Machine Learning

Lecture 06

LDA (Linear Discriminant Analysis)

Introduction

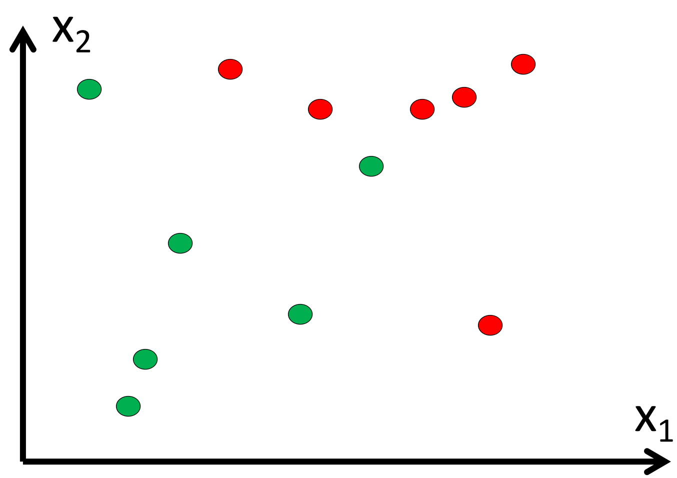
- LDA is one of the oldest, simple and powerful linear classifier.
- LDA classifies objects in two or more groups by forming a linear combination of features.
- LDA can also be used as a dimension reduction technique much like PCA.
  - PCA maximizes variance while LDA maximizes class separability.
  - PCA is an unsupervised learning method while LDA is supervised.
- Addresses some of the well-known shortcomings of Logistic Regression (LR):
  - LR is natively a binary classifier
  - LR can become unstable for perfectly separable classes
  - LR can become unstable with few instances of features

Preparing data for LDA

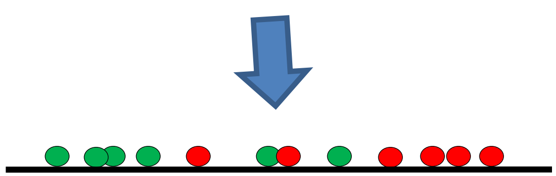
- Assumptions of LDA
  - **Normal distribution:** Attributes are normally distributed, i.e., univariate distribution of each feature follows a Gaussian. You might need to transform the data to make it look like Normal (log / Box Cox transformation).
  - **Equal Variance: E**qual variances for attributes across classes (same covariance matrix).
  - **Independence**: Exclusive and independent features with no perfect correlation
  - **No outliers:** As the outliers can skew the distribution, you might need to handle your outliers.
- LDA is quite robust to violations in assumptions. So, it can work reasonably well even when these assumptions are violated. LDA does feature scaling by design, so a separate scaling is not needed.

Class separation by LDA

- PCA searches for a projection in which the variance in data is maximum (later)
- LDA searches for a projection that maximizes the separability among classes
- **Example:** Reducing a 2D graph to a 1D graph



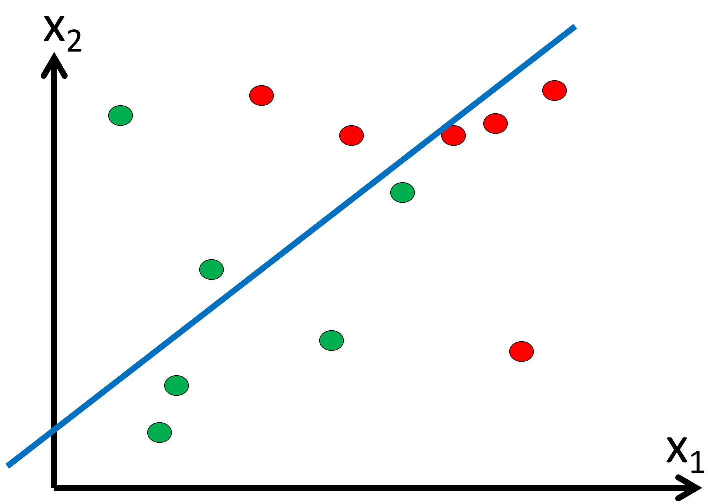
We simply ignore the useful information in attribute x_2 and use only x_1 to separate classes

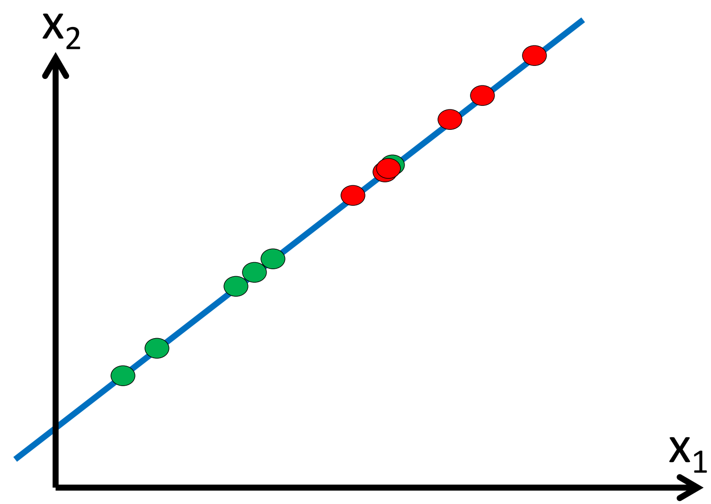


Projection of data points onto x_1

Class separation by LDA – cont'd

- Reducing a 2D graph to a 1D graph using LDA:



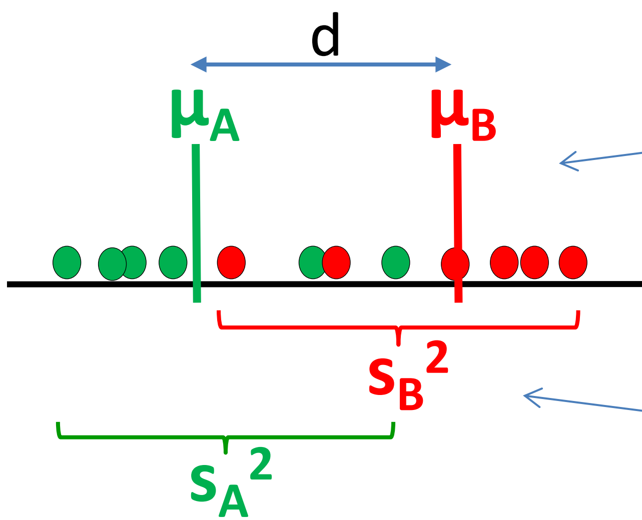


- LDA uses both features to create a new axis and projects the data onto this new axis in a way to maximize the separation between classes
- How does LDA do this?
  – By maximizing the distance between the means of classes (distance between projected means is not a good measure as it doesn't account for the variance within each class)
  – By minimizing the variance within each class

Class separation by LDA – cont'd

- Simultaneous satisfaction of 2 criteria (by R. Fisher):

1. Maximizing the distance "**d**" between means
2. Minimize the variation (called "scatter" by LDA and represented by $\mathbf{s^2}$) within each class



Means of the classes are as far away as possible from each other (maximize between-class scatter)

Data points of the same class are as much clustered-together as possible (minimize within-class scatter)

3. To satisfy both (1) and (2) simultaneously, optimize:

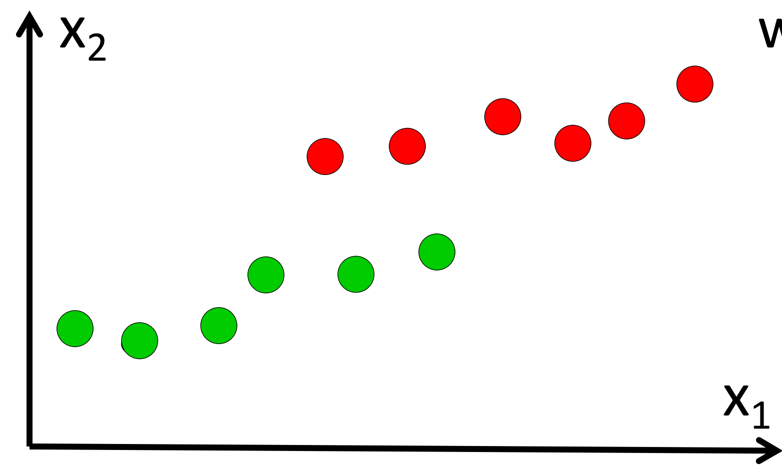
$$\frac{(\mu_A - \mu_B)^2}{s_A^2 + s_B^2}$$

numerator → ideally large

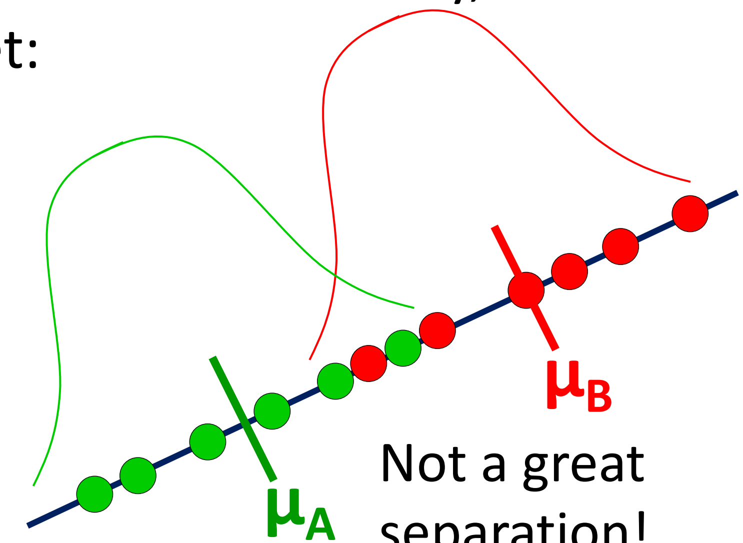
denominator → ideally small

Class separation by LDA – cont'd

- **Example**: Suppose a two-class problem has the following scatter plot:

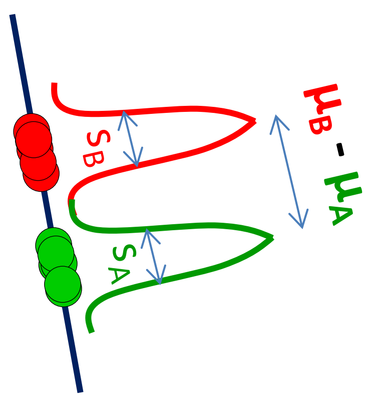


If we were to maximize the distance between the means only, this is what we get:



Not a great separation!

What if we maximize the distance d and minimize the scatter (variance) simultaneously:



We get a good separation…

LDA for 3 or more classes

- We're indeed optimizing the ratio of between-cluster scatter to within-cluster scatter:

The within-class scatter S_W:

scatter matrix for each class

Mean vector: $\boldsymbol{\mu}_\mathbf{i} = (1/N_i)\sum_{x\in D_i}^{n} \mathbf{x}_\mathbf{k}$

$$S_W = \sum_{i=1}^{c} S_i = \sum_{i=1}^{c}\left(\sum_{x\in D_i}(\mathbf{x}-\boldsymbol{\mu}_i)(\mathbf{x}-\boldsymbol{\mu}_i)^T\right)$$

The between-class scatter S_B:

size of the respective class

$$S_B = \sum_{i=1}^{c} N_i(\boldsymbol{\mu}_\mathbf{i}-\boldsymbol{\mu})(\boldsymbol{\mu}_\mathbf{i}-\boldsymbol{\mu})$$

sample mean overall mean

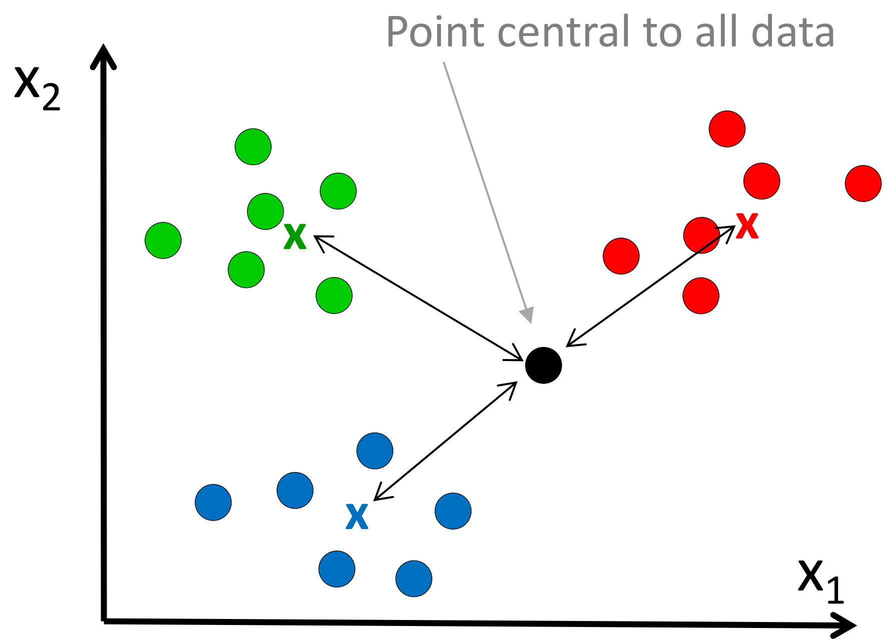
- We then solve for the eigenvalue problem for the matrix $S^{-1}{}_W S_B$ to obtain eigenvalues and eigenvectors.

Eigenvalue problem: $Av = \lambda v$ where $A = S_W{}^{-1} S_B$ and λ is the eigenvalue

- We'll pick the k largest eigenvalues and associated eigenvectors (linear discriminants) and will project the observations onto the subspace spanned by these vectors.

LDA for 3 or more classes

- The idea could easily be extended to multi-class problems:

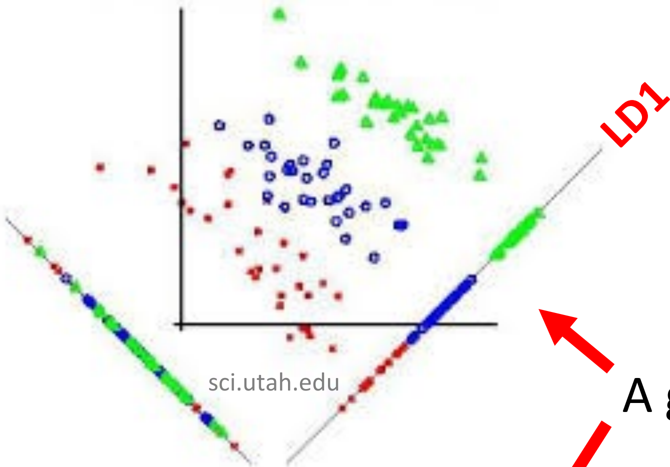


Maximize $$\frac{d_1^2 + d_2^2 + d_3^2}{s_1^2 + s_2^2 + s_3^2}$$

- We search for a space projection matrix W that maximizes the ratio: $|W^T S_B W| \,/\, |W^T S_W W|$

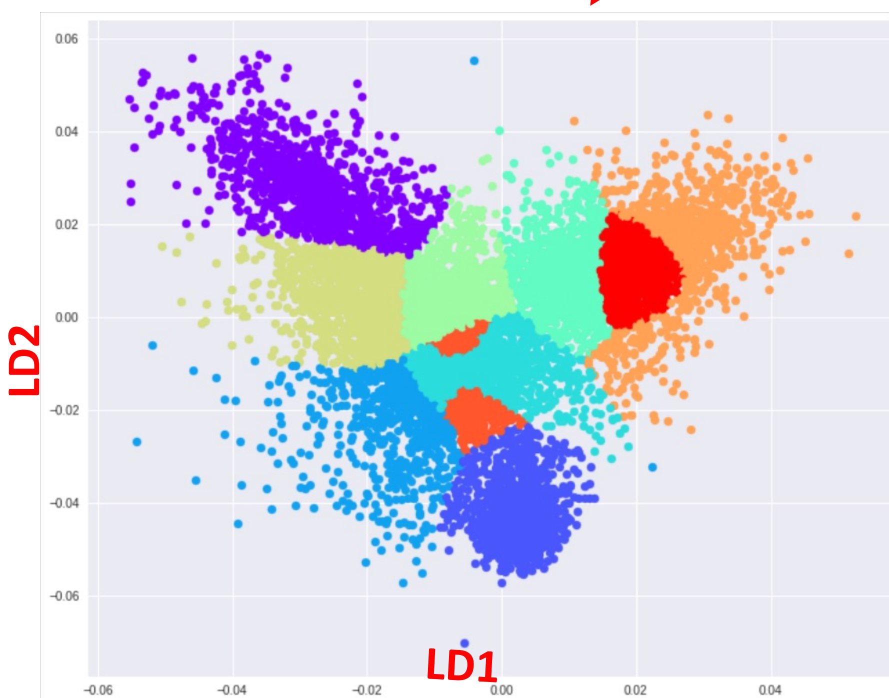
LDA as a Dimension Reduction Tool

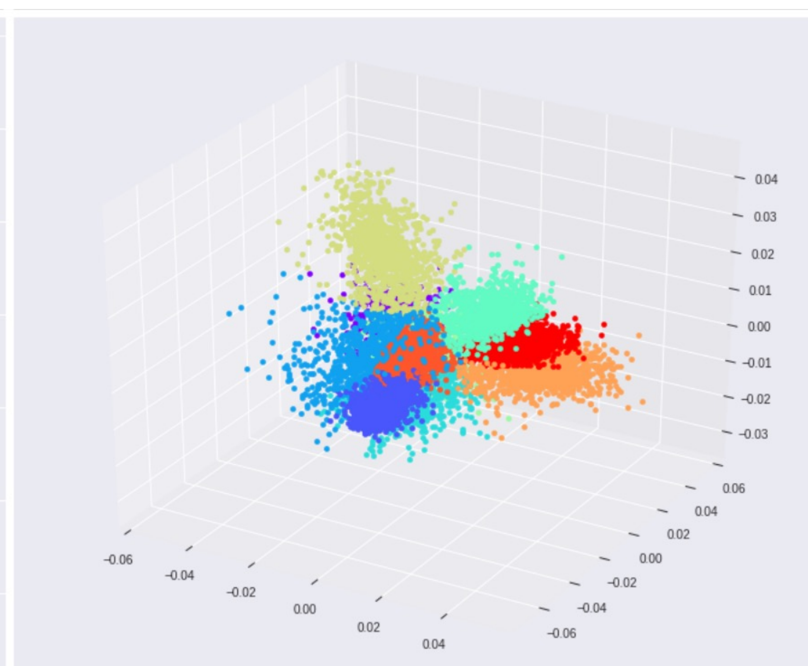
2 class problem



LD1: 1st and only discriminant (a new axis LDA creates) that accounts for the most variation (separability) between classes

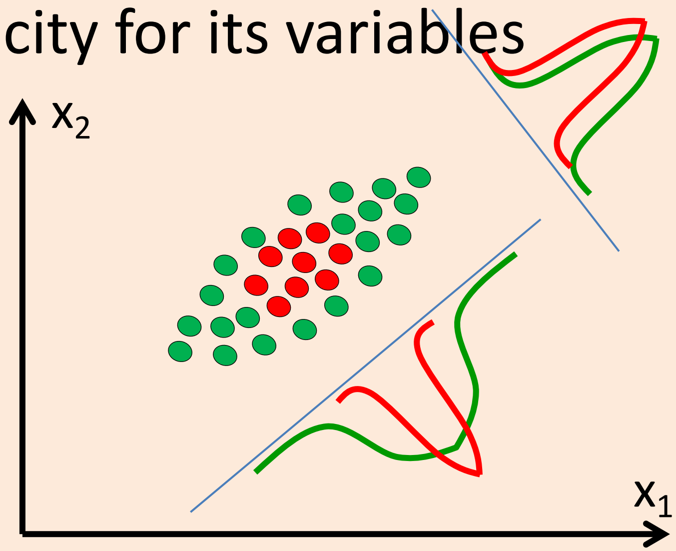
A good way of visualizing high dimensional data





Pros and Cons

- **Pros**
  - Fast and simple
  - Handles both binary and multi-class problems
  - When its assumptions are met, LDA performs slightly better than Logistic Regression even if the number of observations is small
- **Cons**
  - LDA is a parametric method and requires normal distribution and homoscedasticity for its variables
  - Very sensitive to outliers
  - Suffers from multicollinearity
  - LDA will fail if discriminatory information isn't in the mean but in the variance of data



LDA w/ Python

```
#using scikit-learn library
from sklearn.discriminant_analysis import
                                  LinearDiscriminantAnalysis
from sklearn.metrics import accuracy_score

Clf = LinearDiscriminantAnalysis()
Clf.fit(X_train, y_train)
accuracy_score(y_test, Clf.predict(X_test))

# Hyperparameters tuned:
# LDA has a closed form solution and therefore has
# no hyperparameters (except a regularization
# parameter implemented in sklearn)
```

Ref https://medium.freecodecamp.org/an-illustrative-introduction-to-fishers-linear-discriminant-9484efee15ac

References

- Linear Discriminant Analysis – bit by bit
  - https://sebastianraschka.com/Articles/2014_python_lda.html
- Linear Discriminant Analysis for Starters
  - https://eigenfoo.xyz/lda/