2020 BIG CONTEST

KBO 정규시즌 팀별 승률, 타율, 방어율 예측

Team _ Big Contact

남상대 (팀장) nsd96@naver.com
김동현 ddhh445@naver.com
김민영 kmy0433@naver.com
이윤정 lyj000105@naver.com
임창건 ckdrjs96@gmail.com

목차

STEP 1
문제 정의

STEP 2
데이터 개요 및
분석목표

STEP 3
데이터 전처리

STEP 4
EDA 및
모델 선정 검증 과정

STEP 5
모델 구축 및 훈련

STEP 6
결과

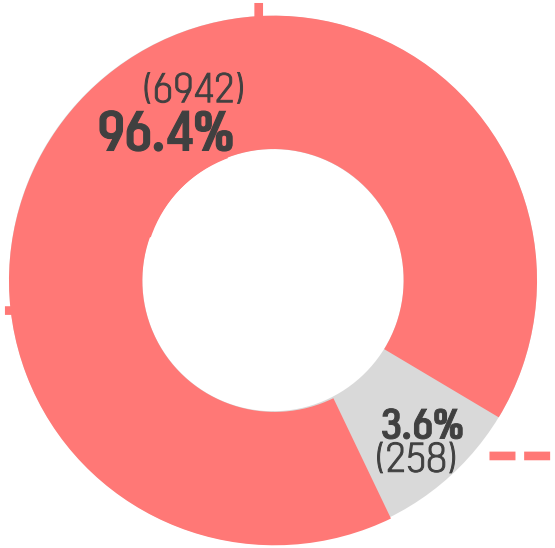
STEP 7
결론 및 느낀 점

STEP 8
향후 계획

1. 문제 정의

제공데이터

2016년 – 2020년 7월 20일
야구 데이터



수집데이터

2020년 7월 21일
- 2020년 9월 28일

예측

2020.09.28-종료시점까지의
승률, 타율, 방어율 예측

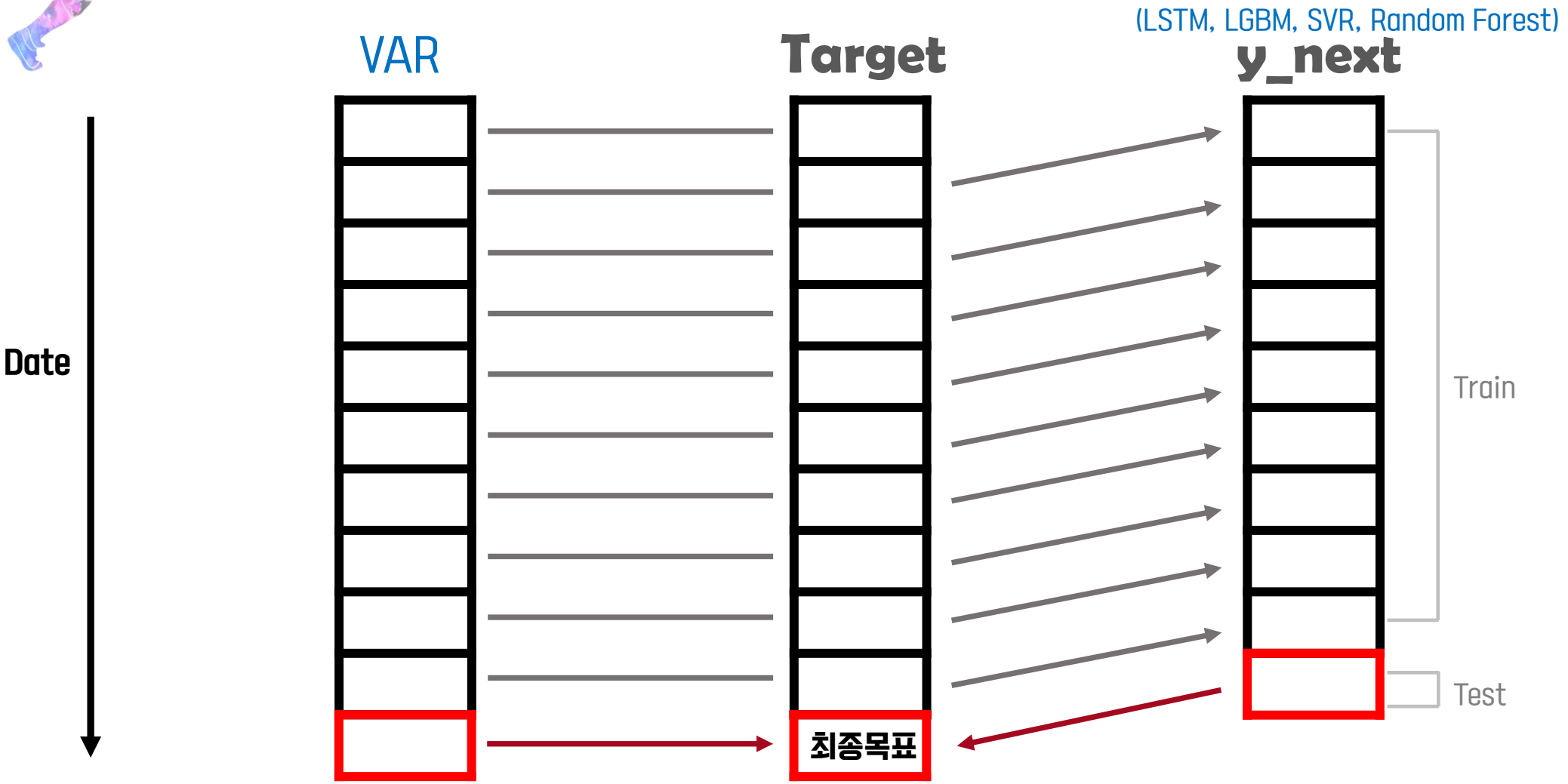
2. 데이터 개요 및 분석목표

기본 제공 데이터

개인타자	개인투수	경기	등록선수	선수	팀
2016~2020	2016~2020	2016~2020	2016~2020	2016~2020	2016~2020

2. 데이터 개요 및 분석목표

분석 목표 도식화



3. 데이터 전처리

1) EDA를 통한 이상치 제거

FIRST STEP

< 사전 조사 결과를 바탕으로 COLD GAME DATA를 파악한 후 제거 >

1) 2017.07.02. KT vs 넥슨 경기

: 강우 콜드 처리 -> 넥센 승

2) 2017.07.02. 삼성 vs SK 경기

: 우천 취소 -> SK 승

3) 2017.07.02 KIA vs LG 경기

: 강우콜드 선언

→ **20170702 WOKTO, 20170702SSSKO, 20170702HTLGO은 이상치로 간주 후 제거**

In [96]:
```
#최종적으로 콜드게임으로 인한 이상치로 규정한 경기들:
cold_list=['20170702WOKTO','20170702SSSKO','20170702HTLGO','20190407HHLT
O','20200610HTKTO']
```

3. 데이터 전처리

1) EDA를 통한 이상치 제거

SECOND STEP

< 팀타자 이상치 제거>

: 운의 요소가 많은 지표들에 대하여 이상치 검정을 시행

: n개 이상의 변수에 대해서 이상치 판정

→ 운 적인 요소가 많이 작용했던 경기로 간주하고 제거함.

→ 즉, n을 높일수록 보수적인 이상치 선정

➔ Box Plot을 통해 해당 변수들의 이상치 분포를 확인함.

➔ **팀타자의 이상치 변수 선정 : 'CS' , 'HP', 'ERR', 'P_HRA_RT', 'P_AB_CN', 'P_HIT_CN'**

3. 데이터 전처리

1) EDA를 통한 이상치 제거

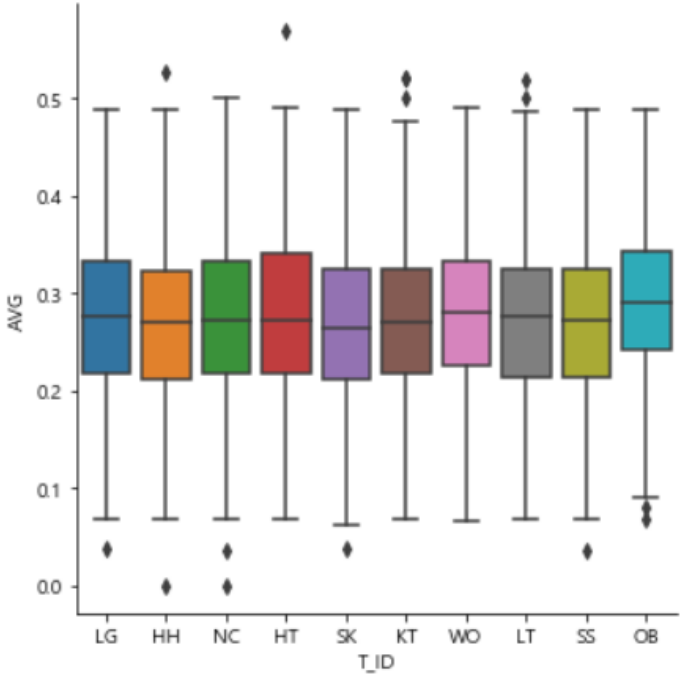
SECOND STEP

< Box_plot 예시 >

```
team_list=team['T_ID'].unique()

sns.factorplot(data=team,x='T_ID',y='AVG',kind='box')
```

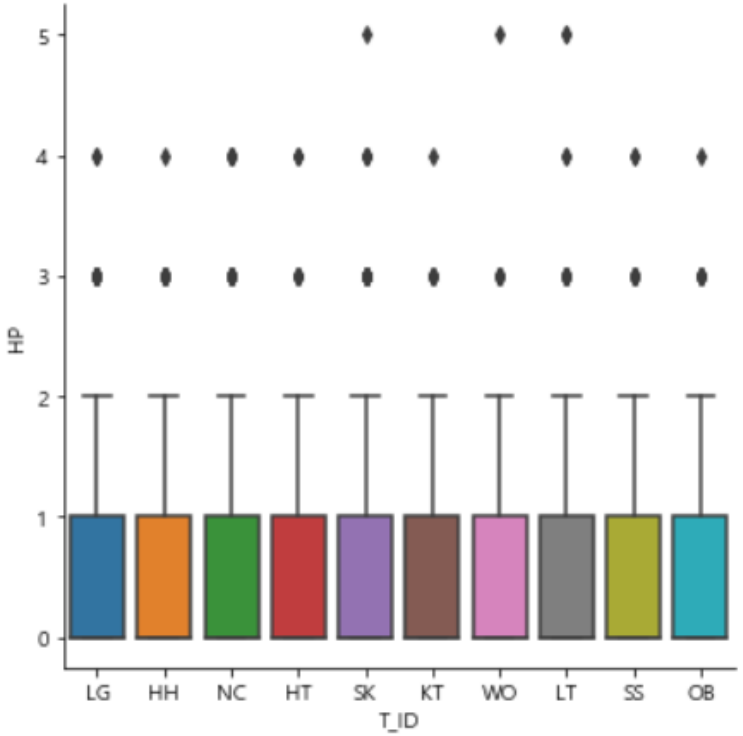
```
<seaborn.axisgrid.FacetGrid at 0x201c519df08>
```



팀별 AVG 변수

```
sns.factorplot(data=team,x='T_ID',y='HP',kind='box')
```

```
<seaborn.axisgrid.FacetGrid at 0x201c82b8308>
```



HP 변수(사구)

3. 데이터 전처리

1) EDA를 통한 이상치 제거

< 이상치 제거 코드 >

```
from collections import Counter
def detect_outliers(df, n, features):
    outlier_indices = []
    for col in features:
        Q1 = np.percentile(df[col], 25)
        Q3 = np.percentile(df[col], 75)
        IQR = Q3 - Q1
        outlier_step = 1.5 * IQR
        outlier_list_col = df[(df[col] < Q1 - outlier_step) | (df[col] > Q3
+ outlier_step)].index
        outlier_indices.extend(outlier_list_col)
    outlier_indices = Counter(outlier_indices)
    multiple_outliers = list(k for k, v in outlier_indices.items() if v >
n)
    return multiple_outliers

#운적인 지표 및 타율에 대해서만 이상치 제거 시행.
#2개 이상의 피쳐에 대해서 이상치가 발생했다면 이상치로 규정(운적인 요소가
많이 작용했을 가능성이 높기 때문에)
```

n : 이상치로 규정된 운 적인 요소가 몇 개 이상일 때 경기 전체를 이상치로 규정할 것인지 정하는 파라미터

SECOND STEP

3. 데이터 전처리

1) EDA를 통한 이상치 제거

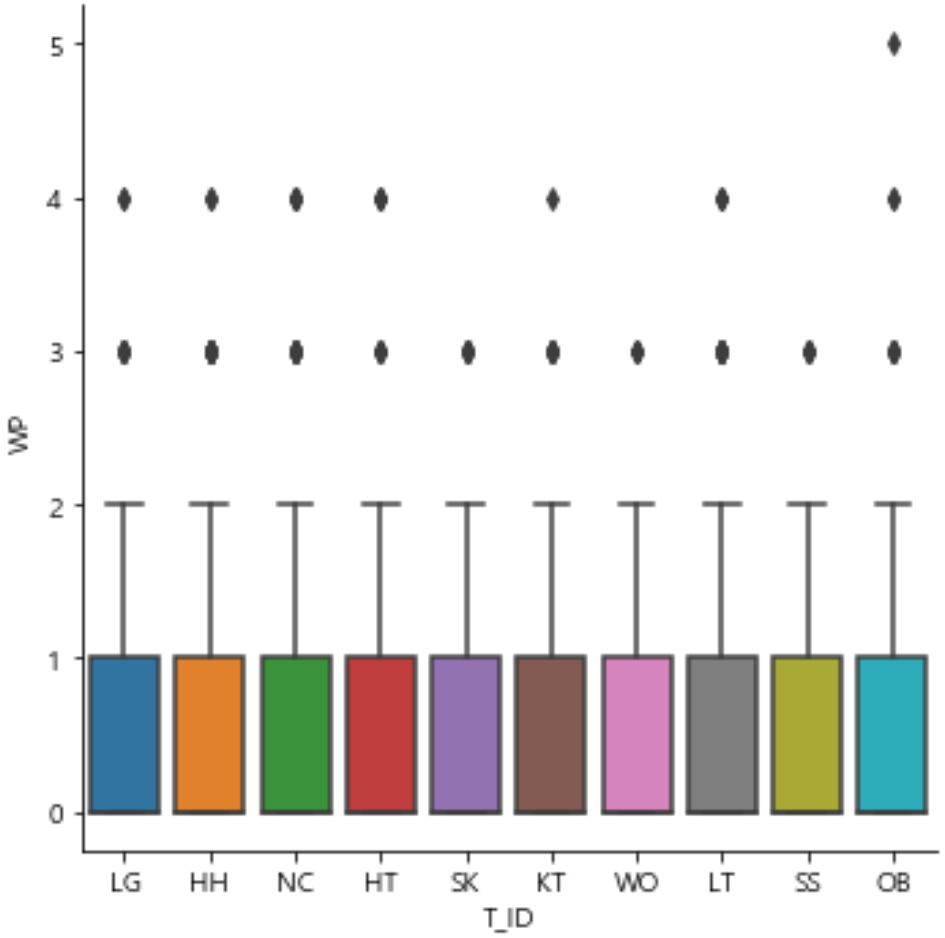
THIRD STEP

< 팀투수 이상치 제거>

: Box Plot을 통해 해당 변수들의 이상치 분포를 확인함.

: 팀투수의 이상치 변수 선정

'CS' , 'HP', 'ERR', 'WP', 'ER'



WP변수

3. 데이터 전처리

2) 세이버 메트릭스 지표 추가

세이버 메트릭스 ?

야구 경기에서 일어나는 선수의 모든 행위를 기록
가장 보편적으로 사용되고 권위 있는 야구 통계 지표

관련 논문

- 한국 프로야구 타자의 경기력 요인 분석
- 2016-2018 한국프로야구 세이버 메트릭스 지표분석
- 세이버 메트릭스 소개 및 통계적 측면의 한국 프로야구 기록 분석

지수	공식
OPS	OPS=출루율+장타율 출루율=(안타+4사구)(타수+4사구+희비) 장타율={1루타+(2루타×2)+(3루타×3)+(홈런×4)}/타수
GPA	GPA={(출루율×1.8)+장타율}/4
SECA	SECA={2루타+(3루타×2)+(홈런×3)+4사구+도루−도실}/타수
TA	TA=(루타수+몸에맞는볼+볼넷+도루)(타수−안타+도실+병살타)
RC	RC=A×B/C A=(안타+4사구−도실−병살타) B={루타수+0.52×(도루+희생타)+0.26×(4사구−고의4구)} C=(타수+4사구+희생타)
RC/27	RC/27=(RC×27)/A A=(타수-안타+희타+희비+도실+병살타)

지표 예시

OPS, GPA, SECA. TA, RC, XR, BR, ISO, PSN, wOBA, OW%, BABIP …

3. 데이터 전처리

3) 트랜드 반영 변수 추가

AVG
ex. 타율

−

avg_AVG
ex. 총 데이터에 대한 타율 평균

=

AVG_trend
ex. 트랜드 변수

+ 상승세

− 하락세

```
df['AVG_trend'] = (df['AVG'] - df['avg_AVG']).astype(np.float16) df.drop(['avg_AVG'],axis=1,inplace=True)
```

시계열성을 반영하기 위하여 트랜드, 시차, rolling window, expanding window 변수 추가 고민

➔ 트랜드 외 변수에서 데이터 손실 발생 ➔ **트랜드 변수만 추가**

트랜드 값이 양수 : 상승세, 음수 : 하락세

4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

- Regression 모델 : ridge, Lasso, best subset selection, GLM
- 시계열 모델 : VAR, ARIMA, SARIMA, FBprophet

Non - Parametric Model

- SVR
- KNN
- Tree 모델 : Random Froest, LGBM
- LSTM, DNN

4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

I. Regression 모델 : ridge, Lasso, best subset selection, GLM

1. **사용전제**를 만족하지 않음.

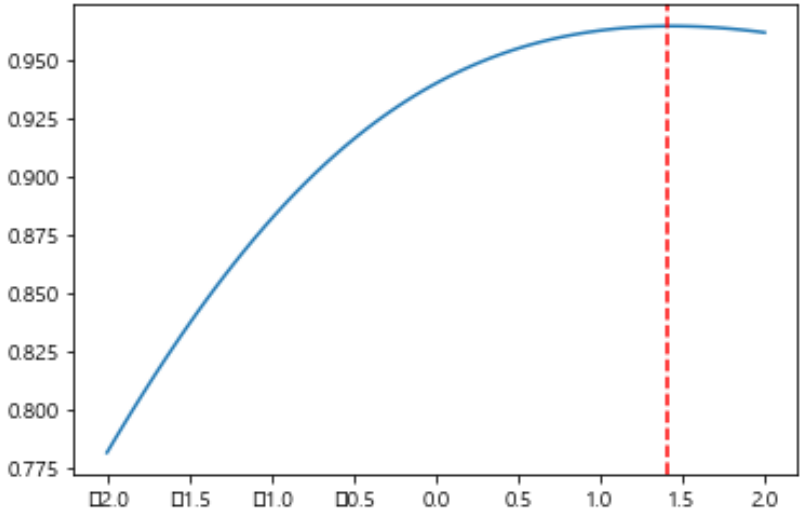
- 사용 전제 : **정규성 만족**
- LG 데이터의 SHAPIRO-WILK 정규성 검정

➔ p-value가 매우 小(귀무가설 기각)

➔ for 정규성 만족

➔ box – cox 변환

➔ Optimal lambda 탐색 후 변환



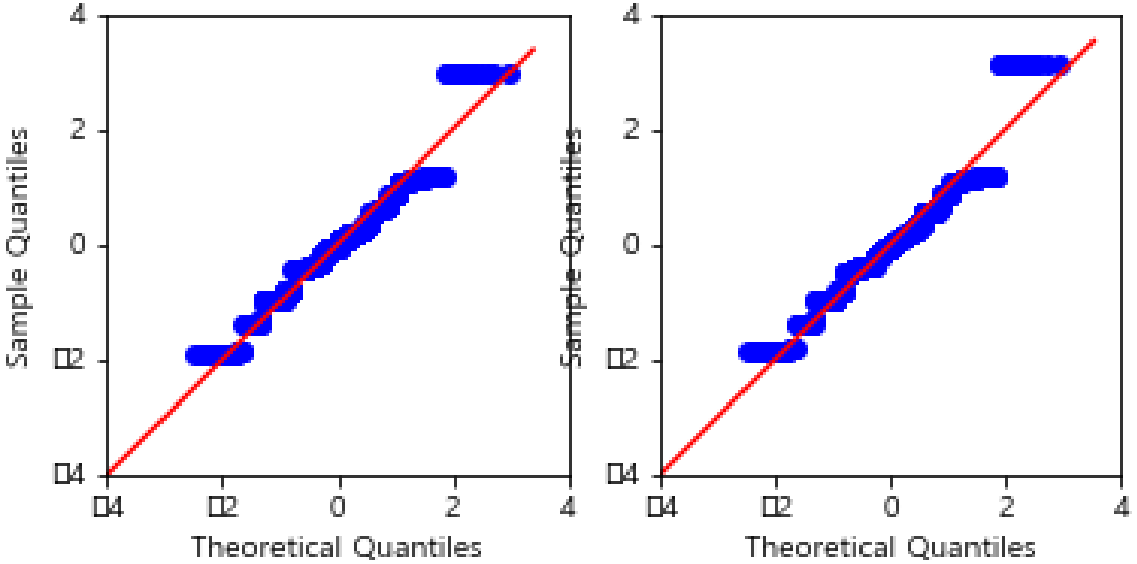
4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

I. Regression 모델 : ridge, Lasso, best subset selection, GLM

근거1



➔ 정규성을 만족하지 않음.

```
from scipy.stats import shapiro

shapiro(y2)
#optimal lambda에서 box-cox변환 취해도, shapiro wilk test 결과 정규성을 만족하지 않는다.
```

```
(0.9315124154090881, 2.7151434659955704e-16)
```

4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

1. Regression 모델 : ridge, Lasso, best subset selection, GLM

2. Our Task : 예측 > 해석

선호 모델 : non- parametric model > parametric model

예측에 용이 해석에 용이

3. Target value : 연속적 x, 범주적

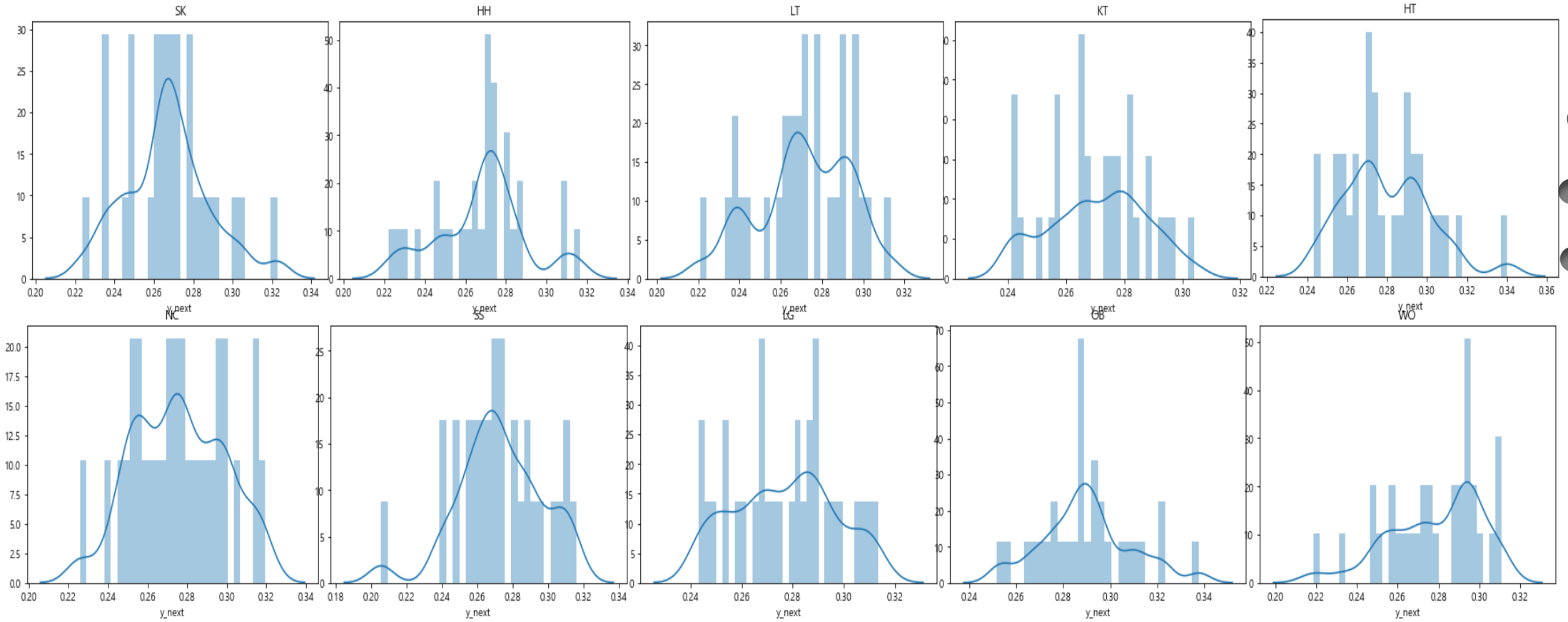
➔ exponential family에 속하는 특정 분포를 띈다고 가정하기 어려움.

➔ GLM 등 parametric model 사용 어려움.

4. EDA 및 모델 선정 검증 과정

모델 후보

<보충> non- parametric model 사용 근거



➔ 범주형에 가까움. Y의 분포에 대한 가정을 세우기 어렵다고 판단함.

4. EDA 및 모델 선정 검증 과정

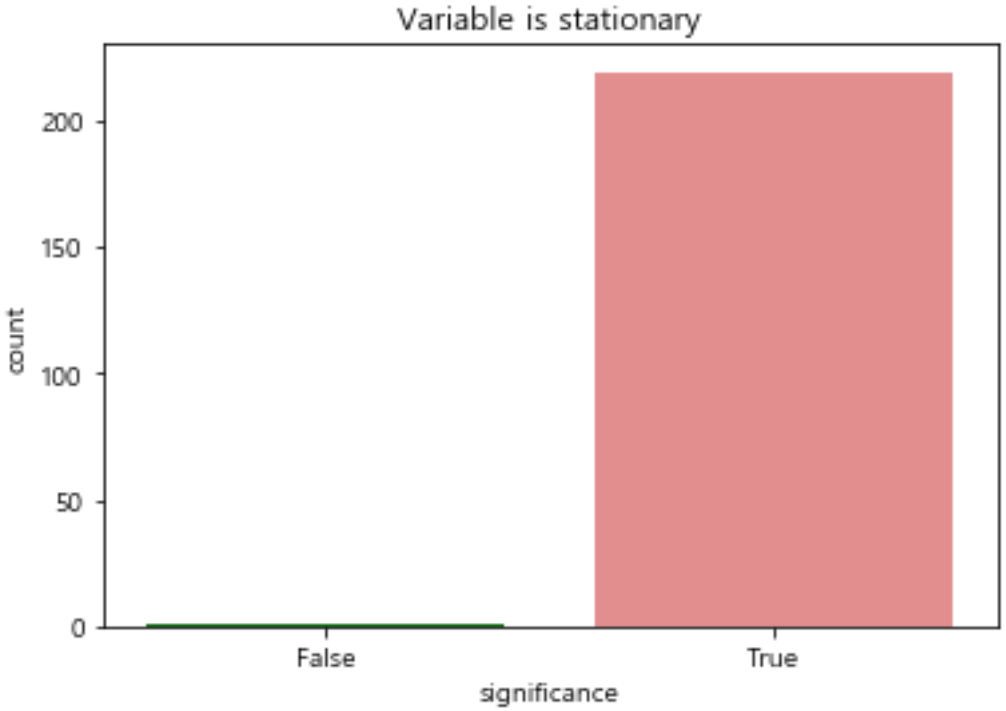
모델 후보

Parametric Model

II. 시계열 모델 : VAR, ARIMA, SARIMA, FBprophet

1. 정상성 검정 시행 (모든 팀별 데이터, 모든 칼럼)

근거1



4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

II. 시계열 모델 : VAR, ARIMA, SARIMA, FBprophet

1. 정상성 검정 시행 (모든 팀별 데이터, 모든 칼럼)

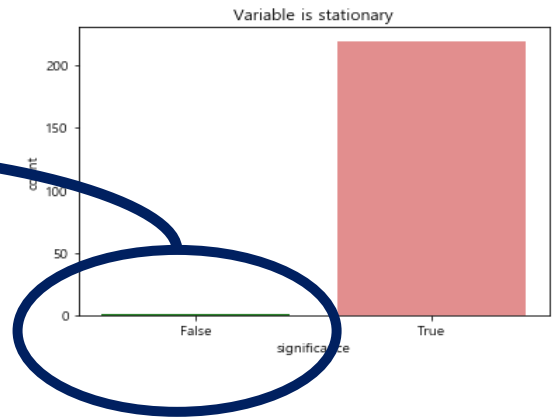
```
In [45]: df[df['significance']==False]['p-value']

Out[45]: 147    0.05187
         Name: p-value, dtype: float64
```

: 변수 1개 제외 모두 정상성 만족

: 변수 1개조차 유의수준과 큰 차이 없음.

➔ **시계열 모델 적합**



4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

II. 시계열 모델 : VAR, ARIMA, SARIMA, FBprophet

2. 예측력 비교 : 다변량 시계열 > 단변량 시계열

➔ 다변량 시계열, 변수 상관관계 고려 가능한 VAR 모델 적합하다고 판단함.

➔ Grangercausality test 진행

y_y	1.0	0.1068	0.3104	0.0916	0.2107	0.3077	0.4547	0.0265	0.0660	0.0955	...	0.1987	0.0856	0.1591	0.0278	0.1071	0.0340

➔ p값 추출 가능

* p값 : y에 대한 나머지 변수들의 인과성 여부

4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

II. 시계열 모델 : VAR, ARIMA, SARIMA, FBprophet

2. 예측력 비교 : 다변량 시계열 > 단변량 시계열

➔ VAR 모델에 사용될 변수 후보군 선정

1) p값이 0.05보다 작은 수

2) 공적분 여부 검정

```
#인과관계 있는 애들만 공적분 검정
cointegration_test(team_HH[['y','RBI' ,'H3','P_HRA_RT','P_HIT_CN']])
```

```
Name    ::  Test Stat > C(95%)    =>   Signif
 ----------------------------------------
y       ::  362.19    > 60.0627   =>   True
RBI     ::  243.18    > 40.1749   =>   True
H3      ::  144.47    > 24.2761   =>   True
P_HRA_RT ::  64.48     > 12.3212   =>   True
P_HIT_CN ::  0.9       > 4.1296    =>   False
```

4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

II. 시계열 모델 : VAR, ARIMA, SARIMA, FBprophet

3. 시차 고려 가능

: AIC를 기준으로 선정

: lag가 너무 큰 값을 갖지 않도록 조정

```
model =sm.tsa.VAR(df_train)
for i in [60,61,62,63,64,65,66,67,68]:
    result = model.fit(i)
    print('Lag Order =', i)
    print('AIC : ', result.aic)
    print('BIC : ', result.bic)
    print('FPE : ', result.fpe)
    print('HQIC: ', result.hqic, '₩n')
```

```
Lag Order = 60
AIC :  -8.30716122814719
BIC :  3.3082219497702017
FPE :  0.0004621080014929853
HQIC:  -3.772007939854241

Lag Order = 61
AIC :  -8.335172695045834
BIC :  3.489368874785715
FPE :  0.0004681577708710447
HQIC:  -3.717992023816368

Lag Order = 62
AIC :  -8.32009015310964
BIC :  3.714189687241305
FPE :  0.0004963277382370691
HQIC:  -3.6206422965933207

Lag Order = 63
AIC :  -8.286187170642986
BIC :  3.958413475526317
FPE :  0.0005375136728504313
HQIC:  -3.504231206998515
```

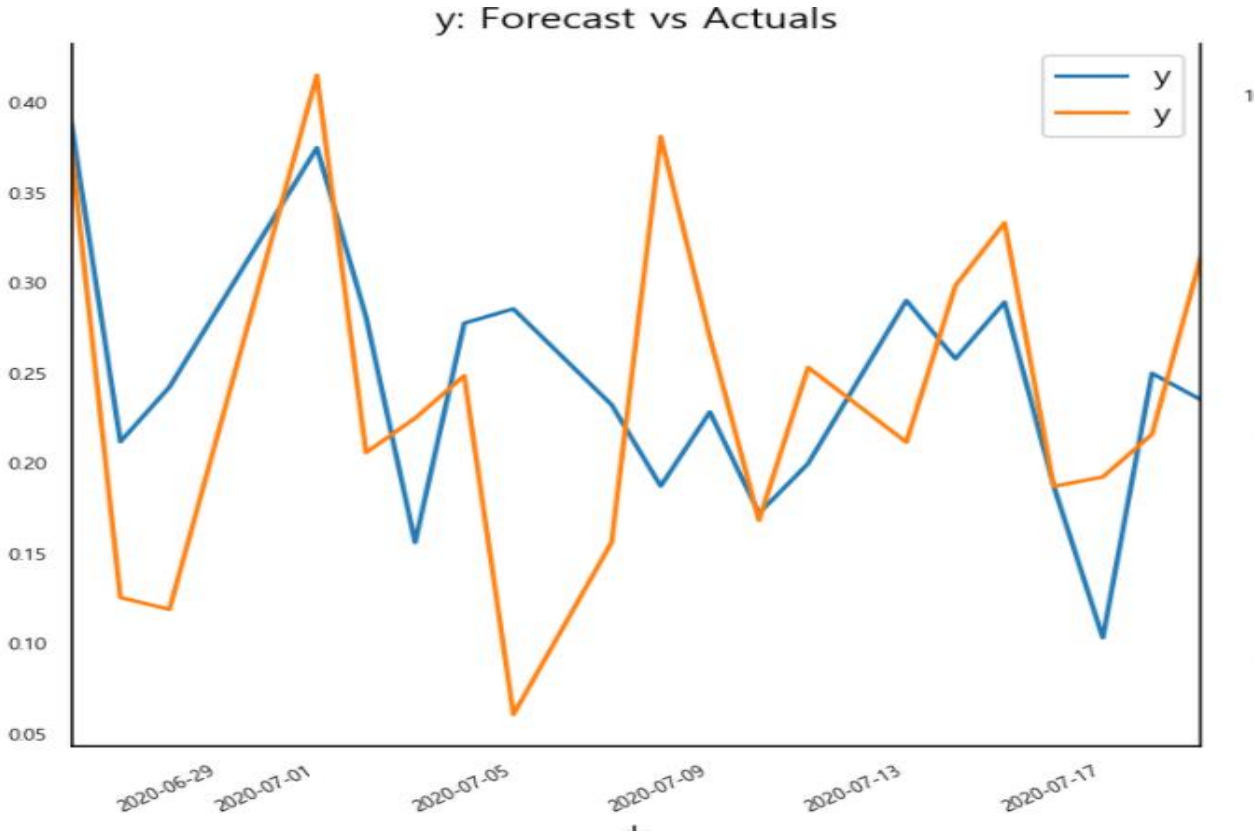
4. EDA 및 모델 선정 검증 과정

모델 후보

Parametric Model

II. 시계열 모델 : VAR, ARIMA, SARIMA, FBprophet

4. 성능 확인



4. EDA 및 모델 선정 검증 과정

모델 후보

Non - Parametric Model

III. SVR

1. 데이터의 특성에 따라 커널 설정 가능함.

: 다양한 데이터셋에서 잘 작동하는 강력한 모델 .

: 현재 task) 차원의 수에 비해 데이터 수가 적음. -> 커널을 가우시안으로 설정.

: 고차원의 데이터에서도 잘 작동.

2. SVR은 시간이 오래 걸리는 것이 단점이지만, 해당 task는 데이터량이 적어 적합.

3. Scaling을 통한 성능 향상 기대

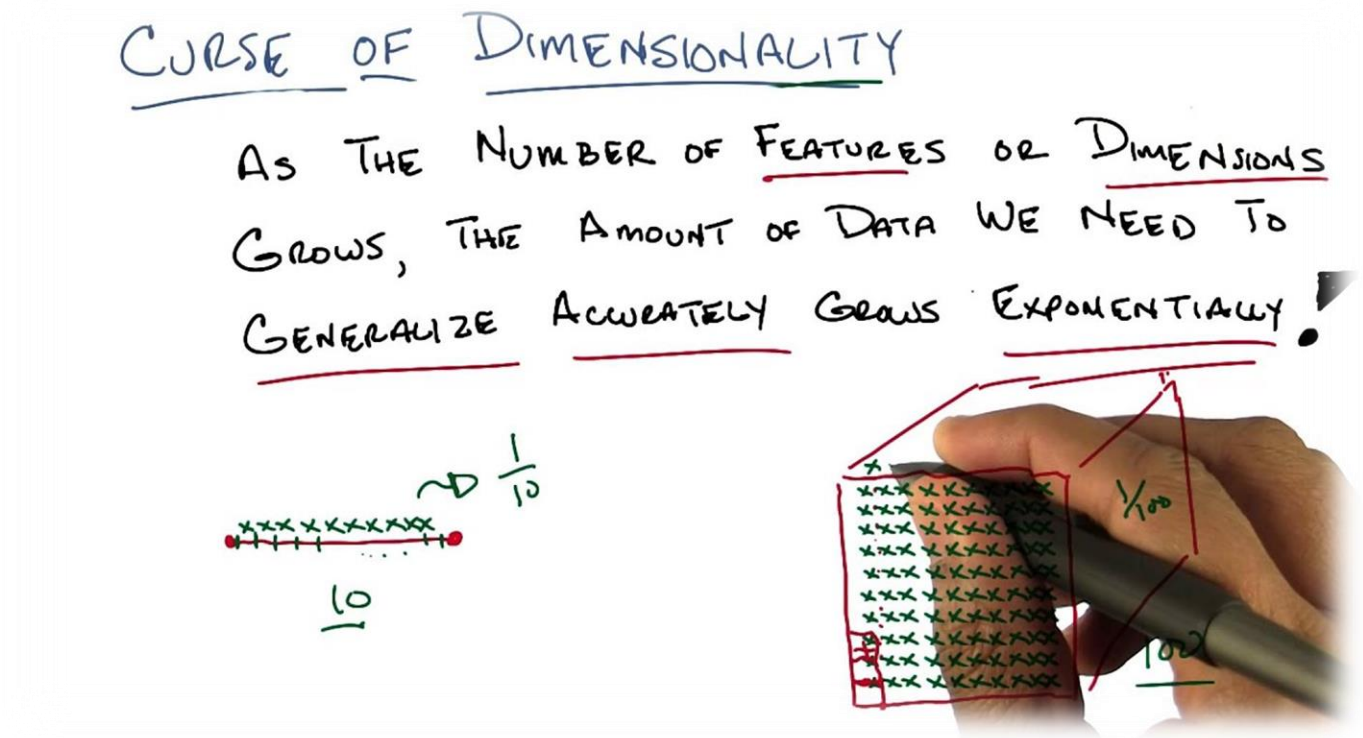
: SVR은 Scale에 민감하기에 적절한 처리 수행 가능

4. EDA 및 모델 선정 검증 과정

모델 후보

Non - Parametric Model

IV. KNN



관측치 개수 << 독립변수의 개수 ➔ 차원의 저주 우려

4. EDA 및 모델 선정 검증 과정

모델 후보

Non - Parametric Model

V. Tree 모델 : Random Froest, LGBM

1. Model variance를 잡아줄 수 있음.

: 단일 tree model은 관측치 개수가 적어 model variance를 잡기 어렵지만, Boosting 계열 모델은 model variance 잡아줄 수 있음.

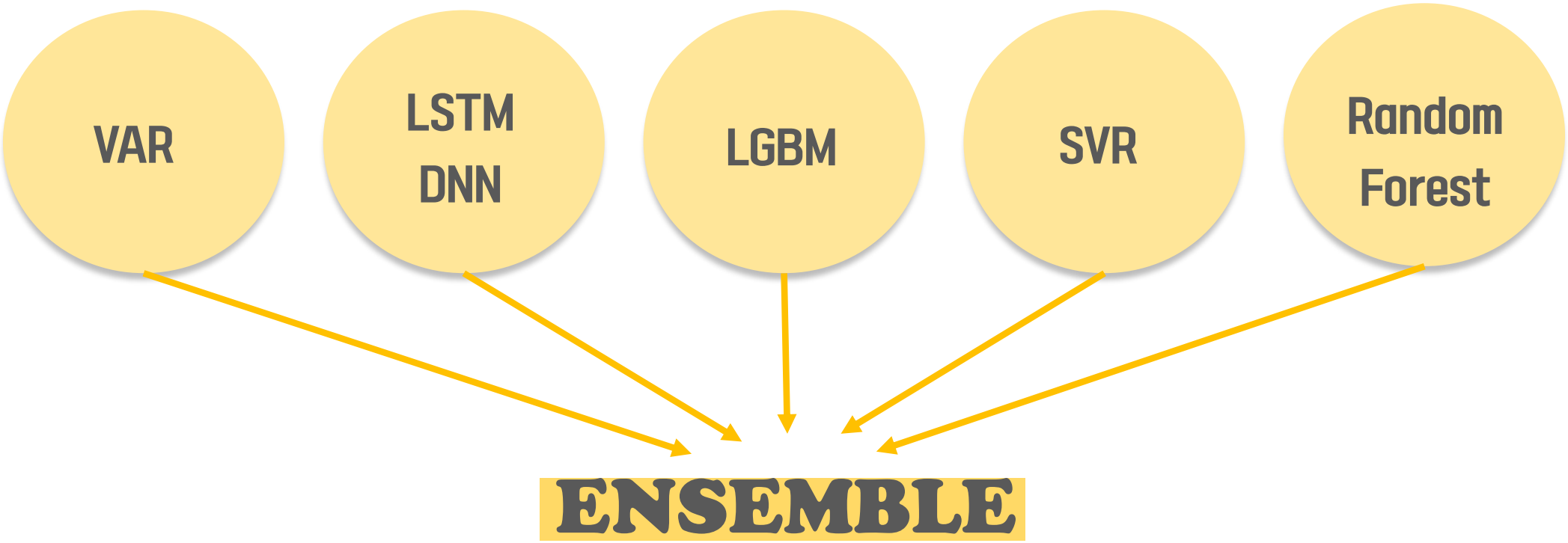
2. Model flexibility 규제 가능

: 적은 양의 데이터에 대해 과적합이 발생할 수 있는 단점 존재.

But, parameter tuning을 통해 규제 가능

5. 모델 구축 및 훈련

최종 모델



5. 모델 구축 및 훈련

VAR

: 다변량 시계열 모델

: 시간의 변화에 따른 결과 예측 가능

: 내생변수와 외생변수 구분 불필요

: 방어율과 타율 inference에 사용

Modeling

```
class Var_automation:
    def __init__(self, df, target, predict_range, mode):
        super(Var_automation, self).__init__()
```

EDA 후 Dataframe

Target 변수 (ERA, AVG)

예측하고 싶은 경기 수

Train or Inference

5. 모델 구축 및 훈련

VAR

<EDA>

: 방어율, 타율 별 EDA 코드 작성

: 각 팀 별 2016-2020년 팀 데이터 합치기

: 아웃라이어 제거

: 시계열 모델 사용하기 위하여 각 경기별 날짜, 시간에 Iindex 부여

: 트랜드 feature 추가

: categorical feature 제외

(모델 검증 과정에서 singular matrix 문제 발생)

5. 모델 구축 및 훈련

VAR

<Modeling 과정>

1) 자기상관 검정 : 자기상관이 낮음을 확인
2) 인과관계 검정 : grangers_causation_matrix 함수를 통해 유효한 변수 리턴
3) 공적분 검정 : cointegration_test 함수를 통해 유효한 변수 리턴
4) 단위근 검정 : adfuller_test 함수를 통해 유효한 변수 리턴
5) AIC test : aic_test 함수를 통해 최적 lag(시차) 리턴
6) Inference : run함수를 통해 var모델이 예측한 값의 평균 리턴
7) Class 자동화 : class를 활용해 모든 검정 과정 및 inference 과정을 자동화

5. 모델 구축 및 훈련

LSTM, DNN

<LSTM>

: AVG, OPS 예측

- 실험 대상 모델 : bidirectional, stateful, stack, state_stack
- Grid Search : look_back (3, 5, 10) / lstm layter (5, 10, 15)

➔ 최적 모델, 최적 파라미터 탐색 완료

➔ Stateful, look_back : 5, layer : 5

<DNN>

: 승률 예측

- Input Feature : AVG , OPS, team_code

5. 모델 구축 및 훈련

LGBM, SVR, Random Forest

<모델링 과정>

1) 모델 선정
2) 범주형 변수 지정
3) Train, Validation set 비율 지정 (0.85:0.15)
4) 주요 파라미터 지정
5) Grid Search -> 최적 파라미터 탐색 (팀별)
6) 최종 model fit

```
# 파라미터 지정

lgb_params_grid = {
    'num_leaves': [5,10,20,30],
    'min_data_in_leaf': [10,30, 50, 100],
    'lambda_l1': [0, 1, 1.5],
    'lambda_l2': [0, 1]
}

svr_params_grid = {
    'kernel':['rbf','poly'],
    'degree':[2,3,4,5,6,7],
    'epsilon':[0.1,0.2,1,10,20,30]
}

rf_params_grid = {
    'n_estimators': [10, 100],
    'max_depth': [6,8,10,12],
    'min_samples_leaf': [8,12,18],
    'min_samples_split': [8,16,20]
}
```

6. 결과 분석

최종 결과 도출

$$\frac{\text{VAR} + \text{LSTM} + \text{LGBM} + \text{SVR} + \text{RF} + \text{2020년 데이터}}{6}$$

* RF = Random Forest
* 가장 최신 데이터 반영

6. 결과 분석

최종결과

	예측 방어율	예측 타율	예측 승률
WO	4.567643	0.270651	0.550921
OB	4.506169	0.284679	0.54775
NC	4.643764	0.278741	0.59196
SK	5.427254	0.257827	0.410641
LG	4.62773	0.274902	0.492944
KT	4.918385	0.272521	0.502009
LT	5.027182	0.271161	0.488329
HT	5.307015	0.27756	0.45959
SS	5.322334	0.268757	0.468494
HH	5.200824	0.256868	0.39356

7. 결론 및 느낀 점

모델 별 성능 비교

< EX. HH(한화)데이터>

	ERA
LGBM	5.297633
RF	5.074899
SVR	5.470508
VAR	5.264835
LSTM	4.846072
2020	5.250997

	AVG
LGBM	0.246836
RF	0.255581
SVR	0.264975
VAR	0.264095
LSTM	0.275041
2020	0.234677

	ODDS
LGBM	0.333691
RF	0.383767
SVR	0.402271
DNN	0.454513

7. 결론 및 느낀 점

기대효과

선수 영입과 합리적인 연봉 협상의 근거

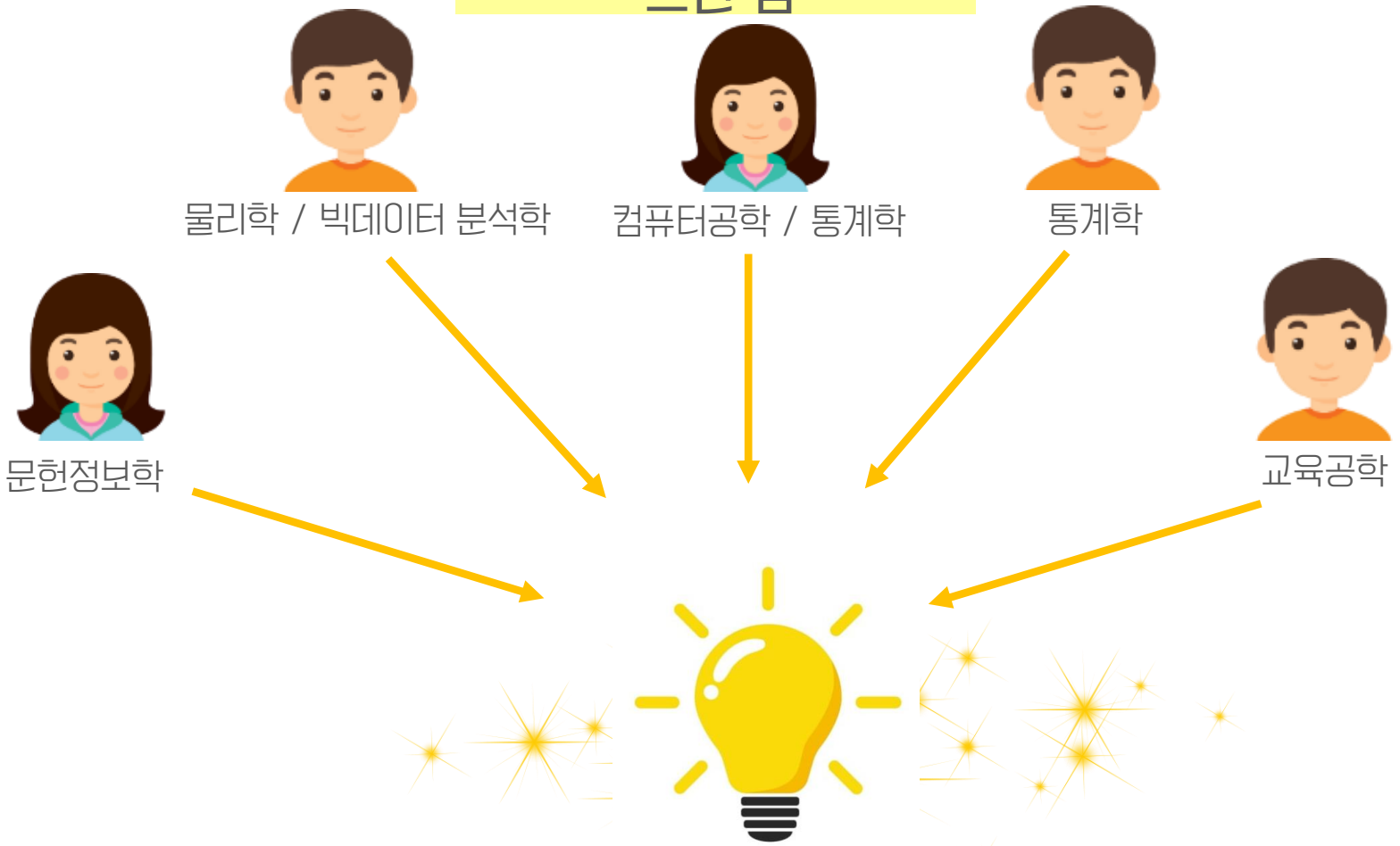
약점 발견 , 개선 가능

팬들의 경기 몰입감 극대화

구단의 포스트시즌 준비 가능

7. 결론 및 느낀 점

느낀 점



'데이터 분석 ' 이라는 공통의 관심사를 공유하고 있는 다양한 학과의 전공생들이 함께 프로젝트를 진행하며,
다채로운 관점과 **융합적 연구**의 중요성을 크게 배웠습니다.

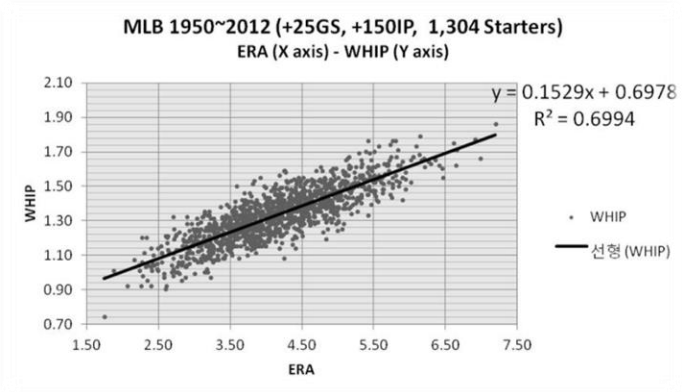
7. 결론 및 느낀 점

느낀 점

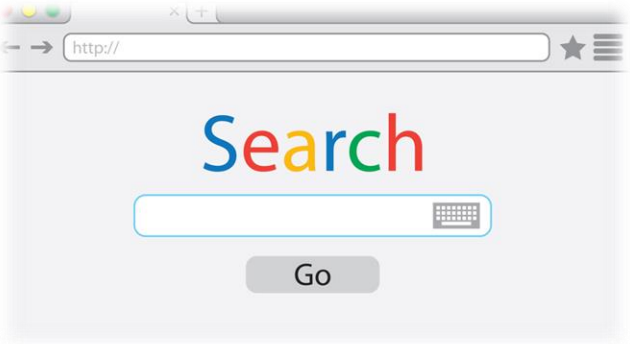
데이터 분석부터 전처리, 모델 선정, 모델 설계 등의 **모든 단계를 직접 수행**하며
단순히 이론 공부에서 그치지 않고, **적용을 통한 실질적 경험**을 쌓을 수 있었습니다.

8. 향후계획

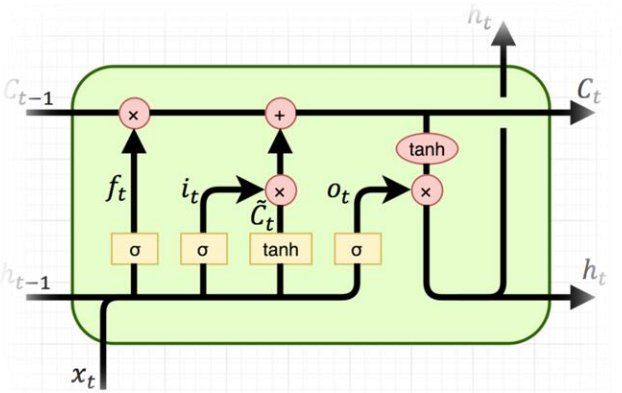
아쉬운 점 및 향후계획



WHIP 계열 변수 추가하기



9/20 이후 데이터 중 구하지 못한
특정 변수들 추가 탐색



LSTM 모델의 한계점 극복 노력
(먼 미래를 예측할수록 성능이 감소하는 문제점)

2016년 이전 데이터 추가 탐색

참고문헌

1. 논문

: Harris Drucker, Chris J.C. Burges, Linda Kaufman, Alex Smola, Vladimir Vapnik, "
: Support Vector Regression Machines", NIPS, 1997.
: Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu,
:"LightGBM: A Highly Efficient Gradient Boosting Decision Tree", NIPS, 2017.
: LEO BREIMAN, "Random Forests", 2001.
: 기계학습 기법을 이용한 한국프로야구 승패 예측 모델(2019) 서영진,문형우,우용태
: 2016~2018 한국프로야구 세이버 메트릭스 지표 분석(2019) 권순규, 이규원, 최형준

2. 데이터 자료

: http://www.kbreport.com/teams/main
: http://www.statiz.co.kr/main.php
: http://cpbpoint.mbcplus.com/

3. 이미지 및 png 자료

: <a href='https://pngtree.com/so/아이디어'>아이디어 png from pngtree.com</a>
: <a href='https://pngtree.com/so/여러-가지-빛깔의'>여러-가지-빛깔의 png from pngtree.com</a>
: https://www.flaticon.com
: http://pptbizcam.co.kr/?p=5441

2020 BIG CONTEST

감사합니다

Team _ Big Contact

남상대 (팀장) nsd96@naver.com

김동현 ddhh445@naver.com

김민영 kmy0433@naver.com

이윤정 lyj000105@naver.com

임창건 ckdrjs96@gmail.com