高级编程技术期末大作业

天池新人赛：快来一起挖掘幸福感!

数据科学与计算机学院　17计算机科学与技术

陈鸿峥*　　冯家苇*　　符　尧*　　傅　畅*

17341015　　17341035　　17341037　　17341038

一、 题目描述

幸福感与每个人息息相关，每个人对幸福感都有自己的衡量标准。如果能发现影响幸福感的共性，生活中是不是将多一些乐趣；如果能找到影响幸福感的政策因素，便能优化资源配置来提升国民的幸福感。

故本赛题[1]将利用调查问卷得到的数据对人们的幸福感进行预测。本赛题使用的数据来自中国人民大学中国调查与数据中心主持之《中国综合社会调查（CGSS）》项目，其中的变量包括个体变量（性别、年龄、地域、职业、健康、婚姻与政治面貌等等）、家庭变量（父母、配偶、子女、家庭资本等等）、社会态度（公平、信用、公共服务等等）等，通过这些指标来预测人们对于幸福感的评价。

输入文件说明：

- `complete`文件为变量完整版数据
- `abbr`文件为变量精简版数据
- `index`文件中包含每个变量对应的问卷题目，以及变量取值的含义
- `survey`文件是数据源的原版问卷，作为补充以方便理解问题背景

提交结果为`csv`文件，其中包含`id`和`happiness`的预测值两列。注意**预测值可以不为整数**。评价指标为均方误差(MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

其中，n为测试集样本数，y_i为第i个样本的真实值，$\hat{y}_i$为第i个样本的预测值。

二、 数据清洗

由于完整数据集包含140个指标，每个指标的内容及表示方法都不同，因此读入数据后的第一步是进行数据清洗。涉及以下几个方面：

---

*以学号为序，不代表贡献大小，小组分工如下：**陈鸿峥负责实验报告撰写；冯家苇和符尧负责具体模型实施及测试；傅畅负责PPT制作。模型构建及改进由四个人讨论一起完成。**

[1]阿里巴巴天池大赛-【新人赛】快来一起挖掘幸福感! https://tianchi.aliyun.com/competition/entrance/231702/information

- 变量的编码：如出生年份是一个连续型变量（从1921年到1997年连续变化），而城市属于离散型变量（不同城市之间只是编号不同，但城市编号不反映大小关系）。
- 缺失数据的填充：因为各种原因，人们在填写调查问卷时总会缺失一些信息，或者因为调查问卷本身的设计导致调查者不需要填写某一指标。但对于数据分析来说，不同样本的维度不同是很难处理的，因此需要把这些缺失的信息用合理的数据填充上去。
- 特征的选择：并非所有指标都对我们的幸福感预测有用，如调查时间存在非常强的随机性且明显与幸福感无关故可以直接删除。而有些指标相互之间存在强相关关系，那么可以只保留一个进行分析。

1. 负数数据处理

将含有负数的表项按照含义置为0或NaN(Not a Number)，举例如下所示

```
data.loc[data['income'] < 0, 'income'] = 0
data.loc[data['join_party'] < 0, 'join_party'] = np.nan
```

2. 创造新特征

相比起受访者受教育的年份，我们可能更在意受访者受教育的年龄，所以我们需要对这些数据进行处理（简单的计算），进而创造出新的特征，如下

```
data['edubir'] = data['edu_yr'] - data['birth']
```

其他如收入比率、家庭收入占比等可以通过现有数据计算出来的，也应先计算出来，如下

```
data['income/family_income'] = data['income'] / (data['family_income'] + 1)
data['all_income/family_income'] = (data['income'] + data['s_income']) / (data['
    ↪ family_income'] + 1)
```

一些数据均值也可以先计算出来，如下

```
data['province_income_mean'] = data.groupby(['province'])['income'].transform('
    ↪ mean').values
```

完整的数据处理过程请见代码文件。

3. 归一化处理

将新的特征和原有的特征进行整合，并进行归一化，防止数据之间差异太大。

```
min_max_scaler = MinMaxScaler()
X_train = min_max_scaler.fit_transform(X_train)
```

三、 模型搭建

本次比赛我们采用了集成学习(ensemble learning)的方法，称为RSXLC-BRR框架。我们采用五个子模型（岭回归、支持向量回归、三种梯度提升方法），最后采用一个线性模型（贝叶

斯岭回归）对这五个子模型的预测效果进行整合。模型框架见图1。

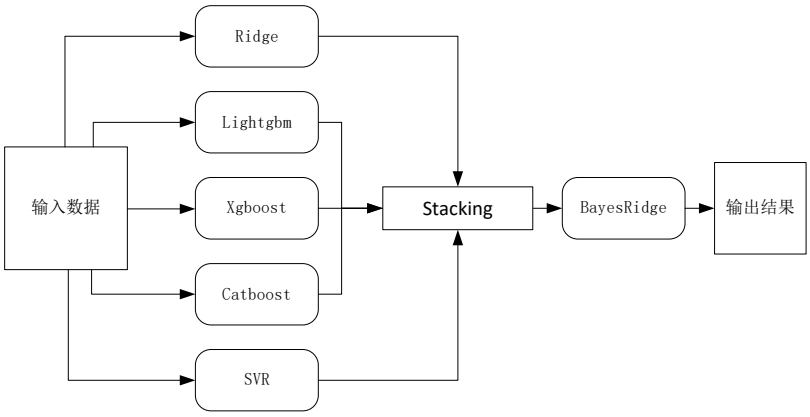


图 1: RSXLC-BRR模型框架

由于最终评价指标是MSE，故我们所有的模型都是回归模型，主要优化下面这条最小二乘表达式

$$\min_{\mathbf{w}} \quad \|X\mathbf{w}-\mathbf{y}\|_2^2$$

其中$X=\begin{bmatrix}\mathbf{x}_1 & \cdots & \mathbf{x}_n\end{bmatrix}^{\mathrm{T}}$为样本特征的集合，$\mathbf{w}$为需要训练的权重（注意这里为了表达方便，将偏置量$\mathbf{b}$也融入其中），$\mathbf{y}$为真实幸福感指数。

1. RidgeRegression

岭回归(Ridge Regression)即在原式的基础上添加一个L2正则化项（令$\mathbf{w}$的各项元素之间差距不会太大），使得优化目标变为

$$\min_{\mathbf{w}} \quad \|X\mathbf{w}-\mathbf{y}\|_2^2+\lambda\|\mathbf{w}\|_2^2$$

通过求梯度可以解得显式解

$$\mathbf{w}=(X^{\mathrm{T}}X+\lambda I)^{-1}X^{\mathrm{T}}\mathbf{y}$$

2. SVR

支持向量回归(Support Vector Regression, SVR)则是在岭回归的基础上添加一个边界范围。

$$\begin{aligned}\min_{\mathbf{w}} \quad & \frac{1}{2}\|\mathbf{w}\|_2^2\\ \text{s.t.} \quad & -\varepsilon\le y_i-\mathbf{x}_i^{\mathrm{T}}\mathbf{w}\le\varepsilon,\ \forall i\end{aligned}$$

通过求拉格朗日函数及拉格朗日对偶问题，同样可以得到迭代解法。

3. XGBoost

XGBoost即eXtreme Gradient Boosting，最早由华盛顿大学陈天奇提出，作为现在数据挖掘比赛中的利器，也被使用到我们的模型中。

首先阐述梯度提升(gradient boosting)这一集成学习常用方法的思想。即对目标函数$f(\mathbf{x})$进行多次逼近，通过不断拟合残差达到逼近的效果，可以按照下式不断迭代

$$
\begin{aligned}
f_1(\mathbf{x}) &= \widehat{f}(\mathbf{x}) & h_1(\mathbf{x}) &= f(\mathbf{x}) - f_1(\mathbf{x}) \\
f_2(\mathbf{x}) &= f_1(\mathbf{x}) + \widehat{h_1}(\mathbf{x}) & h_2(\mathbf{x}) &= f(\mathbf{x}) - f_2(\mathbf{x}) \\
f_3(\mathbf{x}) &= f_2(\mathbf{x}) + \widehat{h_2}(\mathbf{x}) & h_3(\mathbf{x}) &= f(\mathbf{x}) - f_3(\mathbf{x}) \\
&\vdots & &\vdots \\
f_n(\mathbf{x}) &= f_{n-1}(\mathbf{x}) + \widehat{h_{n-1}}(\mathbf{x})
\end{aligned}
$$

对于最小二乘问题，负梯度就是每次计算得到的残差。由于每次都新增一个残差项，因此被称为梯度提升。

而回归/决策树(decision tree)则是每次选择信息增益最大/基尼指数最小的特征作为结点进行划分，最终形成一棵树，每次预测时根据树结点的判断寻找对应的叶结点，即得到最终的预测值。

将梯度提升和决策树结合起来就得到梯度提升回归树(Gradient Boosting Regression Tree, GBRT)，其核心在于，它的每棵树都是从上一次训练的所有树的残差中进行学习，进而拟合一棵回归/分类树，如图2所示。

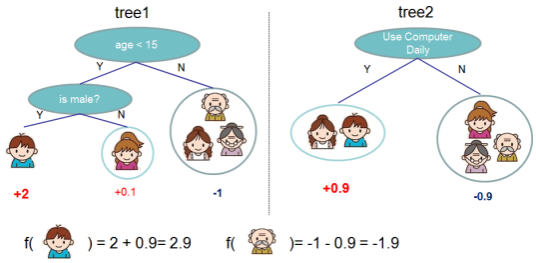


图 2: 树集成模型，图源自XGBoost原始论文

故对于这一模型而言，其预测值就是

$$
\hat{y}_i = \sum_{k=1}^{K} f_k(\mathbf{x}_i)
$$

有目标优化问题

$$\min \quad \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k)$$

其中，l为损失函数(loss)，Ω为每棵树的复杂度，为惩罚项。

通过对目标函数进行Taylor展开并求梯度，可以得到迭代更新的表达式，具体请见XGBoost的原始论文。

相比起传统的GBRT，XGBoost的优势在于以下几点：

- 传统GBRT只用了一阶梯度信息，而XGBoost采用了二阶Taylor展开，精度更高
- XGBoost在目标函数中添加了正则化项，用于控制模型的复杂度
- 通过在迭代式中添加一个缩减(shrinkage)项（类似于学习率），防止越过最优解，同时防止过拟合
- 并行化处理

4. LightGBM

之后的几种梯度提升方法都可以看作是对XGBoost的一种改进。

LightGBM是微软提出的一种高效的大规模并行梯度提升的框架，其相比于XGBoost又有以下几点改进：

- 更快的训练速度和更高的效率
- 更低的内存占用
- 更高的准确率

这些都源于其采用了leaf-wise的决策树增长方式，而不是level-wise的增长方式，如图3所示。

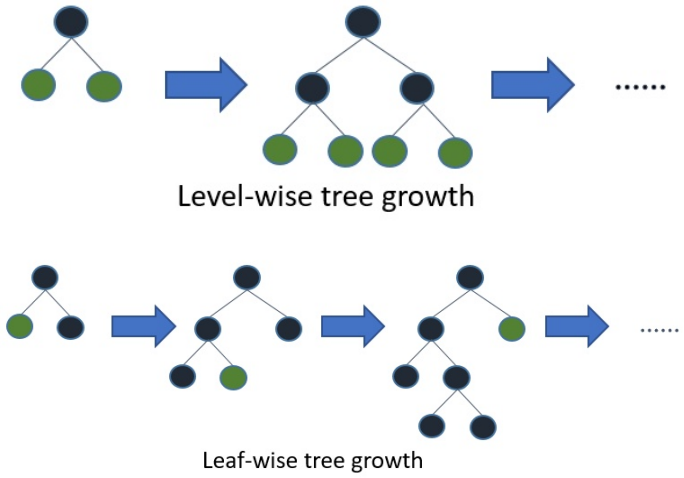


图 3: 不同的树增长方式

5. CatBoost

CatBoost则是俄罗斯Yandex公司提出的梯度提升框架。

传统GBDT算法通常对特征进行独热编码(ont-hot encoding)后进行训练。而CatBoost则对离散型变量进行特殊的处理，先将样本随机排序，然后选取前$p-1$个样本做平均y值代替，同时加入了a和P作为超参数进行平衡，如下

$$\frac{\sum_{j=1}^{p-1}[x_{\sigma_j,k}=x_{\sigma_p,k}]Y_{\sigma_j}+a\cdot P}{\sum_{j=1}^{p-1}[x_{\sigma_j,k}=x_{\sigma_p,k}]+a}$$

CatBoost还结合上述样本排序方法，采用了一种新的计算方法来生成树结构，可以有效缓解过拟合。

6. BRR

最后一个模型则是贝叶斯岭回归(Bayesian Ridge Regression)，将前面五个模型的预测值再做一次回归分析。

五个子模型的预测值相当于贝叶斯回归中的取样结果，即用先验分布推后验分布，最终得到更为精确的拟合曲线。

四、 具体实施

由于比赛方没有提供验证集，故对于每一个子模型，我们都采用K折交叉验证的方法，将数据集划分为k等分，其中$k-1$份用于训练，剩下一份用于验证。

这里只截取最后一个综合模型(`BayesianRidge`)的代码，其他模型实现方法类似，请见源文件`model.py`。

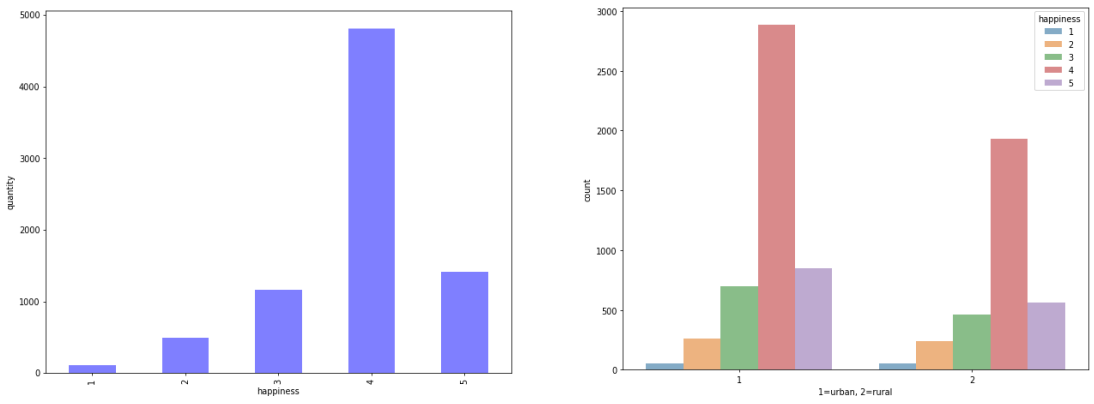
```
train_stack = np.vstack([oof_lgb, oof_xgb, oof_cb, oof_svr, oof_rig]).transpose()
test_stack = np.vstack([predictions_lgb, predictions_xgb, predictions_cb,
    ↪ predictions_svr, predictions_rig]).transpose()

folds_stack = RepeatedKFold(n_splits=5, n_repeats=2, random_state=4590)
oof_stack = np.zeros(train_stack.shape[0])
predictions = np.zeros(test_stack.shape[0])

for fold_, (trn_idx, val_idx) in enumerate(folds_stack.split(train_stack, target))
    ↪ :
    print("fold {}".format(fold_))
    trn_data, trn_y = train_stack[trn_idx], target.iloc[trn_idx].values
    val_data, val_y = train_stack[val_idx], target.iloc[val_idx].values

    clf_3 = BayesianRidge()
    clf_3.fit(trn_data, trn_y)

    oof_stack[val_idx] = clf_3.predict(val_data)
    predictions += clf_3.predict(test_stack) / 10
```

画图的代码请见show.py。

五、 实验过程

1. 数据处理

首先观察数据集总体的分布。如图4，观察图（a）可以发现，有60%左右的被调查者选择了幸福指数4，意味着数据集分布有一定的偏斜。这将对模型提出更高的要求；观察图（b）可以发现，城乡幸福感有相似的分布。这里我们采用独热(one-hot)编码的方式区分这两类样本。



(a) (b)

图 4: 数据分析图。左图为幸福感统计直方图，右图为城乡幸福感统计直方图

图5展示了数据集中有缺失的几类特征。对于缺失较少的数据，我们采用补0或补平均值的方式；对于缺失较多的数据，我们通过实验对比选择性删除。

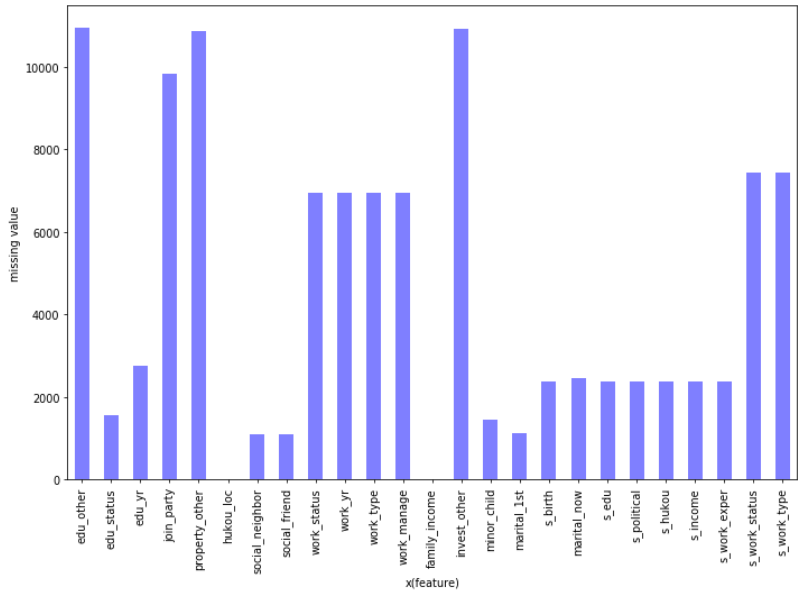


图 5: 缺失值统计

图6和图7展示了数据集处理后的原特征以及新加入的特征与幸福指数的相关性。这里选取了相关程度较高的（≥ 0.1）与较低的（≤ 0.015）一些特征。在相关性较强的特征中，可以发现部分新加入的特征（从首个到`class_10_diff`）有良好的正相关性；“沮丧频率”（depression，1表示频率最高）是原特征中正相关性最高的；而像“与3年前地位对比”(`status_3_before`)以及“与同龄人地位对比”(`status_peer`)等特征则具有较强的负相关性。在相关性较弱的特征中，我们通过实验对比选择性删除了部分，例如出生年份(birth)实际上具有较低的相关性，我们已将其删去。

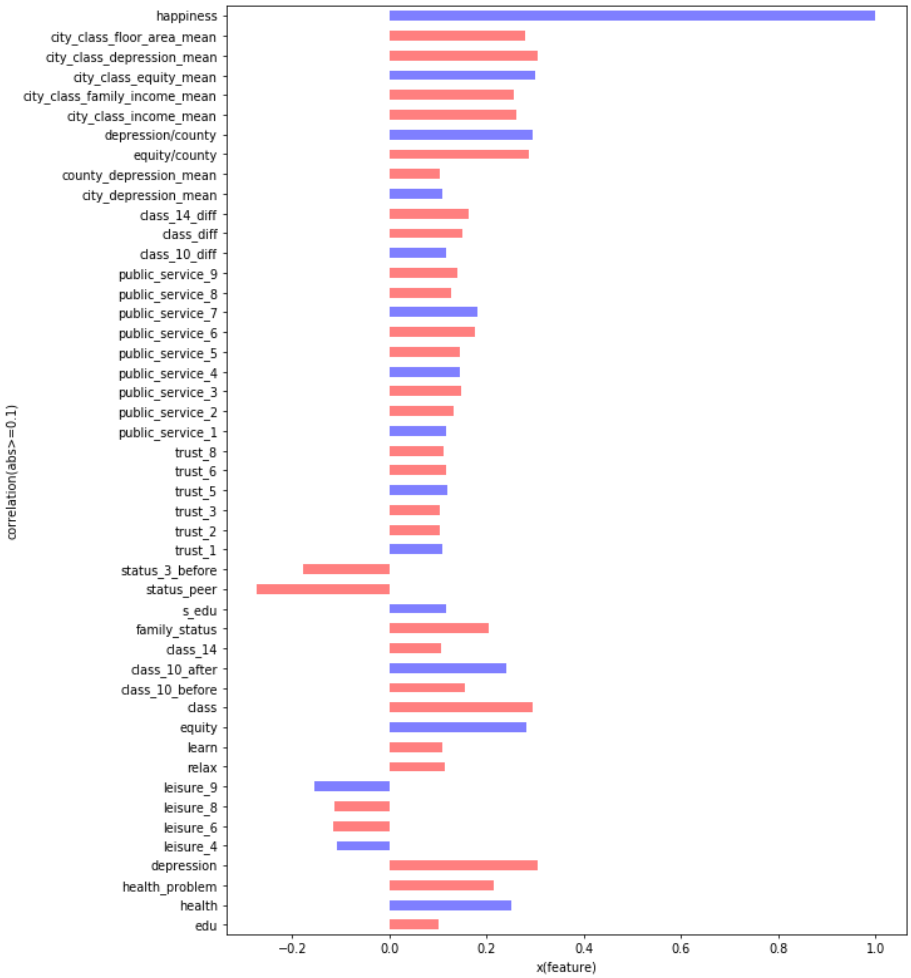


图 6: 相关性分析（≥ 0.1）

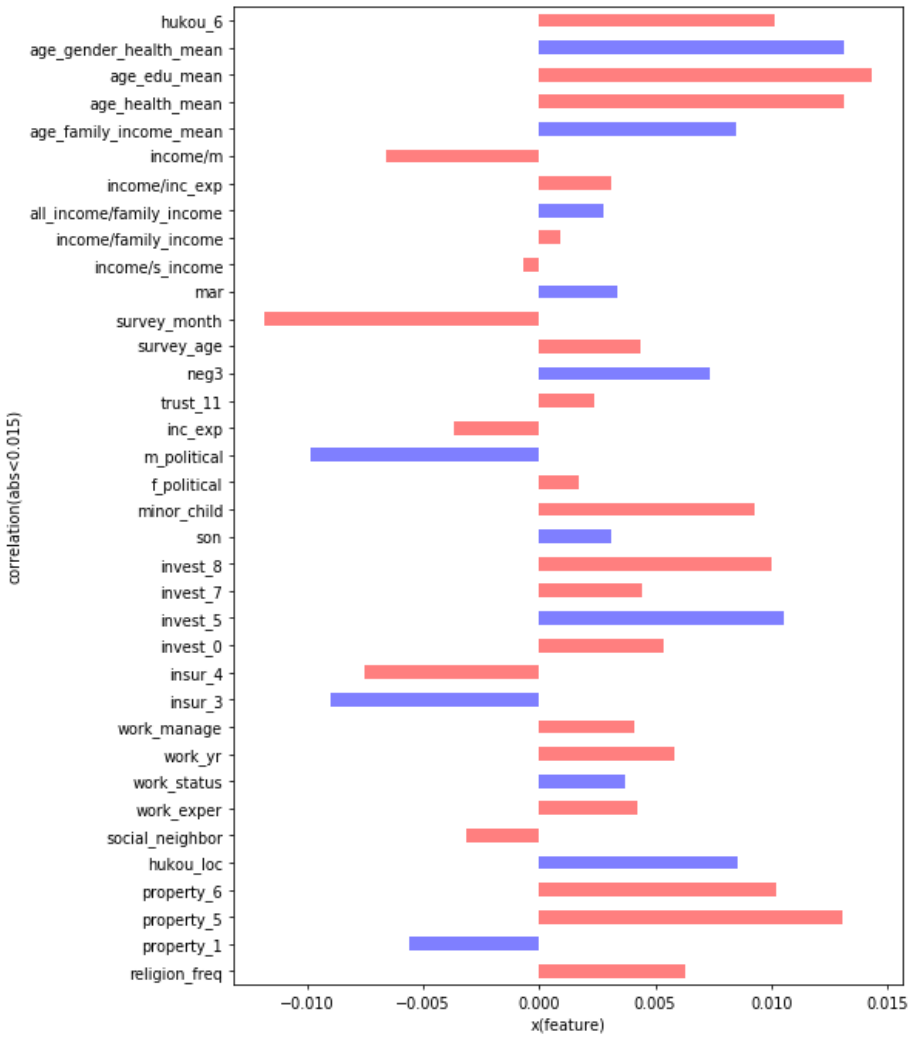


图 7: 相关性分析（≤ 0.015）

2. 训练过程

这里选取了子模型LightGBM的训练过程中Loss（即MSE）的变化，如图8所示，可以看到在训练集上模型学的很好，同时在验证集上取得不错的效果。

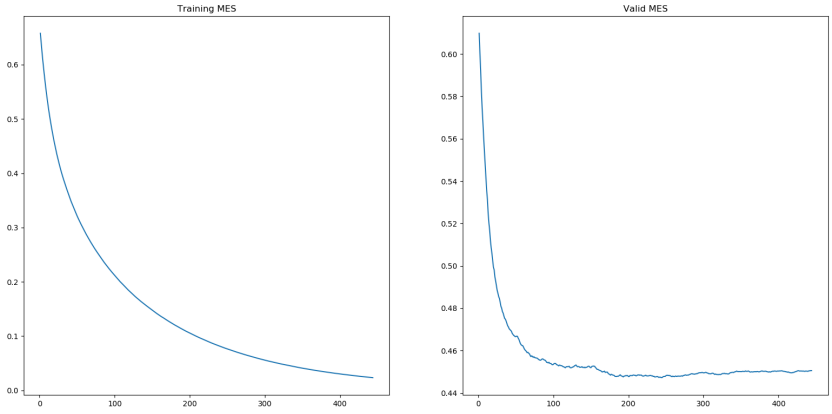


图 8: LightGBM模型训练过程。左图为训练集上的Loss曲线，右图为验证集的Loss曲线

通过`model.coef_`可查看五个模型的权重分布，如图9所示。

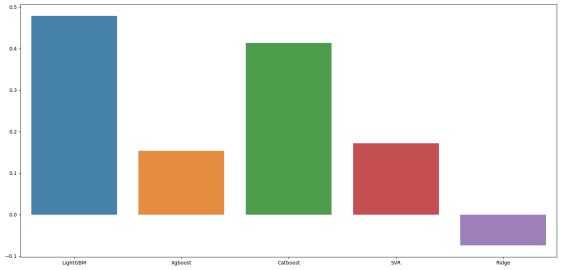


图 9: BayesRidge为每个模型赋予了不同的权重

3. 最终排名

我们修改了不同的超参数，提交了两个版本，都取得了不错的成绩，如图10所示。具体成绩见图11。

图 10: 最终排名

(a) 账号jfjcaa　　(b) 账号f.y.

图 11: 具体成绩

PPT请见`report.pptx`，最终提交结果请见`happiness_submit.csv`。

六、 参考资料

1. Tianqi Chen and Carlos Guestrin. *XGBoost: A Scalable Tree Boosting System*. In 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016.
2. 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*. Advances in Neural Information Processing Systems 30 (NeurIPS), 2017.
3. Anna Veronika Dorogush, Vasily Ershov, Andrey Gulin. *CatBoost: gradient boosting with categorical features support*. Workshop on ML Systems at NIPS 2017.
4. Wikipedia. *Bayesian linear regression*.
5. 周志华. 机器学习. 清华大学出版社. 2016.