第三回 GCI コンペ

化学システム工学専攻修士２年　菅原悠樹

手法の大枠

データセットの特徴として

- 時系列
- ほぼカテゴリー特徴量
- 正例・負例の偏り

がありました。

それらに対していくつかの対処をして作成した特徴量を LightGBM, XGBoost にかけました。

xxx と xxx_raw と書かれた feature については xxx の方を削除しました。また、単一の値の police_department や種類が多様すぎる fine_grained_location も削除しました。

今回の手法は、こちらのサイトの基本編と応用編をほぼ再現しただけなので、詳しくはそちらを参照してください。ここには、手法の細かな説明はしないで、再現しながら思っていたことを書いていきます。すべてについてちゃんとデータを可視化したわけではないので想像が多いです。**サイトの方を見たほうが有益です。**

時系列に対して

時間的に近いサンプルをもとにした特徴量を作成しました。具体的には、train と test をつなげて日時で sort した後に、各サンプルの前 n 時間, 後 n 時間, 前後 n/2 時間ずつのサンプルの is_arrested の平均値や、サンプルの出現数の特徴を作りました。

(n は 1800s ~ 84H の様々な値を用いました。)

ここでは、パーティーなどが行われて違反者が増える期間などの情報が抽出されるのかなと思います。

Categorical feature に対して

もとのカテゴリー特徴量だけでなく、それぞれのカテゴリー特徴量の組み合わせで新たな組合せカテゴリー特徴量を作成しました。

また、カテゴリーの特徴量はただ数字に置き換えてもあまりうまくいかないので、以下のように特徴を作成しました。

1. Count encoding

   カテゴリーの値を出てきた回数で置き換える。交通量が多い場所とかの情報がとれる（？）

2. Count unique

   location_raw, officer_id, violation_raw, stop_date などの、ある程度種類のあるカテゴリー特徴量について、互いに何種類の値を持つかについての特徴を作成。いろんな警官が配置される場所とかの情報が取れる（？）

3. Likelihood encoding

カテゴリーの値をその値を持つ y の平均値で置き換える。逮捕しやすい警官とかの情報がとれる（？）

4. Next Appearance
   それぞれのカテゴリーの値が次に出てくるまでの時間を特徴に追加。逮捕した警官は逮捕者を署へ連行するなどの手続きでしばらく登場しない（？）
5. 行:カテゴリカル変数 1, 列:カテゴリカル変数 2 として出現回数の行列を作成し、
   LDA と NMF を用いて、各カテゴリ値の潜在的意味を特徴量に追加

正例・負例の偏りに対して

train の全データのうち 2%程度しか正例がなく、負例サンプルばかりがあってもあんまり役に立たない上に数少ない正例が埋もれてしまうので、なんらかの対処が必要でした。今回は under sampling + bagging を使用しました。

under sampling とは負例をランダムに正例と同数サンプリングし、残りの負例を使用しないことです。これにより多少精度が下がりますが、under sampling を複数回行ってサブセットを複数作成しそれぞれの予測結果を ensemble(bagging)すると、全データで学習したときと同等以上の精度が出ることが多いです。

偏りのあるデータに対しては、学習時の sample_weights を調整したり over sampling したりの手法があると思いますが、決定木系で sample_weights が効くイメージが僕にはなく、特徴量がかなり膨大なので over sampling は難しそうでした。（一般にも under sampling + bagging が使われている印象があります。）

その他

複数の予測を混ぜ合わせる時には、今回は rank averaging を用いました。Rank averaging とは、それぞれの値の順序をつけて、その順序で平均を取ります。今回は評価指標が AUC で、予測値の順序のみが重要だったためです。

感想

時系列データは全区間の前半部分を train, 後半を test にする場合が多いと思いますが、今回は恐らくランダムシャッフルして train, test を分けているだけだったので、思いっきり未来情報を使おうと思っていました。未来の情報を使って過去を予測することになっているので実際の予測現場では使えませんがコンペなので…。(未来情報が使えないと test に含まれている期間初期のデータとか予測できませんし…)

どのような要素が予測対象に影響しているか知りたいときは未来情報もありかもしれないですね(？)

1 位の人（研究室同期）よりかなり特徴は作りこめたと思ったのですが、精度で負けました。LightGBM は不要な特徴にロバストですが無駄に作りすぎたか、あるいは一部の実装がまずくて train と test で一貫性が低い特徴を作っていたのかもしれないです。削除してしまった fine_grained_location の名前から何か法則を見い出したり、同じような地名をまとめて特徴を作るのもありだったかもしれません。

参考

https://www.rco.recruit.co.jp/career/engineer/blog/kaggle_talkingdata_basic/

https://www.rco.recruit.co.jp/career/engineer/blog/kaggle_talkingdata_advanced/