Feature Selection :

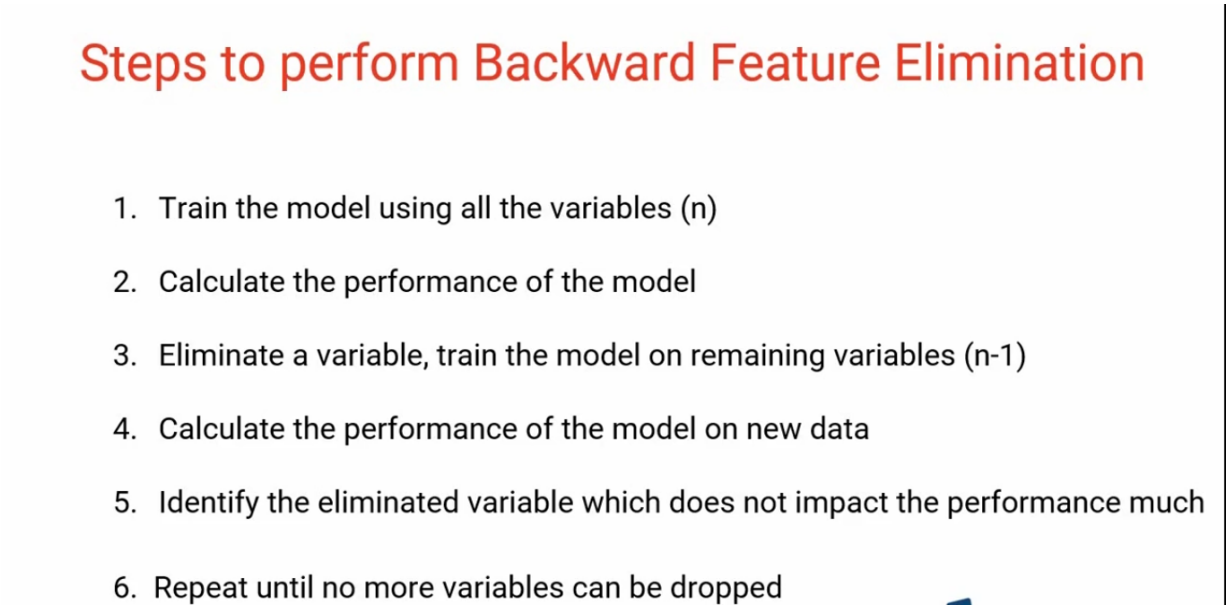
Here I covered two Feature slection Techniques That are

1. Backward Feature Elimination
2. Forward Feature Selection

When It Use ??

- When No missing Values in Dataset.
- Variance of all the variable are high.
- Low Correlation between independents variable.

These are the assumption , when it occure then we go for Backward Feature elimination or Forward Feature Selection.



You Nedd to install mlxtend library to perform these two opeartion

- **!pip install mlxtend**

1. Backward Feature Elimination Implementation

- In backward elimination, we start with all the features and removes the least significant feature at each iteration which improves the performance of the model. We repeat this until no improvement is observed on removal of features.

In [1]:
```
import warnings
warnings.simplefilter('ignore')
#importing the libraries
import pandas as pd
#reading the file
data = pd.read_csv('backward_feature_elimination.csv')
#shape of the data
print(data.shape)
# first 5 rows of the data
data.head()
```

```
(12980, 9)
```

Out[1]:

	ID	season	holiday	workingday	weather	temp	humidity	windspeed	count
0	AB101	1	0	0	1	9.84	81	0.0	16
1	AB102	1	0	0	1	9.02	80	0.0	40
2	AB103	1	0	0	1	9.02	80	0.0	32
3	AB104	1	0	0	1	9.84	75	0.0	13
4	AB105	1	0	0	1	9.84	75	0.0	1

In [2]:
```
# checking missing values in the data
data.isnull().sum()
```

Out[2]:
```
ID            0
season        0
holiday       0
workingday    0
weather       0
temp          0
humidity      0
windspeed     0
count         0
dtype: int64
```

- See here No Missing Value

In [3]:
```
# creating the training data
X = data.drop(['ID','count'], axis=1)
y = data['count']
```

In [4]:
```
X.shape, y.shape
```

Out[4]:
```
((12980, 7), (12980,))
```

In [5]:
```
from mlxtend.feature_selection import SequentialFeatureSelector as sfs
from sklearn.linear_model import LinearRegression
```

In [6]:
```
lreg = LinearRegression()
sfs1 = sfs(lreg, k_features=5, forward=False, verbose=1, scoring='neg_mean_square
```

- in sfs

  1st params = model_name

  2nd params= how many feature you want to select

  3rd params = forward false means it use backward feature eliminition

  4th params= verbose is for print score with each itration

  5th params = scoring is the score.

In [7]:
```
sfs1 = sfs1.fit(X, y)
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worker
s.
[Parallel(n_jobs=1)]: Done   7 out of   7 | elapsed:    0.1s finished
Features: 6/5[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concu
rrent workers.
[Parallel(n_jobs=1)]: Done   6 out of   6 | elapsed:    0.1s finished
Features: 5/5
```

In [8]:
```
feat_names = list(sfs1.k_feature_names_)
print(feat_names) #it return that it select
```

```
['holiday', 'workingday', 'weather', 'temp', 'humidity']
```

In [9]:
```
new_data = data[feat_names]
new_data['count'] = data['count']
```

In [10]:
```
# first five rows of the new data
new_data.head()
```

Out[10]:

	holiday	workingday	weather	temp	humidity	count
0	0	0	1	9.84	81	16
1	0	0	1	9.02	80	40
2	0	0	1	9.02	80	32
3	0	0	1	9.84	75	13
4	0	0	1	9.84	75	1

In [11]:
```
# shape of new and original data
new_data.shape, data.shape
```

Out[11]:
```
((12980, 6), (12980, 9))
```

- See here new_data column is 6 as backward feature elimination eliminate 3 feature.

2. Forward_feature_selection

Steps to perform Forward Feature Selection

1. Train n model using each feature (n) individually and check the performance
2. Choose the variable which gives the best performance
3. Repeat the process and add one variable at a time
4. Variable producing the highest improvement is retained
5. Repeat the entire process until there is no significant improvement in the model's performance

In [12]:
```
#importing the libraries
import pandas as pd
#reading the file
data = pd.read_csv('forward_feature_selection.csv')
#shape of the data
print(data.shape)
# first 5 rows of the data
data.head()
```

```
(12980, 9)
```

Out[12]:

	ID	season	holiday	workingday	weather	temp	humidity	windspeed	count
0	AB101	1	0	0	1	9.84	81	0.0	16
1	AB102	1	0	0	1	9.02	80	0.0	40
2	AB103	1	0	0	1	9.02	80	0.0	32
3	AB104	1	0	0	1	9.84	75	0.0	13
4	AB105	1	0	0	1	9.84	75	0.0	1

In [13]:
```
# checking missing values in the data
data.isnull().sum()
```

Out[13]:
```
ID            0
season        0
holiday       0
workingday    0
weather       0
temp          0
humidity      0
windspeed     0
count         0
dtype: int64
```

In [14]:
```
# creating the training data
X = data.drop(['ID', 'count'], axis=1)
y = data['count']
```

In [15]:
```
X.shape, y.shape
```

Out[15]:
```
((12980, 7), (12980,))
```

In [16]:
```
# importing the models
from mlxtend.feature_selection import SequentialFeatureSelector as sfs
from sklearn.linear_model import LinearRegression
```

In [17]:
```
# calling the linear regression model
lreg = LinearRegression()
sfs1 = sfs(lreg, k_features=4, forward=True, verbose=2, scoring='neg_mean_squared
```

- Here all are same as backward feature elimination but in forward we write True here so it use forward feature selection
- in sfs

  1st params = model_name

  2nd params= how many feature you want to select

  3rd params = forward true as it use forward feature selection

  4th params= verbose is for print score with each itration

  5th params = scoring is the score.

In [18]:
```
sfs1 = sfs1.fit(X, y)
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent worker
s.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   7 out of   7 | elapsed:    0.1s finished

[2020-10-23 10:33:57] Features: 1/4 -- score: -23364.955503181[Parallel(n_jobs=
1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   6 out of   6 | elapsed:    0.0s finished

[2020-10-23 10:33:57] Features: 2/4 -- score: -21454.899339219788[Parallel(n_jo
bs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.0s finished

[2020-10-23 10:33:57] Features: 3/4 -- score: -21458.278788564392[Parallel(n_jo
bs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    0.0s finished

[2020-10-23 10:33:57] Features: 4/4 -- score: -21477.98857350279
```

In [19]:
```
feat_names = list(sfs1.k_feature_names_)
print(feat_names)
```

```
['holiday', 'workingday', 'temp', 'humidity']
```

In [20]:
```
# creating a new dataframe using the above variables and adding the target variab
new_data = data[feat_names]
new_data['count'] = data['count']
# first five rows of the new data
new_data.head()
```

Out[20]:

	holiday	workingday	temp	humidity	count
0	0	0	9.84	81	16
1	0	0	9.02	80	40
2	0	0	9.02	80	32
3	0	0	9.84	75	13
4	0	0	9.84	75	1

In [21]:
```
# shape of new and original data
new_data.shape, data.shape
```

Out[21]:
```
((12980, 5), (12980, 9))
```

- By this we can reduce our dimensionality
- Till now we study all feature selection method, in my next notebook i am going to describe feature extraction method where most famous feature extraction are PCA,t-sne are used.