In [2]:

```
import pandas as pd
columns=['preg','Glucose','BloodP','Skin','Insulin','BMI','Func','Age',
'Class']
df=pd.read_csv("E:\\java\\indians-diabetes.data.csv",names=columns)
```

In [3]:

```
print(df.head(5))
```

```
   preg  Glucose  BloodP  Skin  Insulin   BMI   Func  Age  Class
0     6      148      72    35        0  33.6  0.627   50      1
1     1       85      66    29        0  26.6  0.351   31      0
2     8      183      64     0        0  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
```

In [4]:

```
print(df.isna().sum())
```

```
preg       0
Glucose    0
BloodP     0
Skin       0
Insulin    0
BMI        0
Func       0
Age        0
Class      0
dtype: int64
```

In [5]:

```
print(df.shape)
```

```
(768, 9)
```

In [6]:

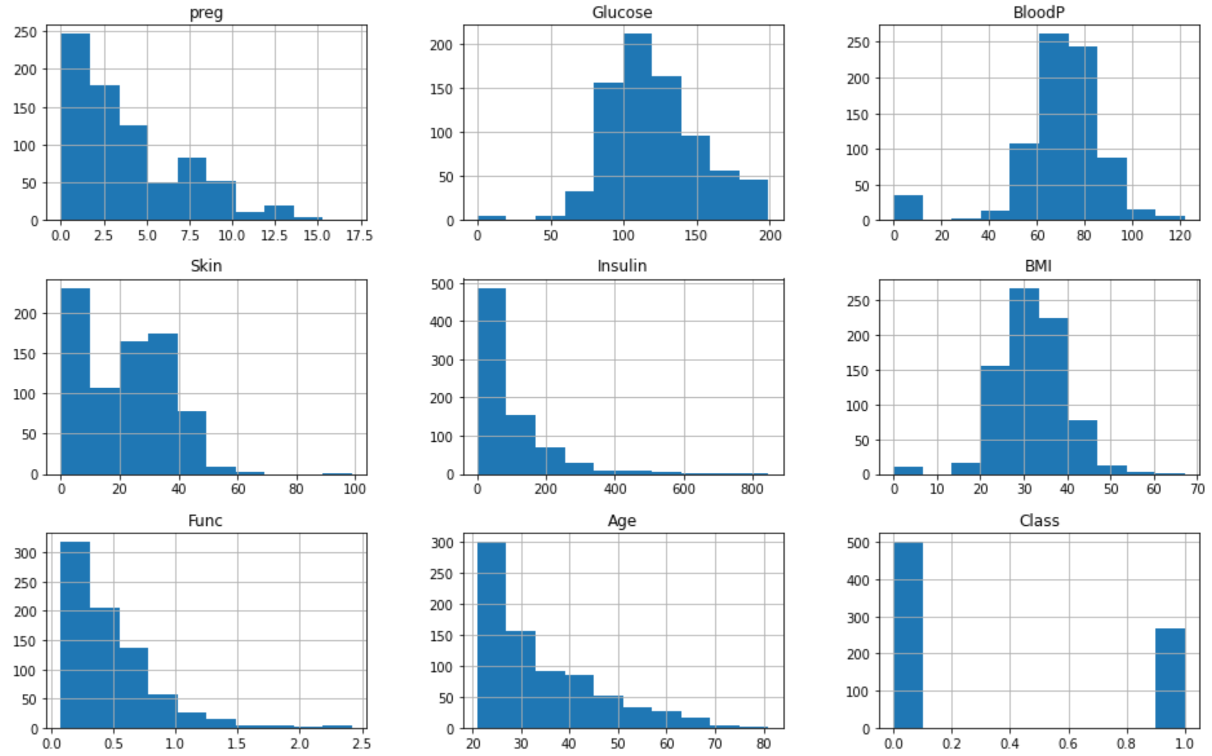
```
print(df.describe())
```

```
             preg     Glucose      BloodP        Skin     Insulin
   BMI  \
count  768.000000  768.000000  768.000000  768.000000  768.000000  768.
000000
mean     3.845052  120.894531   69.105469   20.536458   79.799479   31.
992578
std      3.369578   31.972618   19.355807   15.952218  115.244002    7.
884160
min      0.000000    0.000000    0.000000    0.000000    0.000000    0.
000000
25%      1.000000   99.000000   62.000000    0.000000    0.000000   27.
300000
50%      3.000000  117.000000   72.000000   23.000000   30.500000   32.
000000
75%      6.000000  140.250000   80.000000   32.000000  127.250000   36.
600000
max     17.000000  199.000000  122.000000   99.000000  846.000000   67.
100000

             Func         Age       Class
count  768.000000  768.000000  768.000000
mean     0.471876   33.240885    0.348958
std      0.331329   11.760232    0.476951
min      0.078000   21.000000    0.000000
25%      0.243750   24.000000    0.000000
50%      0.372500   29.000000    0.000000
75%      0.626250   41.000000    1.000000
max      2.420000   81.000000    1.000000
```

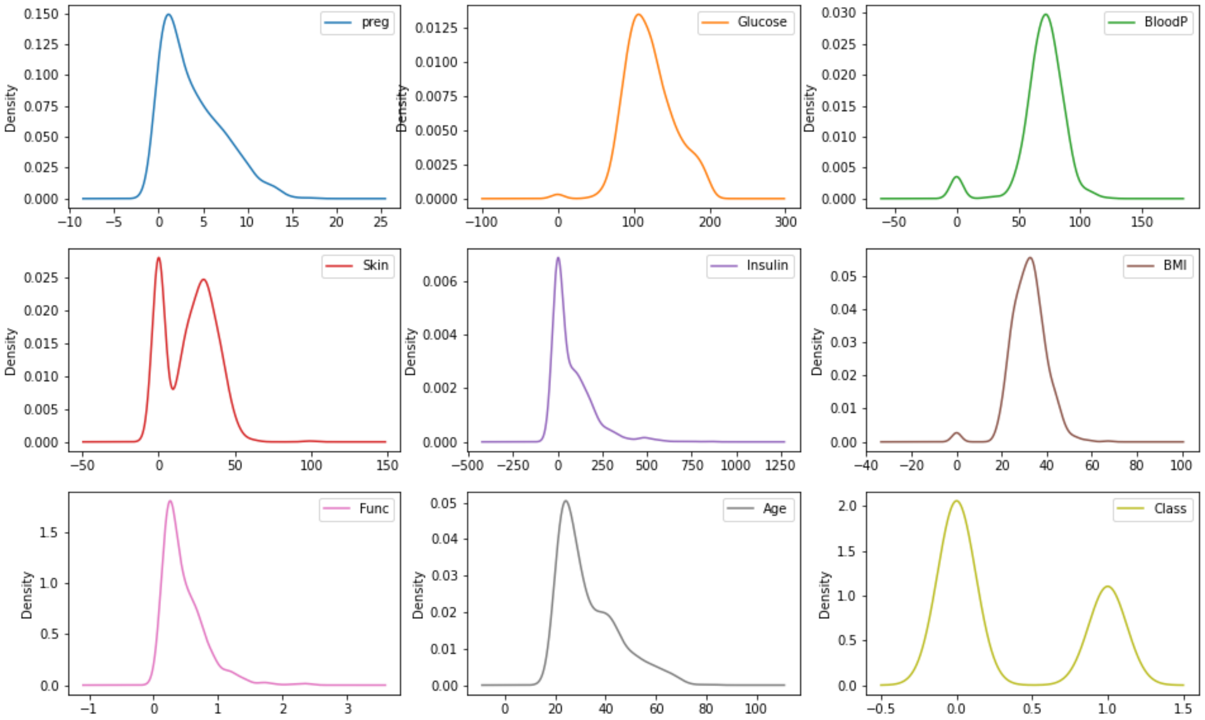
In [7]:

```
import matplotlib.pyplot as plt
df.hist(figsize=(16,10))
plt.show()
```



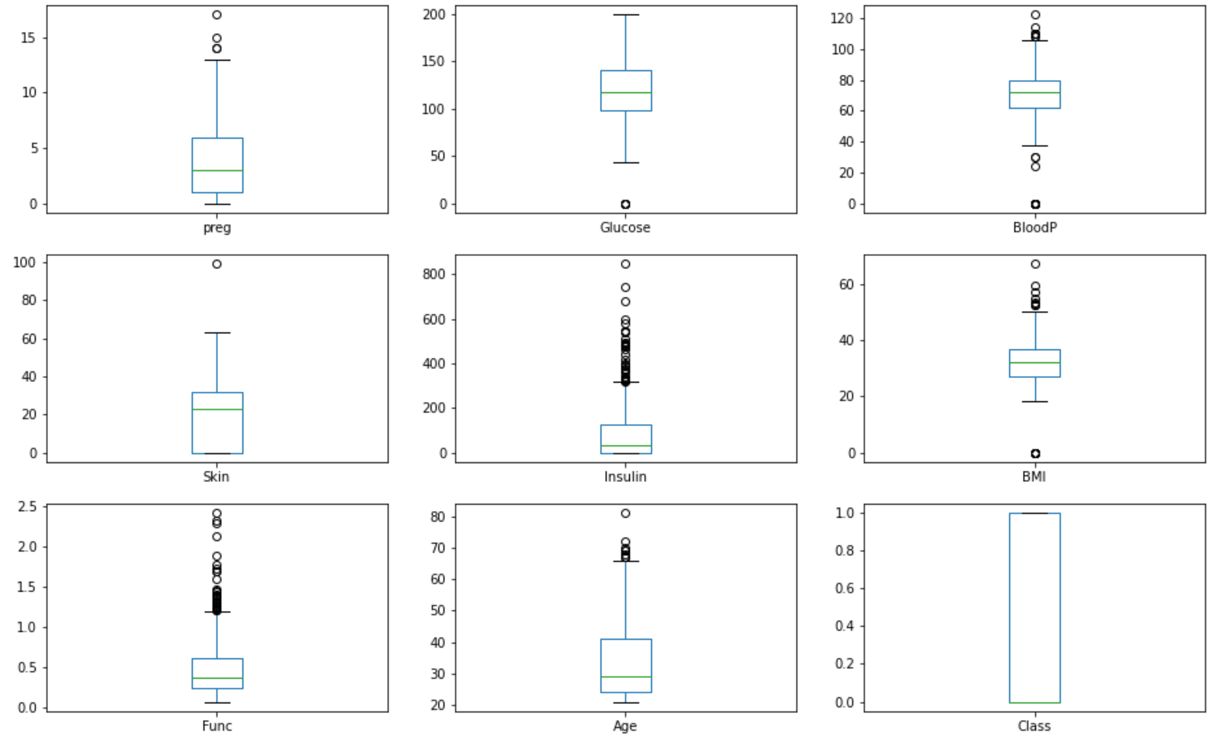
In [8]:

```
df.plot(kind='density',subplots=True,layout=(3,3),sharex=False,figsize=(16,10))
plt.show()
```



In [9]:

```
df.plot(kind='box',layout=(3,3),subplots=True,sharex=False,figsize=(16,
10))
plt.show()
```



In [10]:
```
correlation=df.corr()
```

In [11]:
```
correlation
```

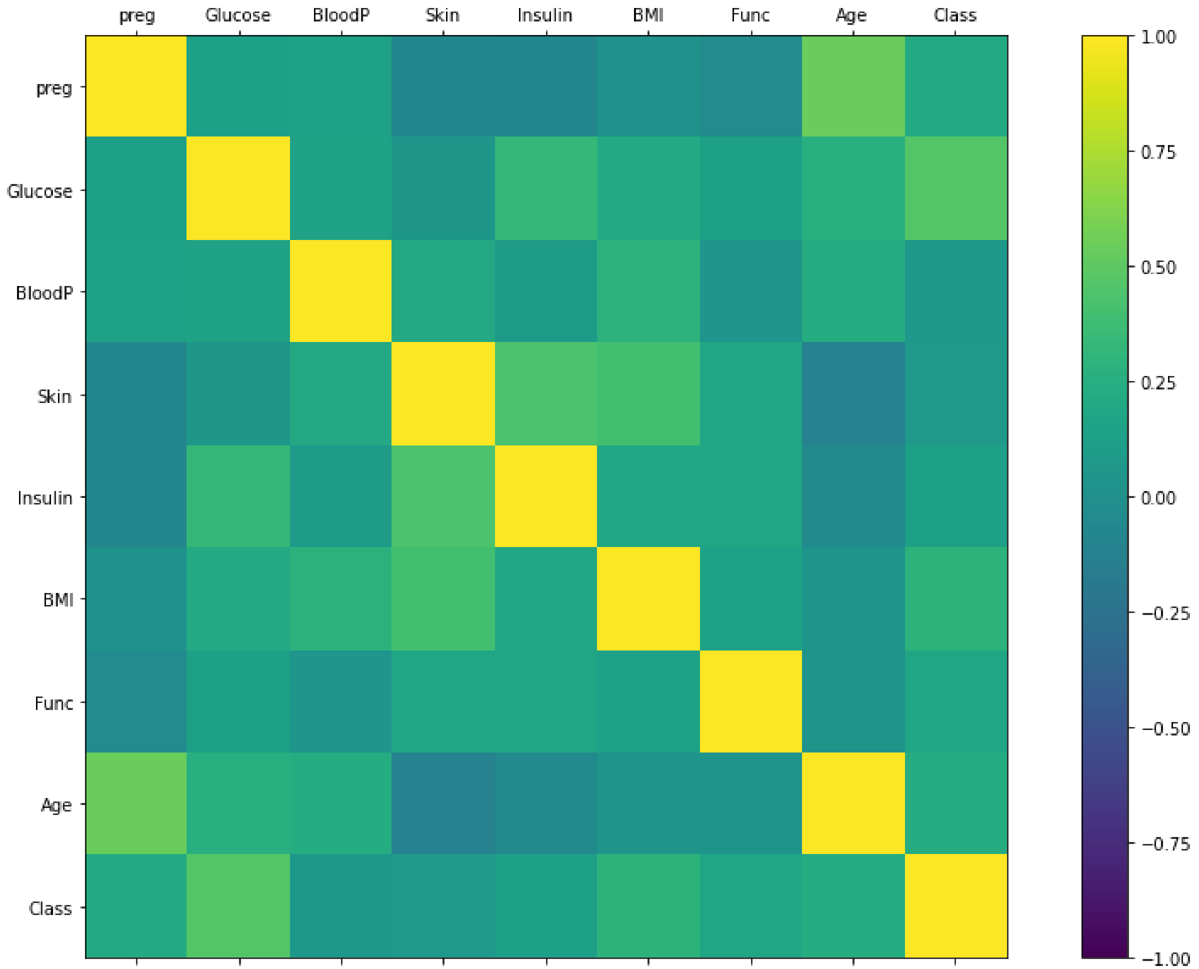
Out[11]:

	preg	Glucose	BloodP	Skin	Insulin	BMI	Func	Age	
preg	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	0.2
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.4
BloodP	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.0
Skin	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	0.0
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.042163	0.1

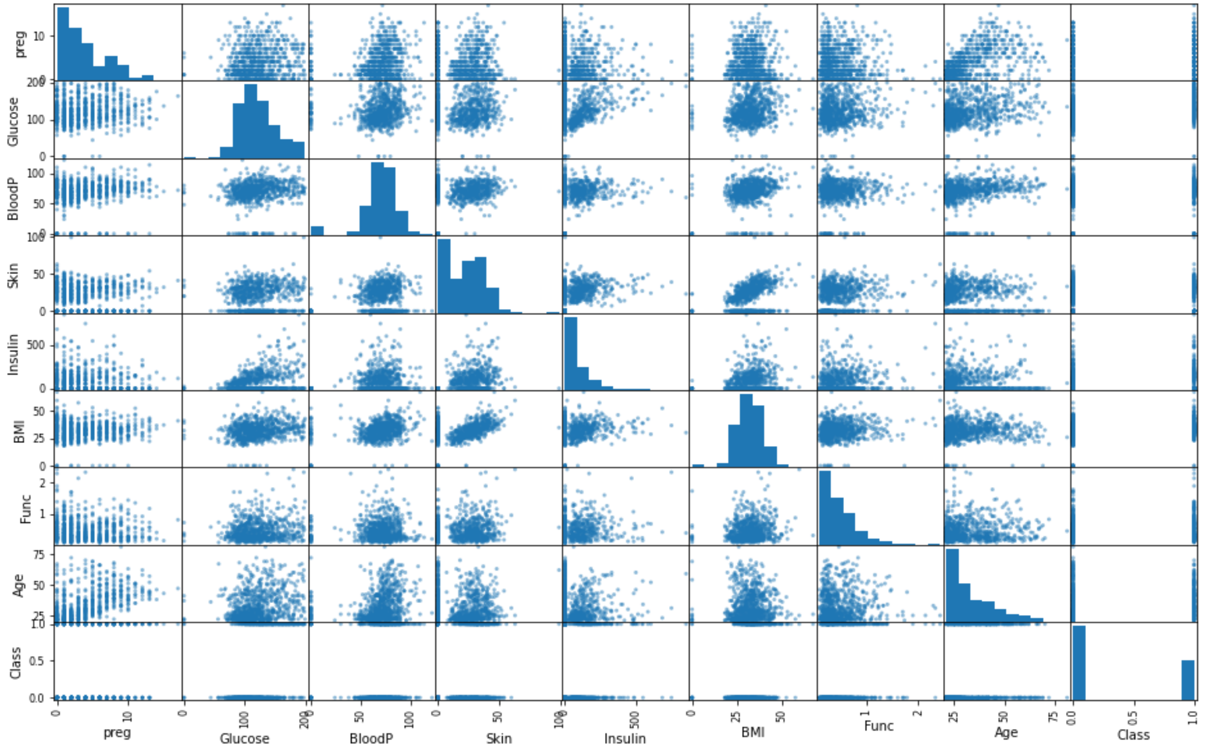
	preg	Glucose	BloodP	Skin	Insulin	BMI	Func	Age	
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.2
Func	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.1
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	0.2
Class	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	1.0

In [12]:

```
import numpy as np
fig=plt.figure(figsize=(16,10))
ax=fig.add_subplot(111)
cax=ax.matshow(correlation,vmax=1,vmin=-1)
fig.colorbar(cax)
ticks=np.arange(len(columns))
ax.set_xticks(ticks)
ax.set_xticklabels(columns)
ax.set_yticks(ticks)
ax.set_yticklabels(columns)
plt.show()
```



In [13]:
```
from pandas.plotting import scatter_matrix
scatter_matrix(df,figsize=(16,10))
plt.show()
```



In [14]:
```
x=df.iloc[:,:-1].values
y=df.iloc[:,-1:].values
```

In [15]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test = train_test_split(x,y,test_size=0.3,rand
om_state=7)
```

In [16]:
```
import warnings
warnings.filterwarnings(action='ignore')
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
model.fit(X_train,Y_train)
Y_pred=model.predict(X_test)
```

In [17]:

```
from sklearn.metrics import accuracy_score
print("Accuracy :",accuracy_score(Y_test,Y_pred))
```

```
Accuracy : 0.7489177489177489
```

In [18]:

```
from sklearn.metrics import confusion_matrix
conf_matrix=confusion_matrix(Y_test,Y_pred)
print(conf_matrix)
```

```
[[127  20]
 [ 38  46]]
```

In [19]:

```
accuracy = (conf_matrix[0][0]+conf_matrix[1][1])/np.sum(conf_matrix)
```

In [20]:

```
print(accuracy)
```

```
0.7489177489177489
```

In [21]:

```
from sklearn.metrics import classification_report
report = classification_report(Y_test,Y_pred)
print(report)
```

```
              precision    recall  f1-score   support

           0       0.77      0.86      0.81       147
           1       0.70      0.55      0.61        84

    accuracy                           0.75       231
   macro avg       0.73      0.71      0.71       231
weighted avg       0.74      0.75      0.74       231
```

In [22]:

```
x_input=[[4,110,92,0,0,37.6,0.191,30]]
result=model.predict(x_input)
print(result)
```

```
[0]
```

In [ ]: