Markov Chains

PYTHON PROGRAMMING

A DATASCIENCE AND MACHINE LEARNING COURSE

SUMMARY

- Probabilities
- Expectations
- What is weather tomorrow?
- Independence
- Dependence
- Assumptions
- Transition
- Problem
- Solution
- Monte Carlo
- Convergence
- Exercise

Probabilities

- Two designs
- Marketing survey
  - Which design to push to market?

Design	Retail Price (GBP)	Margin (GBP)	Probability of buying
A	24.99	17.45	0.42
B	39.99	24.23	0.35

3

A company introduces two types of product designs, and marketing analysts have produced estimates of the likelihood of purchase based on focus group surveys.

You want to push only one of the designs to the market.

Which one should you choose?

4

Expectations

- Survey of 1000
  - Interest/person = 1 for yes, 0 for no
  - Design A - 420 interested
    - ((420 x 1) + (580 x 0))/ 1000 x 17.45 = 7.24 GBP
  - Design B - 350 interested
    - ((350 x 1) + (650 x 0))/ 1000 x 24.23 = 8.48 GBP

You want to maximize your profit to infer which product is best you survey 1000 customers.

In statistics, measurements ("surveys") are observations of random variables.

Here we are measuring the InterestLevel variable. This could be recorded as 0 for no interest and 1 for interest. We have 420 * 1 interested and (1000 â€" 420) not interested.

The expectation value of a random variable may be interpreted as average of many measurements.

We do not care about calculating an expectation of interest level, but of profit.

Expectation of Profit = Margin * Expectation of InterestLevel

Design A: Predict customer buys according to expectation of InterestLevel, so making profit (420/1000) * 17.45 = 72.4 GBP and likewise 8.48 GBP for B.

So B wins!

5

Probabilities and Expectations

- Given the sale of both
  - How much would be made on average?
  - 1000 products
    - 500 Design A: 500 x 0.42 = 210 sold
      - profit: 210 x 17.45 = 3664.5 GBP
    - 500 Design B: 500 x 0.35 = 175 sold
      - profit: 175 x 24.23 = 4240.25 GBP
- Overall - per attempt
  - (3664.5 + 4240.25)/1000 = 7.90 GBP

Flip a coin to partition 1000 product into A and B products without bias, thus into groups of c. 500. Selling each product we expect:

for design A: 500 * 0.42 = 210 sold, with profit of 210 * 17.45 = 3664.5 GBP

for design B: 500 * 0.35 = 175 sold, with profit of: 175 * 24.23 = 4240.25 GBP

Overall, the total profit we made after 1000 attempts is 3664.5 + 4240.25 = 7904.75 GBP.

So, the average per attempt -- our expectation value -- is 7.90 GBP.

Therefore, in statistics, if x_i is the value we measure for an observable i (in this example the profit margins), and if p_i is the probability for observable i to occur (in this example making a sale), then the expectation value is defined as:

Expectation of Variable = Sum over all observations * probability of observing

What is weather tomorrow?

Can we predict weather?

7

Independence

- Simplify
  - If it does not rain then it will be sunny
- Suppose
  - The weather today has no affect on the weather tomorrow
- Probabilities
  - Sun: 60%
  - Rain: 40%

For simplicity, let's assume there are only two possible weathers:

Raining

Not raining -- and we will simply call it "Sunny"

If the weather today has no bearing on what would be the weather tomorrow, then we say the weathers today and tomorrow are "independent".

We need some numbers, so let's assume that on the average, on any given day in a climate zone:

60% chance of sun

40% chance of rain

8

The Independent Future

- Given it is sunny today
  - what are the chances of rain tomorrow?
    - 60% sun
    - 40% rain
  - What about the day after tomorrow?
  - 60% sun
    - 40% rain

So one can ask: Given it is sunny today, what would be the chance of rain tomorrow?

As the two days are independent, what happens today has no effect on tomorrow. So tomorrow there will be 60% chance it will be sunny and 40% change it will rain -- just like any other day.

How about the day after tomorrow? Just like any other day!

9

The Far Future

- Probability of rain tomorrow and the day after?
  - Probability of rain tomorrow
    - X
  - Probability of rain the day after
    - 0.4 * 0.4 = 0.16

What is probability of rain both tomorrow and the day after?

There is 40% chance it will rain tomorrow, and 40% chance it will rain on the day after...

For a 100 days:

40 out of 100 times, we will get rain tomorrow

out of the 40 times, 40 * 0.40 = 16 times we will get rain the day after

Hence, we will get 16 times when both tomorrow and the day after rains: the probability is 16 / 100 = 0.16.

10

Dependence

- What if the weather tomorrow depends on the weather today?
  - If it is sunny today
    - 60% sun tomorrow
    - 40% rain tomorrow
  - If it is raining today
    - 40% sun tomorrow
    - 60% rain tomorrow

Let us examine a more complex situation.

What if the weather today does have an effect on the weather tomorrow?

If it rains today then it is more likely to rain tomorrow, and likewise, if it is sunny, then it is more likely to be sunny the next day.

What will happen then?

11

The Dependent Future

- Given sun today
  - Tomorrow there is 40% chance of rain
- If tomorrow rains
  - The day after tomorrow there is 60% chance of rain
- Probability of raining in the next two days
  - Probability it rains tomorrow
    - X
- Probability it rains the day after given it rains tomorrow
    - 0.4 * 0.6 = 0.24

This time, what happens tomorrow, or the day after tomorrow can be affected by what happens today.

If today is sunny, then there is 40% chance tomorrow will rain. If tomorrow rains, then there is 60% chance the day after will rain too.

The probability for rain in the next two days will be: 0.4 * 0.6 = 0.24.

If today is raining, then there is 60% chance tomorrow will rain. Therefore, the probability for rain in the next two days will be: 0.6 * 0.6 = 0.36.

12

Markov Chains

- A sequence of observations
  - rain, sun, sun, sun, rain
  - next depends on the previous
- Probability Notation
  - P(S|S): P of sun tomorrow given sun today
  - P(R|S): P of rain tomorrow given sun today
  - P(S|R): P of sun tomorrow given rain today
  - P(R|R): P of rain tomorrow given rain today
  - P(R|S,R): P of ain in two days, given sun today and rain tomorrow
  - P(R|R,R): P of of rain in two days, given rain today and tomorrow

The previous example where the likelihood of one event occurring is based on what has happened before is described as a "Markov process", named after the Russian mathematician Andrey Markov. Markov published his first paper on the process in 1906.

This sequence of events â€" e.g., the weather events of subsequent days -- is often referred to as a "Markov chain".

Markov chain has been extremely useful in describing any sequence of events which are not independent to each other, but depend on history. It is a crucial tool in time-series analysis, and is widely used in both the financial sector (to predict stocks etc) and sciences (movement of atoms, predator-prey populations etc.)

To describe the Markov chain, it is useful to use some mathematical notations:

â€¢ P(S|S): Probability it is going to be sunny tomorrow given it is sunny today
â€¢ P(R|S): Probability it is going to rain tomorrow given it is sunny today â€¢ P(S|R): Probability it is going to be sunny tomorrow given it is raining today
â€¢ P(R|R): Probability it is going to rain tomorrow given it is raining today

And for more complicated cases: â€¢ P(R|S,R): Probability it is going to rain the day after, given it is sunny today and rain tomorrow â€¢ P(R|R,R): Probability it is going to rain the day after, given it is rain today and rain tomorrow

13

Markov Assumptions

- The first order Markov Chain assumes
  - Tomorrow only depends on today
    - not days further in the past
    - not days in the future

The first order Markov assumption is given as:

"The next event is only affected directly by the event immediately previous to it."

In other words, today's weather is only affected by the weather yesterday. It is not affected directly by days further into the past, nor by days in the future.

Of course, it does not mean history does not matter. What happened the day before yesterday will have an affect on yesterday, and therefore will have an indirect influence on what happens today. For this reason, the first order assumption has been used to simulate some very complicated events like fluctuations in the stock markets!

14

Example

- Suppose
    - today there is 80/20 chance it is going to rain
- What is the chance of raining tomorrow?

Let's look at our weather example again.

Transition Matrix I

- Formula for the transition matrix

$$P(\text{sun}_{tw}) = P(S|S) \times P(\text{sun}_{td}) + P(S|R) \times P(\text{rain}_{td})$$

$$P(\text{rain}_{tw}) = P(R|S) \times P(\text{sun}_{td}) + P(R|R) \times P(\text{rain}_{td})$$

15

We will start took at another key concept in a Markov process -- the transition matrix.

A transition matrix provides the way of calculating the probabilities of events in the next day based on the probabilities of the previous day.

To see how this works lets calculate the probability it is going to rain or be sunny tomorrow.

Just like the calculation of expectation values, the probabilities are sum over all probable paths.

Transition Matrix II

- As a matrix equation

$$\begin{pmatrix} P(\text{sun}) \\ P(\text{rain}) \end{pmatrix} = \begin{pmatrix} P(S|S) & P(S|R) \\ P(R|S) & P(R|R) \end{pmatrix} \times \begin{pmatrix} P(\text{sun}) \\ P(\text{rain}) \end{pmatrix}$$

tomorrow | transition matrix | today

16

For those who did linear algebra, we can see that the pair of equations in the last slide can be rewritten into a matrix equation.

For those who did not do linear algebra, or has forgotten it, don't worry. A matrix is just a collection of numbers written on a rectangular grid. A vector is a matrix with only one column or row. This transformation is just a definition of matrix multiplication.

The matrix formed as a result in the middle, containing the probabilities P(S|S), P(R|S) etc. is called the "transition matrix".

A transition matrix provides the way of calculating the probabilities of events in the next day based on the probabilities of the previous day.

Transition Matrix II

- Matrix multiplication gives probability

$$\begin{pmatrix} 0.44 \\ 0.56 \end{pmatrix} = \begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix} \times \begin{pmatrix} 0.2 \\ 0.8 \end{pmatrix}$$

tomorrow transition matrix today

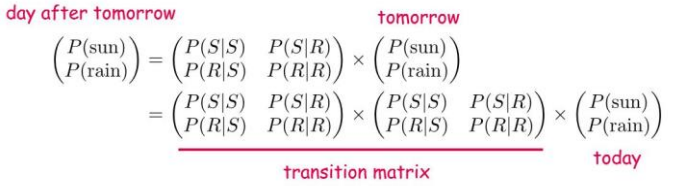
17

Plugging in the numbers, we get that it will be 44% likelihood that it will be sunny tomorrow, and 56% likelihood it will be rain.

Multiplying the weather probabilities of today with the transition matrix allows us to compute the probabilities of tomorrow in one (general) operation.

The Day After Tomorrow? I

- The same formula applies



18

How about the day after tomorrow?

The same formula applies. If we substitute in the expression for tomorrow's probabilities, we notice that:

day after tomorrow = (transition matrix) \^ 2 * today

The Day After Tomorrow? II

- Probabilities for the day after tomorrow

day after tomorrow — tomorrow

$$\begin{pmatrix} 0.488 \\ 0.512 \end{pmatrix} = \begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix} \times \begin{pmatrix} 0.44 \\ 0.56 \end{pmatrix}$$

$$= \underbrace{\begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix} \times \begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix}}_{\text{transition matrix}} \times \underbrace{\begin{pmatrix} 0.2 \\ 0.8 \end{pmatrix}}_{\text{today}}$$

19

Plugging in the numbers, we get the probabilities for rain and sunny weather the day after tomorrow, based on the probabilities of today.

One year later? I

- How about 365 days later?

$$\begin{pmatrix} P(\text{sun}) \\ P(\text{rain}) \end{pmatrix} = \begin{pmatrix} \quad ? \quad \end{pmatrix} \times \begin{pmatrix} P(\text{sun}) \\ P(\text{rain}) \end{pmatrix}$$

365 days later

today

20

How about 365 days later?

One year later? II

- How do you compute M^365?
  - use a computer!

$$\begin{pmatrix} P(\text{sun}) \\ P(\text{rain}) \end{pmatrix} = \begin{pmatrix} P(S|S) & P(S|R) \\ P(R|S) & P(R|R) \end{pmatrix}^{365} \times \begin{pmatrix} P(\text{sun}) \\ P(\text{rain}) \end{pmatrix}$$

365 days later today

$$\begin{pmatrix} ? \\ ? \end{pmatrix} = \begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix}^{365} \times \begin{pmatrix} 0.2 \\ 0.8 \end{pmatrix}$$

21

Applying the same logic, we quickly notice that for any n days after today, the probability of rain or sunny can be calculated using

after n days = (transition matrix) \^ n * today

How do we compute a matrix to the power of 365?

Answer: use a computer.

Use Python!

Calculating

- Python + NumPy
  - Efficient
- Matrix multiplication
- Calculate
  - Transition X state: @
  - Transition ^ 365: for loop

```
import numpy as np

transition = np.array([[0.6, 0.4],
                       [0.4, 0.6]])

today = np.array([0.2, 0.8])

tomorrow = transition @ today

# 365 days later
day = today
for _ in range(365):
  day = transition @ day

print(day)

# OUTPUT:
# array([0.5, 0.5])
```

There is no pre-defined function in Python to perform matrix multiplications. You can certainly write a function yourself.

A much easier, and more efficient way is to use numpy. The numpy package includes many linear algebraic functions, and matrix multiplication is one of them.

The numpy function for matrix multiply is np.matmul(). There is also a short hand operator: \@

To calculate the probabilities for tomorrow, we need to create the vector for today as it is written in the previous slide, and also the transition matrix (as a two dimensional numpy array), and multiply.

Interestingly, notice that after 365 days, there is 50/50 chance it will be raining.

Question

- Today is sunny
  - In the next 365 days
    - how many days do we expect will be rain?

A more complex question:

Given that today is sunny, and if we count the number of days in the next year (365 days), how many days do we expect to have rain?

24

Problem

- Not
  - Will it rain in 365 days time
- Rather
  - If we count number of raining days in a year
  - If we repeat this over many many years
- On average:
  - How many days in a year will it rain?
    - (Ignoring leap years)

We are not asking what happens on the 365th day.

Rather that if we count the number of rain days over the next 365, and if we repeat the experiment over and over, on average, what will we get?

(For simplicity, we will ignore leap years.)

25

Solution

- Hard way
  - Pen and paper: solve a geometric series
- Easy way
  - Use a computer (Python)
  - Work out whether it will rain for every day of the year
  - Many times
  - Compute the averages

There are in general two ways of solving this problem.

Mathematically:

Write out the probabilities of rain for every day in the next 365 days.

Then calculate the expectation value for the total number of raining days by adding the probabilities up. This is equivalent to geometric series of 365 elements.

Computationally: Monte Carlo simulation.

Count the number of rain days by throwing a dice on each day to see if it is going to rain or not. Over and over again

Then the average number of raining days is our answer!

26

Monte Carlo

- History
  - First published by Stanislaw Ulam, 1946
  - Radiation shielding in nuclear weapons programme
- Use
  - Estimate expectation values of random variables
  - Integration
- Methodology
  - Generate an outcome for an event based on given probabilities
  - Repeat many number of times
  - Average = the expectation value

Monte Carlo simulation was first published by Physicist Stanislaw Ulam in 1946 at Los Alamos National Labs.

The goal is to calculate the expected outcomes of a complex process by interpreting the process as a probability and solving with brute force.

We simulate an event based with a series of inter-dependent sub-events, randomly selecting outcomes of sub-events based on their probability.

Repeat the event over and over.

If we do this long enough, then the average value of the final outcomes is the expectation value we are after.

Monte Carlo process relies on computers to be able to repeat short computations typically millions of times. Though idea of Monte Carlo simulation was simple enough to use at any time, it only became successful after 1946 when digital computers became available.

This simulation method is widely used, and is one of the most powerful tools available to a data scientist. This method is used in almost all scientific disciplines for computer simulation, and they are often used together with the Markov chains for time-series analysis.

Monte Carlo simulations can also compute areas under curves (integration) for computing area under a curve. It is particularly efficient method for curves whose form is not known at the time of computation, and/or those with high and narrow peaks.

Expectation values of continues variables are integrations so this is just another form of expectation calculation.

27

One Event

- One event
  - eg. Counting number of raining days in 365 days
    - Every day will be either rain or sunny
    - Every repeat of such event will yield different result

To implement Monte Carlo simulation in Python first implement for one event, and then repeat.

Here "one event" means we go through 365 days, and for each day we determine if it is sunny or rain, and count the raining days. The outcome of one event is the number of raining days in 365 days.

As this is a probabilistic process, the result will be different each time we run the script.

State

- State is a pair
  - tuple, list or np.array of length two
    - probability of sun
    - probability of rain
- Given today is sunny
  - P(sun_today) = (1, 0)

```
import numpy as np

today = np.array([1,0])
```

All the probabilistic information we need about the possible outcomes of an even is its state.

The state of a day is the probability of it being sunny or rain.

We follow what we have done before, we use a 2-element sequence to represent this state: a numpy array.

0th element is probability of sunny

1st element is probability of rain

Transition

- Encode transition matrix
  - list of list or NumPy array

$$\begin{pmatrix} P(S|S) & P(S|R) \\ P(S|S) & P(R|R) \end{pmatrix}$$

```
import numpy as np

today = np.array([1,0])

transition = np.array(
  [[0.6, 0.4],
   [0.4, 0.6]]
)
```

The transition matrix can be represented by a 2D numpy array.

Probability

- NumPy provides matrix multiplication

tomorrow = transition * today

$$\begin{pmatrix} P(\text{sun}) \\ P(\text{rain}) \end{pmatrix} = \begin{pmatrix} P(S|S) & P(S|R) \\ P(R|S) & P(R|R) \end{pmatrix} \times \begin{pmatrix} P(\text{sun}) \\ P(\text{rain}) \end{pmatrix}$$

use NumPy matrix multiplication

```
import numpy as np

today = np.array([1,0])

transition = np.array(
  [[0.6, 0.4],
   [0.4, 0.6]]
)

tomorrow = transition @ today
```

We can use matrix multiplication to calculate the state of the next day given state of the previous day.

Sample Space

- State gives probabilities
  - outcome is binary: sun or rain
- Pick one based on the probabilities
- To pick with probability 0.4?
  - Random float from 0 to 1
  - If less than 0.4, score positive
  - otherwise, score negative

```
import numpy as np

today = np.array([1,0])

transition = np.array(
  [[0.6, 0.4],
   [0.4, 0.6]]
)

tomorrow = transition @ today
p_sun, p_rain = tomorrow

test = np.random.random()
if test < p_sun:              # sun
  tomorrow = np.array([1,0])
else:                         # rain
  tomorrow = np.array([0,1])
```

Now that we have calculated the probabilities of the next day, how do we determine what happens?

This is a random process, the next day will either rain or not-rain based on the calculated probabilities.

There is 0.4 chance that it is going to rain.

Generate a random number from 0 to 1, and if that number is less than 0.4, its a "hit", and therefore will rain. Otherwise, it will be sunny.

To generate random number, we can use the random() function included in numpy.

Sequencing

- Repeat for every day
  - tomorrow becomes today
- Count the number of days it rains
  - Use a for loop
- Output of every run should be different

```
import numpy as np

today = np.array([1,0])

transition = np.array(
  [[0.6, 0.4],
   [0.4, 0.6]]
)

n_rain_days = 0
for _ in range(365):
  today = transition @ today  # tomorrow
  p_sun, p_rain = today
  if np.random.random() < p_sun:
    today = np.array([1,0])
  else:
    today = np.array([0,1])
    n_rain_days += 1
```

Now that we know how to calculate for one day, we can continue for the rest of the 365 days.

Apply same steps for calculating one day, and propagate through. Count the number of raining days.

Evey day is a "today", and "tomorrow" becomes "today" in the next step.

If we run the script at this stage the number of raining days will be different each run.

n Events

- Repeat the one Monte Carlo simulation
  - n times
- Remember the number of raining days each time
  - Calculate average after n events
  - For large n
  - Average aproximates solution

```
import numpy as np

today = np.array([1,0])

transition = np.array(
  [[0.6, 0.4],
   [0.4, 0.6]]
)

n_events = 10
n_rain_days = 0
for _ in range(n_events):
  for _ in range(365):
    today = transition @ today
    p_sun, p_rain = today
    if np.random.random() < p_sun:
      today = np.array([1,0])
    else:
      today = np.array([0,1])
      n_rain_days += 1

print(
 'days of rain:',
  n_rain_days / n_events
)
```

Now you know how to count the number of raining days for one Monte Carlo event, repeat, and do this many number of times!

Don't forget to reset the raining day counter to 0 for each Monte Carlo event.

Accumulate a total for computing the average number of raining days over all those repeated events.

34

Convergence

- As n_events increase
  - The calculated average converges to a stable value
- In theory
  - average tends to the true solution as n_events tends to infinity
- In practice
  - average converges after n_events > 10000

Test the script for different sizes of n_events.

Notice that for small n_events, the outcomes will fluctuate. As n_events gets larger, those fluctuations decrease in size, and eventually the average number of raining days settles to a constant number ~ 182.5.

All Monte Carlo simulation should behave this way. The averages will converge to the true answer for large n, and as n approaches infinity the answer approaches the true answer. In practice, our simulation converges to a stable value after around 10,000 events.

For real life applications obtain satisfactory results after several million or more iterations.

You must check your results are converged, i.e., the variance of results is small for runs for the same N.

Results before convergence are inaccurate.

Exercise

36

Exercise Problem

- Markov Weather
  - Change the n_event numbers
    - Run the script many times, observe variations in the output
    - Does the size of variations change as you increase n_events?
  - Do you get a different outcome if you change the probabilities in the transition matrix?
- Markov Stocks
  - Write a program to predict stock prices using Monte Carlo simulation and Markov Chains
  - Follow the instructions given in the exercise
- Extra
  - Write a script that would automatically check for convergence

Follow the link www.tinyurl.com/wbsadvex

Go to the folder WBS Adv

The code demonstrated is contained in MC Markov Chain Weather Example

Change the n_event numbers, and run the script many times, observe variations in the output.

Does the size of variations change as you increase n_events?

Do you get a different outcome if you change the probabilities in the transition matrix?

Open the exercise: MC Markov Chain Stocks exercise

Write a program to predict stock prices using Monte Carlo simulation and Markov Chain

Follow the instructions given in the exercise

Extra: Can you write a script that would automatically check for convergence?

SUMMARY

- Probabilities
- Expectations
- What is weather tomorrow?
- Independence
- Dependence
- Assumptions
- Transition
- Problem
- Solution
- Monte Carlo
- Convergence
- Exercise

Thank you

MARKOV CHAINS

A DATASCIENCE AND MACHINE LEARNING COURSE