Submitted By

- Ruthvic Punyamurtula (Problems - 3,4)
- Sai Charan Kothapalli (problems - 1,2)

Resources

- youtube demo - part 2
- Source Code

Introduction

This is lab assignment 2 of cs5590 - python/Deep Learning class. This lab is based on the tasks done in ICE 5, ICE 6 & ICE 7 which can be found here.

Objective

In this assignment, we used kaggle datasets & implemented

- Naïve Baye's, SVM and KNN implementation
- K-Means Clustering
- NLP pipeline
- Multiple Linear Regression

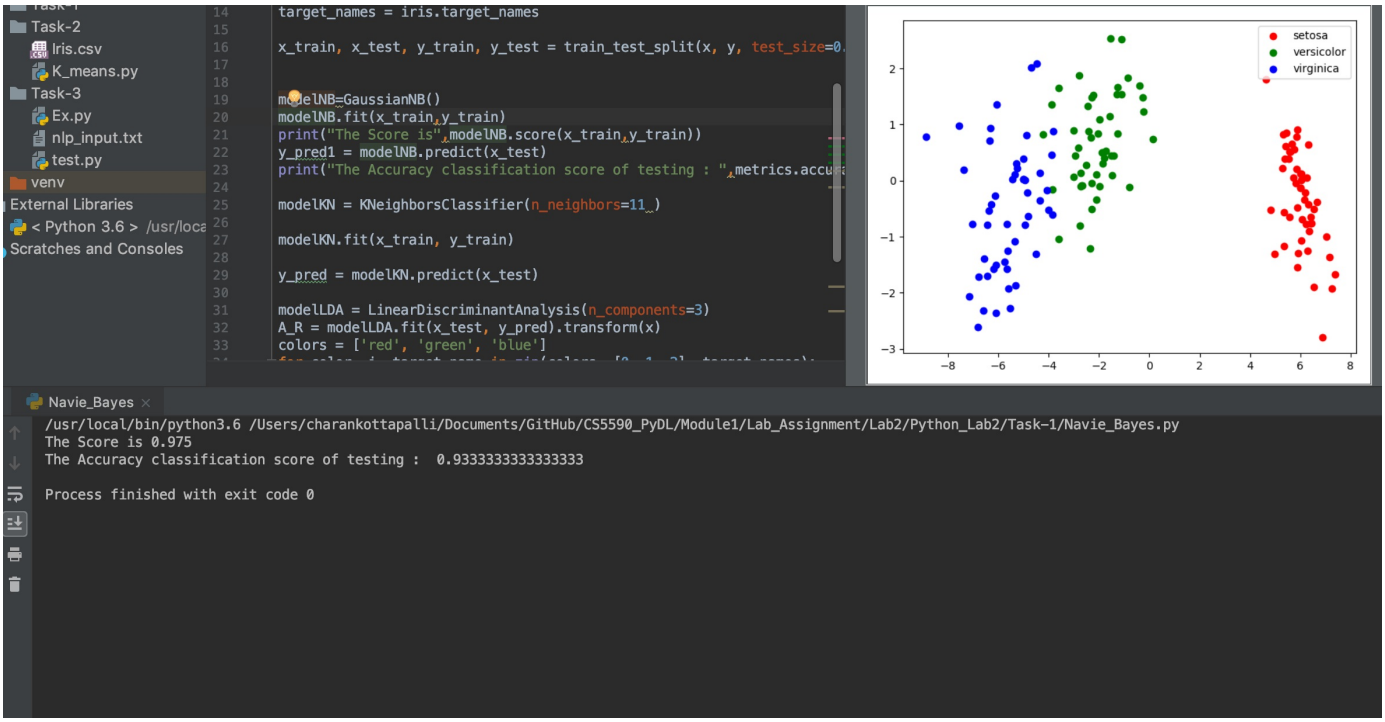
Approaches/Methods

Used pandas Dataframe for data cleaning, used NLTK for nlp pipeline, performed silhoutte score calculation for evaluating k-means

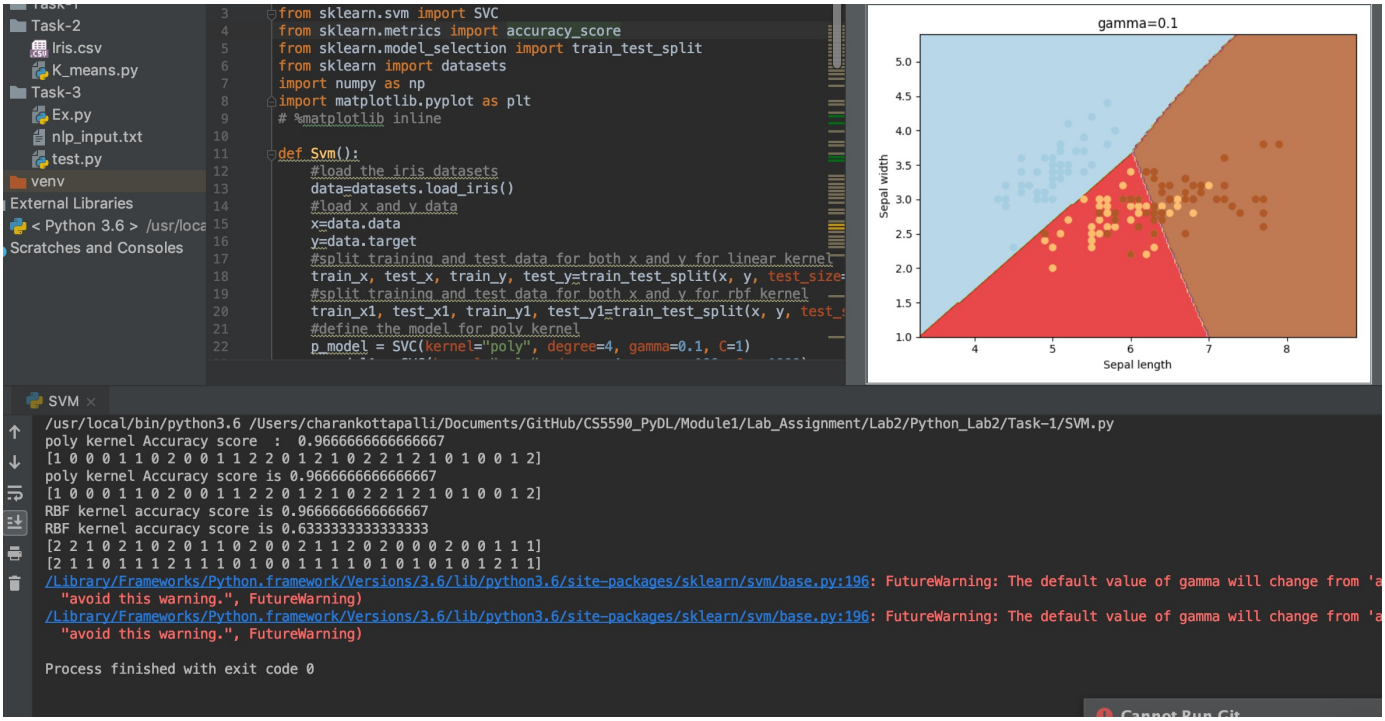
Workflow

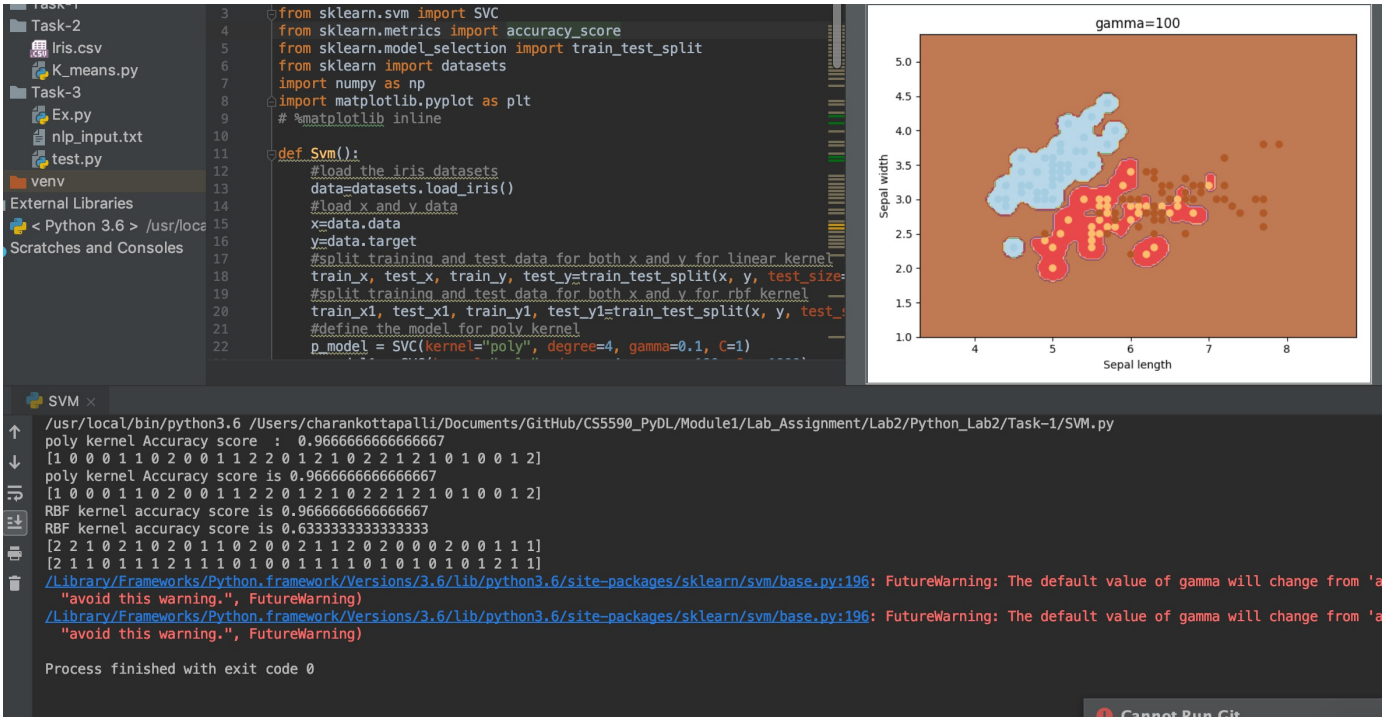
1. Apply Classification algorithms - Naive Bayes, SVM, KNN

Naive bayes

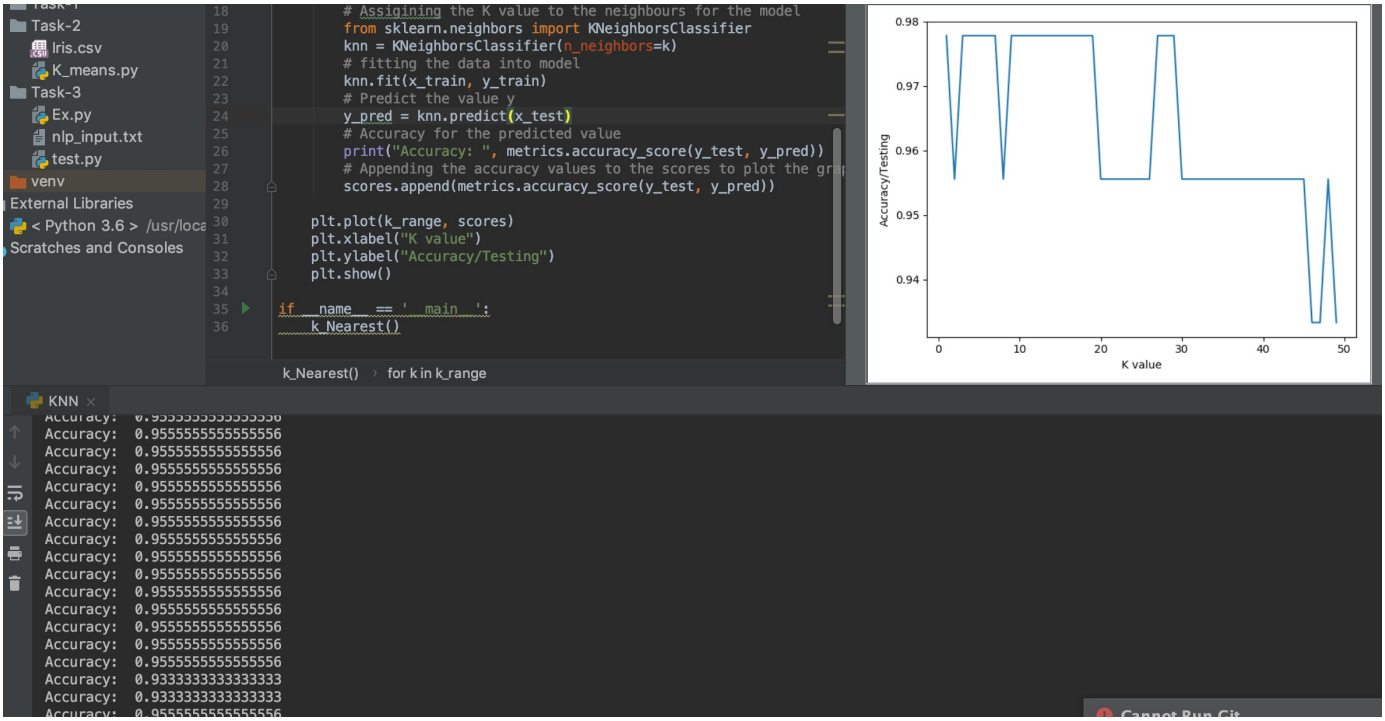


SVM


Task-2
Iris.csv
K_means.py
Task-3
Ex.py
nlp_input.txt
test.py
venv
External Libraries
< Python 3.6 > /usr/loca
Scratches and Consoles
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn import datasets
import numpy as np
import matplotlib.pyplot as plt
# %matplotlib inline
def Svm():
#load the iris datasets
data=datasets.load_iris()
#load x and y data
x=data.data
y=data.target
#split training and test data for both x and y for linear kernel
train_x, test_x, train_y, test_y=train_test_split(x, y, test_size
#split training and test data for both x and y for rbf kernel
train_x1, test_x1, train_y1, test_y1=train_test_split(x, y, test_
#define the model for poly kernel
p_model = SVC(kernel="poly", degree=4, gamma=0.1, C=1)
gamma=0.1
Sepal width
Sepal length
SVM
/usr/local/bin/python3.6 /Users/charankottapalli/Documents/GitHub/CS5590_PyDL/Module1/Lab_Assignment/Lab2/Python_Lab2/Task-1/SVM.py
poly kernel Accuracy score : 0.9666666666666667
[1 0 0 0 1 1 0 2 0 0 1 1 2 2 0 1 2 1 0 2 2 1 2 1 0 1 0 0 1 2]
poly kernel Accuracy score is 0.9666666666666667
[1 0 0 0 1 1 0 2 0 0 1 1 2 2 0 1 2 1 0 2 2 1 2 1 0 1 0 0 1 2]
RBF kernel accuracy score is 0.9666666666666667
RBF kernel accuracy score is 0.6333333333333333
[2 2 1 0 2 1 0 2 0 1 1 0 2 0 0 2 1 1 2 0 2 0 0 0 2 0 0 1 1 1]
[2 1 1 0 1 1 1 2 1 1 1 0 1 0 0 1 1 1 1 0 1 0 1 0 1 0 1 2 1 1]
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-packages/sklearn/svm/base.py:196: FutureWarning: The default value of gamma will change from 'a
"avoid this warning.", FutureWarning)
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-packages/sklearn/svm/base.py:196: FutureWarning: The default value of gamma will change from 'a
"avoid this warning.", FutureWarning)
Process finished with exit code 0


Task-2
Iris.csv
K_means.py
Task-3
Ex.py
nlp_input.txt
test.py
venv
External Libraries
< Python 3.6 > /usr/loca
Scratches and Consoles
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn import datasets
import numpy as np
import matplotlib.pyplot as plt
# %matplotlib inline
def Svm():
#load the iris datasets
data=datasets.load_iris()
#load x and y data
x=data.data
y=data.target
#split training and test data for both x and y for linear kernel
train_x, test_x, train_y, test_y=train_test_split(x, y, test_size
#split training and test data for both x and y for rbf kernel
train_x1, test_x1, train_y1, test_y1=train_test_split(x, y, test_
#define the model for poly kernel
p_model = SVC(kernel="poly", degree=4, gamma=0.1, C=1)
gamma=100
Sepal width
Sepal length
SVM
/usr/local/bin/python3.6 /Users/charankottapalli/Documents/GitHub/CS5590_PyDL/Module1/Lab_Assignment/Lab2/Python_Lab2/Task-1/SVM.py
poly kernel Accuracy score : 0.9666666666666667
[1 0 0 0 1 1 0 2 0 0 1 1 2 2 0 1 2 1 0 2 2 1 2 1 0 1 0 0 1 2]
poly kernel Accuracy score is 0.9666666666666667
[1 0 0 0 1 1 0 2 0 0 1 1 2 2 0 1 2 1 0 2 2 1 2 1 0 1 0 0 1 2]
RBF kernel accuracy score is 0.9666666666666667
RBF kernel accuracy score is 0.6333333333333333
[2 2 1 0 2 1 0 2 0 1 1 0 2 0 0 2 1 1 2 0 2 0 0 0 2 0 0 1 1 1]
[2 1 1 0 1 1 1 2 1 1 1 0 1 0 0 1 1 1 1 0 1 0 1 0 1 0 1 2 1 1]
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-packages/sklearn/svm/base.py:196: FutureWarning: The default value of gamma will change from 'a
"avoid this warning.", FutureWarning)
/Library/Frameworks/Python.framework/Versions/3.6/lib/python3.6/site-packages/sklearn/svm/base.py:196: FutureWarning: The default value of gamma will change from 'a
"avoid this warning.", FutureWarning)
Process finished with exit code 0

KNN



2. Apply K-Means clustering

Code Snippets

```
def tup_to_dict(tup, dict):
    for a, b in tup:
        dict.setdefault(a, []).append(b)
    return dict

# function to sort dictionary whose values are list of tuples
def sort_dict(dict):
    for idx,list_of_tups in dict.items():
        # key - idx, value = list_of_tups
        dict[idx] = sorted(list_of_tups,key=lambda x: x[1]) # sorts on 1st value of list
    return dict
```

Output

```
C:\Users\ruthv\Anaconda3\python.exe C:/Users/ruthv/Documents/GitHub/CS5590_PyDL/Module1/Lab_Assignment/Lab1/Source/Tuples.py
Output before sorting is :
{'John': [('Physics', 80), ('Science', 95)], 'Daniel': [('Science', 90), ('History', 75)], 'Mark': [('Maths', 100), ('Social', 95)]}
Output after sorting is :
{'John': [('Physics', 80), ('Science', 95)], 'Daniel': [('History', 75), ('Science', 90)], 'Mark': [('Social', 95), ('Maths', 100)]}
```

3. Read given text file and perform lemmatization, tokenization,

tri-grams and pick top 10 trigrams

Code Snippets

Code snippets

4. Perform Multiple linear regression

Code Snippets

```
y = df['Video_views']
x = df.drop(['Rank','Channel_name', 'Video_views'], axis=1)
print(x.columns.values)
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, random_state=42, test_size=.30)
```

```
['Grade' 'Video_Uploads' 'Subscribers']
```

```
from sklearn import linear_model
lr = linear_model.LinearRegression()
model = lr.fit(X_train, y_train)
##Evaluate the performance and visualize results
print ("R^2 is: \n", model.score(X_test, y_test))
y_predicted = model.predict(X_test)
from sklearn.metrics import mean_squared_error
print ('RMSE is: \n', mean_squared_error(y_test, y_predicted))
```

```
R^2 is:
 0.5089918039044261
RMSE is:
 1.48183433733783e+18
```

Output