정보이론: 정보량 (Information), 엔트로피 (Entropy), 쿨백 라이블러 발산 (KL-Divergence), 크로스 엔트로피 (Cross - Entropy), maximum likelihood

참고!

entropy : http://jaejunyoo.blogspot.com/2018/02/minimizing-negative-log-likelihood-in-kor.html
KL-Divergence : https://brunch.co.kr/@chris-song/69
KL-Divergence & cross-entropy: https://ratsgo.github.io/statistics/2017/09/22/information/
Binary Cross Entropy: https://curt-park.github.io/2018-09-19/loss-cross-entropy/
cross entropy: https://sevity.tistory.com/41

정보량 (Information)

- 정보량은 degree of surprise, 즉 '놀람의 정도' 로 정의됨
- 정보이론의 핵심 아이디어는 잘 일어나지 않는 사건(unlikely event)은 자주 발생하는 사건보다 정보량이 많다(informative)는 것
- 확률의 역수로 표현된다.
  - $$\frac{1}{p(x)}$$
- 두 사건이 동시에 발생할 확률은 곱해준다
  - $$\frac{1}{p(x)} \times \frac{1}{p(y)}$$
  - 로그를 씌우면 곱셈을 덧셈으로 바꿀 수 있다.
  - 최종정보량 information = $\log \frac{1}{p(x)} + \log \frac{1}{p(y)} = -\sum \log p(i)$

정보량을 공식화한 표현은 다음과 같다.

- 자주 발생하는 사건은 낮은 정보량을 가진다. 발생이 보장된 사건은 그 내용에 상관없이 전혀 정보가 없다는 걸 뜻한다.
- 덜 자주 발생하는 사건은 더 높은 정보량을 가진다.
- 독립사건(independent event)은 추가적인 정보량(additive information)을 가진다. 예컨대 동전을 던져 앞면이 두번 나오는 사건에 대한 정보량은 동전을 던져 앞면이 한번 나오는 정보량의 두 배이다.

위 세가지 조건을 만족하는 함수

$$I(x) = -\log P(x)$$

엔트로피 (Entropy)

- 주어진 결과에 도달할 수 있는 방법의 가짓수를 정량화 한다.

8명의 친구들이 두 대의 택시를 나눠타고 브로드웨이 쇼를 보러 가는 것을 상상해보죠. 다음과 같이 두 가지의 시나리오를 생각해보겠습니다:

- 네 명씩 택시를 탄다:

```
# fill the first, then the second
assignment_1 = [1, 1, 1, 1, 2, 2, 2, 2]

# alternate assignments
assignment_2 = [1, 2, 1, 2, 1, 2, 1, 2]

# alternate assignments in batches of two
assignment_3 = [1, 1, 2, 2, 1, 1, 2, 2]

# etc.
```

- 모든 친구들이 하나의 택시에 어떻게든 우겨 탄다:

```
assignment_1 = [1, 1, 1, 1, 1, 1, 1, 1]
```

두 번째 경우보다 첫 번째 경우가 가능한 경우의 수가 많기 때문에 **첫 번째 결과(outcome)가 더 높은 엔트로피 값을 갖게 됩**니다.

엔토로피 계산 식

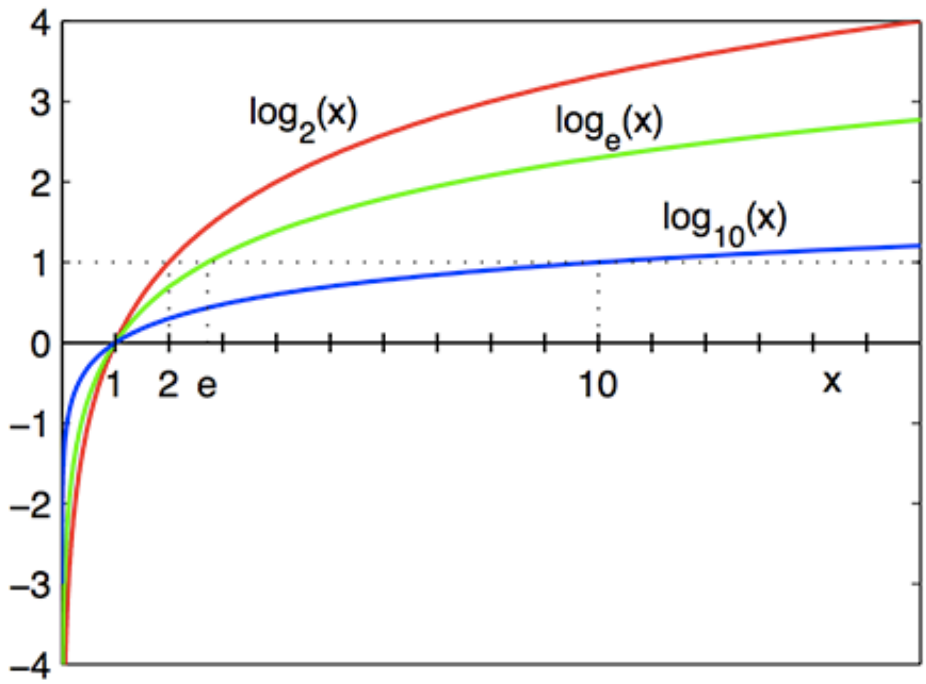
$$H(p) = -\sum_{i=1}^{n} p_i \log p_i$$

=> 개별 확률변수에 대한 정보량을 구해 합하는것
=> 정보량이 클수록, 가능성이 낮은 결과

식 해석

위 식은 다음 전재를 갖게 된다.

```
* p(i) = i 에 대한 확률.
* i = 확률 변수 집합 (벡터의 feature)
* p 는 확률밀도함수이고, p(i) 는 i 변수의 확률이다.
    * 확률밀도함수:
https://www.evernote.com/l/AGLHOdl0w9hDbYMENC1jSshBfPPbvbZe4Yg
* 그렇기 때문에 p(i) + ... p(n) 의 합은 1 이다.
```

- 수식의 log 덕에 P(i) 가 1인 경우가 존재하면, 엔트로피는 0 이다. (log 1 은 0이고)
- p(i)의 확률 값이 적을 수록 엔트로피는 커진다.
    - 

    - 초록색 선을 보면, x 가 적을 수록 y 가 작아진다. (음수)

    - p(i)~ p(n) 의 전체 합에 역(음수)를 붙여 값이 적을 수록, 큰 수로 변환된다.

따라서 엔트로피값이 클 수록 더 많은 경우의 수를 갖게 된다.

- p(i)~p(n) 의 확률이 비등비등 하면 더큰 엔트로피값을 갖게 된다.
- p(i) 가 1 인경우 다른 p(n-1) 은 모두 0으로, 작은 엔트로피 값을 갖게된다.

엔트로피는 가능한 이벤트들에 대한 **가중치 평균 로그 확률(weighted-average log probability)** 이고,
확률 분포에 내재한 불확실성을 측정하는 방법이라고 함.

- **변수들의 확률이 비등비등하면 (엔트로피 값이 높고), 어떤 결과가 나올지 알수 없어진다. (불확실성이 높아진다)**

즉,
어떤 이벤트에 대해 엔트로피가 높다는 것은 해당 결과값을 얻을 것이라는 믿음에 대한 확실성이 덜 하다는 것을 뜻한다.
<->
변수들의 확률이 비등비등하면 (엔트로피 값이 높고), 어떤 결과가 나올지 알수 없어진다. (불확실성이 높아진다)

엔트로피를 계산해보자

```
[5]: from math import log

p = {'rain': .14, 'snow': .37, 'sleet': .03, 'hail': .46}

def entropy(prob_dist):
    return -sum([ p * log(p) for p in prob_dist.values() ])

print(entropy(p))
```

```
1.1055291211185652
```

두개의 확률 분포값으로 엔트로피를 비교해보자

```python
p_2 = {'rain': .01, 'snow': .37, 'sleet': .03, 'hail': .59}

p_3 = {'rain': .01, 'snow': .01, 'sleet': .03, 'hail': .95}

p_2_hat = entropy(p_2)
p_3_hat = entropy(p_3)

print('p_2_hat: {}'.format(p_2_hat))
print('p_3_hat: {}'.format(p_3_hat))
```

```
p_2_hat: 0.8304250977453105
p_3_hat: 0.24602877030753434
```

엔트로피 계산 결과

- p_2 가 엔트로피가 더 크다
- p_2 가 불확실성이 높다. (확실성이 덜 하다)
- **p_3 가 확실성이 높다.**

기울기하강 알고리즘에서 크로스 엔트로피를 손실함수로 사용하는 이유가 무엇일까?
유추해볼수 있는 대목이다.

잠깐, 크로스 엔트로피 공식 먼저 한번 보고..

크로스 엔트로피

$$H_{p,q}(X) = -\sum_{i=1}^{N} p(x_i) \log q(x_i)$$

엔트로피

$$H(p) = -\sum_{i=1}^{n} p_i \log p_i$$

크로스 엔트로피와 엔트로피 식이 유사하지만 마지막 q 와 p 만 다르다.
엔트로피는 자신의 확률변수와 확률 분포를 곱하지만,
크로스 엔트로피는 자신의 확률변수와 비교될 확률분포와 곱하는 것으로
엔트로피는 엔트로피이되 두 확률분포가 교차로 곱해진다는 의미로 크로스(cross) 엔트로피라는 이름이 붙은 것 같습니다.
라고..

여기서 크로스 엔트로피로 더 다가기 위해 필요한 중간 과정

쿨백 라이블러 발산 (KL-Divergence)

- 두 확률분포의 차이를 계산하는데 사용하는 함수.
- 두 확률간 상대적인 entropy를 뜻하며, cost function이 될 수 있다.
- Q(x) 는 예측, P(x) 는 실제 확률이라고 하면, 이 값은 아래처럼 계산된다.
- KL-divergence = 상대적인 엔트로피

내맘대로 정리를 해보자면
학습을 통해서 랜덤변수 x 일 때 y 일 예측한 정보량의 확률 분포와 (아래 식에서 P, 엔트로피)
정답셋에서 랜덤변수 x 일 때 y 일 정보량의 확률 분포 (아래 식에서 Q, 엔트로피)의 차이를 계산하기 위한 함수
즉, loss function !

KL-Divergence 공식

$$D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)}$$

KL-Divergence 공식

위 식을 전개해보면 무릎을 탁 칠걸?

로그함수의 성질을 가져와서 전개하자.

로그의 성질

a > 0, a ≠ 1, M > 0, N > 0, L > 0, k가 실수일 때

- $\log_a 1 = 0,\ \log_a a = 1$
- $\log_a MN = \log_a M + \log_a N$
- $\log_a \frac{M}{N} = \log_a M - \log_a N$
- $\log_a L^k = k \log_a L$

위에 나누기를 빼기로 바꾸는 성질만 알고 있으면, 어렵지 않은 전개..

$$\begin{aligned} D_{KL}(P||Q) &= -\sum_x P(x)\log\left(\frac{Q(x)}{P(x)}\right) \\ &= -\sum_x P(x)\{\log Q(x) - \log P(x)\} \\ &= -\sum_x \{P(x)\log Q(x) - P(x)\log P(x)\} \\ &= \boxed{-\sum_x P(x)\log Q(x)} + \boxed{\sum_x P(x)\log P(x)} \\ &= H(P,Q) - H(P) \end{aligned}$$

- 빨간색 박스 영역이 크로스 엔트로피 식과 동일하다.
- 위 식을 크로스 엔트로피를 기준으로 다시 정리하면 다음과 같다.

$$H(P,Q) = H(P) + D_{KL}(P||Q)$$

-
- 고로 **크로스 엔트로피**는 DKL 그러니까, **두 확률분포의 차이를 계산**하는 것이다.

그리고 크로스 엔트로피 (Cross Entropy)

Cross Entropy

Cross Entropy는 두 개의 확률분포 p와 q에 대해 하나의 사건 X가 갖는 정보량으로 정의된다. 즉, 서로 다른 두 확률분포에 대해 같은 사건이 가지는 정보량을 계산한 것이다. 이는 q에 대한 정보량을 p에 대해서 평균낸 것으로 볼 수 있다 [4].

Cross Entropy:

> $$H_{p,q}(X) = -\sum_{i=1}^{N} p(x_i) \log q(x_i)$$

이를 크로스 엔트로피를 기준으로 다시 정리하면 아래와 같습니다.

$$H(P,Q) = H(P) + D_{KL}(P||Q)$$

위 식을 보면 KLD는 딥러닝 모델의 손실함수(loss function)로 자주 쓰이는 크로스 엔트로피와 깊은 관련을 맺고 있다는 걸 알 수 있습니다. 딥러닝 모델을 학습할 때 크로스 엔트로피를 최소화하는 방향으로 파라메터(가중치)들을 업데이트 합니다. 위 식에서 $P(x)$를 우리가 가지고 있는 데이터의 분포, $Q(x)$를 모델이 추정한 데이터의 분포라고 본다면, 크로스 엔트로피 $H(P,Q)$를 최소화한다는 것은 KLD를 최소화하는 것과 그 의미가 완전히 같습니다(equivalent). 왜냐하면 데이터 분포 $P(x)$는 학습 과정에서 바뀌지 않기 때문입니다. 결과적으로 크로스 엔트로피 최소화는 우리가 가지고 있는 데이터의 분포 $P(x)$와 모델이 추정한 데이터의 분포 $Q(x)$ 간에 차이를 최소화한다는 정도로 이해하면 좋을 것 같습니다.

Gradient decent 알고리즘에서 편미분으로 기울기를 구하는데, 이때 상수는 취급하지 않고(고정하고) 계산에서 지워 버리는데,
P(x) 는 이미 알고있는 학습 데이터의 확률분포이고, 변하지 않는 상수이니, 제거 가능하고
그러면 KLD 와 CrossEntropy 는 완전히 같다는 의미로 이해하기로 했음.. 음.. 그냥 그렇게 생각하기로 했음. (위에 파란색 박스부분을 말하는 것)

결국 Cross Entropy 란,

- **두 가능성의 확률분포의 차이를 계산하는데 사용하는 함수.**

어? 딥러닝에서 사용한 크로스 엔트로피 식과 다른데???

Cross entropy는 기계학습에서 손실함수(loss function)을 정의하는데 사용되곤 한다. 이때, p는 true probability로써 true label에 대한 분포를, q는 현재 예측모델의 추정값에 대한 분포를 나타낸다 [13].

Binary cross entropy는 두 개의 class 중 하나를 예측하는 task에 대한 cross entropy의 special case이다.

Binary Classification Task에서의 Cross Entropy:

$$
\begin{aligned}
H_{p,q}(Y|X) &= -\sum_{i=1}^{N} \sum_{y\in\{0,1\}} p(y_i \mid x_i) \log q(y_i \mid x_i) \\
&= -\sum_{i=1}^{N} \left[p(y_i = 1 \mid x_i) \log q(y_i = 1 \mid x_i) + p(y_i = 0 \mid x_i) \log q(y_i = 0 \mid x_i) \right] \\
&= -\sum_{i=1}^{N} \left[p(y_i = 1 \mid x_i) \log q(y_i = 1 \mid x_i) + \{1 - p(y_i = 1 \mid x_i)\} \log\{1 - q(y_i = 1 \mid x_i)\} \right] \\
&= -\sum_{i=1}^{N} \left[p(y_i) \log q(y_i) + \{1 - p(y_i)\} \log\{1 - q(y_i)\} \right]
\end{aligned}
$$

자 그럼 Maximum likelihood 에 대해 떠오르는 생각

Cross Entropy 를 낮춘다.
= 상대적 엔트로피를 낮춘다.
= 학습한 데이터의 분포와, 예측 데이터의 분포의 차이를 낮춘다.

엔트로피를 낮춘다
= 정보량이 적어진다
= 가능성이 높아진다. (브라질과 중국이 축구를 하면 브라질이 100퍼 이기지)
= 가능성을 높인다?
= Maximum likelihood (가능도) 를 높인다.

어떻게?
기울기하강 알고리즘을 사용해서!