ML Ready Datasets

ESS 469/569

University of Washington

Let us review

By now, we all should feel comfortable with the following concepts:

- Data considerations (modality, dimensions, distributions, etc.)
- Using version control and Python
- Manipulating arrays, manipulating tabular dataframes
- Resampling
- Basic statistical descriptors
- Transformations (spectral and other feature extraction)
- Signal vs. noise
- Dimensionality

Now, how do we go from a question to a ML solution?

When starting a ML project, it is normal to not know exactly how you will solve it with ML/AI techniques—especially when such techniques often are black boxes!

Consider your problem

Before jumping to a specific algorithm, define your problem in the *most general terms*.

Consider your problem

Before jumping to a specific algorithm, define your problem in the *most general terms*.

That is, tell us (or yourself or anyone who will listen) what your question is. Are you trying to transform A into B? Predict Y from X? Determine how one variable is related to another?

Consider your problem

Before jumping to a specific algorithm, define your problem in the *most general terms*.

That is, tell us (or yourself or anyone who will listen) what your question is. Are you trying to transform A into B? Predict Y from X? Determine how one variable is related to another?

If the problem is computable, then it likely can be addressed using some ML method.

Then, get a sense for what is realistically possible

Review existing literature (*not* necessarily ML-specific literature!)

Then, get a sense for what is realistically possible

Review existing literature (*not* necessarily ML-specific literature!)

If experts can only achieve x accuracy for some task, that should be your first benchmark.

Then, get a sense for what is realistically possible

Review existing literature (*not* necessarily ML-specific literature!)

If experts can only achieve x accuracy for some task, that should be your first benchmark.

If x accuracy is “good enough”, then your solution may not need to do better!

Look into general ML approaches

If your problem involves recognizing textures in images, read papers that describe both the history and state-of-the-art solutions to that general problem. Likewise, if your project involves dividing data into classes on the basis of several variables, read papers on classification methods.

Look into general ML approaches

If your problem involves recognizing textures in images, read papers that describe both the history and state-of-the-art solutions to that general problem. Likewise, if your project involves dividing data into classes on the basis of several variables, read papers on classification methods.

Like in the previous step, get a sense for what is possible for your general question.

Look into general ML approaches

If your problem involves recognizing textures in images, read papers that describe both the history and state-of-the-art solutions to that general problem. Likewise, if your project involves dividing data into classes on the basis of several variables, read papers on classification methods.

Like in the previous step, get a sense for what is possible for your general question.

Determine whether your final approach will be *narrow* or *general* (and to what extent).

Compile your data

Get your data into one location (e.g., your home folder).

Compile your data

Get your data into one location (e.g., your home folder).

This process can take some time, so do it early.

Compile your data

Get your data into one location (e.g., your home folder).

This process can take some time, so do it early.

However you assemble your data, you should document every step!

Organize your data

Okay, you now have a folder with all of your data. Now what?

Organize your data

Okay, you now have a folder with all of your data. Now what?

It is time to make it easy to work with!

Organize your data

Okay, you now have a folder with all of your data. Now what?

It is time to make it easy to work with!

Organize your data into machine-readable formats and data structures. For example, arrange data in numpy arrays, Xarrays, or pandas. Or, save data and its attributes in Zarr, H5, CSV formats for easy retrieval.

Now what?

What is the one thing you must do when presenting a project in this class?

Now what?

What is the one thing you must do when presenting a project in this class?

Characterize your data. Histograms, crossplots, box plots, correlation matrices—we want to see them all.

Now what?

What is the one thing you must do when presenting a project in this class?

Characterize your data. Histograms, crossplots, box plots, correlation matrices—we want to see them all.

Tell us (and yourself) something about the data that will go through your eventual ML/AI pipeline.

Then, consider data manipulations

For example: - Extract statistical, temporal, or spectral features (use tsfresh, tsfel, ...) - Transform the data into Fourier or Wavelet space (use scipy fft or cwt module) - Reduce dimensionality by taking the PCA or ICA of the data. Save these features into file or metadata (use scikit-learn PCA or FastICA module).

Finally, consider data augmentation

Say you have a small[1] dataset. One thing you might do to address this issue is augment your data (e.g., create modified copies of your data).

[1] The definition of "small" is problem dependent. 1000 observations may be more than enough for simple regression analyses. The same number of observations may not be adequate for image segmentation tasks. Consider the extent of your problem space.

Finally, consider data augmentation

Say you have a small[1] dataset. One thing you might do to address this issue is augment your data (e.g., create modified copies of your data).

Bootstrap your data. Or use Monte Carlo methods to propagate uncertainties. If you have images, skew, stretch, rotate, and mirror them.

[1]The definition of “small” is problem dependent. 1000 observations may be more than enough for simple regression analyses. The same number of observations may not be adequate for image segmentation tasks. Consider the extent of your problem space.

Save your processed data

Since all of you are model coders, you will have saved the entire data processing workflow in well annotated notebooks (or scripts or .md files).

Save your processed data

Since all of you are model coders, you will have saved the entire data processing workflow in well annotated notebooks (or scripts or .md files). Consider using built in modules for the data pre-processing tasks (e.g., scikit-learn Pipeline).