Week 07: Machine Learning - III

Data Science Bootcamp
Fall, 2021

Instructor: Sagar Patel

Where are we?

Week 1	Week 2	Week 3	Week 4	Week 5
Introduction	**SQL**	**Python Libraries**	**Exploratory Data Analysis**	**Machine Learning - I**
Introduction to Python; Getting started with Git and GitHub	Relational databases; Selecting, joining and updating tables using SQL	Python packages and libraries; Working with numpy, pandas, matplotlib and seaborn	What makes data good? The “recipe” for performing EDA	Introduction to Machine Learning; Types of Learning

Where are we?

Week 6 — Week 7 — Week 8 — Week 9 — Week 10

Machine Learning - II

Linear and Logistic Regression; Clustering; Confusion Matrix

Machine Learning - III

Machine Learning - IV

Project - I

Project - II

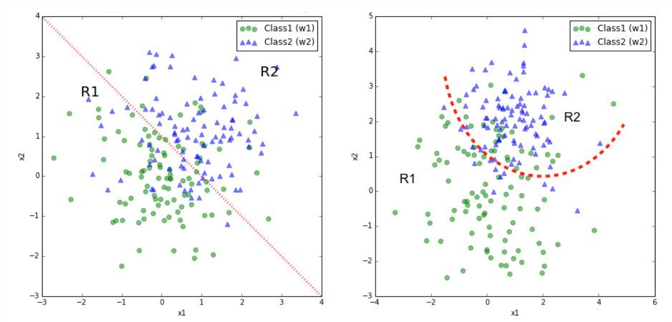
Agenda

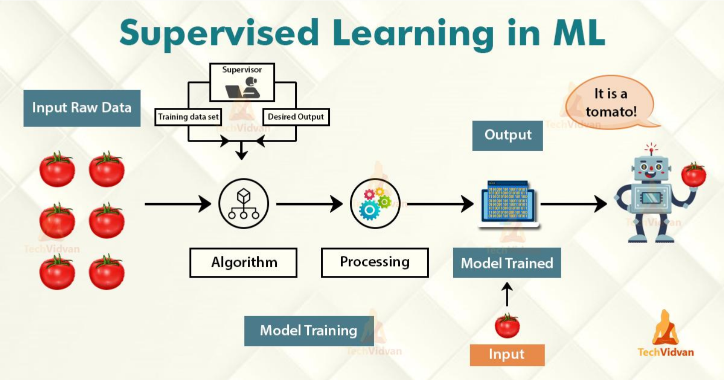
- Supervised Learning
    - Generalization, Overfitting and Underfitting
    - Supervised Machine Learning Algorithms (Notebook)
- Unsupervised Learning and Preprocessing
    - Challenges
    - Preprocessing and Scaling
    - Dimensionality Reduction, Feature Extraction and Manifold Learning

Supervised Machine Learning

Supervised Learning

- Algorithms trained using **labeled** data
- The model takes direct **feedback** to check if it's predicting the correct output or not
- Can be categorized in **Classification** and **Regression**
- Example: Tomato Detector





Generalization

- If the model is able to make **accurate** predictions on unseen data
  - **Generalization** is taking place from the training set to the test set

- Usually we build a model which is able to generalize as much as possible
  - However, there are **some** cases where this can go wrong

Age	Number of cars	Owns a house?	Number of children	Marital status	Owns a dog?	Bought a boat
66	1	Yes	2	widowed	no	yes
52	2	Yes	3	married	no	yes
22	0	No	0	married	yes	no
25	1	No	1	single	no	no
44	0	No	2	divorced	yes	no
39	1	Yes	2	married	yes	no
26	1	No	2	single	no	no
40	3	Yes	1	married	yes	no
53	2	Yes	2	divorced	no	yes
64	2	Yes	3	divorced	no	no

Example: data about customers

Hypothesis

- Naturally, the **rule** we would come up with would be
  - If the customer is older than 45, and has less than 3 children or is not divorced, then they want to buy a boat

Hypothesis

- Naturally, the **rule** we would come up with would be
  - If the customer is older than 45, and has less than 3 children or is not divorced, then they want to buy a boat
- When asked, how this fits
  - It is 100% **accurate** (which is not wrong, technically)

Hypothesis

- Naturally, the **rule** we would come up with would be
  - If the customer is older than 45, and has less than 3 children or is not divorced, then they want to buy a boat
  - When asked, how this fits
    - It is 100% **accurate** (which is not wrong, technically)
- While we can make up many rules which work on this data
  - We are not interested in making predictions for **this** dataset; we already know the answers

Hypothesis

- Naturally, the **rule** we would come up with would be
  - If the customer is older than 45, and has less than 3 children **or is not divorced**, then they want to buy a boat
  - When asked, how this fits
    - It is 100% **accurate** (which is not wrong, technically)
- While we can make up many rules which work on this data
  - We are not interested in making predictions for **this** dataset; we already know the answers
- We want to know if **new customers** are likely to buy a boat

Hypothesis

- Naturally, the **rule** we would come up with would be
  - If the customer is older than 45, and has less than 3 children **or is not divorced**, then they want to buy a boat
  - When asked, how this fits
    - It is 100% **accurate** (which is not wrong, technically)
- While we can make up many rules which work on this data
  - We are not interested in making predictions for **this** dataset; we already know the answers
- We want to know if **new customers** are likely to buy a boat

The only measure of whether an algorithm will perform well on new data is the evaluation on the test set

Overfitting

- Alternatively, if the rule was
  - People **older than 50** want to buy a boat
- We would trust it more than the rule involving children and marital status in addition to age
  - Therefore, we always want to find the **simplest model**

Building a model that is too complex for the amount of information provides, is called **overfitting**

- Overfitting occurs when the model works really well on training set but is unable to generalize

Overfitting

- Alternatively, if the rule was
  - People **older than 50** want to buy a boat
- We would trust it more than the rule involving children and marital status in addition to age
  - Therefore, we always want to find the **simplest model**

Building a model that is too complex for the amount of information provides, is called **overfitting**

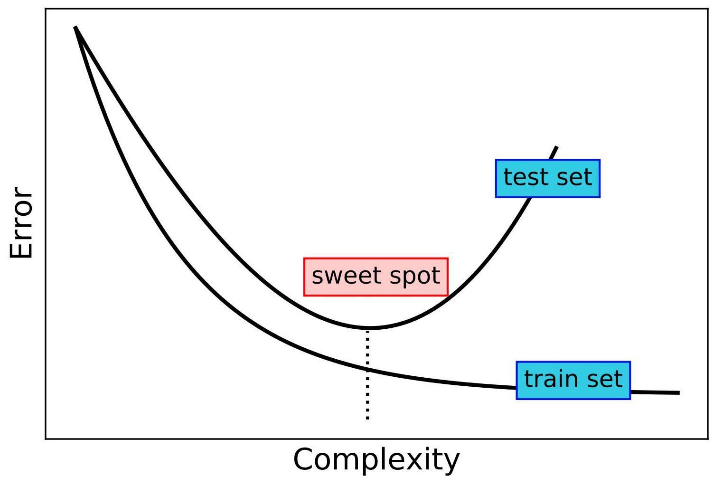
- **Overfitting** occurs when the model works really well on training set but is unable to generalize

Underfitting

- On the other hand, if the model is **too simple**
  - Everybody who owns a house buys a boat
- We might not be able to able to capture all aspects of and variability in the data
  - The model might do badly even on the training set

Choosing too simple a model is called **underfitting**

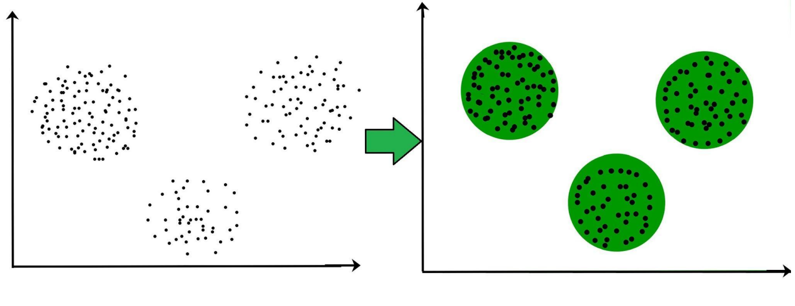
The sweet spot

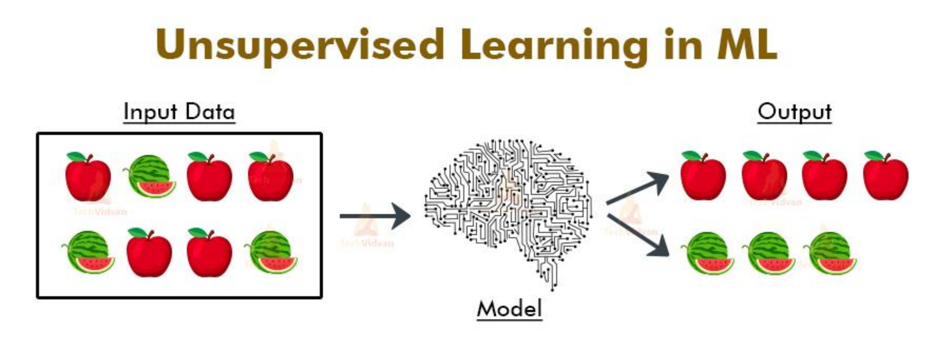


Unsupervised Machine Learning

Unsupervised Learning

- Algorithms trained using **unlabeled** (/unknown) data
- There is no **feedback**
- Can be categorized in **Clustering** and **Association**
- Example: Fruit classifier





Challenges in Unsupervised Learning

- A major **challenge** in unsupervised learning is evaluating whether the algorithm learned something useful
- Example - Facial Recognition
  - Our clustering algorithm could have grouped together all the pictures that show faces in profile and all the full-face pictures
  - This would certainly be a possible way to divide a collection of pictures of people's faces, but it is not the one we are looking for

Challenges in Unsupervised Learning

- A major **challenge** in unsupervised learning is evaluating whether the algorithm learned something useful
- Example - Facial Recognition
  - Our clustering algorithm could have grouped together all the pictures that show faces in profile and all the full-face pictures
  - This would certainly be a possible way to divide a collection of pictures of people's faces, but it is not the one we are looking for

There is no way for us to "tell" the algorithm what we are looking for

Challenges in Unsupervised Learning

- As a consequence, unsupervised algorithms are used often in an **exploratory setting**
  - Rather than as a part of a **larger automatic system**

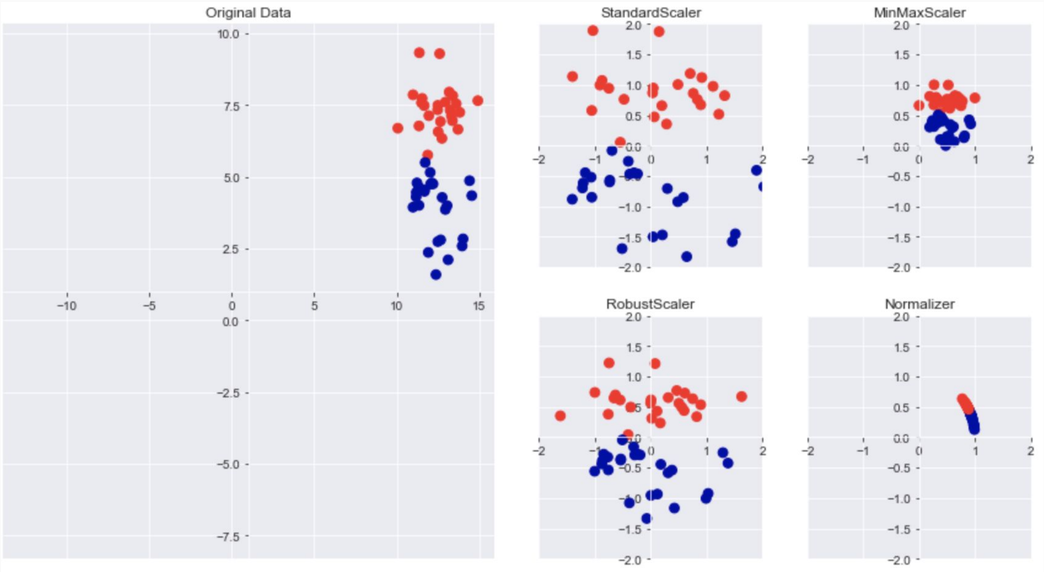
- Therefore, sometimes using unsupervised algorithms can be used as a **preprocessing** step for supervised algorithms
  - Could possibly **improve the accuracy of supervised algorithms**
  - Or **reduced** memory and time consumption

Preprocessing and Scaling

- **Neural networks** and **SVMs** are very **sensitive** to **scaling** of data
    - Therefore, adjusting the **features** so that the data representation is more suitable for these algorithms is a priority

- Often, this is a simple **per-feature rescaling** and **shift** of the data

Different kinds of Preprocessing



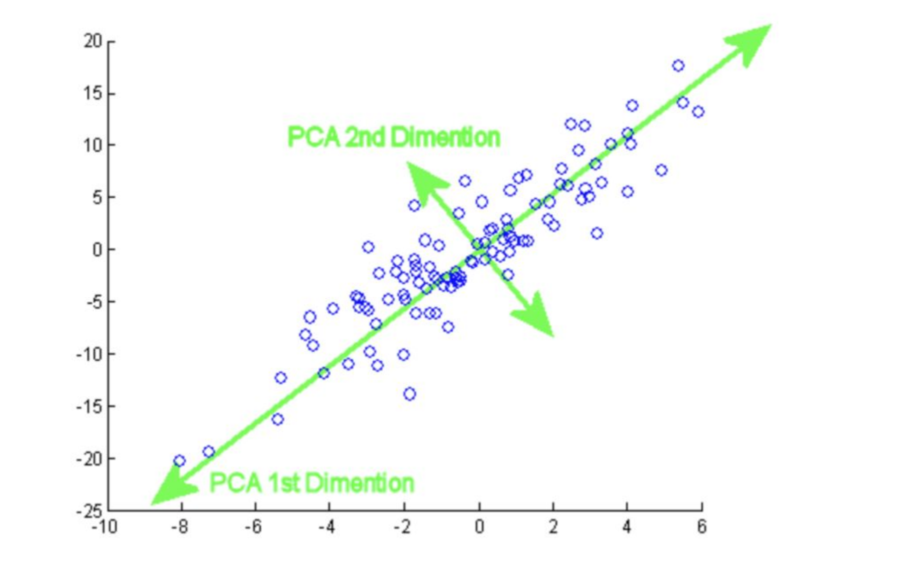
Dimensionality Reduction and Feature Extraction

- **Transforming data** using unsupervised learning can be pretty handy
  - Visualization, compressing data, finding a representation that is informative
- The most commonly used algorithms in unsupervised learning
  - Principal Component Analysis (PCA)
  - Non-negative Matrix Factorization (NMF) - for **feature extraction**
  - t-SNE which is used **visualization** using two-dimensional scatter plots

Principal Component Analysis (PCA)

- Method of **"rotating"** the dataset in a way such that the rotated features are statistically **uncorrelated**
  - Often followed by selecting only a subset of the new features
  - According to how important they are in explaining the data
- We can use PCA for **dimensionality reduction** by retaining only **some** of the principal components

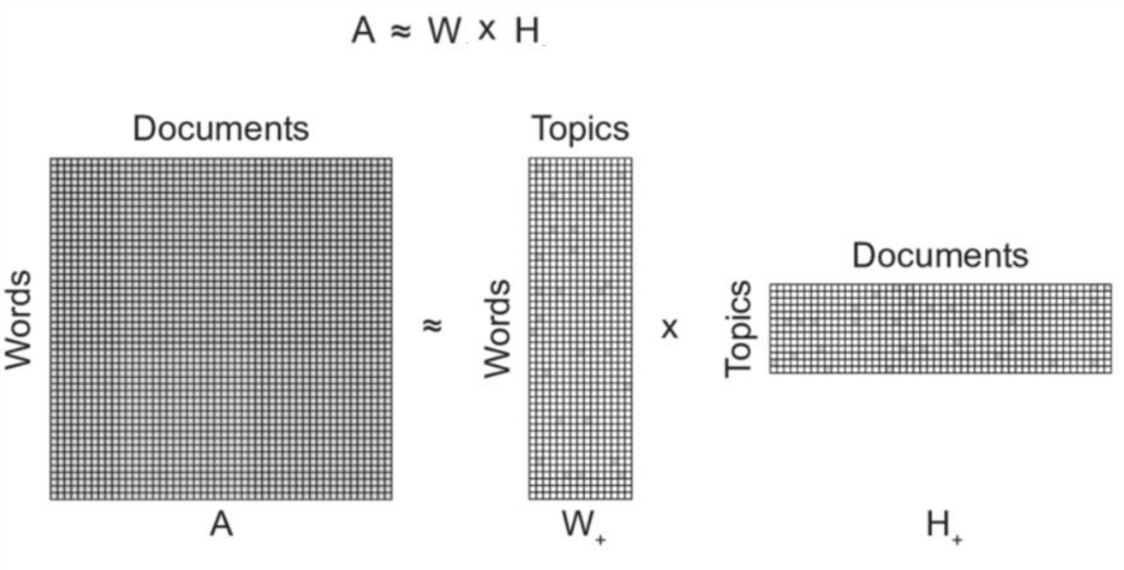
Principal Component Analysis (PCA)



Non-negative Matrix Factorization (NMF)

- Works similarly to PCA and can also be used for **dimensionality reduction**
- In PCA, we wanted components that were orthogonal and that explained as much variance of the data as possible
  - In NMF, we want the **components and coefficients** to be **non-negative**

Non-negative Matrix Factorization (NMF)

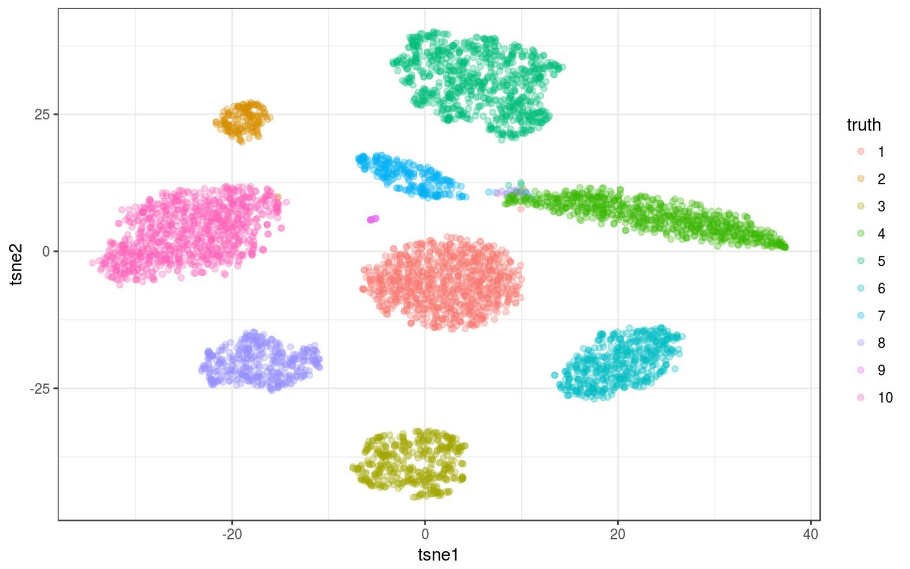


Manifold Learning with t-SNE

- Mainly aimed at **Visualization**, and so are rarely used to generate more than two new features
    - t-SNE computes a **new representation of the training data**, but **do not allow transformations of new data**

These algorithms can not be applied to a test set; rather they can only transform the data they were trained for

Manifold Learning with t-SNE



Summary

- You need to make sure that your model is neither “too simple” nor “too complex”
- Unsupervised Learning Algorithms do help Supervised Learning in preprocessing the data
    - For dimensionality reduction problems, PCA should be your go-to

That’s all Folks!

See you in the next session :)

Give us a feedback: https://bit.ly/3g6ZDlD