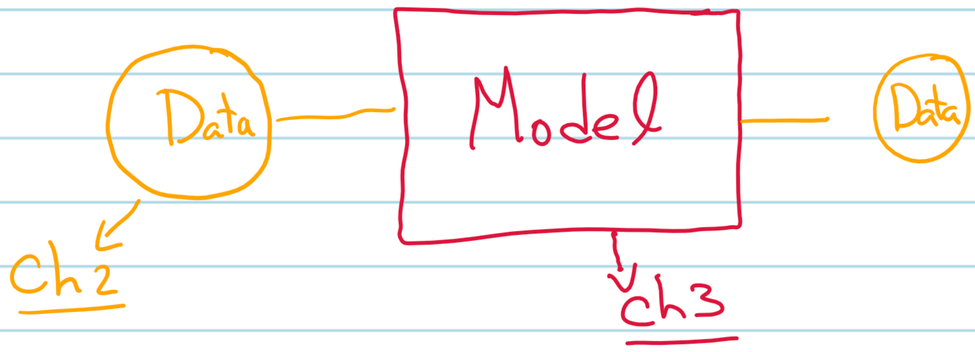
Machine Learning
in Physics _ Winter 2020

Sadegh Raeisi

Chpt. 2 _ P 1



In this chapter:

- ☐ Types of data
- ☐ Cleaning the data
- ☐ Data reduction & feature selection/transformation

Types of data

* Numerical data: X_i or $Y \in \mathbb{R}$ ⟶ This is often what we want

* Categorical

* Text

* Audio

* ⋮

*

Basically, any sort of inputs.

But in order to feed the data to the model, we often need to represent it in a language that the model can work with.

Example: $f_w(X) = w_0 + w_1 x_1 + w_2 x_2 + \cdots$ $\quad w_i \in \mathbb{R}$

$x_i \in \mathbb{R}$ ✓

$x_i \in \{\text{Apple}, \text{Orrange}\}$?

How do we deal with categorical data?

X_i is a sentence.

⋮

Categorical Data

Typically we say X_i or $Y \in \mathbb{Z}_n$, but we should be careful

$$X_i \in \{0, 1, 2\}$$

$$X_i \in \{\text{Apple}, \text{Orange}, \text{Banana}\} \longrightarrow \{0, 1, 2\}$$

are not the same. $2 > 0$ but $\text{Apple} \overset{?}{>} \text{Orange}$

is not meaningful.

One-Hot encoding

$: \mathbb{Z}_n \longrightarrow \mathbb{Z}_2^{\otimes n}$

$\{0, 1, 2\}$ → $(1,0,0)$ $(0,1,0)$ $(0,0,1)$

$x_i \longrightarrow$ 3 new features.

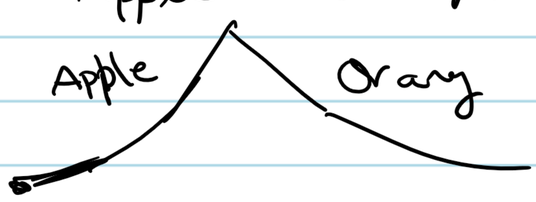
$\{\text{Apple}, \text{Orange}, \text{Banana}\}$ → $(1,0,0)$ $(0,1,0)$ $(0,0,1)$

We do this for both X_i & Y i.e. features & labels.

What encoding we use, depends on the model.

For instance, decision trees can often handle categorical data (at least conceptually)

Apple or Orange is a valid question.



But often, we need to at least convert it to numbers.

[Separate lec for text]

Data cleaning

- Missing data
- Outliers
- Saling
- Transformations

Missing Data →

. Drop the missing data ?

In a row → loose too many samples ?

In a Col. → loose too much info ?

What do you think we should do?

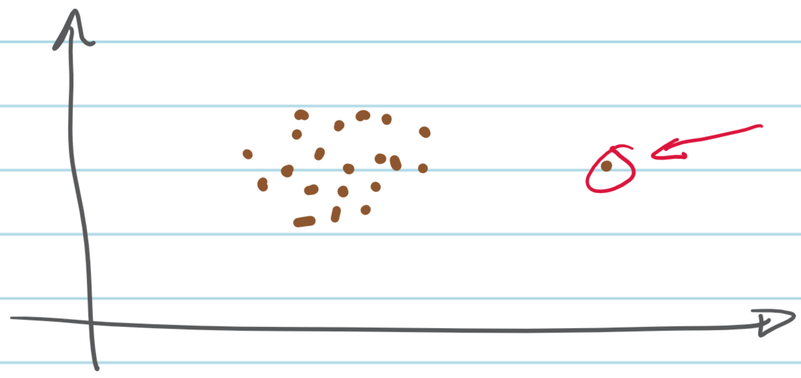
For a few samples, we can drop the samples but we can also train models that can make a guess for the missing data. → Imputation

Similarly, for a col, we need to evalute the importance of the feature, based on that, we may be able to drop a col that has too many missing data and does not play a huge role in our task.

→ If filled, add a flag col to know for which samples, it was missing.

Outliers

Depending on your data and how it is generated, you may have data points that are far off.



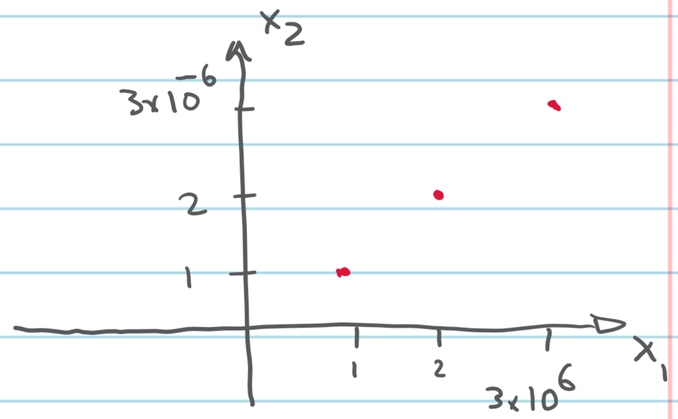
Some algorithms are robust to outliers and some are sensetive. Depending on that, you may need to either correct them e.g. 10 m → 10 km or treat them like missing data.

Question: Can you think of any algorithm that is sensetive to outliers?

Scaling:

Consider this example:

$$f(x) = w_0 + w_1 x_1$$



Try to fit a model to this data.

$f(x) = 0$ would be an answer with loss of

$$l = \sqrt{\frac{10^{-12} \times 6}{3}} \simeq 10^{-6}$$ which is fairly small.

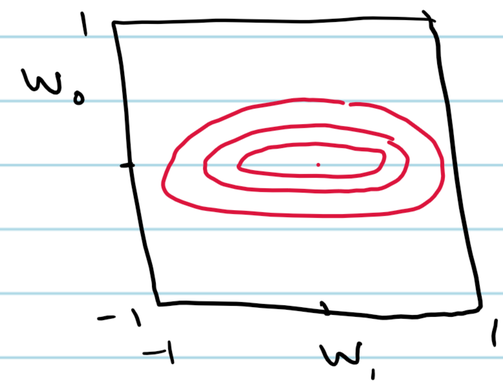
Also

It is easy to see that $f_{real}(x) = 10^{-12} x$

But for the optimization

It is really challenging

to converge to $w_0 = 0,\ w_1 = 10^{-12}$.



Either the learning rate should be $\sim 10^{-12}$ which takes forever to converge (w_0 is on a diff. scale) or if lr is to large, $w_0 \to 0$ but w_1 would miss the optimal value of 10^{-12}.

* **Standardization** $S: X \to X_S$: $\text{mean}(X_S) = 0$

$$X_S = \frac{X - \text{mean}(X)}{\text{Var}(X)}$$ $\text{Var}(X_S) = 1$

* **Scale to a range** $R(\text{min,max}) X \to X_R$, $\min(X) = \min$

$\text{Max}(X) = \max$

$$X_R = X \frac{\max - \min}{X_{max} - X_{min}} - \min$$

Other transformation

There are other transformation. For instance, for some applications, we need the data to be more normal and there are transformations for this.

Data reduction

Sometimes, there are too many features in our data, and not all of them are informative for training our model. One strategy is to reduce the features. There are two approaches:

* Feature Selection
* Feature transformation

There are two ways to decide what to remove.

One is to get rid of features that are not changing much.

The other is to look at the labels and see which features contribute less. For instance, for

$$f(\vec{X}) = w_0 + w_1 x_1 + w_2 x_2 + \dots w_n x_n$$

If some w's are too small, we can drop them.

Or with DT, the importance of the features can be used for this.

Transformation

* linear → PCA

X → SVD → Keep the more important components.

(SVD: Singular Value Decom.)

* Non-linear : → Manifold learning

These are non-linear transformation that try to, reduce the # features with the specific objective e.g. preserving the neighbourhood or shortest path.

(See nb from last year for more info.).