Ｊ Leagueの観客数予測に挑戦！

林　健

1. 今回の目的

SignateのJリーグの観客数予測に挑戦してコンペに慣れること。
※実際のコードはGithubに載せるのでこのスライドでは基本的に省略
コード：https://github.com/khayashi0902/J_league.git

＜Jリーグの観客数予測について＞
2012~2014年前半のJリーグの観客数から2014年のJリーグの観客数の予測を行う。

2. データについて（説明変数の確認）

データは以下のように説明変数が42個存在する
※欠損値は存在しない

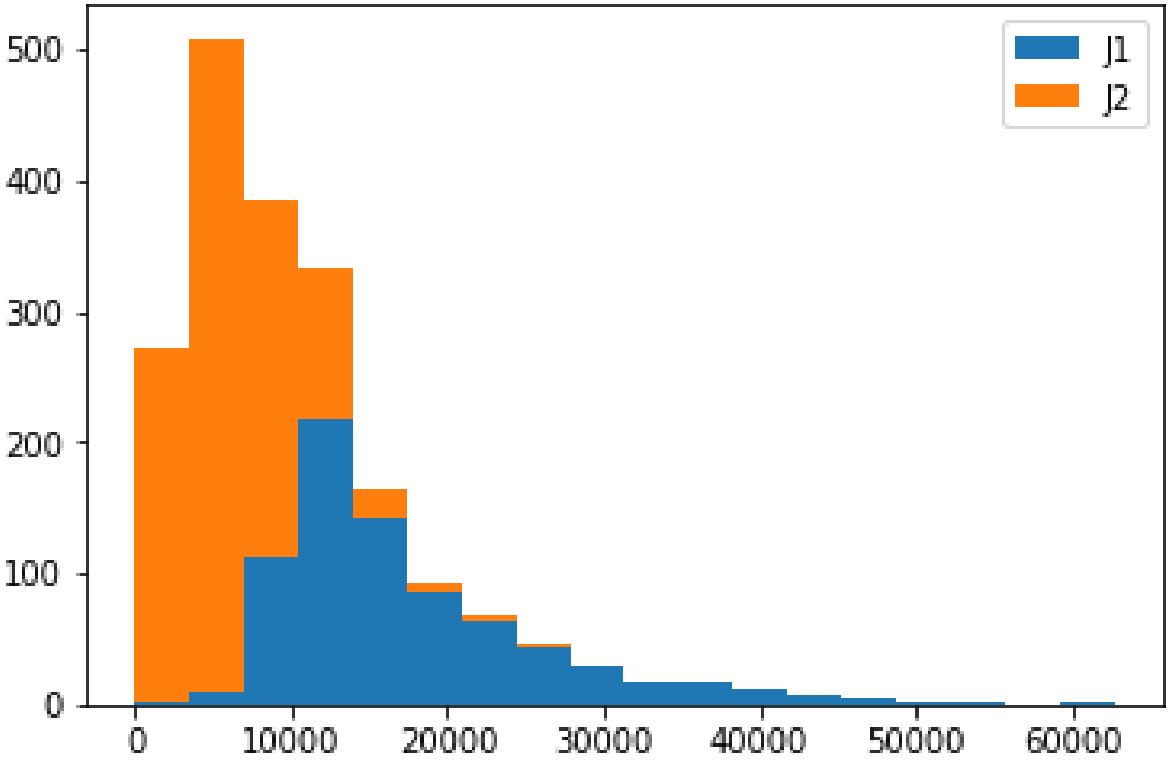
変数	定義
id	試合のid
y(目的変数)	観客数
year	年(2012,2013,2014)
stage	ステージ(J1,J2)
match	第〇節
gameday	試合日、曜日
time	キックオフ時間
home	ホームチーム名

変数	定義
away	アウェイチーム名
stadium	スタジアム名
tv	テレビで放送されたところ
home_score	ホームチーム得点数
away_score	アウェイチーム得点数
weather	天気
temperature	気温
humidity	湿度

変数	定義
referee	主審
home_team	ホームチーム名
home01~home11	ホームチーム選手名
away_team	アウェイチーム名
away_01~11	アウェイチーム選手名
address	スタジアムの住所
capa	スタジアムの収容人数

数値でない変数をどう数値化するかが重要そう！

2. データについて（目的変数の分布）



まず目的変数の分布を確認

- J1とJ2で大きく分布が違う

⇒後々J1とJ2でデータを分ける

2. データについて（今後の方針）

①まずとりあえず説明変数を数値化して一度モデルを作る

②そこから各変数の分布を確認して、説明変数の数値化の方法を改善。必要に応じて新たに外部変数をいれるなどの工夫を行う

③様々なモデルを試して、どのモデルを使うかを決める

④その後改善を繰り返す

3. 変数の数値化①

とりあえず数値がないものはほとんどマッピングで対応
→マッピングと書かれているものはLabelEncoderを利用して自動的にマッピングした。

変数	数値化方法
id	そのまま
y(目的変数)	そのまま
year	そのまま
stage	J1→0,J2→1
match	第〇節→〇を抽出
gameday	月(month)と曜日(day)を抽出
time	〇時(hour)の部分だけ抽出
home	マッピング

変数	定義
away	マッピング
stadium	マッピング
tv	'NHK総合'を含む(0),'NHK'を含む(1),'BS'を含む(2),その他(3)とする
	tvの数も抽出(tv_count)
home_score	そのまま
away_score	そのまま
weather	"雨"もしくは"雪"を含む(0),"晴"を含む(2),'屋内'を含む(3),その他(1)とする
temperature	そのまま

変数	定義
humidity	〇%→〇を抽出
referee	マッピング
home_team	homeと被るので削除
home01~home11	マッピング
away_team	awayと被るので削除
away_01~11	マッピング
address	都道府県でマッピング
capa	そのまま

3. 変数の数値化①

一旦単純に重回帰分析をおこない、そのまま提出する。
→念のため1次、2次、3次で分析を行った。
＜結果＞

- 自由度調整済み決定係数

1次 train: 0.700699　test : 0.648247
2次 train: 0.942　　test : 0.375
3次 train: 1.000　　test : -0.020

- RMSE

1次 train: 4443.847 test : 4751.291
2次 train: 1963.165 test : 6325.856
3次 train: 0.000　　test : 8085.446

→2次、3次は過学習が激しい⇒1次で提出

<提出結果＞
RMSE→6,024.96845、768位/930人中

4. 改善案①

①意味のないマッピングデータを意味のあるものにする or 削除
・選手名→日本代表の数(外部データ)
・チーム→順位(外部データ)を代入
・ホームチーム→ホームの試合での平均観客数(外部データ)を使用
・レフェリー→削除
②処理の改善
ただの重回帰分析から改善する。
→多い変数を自動で選択してくれるLassoやRidge、多くのコンペティションで上位を独占しているLGBMなどに挑戦
③その他一部改善
数値の標準化、外れ値の処理など

4. 改善案①

普通の重回帰分析(1次)
RMSE:3630.72572
順位:426位/931人中

Lasso回帰(1次)
RMSE:4125.57155
順位:674位/931人中

Ridge回帰 (2次)
RMSE:3922.85376
順位:621位/931人中

LGBM
RMSE:3599.20752
順位:404位/931人中

変数が多いことから自動で変数を減らしてくれるLasso回帰などに期待していたがそれよりも普通の重回帰分析やLGBMの方が良い結果となった。→Ridge回帰とLGBMを使用する

5. 改善案②

改善案①では変数がとにかく多くそれをLasso回帰やRidge回帰で減らそうと考えていたが、あまり効果的でなかった。Jリーグへの知識もあるので、自分なりに意味のない変数や意味が被る変数を減らすことにした。⇒これらの案の中から1個1個試してみて実行した

①似ている変数を減らす or 統合

yとの相関係数、重回帰分析の時の標準化などを考慮し決めた。

・月(month)と節(match)

・放送局数(tv_count)とテレビ局(tv)

②その他いらない変数を減らす

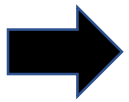
・湿度(humidity),温度(temperature), 天気(weather), ホームチーム得点(home_score), アウェイチーム得点(away_score), 時間(time)→削除

・曜日(day) → 土休日(1)かそうじゃないか(0)のみにする

5. 改善案②

③ one-hotエンコーディングを利用してダミー変数化
・ステージ(stage)をダミー変数化する
・ダミー変数ではないが、順位(home_rank,away_rank)もJ1とJ2にわけ、home_rank_j1,home_rank_j2,away_rank_j1,away_rank_j1を作成
→違うリーグの順位の部分は0にする
※one-hotエンコーディングとはJ1かでないかという変数とJ2かJ2でないかという変数をつくること。同じように順位も下図のようにした。

home_rank	away_rank
2	11
7	9
17	16
1	3
18	12



home_rank_j1	home_rank_j2	away_rank_j1	away_rank_j2
2.0	0.0	11.0	0.0
7.0	0.0	9.0	0.0
17.0	0.0	16.0	0.0
1.0	0.0	3.0	0.0
18.0	0.0	12.0	0.0

5. 改善案②

改善案②の中で効果があったものはtvとtv_countの削除と順位をJ1とJ2に分けたことのみだった。
結果は…

Ridge回帰 (2次)
RMSE:4,253.62204
順位:684位/931人中

LGBM
RMSE:3,505.95881
順位:313位/931人中

最終的に改善案②で思いついたことのほとんどが意味がないものだった。またその効果もかなり薄かった。またLGBMがどの結果でも最も結果が良かった。LGBMは過学習を起こしやすいプログラムでもあるのでその特徴を考慮して変数を減らそうとしたがなぜ変数が多いほうが結果が良いのかはわからなかった…

6. 感想、今後の展望

今回は良い結果を得ることを目標としていたので、思ったよりも良い結果が得ることができずとても悔しく思っている。全体の1/3程度がやっとだったのはやっぱりもう少しできることがあったのかなと思った。しかしまだいろいろやりたいと思っていたができなかったこともあった。例えば今回、各モデルのパラメーターを全くいじらなかったので、より良いパラメーターを探すためにグリッドサーチを使ったらもう少し結果が改善されたかもしれないと思った。またもっと結果が良くなればこの結果を利用し、観客数予測からチケットの値段を変えるダイナミックプライシングなどに活用できると考えられる。しかしそのためには後からわかる変数(順位)などは使わないように予測する必要があるためさらに難しいとも感じた。