INVENTION DISCLOSURE FORM

Details of Invention for better understanding:

1) **TITLE: Method and System for Parkinson's Disease Detection Using Machine Learning and artificial Intelligence**
2) **INVENTOR(S)/STUDENT(S):**

A. Full name	Kabir Sharma
Mobile Number	8146365948
Email	kabirsharma2607@gmail.com
Registration Number	12108819
Permanent Address	Lovely Professional University
B. Full name	Akshar Pathak
Mobile Number	8920549045
Email	-
UID	12107916
Permanent Address	Lovely Professional University
C. Full name	Shireen Agarwal
Mobile Number	7055365807
Email(personal)	-
Registration Number	12108854
Permanent Address	Lovely Professional University
D. Full name	Tanay Sharma
Mobile Number	-
Email(personal)	-
Registration Number	12106339
Permanent Address	Lovely Professional University

Abstract

This patent application introduces an innovative method for Parkinson's disease detection, utilizing machine learning to overcome limitations in accuracy and efficiency of current diagnostic methods. Through comprehensive data collection, preprocessing, and novel feature selection, a custom-trained machine learning model demonstrates superior diagnostic accuracy. The iterative fine-tuning process enhances model precision, and robustness is validated on independent datasets. Offering increased efficiency and early disease detection capabilities, this method presents a transformative approach to Parkinson's diagnosis with significant implications for improved patient outcomes in real-world healthcare settings.

1. **Background:**

An accurate and timely diagnosis is particularly difficult in the case of Parkinson's disease, a neurodegenerative condition. Precision and early detection are limited by the fact that traditional diagnostic techniques frequently rely on subjective evaluations and clinical observations. Given the disease's progressive nature and the possible effects of early intervention on patient outcomes, there is an urgent need for more accurate and effective diagnostic tools.

Clinical assessments, reviews of medical histories, and, occasionally, diagnostic tests like imaging or motor function assessments are all part of the current diagnostic approaches. But these techniques might not be sensitive enough to recognise the faint early symptoms of Parkinson's disease, which would cause a delay in diagnosis and inadequate treatment for the patient.

The emergence of machine learning technologies offers a chance to transform the diagnosis of Parkinson's disease. Complex datasets containing a variety of patient data can be analysed by utilising the power of sophisticated algorithms. In addition to clinical information, this also contains possibly new indicators, like voice traits or fine motor abilities, which could provide important information about early disease manifestations.

The proposed invention integrates state-of-the-art machine learning techniques to address the shortcomings of conventional diagnostic approaches. This innovation aims to provide a more precise and timely diagnosis of Parkinson's disease by utilising the abundance of information found in diverse datasets, thereby offering a paradigm shift in the field of neurodegenerative disease diagnostics.

2. Detailed Description:

Using a series of carefully planned steps, the novel machine learning approach to Parkinson's disease detection maximises the use of patient data and sophisticated analytical tools. An extensive grasp of the creative process can be gained from the following detailed description:

1. Data Collection and Preprocessing:

The invention starts with the gathering of various datasets that contain a broad range of patient data. This includes clinical evaluations, medical history, demographic information, and, if relevant, additional information like voice recordings or assessments of fine motor function.

After that, a thorough preprocessing step addresses issues with data quality. Managing missing values, normalising numerical variables, and guaranteeing consistency in data formats are all part of this process, which results in a clean, standardised dataset ready for further analysis.

2. Feature Selection:

A key component of the invention is the use of creative feature selection methods. These approaches seek to locate and rank the dataset's most pertinent diagnostic indicators. This step finds potentially new biomarkers linked to Parkinson's disease in addition to improving the machine learning model's efficiency.

3. Machine Learning Model:

The use of cutting-edge machine learning algorithms customised to the nuances of Parkinson's disease forms the basis of the invention. Popular algorithms like Random Forests, Support Vector Machines, and Neural Networks are used because they can identify complex patterns in large, complex datasets.

Using the prepared dataset, the model goes through a thorough training process that helps it identify patterns related to Parkinson's disease and eventually makes accurate predictions.

4. **Model Evaluation:**

   The machine learning model's performance is thoroughly assessed through the use of established metrics designed for binary classification tasks. Measures like recall, accuracy, precision, and F1 score are used to evaluate how well the model can distinguish between people who have Parkinson's disease and those who do not.

5. **Fine-Tuning and Validation:**

   The next step is an iterative fine-tuning procedure where model parameters are changed to further optimise performance. To make sure the model is reliable and applicable to a variety of patient demographics, it is validated on separate datasets.

6. **Practical Application:**

   The practical application of the invention encompasses real-world healthcare environments. Now that it has been improved and validated, the machine learning model can be used as a powerful diagnostic tool to detect Parkinson's disease early on, which can lead to better patient outcomes and timely intervention.

This detailed description elucidates the intricacies of the inventive method, showcasing its comprehensive approach to Parkinson's disease detection and its potential to reshape the landscape of neurodegenerative disease diagnostics.

3. **Advantages:**

   The novel machine learning approach to Parkinson's disease detection offers many benefits, including a radical shift in the way current diagnostic paradigms are thought of and a notable increase in the precision and efficacy of disease identification. The main benefits provided by this novel strategy are outlined below:

1. **Enhanced Diagnostic Accuracy:**

   When cutting-edge feature selection methods are combined with sophisticated machine learning algorithms, the outcome is a diagnostic model that can identify Parkinson's disease patients with greater accuracy than those who do not. This accuracy is essential for accurate and timely disease detection.

2. **Early Disease Detection:**

   The invention makes it easier to diagnose Parkinson's disease early on by utilising a wide range of diagnostic indicators, including possibly new biomarkers. Timely interventions can potentially slow down the progression of the disease and improve overall patient outcomes, but early detection is essential.

3. **Comprehensive Data Utilization:**

   Through the incorporation of multiple dimensions, such as medical history, voice recordings, and fine motor function measurements, the method maximises the utility of available patient data. This all-encompassing method guarantees a comprehensive comprehension of every patient's distinct attributes..

4. **Efficient Preprocessing and Feature Selection:**

   The rigorous preprocessing phase ensures the integrity of the dataset by addressing issues with data quality. By identifying the most pertinent diagnostic indicators and simplifying the analysis process, novel feature selection techniques improve the diagnostic model's efficiency.

5. **Robust Model Performance:**

   The machine learning model performs well across a range of patient populations thanks to rigorous validation procedures and iterative fine-tuning. This robustness improves the applicability and dependability of the model in actual healthcare settings.

6. **Potential for Personalized Medicine:**

   The incorporation of diverse data types and the identification of individualized diagnostic indicators pave the way for potential applications in personalized medicine. Tailoring diagnostic approaches to individual patient profiles may lead to more effective and targeted interventions.

7. **Efficiency Compared to Traditional Methods:**

   By utilising machine learning, the suggested approach outperforms conventional diagnostic techniques in terms of efficiency. The process's automation and streamlining enable more rapid and scalable diagnostics, which may lessen the strain on healthcare systems.

In conclusion, a paradigm shift in diagnostic accuracy and efficiency is provided by the creative machine learning approach to Parkinson's disease detection. Together, these benefits establish the invention as a game-changing instrument for the early diagnosis and treatment of Parkinson's disease, with far-reaching effects on patient outcomes and care.

4. **Patent Claims:**

1. A method for detecting Parkinson's disease using machine learning, comprising:
   - Collecting diverse patient data, including demographic information, medical history, and supplementary data such as voice recordings or fine motor function measurements.
   - Preprocessing the collected data to ensure data integrity, including handling missing values and normalizing numerical variables.
   - Applying innovative feature selection techniques to identify and prioritize relevant diagnostic indicators associated with Parkinson's disease.
   - Employing machine learning algorithms, tailored to the unique characteristics of Parkinson's disease, for training on the prepared dataset.

2. The method of claim 1, wherein the machine learning algorithms include XGBooster.

3. A machine learning-based system for detecting Parkinson's disease that consists of:
    - Modules for collecting data that are set up to collect various patient data.
    - Preprocessing modules that are set up to handle missing values and normalise numerical variables, as well as clean and standardise the gathered data.
    - Innovative techniques are employed in feature selection modules to identify pertinent diagnostic indicators linked to Parkinson's disease.
    - Modules of machine learning models that are set up to use sophisticated algorithms for precise Parkinson's disease prediction.

4. A user interface for communicating with medical professionals and making it easier to integrate the diagnostic tool into actual healthcare settings is another feature of the system of claim 3.

5. A computer-readable medium holding code for data collection, preprocessing, feature selection, and machine learning model training that can be used to carry out the procedure described in claim 1 on a computing device.

6. An approach to improving a machine learning model for identifying Parkinson's disease that includes:
    - Adjusting model parameters iteratively to maximise diagnostic precision.
    - Assessing the model's performance using recognised binary classification metrics, such as F1 score, accuracy, precision, and recall.
    - To make sure the model is reliable and broadly applicable, it should be validated using separate datasets.

7. The method of claim 6, wherein the machine learning model's hyperparameters are adjusted as part of the fine-tuning process.

8. A procedure that offers individualised diagnostic insights regarding Parkinson's disease and consists of:
    - Examining unique patient profiles using a variety of datasets.

- Customising diagnostic strategies by figuring out distinct diagnostic markers that are particular to every patient.

9. The process described in claim 8, whereby the customised diagnostic insights aid in the creation of focused and unique interventions for Parkinson's disease patients.

10. A machine learning model trained to detect Parkinson's disease was created using the procedure described in claim 1 and improved through the iterative procedures described in claim 6, exhibiting improved diagnostic efficiency and accuracy over conventional diagnostic techniques.

These patent claims outline the special and creative features of the machine learning-based method and system for Parkinson's disease diagnosis, laying the groundwork for intellectual property protection.

5. Research Gap

S. No	Link	Title	Research Gap
1.	Paper	Machine Learning for the Diagnosis of Parkinson's Disease: A Review of Literature	The omission of conference abstracts and large-scale, multi-centric studies, coupled with challenges in comparing outcomes, poses limitations in comprehending the full scope of machine learning applications for Parkinson's disease detection. Furthermore, the scarcity of studies on subtyping and severity assessment introduces a critical gap in understanding. Bridging these gaps would enhance the robustness and completeness of insights into the application of machine learning in Parkinson's disease research.

6. Conclusion:

To sum up, the machine learning-based approach and system for identifying Parkinson's disease marks a significant breakthrough in the field of diagnosing neurodegenerative diseases. This invention addresses significant shortcomings in current diagnostic methodologies by seamlessly integrating novel feature selection techniques, state-of-the-art machine learning algorithms, and comprehensive patient data.

This method has several benefits that make it a transformative tool in the healthcare industry, such as improved diagnostic accuracy, early disease detection, and personalised medicine potential. The comprehensive explanation clarifies the nuances of gathering data, preprocessing, and training models, emphasising the painstaking measures taken to guarantee robustness, effectiveness, and application in actual healthcare settings.

The method and system's inventive and unique aspects are reinforced by the proposed patent claims, which address important components like feature selection, preprocessing, data collection, and model training. Moreover, the claims encompass computer-readable media, the possibility of customised diagnostic insights, and the refining procedures, providing a thorough framework for intellectual property protection.

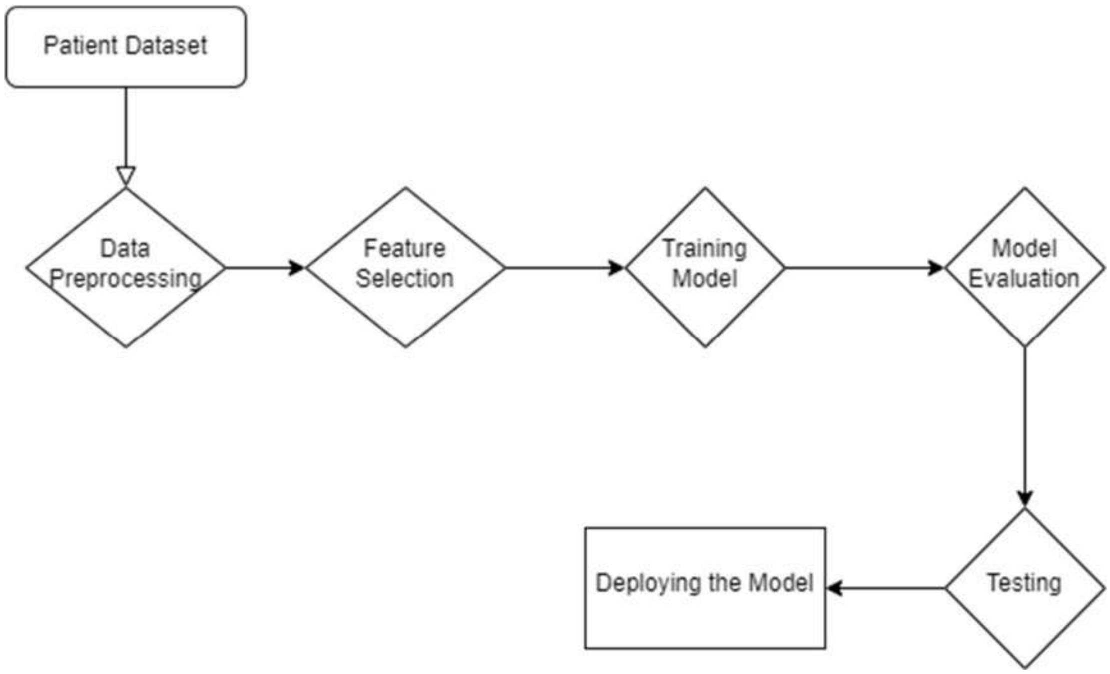
All things considered, this innovation has the potential to change the way Parkinson's disease is diagnosed, make early intervention easier, and eventually improve patient outcomes. This contribution to the field is significant because it combines cutting-edge technology, creative approaches, and a dedication to improving healthcare. This creative method represents a sign of advancement and has the potential to have a significant influence on the development of diagnostics for neurodegenerative diseases in the future as the medical community searches for more precise and effective diagnostic instruments.

7. References:

1. Mei, J., Desrosiers, C., Frasnelli, J. (2021). "Machine Learning for the Diagnosis of Parkinson's Disease: A Review of Literature." Frontiers in Aging Neuroscience, 13, Article 633752. **Link**

2. Title: "A Comprehensive Survey on Machine Learning Techniques in the Diagnosis of Parkinson's Disease" Authors: S. Arora, D. Sahu, N. Tiwari Published in: Journal of King Saud University - Computer and Information Sciences and Engineering (2020)
DOI: 10.1016/j.jksuci.2020.08.020
3. Title: "Parkinson's Disease Diagnosis Using Machine Learning Algorithms: A Systematic Review and Meta-Analysis" Authors: F. Khedher, I. Monacelli, M. P. Trivella, et al. Published in: Computers in Biology and Medicine (2020)
DOI: 10.1016/j.compbiomed.2020.103977
4. Title: "Parkinson's Disease Detection from Handwriting using Machine Learning: A Review" Authors: R. Karthikeyan, S. Ravi Published in: Biocybernetics and Biomedical Engineering (2019)
DOI: 10.1016/j.bbe.2019.10.011
5. Title: "An Ensemble Deep Learning-Based Approach for Automatic Detection of Parkinson's Disease" Authors: D. Zhang, Y. Wang, C. Zhou, et al. Published in: Computers in Biology and Medicine (2020)
DOI: 10.1016/j.compbiomed.2020.103943
6. Title: "Machine Learning in the Detection of Parkinson's Disease" Authors: S. J. Karamzadeh, R. Amiri, M. R. Calvo, et al. Published in: Journal of Neuroscience Methods (2020)
DOI: 10.1016/j.jneumeth.2019.108502
7. Abiyev, R. H., and Abizade, S. (2016). Diagnosing Parkinson's diseases using fuzzy neural system. Comput. Mathe. Methods Med. 2016:1267919. doi: 10.1155/2016/1267919
8. Abos, A., Baggio, H. C., Segura, B., Campabadal, A., Uribe, C., Giraldo, D. M., et al. (2019). Differentiation of multiple system atrophy from Parkinson's disease by structural connectivity derived from probabilistic tractography. Sci. Rep. 9:16488. doi: 10.1038/s41598-019-52829-8
9. Abujrida, H., Agu, E., and Pahlavan, K. (2017). “Smartphone-based gait assessment to infer Parkinson's disease severity using crowdsourced data,” in 2017 IEEE Healthcare Innovations and Point of Care Technologies (HI-POCT) (Bethesda, MD), 208–211. doi: 10.1109/HIC.2017.8227621
10. Adams, W. R. (2017). High-accuracy detection of early Parkinson's Disease using multiple characteristics of finger movement while typing. PLoS ONE 12:e0188226. doi: 10.1371/journal.pone.0188226
11. Adeli, E., Shi, F., An, L., Wee, C.-Y., Wu, G., Wang, T., et al. (2016). Joint feature-sample selection and robust diagnosis of Parkinson's disease from MRI data. NeuroImage 141, 206–219. doi: 10.1016/j.neuroimage.2016.05.054

8. Diagrams:



9. Model Used:

XGBoost (Extreme Gradient Boosting):
XGBoost is a popular and powerful machine learning algorithm that belongs to the class of ensemble learning methods. It is particularly effective for structured/tabular data and is widely used in machine learning competitions and real-world applications.
Here are some key aspects of XGBoost:

1. **Gradient Boosting:** XGBoost is an implementation of the gradient boosting framework. Gradient boosting is an ensemble learning technique where weak learners (usually decision trees) are trained sequentially, with each new tree trying to correct the errors made by the previous ones.

2. **Regularization:** XGBoost includes regularization terms in its objective function to control model complexity and help prevent overfitting. Regularization helps in building models that generalize well to unseen data.

3. **Parallel and Distributed Computing:** XGBoost is designed for efficiency and speed. It can be parallelized across CPU cores and can also be distributed across a cluster of machines, making it scalable and able to handle large datasets.

4. **Tree Pruning:** XGBoost uses a technique called "pruning" during the tree-building process. Pruning involves removing parts of the tree that do not provide significant predictive power, which helps prevent the model from becoming too complex and overfitting the training data.

5. **Handling Missing Data:** XGBoost has a built-in mechanism to handle missing data, which is common in real-world datasets. It can automatically learn how to best impute missing values during the training process.

6. **Feature Importance:** XGBoost provides a feature importance score for each feature in the dataset, indicating the contribution of each feature to the model's predictions. This can be useful for feature selection and understanding the importance of different variables in the model.

7. **Wide Applicability:** XGBoost can be applied to various machine learning tasks, including classification, regression, and ranking problems.

10.Source Code:

In [32]:
```
import numpy as np
import pandas as pd
```

Importing Libraries

In [33]:
```
import pandas as pd
import numpy as np
import os,sys
import xgboost as xgb
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Load Dataset

In [34]:
```
df=pd.read_csv("parkinsons.data")
df
```

Out[34]:

	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Ab
0	phon_R01_S01_1	119.992	157.302	74.997	0.00784	0.0000
1	phon_R01_S01_2	122.400	148.650	113.819	0.00968	0.0000
2	phon_R01_S01_3	116.682	131.111	111.555	0.01050	0.0000
3	phon_R01_S01_4	116.676	137.871	111.366	0.00997	0.0000
4	phon_R01_S01_5	116.014	141.781	110.655	0.01284	0.0001
...	...	...	...	...	...	
190	phon_R01_S50_2	174.188	230.978	94.261	0.00459	0.0000
191	phon_R01_S50_3	209.516	253.017	89.488	0.00564	0.0000
192	phon_R01_S50_4	174.688	240.005	74.287	0.01360	0.0000
193	phon_R01_S50_5	198.764	396.961	74.904	0.00740	0.0000
194	phon_R01_S50_6	214.289	260.277	77.973	0.00567	0.0000

195 rows × 24 columns

Renaming columns

In [35]:
```
df.rename(columns=({'MDVP:Fo(Hz)':'avg_fre', 'MDVP:Fhi(Hz)':'max_fre', 'MDVP:Fl
        'MDVP:Jitter(Abs)':'var_fre2', 'MDVP:RAP':'var_fre3', 'MDVP:PPQ':'var_fr
        'MDVP:Shimmer':'var_amp1', 'MDVP:Shimmer(dB)':'var_amp2', 'Shimmer:APQ3'
        'MDVP:APQ':'var_amp5', 'Shimmer:DDA':'var_amp6'}),inplace=True)
df
```

Out[35]:

	name	avg_fre	max_fre	min_fre	var_fre1	var_fre2	var_fre3	var_fre4	var_fre5
0	phon_R01_S01_1	119.992	157.302	74.997	0.00784	0.00007	0.00370	0.00554	0.01109
1	phon_R01_S01_2	122.400	148.650	113.819	0.00968	0.00008	0.00465	0.00696	0.01394
2	phon_R01_S01_3	116.682	131.111	111.555	0.01050	0.00009	0.00544	0.00781	0.01633
3	phon_R01_S01_4	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.01505
4	phon_R01_S01_5	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.01966
...	...	...	...	...	...	...	...	...	...
190	phon_R01_S50_2	174.188	230.978	94.261	0.00459	0.00003	0.00263	0.00259	0.00790
191	phon_R01_S50_3	209.516	253.017	89.488	0.00564	0.00003	0.00331	0.00292	0.00994
192	phon_R01_S50_4	174.688	240.005	74.287	0.01360	0.00008	0.00624	0.00564	0.01873
193	phon_R01_S50_5	198.764	396.961	74.904	0.00740	0.00004	0.00370	0.00390	0.01109
194	phon_R01_S50_6	214.289	260.277	77.973	0.00567	0.00003	0.00295	0.00317	0.00885

195 rows × 24 columns

Dimensions of Dataset

In [36]:
```
df.shape
```

Out[36]: (195, 24)

Peak at the Data

In [37]:
```
df.head(20)
```

Out[37]:

	name	avg_fre	max_fre	min_fre	var_fre1	var_fre2	var_fre3	var_fre4	var_fre5
0	phon_R01_S01_1	119.992	157.302	74.997	0.00784	0.00007	0.00370	0.00554	0.01109
1	phon_R01_S01_2	122.400	148.650	113.819	0.00968	0.00008	0.00465	0.00696	0.01394
2	phon_R01_S01_3	116.682	131.111	111.555	0.01050	0.00009	0.00544	0.00781	0.01633
3	phon_R01_S01_4	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.01505
4	phon_R01_S01_5	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.01966
5	phon_R01_S01_6	120.552	131.162	113.787	0.00968	0.00008	0.00463	0.00750	0.01388
6	phon_R01_S02_1	120.267	137.244	114.820	0.00333	0.00003	0.00155	0.00202	0.00466
7	phon_R01_S02_2	107.332	113.840	104.315	0.00290	0.00003	0.00144	0.00182	0.00431
8	phon_R01_S02_3	95.730	132.068	91.754	0.00551	0.00006	0.00293	0.00332	0.00880
9	phon_R01_S02_4	95.056	120.103	91.226	0.00532	0.00006	0.00268	0.00332	0.00803
10	phon_R01_S02_5	88.333	112.240	84.072	0.00505	0.00006	0.00254	0.00330	0.00763
11	phon_R01_S02_6	91.904	115.871	86.292	0.00540	0.00006	0.00281	0.00336	0.00844
12	phon_R01_S04_1	136.926	159.866	131.276	0.00293	0.00002	0.00118	0.00153	0.00355
13	phon_R01_S04_2	139.173	179.139	76.556	0.00390	0.00003	0.00165	0.00208	0.00496
14	phon_R01_S04_3	152.845	163.305	75.836	0.00294	0.00002	0.00121	0.00149	0.00364
15	phon_R01_S04_4	142.167	217.455	83.159	0.00369	0.00003	0.00157	0.00203	0.00471
16	phon_R01_S04_5	144.188	349.259	82.764	0.00544	0.00004	0.00211	0.00292	0.00632
17	phon_R01_S04_6	168.778	232.181	75.603	0.00718	0.00004	0.00284	0.00387	0.00853
18	phon_R01_S05_1	153.046	175.829	68.623	0.00742	0.00005	0.00364	0.00432	0.01092
19	phon_R01_S05_2	156.405	189.398	142.822	0.00768	0.00005	0.00372	0.00399	0.01116

20 rows × 24 columns

Statistical Summary

In [38]:
```
df.describe()
```

Out[38]:

	avg_fre	max_fre	min_fre	var_fre1	var_fre2	var_fre3	var_fre4	va
count	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.(
mean	154.228641	197.104918	116.324631	0.006220	0.000044	0.003306	0.003446	0.(
std	41.390065	91.491548	43.521413	0.004848	0.000035	0.002968	0.002759	0.(
min	88.333000	102.145000	65.476000	0.001680	0.000007	0.000680	0.000920	0.(
25%	117.572000	134.862500	84.291000	0.003460	0.000020	0.001660	0.001860	0.(
50%	148.790000	175.829000	104.315000	0.004940	0.000030	0.002500	0.002690	0.(
75%	182.769000	224.205500	140.018500	0.007365	0.000060	0.003835	0.003955	0.(
max	260.105000	592.030000	239.170000	0.033160	0.000260	0.021440	0.019580	0.(

8 rows × 23 columns

Information of the dataset

In [39]:
```
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   avg_fre   195 non-null    float64
 2   max_fre   195 non-null    float64
 3   min_fre   195 non-null    float64
 4   var_fre1  195 non-null    float64
 5   var_fre2  195 non-null    float64
 6   var_fre3  195 non-null    float64
 7   var_fre4  195 non-null    float64
 8   var_fre5  195 non-null    float64
 9   var_amp1  195 non-null    float64
 10  var_amp2  195 non-null    float64
 11  var_amp3  195 non-null    float64
 12  var_amp4  195 non-null    float64
 13  var_amp5  195 non-null    float64
 14  var_amp6  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
```

Duplicate Entries

In [40]:
```
df.duplicated().sum()
```

Out[40]: 0

unwanted columns

In [41]:
```
df.drop(columns="name",axis=1,inplace=True)
df
```

Out[41]:

	avg_fre	max_fre	min_fre	var_fre1	var_fre2	var_fre3	var_fre4	var_fre5	var_amp1	var_an
0	119.992	157.302	74.997	0.00784	0.00007	0.00370	0.00554	0.01109	0.04374	0.
1	122.400	148.650	113.819	0.00968	0.00008	0.00465	0.00696	0.01394	0.06134	0.
2	116.682	131.111	111.555	0.01050	0.00009	0.00544	0.00781	0.01633	0.05233	0.
3	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.01505	0.05492	0.
4	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.01966	0.06425	0.
...	...	...	...	...	...	...	...	...	...	
190	174.188	230.978	94.261	0.00459	0.00003	0.00263	0.00259	0.00790	0.04087	0.
191	209.516	253.017	89.488	0.00564	0.00003	0.00331	0.00292	0.00994	0.02751	0.
192	174.688	240.005	74.287	0.01360	0.00008	0.00624	0.00564	0.01873	0.02308	0.
193	198.764	396.961	74.904	0.00740	0.00004	0.00370	0.00390	0.01109	0.02296	0.
194	214.289	260.277	77.973	0.00567	0.00003	0.00295	0.00317	0.00885	0.01884	0.

195 rows × 23 columns

Missing values

In [42]:
```
df.isnull().sum()
```

Out[42]:
```
avg_fre      0
max_fre      0
min_fre      0
var_fre1     0
var_fre2     0
var_fre3     0
var_fre4     0
var_fre5     0
var_amp1     0
var_amp2     0
var_amp3     0
var_amp4     0
var_amp5     0
var_amp6     0
NHR          0
HNR          0
status       0
RPDE         0
DFA          0
spread1      0
spread2      0
D2           0
PPE          0
dtype: int64
```

In [43]:
```
df.notnull()
```

Out[43]:

	avg_fre	max_fre	min_fre	var_fre1	var_fre2	var_fre3	var_fre4	var_fre5	var_amp1	var_an
0	True	True	True	True	True	True	True	True	True	T
1	True	True	True	True	True	True	True	True	True	T
2	True	True	True	True	True	True	True	True	True	T
3	True	True	True	True	True	True	True	True	True	T
4	True	True	True	True	True	True	True	True	True	T
...	...	...	...	...	...	...	...	...	...	
190	True	True	True	True	True	True	True	True	True	T
191	True	True	True	True	True	True	True	True	True	T
192	True	True	True	True	True	True	True	True	True	T
193	True	True	True	True	True	True	True	True	True	T
194	True	True	True	True	True	True	True	True	True	T

195 rows × 23 columns

Outliers

In [44]:
```
df.columns
```

Out[44]:
```
Index(['avg_fre', 'max_fre', 'min_fre', 'var_fre1', 'var_fre2', 'var_fre3',
       'var_fre4', 'var_fre5', 'var_amp1', 'var_amp2', 'var_amp3', 'var_amp
4',
       'var_amp5', 'var_amp6', 'NHR', 'HNR', 'status', 'RPDE', 'DFA',
       'spread1', 'spread2', 'D2', 'PPE'],
      dtype='object')
```

In [45]:
```
df.skew()
```

Out[45]:
```
avg_fre      0.591737
max_fre      2.542146
min_fre      1.217350
var_fre1     3.084946
var_fre2     2.649071
var_fre3     3.360708
var_fre4     3.073892
var_fre5     3.362058
var_amp1     1.666480
var_amp2     1.999389
var_amp3     1.580576
var_amp4     1.798697
var_amp5     2.618047
var_amp6     1.580618
NHR          4.220709
HNR         -0.514317
status      -1.187727
RPDE        -0.143402
DFA         -0.033214
spread1      0.432139
spread2      0.144430
D2           0.430384
PPE          0.797491
dtype: float64
```

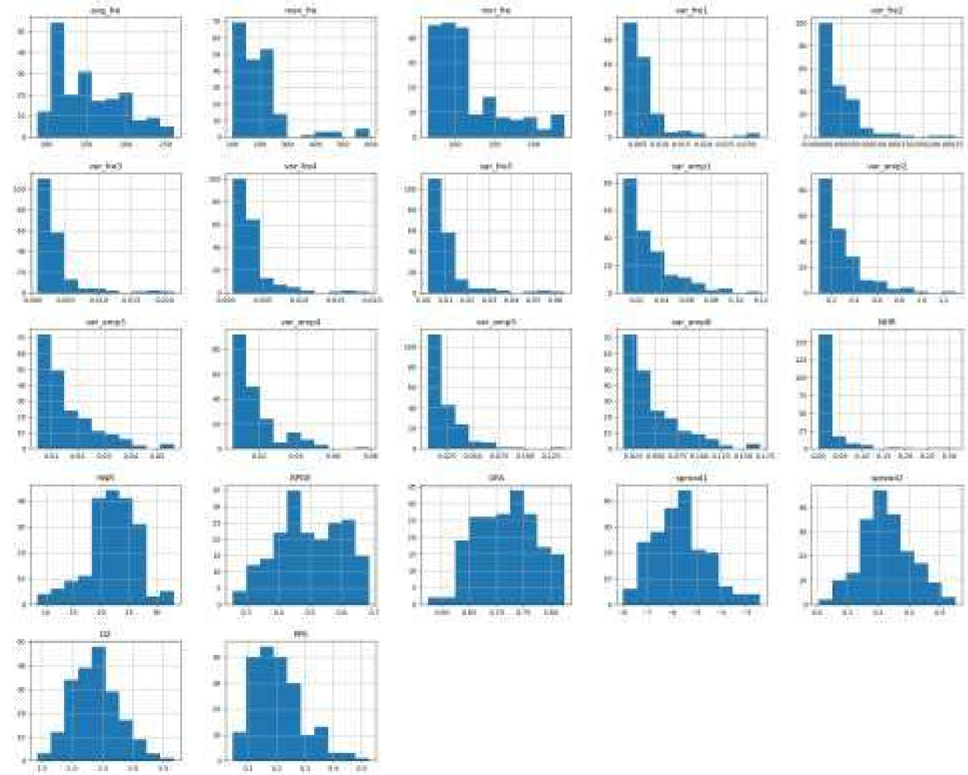
Determining Depentend & Independent Variables

In [46]:
```
# get features and labels

x=df.loc[:,df.columns!='status'].values[:,1:]
x1=df.loc[:,df.columns!='status']
y=df.loc[:,'status'].values
y1=df.loc[:,'status']
```

Analyzing Features

In [47]:
```
x1.hist(figsize=(25,20))
plt.show()
```



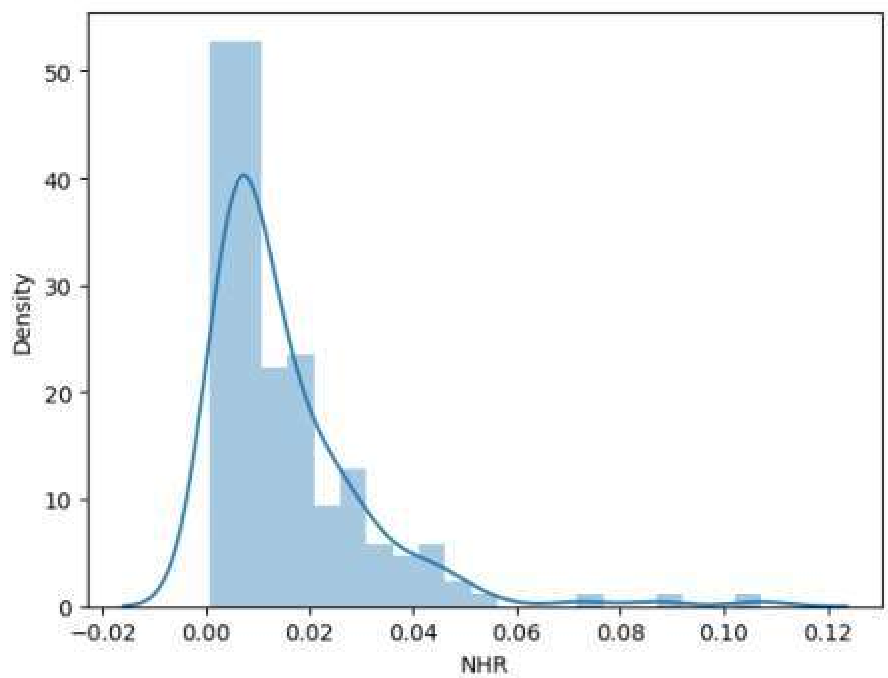
In [48]:
```
df=df[df.max_fre<=300]
df=df[df.var_fre1<=0.02]
df=df[df.var_fre2<=0.0001]
df=df[df.var_fre3<=0.01]
df=df[df.var_fre4<=0.01]
df=df[df.var_fre5<=0.02]
df=df[df.var_amp1<=0.10]
df=df[df.var_amp2<=1.0]
df=df[df.var_amp3<=0.04]
df=df[df.var_amp4<=0.050]
df=df[df.var_amp5<=0.075]
df=df[df.var_amp6<=0.125]
df=df[df.NHR<=0.15]
```

In [49]:
```
df.skew()
```

Out[49]:
```
avg_fre      0.608391
max_fre      0.290164
min_fre      1.247241
var_fre1     0.843153
var_fre2     0.756592
var_fre3     0.811867
var_fre4     1.142506
var_fre5     0.811544
var_amp1     1.077428
var_amp2     1.138932
var_amp3     1.128533
var_amp4     1.376069
var_amp5     1.096979
var_amp6     1.128416
NHR          2.635106
HNR         -0.035596
status      -1.057890
RPDE        -0.066659
DFA         -0.132660
spread1      0.283933
spread2      0.158902
D2           0.485240
PPE          0.535763
dtype: float64
```

In [50]:
```
sns.distplot(df['NHR'])
```

Out[50]: <Axes: xlabel='NHR', ylabel='Density'>

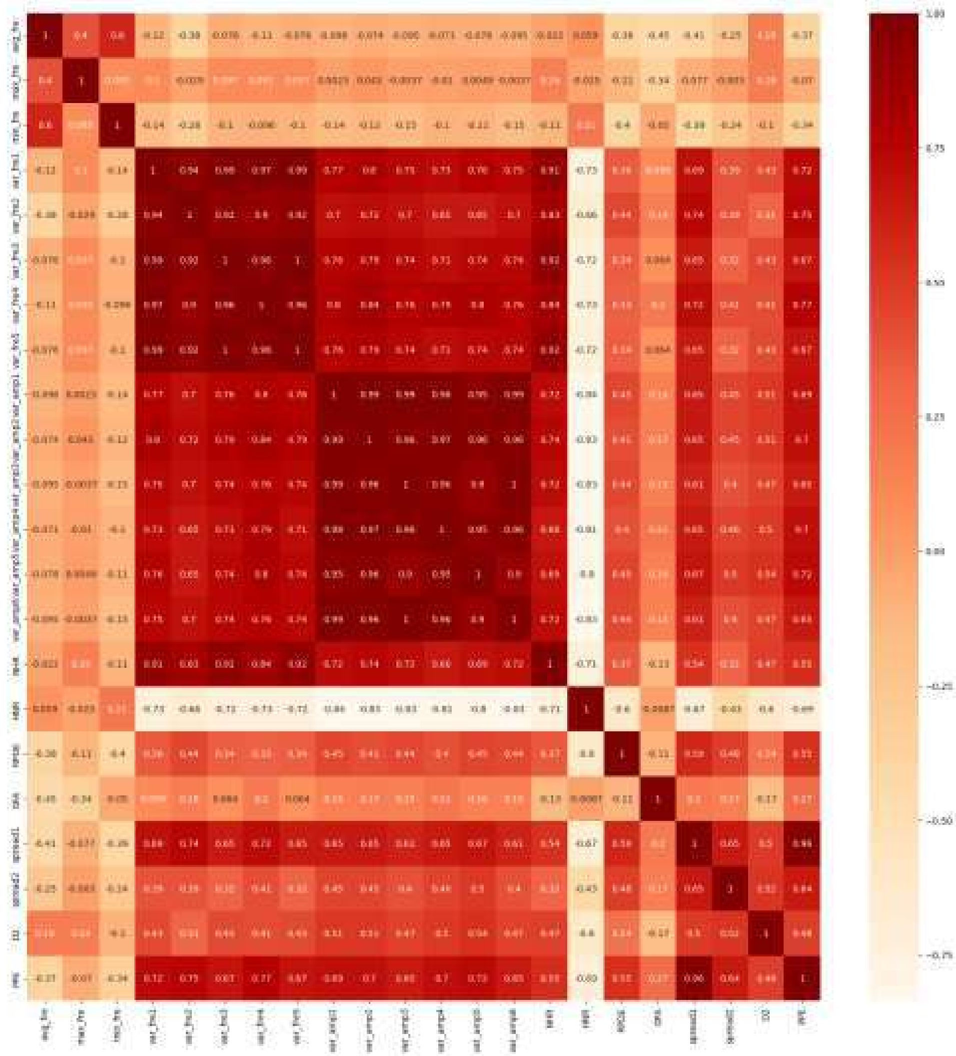


In [51]:
```
df=df[df.NHR<=0.06]
df.skew()
```

Out[51]:
```
avg_fre      0.629564
max_fre      0.328258
min_fre      1.245583
var_fre1     0.699469
var_fre2     0.769365
var_fre3     0.813203
var_fre4     1.212263
var_fre5     0.812495
var_amp1     1.063387
var_amp2     1.136743
var_amp3     1.116058
var_amp4     1.381370
var_amp5     1.098219
var_amp6     1.115979
NHR          1.327245
HNR          0.174386
status      -1.064996
RPDE        -0.061493
DFA         -0.133070
spread1      0.298066
spread2      0.123992
D2           0.194425
PPE          0.553609
dtype: float64
```

Correlation Matrix

In [52]:
```
correl=x1.corr()
plt.figure(figsize=(20,20))
sns.heatmap(correl,annot=True,cmap='OrRd')
plt.show()
```



In [53]:
```
#Scale the features to between -1 and 1
scaler=MinMaxScaler((-1,1))
x1=scaler.fit_transform(x)
y1=y
```

In [54]:
```
#Split the dataset

xtrain,xtest,ytrain,ytest=train_test_split(x1, y1, test_size=0.2)
```

In [55]:
```
# Train the model
from xgboost import XGBClassifier


model=XGBClassifier()
model.fit(xtrain,ytrain)
predict=model.predict(xtest)
```

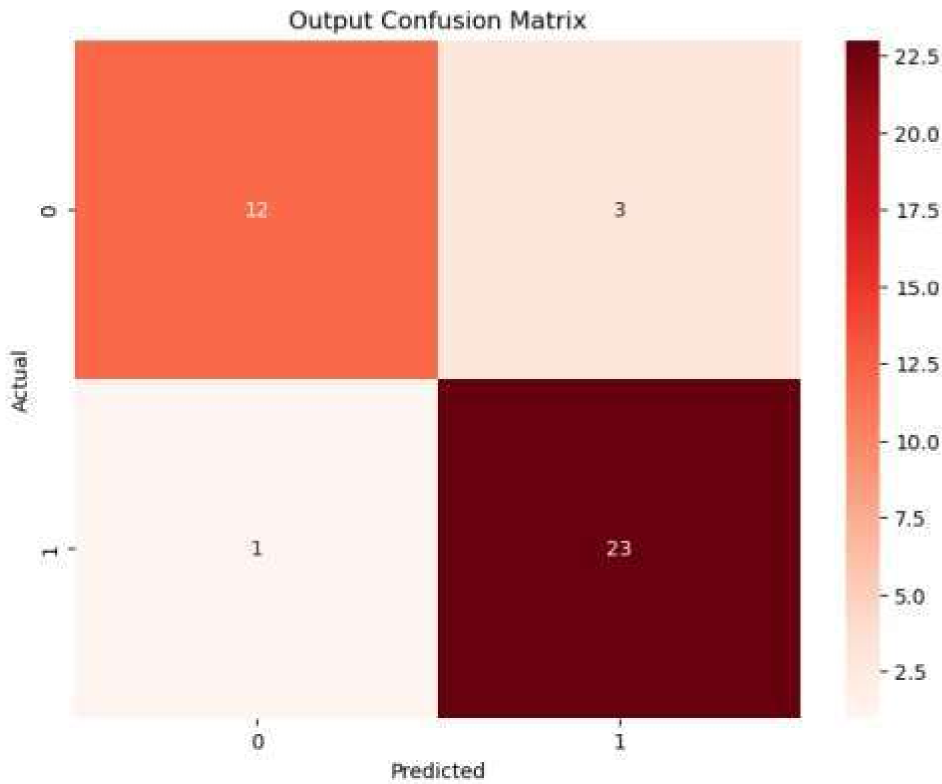
In [56]:
```
print(accuracy_score(ytest,predict)*100)
```

```
89.74358974358975
```

Implementing Confusion Matrix

In [57]:
```
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(ytest,predict)
plt.figure(figsize=(8,6))
fg=sns.heatmap(cm,annot=True,cmap="Reds")
figure=fg.get_figure()
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title("Output Confusion Matrix")
```

Out[57]:
```
Text(0.5, 1.0, 'Output Confusion Matrix')
```



Output Display

In [58]:
```
pd.DataFrame({'actual':ytest,'predict':predict})
```

11/10/23, 1:53 PM parkinsons using xgboost - Jupyter Notebook

Out[58]:

	actual	predict
0	0	0
1	1	1
2	0	0
3	0	0
4	0	0
5	1	1
6	0	1
7	1	1
8	1	1
9	0	0
10	0	1
11	1	1
12	1	1
13	0	0
14	0	0
15	1	1
16	1	1
17	1	0
18	0	0
19	1	1
20	1	1
21	1	1
22	0	1
23	0	0
24	1	1
25	1	1
26	1	1
27	1	1
28	0	0
29	1	1
30	1	1
31	1	1
32	0	0
33	1	1
34	1	1
35	1	1

	actual	predict
36	1	1
37	0	0
38	1	1

Prediction With New Input

In [61]:
```
newinput=[[274.688,240.005,174.287,0.01360,0.01008,0.00624,0.00564,0.01873,1.02
```

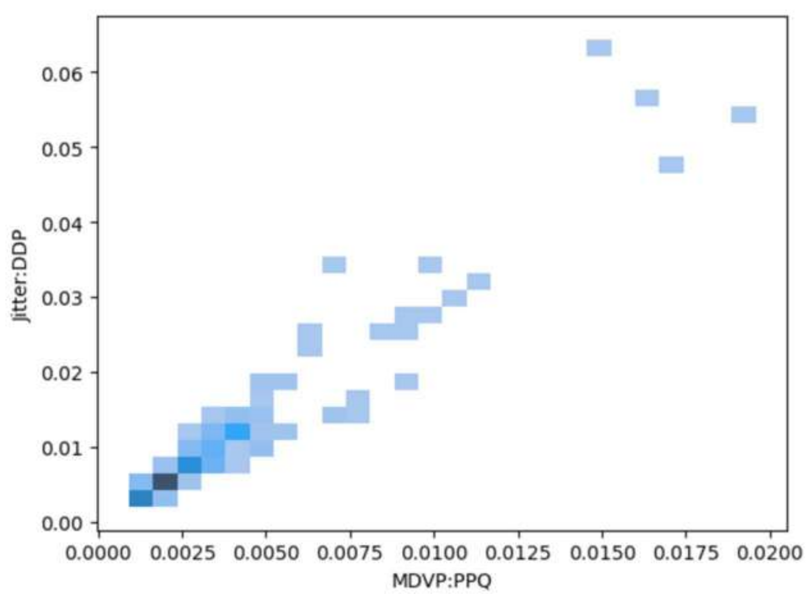
In [62]:
```
output=model.predict(newinput)
output
```

Out[62]:
```
array([1])
```

In [ ]:

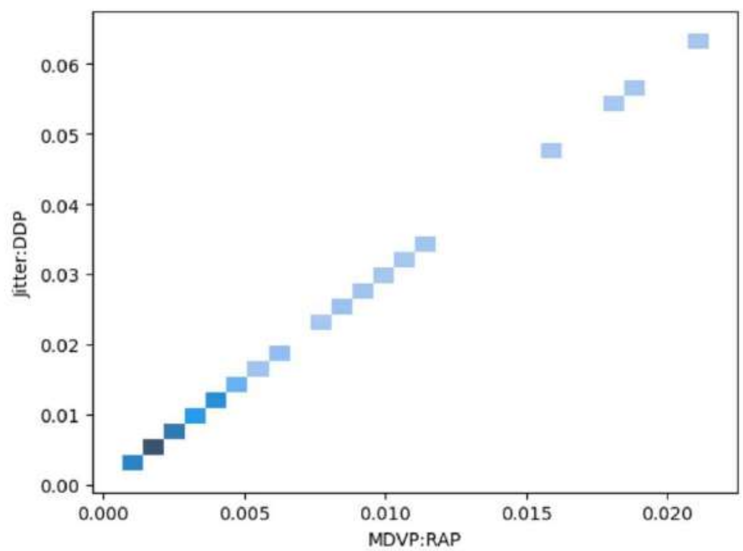
In [16]:
```
sns.histplot(x = df["MDVP:PPQ"], y = df["Jitter:DDP"])
```

Out[16]:
```
<Axes: xlabel='MDVP:PPQ', ylabel='Jitter:DDP'>
```



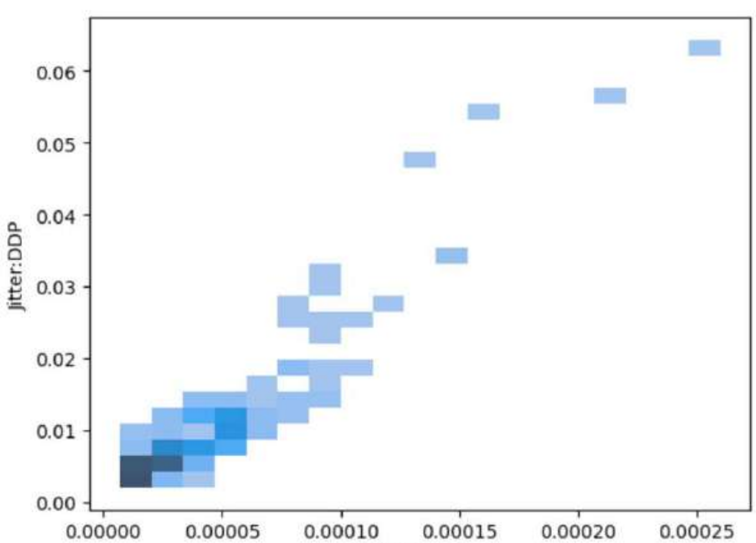
```
In [15]: sns.histplot(x = df["MDVP:RAP"], y = df["Jitter:DDP"])
```

Out[15]: <Axes: xlabel='MDVP:RAP', ylabel='Jitter:DDP'>



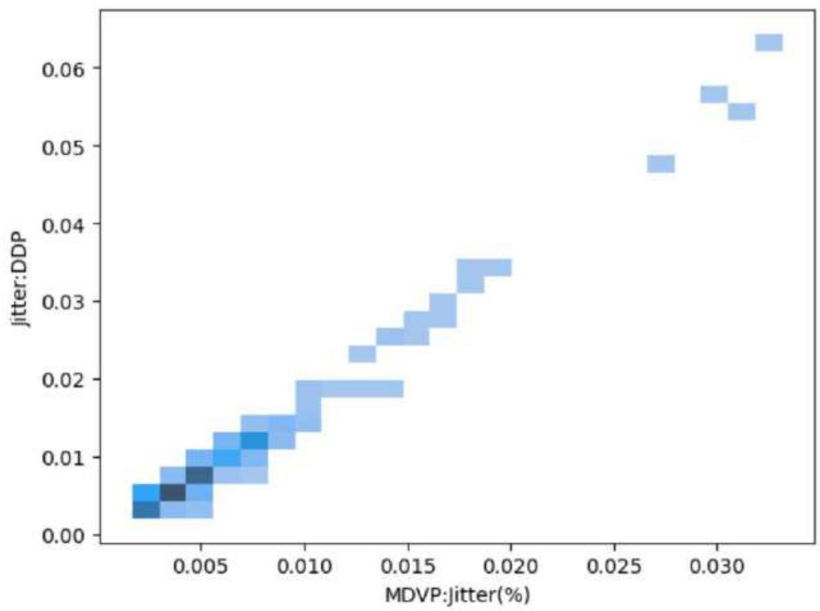

```
In [14]: sns.histplot(x = df["MDVP:Jitter(Abs)"], y = df["Jitter:DDP"])
```

Out[14]: <Axes: xlabel='MDVP:Jitter(Abs)', ylabel='Jitter:DDP'>



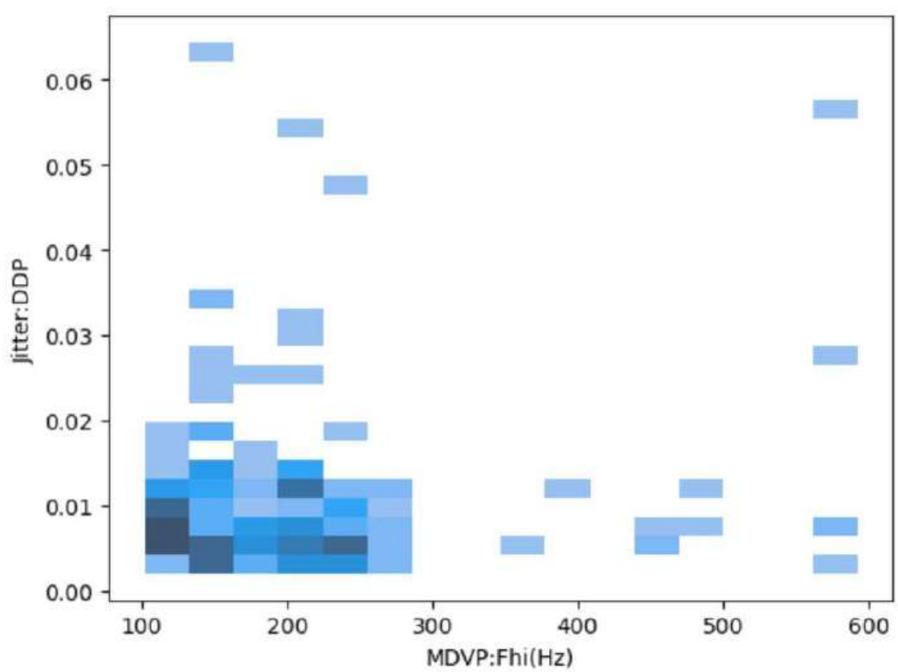
In [13]:
```
sns.histplot(x = df["MDVP:Jitter(%)"], y = df["Jitter:DDP"])
```

Out[13]: <Axes: xlabel='MDVP:Jitter(%)', ylabel='Jitter:DDP'>



In [12]:
```
sns.histplot(x = df["MDVP:Fhi(Hz)"], y = df["Jitter:DDP"])
```

Out[12]: <Axes: xlabel='MDVP:Fhi(Hz)', ylabel='Jitter:DDP'>



In [11]:
```
sns.histplot(x = df["MDVP:Fo(Hz)"], y = df["Jitter:DDP"])
```
Out[11]: <Axes: xlabel='MDVP:Fo(Hz)', ylabel='Jitter:DDP'>