Week 8

???

8.1 Machine Learning

10/22:

- Lecture 12 recap.
  - Different electronic parameters capture different features of molecules.
  - Electronic parameters.
    - Hammett parameters (σ).
      - ➢ Examples include σ_p, σ_m, σ^+, and σ^-.
    - Nucleophilicity and electrophilicity.
      - ➢ Examples include Mayr and Swain-Scott.
    - NMR or IR shifts.
    - You can also parameterize via the energy of certain electrons (e.g., σ, σ^*, lp, etc.).
  - Steric parameters.
    - A values: Historic.
    - Sterimol (L, B_1, and B_5): Common.
    - Taft (E_s) and Charton for stereoelectronic.
    - Bite angle, cone angle, and PBV for sterics in catalysis.
  - Why do we use parameters?
    - Correlating parameters to reaction outcomes (e.g., rate, selectivity, etc.) lets us...
      - ➢ Predict reaction outcomes;
      - ➢ Design better catalysts;
      - ➢ Learn something about the reaction mechanism (this is especially important for this class).
- Announcements.
  - Don't forget the exam!
  - This is Masha's last lecture. Fill out the teaching evaluations for Masha and Jonathan at the end of the course! Masha's evals will influence her tenure decision, and Jonathan's could help win him a teaching award.
- Today: More complex relationships between the input parameters from last time and our output.
  - This is machine learning (ML)!
  - Masha will focus on the applications of ML to organic chemistry, but please read more about the math and other applications if you're interested!
- There will be a lot of vocab in this lecture, starting with the definition of **AI**.

- **Artificial intelligence**: The development of computer systems able to perform tasks that normally require human intelligence. *Also known as* **AI**.
- Examples of such tasks.
  - Speech recognition, decision making, visual perception.
  - Not just things like calculus, but things that require a "greater" level of intelligence.
- Under the umbrella of AI falls **ML**.
- **Machine learning**: A subfield of AI that allows computer systems to learn and adapt without explicit instructions or programming. *Also known as* **ML**.
  - ML is characterized by the computer system being able to do things that we didn't explicitly program it to do.
- In the context of ML, we also have an explicit definition of **learning**.
- **Learning**: A computer program is said to form some experience (E) with respect to a task (T) and a measure of performance (P; aka the "performance metric"), if it's performance on T — as measured by P — improves with E.
  - This gets into the Turing test, and what it really means to know and to learn and to be conscious. This is more the realm of philosophy, and we won't get into that.
  - It's not like it did great from the beginning; it's that it had to get better with more experience.
- Reviews on the subject of ML in chemistry.
  - A great one to start for organic chemists: Williams et al. (2021). Four big-name corresponding authors.
    - Tobias Gensch: He's new, but we'll know him soon.
    - Sigman: The pioneer of multivariate linear regression.
    - Doyle: First to publish ML in chemistry; her 2018 *Science* paper — Ahneman et al. (2018) — exploded the field.
    - Anslyn: Wrote our textbook; the gold standard of Phys Orgo.
  - Any review publsihed by Doyle or Sigman will be great to read.
  - There are also great reviews from Connor Coley, Bill Green, and Klavs Jensen.
- Types of learning: **Supervised** and **unsupervised**.
- **Supervised** (learning): ML that has **labeled** training data.
  - This type of ML analyzes the labeled data and then makes a guess on unlabeled data. After the model guesses, we evaluate its performance.
  - Example: Show my model 100 reactions (with their yields labeled), and then have it guess the yield of a new reaction it's never seen before.
  - This is called "supervised" learning, because after the model guesses, *we* need to show it the right answer (i.e., the label).
  - Example: Spam filters.
    - These separate spam from "ham," the technical term for good emails.
    - We train such models by showing them a bunch of spam emails and a bunch of ham emails so that they "learn" what spam looks like.
    - The model looks for typos, weird email addresses, requests for money, etc.
- **Labeled** (training data): A set of data in which each data point (or datum) has an input and output label.

- **Unsupervised** (learning): ML that has **unlabeled** training data.
  - This type of ML tries to uncover relationships between data and find patterns.
  - This is "unsupervised" because there is no right answer, no guidance, no yield.
  - A common approach: **Clustering**.
  - Example: Netflix recommends movies that are similar to each other (i.e., which share common actors, common runtime, common genre labels, common people who have watched them, etc.).
- **Clustering**: Grouping together similar data.
- A really common approach is to do both of these at once in **semi-supervised** learning, our secret third option.
- **Semi-supervised** (learning): ML that splits data into a small labeled dataset and a big unlabeled dataset.
  - We group data together and assign a label to the group.
  - Example: Image classification, i.e., to answer the question, "which photos are of the same animal?"
    - An unsupervised ML finds similar images, and then a few of those get labeled "cat," so the whole group gets labeled "cat."
    - This is how self-driving cars and Captcha work. When you help Captcha find all the images with stairs, you're (nonconsensually) providing labels to help train image recognition models!
- We'll focus on supervised learning for the rest of today.
- Two types of supervised learning: **Classification** and **regression**.
- **Classification**: The output/label is a category.
  - There are a finite number of options.
  - Example: Photos are "cat," "dog," or "human."
- **Regression**: The output/label is a continuous number.
  - There are an infinite number of options.
  - Example: We could model the cost of a house as a function of house properties (e.g., the year it was built, the year we're trying to buy it, the cost of the surrounding homes, the neighborhood school system, etc.).
  - Example: Model ΔG as a function of reaction parameters.
    - This is Hammett plots! That was linear regression, so that's why we call this, "regression."[1]
  - Far more common in chemistry.
  - Formal definition: A statistical technique for determining the relationship between independent or explanatory variables (x) and dependent or response variables (y).
- **Linear regression**: Describe the relationship between x and y as a straight line.
  - Fitting to $y = mx + b$.
  - Example: $\log(k_X/k_H) = \rho\sigma$.
  - Some people don't call this ML; they call this "statistics." But that distinction is really only fought over by people who care about semantics or credit. So you may hear some strong opinions in the field (e.g., Sigman doesn't call it ML), but it's just labels at the end of the day (in Masha's opinion).

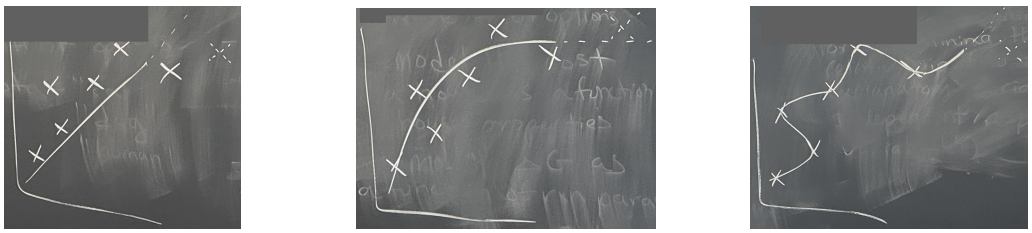
[1] Is there a "second time you hear it" effect in psychology that mimics ML? Unlikely to place emphasis on something the first time we hear it (e.g., Dad saying that there are crazy jobs for smart people/Maya telling me about the email tracker), but more likely when we hear it again (e.g., Carina Hong's job/Dylan Miars telling me about the email tracker). Relation to retention in learning!

- **Multivariate** (linear regression): Multiple x and y variables.
  - This is all the work of Matt Sigman, building off of his classic Hammett paper (Figure 7.8) that we reviewed last lecture.
- The ML workflow: Here are the steps if you want to go into lab and plan a project.
  0. Know or define your goal and application.
     - Why do you want this model?
     - What do you want it to do?
     - Why do you want to use it?
     - Why would anyone care about it?
     - A model that can predict yield to a decimal point will need tens of thousands of data points.
       - If you want a model to help you refine a ligand for a reaction that you've already studied pretty well, that's a good use.
     - ML is fundamentally an engineering solution, so you better have a practical use for this tool you've built.
  1. Data collection.
     - How much data?
     - Labeled or unlabeled?
     - Will this data come from the literature or from experiment?
       - This is an especially relevant consideration in chemistry.
     - Be aware of bias; we, as a field, tend to overreport high-yield reactions and underreport low-yield reactions. So if we train a model based just on the literature, it will think all chemical reactions are high yield.
     - The adage here: Garbage in = garbage out.
       - If you train a model on bad data, you're going to have a bad model.
     - So we can go into the lab and get a bunch of low-yield reactions and feel good about it, which almost never happens!
     - Warning: If you go to Reaxys and dump a bunch of data into a model, the error rate is 30-60% (typos and such).
       - Notoriously, the patent database used to be 60% wrong due to bad data-scraping.
  2. **Parameterization**.
     - Categorical descriptors (common for solvents, salts, and additives).
       - Tell your model that these 10 were run in toluene, these 10 in DCM, etc. That's a common category approach.
     - Chemically meaningful parameters.
       - This is all of last lecture: σ values, sterimol values, etc.
       - These are chemically meaningful because they capture a feature important to reactivity.
     - Graph networks: Atoms are nodes, bonds are edges, etc.
     - Molecular fingerprints: Lists of functional groups.
       - Example: My molecule contains a ketone, an ester, two methynes, etc.
     - SMILES, or its derivatives: This is how ChemDraw encodes molecules.
       - Example: Cyclohexane is "C1CCCCC1".
       - This string is just text, so then we can use LLMs.
       - Masha doesn't think these work that well, but they do exist.
     - Most ML papers mess up by this point: They either got bad data, or misrepresented it.
     - Most ML applications use millions or billions of datapoints, so chemistry is a bit unique in that it uses dozens of data points. How to make that tenable is a big question in both the chemistry and computer science communities!

3. Data preprocessing: Cleaning up the data for modeling.
   - Technique: Normalization.
     - Make all parameters lie in the range of 0-1.
       - ➢ To do this, just divide by the maximum value.
     - Example: Sterimol values of 1, 4, and 8 become 0.125, 0.5, and 1; charge values of 0.01, 0.04, and 0.08 become 0.125, 0.5, and 1.
     - Normalization helps us make sure that the model doesn't think sterimol values are 100 times more important than charge. Essentially, it prevents bias toward parameters with large values.
   - Technique: Reduce the number of parameters (if needed).
     - Sigman recommends 8 data points per single parameter.
     - If we have more parameters than data points, we have an **overfit** model (see Figure 8.1c), and that is no good.
   - The simplest model architecture (Occam's razor) is the best model.
4. Data sampling: Splitting data into **training**, **validation**, and **test** sets.
   - We also sometimes have an **out-of-sample** set.
   - So we have a model, but just like in science, we need the model to make useful predictions for it to be a good model.
   - In the validation set, the model is going to show us it's performance.
     - Example: If linear regression does 70% right and multivariate linear regression does 90% right, we go with the multivariate architecture.
   - Then the performance of the model that we report is the performance on the final (test) set.
   - Overexposing your model to the test set invalidates the model; this is called **data leakage**.
   - Note: Chemists tend to mix up the "validation" and "test" sets in their writing.
     - Just make sure that we have real evidence that the model works on *unseen* data.
     - If you see people say, "we used the validation set to test the model, and then applied the model to the test set," that's not a bad thing; that's just a semantic error.
5. Training and testing: Evaluate the performance of different model architectures based on **metrics**.
   - Example metrics: Accuracy, percent data explained, R^2, RMSE (root-mean-square error).
   - Here, we run control and baseline models to predict the average and mode.
6. Interpretation and prediction: Use the trained model to predict reaction results, guide catalyst design, or present mechanistic hypotheses.
   - Best practice: Test the model experimentally.
   - Relating back to Point 0: Make sure the model is worth making.
     - Make sure you use it for something cool; you don't need a chainsaw to hammer in a nail.

- **Parameterization**: Converting chemical information to **machine-readable** formats.
- **Machine-readable** (input): A number, binary value, graph representation, etc.
- **Training** (set): The set on which the model learns the trends.
  - Roughly 70% of the data.
- **Validation** (set): The subset of the training set that help you choose between model architectures.
- **Test** (set): The set on which we evaluate the model performance.
  - Roughly 30% of the data.
    - The actual numbers here are up to you! People do everything from 90 : 10 to 60 : 40.
  - We really only want to use the test set once or twice.

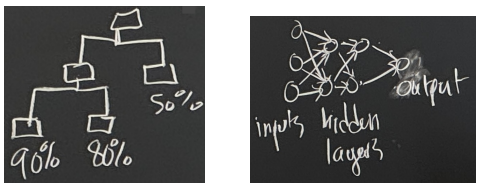
- **Out-of-sample** (set): A set that helps us further validate generalizability or **extrapolation**. *Also known as* **experimental validation** (set).
- **Data leakage**: The mixing of data between the test set and the training set. *Also known as* **poor data hygeine**.
- **Extrapolation**: The ability to make predictions beyond the training set.
- Memo: Extrapolation vs. **interpolation**?
  - There definitely is a difference.
  - For an example of true extrapolation, see Sigman's paper on extrapolating a model trained on ligands with ee below 80% to find ligands with ee beyond 80%.
- Don't report your training set performance!
- **Overfit** (model): A model that can predict the training set, but not new data.
  - Such models cannot generalize, and they definitely cannot extrapolate.
  - See Figure 8.1c.
- There are actually three types of fitting.



(a) Underfit. (b) Good fit. (c) Overfit.

Figure 8.1: Fitting machine learning models.

  - Figure 8.1a: Underfitting.
    - Doesn't generalize to new data.
  - Figure 8.1b: Good fitting.
    - Generalizes well to new data.
  - Figure 8.1c: Overfitting.
    - Doesn't generalize at all to new data.
    - This is tempting to chemists, because it gives them a good-looking model. But that's not actual model; that's fraud!
- Model architectures.



(a) Decision tree. (b) Neural network.

Figure 8.2: Machine learning model architectures.

  - These exist on a spectrum from models with low complexity and high interpretability to models with high complexity and low interpretability.
  - Lowest complexity and highest interpretability: Linear regression.
  - k-nearest neighbors (knn).
    - Basic idea: Our ligand is close to something with high ee, so we'll probably get high ee, too.
  - Decision trees (Figure 8.2a).
    - Answer questions such as, "high electronegativity or low electronegativity," and correlates that to ees.
  - Random forest.
    - A subset of decision trees.
    - Make a lot of trees and average the result.
    - Called a "forest" because there are many trees!
  - Highest complexity and lowest interpretability: Neural networks (Figure 8.2b).
    - Like the brain: Inputs, through hidden layers, that converge on an output.
    - This year's physics Nobel Prize went to neural networks; Masha's not quite sure how these are physics, but they really have changed their game.
    - They're very powerful, but extremely complex (so often overfit and not very interpretable).
      - Called "black box models."
      - Not good for mechanisms!

- Masha also has some additional notes on model architectures that she will post on Canvas.