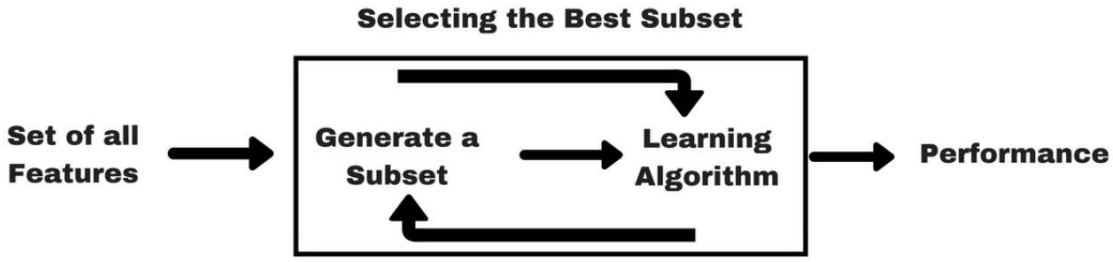
Step Forward, Step Backward and Exhaustive Feature Selection | Wrapper Method

- Use combinations of variables to determine predictive power
- Find the best combination of variables
- Computationally expensive than filter method
- Perform better than filter method
- Not recommended on high number of features

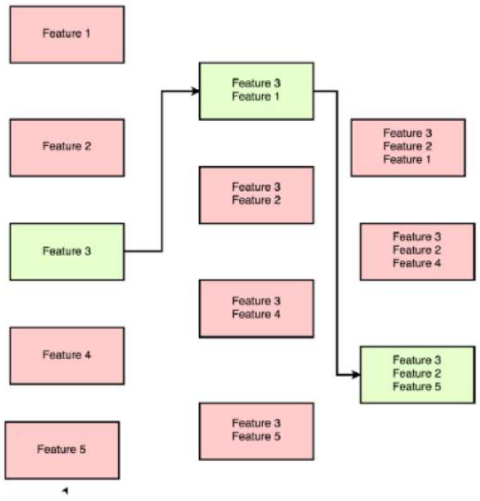
Wrapper Methods

- **Type of methods**
  - Subset Selection (Exhaustive Feature Selection)
  - Forward Step Selection
  - Backward Step Selection (Recursive Feature Selection)



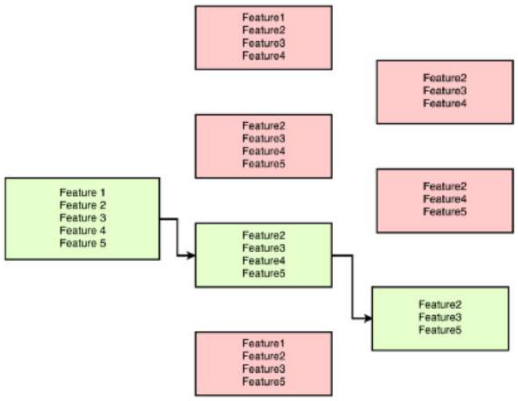
Wrapper Methods

- Forward Step Selection



Wrapper Methods

- Backward Step Selection (Recursive Feature Selection)



Wrapper Methods

- Subset Selection (Exhaustive Feature Selection)
  - fits the model with each possible combinations of N features.
  - requires massive computational power

    (Y = B0 , Y = B0 + B1*X1 , Y = C0 + C1*X2 , Y = D0 + D1*X1 + D2*X2)
  - Use test error to evaluate model performance

Use of mlxtend in Wrapper Method

In [3]:
```
!pip install mlxtend
```

```
Requirement already satisfied: mlxtend in c:\users\ibra5\appdata\local\progra
ms\python\python38\lib\site-packages (0.22.0)
Requirement already satisfied: numpy>=1.16.2 in c:\users\ibra5\appdata\roamin
g\python\python38\site-packages (from mlxtend) (1.23.5)
Requirement already satisfied: pandas>=0.24.2 in c:\users\ibra5\appdata\local
\programs\python\python38\lib\site-packages (from mlxtend) (1.5.3)
Requirement already satisfied: matplotlib>=3.0.0 in c:\users\ibra5\appdata\lo
cal\programs\python\python38\lib\site-packages (from mlxtend) (3.7.1)
Requirement already satisfied: scipy>=1.2.1 in c:\users\ibra5\appdata\roaming
\python\python38\site-packages (from mlxtend) (1.10.1)
Requirement already satisfied: scikit-learn>=1.0.2 in c:\users\ibra5\appdata
\roaming\python\python38\site-packages (from mlxtend) (1.2.2)
Requirement already satisfied: joblib>=0.13.2 in c:\users\ibra5\appdata\roami
ng\python\python38\site-packages (from mlxtend) (1.2.0)
Requirement already satisfied: setuptools in c:\users\ibra5\appdata\local\pro
grams\python\python38\lib\site-packages (from mlxtend) (47.1.0)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\ibra5\appda
ta\local\programs\python\python38\lib\site-packages (from pandas>=0.24.2->mlx
tend) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\ibra5\appdata\local\p
rograms\python\python38\lib\site-packages (from pandas>=0.24.2->mlxtend) (202
2.7.1)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\ibra5\appdata\loc
al\programs\python\python38\lib\site-packages (from matplotlib>=3.0.0->mlxten
d) (3.0.9)
Requirement already satisfied: pillow>=6.2.0 in c:\users\ibra5\appdata\local
\programs\python\python38\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(9.4.0)
Requirement already satisfied: cycler>=0.10 in c:\users\ibra5\appdata\local\p
rograms\python\python38\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(0.11.0)
Requirement already satisfied: importlib-resources>=3.2.0; python_version <
"3.10" in c:\users\ibra5\appdata\local\programs\python\python38\lib\site-pack
ages (from matplotlib>=3.0.0->mlxtend) (5.12.0)
Requirement already satisfied: packaging>=20.0 in c:\users\ibra5\appdata\loca
l\programs\python\python38\lib\site-packages (from matplotlib>=3.0.0->mlxten
d) (23.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\ibra5\appdata\lo
cal\programs\python\python38\lib\site-packages (from matplotlib>=3.0.0->mlxte
nd) (1.4.4)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\ibra5\appdata\loc
al\programs\python\python38\lib\site-packages (from matplotlib>=3.0.0->mlxten
d) (1.0.7)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\ibra5\appdata\lo
cal\programs\python\python38\lib\site-packages (from matplotlib>=3.0.0->mlxte
nd) (4.39.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\ibra5\appdata
\roaming\python\python38\site-packages (from scikit-learn>=1.0.2->mlxtend)
(3.1.0)
Requirement already satisfied: six>=1.5 in c:\users\ibra5\appdata\local\progr
ams\python\python38\lib\site-packages (from python-dateutil>=2.8.1->pandas>=
0.24.2->mlxtend) (1.16.0)
Requirement already satisfied: zipp>=3.1.0; python_version < "3.10" in c:\use
rs\ibra5\appdata\local\programs\python\python38\lib\site-packages (from impor
tlib-resources>=3.2.0; python_version < "3.10"->matplotlib>=3.0.0->mlxtend)
(3.15.0)
```

```
WARNING: You are using pip version 20.1.1; however, version 23.1.2 is availab
le.
You should consider upgrading via the 'c:\users\ibra5\appdata\local\programs
\python\python38\python.exe -m pip install --upgrade pip' command.
```

How it works

Sequential feature selection algorithms are a family of greedy search algorithms that are used to reduce an initial d-dimensional feature space to a k-dimensional feature subspace where k < d.

In a nutshell, SFAs remove or add one feature at the time based on the classifier performance until a feature subset of the desired size k is reached. There are 4 different flavors of SFAs available via the SequentialFeatureSelector:

- Sequential Forward Selection (SFS)
- Sequential Backward Selection (SBS)
- Sequential Forward Floating Selection (SFFS)
- Sequential Backward Floating Selection (SBFS)

Step Forward Selection (SFS)

In [4]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [5]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier
from sklearn.metrics import roc_auc_score
from mlxtend.feature_selection import SequentialFeatureSelector as SFS
```

In [4]:
```
from sklearn.datasets import load_wine
from sklearn.preprocessing import StandardScaler
```

In [19]:
```
data = pd.read_csv('0.9_5subjectslabelled_data.csv', nrows=31437, usecols=lamb
data.head()
```

Out[19]:

	AccX	AccY	AccZ	Gyro_X	Knee Angles	Gait Cycle Phase
0	-0.181472	-0.088708	-0.665352	0.087145	67.223821	5
1	-0.181443	-0.088745	-0.659870	0.103506	67.217858	5
2	-0.183826	-0.089735	-0.654215	0.117635	67.154903	5
3	-0.188545	-0.091706	-0.648504	0.129259	67.011479	5
4	-0.195535	-0.094645	-0.642857	0.138193	66.799616	5

In [20]:
```
data.keys()
```

Out[20]:
```
Index(['AccX', 'AccY', 'AccZ', 'Gyro_X', 'Knee Angles', 'Gait Cycle Phase'],
dtype='object')
```

In [21]:
```
X = data.drop('Gait Cycle Phase', axis = 1)
y = data['Gait Cycle Phase']

X.shape, y.shape
```

Out[21]:
```
((31436, 5), (31436,))
```

In [22]:
```
X.isnull().sum()
```

Out[22]:
```
AccX           0
AccY           0
AccZ           0
Gyro_X         0
Knee Angles    0
dtype: int64
```

In [23]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, ran
X_train.shape, X_test.shape
```

Out[23]:
```
((25148, 5), (6288, 5))
```

Step Forward Feature Selection (SFS)

In [24]:
```
sfs = SFS(RandomForestClassifier(n_estimators=100, random_state=0, n_jobs = -1
         k_features = 5,
          forward= True,
          floating = False,
          verbose= 2,
          scoring= 'accuracy',
          cv = 4,
          n_jobs= -1
         ).fit(X_train, y_train)
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 out of   5 | elapsed:   55.4s remaining:  1.4
min
[Parallel(n_jobs=-1)]: Done   5 out of   5 | elapsed:   57.4s remaining:
0.0s
[Parallel(n_jobs=-1)]: Done   5 out of   5 | elapsed:   57.4s finished

[2023-06-25 02:51:03] Features: 1/5 -- score: 0.37859869572132976[Parallel(n_
jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   4 out of   4 | elapsed:   36.4s remaining:
0.0s
[Parallel(n_jobs=-1)]: Done   4 out of   4 | elapsed:   36.4s finished

[2023-06-25 02:51:39] Features: 2/5 -- score: 0.6335295053284555[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   3 out of   3 | elapsed:   21.9s finished

[2023-06-25 02:52:01] Features: 3/5 -- score: 0.8065452521075234[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 out of   2 | elapsed:   25.9s finished

[2023-06-25 02:52:27] Features: 4/5 -- score: 0.8644425003976459[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:   12.3s finished

[2023-06-25 02:52:39] Features: 5/5 -- score: 0.8954191188166056
```

In [26]:
```
sfs.k_feature_names_
```

Out[26]:
```
('AccX', 'AccY', 'AccZ', 'Gyro_X', 'Knee Angles')
```

In [27]:
```
sfs.k_feature_idx_
```

Out[27]:
```
(0, 1, 2, 3, 4)
```

In [28]:
```
sfs.k_score_
```

Out[28]:
```
0.8954191188166056
```

In [29]:
```
pd.DataFrame.from_dict(sfs.get_metric_dict()).T
```

Out[29]:

	feature_idx	cv_scores	avg_score	feature_names	ci_bound	std_dev	std_err
1	(4,)	[0.37967233974868775, 0.3798313981231112, 0.37...	0.378599	(Knee Angles,)	0.002144	0.001337	0.000772
2	(0, 4)	[0.6230316526165103, 0.6433911245427072, 0.637...	0.63353	(AccX, Knee Angles)	0.012398	0.007734	0.004465
3	(0, 2, 4)	[0.8067440750755527, 0.8115158263082551, 0.807...	0.806545	(AccX, AccZ, Knee Angles)	0.006398	0.003991	0.002304
4	(0, 1, 2, 4)	[0.8627326228725942, 0.8719580085891522, 0.860...	0.864443	(AccX, AccY, AccZ, Knee Angles)	0.007046	0.004396	0.002538
5	(0, 1, 2, 3, 4)	[0.8956577063782408, 0.8997932241132496, 0.892...	0.895419	(AccX, AccY, AccZ, Gyro_X, Knee Angles)	0.004436	0.002768	0.001598

In [30]:
```
sfs = SFS(RandomForestClassifier(n_estimators=100, random_state=0, n_jobs = -1
         k_features = (1, 5),
          forward= True,
          floating = False,
          verbose= 2,
          scoring= 'accuracy',
          cv = 4,
          n_jobs= -1
         ).fit(X_train, y_train)
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 out of   5 | elapsed:   48.4s remaining:  1.2
min
[Parallel(n_jobs=-1)]: Done   5 out of   5 | elapsed:   48.9s remaining:
0.0s
[Parallel(n_jobs=-1)]: Done   5 out of   5 | elapsed:   48.9s finished

[2023-06-25 02:54:52] Features: 1/5 -- score: 0.37859869572132976[Parallel(n_
jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   4 out of   4 | elapsed:   32.6s remaining:
0.0s
[Parallel(n_jobs=-1)]: Done   4 out of   4 | elapsed:   32.6s finished

[2023-06-25 02:55:25] Features: 2/5 -- score: 0.6335295053284555[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   3 out of   3 | elapsed:   21.3s finished

[2023-06-25 02:55:46] Features: 3/5 -- score: 0.8065452521075234[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 out of   2 | elapsed:   24.3s finished

[2023-06-25 02:56:10] Features: 4/5 -- score: 0.8644425003976459[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:   11.4s finished

[2023-06-25 02:56:22] Features: 5/5 -- score: 0.8954191188166056
```

In [31]:
```
sfs.k_score_
```

Out[31]:
```
0.8954191188166056
```

In [32]:
```
sfs.k_feature_names_
```

Out[32]:
```
('AccX', 'AccY', 'AccZ', 'Gyro_X', 'Knee Angles')
```

Step Backward Selection (SBS)

In [34]:
```
sfs = SFS(RandomForestClassifier(n_estimators=100, random_state=0, n_jobs = -1
         k_features = (1, 5),
          forward= False,
          floating = False,
          verbose= 2,
          scoring= 'accuracy',
          cv = 4,
          n_jobs= -1
         ).fit(X_train, y_train)
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 out of   5 | elapsed:   57.2s remaining:  1.4
min
[Parallel(n_jobs=-1)]: Done   5 out of   5 | elapsed:   59.8s remaining:
0.0s
[Parallel(n_jobs=-1)]: Done   5 out of   5 | elapsed:   59.8s finished

[2023-06-25 03:01:28] Features: 4/1 -- score: 0.8644425003976459[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   4 out of   4 | elapsed:   30.4s remaining:
0.0s
[Parallel(n_jobs=-1)]: Done   4 out of   4 | elapsed:   30.4s finished

[2023-06-25 03:01:58] Features: 3/1 -- score: 0.8065452521075234[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   3 out of   3 | elapsed:   26.2s finished

[2023-06-25 03:02:24] Features: 2/1 -- score: 0.6335295053284555[Parallel(n_j
obs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 out of   2 | elapsed:   22.9s finished

[2023-06-25 03:02:47] Features: 1/1 -- score: 0.37859869572132976
```

In [37]:
```
sbs = sfs
```

In [38]:
```
sbs.k_score_
```

Out[38]:
```
0.8954191188166056
```

In [39]:
```
sbs.k_feature_names_
```

Out[39]:
```
('AccX', 'AccY', 'AccZ', 'Gyro_X', 'Knee Angles')
```

Exhaustive Feature Selection (EFS)

In [40]:
```
from mlxtend.feature_selection import ExhaustiveFeatureSelector as EFS
```

In [42]:
```
efs = EFS(RandomForestClassifier(n_estimators=100, random_state=0, n_jobs=-1),
          min_features= 4,
          max_features= 5,
          scoring='accuracy',
          cv = None,
          n_jobs=-1
         ).fit(X_train, y_train)
```

```
Features: 6/6
```

C(13, 4) + C(13, 5) = 715 + 1287

In [43]:
```
715 + 1287
```

Out[43]:
```
2002
```

In [44]:
```
help(efs)
```

```
Help on ExhaustiveFeatureSelector in module mlxtend.feature_selection.exhau
stive_feature_selector object:

class ExhaustiveFeatureSelector(sklearn.base.BaseEstimator, sklearn.base.Me
taEstimatorMixin)
 |  ExhaustiveFeatureSelector(estimator, min_features=1, max_features=1, pr
int_progress=True, scoring='accuracy', cv=5, n_jobs=1, pre_dispatch='2*n_jo
bs', clone_estimator=True, fixed_features=None, feature_groups=None)
 |
 |  Exhaustive Feature Selection for Classification and Regression.
 |     (new in v0.4.3)
 |
 |  Parameters
 |  ----------
 |  estimator : scikit-learn classifier or regressor
 |
 |  min_features : int (default: 1)
 |      Minumum number of features to select
 |
```

In [45]:
```
efs.best_score_
```

Out[45]:
```
1.0
```

In [46]:
```
efs.best_feature_names_
```

Out[46]:
```
('AccX', 'AccY', 'AccZ', 'Gyro_X')
```

In [47]:
```
efs.best_idx_
```

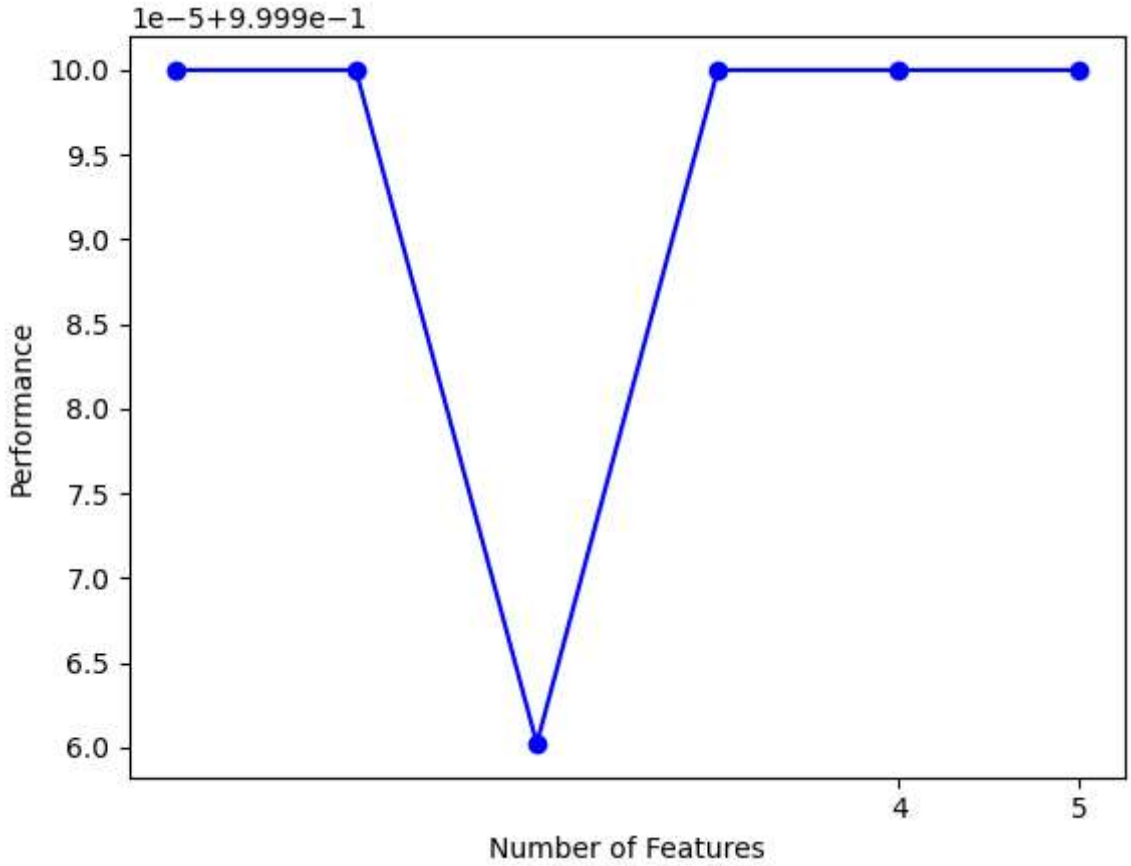
Out[47]:
```
(0, 1, 2, 3)
```

In [48]:
```
from mlxtend.plotting import plot_sequential_feature_selection as plot_sfs
```

In [49]:
```
plot_sfs(efs.get_metric_dict(), kind='std_dev')
```

```
C:\Users\ibra5\AppData\Roaming\Python\Python38\site-packages\numpy\core\_meth
ods.py:265: RuntimeWarning: Degrees of freedom <= 0 for slice
  ret = _var(a, axis=axis, dtype=dtype, out=out, ddof=ddof,
C:\Users\ibra5\AppData\Roaming\Python\Python38\site-packages\numpy\core\_meth
ods.py:257: RuntimeWarning: invalid value encountered in double_scalars
  ret = ret.dtype.type(ret / rcount)
```

Out[49]:
```
(<Figure size 640x480 with 1 Axes>,
 <Axes: xlabel='Number of Features', ylabel='Performance'>)
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: