Working with Data

Fraida Fund

Contents

Garbage in, garbage out

Any machine learning project has to start with high-quality data.

There is a “garbage in, garbage out” rule: If you use “garbage” to train a machine learning model, you will only get “garbage” out. (And: Since you are testing on the same data, you might not even realize it is “garbage” at first! You may not realize until the model is already deployed in production!)

Before using any data

- Consider ethics concerns
- Prepare a held-out test set
- Make and check assumptions
- Check for missing data
- Identify potentially predictive features
- Look for patterns you *don't* want model to learn

Ethics concerns

Some data ethics concerns

- Bias
- Privacy
- Consent

…are just a few.

- Many social media datasets used for "offensive post" classification have biased labels (especially if they were produced without adequate training procedures in place). For example, they may label posts containing African-American dialects of English as "offensive" much more often. Source, User-friendly article
- On the anonymity of the Facebook dataset
- 70,000 OkCupid Users Just Had Their Data Published; OkCupid Study Reveals the Perils of Big-Data Science; Ethics, scientific consent and OKCupid
- IBM didn't inform people when it used their Flickr photos for facial recognition training

Prepare a held-out test set

- If we plan to use some data for machine learning, we *must* set aside a "test set" before we do *anything* with the data.
- We will explain in a future lesson why this is so important.

Our next steps - checking assumptions, handling missing data, identifying potentially predictive features, identifying "bad" patterns - will be on the part of the data that is not "held out" as a test set.

Make and check assumptions

It's always a good idea to "sanity check" your data - before you look at it, think about what you expect to see. Then check to make sure your expectations are realized.

Look at plots of data, summary statistics, etc. and consider general trends.

Example: author citation data (1)

Data analysis: use PubMed, and identify the year of first publication for the 100,000 most cited authors.

What are our expectations about what this should look like?

Example: author citation data (2)

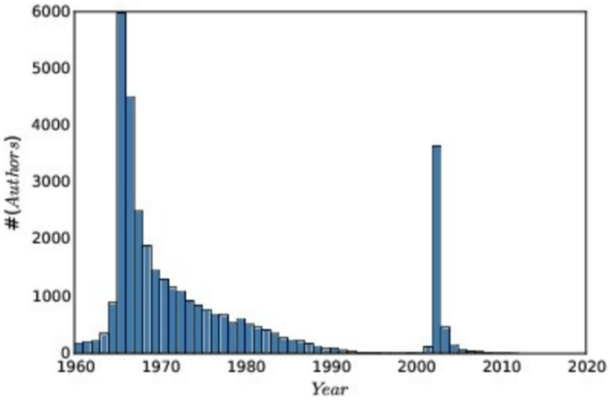

Figure 1: Does this look reasonable?

We can think of many potential explanations for this pattern, even though it is actually a data artifact.

The true explanation: in 2002, PubMed started using full first names in authors instead of just initials. The same author is represented in the dataset as a “new” author with a first date of publication in 2002.

Example: author citation data (3)

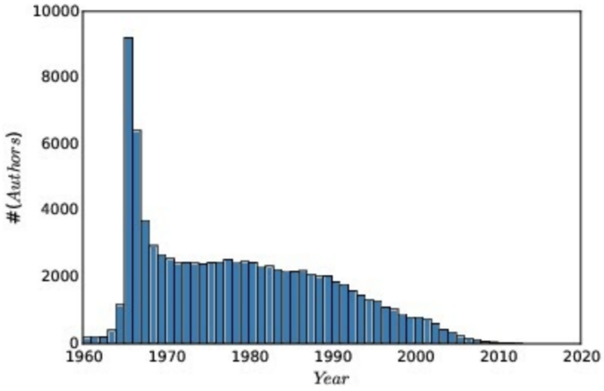

Figure 2: The real distribution, after name unification. Example via Steven Skiena @ Stony Brook U.

Handling unreasonable data

How should you handle unreasonable values, data that does not match expectations, or "outliers"? It depends!

- e.g. suppose in a dataset of voter information, some have impossible year of birth - would make the voter over 120 years old. (The reason: Voters with no known DOB, who registered before DOB was required, are often encoded with a January 1900 DOB.)
- **not** a good idea to just remove outliers unless you are sure they are a data entry error or otherwise not a "true" value.
- Even if an outlier is due to some sort of error, if you remove them, you may skew the dataset (as in the 1/1/1900 voters example).

Consider the possibility of:

- Different units, time zones, etc. in different rows
- Same value represented several different ways (e.g. names, dates)
- Missing data encoded as zero

Look for missing data

Indicators of missing data

- Rows that have `NaN` values
- Rows that are *not there*

Examples of missing data

- Twitter API terms of use don't allow researchers to share tweets directly, only message IDs (except for limited distribution, e.g. by email). To reproduce the dataset, you use the Twitter API to download messages using their IDs. But, tweets that have been removed are not available - the distribution of removed tweets is not flat! (For example: you might end up with a dataset that has offensive posts but few "obvious" offensive posts.)
- A dataset of Tweets following Hurricane Sandy makes it looks like Manhattan was the hub of the disaster, because of power blackouts and limited cell service in the most affected areas. Source
- The City of Boston released a smartphone app that uses accelerometer and GPS data to detect potholes and report them automatically. But, low income and older residents are less likely to have smartphones, so this dataset presents a skewed view of where potholes are. Source

Types of "missingness"

- Completely random
- Correlated with something that is in data
- Correlated with something not in data

These are often referred to using this standard terminology (which can be confusing):

- Missing *completely* at random: missingness not correlated with any feature or the target variable.
- Missing at random: missingness correlated with something that is in data.
- Missing not at random: missingness correlated with something that is not in data.

Handling missing data

How should you handle little bits of missing data? It always depends on the data and the circumstances. Some possibilities include:

- omit the row
- fill with mean, median, max, mode…
- fill back/forward (ordered rows)
- train a model on the rest of the data to "predict" the missing value

You generally have to know why the data is missing, to understand the best way to handle it.

Identify predictive features

"Predictive" means "related to the target variable" (any kind of relationship!)

How do we look for predictive features?

- Numeric (continuous) features
- Categorical features
- Graphical features
- Text features

“Bad patterns” (and data leakage)

When looking for predictive features, also ask yourself - are there patterns in the data that you *don’t* want your model to learn?

COVID-19 chest radiography (1)

- **Problem**: diagnose COVID-19 from chest radiography images
- **Input**: image of chest X-ray (or other radiography)
- **Target variable**: COVID or no COVID

COVID-19 chest radiography (2)

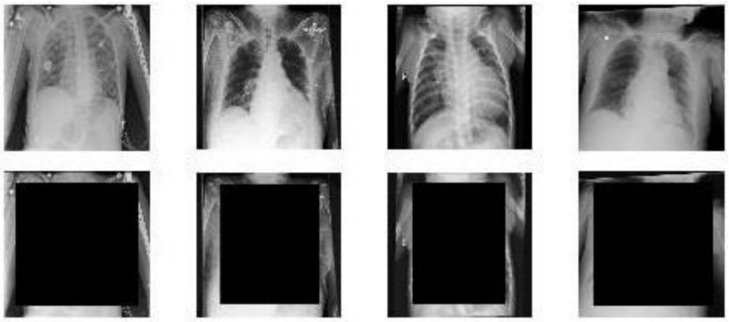

Fig. 5. Original and transformed samples from the 4 datasets, 300 sized black square (Left to right: COV, NIH, CHE, KAG)

Figure 3: Neural networks can classify the source dataset of these chest X-ray images, even *without lungs*! Source

Between January and October 2020, more than 2000 papers were published that claimed to use machine learning to diagnose COVID-19 patients based on chest X-rays or other radiography. But a later review found that “none of the models identified are of potential clinical use due to methodological flaws and/or underlying biases”.

To train these models, people used an emerging COVID-19 chest X-ray dataset, along with one or more existing chest X-ray dataset, for example a pre-existing dataset used to try and classify viral vs. bacterial pneumonia.

The problem is that the chest X-rays for each dataset were so “distinctive” to that dataset, that a neural network could be trained with high accuracy to classify an image into its source dataset, even without the lungs showing!

COVID-19 chest radiography (2)

Findings:

- some non-COVID datasets were pediatric images, COVID images were adult
- there were dataset-level differences in patient positioning
- many COVID images came from screenshots of published papers, which often had text, arrows, or other annotations over the images. (Some non-COVID images did, too.)

COVID-19 chest radiography (3)

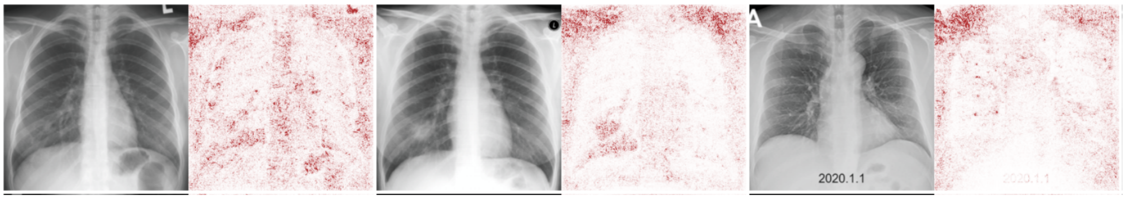

Figure 4: Saliency map showing the “important” pixels for classification. Source

These findings are based on techniques like

- saliency maps, where the model is made to highlight the part of the image (the pixels) that it considered most relevant to its decision.
- using generative models and asking it to take a COVID-negative X-ray and make it positive (or v.v.)

Many of the findings are not easy to understand without domain knowledge (e.g. knowing what part of the X-ray *should* be important and what part should not be.) For example: should the diaphragm area be helpful?

Data leakage

In machine learning, we train models on a training set of data, then evaluate their performance on a set of data that was not used in training. “Data leakage” can occur when

- information “leaks” between the held-out test set and the training set
- or, information from the target variable (that should not/will not be available when making a prediction) leaks into the feature data
- or any scenario where the model may learn a “pattern” that will not be present during the “real” task.

Data leakage: the model uses something (a feature, a pattern in the data, actual data points) that will not be available during “real” prediction task.

Some types of data leakage

- Learning from a feature that is a proxy for target variable, but that won’t be available
- Learning from adjacent temporal data
- Learning from duplicate data
- Learning from features that are not available at prediction time (e.g. data from the future)

Signs of potential data leakage (after training)

- Performance is “too good to be true”
- Unexpected behavior of model (e.g. learns from a feature that shouldn’t help)

Detecting data leakage

- Exploratory data analysis
- Study the data before, during, and after you use it!
- Explainable ML methods
- Early testing in production

Many more data problems...

- **Data is not representative of your target situation**. For example, you are training a model to predict the spread of infectious disease for a NYC-based health startup, but you are using data from another country.
- **Data or situation changes over time**. For example, imagine you train a machine learning model to classify loan applications. However, if the economy changes, applicants that were previously considered credit-worthy might not be anymore despite having the same income, as the lender becomes more risk-averse. Similarly, if wages increase across the board, the income standard for a loan would increase.