**□ 모듈프로젝트 산출물**

클라우드 AI 융합 전문가 양성 과정

|  |
| --- |
| 모듈 프로젝트 3   * **행동 분석에 따른 순위 예측 –** * 2조 : 김민준, 박인호, 정지원, 허현준 |

**2021년 9월 30일**

**클라우드 AI 융합 전문가**

**이름 : 정지원**

|  |
| --- |
| **목 차**  Ⅰ. 이상치 제거  Ⅱ. 컬럼 추가  Ⅲ. 알고리즘  Ⅳ. 소감문 |

**Ⅰ. 데이터 전처리  
1.1. 결측지 제거**

|  |
| --- |
| **주요 코드** |
|  |

**1.2. IQR 방식**

**- 게임순위를 예측해야 하기 때문에 IQR방식으로 나온 이상치들은 단순히 잘하는 사람들의 데이터이기 때문에 바로 제거하지 않음**

**- 극상위, 극하위 유저 추출**

|  |
| --- |
| **주요 코드** |
|  |

**1.3. 가한 데미지 0인데 유효한 킬이나 기절이 발생한 경우**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**1.4 1등이 존재하지 않는 매치**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**1.5 실질적 플레이어가 한 명뿐인 매치**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**1.6 듀오, 스쿼드 등 그룹 인원수가 일치하지 않는 경우**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**1.7 매치 킬 수가 이상한 경우**

**- GAME 매치에 최대 100명이 들어가기 때문에 매치당 킬 수의 합은 최대 ’99’ 킬이다. 그렇지 않은 경우는 오류가 난 데이터 이므로 이를 색출**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**1.8 매치 내 인원수보다 킬 수 더 많은 경우**

**- 배틀그라운드 게임의 경우 한 매치에 참여한 사람수와 총 킬수와 같거나 크면 안됌**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**1.9 운전한 거리에 비해 로드킬 수가 과하게 많은 경우 제거**

**- 과하게 많은 기준은 로드킬 하는데 필요한 운전한 거리는 1m로 측정**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**1.10 무기 사용이 0인데 1등을 한 플레이어**

**- 무기를 사용하지 않고 1등을 할 수 있는 경우(duo, squad) 에서는 가능**

**- 그러나 ‘Solo’ 모드에서 무기를 사용하지 않고 'solo'가 1등을 할 수 없음.**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**1.11 혼자 게임하는데 팀원을 살린 경우**

|  |
| --- |
| **주요 코드** |
|  |

**Ⅱ. 컬럼추가**

**2.1. 샘플링**

**- 용량이 큰 데이터셋 때문에 효율적인 분석을 위하여 약 2만개 데이터를 sample함수로 추출**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**2.2. 매치별 지표 백분위 순위 컬럼 추가**

**- Rank함수를 이용하여 정규화된 지표를 추가시킴**

|  |
| --- |
| **데이터 프레임** |
|  |
| **주요 코드** |
|  |

**2.3. 그룹화 칼럼 추가**

**- 매치별 통계와 그룹별 통계의 정보를 포함하는 칼럼은 concat을 이용하여 생성 및 추가**

|  |
| --- |
| **데이터 프레임** |
|  |
| **주요 코드** |
| **데이터 프레임** |
|  |
| **주요코드** |
|  |

**2.4. 문자형 데이터 이산화 칼럼 추가**

**- 총 84개 컬럼 추가**

|  |
| --- |
| **결과** |
|  |
| **주요 코드** |
|  |

**2.5. features 관계성 시각화**

**- 히트맵을 통해 상관관계를 시각화**

|  |
| --- |
| **시각화** |
|  |
| **주요 코드** |
|  |

**2.6. 상관 계수 처리**

**- 피어슨의 상관계수를 통해 절대값 0.1 보다 작으면 상관관계가 없다라고 판단**

**- 훈련 데이터셋에서는 상관 계수를 처리한 버전과 처리하지 않은 버전 모두 사용**

|  |
| --- |
| **시각화** |
|  |
| **주요 코드** |
|  |

**2.7. 다중공선성**

**-** 다중공선성이 궁극적으로 오버피팅을 야기할 수 있음

- scaler를 통해 컬럼별 범위를 정규화 화

- VIF가 10 이상인 경우 일반적으로 다중공선성이 매우 높은 변수라고 판단후 삭제

|  |
| --- |
| **시각화** |
|  |
| **주요 코드** |
| - MinMaxScaler를 사용해 컬럼의 값들을 정규화  - 정규화 한 값들을 데이터프레임 형태로 리턴    - 스케일링한 데이터를 함수로 호출  - 다중공선성 수치와 수치에 따른 컬럼이름을 담을 데이터 프레임 생성  - 다중 공선성이 제일 큰 값의 인덱스를 가지고 스케일링한 데이터에서 해당 컬럼 삭제  - 위 3단계 반복, 다중공선석 수치가 10보다 모두 작은 경우 삭제 |

**Ⅲ. 알고리즘**

**3.1 GridSearchCV**

**- 각 모델에서 사용하는 파라미터를 GridSerachCV로 하이퍼 파라미터를 찾으려고 시도함**

**- - ex) Light Gradient Boosting Machine**

**3.1.1 파라미터 설명 (Booster Parameters 기준)**

**1. min\_child\_weight [default=1] (Should be tuned using CV)**

1. **child의 관측(?)에서 요구되는 최소 가중치의 합**
2. **over-fitting vs under-fitting을 조정하기 위한 파라미터.**
3. **너무 큰 값이 주어지면 under-fitting.**

**2. max\_depth [default=6] (Should be tuned using CV)**

1. **트리의 최대 깊이.**
2. **일반적으로 3-10**

**3. max\_leaf\_nodes**

1. **종 노드의 최대 개수. (max number of terminal nodes)**
2. **이진 트리가 생성되기 때문에 max\_depth가 6이면 max\_leaf\_nodes는 2^6개가 됨.**

**4. gamma [default=0]**

1. **분할을 수행하는데 필요한 최소 손실 감소를 지정**
2. **알고리즘을 보수적으로 만든다. loss function에 따라 조정**

**5. subsample [default=1]**

1. **각 트리마다의 관측 데이터 샘플링 비율.**
2. **값을 적게 주면 over-fitting을 방지하지만 값을 너무 작게 주면 under-fitting.**
3. **일반적으로 0.5-1**

**6. colsample\_bytree [default=1]**

1. **각 트리마다의 feature 샘플링 비율.**
2. **일반적으로 0.5-1**

**7. lambda [default=1]**

1. **가중치에 대한 L2 정규화 용어**

**8. alpha [default=0]**

1. **가중치에 대한 L1 정규화 용어**

|  |
| --- |
| **하이퍼 파라미터 결과** |
|  |
| **주요 코드** |
|  |

**3.2. xgboost**

|  |  |
| --- | --- |
| **결과** | |
| **튜닝 전** | **튜닝 후** |
|  |  |
| **설명** | |
| - XGBoost는 Extreme Gradient Boosting의 약자  - Boosting 기법을 이용하여 구현한 알고리즘은 Gradient Boost 가 대표적인데 이 알고리즘을 병렬 학습이 지원되도록 구현한 라이브러리  - Regression, Classification 문제를 모두 지원하며, 성능과 자원 효율이 좋아서, 인기 있게 사용되는 알고리즘 | |
| **주요 코드** | |
|  |  |

**3.3. Light Gradient Boosting Machine(LGBM)**

|  |  |
| --- | --- |
| **결과** | |
| **튜닝 전** | **튜닝 후** |
|  |  |
| **설명** | |
| C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\Content.Word\제목 없음.png | |
| **주요 코드** | |
|  |  |

**3.4. Random Forest Regressor**

|  |  |
| --- | --- |
| **결과** | |
| **튜닝 전** | **튜닝 후** |
|  |  |
| **설명** | |
| 제목 없음 | |
| **주요 코드** | |
|  |  |

**3.5. Gradient Boosting Regressor**

|  |  |  |
| --- | --- | --- |
| **결과** | | |
| **튜닝 전** | **튜닝 후** | |
|  |  | |
| **설명** | | |
| - Gradient Boosting Algorithm (GBM)은 회귀분석 또는 분류 분석을 수행할 수 있는 예측모형이며 예측모형의 앙상블 방법론 중 부스팅 계열에 속하는 알고리즘.- Gradient Boosting Algorithm은 Tabular format 데이터 (엑셀형태와 같이 X-Y Grid로 되어있는 데이터)에 대한 예측에서 엄청난 성능을 보여주고, 머신러닝 알고리즘 중에서도 가장 예측 성능이 높다고 알려진 알고리즘- LightGBM, CatBoost, XGBoost 같은 파이썬 패키지들이 모두 Gradient Boosting Algorithm을 구현한 패키지- GBM은 계산량이 상당히 많이 필요한 알고리즘이기 때문에, 이를 하드웨어 효율적으로 구현하는 것이 필요한데, 위 패키지들은 모두 GBM을 효율적으로 구현하려고한 패키지들이라고 볼 수 있음. C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\Content.Word\제목 없음.png | | |
| **주요 코드** | | |
|  | |  |

**3.6. Extra Trees Regressor**

|  |  |
| --- | --- |
| **결과** | |
| **튜닝 전** | **튜닝 후** |
|  |  |
| **설명** | |
| 제목 없음 | |
| **주요 코드** | |
|  |  |

**3.7. 5개 모델 성능 비교**

|  |
| --- |
| **데이터 프레임** |
|  |
| **주요 코드** |
| * 하이퍼 파라미터로 5개모델 객체 생성 후 리스트에 넣음 * X,y\_train(test)\_list는 컬럼추가가 다른 데이터셋 2개와 앞 2개 데이터에서 상관계수 처리까지 한 데이터셋 2개, 총 4개의 데이터셋을 넣음 * 빈 데이터프레임 생성      * 모델 리스트에 담긴 순서대로 학습 및 예측 * 각 성능 지표들을 계산 후 데이터 프레임 열 추가 * 추가되는 열 이름은 1,2번째는 컬럼V1 , V2 + 모델이름 * 3,4 번쨰는 상관계수를 처리한 데이터 프레임이기 때문에 컬럼V1, V2 + 상관 계수+ 모델이름 으로 설정 * 행의 인덱스를 num으로 증가 시키면서 덮어쓰기 방지 |

**3.7. AutoML**

**-** Pycaret에서 제공하는 AutoML을 사용해 모델 13개에 대해 성능을 비교

**-** 상위 3개의 모델을 앙상블기법 중 하나인 블렌딩을 통해 하나의 모델로 만든 후 10-fold 교차 검증

|  |
| --- |
| **결과** |
|  |
| **블렌딩 결과** |
|  |
| **설명** |
| 제목 없음 |
| **주요 코드** |
|  |

**Ⅳ. 소감문**

|  |  |
| --- | --- |
| 성명 | 프로젝트 후 소감 |
| 김민준 | 이번 프로젝트는 정형데이터에 대한 인사이트를 얻기 위해 데이콘 과제(저희 조는 케글)  를 하나 정해서 머신러닝을 하기 위한 전과정을 직접 시행해 보는 프로젝트였습니다.  알고리즘에 데이터를 넣기 전 성능 향상을 위해 전처리 과정을 거치는데 이 부분에서 성  능 차이가 많이 나기 때문에 중요했습니다. 다른 조와 달리 우리 조는 해당 주제의 데이  터에 대한 사전 지식이 있기 때문에 시각화 과정을 거의 거치지 않고 이상치를 쳐내는  함수를 만들 수 있었습니다. 알고리즘의 적용의 경우 sklearn에서 패키지로 제공하기 때  문에 그 원리를 정확히 몰라도 쉽게 적용할 수 있었습니다. 여기에 추가로 AutoML을 사  용하여 여러 모델의 성능을 한눈에 비교해보고 블랜딩을 적용시켜 하나의 모델로 만들기  도 했습니다. 이 밖에도 데이터셋의 다양화, parameter 최적화를 통해 가장 작은 MSE를  얻기 위해 노력했고 실제 성과도 있어서 만족스러웠습니다. 쉽지 않은 프로젝트였지만 함  께 열심히 해준 팀원들과 팀장님에게 감사합니다. |
| 박인호 | 어떤 주제로 선정할지 조원들과 고민 중 저희가 가장 흥미롭게 할 수 있는 주제로 해보려고 ‘배틀그라운드’ 게임 소재로 정했습니다. 머신러닝 알고리즘이 혼자 학습할 때는 상당히 어려웠는데 평소 자주 해보던 게임으로 알고리즘을 학습시켜보니 이해하는 데에 큰 도움이 되었습니다. 그리고 조원들의 큰 도움이 있어서 모르는 부분에 대해서 다시 공부해 볼 수 있는 기회가 되었고, 조원들 옆에서 많은 자극을 받은 프로젝트였던 거 같습니다. |
| 정지원 | 지금까지는 주제가 정해져 있었지만 이번에는 자유 주제로 프로젝트를 하게 되었고, 주제를 정하는 것이 가장 큰 과제라고 생각했습니다. 첫 회의는 대면으로 진행 하였습니다.  온라인 전에 얼굴만 봤던 동료들이었는데 이렇게 만나서 얼굴도 보고, 이야기도 나누면서 회의를 하다 보니 더 빠르게 서로에 대해서 알 수 있었고, 더욱 친해질 수 있었던 것 같습니다. 첫날에는 거의 주제를 선정하는 데에 시간을 보냈고, 친목 도모의 시간을 가졌습니다. 고민 끝에 저희는 ‘배틀그라운드’라는 유명한 슈팅 게임에 대한 주제를 정하게 되었습니다. 저는 게임을 하지 않아 생소했던 주제였지만 조원들과는 즐겁게 잘 할 수 있겠다는 확신이 들었습니다. 하지만 컬럼들을 봐도 무슨 말인지 몰랐고, 집에 와서 ‘배틀그라운드’ 게임에 대해 먼저 공부를 했습니다. 게임에 관한 공부를 한 건 살면서 처음인 것 같습니다.. 저희가 선택한 ‘배틀그라운드’ 데이터는 양이 너무 많아서 전 처리하는 과정이 쉽지가 않았습니다. 지금까지는 강의 내용을 따라하면 적용이 되는 코딩이었다고 하면 이번에는 그 과정을 수행하기 전에 응용하고 모르는 전처리 학습은 찾아보면서 공부를 해야 했고, 시간도 많이 걸려 더 조급했던 것 같습니다. 그래서 더 도전해보고 싶었고 오기가 생겼던 것 같습니다. 코딩은 깊게 들어가면 깊게 들어갈수록 신기했고, 흥미가 계속해서 생겼습니다. 이런 시행착오가 많아 부족한 점도 많고, 미안한 부분도 많고, 아쉬운 점도 많았지만, 막내지만 조장의 역할을 너무 잘해주었던 현준이, 추진력으로 프로젝트 방향을 잘 잡아주었던 민준이, 배려심이 많아 프로젝트 중 제 고민을 잘 들어주었던 인호까지 2조라는 팀과 함께해서 좋은 결과, 좋은 추억을 만들 수 있었던 것 같습니다. 감사합니다. |
| 허현준 | 캐 글에서 진행했던 게임 유저의 행동 패턴에 따른 순위 예측 대회를 팀원들과 같이하면서 다른 사람과 우리 팀원들 모두 문제 해결방식이 다양하고 재밌었다. 팀원들과의 적극적인 의사소통으로 인해 빠르게 방향성을 잘 잡고 프로젝트를 잘 마무리할 수 있었다. 또한 다른 팀들의 발표를 보면서 다양한 전처리 방식과 알고리즘을 배웠다.  특히 칼럼별 범위 값이 많이 차이 나면 그것을 조정할 수 있게 해주는 스케일러에 대한 것이 제일 기억에 남고 좋은 성능을 내지만 우리는 사용하지 않았던 Cat boost 알고리즘에 대한 것도 알 수 있었다. 따라서 이번에 진행했던 프로젝트에 다음에 적용해볼 생각이다.  마지막으로는 머신러닝을 딥러닝이 완전하게 대체할 수 있는지에 대한 것도 생각해볼 수 있었던 프로젝트였다 |