|  |  |  |  |
| --- | --- | --- | --- |
| 딥러닝 기반 핵심 산업별 빅데이터 분석  <머신러닝&딥러닝 파일럿 프로젝트> | | | |
| 주 제 | Otto Group Product Classification Challenge | **링 크** | https://www.kaggle.com/c/otto-group-product-classification-challenge |
| 팀 명 | 하드캐리 | **일 자** | 2018년 11월 23일 |
| 팀 장 | 강호영<hoyoungk12@naver.com> | **팀 원** | 이상훈, 유영재 |

**1. 과제 개요**

|  |  |
| --- | --- |
| **팀의 목표** | - 수업에서 다뤘던 딥러닝의 다양한 기법들을 실제 실행하고 튜닝하면서 모든 팀원들이 딥러닝을 체득화  - 도전하는 모델 자체가 실용성을 가지고 있는 것 |
| **목표에 부합하는**  **데이터셋** | - 딥러닝 기법을 바로 적용해볼 수 있기 위해 데이터 전처리는 최소화할 수 있는 데이터  *Ex) feature들이 잘 정리되어 있고 가급적 수치적으로 표현된 데이터*  - 데이터가 분산되지 않고 train과 test로 형식으로 정리된 데이터  - 짧은 시간에 완성해야 하므로 도메인에 대한 전문 지식이 필요하지 않은 데이터  - 예측모델 자체가 기업이 현업에서 필요로 하는 실용성을 갖춘 데이터 |
| **프로젝트 주제** | Otto Group Product Classification Challenge (제품 카테고리 분류 모델) |
| **Otto Group 개요** | - Otto Group은 세계적으로 유명한 e-commerce 회사  - 20개국이 넘는 곳에 자회사를 갖춤, 매일 전 세계에 수백만 개의 제품들을 판매 |
| **제품 카테고리**  **분류가**  **필요한 이유** | - 제품의 성능에 대한 일관된 분석이 판매에 중요한 요소  > but 다양한 글로벌 인프라로 동일한 제품들이 다르게 분류되어 일관된 분석이 나오지 않는 문제  > 이를 해결하기 위해 유사한 제품들끼리 분류에 군집화하는 작업이 필요 |

**2. 데이터 설명**

|  |  |  |  |
| --- | --- | --- | --- |
| **데이터 종류** | train | | Test |
| **데이터 구성** | id (int형)  feat\_1 ~ feat\_93 (int형)  target (str형) | | id (int형)  feat\_1 ~ feat\_93 (int형) |
| **각 열의 특징** | **id** | 각 제품에 대해 번호를 매김 | |
| **Feat** | 93가지의 특징 중 각 제품에 해당하는 요소를 수치화한 데이터    각 feature마다 최대값이 다른 것을 확인할 수 있음 | |
| **target** | 제품 카테고리 | |
| **제출 형태** | 각 제품마다 분류군(9가지)에 대한 예상 확률 값을 담아 csv파일 제출 | | |
| **평가 방식** | 다중 클래스 로그 손실을 이용하여 평가    결론: 손실 값이 낮을수록 예상을 잘하는 모델 | | |
| **데이터 선택한 이유** | - 데이터 종류가 2가지로 단순함  - 대부분의 데이터가 머신 러닝을 바로 적용할 수 있는 숫자형 데이터  - 실제 기업에서 필요로 하는 실용성  - 도메인에 대한 깊은 전문지식이 필요하지 않은 용이한 접근성 | | |

**3. 과제 수행 내역**

**1) 목표**

-상품을 특징에 따라 효과적으로 분류하는 알고리즘 완성

-평가방식에 따라 최종점수 상위 30%(1050등)내 진입

-조원 모두가 딥러닝(NN)을 익숙하게 다루도록 성장

**2) 데이터 전처리**

**(1) 전처리의 필요성**

-전처리가 크게 필요하지 않은 데이터라고 생각했으나 실제 살펴본 결과 데이터 값에 0이 많고 최대값과의 차이가 크기 때문에 모델링 과정에서 데이터의 특성을 잘 반영하지 못하는 경향을 보임. 따라서 데이터를 평준화시킬 필요성이 발생.

-보다 좋은 모델을 만들기 위해 93개의 변수를 줄이거나 새로운 변수를 추가할 수 있는 가능성 탐색.

**(2) 데이터 표준화**

방법 1) (x - x평균)\*\*2

방법 2) (x-x\_mean)/x\_std

방법 3) Log(x+1)

**(2) 변수 설정**

모델1) 93개 feature를 모두 사용

모델2) 행별 feature의 합과 분산을 추가

**3) 활용 모델**

**사용 모델: NN**

-분류방법을 활용할 수 있는 과제였기 때문에 분류 알고리즘의 하나인 NN 모델을 활용

**4) 모델링**

**Try 1) 단순 딥러닝**

-데이터를 통으로 활용

-데이터 전처리

방법 1) (x - x평균)\*\*2

방법 2) (x-x\_mean)/x\_std

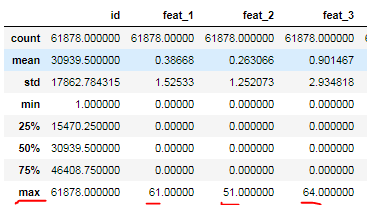
-활성함수, 레이어수, 노드개수, 자료형태, 러닝레이트 등을 변화시키며 최적의 모델 탐색

-결과: relu방식이 softmax방식 보다 성능이 안 좋아지는 현상이 발생

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 작성자 | 방법 | 레이어 | 노드  개수 | 자료형태 | learning  rate | Acc | step | drop | 점수 |
| 유영재 | adam과 gradient descent | 3 | 512 | Raw data | 1.00E-0.3 | 67.9% | 6000 |  | 5.7 |
| 강호영 | 기본 | 1 | 512 | 정규화 | 0.1 | 68.29% | 4000 |  |  |
| 강호영 | softmax-adam | 3 | 512 | 정규화 | 1.00E-03 | 74.93% | 4000 |  |  |
| 이상훈 | softmax-gradient descent | 1 | 512 | Mean  Square | 0.01 | 60.34% | 6000 |  |  |
| 강호영 | softmax-dropout-adam | 4 | 256 | 정규화 | 0.01 | 76.96% | 3000 |  |  |
| 유영재 | softmax adam | 4 | 256 | 정규화 | 1.00E-03 | 83.24 | 4000 |  | 15 |
| 이상훈 | sigmoid\_softmax\_softmax\_adam | 4 | 256 | Raw data | 1.00E-03 | 91% | 6000 |  | 7.2 |

→데이터의 문제?

: 데이터가 0이 너무 많고, feature마다 scale차이가 매우 상이함



.→다른 방식으로 데이터 전처리를 해야 할 필요성 발생

**Try 2) log 방식으로 데이터 표준화처리**

-relu, xavier를 적용할 수 있게 됨

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 작성자 | 방법 | 레이어 | 노드  개수 | 자료형태 | learning  rate | Acc | step | drop | 점수 |
| 강호영 | relu, xavier, drop out | 3 |  | 로그,  sum, var | 0.01 | 84.17% | 5000 | 0.7 |  |
| 유영재 | relu, xavier, | 3 | 512 | 로그 | 1.00E-05 | 66.72% | 5000 |  |  |
| 유영재 | relu, xavier, drop out | 3 | 512 | 로그 | 1.00E-05 | 49.10% | 2000 | 0.75 |  |
| 유영재 | relu, xavier | 3 | 128 | 로그 | 1.00E-05 | 64.86% | 6000 |  |  |
| 유영재 | relu, xavier | 3 | 128 | 로그 | 1.00E-04 | 66.67% | 4000 |  |  |
| 유영재 | relu, xavier | 3 | 256 | 로그 | 1.00E-04 | 88.47% | 6000 |  |  |
| 유영재 | relu, xavier, drop out | 3 | 256 | 로그 | 1.00E-03 | 88.05% | 6000 | 0.75 |  |
| 유영재 | relu, xavier | 3 | 256 | 로그 | 5.00E-04 |  | 10000 |  |  |
| 이상훈 | sigmoid\_softmax\_softmax\_adam\_10 | 3 | 512 | 로그 | 1.00E-03 | 95% | 10000 |  | 8.5. |
| 유영재 | **relu, xavier** | **3** | **256** | **로그** | **1.00E-03** | **97.79%** | **12000** |  | **1.9** |
| 강호영 | relu, xavier, | 3 | 256 | 로그, sum | 0.001 | 91.27% | 10000 |  | 4.7 |
| 강호영 | relu, xavier | 2 | 256 | 로그 | 0.005 | 86.44% | 5000 |  |  |
| 유영재 | **relu, xavier** | **3** | **256** | **로그** | **1.00E-03** | **95.71%** | **6000** |  | **1.41** |
| 유영재 | **relu, xavier, drop out** | **3** | **256** | **로그** | **1.00E-03** | **90.26%** | **10000** | **0.75** | **0.58** |
| 강호영 | relu, xavier | 2 | 256 | 로그 | 0.01 | 90.26 | 5000 |  | 4.3 |
| 유영재 | **relu, xavier** | **3** | **256** | **로그** | **1.00E-03** | **88.71%** | **6000** |  | **0.57** |
| 강호영 | relu, xavier | 2 | 512 | 로그 | 0.01 | 91.9 | 5000 |  | 3.9 |

-Train데이터 자체가 적어 과적합이 발생. Accuracy가 90%이상으로 올라갈수록 점수가 안 좋아지는 현상 발생

→과적합을 어떻게 해결할 수 있을까?

**Try 3) Train-Validation-Test**

|  |  |
| --- | --- |
| Train Data | |
| Train | **Validation** |

-Train Data를 Train과 Validation set으로 분리

-Train set에서 일정 비율을 반복 추출해 학습, Validation set을 통해 Accuracy가 좋아지면 weight값을 보존, Accuracy가 떨어지면 학습 데이터를 재 추출해 학습 반복.

추출방법 1) Train Set과 Validation Set을 학습을 반복할 때마다 재설정

결과: Accuracy가 일정부분 좋아지긴 함

학습한 데이터의 일부가 Validation Set으로 들어갔기 때문에 효과를 장담할 수 없음

추출방법 2) Validation Set은 보존, Train Set에서 지속적으로 일정 비율을 비복원추출

결과: 최종점수 1.2점. Train Data가 약 6만8천개로 거기서 다시 일정 비율을 추출한 결과 훈련데이터가 너무 작아져서 학습이 효과적으로 이루어지지 못함.

**4. 결과 보고**

.

**감사합니다**