markov

May 10, 2018

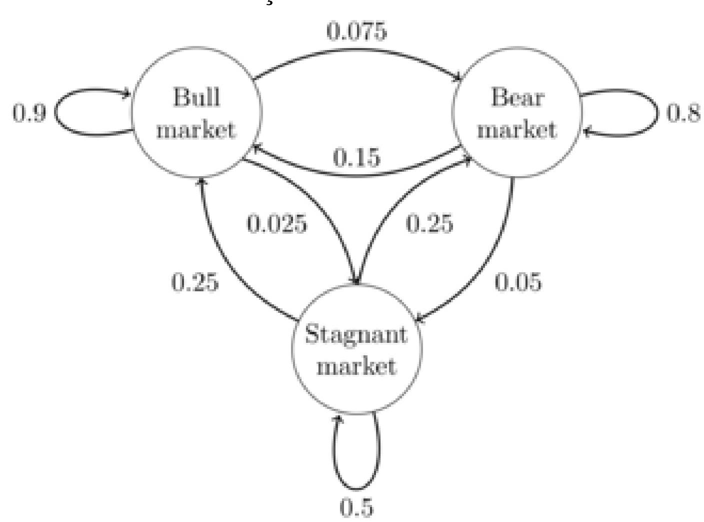
Cadeias de Markov Referência: http://www.math.ou.edu/~epearse/resources/Math3333-LinearAlgebra/Markov-chain-basics.pdf Referência: https://en.wikipedia.org/wiki/Markov_chain

---

Uma cadeia de Markov é um modelo estocástico que descreve uma sequência de possíveis eventos nos quais a probabilidade de cada evento depende apenas do estado, ou evento, imediatamente anterior.

Cadeias de Markov têm muitas aplicações com modelos estatísticos de processos reais, tal como o estudo do controle de velocidade em veículos motores, filas de clientes chegando num aeroporto, taxas de câmbio, crescimento populacional de certas espécies de animais. O algoritmo conhecido como PageRank, que foi originalmente proposto para o motor de busca do Google, é baseado em um processo de Markov.

A representação de uma cadeia de Markov é dada por uma matriz aleatória que representa a probabilidade de transição entre os estados.



O diagrama de estados representa um mercado de ações hipotético em que o diagrama representa a chance de uma das opções ter uma semana de alta, tem-se Bull market, Bear market, e Stagnant market. De acordo com o diagrama Bull é seguido de outra semana de alta 90% das vezes, Bear 7.5% das vezes, e Stagnant 2.5% das vezes, enquanto que Bear é seguido de outra semana de alta 80% das vezes, Bull 15% das vezes, e Stagnant 5% das vezes, e Stagnant é seguido por uma semana de alta 50% das vezes, Bull 25% das vezes, e Bear 25% das vezes.

A matriz de transição que representa o diagrama acima é:

$$P = \begin{bmatrix} 0.9 & 0.075 & 0.025 \\ 0.15 & 0.8 & 0.05 \\ 0.25 & 0.25 & 0.5 \end{bmatrix}.$$

A potência dessa matriz (P^d) representa probabilidade de começando no estado i terminar no estado j depois de d transições.

O autovetor associado ao maior autovalor, no caso 1 ($\lambda = 1$), representa a probabilidade de cada estado após infinitas transições, essa probabilidade é dada por cada componente do autovetor. A primeira componente representa a probabilidade do primeiro estado, a segunda componente do segundo estado, e assim por diante.

```
In [78]: import numpy as np



         # definindo a matriz de transição
         #P = np.array([[0.25, 0.25, 0.5], [0.25, 0.5, 0.25], [0.5, 0.25, 0.25]])
         #P = np.array([[0, 0.25, 0.5], [0.25, 0, 0.25], [0.5, 0.25, 0]])
         P = np.array([[0.9, 0.075, 0.025], [0.15, 0.8, 0.05], [0.25, 0.25, 0.5]])
         print(P)
```

```
[[0.9   0.075 0.025]
 [0.15  0.8   0.05 ]
 [0.25  0.25  0.5  ]]
```

```
In [79]: # Probabilidade de transição após 2 saltos

         print( np.linalg.matrix_power(P, 2) )
```

```
[[0.8275  0.13375 0.03875]
 [0.2675  0.66375 0.06875]
 [0.3875  0.34375 0.26875]]
```

```
In [80]: # Probabilidade de transição após 3 saltos

         print( np.linalg.matrix_power(P, 3) )
```

```
[[0.7745  0.17875 0.04675]
 [0.3575  0.56825 0.07425]
 [0.4675  0.37125 0.16125]]
```

```
In [81]: # Probabilidade de transição após 10 saltos

         print( np.linalg.matrix_power(P, 10) )
```

```
[[0.64328888 0.29579253 0.06091859]
 [0.59158507 0.34309614 0.06531879]
 [0.60918588 0.32659396 0.06422016]]
```

In [82]:
```python
# Probabilidade de transição após 100 saltos

print( np.linalg.matrix_power(P, 100) )
```

```
[[0.625  0.3125 0.0625]
 [0.625  0.3125 0.0625]
 [0.625  0.3125 0.0625]]
```

In [83]:
```python
# computa os autovalores e autovetores da matriz A

autovalores, autovetores = np.linalg.eig(P.T)

print( autovalores )
print( autovetores )


print('\nProbabilidade dos estados após infinitas transições \n%s'%(autovetores[:, 0] /
```

```
[1.         0.74142136 0.45857864]
[[ 0.89087081  0.7365804  -0.27569188]
 [ 0.4454354  -0.67339179 -0.52773313]
 [ 0.08908708 -0.06318861  0.803425  ]]

Probabilidade dos estados após infinitas transições
[0.625  0.3125 0.0625]
```