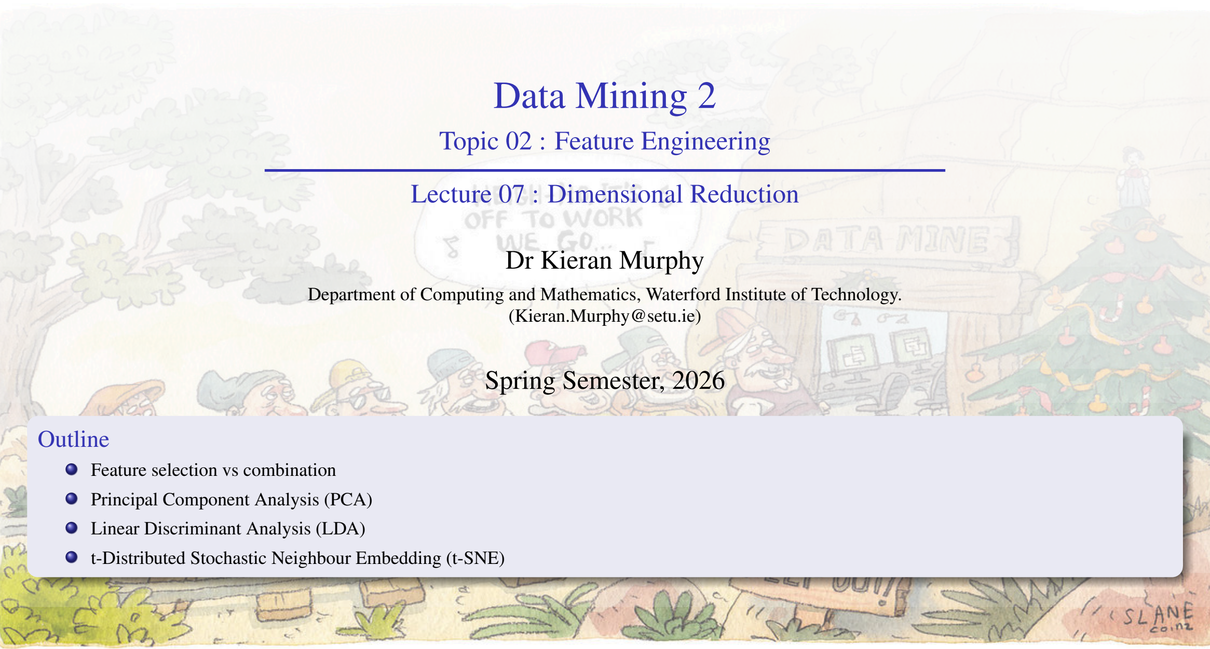


Data Mining 2

Topic 02 : Feature Engineering

Lecture 07 : Dimensional Reduction

Dr Kieran Murphy

Department of Computing and Mathematics, Waterford Institute of Technology.
(Kieran.Murphy@setu.ie)

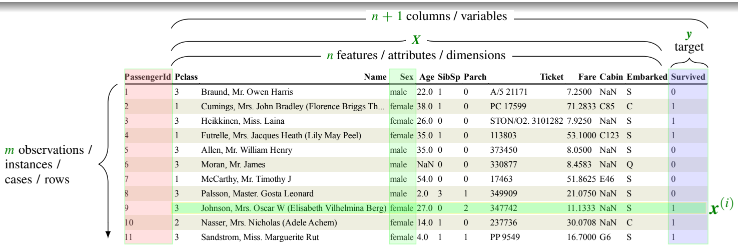
Spring Semester, 2026

Outline

- Feature selection vs combination
- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- t-Distributed Stochastic Neighbour Embedding (t-SNE)

Outline

Motivation — The Problem — Curse of Dimensionality



$n+1$ columns / variables
$\boldsymbol{X}$
n features / attributes / dimensions
$\boldsymbol{y}$
target
m observations / instances / cases / rows
$\boldsymbol{x}^{(i)}$

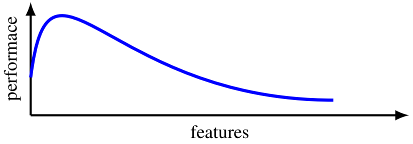
PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Survived
1	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	0
2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	1
3	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	1
4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	1
5	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	0
6	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q	0
7	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S	0
8	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S	0
9	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S	1
10	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C	1
11	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	G6	S	1

If the number of features/attributes/dimensions is too large then:

- Data points are sparse — long distances between randomly distributed points.
- Number of observations, m, needed to maintain the same density of point (needed to get good/reliable statistical estimates) increases exponentially with dimension.

For a given sample size, m, there is a maximum number of features above which the performance of our classifier will degrade rather than improve.

Information that is lost by discarding some features is compensated by a more accurate mapping in lower-dimensional space.



Motivation — The Solution I

Feature Extraction

- Create new features by combining existing ones:

$$[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_n] \xrightarrow{\text{feature extraction}} f([\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_n]) = [{}^{new}\boldsymbol{x}_1, {}^{new}\boldsymbol{x}_2, \ldots]$$

(Churn: `Day_Min` + `Eve_Min` + `Night_Min` $\rightarrow$ `Total_Min`).

- Manual, problem specific, not scalable, . . .

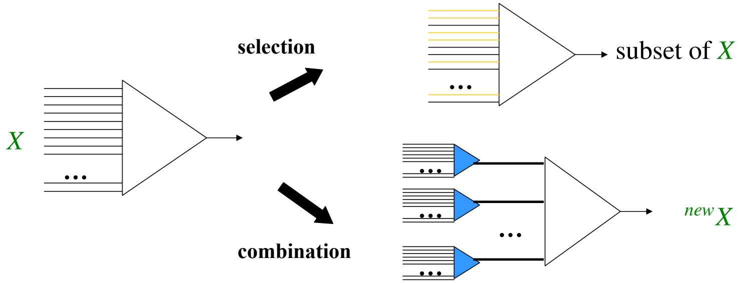
Feature Selection

$$[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_n] \xrightarrow{\text{feature selection}} = [\boldsymbol{x}_1, \cancel{\boldsymbol{x}_2}, \cancel{\boldsymbol{x}_3}, \boldsymbol{x}_4, \ldots]$$

- Pick a subset of features that gives/preserves most of the output prediction capabilities.
- Filters — filter out features with small potential to predict outputs well – use univariate analysis — done before classification
- Wrappers — select features that directly optimises the accuracy of the classifier (Assignment).

Motivation — The Solution II

Feature Combination



- We want to replace a high dimensional input with a small set of features (obtained by combining inputs).
- Linear techniques
  - Principle Component Analysis (PCA)
  - Fisher's Linear Discriminant Analysis (LDA)
- Non-linear techniques
  - t-distributed stochastic neighbour embedding (t-SNE)

Outline

Principal Component Analysis (PCA) I

To understand PCA consider the following two-dimensional problem:

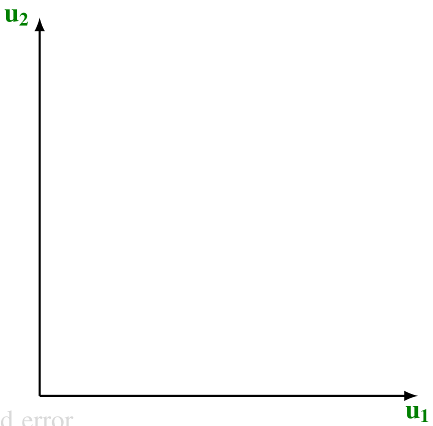
- Assume data follows a Gaussian distribution as shown in the diagram.
- Any point/observation/vector can be represented by their 2D coordinates

$$\boldsymbol{x} = x^{(1)}\boldsymbol{u}_1 + x^{(2)}\boldsymbol{u}_2 = \left(x^{(1)}, x^{(2)}\right)_{\boldsymbol{u}_1,\boldsymbol{u}_2}$$

- We want a 1D representation ${}^{new}\boldsymbol{x}$ that is 'close' to $\boldsymbol{x}$.

$${}^{new}\boldsymbol{x} =$$

- Here 'closeness' is measured by the mean squared error over all points in the distribution.



Principal Component Analysis (PCA) I

To understand PCA consider the following two-dimensional problem:

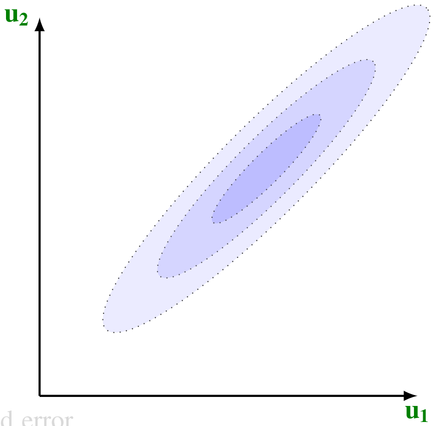
- Assume data follows a Gaussian distribution as shown in the diagram.
- Any point/observation/vector can be represented by their 2D coordinates

$$\boldsymbol{x} = x^{(1)}\boldsymbol{u}_1 + x^{(2)}\boldsymbol{u}_2 = \left(x^{(1)}, x^{(2)}\right)_{\boldsymbol{u}_1,\boldsymbol{u}_2}$$

- We want a 1D representation ${}^{new}\boldsymbol{x}$ that is 'close' to $\boldsymbol{x}$.

$${}^{new}\boldsymbol{x} =$$

- Here 'closeness' is measured by the mean squared error over all points in the distribution.



Principal Component Analysis (PCA) I

To understand PCA consider the following two-dimensional problem:

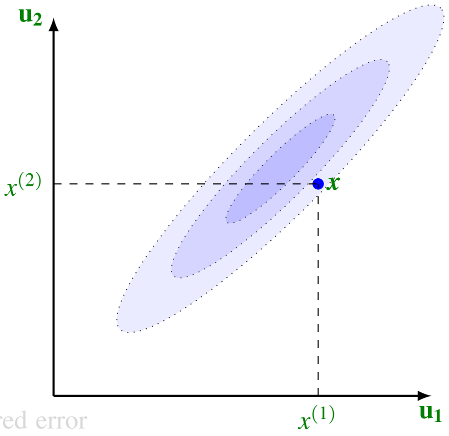
- Assume data follows a Gaussian distribution as shown in the diagram.
- Any point/observation/vector can be represented by their 2D coordinates

$$\boldsymbol{x} = x^{(1)}\boldsymbol{u}_1 + x^{(2)}\boldsymbol{u}_2 = \left(x^{(1)}, x^{(2)}\right)_{\boldsymbol{u}_1,\boldsymbol{u}_2}$$

- We want a 1D representation ${}^{new}\boldsymbol{x}$ that is 'close' to $\boldsymbol{x}$.

$${}^{new}\boldsymbol{x} =$$

- Here 'closeness' is measured by the mean squared error over all points in the distribution.



Principal Component Analysis (PCA) I

To understand PCA consider the following two-dimensional problem:

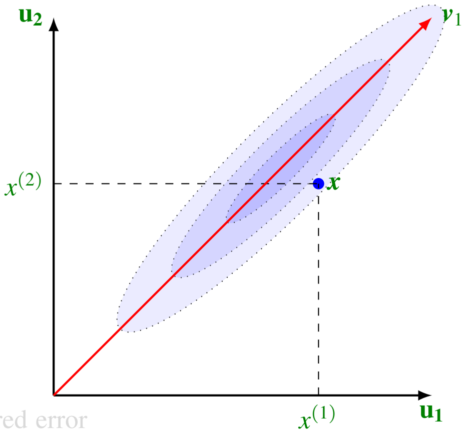
- Assume data follows a Gaussian distribution as shown in the diagram.
- Any point/observation/vector can be represented by their 2D coordinates

$$\boldsymbol{x} = x^{(1)}\boldsymbol{u}_1 + x^{(2)}\boldsymbol{u}_2 = \left(x^{(1)}, x^{(2)}\right)_{\boldsymbol{u}_1,\boldsymbol{u}_2}$$

- We want a 1D representation ${}^{new}\boldsymbol{x}$ that is 'close' to $\boldsymbol{x}$.

$${}^{new}\boldsymbol{x} =$$

- Here 'closeness' is measured by the mean squared error over all points in the distribution.



Principal Component Analysis (PCA) I

To understand PCA consider the following two-dimensional problem:

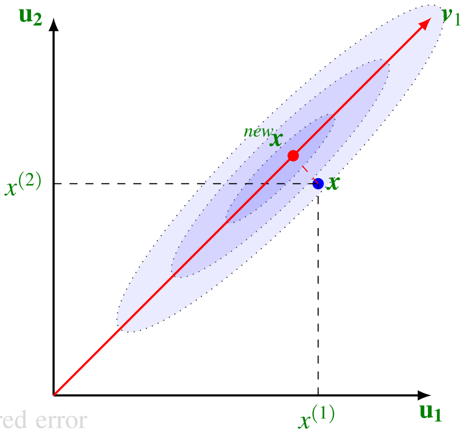
- Assume data follows a Gaussian distribution as shown in the diagram.
- Any point/observation/vector can be represented by their 2D coordinates

$$\boldsymbol{x} = x^{(1)}\boldsymbol{u}_1 + x^{(2)}\boldsymbol{u}_2 = \left(x^{(1)}, x^{(2)}\right)_{\boldsymbol{u}_1,\boldsymbol{u}_2}$$

- We want a 1D representation ${}^{new}\boldsymbol{x}$ that is 'close' to $\boldsymbol{x}$.

$${}^{new}\boldsymbol{x} = \ ? \ \boldsymbol{v}_1$$

- Here 'closeness' is measured by the mean squared error over all points in the distribution.



Principal Component Analysis (PCA) I

To understand PCA consider the following two-dimensional problem:

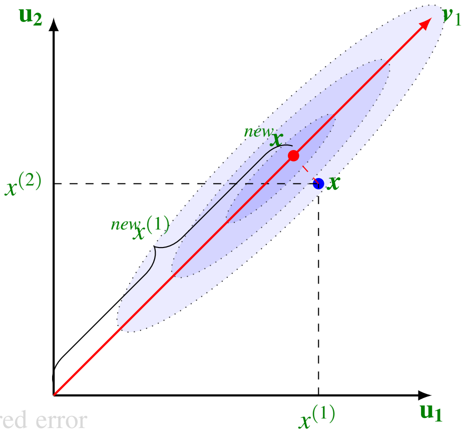
- Assume data follows a Gaussian distribution as shown in the diagram.
- Any point/observation/vector can be represented by their 2D coordinates

$$\boldsymbol{x} = x^{(1)}\boldsymbol{u}_1 + x^{(2)}\boldsymbol{u}_2 = \left(x^{(1)}, x^{(2)}\right)_{\boldsymbol{u}_1,\boldsymbol{u}_2}$$

- We want a 1D representation ${}^{new}\boldsymbol{x}$ that is 'close' to $\boldsymbol{x}$.

$${}^{new}\boldsymbol{x} = {}^{new}x^{(1)}\boldsymbol{v}_1 = \left({}^{new}x^{(1)}\right)_{\boldsymbol{v}_1}$$

- Here 'closeness' is measured by the mean squared error over all points in the distribution.



Principal Component Analysis (PCA) I

To understand PCA consider the following two-dimensional problem:

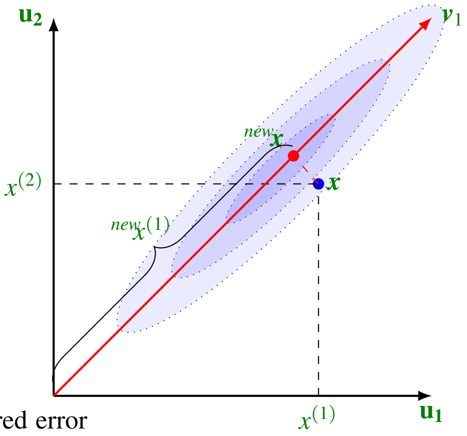
- Assume data follows a Gaussian distribution as shown in the diagram.
- Any point/observation/vector can be represented by their 2D coordinates

$$\boldsymbol{x} = x^{(1)}\boldsymbol{u}_1 + x^{(2)}\boldsymbol{u}_2 = \left(x^{(1)}, x^{(2)}\right)_{\boldsymbol{u}_1,\boldsymbol{u}_2}$$

- We want a 1D representation ${}^{new}\boldsymbol{x}$ that is 'close' to $\boldsymbol{x}$.

$${}^{new}\boldsymbol{x} = {}^{new}x^{(1)}\boldsymbol{v}_1 = \left({}^{new}x^{(1)}\right)_{\boldsymbol{v}_1}$$

- Here 'closeness' is measured by the mean squared error over all points in the distribution.



Principal Component Analysis (PCA) II

- It can be shown that the ‘optimal’ 1D representation consists of projecting the vector $\boldsymbol{x}$ over the direction of maximum variance in the data (e.g., the longest axis in the ellipse).
- Generalising this result we have:

> The optimal approximation of a random vector $\boldsymbol{x}$ in N dimensional space by a linear combination M where $(M < N)$ independent vectors is obtained by projecting the random vector $\boldsymbol{x}$ onto the eigenvectors, $\boldsymbol{v}_i$ corresponding to the largest eigenvalues λ of the covariance matrix of $\boldsymbol{x}$.

- Data needs to be normalised — otherwise bias towards dimensions with larger range/variance.
- The eigenvalues give the proportion of the observed variance that is represented by the principal components.
  - Can be used to evaluate the effectiveness of the PCA.
- Note method is unsupervised — does not take the target into account.

Python Implementation — Iris dataset

The Iris dataset represents 3 kind of Iris flowers (Setosa, Versicolour and Virginica) with 4 attributes: sepal length, sepal width, petal length and petal width.

```
from sklearn import datasets
from sklearn.decomposition import PCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

iris = datasets.load_iris()

X, y = iris.data, iris.target

pca = PCA(n_components=4)
X_r = pca.fit(X).transform(X)

print('Explained variance ratio:\n%s'
    % str(pca.explained_variance_ratio_))
```

```
Explained variance ratio:
[0.92461872 0.05306648 0.01710261 0.00521218]
```

92% of the variation in the data is captured in the 1st principal component, etc.

Python Implementation — Churn dataset

Applying PCA to our Churn dataset we get principal components:

```
Explained variance ratio:
['0.1206', '0.1198', '0.1175', '0.1163', '0.1128', '0.06287', '0.06043', '0.05927', '0.05792', '0.05761', '0.05671',
'0.05559', '0.002526', '4.259e-07', '4.607e-08', '1.315e-08', '2.807e-09']
```

- Largest principal components only explains 12% of the variance in the data.
- Need 12 principal components to explain over 95% of the variance in the data.

$\Rightarrow$ PCA not suitable for this dataset.

Limitations of PCA

- PCA is a linear method.
- Can over estimate the “true” dimensionality, due to non-linear correlations.

Python Implementation — Churn dataset

Applying PCA to our Churn dataset we get principal components:

```
Explained variance ratio:
['0.1206', '0.1198', '0.1175', '0.1163', '0.1128', '0.06287', '0.06043', '0.05927', '0.05792', '0.05761', '0.05671',
'0.05559', '0.002526', '4.259e-07', '4.607e-08', '1.315e-08', '2.807e-09']
```

- Largest principal components only explains 12% of the variance in the data.
- Need 12 principal components to explain over 95% of the variance in the data.

$\Rightarrow$ PCA not suitable for this dataset.

Limitations of PCA

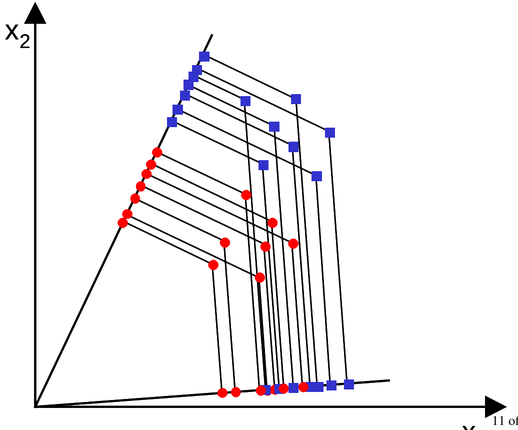
- PCA is a linear method.
- Can over estimate the “true” dimensionality, due to non-linear correlations.

Linear Discriminant Analysis (LDA) I

The objective of LDA is to perform dimensionality reduction while preserving as much of the class discriminatory information as possible

To understand LDA, consider the following 2-dimensional dataset with 2 cases:

- Assume we have a set of 2D samples, some of which belong to class RED, and the remainder belong to class BLUE.
- Consider all possible lines going (through the origin) and the resulting projection of the dataset onto those lines.
- Of all possible lines we would like to select the one that maximises the separability of the classes.

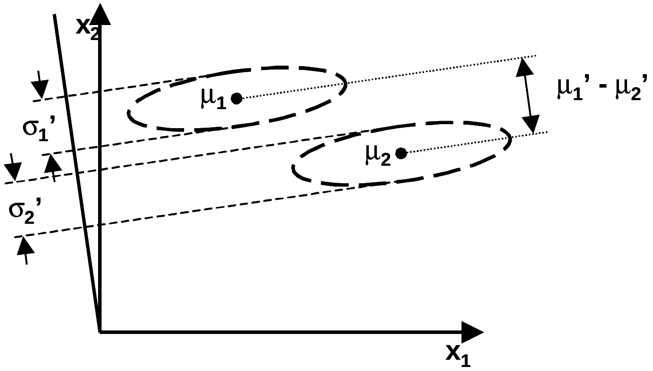


Linear Discriminant Analysis (LDA) II

In general we want

- Maximise separation between the projected class means.
- Minimise variance within each projected class.

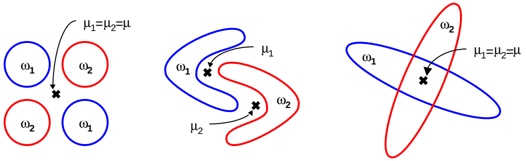
$$\text{maximize } \frac{(\mu_1' - \mu_2')^2}{\sigma_1'^2 - \sigma_2'^2}$$



Limitations of Linear Discriminant Analysis I

LDA assumes unimodal Gaussian likelihoods

If the densities are significantly non-Gaussian, LDA may not preserve any complex structure of the data needed for classification

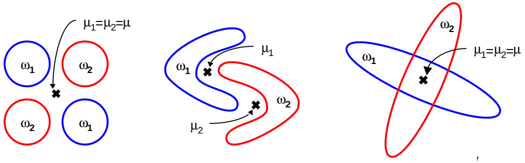


In each dataset above, the LDA will fail due to bi-modal distributions,
non-linearly separable, same mean but different orientation.

Limitations of Linear Discriminant Analysis I

LDA assumes unimodal Gaussian likelihoods

If the densities are significantly non-Gaussian, LDA may not preserve any complex structure of the data needed for classification

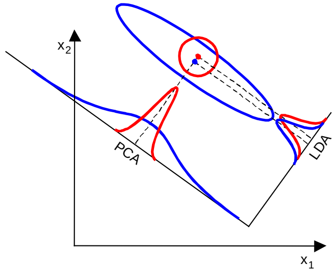


In each dataset above, the LDA will fail due to bi-modal distributions, non-linearly separable, same mean but different orientation.

LDA focus on mean not variance

LDA will fail when the discriminatory information is not in the mean but rather in the variance of the data.

In the diagram, the mean of both classes are similar but their variance differ. This is not captured by LDA.

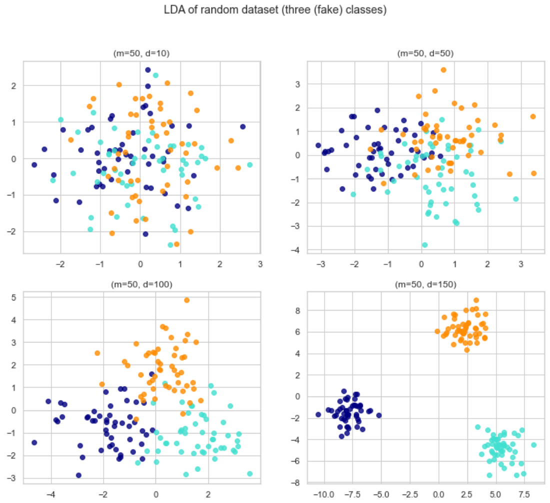


Limitations of Linear Discriminant Analysis II

LDA has a tendency to overfit training data.

To illustrate this problem, we generate an artificial dataset:

- Generate 150 d-dimensional points, with the exact same likelihood: a multivariate Gaussian with zero mean and identity covariance.
- Allocate the points to one of three classes at a random.
- The classes should be identical,
- Perform LDA.
- As we arbitrarily increase the dimensions, classes appear to separate better, even though they come from the same distribution.



LDA has a tendency to overfit training data

```python
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

np.random.seed(42)
colors = ['navy', 'turquoise', 'darkorange']

m = 50

fig, axs = plt.subplots(2,2,figsize=(12,10))
for k, d in enumerate([10,50,100,150]):

    X = np.random.normal(size=3*m*d).reshape(3*m,d)
    y = np.array([0] * m + [1] * m + [2] * m)

    lda = LinearDiscriminantAnalysis(n_components=2)
    X_r = lda.fit(X, y).transform(X)

    for color, i in zip(colors, [0, 1, 2]):
        fig.axes[k].scatter(X_r[y == i, 0], X_r[y == i, 1], alpha=.8, color=color)
    fig.axes[k].set_title(f'(m={m}, d={d})')
plt.suptitle("LDA of random dataset (three (fake) classes)")

plt.show()
```

PCA vs LDA

- The goal of PCA is to represent the samples accurately in a lower-dimensional space — how to represent the data (signal) optimally?
- The goal of LDA is to enhance the class-discriminatory information in the lower-dimensional space — how to represent the classes optimally?

LDA, in contrast to PCA, is a supervised method, using known class labels.

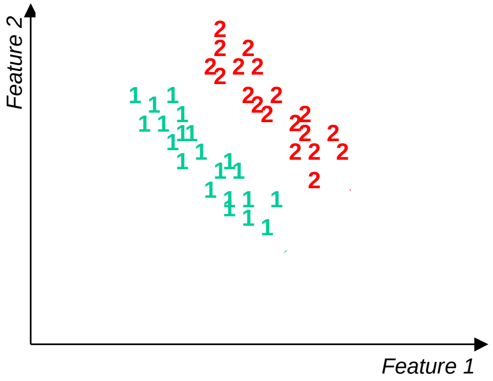
- Consider a 2D problem with two cases as shown.
- The structure of the data depends on direction of 'viewer'.
- PCA and LDA are directions that maximise the spread of the data and the separation between the classes respectively.

PCA vs LDA

- The goal of PCA is to represent the samples accurately in a lower-dimensional space — how to represent the data (signal) optimally?
- The goal of LDA is to enhance the class-discriminatory information in the lower-dimensional space — how to represent the classes optimally?

LDA, in contrast to PCA, is a supervised method, using known class labels.

- Consider a 2D problem with two cases as shown.
- The structure of the data depends on direction of 'viewer'.
- PCA and LDA are directions that maximise the spread of the data and the separation between the classes respectively.

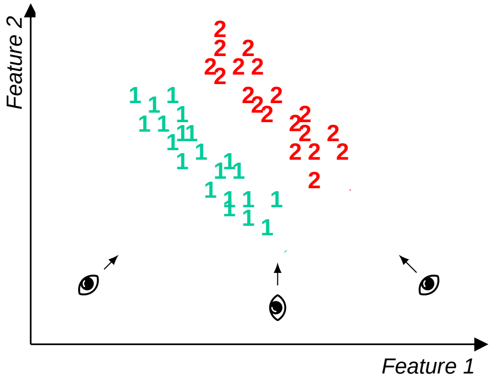


PCA vs LDA

- The goal of PCA is to represent the samples accurately in a lower-dimensional space — how to represent the data (signal) optimally?
- The goal of LDA is to enhance the class-discriminatory information in the lower-dimensional space — how to represent the classes optimally?

LDA, in contrast to PCA, is a supervised method, using known class labels.

- Consider a 2D problem with two cases as shown.
- The structure of the data depends on direction of ‘viewer’.
- PCA and LDA are directions that maximise the spread of the data and the separation between the classes respectively.

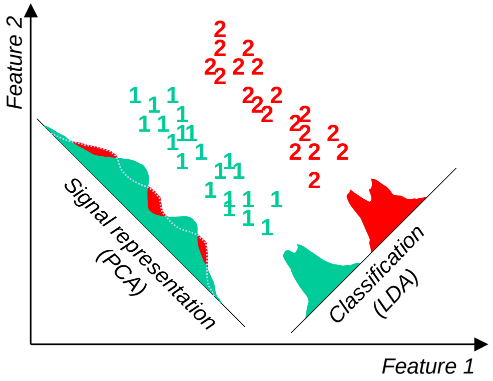


PCA vs LDA

- The goal of PCA is to represent the samples accurately in a lower-dimensional space — how to represent the data (signal) optimally?
- The goal of LDA is to enhance the class-discriminatory information in the lower-dimensional space — how to represent the classes optimally?

LDA, in contrast to PCA, is a supervised method, using known class labels.

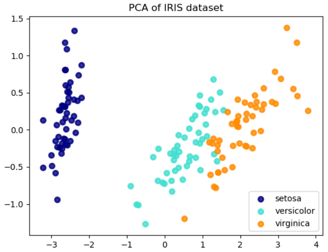
- Consider a 2D problem with two cases as shown.
- The structure of the data depends on direction of 'viewer'.
- PCA and LDA are directions that maximise the spread of the data and the separation between the classes respectively.

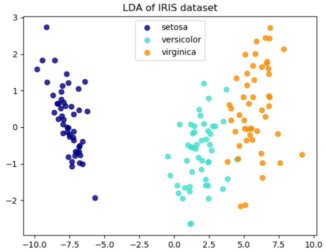


PCA vs LDA — Iris Dataset*

The Iris dataset represents 3 kind of Iris flowers (Setosa, Versicolour and Virginica) with 4 attributes: sepal length, sepal width, petal length and petal width.

- Principal Component Analysis (PCA) applied to this data identifies the combination of attributes (principal components, or directions in the feature space) that account for the most variance in the data.
- Linear Discriminant Analysis (LDA) tries to identify attributes that account for the most variance between classes.





Comparison of LDA and PCA 2D projection of Iris dataset

Outline

t-Distributed Stochastic Neighbour Embedding (t-SNE)

T-distributed Stochastic Neighbour Embedding (t-SNE)
is a nonlinear dimensionality reduction technique well-suited for embedding high-dimensional data for visualisation in a low-dimensional space of two or three dimensions. It models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modelled by nearby points and dissimilar objects are modelled by distant points with high probability.

Disadvantages of t-SNE

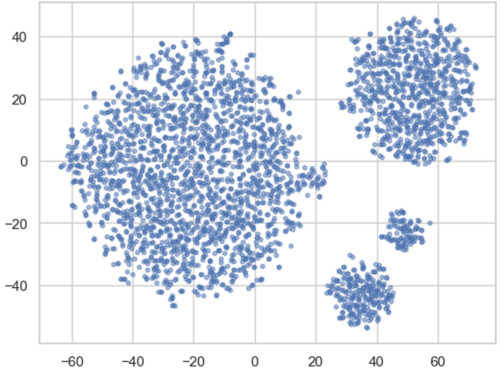
- High computational complexity. The implementation in scikit-learn is unlikely to be feasible in a real task. Try Multicore-TSNE instead, for larger datasets.
- The plot can change a great deal depending on the random seed, which complicates interpretation. In general, you shouldn?t make any far-reaching conclusions based on such graphs because it can equate to plain guessing.

t-SNE an Churn Dataset I

Applying t-SNE to the Churn dataset (AFTER preparing and normalising the data)

3
```
from sklearn.manifold import TSNE
tsne = TSNE(random_state=142)
tsne_repr = tsne.fit_transform(X_train)
```

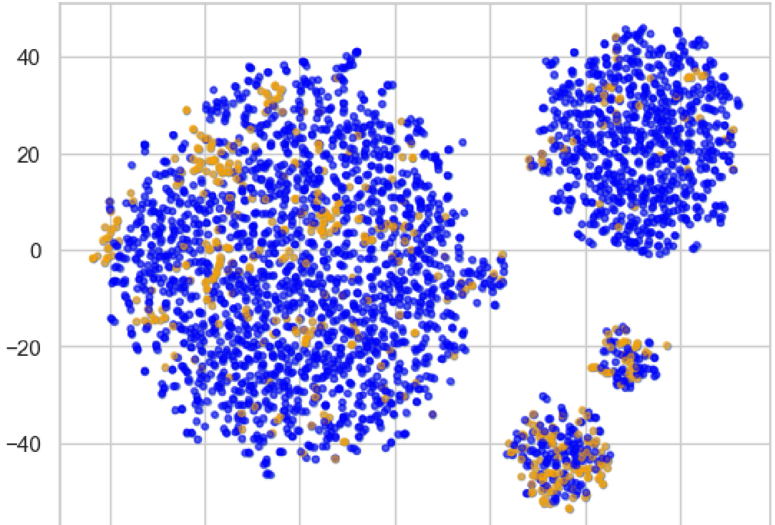
After 15 seconds† we then plot the resulting t-SNE representation



OK, we see 4 blobs, so what?

t-SNE an Churn Dataset II

We could colour this t-SNE representation according to the churn (blue for loyal customers, and orange for those who churned).

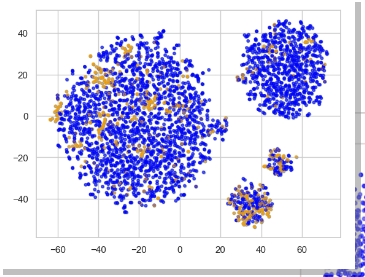


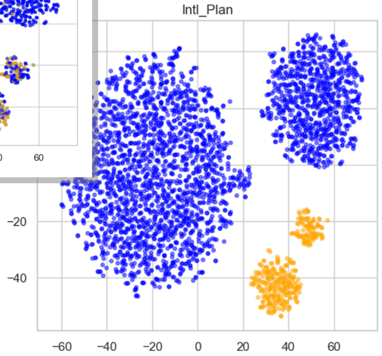
OK, the churn rate appears differ between blobs — as customers who churned are concentrated in a few areas of the lower dimensional feature space.

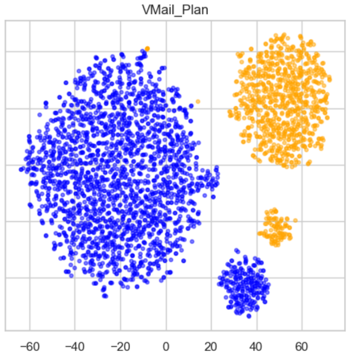
But things are still not that clear.

t-SNE an Churn Dataset III

To better understand the picture, we can also colour it with the remaining binary features: `Intl_Plan` and `VMail_PLan`. Orange dots here indicate instances that are positive for the corresponding binary feature.







Now we see many dissatisfied customers who canceled their subscription are crowded together in one cluster representing the people with the international plan but no voice mail plan.