MARKOV CHAINS MATH MODEL (WEATHER PREDICTION)

By Najia Jahan

Professor Erik K. Grimmelmann, Ph. D.
Final Project
CSC 30100
Spring 2023

Table of Contents

Abstract:

This report explores the use of Markov chains in building a weather model. Markov chains are a type of stochastic process that can be used to model the transition of weather conditions from one state to another. The report provides an overview of the theory behind Markov chains, including the concepts of state space, transition probabilities, and stationary distributions. The report then presents a detailed methodology for building a weather model using Markov chains, including data collection, model design, and simulation. The report concludes with a discussion of the results obtained from the weather model, including the accuracy of the predictions and the limitations of the model. The report highlights the potential of Markov chains as a powerful tool for modeling complex systems and their applications in weather forecasting.

Introduction:

A Markov chain is a mathematical system that experiences transitions from one state to another according to certain probabilistic rules. Markov chains are basically represented as a collection of a finite or a countable set of states and a set of transitions between those states. The defining characteristic of a Markov chain is that the probability of transitioning to any particular state is dependent solely on the current state and time elapsed.

From the probability distribution below we can understand the definition more clearly:

$$P(X_{n+1} = x \mid X_1 = x_1, X_2 = x_2, X_3 = x_{3,\ldots\ldots} X_n = x_n)$$

$$P(X_{n+1} = x \mid X_{n+1} = x_n)$$

The probability that (n+1)th step will be x depends only on the nth step, not the complete sequence of steps that came before n.

The state space, or set of all possible states, can be anything: letters, numbers, words, weather conditions, stock performances, search results, or game scores. Markov chains are widely utilized in economics, game theory, queueing (communication) theory, genetics, and finance. While it is possible to discuss Markov chains with any size of state space, the initial theory and most applications are focused on cases with a finite (or countably infinite) number of states.

History:

Markov chains were developed by the Russian mathematician Andrey Markov, specifically by Andrey Andreyevich Markov. He is known for his work on stochastic processes and probability theory, including the study of sequences of events with probabilistic properties. Markov's groundbreaking research on Markov chains, published in the late 19th and early 20th centuries, laid the foundation for the field of Markov processes and had significant implications for various disciplines, including mathematics, statistics, physics, and computer science. His contributions to the theory of Markov chains have had a huge impact and continues to influence in diverse areas of study.

Markov Property:

Markov chains are stochastic process. The mathematical definition of stochastic process is that it is a sequence of random events or phenomena that evolve over time, where the behavior of the system is influenced by random factors at each time step. However, the Markov chains are different from the general stochastic process as it needs to be memoryless which is known as Markov property.

Suppose we have a sequence of random variables $\{Xn\}$. This sequence will be called a Markov chain if it has the Markov property:

$$P\{X_k = i \mid X_{k-1} = j, X_{k-2}, \cdots, X_1\} = P\{X_k = i \mid X_{k-1} = j\}$$

This equation states that the probability of being in state i at time k, given that we were in state j at time k-1, is equal to the probability of being in state i at time k, given that we were in state j at any time in the past. This means that the Markov chain is memoryless. It does not remember how it got to its current state. All it knows is its current state, and it uses that information to predict its future states.

States are usually labelled {(0,)1, 2, ⋯}, and state space can be finite or infinite.

- First order Markov assumption (memoryless):

$$P(q_t = i \mid q_{t-1} = j, q_{t-2} = k, \cdots) = P(q_t = i \mid q_{t-1} = j)$$

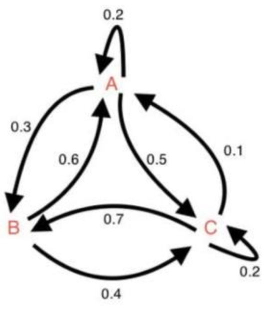
This means that the probability of being in state i at time t, given that we were in state j at time t−1, is equal to the probability of being in state i at time t, given that we were in state j at any time in the past.

- Homogeneity: $P(q_t = i \mid q_{t-1} = j) = P(q_{t+l} = i \mid q_{t+l-1} = j)$

This Homogenity equation states that the probability of being in state i at time t, given that we were in state j at time t−1, is equal to the probability of being in state i at time t+l, given that we were in state j at time t+l−1. This means that the Markov chain is time-invariant. The transition probabilities are the same regardless of when the chain is started.

Transition Matrix:

Transition matrix of Markov Chains represents the probability distribution of the state's transitions. If the system has N states then transition matrix will be a N×N matrix. Given that it's a stochastic matrix, the sum of probabilities in each row is one. A directed, connected graph can be converted into a transition matrix, with each element representing a probability weight associated to an edge connecting two nodes.



For the Markov chains example above the Transition Matrix will look like this:

	A	B	C
A	0.2	0.3	0.5
B	0.6	0	0.2
C	0.1	0.7	0.2

From the transition matrix, we can see each row's probability distributions add up to 1.

The transition matrix is denoted by $P = (p_{ij})$.

$$p_{ij} = P(X_{t+1} = j \mid X_t = i)$$

In this equation X_t is considered as current state, and a row represent currently happening (transition from X_t). The column represents the next or transition to X_{t+1}. Entry (i, j) depends on the probability that Next = j, while now = i, which signifies the probability of going from state i to state j.

Stationary Distribution of Markov Chains:

A Markov chain with a finite state has a stationary probability distribution. This is a probability distribution that remains unchanged as time progresses in the Markov chain. Stationary distribution is typically denoted as a row vector π whose entries are probabilities that sums to 1, and gives a transition matrix P.

it satisfies the equation, $\pi = \pi P$

Lets consider the following transition matrix,

$$P = [\quad [0.8, 0.2], \\ [0.5, 0.5]]$$

P can be written as:

$$\pi (P - I) = 0, \text{ where I is the identity matrix}$$

$$\pi(P - I) = [\pi_1, \pi_2] * [[0.8-1, 0.2], [0.5, 0.5-1]] = [0, 0]$$

This gives us two equations:

$$0.8\pi_1 + 0.5\pi_2 - \pi_1 = 0$$

$$0.2\pi_1 + 0.5\pi_2 - \pi_2 = 0$$

We also know that the probabilities must sum to 1: $\pi_1 + \pi_2 = 1$

Solving this system of equations, we get:

$$\pi_1 = 0.625$$

$$\pi_2 = 0.375$$

So, the stationary distribution of this Markov chain is [0.625, 0.375]. This means that over time, if the system starts in state 1, then it will remain in that state about 62.5% of the time, and will be in state 2 about 37.5% of the time. On the other hand, if the system starts in state 2, then it will remain in that state about 37.5% of the time, and will be in state 1 about 62.5% of the time.

Knowing the stationary distribution of a Markov chain is important because it provides information about the long-term behavior of the system. It can be used to predict future probabilities or optimize the system by finding ways to adjust transition probabilities to achieve a desired stationary distribution.

How Does Markov Chains Work:

Here are the general steps for creating and working with Markov chains:

- **Define the states/events:** Identify all the possible states that the system can be in. For example, suppose the weather has three conditions, "sunny," "cloudy," and "rainy". So, to make a markov chains weather model we will gave 3 states.
- **Create the transition matrix:** The transition matrix for N states is a NxN square matrix that represents the probability of transitioning from one state to another in a single time step. Each element in the matrix represents the probability of going from one state to another. The sum of each row of the matrix should equal 1.
- **Determine the initial state probabilities:** These are the probabilities that the system starts in a particular state. For the weather model, if it is currently sunny, the initial state probabilities might be [1 0 0], meaning that the probability of starting in the sunny state is 1 and the probabilities of starting in the other states are 0.
- **Simulate the Markov chain:** To simulate the Markov chain, we start from a specific state and use the transition matrix to determine the probabilities of moving to each of the other states in the next time step. Then we randomly choose the next state based on these probabilities and repeat the process for as many time steps as desired.
- **Analyze the Markov chain:** After simulating the Markov chain getting transition matrix and probabilities, we can analyze it to answer questions such as "What is the probability

of being in a particular state after a certain number of time steps?" or "What is the expected number of time steps until the system reaches a certain state?"

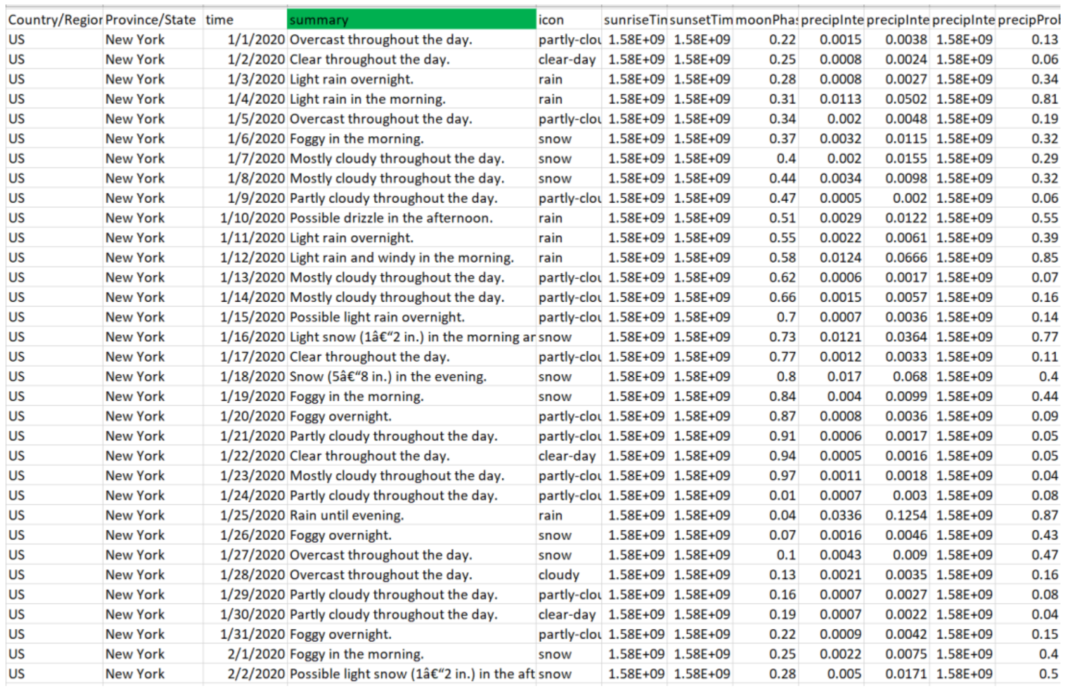
Real-life Usage of Markov Chains:

- **Finance and Stock Market Analysis:** Markov chains are very useful in finance and marketing. They can be used to model stock prices and predict future trends in the stock market. The model can be trained on historical data to estimate the probabilities of stock prices moving up, down, or staying the same based on the current price and other relevant factors.
- **Text Generation Tool and Analysis:** Markov chains can be used to generate text that closely resembles the style of a given author or genre. The model can be trained on a large corpus of text to learn the patterns of word usage and generate new text based on these patterns. In this context, the states will be different chunks of words and transition probabilities would be the likelihood of going to one sequence of words to other.
- **Speech Recognition:** Hidden Markov models (HMMs) are commonly used in speech recognition to model the probability of a particular sound given the previous sounds in a sequence. HMMs can be trained on large datasets of speech recordings to improve the accuracy of speech recognition systems.
- **Genetics:** Markov chains can be used to model DNA sequences and identify patterns in the sequence. The model can be used to predict the likelihood of a particular genetic mutation over the generations occurring based on the surrounding DNA sequence. They are also used to analyze the evolutionary history of species by modeling the mutations that occur in DNA sequences over time.

- **Social Networks and Advertising:** Markov chains can be used to model the dynamics of social networks, such as the likelihood of a user following another user or the probability of a user posting a particular type of content given their previous posts. This information can be used to improve recommendation systems and targeted advertising.

Markov Chains Weather Model:

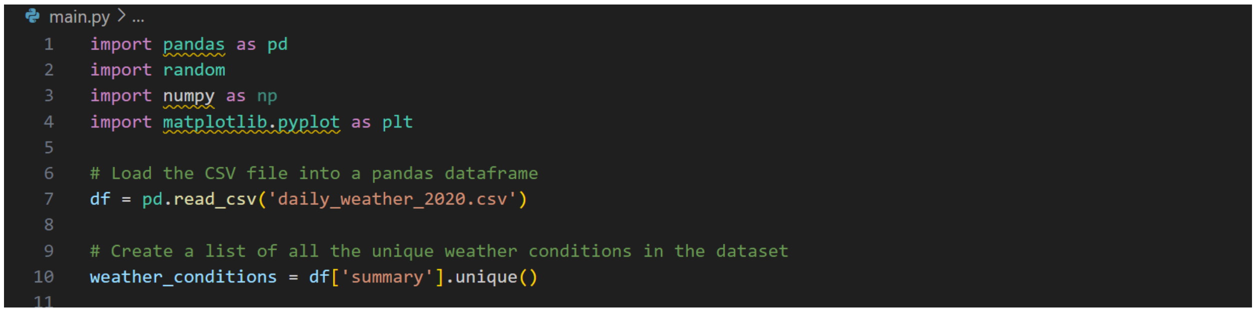
For this project we will be utilizing Markov Chains properties to build a weather prediction model. For building this model, we first need a dataset of weather conditions for a given time period. Extracting that data, we will create a transition matrix that shows the probabilities of transitioning from one weather condition to another. Then that transition matrix will be utilized to generate a sequence of weather conditions for a given time period for the future. This dataset should include the date and a categorical variable representing the weather condition (e.g., sunny, rainy, cloudy, etc.). For the data, we will use a .csv file that has 30687 data. It contains historical daily weather data for 163 countries(with provincial data for some) from Jan 1, 2020 up to April 21, 2020. The snippet of the dataset below is displaying the weather data for New York for the month of January, 2020.



Country/Regior	Province/State	time	summary	icon	sunriseTin	sunsetTim	moonPha:	precipInte	precipInte	precipInte	precipProl
US	New York	1/1/2020	Overcast throughout the day.	partly-clou	1.58E+09	1.58E+09	0.22	0.0015	0.0038	1.58E+09	0.13
US	New York	1/2/2020	Clear throughout the day.	clear-day	1.58E+09	1.58E+09	0.25	0.0008	0.0024	1.58E+09	0.06
US	New York	1/3/2020	Light rain overnight.	rain	1.58E+09	1.58E+09	0.28	0.0008	0.0027	1.58E+09	0.34
US	New York	1/4/2020	Light rain in the morning.	rain	1.58E+09	1.58E+09	0.31	0.0113	0.0502	1.58E+09	0.81
US	New York	1/5/2020	Overcast throughout the day.	partly-clou	1.58E+09	1.58E+09	0.34	0.002	0.0048	1.58E+09	0.19
US	New York	1/6/2020	Foggy in the morning.	snow	1.58E+09	1.58E+09	0.37	0.0032	0.0115	1.58E+09	0.32
US	New York	1/7/2020	Mostly cloudy throughout the day.	snow	1.58E+09	1.58E+09	0.4	0.002	0.0155	1.58E+09	0.29
US	New York	1/8/2020	Mostly cloudy throughout the day.	snow	1.58E+09	1.58E+09	0.44	0.0034	0.0098	1.58E+09	0.32
US	New York	1/9/2020	Partly cloudy throughout the day.	partly-clou	1.58E+09	1.58E+09	0.47	0.0005	0.002	1.58E+09	0.06
US	New York	1/10/2020	Possible drizzle in the afternoon.	rain	1.58E+09	1.58E+09	0.51	0.0029	0.0122	1.58E+09	0.55
US	New York	1/11/2020	Light rain overnight.	rain	1.58E+09	1.58E+09	0.55	0.0022	0.0061	1.58E+09	0.39
US	New York	1/12/2020	Light rain and windy in the morning.	rain	1.58E+09	1.58E+09	0.58	0.0124	0.0666	1.58E+09	0.85
US	New York	1/13/2020	Mostly cloudy throughout the day.	partly-clou	1.58E+09	1.58E+09	0.62	0.0006	0.0017	1.58E+09	0.07
US	New York	1/14/2020	Mostly cloudy throughout the day.	partly-clou	1.58E+09	1.58E+09	0.66	0.0015	0.0057	1.58E+09	0.16
US	New York	1/15/2020	Possible light rain overnight.	partly-clou	1.58E+09	1.58E+09	0.7	0.0007	0.0036	1.58E+09	0.14
US	New York	1/16/2020	Light snow (1â€“2 in.) in the morning ar	snow	1.58E+09	1.58E+09	0.73	0.0121	0.0364	1.58E+09	0.77
US	New York	1/17/2020	Clear throughout the day.	partly-clou	1.58E+09	1.58E+09	0.77	0.0012	0.0033	1.58E+09	0.11
US	New York	1/18/2020	Snow (5â€“8 in.) in the evening.	snow	1.58E+09	1.58E+09	0.8	0.017	0.068	1.58E+09	0.4
US	New York	1/19/2020	Foggy in the morning.	snow	1.58E+09	1.58E+09	0.84	0.004	0.0099	1.58E+09	0.44
US	New York	1/20/2020	Foggy overnight.	partly-clou	1.58E+09	1.58E+09	0.87	0.0008	0.0036	1.58E+09	0.09
US	New York	1/21/2020	Partly cloudy throughout the day.	partly-clou	1.58E+09	1.58E+09	0.91	0.0006	0.0017	1.58E+09	0.05
US	New York	1/22/2020	Clear throughout the day.	clear-day	1.58E+09	1.58E+09	0.94	0.0005	0.0016	1.58E+09	0.05
US	New York	1/23/2020	Mostly cloudy throughout the day.	partly-clou	1.58E+09	1.58E+09	0.97	0.0011	0.0018	1.58E+09	0.04
US	New York	1/24/2020	Partly cloudy throughout the day.	partly-clou	1.58E+09	1.58E+09	0.01	0.0007	0.003	1.58E+09	0.08
US	New York	1/25/2020	Rain until evening.	rain	1.58E+09	1.58E+09	0.04	0.0336	0.1254	1.58E+09	0.87
US	New York	1/26/2020	Foggy overnight.	snow	1.58E+09	1.58E+09	0.07	0.0016	0.0046	1.58E+09	0.43
US	New York	1/27/2020	Overcast throughout the day.	snow	1.58E+09	1.58E+09	0.1	0.0043	0.009	1.58E+09	0.47
US	New York	1/28/2020	Overcast throughout the day.	cloudy	1.58E+09	1.58E+09	0.13	0.0021	0.0035	1.58E+09	0.16
US	New York	1/29/2020	Partly cloudy throughout the day.	partly-clou	1.58E+09	1.58E+09	0.16	0.0007	0.0027	1.58E+09	0.08
US	New York	1/30/2020	Partly cloudy throughout the day.	clear-day	1.58E+09	1.58E+09	0.19	0.0007	0.0022	1.58E+09	0.04
US	New York	1/31/2020	Foggy overnight.	partly-clou	1.58E+09	1.58E+09	0.22	0.0009	0.0042	1.58E+09	0.15
US	New York	2/1/2020	Foggy in the morning.	snow	1.58E+09	1.58E+09	0.25	0.0022	0.0075	1.58E+09	0.4
US	New York	2/2/2020	Possible light snow (1â€“2 in.) in the aft	snow	1.58E+09	1.58E+09	0.28	0.005	0.0171	1.58E+09	0.5

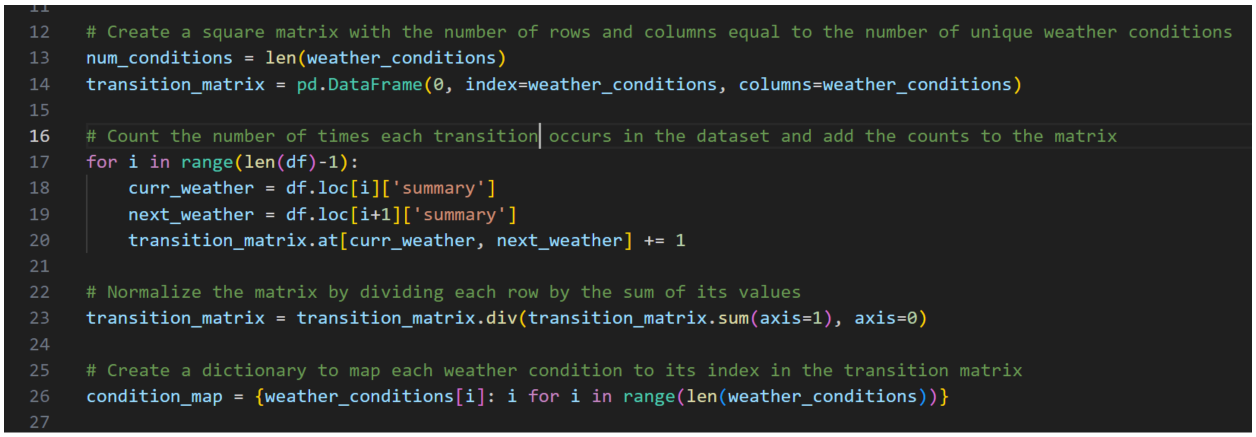
Here, the summary contains the weather conditions for each day. Therefore, they are our states for the model. There are a total of 12 unique weather conditions and so, we will have a transition matrix of 12x12 size. This means there can be 144 possible transitions.

Python Code and Analysis:



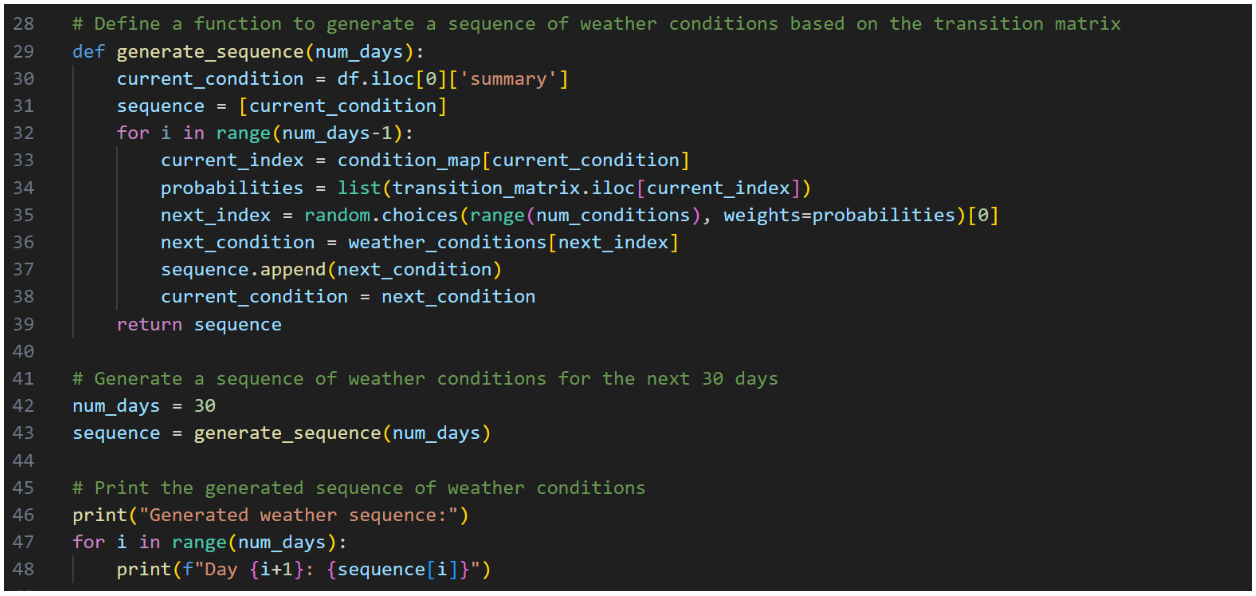
```python
import pandas as pd
import random
import numpy as np
import matplotlib.pyplot as plt

# Load the CSV file into a pandas dataframe
df = pd.read_csv('daily_weather_2020.csv')

# Create a list of all the unique weather conditions in the dataset
weather_conditions = df['summary'].unique()

```

Adding necessary libraries to perform required calculations and simulation.



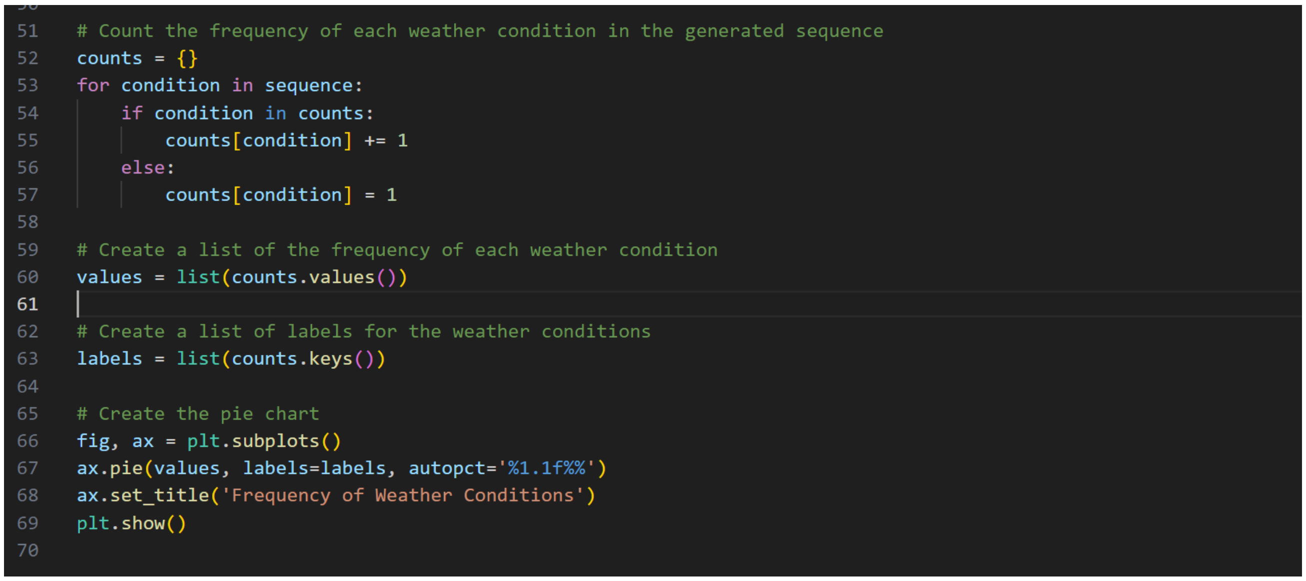
```python
# Create a square matrix with the number of rows and columns equal to the number of unique weather conditions
num_conditions = len(weather_conditions)
transition_matrix = pd.DataFrame(0, index=weather_conditions, columns=weather_conditions)

# Count the number of times each transition occurs in the dataset and add the counts to the matrix
for i in range(len(df)-1):
    curr_weather = df.loc[i]['summary']
    next_weather = df.loc[i+1]['summary']
    transition_matrix.at[curr_weather, next_weather] += 1

# Normalize the matrix by dividing each row by the sum of its values
transition_matrix = transition_matrix.div(transition_matrix.sum(axis=1), axis=0)

# Create a dictionary to map each weather condition to its index in the transition matrix
condition_map = {weather_conditions[i]: i for i in range(len(weather_conditions))}
```

It then creates a list of all the unique weather conditions in the dataset and a square matrix with the number of rows and columns equal to the number of unique weather conditions. The for loop iterates over the rows of the dataset and counts the number of times each transition occurs in the dataset. It then adds the counts to the matrix. The matrix is then normalized by dividing each row by the sum of its values. The dictionary is created to map each weather condition to its index in the transition matrix. This will be used later to generate the weather sequence.



```python
# Define a function to generate a sequence of weather conditions based on the transition matrix
def generate_sequence(num_days):
    current_condition = df.iloc[0]['summary']
    sequence = [current_condition]
    for i in range(num_days-1):
        current_index = condition_map[current_condition]
        probabilities = list(transition_matrix.iloc[current_index])
        next_index = random.choices(range(num_conditions), weights=probabilities)[0]
        next_condition = weather_conditions[next_index]
        sequence.append(next_condition)
        current_condition = next_condition
    return sequence

# Generate a sequence of weather conditions for the next 30 days
num_days = 30
sequence = generate_sequence(num_days)

# Print the generated sequence of weather conditions
print("Generated weather sequence:")
for i in range(num_days):
    print(f"Day {i+1}: {sequence[i]}")
```

This section of code defines a function generate_sequence that takes a number of days as input and returns a sequence of weather conditions for those days. The function initializes the current weather condition as the first weather condition in the dataset, and then uses the transition matrix to probabilistically generate the next weather condition based on the current condition. This process is repeated for the desired number of days, and the resulting sequence of weather conditions is returned. Finally, the function is called with a num_days parameter of 30, and the resulting sequence is printed to the console for each day.

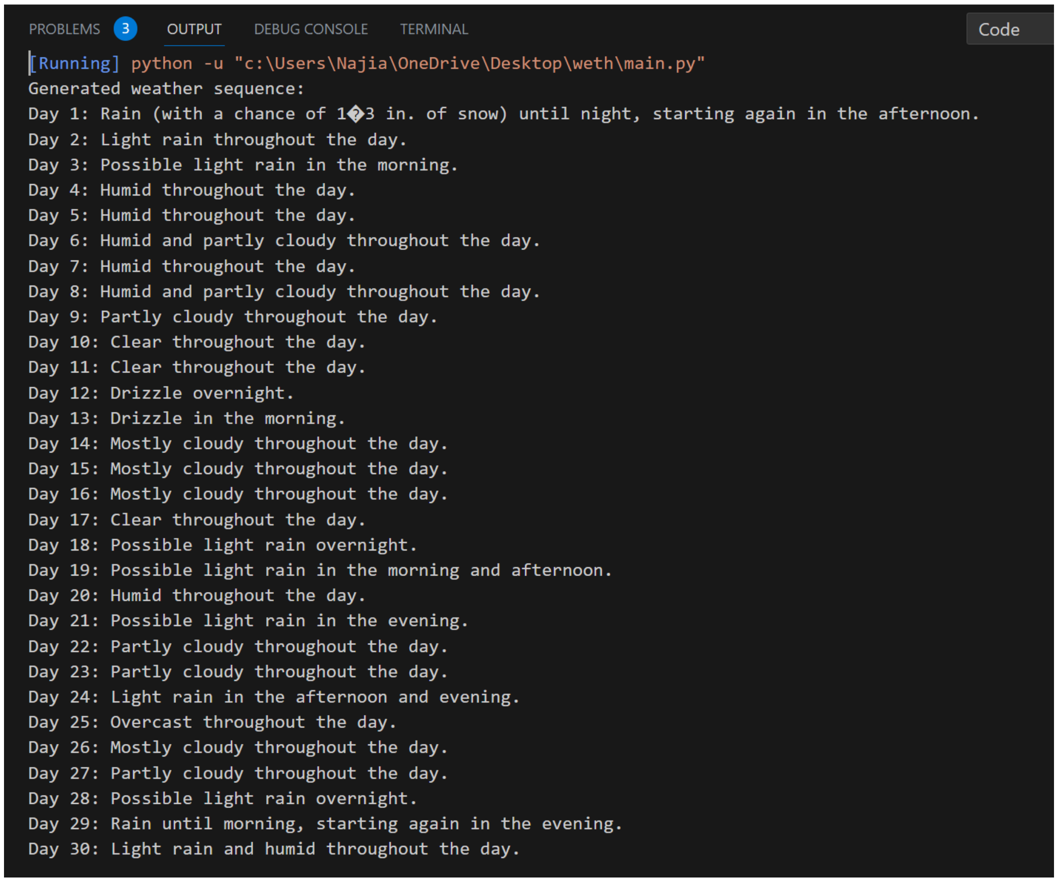


```python
# Count the frequency of each weather condition in the generated sequence
counts = {}
for condition in sequence:
    if condition in counts:
        counts[condition] += 1
    else:
        counts[condition] = 1

# Create a list of the frequency of each weather condition
values = list(counts.values())

# Create a list of labels for the weather conditions
labels = list(counts.keys())

# Create the pie chart
fig, ax = plt.subplots()
ax.pie(values, labels=labels, autopct='%1.1f%%')
ax.set_title('Frequency of Weather Conditions')
plt.show()
```

The code above generates a pie chart that shows the frequency of each weather condition in the generated sequence of weather conditions.

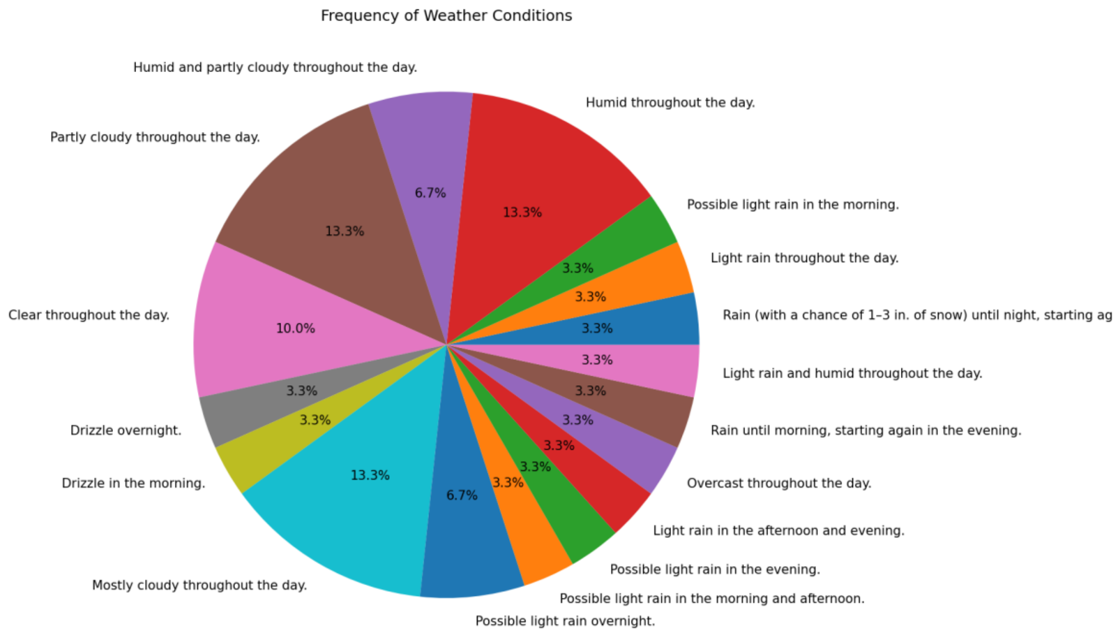
Output of the Code Simulation:

Terminal output:



```
[Running] python -u "c:\Users\Najia\OneDrive\Desktop\weth\main.py"
Generated weather sequence:
Day 1: Rain (with a chance of 1�3 in. of snow) until night, starting again in the afternoon.
Day 2: Light rain throughout the day.
Day 3: Possible light rain in the morning.
Day 4: Humid throughout the day.
Day 5: Humid throughout the day.
Day 6: Humid and partly cloudy throughout the day.
Day 7: Humid throughout the day.
Day 8: Humid and partly cloudy throughout the day.
Day 9: Partly cloudy throughout the day.
Day 10: Clear throughout the day.
Day 11: Clear throughout the day.
Day 12: Drizzle overnight.
Day 13: Drizzle in the morning.
Day 14: Mostly cloudy throughout the day.
Day 15: Mostly cloudy throughout the day.
Day 16: Mostly cloudy throughout the day.
Day 17: Clear throughout the day.
Day 18: Possible light rain overnight.
Day 19: Possible light rain in the morning and afternoon.
Day 20: Humid throughout the day.
Day 21: Possible light rain in the evening.
Day 22: Partly cloudy throughout the day.
Day 23: Partly cloudy throughout the day.
Day 24: Light rain in the afternoon and evening.
Day 25: Overcast throughout the day.
Day 26: Mostly cloudy throughout the day.
Day 27: Partly cloudy throughout the day.
Day 28: Possible light rain overnight.
Day 29: Rain until morning, starting again in the evening.
Day 30: Light rain and humid throughout the day.
```

Pie-chart of the probabilities:



Comparison with Other Estimation Models:

There are some other models which are used to build similar kind of weather prediction model. For example, Numerical weather prediction (NWP) which utilizes mathematical models that simulate the behavior of the atmosphere and oceans to forecast future weather conditions. The Markov chains weather model has the following advantages compared to numerical weather prediction models or machine learning-based models:

Advantages:

- Simplicity and interpretability.
- Works well with limited data.
- Computational efficiency.
- Provides a clear representation of transition probabilities.

However, the Markov chains weather model has the following limitations:

Disadvantages:

- Assumes the Markov property, which may not hold for all weather phenomena.
- Limited spatial and temporal resolution.
- Lacks explicit incorporation of physical dynamics.
- May struggle to accurately predict extreme or rare weather events.

In contrast, numerical weather prediction models offer detailed and accurate forecasts by simulating atmospheric dynamics, while machine learning-based models leverage large datasets and algorithms to capture complex relationships.

Conclusion:

This report effectively explains the Markov chains weather model, covering key concepts such as state space, transition matrix, and stationary distribution. It provides practical implementation details and includes a demonstration using historical weather data. The report also highlights real-life applications of Markov chains in diverse fields. Overall, it offers a clear and comprehensive understanding of the model's theory, implementation, and potential applications.

Reference:

http://scihi.org/andrey-markov/

https://brilliant.org/wiki/markov-chains/

https://builtin.com/machine-learning/markov-chain

https://rams.atmos.colostate.edu/at540/fall03/fall03Pt7.pdf

https://www.stat.auckland.ac.nz/~fewster/325/notes/ch8.pdf

https://web2.qatar.cmu.edu/~gdicaro/15281/additional/Ross-ch4.pdf

https://analyticsindiamag.com/5-real-world-use-cases-of-the-markov-chains/

https://www.kaggle.com/datasets/vishalvjoseph/weather-dataset-for-covid19-predictions