In [57]:
```
## KNN multi-label
```

In [58]:
```
% matplotlib inline

from IPython.core.display import display, HTML
display(HTML("<style>.container { width:90% !important; }</style>"))
import numpy as np
import pandas as pd
pd.set_option('display.max_columns', 100)

df = pd.read_csv('User_Knowledge.csv')

df.loc[df.UNS == 'very_low','grade'] = 0
df.loc[df.UNS == 'Low','grade'] = 1
df.loc[df.UNS == 'Middle','grade'] = 2
df.loc[df.UNS == 'High','grade'] = 3

df.sample(5)
```

Out[58]:

	STG	SCG	STR	LPR	PEG	UNS	grade
115	0.285	0.640	0.18	0.61	0.45	Middle	2.0
158	0.465	0.258	0.73	0.18	0.59	Middle	2.0
131	0.400	0.180	0.26	0.26	0.67	Middle	2.0
186	0.495	0.820	0.67	0.01	0.93	High	3.0
254	0.780	0.610	0.71	0.19	0.60	Middle	2.0

Attribute Information

- STG (The degree of study time for goal object materails), (input value)
- SCG (The degree of repetition number of user for goal object materails) (input value)
- STR (The degree of study time of user for related objects with goal object) (input value)
- LPR (The exam performance of user for related objects with goal object) (input value)
- PEG (The exam performance of user for goal objects) (input value)
- UNS (The knowledge level of user) (target value)
  - Very Low: 50
  - Low:129
  - Middle: 122
  - High 130

In [59]:
```
y = list(df['UNS'])
y_grade = df.grade

# feature selection, dropping SCG
X = df.drop(columns=['SCG']).iloc[:,0:4]

X.sample()
```

Out[59]:

	STG	STR	LPR	PEG
173	0.4	0.58	0.75	0.16

In [60]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_
state=0)
sc = StandardScaler()
X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)
```

KNN

In [61]:
```
# Classifier implementing the k-nearest neighbors vote

from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
knn_pca = KNeighborsClassifier(n_neighbors=3)

knn.fit(X_train, y_train)
print("KNN Classfier",knn.score(X_test,y_test))
```

```
KNN Classfier 0.953488372093
```

In [62]:
```
from sklearn.metrics import confusion_matrix
label = ['very_low', 'Low','Middle','High']

print(confusion_matrix(y_test, knn.predict(X_test), label))
```

```
[[ 5  2  0  0]
 [ 0 32  0  0]
 [ 0  1 27  0]
 [ 0  0  1 18]]
```

KNN (PCA transformed)

In [63]:
```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_train_pca = pca.fit_transform(X_train)
X_test_pca = pca.transform(X_test)

knn_pca.fit(X_train_pca, y_train)
print("MLP (PCA transformed) accuracy: ", knn_pca.score(X_test_pca, y_test))
```

```
MLP (PCA transformed) accuracy:  0.523255813953
```

In [64]:
```
pd.Series(y_test).value_counts().head(1)/len(y_test)
```

Out[64]:
```
Low    0.372093
dtype: float64
```