Haydon Behl

016390257

MATH 178

April 20, 2025

Project 2 - Markov Model Weather Forecasting

1. Introduction & Project Overview

In this project, I used a Markov chain model to predict tomorrow's weather in Seattle based on today's weather and the month. My goal is to build a model that captures both day-to-day transitions and seasonal patterns. I chose this project because weather is a real-world example of a dynamical system that changes over time in ways that are difficult to solve by hand. Using computer simulations in python allows us to estimate the probability of different weather outcomes.

2. Model Overview and Intuition

Initial EDA with the data presented me with what I hoped to see–the numerical data is not sufficiently correlated with the next-day's weather to be best fit to an MLR or logistic model. Still, I tried fitting both and found the best fit with a logistic regression model that earned 65% accuracy. This would be the baseline score I would work to beat with a different model altogether. After some research into weather forecasting, I found that the most human-interpretable and high-scoring model could lay in a Markov chain model.

A Markov chain models how a system transitions between discrete states with probabilities that depend only on the current state. Markov chains are very easy to interpret after construction, since they exist as a probability table of a next state given the current state, as reviewed in class. We first implemented a basic first-order Markov model using only the weather categories (drizzle, rain, sun, snow, fog). After counting

day-to-day transitions over the historical record and normalizing to probabilities, this simple chain achieved around 64.18% validation accuracy. Below is the final transition table after fitting.

```
Transition probability matrix:
           drizzle      rain       sun      snow       fog
drizzle  0.301887  0.358491  0.283019  0.000000  0.056604
rain     0.028081  0.673947  0.224649  0.017161  0.056162
sun      0.025039  0.231612  0.682316  0.007825  0.053208
snow     0.038462  0.384615  0.192308  0.384615  0.000000
fog      0.009901  0.316832  0.396040  0.000000  0.277228
If today is drizzle, tomorrow is most likely: rain
If today is rain   , tomorrow is most likely: rain
If today is sun    , tomorrow is most likely: sun
If today is snow   , tomorrow is most likely: rain
If today is fog    , tomorrow is most likely: sun
```

While this is well above the 20% baseline of random guessing among five equally likely states, it still missed many patterns—especially those driven by seasonal changes. I didn't want to stop here, because my initial findings with the logistic regression model had slightly better results (~65%) that I wanted to beat with the Markov chain model.

My exploratory data analysis (EDA) revealed strong monthly and seasonal effects in Seattle's climate. Monthly averages of precipitation and temperature showed that winter months are much wetter and cooler, whereas summer months are drier and warmer.

Likewise, the frequency of each weather category varied by month—for example, “sun” days dominated the summer, while “rain” days were most common in winter. These clear seasonal trends suggested that conditioning only on today’s weather state ignored a crucial source of information—time of year.

To capture these effects, I extended our state definition to a composite of (weather, month) tuples (e.g. “rain_01” for rain in January). By building transition probabilities between these composite states and the pure weather states for the next day, our model learns distinct patterns for each month. This seasonal conditioning boosted our validation accuracy to 66.92%, demonstrating that incorporating the month significantly improves day-to-day weather predictions. The final table:

```
P(next_weather | today_weather, today_month):
             drizzle       fog      rain      snow       sun
drizzle_01  0.600000  0.000000  0.300000  0.000000  0.100000
drizzle_02  0.000000  0.000000  1.000000  0.000000  0.000000
drizzle_03  0.000000  0.333333  0.333333  0.000000  0.333333
drizzle_04  0.000000  0.000000  0.666667  0.000000  0.333333
drizzle_05  0.000000  0.000000  0.000000  0.000000  1.000000
drizzle_06  0.000000  0.000000  0.500000  0.000000  0.500000
drizzle_07  0.250000  0.125000  0.125000  0.000000  0.500000
drizzle_08  0.250000  0.000000  0.125000  0.000000  0.625000
drizzle_09  0.600000  0.000000  0.400000  0.000000  0.000000
drizzle_10  0.500000  0.000000  0.500000  0.000000  0.000000
drizzle_11  0.000000  0.333333  0.666667  0.000000  0.000000
drizzle_12  0.500000  0.000000  0.000000  0.000000  0.500000
fog_01      0.000000  0.411765  0.411765  0.000000  0.176471
fog_02      0.000000  0.000000  0.666667  0.000000  0.333333
fog_03      0.000000  0.500000  0.500000  0.000000  0.000000
fog_04      0.000000  0.000000  0.666667  0.000000  0.333333
fog_05      0.000000  0.000000  0.400000  0.000000  0.600000
fog_06      0.000000  0.000000  0.000000  0.000000  1.000000
fog_07      0.100000  0.000000  0.100000  0.000000  0.800000
fog_08      0.000000  0.166667  0.000000  0.000000  0.833333
fog_09      0.000000  0.285714  0.214286  0.000000  0.500000
fog_10      0.000000  0.578947  0.210526  0.000000  0.210526
fog_11      0.000000  0.000000  0.666667  0.000000  0.333333
fog_12      0.000000  0.250000  0.250000  0.000000  0.500000
rain_01     0.016393  0.147541  0.754098  0.016393  0.065574
rain_02     0.053333  0.026667  0.773333  0.013333  0.133333
rain_03     0.041096  0.013699  0.712329  0.068493  0.164384
rain_04     0.049180  0.032787  0.540984  0.000000  0.377049
rain_05     0.000000  0.075000  0.600000  0.000000  0.325000
rain_06     0.023810  0.000000  0.523810  0.000000  0.452381
rain_07     0.062500  0.125000  0.500000  0.000000  0.312500
rain_08     0.041667  0.083333  0.416667  0.000000  0.458333
rain_09     0.000000  0.083333  0.500000  0.000000  0.416667
rain_10     0.000000  0.096774  0.758065  0.000000  0.145161
rain_11     0.040000  0.026667  0.786667  0.013333  0.133333
rain_12     0.013158  0.052632  0.723684  0.039474  0.171053
snow_01     0.125000  0.000000  0.125000  0.750000  0.000000
snow_02     0.000000  0.000000  0.250000  0.250000  0.500000
snow_03     0.000000  0.000000  0.500000  0.166667  0.333333
snow_04     0.000000  0.000000  1.000000  0.000000  0.000000
snow_11     0.000000  0.000000  0.000000  0.000000  1.000000
snow_12     0.000000  0.000000  0.666667  0.333333  0.000000
sun_01      0.035714  0.035714  0.285714  0.035714  0.607143
sun_02      0.000000  0.037037  0.296296  0.074074  0.592593
sun_03      0.000000  0.027778  0.388889  0.000000  0.583333
sun_04      0.000000  0.019231  0.423077  0.019231  0.538462
sun_05      0.012821  0.025641  0.192308  0.000000  0.769231
sun_06      0.013333  0.013333  0.240000  0.000000  0.733333
sun_07      0.055556  0.077778  0.055556  0.000000  0.811111
sun_08      0.046512  0.034884  0.162791  0.000000  0.755814
sun_09      0.030769  0.123077  0.200000  0.000000  0.646154
sun_10      0.051282  0.051282  0.282051  0.000000  0.615385
sun_11      0.000000  0.156250  0.250000  0.000000  0.593750
sun_12      0.000000  0.064516  0.387097  0.032258  0.516129

Example:
  Today = rain, month = 01 → tomorrow most likely: rain
```

3. Data and Code

I used daily weather data for Seattle from 2012 to 2018 retrieved from kaggle. The main steps in my code were:

1. Load and preprocess the data (convert dates, extract month).
2. Define weather and composite states.
3. Count transitions between states over consecutive days.
4. Normalize counts to probabilities to build the transition table.
5. Use the table to predict the next weather state.
6. Validate by measuring the percentage of correct predictions over the whole dataset.

Using this pipeline I was able to sufficiently predict the next day's weather in Seattle after some iteration on the model. All of my code is visible on my Github.

4. Results and Simulation

- Baseline logistic regression accuracy: 65.00%
- Basic first-order Markov chain accuracy: 64.18%
- Composite-state Markov chain accuracy: 66.92%

Using my composite Markov model, I simulated predictions for each day and compared them to actual outcomes. The model achieved 66.92% accuracy on historical data. This means it correctly predicted tomorrow's weather about two out of every three days.

5. Context and Interpretation

In real life, weather depends on factors like temperature, humidity, and pressure. Our simplified model uses only past weather and month, but still captures important patterns like rainy winters and sunnier summers. A 66.92% accuracy shows that, even with limited information, the composite Markov chain gives a useful forecast.

6. Conclusion

I built a computational model to predict Seattle weather using a Markov chain with composite states. Code simulations made it possible to handle large data that would be impossible to solve by hand. Our model's accuracy indicates it can be a baseline weather predictor. Future improvements could include additional features or more machine-learning methods, perhaps moving forward with a Hidden Markov Model for improvements in extracting underlying information from the data.

Relevant Sources

Data: https://www.kaggle.com/datasets/ananthr1/weather-prediction

Codebase: https://github.com/Exidekat/math178weather