18. Feature Selection Techniques

- Data consist of many columns representing number of features, so we will select only those columns which are important for ML model building
- Feature selection is we select certian columns from many available columns
- columbn = feature
- A features is an attribute that has an impact on a problem or is useful for the problem, and choosing the important features for the mdoel is known as feature selection
- One should have domain knowledge in order to select appropriate features from data

18.1 Feature Selection by Forward Elimination

- In this example, we will select feature even if we don't have domain knowledge.
- From following example, consider these points:

1. From layer 1, we will select only that feature which will have highest accuracy, for example feature 2 has highest accuracy
2. Then we will merge all remaining features, with the highest selected features, ie feature 3 (accuracy = 75%)
3. From layer 2, we will select features set, for example feature set (feature 3 + feature 1), only if it will have higher accuracy than 75%, otherwise we will move to the next step with feature 3 only
4.

18.2 Backward Elimination

- it move oppostie, first groups of multiple features are carried further having good accuracy score, then remove one feature and move and so on

18.3 Implementation

In [12]:
```
import pandas as pd
from mlxtend.feature_selection import SequentialFeatureSelector
```

In [15]:
```
dataset = pd.read_csv('diabetes.csv')
dataset.head(3)
```

Out[15]:

	Glucose	BloodPressure	SkinThickness	BMI	Age	Outcome
0	148	72	35	33.6	50	1
1	85	66	29	26.6	31	0
2	183	64	0	23.3	32	1

In [18]:
```
x = dataset.iloc[:,:-1]
x.head(3)
```

Out[18]:

	Glucose	BloodPressure	SkinThickness	BMI	Age
0	148	72	35	33.6	50
1	85	66	29	26.6	31
2	183	64	0	23.3	32

In [19]:
```
y = dataset['Outcome']
y.head(3)
```

Out[19]:
```
0    1
1    0
2    1
Name: Outcome, dtype: int64
```

In [22]:
```
x.shape
```

Out[22]:
```
(768, 5)
```

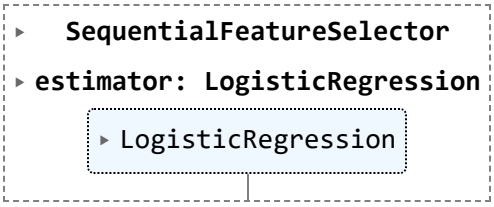
There are 5 features

In [20]:
```
from sklearn.linear_model import LogisticRegression
```

In [21]:
```
lr = LogisticRegression()
```

In [24]:
```
#fs = SequentialFeatureSelector(estimator, k_feature, )
fs = SequentialFeatureSelector(lr, k_features=5, forward=True)
fs.fit(x,y)
```

Out[24]:



In [25]:
```
fs.feature_names
```

Out[25]:
```
['Glucose', 'BloodPressure', 'SkinThickness', 'BMI', 'Age']
```

In [27]:
```
fs.k_feature_names_
```

Out[27]:
```
('Glucose', 'BloodPressure', 'SkinThickness', 'BMI', 'Age')
```

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

Out[28]:
```
0.7682794329853152
```

In [ ]:
```
5 - 0.7682794329853152
```

Now we will select 4 features and see accuracy and then 3 features and see its accuracy and so on..

In [29]:
```
fs = SequentialFeatureSelector(lr, k_features=4, forward=True)
fs.fit(x,y)
```

Out[29]:
SequentialFeatureSelector
estimator: LogisticRegression
LogisticRegression

In [30]:
```
fs.k_feature_names_
```

Out[30]:
```
('Glucose', 'BloodPressure', 'BMI', 'Age')
```

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

Out[31]:
```
0.7682709447415329
```

In [32]:
```
fs = SequentialFeatureSelector(lr, k_features=3, forward=True)
fs.fit(x,y)
```

Out[32]:
SequentialFeatureSelector
estimator: LogisticRegression
LogisticRegression

In [33]:
```
fs.k_feature_names_
```

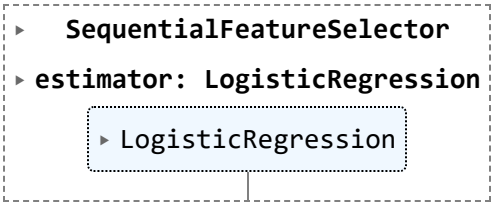
Out[33]:
```
('Glucose', 'BMI', 'Age')
```

In [34]:
```
fs.k_score_
```

Out[34]:
```
0.7683048977166624
```

In [35]:
```
fs = SequentialFeatureSelector(lr, k_features=2, forward=True)
fs.fit(x,y)
```

Out[35]:



In [36]:
```
fs.k_feature_names_
```

Out[36]:
```
('Glucose', 'BMI')
```

In [37]:
```
fs.k_score_
```

Out[37]:
```
0.7591206179441474
```