1 Introduction to Data Mining

In this course, we will focus on **Web Mining and Recommender Systems**. We will build models to help us understand data in order to gain insights and make predictions.

(Example.) Recommender System.

- **Prediction:** What (star-) rating will a person give to a product? For example, `rating("julian", "Pitch Black")`.
  - If we can solve a prediction task, then we can build things like a recommender system. For example, if we can predict what stars you will give to a particular movie, we can suggest those movies to that user.
- **Application:** Build a system to recommend products that people are interested in.
  - Would this person like this movie?
- **Insights:** How are opinions influenced by factors like time, gender, age, and location?

(Example.) Social Networks.

- **Prediction:** Whether two users of a social network are likely to be friends.
- **Application:** Friend recommendation features.
  - Does this person know the other person? (e.g., "people you may know").
- **Insights:** What are the features around which friendships form?
  - Do you become friends because you have some feature in common (e.g., same group)?

(Example.) Advertising.

- **Prediction:** Will I click on an advertisement?
- **Application:** Recommend relevant (or likely to be clicked on) advertisements to maximize revenue.
  - Google!
- **Insights:** What products tend to be purchased together? What do people purchase at different times of the year?

(Example.) Medical Informatics.

- **Prediction:** What symptoms will a person exhibit on their next visit to the doctor?
- **Application:** Recommend preventative treatment.
- **Insights:** How do diseases progress? How do different people progress through those stages?

1.1 Requirements for Data Mining

What do we need to do data mining?

1. Are the data associated with meaningful outcomes?

- Are the data **labeled**?
- Are the instances (relatively) independent?

> (Example.) If we're trying to build a recommender system to predict ratings (e.g., who likes this movie?), then the **label** would be the rating that we're trying to track.
>
> However, if we're trying to predict which reviews are sarcastic, then there are no labels – it's not possible to objectively identify sarcastic reviews.

2. Is there a clear objective to be optimized?

   - How will we know if we've modeled the data well?
   - Can actions be taken based on our findings?

> (Example.) Who likes this movie? How wrong were our predictions on average?

3. Is there enough data?

   - Are our results statistically significant?
   - Can features be collected?
   - Are the features useful/relevant/predictive?

2 Machine Learning Basics

In this section, we'll talk a bit more about the basics of machine learning.

2.1 Supervised Learning

Definition 2.1: Supervised Learning

Supervised Learning is the process of trying to infer from labeled data the underlying function that produced the labels associated with the data.

More abstractly, given **labeled training data** of the form

$$\{(\text{data}_1, \text{label}_1), \ldots, (\text{data}_n, \text{label}_n)\},$$

we want to estimate the function

$$f(\text{data}) \mapsto \text{labels}.$$

In other words, given the data, can we predict what the label will be? Can we figure out what form this function should take?

(Example.) Suppose we want to build a movie recommender, e.g., given a list of films, which of these films will I rate the highest?

- The **labels** here would be the **ratings** that others have given to each movie, and that I have given to other movies in the past. So, we have this huge collection of data consisting of movie ratings.
- The **data** here would be the **features** about the movie and the users who evaluated it. Essentially, the features we care to extract, anything that we think may help us predict the rating. For example:
  - Movie Features: genre, actors, MPAA rating, length, etc.
  - User Features: age, gender, location, etc.

Can we build our first supervised machine learning (or regression) model for this example? Specifically, can we estimate a star rating given the features associated with a user and the features associated with a movie?

$$f(\text{user features, movie features}) \overset{?}{\mapsto} \text{star rating}.$$

There are several solutions that we may try.

1. Design a system based on prior knowledge.

```
def prediction(user, movie):
    if user['age'] < 14:
        if movie['mpaa_rating'] == 'G':
            return 5.0
        else
            return 1.0
    else if user['age'] <= 18:
        if movie['mpaa_rating'] == 'PG':
            return 5.0
    ...
```

Disadvantages:

- Depends on possibly false assumptions about how users relate to items.
- Cannot adapt to new data/information (e.g., books).

Advantages:

- Requires no data!

Basically, hardcoding is bad for this example.

2. We can identify words that were frequently mentioned in social media posts, and recommend movie whose plot synopses use similar types of language.

Disadvantages:

- Depends on possibly false assumptions about how users relate to items.
- May not be adaptable to new settings.

Advantages:

- Requires data, but does not required *labeled* data.

3. Identify which attributes (e.g., actors, genres) are associated with positive ratings. Recommend movies that exhibit those attributes.

Disadvantages:

- Requires a (possibly large) dataset of movies with labeled ratings.

Advantages:

- Directly optimizes a measure we care about (predicting ratings).
- Easy to adapt to new settings and data.

2.2 Supervised vs. Unsupervised Learning

Learning approaches attempt to model data in order to solve a problem.

- The **unsupervised learning** approach finds patterns/relationships/structure in data, but are not optimized to solve a particular predictive task.
- The **supervised learning** approach aims to directly model the relationship between input and output variables, so that the output variables can be predicted accurately given the input.

2.3 Regression

Regression is one of the simplest supervised learning approaches to learn relationships between input variables (features) and output variables (predictions).

2.3.1 Linear Regression

Linear regression assumes a predictor of the form

$$X\theta = y,$$

where

- X is the matrix of features (the data); it's the representation of data as a matrix associated with users and items,
- θ are the unknowns (which features are relevant); it somehow explains the relationship between the features and the labels we're trying to predict,
- y is the vector of outputs (labels that we're trying to predict).

(Motivation.) Height vs. Weight.

Suppose we're given a scattered plot, where the x-axis is the height and the y-axis is the weight. Can we find a line that approxiately fits the data?

We can model this relationship by using the classic equation

$$y = mx + b.$$

Here, m is the slope and b is the intercept. In our case here, we're trying to predict

$$\text{weight} = m \cdot \text{height} + b.$$

m and b are the unknowns, and we need to fit those from the data so that the line follows the data as accurately as possible.

If we can find such a line, then we can use it to make **predictions** (i.e., estimate a person's weight given their height).

(Motivation.) Height vs. Weight vs. Age.

We can generalize this to multiple dimensions! We can use the equation

$$\text{weight} = m_1 \cdot \text{height} + m_2 \cdot \text{age} + b.$$

In matrix form, this would be

$$y = \begin{bmatrix} m_1, m_2, b \end{bmatrix} \cdot \begin{bmatrix} \text{weight} \\ \text{age} \\ 1 \end{bmatrix}.$$

Here, we might have a two-dimensional graph with height, age, and weight. Then, instead of a line, we might have a plane.