U18ISI6204 – Machine Learning Techniques

LAB- EXPERIMENT 9

-

NAME: Aaron Mathew

ROLL_NO: 20BIS001

Using Principal component Analysis as Dimensionality reduction component implement Logistic Regression for detecting credit card frauds

OBJECTIVE OF THE EXERCISE/EXPERIMENT

To perform logistic regression along with PCA on the given dataset, using scikit library

STEP 2: ACQUISITION

PROCEDURE:

STEP-1: Start the program.

STEP-2: import all the necessary libraries

iv) Numpy – array manipulation
v) Pandas – dataframe manipulation
vi) Matplotlib and seaborn – for data visualization
vii) Sklearn.model_selection – train test data split
viii) Sklearn.metrics –f1 score.
ix) Sklearn,linear_model– for logistic regression
x) Sklearn.decomposition – for PCA
xi) Sklearn.preprocessing – for Normalisation

STEP-3: Loading the dataset using read_csv method in pandas module.

STEP-4: Analyze the dataset using info method, which gives its data types and number of non- null values in each columns.

STEP-5: Perform basic statistic operation using describe() method.

STEP-6: Use heatmaps, correlation matrix, regression plots and pairplots in seaborn to find the relationship between features.

STEP-7: Normalize the data points

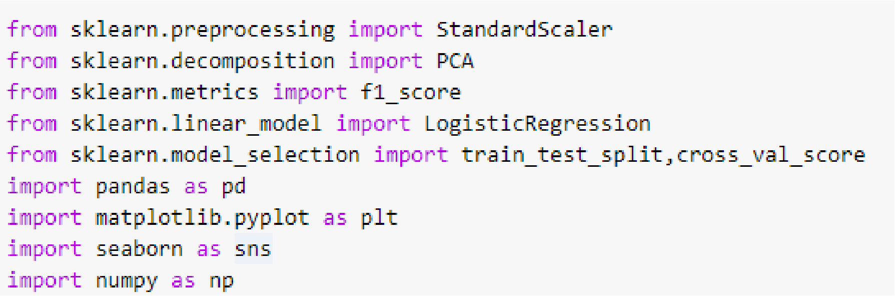
STEP-8: Using selective feature, perform PCA in order to reduce number of feature from 30 to 11.

STEP-9: Implement logistic regression with 11 PCA variable and calculate f1 score.

STEP-10: Stop the program.

PROGRAM:

```
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition  import PCA
from sklearn.metrics import f1_score
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, cross_val_score
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

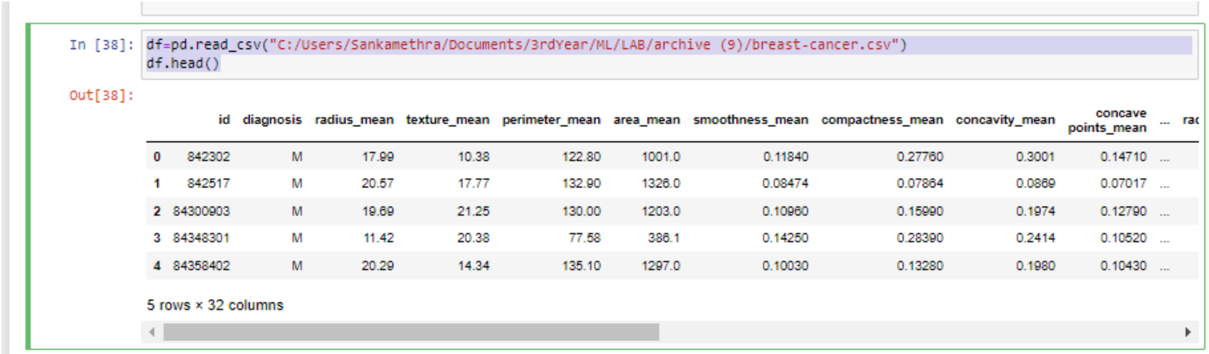


```
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import f1_score
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split,cross_val_score
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

```
import numpy as np
```

Loading dataset

df=pd.read_csv("C:/Users/Sankamethra/Documents/3rdYear/ML/LAB/archive (9)/breast-cancer.csv")
df.head()



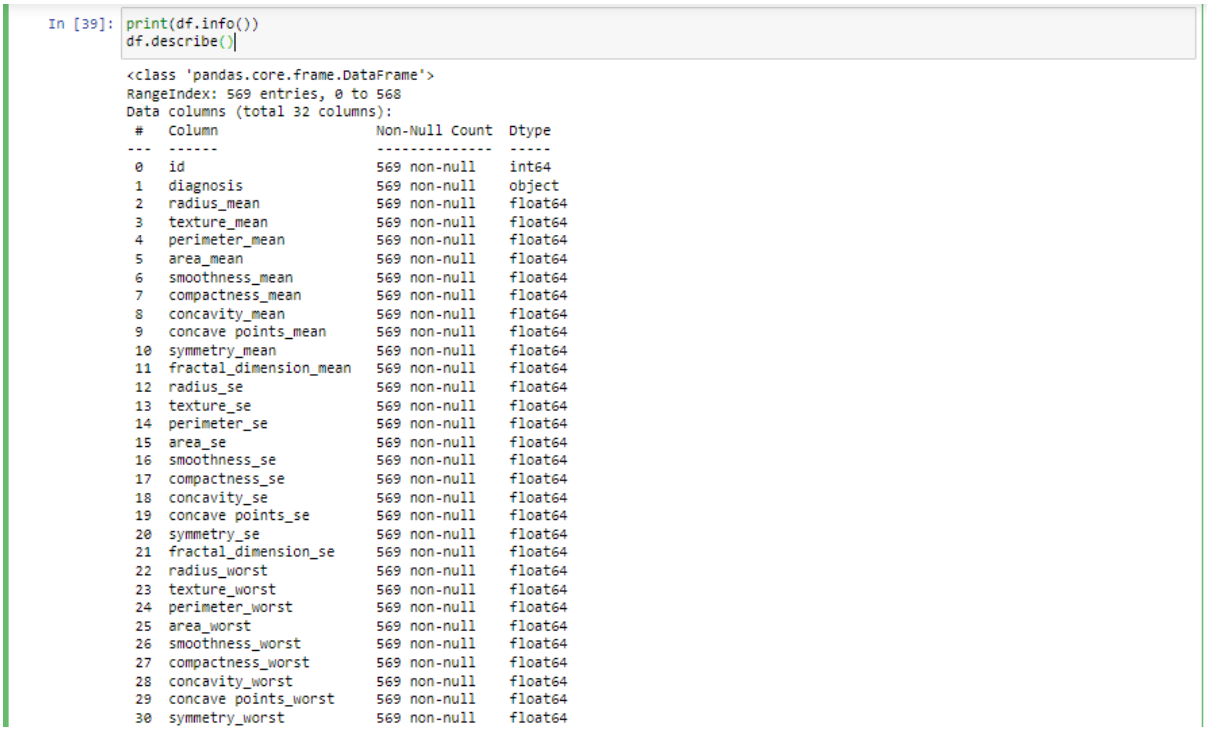
In [38]:
```
df=pd.read_csv("C:/Users/Sankamethra/Documents/3rdYear/ML/LAB/archive (9)/breast-cancer.csv")
df.head()
```

Out[38]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	rac
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	

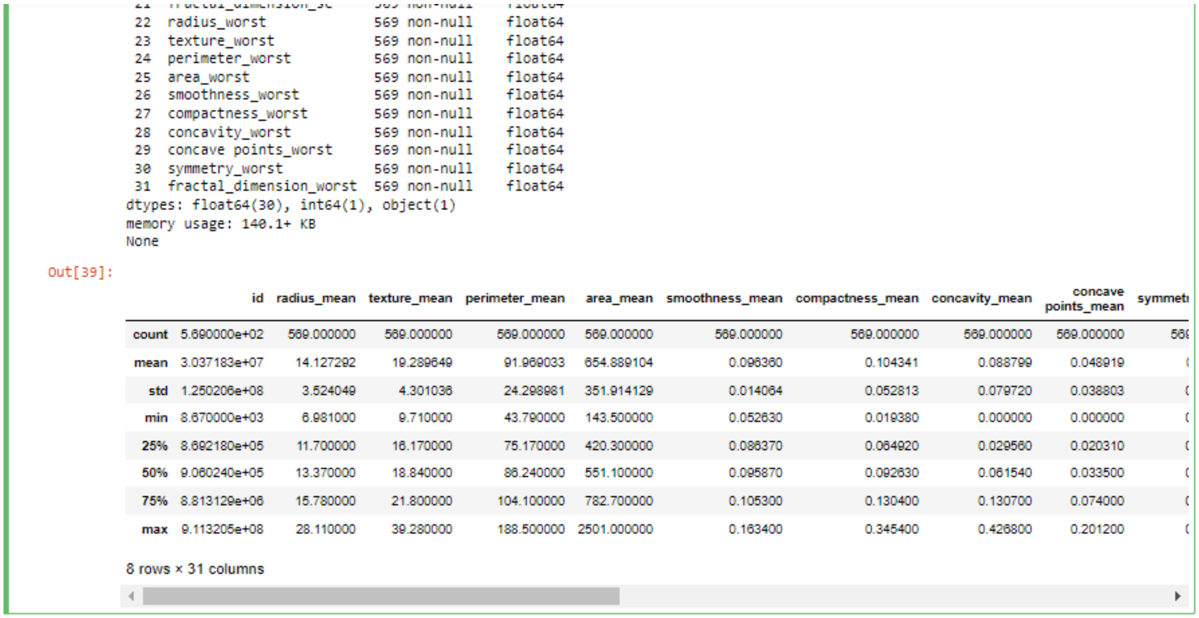
5 rows × 32 columns

print(df.info())
df.describe()



In [39]:
```
print(df.info())
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       569 non-null    int64  
 1   diagnosis                569 non-null    object 
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
 16  smoothness_se            569 non-null    float64
 17  compactness_se           569 non-null    float64
 18  concavity_se             569 non-null    float64
 19  concave points_se        569 non-null    float64
 20  symmetry_se              569 non-null    float64
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
```



```
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
 31  fractal_dimension_worst  569 non-null    float64
dtypes: float64(30), int64(1), object(1)
memory usage: 140.1+ KB
None
```

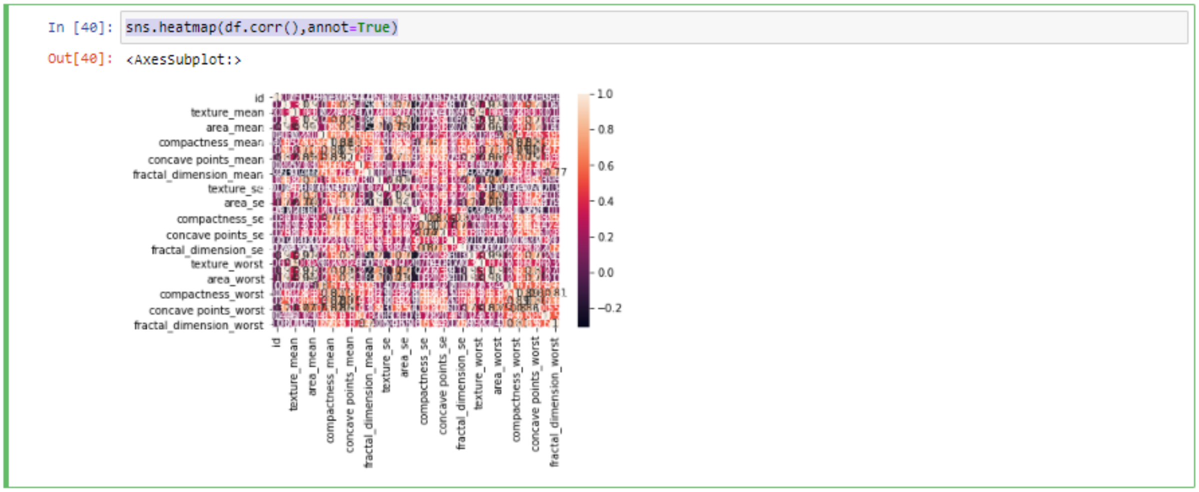
Out[39]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetı
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	56
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	(
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	(
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	(
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	(
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	(
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	(
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	(

8 rows × 31 columns

Correlation between columns

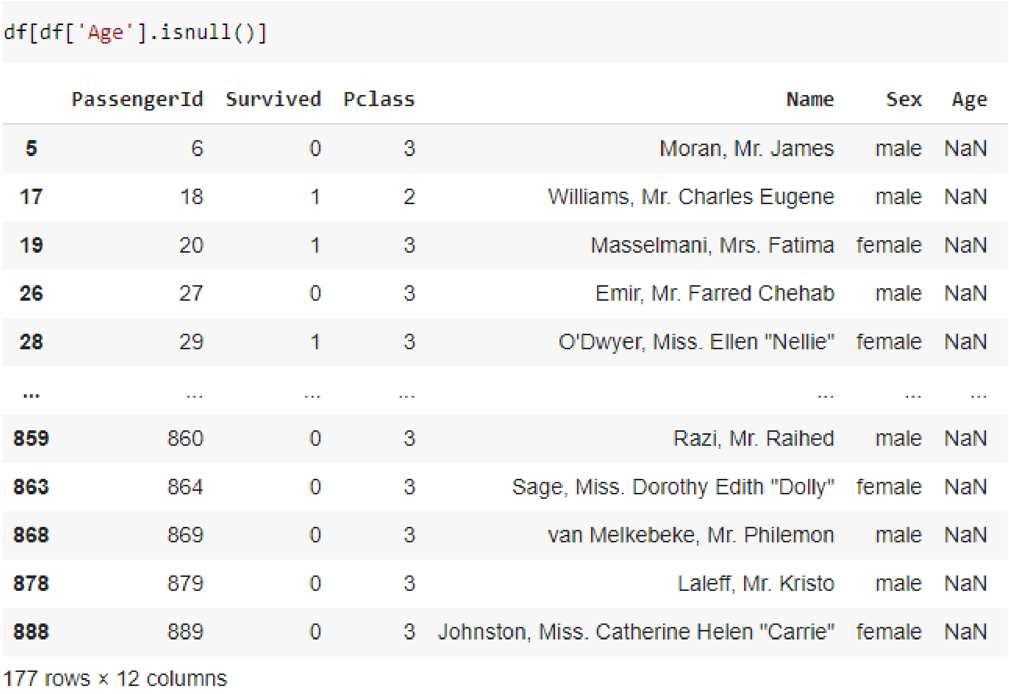
sns.heatmap(df.corr(),annot=True)



```
In [40]: sns.heatmap(df.corr(),annot=True)
Out[40]: <AxesSubplot:>
```

Missing rows for the values in age.

df[df['Age'].isnull()]

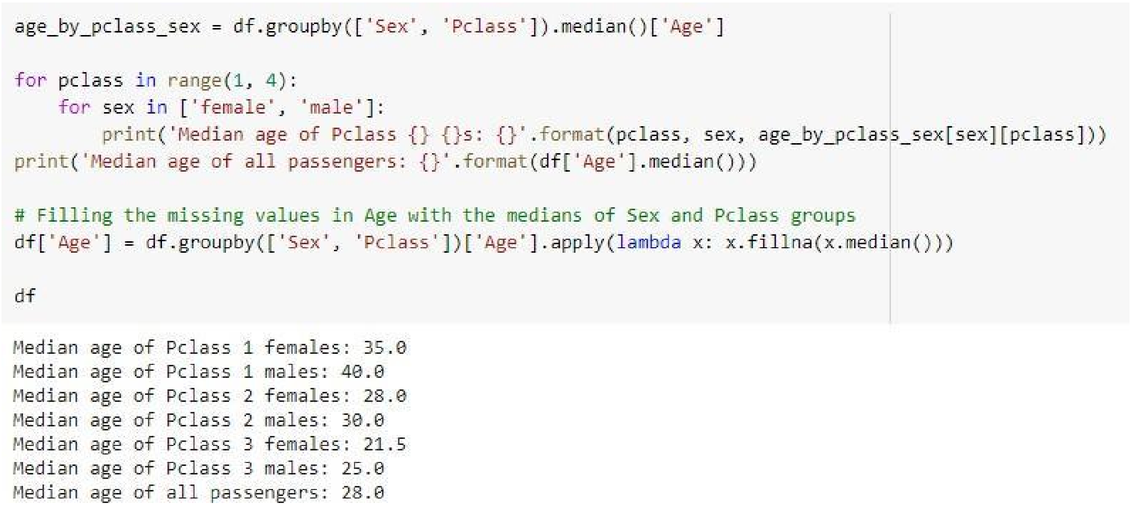


```
df[df['Age'].isnull()]
```

	PassengerId	Survived	Pclass	Name	Sex	Age
5	6	0	3	Moran, Mr. James	male	NaN
17	18	1	2	Williams, Mr. Charles Eugene	male	NaN
19	20	1	3	Masselmani, Mrs. Fatima	female	NaN
26	27	0	3	Emir, Mr. Farred Chehab	male	NaN
28	29	1	3	O'Dwyer, Miss. Ellen "Nellie"	female	NaN
...	...	...	...	...	...	...
859	860	0	3	Razi, Mr. Raihed	male	NaN
863	864	0	3	Sage, Miss. Dorothy Edith "Dolly"	female	NaN
868	869	0	3	van Melkebeke, Mr. Philemon	male	NaN
878	879	0	3	Laleff, Mr. Kristo	male	NaN
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN

177 rows × 12 columns

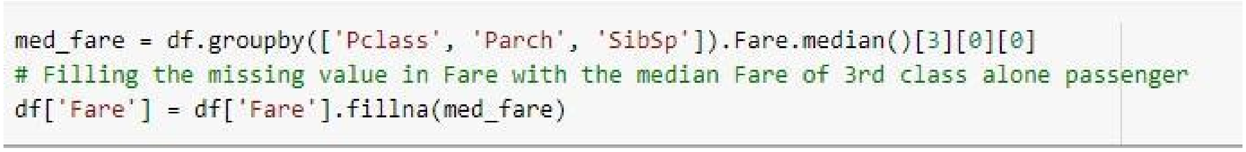
Filling the missing values in age with median of corresponding data values in pclass and sex.



```
age_by_pclass_sex = df.groupby(['Sex', 'Pclass']).median()['Age']

for pclass in range(1, 4):
    for sex in ['female', 'male']:
        print('Median age of Pclass {} {}s: {}'.format(pclass, sex, age_by_pclass_sex[sex][pclass]))
print('Median age of all passengers: {}'.format(df['Age'].median()))

# Filling the missing values in Age with the medians of Sex and Pclass groups
df['Age'] = df.groupby(['Sex', 'Pclass'])['Age'].apply(lambda x: x.fillna(x.median()))

df
```

```
Median age of Pclass 1 females: 35.0
Median age of Pclass 1 males: 40.0
Median age of Pclass 2 females: 28.0
Median age of Pclass 2 males: 30.0
Median age of Pclass 3 females: 21.5
Median age of Pclass 3 males: 25.0
Median age of all passengers: 28.0
```

Filling missing values in fare with median of datas corresponding to pclass Sibsp and Parch columns.



```
med_fare = df.groupby(['Pclass', 'Parch', 'SibSp']).Fare.median()[3][0][0]
# Filling the missing value in Fare with the median Fare of 3rd class alone passenger
df['Fare'] = df['Fare'].fillna(med_fare)
```