로지스틱 회귀

	linear classification	linear regression	logistic regression
$\mathcal{Y}$	$\{-1,+1\}$	$\mathbb{R}$	$\{-1,+1\}$
$\hat{y}=h(\mathbf{x})$	$\text{sign}(\mathbf{w}^\top\mathbf{x})$	$\mathbf{w}^\top\mathbf{x}$	$\theta^*(\mathbf{w}^\top\mathbf{x})$
$\mathrm{e}(\hat{y},y)$	0-1 loss $[\![\hat{y}\neq y]\!]$	squared error $(\hat{y}-y)^2$	cross-entropy error $[\![y=+1]\!]\ln\frac{1}{\hat{y}}+[\![y=-1]\!]\ln\frac{1}{1-\hat{y}}$
$\mathrm{E}_{\text{in}}(h)$	$\frac{1}{N}\sum_{n=1}^N[\![h(\mathbf{x}_n)\neq y_n]\!]$	$\frac{1}{N}\sum_{n=1}^N(h(\mathbf{x}_n)-y_n)^2$	$\frac{1}{N}\sum_{n=1}^N\ln\left(1+e^{-y_n\mathbf{w}^\top\mathbf{x}_n}\right)$
opt.	combinatorial optimization (NP-hard)	set $\nabla\mathrm{E}_{\text{in}}(\mathbf{w})=0$ (closed-form solution exists)	set $\nabla\mathrm{E}_{\text{in}}(\mathbf{w})=0$ iterative optimization (*e.g.* gradient descent)

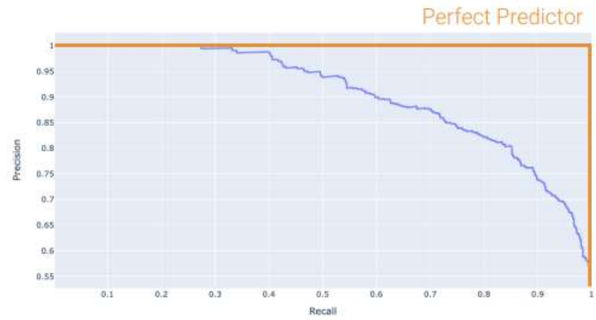
⋆ logistic sigmoid $\theta(s)=1/(1+e^{-s})$

- $1-\theta(s)=\theta(-s)$, 즉 $1-P(+1|x)=P(-1|x)$
- 실제 확률 $f(\mathrm{x})=\mathbb{P}[y=+1|\mathrm{x}]$이지만 이 값은 알 수 없음
- MSE에 기반하여, $E_{\text{in}}(h)=\frac{1}{N}\sum_{n=1}^{N}[h(\mathrm{x}_n)-\frac{1}{2}(1+y_n)]^2$
- 대부분의 영역에서 기울기가 0이라 최소화하기 매우 어려움
- 크로스 엔트로피 오차가 필요함
- $E_{\text{in}}(h)=\frac{1}{N}\sum_{n=1}^{N}\ln(1+e^{-y_n\mathrm{w}^\top\mathrm{x}_n})$
- 업데이트 : $\mathrm{w}\leftarrow\mathrm{w}-\eta\nabla E_{\text{in}}(\mathrm{w})$
- 각 반복에서 O(N)의 시간이 걸림
- Stochastic Gradient Descent로 O(1)로 할 수 있음
- 초기 가중치 w(0)는 평균이 0, 작은 분산을 가지는 정규분포에서 독립적으로 추출
- 최종 가중치를 정하는 방법
1. 반복횟수의 Upper Bound에서 멈추기 : 좋은 값이라는 보장 X
2. 기울기가 작은 Lower Bound : 오차가 크지만 기울기가 낮은 구간
- 따라서 반복횟수, 오차, 기울기 모두 사용하는 것이 바람직
- 경사하강법의 종류
1. Batch(일괄) : 모든 값을 사용
2. 무작위적(Stochastic) : 하나의 값만을 사용
3. 소형 일괄(Mini-Batch) : $2\leq b\leq 100$의 값 사용

분류기 평가

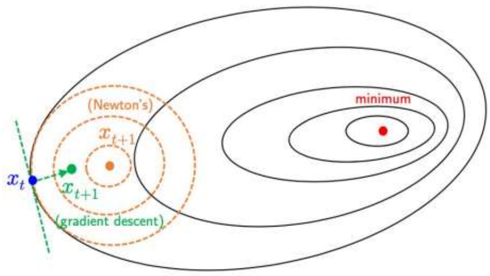
		Prediction	
		0	1
Actual	0	True negative (TN)	False positive (FP)
	1	False negative (FN)	True positive (TP)

- Accuracy : $\frac{TP+TN}{n}$
- Precision : $\frac{TP}{TP+FP}$ False Positive에 초점
- Recall(TPR, Sensitivity) : $\frac{TP}{TP+FN}$ False Negative에 초점
- FPR : $\frac{FP}{FP+TN}$
- 낮은 T : 많은 FP, 적은 FN, Recall 커짐
- 높은 T : 많은 FN, 적은 FP, Precision 커짐
- Precision-Recall Curve
- 좋은 분류기는 AUC = 1, 무작위 분류기는 AUC = .5
- Threshold는 좌측이 높고 우측이 낮음



피처 엔지니어링

- 단점 : 중복된 피처, 지나치게 많은 피처, 과적합
- Constant Feature Function : 모든 값이 1인 열을 x에 추가
- UID와 같이 정보가 없는 피처 제거
- 양적 변수 : 대한 비선형 변환, 정규화, 표준화
- 카테고리형 변수 : 원 핫 인코딩
- Null : 보간, is_null 열 만들기, 카테고리 만들기
- 도메인 지식을 활용한 피쳐 변환
- Bag of Words : 단어의 빈도 딕셔너리
- 단점 : 희소한 벡터, 단어 순서 소실, 없는 단어 처리 불가
- N-gram : N개의 연속된 단어를 하나로 처리
- 단점 : 더 희소한 벡터($O(m^2)$), 없는 덩어리가 있을 확률 높음



First Order 최적화(Gradient Descent, Jacobian)

- $x\leftarrow x-\epsilon f'(x)$
- Critical Points(Stationary -) : $\nabla_x f(x)=0$
- Local Min/Max와 안장점(Saddle Point)
- 고차원 딥러닝에서 안장점이 최대/최소보다 많음
- 데이터가 매우 많으면 기울기 계산은 오래 걸림
- SGD : m개의 x를 임의로 뽑아서 계산
- 장점 : N이 늘어나도 계산 시간이 고정적, 이론보다 잘 작동함
- 단점 : 지역적 최소나 안장점에 갇힘, Gradient Noise, 지그재그

• gradient estimate

$$\hat{\boldsymbol{g}}=\frac{1}{m}\nabla_{\boldsymbol{\theta}}\sum_i L(f(\boldsymbol{x}^{(i)};\boldsymbol{\theta}),\boldsymbol{y}^{(i)})$$

• stochastic gradient descent (sgd)

$$\boldsymbol{\theta}\leftarrow\boldsymbol{\theta}-\epsilon_k\hat{\boldsymbol{g}}$$

• method of momentum ($\alpha\in[0,1)$)

$$\boldsymbol{v}\leftarrow\alpha\boldsymbol{v}-\epsilon\hat{\boldsymbol{g}}$$
$$\boldsymbol{\theta}\leftarrow\boldsymbol{\theta}+\boldsymbol{v}$$

• Nesterov momentum (corrected momentum)

$$\boldsymbol{v}\leftarrow\alpha\boldsymbol{v}-\epsilon\nabla_{\boldsymbol{\theta}}\left[\frac{1}{m}\sum_{i=1}^{m}L\left(f(\boldsymbol{x}^{(i)};\boldsymbol{\theta}+\alpha\boldsymbol{v}),\boldsymbol{y}^{(i)}\right)\right]$$
$$\boldsymbol{\theta}\leftarrow\boldsymbol{\theta}+\boldsymbol{v}$$

• AdaGrad ($\boldsymbol{r}$: for gradient accumulation)

$$\boldsymbol{r}\leftarrow\boldsymbol{r}+\hat{\boldsymbol{g}}\odot\hat{\boldsymbol{g}}$$
$$\Delta\boldsymbol{\theta}\leftarrow-\frac{\epsilon}{\sqrt{\delta+\boldsymbol{r}}}\odot\hat{\boldsymbol{g}}$$
$$\boldsymbol{\theta}\leftarrow\boldsymbol{\theta}+\Delta\boldsymbol{\theta}$$

• RMSProp (gradient accumulation by EWMA[1])

$$\boldsymbol{r}\leftarrow\rho\boldsymbol{r}+(1-\rho)\hat{\boldsymbol{g}}\odot\hat{\boldsymbol{g}}$$
$$\Delta\boldsymbol{\theta}\leftarrow-\frac{\epsilon}{\sqrt{\delta+\boldsymbol{r}}}\odot\hat{\boldsymbol{g}}$$
$$\boldsymbol{\theta}\leftarrow\boldsymbol{\theta}+\Delta\boldsymbol{\theta}$$

• Adam (a reasonable default choice)

$$\boldsymbol{s}\leftarrow\rho_1\boldsymbol{s}+(1-\rho_1)\hat{\boldsymbol{g}} \quad \text{(momentum)}$$
$$\boldsymbol{r}\leftarrow\rho_2\boldsymbol{r}+(1-\rho_2)\hat{\boldsymbol{g}}\odot\hat{\boldsymbol{g}} \quad \text{(RMSProp)}$$
$$\hat{\boldsymbol{s}}\leftarrow\frac{\boldsymbol{s}}{1-\rho_1^t},\quad \hat{\boldsymbol{r}}\leftarrow\frac{\boldsymbol{r}}{1-\rho_2^t} \quad \text{(bias correction)}$$
$$\boldsymbol{\theta}\leftarrow\boldsymbol{\theta}-\epsilon\frac{\hat{\boldsymbol{s}}}{\sqrt{\hat{\boldsymbol{r}}+\delta}}$$

EWMA(Exponentially Weighted Moving Average)

- $v_t\leftarrow\alpha v_{t-1}+(1-\alpha)g_t$, $v_1=g_1$
- 다음 스텝의 값을 결정하기 위해 기울기와 현재 스텝의 값을 모두 고려
- a가 큰 값일수록 이전 값 많이 반영, 부드러운 곡선

- $1+\alpha+\alpha^2+... = \frac{1}{1-\alpha}$개 최근 점의 가중평균
- Bias Correction : 초기에 최근 데이터가 부족하므로 편향이 있음
- $v_t \leftarrow \frac{v_t}{1-\alpha^t}$

모멘텀
- $\theta \leftarrow \theta - \epsilon(g+cv)$, g는 기울기, c는 상수, v는 속도(EWMA)
- Nesterov 모멘텀 : 이전 속도대로 간 후 지점의 기울기 반영(기존 모멘텀에 Correction Factor 적용)
- $\theta \leftarrow \theta + \alpha v - \epsilon \nabla_\theta J(\theta + \alpha v)$

AdaGrad(Per Parameter Adaptive Learning Rate)
- 차원(방향)마다 스텝 크기를 다르게 설정
- $\epsilon_j = \frac{\epsilon}{\sqrt{\sum_{all}(g_j \cdot g_j)}}$, gj는 차원 j에서의 기울기
- 스텝 크기가 단조감소함
- 학습이 지나치게 빨리 멈출 수 있음
- Adadelta : 모든 기울기 대신 δ개의 기울기 사용
- RMSProp : AdaGrad와 유사하나 기울기 대신 EWMA를 반영

Adam
- RMSProp + 모멘텀, 대부분의 경우 사용

0. s=0, r=0, t=0
1. g 계산 / 2. 모멘텀 계산 : $s \leftarrow \rho_1 s + (1-\rho_1)g$
3. RMSProp : $r \leftarrow \rho_2 r + (1-\rho_2) g \cdot g$
4. Bias Correction : $\hat{s} \leftarrow \frac{s}{1-\rho_1^t}$, $\hat{r} \leftarrow \frac{r}{1-\rho_2^t}$
5. $\theta \leftarrow \theta - \epsilon \frac{\hat{s}}{\sqrt{\hat{r}+\delta}}$

Second Order 최적화(Hessian, Newtons Method)
- $x \leftarrow x - \frac{f'(x)}{f''(x)}$, 2차미분은 $O(n^3)$의 복잡도
- 음수 곡률 : 기울기 기반 예측보다 빠르게 f가 감소
- 곡률 없음 : 기울기 기반 예측대로 f가 감소
- 양수 곡률 : 예측보다 느리게 f가 감소하다가 늘어남
- $\nabla f(x) = f'(x) + (x-x_0)f''(x_0) = 0$이 되는 지점 찾기
- $x = x_0 - \frac{f'(x_0)}{f''(x_0)} = x_0 - H^{-1}g$
- 하이퍼파라미터가 없으나, 비효율적임($O(n^2)$의 원소, $O(n^3)$ 시간)
- Quasi-Newton : H^{-1}대신 M_t행렬로 근사, $O(n^2)$ 복잡도
- BGFS : 행렬 저장에 $O(n^2)$ 복잡도
- L-BFGS : Full Batch, Deterministic한 경우에 잘 작동

정칙화 : 과적합에 강해지지만 과소적합에 약해짐

1. 파라미터에 제약 2. 목적 함수에 항 추가 3. 앙상블
- f가 H에 있지 않은 경우 편향, 과소적합 발생
- f가 H에 있지만 H가 너무 큰 경우 분산, 과적합 발생
- $E_{aug} = E_{\text{in}}(H) + \Omega(h)$, $E_{aug}(w) = E_{\text{in}}(w) + \frac{\lambda}{N}\Omega(w)$
- 람다는 정칙화 파라미터, 오메가는 정칙화기
- Norm Penalty : $L_{aug} = L_{train} + \Omega(w)$
- L1 : $\sum_q |w_q|$, Variable Shrinking, Feature Selection -> Sparse
- L2 : $\sum_q w_q^2$, Variable Shrinking, Weight Decay
- Early Stopping : E_{val}을 계산하여 내려가지 않으면 일찍 멈춤
- 최소값을 찾는 데에 장애물을 만들지 않고 반복이 줄어듦
- E_{val}을 계산하는 것의 손실이 발생
- 앙상블 : 서로 다른 모델의 오류 평균이 0일 것이라는 가정
- Voting : (가중) 투표
- Bagging : x를 여러 표본으로 추출, 분산은 감소, 편향은 그대로
- Boosting : 이전 단계에서 실패한 데이터의 가중치를 상승시키고 다시 표집, 분산과 편향 모두 감소

결정트리
- 가장 간단한 결정 트리를 찾는 과정은 NP-Complete
- 각 클래스가 완전히 분리되거나(Pure), 더 나눌 수 없을(Unsplittable) 때
- 최선의 피처 x와 값 β를 찾은 후, 데이터를 $x<\beta, x \geq \beta$로 나눔
- 엔트로피 : $-\sum_c p_c \lg p_c$, $p_c = \frac{N_c}{\sum_c N_c}$, $L = \frac{N_1 S_1 + N_2 S_2}{N_1 + N_2}$
- 과적합 : 결정트리의 완전한 분리는 과적합 문제를 유발함
- 특별 규칙 : 데이터의 개수가 일정 이하, 트리의 깊이가 일정 이상
- 프루닝 : 검증 세트에 대해 오차가 변하지 않는 부분 트리 제거
- 랜덤 포레스트 : 여러 트리의 Bagging, 피처를 $\sqrt{p}$개만 사용

군집화
- K-Means : 소속이 바뀌는 점이 없을 때까지 반복
- 직관에서 벗어나는 군집이 발생 가능, 시작점이 중요
- Inertia : 모든 군집의 군집 내 거리 제곱의 합 $\sum d^2$
- Inertia를 최적화하는 알고리즘은 NP-Hard
- 또한 Inertia는 Blobiness를 고려하지 않음
- Distortion : 모든 군집의 군집 내 거리 정규화의 합, $\sum \frac{\sum d}{l}$
- Agglomerative Clustering : 원하는 k를 얻을 때까지 가장 가까운 클러스터를 합치는 것을 반복

k 결정
- Elbow Method : k에 따른 Inertia의 급격한 감소를 통해 결정
- 실루엣 스코어 : S가 높을수록 잘 군집화되어 있음
- 평균 실루엣 스코어가 높을수록 적절한 k값
- A는 군집 내 다른 점과의 거리 평균
- B는 가장 가까운 군집 내 다른 점과의 거리 평균
- $S = \frac{B-A}{\max(A,B)}$
- A=0이면 S=1이고, 이때 군집 내 모든 점이 같은 위치임
- A>B면 S<0이 될 수도 있음

빅데이터
- 기존이 도구로 빠르게 분석, 처리하기 어려운 크기의 데이터
- 비구조화, 반구조화, 구조화된 데이터
- ETL : 떨어진 원천에서 데이터 추출, 기준 스키마로 변환, DB에 적재
- 데이터레이크 : 비구조화된 데이터 그대로의 모음
- 단점 : 노이즈가 많음, 데이터 거버넌스와 계획이 잘 되지 않음, Dirty Data가 많음, 이전의 도구와 호환되지 않음
- 장점 : 새로운 스킬들이 등장, 기술력의 발전, 조직의 개선
- 데이터베이스의 속성 : 현재를 저장함, 조직 내에 다양한 데이터베이스가 존재, 데이터 포맷이 상이함, 히스토리를 저장하지 않기도 함
- Star Scheme : 테이블에 중복된 데이터가 저장되어 있으면 여러 테이블로 분할할 수 있음
- 빅데이터 하드웨어 : 큰 용량을 위해 저렴하지만 실패가 잦은 하드웨어 사용
- 분산 파일 시스템 : 실패에 관대한 파일 시스템, 파일을 분할하여 여러 기기에 저장
- 분산 컴퓨팅(Map Reduce)
- Map : 데이터를 분할하여 각 자원에 할당하고 계산
- Deterministic Map : 실패 시 해당 부분만 재계산이 가능해야 함
- Reduce : 완성한 데이터를 다시 합침
- Commutative Reduce : $R(a,b) = R(b,a)$
- Associative Reduce : $R(R(a,b),c) = R(a,R(b,c))$
- 부동소수점 연산은 Associativity가 보장되지 않음