리그 오브 레전드
15분 시점 승패 예측

발표자
AI_16 조예찬

발표일자
2022-12-05

로그인 상태 유지

목차

X 확인

15분 승패 예측

프로젝트 목표 제시

문제 의식

리그 오브 레전드는 15분 부터 항복 투표를 할 수 있음

이기지 못할 게임이라면 항복을 하는 것이 체력과 정신력을 아낄 수 있는 방법임

목표 제시

- 15분에 확인할 수 있는 정보를 통해 게임 승패를 예측하여 항복 투표에 도움을 주는 예측 모델을 만듬
- 게임의 승패에 영향을 끼치는 주요 특성들을 파악하여 게임을 이기려면 어떤 점에 집중 해야 하는지 확인

X 확인

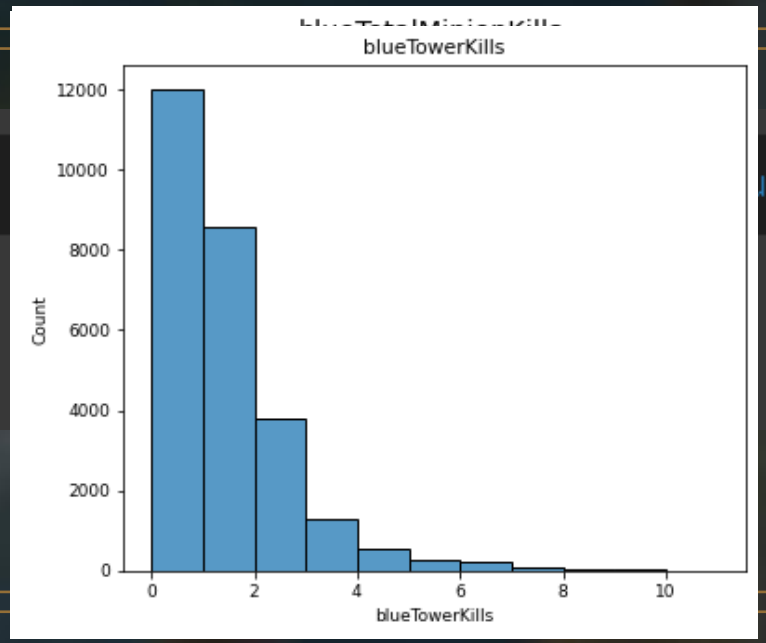
데이터 수집 및 가공

데이터 수집

챌린저 랭크 게임 26779판의 15분 시점 데이터

컬럼 설명

Target	blueWins	블루 팀의 승패 여부
	blueKill	블루 팀의 총 처치 수
	blueDeath	블루 팀의 총 사망 수
	blueWardPlaced	블루 팀의 총 와드 설치 횟수
	blueTotalLevel	블루 팀의 총 레벨 합
	blueTotalMinionKills	블루 팀의 총 미니언 처치 수
	blueTowerKills	블루 팀의 총 타워 처치 수



X 확인

데이터 수집 및 가공

EDA

redKill	–〉 blueDeath와 같은 값이기 때문에 삭제
redDeath	–〉 blueKill과 같은 값이기 때문에 삭제
blueFirstTower, redFirstTower	–〉 각각 Boolean으로 되어있는 값을 하나의 컬럼으로 합침 (‘None’,’Blue’,’Red’)
	FirstInhibitor, FirstDragon 또한 하나의 컬럼으로 합침
blueDragon, redDragon	–〉 Data Leakage를 보여 삭제
redWins	–〉 Target과 반대의 값. 삭제

이후 데이터셋을 분할하여 Train, Validation, Test Set로 만듦

15분 승패 예측

가설 설정

1. 와드 설치수는 게임의 승리에 있어서 가장 중요한 특성일 것이다.

리그오브레전드를 해본 사람이라면 와드 설치의 중요성에 대해 귀에 딱지가 앉도록 들었을 것이다. 적의 위치를 정확히 파악하고 동선을 제한시키면서 본인의 활동범위는 확보할 수 있는 중요한 수단으로써 와드는 사용되고 있기 때문이다. 따라서 와드의 설치수가 게임의 승리에 가장 큰 영향을 끼칠 것이라는 가설을 세워보았다.

1. 아니다, 게임의 승리에는 킬과 데스가 더욱 중요할 것이다.

와드를 통해 전략적인 정보를 얻어내는 것도 중요하지만 결국 모든 데이터는 킬과 데스로 표현되기 마련이다. 용과 전령같은 오브젝트를 얻어내는 것도, 타워를 부수는 것도, 미니언을 많이 잡아내는 것도 결국에는 킬과 데스로 귀결되므로 킬과 데스가 가장 중요한 특성일 것이다.

X 확인

15분 승패 예측

머신러닝 모델링

기저모델 설정

최빈값: 0.500

```
y_train.value_counts(normalize=True).max()
```

```
0.5000583498657953
```

모델 설계 및 학습

- 승패를 예측하는 것이므로, 분류모델을 이용한다. (XGBooster Classifier)
- 모델의 성능을 평가하기 위해 Accuracy, F1 score, AUC score를 활용하도록 한다.
- Cross Validation을 통해 모델의 안정적인 성능을 확보한다.
- Validation Set을 이용해 모델의 성능을 비교한다.
- Randomized CV와 HyperOpt를 활용해 하이퍼파라미터 튜닝을 진행하고 비교하였다.

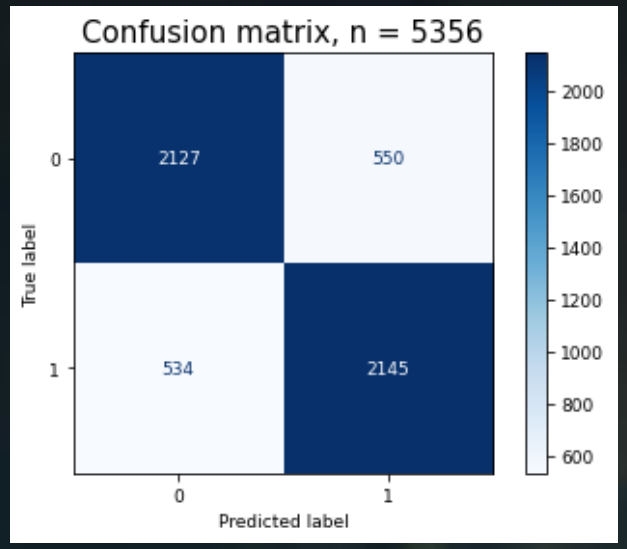
```
def make_pipe(X_train, y_train):

    pipe = make_pipeline(
        OrdinalEncoder(),
        XGBClassifier(n_estimators=1000
                      , random_state=42
                      , n_jobs=-1
                      # , early_stopping_rounds=50
                     )
    )

    # 튜닝할 하이퍼파라미터의 범위를 지정해 주는 부분
    dists = {
        'xgbclassifier__max_depth': [6,7],
        'xgbclassifier__learning_rate':[0.03,0.04]
    }

    clf = RandomizedSearchCV(
        pipe,
        param_distributions=dists,
        n_iter=4,
        cv=5,
        scoring='f1',
        verbose=1,
        n_jobs=-1,
        random_state=42
    )

    clf.fit(X_train, y_train)

    return clf
```

X 확인

머신러닝 모델링

성능검증

Confusion Matrix



Accuracy

0.797

```
accuracy_score(y_test, y_test_pred)
0.7976101568334578
```

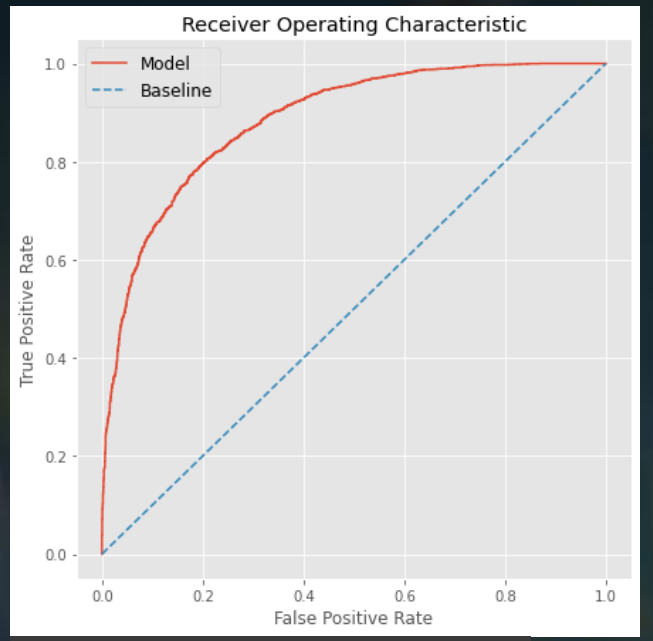
F1 score

0.798

```
f1_score(y_test, y_test_pred)
0.7982880535913659
```

$$F1\ Score = 2 \times \frac{recall \times precision}{recall + precision}$$

ROC_AUC score 0.884




```
print("Validation AUC:", auc_score)
Validation AUC: 0.8846512039084828
```

기준모델보다 더 좋은 성능을 보이는 모델이다.

X 확인

머신러닝 모델링

특성 중요도

Permutation Importance

한 특성 내부의 데이터를 무작위로 배열하여 그 특성의 영향을 없앤 상태와 기존 모델의 결과를 비교하는 방법

그렇게 하여 특성의 중요도를 평가할 수 있게 된다.

blueDeath	0.037828
blueKill	0.035703
redTotalLevel	0.035019
blueTotalLevel	0.027908
blueTotalMinionKills	0.022844
blueAssist	0.009021
redTotalMinionKills	0.006388
blueWardPlaced	0.005556
blueTowerKills	0.005089
redAssist	0.004608
redWardPlaced	0.003572
redTowerKills	0.003298
redCurrentGolds	0.003122
redTotalJungleMinionKills	0.003052
blueTotalJungleMinionKills	0.001977
redMidTowerKills	0.001930
blueRiftHeralds	0.000865
blueWardKills	0.000752
redInhibitor	0.000728
blueInhibitor	0.000407
redBotTowerKills	0.000211
redTopTowerKills	0.000040
FirstTower	0.000000
FirstInhibitor	0.000000
FirstDragon	0.000000
redFirstTowerLane	0.000000
blueFirstTowerLane	0.000000
blueMidTowerKills	-0.000135
blueCurrentGolds	-0.000357
blueBotTowerKills	-0.000364
blueTopTowerKills	-0.001113
redRiftHeralds	-0.001161
redWardKills	-0.001301

dtype: float64

1. blueDeath
2. blueKill
3. redTotalLevel
4. blueTotalLevel
5. blueTotalMinionKills

8. BlueWardPlaced

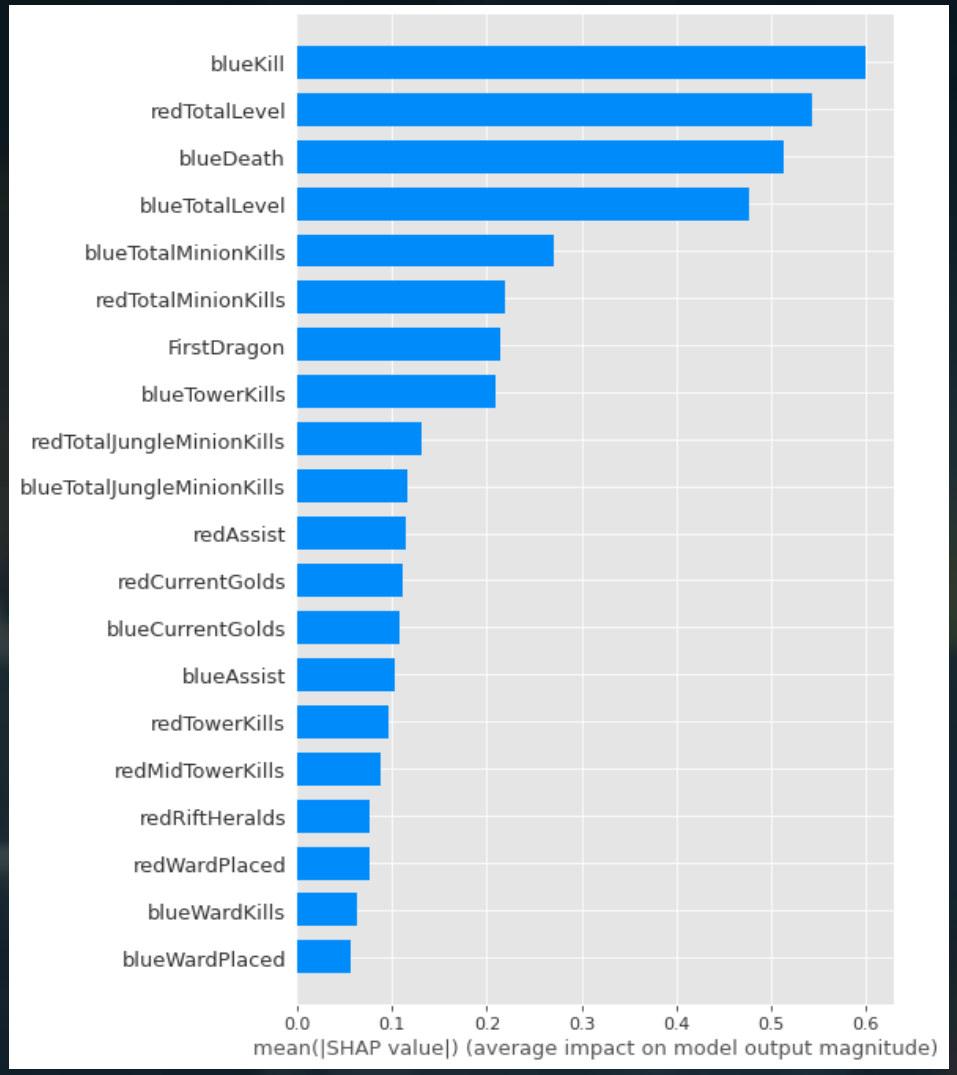
머신러닝 모델링

특성 중요도

SHAP

하나의 샘플 내에서 각 특성이 결과에 얼마나 기여하는지를 나타내는 방법

특성의 각 샘플에서의 영향력을 평균 내어 전체적인 영향력을 평가할 수 있다.



1. blueKill
2. redTotalLevel
3. blueDeath
4. blueTotalLevel
5. blueTotalMinionKills

20. blueWardPlaced

머신러닝 모델링

특성 중요도

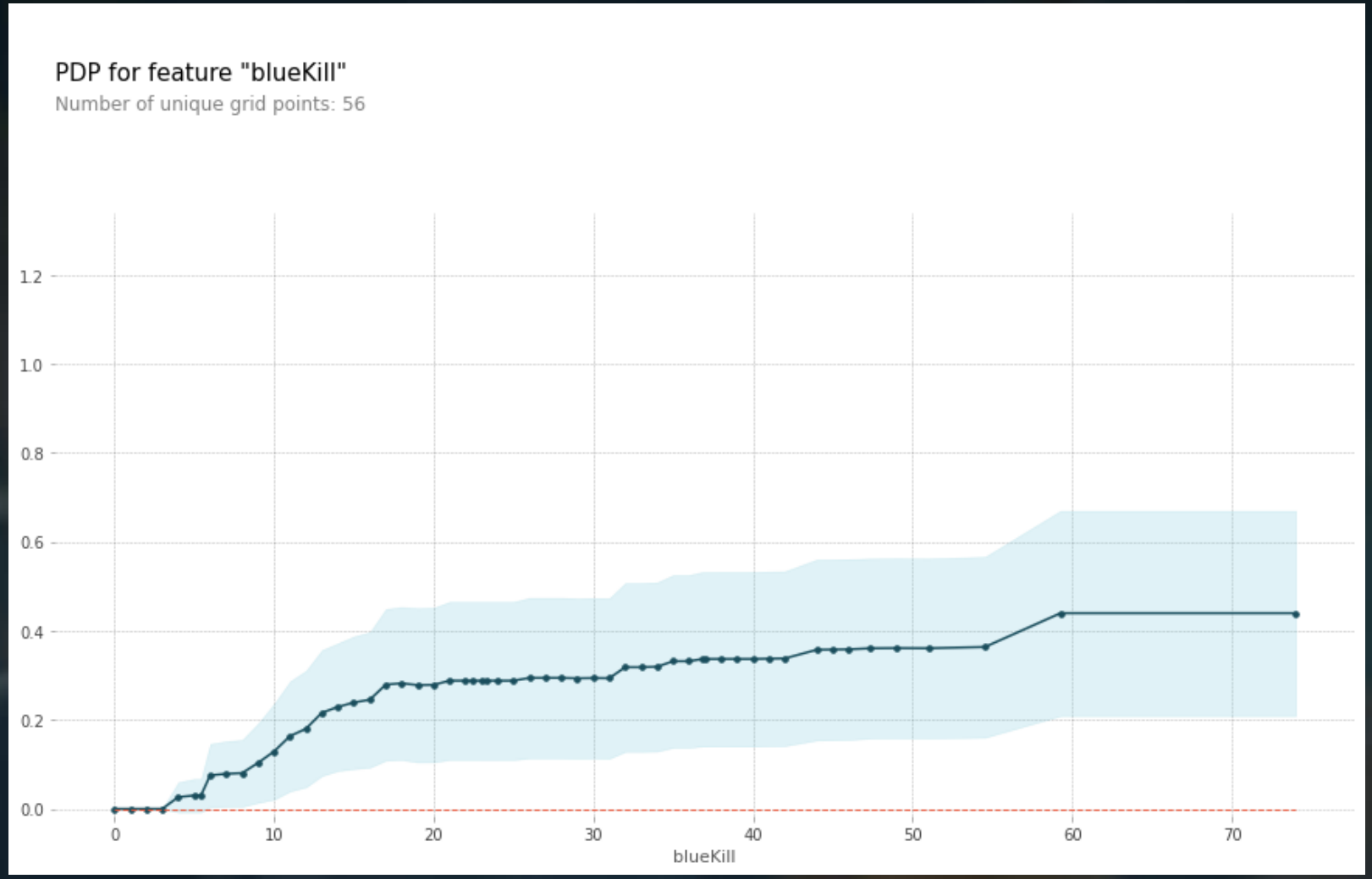
Partial Dependence Plot

한 특성이 타겟에 대해 어떤 영향을 끼쳤는지 알기 위한 그래프.

blueKill

20킬 전까지 가파르게 승률이 상승하다

그 이후로 완만해진 모습



머신러닝 모델링

특성 중요도

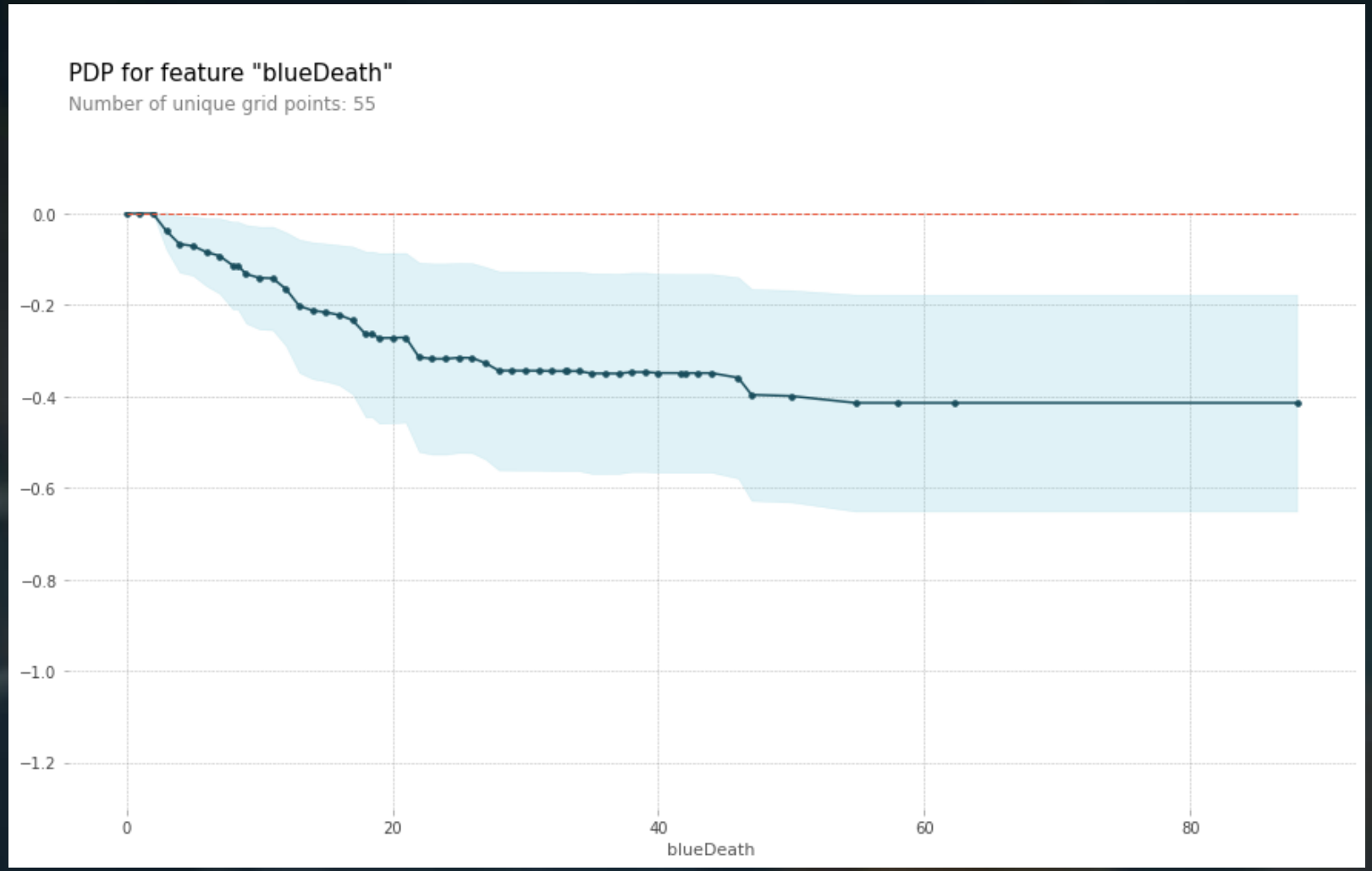
Partial Dependence Plot

한 특성이 타겟에 대해 어떤 영향을 끼쳤는지 알기 위한 그래프.

blueDeath

음수로 부정적인 영향.

blueKill과 마찬가지로 20까지는 기울기가 가파르다가 그 이후로 완만해짐



머신러닝 모델링

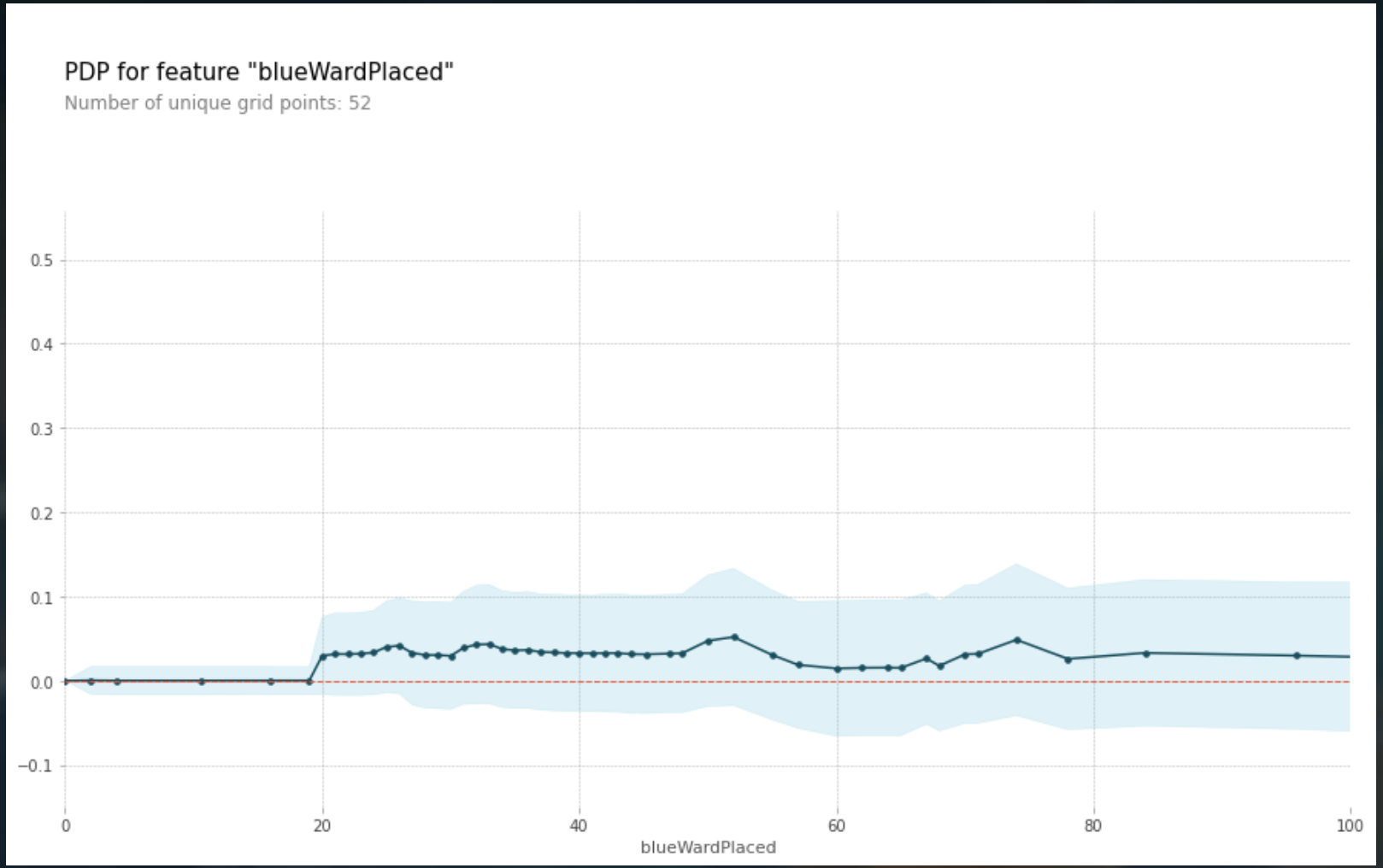
특성 중요도

Partial Dependence Plot

한 특성이 타겟에 대해 어떤 영향을 끼쳤는지 알기 위한 그래프.

blueWardPlaced

앞선 두 그래프와는 다르게 그닥 타겟에 영향을 끼치지 못하는 모습.



15분 승패 예측

가설 검정

가설 1. 와드 설치수는 게임의 승리에 있어서 중요한 특성 중 하나일 것이다.

가설 2. 아니다, 게임의 승리에는 킬과 데스가 더욱 중요할 것이다.

모델은 Accuracy: 80%, F1 score: 0.80, AUC score: 0.88 을 보이는 준수한 모델이다.

Permutation score

1. blueDeath
2. blueKill
3. redTotalLevel
4. blueTotalLevel
5. blueTotalMinionKills

-> 가설 2를 지지

SHAP

1. blueKill
2. redTotalLevel
3. blueDeath
4. blueTotalLevel
5. blueTotalMinionKills

-> 가설 2를 지지

Partial Dependence Plot

blueKill과 blueDeath는 타겟에 대해 유의미한 변화를 보여주는 반면, blueWardPlaced는 유의미한 변화를 보여주지 않음.

-> 가설 2를 지지

따라서,

가설 1을 기각하고

가설 2를 채택한다.

X 확인

결론

결과 해설

킬과 데스가 중요하다는 결론은 사실 기존의 통념에서 크게 벗어나지 않는다.
와드의 수가 중요하게 작용하지 않는다는 사실은 놀랍지만 이해할만 하다. 일반 와드의 개수는 서포터를 제외하고는 모두 동일하게 주어지고, 와드의 설치 '양'보다는 와드의 설치의 '질'이 더욱 중요하기 때문이다.
앞서 설계한 모델을 통해 항복 여부를 판단하는데 도움을 줄 수 있다.
현재 특성의 상황에 따라 적절한 조언을 해주던지 (ex. 레벨을 올리는데 초점을 맞추세요!, 이번 오브젝트는 중요합니다! 등) 너무 승률이 낮으면 항복을 권유하는 시스템을 만들 수도 있을 것 같다.

한계점

리그 오브 레전드에는 여러 컨셉의 챔피언들이 있다. 이번에 사용한 데이터는 각 챔피언들의 특성을 담아내지 못한다.