Python and Deep Learning Programming

Lab Assignment1

Team Members:

Vishnu vardhan Reddy Manne
Sai Jaswanth Gattidi
Vandith Thotla

Department of Computer Science

University of Missouri, Kansas City

Introduction:

This Lab assignment will help us to get experienced with some of the concepts in python like tuples, dictionaries, OOPS concepts and BeautifulSoup package. This lab work also contains the application of some of the machine learning algorithms like Navie Bayes, SVM, KNN, Multiple regression and K-Means clustering.

Objectives:

The objectives of this Lab are:

- To get all the possible subsets without any null or duplicate combinations in a given set.
- Creating multiple dictionaries and merging them and then sorting the merged dictionary content based on value.
- Writing a python program to create Airline Booking Reservation system with classes Flight, Person, Employee, Passenger by using Inheritance and Method Overriding concepts
- Using beautifulsoup4 and requests package to get the contents of a webpage and then printing all the course codes and their respective course descriptions from the webpage.
- Picking any dataset that includes both numeric and non-numeric data and performing exploratory data analysis (EDA) on the chosen dataset like handling null values, finding the correlation between the features, replacing the null values with mean, encoding the non-numeric features. Applying the classification Algorithms: Naïve Bayes, KNN, and SVM on the chosen dataset and reporting the classification algorithm with the best result.
- Picking the dataset of our choice and performing exploratory data analysis and then applying the K-means on the dataset and visualizing the clusters using matplotlib or seaborn. Reporting the K value using Elbow method and evaluating the k value using silhouette score.
- To take an input file and read it and apply Tokeniation, Lemmatization and finding the trigrams for the words, top10 most repeated trigrams, extracting sentences with most repeated trigrams and concatenating them and printing the result.

- To pick a dataset and apply Multiple regression and reporting the R squared, and RMSE values before and after performing the Exploratory data analysis on the chosen dataset.

Approaches/Methods:

1.

Code:

```python
import itertools
# creating an empty list
Elements_list = []
# number of elements as input
n = int(input("Enter the number of elements of array: "))
print("Enter the values for elements:")
# iterating till the range
for i in range(0, n):
    elements = int((input()))
# adding the elements to the Elements_list
    Elements_list.append(elements)
for i in range(1,len(Elements_list)+1):
# Prints all possible combinations of r elements in a given array of size n
   a = (list(itertools.combinations(Elements_list,i)))
   print(list(set(a)))
```

Output:

```
C:\Users\vishn\PycharmProjects\Lab1\venv\Scripts\python.exe C:/Users/vishn/PycharmProjects/Lab1/1st_solution.py
Enter the number of elements of array: 3
Enter the values for elements:
1
2
2
[(2,), (1,)]
[(1, 2), (2, 2)]
[(1, 2, 2)]

Process finished with exit code 0
```

- Here itertools is a module that implements a number of iterator building blocks.

- We create an empty list and then, we take the number of elements as input to the array from the terminal. Then we will append the elements taken as input from the terminal to the empty list created.
- Then we will iterate for i in the range of 0 to n (which is size of the list) and print all the combinations of elements in the given array of size n.
- Here we get some of the duplicate combinations so we use list(set(a)) in order to eliminate them. Then we will get all the possible subsets without any null and duplicate combinations.

2.

Code:

```
# Create first dictionary
dict1 = {'Vishnu': 5, 'Vandith': 7, 'Jeswanth': 10}
# Create second dictionary
dict2 = {'Haneesh': 8, 'Vishnu': 20, 'Harun': 11}
# Merge contents of dict2 in dict1
dict1.update(dict2)
print('Updated dictionary 1 :')
print(dict1)
# Create a list of tuples sorted by index 1 i.e. value field
listofTuples = sorted(dict1.items(), key=lambda x: x[1])
# Iterate over the sorted sequence
for elem in listofTuples:
    print(elem[0], " ::", elem[1])
```

Output:

```
C:\Users\vishn\PycharmProjects\Lab1\venv\Scripts\python.exe C:/Users/vishn/PycharmProjects/Lab1/2nd_solution.py
Updated dictionary 1 :
{'Vishnu': 20, 'Vandith': 7, 'Jeswanth': 10, 'Haneesh': 8, 'Harun': 11}
Vandith  :: 7
Haneesh  :: 8
Jeswanth  :: 10
Harun  :: 11
Vishnu  :: 20

Process finished with exit code 0
```

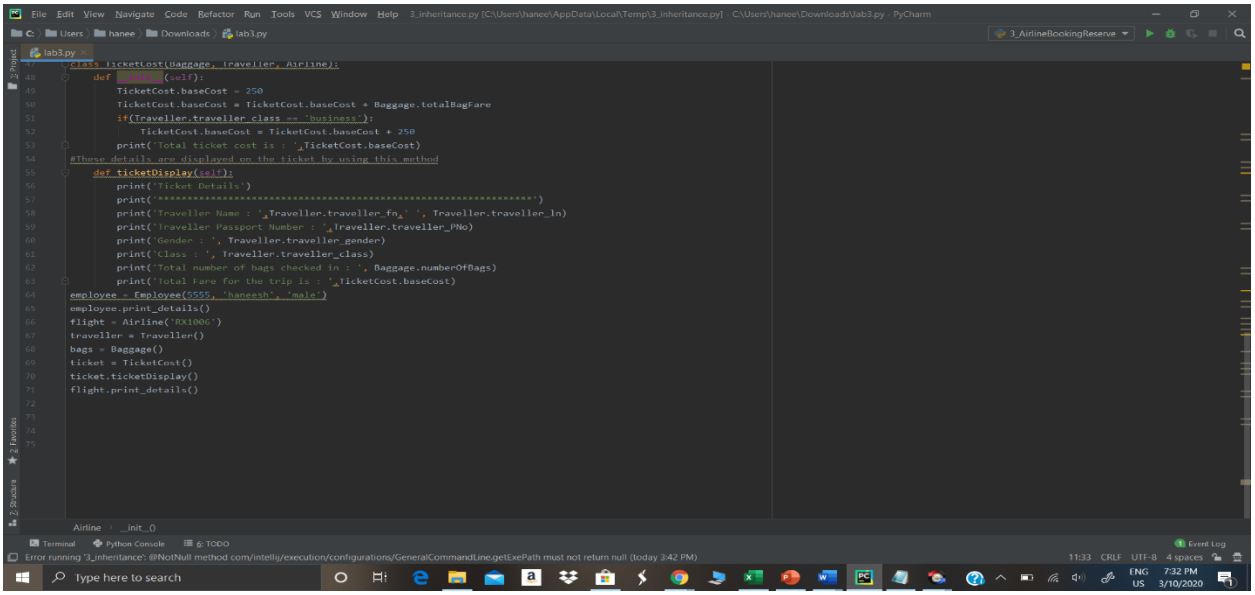
- Here we create two dictionaries dict1 & dict2 and we merge the contents of dictionary2(dict2) to dictionary1(dict1) using the update function.

- Then in order to sort the dictionary content based on value we create a sorted list of tuples and then will iterate over this sorted list of values and print the key value pairs in sorted order of values.

3(a).

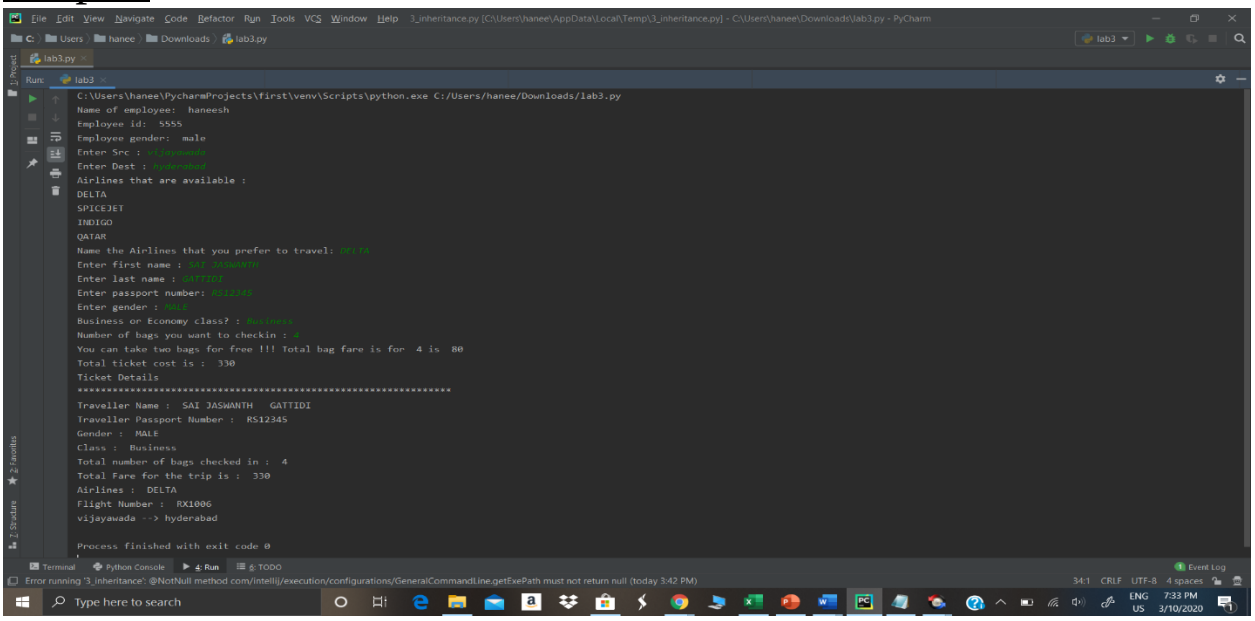
Code:

```
class Airline:
    def __init__(self, flightNo):
        Airline.source = input('Enter Src : ')
        Airline.destination = input('Enter Dest : ')
        print('Airlines that are available :')
        print('DELTA')
        print('SPICEJET')
        print('INDIGO')
        print('QATAR')
        Airline.airlinesName = input('Name the Airlines that you prefer to travel: ')
        self.flightNo = flightNo
#This method prints the details of the airlines
    def print_details(self):
        print('Airlines : ', Airline.airlinesName)
        print('Flight Number : ', self.flightNo)
        print(Airline.source, '-->', Airline.destination)
#Inheritance
class Employee(Airline):
    def __init__(self, employee_id, employee_name, employee_gender):
        self.employee_name = employee_name
        self.employee_id = employee_id
        self.employee_gender = employee_gender
#Method Overriding
    def print_details(self):
        print("Name of employee: ", self.employee_name)
        print('Employee id: ', self.employee_id)
        print('Employee gender: ', self.employee_gender)
#This class inputs all the Traveller details
class Traveller:
    def __init__(self):
        Traveller.traveller_fn = input('Enter first name : ')
        Traveller.traveller_ln = input('Enter last name : ')
        Traveller.traveller_PNo = input('Enter passport number: ')
        Traveller.traveller_gender = input('Enter gender : ')
        Traveller.traveller_class = input('Business or Economy class? : ')
#This class is used to calculate the baggage fare
class Baggage:
    def __init__(self):
        Baggage.numberOfBags = int(input('Number of bags you want to checkin : '))
        Baggage.totalBagFare =0
        Baggage.numberOfBags = Baggage.numberOfBags
        if(Baggage.numberOfBags > 3):
            for i in range(Baggage.numberOfBags-3):
                Baggage.totalBagFare += 80
        print('You can take two bags for free !!! Total bag fare is for ', Baggage.numberOfBags,'is ',Baggage.totalBagFare)
#This class calculates the ticket cost based on the class, flight and bags
class TicketCost(Baggage, Traveller, Airline):
    def __init__(self):
        TicketCost.baseCost = 250
        TicketCost.baseCost = TicketCost.baseCost + Baggage.totalBagFare
        if(Traveller.traveller_class == 'business'):
            TicketCost.baseCost = TicketCost.baseCost + 250
        print('Total ticket cost is : ',TicketCost.baseCost)
#These details are displayed on the ticket by using this method
    def ticketDisplay(self):
        print('Ticket Details')
        print('*************************************************************')
        print('Traveller Name : ',Traveller.traveller_fn,' ', Traveller.traveller_ln)
        print('Traveller Passport Number : ',Traveller.traveller_PNo)
        print('Gender : ', Traveller.traveller_gender)
        print('Class : ', Traveller.traveller_class)
        print('Total number of bags checked in : ', Baggage.numberOfBags)
        print('Total Fare for the trip is : ',TicketCost.baseCost)
employee = Employee(5555, 'haneesh', 'male')
employee.print_details()
flight = Airline('BX1006')
```



```
class TicketCost(Baggage, Traveller, Airline):
    def __init__(self):
        TicketCost.baseCost = 250
        TicketCost.baseCost = TicketCost.baseCost + Baggage.totalBagFare
        if(Traveller.traveller_class == 'business'):
            TicketCost.baseCost = TicketCost.baseCost + 250
        print('Total ticket cost is : ',TicketCost.baseCost)
#These details are displayed on the ticket by using this method
    def ticketDisplay(self):
        print('Ticket Details')
        print('*****************************************************************')
        print('Traveller Name : ',Traveller.traveller_fn,' ', Traveller.traveller_ln)
        print('Traveller Passport Number : ',Traveller.traveller_PNo)
        print('Gender : ', Traveller.traveller_gender)
        print('Class : ', Traveller.traveller_class)
        print('Total number of bags checked in : ', Baggage.numberOfBags)
        print('Total Fare for the trip is : ',TicketCost.baseCost)
employee = Employee(5555, 'haneesh', 'male')
employee.print_details()
flight = Airline('RX1006')
traveller = Traveller()
bags = Baggage()
ticket = TicketCost()
ticket.ticketDisplay()
flight.print_details()
```

Output:



```
C:\Users\hanee\PycharmProjects\first\venv\Scripts\python.exe C:/Users/hanee/Downloads/lab3.py
Name of employee:  haneesh
Employee id:  5555
Employee gender:  male
Enter Src : vijayawada
Enter Dest : hyderabad
Airlines that are available :
DELTA
SPICEJET
INDIGO
QATAR
Name the Airlines that you prefer to travel: DELTA
Enter first name : SAI JASWANTH
Enter last name : GATTIDI
Enter passport number: RS12345
Enter gender : MALE
Business or Economy class? : Business
Number of bags you want to checkin : 4
You can take two bags for free !!! Total bag fare is for  4 is  80
Total ticket cost is :  330
Ticket Details
*****************************************************************
Traveller Name :  SAI JASWANTH  GATTIDI
Traveller Passport Number :  RS12345
Gender :  MALE
Class :  Business
Total number of bags checked in :  4
Total Fare for the trip is :  330
Airlines :  DELTA
Flight Number :  RX1006
vijayawada --> hyderabad

Process finished with exit code 0
```

- We have created an Airline class that will take input from the terminal like source, destination, and preferred airlines then airline class will generate the flight number for the provided details.
- Employee class will display the details of the employee. Employee class will use inheritance concept and overrides the printing method of the Airline class.
- Traveller class will take all the details of the traveller. Baggage class would take the number of check-in bags from the traveller and calculate the baggage fare. TicketCost class takes the inputs from all the classes and calculates the fare of the ticket. A method in TicketCost would display all the ticket details.

4.

Code:

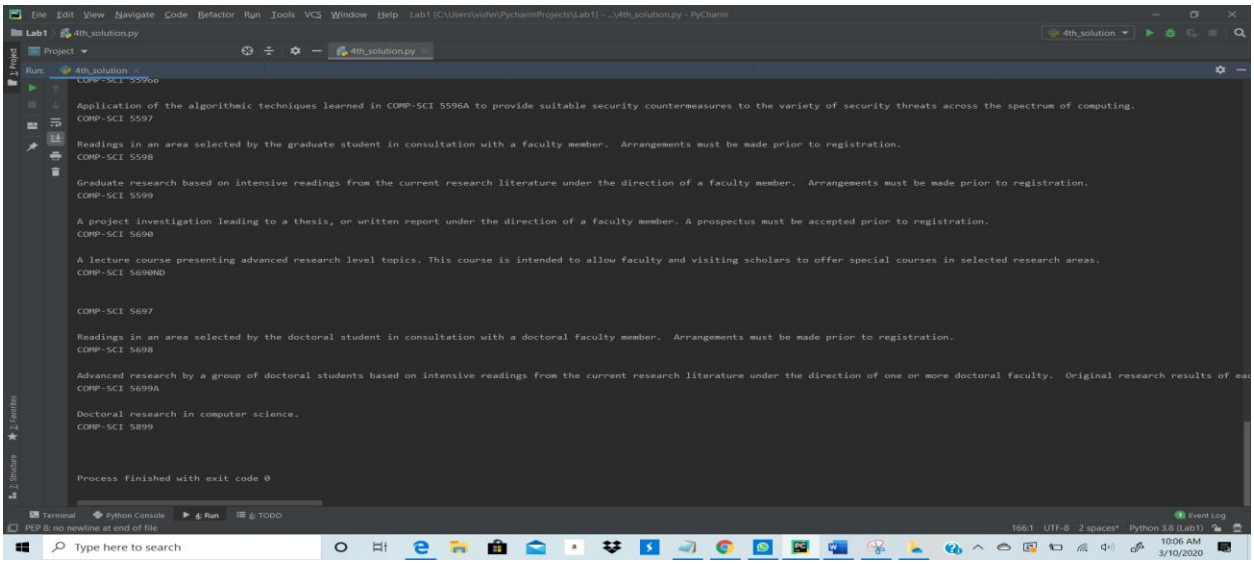
```python
import requests
from bs4 import BeautifulSoup
url = 'https://catalog.umkc.edu/course-offerings/graduate/comp-sci/'
res = requests.get(url)
html_page = res.content
soup = BeautifulSoup(html_page, 'html.parser')
text = soup.find_all(text=True)
# Finding all the divs with class courseblock
res = soup.find_all('div', {'class': 'courseblock'})
# printing all the course names and course descriptions
for a in res:
  res1 = a.find('span', {'class': 'code'}).text
  res2 = a.find('p', {'class': 'courseblockdesc'}).text
  print(res1)
  print(res2)
```

Output:

```
C:\Users\vishn\PycharmProjects\Lab1\venv\Scripts\python.exe C:/Users/vishn/PycharmProjects/Lab1/4th_solution.py
COMP-SCI 5101

A review of mathematical logic, sets, relations, functions, mathematical induction, and algebraic structures with emphasis on computing applications. Recurrence relations and their use in the analysis
COMP-SCI 5102

This course covers concurrency and control of asynchronous processes, deadlocks, memory management, processor and disk scheduling, parallel processing, and file system organization in operating system
COMP-SCI 5103

A review of linear and hierarchical data structures, including stacks, queues, lists, trees, priority queues, advanced tree structures, hashing tables, dictionaries and disjoint-sets. Asymptotic analys
COMP-SCI 5514

Fiber optic cable and its characteristics, optical sources and transmitters, optical detectors and receivers, optical components such as couplers and connectors, WDM and OFDM techniques, modulation and
COMP-SCI 5525

Cloud computing systems operate in a very large scale, and are impacting the economics and the assumptions behind computing significantly. This special topics course provides a comprehensive overview o
COMP-SCI 5531

Components of an operating system, scheduling/routing mechanisms, process control blocks, design and test various operating system components.
COMP-SCI 5540

This course will introduce the essential characteristics of Big Data and why it demands rethinking how we store, process, and manage massive amounts of structured and unstructured data. It will cover t
COMP-SCI 5542

Big Data analytics focus on analyzing large amounts of data to find useful information and to make use of the information for better business decisions. This course introduces students to the practice
COMP-SCI 5543

This course teaches students fundamental theory and practice in the field of big data analytics and real time distributed systems for real time big data applications. In this course, students will have
COMP-SCI 5551

Current concepts in software architecture and design, comparative analysis for design, object-oriented software design, software quality criteria for evaluation of software design. Introduction to metr
COMP-SCI 5552A
```

```
COMP-SCI 5553

The course introduces a number of basic concepts and enabling technologies of software architecture, including architecture styles, architecture description languages, architecture-implementation mappi
COMP-SCI 5555

Software methods and tools are extensively used in current software production to improve software productivity and quality. In this course, we are going to learn a number of popular software methods a
COMP-SCI 5560

This course teaches students fundamental theory and practice in the field of knowledge discovery and management and also provides them with hands-on experience through application development.
COMP-SCI 5561

AI systems and their languages, implementations and applications, case studies of various expert systems, current research topics in AI, logic programming using PROLOG.
COMP-SCI 5565

Introduction to Machine Learning; Multivariate Distributions; Information Theory; Linear Algebra (Eigenanalysis); Supervised/Unsupervised Learning, Classification/Regression; Linear/Non-linear Learning
COMP-SCI 5566

This course introduces students to the field of Bioinformatics with a focus on understanding the motivation and computer science behind existing Bioinformatic resources, as well as learning the skills
COMP-SCI 5567

This course teaches the theoretical basis of methods for learning from data, illustrated by examples of applications to several domains. Recommended preparation: COMP-SCI 5565.
COMP-SCI 5568

Many real world systems are probabilistic in nature. Probability theory gives us the basic tools for modeling many real world systems, allowing us to understand complex behavior. Probabilistic graphica
COMP-SCI 5570

Covers in detail, architecture of centralized database systems, database processing, management of concurrent transactions, query processing, query optimization, data models, database recovery, datawar
COMP-SCI 5572

This course covers in detail the architecture of mobile and wireless network. It discusses and develops reveland concepts and algorithms for building mobile database systems (MDS), which is necessary f
COMP-SCI 5573

This course deals with information security and assurance and covers the concepts necessary to secure the cyberspace. It introduces security models, assurance policies, security policies and procedures
```

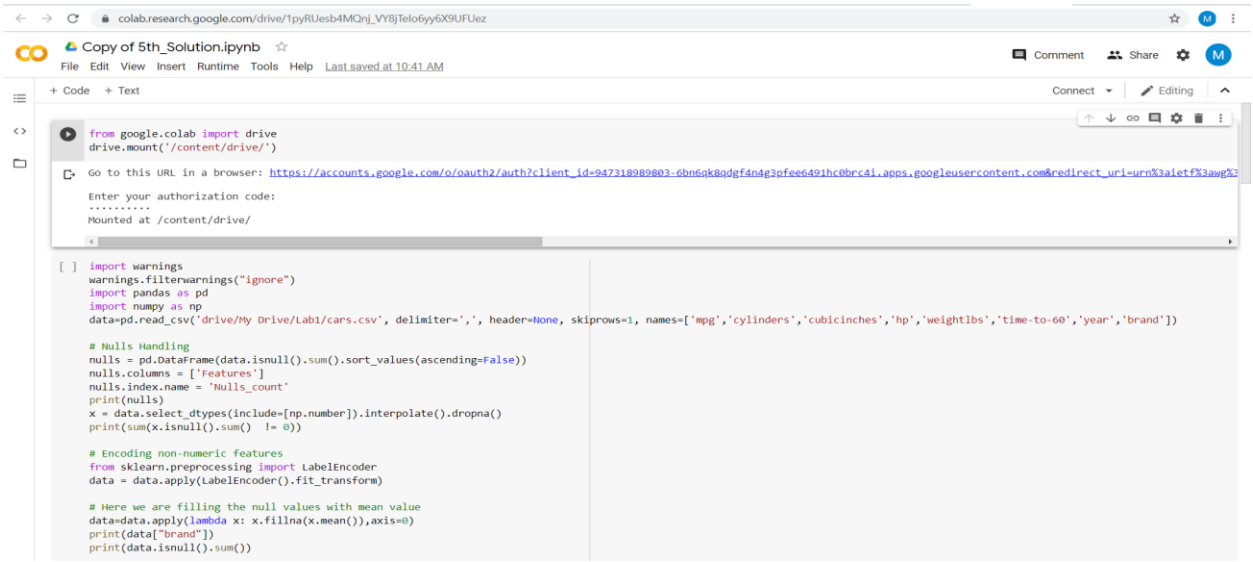
```
COMP-SCI 5574

This course will cover topics related to managing large scale semistructured data modeled using the Extensible Markup Language XML and the Resource Description Framework (RDF). This will include storin
COMP-SCI 5581

Parallelism in computer architecture, pipelined processors, array processors and multi-processor systems, algorithms for SISD, SIMD, MISD and MIMD organizations, vectorization, pipelining algorithms.
COMP-SCI 5582

The image is an essential form of information representation and communication in modern society. This course focuses on topics of computer vision, teaching computer how to understand images. Topics in
COMP-SCI 5590

Selected topics in specific areas of computer science.  May be repeated for credit when the topic varies.
COMP-SCI 5590AW


COMP-SCI 5590BD


COMP-SCI 5590CC


COMP-SCI 5590CI


COMP-SCI 5590CN


COMP-SCI 5590HI


COMP-SCI 5590MT


COMP-SCI 5590NN
```



- Using Beautifulsoup we will first get the content of the webpage and converted all the html content to plain text.
- Then we have inspected the webpage and we get to know that courseblock represents the subject name, code represents the subject code and courseblockdesc gives the description of the subject.
- Then we find all the div tags with class:courseblock. Then we have printed all the course code and their respective descriptions from the given URL.

5(a).

Code & Output:



colab.research.google.com/drive/1pyRUesb4MQnj_VY8jTeIo6yy6X9UFUez

Copy of 5th_Solution.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:41 AM

Comment Share

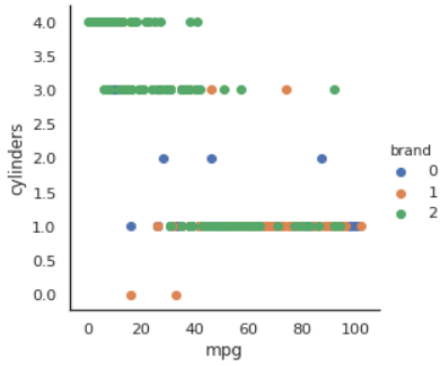
+ Code + Text

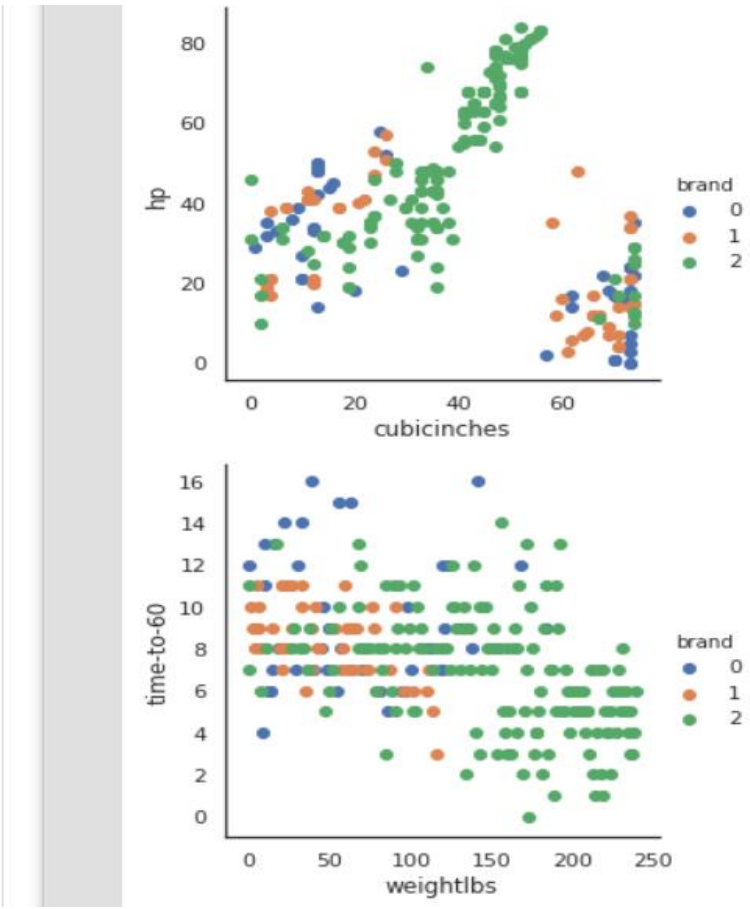
Connect Editing

```
# Visualize data to analyze our features correlations
import seaborn as sns
sns.set(style="white", color_codes=True)
import matplotlib.pyplot as plt
sns.FacetGrid(data, hue='brand', height=4).map(plt.scatter, 'mpg', 'cylinders').add_legend()
plt.show()
sns.FacetGrid(data, hue='brand', height=4).map(plt.scatter, 'cubicinches', 'hp').add_legend()
plt.show()
sns.FacetGrid(data, hue='brand', height=4).map(plt.scatter, 'weightlbs', 'time-to-60').add_legend()
plt.show()

# Split data into train and test
from sklearn.model_selection import train_test_split
x = data.drop(['brand'], axis=1)
y = data['brand']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size =0.2, random_state=0)
```

```
            Features
Nulls_count
brand             0
year              0
time-to-60        0
weightlbs         0
hp                0
cubicinches       0
cylinders         0
mpg               0
0
0     2
1     0
2     2
3     2
4     2
     ..
```

```
     ..
256    2
257    1
258    2
259    2
260    2
Name: brand, Length: 261, dtype: int64
mpg            0
cylinders      0
cubicinches    0
hp             0
weightlbs      0
time-to-60     0
year           0
brand          0
dtype: int64
```





- In this program we will first import the required libraries.
  - We have taken cars.csv as the sample dataset which contains both the numeric and non-numeric data. We have applied exploratory data analysis and cleaned the dataset by handling all the null values by filling them with the mean values and encoded the non-numeric features using Label encoder.
- Then we have visualized all the correlation features and plotted them using seaborn. Then by using Train_test_split method we have divided the dataset into train and test datasets with test dataset of size = 0.2 of the cars.csv dataset.

5(b)

Code & Output:

colab.research.google.com/drive/1pyRUesb4MQnj_VY8jTeIo6yy6X9UFUez#scrollTo=fQV_U1Vjnh4M

Copy of 5th_Solution.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:41 AM

+ Code + Text

```
# KNN method
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(x_train, y_train)
# Evaluate model
score = round(knn.score(x_train, y_train)* 100, 2)
print('K-Neighbors accuracy training score: ', score)
print('Classification report:')
y_pred = knn.predict(x_test)
print(classification_report(y_test, y_pred))
```

```
K-Neighbors accuracy training score:  85.1
Classification report:
              precision    recall  f1-score   support

           0       0.58      0.88      0.70         8
           1       0.75      0.50      0.60        12
           2       0.88      0.88      0.88        33

    accuracy                           0.79        53
   macro avg       0.74      0.75      0.73        53
weighted avg       0.81      0.79      0.79        53
```

colab.research.google.com/drive/1pyRUesb4MQnj_VY8jTeIo6yy6X9UFUez#scrollTo=fQV_U1Vjnh4M

Copy of 5th_Solution.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:41 AM

+ Code + Text

```
# Naive Bayes method
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report
nb = GaussianNB()
nb.fit(x_train, y_train)
# Evaluate model
score = round(nb.score(x_train, y_train)* 100, 2)
print('Naive Bayes accuracy training score: ', score)
print('Classification report:')
y_pred = nb.predict(x_test)
print(classification_report(y_test, y_pred))
```

```
Naive Bayes accuracy training score:  72.12
Classification report:
              precision    recall  f1-score   support

           0       0.40      0.75      0.52         8
           1       0.64      0.58      0.61        12
           2       0.89      0.73      0.80        33

    accuracy                           0.70        53
   macro avg       0.64      0.69      0.64        53
weighted avg       0.76      0.70      0.71        53
```

colab.research.google.com/drive/1pyRUesb4MQnj_VY8jTeIo6yy6X9UFUez#scrollTo=fQV_U1Vjnh4M

Copy of 5th_Solution.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:41 AM

+ Code + Text

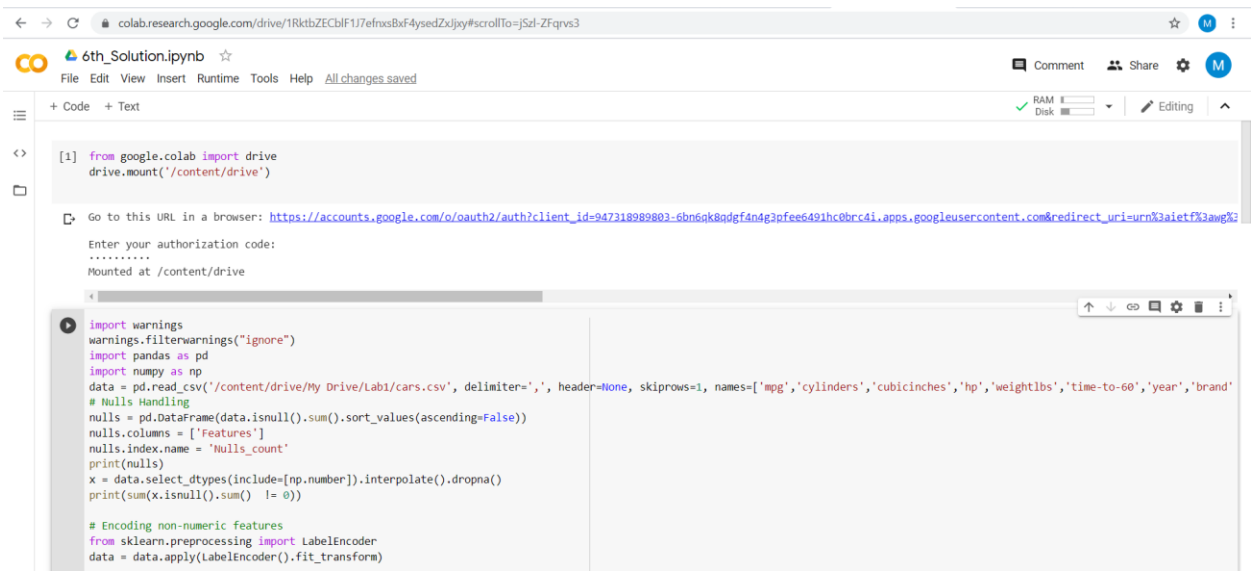
```
# SVM method
from sklearn.svm import SVC, LinearSVC
svc = SVC()
svc.fit(x_train, y_train)
# Evaluate model
score = round(svc.score(x_train, y_train)* 100, 2)
print('Support Vector Machines score: ', score)
print('Classification report:')
y_pred = svc.predict(x_test)
print(classification_report(y_test, y_pred))
```

```
Support Vector Machines score:  70.19
Classification report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         8
           1       0.45      0.42      0.43        12
           2       0.74      0.94      0.83        33

    accuracy                           0.68        53
   macro avg       0.40      0.45      0.42        53
weighted avg       0.56      0.68      0.61        53
```

- After Splitting the dataset into train and test datasets we have applied all the three classification algorithms: K-nearest neighbors, Naïve bayes, and support vector machines and fitted the models with the training data.
- Calculated the accuracy score for each of the classification algorithm and found that ‘k-Nearest neighbors’ with k=3 has the highest accuracy 85.1 when compared with other two classification algorithms ‘support vector machines’ (70.10), and ‘Naïve bayes’ (72.12)

6(a).

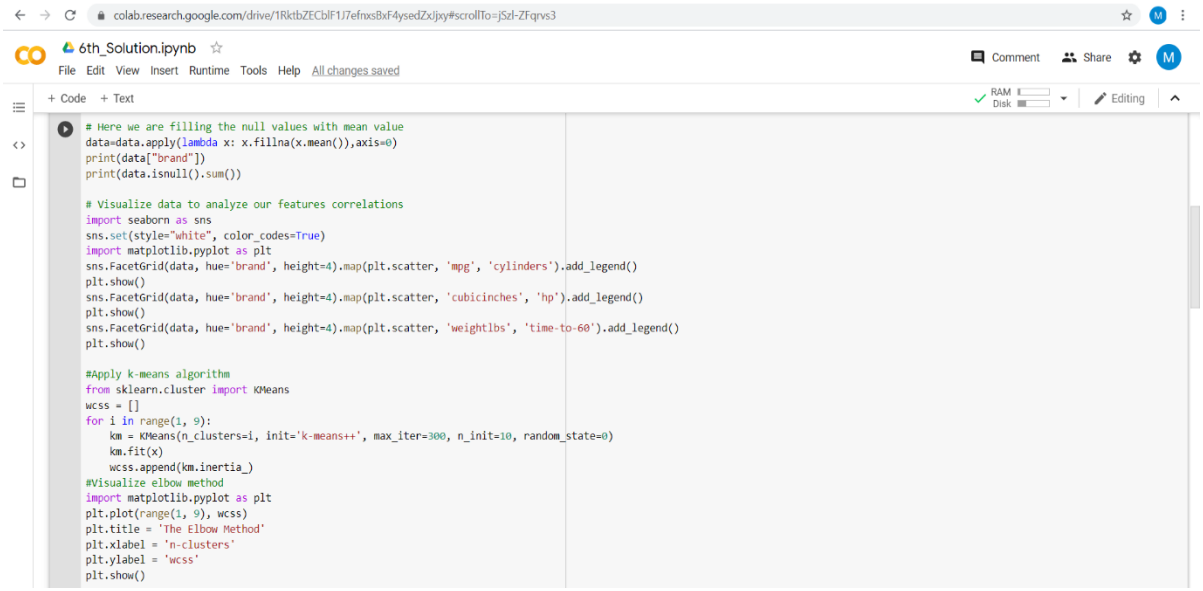
Code & Output:



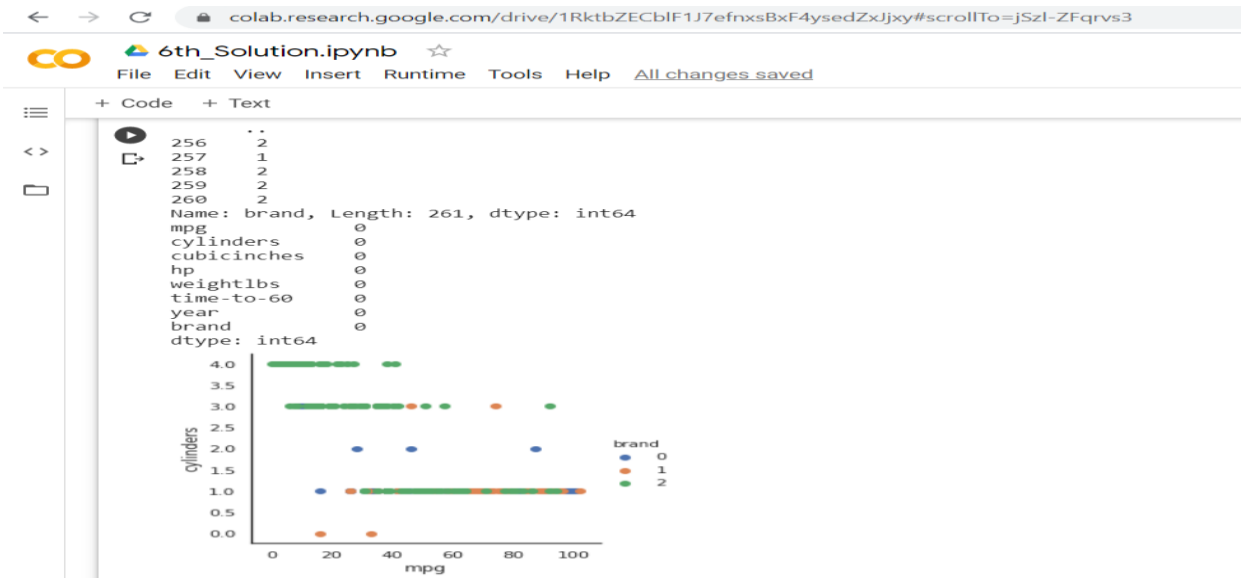
```
[1] from google.colab import drive
    drive.mount('/content/drive')
```

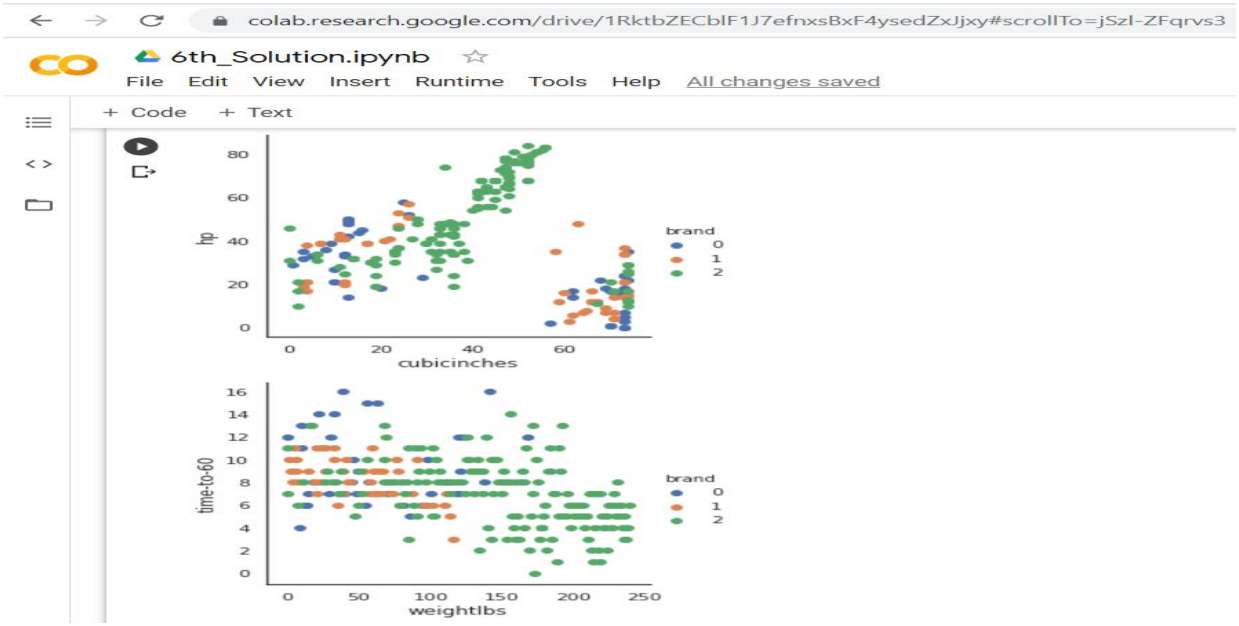
```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3aietf%3awg%2

Enter your authorization code:
··· ·····
Mounted at /content/drive
```

```
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
data = pd.read_csv('/content/drive/My Drive/Lab1/cars.csv', delimiter=',', header=None, skiprows=1, names=['mpg','cylinders','cubicinches','hp','weightlbs','time-to-60','year','brand'
# Nulls Handling
nulls = pd.DataFrame(data.isnull().sum().sort_values(ascending=False))
nulls.columns = ['Features']
nulls.index.name = 'Nulls_count'
print(nulls)
x = data.select_dtypes(include=[np.number]).interpolate().dropna()
print(sum(x.isnull().sum()  != 0))

# Encoding non-numeric features
from sklearn.preprocessing import LabelEncoder
data = data.apply(LabelEncoder().fit_transform)
```



```
# Here we are filling the null values with mean value
data=data.apply(lambda x: x.fillna(x.mean()),axis=0)
print(data["brand"])
print(data.isnull().sum())

# Visualize data to analyze our features correlations
import seaborn as sns
sns.set(style="white", color_codes=True)
import matplotlib.pyplot as plt
sns.FacetGrid(data, hue='brand', height=4).map(plt.scatter, 'mpg', 'cylinders').add_legend()
plt.show()
sns.FacetGrid(data, hue='brand', height=4).map(plt.scatter, 'cubicinches', 'hp').add_legend()
plt.show()
sns.FacetGrid(data, hue='brand', height=4).map(plt.scatter, 'weightlbs', 'time-to-60').add_legend()
plt.show()

#Apply k-means algorithm
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 9):
    km = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=0)
    km.fit(x)
    wcss.append(km.inertia_)
#Visualize elbow method
import matplotlib.pyplot as plt
plt.plot(range(1, 9), wcss)
plt.title = 'The Elbow Method'
plt.xlabel = 'n-clusters'
plt.ylabel = 'wcss'
plt.show()
```





- We have taken cars.csv as the sample dataset which contains both the numeric and non-numeric data. We have applied exploratory data analysis and cleaned the dataset by handling all the null values by filling them with the mean values and encoded the non-numeric features using Label encoder.
- Then we have visualized all the correlation features and plotted them using seaborn.
- Used K-means clusterings to get the best value for k which is 2 by using the elbow method.

6(b).

Code & Output:

colab.research.google.com/drive/1RktbZECblF1J7efnxsBxF4ysedZxJjxy#scrollTo=jSzl-ZFqrvs3

6th_Solution.ipynb

File Edit View Insert Runtime Tools Help All changes saved

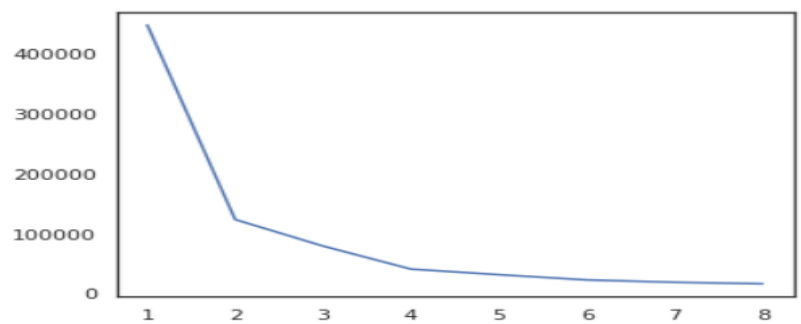
Comment Share

+ Code + Text

RAM Disk Editing

```
#Found k=2
km = KMeans(n_clusters=2)
from sklearn.metrics import silhouette_score
km.fit(x)
x_pred = km.predict(x)
print('Silhouette score for k=2:', silhouette_score(x, x_pred))
#Found k=3
km = KMeans(n_clusters=3)
from sklearn.metrics import silhouette_score
km.fit(x)
x_pred = km.predict(x)
print('Silhouette score for k=3:', silhouette_score(x, x_pred))
#Found k=4
km = KMeans(n_clusters=4)
from sklearn.metrics import silhouette_score
km.fit(x)
x_pred = km.predict(x)
print('Silhouette score for k=4:', silhouette_score(x, x_pred))
```

```
             Features
Nulls_count
brand               0
year                0
time-to-60          0
weightlbs           0
hp                  0
cubicinches         0
cylinders           0
mpg                 0
0
0      2
1      0
2      2
3      2
4      2
      ..
```

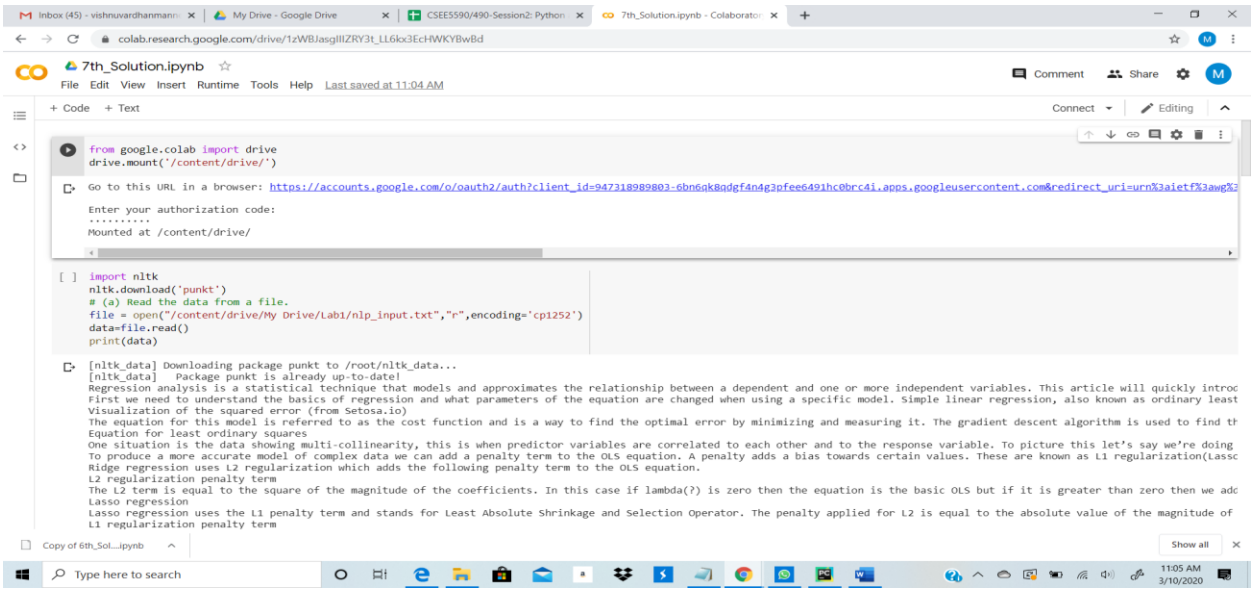



```
Silhouette score for k=2: 0.656735637586314
Silhouette score for k=3: 0.5891108798858462
Silhouette score for k=4: 0.5187172839639153
```

- Calculated the Silhouette score for k=2,3,4 and when the number of clusters k=2 we got the highest silhouette score which is 0.656

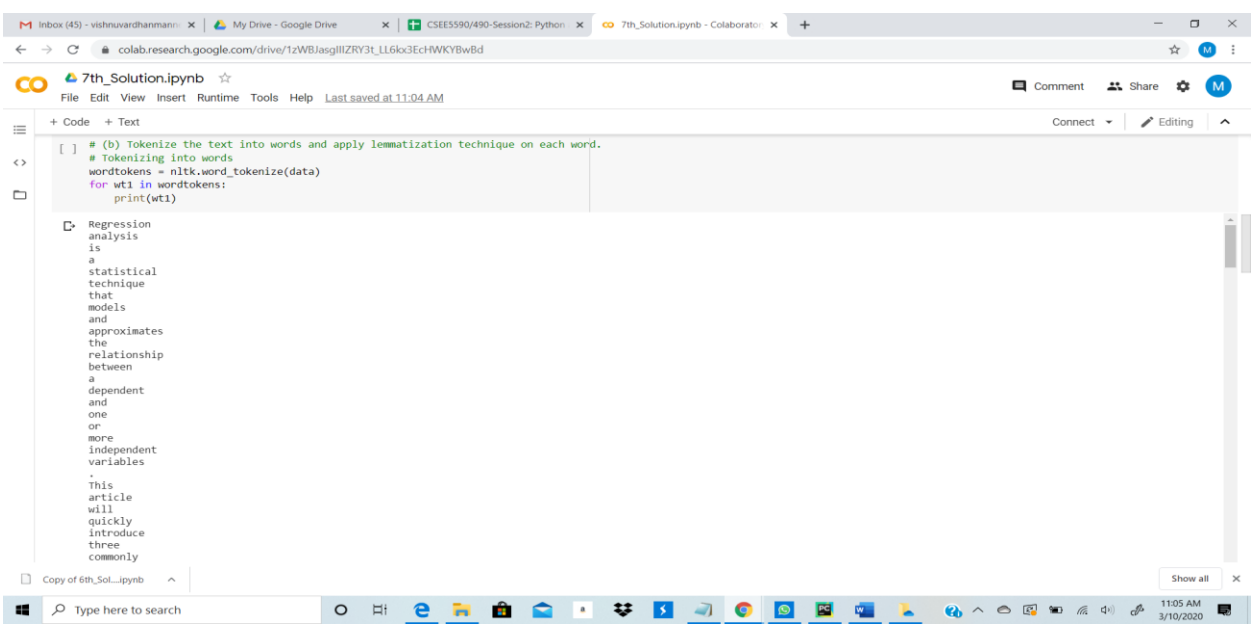
7(a).

Code & Output:

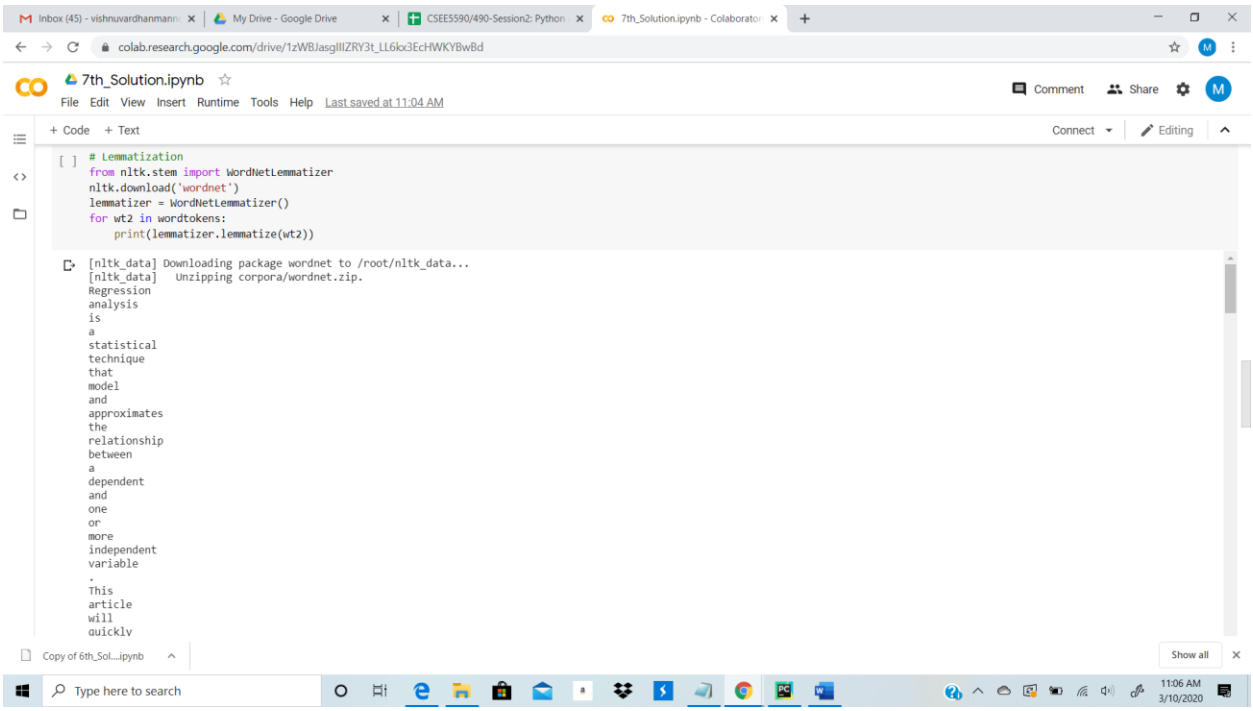


- Import the nltk library and then read the given nlp_input.txt file with encoding cp1252.
- Used file.read() to get the complete data in the give file and printed the result.

7(b).



- We have used word_tokenize function for tokenizing text into words.

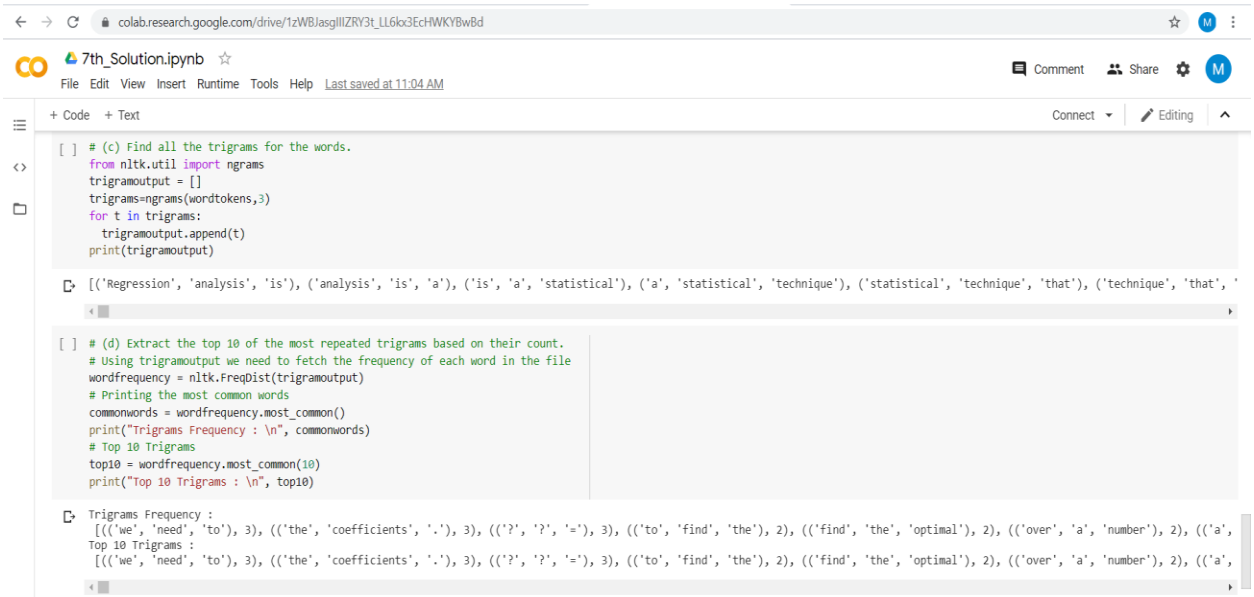


```
# Lemmatization
from nltk.stem import WordNetLemmatizer
nltk.download('wordnet')
lemmatizer = WordNetLemmatizer()
for wt2 in wordtokens:
    print(lemmatizer.lemmatize(wt2))
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
Regression
analysis
is
a
statistical
technique
that
model
and
approximates
the
relationship
between
a
dependent
and
one
or
more
independent
variable
.
This
article
will
quickly
```

- We have imported wordnetlemmatizer and applied lemmatization on each word.

7(c,d).



```
# (c) Find all the trigrams for the words.
from nltk.util import ngrams
trigramoutput = []
trigrams=ngrams(wordtokens,3)
for t in trigrams:
  trigramoutput.append(t)
print(trigramoutput)
```

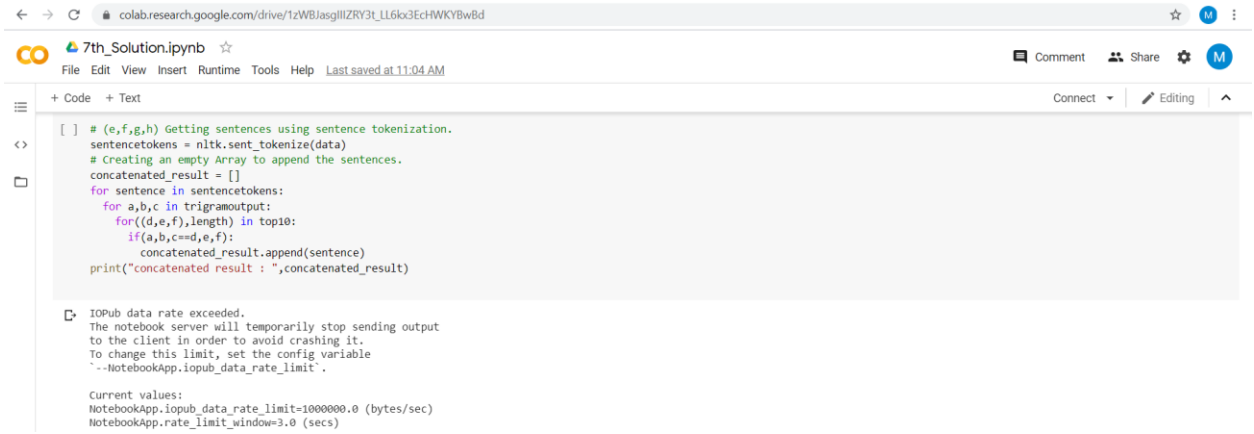
```
[('Regression', 'analysis', 'is'), ('analysis', 'is', 'a'), ('is', 'a', 'statistical'), ('a', 'statistical', 'technique'), ('statistical', 'technique', 'that'), ('technique', 'that', '
```

```
# (d) Extract the top 10 of the most repeated trigrams based on their count.
# Using trigramoutput we need to fetch the frequency of each word in the file
wordfrequency = nltk.FreqDist(trigramoutput)
# Printing the most common words
commonwords = wordfrequency.most_common()
print("Trigrams Frequency : \n", commonwords)
# Top 10 Trigrams
top10 = wordfrequency.most_common(10)
print("Top 10 Trigrams : \n", top10)
```

```
Trigrams Frequency :
 [(('we', 'need', 'to'), 3), (('the', 'coefficients', '.'), 3), (('?', '?', '='), 3), (('to', 'find', 'the'), 2), (('find', 'the', 'optimal'), 2), (('over', 'a', 'number'), 2), (('a',
Top 10 Trigrams :
 [(('we', 'need', 'to'), 3), (('the', 'coefficients', '.'), 3), (('?', '?', '='), 3), (('to', 'find', 'the'), 2), (('find', 'the', 'optimal'), 2), (('over', 'a', 'number'), 2), (('a',
```

- Imported the ngrams library and the value of n should be 3 in order to obtain all the trigrams.
- To extract the top 10 most repeated trigrams first we used FreqDist function to obtain the word frequency of trigrams. Then have used most_common(10) function to get the top 10 trigrams.

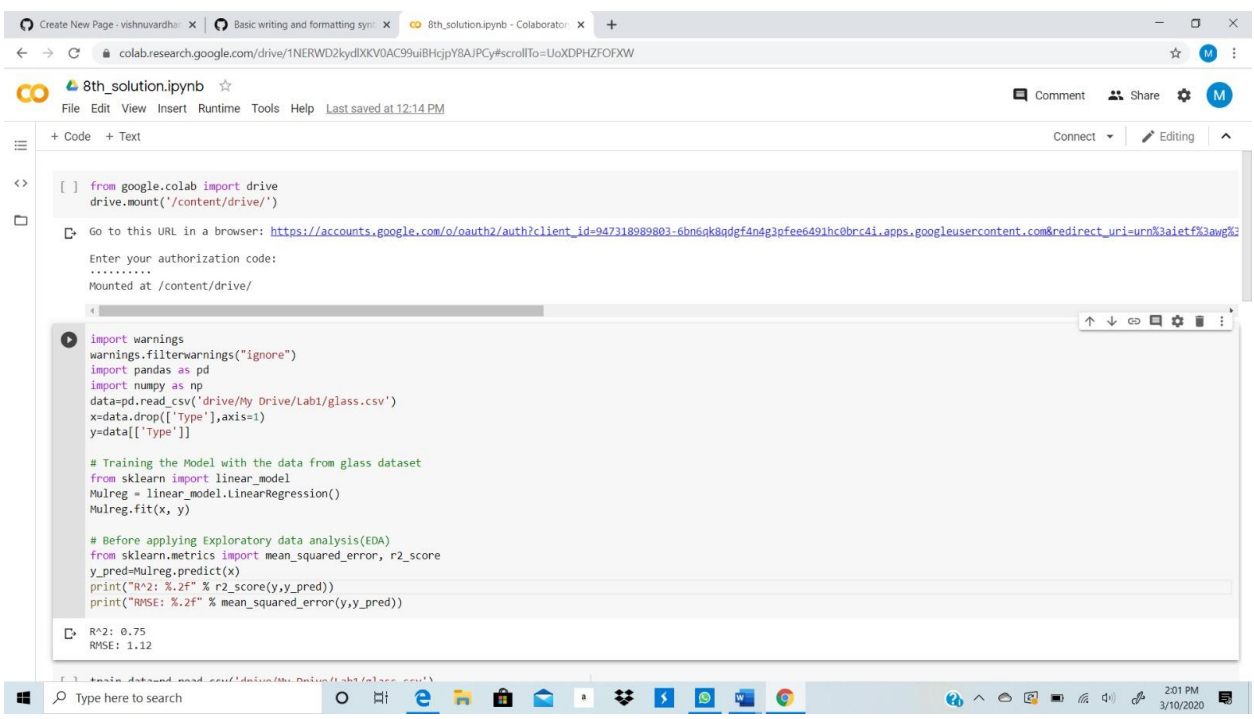
7(e,f,g).

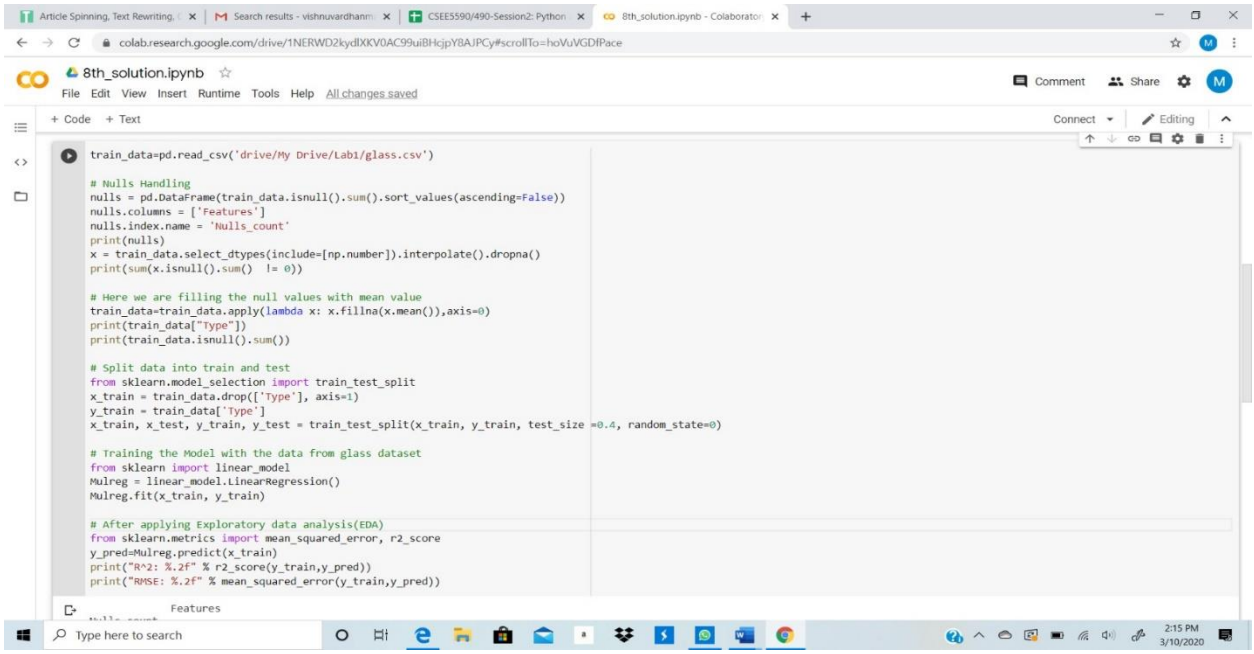


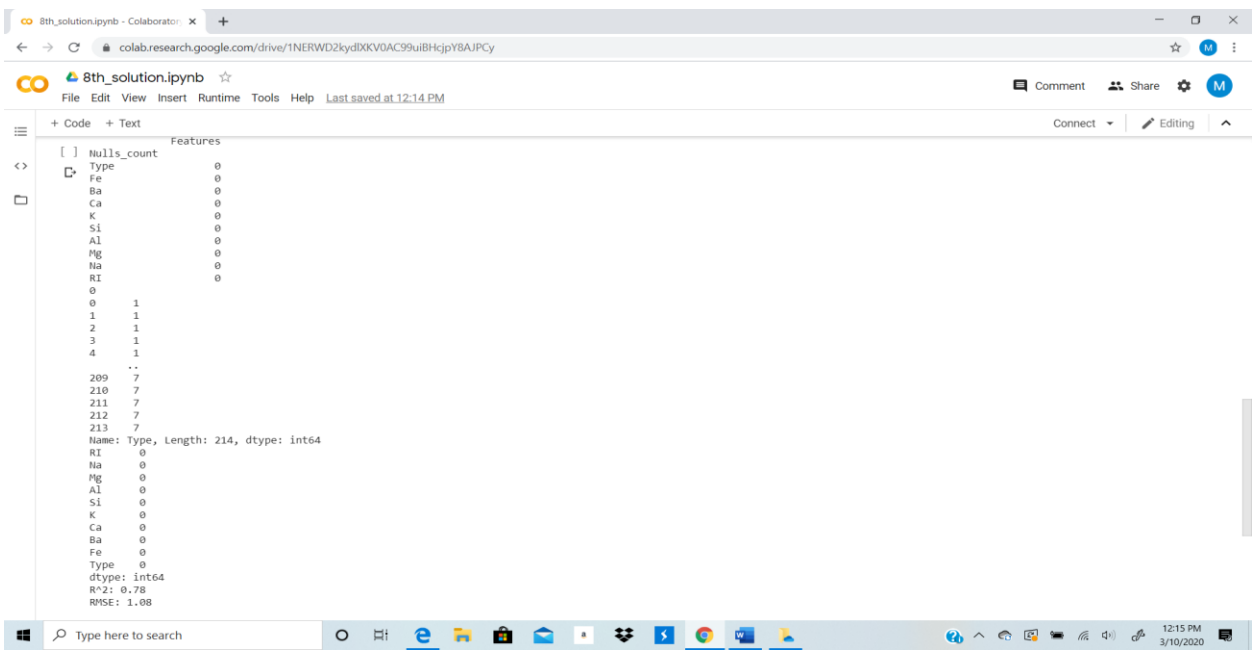
- We have used sent_tokenize to get every sentence and created an empty array to append each sentence. Then we have iterated sentence with trigrams to get sentence with most trigrams with most recurrence and concatenated the result and printed the output.

8.

Code & Output:







- Imported the required libraries and we have used glass.csv dataset. Dropped the target variable 'Type' from training data and used linear regression for fitting the model. Before applying the Exploratory data analysis we have calculated the RMSE and R squared scores.
- Now we have cleaned the dataset by applying exploratory data analysis and used linear regression for fitting the model again. Now again we have calculated the RMSE and R squared scores.

- Before applying EDA the R_squared and RMSE values are 0.75 and 1.12.
- After applying EDA the R_squared and RMSE values are 0.78 and 1.08.
- The R_squared value is increased and RMSE value is decreased slightly after cleaning the dataset.

Workflow:

The work flow for all algorithms in this lab assignment is as follows:

- Created Python files using PyCharm IDE and used Google colab.
- Downloaded the datasets from Kaggle.
- Pre-processing of the data.
- Split the data into train and test using train_test_split().
- Trained the model with that data.
- Metrics calculation and plotting the data using matplotlib and seaborn.

Datasets:

- Cars.csv
- Glass.csv

Evaluation & Discussion:

We ensured that all the objectives of the lab assignment are met.

Conclusion:

We have completed the assignment on fundamentals of python and have trained the classifier with models like SVM, KNN, Naive Byes. We have also applied K-means clustering and used Elbow method and silhouette score to find the best k value and also applied Multiple regression technique to dataset and evaluated R2, RMSE scores before and after EDA successfully.