PROJECT : Parkinson's Disease Detection

Attribute Information:

name - ASCII subject name and recording number

MDVP:Fo(Hz) - Average vocal fundamental frequency

MDVP:Fhi(Hz) - Maximum vocal fundamental frequency

MDVP:Flo(Hz) - Minimum vocal fundamental frequency

MDVP:Jitter(%),MDVP:Jitter(Abs),MDVP:RAP,MDVP:PPQ,Jitter:DDP - Several measures of variation in fundamental frequency

MDVP:Shimmer,MDVP:Shimmer(dB),Shimmer:APQ3,Shimmer:APQ5,MDVP:APQ,Shimmer:DDA - Several measures of variation in amplitude

NHR,HNR - Two measures of ratio of noise to tonal components in the voice

status - Health status of the subject (one) - Parkinson's, (zero) - healthy

RPDE,D2 - Two nonlinear dynamical complexity measures

DFA - Signal fractal scaling exponent

spread1,spread2,PPE - Three nonlinear measures of fundamental frequency variation

Import required libraries

In [ ]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import GridSearchCV
df=pd.read_csv('/content/archive(4).zip')
df
```

Out[ ]:

	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	...	Shimmer:DDA	NHR	HNR	status	RPDE	DFA	spread1	spread2	D2	PPE
0	phon_R01_S01_1	119.992	157.302	74.997	0.00784	0.00007	0.00370	0.00554	0.01109	0.04374	...	0.06545	0.02211	21.033	1	0.414783	0.815285	-4.813031	0.266482	2.301442	0.284654
1	phon_R01_S01_2	122.400	148.650	113.819	0.00968	0.00008	0.00465	0.00696	0.01394	0.06134	...	0.09403	0.01929	19.085	1	0.458359	0.819521	-4.075192	0.335590	2.486855	0.368674
2	phon_R01_S01_3	116.682	131.111	111.555	0.01050	0.00009	0.00544	0.00781	0.01633	0.05233	...	0.08270	0.01309	20.651	1	0.429895	0.825288	-4.443179	0.311173	2.342259	0.332634
3	phon_R01_S01_4	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.01505	0.05492	...	0.08771	0.01353	20.644	1	0.434969	0.819235	-4.117501	0.334147	2.405554	0.368975
4	phon_R01_S01_5	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.01966	0.06425	...	0.10470	0.01767	19.649	1	0.417356	0.823484	-3.747787	0.234513	2.332180	0.410335
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
190	phon_R01_S50_2	174.188	230.978	94.261	0.00459	0.00003	0.00263	0.00259	0.00790	0.04087	...	0.07008	0.02764	19.517	0	0.448439	0.657899	-6.538586	0.121952	2.657476	0.133050
191	phon_R01_S50_3	209.516	253.017	89.488	0.00564	0.00003	0.00331	0.00292	0.00994	0.02751	...	0.04812	0.01810	19.147	0	0.431674	0.683244	-6.195325	0.129303	2.784312	0.168895
192	phon_R01_S50_4	174.688	240.005	74.287	0.01360	0.00008	0.00624	0.00564	0.01873	0.02308	...	0.03804	0.10715	17.883	0	0.407567	0.655683	-6.787197	0.158453	2.679772	0.131728
193	phon_R01_S50_5	198.764	396.961	74.904	0.00740	0.00004	0.00370	0.00390	0.01109	0.02296	...	0.03794	0.07223	19.020	0	0.451221	0.643956	-6.744577	0.207454	2.138608	0.123306
194	phon_R01_S50_6	214.289	260.277	77.973	0.00567	0.00003	0.00295	0.00317	0.00885	0.01884	...	0.03078	0.04398	21.209	0	0.462803	0.664357	-5.724056	0.190667	2.555477	0.148569

195 rows × 24 columns

In [ ]:
```
df.head()
```

Out[ ]:

	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	...	Shimmer:DDA	NHR	HNR	status	RPDE	DFA	spread1	spread2	D2	PPE
0	phon_R01_S01_1	119.992	157.302	74.997	0.00784	0.00007	0.00370	0.00554	0.01109	0.04374	...	0.06545	0.02211	21.033	1	0.414783	0.815285	-4.813031	0.266482	2.301442	0.284654
1	phon_R01_S01_2	122.400	148.650	113.819	0.00968	0.00008	0.00465	0.00696	0.01394	0.06134	...	0.09403	0.01929	19.085	1	0.458359	0.819521	-4.075192	0.335590	2.486855	0.368674
2	phon_R01_S01_3	116.682	131.111	111.555	0.01050	0.00009	0.00544	0.00781	0.01633	0.05233	...	0.08270	0.01309	20.651	1	0.429895	0.825288	-4.443179	0.311173	2.342259	0.332634
3	phon_R01_S01_4	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.01505	0.05492	...	0.08771	0.01353	20.644	1	0.434969	0.819235	-4.117501	0.334147	2.405554	0.368975
4	phon_R01_S01_5	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.01966	0.06425	...	0.10470	0.01767	19.649	1	0.417356	0.823484	-3.747787	0.234513	2.332180	0.410335

5 rows × 24 columns

In [ ]:
```
df.tail()
```

Out[ ]:

	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	...	Shimmer:DDA	NHR	HNR	status	RPDE	DFA	spread1	spread2	D2	PPE
190	phon_R01_S50_2	174.188	230.978	94.261	0.00459	0.00003	0.00263	0.00259	0.00790	0.04087	...	0.07008	0.02764	19.517	0	0.448439	0.657899	-6.538586	0.121952	2.657476	0.133050
191	phon_R01_S50_3	209.516	253.017	89.488	0.00564	0.00003	0.00331	0.00292	0.00994	0.02751	...	0.04812	0.01810	19.147	0	0.431674	0.683244	-6.195325	0.129303	2.784312	0.168895
192	phon_R01_S50_4	174.688	240.005	74.287	0.01360	0.00008	0.00624	0.00564	0.01873	0.02308	...	0.03804	0.10715	17.883	0	0.407567	0.655683	-6.787197	0.158453	2.679772	0.131728
193	phon_R01_S50_5	198.764	396.961	74.904	0.00740	0.00004	0.00370	0.00390	0.01109	0.02296	...	0.03794	0.07223	19.020	0	0.451221	0.643956	-6.744577	0.207454	2.138608	0.123306
194	phon_R01_S50_6	214.289	260.277	77.973	0.00567	0.00003	0.00295	0.00317	0.00885	0.01884	...	0.03078	0.04398	21.209	0	0.462803	0.664357	-5.724056	0.190667	2.555477	0.148569

5 rows × 24 columns

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 24 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   name              195 non-null    object 
 1   MDVP:Fo(Hz)       195 non-null    float64
 2   MDVP:Fhi(Hz)      195 non-null    float64
 3   MDVP:Flo(Hz)      195 non-null    float64
 4   MDVP:Jitter(%)    195 non-null    float64
 5   MDVP:Jitter(Abs)  195 non-null    float64
 6   MDVP:RAP          195 non-null    float64
 7   MDVP:PPQ          195 non-null    float64
 8   Jitter:DDP        195 non-null    float64
 9   MDVP:Shimmer      195 non-null    float64
 10  MDVP:Shimmer(dB)  195 non-null    float64
 11  Shimmer:APQ3      195 non-null    float64
 12  Shimmer:APQ5      195 non-null    float64
 13  MDVP:APQ          195 non-null    float64
 14  Shimmer:DDA       195 non-null    float64
 15  NHR               195 non-null    float64
 16  HNR               195 non-null    float64
 17  status            195 non-null    int64  
 18  RPDE              195 non-null    float64
 19  DFA               195 non-null    float64
 20  spread1           195 non-null    float64
 21  spread2           195 non-null    float64
 22  D2                195 non-null    float64
 23  PPE               195 non-null    float64
dtypes: float64(22), int64(1), object(1)
memory usage: 36.7+ KB
```

Checking missing values

In [ ]:
```
df.isna().sum()
```

Out[ ]:
```
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 [ ]:
```
df.dtypes
```

Out[ ]:
```
name               object
MDVP:Fo(Hz)       float64
MDVP:Fhi(Hz)      float64
MDVP:Flo(Hz)      float64
MDVP:Jitter(%)    float64
MDVP:Jitter(Abs)  float64
MDVP:RAP          float64
MDVP:PPQ          float64
Jitter:DDP        float64
MDVP:Shimmer      float64
MDVP:Shimmer(dB)  float64
Shimmer:APQ3      float64
Shimmer:APQ5      float64
MDVP:APQ          float64
Shimmer:DDA       float64
NHR               float64
HNR               float64
status              int64
RPDE              float64
DFA               float64
spread1           float64
spread2           float64
D2                float64
PPE               float64
dtype: object
```

Statistical measures of the dataset

In [ ]:
```
df.describe()
```

Out[ ]:

	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	MDVP:Shimmer(dB)	...	Shimmer:DDA	NHR	HNR	status	RPDE	DFA	spread1	spread2
count	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	...	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000
mean	154.228641	197.104918	116.324631	0.006220	0.000044	0.003306	0.003446	0.009920	0.029709	0.282251	...	0.046993	0.024847	21.885974	0.753846	0.498536	0.718099	-5.684397	0.226510
std	41.390065	91.491548	43.521413	0.004848	0.000035	0.002968	0.002759	0.008903	0.018857	0.194877	...	0.030459	0.040418	4.425764	0.431878	0.103942	0.055336	1.090208	0.083406
min	88.333000	102.145000	65.476000	0.001680	0.000007	0.000680	0.000920	0.002040	0.009540	0.085000	...	0.013640	0.000650	8.441000	0.000000	0.256570	0.574282	-7.964984	0.006274
25%	117.572000	134.862500	84.291000	0.003460	0.000020	0.001660	0.001860	0.004985	0.016505	0.148500	...	0.024735	0.005925	19.198000	1.000000	0.421306	0.674758	-6.450096	0.174351
50%	148.790000	175.829000	104.315000	0.004940	0.000030	0.002500	0.002690	0.007490	0.022970	0.221000	...	0.038360	0.011660	22.085000	1.000000	0.495954	0.722254	-5.720868	0.218885
75%	182.769000	224.205500	140.018500	0.007365	0.000060	0.003835	0.003955	0.011505	0.037885	0.350000	...	0.060795	0.025640	25.075500	1.000000	0.587562	0.761881	-5.046192	0.279234
max	260.105000	592.030000	239.170000	0.033160	0.000260	0.021440	0.019580	0.064330	0.119080	1.302000	...	0.169420	0.314820	33.047000	1.000000	0.685151	0.825288	-2.434031	0.450493

8 rows × 23 columns

Distribution of target variable

- 1--> Not Healthy
- 0--> Healthy

In [ ]:
```
df['status'].value_counts()
```

Out[ ]:
```
1    147
0     48
Name: status, dtype: int64
```

Split the data into Training & Testing data

In [ ]:
```
x=df.drop(['name','status'],axis=1)
x
```

Out[ ]:

	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	MDVP:Shimmer(dB)	...	MDVP:APQ	Shimmer:DDA	NHR	HNR	RPDE	DFA	spread1	spread2	D2	PPE
0	119.992	157.302	74.997	0.00784	0.00007	0.00370	0.00554	0.01109	0.04374	0.426	...	0.02971	0.06545	0.02211	21.033	0.414783	0.815285	-4.813031	0.266482	2.301442	0.28465
1	122.400	148.650	113.819	0.00968	0.00008	0.00465	0.00696	0.01394	0.06134	0.626	...	0.04368	0.09403	0.01929	19.085	0.458359	0.819521	-4.075192	0.335590	2.486855	0.36867
2	116.682	131.111	111.555	0.01050	0.00009	0.00544	0.00781	0.01633	0.05233	0.482	...	0.03590	0.08270	0.01309	20.651	0.429895	0.825288	-4.443179	0.311173	2.342259	0.33263
3	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.01505	0.05492	0.517	...	0.03772	0.08771	0.01353	20.644	0.434969	0.819235	-4.117501	0.334147	2.405554	0.36897
4	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.01966	0.06425	0.584	...	0.04465	0.10470	0.01767	19.649	0.417356	0.823484	-3.747787	0.234513	2.332180	0.41033
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
190	174.188	230.978	94.261	0.00459	0.00003	0.00263	0.00259	0.00790	0.04087	0.405	...	0.02745	0.07008	0.02764	19.517	0.448439	0.657899	-6.538586	0.121952	2.657476	0.13305
191	209.516	253.017	89.488	0.00564	0.00003	0.00331	0.00292	0.00994	0.02751	0.263	...	0.01879	0.04812	0.01810	19.147	0.431674	0.683244	-6.195325	0.129303	2.784312	0.16889
192	174.688	240.005	74.287	0.01360	0.00008	0.00624	0.00564	0.01873	0.02308	0.256	...	0.01667	0.03804	0.10715	17.883	0.407567	0.655683	-6.787197	0.158453	2.679772	0.13172
193	198.764	396.961	74.904	0.00740	0.00004	0.00370	0.00390	0.01109	0.02296	0.241	...	0.01588	0.03794	0.07223	19.020	0.451221	0.643956	-6.744577	0.207454	2.138608	0.12330
194	214.289	260.277	77.973	0.00567	0.00003	0.00295	0.00317	0.00885	0.01884	0.190	...	0.01373	0.03078	0.04398	21.209	0.462803	0.664357	-5.724056	0.190667	2.555477	0.14856

195 rows × 22 columns

In [ ]:
```
y=df['status']
y
```

Out[ ]:
```
0      1
1      1
2      1
3      1
4      1
      ..
190    0
191    0
192    0
193    0
194    0
Name: status, Length: 195, dtype: int64
```

In [ ]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
x_train
```

Out[ ]:

	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	MDVP:Shimmer(dB)	...	MDVP:APQ	Shimmer:DDA	NHR	HNR	RPDE	DFA	spread1	spread2	D2	PPE
38	180.198	201.249	175.456	0.00284	0.00002	0.00153	0.00166	0.00459	0.01444	0.131	...	0.01190	0.02177	0.00231	26.738	0.403884	0.766209	-6.452058	0.212294	2.269398	0.141929
31	199.228	209.512	192.091	0.00241	0.00001	0.00134	0.00138	0.00402	0.01015	0.089	...	0.00762	0.01513	0.00167	30.940	0.432439	0.742055	-7.682587	0.173319	2.103106	0.06850
173	113.715	116.443	96.913	0.00349	0.00003	0.00171	0.00203	0.00514	0.01472	0.133	...	0.01148	0.02245	0.00478	26.547	0.380253	0.766700	-5.943501	0.192150	1.852542	0.17967
12	136.926	159.866	131.276	0.00293	0.00002	0.00118	0.00153	0.00355	0.01259	0.112	...	0.01140	0.01968	0.00581	25.703	0.460600	0.646846	-6.547148	0.152813	2.041277	0.13851
109	193.030	208.900	80.297	0.00766	0.00004	0.00450	0.00389	0.01351	0.03044	0.275	...	0.02084	0.05312	0.00947	21.934	0.497554	0.740539	-5.845099	0.278679	2.608749	0.18566
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
106	155.078	163.736	144.148	0.00168	0.00001	0.00068	0.00092	0.00204	0.01064	0.097	...	0.00928	0.01567	0.00233	29.746	0.334171	0.677930	-6.981201	0.184550	2.129924	0.10680
14	152.845	163.305	75.836	0.00294	0.00002	0.00121	0.00149	0.00364	0.01828	0.158	...	0.01246	0.03191	0.00609	24.922	0.474791	0.654027	-6.105098	0.203653	2.125618	0.17010
92	148.272	164.989	142.299	0.00459	0.00003	0.00250	0.00256	0.00750	0.04190	0.383	...	0.03051	0.07150	0.01914	18.780	0.454444	0.734504	-5.952058	0.087840	2.344336	0.18648
179	148.143	155.982	135.041	0.00392	0.00003	0.00204	0.00231	0.00612	0.01450	0.131	...	0.01263	0.02175	0.00540	23.683	0.398499	0.778349	-5.711205	0.240875	2.845109	0.19273
102	139.224	586.567	66.157	0.03011	0.00022	0.01854	0.01628	0.05563	0.09419	0.930	...	0.06023	0.16654	0.25930	10.489	0.596362	0.641418	-3.269487	0.270641	2.690917	0.44477

136 rows × 22 columns

In [ ]:
```
x_test
y_train
y_test
```

Out[ ]:
```
138    1
16     1
155    1
96     1
68     1
153    1
55     1
15     1
112    1
111    1
184    0
18     1
82     1
9      1
164    1
117    1
69     1
113    1
192    0
119    1
123    1
144    1
66     1
45     0
158    1
115    1
67     1
93     1
30     0
101    1
118    1
75     1
24     1
172    0
127    1
169    0
19     1
168    0
73     1
5      1
135    1
122    1
167    0
85     1
56     1
95     1
35     0
190    0
42     0
65     0
104    1
159    1
78     1
76     1
29     1
136    1
60     0
51     0
165    0
Name: status, dtype: int64
```

Data Normalization

In [ ]:
```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit(x_train)
x_train=scaler.transform(x_train)
x_test=scaler.transform(x_test)
```

Model Creation using

- KNN
- Naivebayes
- SVM

In [ ]:
```
from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
y_pred=knn.predict(x_test)
y_pred
```

Out[ ]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0])
```

In [ ]:
```
from sklearn.metrics import confusion_matrix,accuracy_score
matr=confusion_matrix(y_pred,y_test)
matr
score=accuracy_score(y_test,y_pred)
score
```

Out[ ]:
```
0.8983050847457628
```

GridSearchCV (find the optimal hyperparameters of a model) *imporve performance of model*

In [ ]:
```
knn1=KNeighborsClassifier()
```

In [ ]:
```
param={'n_neighbors':[3,5,7,9],'weights':['uniform','distance']}
clf=GridSearchCV(knn1,param,cv=10,scoring='accuracy')
clf.fit(x_train,y_train)
```

Out[ ]:

▸ **GridSearchCV**

▸ **estimator: KNeighborsClassifier**

▸ KNeighborsClassifier

In [ ]:
```
print(clf.best_params_)
```

```
{'n_neighbors': 3, 'weights': 'distance'}
```

In [ ]:
```
knn2=KNeighborsClassifier(n_neighbors=3,weights='distance')
knn2.fit(x_train,y_train)
y_pred1=knn2.predict(x_test)
y_pred1
```

Out[ ]:
```
array([1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1,
       1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0])
```

Performance Evaluation

In [ ]:
```
print('accuracy_score',accuracy_score(y_test,y_pred1))
```

```
accuracy_score 0.9152542372881356
```

- Naivebayes
- SVM

In [ ]:
```
from sklearn.naive_bayes import BernoulliNB
from sklearn.svm import SVC
model=BernoulliNB()
model3=SVC()
lst=[model,model3]
```

Performance Evaluation

In [ ]:
```
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
for i in lst:
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  print(i)
  print(confusion_matrix(y_test,y_pred))
  print("accuracy_score:",accuracy_score(y_test,y_pred))
  print("classification_report:",classification_report(y_test,y_pred))
```

```
BernoulliNB()
[[11  4]
 [ 7 37]]
accuracy_score: 0.8135593220338984
classification_report:               precision    recall  f1-score   support

           0       0.61      0.73      0.67        15
           1       0.90      0.84      0.87        44

    accuracy                           0.81        59
   macro avg       0.76      0.79      0.77        59
weighted avg       0.83      0.81      0.82        59

SVC()
[[ 8  7]
 [ 0 44]]
accuracy_score: 0.8813559322033898
classification_report:               precision    recall  f1-score   support

           0       1.00      0.53      0.70        15
           1       0.86      1.00      0.93        44

    accuracy                           0.88        59
   macro avg       0.93      0.77      0.81        59
weighted avg       0.90      0.88      0.87        59
```