Nikit Gokhe

Class: TY Comp D1

Roll No. 324022

Gr No. 21810522

Assignment 1

Problem Statement:

Perform the following using R/Python on suitable data sets, read data from different formats (like csv, xls), indexing and selecting data, sort data, describe attributed of data, checking data types of each column, counting unique values of data, format of each column, converting variable data type, identifying missing values and fill in the missing values.

Objectives:

1. Change datatype of the attribute wherever required.
2. Get rid of the missing values from the data.
3. Drop the not useful columns.

Theory:

Why Python for Data Science?: Python is open source, interpreted, high level language and provides great approach for object-oriented programming. It is one of the best languages used by data scientist for various data science projects/application. Python provide great functionality to deal with mathematics, statistics, and scientific function. It provides great libraries to deals with data science application

Pandas in Python: Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc. In this tutorial, we will learn the various features of Python Pandas and how to use them in practice.

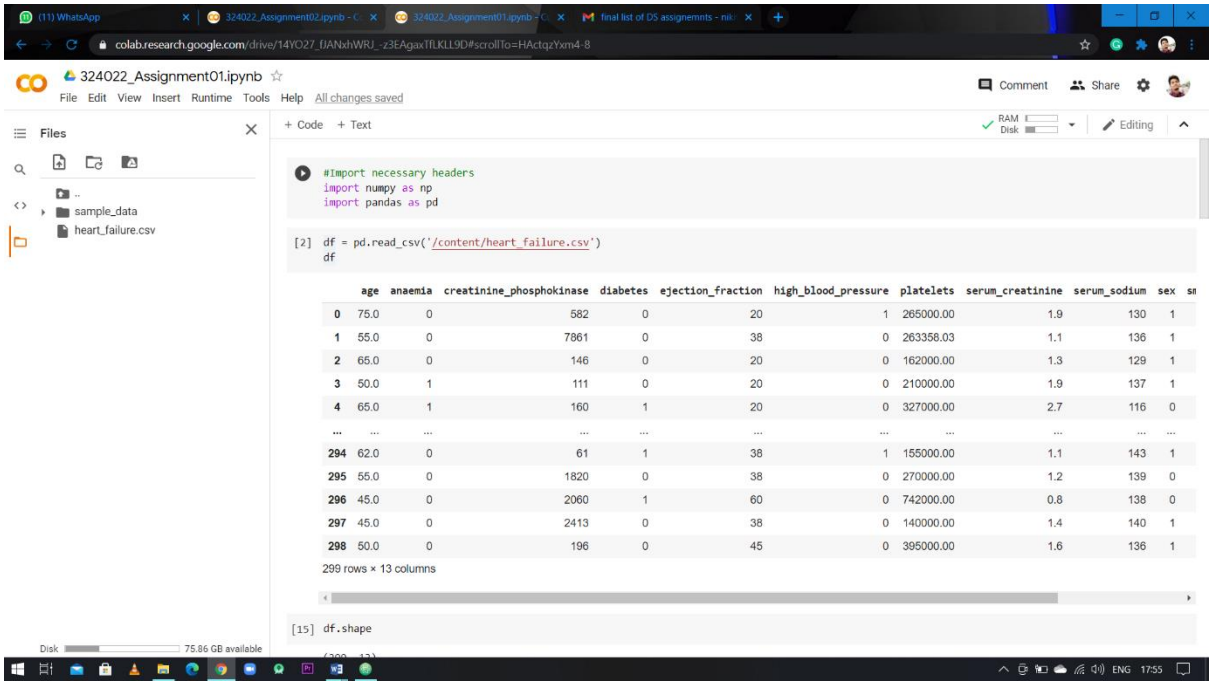
NumPy in Python: NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. An introduction to Matplotlib is also provided. All this is explained with the help of examples for better understanding.

Dataset:

Heart_failure dataset obtained from Kaggle.com

Link: https://www.kaggle.com/sagar029/heart-failure

Expected Output/sample code:



324022_Assignment01.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Files

- ..
- sample_data
- heart_failure.csv

[15]
```
df.shape
```

```
(299, 13)
```

[16]
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 299 entries, 0 to 298
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   age                       299 non-null    float64
 1   anaemia                   299 non-null    int64
 2   creatinine_phosphokinase  299 non-null    int64
 3   diabetes                  299 non-null    int64
 4   ejection_fraction         299 non-null    int64
 5   high_blood_pressure       299 non-null    int64
 6   platelets                 299 non-null    float64
 7   serum_creatinine          299 non-null    float64
 8   serum_sodium              299 non-null    int64
 9   sex                       299 non-null    int64
 10  smoking                   299 non-null    int64
 11  time                      299 non-null    int64
 12  DEATH_EVENT               299 non-null    int64
dtypes: float64(3), int64(10)
memory usage: 30.5 KB
```

[17]
```
df.dtypes
```

```
age                       float64
anaemia                     int64
creatinine_phosphokinase    int64
diabetes                    int64
ejection_fraction           int64
high_blood_pressure         int64
platelets                 float64
```

Disk 75.86 GB available

[3]
```
df.head()
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smok
0	75.0	0	582	0	20	1	265000.00	1.9	130	1	
1	55.0	0	7861	0	38	0	263358.03	1.1	136	1	
2	65.0	0	146	0	20	0	162000.00	1.3	129	1	
3	50.0	1	111	0	20	0	210000.00	1.9	137	1	
4	65.0	1	160	1	20	0	327000.00	2.7	116	0	

[4]
```
#Make a different copy for doing all the changes and processing
dfTemp = df.copy()
```

[5]
```
df.tail(3)
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	sm
296	45.0	0	2060	1	60	0	742000.0	0.8	138	0	
297	45.0	0	2413	0	38	0	140000.0	1.4	140	1	
298	50.0	0	196	0	45	0	395000.0	1.6	136	1	

[12]
```
df.columns
```

```
Index(['age', 'anaemia', 'creatinine_phosphokinase', 'diabetes',
       'ejection_fraction', 'high_blood_pressure', 'platelets',
       'serum_creatinine', 'serum_sodium', 'sex', 'smoking', 'time',
       'DEATH_EVENT'],
      dtype='object')
```

Disk 75.86 GB available

324022_Assignment01.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Files

- ..
- sample_data
- heart_failure.csv

[14]
```
c_df = df[['age','sex']]
print(c_df)
```

```
      age  sex
0    75.0    1
1    55.0    1
2    65.0    1
3    50.0    1
4    65.0    0
..    ...  ...
294  62.0    1
295  55.0    0
296  45.0    0
297  45.0    1
298  50.0    1

[299 rows x 2 columns]
```

[7]
```
dfTemp.describe()
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	seru
count	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.00000	29
mean	60.833893	0.431438	581.839465	0.418060	38.083612	0.351171	263358.029264	1.39388	13
std	11.894809	0.496107	970.287881	0.494067	11.834841	0.478136	97804.236869	1.03451	
min	40.000000	0.000000	23.000000	0.000000	14.000000	0.000000	25100.000000	0.50000	11
25%	51.000000	0.000000	116.500000	0.000000	30.000000	0.000000	212500.000000	0.90000	13
50%	60.000000	0.000000	250.000000	0.000000	38.000000	0.000000	262000.000000	1.10000	13
75%	70.000000	1.000000	582.000000	1.000000	45.000000	1.000000	303500.000000	1.40000	14
max	95.000000	1.000000	7861.000000	1.000000	80.000000	1.000000	850000.000000	9.40000	14

Disk 75.86 GB available

324022_Assignment01.ipynb

```
[8] print("Min Age : ", dfTemp['age'].min())
    print("Max Age : ", dfTemp['age'].max())
```

```
Min Age :  40.0
Max Age :  95.0
```

```
[9] dfTemp[['age', 'anaemia', 'diabetes']][dfTemp['diabetes'] == 1]
```

	age	anaemia	diabetes
4	65.0	1	1
7	60.0	1	1
19	48.0	1	1
21	65.0	1	1
23	53.0	0	1
...	...	...	...
290	45.0	0	1
292	52.0	0	1
293	63.0	1	1
294	62.0	0	1
296	45.0	0	1

125 rows × 3 columns

```
[10] #Number of unique values of columns
     dfTemp.nunique()
```

324022_Assignment01.ipynb

```
[10] #Number of unique values of columns
     dfTemp.nunique()
```

```
age                       47
anaemia                    2
creatinine_phosphokinase 208
diabetes                   2
ejection_fraction         17
high_blood_pressure        2
platelets                176
serum_creatinine          40
serum_sodium              27
sex                        2
smoking                    2
time                     148
DEATH_EVENT                2
dtype: int64
```

```
[11] #Count of each unique value in a column
     print(dfTemp['sex'].value_counts())
```

```
1    194
0    105
Name: sex, dtype: int64
```

```
[ ]
```

Conclusion:

using R/Python on suitable data sets, read data from different formats (like csv, xls), indexing and selecting data, sort data, describe attributed of data, checking data types of each column, counting unique values of data, format of each column, converting variable data type, identifying missing values and fill in the missing values.