Prédiction de match NBA

Réalisé par :

FURAIJI Dalya
ZERZERI Nabil

Sommaire

- Introduction
- DataSet
- Nettoyage
- Train / Validation
- Algo / Modele
- Résultats de prédictions
- Interprétation des résultats
- Discussion
- Code
- Contributions

Introduction

Qui n’a jamais tenté de prédire le résultat d’un match ? La NBA propose des centaines de matchs tout les ans, de quoi rêver de fortune si on peut prédire le résultat des matchs.

Nous allons donc tenter de produire un algorithme de machine learning qui nous donnera les résultats des matchs auxquels nous allons parier.

DataSet

La base de données :

https://www.kaggle.com/pablote/nba-enhanced-stats

2 fichiers csv :

- 2017-2018

  4920 lignes

  123 colonnes

- 2015-2017

  2460 lignes

  123 colonnes

Nettoyage

Pas de nettoyage car :

- Pas de valeur manquante
- Pas d’entrée dupliquée

Algo / Modèle

- Random Forest
- K nearest neighbors
- SVM
- Gradient Boosting Clasifier

Train / Validation

Nous avons utilisé la fonction train_test_split de sklearn.model_selection avec 60% train et 40% test.

Nous utilisons ensuite ces (x et y) test et train pour calculer le score de nos modèles.

Résultats de prédictions

```
2                       1                       1                       1                       1
[[0.31 0.69]]           [[0.76 0.24]]           [[0.61 0.39]]           [[0.62 0.38]]           [[0.64 0.36]]
2                       1                       1                       1                       1
[[0.28399549 0.71600451]] [[0.59006072 0.40993928]] [[0.65928791 0.34071209]] [[0.60068569 0.39931431]] [[0.65247942 0.34752058]]
2                       1                       1                       1                       1
[[0.48333333 0.51666667]] [[0.5 0.5]]           [[0.51666667 0.48333333]] [[0.58333333 0.41666667]] [[0.56666667 0.43333333]]
Match PHO vs SA         Match GS vs MIN         Match IND vs MEM        Match OKC vs HOU        Match MIA vs MEM
2                       1                       1                       1                       1
------------------      ------------------      ------------------      ------------------      ------------------
2                       1                       1                       2                       1
[[0.27 0.73]]           [[0.71 0.29]]           [[0.73 0.27]]           [[0.43 0.57]]           [[0.62 0.38]]
2                       1                       1                       2                       1
[[0.29076451 0.70923549]] [[0.71407582 0.28592418]] [[0.59070706 0.40929294]] [[0.34965751 0.65034249]] [[0.60428416 0.39571584]]
2                       1                       1                       2                       1
[[0.43333333 0.56666667]] [[0.61666667 0.38333333]] [[0.63333333 0.36666667]] [[0.43333333 0.56666667]] [[0.61666667 0.38333333]]
Match PHI vs MIA        Match UTA vs SAC        Match TOR vs NO         Match DEN vs LAC        Match GS vs OKC
1                       1                       1                       2                       2
------------------      ------------------      ------------------      ------------------      ------------------
1                       2                       2                       1                       1
[[0.51 0.49]]           [[0.43 0.57]]           [[0.48 0.52]]           [[0.78 0.22]]           [[0.53 0.47]]
2                       2                       1                       1                       1
[[0.49581362 0.50418638]] [[0.39432092 0.60567908]] [[0.60387799 0.39612201]] [[0.64684853 0.35315147]] [[0.6302287 0.3697713]]
1                       1                       1                       1                       1
[[0.6 0.4]]             [[0.53333333 0.46666667]] [[0.56666667 0.43333333]] [[0.53333333 0.46666667]] [[0.56666667 0.43333333]]
Match CLE vs BOS        Match WAS vs DET        Match CHI vs SA         Match ATL vs NY         Match IND vs CHI
1                       1                       2                       1                       1
------------------      ------------------      ------------------      ------------------      ------------------
2                       1                       1                       2                       1
[[0.36 0.64]]           [[0.53 0.47]]           [[0.54 0.46]]           [[0.2 0.8]]             [[0.55 0.45]]
2                       2                       1                       2                       1
[[0.35720124 0.64279876]] [[0.43838532 0.56161468]] [[0.63578143 0.36421857]] [[0.3743633 0.6256367]]  [[0.68762449 0.31237551]]
2                       2                       1                       2                       1
[[0.45 0.55]]           [[0.46666667 0.53333333]] [[0.55 0.45]]           [[0.3 0.7]]             [[0.61666667 0.38333333]]
Match LAL vs LAC        Match POR vs MIL        Match DAL vs POR        Match BKN vs NY         Match CHA vs DAL
2                       2                       2                       2                       2
------------------      ------------------      ------------------      ------------------      ------------------
1                       1                       1                       1
[[0.76 0.24]]           [[0.61 0.39]]           [[0.62 0.38]]           [[0.64 0.36]]           C:\Users\nabza\Desktop\FAC
1                       1                       1                       1
[[0.59006072 0.40993928]] [[0.65928791 0.34071209]] [[0.60068569 0.39931431]] [[0.65247942 0.34752058]]
1                       1                       1                       1
```

Interprétation des résultats

Méthode	Précision	Prédiction
Gradient Tree Boosting	73%	13/20
KNN	73%	15/20
RandomForest	73%	14/20
SVM	58 %	NA

Discussion

Il est difficile de trouver un modèle de prédiction sans avoir assez d'informations sur les matchs.

On a donc essayé d'utiliser le moins d'informations révélatrices sur les résultats des matchs à prédire, mais on en a utilisé 6 (sur la centaine disponible).

C'est relativement correct, et on trouve un pourcentage de réussite assez intéressant pour un modèle de prédiction.

Code

```
33
34  def prepareXandY(df):
35      feature_cols = ['opptPTS', 'teamDrtg', 'teamPF', 'teamTO', 'teamORB', 'teamFGA']
36      x = df[feature_cols]
37      y = df['opptRslt']
38      x_train, x_test, y_train, y_test = train_test_split(x, y , test_size=0.4, random_state=2)
39      return x_train, x_test, y_train, y_test
40
41
42
43  def randomForest(df):
44      x_train, x_test, y_train, y_test = prepareXandY(df)
45      model = RandomForestClassifier(n_estimators = 1900, criterion = "gini", max_features = "sqrt", n_jobs=-1)
46      model.fit(x_train,y_train)
47      y_predict = model.predict(x_test)
48      print('accuracy of RandomForest :')
49      print(accuracy_score(y_test, y_predict))
50
51  def predictRandomForest(df,team1,team2):
52      x_train, x_test, y_train, y_test = prepareXandY(df)
53      clf = RandomForestClassifier()
54      clf.fit(x_train, y_train)
55      filename = 'nba_pred_modelv1.sav'
56      pickle.dump(clf, open(filename, 'wb'))
57      nba_pred_modelv1 = pickle.load(open(filename, 'rb'))
58
59      g1 = gameToPredict(df, team1, team2)
60      pred = nba_pred_modelv1.predict(g1)
61      prob = nba_pred_modelv1.predict_proba(g1)
62      if(pred[0]=='Win'):
63          print(2)
64      else:
65          print(1)
66      print(prob)
67
```

"te