KB 국민은행

KB IT's Your Life 1기

Machine Learning - 배구 경기 결과 예측

1조
박세희
박준석
백정은
심범수

Contents

01

데이터 분석

- EDA
- feature engineering
- feature extraction

01 EDA

- **대한민국 프로 배구 리그 데이터**
  - 경기 날짜, 팀 별 기록/선수 상세 정보

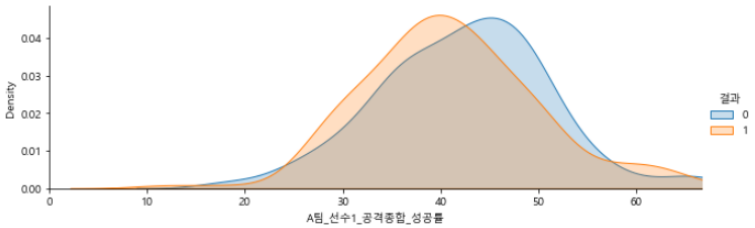
    공격득점, 블로킹득점, 서브득점, 상대범실, 전체득점, 디그성공, 리시브정확.. etc

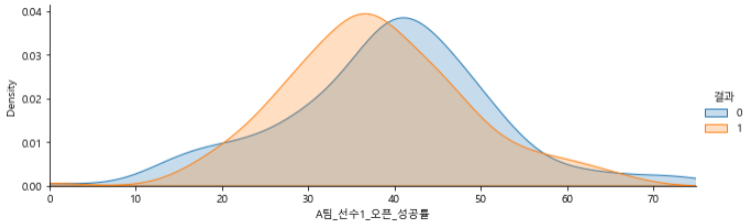
	경기번호	팀명	경기날짜	결과	No.	이름	출전세트_1set	출전세트_2set	출전세트_3set	출전세트_4set	...	블로킹_시도	블로킹_성공	블로킹_유효블락	블로킹_실패	블로킹_범실	블로킹_세트당	블로킹_점유율	블로킹_어시스트	벌칙_벌칙	범실_범실
0	1	IBK기업은행	2017-10-14	0	19	메디(L)	O	O	O	O	...	20	3	5	6	1	0.6	21.28	1	0	7

- **데이터 추출 방식**
  - 셀러니움 라이브러리로 웹페이지 자동 스크랩
  - 시즌, 선수별 기록 비교 페이지 크롤링

01 EDA

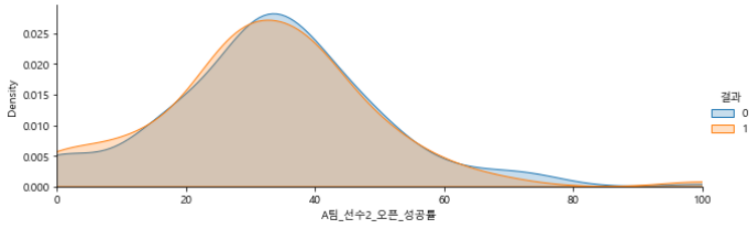
- **유의한 특성**

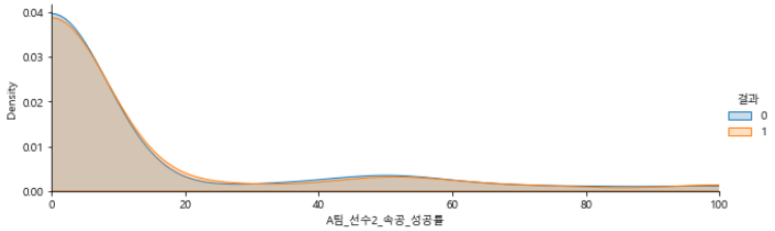




[공격종합성공률, 범실, 오픈, 퀵오픈 …]

- **유의하지 않은 특성**





[시간차성공률, 서브성공률, 이동성공률 …]

01 Feature Engineering

- **칼럼 추가**

$$\text{A팀 공격종합성공률} = \frac{1}{5}\sum_{i}^{5} A\text{팀 } i\text{번째 선수 공격종합성공률}$$

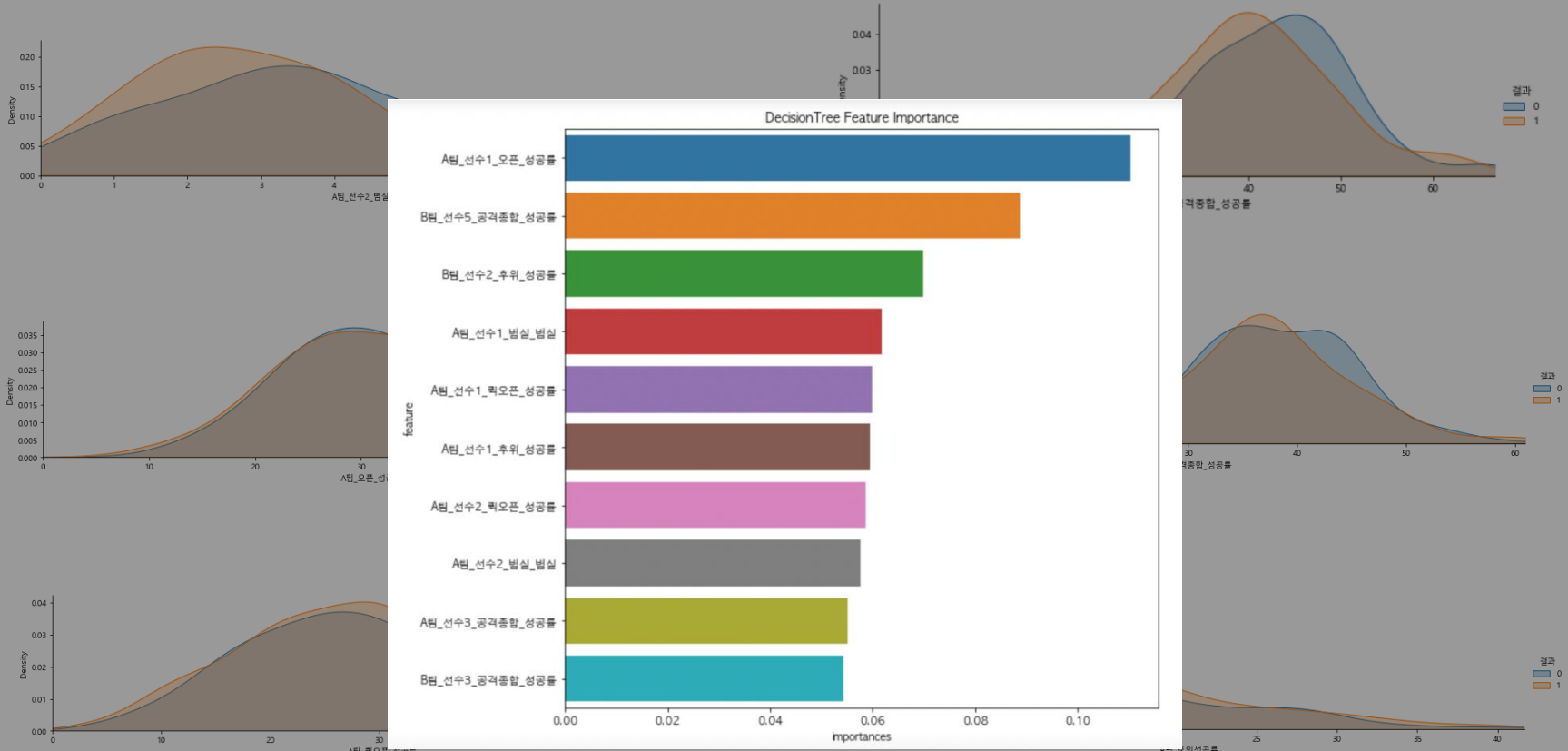
$$\text{B팀 공격종합성공률} = \frac{1}{5}\sum_{i}^{5} B\text{팀 } i\text{번째 선수 공격종합성공률}$$

⋮

10개의 팀별 칼럼 추가

A팀_공격종합_성공률	B팀_공격종합_성공률	A팀_오픈_성공률	B팀_오픈_성공률	A팀_범실	B팀_범실	A팀_퀵오픈_성공률	B팀_퀵오픈_성공률	A팀_후위성공률	B팀_후위성공률
22.526	37.386	11.766	26.924	4.8	5.0	31.090	24.224	1.666	20.000
35.134	44.600	34.498	35.770	3.0	3.4	29.104	15.368	18.430	6.666

01 Feature Extraction



01 Feature Extraction

column	example
A팀_선수1_공격종합_성공률	
A팀_선수1_오픈_성공률	
A팀_선수1_후위_성공률	
A팀_선수1_퀵오픈_성공률	
A팀_선수1_범실_범실	
A팀_선수2_퀵오픈_성공률	
A팀_선수2_범실_범실	
A팀_선수4_후위_성공률	
A팀_선수4_범실_범실	
A팀_선수5_공격종합_성공률	
A팀_선수5_오픈_성공률	
A팀_선수5_범실_범실	
B팀_선수1_공격종합_성공률	
B팀_선수1_오픈_성공률	
B팀_선수1_퀵오픈_성공률	
B팀_선수1_범실_범실	

column	example
B팀_선수1_후위_성공률	
B팀_선수2_공격종합_성공률	
B팀_선수2_후위_성공률	
A팀_공격종합_성공률	
B팀_공격종합_성공률	
A팀_범실	
B팀_범실	
A팀_후위성공률	
B팀_후위성공률	

02

Machine Learning

- ML
- Auto ML

02 단일 알고리즘 모델 비교

Train score 95% 이하	train	test	Hyperparameter
Logistic Regression	0.653846	0.540229	X
Decision Tree	0.819230	0.643678	max_depth = 8, min_samples_split=20, max_features=0.7
RandomForest	0.776923	0.666666	n_estimators = 70, n_jobs = -1, max_depth = 2, max_features = 0.1
GradientBoost	0.953692	0.620689	learning_rate=0.05, n_estimators=50, subsample=0.8, max_features='auto'
XGBoost	0.838461	0.701149	n_estimators = 10, n_jobs = -1, max_depth = 2, learning_rate = 0.4
LightGBM	0.919230	0.666666	learning_rate=0.01, n_estimators=2500, subsample=0.7, max_depth=2, n_jobs = -1
RidgeClassifier	0.661538	0.551724	alpha = 1.5

02 단일 알고리즘 모델 비교

Train_test_diff 20% 이하	train	test	Hyperparameter
Logistic Regression	0.653846	0.540229	X
Decision Tree	0.734615	0.597701	max_depth =4, min_samples_split=30, max_features=0.7
RandomForest	0.646153	0.597701	n_estimators = 10, n_jobs = -1, max_depth = 1, max_features = 0.3
GradientBoost	0.788461	0.609195	learning_rate=0.0001, n_estimators=1000, subsample=0.8
XGBoost	0.838461	0.701149	n_estimators = 10, n_jobs = -1, max_depth = 2, learning_rate = 0.4
LightGBM	0.846153	0.643678	learning_rate=0.01, n_estimators=680, subsample=0.8, max_depth=2, n_jobs = -1
RidgeClassifier	0.661538	0.551724	alpha = 1.5

02 voting

Random Forest	XGB Classifier	LGB Classifier
0.7769	0.8384	0.9307
0.6667	0.7011	0.6551

Soft voiting

Voting Classifier
0.8884
0.6781

- **Voting 알고리즘의 결과?**
  - test score 기준 2위를 달성
  - train_test_diff 기준 2위를 달성

그러나, 두 기준 모두
XGB Classifer 단일모델이 우수하다.

=> Voting 알고리즘을 적용하지 않는다.

02 Auto ML 해석

성능 top3 모델 Tuning

- GradientBoostingClassifier
- AdaBoostClassifier
- LGBMClassifier

	Accuracy
GradientBoosting	0.5585
AdaBoost	0.5463
LGBM	0.5458

모델 Blending & Tuning

- GradientBoostingClassifier

	Accuracy
Mean	0.6363
Std	0.0883

02 Auto ML vs ML

Auto ML

	Accuracy
GradientBoosting	0.6363
XGBM	X
LGBM	0.6076

ML

	Accuracy
GradientBoosting	0.6436
XGBM	0.7011
LGBM	0.6551

- XGB는 지원하지 않아 비교할 수 없었다.
- GradientBoosting, LGBM 모두 ML이 더 좋은 성능을 보였다.

03

회고

- 어려웠던점
- 좋았던점

03 회고

- **작업방식**
  - 교수님이 제공해주신 컬럼을 기반으로 분석 작업 진행 후 classifier 진행
  - 이후 필요한 컬럼들을 재선택하여 분석 후 classifier 진행
  - AutoML과의 성능 비교

- **아쉬웠던 점**
  - 각 포지션에 대한 정보를 모델에 활용하지 못하였다.

- **가장 중요하게 여긴 부분**
  - 친구한테 물어본 것
  - 어떠한 컬럼을 기반으로 최고의 수치를 만들어낼 수 있는지를 중점으로 분석을 진행

- **잘 된 점**
  - Train score와 Test score의 차이를 예상보다 더 줄일 수 있었음.

감사합니다.