R E P O R T

머신러닝 중간고사

학 과 머신러닝 02분반

교수님 전동산

학 과 컴퓨터공학과

학 번 2254747

이 름 유우림

동아대학교
DONG-A UNIVERSITY

1주차_2차시

미분: Derivative
상수: constant
변수: Variable
계수: Coefficient

1. 미분

1.1 미분개요: Derivative

- 다항식, 삼각, 지수, 로그

- **미분**: gradient(**기울기, 변화량**)

$$y = cos(x - \frac{\pi}{2}) = sin(x) = 0$$

길이가 1짜리인 막대기의 그림자 길이를 생각하면 됨

- cos(x): cos(0)의 값은 막대기가 x축에 누워있음
- sin(x): sin(0)의 값은 막대기가 -y축의 방향으로 서있음
  - cf) 각도를 빼면 막대기는 역시계방향으로 돌아감(그래프로 그려 보면 됨)

$$\frac{d}{dx}sin(x) = cos(x), \frac{d}{dx}cos(x) = -sin(x), \frac{d}{dx}tan(x) = sec^2(x)$$

- **미분 형태**

$\frac{d}{dx}lnx = \frac{1}{x}$, $\frac{d}{de^x}e^x = e^x$, $\frac{d}{df(x)}e^{f(x)} = e^{f(x)}f(x)'$, $\frac{d}{df(x)}lnf(x) = \frac{f(x)'}{f(x)}$

1.2 편미분(Partial Derivative)

- 여러가지의 변수를 가지는 함수를 하나의 변수에 대해서만 미분

함수Z를 변수x에 대해 편미분 Notation → $\nabla_x Z$

Chain rule

- 여러 함수를 거친 함수의 변화량을 알아보는 방법

2주차_2차시

Linear Algebra: 벡터/행렬

1. Scalar

실수: Real Num

허수: Imagnary Num

복소수: Complex Num

Vector: Scalar들의 집합

Matrix: 행렬

- 행: row
- 열: colum

2. Vector : (m×1)

3. Matrix: (m×n)

4. Tensor: (m×n×k)

5. Unit Vector

- ||x||
- 벡터의 크기: Norm
  - L1 norm = $\sum |x_i| = ||x||_1$
  - L2 norm = $\sqrt{\sum x_i^2} = ||x||_2$, $||x||_2{}^2 = \sum x_i^2$

- Unit Vector: $U_x = \frac{x}{||x||_2}$

6. Bases Vector

- Vector Space: Basis Vector들의 공간
- 선형 독립(Linear Independent)
- 선형 조합(Linear Combination, Weighted Sum) = Span

💡 만약 Span 공간이 close하고 Linear Independent하다면 → Vector Space를 형성한다.

- Basis Vector를 Rank라고 부름

> 💡 Full Rank를 되지 않는다면, Inverse Matrix를 가지지 않음

- orthogonal : Linear Independent한 Basis Vector끼리의 각도가 90도일 경우

> 💡 orthogonal 할 경우, Linear Independent함. (Linear Independent하다고 해서, 꼭 orthogonal 하지는 않음)

- orthonarmal : Linear Independent한 Basis Vector의 길이가 1이면서, Basis Vector끼리의 각도가 90도일 경우

정리:

Basis Vector는 Linear Independent해야 되고, Span을 Close해서 Vector Space를 만들 수 있어야 한다.

- Basis Vector가 직각을 이루면 → othogonal
- Basis Vector의 크기가 다 1이고 + 직각을 이루면 → orthonormal

Span

- Basis Vector를 Weighted Sum, Linear Combination하는 과정
- 다른 말로, Vector Space를 만드는 과정

→ **중요한건 Vector Space를 만들어야 된다는 것이다**

3주차_2차시

review

Basis Vector — span —> Vector Space

Basis Vector = $(x_1 ... x_n)$

span = $span(x_1 ... x_n) = [w_1 x_1 + ... + w_n x_n]$

Linear Combination = $w_1 x_1, w_n x_n$

- Vector Space = Colum Space
- 벡터의 크기: Norm
  - L1 norm = $\sum |x_i| = ||x||_1$
  - L2 norm = $\sqrt{\sum x_i^2} = ||x||_2$, $||x||_2^{\ 2} = \sum x_i^2$

- 1차 방정식: First-order equation(Linear Eqn)
- 2차 방정식: Second-order eqution(Quadratic Eqn)
- 3차 방정식: Cubic Eqn
- 다항식: Polynomial Equation

행렬

$$Z = X, Y \text{행렬곱} = (m \times n)(n \times k) = Z \in R^{m \times k}$$

- 항등행렬: Identity

Loss function: 실제값과 예측값간의 차이를 표현하는 함수

- 절댓값(MAE) : L1
- 제곱근(MSE): L2

4주차_1차시 - 실습

4주차_2차시

ML/DL

- 분류(Classificatio)
- 추정(Inference) : 값을 추론/추정

독립변수1개 : SLP

독립변수 여러개 : MLP

> **편미분**
>
> 2차 함수인 Loss function의 값이 최소가 되는 x값을 찾아야 함
>
> → 편미분: 변수가 여러개인 함수를 하나의 변수에 대해서만 미분하는 것 → 미분을 통해 기울기가 0인 x값이 곧, Loss function이 최소가 되는 값임(실제 값과 가장 차이가 적은 값)
>
> → 그렇게 여러 개의 변수의 최소 값을 하나씩 찾아감

편미분

$$L(\alpha,\beta)=\sum(y-\hat{y})^2=\sum(y-\alpha-\beta x)^2$$

$\hat{y}$ → $y=\alpha+\beta x$

1) $\frac{\partial L}{\partial \alpha}=-2\sum(y-\alpha-\beta x)=0$

$=\sum(y-\alpha-\beta x)=0$

$=\sum y-n\alpha-\beta\sum x=0$

$\therefore \alpha=\frac{1}{n}\sum y-\frac{1}{n}\beta\sum x$

$\alpha^*=\bar{y}-\beta\bar{x}$

2) $\frac{\partial L}{\partial \beta}=-2x\sum(y-\alpha-\beta x)=0$

$=\sum x(y-\alpha-\beta x)=0$ (cf) $\alpha^*=\bar{y}-\beta\bar{x}$

$=\sum x(y-\bar{y}+\beta\bar{x}-\beta x)$

$=\sum x(y-\bar{y})-\beta\sum x(x-\bar{x})=0$

$\therefore \beta^*=\frac{\sum x(y-\bar{y})}{\sum x(x-\bar{x})}$

(+) 참고로:

$\beta^*=\frac{\sum(x-\bar{x})(y-\bar{y})}{\sum(x-\bar{x})^2}=\frac{\sum x(y-\bar{y})-\sum\bar{x}(y-\bar{y})}{\sum x(x-\bar{x})-\sum\bar{x}(x-\bar{x})}=\frac{\sum x(y-\bar{y})}{\sum x(x-\bar{x})}$

> 편차의 합은 항상 0이기 때문에 분모, 분자의 $\sum(x-\bar{x})$는 날라감

벡터의 편미분

1) $\frac{\partial(Y^T \cdot AX)}{\partial X} = Y^T \cdot A$

2) $\frac{\partial(Y^T \cdot AX)}{\partial Y} = (AX)^T = X^T A^T$

3) $\frac{\partial(X^T \cdot AX)}{\partial X} = (AX)^T + X^T A$ cf) $f \cdot g \rightarrow f'g + fg'$

$= X^T A^T + X^T A$

$= X^T [A + A^T]$

▼ **행렬 θ 유도 과정**

▼ **사진**

$$X\theta = Y$$

$$(X^T \cdot X)\theta = X^T \cdot Y \qquad \theta = X^{-1} \cdot Y \quad \rightarrow X\text{가 정방행렬이 아니기 때문에 불가능}$$

$$\cancel{(X^T \cdot X)^{-1} \cdot (X^T \cdot X)}\theta = (X^T \cdot X)^{-1} \cdot (X^T \cdot Y)$$

목표는 선형 회귀의 파라미터 θ를 구하는 식, 즉 $\theta = (X^T X)^{-1} X^T y$를 유도하는 것입니다. 이는 최소제곱법을 사용하여 잔차(residual)의 제곱합을 최소화하는 θ를 찾는 과정입니다.

1. 문제 정의

우리는 다음과 같은 모델을 가지고 있습니다:

여기서:

$$y = X\theta + \epsilon$$

- y는 관측된 출력 벡터입니다.
- X는 입력 데이터 행렬입니다.
- θ는 우리가 추정하려는 파라미터 벡터입니다.
- ϵ은 오차 벡터입니다.

2. 오차 제곱합 정의

잔차의 제곱합을 다음과 같이 정의합니다.

이 식을 최소화하는 θ를 찾는 것이 목표입니다.

$$S(\theta) = (y - X\theta)^T (y - X\theta)$$

3. 오차 제곱합을 전개하기

위의 식 $S(\theta)$를 전개하면 다음과 같이 됩니다.

$$S(\theta) = y^T y - 2\theta^T X^T y + \theta^T X^T X\theta$$

이 식은 $y^T y$, $-2\theta^T X^T y$, $\theta^T X^T X\theta$의 합으로 나타날 수 있습니다.

- $-y^T(X\theta) - (X\theta)^T y = -2\theta^T X^T y$

4. θ에 대해 편미분하기

이제 이 식을 θ에 대해 미분하고, 최소값을 찾기 위해 0으로 설정합니다.

1. 첫 번째 항 $y^T y$는 θ와 독립적이므로 미분하면 0입니다.
2. 두 번째 항 $-2\theta^T X^T y$를 θ에 대해 미분하면 $-2X^T y$가 됩니다.
3. 세 번째 항 $\theta^T X^T X\theta$를 θ에 대해 미분하면 $2X^T X\theta$가 됩니다.

따라서,

$$\frac{\partial S(\theta)}{\partial \theta} = -2X^T y + 2X^T X\theta$$

이 결과를 0으로 설정하여 최소값을 찾습니다:

$$-2X^T y + 2X^T X\theta = 0$$

양변을 2로 나누어 단순화하면:

$$X^T X\theta = X^T y$$

5. 최종적으로 θ에 대한 해 구하기

이제 $X^T X$가 가역 행렬이라고 가정하고, 양변에 $(X^T X)^{-1}$를 곱하여 θ를 구합니다:

$$\theta = (X^T X)^{-1} X^T y$$

5주차_1차시

1. Introduction

MLP: Multi-Layer Perceptron

Loss Function

- MAE: Mean Absolute Error
- MSE: Mean Squaree Error

ML/DL

- 분류(Classificatio)
- 추정(Inference)

Optimization방법 통해 LossFunction을 줄이는 방향으로 traing 시켜야 된다.

Overfiting(과적합): training data에 너무 fit됐다는 뜻이다.
traning data와 test data는 비슷한 데이터로 골라야된다.

기호주의 인공지능(Symblism AI)

- 컴퓨터의 작동 방식에 맞추어 규칙 기반 (Rule-baased)으로 인공지능을 학습
- (1) 인간의 지식을 기호화 → (2) 기호 간 관계 정의 → (3) 연역적 추론

ex)
knowledge1: 사람은 죽는다.
knowledge2: 소크라테스는 사림이다.
Inference: 소크라테스는 죽는가?
Result: 죽는다.

Symblism AI의 한계

- 현실 세계의 모든 형상/규칙/개념을 기호화 하는 것은 불가능함
- Ex. 기호주의 기법으로 고양이를 판별하는 인공지능
  → 밝기, 자세, 품종, 크기 등 모든 자연 형상을 고려하지 못함

연결주의 인공지능(ConnectionAI)

- 인간이 학습하는 방법을 구현/모방해 신경망(Neural Network) 기반으로 인공지능을 학습
- 대표적인 알고리즘으로 perceptron이 존재함 (1958)

각 노드들을 고유의 bias각 가지고 있고, 노드와 노드간에 weight가 있음

$y = b + w_1x_1 + w_2x_2 + ...$

각 bias(편향)와 weight(가중치)값을 통해 출력 값 조정함

XOR 문제와 AI winter

- 기호주의 인공지능을 지지하던 M.Minsky는 perceptron의 한계를 증명한 책을 출판

  → Perceptron은 어떤 방법으로도 XOR 문제를 풀 수 없음
- 이후 연결주의 인공지능 연구는 관심을 받지 못함 → AI winter

> 💡 결정 경계: Decisoion Boudary

Mluti-Layer Perceptron(MLP, 1986)

- 여러 층의 perceptron으로 구성된 MLP와 오차역전파법이 개발됨
- XOR 문제를 해결함으로써 인공지능 연구가 다시 활발해지는 계기가 됨

> 💡 인공지능 역사의 흐름
> 1956: Dartmouth Conference
> 1957: Perceptron 발표
> 1969: XOR 문제
> ~ AI winter(AI 빙하기)
> 1986: Multi-Layer Perceptron, Backpropagation 발표
> 이후 Deep Neural Network

2. Applications

Regression(output: real number)

- 값을 추론하는 것

Supervised Learning(지도 학습)

- 입력데이터와 대응되는 정답 데이터를 함께 이용해 학습하는 기법
- 대표적인 응용 분야로 Regression(회귀)와 Classification(분류)이 존재함

Classification

1. Binary Classification
2. Multi-label Classification

Logic gate: 논리 회로

- 입력 값에 대해 논리 연산을 수행하여 하나의 출력 값을 얻는 전자회로
- 대표적인 논리 회리 회로로 AND, OR, NOT gate가 존재함

  NOT gate: 어떤 값의 반대 값

  NAND gate: 두 변수의 AND값의 반대 값

  NOR gate: 두 변수의 OR값의 반대 값

  XOR: 둘 중 하나만 1일 경우 1이 나옴 (0, 0 / 1, 1 이면 0이 됨)

  - NAND gate, OR gate, AND gate 서로 연결하여 XOR gate 구현가능

    s_1 : x_1 NAND x_2

    s_2 : x_1 OR x_2

    y : s_1 AND s_2

3. Multi-layer Perceptron (MLP)

- 여러 개의 층으로 구성된 perceptron 모델
- 입력층, 은닉층, 출력층으로 구성됨
- 각 입력(x)에 대응되는 weight(w), 1개의 노드에 입력되는 bias(b) 존재
- 가중합으로 얻어진 결과치에 활성화 함수(h()) 적용

💡 Network parameter
Hyper parameter: 사람이 지정

💡 각 노드끼리 완전히 연결되어 있다고 해서
Fully Connected layer = MLP

FC layer = MLP

💡 $Y = h(W^tX + b)$: 1-layer perceptron 계산
$Y = h(W_2^t h(W_1^t X + b_1) + b_2)$: 2-layer perceptron 계산

전 결합 계층(Fully Connected layer, FC layer)

- 각 층별로 모든 노드가 연결되어 weight를 가지는 계층을 FC layer라고 함

활성화 함수(Activated Function): h()

Softmax function: Normalization

- 여러 개의 입력을 받아 각각의 확률 값으로 출력(abs, square, 지수)
- 0~1 사이 실수를 출력하고, 모든 출력의 합은 1 (확률로 해석)
- Multi-label classification 모델의 출력층에 주로 사용됨
- sigmoid를 통해 0~1사이의 값으로 정규화할 수 있음

💡 Regularization(규제화)
- EX. weight를 너무 많이 주지마라(L1 R, L2 R)

Loss function

- 학습 모델이 얼마나 잘못 예측하고 있는지를 표현하는 지표
- 값이 낮을수록 모델이 정확하게 예측했다고 해석할 수 있음
  - MAE(Mean Absolute Error, 평균 절대 오차)
  - MSE(Mean Squared Error, 평균 제곱 오차)
  - CEE(Cross Entropy Error, 교차 엔트로피 오차)

  MAE, MSE는 이미지 처리를 할 때, CEE는 분류를 할 때 많이 사용함

$$MAE(y, y^{'}) = \frac{1}{n}\sum_{i=1}^{n}|y_1 - y_1^{'}|$$

$$MSE(y, y^{'}) = \frac{1}{n}\sum_{i=1}^{n}(y_1 - y_1^{'})^2$$

$$CEE(y, y^{'}) = -(\sum_{i=1}^{n} y_1 \times log(y_1^{'})) = -(1 \times log(y_1^{'}))$$

💡 엔트로피: 불확실성의 양

5주차_2차시

1. Introduction

Gradient Descent Method (GD Method / 경사하강법)

- 역전파를 위한 최적화

full-batch GD

Mini-batch GD: 잘라서

Stocastic GD

💡 Mini-batch GD = Stocatic GD

2. Concept of Backpropagation

💡 **편미분**

2차 함수인 Loss function의 값이 최소가 되는 x값을 찾아야 함

→ 편미분: 변수가 여러개인 함수를 하나의 변수에 대해서만 미분하는 것 → 미분을 통해 기울기가 0인 x값이 곧, Loss function이 최소가 되는 값임(실제 값과 가장 차이가 적은 값)

→ 그렇게 여러 개의 변수의 최소 값을 하나씩 찾아감

Backprogapagtion

- 계산 결과와 정답의 오차를 구해서 오차에 관여하는 노드 값들의 가중치와 편향을 수정하는데,
  오차가 작아지는 방향으로 반복해서 수정하는 기법
- 아무리 깊고 복잡한 층으로 구성되어 있다 하더라고 **Chain Rule을 활용**하여 미분 값을 얻어낼 수 있다.
- Forward Pass시 **Local Gradient를 미리 계산**하여 저장해 둔다.
- 저장해둔 **Local Gradient와 Global Gradient를 Backward Pass시 곱**하여 최종 미분 값을 얻는다.

💡 한 노드의 weigth, local gradient값이 아래와 같을 경우
Weight : 5
Local Gradient: -0.2

Local Gradient의 값은 weight값이 1증가 할때마다 증가하는 output의 값을 의미
EX 1) weight가 2증가해서 7이 됨 → output값: -0.2 × 2 만큼 증가한다.

EX 2) weight가 0.1증가해서 5.1이 됨 →outputr값: -0.2 × 0.1 만큼 증가한다.

💡 더하기: 현재 노드의 Global g → 이전 노드의 Global g가 됨

곱하기: 현재 노드의 Global g × (현재노드를 이전노드에 대해 미분값)

→ 이전 노드의 Global g가 됨

Gradient Descent(GD) Method (경사하강법)

- 주어진 어떤 지점에서부터 오차가 더 작은 곳으로 이동하려는 방법

μ : 학습률(learning rate, step size)

Δw : 탐색 방향

$$W^{t+1} \leftarrow W^t - \mu \nabla f(W^t), where \nabla f(W^t) \Delta w < 0$$

해석:

기울기가 0보다 작다는 건, 다음 W값이 더 작아지기 때문에 빼주는 거임

기울기가 0보다 크다면, 다음 W값은 더 커지기 때문에 더해줌.

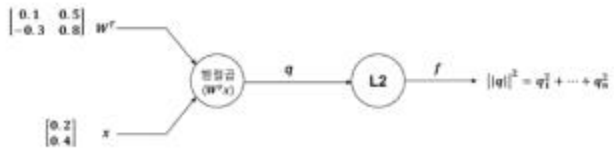
Sigmoid 계층의 backpropagation

▶ 사진

$$\frac{d}{dx} sigmoid(x) = \frac{d}{dx}(1 + e^{-1})^{-1} = sigmoid(x) \times (1 - sigmoid(x))$$

Deep Meural Network (MLP)

EX.



Loss 2 function을 거친 f(x)함수를 backpropagation할때, W^t의 gradient알아내는 방법

$$f(q) = \sum q^2, (q = y - y^{'})$$

$$\frac{\theta f(q)}{\theta q(x)} \times \frac{\theta q(x)}{\theta W^t} = (\frac{\theta f(q)}{\theta q(x)} = 2q(x)) \times (\frac{\theta W^t x}{\theta W^t} = x) = 2q(x)x$$

but, $2q(x)$는 행렬곱을 한$(W^t(n \times m) \times x(m \times 1))$, n×1의 행렬임, x = m ×1임

$\rightarrow (n \times 1)(m \times 1)$은 디맨션이 맞지 않음

우리는 W^t의 gradient를 구하기 때문에 n×m을 만들어야 됨

그래서, 아래 식처럼 해서 W^t의 gradient를 구할 수 있음

$$2q(x)x^{-1} = (n \times 1)(1 \times m) \rightarrow (n \times m)$$

Loss 2 function을 거친 f(x)함수를 backpropagation할때, x의 gradient알아내는 방법

$$\frac{\theta f(q)}{\theta q(x)} \times \frac{\theta q(x)}{\theta x} = (\frac{\theta f(q)}{\theta q(x)} = 2q(x)) \times (\frac{\theta W^t x}{\theta x} = W^t) = 2q(x)W^t$$

but, $2q(x)W^t \rightarrow (n \times 1)(n \times m)$ 그래서 아래 식처럼 해야됨

$$2(W^t)^t q(x) \rightarrow (m \times n)(n \times 1) \rightarrow (m \times 1)$$

6주차_1차시

MLP

: Multi-Layer Perceptron = Fully Connected layer, FC layer

활성화 함수(Activated Function): h()

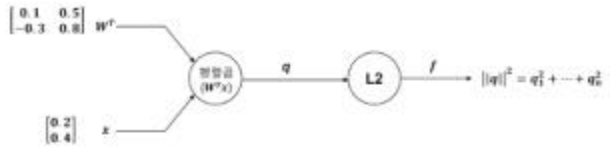
Softmax function: Normalization

- 여러 개의 입력을 받아 각각의 확률 값으로 출력(abs, square, 지수)
- 0~1 사이 실수를 출력하고, 모든 출력의 합은 1 (확률로 해석)
- Multi-label classification 모델의 출력층에 주로 사용됨
- sigmoid를 통해 0~1사이의 값으로 정규화할 수 있음

6주차_2차시

Deep Meural Network (MLP)

EX.



Loss 2 function을 거친 f(x)함수를 backpropagation할때, W^t의 gradient알아내는 방법

$$f(q) = \sum q^2, (q = y - y^{'})$$

$$\frac{\theta f(q)}{\theta q(x)} \times \frac{\theta q(x)}{\theta W^t} = (\frac{\theta f(q)}{\theta q(x)} = 2q(x)) \times (\frac{\theta W^t x}{\theta W^t} = x) = 2q(x)x$$

but, $2q(x)$는 행렬곱을 한$(W^t(n \times m) \times x(m \times 1))$, n×1의 행렬임, x = m ×1임

$\rightarrow (n \times 1)(m \times 1)$은 디맨션이 맞지 않음

우리는 W^t의 gradient를 구하기 때문에 n×m을 만들어야 됨

그래서, 아래 식처럼 해서 W^t의 gradient를 구할 수 있음

$$2q(x)x^{-1} = (n \times 1)(1 \times m) \rightarrow (n \times m)$$

Loss 2 function을 거친 f(x)함수를 backpropagation할때, x의 gradient알아내는 방법

$$\frac{\theta f(q)}{\theta q(x)} \times \frac{\theta q(x)}{\theta x} = (\frac{\theta f(q)}{\theta q(x)} = 2q(x)) \times (\frac{\theta W^t x}{\theta x} = W^t) = 2q(x)W^t$$

but, $2q(x)W^t \rightarrow (n \times 1)(n \times m)$ 그래서 아래 식처럼 해야됨

$$2(W^t)^t q(x) \rightarrow (m \times n)(n \times 1) \rightarrow (m \times 1)$$