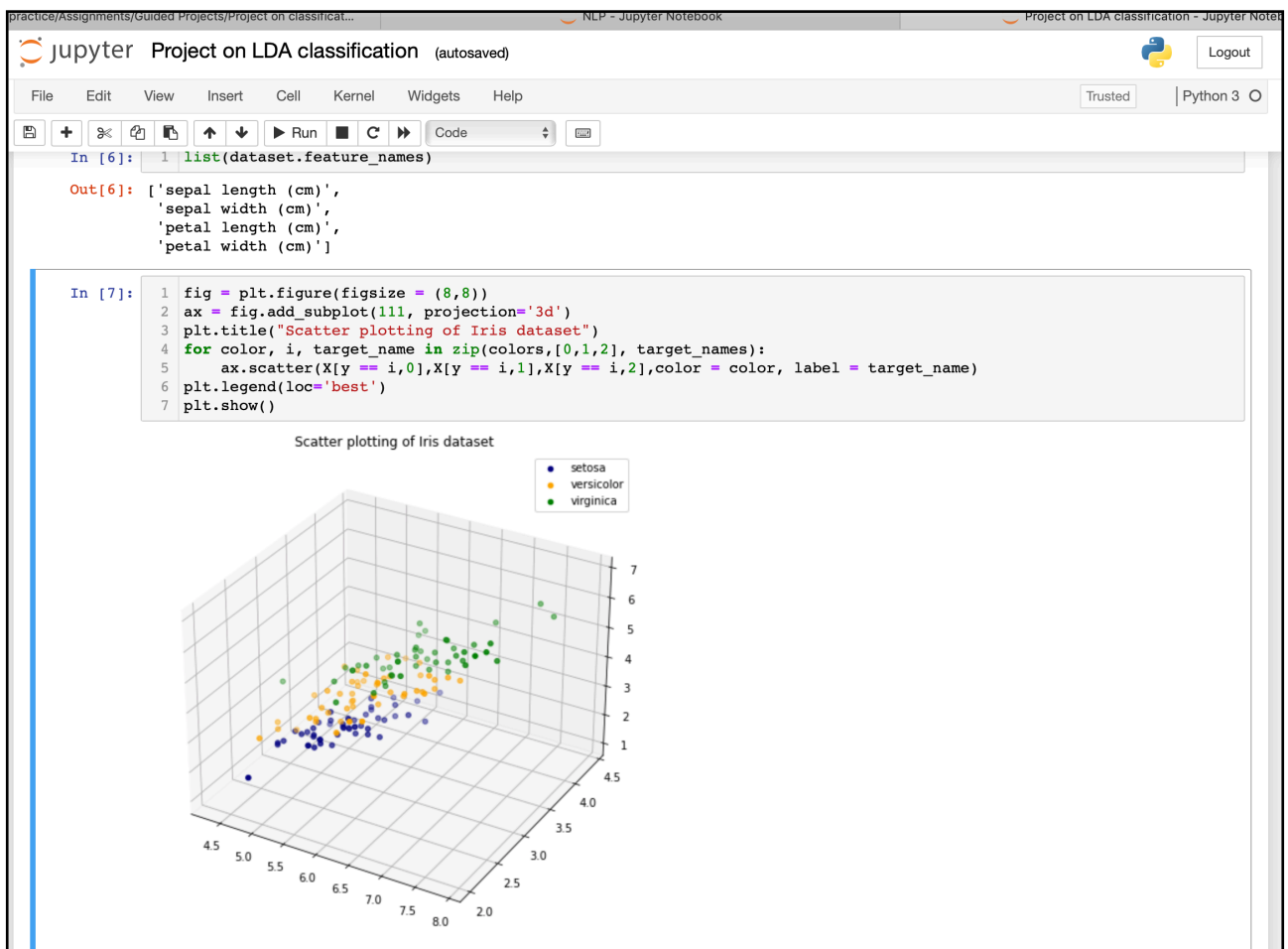
Iris dataset -> reduce dimensionality to one or two -> classify —> test the performance of LDA for dimensionality reduction

Importing the libraries and capturing images:



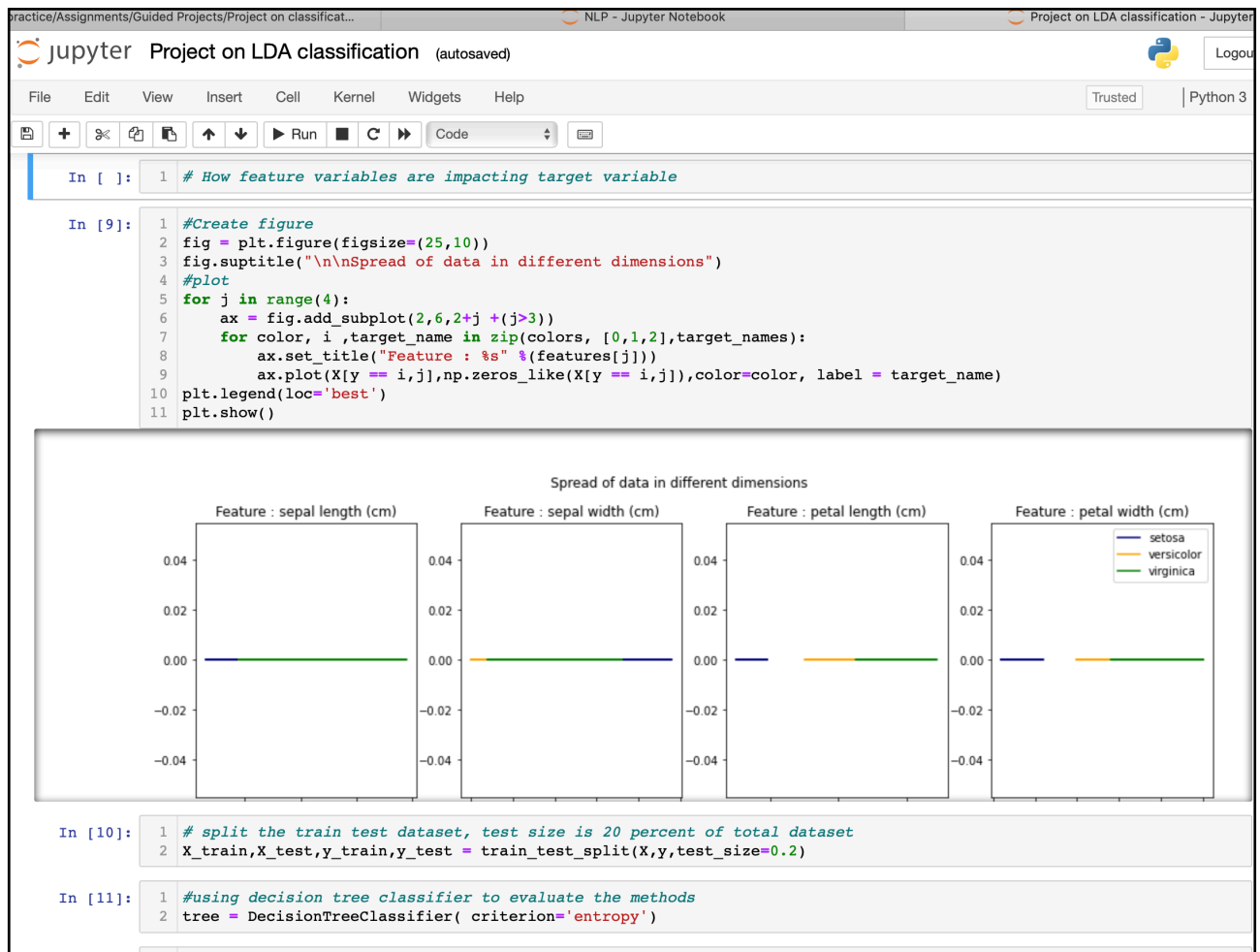
```
practice/Assignments/Guided Projects/Project on classificat... NLP - Jupyter Notebook Project on LDA classification - Jupyter Noteb
jupyter Project on LDA classification (autosaved)
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3
In [1]: 1 ## Import the libraries
In [2]: 1 import numpy as np
        2 from sklearn.datasets import load_iris
        3 from sklearn.model_selection import train_test_split
        4 import math
        5 import matplotlib.pyplot as plt
        6 from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
        7 from sklearn.tree import DecisionTreeClassifier
In [3]: 1 ## Reading data and Plotting
In [4]: 1 dataset = load_iris()
        2 X= dataset.data
        3 y= dataset.target
        4 target_names=dataset.target_names
        5 features =dataset.feature_names
        6 print(X.shape)
        7 # there are 150 samples with 4 dimensionalities
        8 colors = ['navy','orange','green']
(150, 4)
In [5]: 1 list(dataset.target_names)
Out[5]: ['setosa', 'versicolor', 'virginica']
In [6]: 1 list(dataset.feature_names)
Out[6]: ['sepal length (cm)',
        'sepal width (cm)',
        'petal length (cm)',
        'petal width (cm)']
In [7]: 1 fig = plt.figure(figsize = (8,8))
        2 ax = fig.add_subplot(111, projection='3d')
        3 plt.title("Scatter plotting of Iris dataset")
        4 for color, i, target_name in zip(colors, [0,1,2], target_names):
```

## Plotting the images



Splitting the training and testing data into 80 :20.

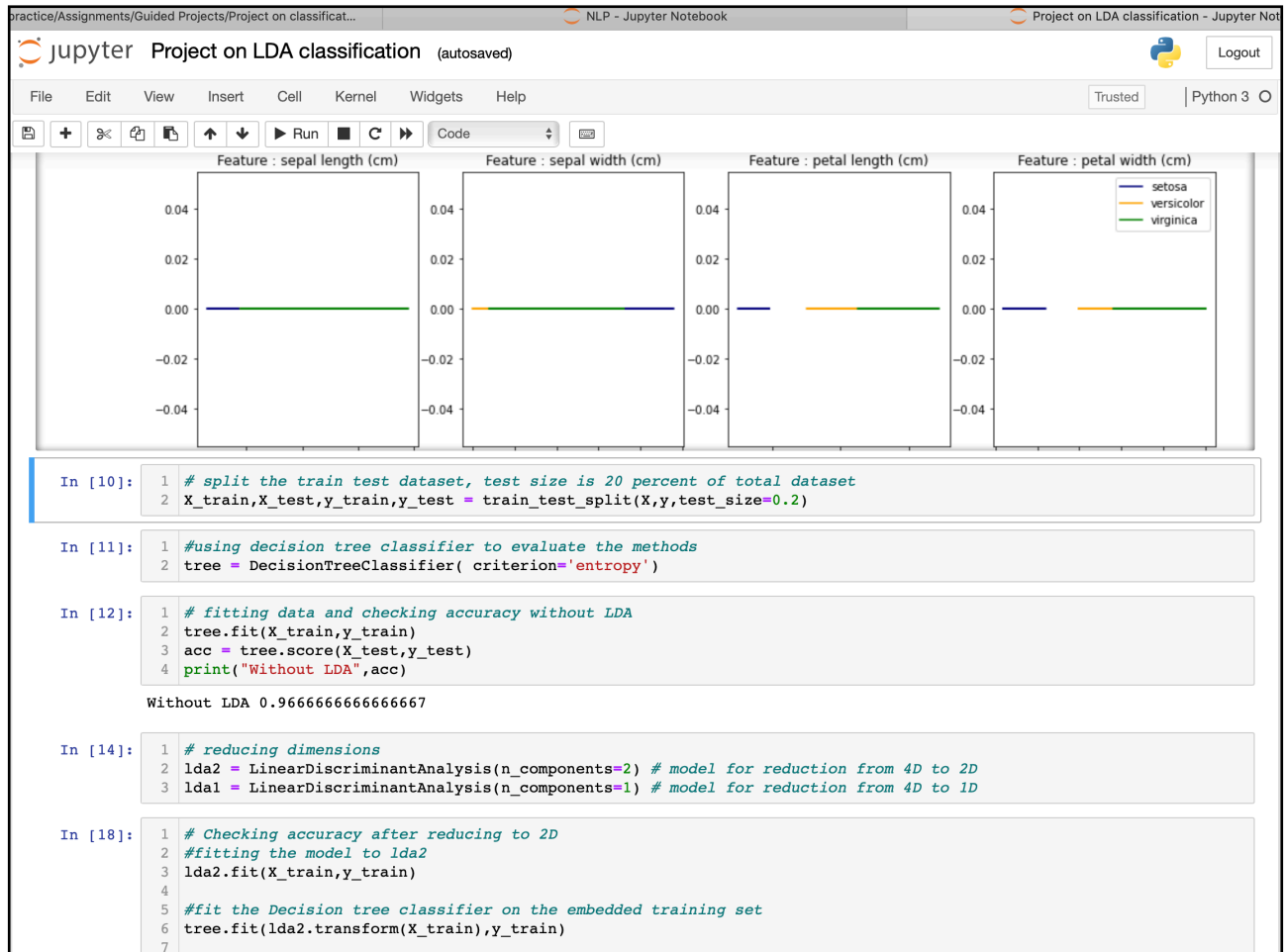
This is hyper parameter and can be varied to 85:15 or 80:20 ratio also.



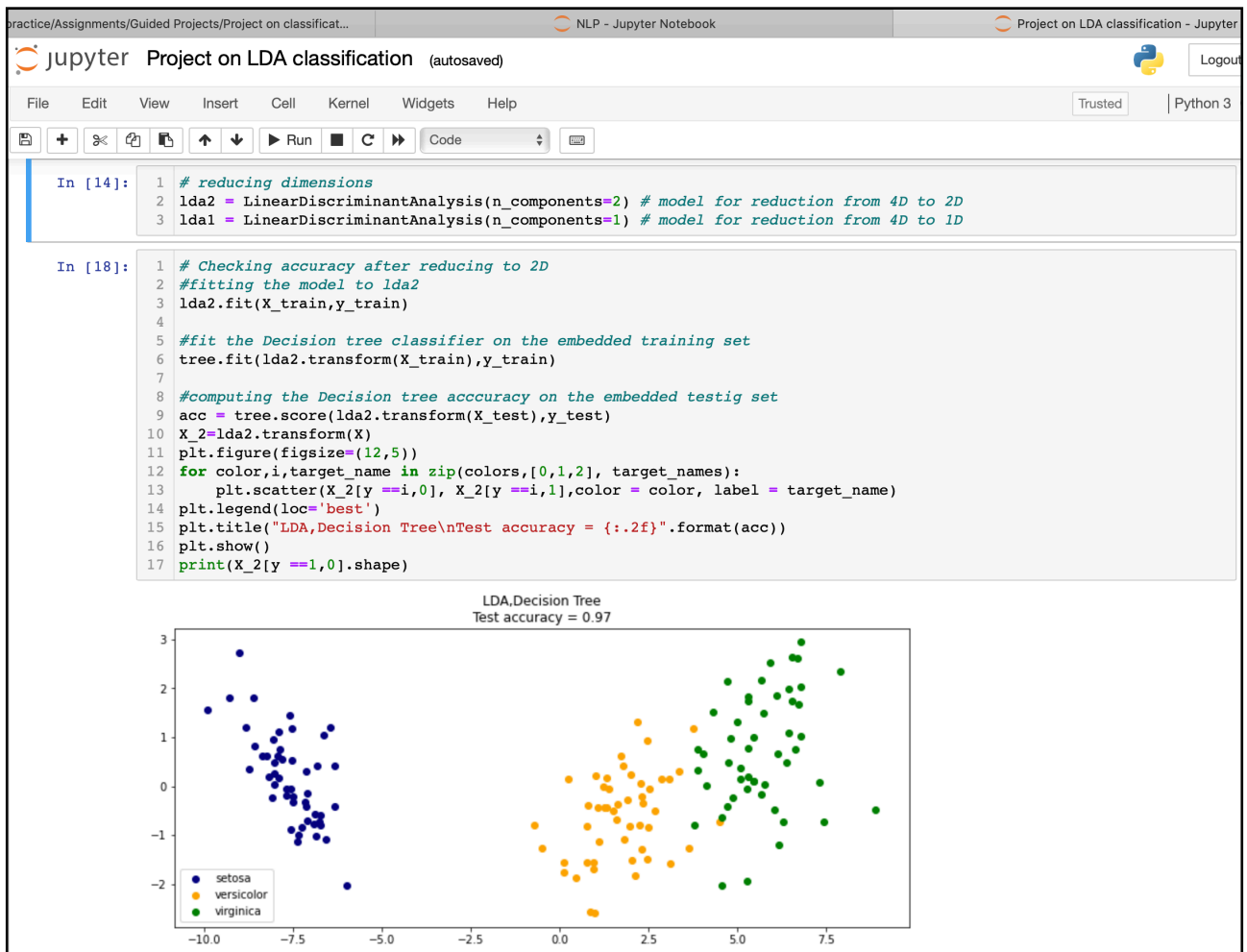
## Using Decision Tree Classifier

### Checking the Performance without LDA dimension reduction

### Reducing Dimensions to 2D and 1D



## Checking the Performance for 2D



## Checking the performance using 1D

