Dimensionality reduction

Feature extraction with **Linear Discriminant Analysis (LDA)**

see Ch. 05 in book “Python Machine Learning” by Raschka & Mirjalili

Agenda

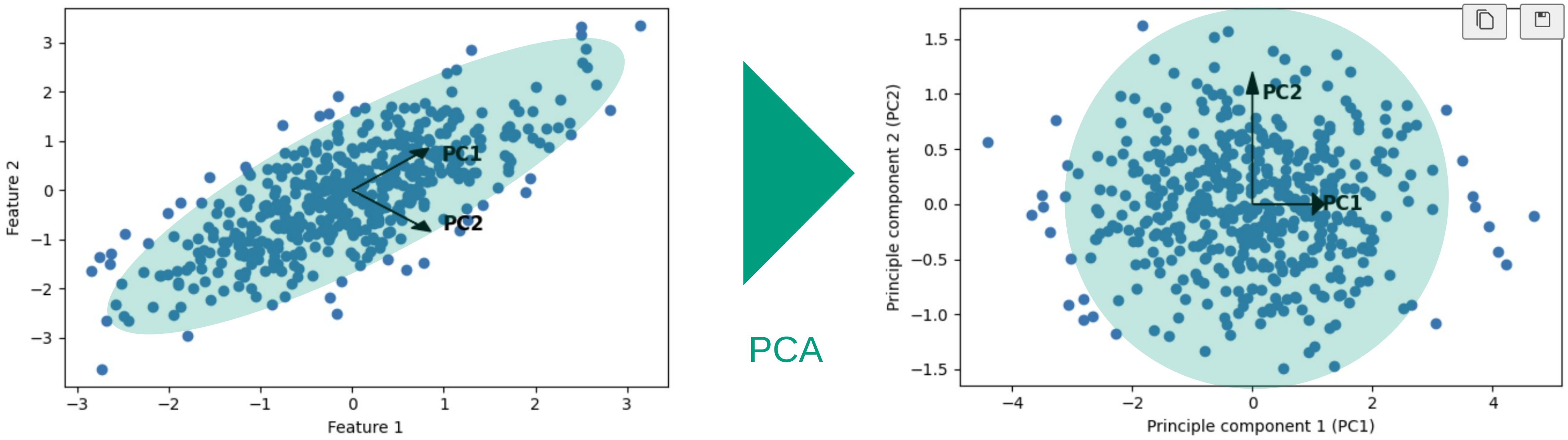
- Short recap of PCA
- Round of Chapter 5 in the book with Linear Discriminant Analysis
- Start of Chapter 6 with Pipelines

RECAP – Dimensionality reduction with PCA

- Feature **selection** vs. feature **extraction**
  - Selection: **Maintains** a subset of the **original features**
  - Extraction: **transforms** or **projects** data into a feature **subspace** (constructs new features)
- Principle Component Analysis (PCA)
  - A **linear transformation** technique
  - **unsupervised learning** techniques
  - Often used for **dimensionality reduction** (**feature extraction**)

RECAP – Dimensionality reduction with PCA

- Aims to estimate the **dimensions** of the dataset with **greatest variance** (principle components)
- Can be computed as the **eigenvalue decomposition** of the **covariance matrix** of the dataset.



Feature extraction using Linear Discriminant Analysis

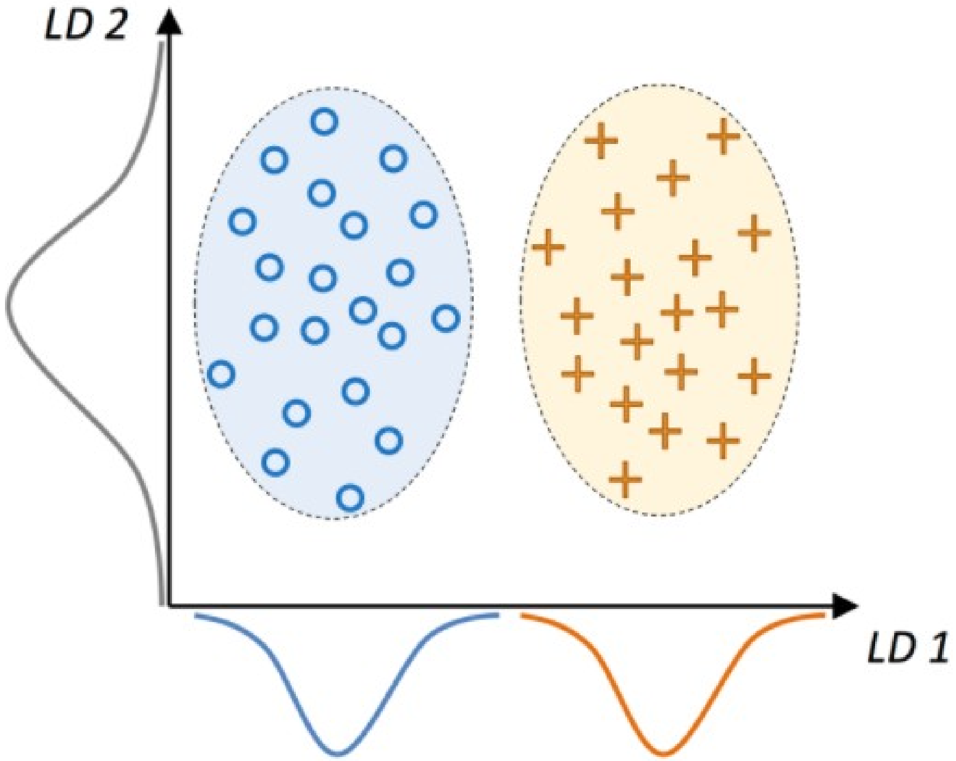
- What do you think *linear discriminant analysis* aims to accomplish?
- As the name suggests, we aim to **transform** the data into a **subspace** where it is as easy as possible to “**linearly discriminate**” between samples of different classes

Feature extraction using Linear Discriminant Analysis

- **Linear Discriminant Analysis (LDA)** can be used as a technique for feature extraction
  - **Increases** the computational **efficiency**
  - **Reduces overfitting** due to curse of dimensionality in models that are not regularized
- General concept behind LDA is **very similar** to PCA
  - Both are **linear transformation techniques** can be used for **feature extraction**
  - Both decompose matrices into **eigenvalues** and **eigenvectors** that form basis for new feature subspace
- However there are some **differences** as well
  - PCA is an **unsupervised** algorithm, LDA is a **supervised** algorithm
  - PCA aims to find orthogonal components of **basis** that **maximizes variance** in dataset
    LDA finds the **feature subspace** that optimizes **linear class separability**

Feature extraction using Linear Discriminant Analysis

- The concept of LDA
- In the binary classification example to the right a dataset has been projected onto two *linear discriminants*.
- Which of the **LD**’s would you choose to classify the dataset?



Feature extraction using Linear Discriminant Analysis

- Underlying assumptions for LDA:
  - The data is **normally distributed**
  - Individual classes have **identical covariance matrices**
  - The features of the dataset are **statistically independent** of one another
- However, if one or more of the assumptions are slightly violated, LDA can still work reasonably well for dimensionality reduction
- In the next slides we will look at the inner workings of LDA with an example
- Assume that you have a dataset with input matrix, $\boldsymbol{X}$, that has d number of features and a binary target, $\boldsymbol{y}$

Inner Workings of LDA

1. Standardize the dataset.
2. For each class, compute the *d*-dimensional mean vector.
3. Construct the between-class scatter matrix, S_B, and the within-class scatter matrix, S_W.
4. Compute the eigenvectors and corresponding eigenvalues of the matrix, $S_W^{-1}S_B$.
5. Sort the eigenvalues by decreasing order to rank the corresponding eigenvectors.
6. Choose the *k* eigenvectors that correspond to the *k* largest eigenvalues to construct a *d* x *k*-dimensional transformation matrix, ***W***; the eigenvectors are the columns of this matrix.
7. Project the samples of ***X*** onto the new feature subspace using the transformation matrix, ***W***.

$$x' = xW$$

LDA - Examples

- `LDA_from_scratch.ipynb`
  - Performing LDA from scratch
- `LDA_and_logistic_regression_with_sklearn.ipynb`
  - Performing LDA and then logistic regression on the feature-transformed dataset using the scikit-learn methods

PCA

Mean centered
data (or standard scaled)
data matrix
$\boldsymbol{X}$

EVD of Cov($\boldsymbol{X}$) to find the tranformation matrix

$\boldsymbol{x}' = \boldsymbol{x}\boldsymbol{W}$

$\boldsymbol{X}' = \boldsymbol{X}\boldsymbol{W}$

LDA

$$\boldsymbol{m}_i = \frac{1}{n_i}\sum_{\boldsymbol{x}\in D_i}^{c} \boldsymbol{x}_m$$

$$\boldsymbol{S}_i = \sum_{\boldsymbol{x}\in D_i}^{c} (\boldsymbol{x}-\boldsymbol{m}_i)(\boldsymbol{x}-\boldsymbol{m}_i)^T$$

$$\boldsymbol{S}_W = \sum_{i=1}^{c} \boldsymbol{S}_i$$

$$\boldsymbol{S}_B = \sum_{i=1}^{c} n_i(\boldsymbol{m}_i-\boldsymbol{m})(\boldsymbol{m}_i-\boldsymbol{m})^T$$

$\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$:

In case of unbalanced classes

$$\Sigma_i = \frac{1}{n_i}\boldsymbol{S}_W = \frac{1}{n_i}\sum_{\boldsymbol{x}\in D_i}^{c} (\boldsymbol{x}-\boldsymbol{m}_i)(\boldsymbol{x}-\boldsymbol{m}_i)^T$$

Limits number of features vs samples,

could combine with PCA.

Eigenvaluedecomp. of $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$: to find tranformation matrix

Max number of LD's:
c - 1

$\boldsymbol{x}' = \boldsymbol{x}\boldsymbol{W}$

$\boldsymbol{X}' = \boldsymbol{X}\boldsymbol{W}$

Thank you for listening