Predição

October 31, 2019

0.1 Predição

0.1.1 O problema do Boston Bruins

Em uma série de 7 jogos, o Boston perdeu os dois primeiros jogos por 0-1 e 2-3, depois venceu os próximos dois por 8-1 e 4-0. Neste ponto da série, qual é a **probabilidade de o Boston vencer o próximo jogo** e qual a **probabilidade de vencer o campeonato**?

Suposições - É razoável acreditar que os **gols no hóquei** seja pelo menos aproximadamente um **processo de Poisson**; - Podemos assumir que, contra um adversário em particular, cada equipe, a longo prazo, tem algumas médias de gols, denotados .

Estratégia para responder a essa pergunta é

- Use as **estatísticas dos jogos anteriores** para escolher uma distribuição **priori** para .
- Use a pontuação dos **quatro primeiros jogos** para estimar para cada equipe.
- Use as **distribuições posteriores de** para calcular a **distribuição de gols** para cada equipe, a **distribuição de diferencia de gols** e a **probabilidade de que cada time ganhe o próximo jogo**.
- Calcule a **probabilidade** de cada equipe **vencer** a série.

A **distribuição priori** é aproximadamente **gaussiana**, com média de 2,8 e desvio padrão de 0,3.

0.1.2 Processos de Poisson

Os processos de Poisson foi usado para **modelar os gols marcados** em um jogo de hóquei;

- É uma versão contínua de um processo de Bernoulli;
- Em muitos sistemas reais, a probabilidade de um evento muda com o tempo.
  - Os gols são **mais ou menos prováveis** em momentos diferentes **durante um jogo**.

Podemos calcular a distribuição de gols por jogo de forma eficiente e a distribuição do tempo entre os gols.

0.1.3 As posteriores

[4]:
```
def Likelihood(self, data, hypo):
    lam = hypo
    k = data
    like = thinkbayes.EvalPoissonPmf(k, lam)
    return like
```

Cada **hipótese** é um **valor possível de** ; **data** é o **número observado de gols**, k. suite1 = Hóquei ('bruins') suite1.UpdateSet ([0, 2, 8, 4])

suite2 = Hóquei ('canucks') suite2.UpdateSet ([1, 3, 1, 0]

- atualizá-los com os gols dos quatro primeiros jogos.

0.1.4 A distribuição de gols

Para calcular a **probabilidade** de cada **equipe vencer** o próximo jogo, precisamos calcular a **distribuição de gols(único jogo)** para cada equipe;

Não sabemos o valor de lambda, mas temos uma distribuição para os **valores posséveis de lambda (distribuição de gols é Poisson)**;

[5]:
```
def MakePoissonPmf(lam, high):
    pmf = Pmf()
    for k in xrange(0, high+1):
        p = EvalPoissonPmf(k, lam)
        pmf.Set(k, p)
    pmf.Normalize()
    return pmf
```

A **distribuição geral de gols** é uma **mistura** dessas distribuições de Poisson

[6]:
```
def MakeGoalPmf(suite):
    metapmf = thinkbayes.Pmf()

    for lam, prob in suite.Items():
        pmf = thinkbayes.MakePoissonPmf(lam, 10)
        metapmf.Set(pmf, prob)

    mix = thinkbayes.MakeMixture(metapmf)
    return mix
```

0.1.5 A probabilidade de ganhar

Para obter a **probabilidade de vitória**, primeiro calculamos a **distribuição do diferencial de gols**:

[7]:
```
def winnig():
    goal_dist1 = MakeGoalPmf(suite1)
    goal_dist2 = MakeGoalPmf(suite2)
    diff = goal_dist1 - goal_dist2
```

Se o diferencial de gols for **positivo**, os **Bruins** vencem; se **negativo**, os **Canucks** vencem; se 0, é um empate:

[8]:
```
def prob():
    p_win = diff.ProbGreater(0)
    p_loss = diff.ProbLess(0)
    p_tie = diff.Prob(0)
```

Com as distribuições da seção anterior, **p_win é de 46%**, **p_loss é de 37%** e **p_tie é de 17%.**

0.1.6 Morte súbita

A estatística importante não é de **gols** por jogo, mas de **tempo** até o **primeiro gol**

Tempo entre os gols é distribuído **exponencialmente**;

Temos uma distribuição posteriori de valores possíveis - criamos um meta-Pmf e - calculamos uma mistura de Pmfs

[9]:
```
def MakeGoalTimePmf(suite):
    metapmf = thinkbayes.Pmf()

    for lam, prob in suite.Items():
        pmf = thinkbayes.MakeExponentialPmf(lam, high=2, n=2001)
        metapmf.Set(pmf, prob)

    mix = thinkbayes.MakeMixture(metapmf)
    return mix
```

O **Bruins** têm mais chances de marcar em **tempos mais curtos**;

O tempo de **Canucks** marcar é provável que seja **mais longo**;

[13]:
```
def prob_prorro():
    time_dist1 = MakeGoalTimePmf(suite1)
    time_dist2 = MakeGoalTimePmf(suite2)
    p_overtime = thinkbayes.PmfProbLess(time_dist1, time_dist2)
```

Para os Bruins, a probabilidade de ganhar na prorrogação é de 52%.

Para os Bruins, a chance geral de ganhar o próximo jogo é de 55%.

[12]:
```
def prob_prox():
    p_tie = diff.Prob(0)
    p_overtime = thinkbayes.PmfProbLess(time_dist1, time_dist2)
```

```
    p_win = diff.ProbGreater(0) + p_tie * p_overtime
```

A chance dos Bruins de vencer a série é de 57%

[14]:
```
def prob_serie():
    # win the next two
    p_series = p_win**2

    # split the next two, win the third
    p_series += 2 * p_win * (1-p_win) * p_win
```

0.1.7 Discussão

- Análise neste capítulo é baseada em decisões de modelagem;
- Usou apenas os quatro primeiros jogos da série do campeonato,
- Se usar mais jogos anteriores, talvez seja melhor dar mais peso aos jogos recentes.
- Poderíamos usar os resultados de todos os jogos da temporada regular para estimar a taxa de pontuação de cada equipe;
- Os resultados são sensíveis a priori.

[ ]:
```

```