Each condition should fall into one of the following

* Categorical (e.g. crystallization method)
* Continuous (e.g. pH)
* Continuous plus binary (e.g. sodium chloride)
* Bicontinuous plus binary (e.g. PEG)

Metadata has top row with condition and row 2 as encoding

Within the metadata, features are encoded as follows:

* Categorical: by name, convert to one-hot vector within the model and use weights on those
* Continuous: value only
* Continuous plus binary: value can include 0; use zero vs non-zero to handle the binary prediction and use only those with the continuous value for the continuous condition; do loss separately for each set of weights
* Bicontinuous plus binary: subcondition, concentration separated by commas like (for PEG) 4000, 0.2 and train the binary on whether the concentration is non-zero

Training

* Split into 2020+ test set and before-2020 train, split the train into 2018-2019 validation and before-2018 train. Try different hyperparameters (beta, delta) on 2018-2019 validation set and use gradient descent on the before-2018 train set
* Since loss is too variable, iterate until weights converge (less than 0.2 difference in last 1000 steps)
* Once converged, fix the weights and run over full train set to get optimal separator thresholds

Testing

* For categorical predictions, use argmax and test proportion correct
* For continuous, use MAE of point estimate and interval size as long as the interval is around its reported confidence
* For continuous plus binary, using the determined thresholds, assign 0/1 and compute desired statistics; only for cases where the condition actually was present; for the continuous part, use MAE of point estimate and interval size as long as the capture rate is around its reported confidence
* For bicontinuous plus binary, using the determined thresholds, assign 0/1 and compute desired statistics; only for cases where the condition actually was present; use MAE of point estimate and interval size for each of the two variables as long as the capture rates are around their reported confidences

Order of things to get done and timeline:

1. Write script to download metadata from BMCD and improve data scripts for metadata from RCSB (also need to get addition dates for test/train splits) (3 hours for BMCD script, 6 hours for improvement on RCSB script, 5 hours for miscellaneous improvements and data formatting)
2. Redownload all RCSB cif files to get a final set of amino acid sequences to be extracted and fit (1 hour)
3. Determine a way to give a probability or something equivalent for the binary objects (e.g. your protein is more likely to use this condition than 98% of proteins (using yhat density); modeled thresholds predict your protein will/will not use this condition (from TPR-FPR)) (30 mins)
4. Create a multi-class estimator (3 hours for math, 3 hours for write up, 3 hours for implementation)
5. Create a bicontinuous kernel density estimator (2 hours for math, 2 hours for write up, 4 hours for implementation)
6. Once models are done, thoroughly check model math and code one more time (at least understand why too narrow intervals make the pH weights break (1.5 hours per mode \* 4 models)
7. Join all the models into one system that can query a protein and then manage all the weight updating for each condition (1 hour for planning, 5 hours implementation)
8. Create test functions that give summaries and all density objects (5 hours implementation)
9. Run test/train/validation split by newest additions (3 hours creating workflows, 2 hours troubleshooting)
10. Write up results (10 hours processing data, 3 hours figuring out GitHub, 7 hours writing)
11. Conda package (5 hours figuring out Conda, 5 hours formatting and compiling)

Total time: 84.5 hours

Potential breakdown: 1 full day processing metadata, scattered hours redownloading data, 1 day for multi-class estimator, 1 day for bicontinuous estimator, 1 half day to check math/code, 1 full day creating model that can run everything at once and test on new data, 1 half day to get the test/train/validation working properly, 2 days to write up results, 1 full day to create the Conda package

Total devoted days: 9

Future steps: implement a structural minhash algorithm to get the similarity between structures and feed this in instead of the amino acid minhashes. Use alphafold to predict the structure when fed in