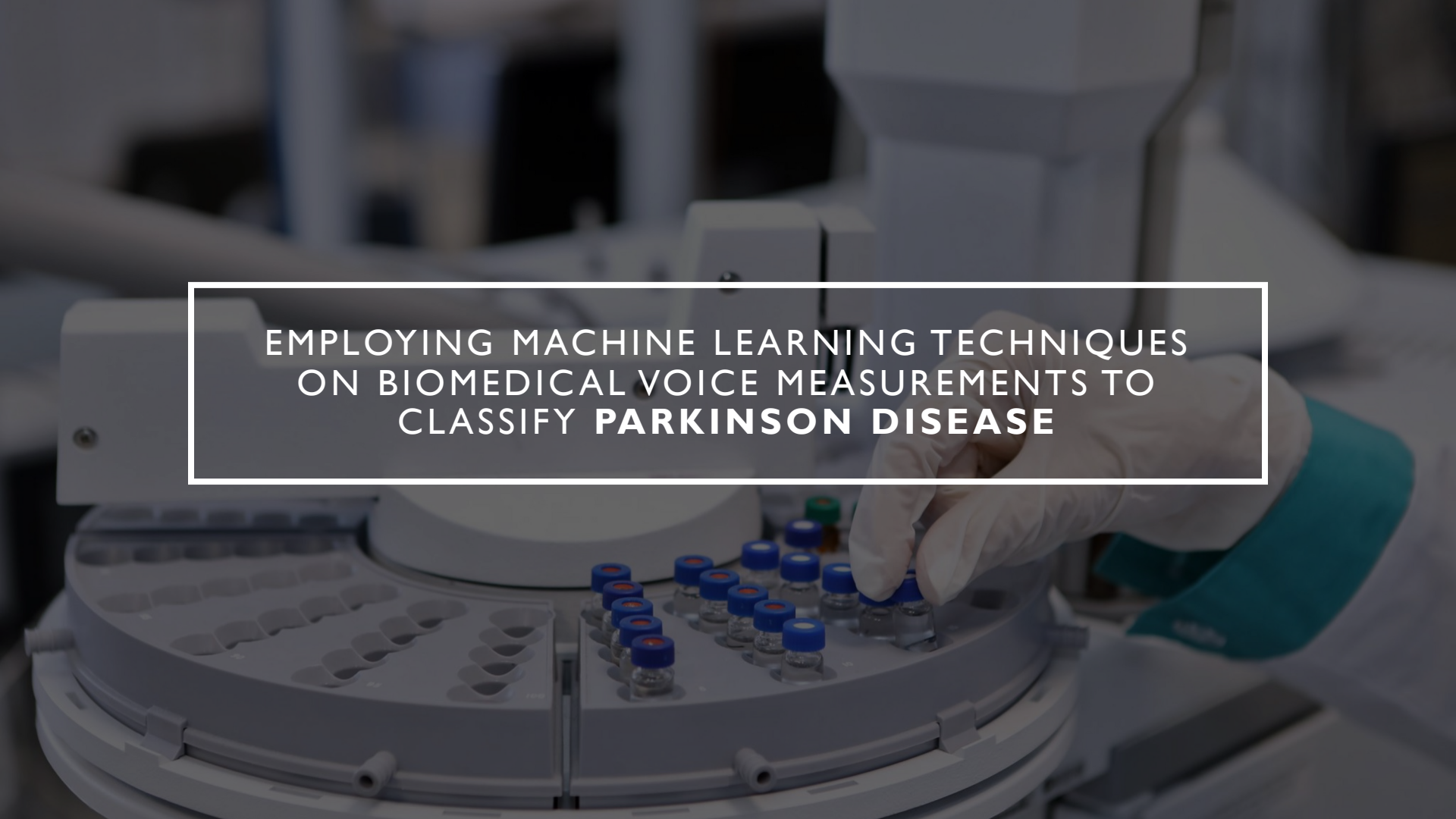


EMPLOYING MACHINE LEARNING TECHNIQUES ON BIOMEDICAL VOICE MEASUREMENTS TO CLASSIFY **PARKINSON DISEASE**

AGENDA

- Parkinson's Disease
- Exploratory Data Analysis
- Feature Engineering
- Classification Models
- Results

INTRODUCTION

Parkinson's

- degenerative brain disorder

Causes

- Neuron impairment and death in the areas of brain that control brain movement

Symptoms

- Tremors
- Stiffness
- Difficulty maintaining balance and coordination
- Speech changes
  - Dysphonia - impairment in productions of vocal sounds

Dataset

Parkinsons Dataset

Dataset Characteristics: Multivariate

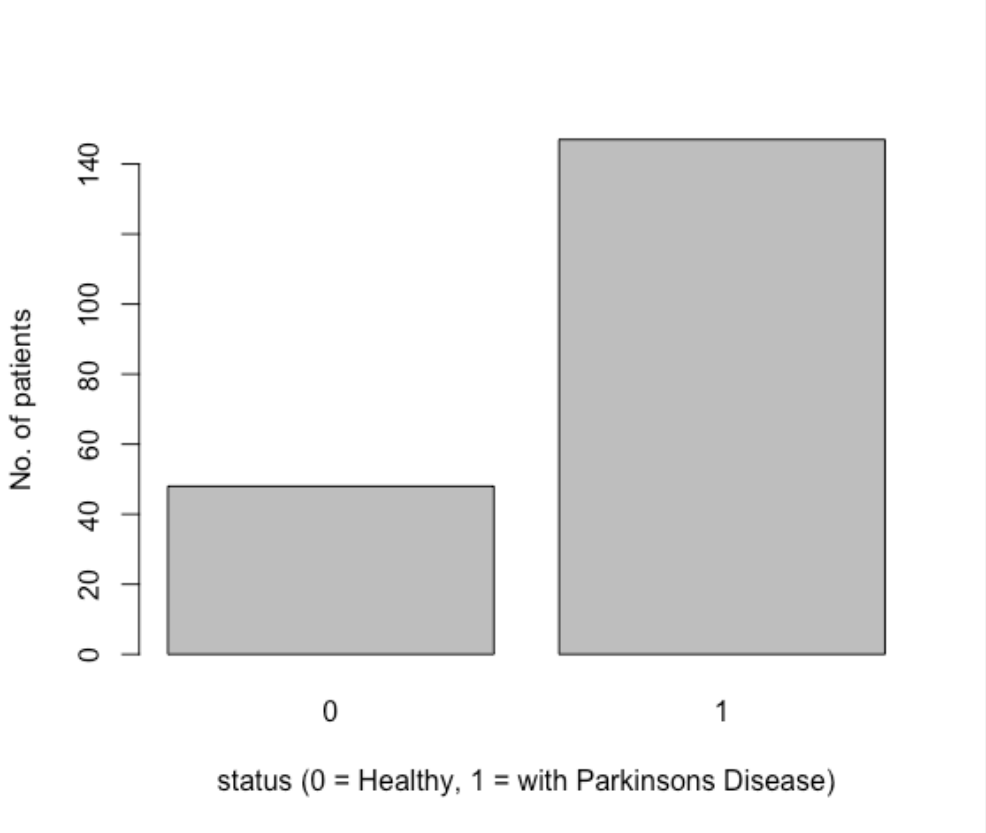
Number of instances: 195

Number of attributes: 24

Goal: Classify Parkinson Disease - Binary (0/1) -> (Healthy people/ People with Parkinson Disease)

The data was collected from 31 people, 23 were detected with Parkinson's.

195 voice recordings were converted into various measures of vocal signals.



EXPLORATORY DATA ANALYSIS

- There are no null values in the Parkinson's Dataset
- All the record inputs are unique
- Among 195 total entries, there are 48 healthy people and 147 patients that were detected with Parkinson's.

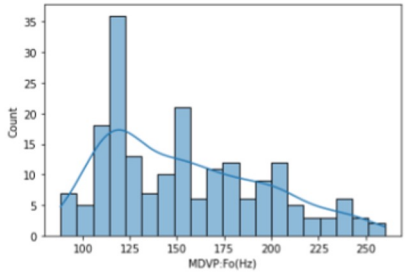
Descriptive Analysis

```
df.describe().transpose()
```

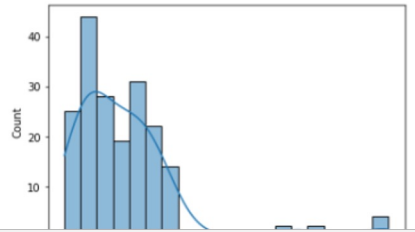
	count	mean	std	min	25%	50%	75%	max
MDVP:Fo(Hz)	195.0	154.228641	41.390065	88.333000	117.572000	148.790000	182.769000	260.105000
MDVP:Fhi(Hz)	195.0	197.104918	91.491548	102.145000	134.862500	175.829000	224.205500	592.030000
MDVP:Flo(Hz)	195.0	116.324631	43.521413	65.476000	84.291000	104.315000	140.018500	239.170000
MDVP:Jitter(%)	195.0	0.006220	0.004848	0.001680	0.003460	0.004940	0.007365	0.033160
MDVP:Jitter(Abs)	195.0	0.000044	0.000035	0.000007	0.000020	0.000030	0.000060	0.000260
MDVP:RAP	195.0	0.003306	0.002968	0.000680	0.001660	0.002500	0.003835	0.021440
MDVP:PPQ	195.0	0.003446	0.002759	0.000920	0.001860	0.002690	0.003955	0.019580
Jitter:DDP	195.0	0.009920	0.008903	0.002040	0.004985	0.007490	0.011505	0.064330
MDVP:Shimmer	195.0	0.029709	0.018857	0.009540	0.016505	0.022970	0.037885	0.119080
MDVP:Shimmer(dB)	195.0	0.282251	0.194877	0.085000	0.148500	0.221000	0.350000	1.302000
Shimmer:APQ3	195.0	0.015664	0.010153	0.004550	0.008245	0.012790	0.020265	0.056470
Shimmer:APQ5	195.0	0.017878	0.012024	0.005700	0.009580	0.013470	0.022380	0.079400
MDVP:APQ	195.0	0.024081	0.016947	0.007190	0.013080	0.018260	0.029400	0.137780
Shimmer:DDA	195.0	0.046993	0.030459	0.013640	0.024735	0.038360	0.060795	0.169420
NHR	195.0	0.024847	0.040418	0.000650	0.005925	0.011660	0.025640	0.314820
HNR	195.0	21.885974	4.425764	8.441000	19.198000	22.085000	25.075500	33.047000
status	195.0	0.753846	0.431878	0.000000	1.000000	1.000000	1.000000	1.000000
RPDE	195.0	0.498536	0.103942	0.256570	0.421306	0.495954	0.587562	0.685151
DFA	195.0	0.718099	0.055336	0.574282	0.674758	0.722254	0.761881	0.825288
spread1	195.0	-5.684397	1.090208	-7.964984	-6.450096	-5.720868	-5.046192	-2.434031
spread2	195.0	0.226510	0.083406	0.006274	0.174351	0.218885	0.279234	0.450493
D2	195.0	2.381826	0.382799	1.423287	2.099125	2.361532	2.636456	3.671155
PPE	195.0	0.206552	0.090119	0.044539	0.137451	0.194052	0.252980	0.527367

Distributions

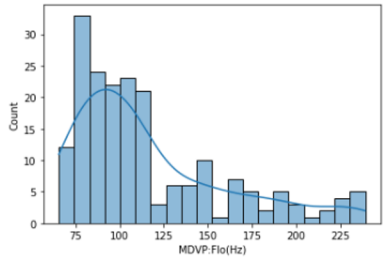
```
sns.histplot(data = df, x = 'MDVP:Fo(Hz)', bins=20, kde=True, edgecolor='black')
plt.show()
```



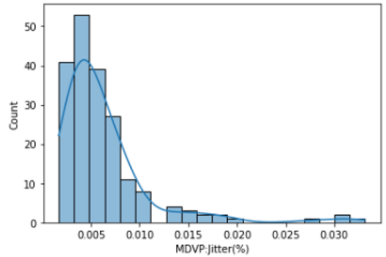

```
[10] sns.histplot(data = df, x = 'MDVP:Fhi(Hz)', bins=20, kde=True, edgecolor='black')
     plt.show()
```



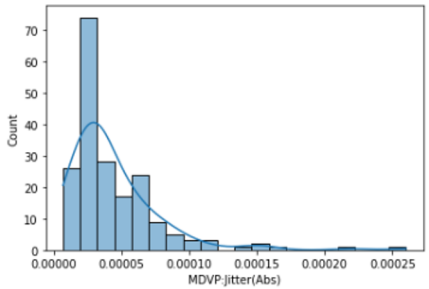

```
sns.histplot(data = df, x = 'MDVP:Flo(Hz)', bins=20, kde=True, edgecolor='black')
plt.show()
```



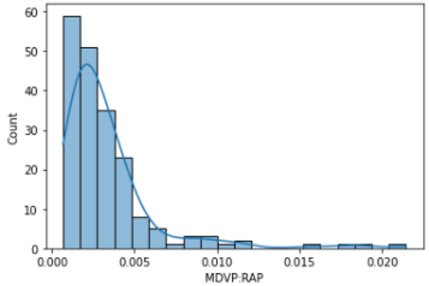

```
[ ] sns.histplot(data = df, x = 'MDVP:Jitter(%)', bins=20, kde=True, edgecolor='black')
    plt.show()
```



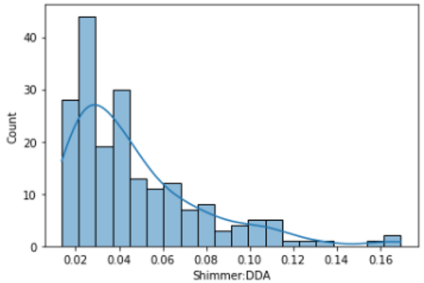
```
sns.histplot(data = df, x = 'MDVP:Jitter(Abs)', bins=20, kde=True, edgecolor='black')
plt.show()
```



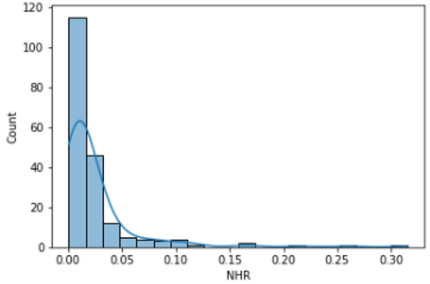

```
[ ] sns.histplot(data = df, x = 'MDVP:RAP', bins=20, kde=True, edgecolor='black')
    plt.show()
```




```
sns.histplot(data = df, x = 'Shimmer:DDA', bins=20, kde=True, edgecolor='black')
plt.show()
```

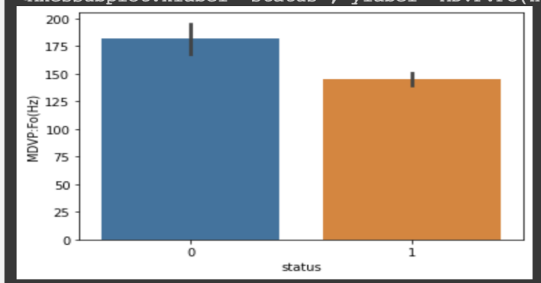



```
[ ] sns.histplot(data = df, x = 'NHR', bins=20, kde=True, edgecolor='black')
    plt.show()
```



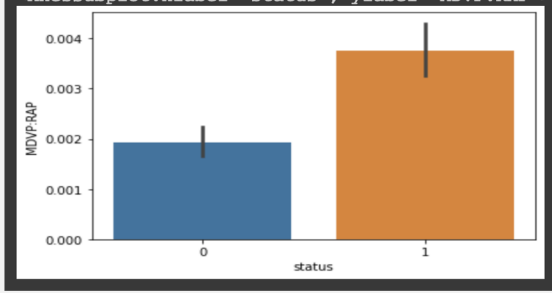
Bivariate Analysis

```
<AxesSubplot:xlabel='status', ylabel='MDVP:Fo(Hz)'>
```



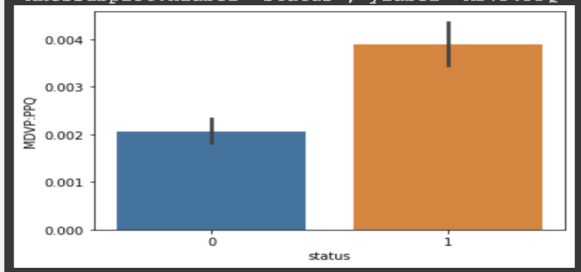

```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="MDVP:RAP", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='MDVP:RAP'>
```



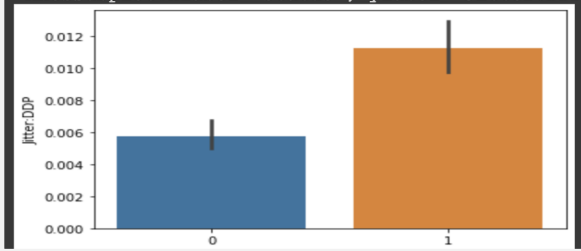

```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="MDVP:PPQ", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='MDVP:PPQ'>
```



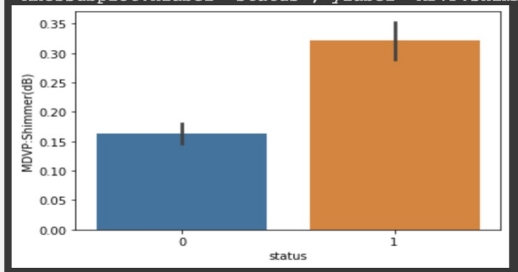

```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="Jitter:DDP", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='Jitter:DDP'>
```



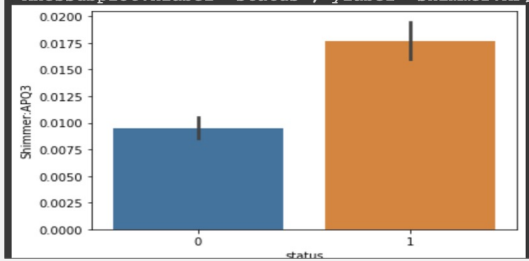

```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="MDVP:Shimmer(dB)", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='MDVP:Shimmer(dB)'>
```




```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="Shimmer:APQ3", data = df)
```

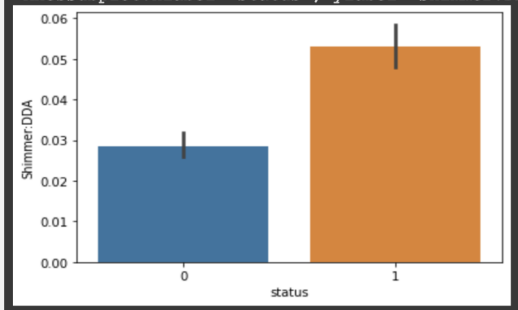
```
<AxesSubplot:xlabel='status', ylabel='Shimmer:APQ3'>
```



Bivariate Analysis

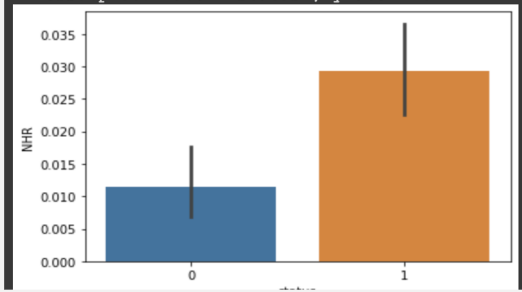
```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="Shimmer:DDA", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='Shimmer:DDA'>
```



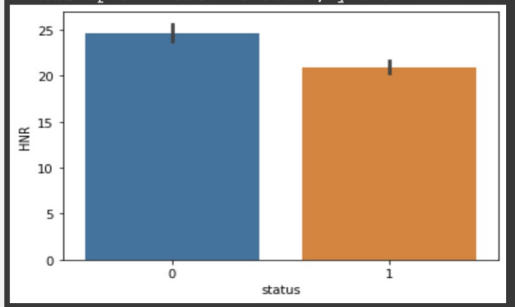

```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="NHR", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='NHR'>
```



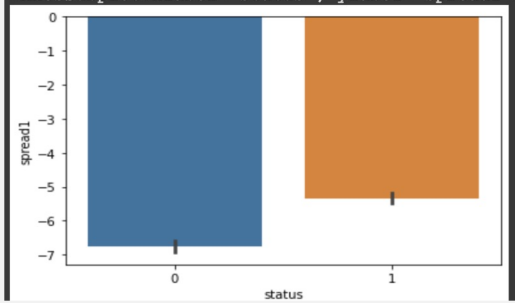

```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="HNR", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='HNR'>
```



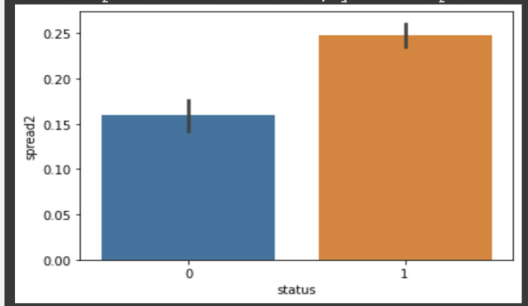

```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="spread1", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='spread1'>
```



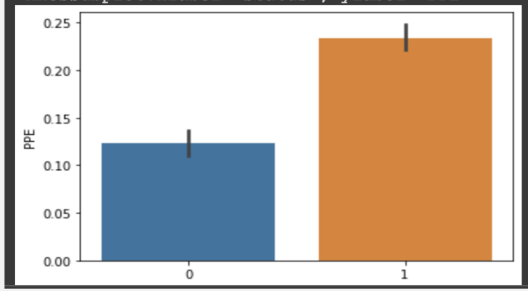

```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="spread2", data = df)
```

```
<AxesSubplot:xlabel='status', ylabel='spread2'>
```




```
plt.figure(figsize=(6,4))
sns.barplot(x="status", y="PPE", data = df)
```

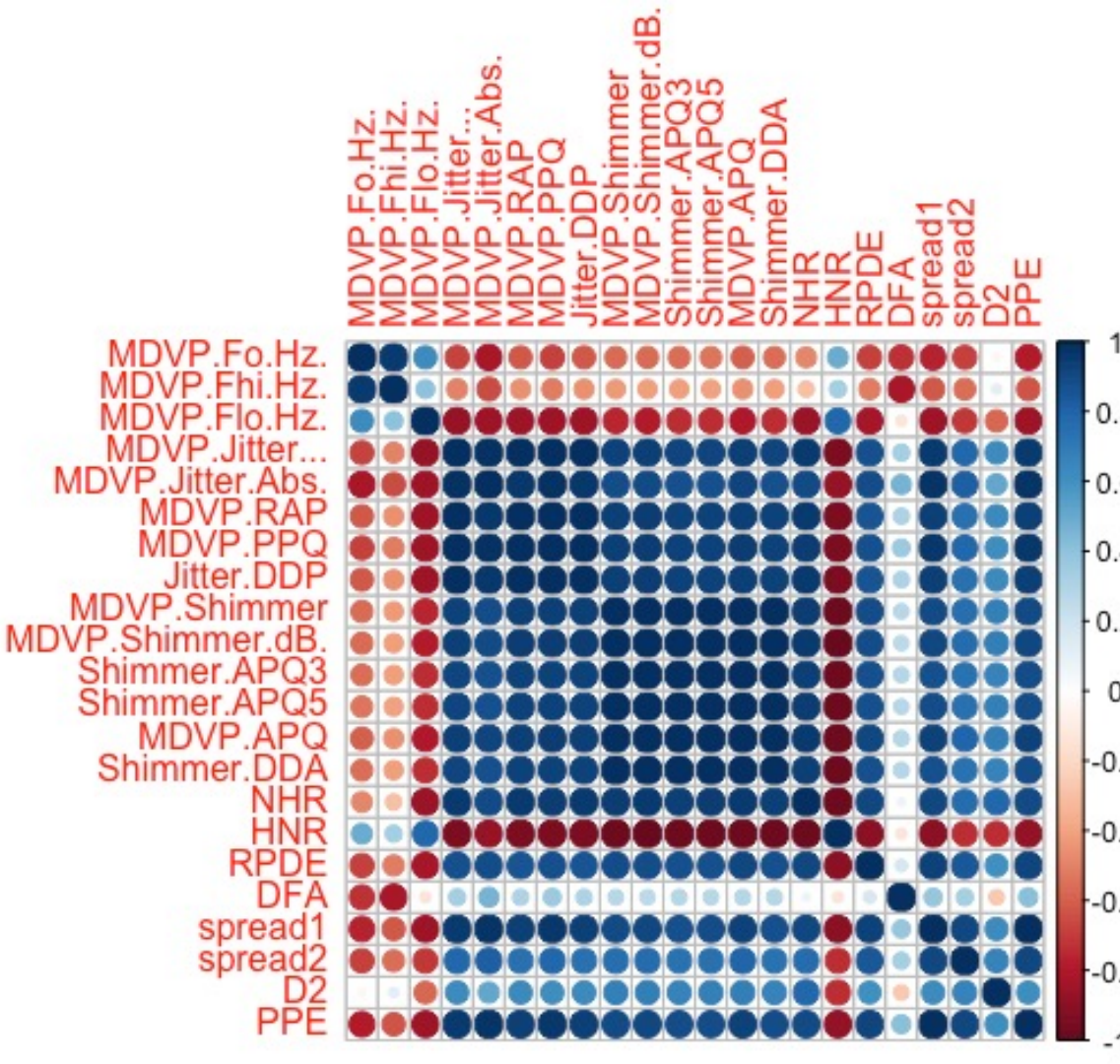
```
<AxesSubplot:xlabel='status', ylabel='PPE'>
```



EXPLORATORY DATA ANALYSIS

- Before finding the correlation between numeric attributes, remove the column ‘name’ and ‘status’ (categorical)
- Using Spearman correlation coefficient to create correlation matrix
- With a cut-off of 0.9, the highly correlated features are:

• PPE, spread1, MDVP:PPQ, MDVP:Jitter…, MDVP:Jitter(Abs), MDVP:RAP, MDVP:APQ, Jitter:DDP, MDVP:Shimmer, MDVP:Shimmer(dB), Shimmer:APQ3, Shimmer:APQ5, Shimmer:DDA, HNR, MDVP:Fo(Hz)



FEATURE SELECTION

```
[ ] feature_names=["MDVP:Fo(Hz)","MDVP:Fhi(Hz)", "MDVP:Flo(Hz)","MDVP:Jitter(Abs)","MDVP:PPQ",
        "Jitter:DDP","MDVP:Shimmer(dB)","MDVP:APQ",'Shimmer:DDA', 'NHR', 'HNR','RPDE', 'DFA','spread2', 'D2', 'PPE']
```

MODEL 1 - LOGISTIC REGRESSION

- L

	feature	coefficient	odds_ratio
0	MDVP:Fo(Hz)	-0.487664	0.614059
1	MDVP:Fhi(Hz)	-0.371685	0.689571
2	MDVP:Flo(Hz)	-0.155117	0.856315
3	MDVP:Jitter(Abs)	-0.554254	0.574501
4	MDVP:PPQ	0.161669	1.175471
5	Jitter:DDP	0.628325	1.874468
6	MDVP:Shimmer(dB)	0.164918	1.179297
7	MDVP:APQ	0.734422	2.084278
8	Shimmer:DDA	-0.291246	0.747332
9	NHR	-0.180813	0.834592
10	HNR	-0.017442	0.982709
11	RPDE	-0.420846	0.656491
12	DFA	0.108283	1.114363
13	spread2	0.436955	1.547986
14	D2	0.906703	2.476146
15	PPE	1.704927	5.500986

```
lr = LogisticRegression()
lr.fit(X_train, y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
coef = lr.coef_[0]
intercept = lr.intercept_
```

```
feature_names=["MDVP:Fo(Hz)","MDVP:Fhi(Hz)", "MDVP:Flo(Hz)","MDVP:Jitter(Abs)","MDVP:PPQ",
      "Jitter:DDP","MDVP:Shimmer(dB)","MDVP:APQ",'Shimmer:DDA', 'NHR', 'HNR','RPDE', 'DFA','spread2', 'D2', 'PPE']
```

```
lr_acc = accuracy_score(y_test, lr.predict(X_test))
print(f"Accuracy Score is {lr_acc}")
```

```
Accuracy Score is 0.8974358974358975
```

MODEL 2 – DECISION TREE

```
dtc = DecisionTreeClassifier()
dtc.fit(X_train, y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
dtc_acc = accuracy_score(y_test, dtc.predict(X_test))
print(f"Accuracy Score is {dtc_acc}")
```

```
Accuracy Score is 0.8974358974358975
```

MODEL 3 - RANDOM FOREST

```
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
rf_acc = accuracy_score(y_test, rf.predict(X_test))
print(f"Accuracy Score is {rf_acc}")
```

```
Accuracy Score is 0.9487179487179487
```

MODEL 4 - GRADIENT BOOSTING

```
gb = GradientBoostingClassifier()
gb.fit(X_train, y_train)
```

```
▾ GradientBoostingClassifier
GradientBoostingClassifier()
```

```
gb_acc = accuracy_score(y_test, gb.predict(X_test))
print(f"Accuracy Score is {gb_acc}")
```

```
Accuracy Score is 0.9487179487179487
```

MODEL 5 - SVM

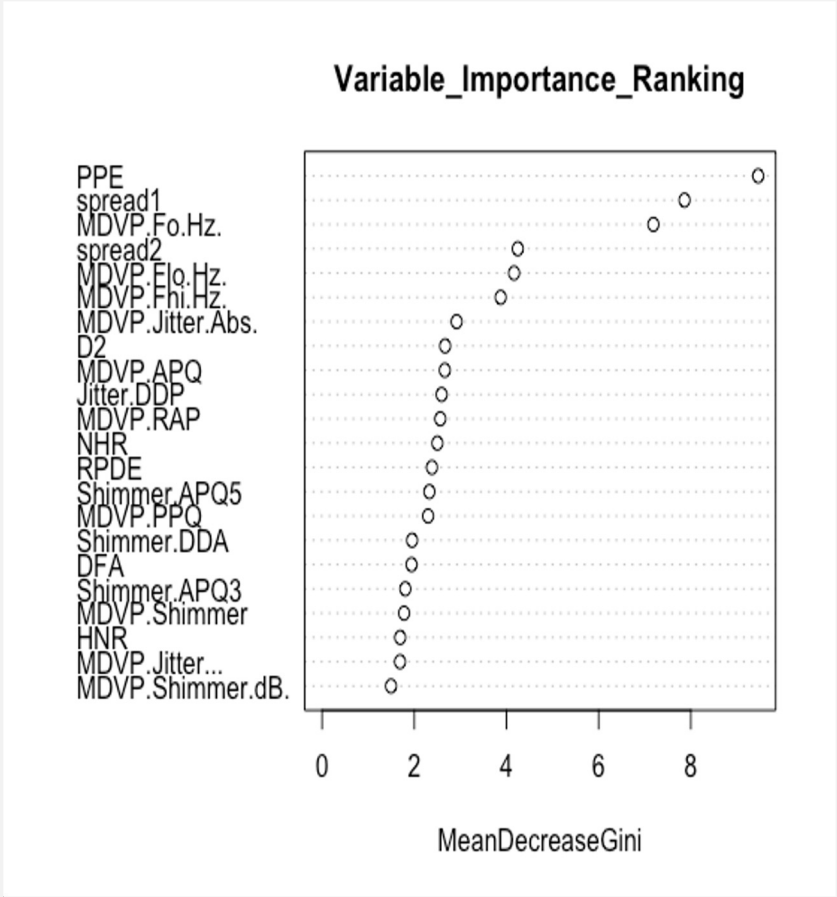
```
svc=SVC()
svc.fit(X_train, y_train)
y_pred=svc.predict(X_test)
print('Accuracy Score is {0:0.3f}'. format(accuracy_score(y_test, y_pred)))
```

```
Accuracy Score is 0.923
```

Importance Features: Variable Importance

- Random Forest can be used to rank the importance of variables in a classification problem.
- Interpretation: MeanDecreaseAccuracy Table represents how much removing each variable reduces the accuracy of the model.
- “varImp” function from “RandomForest” package -> Feature Model

```
> importance <- varImp(Variable_Importance_Ranking)
> #summarize importance
> print(importance)
                  Overall
MDVP.Fo.Hz.      7.187774
MDVP.Fhi.Hz.     3.874204
MDVP.Flo.Hz.     4.164955
MDVP.Jitter...   1.690959
MDVP.Jitter.Abs. 2.918349
MDVP.RAP         2.564392
MDVP.PPQ         2.299899
Jitter.DDP       2.593390
MDVP.Shimmer     1.779224
MDVP.Shimmer.dB. 1.495281
Shimmer.APQ3     1.805764
Shimmer.APQ5     2.327622
MDVP.APQ         2.662474
Shimmer.DDA      1.952547
NHR              2.501597
HNR              1.694664
RPDE             2.383447
DFA              1.941248
spread1          7.862932
spread2          4.245931
D2               2.669413
PPE              9.461802
```



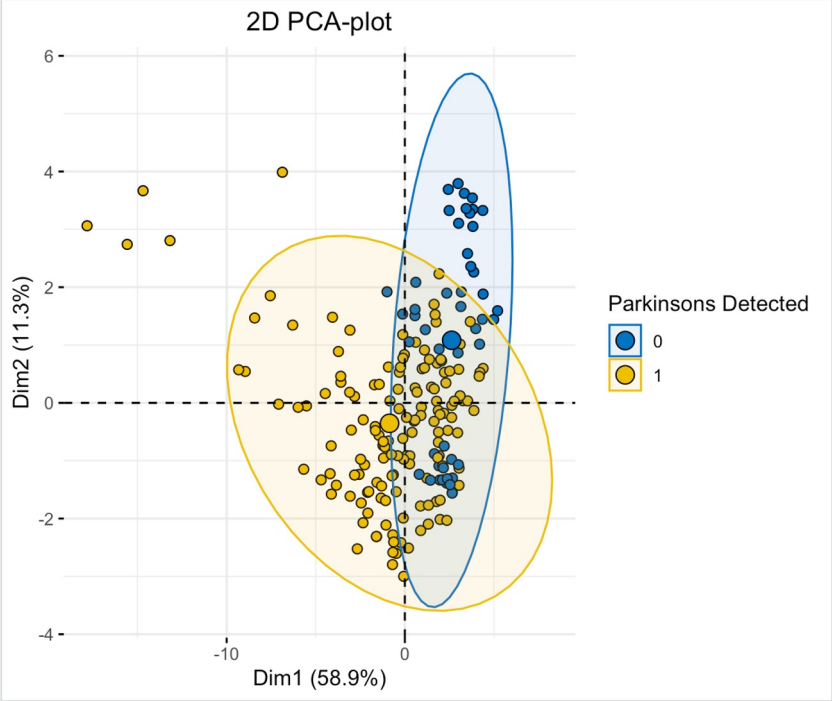
Alternative Approach: Perform PCA in R

FEATURE ENGINEERING

- Principal Component Analysis
  - Ensured the data is centered and scaled
  - 22 principal components have been generated (PC1 to PC22), which also correspond to the number of variables in the data.
  - Cumulative Proportion

```
> #Doing Principle Component Analysis on the Dataset
> pd.pca <- prcomp(pd_data1, center = TRUE, scale = TRUE)
> summary(pd.pca)
Importance of components:
                          PC1   PC2     PC3     PC4     PC5     PC6    PC7     PC8     PC9    PC10    PC11
Standard deviation     3.600 1.577 1.24178 1.21037 0.98687 0.85388 0.7431 0.60200 0.53837 0.47342 0.37492
Proportion of Variance 0.589 0.113 0.07009 0.06659 0.04427 0.03314 0.0251 0.01647 0.01317 0.01019 0.00639
Cumulative Proportion  0.589 0.702 0.77209 0.83868 0.88295 0.91609 0.9412 0.95767 0.97084 0.98103 0.98742
                          PC12    PC13    PC14   PC15    PC16    PC17    PC18    PC19    PC20      PC21
Standard deviation     0.32379 0.26408 0.19536 0.1484 0.13337 0.11161 0.08494 0.05913 0.03294 0.0006015
Proportion of Variance 0.00477 0.00317 0.00173 0.0010 0.00081 0.00057 0.00033 0.00016 0.00005 0.0000000
Cumulative Proportion  0.99218 0.99535 0.99709 0.9981 0.99890 0.99946 0.99979 0.99995 1.00000 1.0000000
                          PC22
Standard deviation     0.000182
Proportion of Variance 0.000000
Cumulative Proportion  1.000000
```

- PCA Variable Coordinates
  - PC1 - captures most variation
  - PC2 - captures second most




```
> # Results for Variables
> pd.pca.var <- get_pca_var(pd.pca)
> pd.pca.var$coord          # Coordinates
                      Dim.1       Dim.2
MDVP.Fo.Hz.       0.19197801  0.87252840 -
MDVP.Fhi.Hz.     -0.02416324  0.54991186 -
MDVP.Flo.Hz.      0.22973318  0.62354017
MDVP.Jitter...   -0.91618089  0.12895436
MDVP.Jitter.Abs. -0.86998709 -0.12104464
MDVP.RAP         -0.89929931  0.18295301
MDVP.PPQ         -0.92343921  0.10737349
Jitter.DDP       -0.89928246  0.18296500
MDVP.Shimmer     -0.93593176  0.08286388
MDVP.Shimmer.dB. -0.94230805  0.12025120
Shimmer.APQ3     -0.91207818  0.09288902
Shimmer.APQ5     -0.90849483  0.07896712
MDVP.APQ         -0.91303172  0.07530644 -
Shimmer.DDA      -0.91208336  0.09289047
NHR              -0.84200637  0.26888117
HNR               0.87134487 -0.05844654
RPDE             -0.52819116 -0.39195092 -
DFA              -0.14763734 -0.48999985
spread1          -0.80528796 -0.37731187 -
spread2          -0.54262903 -0.32198314 -
D2               -0.55766929  0.20939913 -
PPE              -0.83176114 -0.33732894 -
```

Biplot - Contribution of each variable