CORK INSTITUTE OF TECHNOLOGY
INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

scikit learn

# Machine Learning

## Machine Learning

Lecture: Dimensionality Reduction

Ted Scully

# Dimensionality Reduction

- Dimensionality reduction is the process of converting a dataset that typically has a large number of features into a dataset with a **reduced number of features** (in some cases a significantly reduced number of features).

- This is **not** the same thing as **feature selection** as the reduced set of features may not be a subset of the original features.

- Dimensionality reduction algorithms such as PCA performs a **linear transformation** of *d* dimensional input to d dimensional feature vectors such that k < d, while attempting to that explain a maximum amount of the variance.

- Can be used very effectively for reducing the dimensionality when you have a very large number of features. A typical application would be image analysis.

# PCA

- PCA is by far the most popular method used for dimensionality reduction.
- The objective of the PCA algorithm is to find a surface on to which to project the data so as to **minimize the projection error**
- More specify we want to identify k vectors that minimize the projection error where the projection error is defined as:
- $\frac{1}{m}\sum_{i=1}^{m} | x^i - x^i_{mapped} |^2$
- Note the PCA expects that you data has been **standardized** in advance.

# PCA

- PCA is a parameterized algorithm. We specify in advance the a parameter value k, which specifies the number of features vectors the algorithm will return.

- Typically the selection of a value for k is made depending on the **retained variance from running PCA with the value k**. You can think of the retained variance as a measure of how effective the projection process employed by PCA.

- You will hear phrases such as selecting a value of k resulted in “99% of the variance retained”. Another common measure would be “95% of the variance retained”.

```
from sklearn import decomposition
from sklearn import datasets
from sklearn import cross_validation
from sklearn import tree
from sklearn.preprocessing import StandardScaler
import numpy as np

iris = datasets.load_iris()
X = iris.data
y = iris.target

scaler = StandardScaler()
X = scaler.fit_transform(X)

clf = tree.DecisionTreeClassifier()
scores = cross_validation.cross_val_score(clf, X, iris.target, cv=10)
print ("Original Accuracy", np.mean(scores))

pca = decomposition.PCA(n_components=2)
pca.fit(X)
X = pca.transform(X)
print ("Explained Variance: ",np.sum(pca.explained_variance_ratio_))

clf = tree.DecisionTreeClassifier()
scores = cross_validation.cross_val_score(clf, X, iris.target, cv=10)

print (np.mean(scores))
```

Original Accuracy 0.95
Explained Variance:  0.958
Accuracy Using Two New Features  0.91