Feature Selection And Classification

Import Data

In [2]:
```
import pandas as pd
import numpy as np

%matplotlib notebook
```

In [3]:
```
data = pd.read_csv("data_merged.csv", sep=";")
```

In [4]:
```
data.head()
```

Out[4]:

	uuid	Act_Raw_Score	ActieveCopingPercentage	ActieveCopingScore	Age	AlgIntakeOpleidingsniveauScore_Raw	AlgIntake
0	-9214014786609792531	16.0	38.89	26.0	44.0	1.0	4.0
1	-9204323589684605317	14.0	47.22	29.0	40.0	6.0	1.0
2	-9189315961929324040	18.0	61.11	34.0	30.0	5.0	2.0
3	-9187839909081422277	18.0	72.22	38.0	48.0	9.0	4.0
4	-9184078185923068786	16.0	55.56	32.0	69.0	3.0	3.0

5 rows × 67 columns

Feature Selection

In [10]:
```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
import math
```

In [11]:
```
# Separate dataframe into scores, go (yes:no) and finished (yes/no) labels
X = data.iloc[:,1:-3]
s = data.iloc[:,-3]
y= data.iloc[:,-1]
```

In [12]:
```
# Fill in null values
X = X.fillna(X.mean().apply(lambda x: math.floor(x)))
X = X.astype(np.float64)
X.describe()
```

Out[12]:

	Act_Raw_Score	ActieveCopingPercentage	ActieveCopingScore	Age	AlgIntakeOpleidingsniveauScore_Raw	AlgIntakeWoonsituat
count	2376.000000	2376.000000	2376.000000	2376.000000	2376.000000	2376.000000
mean	17.650673	44.314373	27.948232	48.234428	4.874158	3.948653
std	3.786628	13.244941	4.768698	14.487983	2.175594	10.818069
min	7.000000	0.000000	12.000000	18.000000	1.000000	0.000000
25%	15.000000	36.110000	25.000000	37.000000	3.000000	2.000000
50%	18.000000	44.440000	28.000000	49.000000	5.000000	3.000000
75%	20.000000	52.780000	31.000000	59.000000	7.000000	4.000000
max	28.000000	88.890000	44.000000	89.000000	10.000000	99.000000

8 rows × 64 columns

Pipeline: Recursive Feature Elimination with Cross Validation + GridSearchCV

In [6]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import StratifiedKFold
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier

from yellowbrick.features import RFECV
import matplotlib.pyplot as plt
```

In [7]:
```
# Extend Pipeline class to get access to the features' importance of the model
class PipelineRFE(Pipeline):

    def fit(self, X, y=None, **fit_params):
        super(PipelineRFE, self).fit(X, y, **fit_params)
        self.feature_importances_ = self.named_steps['RFC'].feature_importances_
#         self.support_ = self.named_steps['RFC'].support_
        return self
```

In [8]:
```
pipeline = [
    ('scaler', StandardScaler()),
    ('RFC', RandomForestClassifier(class_weight="balanced", n_estimators=150))
]
estimator = PipelineRFE(pipeline)
```

In [25]:
```
# Stratified cross validation for class imbalance
cv = StratifiedKFold(2)

## Recursive Feature Elimination (each take about 5 minutes to complete given the current parameters)

# ROC AUC --> BLUE
clf_roc = RFECV(estimator, step=5, cv=cv, scoring="roc_auc")
clf_roc.fit(X, s)
clf_roc.finalize()

# Recall --> GREEN
clf_recall = RFECV(estimator, step=5, cv=cv, scoring="recall")
clf_recall.fit(X, s)
clf_recall.finalize()

# Precision --> RED
clf_precision = RFECV(estimator, step=5, cv=cv, scoring="precision")
clf_precision.fit(X, s)
clf_precision.finalize()

# Accuracy --> PURPLE
clf_accuracy = RFECV(estimator, step=5, cv=cv, scoring="accuracy")
clf_accuracy.fit(X, s)
clf_accuracy.finalize()
```