2.3 Recursive Feature elimination:

It is a greedy optimization algorithm which aims to find the best performing feature subset. It repeatedly creates models and keeps aside the best or the worst performing feature at each iteration. It constructs the next model with the left features until all the features are exhausted. It then ranks the features based on the order of their elimination.

In [1]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
```

In [2]:

```
data = pd.read_csv('paribas.csv',nrows =20000)
```

In [3]:

```
# feature selection using corr()

num_cols = ['int16','int32','int64','float16','float32','float64']

numerical_columns = list(data.select_dtypes(include = num_cols).columns)
data = data[numerical_columns]
```

In [4]:

```
correlated_features = set()
correlation_matrix = data.corr()
```

In [5]:

```
for i in range(len(correlation_matrix.columns)):
    for j in range(i):
        if abs(correlation_matrix.iloc[i,j]) > 0.8:
            col_name = correlation_matrix.columns[i]
            correlated_features.add(col_name)
```

In [6]:

```
train_features,test_features,train_labels,test_labels = train_test_split(
data.drop(labels = ['target','ID'],axis =1),
data['target'],
random_state = 41,
test_size=0.2)
```

In [7]:

```
train_features.drop(labels = correlated_features , axis =1 , inplace =True)
test_features.drop(labels = correlated_features , axis =1 , inplace =True)
```

In [8]:

```
train_features.shape
```

Out[8]:

```
(16000, 57)
```

RECURSIVE FEATURE ELIMINATION

our algorithm automatically chosses the features we donot include k_features parameter here

In [29]:

```
from sklearn.feature_selection import RFECV
from sklearn.metrics import roc_auc_score
from sklearn.ensemble import RandomForestClassifier as rfc
```

In [38]:

```
feature_selector = RFECV(estimator = rfc(n_jobs=1),
                      cv = 10,
                      verbose = 2,
                       scoring = 'roc_auc'
                      )
```

In [39]:

```
features = feature_selector.fit(np.array(train_features.fillna(0)) , train_labels)
```

```
estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)

Fitting estimator with 12 features.

c:\users\dell\appdata\local\programs\python\python36\lib\site-packages
\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n_
estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)

Fitting estimator with 11 features.

c:\users\dell\appdata\local\programs\python\python36\lib\site-packages
\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n_
estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)

Fitting estimator with 10 features.

c:\users\dell\appdata\local\programs\python\python36\lib\site-packages
\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n_
estimators will change from 10 in version 0.20 to 100 in 0.22.
```

In [43]:

```
filtered_features = train_features.columns[features.support_ ]
len(filtered_features)
```

Out[43]:

```
12
```

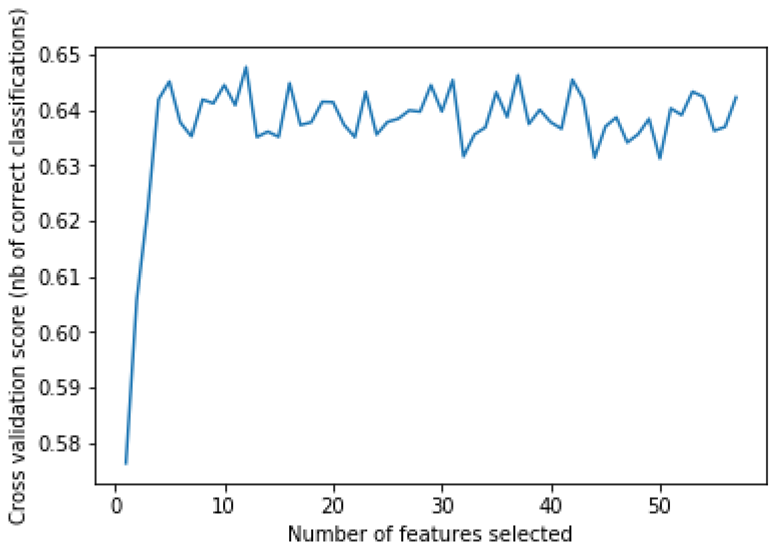
In [41]:

```
clf = RandomForestClassifier(n_estimators=100, random_state=41, max_depth=3)
clf.fit(train_features[filtered_features].fillna(0), train_labels)

train_pred = clf.predict_proba(train_features[filtered_features].fillna(0))
print('Accuracy on train data set {}'.format(roc_auc_score(train_labels , train_pred[:,

test_pred = clf.predict_proba(test_features[filtered_features].fillna(0))
print('Accuracy on test dataset {}'.format(roc_auc_score(test_labels,test_pred[:,1])))
```

```
Accuracy on train data set 0.7027906573527073
Accuracy on test dataset 0.7087839516824849
```

In [46]:

```
%matplotlib inline
import matplotlib.pyplot as plt
plt.figure()
plt.xlabel("Number of features selected")
plt.ylabel("Cross validation score (nb of correct classifications)")
plt.plot(range(1, len(features.grid_scores_) + 1), features.grid_scores_)
plt.show()
```



links for the methods() used

***RandomForestClassifier():* [*https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html)*](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html)**

RFECV () :https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFECV.html (https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFECV.html)