# [2017.08.12.토]WTTE-RNN 정리자료

**YBigTa 10기 : 김태한, 손진원**

원본링크 (https://ragulpr.github.io/2016/12/22/WTTE-RNN-Hackless-churn-modeling/)

위 링크의 게시글(WTTE-RNN-Less hacky churn prediction)을 번역하고 공부한 글입니다.

---

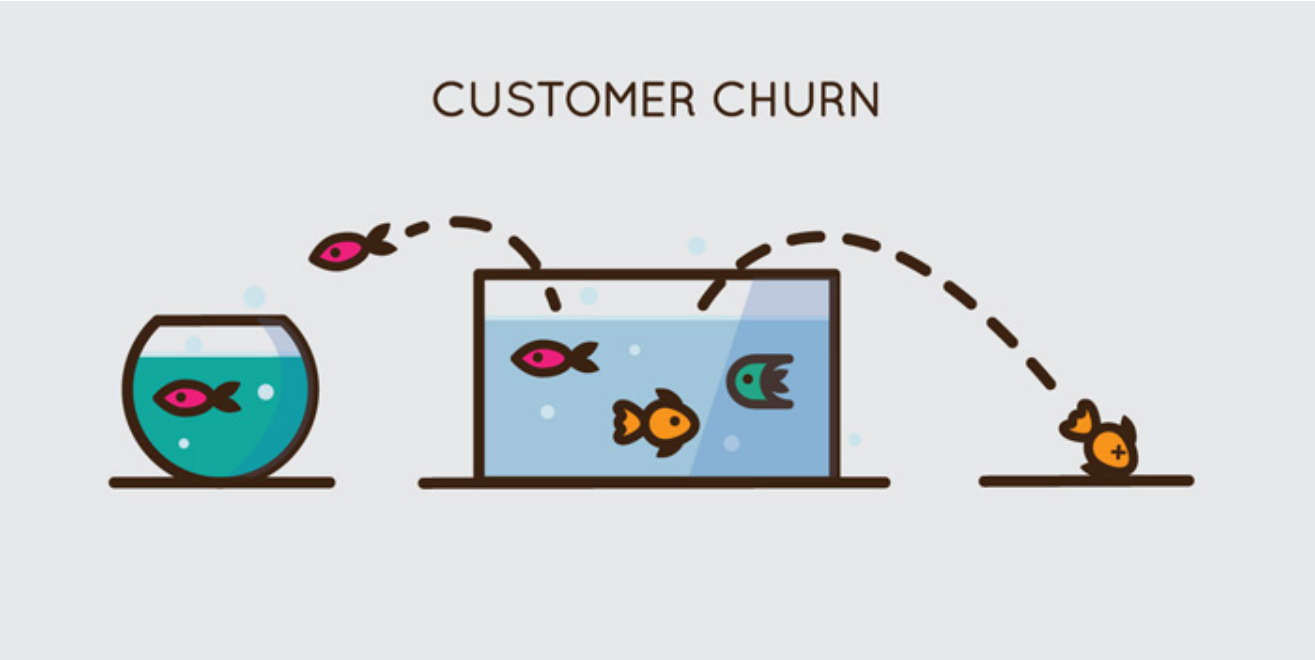
## 1. Churn prediction의 어려움

Churn의 정의 : 매일경제용어사전 Churn

- (통신회사 마케팅 용어의 경우) 요금 등의 이유로 사업자를 자주 바꾸는 고객 --> 고객이탈

Churn rate 정의 : 위키피디아 정의

- 특정 기간에 그룹으로 부터 벗어나는 개인의 수나 품목의 수를 측정하는 용어




(이미지 출처 Churn rate 정의 https://sendpulse.com/support/glossary/churn-rate)

> 고객 n은 feature vector $x^n$과 binary 타겟 $y^n$을 갖는다.
>
> $$y^n = \begin{cases} 1, & \text{if customer n will churn .} \\ 0, & \text{if customer n won' churn.} \end{cases}$$
>
> 우리가 구축하고 싶은 머신러닝 모델은 고객 n의 churning 확률을 추측하기 위하여 feature $x^n$ 을 사용한다.

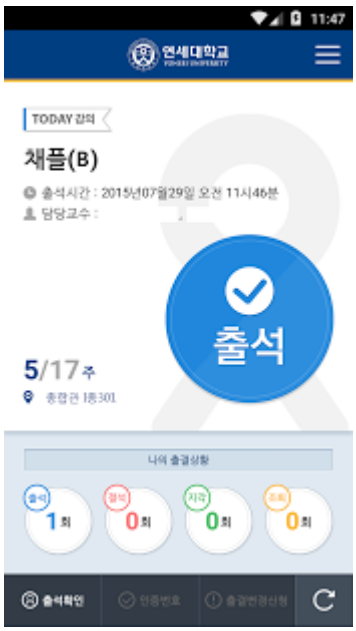
하지만 현실세계에서 이러한 모형은 잘 맞지 않는다. 문제는 언어적인 오류에서 기인한다 - churn을 둘러싼 우리의 직관은 수수께끼문제처럼 보인다.

- **'will'** 의 의미는 무엇인가? 우리 모두는 언젠가는 'churn'한다.
- **'customer'** 의 의미는 무엇인가? 특정 시점의 고객? 구독계획? 특정 고객 id의 non-churned 기간?
- feature vector의 shape는 어떤 형태인가? 정적인 feature로 고정된 너비인가? 아니면 서로 다른 측정 오차를 내포하는 시간의 흐름에 걸친 합쳐진 형태인가? 정말로 각 고객들에 대한 시계열 자료인가?

- **'churn'**의 의미는 무엇인가? 매우 어려운 질문이다.
- 당신은 아마도 현재 churn rate를 알지 못할 것이다. 당신은 churn rate를 예측할 churn-model이 필요하다.

## churn 예측 = non-event 예측

churn을 예측하는 것 보다는, non-churn을 예측하는 편이 낫다.  만약 미래에 어떤 사건이 발생한다면, 우리는 과거의 어느 시점으로 부터 해당 이벤트 까지의 시간을 정의할 수 있다. 만약 어떤 고객이 미래 이벤트 시점까지 더 긴시간 동안 떠나 있다면, 그 고객은 더욱 churned한 것이라고 정의할 수 있다.

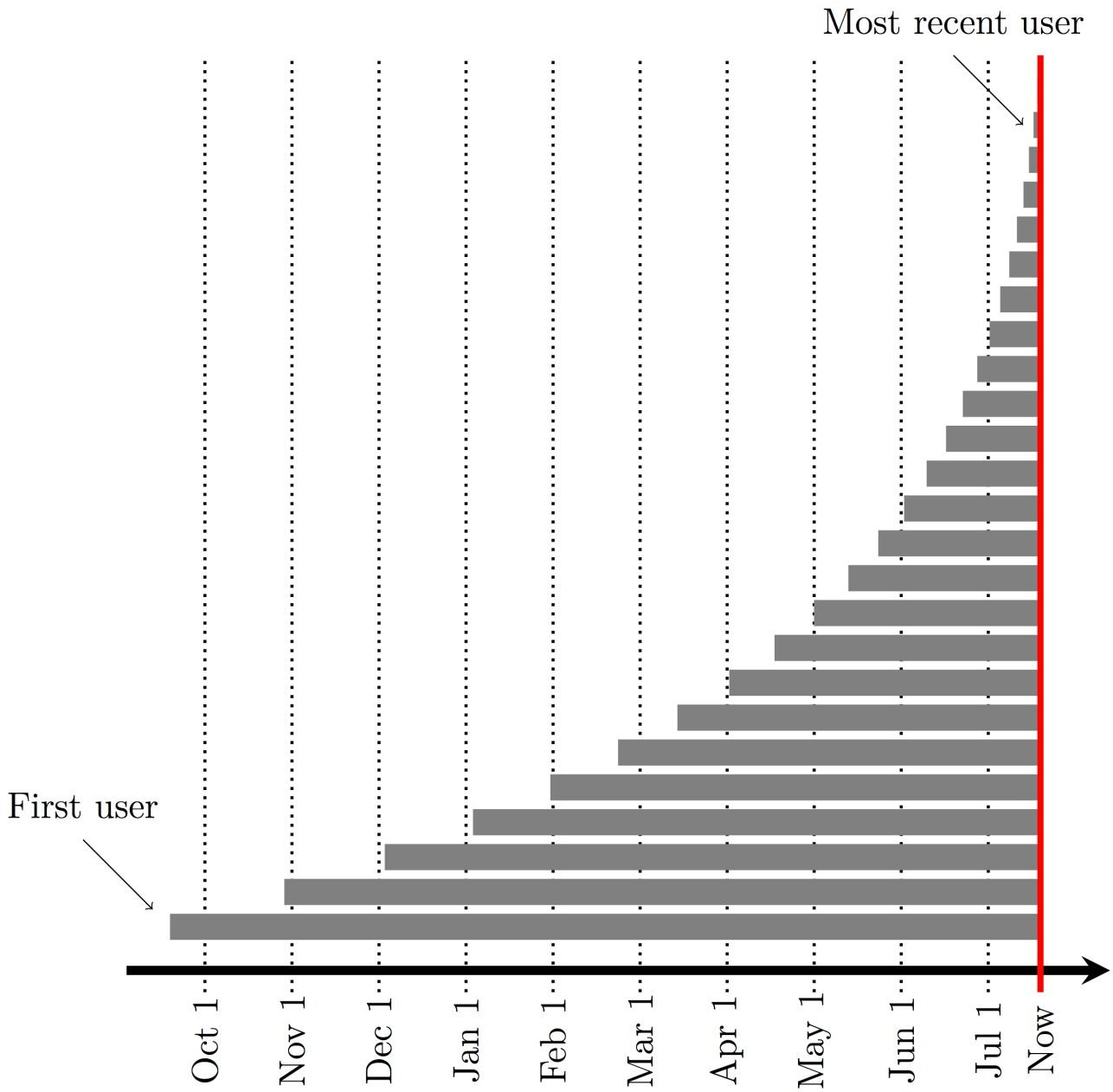



(전자출결 예시)

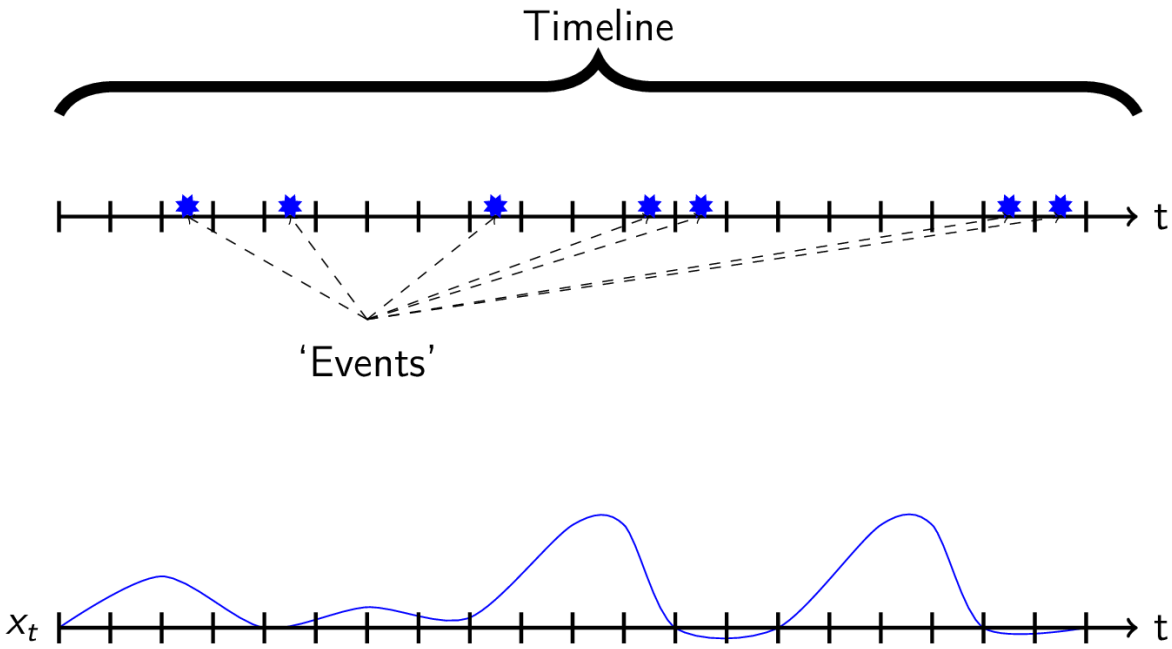
(이미지 출처 : https://play.google.com/store/apps/details?id=kr.ac.yonsei.attendance&hl=ko)

전자출결 단말기 기준 예시 : 오전 9시 수업의 경우 단말기는 8시 50분 부터 9시 6분까지 출결 체크를 한다. 16분 사이에 churn은 출석체크가 되는 것. non-event는 출석체크 되지 않는 시간 즉 출석체크 사이의 기간.

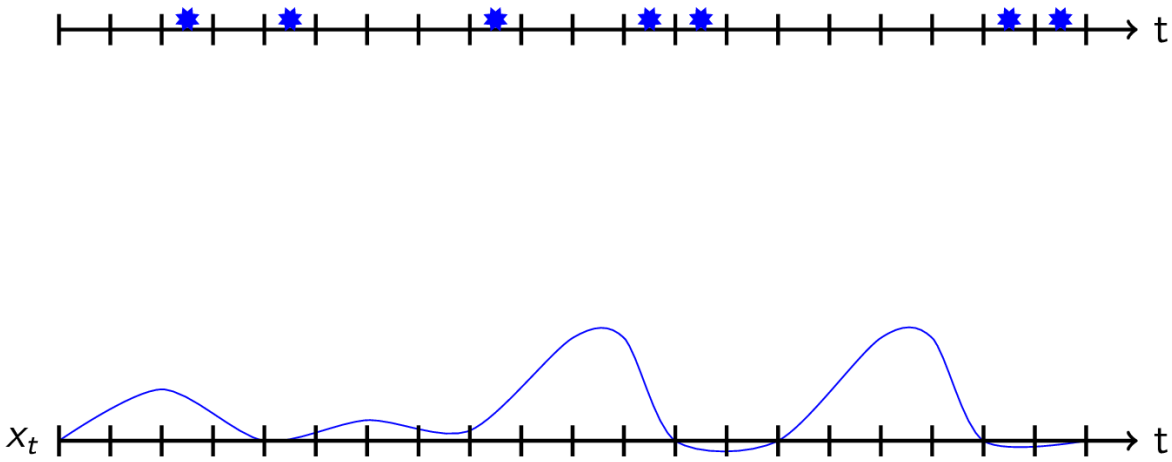
우리가 사용할 raw data는 각 고객에 대한 records의 series이다. 각 고객을 시작시점 부터 현재시점 까지의 timeline이라고 생각해보라.  각 timeline을 쌓으면 아래와 같은 그림이 된다.



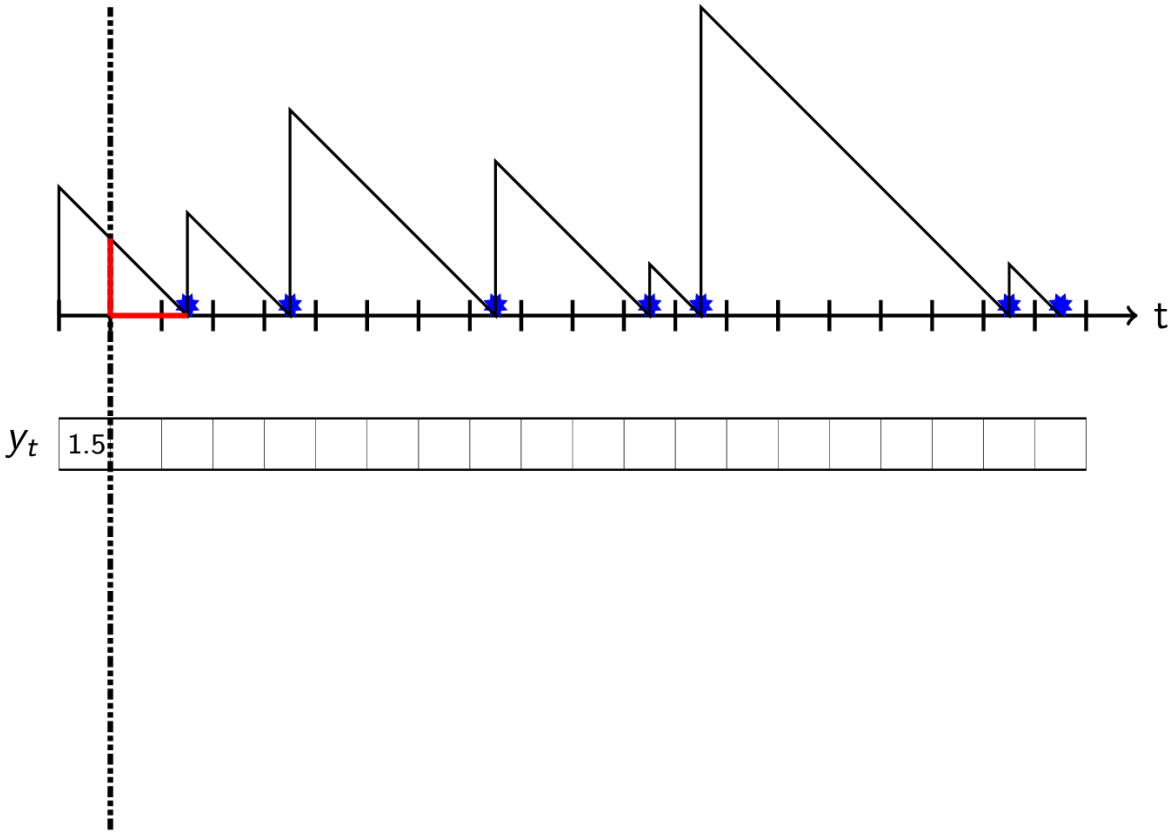

관심있는 churn의 정의에 따라서 *events*(구매 혹은 로그인 등)와 이벤트 예측에 사용될 *features*(클릭, 구매, 로그인 등) 두 가지로 dataset을 나누어야한다. 아래 그림은 개별 timeline에 대한 *events*와 *features* 구분의 예시이다.




과거 데이터로 순차적인 미래를 예측하기 위하여 features를 이용한다.




다음으로 소개할 노하우는 예측하고 싶은 문제를 정의하는 것이다. 가장 자연스러운 문제는 **각각의 timestep t에서 다음 event까지의 시간** $y_t$ 를 예측하는 것이다. 나는 이 문제를 **TTE**라고 부른다. 아래는 TTE 문제를 시각화한 것이다.

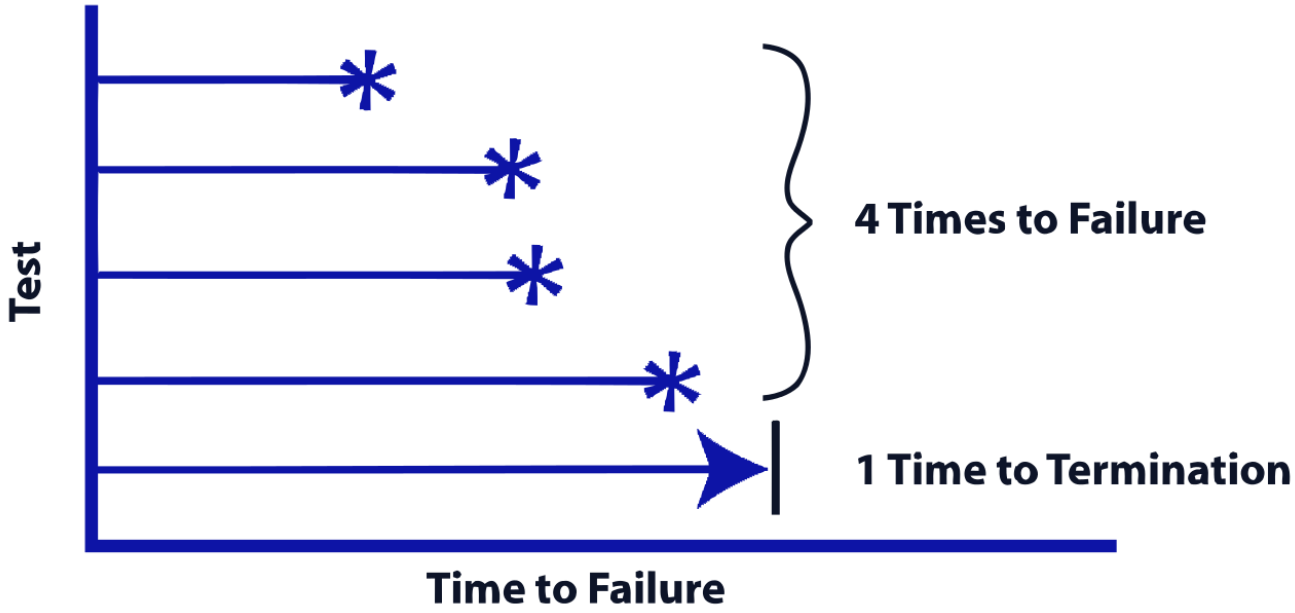


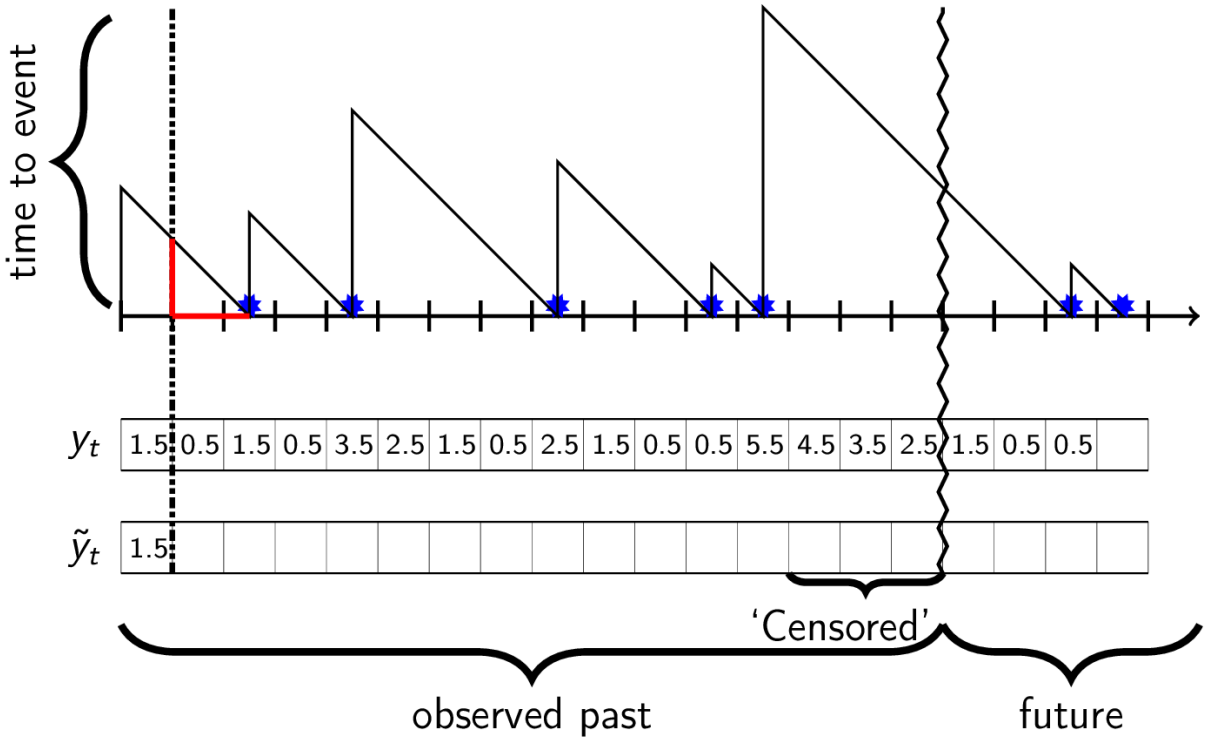

만약 어느 고객이 다음 구매까지 더 긴 시간이 소요된다면, 그 고객은 more churned라고 말할 수 있다. 그렇다면 이러한 파동을 점추정하는 것이 쉬운가? 회귀분석 문제로 접근하는 것은 쉬운가? Nope.

만약 특정 고객이 다시는 구매를 하지 않는 다면 $y_t$ -> $\infty$ 이다. 문제는 우리가 영원히 기다려야 한다는 것. 여기서 이러한 유형의 데이터의 근본적인 문제가 등장한다 : **censoring**.

## Censored data

현실세계에서는 관찰된 과거로부터 event-data를 기록한다. (첫번째 고객의 출현 부터 현재까지). 이것은 마지막 event 발생 시점 이후로 다음 시점(unseen) event 까지의 실제 시간에 대한 데이터가 없다는 것이다. 우리가 가진 것은 학습을 위해 사용가능한 lower bound $\hat{y}_t \leq y_t$ 이다. 이처럼 부분적 관측을 right censored data라고 한다.




censored 된 관측 $\tilde{y}_t$는 "시점 t에서 사건 발생까지는 적어도 $\hat{y}_t$남았다."라고 해석할 수 있다. censored 데이터로 어떻게 모형을 학습할 것인가?

## 2. Censored data분석에 쓰이는 모델s

churn-models 문제의 대부분은 sliding box model을 사용한다.

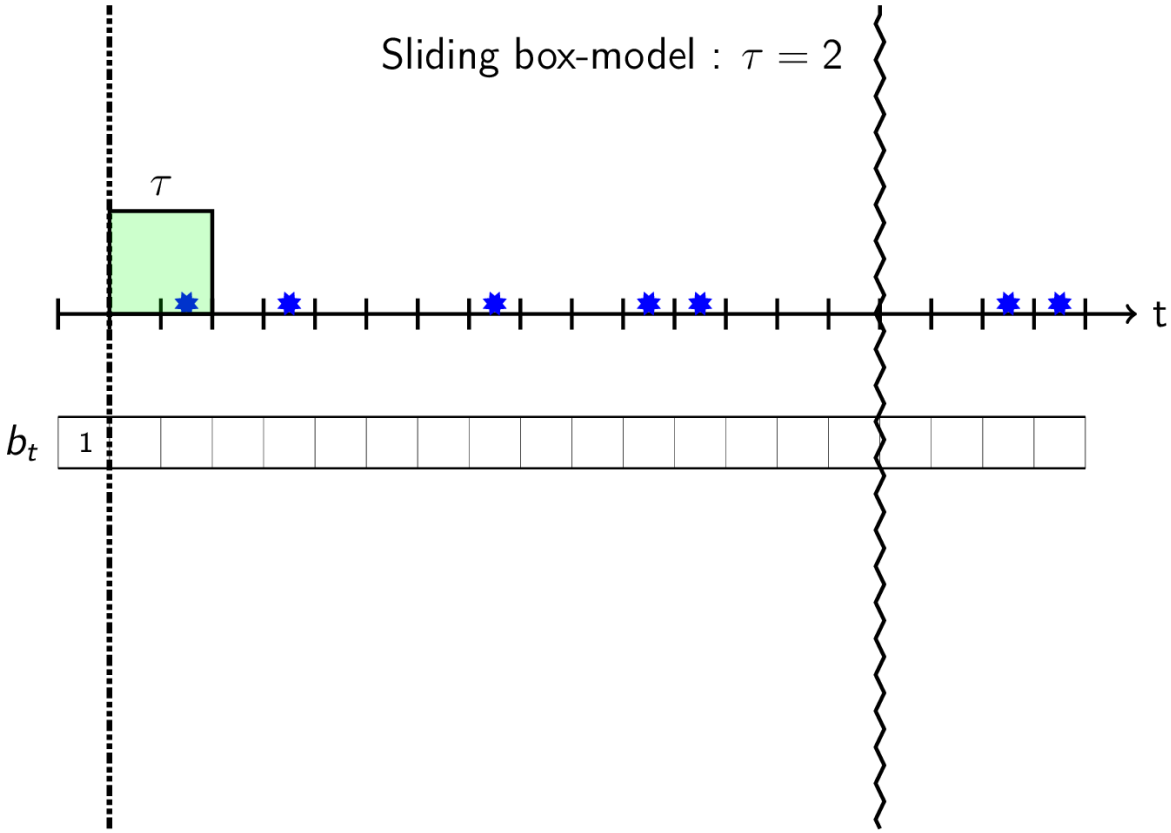
## Sliding box model

TTE 를 직접적으로 예측하는 것 대신에, 우리는 특정 사건이 시점 $\tau$이내에 발생할 것인지를 예측한다.

timesetp $t$에서의 관측된 target value를 $b_t$라고 정의한다.

$$b_t = \begin{cases} 1, & \text{if event in } [t, t+\tau) \text{ .} \\ 0, & \text{if no event in } [t, t+\tau). \\ unknown, & \text{else} \end{cases}$$

이 수식은 아래 그림과 같이 sliding box로 나타낼 수 있다.




위 그림을 보면 마지막 $\tau$ 단계에서 사건이 발생하지 않는 unknowns가 등장한다. (blank -- 0도 1도 아님)

확률 목적 함수를 세우기 위해서, 시간에 따라 변화하는 모수 $\theta_t$를 갖는 베르누이 분포로 부터 $b_t$가 독립적으로 도출되었다고 생각해보자. $\theta_t$는 timestep t로부터 $\tau$ time 이내에 사건이 발생할 확률이다.

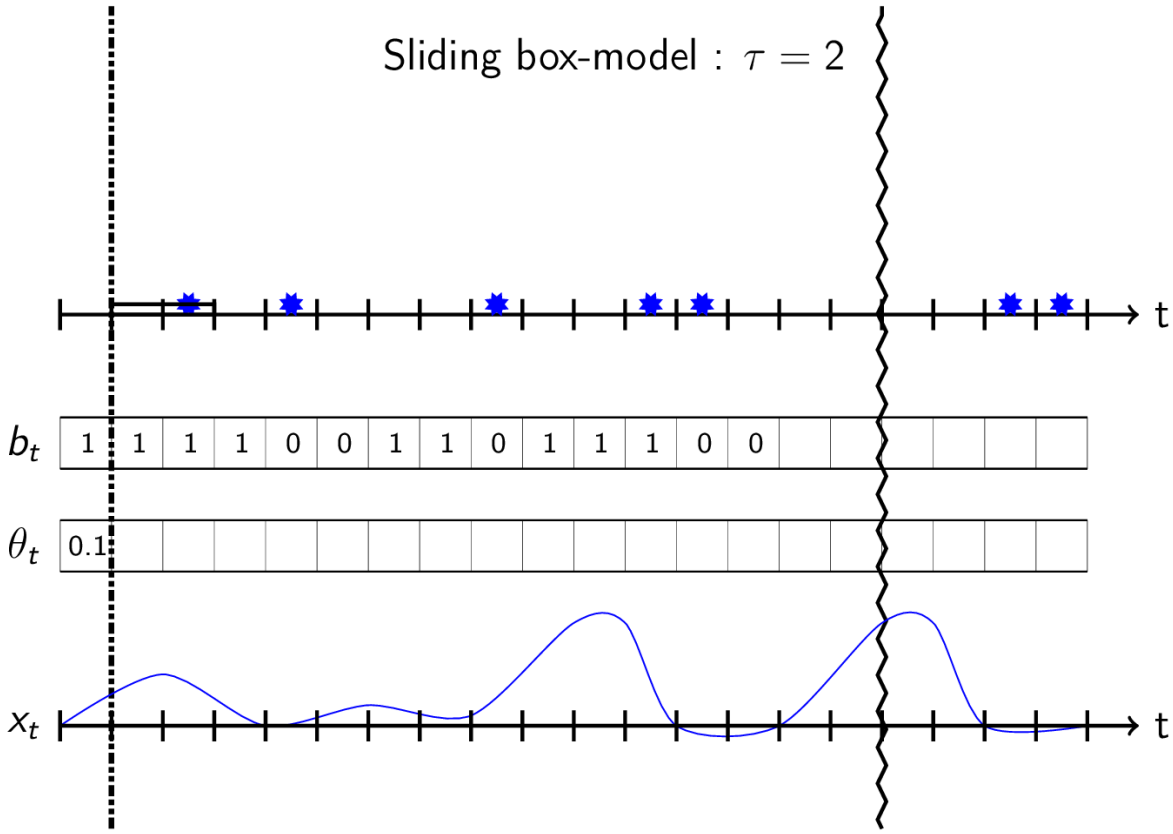
$$B_t \sim Bernoulli(\theta_t)$$

$$Pr(B_t = b_t) = \theta_t^{b_t} * (1-\theta_t)^{1-b_t}$$

시점 $\theta_t$에서 $\theta_t = g(x_{0:t})$가 되고 데이터 $x_{0:t}$를 갖는 머신러닝모델 $g$를 생각해보자. 목적함수는 아래와 간다.

$$\underset{g}{\text{maximize}} \; log(\mathcal{L}(\theta_t)) := log(\theta^{b_t} * (1-\theta_t)^{1-b_t})$$

이 모델에 대한 시각화는 아래 그림과 같이 나타낼 수 있다. box의 높이(height)는 사건의 발생 확률 $\theta_t$를 나타낸다.




churn-model 활용하기

churn-model의 명확한 이점이 있다.

- 단순함과 명확성. 작동원리를 이해하기 쉽다.
- 유연성. 어떤 binary 예측 알고리즘에도 사용가능하다. Xgboost나 랜덤포레스트 혹은 char-level RNN까지.

단점

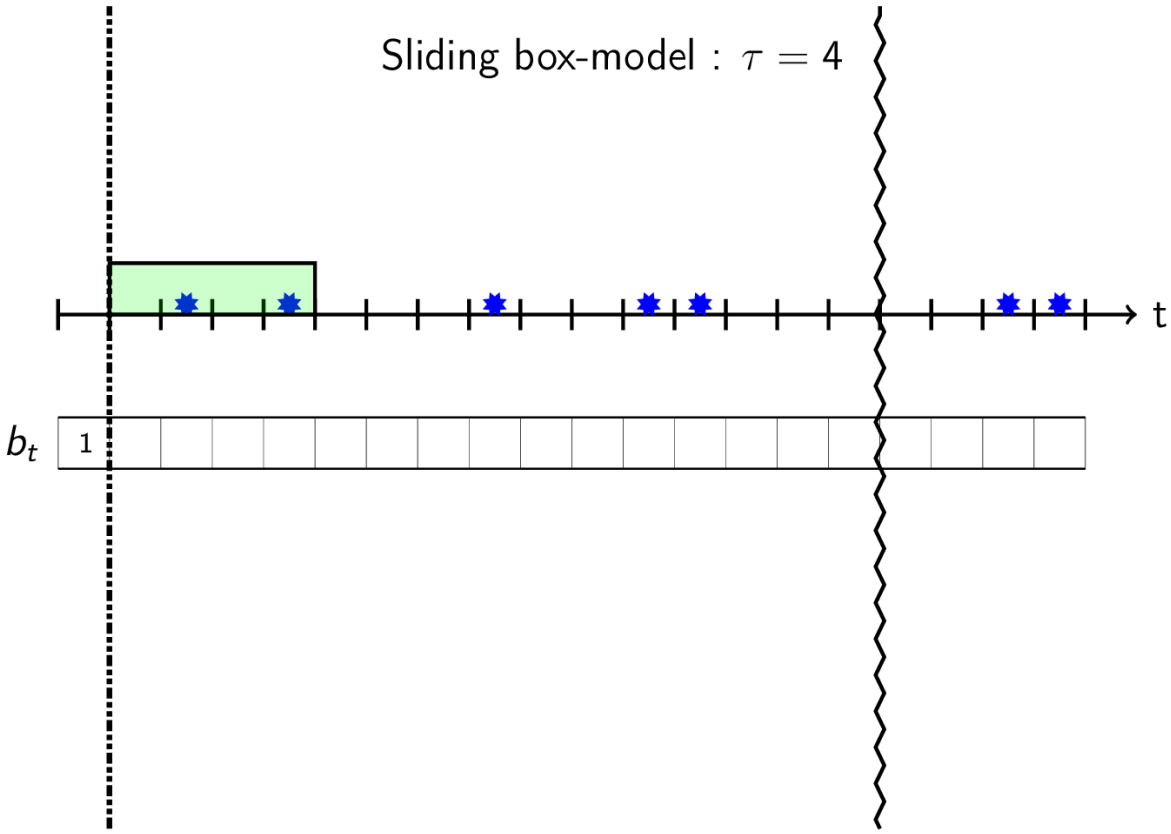
- 예측이 유익하지 못함.

'30일 동안 이벤트가 발생하지 않는다.'에 대한 예측된 확률은 과연 유효한가?

binary timeintervals 대신에 multiple timeintervals을 예측할 수도 있다. 하지만 이 경우 더 많은 hyperparameters를 필요로 한다. 어떤 경우라도 우리는 예측 가능한 미래에 event가 없는 시점을 예측하고 싶어하기 때문에 아마도 가능하다면 큰 $\tau$를 원할 것이다. 이경우에도 문제가 생긴다

- 마지막 $\tau$ timesteps를 학습에 사용할 수 없다.

아래 그림에서 사건이 발생한 곳은 $b_t = 1$ 라고 할 수 있다. 하지만 boundary 바깥의 사건이 발생하지 않은 구간을 배제할 수 없다. 이러한 생략을 명쾌하게 설명할 수 없다면, 어떤 biases나 class-imbalances가 발생하는지 알 수 없다. 가장 쉽고 안전한 방법은 마지막 $\tau$ steps의 모든 관측을 생략하는 것이다. 이것은 $\tau$가 클수록 학습할 최근 데이터가 더 줄어든다는 것이다. $\tau$를 크게 잡는 것은 모든 target을 1로 만드는 문제가 발생한다.




이 경우 학습하기에는 매우 blunt한 signal이다. 반대로 만약 $\tau$를 너무 작제 잡으면 output이 의미 없어진다.  또 너무 큰 $\tau$로는 학습할 수 없다. sliding box 모델을 오약하자면

- 이용하기에 매우 까다롭고 어려움

parameter $\tau$ 가 이 model의 대부분을 설명한다. $\tau$를 조절하는 것은 data pipeline을 변경하는 것이며 또한 예측된 결과와 모델의 성능의 의미를 변화시키는 것이다. 최적의 균형점을 찾는 일은 어렵고 시간이 많이 소모되는 반복작업이다.

특정 환경에서 binary target과 $\tau$는 쉽게 정의되고 sliding box 모델은 잘 들어 맞을 수 있다. Moz developer blog

는 그들이 이와 유사한 문제에 성공적으로 RNN을 사용한 글을 공유하였다.

## Making it a learning to rank-problem

churn을 정의하고 예측하는 것 대신에, 누가 다른이들 보다 더 churned한지 예측할 수 있다.  churn-model을 흔히 사용하는 분야는 riskset에 따라서 고객의 순위를 매기는 점수를 사용하는 것이다. 순위 매기기, learning to rank는 머신러닝 토픽 중 하나이고 censored data를 모델에 포함할 수도 있다.

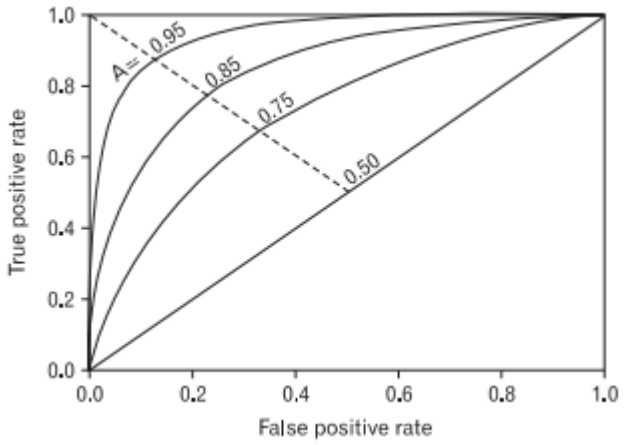
만약 사건 발생까지 최소한 7일이 남았다는 것을 알고있다면, 3<7이기 때문에 사건 발생까지 3일이 남은 시점도 알 수 있다. 순위매기기는 binary target을 계산한 모든 쌍체비교에 의하여 정의된다는 점을 알아두자.

$$r_{ij} = \begin{cases} 1, & \text{if } \hat{y}_i \leq \hat{y}_j \text{ and obs } i \text{ not censored .} \\ 0, & \text{if } \hat{y}_i > \hat{y}_j \text{ and both not censored.} \\ unknown, & \text{else} \end{cases}$$

머신러닝 모델 $g$는 관측 $i$과 $j$로 부터 features를 사용하고 다음과 같은 예측을 한다.

$$\hat{r}_{ij} = g(x_{0:i}, x_{0:j})$$

예시를 들면, censored된 data를 갖지 않고 target 을 discrete한 $\hat{r}_{ij} \in \{0,1\}$ 로 고정하면 $\|\hat{r}_{ij} - r_{ij}\|$를 최소화 하는 것이 AUC 를 최적화 하는 것이다.




(이미지출처 : http://newsight.tistory.com/53)

- **AUC** : ROC커브의 밑 면적. 1에 가까울 수록 성능이 좋다.
- true positive rate (=민감도, sensitivity) : 실제로 1인 값을 1이라고 예측한 비율
- false positive rate (= 1- 특이도,  1- specificity ) : 실제로 0인 값을 1이라고 잘못 예측한 비율

이 문제에 대한 확률적인 모델은 생각해보지 않았지만 만약 이 문제를 continuously 하게 $\hat{r}_{ij} \in [0,1]$로 만들 수 있다면 어떠한 binary 예측 알고리즘을 이용하여도 결과를 얻을 수 있다.

dataset이 모든 관측의 쌍체조합이기 때문에 학습 dataset은 이차적으로 증가한다. 또한 개별 고객이 churned인지 아닌지를 말할 수 없는 문제가 존재한다. 단지 이 고객들은 다른 고객보다 more churned라고 말할 수 있을 뿐이다.

최근 랭킹학습과 AUC최적화에 대한 멋진 연구들이 진행 중이다. this article of the year 2015를 추천한다.

## 3. churn-model에서 원하는 것

Churn 모델링은 과학이라기 보다는 기술에 가깝다. 결국 우리는 churned 고객이 무엇인지에 대한 답을 내려야 한다. 이 대답에 대한 결론을 내릴때 몇 가지 팁이 있다. 'churn'을 정의할 때의 목표는 아래와 같다.

- 부활(churn 이벤트 재발생) 확률을 최소화 하는 것
    - 영속성. 이해관계자들은 churned한 고객들이 영원히 churn 상태일 것이고 미래에 어떠한 가치도 가져올 것이라고 생각하지 않는다.
- detection(탐지) 확률을 최대화
    - 측정가능성. 만약 당신의 churn에 대한 정의에 상응하는 현실이 측정가능하지 않고 예측가능하지 않다면 당신의 churn-정의는 쓸모가 없다.
- 당신의 정의에대한 해석가능성 극대화
    - churn에 대한 좋으면서도 안전한 정의는 사람들의 일반적 생각에 상응한다. 자연어에서의 오류는 당신이 이러한 가정을 바꿀 수 없다는 것이다. (아마도 같은 단어에 대해서 사람마다 생각하는 정의가 다르기에 발생하는 오류) 그래서 만일 이러한 개념을 명확하게 model할 수 없다면 churn에 대해 이야기하지 말라. 그 대신에 active vs. inactive 한 고객 상태 등 다른 용어를 사용하라.

마지막이 가장 중요한 포인트이다. 누군가는 'churn'모델링이 명백한 서비스에 대해서만 시도할 수 있는것이라 말할 수도 있다. Netflix는 churn-rates에 대한 과도한 보도로 주주들과 소송싸움을 한 경험이 있다. 법원은 churn에 대한 명확한 정의가 부재하다는 이유로 재판을 중지했다.

churn 모델링에 대한 명확한 방법이 존재하지 않기에 어떤 업무에 적용할 것인지 구체적으로 정해야 한다. 우리는 feature-engineering과 크고 무거운 modeling loop를 병목현상(여기선 걸림돌 정도)으로 생각한다. 이 케이스에서 좋은 머신러닝 모델은 다음과 일을 할 수 있다.

- 1. 반복되는 사건을 다룰 수 있다.
  2. 시간에 따라 변화하는 공변량을 다룰 수 있다.
  3. 일시적인 패턴을 학습할 수 있다.
  4. 변하하는 길이를 가진 sequences를 다룰 수 있다.
  5. censored data를 학습할 수 있다.
  6. 유연한 예측이 가능하다.

1~4번은 러닝머신 알고리즘으로써 RNN을 활용한다면 가능하다.

마지막 5~6번은 smart한 목적함수를 사용함으로써 달성 가능하다.

WTTE-RNN을 만나보자.

## 4. WTTE - RNN

이 모델을 이해하기 위해 알아야할 사항들은 매우 간단하다.

- $y_t^n$은 시점 $t = 0, 1, \ldots, T_n$에서 $n = 1, \ldots, N$ 에 대한 **TTE**이다.
- $x_{0:t}^n$ 데이터는 시점 t까지 존재한다.
- $u_t^n$은 데이터가 censored 되었다면 $u_t^n = 0$ 혹은 censored되지 않았다면 $u_t^n = 1$ 이다.

특별한 목적함수는 생존분석에서 유래한다. 목적은 아래 수식과 같이 최대화 하는 것이다.

$$\sum_{n=1}^{N} \sum_{t=0}^{T_n} u_t^n \cdot log[Pr(Y_t^n = y_t^n | x_{0:t}^n)] + (1 - u_t^n) \cdot log[Pr(Y_t^n > y_t^n | x_{0:t}^n)]$$

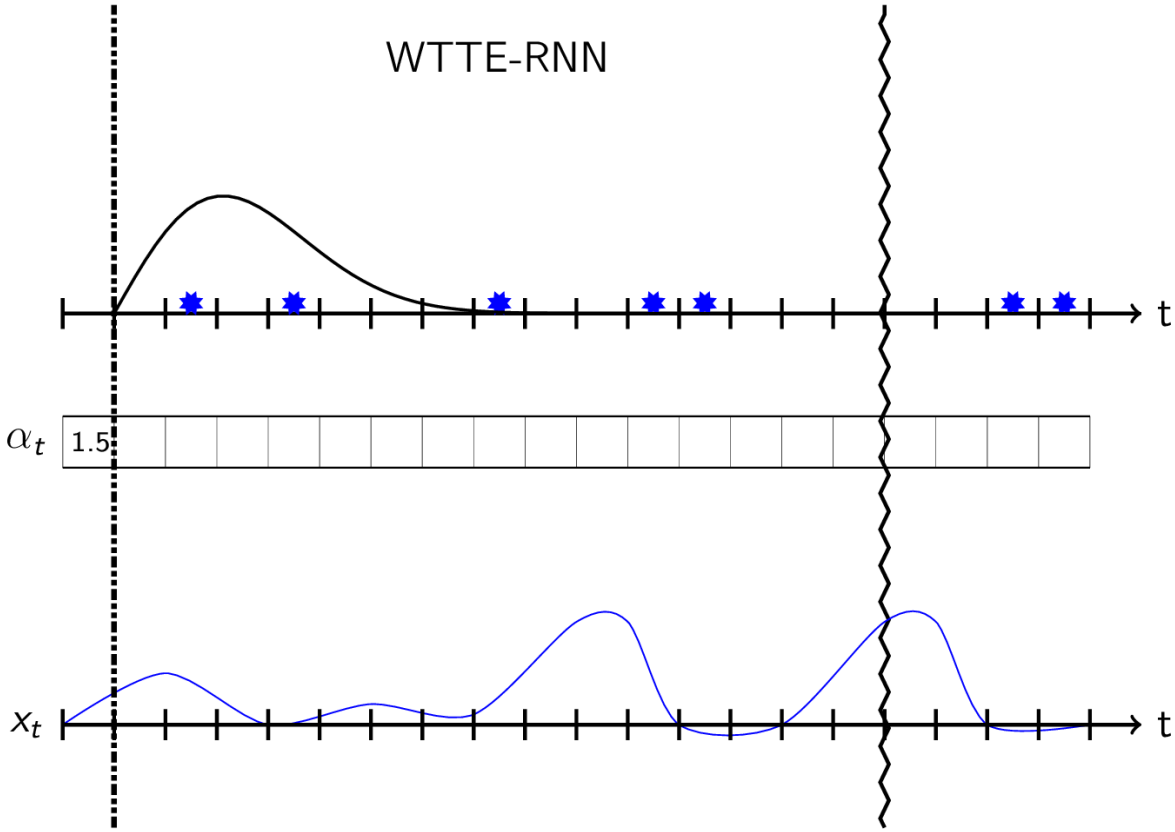
여기서 $Y_t^n$은 랜덤 실험이다.  이 문제를 다음과 같이 머신러닝 문제에 접목할 수 있다.

- 1. 각 시점에서 사건발생까지의 시간 $Y_t$는 $Pr(Y_t \leq y_t || \theta_t)$로 정의된 모수 $\theta_t$를 갖는 임의의 분포를 따른다고 가정한다.
  2. 모수 $\theta_t$는 timestep $t$를 인풋으로 하는 feature history를 갖는 머신러닝 모델(RNN 같은)의 output이다. 즉, $\theta_t = g(x_{0:t})$이다.
  3. censored data에 대해서 특별한 log-likelihood loss를 사용하여 머신러닝 모델을 학습한다.
  4. 각 step에서 다음 사건까지의 시간에 대한 *분포*를 예측할 수 있다.

WTTE-RNN이라고 할 수 있는 경우는 아래와 같다.

- $Y \sim Weibull$ with 모수 $\alpha_t \; and \; \beta_t$ . Y는  모수 $\alpha_t \; and \; \beta_t$ 를 갖는 Weibull 분포다.
- $\theta_t = \begin{pmatrix} \alpha_t \\ \beta_t \end{pmatrix} = g(x_{0:t})$ 는 RNN의 output이다.

다음 사건 발생까지의 시간에 대한 분포를 순차적으로 예측하는 과정은 아래와 같이 시각화 할 수 있다.



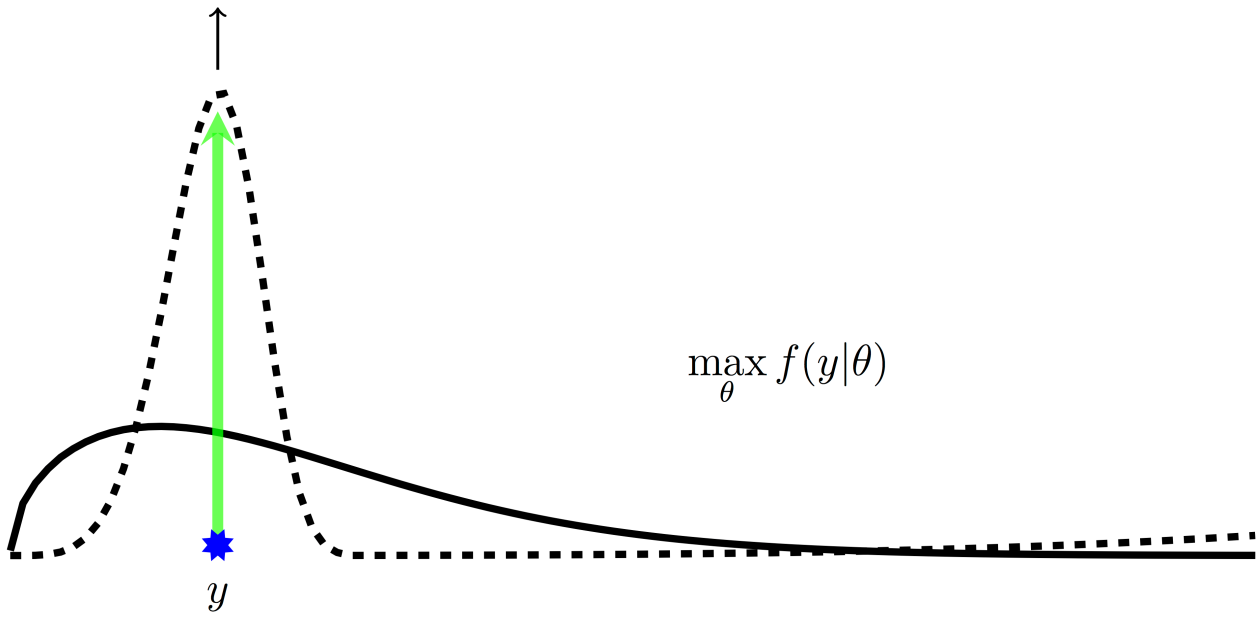

$\alpha_t$와 $\beta_t$는 feature data $x_t$에 의해 통제된다.

## censored data의 학습

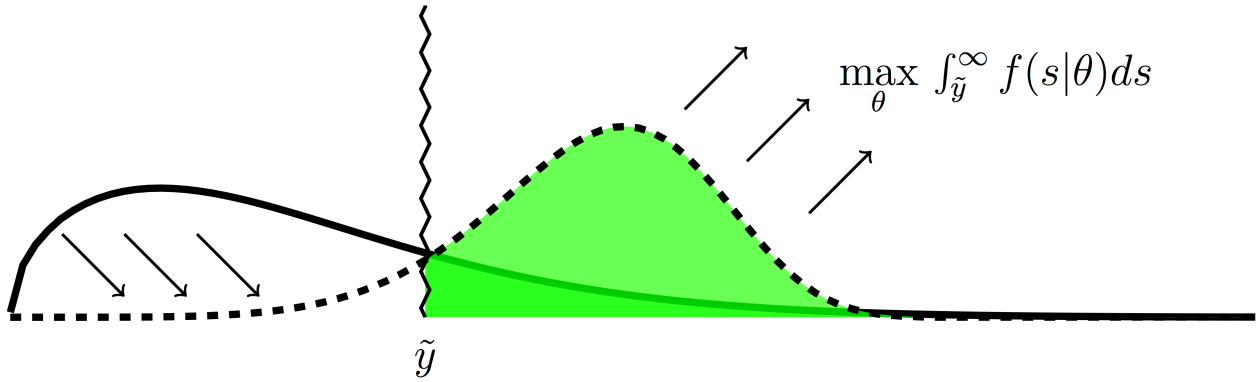
관측하지 않은 것을 어떻게 학습할 것인가? 마법처럼 보이지만 (불가능해 보이지만) 특별한 loss-function과 몇몇 가정과 상상력이 있다면 가능하다. 기본적인 idea는 생존분석에서 차용하였다. censoring이 어떻게 발생하였는지에 관한 가정이 주어졌을때 datapoint가 censored되었는지에 대한 결합확률분포의 likelihood를 이용할 수 있다. 관측된(censored된) TTE는 아래와 같이 쓸 수 있다.

$$\mathcal{L}(\theta) \propto \begin{cases} Pr(Y = y|\theta), & \text{if uncensored .} \\ Pr(Y > \tilde{y}|\theta), & \text{if right uncensored.} \end{cases}$$

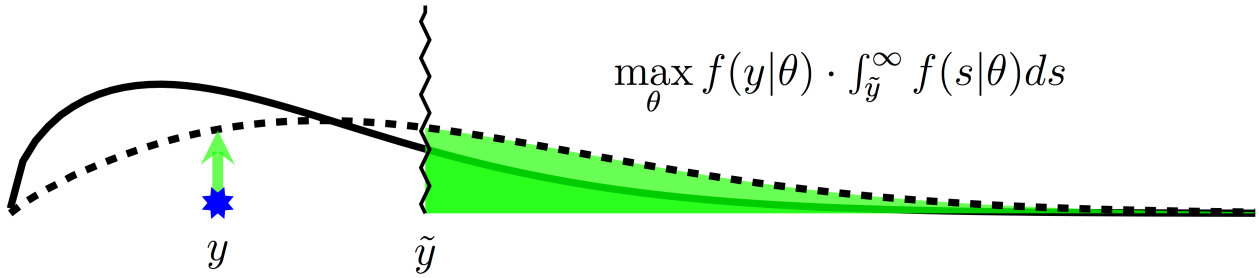
몇 가지 수학적인 가정과 증명이 필요하지만 다음과 같이 작동원리에 대한 직관은 명확하다.




**(a)** Uncensored observation: Push the pdf up at event




**(b)** Censored observation: Push mass over the point of censoring




**(c)** Uncensored and censored observation: Compromise

### Weibull 분포의 장점

와이불 분포는 60-70년대에 유행하였고 과학자와 엔지니어들 사이에서 인기있었다.

와이블 분포는 아래와 같은 특징이 있다.

- 연속형, 이산형 둘다 사용가능
- 다양한 표현 가능. 두 모수를 조정하면 다양한 형태를 가질 수 있다.
- PDF, CDF, PMF, 기대값, 중위수, 최빈값, Quantile 함수(CDF의 역함수)에 대하여 닫혀있다.
- 정규분포처럼 자연현상에 많이 등장하기에 다양한 종류의 예측에 많이 사용된다.

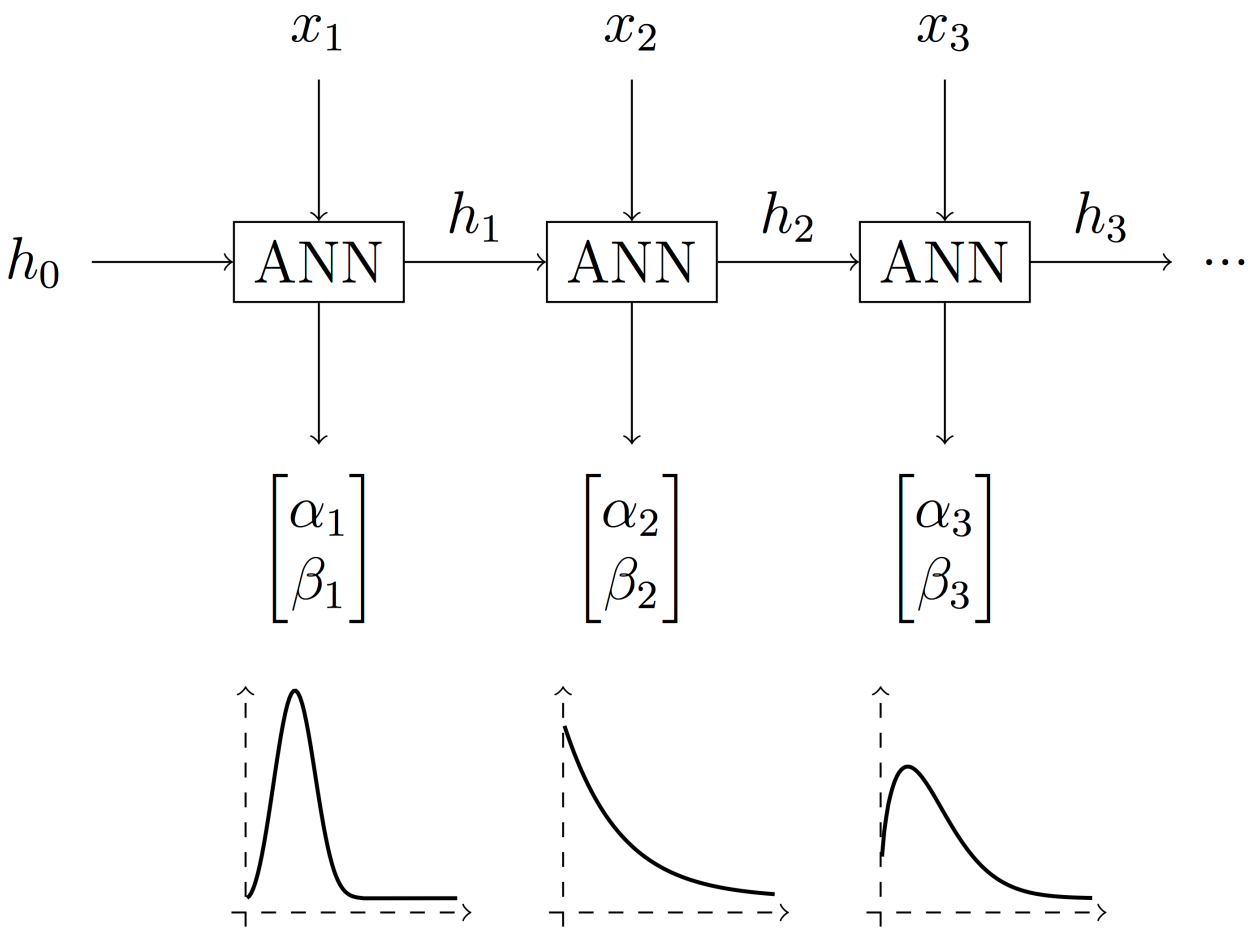
- Weakest link property : If a system breaks with the failure of any of its independent identical components then the time to failure is approximately Weibull distributed.
- 정규화 메커니즘에 의해 만들어졌다. $\beta$의 크기를 조절하면 뾰족한 정도를 조절할 수 있다.

지수분포와 이산 기하분포는 $\beta = 1$인 경우이다. $\beta = 2$인 경우는 레일리분포이다.

## 5. 구현&실험

2차원의 output layer를 가진 RNN 모형. 활성화 함수의 경우 output $\beta$에 대해서는 SoftPlus 를 추천하고 output $\alpha$에 대해서는 지수분포를 추천하다.

**softplus** : $f(x) = ln(1 + e^x)$




### churn-model에 WTTE-RNN 적용하기

churn 예측에 대해서 필자는 non event 사건이 발생하지 않는 것에 중점을 두라고 주장하였다. 사건발생에 집중하는 것 대신에 우리는 사건이 발생하지 않는 구간(사건발생 사이의 시간)에 집중하는 것이 좋다. 실시간으로 사건이 발생한 이후의 시간만 알기 때문에 단지 사건 발생까지의 시간만 예측하면 된다.

| **Ground truth world** | **actual churners** | **customer with TTE $y_t > \tau$ is churned** |
|---|---|---|
| Prediction world | predicted churners | $Pr(Y_t \leq \tau) < \theta^*$ the customer is predicted as churned |
| training world | observed churners | If $y_t < \tau$ customer was active. If $\hat{y}_t \geq \tau$ customer was churned |

## 6. 요약

WTTE-RNN은

- 이산형, 연속형 자료에 모두 적용가능하다
- censored된 (절단된) 데이터 학습이 가능하다
- 일시적 feature와 시간에 따라 변화하는 공변량을 이용할 수 있다.
- 장기간의 일시적 패턴을 학습할 수 있다.

임의적인 window size 조절이 필요 없기에 (아마 $\tau$ 길이 설정) sliding box모델 보다 더욱 신뢰할 수 있다.  이용가능한 모든 data를 사용하고 해석가능한 결과를 얻을 수 있다.

또한 현실이 아니라 모형이기에 몇 가지 가정이 필요하다. 하지만 명쾌한 가정이고 애매한 것이 아니다.

- 주어진 feature에 대해서 다음 사건 발생까지의 시간은 Weibull 분포를 따른다.
- Assume uninformative censoring -- censoring에 대한 충분한 정보는 없다?