In [ ]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from collections import Counter
sns.set_theme(style="ticks", color_codes=True)

data = pd.read_csv('https://raw.githubusercontent.com/eecastillo/parkinson_analisis/master/R_project/Shiny/files/clean_handwriting.csv',sep='|')
```

In [ ]:
```
data_df = pd.DataFrame(data)
data_df
```

Out[ ]:

	N	Objectives	Type of diagnosis	Source of data	Number of subjects (n)	Machine learning method(s)	Outcomes	Training accuracy	Accuracy	Sensitivity	Precision	Specificity	AUC	Recall	PPV	NPV
0	1	Classification of PD from MSA	Differential diagnosis	collected from participants	150; 54 HC + 65 PD + 31 MSA	SVM with leave-one-out- cross validation	MSA vs PD: accuracy = 0.79 sensitivity = 0.71 ...	NaN	0.7900	0.7100	NaN	0.8600	NaN	NaN	NaN	NaN
1	2	Classification of PD from MSA	Differential diagnosis	collected from participants	150; 54 HC + 65 PD + 31 MSA	SVM with leave-one-out- cross validation	MSA vs HC: accuracy = 0.79 sensitivity = 0.84 ...	NaN	0.7900	0.8400	NaN	0.7400	NaN	NaN	NaN	NaN
2	3	Classification of PD from MSA	Differential diagnosis	collected from participants	150; 54 HC + 65 PD + 31 MSA	SVM with leave-one-out- cross validation	MSA vs subsample of PD: accuracy = 0.84 sensit...	NaN	0.8400	0.7700	NaN	0.9000	NaN	NaN	NaN	NaN
3	4	Classification of PD from MSA	Differential diagnosis	collected from participants	151; 59 HC + 62 PD + 30 MSA	SVM with leave-one-out- cross validation	accuracy = 77.17% sensitivity = 83.33% specifi...	NaN	0.7717	0.8333	NaN	0.7419	NaN	NaN	NaN	NaN
4	5	Classification of PD from HC	Diagnosis	collected from participants	94; 50 HC + 44 PD	CNN with 85 subjects for training and 9 for te...	training accuracy = 95.24% testing accuracy = ...	0.9524	0.8888	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
57	58	Classification of PD, HC and SWEDD	Diagnosis and differential diagnosis	PPMI database	741; 262 HC + 408 PD + 71 SWEDD	LSSVM-RBF with 10- fold cross validation	PD vs HC accuracy = 95.37%	NaN	0.9537	NaN	NaN	NaN	NaN	NaN	NaN	NaN
58	59	Classification of PD, HC and SWEDD	Diagnosis and differential diagnosis	PPMI database	741; 262 HC + 408 PD + 71 SWEDD	LSSVM-RBF with 10- fold cross validation	PD vs SWEDD accuracy = 96.04%	NaN	0.9604	NaN	NaN	NaN	NaN	NaN	NaN	NaN
59	60	Classification of PD, HC and SWEDD	Diagnosis and differential diagnosis	PPMI database	741; 262 HC + 408 PD + 71 SWEDD	LSSVM-RBF with 10- fold cross validation	SWEDD vs HC accuracy = 93.03%	NaN	0.9303	NaN	NaN	NaN	NaN	NaN	NaN	NaN
60	61	Classification of PD from HC	Diagnosis	PPMI database	408; 204 HC + 204 PD	CNN (VGG and ResNet)	ResNet50 accuracy = 88.6%	NaN	0.8860	NaN	NaN	NaN	NaN	NaN	NaN	NaN
61	62	Classification of PD from HC	Diagnosis	PPMI database	754; 158 HC + 596 PD	FCN, GCN with 5-fold cross validation	AUC = 95.37%	NaN	NaN	NaN	NaN	NaN	0.9537	NaN	NaN	NaN

62 rows × 25 columns

In [ ]:
```
ML_methods = ["cross validation", "LS-SVM", "PNN", "SVM-RBF","SVM-linear","SCFW-KELM","SVM","FKNN",
              "ECFA-SVM","DNN","SMO","Pegasos","AdaBoost","FBANN","rotation forest","NNge",
              "logistic regression","KNN","naive Bayes","decision tree","random forest","CNN",
              "LSSVM-RBF","MLP","joint learning","ELM","NN","EER"]
```

In [ ]:
```
data_df=data_df.rename(columns={'Machine learning method(s)': 'ML_method'})
```

In [ ]:
```
orderDf(data_df,'Accuracy',False)
```

Out[ ]:

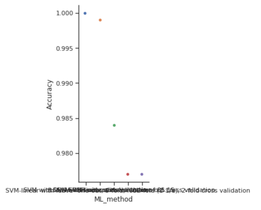
	ML_method	Accuracy
6	SVM with leave-one-out cross validation	1.0000
56	LSSVM-RBF with cross validation	0.9990
28	CNN with train-validation ratio of 85:15	0.9840
38	SVM-linear with leave- one-out, 5-fold, .632-f...	0.9770
15	SVM-linear with leave- one-out cross validation	0.9770
18	SVM-RBF with 10-fold cross-validation	0.9750
25	CNN with train-validation ratio of 85:15	0.9680
58	LSSVM-RBF with 10- fold cross validation	0.9604
57	LSSVM-RBF with 10- fold cross validation	0.9537
42	MLP, XgBoost, random forest, SVM with 5-fold c...	0.9530
27	CNN with train-validation ratio of 85:15	0.9520
48	joint learning with 10- fold cross validation	0.9489
46	SVM-linear with leave- one-out cross validation	0.9459
45	naive Bayes, SVM-RBF with 10-fold cross valida...	0.9459
22	SVM-RBF with 10-fold cross validation	0.9420
17	SVM-linear with leave- one-out cross validation	0.9375
26	CNN with train-validation ratio of 85:15	0.9370
5	SVM-linear with leave- one-out cross validation	0.9362
59	LSSVM-RBF with 10- fold cross validation	0.9303
41	random forest (for feature selection and clini...	0.9300
53	SVM with 2-fold cross validation	0.9235
49	joint learning with 10- fold cross validation	0.9212
47	joint learning with 10- fold cross validation	0.9112
51	SVM, ELM with train-test ratio of 80:20	0.9097
23	SVM-RBF with 10-fold cross validation	0.9050
33	SVM-linear with leave- one-out cross validation	0.8910
34	SVM-linear with leave- one-out cross validation	0.8890
4	CNN with 85 subjects for training and 9 for te...	0.8888
60	CNN (VGG and ResNet)	0.8860
30	SVM-linear with leave- one-out cross validation	0.8800
16	SVM-linear with leave- one-out cross validation	0.8780
19	SVM-RBF with 10-fold cross validation	0.8692
32	SVM-linear with leave- one-out cross validation	0.8580
52	multi-kernel SVM with 10-fold cross validation	0.8578
36	CNN-DL, CR-ML, RA- ML with 5-fold cross- valid...	0.8570
21	SVM-RBF with 10-fold cross validation	0.8530
2	SVM with leave-one-out- cross validation	0.8400
54	SVM with 2-fold cross validation	0.8391
39	RLDA with JFSS with 10-fold cross validation	0.8190
55	SVM with 2-fold cross validation	0.8084
43	SVM with leave-one-out cross validation	0.8000
35	CNN-DL, CR-ML, RA- ML with 5-fold cross- valid...	0.8000
40	RFS-LDA with 10-fold cross validation	0.7980
0	SVM with leave-one-out- cross validation	0.7900
1	SVM with leave-one-out- cross validation	0.7900
24	SVM-linear with stratified 10-fold cross valid...	0.7833
3	SVM with leave-one-out- cross validation	0.7717
31	boosted logistic regression with nested cross-...	0.7620
14	SVM-linear with leave- one-out cross validation	0.7187
50	RLDA with 8-fold cross validation	0.7050
44	SVM with leave-one-out cross validation	0.6800
20	SVM-RBF with 10-fold cross validation	0.6570
13	SVM-linear with leave- one-out cross validation	0.4186

In [ ]:
```
def orderDf(df,col,order):
  df.dropna(subset = [col], inplace=True)
  temp = df.sort_values(by=[col],ascending=order)
  return temp[['ML_method',col]]
```

In [ ]:
```
def topSpec(df,col,top):
  temp = orderDf(df,col,False)
  return temp.head(top)
```

In [ ]:
```
def plotTopSpec(df,col,top):
  df = topSpec(data_df,col,top)
  sns.catplot(x="ML_method", y=col, data=df)
```

In [ ]:
```
plotTopSpec(data_df,"Accuracy",5)
```



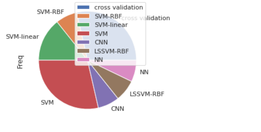
In [ ]:
```
counts = Counter(ML_methods)
print(counts)
```

```
Counter({'cross validation': 1, 'LS-SVM': 1, 'PNN': 1, 'SVM-RBF': 1, 'SVM-linear': 1, 'SCFW-KELM': 1, 'SVM': 1, 'FKNN': 1, 'ECFA-SVM': 1, 'DNN': 1, 'SMO': 1, 'Pegasos': 1, 'AdaBoost': 1, 'FBANN': 1, 'rotation forest': 1, 'NNge': 1, 'logistic regression': 1, 'KNN': 1, 'naive Bayes': 1, 'decision tree': 1, 'random forest': 1, 'CNN': 1, 'LSSVM-RBF': 1, 'MLP': 1, 'joint learning': 1, 'ELM': 1, 'NN': 1, 'EER': 1})
```

In [ ]:
```
def getFreq(substring):
  data = topSpec(data_df,"Accuracy",5)
  return sum(substring in s for s in data["ML_method"])
```

In [ ]:
```
#getFreq(ML_methods[3],data_df["ML_method"])
freq = list(map(getFreq,ML_methods))
#pd.DataFrame(data={ML_methods,freq})
pie = pd.DataFrame(freq, index =ML_methods,columns =['Freq'])
pie=pie.loc[~(pie==0).all(axis=1)]
pie.plot(kind='pie', y='Freq')
```

Out[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb1c23e1150>
```



In [ ]:

In [ ]:

In [ ]: