Lab 2

Classification with MAP criterion
PCA vs MDA feature selection
Synthetic & PHONEME dataset

UPC
telecom BCN
Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Lab2

Objectives:

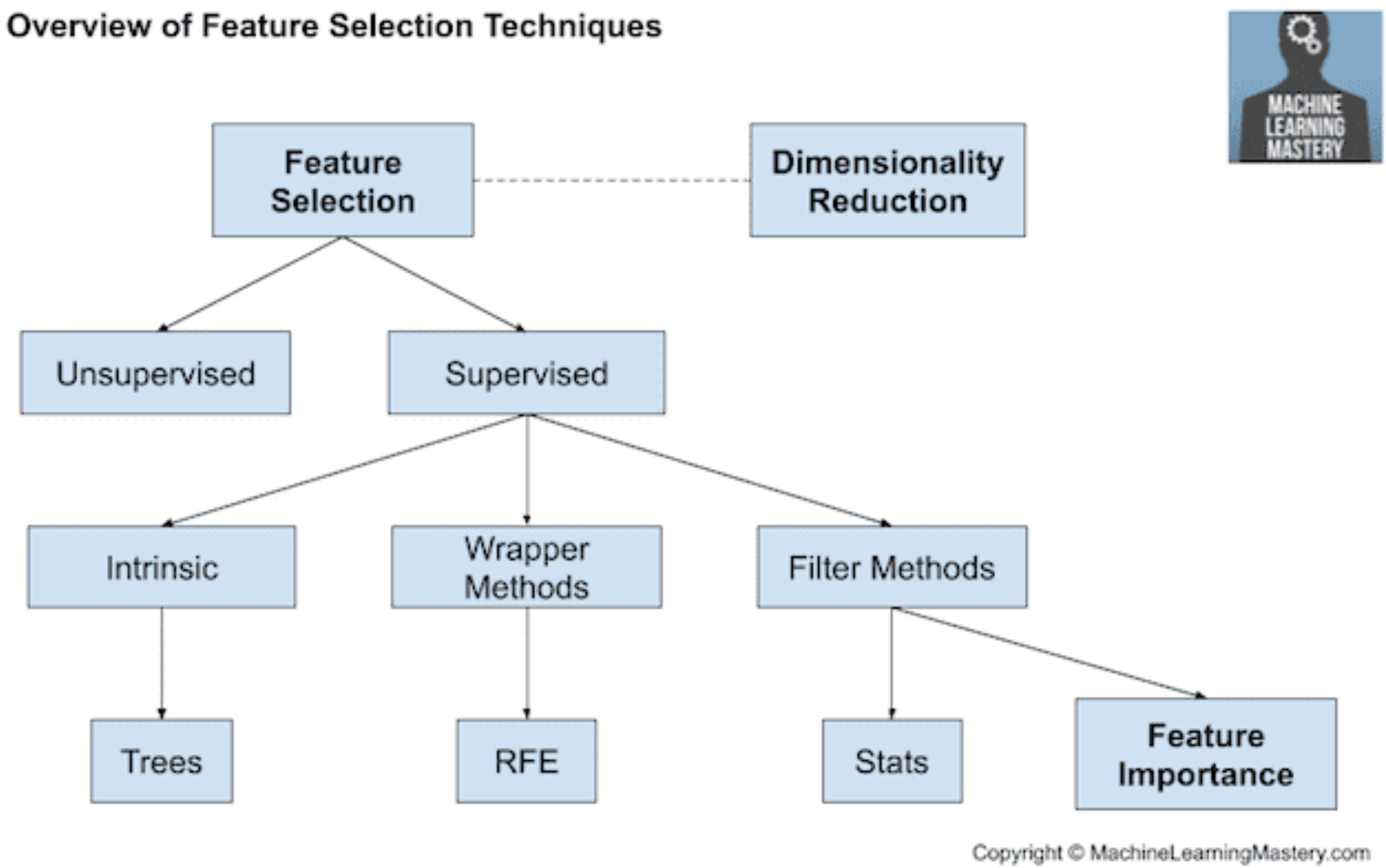
- Dimensionality reduction (feature selection) using PCA and MDA
- Application to a real dataset
- Split the dataset into training and test subsets

Feature selection is the process of selecting a subset of relevant features for use in model construction.

Feature selection techniques help to

- Simplify the classifier model
- Reduce the computational cost / training times
- Avoid the curse of dimensionality
- Improve generalization by reducing overfitting

Feature selection techniques



See https://scikit-learn.org/stable/modules/feature_selection.html

Dimensionality reduction through a linear transform

Goal: Reduce the number of features (assuming column vectors) :

$$\mathbf{z}_k = \mathbf{W}^T \left(\mathbf{x}_k - \boldsymbol{\alpha} \right) \qquad \mathbf{x} \in \mathbb{R}^d, \mathbf{z} \in \mathbb{R}^{d'}, \mathbf{W}^T \in \mathbb{R}^{d' \times d} \qquad d' < d$$

Possible solutions for W:

1. Projection of vectors x_k on subspace that minimizes the reconstruction error (MSE): principal component analysis (PCA)

2. Projection of vectors x_k on the subspace that maximizes the separation between classes: multiple discriminant analysis (MDA)

Take into account

The reduction matrix **W** must be created using the training dataset

Scatter matrix

$$\mathbf{m}_i = \frac{1}{N_i}\sum_{\mathbf{x}\in D_i}\mathbf{x}$$ average of samples from class i

$$\mathbf{m} = \frac{1}{N}\sum_{\mathbf{x}\in\{D_1,\dots,D_c\}}\mathbf{x} = \frac{1}{N}\sum_{i=1}^{c}N_i\mathbf{m}_i$$ average of all samples

$$\mathbf{S}_T = \sum_{\mathbf{x}\in\{D_1,\dots,D_c\}}(\mathbf{x}-\mathbf{m})(\mathbf{x}-\mathbf{m})^T$$ Total data dispersion

---

$$\mathbf{S}_T = \underbrace{\sum_{i=1}^{c}\sum_{\mathbf{x}\in D_i}(\mathbf{x}-\mathbf{m}_i)(\mathbf{x}-\mathbf{m}_i)^T}_{\mathbf{S}_C} + \underbrace{\sum_{i=1}^{c}\sum_{\mathbf{x}\in D_i}(\mathbf{m}_i-\mathbf{m})(\mathbf{m}_i-\mathbf{m})^T}_{\mathbf{S}_B}$$

Sum of intra-class scatter matrices

Inter-class scatter matrix

PCA (Principal Component Analysis)

Objective:

- Maximize: $$\sum_{i=1}^{d'} w_i^T S_T w_i$$
- Constraints: $$w_i^T w_i = E$$

Solution:

- Columns of **W**: d' eigenvectors associated with the largest eigenvalues of $\mathbf{S}_T$

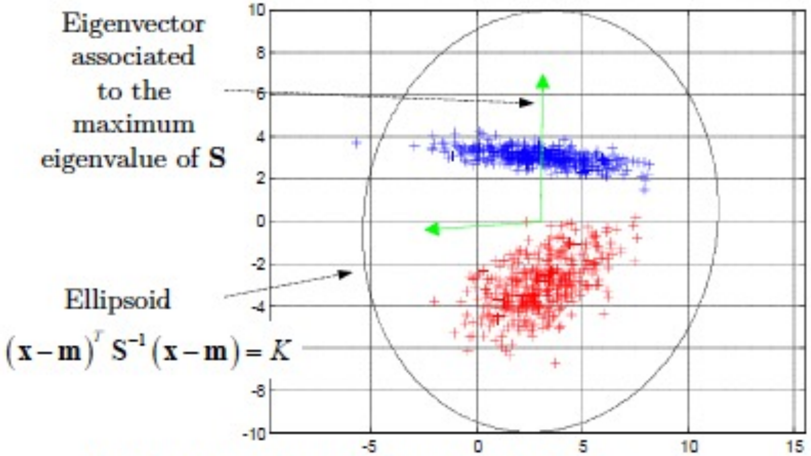
$$\mathbf{S}_T \mathbf{w}_i = \lambda_i \mathbf{w}_i$$

Problem:

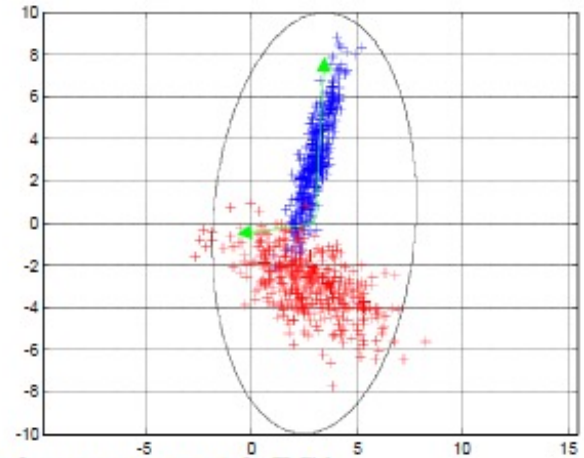
- PCA minimizes the approximation squared error but it does not guarantee the separability of the classes

PCA (Principal Component Analysis)

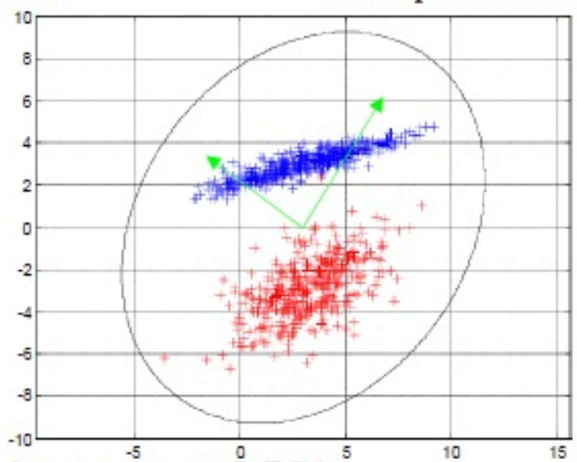
PCA does not guarantee a good separation of classes



Good separation of classes with PCA



Bad separation with PCA



Bad separation with PCA

MDA (Multiple Discriminant Analysis)

Objective:

- Maximize intra-class separability while minimizing the inter-class scatter
- We measure the separability and scatter using the ellipsoid volumes, assuming data Gaussianity

Formulation:

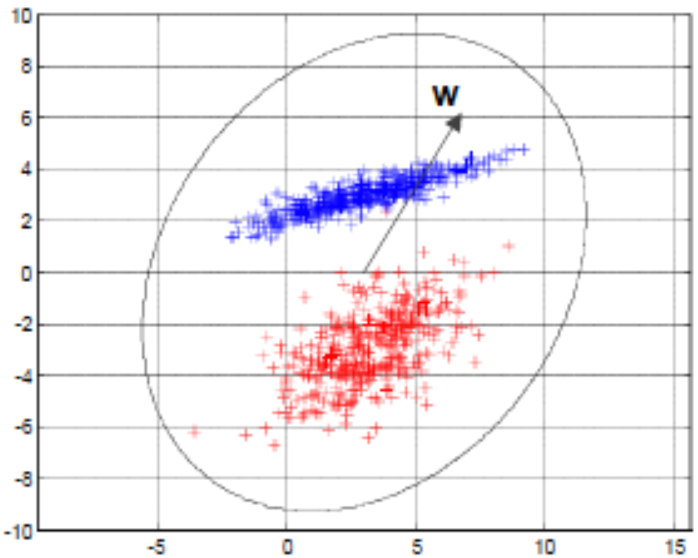
- Maximization: $\mathbf{W} = \arg\max_{\mathbf{W}} \dfrac{\left|\mathbf{W}^T \mathbf{S}_B \mathbf{W}\right|}{\left|\mathbf{W}^T \mathbf{S}_C \mathbf{W}\right|}$

Solution:

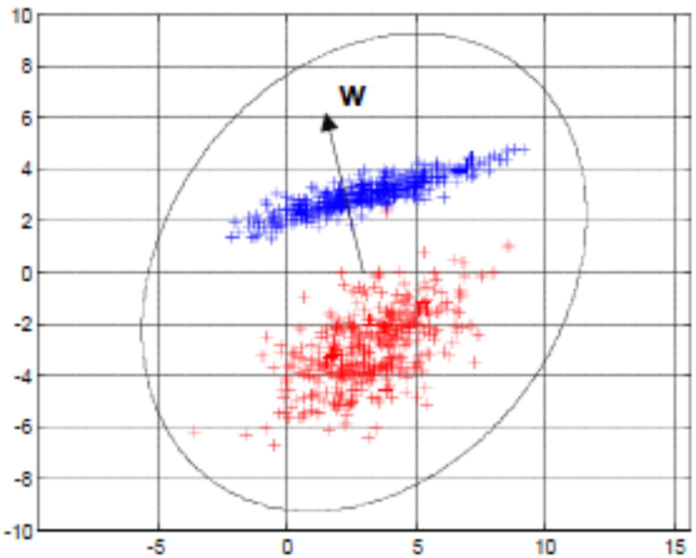
- d’ ≤ min(d,c-1) (c: number of classes)
- **W columns:** eigenvectors associated to the largest eigenvalues:

$$\mathbf{S}_B \mathbf{w}_j = \sigma_j \mathbf{S}_C \mathbf{w}_j \qquad \Rightarrow \qquad \mathbf{S}_C^{-1} \mathbf{S}_B \mathbf{w}_j = \sigma_j \mathbf{w}_j$$

MDA (Multiple Discriminant Analysis)



Bad separation with PCA when projecting onto w



Better separation when projecting onto w

PCA in scikit-learn

n_components = number of components to keep.

```
pca = PCA(n_components=2)
pca.fit(X_train)
X_train_pca1 = pca.transform(X_train)
X_test_pca1 = pca.transform(X_test)
```

If n_components is not set all components are kept

```
pca = PCA()
pca.fit(X_train)
X_train_pca = pca.transform(X_train)
X_test_pca = pca.transform(X_test)
```

MDA in scikit-learn

Linear Discriminant Analysis (LDA ->MDA)

A classifier with a linear decision boundary, generated by fitting class conditional densities to the data and using Bayes' rule. The model fits a Gaussian density to each class, assuming that all classes share the same covariance matrix.

```
lda = LinearDiscriminantAnalysis(solver="svd",store_covariance=True)
ldamodel = lda.fit(X_train, y_train)
y_tpred_lda = ldamodel.predict(X_train)
y_testpred_lda = ldamodel.predict(X_test)
```

The fitted model can also be used to reduce the dimensionality of the input by projecting it to the most discriminative directions, using the transform method.

```
mda = LinearDiscriminantAnalysis(n_components=2)
mda.fit(X_train, y_train)
X_train_mda2 = mda.transform(X_train)
X_test_mda2 = mda.transform(X_test)
```

```
mda = LinearDiscriminantAnalysis()
mda.fit(X_train, y_train)
X_train_mda =
mda.transform(X_train)
X_test_mda = mda.transform(X_test)
```

Phoneme dataset

- A dataset was formed by selecting five phonemes for classification based on digitized speech from TIMIT database (speech recognition)

- Vectors correspond to 5 possible phonemes or classes:
  'aa' (695) 'ao'(1022) 'dcl'(757) 'iy'(1163) 'sh'(872).

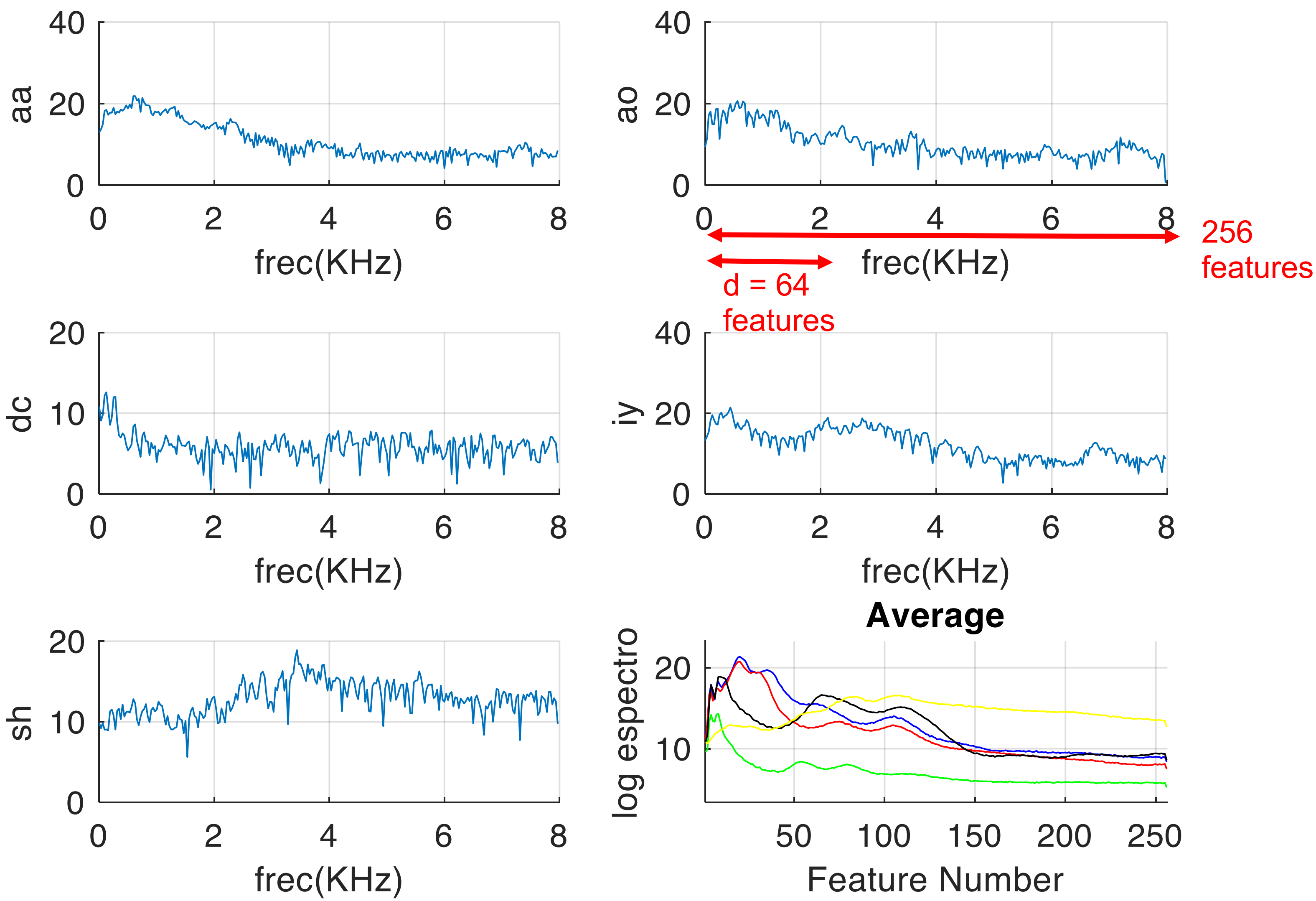
- Each vector has been obtained computing $log\left(\left|TF\left(x(n)\right)\right|^2\right)$ where the sequence $x(n)$ corresponds to part of a recording of a phoneme at a sampling rate of 16 kHz.

- For each vector we initially have 256 features, corresponding to the spectrum between 0 and 8 kHz

- In Lab2 we will work just with the first 64 samples (frequencies 0 to 2 kHz).

https://web.stanford.edu/~hastie/ElemStatLearn/data.html

Example: one vector per class



Data / Classifier design

Dataset split

- Training set X_train y_train (70%)
- Test set X_test, y_test (30%)

Design of a linear (LC) and a quadratic (QC) classifier

- Dataset with d=256 (or 64) features
- Reduced dataset using manual selection of 2 features
- Reduced dataset (dimension d') using PCA

Lab2

Part1 (Mlearn_lab2_1_IntroPCA.ipynb)

- Understand the use of PCA for dimensionality reduction (toy example, digits image dataset)

Part2 (Mlearn_lab2_2_Synthetic_PCA_MDA.ipynb)

- Use synthetic Gaussian datasets (c=3 classes, d=3 features) for different SNR values
- Train Lc and Qc classifiers using all the features
- Train Lc and Qc classifiers after dimensionality reduction using PCA and MDA

Part3 (Mlearn_lab2_3_Phoneme.ipynb)

- Use Phoneme dataset (c=5 classes, d=256 features)
- Train Lc and Qc classifiers using the first d=64 features
- Train Lc and Qc classifiers using d'=2 manually selected features

Part2 (your code) Mlearn_lab2_4_Phoneme_PCA_MDA_surname.ipynb

- Use Phoneme dataset
- Train Lc and Qc classifiers using d' features selected with PCA / MDA
- Show Lc and Qc training/test error curves for varying number of features selected with PCA / MDA