Author: kkitonga
Subject:PCA for visualization
Date:01/11/2022

1.Importation of necessary libraries

In [1]:
```
import pandas as pd                                                    #dealing with dataframes
import numpy as np                                                     #mathematical operations
from sklearn.decomposition import PCA                                  #principal component annalysis
from sklearn.preprocessing import StandardScaler                       #scaling data
import matplotlib.pyplot as plt                                        #data visualization
import seaborn as sns                                                  #data visualization
import warnings
warnings.filterwarnings("ignore")                                      #suppress warnings
```

2.loading data

In [2]:
```
data = pd.read_csv("C:/Users/Karengi/Desktop/MIT/diabetes.csv")
```

3.Making a copy of the data

In [3]:
```
data1 =data.copy()
```

4.Read the data

4.1 First five observations

In [4]:
```
data.head()
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	79	33.6	0.627	50	1
1	1	85	66	29	79	26.6	0.351	31	0
2	8	183	64	20	79	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

4.2 Last five observations

In [5]:
```
data.tail()
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	79	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	20	79	30.1	0.349	47	1
767	1	93	70	31	79	30.4	0.315	23	0

4.3.Data information

In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

5 Exploratory data annalysis

5.Exploratory data annalysis

In [10]:
```
data.describe().T
```

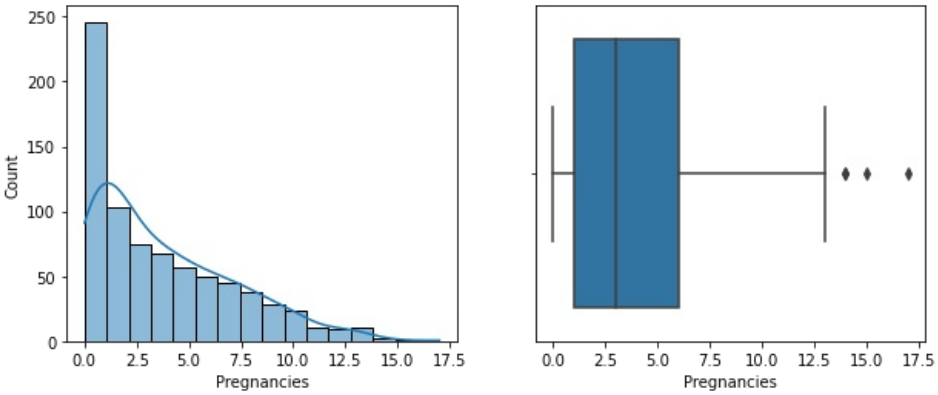
Out[10]:

	count	mean	std	min	25%	50%	75%	max
Pregnancies	768.0	3.845052	3.369578	0.000	1.00000	3.0000	6.00000	17.00
Glucose	768.0	121.675781	30.436252	44.000	99.75000	117.0000	140.25000	199.00
BloodPressure	768.0	72.250000	12.117203	24.000	64.00000	72.0000	80.00000	122.00
SkinThickness	768.0	26.447917	9.733872	7.000	20.00000	23.0000	32.00000	99.00
Insulin	768.0	118.270833	93.243829	14.000	79.00000	79.0000	127.25000	846.00
BMI	768.0	32.450805	6.875374	18.200	27.50000	32.0000	36.60000	67.10
DiabetesPedigreeFunction	768.0	0.471876	0.331329	0.078	0.24375	0.3725	0.62625	2.42
Age	768.0	33.240885	11.760232	21.000	24.00000	29.0000	41.00000	81.00
Outcome	768.0	0.348958	0.476951	0.000	0.00000	0.0000	1.00000	1.00

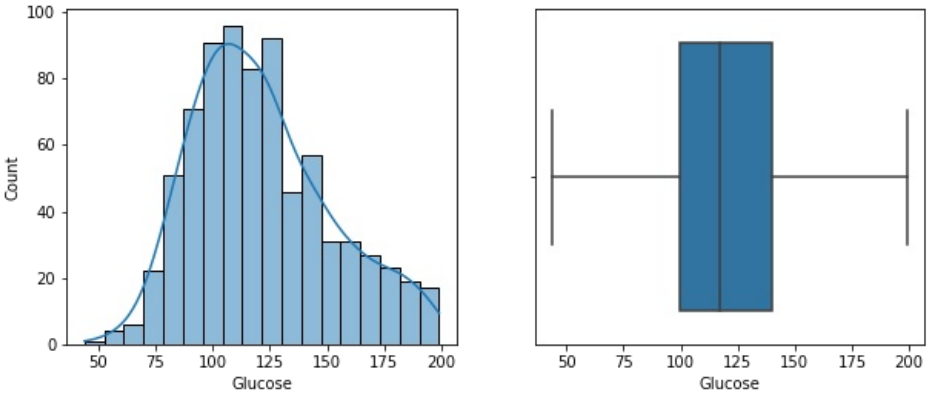
6.Univariate annalysis

In [54]:
```
for col in data.columns[0:8]:
    plt.figure(figsize=(10,4))
    plt.subplot(1,2,1)
    print(col)
    sns.histplot(data[col],kde=True)
    plt.subplot(1,2,2)
    sns.boxplot(data[col])
    plt.show()
```

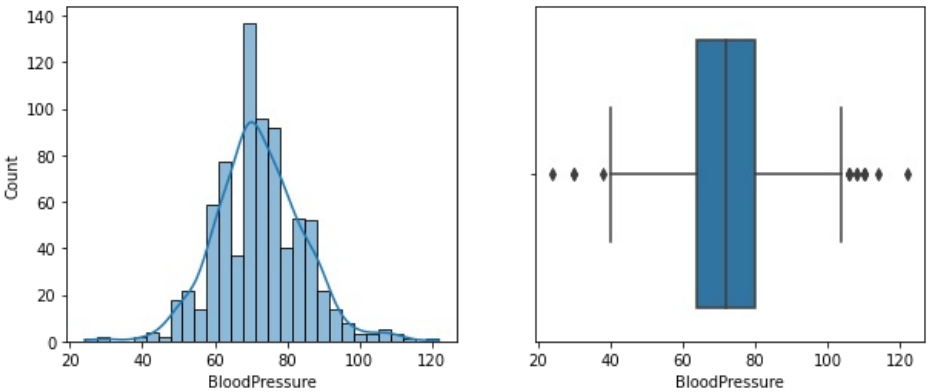
Pregnancies



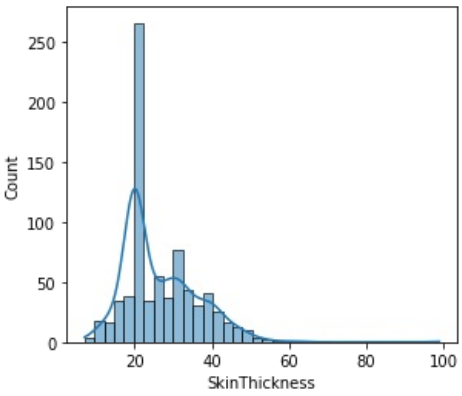
Glucose

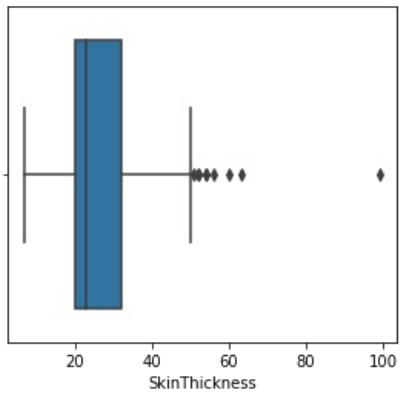


BloodPressure

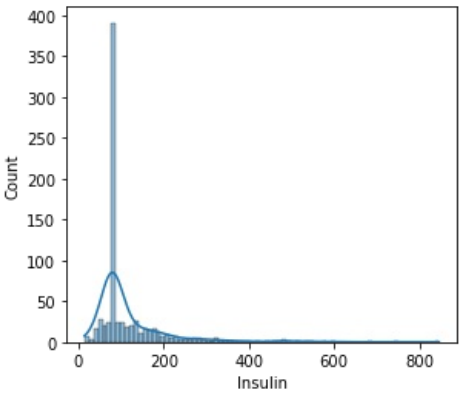


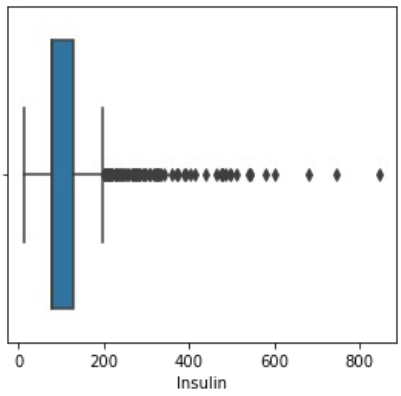
SkinThickness



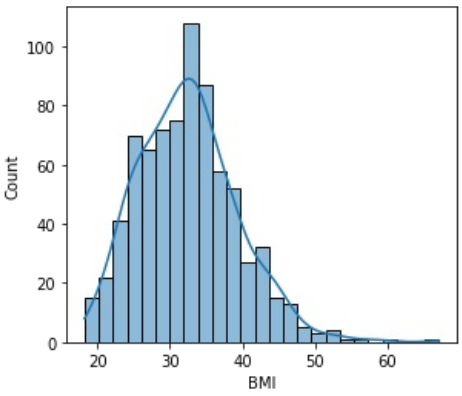


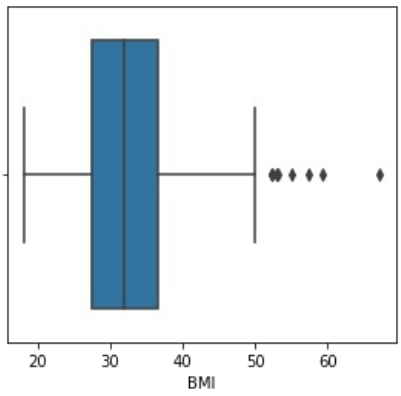
Insulin



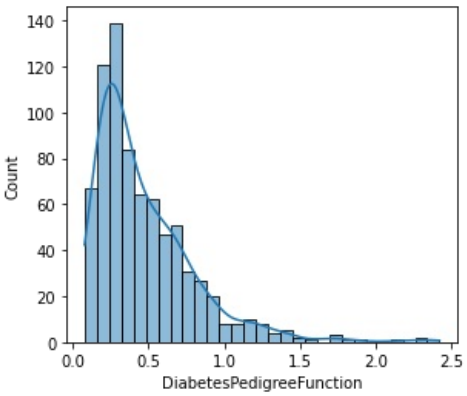


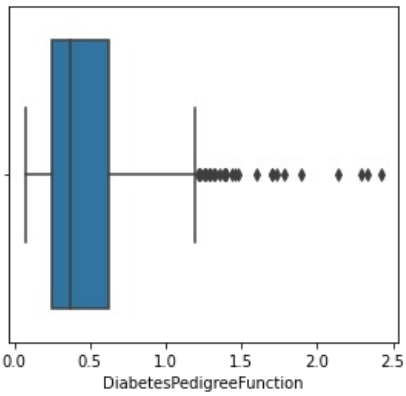
BMI



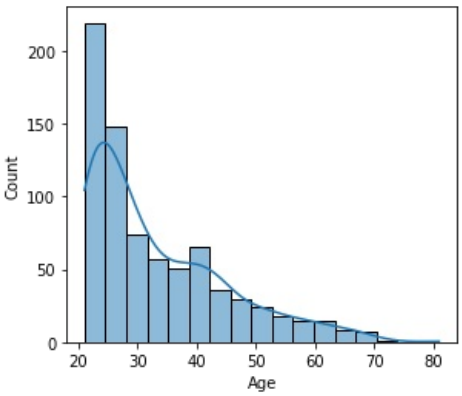


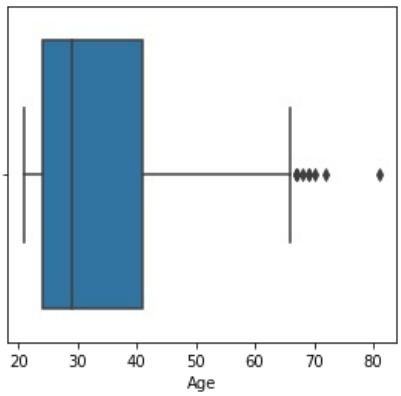
DiabetesPedigreeFunction





Age





7.Dropping outcome variable in 'data' dataframe

In [43]:
```
data = data.drop(['Outcome'],axis=1)
```

In [44]:
```
data.head()                                                    #confirmation 'Outcome' was drop
```

Out[44]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	6	148	72	35	79	33.6	0.627	50
1	1	85	66	29	79	26.6	0.351	31
2	8	183	64	20	79	23.3	0.672	32
3	1	89	66	23	94	28.1	0.167	21
4	0	137	40	35	168	43.1	2.288	33

Perfoming Principal Component Annalysis

1.Scaling data

In [45]:
```
scaler = StandardScaler()
data_scaled = pd.DataFrame(scaler.fit_transform(data),columns=data.columns)  #scaling data and converting it to
```

In [46]:
```
data_scaled.head()
```

Out[46]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	0.639947	0.865461	-0.020645	0.879163	-0.421437	0.167255	0.468492	1.425995
1	-0.844885	-1.205788	-0.516132	0.262357	-0.421437	-0.851535	-0.365061	-0.190672
2	1.233880	2.016154	-0.681294	-0.662852	-0.421437	-1.331821	0.604397	-0.105584
3	-0.844885	-1.074281	-0.516132	-0.354449	-0.260464	-0.633222	-0.920763	-1.041549
4	-1.141852	0.503814	-2.663240	0.879163	0.533672	1.549899	5.484909	-0.020496

2.Defining number of principal components

In [47]:
```
pca = PCA(n_components=data_scaled.shape[1])                   #equivalent to no of columns
```

3.Performing PCA

In [48]:
```
data_pca = pd.DataFrame(pca.fit_transform(data_scaled))        #PCA
```

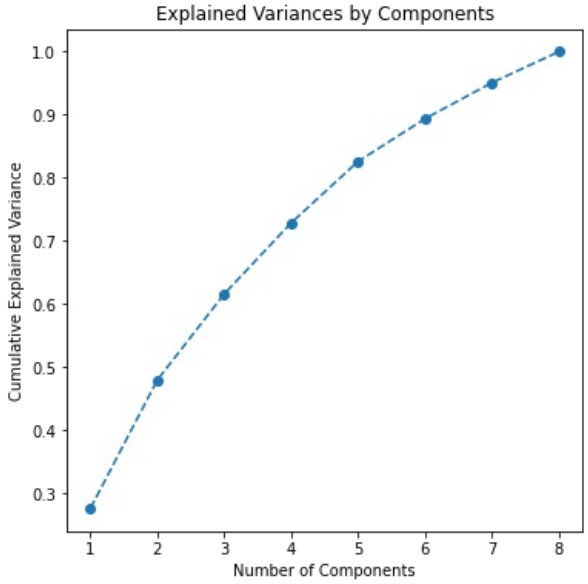
4. Explained variance by each component

In [49]:
```
exp_var = pca.explained_variance_ratio_
```

5:Plotting cumulative variance

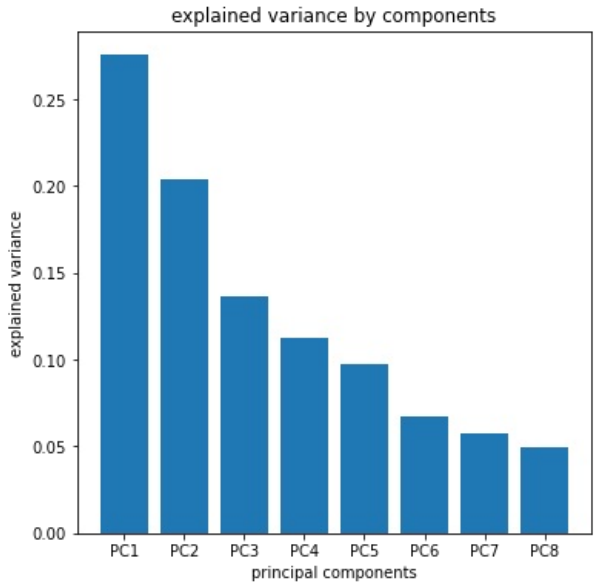
In [50]:
```
plt.figure(figsize = (6,6))
plt.plot(range(1,9), exp_var.cumsum(), marker = 'o', linestyle = '--')
plt.title("Explained Variances by Components")
plt.xlabel("Number of Components")
plt.ylabel("Cumulative Explained Variance")
```

Out[50]:
```
Text(0, 0.5, 'Cumulative Explained Variance')
```



6:Plotting explained variance by each component

In [51]:
```
labels=['PC' + str(x) for x in range (1,9)]
plt.figure(figsize=(6,6))
plt.bar(x=range(1,9),height=exp_var,tick_label=labels)
plt.xlabel('principal components')
plt.ylabel('explained variance')
plt.title('explained variance by components')
plt.show()
```



7.PCA loadings

In [52]:
```
#Loadings for 5 columns that explain more than 80 % of variation
pc_comps = ['PC1','PC2','PC3','PC4','PC5']
data_pca = pd.DataFrame(np.round(pca.components_[:5,:],2),index=pc_comps,columns=data_scaled.columns)
data_pca.T
```

Out[52]:

	PC1	PC2	PC3	PC4	PC5
Pregnancies	0.27	-0.54	0.00	0.16	-0.46
Glucose	0.44	0.01	0.43	-0.29	0.25
BloodPressure	0.37	-0.24	-0.36	-0.13	0.66
SkinThickness	0.39	0.36	-0.35	0.05	-0.43
Insulin	0.33	0.28	0.53	-0.31	-0.18
BMI	0.43	0.31	-0.43	-0.01	-0.04
DiabetesPedigreeFunction	0.20	0.24	0.28	0.87	0.25
Age	0.35	-0.53	0.10	0.13	-0.11

8 :Visualization: first two principle components (Target variable:diabetic status)

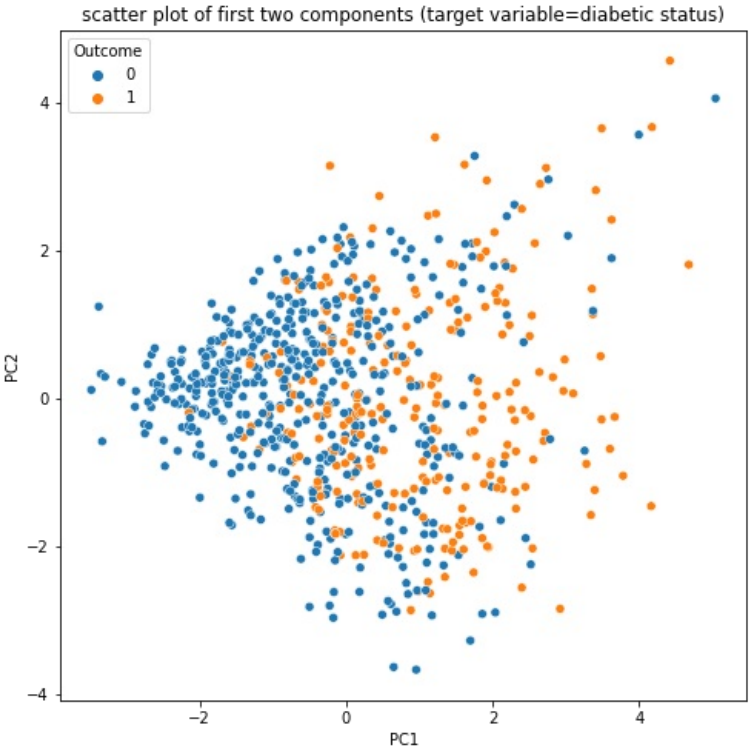
In [55]:
```
pca2 = PCA(n_components=2)
```

In [56]:
```
data_pca2 = pca2.fit_transform(data_scaled)
```

In [57]:
```
data_pca2.shape
```

Out[57]:
```
(768, 2)
```

In [59]:
```
####Conclusion
plt.figure(figsize=(8,8))
sns.scatterplot(data_pca2[:,0],data_pca2[:,1],hue=data1['Outcome'])
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('scatter plot of first two components (target variable=diabetic status)')
plt.show()
```



9.Conclusion

Not quite a clear cut distinction,which is understandable because PC1 AND PC2 combined only capture
less than 50 % of the variation.

In [ ]:
```
```