In [1]:
```
# setting working directory

#importing all libraries for modelling
import random
import os
os.chdir("D:\Anamika\F\Downloads")
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
%matplotlib inline
from xgboost import plot_tree
from matplotlib.pylab import rcParams
from sklearn.metrics import roc_curve ,auc,recall_score,precision_score
from sklearn.metrics import accuracy_score, make_scorer
from sklearn.model_selection import KFold, GridSearchCV
from numpy import sqrt ,argmax
import seaborn as sns
from matplotlib import pyplot
import matplotlib.pyplot as plt
import matplotlib.pylab as plt
%matplotlib inline
rcParams['figure.figsize']= [10,10]

#set seed to have consistent values for all iterations
random.seed(10)
```

In [2]:
```
#loading the dataset
df = pd.read_csv(r"D:\Anamika\F\Semester 7\MDD\parkinsons.csv")
print(df)
```

	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)
0	phon_R01_S01_1	119.992	157.302	74.997	0.00784
1	phon_R01_S01_2	122.400	148.650	113.819	0.00968
2	phon_R01_S01_3	116.682	131.111	111.555	0.01050
3	phon_R01_S01_4	116.676	137.871	111.366	0.00997
4	phon_R01_S01_5	116.014	141.781	110.655	0.01284
5	phon_R01_S01_6	120.552	131.162	113.787	0.00968
6	phon_R01_S02_1	120.267	137.244	114.820	0.00333
7	phon_R01_S02_2	107.332	113.840	104.315	0.00290
8	phon_R01_S02_3	95.730	132.068	91.754	0.00551
9	phon_R01_S02_4	95.056	120.103	91.226	0.00532
10	phon_R01_S02_5	88.333	112.240	84.072	0.00505
11	phon_R01_S02_6	91.904	115.871	86.292	0.00540
12	phon_R01_S04_1	136.926	159.866	131.276	0.00293
13	phon_R01_S04_2	139.173	179.139	76.556	0.00390
14	phon_R01_S04_3	152.845	163.305	75.836	0.00294
15	phon_R01_S04_4	142.167	217.455	83.159	0.00369
16	phon_R01_S04_5	144.188	349.259	82.764	0.00544
17	phon_R01_S04_6	168.778	232.181	75.603	0.00718
18	phon_R01_S05_1	153.046	175.829	68.623	0.00742
19	phon_R01_S05_2	156.405	189.398	142.822	0.00768
20	phon_R01_S05_3	153.848	165.738	65.782	0.00840
21	phon_R01_S05_4	153.880	172.860	78.128	0.00480
22	phon_R01_S05_5	167.930	193.221	79.068	0.00442
23	phon_R01_S05_6	173.917	192.735	86.180	0.00476
24	phon_R01_S06_1	163.656	200.841	76.779	0.00742
25	phon_R01_S06_2	104.400	206.002	77.968	0.00633
26	phon_R01_S06_3	171.041	208.313	75.501	0.00455
27	phon_R01_S06_4	146.845	208.701	81.737	0.00496
28	phon_R01_S06_5	155.358	227.383	80.055	0.00310
29	phon_R01_S06_6	162.568	198.346	77.630	0.00502
..	...	...	...	...	...
165	phon_R01_S42_1	236.200	244.663	102.137	0.00277
166	phon_R01_S42_2	237.323	243.709	229.256	0.00303
167	phon_R01_S42_3	260.105	264.919	237.303	0.00339
168	phon_R01_S42_4	197.569	217.627	90.794	0.00803
169	phon_R01_S42_5	240.301	245.135	219.783	0.00517
170	phon_R01_S42_6	244.990	272.210	239.170	0.00451
171	phon_R01_S43_1	112.547	133.374	105.715	0.00355
172	phon_R01_S43_2	110.739	113.597	100.139	0.00356
173	phon_R01_S43_3	113.715	116.443	96.913	0.00349
174	phon_R01_S43_4	117.004	144.466	99.923	0.00353
175	phon_R01_S43_5	115.380	123.109	108.634	0.00332
176	phon_R01_S43_6	116.388	129.038	108.970	0.00346
177	phon_R01_S44_1	151.737	190.204	129.859	0.00314
178	phon_R01_S44_2	148.790	158.359	138.990	0.00309
179	phon_R01_S44_3	148.143	155.982	135.041	0.00392
180	phon_R01_S44_4	150.440	163.441	144.736	0.00396
181	phon_R01_S44_5	148.462	161.078	141.998	0.00397
182	phon_R01_S44_6	149.818	163.417	144.786	0.00336
183	phon_R01_S49_1	117.226	123.925	106.656	0.00417
184	phon_R01_S49_2	116.848	217.552	99.503	0.00531
185	phon_R01_S49_3	116.286	177.291	96.983	0.00314
186	phon_R01_S49_4	116.556	592.030	86.228	0.00496
187	phon_R01_S49_5	116.342	581.289	94.246	0.00267
188	phon_R01_S49_6	114.563	119.167	86.647	0.00327

189	phon_R01_S50_1	201.774	262.707	78.228	0.00694
190	phon_R01_S50_2	174.188	230.978	94.261	0.00459
191	phon_R01_S50_3	209.516	253.017	89.488	0.00564
192	phon_R01_S50_4	174.688	240.005	74.287	0.01360
193	phon_R01_S50_5	198.764	396.961	74.904	0.00740
194	phon_R01_S50_6	214.289	260.277	77.973	0.00567

	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	...	\
0	0.00007	0.00370	0.00554	0.01109	0.04374	...	
1	0.00008	0.00465	0.00696	0.01394	0.06134	...	
2	0.00009	0.00544	0.00781	0.01633	0.05233	...	
3	0.00009	0.00502	0.00698	0.01505	0.05492	...	
4	0.00011	0.00655	0.00908	0.01966	0.06425	...	
5	0.00008	0.00463	0.00750	0.01388	0.04701	...	
6	0.00003	0.00155	0.00202	0.00466	0.01608	...	
7	0.00003	0.00144	0.00182	0.00431	0.01567	...	
8	0.00006	0.00293	0.00332	0.00880	0.02093	...	
9	0.00006	0.00268	0.00332	0.00803	0.02838	...	
10	0.00006	0.00254	0.00330	0.00763	0.02143	...	
11	0.00006	0.00281	0.00336	0.00844	0.02752	...	
12	0.00002	0.00118	0.00153	0.00355	0.01259	...	
13	0.00003	0.00165	0.00208	0.00496	0.01642	...	
14	0.00002	0.00121	0.00149	0.00364	0.01828	...	
15	0.00003	0.00157	0.00203	0.00471	0.01503	...	
16	0.00004	0.00211	0.00292	0.00632	0.02047	...	
17	0.00004	0.00284	0.00387	0.00853	0.03327	...	
18	0.00005	0.00364	0.00432	0.01092	0.05517	...	
19	0.00005	0.00372	0.00399	0.01116	0.03995	...	
20	0.00005	0.00428	0.00450	0.01285	0.03810	...	
21	0.00003	0.00232	0.00267	0.00696	0.04137	...	
22	0.00003	0.00220	0.00247	0.00661	0.04351	...	
23	0.00003	0.00221	0.00258	0.00663	0.04192	...	
24	0.00005	0.00380	0.00390	0.01140	0.01659	...	
25	0.00006	0.00316	0.00375	0.00948	0.03767	...	
26	0.00003	0.00250	0.00234	0.00750	0.01966	...	
27	0.00003	0.00250	0.00275	0.00749	0.01919	...	
28	0.00002	0.00159	0.00176	0.00476	0.01718	...	
29	0.00003	0.00280	0.00253	0.00841	0.01791	...	
..	...	...	...	...	...	...	
165	0.00001	0.00154	0.00153	0.00462	0.02448	...	
166	0.00001	0.00173	0.00159	0.00519	0.01242	...	
167	0.00001	0.00205	0.00186	0.00616	0.02030	...	
168	0.00004	0.00490	0.00448	0.01470	0.02177	...	
169	0.00002	0.00316	0.00283	0.00949	0.02018	...	
170	0.00002	0.00279	0.00237	0.00837	0.01897	...	
171	0.00003	0.00166	0.00190	0.00499	0.01358	...	
172	0.00003	0.00170	0.00200	0.00510	0.01484	...	
173	0.00003	0.00171	0.00203	0.00514	0.01472	...	
174	0.00003	0.00176	0.00218	0.00528	0.01657	...	
175	0.00003	0.00160	0.00199	0.00480	0.01503	...	
176	0.00003	0.00169	0.00213	0.00507	0.01725	...	
177	0.00002	0.00135	0.00162	0.00406	0.01469	...	
178	0.00002	0.00152	0.00186	0.00456	0.01574	...	
179	0.00003	0.00204	0.00231	0.00612	0.01450	...	
180	0.00003	0.00206	0.00233	0.00619	0.02551	...	
181	0.00003	0.00202	0.00235	0.00605	0.01831	...	
182	0.00002	0.00174	0.00198	0.00521	0.02145	...	

183	0.00004	0.00186	0.00270	0.00558	0.01909	...
184	0.00005	0.00260	0.00346	0.00780	0.01795	...
185	0.00003	0.00134	0.00192	0.00403	0.01564	...
186	0.00004	0.00254	0.00263	0.00762	0.01660	...
187	0.00002	0.00115	0.00148	0.00345	0.01300	...
188	0.00003	0.00146	0.00184	0.00439	0.01185	...
189	0.00003	0.00412	0.00396	0.01235	0.02574	...
190	0.00003	0.00263	0.00259	0.00790	0.04087	...
191	0.00003	0.00331	0.00292	0.00994	0.02751	...
192	0.00008	0.00624	0.00564	0.01873	0.02308	...
193	0.00004	0.00370	0.00390	0.01109	0.02296	...
194	0.00003	0.00295	0.00317	0.00885	0.01884	...

	Shimmer:DDA	NHR	HNR	status	RPDE	DFA	spread1	\
0	0.06545	0.02211	21.033	1	0.414783	0.815285	-4.813031	
1	0.09403	0.01929	19.085	1	0.458359	0.819521	-4.075192	
2	0.08270	0.01309	20.651	1	0.429895	0.825288	-4.443179	
3	0.08771	0.01353	20.644	1	0.434969	0.819235	-4.117501	
4	0.10470	0.01767	19.649	1	0.417356	0.823484	-3.747787	
5	0.06985	0.01222	21.378	1	0.415564	0.825069	-4.242867	
6	0.02337	0.00607	24.886	1	0.596040	0.764112	-5.634322	
7	0.02487	0.00344	26.892	1	0.637420	0.763262	-6.167603	
8	0.03218	0.01070	21.812	1	0.615551	0.773587	-5.498678	
9	0.04324	0.01022	21.862	1	0.547037	0.798463	-5.011879	
10	0.03237	0.01166	21.118	1	0.611137	0.776156	-5.249770	
11	0.04272	0.01141	21.414	1	0.583390	0.792520	-4.960234	
12	0.01968	0.00581	25.703	1	0.460600	0.646846	-6.547148	
13	0.02184	0.01041	24.889	1	0.430166	0.665833	-5.660217	
14	0.03191	0.00609	24.922	1	0.474791	0.654027	-6.105098	
15	0.02316	0.00839	25.175	1	0.565924	0.658245	-5.340115	
16	0.02908	0.01859	22.333	1	0.567380	0.644692	-5.440040	
17	0.04322	0.02919	20.376	1	0.631099	0.605417	-2.931070	
18	0.07413	0.03160	17.280	1	0.665318	0.719467	-3.949079	
19	0.05164	0.03365	17.153	1	0.649554	0.686080	-4.554466	
20	0.05000	0.03871	17.536	1	0.660125	0.704087	-4.095442	
21	0.06062	0.01849	19.493	1	0.629017	0.698951	-5.186960	
22	0.06685	0.01280	22.468	1	0.619060	0.679834	-4.330956	
23	0.06562	0.01840	20.422	1	0.537264	0.686894	-5.248776	
24	0.02214	0.01778	23.831	1	0.397937	0.732479	-5.557447	
25	0.05197	0.02887	22.066	1	0.522746	0.737948	-5.571843	
26	0.02666	0.01095	25.908	1	0.418622	0.720916	-6.183590	
27	0.02650	0.01328	25.119	1	0.358773	0.726652	-6.271690	
28	0.02307	0.00677	25.970	1	0.470478	0.676258	-7.120925	
29	0.02380	0.01170	25.678	1	0.427785	0.723797	-6.635729	
..	...	...	...	...	...	...	...	
165	0.04231	0.00620	24.078	0	0.469928	0.628232	-6.816086	
166	0.02089	0.00533	24.679	0	0.384868	0.626710	-7.018057	
167	0.03557	0.00910	21.083	0	0.440988	0.628058	-7.517934	
168	0.03836	0.01337	19.269	0	0.372222	0.725216	-5.736781	
169	0.03529	0.00965	21.020	0	0.371837	0.646167	-7.169701	
170	0.03253	0.01049	21.528	0	0.522812	0.646818	-7.304500	
171	0.01992	0.00435	26.436	0	0.413295	0.756700	-6.323531	
172	0.02261	0.00430	26.550	0	0.369090	0.776158	-6.085567	
173	0.02245	0.00478	26.547	0	0.380253	0.766700	-5.943501	
174	0.02643	0.00590	25.445	0	0.387482	0.756482	-6.012559	
175	0.02436	0.00401	26.005	0	0.405991	0.761255	-5.966779	
176	0.02623	0.00415	26.143	0	0.361232	0.763242	-6.016891	

177	0.02184	0.00570	24.151	1	0.396610	0.745957	-6.486822
178	0.02518	0.00488	24.412	1	0.402591	0.762508	-6.311987
179	0.02175	0.00540	23.683	1	0.398499	0.778349	-5.711205
180	0.03964	0.00611	23.133	1	0.352396	0.759320	-6.261446
181	0.02849	0.00639	22.866	1	0.408598	0.768845	-5.704053
182	0.03464	0.00595	23.008	1	0.329577	0.757180	-6.277170
183	0.02592	0.00955	23.079	0	0.603515	0.669565	-5.619070
184	0.02429	0.01179	22.085	0	0.663842	0.656516	-5.198864
185	0.02001	0.00737	24.199	0	0.598515	0.654331	-5.592584
186	0.02460	0.01397	23.958	0	0.566424	0.667654	-6.431119
187	0.01892	0.00680	25.023	0	0.528485	0.663884	-6.359018
188	0.01672	0.00703	24.775	0	0.555303	0.659132	-6.710219
189	0.04363	0.04441	19.368	0	0.508479	0.683761	-6.934474
190	0.07008	0.02764	19.517	0	0.448439	0.657899	-6.538586
191	0.04812	0.01810	19.147	0	0.431674	0.683244	-6.195325
192	0.03804	0.10715	17.883	0	0.407567	0.655683	-6.787197
193	0.03794	0.07223	19.020	0	0.451221	0.643956	-6.744577
194	0.03078	0.04398	21.209	0	0.462803	0.664357	-5.724056

	spread2	D2	PPE
0	0.266482	2.301442	0.284654
1	0.335590	2.486855	0.368674
2	0.311173	2.342259	0.332634
3	0.334147	2.405554	0.368975
4	0.234513	2.332180	0.410335
5	0.299111	2.187560	0.357775
6	0.257682	1.854785	0.211756
7	0.183721	2.064693	0.163755
8	0.327769	2.322511	0.231571
9	0.325996	2.432792	0.271362
10	0.391002	2.407313	0.249740
11	0.363566	2.642476	0.275931
12	0.152813	2.041277	0.138512
13	0.254989	2.519422	0.199889
14	0.203653	2.125618	0.170100
15	0.210185	2.205546	0.234589
16	0.239764	2.264501	0.218164
17	0.434326	3.007463	0.430788
18	0.357870	3.109010	0.377429
19	0.340176	2.856676	0.322111
20	0.262564	2.739710	0.365391
21	0.237622	2.557536	0.259765
22	0.262384	2.916777	0.285695
23	0.210279	2.547508	0.253556
24	0.220890	2.692176	0.215961
25	0.236853	2.846369	0.219514
26	0.226278	2.589702	0.147403
27	0.196102	2.314209	0.162999
28	0.279789	2.241742	0.108514
29	0.209866	1.957961	0.135242
..	...	...	...
165	0.172270	2.235197	0.119652
166	0.176316	1.852402	0.091604
167	0.160414	1.881767	0.075587
168	0.164529	2.882450	0.202879
169	0.073298	2.266432	0.100881
170	0.171088	2.095237	0.096220

```
171  0.218885  2.193412  0.160376
172  0.192375  1.889002  0.174152
173  0.192150  1.852542  0.179677
174  0.229298  1.872946  0.163118
175  0.197938  1.974857  0.184067
176  0.109256  2.004719  0.174429
177  0.197919  2.449763  0.132703
178  0.182459  2.251553  0.160306
179  0.240875  2.845109  0.192730
180  0.183218  2.264226  0.144105
181  0.216204  2.679185  0.197710
182  0.109397  2.209021  0.156368
183  0.191576  2.027228  0.215724
184  0.206768  2.120412  0.252404
185  0.133917  2.058658  0.214346
186  0.153310  2.161936  0.120605
187  0.116636  2.152083  0.138868
188  0.149694  1.913990  0.121777
189  0.159890  2.316346  0.112838
190  0.121952  2.657476  0.133050
191  0.129303  2.784312  0.168895
192  0.158453  2.679772  0.131728
193  0.207454  2.138608  0.123306
194  0.190667  2.555477  0.148569

[195 rows x 24 columns]
```

In [3]:
```
df.isnull().sum()
```

Out[3]:
```
name              0
MDVP:Fo(Hz)       0
MDVP:Fhi(Hz)      0
MDVP:Flo(Hz)      0
MDVP:Jitter(%)    0
MDVP:Jitter(Abs)  0
MDVP:RAP          0
MDVP:PPQ          0
Jitter:DDP        0
MDVP:Shimmer      0
MDVP:Shimmer(dB)  0
Shimmer:APQ3      0
Shimmer:APQ5      0
MDVP:APQ          0
Shimmer:DDA       0
NHR               0
HNR               0
status            0
RPDE              0
DFA               0
spread1           0
spread2           0
D2                0
PPE               0
dtype: int64
```

In [4]:
```
#checking for correalation among the variables. Removing the variables with correlation > 0.7
corr = df.corr()
corr.style.background_gradient(cmap='coolwarm')
```

Out[4]:

	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)
MDVP:Fo(Hz)	1	0.400985	0.596546	-0.118003	-0.382027
MDVP:Fhi(Hz)	0.400985	1	0.0849513	0.102086	-0.0291983
MDVP:Flo(Hz)	0.596546	0.0849513	1	-0.139919	-0.277815
MDVP:Jitter(%)	-0.118003	0.102086	-0.139919	1	0.935714
MDVP:Jitter(Abs)	-0.382027	-0.0291983	-0.277815	0.935714	1
MDVP:RAP	-0.0761938	0.0971766	-0.100519	0.990276	0.922911
MDVP:PPQ	-0.112165	0.0911262	-0.0958284	0.974256	0.897778
Jitter:DDP	-0.0762127	0.0971499	-0.100488	0.990276	0.922913
MDVP:Shimmer	-0.0983737	0.00228123	-0.144543	0.769063	0.703322
MDVP:Shimmer(dB)	-0.0737425	0.0434652	-0.119089	0.804289	0.716601
Shimmer:APQ3	-0.0947171	-0.00374325	-0.150747	0.746625	0.697153
Shimmer:APQ5	-0.0706818	-0.00999678	-0.101095	0.725561	0.648961
MDVP:APQ	-0.0777738	0.00493698	-0.107293	0.758255	0.648793
Shimmer:DDA	-0.0947316	-0.00373289	-0.150737	0.746635	0.69717
NHR	-0.0219808	0.163766	-0.10867	0.906959	0.834972
HNR	0.0591444	-0.0248931	0.210851	-0.728165	-0.65681
status	-0.383535	-0.166136	-0.3802	0.27822	0.338653
RPDE	-0.383894	-0.112404	-0.400143	0.360673	0.441839
DFA	-0.446013	-0.343097	-0.0504063	0.0985724	0.175036
spread1	-0.413738	-0.0766578	-0.394857	0.693577	0.735779
spread2	-0.24945	-0.00295361	-0.243829	0.385123	0.388543
D2	0.17798	0.176323	-0.100629	0.433434	0.310694
PPE	-0.372356	-0.069543	-0.340071	0.721543	0.748162

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 24 columns):
name                195 non-null object
MDVP:Fo(Hz)         195 non-null float64
MDVP:Fhi(Hz)        195 non-null float64
MDVP:Flo(Hz)        195 non-null float64
MDVP:Jitter(%)      195 non-null float64
MDVP:Jitter(Abs)    195 non-null float64
MDVP:RAP            195 non-null float64
MDVP:PPQ            195 non-null float64
Jitter:DDP          195 non-null float64
MDVP:Shimmer        195 non-null float64
MDVP:Shimmer(dB)    195 non-null float64
Shimmer:APQ3        195 non-null float64
Shimmer:APQ5        195 non-null float64
MDVP:APQ            195 non-null float64
Shimmer:DDA         195 non-null float64
NHR                 195 non-null float64
HNR                 195 non-null float64
status              195 non-null int64
RPDE                195 non-null float64
DFA                 195 non-null float64
spread1             195 non-null float64
spread2             195 non-null float64
D2                  195 non-null float64
PPE                 195 non-null float64
dtypes: float64(22), int64(1), object(1)
memory usage: 36.6+ KB
```

In [6]:
```
# csv version of correlation
df.corr(method ='pearson')
df.to_csv('corr.csv')
```

In [7]:
```
# final variables for modelling after removing correlation
df_1 =df[['MDVP:Fhi(Hz)','MDVP:Flo(Hz)','MDVP:Jitter(%)','NHR','DFA','spread1'
,'spread2','D2','PPE','status']]
df_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 10 columns):
MDVP:Fhi(Hz)      195 non-null float64
MDVP:Flo(Hz)      195 non-null float64
MDVP:Jitter(%)    195 non-null float64
NHR               195 non-null float64
DFA               195 non-null float64
spread1           195 non-null float64
spread2           195 non-null float64
D2                195 non-null float64
PPE               195 non-null float64
status            195 non-null int64
dtypes: float64(9), int64(1)
memory usage: 15.3 KB
```

In [8]:
```
# checking if there is a null value in the dataset
df_1.isnull().sum()
```

Out[8]:
```
MDVP:Fhi(Hz)      0
MDVP:Flo(Hz)      0
MDVP:Jitter(%)    0
NHR               0
DFA               0
spread1           0
spread2           0
D2                0
PPE               0
status            0
dtype: int64
```

In [9]:
```
# print no of 0s and 1s in the dependant variable
print(df_1[df_1.status==0])
print(df_1[df_1.status==1])

# to check the unique values in a df
#print(df_1['status'].unique())
```

	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	NHR	DFA	spread1
30	206.896	192.055	0.00289	0.00339	0.741367	-7.348300
31	209.512	192.091	0.00241	0.00167	0.742055	-7.682587
32	215.203	193.104	0.00212	0.00119	0.738703	-7.067931
33	211.604	197.079	0.00180	0.00072	0.742133	-7.695734
34	211.526	196.160	0.00178	0.00065	0.741899	-7.964984
35	210.565	195.708	0.00198	0.00135	0.742737	-7.777685
42	247.326	225.227	0.00298	0.00740	0.654172	-7.310550
43	248.834	232.483	0.00281	0.00675	0.634267	-6.793547
44	250.912	232.435	0.00210	0.00454	0.635285	-7.057869
45	255.034	227.911	0.00225	0.00476	0.638928	-6.995820
46	262.090	231.848	0.00235	0.00476	0.631653	-7.156076
47	261.487	182.786	0.00185	0.00432	0.635204	-7.319510
48	128.611	115.765	0.00524	0.00839	0.733659	-6.439398
49	130.049	114.676	0.00428	0.00462	0.754073	-6.482096
50	135.069	117.495	0.00431	0.00479	0.775933	-6.650471
51	134.231	112.773	0.00448	0.00474	0.760361	-6.689151
52	138.052	122.080	0.00436	0.00481	0.766204	-7.072419
53	139.867	118.604	0.00490	0.00484	0.785714	-6.836811
60	237.494	109.379	0.00282	0.00871	0.678874	-7.040508
61	238.987	98.664	0.00264	0.00301	0.686264	-7.293801
62	231.345	205.495	0.00266	0.00340	0.694399	-6.966321
63	234.619	223.634	0.00296	0.00351	0.683296	-7.245620
64	252.221	221.156	0.00205	0.00300	0.673636	-7.496264
65	239.541	113.201	0.00238	0.00420	0.681811	-7.314237
165	244.663	102.137	0.00277	0.00620	0.628232	-6.816086
166	243.709	229.256	0.00303	0.00533	0.626710	-7.018057
167	264.919	237.303	0.00339	0.00910	0.628058	-7.517934
168	217.627	90.794	0.00803	0.01337	0.725216	-5.736781
169	245.135	219.783	0.00517	0.00965	0.646167	-7.169701
170	272.210	239.170	0.00451	0.01049	0.646818	-7.304500
171	133.374	105.715	0.00355	0.00435	0.756700	-6.323531
172	113.597	100.139	0.00356	0.00430	0.776158	-6.085567
173	116.443	96.913	0.00349	0.00478	0.766700	-5.943501
174	144.466	99.923	0.00353	0.00590	0.756482	-6.012559
175	123.109	108.634	0.00332	0.00401	0.761255	-5.966779
176	129.038	108.970	0.00346	0.00415	0.763242	-6.016891
183	123.925	106.656	0.00417	0.00955	0.669565	-5.619070
184	217.552	99.503	0.00531	0.01179	0.656516	-5.198864
185	177.291	96.983	0.00314	0.00737	0.654331	-5.592584
186	592.030	86.228	0.00496	0.01397	0.667654	-6.431119
187	581.289	94.246	0.00267	0.00680	0.663884	-6.359018
188	119.167	86.647	0.00327	0.00703	0.659132	-6.710219
189	262.707	78.228	0.00694	0.04441	0.683761	-6.934474
190	230.978	94.261	0.00459	0.02764	0.657899	-6.538586
191	253.017	89.488	0.00564	0.01810	0.683244	-6.195325
192	240.005	74.287	0.01360	0.10715	0.655683	-6.787197
193	396.961	74.904	0.00740	0.07223	0.643956	-6.744577
194	260.277	77.973	0.00567	0.04398	0.664357	-5.724056

	spread2	D2	PPE	status
30	0.177551	1.743867	0.085569	0
31	0.173319	2.103106	0.068501	0
32	0.175181	1.512275	0.096320	0
33	0.178540	1.544609	0.056141	0
34	0.163519	1.423287	0.044539	0

```
35   0.170183  2.447064  0.057610       0
42   0.098648  2.416838  0.095032       0
43   0.158266  2.256699  0.117399       0
44   0.091608  2.330716  0.091470       0
45   0.102083  2.365800  0.102706       0
46   0.127642  2.392122  0.097336       0
47   0.200873  2.028612  0.086398       0
48   0.266392  2.079922  0.133867       0
49   0.264967  2.054419  0.128872       0
50   0.254498  1.840198  0.103561       0
51   0.291954  2.431854  0.105993       0
52   0.220434  1.972297  0.119308       0
53   0.269866  2.223719  0.147491       0
60   0.066994  2.460791  0.101516       0
61   0.086372  2.321560  0.098555       0
62   0.095882  2.278687  0.103224       0
63   0.018689  2.498224  0.093534       0
64   0.056844  2.003032  0.073581       0
65   0.006274  2.118596  0.091546       0
165  0.172270  2.235197  0.119652       0
166  0.176316  1.852402  0.091604       0
167  0.160414  1.881767  0.075587       0
168  0.164529  2.882450  0.202879       0
169  0.073298  2.266432  0.100881       0
170  0.171088  2.095237  0.096220       0
171  0.218885  2.193412  0.160376       0
172  0.192375  1.889002  0.174152       0
173  0.192150  1.852542  0.179677       0
174  0.229298  1.872946  0.163118       0
175  0.197938  1.974857  0.184067       0
176  0.109256  2.004719  0.174429       0
183  0.191576  2.027228  0.215724       0
184  0.206768  2.120412  0.252404       0
185  0.133917  2.058658  0.214346       0
186  0.153310  2.161936  0.120605       0
187  0.116636  2.152083  0.138868       0
188  0.149694  1.913990  0.121777       0
189  0.159890  2.316346  0.112838       0
190  0.121952  2.657476  0.133050       0
191  0.129303  2.784312  0.168895       0
192  0.158453  2.679772  0.131728       0
193  0.207454  2.138608  0.123306       0
194  0.190667  2.555477  0.148569       0
     MDVP:Fhi(Hz)  MDVP:Flo(Hz)  MDVP:Jitter(%)      NHR       DFA   spread1
\
0         157.302        74.997         0.00784  0.02211  0.815285 -4.813031
1         148.650       113.819         0.00968  0.01929  0.819521 -4.075192
2         131.111       111.555         0.01050  0.01309  0.825288 -4.443179
3         137.871       111.366         0.00997  0.01353  0.819235 -4.117501
4         141.781       110.655         0.01284  0.01767  0.823484 -3.747787
5         131.162       113.787         0.00968  0.01222  0.825069 -4.242867
6         137.244       114.820         0.00333  0.00607  0.764112 -5.634322
7         113.840       104.315         0.00290  0.00344  0.763262 -6.167603
8         132.068        91.754         0.00551  0.01070  0.773587 -5.498678
9         120.103        91.226         0.00532  0.01022  0.798463 -5.011879
10        112.240        84.072         0.00505  0.01166  0.776156 -5.249770
11        115.871        86.292         0.00540  0.01141  0.792520 -4.960234
```

12	159.866	131.276	0.00293	0.00581	0.646846	-6.547148
13	179.139	76.556	0.00390	0.01041	0.665833	-5.660217
14	163.305	75.836	0.00294	0.00609	0.654027	-6.105098
15	217.455	83.159	0.00369	0.00839	0.658245	-5.340115
16	349.259	82.764	0.00544	0.01859	0.644692	-5.440040
17	232.181	75.603	0.00718	0.02919	0.605417	-2.931070
18	175.829	68.623	0.00742	0.03160	0.719467	-3.949079
19	189.398	142.822	0.00768	0.03365	0.686080	-4.554466
20	165.738	65.782	0.00840	0.03871	0.704087	-4.095442
21	172.860	78.128	0.00480	0.01849	0.698951	-5.186960
22	193.221	79.068	0.00442	0.01280	0.679834	-4.330956
23	192.735	86.180	0.00476	0.01840	0.686894	-5.248776
24	200.841	76.779	0.00742	0.01778	0.732479	-5.557447
25	206.002	77.968	0.00633	0.02887	0.737948	-5.571843
26	208.313	75.501	0.00455	0.01095	0.720916	-6.183590
27	208.701	81.737	0.00496	0.01328	0.726652	-6.271690
28	227.383	80.055	0.00310	0.00677	0.676258	-7.120925
29	198.346	77.630	0.00502	0.01170	0.723797	-6.635729
..	...	...	...	...	...	...
141	253.792	91.802	0.00757	0.04238	0.665945	-5.410336
142	219.290	148.691	0.00376	0.01728	0.661735	-5.585259
143	231.508	86.232	0.00370	0.02010	0.632631	-5.898673
144	241.350	164.168	0.00254	0.01049	0.630409	-6.132663
145	263.872	87.638	0.00352	0.01493	0.574282	-5.456811
146	191.759	151.451	0.01568	0.07530	0.793509	-3.297668
147	216.814	161.340	0.01466	0.06057	0.768974	-4.276605
148	216.302	165.982	0.01719	0.08069	0.764036	-3.377325
149	565.740	177.258	0.01627	0.07889	0.775708	-4.892495
150	211.961	149.442	0.01872	0.10952	0.762726	-4.484303
151	224.429	168.793	0.03107	0.21713	0.768320	-2.434031
152	233.099	174.478	0.02714	0.16265	0.754449	-2.839756
153	139.644	98.250	0.00684	0.04179	0.670475	-4.865194
154	128.442	88.833	0.00692	0.04611	0.659333	-4.239028
155	127.349	95.654	0.00647	0.02631	0.652025	-3.583722
156	142.369	94.794	0.00727	0.03191	0.623731	-5.435100
157	134.209	100.757	0.01813	0.10748	0.646786	-3.444478
158	154.284	97.543	0.00975	0.03828	0.627337	-5.070096
159	138.752	112.173	0.00605	0.02663	0.675865	-5.498456
160	124.393	77.022	0.00581	0.02073	0.694571	-5.185987
161	135.738	107.802	0.00619	0.02810	0.684373	-5.283009
162	126.778	91.121	0.00651	0.02707	0.719576	-5.529833
163	131.669	97.527	0.00519	0.01435	0.673086	-5.617124
164	142.830	85.902	0.00907	0.03882	0.674562	-2.929379
177	190.204	129.859	0.00314	0.00570	0.745957	-6.486822
178	158.359	138.990	0.00309	0.00488	0.762508	-6.311987
179	155.982	135.041	0.00392	0.00540	0.778349	-5.711205
180	163.441	144.736	0.00396	0.00611	0.759320	-6.261446
181	161.078	141.998	0.00397	0.00639	0.768845	-5.704053
182	163.417	144.786	0.00336	0.00595	0.757180	-6.277170

	spread2	D2	PPE	status
0	0.266482	2.301442	0.284654	1
1	0.335590	2.486855	0.368674	1
2	0.311173	2.342259	0.332634	1
3	0.334147	2.405554	0.368975	1
4	0.234513	2.332180	0.410335	1
5	0.299111	2.187560	0.357775	1

6	0.257682	1.854785	0.211756	1
7	0.183721	2.064693	0.163755	1
8	0.327769	2.322511	0.231571	1
9	0.325996	2.432792	0.271362	1
10	0.391002	2.407313	0.249740	1
11	0.363566	2.642476	0.275931	1
12	0.152813	2.041277	0.138512	1
13	0.254989	2.519422	0.199889	1
14	0.203653	2.125618	0.170100	1
15	0.210185	2.205546	0.234589	1
16	0.239764	2.264501	0.218164	1
17	0.434326	3.007463	0.430788	1
18	0.357870	3.109010	0.377429	1
19	0.340176	2.856676	0.322111	1
20	0.262564	2.739710	0.365391	1
21	0.237622	2.557536	0.259765	1
22	0.262384	2.916777	0.285695	1
23	0.210279	2.547508	0.253556	1
24	0.220890	2.692176	0.215961	1
25	0.236853	2.846369	0.219514	1
26	0.226278	2.589702	0.147403	1
27	0.196102	2.314209	0.162999	1
28	0.279789	2.241742	0.108514	1
29	0.209866	1.957961	0.135242	1
..	...	...	...	...
141	0.288917	2.665133	0.231723	1
142	0.310746	2.465528	0.209863	1
143	0.213353	2.470746	0.189032	1
144	0.220617	2.576563	0.159777	1
145	0.345238	2.840556	0.232861	1
146	0.414758	3.413649	0.457533	1
147	0.355736	3.142364	0.336085	1
148	0.335357	3.274865	0.418646	1
149	0.262281	2.910213	0.270173	1
150	0.340256	2.958815	0.301487	1
151	0.450493	3.079221	0.527367	1
152	0.356224	3.184027	0.454721	1
153	0.246404	2.013530	0.168581	1
154	0.175691	2.451130	0.247455	1
155	0.207914	2.439597	0.206256	1
156	0.230532	2.699645	0.220546	1
157	0.303214	2.964568	0.261305	1
158	0.280091	2.892300	0.249703	1
159	0.234196	2.103014	0.216638	1
160	0.259229	2.151121	0.244948	1
161	0.226528	2.442906	0.238281	1
162	0.242750	2.408689	0.220520	1
163	0.184896	1.871871	0.212386	1
164	0.396746	2.560422	0.367233	1
177	0.197919	2.449763	0.132703	1
178	0.182459	2.251553	0.160306	1
179	0.240875	2.845109	0.192730	1
180	0.183218	2.264226	0.144105	1
181	0.216204	2.679185	0.197710	1
182	0.109397	2.209021	0.156368	1

[147 rows x 10 columns]

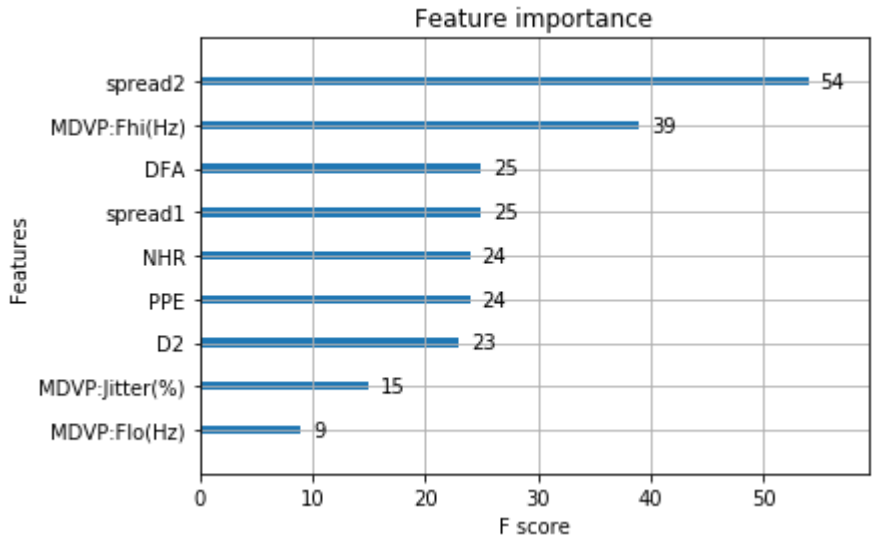
In [10]:
```
#splitting training and testing data
# stratify = y will maintain the ratio of 0s and 1s as population
y= df_1.status
x=df_1.drop(['status'],axis=1)
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size =0.3,stratify=y
)
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(136, 9)
(136,)
(59, 9)
(59,)
```

In [11]:
```
# building the xgboost model with training data and testing it on testing data
from xgboost import XGBClassifier
model = XGBClassifier()
import xgboost as xgb
model.fit(x_train, y_train)
pred1 =model.predict (x_test)
# Model based feature importance
plt.show()
xgb.plot_importance(model)
```

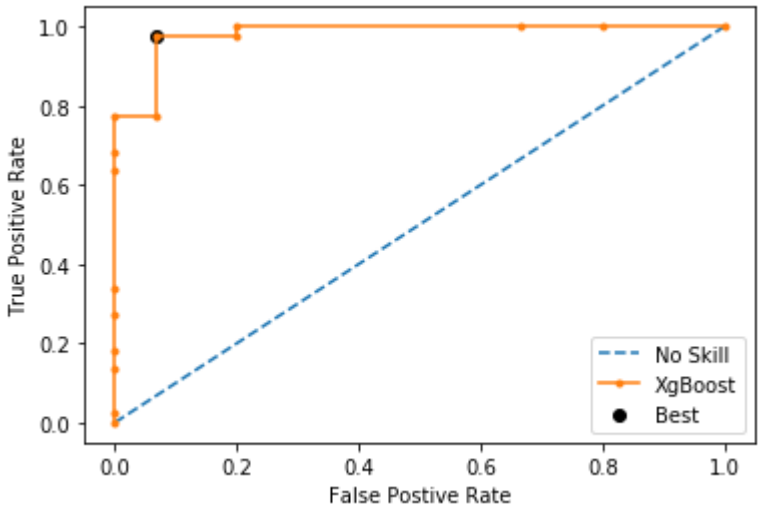
```
[20:14:21] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.
3.0/src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation me
tric used with the objective 'binary:logistic' was changed from 'error' to 'l
ogloss'. Explicitly set eval_metric if you'd like to restore the old behavio
r.

C:\Users\vatsa\Anaconda3\lib\site-packages\xgboost\sklearn.py:888: UserWarnin
g: The use of label encoder in XGBClassifier is deprecated and will be remove
d in a future release. To remove this warning, do the following: 1) Pass opti
on use_label_encoder=False when constructing XGBClassifier object; and 2) Enc
ode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_clas
s - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x1dd6beb15f8>



In [12]:
```
# Plotting the Auc curve and best threshold
yhat = model.predict_proba(x_test)[:,1]
fpr,tpr,thresholds =roc_curve(y_test,yhat)
gmeans =sqrt(tpr* (1-fpr))
ix=argmax(gmeans)
print('Best threshold=%f ,G-Mean =%.3f' % (thresholds[ix],gmeans[ix]))
pyplot.plot([0,1],[0,1],linestyle ='--',label ='No Skill')
pyplot.plot(fpr,tpr,marker='.',label='XgBoost')
pyplot.scatter(fpr[ix],tpr[ix],marker='o',color='black',label='Best')
pyplot.xlabel('False Postive Rate')
pyplot.ylabel('True Positive Rate')
pyplot.legend()
pyplot.show()
```

```
Best threshold=0.521453 ,G-Mean =0.955
```



In [13]:
```
# based on confusion matrix, getting the accuracy, specificity, sensitivity
#default threshold is 0.5
from sklearn.metrics import confusion_matrix
cm1 =confusion_matrix(y_test,pred1)
print(cm1)
total =sum(sum(cm1))

accuracy = (cm1[0,0]+cm1[1,1])/total
print('Accuracy :',accuracy)

specificity = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('specificity :',specificity)

sensitivity = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('sensitivity :',sensitivity)
```

```
[[14  1]
 [ 1 43]]
Accuracy : 0.9661016949152542
specificity : 0.9333333333333333
sensitivity : 0.9772727272727273
```

In [14]:
```
#XGBoost hyper-parameter tuning
#def hyperParameterTuning(X_train, y_train):
 #   param_tuning = {
  #      'learning_rate': [0.01, 0.1],
   #     'max_depth': [3, 5, 7, 10],
    #    'min_child_weight': [1, 3, 5],
     #   'subsample': [0.5, 0.7],
      #  'colsample_bytree': [0.5, 0.7],
       # 'n_estimators' : [100, 200, 500],
        #'objective': ['reg:squarederror']
    #}

    #xgb_model = XGBClassifier()
    #scoring = {'AUC':'roc_auc', 'Accuracy':make_scorer(accuracy_score)}
    #kfold = KFold(n_splits=10, random_state=42)

    #gsearch = GridSearchCV(estimator = xgb_model,
     #                      param_grid = param_tuning,
      #                     scoring =scoring,
       #                    cv = kfold,
        #                   refit='AUC',
         #                  n_jobs = -1,
          #                 verbose = 1)

 #   gsearch.fit(x_train,y_train)

   # return gsearch.best_params_
#hyperParameterTuning(x_train, y_train)
```

In [15]:
```
from sklearn.svm import SVC
svclassifier = SVC(kernel='linear',probability=True)
#linear
#poly, degree =3
#rbf
#linear is performing better, but  the specificity is very low. so not choosin
g this algorithmmmmodel
model=svclassifier.fit(x_train, y_train)
```

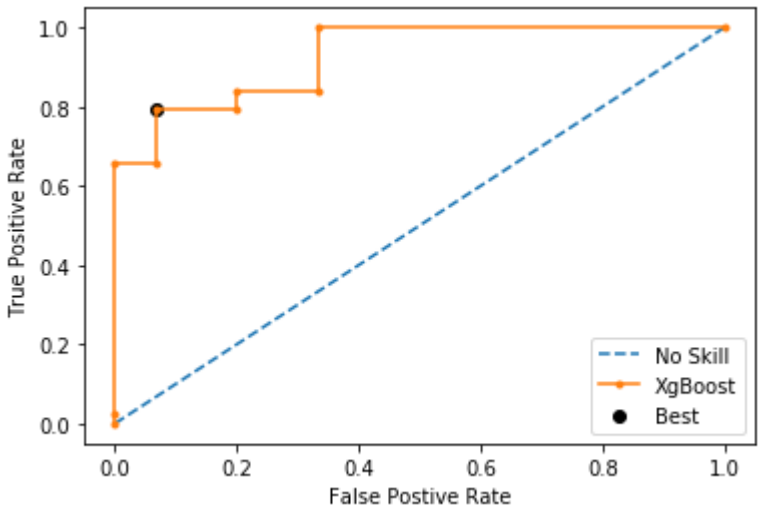
In [16]:
```
y_pred = svclassifier.predict(x_test)
```

In [17]:
```
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
[[10  5]
 [ 3 41]]
              precision    recall  f1-score   support

           0       0.77      0.67      0.71        15
           1       0.89      0.93      0.91        44

    accuracy                           0.86        59
   macro avg       0.83      0.80      0.81        59
weighted avg       0.86      0.86      0.86        59
```

In [18]:
```
# Plotting the Auc curve and best threshold
yhat = model.predict_proba(x_test)[:,1]
fpr,tpr,thresholds =roc_curve(y_test,yhat)
gmeans =sqrt(tpr* (1-fpr))
ix=argmax(gmeans)
print('Best threshold=%f ,G-Mean =%.3f' % (thresholds[ix],gmeans[ix]))
pyplot.plot([0,1],[0,1],linestyle ='--',label ='No Skill')
pyplot.plot(fpr,tpr,marker='.',label='XgBoost')
pyplot.scatter(fpr[ix],tpr[ix],marker='o',color='black',label='Best')
pyplot.xlabel('False Postive Rate')
pyplot.ylabel('True Positive Rate')
pyplot.legend()
pyplot.show()
```

```
Best threshold=0.719299 ,G-Mean =0.862
```



In [19]:
```
# based on confusion matrix, getting the accuracy, specificity, sensitivity
#default threshold is 0.5
from sklearn.metrics import confusion_matrix
cm1 =confusion_matrix(y_test,y_pred)
total =sum(sum(cm1))

accuracy = (cm1[0,0]+cm1[1,1])/total
print('Accuracy :',accuracy)

specificity = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('specificity :',specificity)

sensitivity = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('sensitivity :',sensitivity)
```

```
Accuracy : 0.864406779661017
specificity : 0.6666666666666666
sensitivity : 0.9318181818181818
```

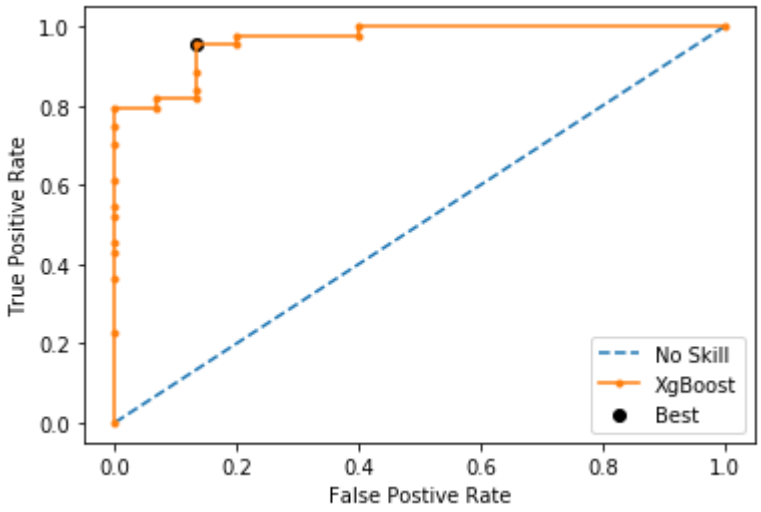
In [20]:
```
from sklearn.ensemble import RandomForestClassifier
model_2=RandomForestClassifier(n_estimators=100)
model_2.fit(x_train,y_train)
pred_2=model_2.predict(x_test)
```

In [21]:
```
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,pred_2))
print(classification_report(y_test,pred_2))
```

```
[[13  2]
 [ 2 42]]
              precision    recall  f1-score   support

           0       0.87      0.87      0.87        15
           1       0.95      0.95      0.95        44

    accuracy                           0.93        59
   macro avg       0.91      0.91      0.91        59
weighted avg       0.93      0.93      0.93        59
```

In [22]:
```
# Plotting the Auc curve and best threshold
yhat = model_2.predict_proba(x_test)[:,1]
fpr,tpr,thresholds =roc_curve(y_test,yhat)
gmeans =sqrt(tpr* (1-fpr))
ix=argmax(gmeans)
print('Best threshold=%f ,G-Mean =%.3f' % (thresholds[ix],gmeans[ix]))
pyplot.plot([0,1],[0,1],linestyle ='--',label ='No Skill')
pyplot.plot(fpr,tpr,marker='.',label='XgBoost')
pyplot.scatter(fpr[ix],tpr[ix],marker='o',color='black',label='Best')
pyplot.xlabel('False Postive Rate')
pyplot.ylabel('True Positive Rate')
pyplot.legend()
pyplot.show()
```

```
Best threshold=0.510000 ,G-Mean =0.910
```



In [23]:
```
# based on confusion matrix, getting the accuracy, specificity, sensitivity
#default threshold is 0.5
from sklearn.metrics import confusion_matrix
cm1 =confusion_matrix(y_test,pred_2)
total =sum(sum(cm1))

accuracy = (cm1[0,0]+cm1[1,1])/total
print('Accuracy :',accuracy)

specificity = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('specificity :',specificity)

sensitivity = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('sensitivity :',sensitivity)
```

```
Accuracy : 0.9322033898305084
specificity : 0.8666666666666667
sensitivity : 0.9545454545454546
```

In [24]:
```
# Method 1 - Voting
from sklearn.ensemble import VotingClassifier
model = VotingClassifier(estimators=[('xg', model), ('rf', model_2)], voting=
'hard')
model.fit(x_train,y_train)
model.score(x_test,y_test)
```

Out[24]:
```
0.9152542372881356
```