Machine Learning Techniques for Text

Module 5: Recommending Music Titles

Dr. Nikos Tsourakis

Course outline

- Module 0: Python Crash Course
- Module 1: Intro to Machine Learning
- Module 2: Detecting Spam Emails
- Module 3: Classifying Topics of Newsgroup Posts
- Module 4: Extracting Sentiments from Product Reviews
- **Module 5: Recommending Music Titles**
- Module 6: Teaching Machines to Translate
- Module 7: Summarizing Wikipedia Articles
- Module 8: Detecting Hateful and Offensive Language
- Module 9: Generating Text in Chatbots
- Module 10: Clustering Speech-to-Text Transcriptions

Overview

- Consumer choices and how they can be influenced are critical factors for every business
- Most people are interested in specific music genres, have favorite authors, or engage in particular hobbies. This information can be extracted from their purchase history or product reviews
- This module seeks to exploit product and user data to create recommender systems for music titles. We will base the discussion on a corpus of customer reviews from the Amazon online store
    - First, we will perform exploratory data analysis to identify possible shortcomings in the samples and carry out an extensive data cleaning task
    - Next, we will introduce two flavors of recommenders that rely either on product reviews or user ratings
    - The implementations will utilize dimensionality reduction techniques, introducing a new method for this task
    - Finally, we will revisit the topic of hyperparameter tuning and discuss a related technique

Module objectives

After completing this module, you should be able to:

- Understanding essential concepts in statistics
- Examining more advanced dimensionality reduction techniques
- Identifying hidden relations between products and customers
- Learning methods to compute optimum values of hyperparameters
- Creating models using autoencoders

Machine Learning Techniques for Text

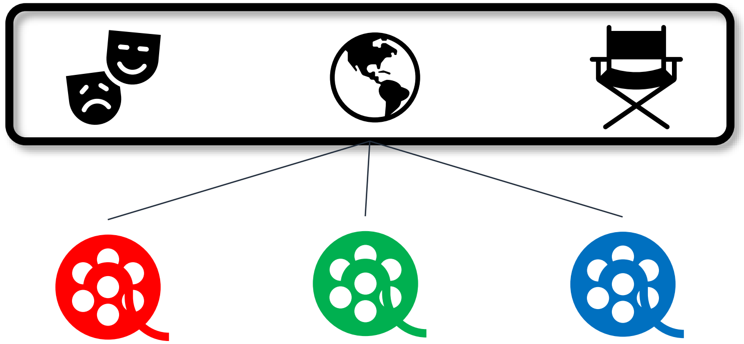
Section 1: Understanding recommender systems

Introduction

- Automatic systems can use various information to recommend content and engage users effectively
- Recommender systems are commonly encountered in various online platforms for product recommendations, news, friends, jobs, and restaurants
- Recommender systems can be categorized into ***content-based*** and ***collaborative-filtering*** types
  - Content-based systems create models based on a customer's past purchases to recommend new items
  - Collaborative filtering relies on mutual preferences as it identifies items that a user might like based on how other similar users rated them

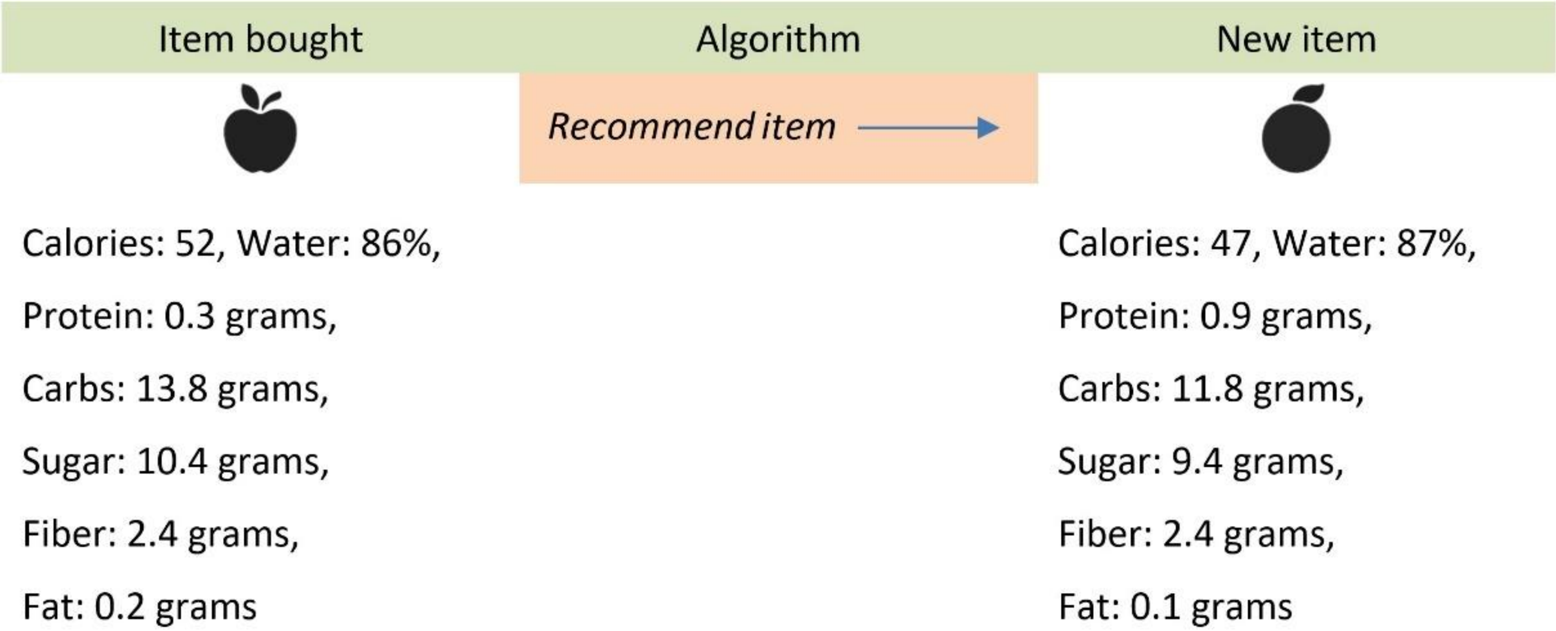
Content-based recommenders

- The idea behind the first category is simple: create a model with the properties of the items already purchased by a customer and run this model on new items to identify those they are likely to buy
- Generally, content-based systems become more accurate the more input a user provides
- For instance, a movie recommender can exploit certain information for a film like the actors and director names, genre, language, etc.



Content-based recommenders

- Suppose that a customer purchases ***apples*** frequently
- The recommender algorithm proposes ***oranges*** as a candidate purchase based on the apple's properties, *calories*, *water percentage*, *protein*, *carbs*, *sugar*, *fiber*, and *fat content*

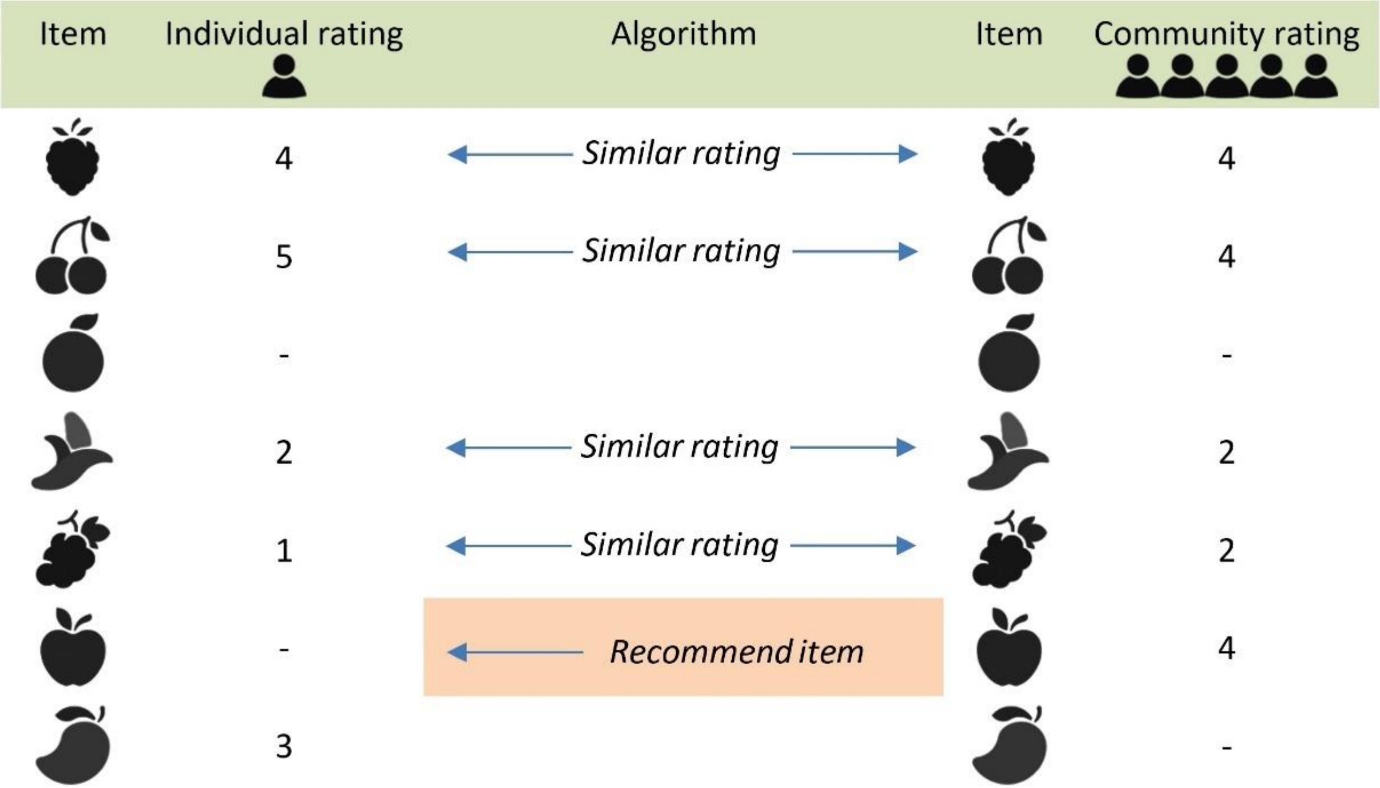


Collaborative-filtering recommenders

- A significant drawback of content-based recommenders is that they base their decision on items belonging to categories people already know to want
- We need the element of surprise by the recommendation
  - 😯 *"I have never thought of this, but I think I like it!"*
- Collaborative filtering systems try to identify similarities among customers based on their past behavior
- People that exhibit similar purchase habits can recommend products to each other
- The benefit, in this case, is that customers are exposed to items for which they never expressed any explicit interest

Collaborative-filtering recommenders

- The task becomes even more interesting when there is some rating for each product, a feature commonly encountered in most e-commerce and similar online services



Machine Learning Techniques for Text

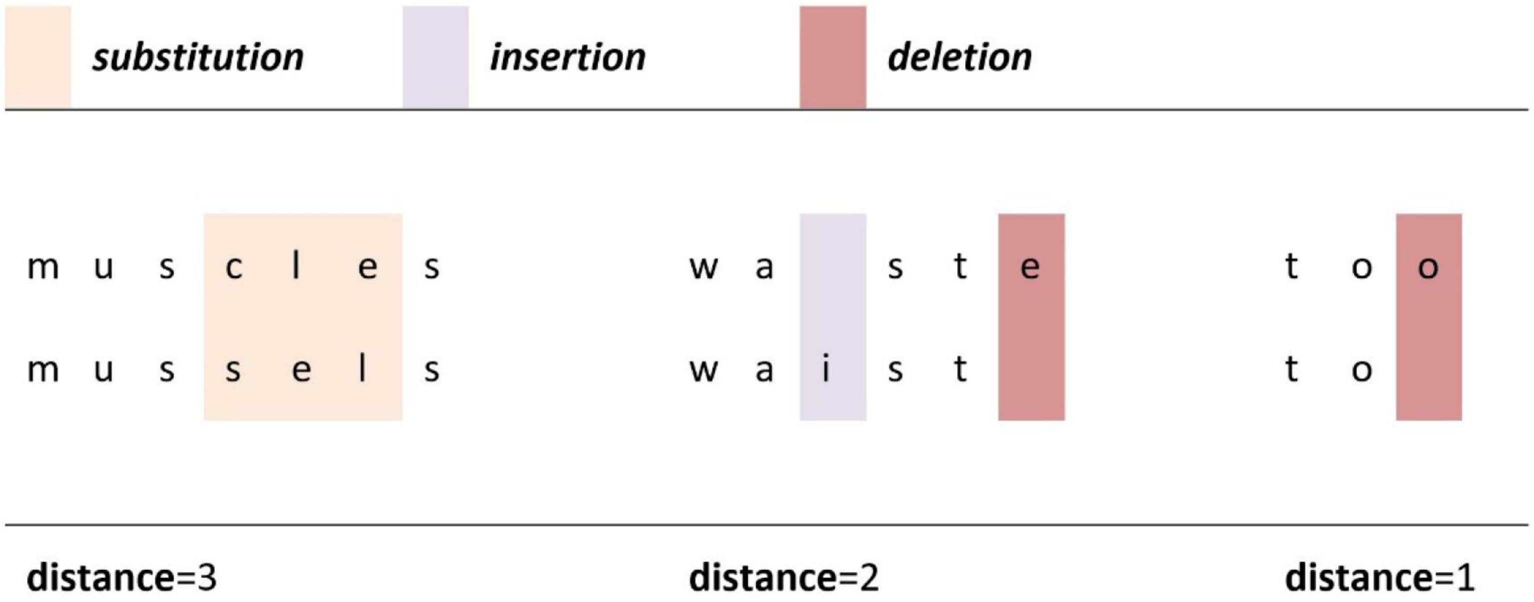
Section 2: Performing exploratory data analysis

Levenshtein distance

- During EDA we identify that the names of several music titles appear with slightly different versions
- Including all versions in our dataset wouldn’t make sense
- The ***Levenshtein distance*** is used for measuring text similarity (in our case the similarity between the two titles)
- To calculate the distance, we need to count the minimum number of character edits to change one word to the other

Levenshtein distance

- Consider three sets of ***homophones*** (words with the same pronunciation but different meanings)

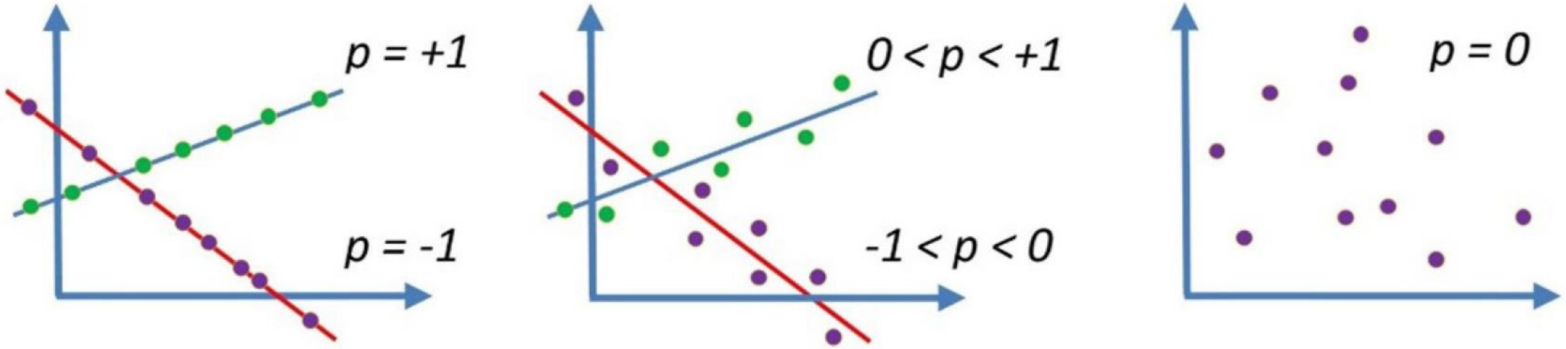


Pearson correlation

- The ***correlation*** between two changing portions (or variables) indicates how the change of the first variable affects the direction of change for the second one
- A typical example is the correlation between ***height*** and ***weight***
  - As height increases, weight tends to increase too, and we can say that these variables are positively correlated
- A correlation coefficient, ***p***, indicates both the direction and strength of this relation in statistics
- A typical variant is the ***Pearson correlation***, which receives values between +1 and −1.

Pearson correlation

- A value of ***+1*** indicates a total positive linear correlation
- A value equal to ***−1*** signifies a total negative linear correlation
- When ***p = 0***, there is no linear correlation between the variables
- Values between ***-1*** to ***+1*** might indicate weak, moderate or strong correlation
- Notice that the values must be interpreted on each use case



Correlation and causation

A common fallacy is that correlation implies causation

Twitter

March 12, 2020

Map tweeted by Manchester Councillor, Kenneth Dobson

Let’s practice!

Tasks

- Exploratory data analysis

https://colab.research.google.com/github/PacktPublishing/Machine-Learning-Techniques-for-Text/blob/main/chapter-05/recommender-systems.ipynb

Section 3: Introducing collaborative filtering

Types of collaborative recommenders

- Collaborative filtering relies on mutual preferences, as it identifies items that a user might like based on how other similar users rated them
- The central paradigm behind this approach is driven by the statement

  “*Show me the items people like me have chosen.*
  I might find them interesting!”

- There are two methods for implementing collaborative filtering systems: ***memory-based*** and ***model-based***
  - In the first case, we utilize user rating data to compute the similarity between users or items
  - In the second case, models are developed incorporating machine learning (ML) algorithms to predict user ratings for unrated items

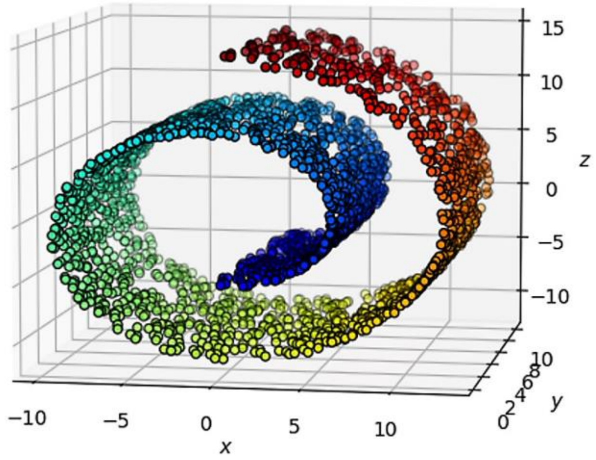
t-distributed Stochastic Neighbor Embedding

- ***t-distributed Stochastic Neighbor Embedding*** (t-SNE) is another dimensionality reduction technique
- It embeds data points from a higher dimensional space into a lower one
- Contrary to PCA, the aim is to preserve the neighborhood of each point as closely as possible – namely, its local structure

t-distributed Stochastic Neighbor Embedding

- ***t-distributed Stochastic Neighbor Embedding*** (t-SNE) is another dimensionality reduction technique
- It embeds data points from a higher dimensional space into a lower one
- Contrary to PCA, the aim is to preserve the neighborhood of each point as closely as possible – namely, its local structure

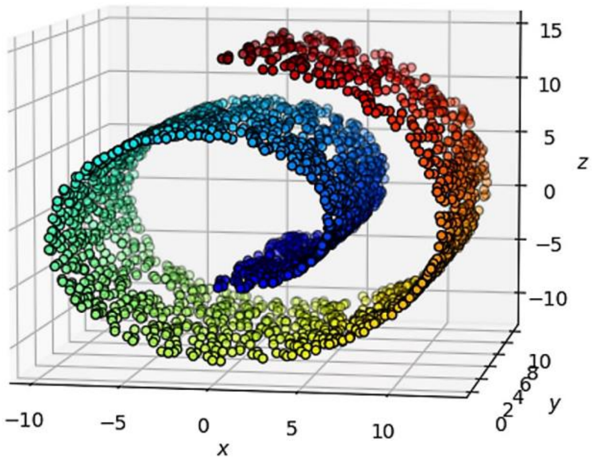
Swiss roll

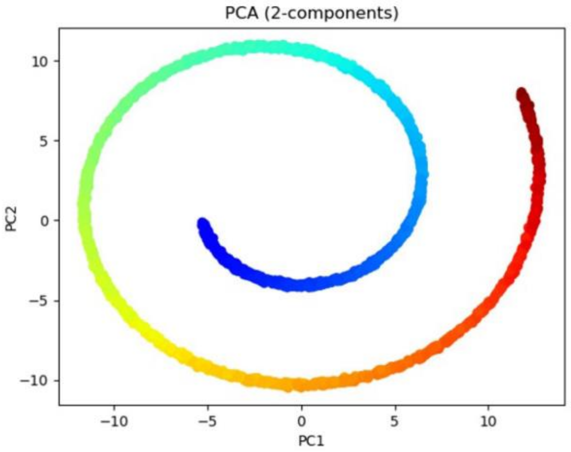


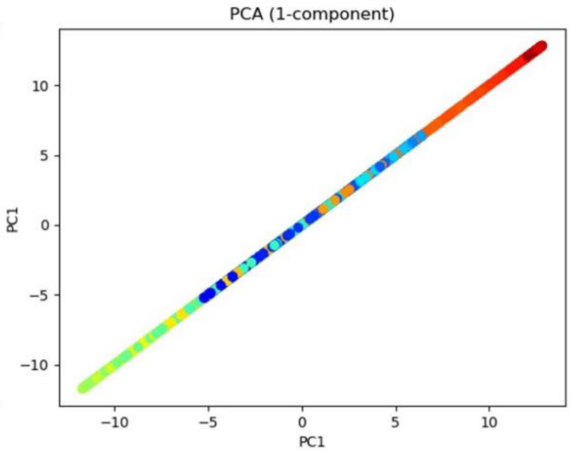
t-distributed Stochastic Neighbor Embedding

- ***t-distributed Stochastic Neighbor Embedding*** (t-SNE) is another dimensionality reduction technique
- It embeds data points from a higher dimensional space into a lower one
- Contrary to PCA, the aim is to preserve the neighborhood of each point as closely as possible – namely, its local structure

Swiss roll



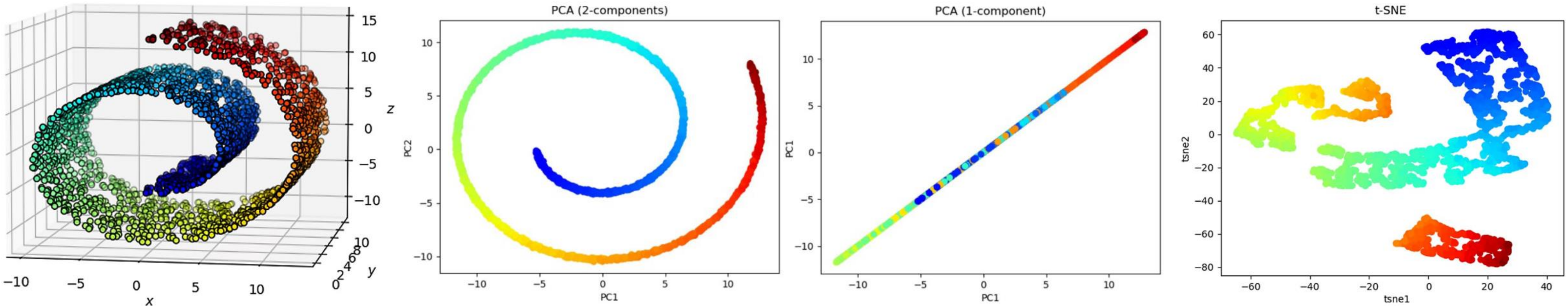




t-distributed Stochastic Neighbor Embedding

- ***t-distributed Stochastic Neighbor Embedding*** (t-SNE) is another dimensionality reduction technique
- It embeds data points from a higher dimensional space into a lower one
- Contrary to PCA, the aim is to preserve the neighborhood of each point as closely as possible – namely, its local structure

Swiss roll



Model-based collaborative systems

- The aim is to develop the necessary models to predict how a specific user would rate an item they have never encountered before
- Consequently, items with a high predicted rating are candidate recommendations for the specific person
- We utilize the rating table from the previous sections and a technique known as ***matrix factorization*** that allows us to discover the latent features underlying the interactions between users and items
- The idea behind factorization is quite simple; express a quantity as a product of smaller ones, called ***factors***
- Consider, for example, the following quantities and their corresponding factors: $6 = 2 \times 3, 588 = 2^2 \times 3 \times 7^2, x^2 + 4x + 3 = (x+3)(x+1)$

Matrix factorization

- Three users (**U1** to **U3**) have rated three movies (**M1** to **M3**), and there is one missing rating depicted with the question mark
- It should be evident that **U3** is more similar to **U2** than **U1** regarding their preferences
- What should **U3**'s rating of M3 be, then?

	M1: Taxi Driver (psychological thriller)		**M2**: Insomnia (psychological thriller)		**M3**: Donnie Brasco (crime drama)	
U1	2		1		0	
U2	5		4		1	
U3	5		3		?	

Matrix factorization

- The aim is to find the latent factor that affect the user ratings
- Suppose that the latent space, in this case, consists of two latent factors
  - The first could refer to whether ***Robert De Niro*** appears in the film and the second as to whether the movie is a ***psychological thriller*** or not
  - Notice that latent factors do not have such clear associations with people, objects, or concepts in practice

	M1: Taxi Driver (psychological thriller)	**M2**: Insomnia (psychological thriller)	**M3**: Donnie Brasco (crime drama)
U1	2	1	0
U2	5	4	1
U3	5	3	**?**

Matrix factorization

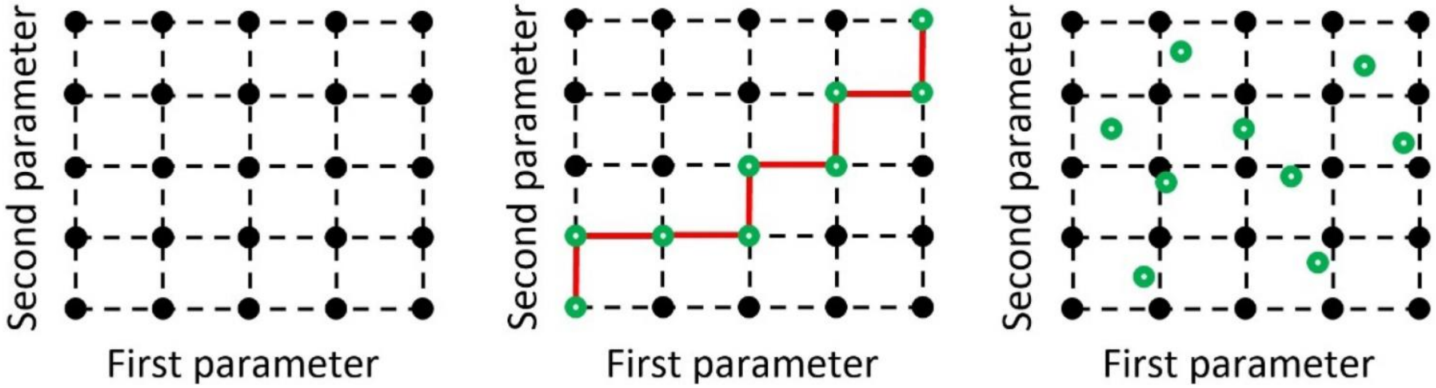
- The aim is to find the latent factor that affect the user ratings
- Suppose that the latent space, in this case, consists of two latent factors
  - The first could refer to whether ***Robert De Niro*** appears in the film and the second as to whether the movie is a ***psychological thriller*** or not
  - Notice that latent factors do not have such clear associations with people, objects, or concepts in practice

	M1: Taxi Driver (psychological thriller)	**M2**: Insomnia (psychological thriller)	**M3**: Donnie Brasco (crime drama)
U1	2	1	0
U2	5	4	1
U3	5	3	?

$$\underset{\boldsymbol{U}(3\times 2)}{\begin{bmatrix}1 & 1\\ 4 & 1\\ 3 & 2\end{bmatrix}} \cdot \underset{\boldsymbol{I}(2\times 3)}{\begin{bmatrix}1 & 1 & 0\\ 1 & 0 & 0\end{bmatrix}} = \underset{\boldsymbol{R}'(3\times 3)}{\begin{bmatrix}2 & 1 & 0\\ 5 & 4 & 0\\ 5 & 3 & \boxed{0}\end{bmatrix}}$$

Parameter tuning

- An ML algorithm can be tuned by adjusting the values of its hyperparameters
- Knowing beforehand which combination provides the best performance is hard until you test them all

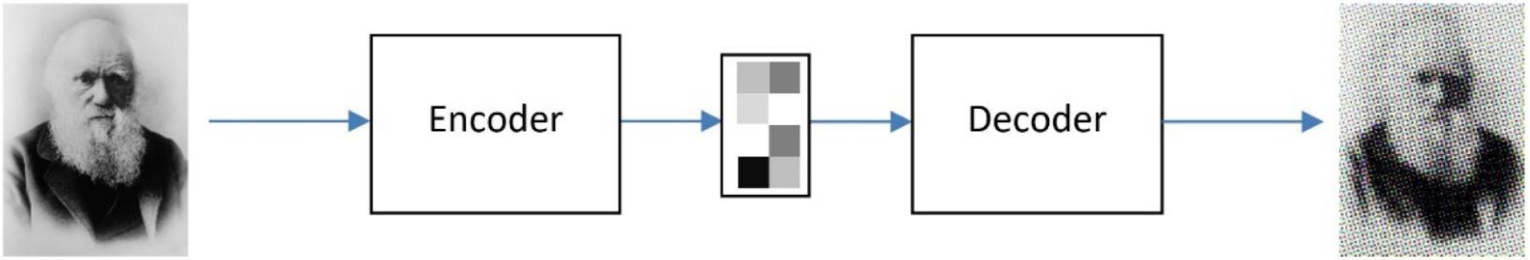


5 hyperparameters with **3** possible values each, require 3×3×3×3×3=**243** different tests

- The ***grid search*** technique helps us quickly tune the algorithm

Autoencoders

- ***Autoencoders*** are neural networks that try to shape their structure so that a given input and output are the same
- An autoencoder network consists of two connected networks, an ***Encoder*** and a ***Decoder*** part
- The encoder takes in the input and converts it into a smaller, more dense representation
- Next, the decoder network uses the compressed encoding from the previous step to reconstruct the original input as accurately as possible



Let’s practice!

Tasks

- Exploratory data analysis
- Content-based filtering
- Collaborative filtering

https://colab.research.google.com/github/PacktPublishing/Machine-Learning-Techniques-for-Text/blob/main/chapter-05/recommender-systems.ipynb

Key takeaways

Visualizations

- Line plots
- Distribution plots

Dimensionality reduction

- Autoencoders
- t-sne

ML algorithms & models

- Restricted Boltzmann Machine

ML concepts

- Parameter tuning

Performance metrics

- Root Mean Squared Error
- Mean Absolute Error
- Levenshtein distance

Machine Learning Techniques for Text

Questions?