# Lab 2: Weather Prediction

## Data Collection

Data was gathered automatically from the forecast.weather.gov site using the Python requests library. Any array of cities was chosen, initially Rochester, Buffalo, and Detroit from the assignment page and later scaled up based on the autograder on GradeScope. CF6 reports were isolated from <pre/> tags and parsed into a dictionary format with a WeatherReport class. The class had utilities for managing date strings and was useful for aggregating different reports into a standard format.
Test data was stored in raw.json, with keys for attributes, training, and testing. Training and testing are divided into key-value pairs of dates and dictionaries of city keys and WeatherReport values as vectors. Multiple training-to-testing ratios were considered for the ~50 months of reports, but we eventually decided on the first 14 for testing and the remainder for training. This decision was made in an attempt to not bias data towards certain months of the year (i.e., training on 12 month cycles for 3 years [36 months]). Better results were achieved when using 2 months of testing data on 4 years of training, but this means that 10 months of the year would have been underrepresented.

## Feature Selection

Feature selection was ultimately our downfall for this lab. In our hubris, we attempted to construct data comparisons automatically, mostly from generating pairs of city-Rochester groupings and comparing attributes like average temperature. Towards the end of development, we started to feel like we were on the right track but time constraints led to poor decisions and a generalized function for producing vectors was produced. The trickiest part with these generations was the complexity of the function for transforming multiple reports into a single vector of booleans (or None when data was missing, which was probably not an intelligent choice but it *seemed* like Python was able to generate branches for missing data). Our lack of foresight to pare down was a good learning experience. Feature selection is (or would have been) broken down as follows

- Rochester is hotter today
    - On previous day, did it rain (more than 0.2) in cities west of Rochester (1 dim. each)?
    - On previous day, did it rain in Rochester?
    - On previous day, did it snow (more than 0.2) in cities west of Rochester (1 dim. each)?
    - On previous day, did it snow in Rochester?
    - On previous day, did western cities have higher DEP than Rochester (1 dim. each)?
    - On previous day, did Rochester have higher DEP than today?

    - On previous day, were western cities warmer than Rochester (1 dim. each)?

- Rochester is hotter than average today
    - Presumably fairly similar to the previous question, but training format might be able to be factored in since the report has the information as a month progresses and there should be historical data. If not using this, making use of the previous 5 days would help.

- Precipitation in Rochester today
    - On previous day, did it rain (more than 0.2) in cities west of Rochester (1 dim. each)?
    - On previous day, did it rain in Rochester?
    - On previous day, did it snow (more than 0.2) in cities west of Rochester (1 dim. each)?
    - On previous day, did it snow in Rochester?
    - On previous day, did western cities have lower DEP than Rochester (1 dim. each)?
    - On previous day, did Rochester have lower DEP than today?

# Training Process

Beyond a manual process, we were unable to fully complete this step. Hyperparameters were adjusted and the best performance was chosen across models. Since data used in each significantly overlapped, it seemed like a reasonable assumption that hyperparameters would behave similarly. If we had more time, we would have attempted some sort of regression approach where hyperparamters could be tweaked to minimize loss (wrong predictions).