```
# reducing features using principle components
# load Libraries
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA from
sklearn import datasets

# load the data
digits = datasets.load_digits()

# standardize the features matrix
X = StandardScaler().fit_transform(digits.data)

# create a pca that will retain 99% of the variance
pca = PCA(n_components=0.99, whiten=True)

# Conduct PCA
X_pca = pca.fit_transform(X)

# show PCA
print('original number of features:', X.shape[1])
print('Reduced number of features:', X_pca.shape[1])

original number of features: 64
Reduced number of features: 54

# Load libraries
from sklearn.decomposition import PCA, KernelPCA
from sklearn.datasets import make_circles

#create linearly inseparable data
x, _ = make_circles(n_samples=1000,random_state=1, noise=0.1,
factor=0.1)

# apply kernel pca with radius basis function (RBF) kernel
kpca = KernelPCA(kernel="rbf", gamma=15, n_components=1)
X_kcpa = kpca.fit_transform(X)

print('original number of features:', X.shape[1])
print('original number of features:', X_kcpa.shape[1])

original number of features: 64
original number of features: 1

from sklearn import datasets
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

# load the iris dataset
iris = datasets.load_iris()
X = iris.data
```

```
y = iris.target

# create an LDA that will reduce the data down to 1 feature
lda = LinearDiscriminantAnalysis(n_components=1)

# run LDA and transform the features
X_lda = lda.fit(X, y).transform(X)

# print the number of features
print('Original number of features:', X.shape[1])
print('Reduced number of features:', X_lda.shape[1])

Original number of features: 4
Reduced number of features: 1

# view the ration of explained variance
lda.explained_variance_ratio_
array([0.9912126])
```