



华东师范大学
East China Normal University

统计方法与机器学习 Project1

姓 名： 杨茜雅

学 号： 10215501435

学 院： 数据科学与工程学院

课 程： 统计方法与机器学习

-

2023 年 11 月

目录

- 1、 数据集简介
- 2、 实验过程
 - 2.1 数据预处理
 - 2.2 相关性分析
 - 2.3 基于机器学习模型的取消率预测
 - 2.3.1 手动选取特征
 - 2.3.2 特征分类
 - 2.3.3 特征预处理
 - 2.3.4 选取评估模型的统计方法
 - 2.3.5 机器学习模型评分展示
 - 2.3.6 随机森林模型
- 3、 随机森林模型优化（参数调优）
- 4、 特征评分

1、数据集简介

数据集包含了从 2015 年 7 月到 2017 年 8 月一家城市酒店和一家度假酒店的预定信息、包括诸如预定时间、逗留时间、成人、儿童、婴儿的数量，以及可用的停车位数量等 32 条信息。原始数据集已经存放在项目文件夹中。

```
Data columns (total 32 columns):
#   Column                                     Non-Null Count  Dtype
---  -
0   hotel                                     119390 non-null object
1   is_canceled                             119390 non-null int64
2   lead_time                               119390 non-null int64
3   arrival_date_year                       119390 non-null int64
4   arrival_date_month                     119390 non-null object
5   arrival_date_week_number               119390 non-null int64
6   arrival_date_day_of_month              119390 non-null int64
7   stays_in_weekend_nights                119390 non-null int64
8   stays_in_week_nights                   119390 non-null int64
9   adults                                  119390 non-null int64
10  children                                119386 non-null float64
11  babies                                  119390 non-null int64
12  meal                                    119390 non-null object
13  country                                 118902 non-null object
14  market_segment                          119390 non-null object
15  distribution_channel                    119390 non-null object
16  is_repeated_guest                       119390 non-null int64
17  previous_cancellations                  119390 non-null int64
18  previous_bookings_not_canceled          119390 non-null int64
19  reserved_room_type                      119390 non-null object
20  assigned_room_type                      119390 non-null object
21  booking_changes                         119390 non-null int64
22  deposit_type                            119390 non-null object
23  agent                                   103050 non-null float64
24  company                                 6797 non-null float64
25  days_in_waiting_list                   119390 non-null int64
26  customer_type                           119390 non-null object
27  adr                                     119390 non-null float64
28  required_car_parking_spaces             119390 non-null int64
29  total_of_special_requests               119390 non-null int64
30  reservation_status                     119390 non-null object
31  reservation_status_date                 119390 non-null object
dtypes: float64(4), int64(16), object(12)
```

2、 实验过程

该项目中我首先对现有数据做了一些预处理，方便后续工作，再进行相关性分析，对数据特征有了一个最初步的了解。最后建立多个机器学习的模型来预测顾客是否会取消订单，并且针对表现最好的 base model 进行优化和特征评分。

2.1 数据预处理

```
print(data.isnull().sum()[data.isnull().sum() != 0])
```

```
children      4
country      488
agent      16340
company     112593
```

通过查看缺失值可以发现 company 特征缺失过多，所以删去。对于缺失值较少的 country 和 children，用字段内的众数填充。对于 agent，则用 0 填充，表示没有指定任何机构。

```
# 缺失值处理
# company 缺失太多，删除
# country、children和agent缺失比较少，用字段内的众数填充
# country和children用字段内的众数填充 agent缺失值用0填充，代表没有指定任何机构
data_new = data.copy(deep=True)
data_new.drop("company", axis=1, inplace=True)
data_new["agent"].fillna(0, inplace=True)
data_new["children"].fillna(data_new["children"].mode()[0], inplace=True)
data_new["country"].fillna(data_new["country"].mode()[0], inplace=True)
```

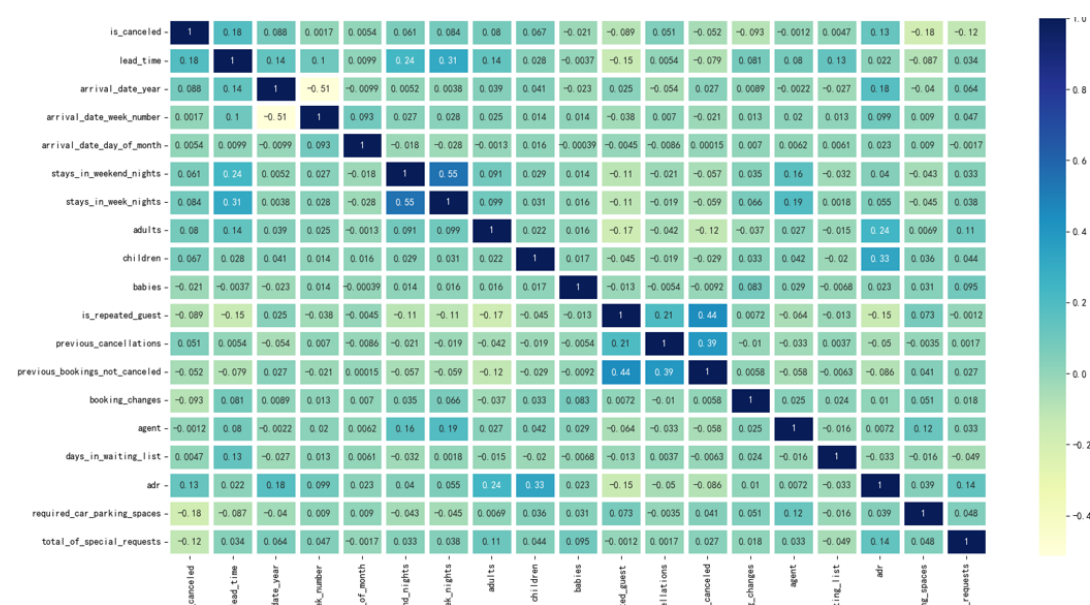
同时对于异常数据进行删除，比如 adults+children+babies=0 的订单是不符合常理的，理应删除。对于用到简写的字段需要进行字段映射处理，增加可读性。最后形成新的数据集 data_new，规模从 (119390, 32) 变为 (87204, 31)。

```
# 处理一下异常值：成人+小孩+婴儿=0的情况都需要删掉
data_new["children"] = data_new["children"].astype(int)
data_new["agent"] = data_new["agent"].astype(int) # 转换数据类型
zero_guests = list(data_new["adults"] + data_new["children"] + data_new["babies"] == 0)
data_new.drop(data_new.index[zero_guests], inplace=True)
# meal字段映射处理
data_new["meal"].replace(["Undefined", "BB", "FB", "HB", "SC"], ["No Meal", "Breakfast", "Full Board", "Half Board", "No Meal"], inplace=True)
# 数据去重
data_new.drop_duplicates(inplace=True)
data_new.to_csv('data_new.csv')
```

2.2 相关性分析

查看每一个特征与 is_canceled 的相关系数

```
Name: is_canceled, dtype: float64
is_canceled      1.000000
lead_time        0.292876
total_of_special_requests 0.234877
required_car_parking_spaces 0.195701
booking_changes  0.144832
previous_cancellations 0.110139
is_repeated_guest 0.083745
adults           0.058182
previous_bookings_not_canceled 0.057365
days_in_waiting_list 0.054301
agent            0.046770
adr              0.046492
babies           0.032569
stays_in_week_nights 0.025542
arrival_date_year 0.016622
arrival_date_week_number 0.008315
arrival_date_day_of_month 0.005948
children         0.004851
stays_in_weekend_nights 0.001323
Name: is_canceled, dtype: float64
```



与是否取消订单相关性 top5 高的特征分别是：提前预定的时长、特殊要求

总量、是否需要停车位、对预定进行的更改、历史取消量。

2.3 基于机器学习模型的取消率预测

预测顾客是否会取消订单时我试用了六个机器学习的 base model，大体思路是将六个 base model 都尝试一遍，并选取表现最优的进行模型优化。我选取的 base model 有：决策树、随机森林、逻辑回归、XGBoost、KNN 和 AdaBoost。

2.3.1 手动选取特征

此处删除所有时间类特征， arrival_date_year、arrival_date_month、arrival_date_week_number。

2.3.2 特征分类

```
#分离特征类型
num_features = ["lead_time", "booking_changes", "total_of_special_requests", "required_car_parking_spaces",
                "days_in_waiting_list",
                "previous_cancellations", "is_repeated_guest", "adults", "previous_bookings_not_canceled", "agent",
                "adr", "babies", "stays_in_weekend_nights", "arrival_date_week_number", "arrival_date_day_of_month",
                "children", "stays_in_week_nights"]

cat_features = ["hotel", "arrival_date_month", "meal", "market_segment", "country", "assigned_room_type",
                "distribution_channel", "reserved_room_type", "deposit_type", "customer_type"]

#分离特征和预测值
features = num_features + cat_features
X = data_new.drop(["is_canceled"], axis=1)[features]
y = data_new["is_canceled"]
```

数据集中有整数型特征和分类型特征，为了方便特征预处理，将两种特征分离到 num_features 和 cat_features 中。

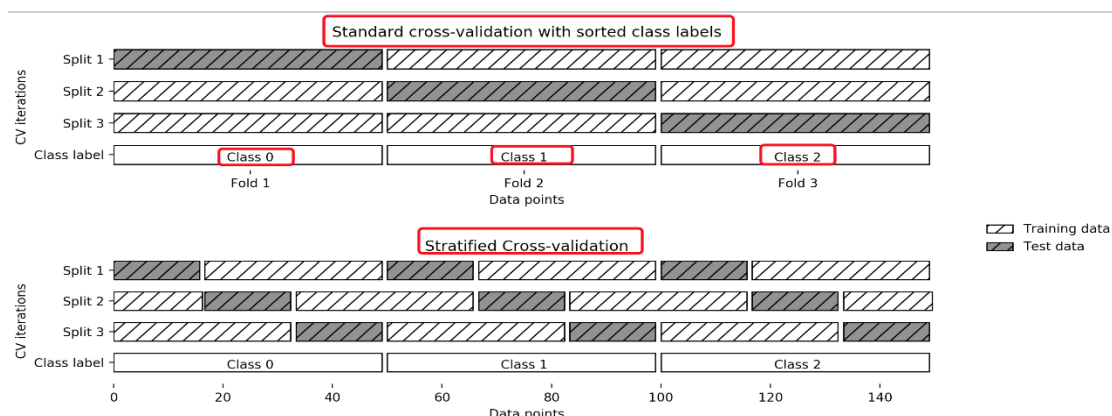
2.3.3 特征预处理

```
# 预处理数值特征:
#整数型特征
# 对于大多数数值型的特征，用0填充缺失值
num_transformer = SimpleImputer(strategy="constant", fill_value=0)

# 分类型特征的预处理:
cat_transformer = Pipeline(steps=[("imputer", SimpleImputer(strategy="constant", fill_value="Unknown")),
                                   ("onehot", OneHotEncoder(handle_unknown='ignore'))])
```

对于整数型特征，用对于最大多数特征都符合逻辑的 0 进行填充；对于分类型特征，用 OneHotEncoder 将其数字化，方便后续评分。

2.3.4 选取评估模型的统计方法



选取分层 K-Fold 交叉验证方法，其优点是对于不平衡的数据集比较友好，该方法可以将数据集先分成 k 折（实验中选择了十折，在用 `train_test_split` 函数划分模型的训练集和测试集时，通常会用四折，也就是 75% 的数据作为训练集，25% 的数据作为测试集；在五折交叉验证的时候，是 80% 的数据作为训练集，20% 的数据作为测试集；十折交叉验证的时候，是 90% 的数据作为训练集，10% 的数据作为测试集。显然，训练集的数据越多，模型的 accuracy 得分就越高。但是十折交叉运算也意味着六个模型都要训练十次，增加十倍的训练时间），然后再从每个 k 折中选择 k 折作为测试集合，这样就可以在口中程度上保证每次测试集合和训练集和都有各个类型的样本，特别是在当数据分类的类型数量相差悬殊的情况下，而本次实验所用的数据集有样本分布不均匀的问题，客户取消和未取消的比例约为 3:7，分层 k 折交叉验证可以使随机划分测试集时不存在一个测试集中全部都是未取消顾客的情况。缺点是在该方法中样本是随机选取的，所以不适合时间序列型数据，因此在手动选取特征值的阶段删去了所有时间型特征，一是为了让模型更加通用，不受限于 2015-2017 的时间范围，二是为了与评估方法更加适配。

2.3.5 机器学习模型评分展示

DT_model cross validation accuracy score: 0.8601 +/- 0.0027 (std) min: 0.8556, max: 0.8654, mean: 0.8601
RF_model cross validation accuracy score: 0.8941 +/- 0.0023 (std) min: 0.8922, max: 0.9004, mean: 0.8941
LR_model cross validation accuracy score: 0.8151 +/- 0.0023 (std) min: 0.8107, max: 0.8187, mean: 0.8151
XGB_model cross validation accuracy score: 0.8772 +/- 0.0018 (std) min: 0.8739, max: 0.8794, mean: 0.8772
KNN_model cross validation accuracy score: 0.8097 +/- 0.0035 (std) min: 0.8043, max: 0.8179, mean: 0.8097
AdaB_model cross validation accuracy score: 0.8298 +/- 0.0034 (std) min: 0.8243, max: 0.8354, mean: 0.8298

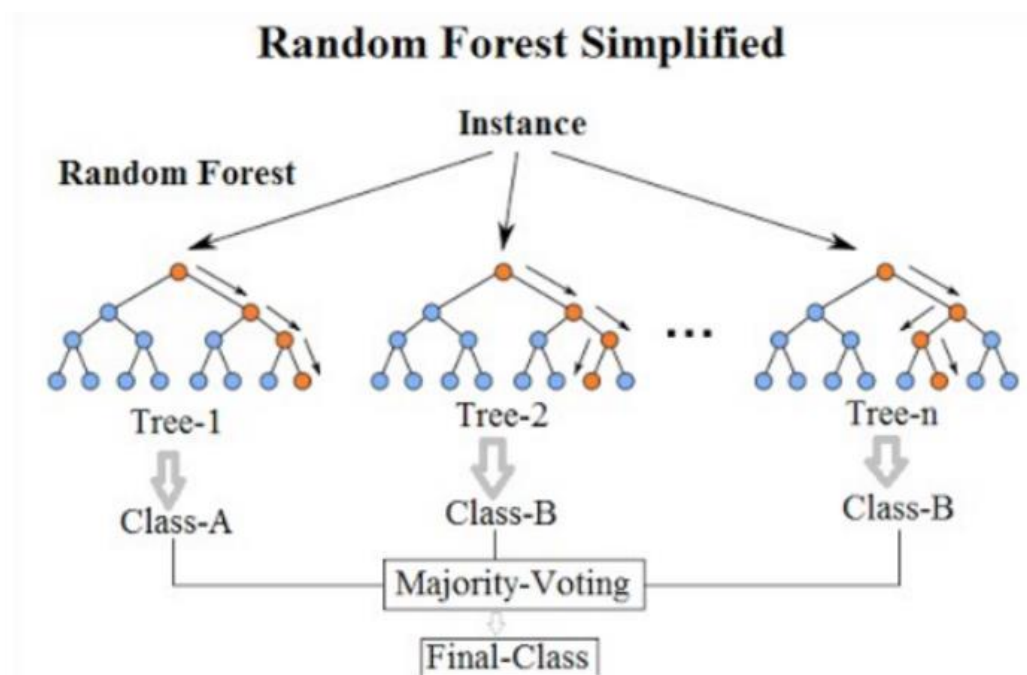
可以看出随机森林是表现最好的模型，XGBoost 次之，表现最差的是 KNN。

对 KNN 表现不好的猜测：在样本不平衡的时候，对稀有类别的预测准确率低，比如一个类的样本容量很大，输入一个样本的时候，K 个临近值中大多数都是大样本容量的那个类。假设实验中取消订单的顾客占比大约百分之 30，即使 KNN 得分在 80 左右，也有可能把所有顾客都预测成了未取消订单的顾客；相比起决策树等模型，KNN 模型的可解释性也不强，是慵懒学习法，基本上不学习，导致实验时的运行速度相对于其他算法更慢。

对随机森林表现好的猜测：对多元共线性不敏感，对非平衡的数据比较稳健。

通过 bootstrap 重采样技术，可以对占比高的样本降采样，也可以对占比少的样本过采样，以此弱化样本不平衡的问题。

2.3.6 随机森林模型



随机森林，就像它的名字所暗示的那样，由大量独立的决策树组成，它们作为一个整体运行。随机森林中的每棵树都给出一个类预测，投票最多的类将成为模型的预测结果。

用数据科学的话说，随机森林模型如此有效的原因是：

作为一个整体运行大量相对不相关的模型（树）将胜过任何单个的模型。

随机森林表现良好的先决条件是：

在我们的特性中需要一些实际的信号，这样使用这些特性构建的模型才能比随机猜测做得更好。

单个树所做的预测(以及因此产生的错误)需要彼此之间具有较低的相关性。

当我们进行分类任务时，新的输入样本进入，就让森林中的每一棵决策树分别进行判断和分类，每个决策树会得到一个自己的分类结果，决策树的分类结果中哪一个分类最多，那么随机森林就会把这个结果当做最终的结果。

随机森林（Random Forest）属于 Bagging。随机森林简单地来说就是用随机的方式建立一个森林，森林由很多的决策树组成，随机森林的每一棵决策树之间是没有关联的。

在我们学习每一棵决策树的时候就需要用到 Bootstrap 方法。在随机森林中，有两个随机采样的过程：对输入数据的行（数据的数量）与列（数据的特征）都进行采样。对于行采样，采用有放回的方式，若有 N 个数据，则采样出 N 个数据（可能有重复），这样在训练的时候每一棵树都不是全部的样本，相对而言不容易出现 overfitting；接着进行列采样从 M 个 feature 中选择出 m 个 ($m < M$)。最近进行决策树的学习。

预测的时候，随机森林中的每一棵树的都对输入进行预测，最后进行投票，

哪个类别多，输入样本就属于哪个类别。这就相当于前面说的，**每一个分类器都比较弱**，但组合到一起投票时就比较强了。

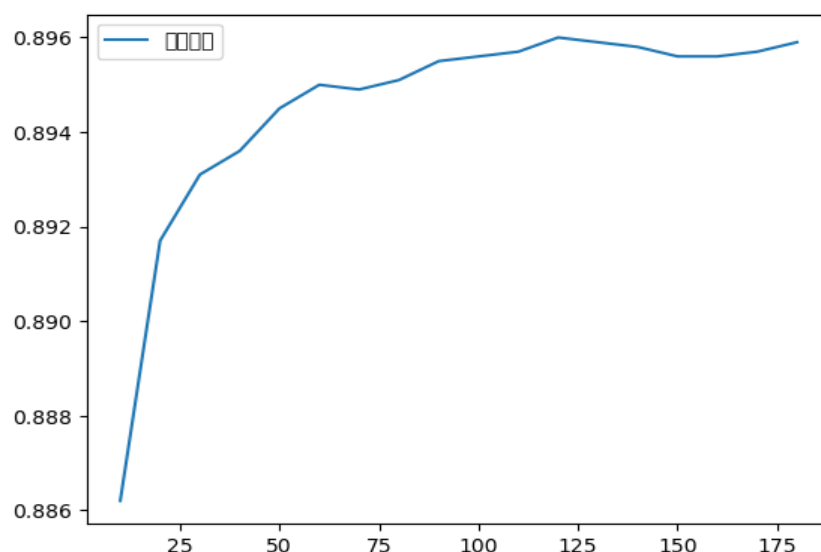
3、 随机森林模型优化（参数调优）

选取表现最优的随机森林模型，并对其进行 `n_estimators` 参数调优。

```
rfc_s = []
for i in range(0, 181, 10):
    rf_model_enh = RandomForestClassifier(n_estimators=i, max_features=0.4, min_samples_split=2, random_state=42)
    split = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
    model_pipe = Pipeline(steps=[('preprocessor', preprocessor), ('model', rf_model_enh)])
    cv_results = cross_val_score(model_pipe, X, y, cv=split, scoring="accuracy", n_jobs=-1)
    min_score = round(min(cv_results), 4)
    max_score = round(max(cv_results), 4)
    mean_score = round(np.mean(cv_results), 4)
    std_dev = round(np.std(cv_results), 4)
    print(f'Enhanced RF model cross validation accuracy score: {mean_score} +/- {std_dev} (std) min: {min_score}, max: {max_score}, mean: {mean_score}')
    rfc_s.append(mean_score)
plt.plot(range(0, 181, 10), rfc_s, label='随机森林')
plt.legend()
```

将范围设置在 0-180 之间，步长为 10，并绘制学习曲线。

```
Enhanced RF model cross validation accuracy score: nan +/- nan (std) min: nan, max: nan, mean: nan, n_estimators: 0
Enhanced RF model cross validation accuracy score: 0.8862 +/- 0.0026 (std) min: 0.8827, max: 0.8903, mean: 0.8862, n_estimators: 10
Enhanced RF model cross validation accuracy score: 0.8917 +/- 0.0021 (std) min: 0.8874, max: 0.8942, mean: 0.8917, n_estimators: 20
Enhanced RF model cross validation accuracy score: 0.8931 +/- 0.0025 (std) min: 0.8894, max: 0.8961, mean: 0.8931, n_estimators: 30
Enhanced RF model cross validation accuracy score: 0.8936 +/- 0.0027 (std) min: 0.889, max: 0.8978, mean: 0.8936, n_estimators: 40
Enhanced RF model cross validation accuracy score: 0.8945 +/- 0.0023 (std) min: 0.8909, max: 0.898, mean: 0.8945, n_estimators: 50
Enhanced RF model cross validation accuracy score: 0.895 +/- 0.0026 (std) min: 0.8914, max: 0.8988, mean: 0.895, n_estimators: 60
Enhanced RF model cross validation accuracy score: 0.8949 +/- 0.0025 (std) min: 0.8916, max: 0.8986, mean: 0.8949, n_estimators: 70
Enhanced RF model cross validation accuracy score: 0.8951 +/- 0.0024 (std) min: 0.8916, max: 0.8988, mean: 0.8951, n_estimators: 80
Enhanced RF model cross validation accuracy score: 0.8955 +/- 0.0025 (std) min: 0.8913, max: 0.8993, mean: 0.8955, n_estimators: 90
Enhanced RF model cross validation accuracy score: 0.8956 +/- 0.0024 (std) min: 0.8928, max: 0.9004, mean: 0.8956, n_estimators: 100
Enhanced RF model cross validation accuracy score: 0.8957 +/- 0.0023 (std) min: 0.8925, max: 0.9004, mean: 0.8957, n_estimators: 110
Enhanced RF model cross validation accuracy score: 0.896 +/- 0.0024 (std) min: 0.8925, max: 0.9003, mean: 0.896, n_estimators: 120
Enhanced RF model cross validation accuracy score: 0.8959 +/- 0.0025 (std) min: 0.8927, max: 0.9006, mean: 0.8959, n_estimators: 130
Enhanced RF model cross validation accuracy score: 0.8958 +/- 0.0023 (std) min: 0.893, max: 0.9, mean: 0.8958, n_estimators: 140
Enhanced RF model cross validation accuracy score: 0.8956 +/- 0.0021 (std) min: 0.893, max: 0.8994, mean: 0.8956, n_estimators: 150
Enhanced RF model cross validation accuracy score: 0.8956 +/- 0.0024 (std) min: 0.8927, max: 0.8998, mean: 0.8956, n_estimators: 160
Enhanced RF model cross validation accuracy score: 0.8957 +/- 0.0023 (std) min: 0.893, max: 0.8994, mean: 0.8957, n_estimators: 170
Enhanced RF model cross validation accuracy score: 0.8959 +/- 0.0022 (std) min: 0.893, max: 0.8999, mean: 0.8959, n_estimators: 180
```



通过学习曲线可以看出：模型在 120 左右的参数区间内表现最好，于是可以根据该图缩小参数范围，本次实验尝试 118、119、121、122 这些参数，结果显示模型在 `n_estimators=118` 的时候表现最优。

```
rf_model_enh = RandomForestClassifier(n_estimators=118,max_features=0.4,min_samples_split=2,random_state=42)
split = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
model_pipe = Pipeline(steps=[('preprocessor', preprocessor), ('model', rf_model_enh)])
cv_results = cross_val_score(model_pipe, X, y, cv=split, scoring="accuracy", n_jobs=-1)
min_score = round(min(cv_results), 4)
max_score = round(max(cv_results), 4)
mean_score = round(np.mean(cv_results), 4)
std_dev = round(np.std(cv_results), 4)
print(f"Enhanced RF model cross validation accuracy score: {mean_score} +/- {std_dev} (std) min: {min_score}, max: {max_score}, mean: {mean_score}")

print("119")
rf_model_enh = RandomForestClassifier(n_estimators=119,max_features=0.4,min_samples_split=2,random_state=42)
split = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
model_pipe = Pipeline(steps=[('preprocessor', preprocessor), ('model', rf_model_enh)])
cv_results = cross_val_score(model_pipe, X, y, cv=split, scoring="accuracy", n_jobs=-1)
min_score = round(min(cv_results), 4)
max_score = round(max(cv_results), 4)
mean_score = round(np.mean(cv_results), 4)
std_dev = round(np.std(cv_results), 4)
print(f"Enhanced RF model cross validation accuracy score: {mean_score} +/- {std_dev} (std) min: {min_score}, max: {max_score}, mean: {mean_score}")

print("121")
rf_model_enh = RandomForestClassifier(n_estimators=121,max_features=0.4,min_samples_split=2,random_state=42)
split = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
model_pipe = Pipeline(steps=[('preprocessor', preprocessor), ('model', rf_model_enh)])
cv_results = cross_val_score(model_pipe, X, y, cv=split, scoring="accuracy", n_jobs=-1)
min_score = round(min(cv_results), 4)
max_score = round(max(cv_results), 4)
mean_score = round(np.mean(cv_results), 4)
std_dev = round(np.std(cv_results), 4)
print(f"Enhanced RF model cross validation accuracy score: {mean_score} +/- {std_dev} (std) min: {min_score}, max: {max_score}, mean: {mean_score}")

print("122")
rf_model_enh = RandomForestClassifier(n_estimators=122,max_features=0.4,min_samples_split=2,random_state=42)
split = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
model_pipe = Pipeline(steps=[('preprocessor', preprocessor), ('model', rf_model_enh)])
cv_results = cross_val_score(model_pipe, X, y, cv=split, scoring="accuracy", n_jobs=-1)
min_score = round(min(cv_results), 4)
max_score = round(max(cv_results), 4)
mean_score = round(np.mean(cv_results), 4)
std_dev = round(np.std(cv_results), 4)
print(f"Enhanced RF model cross validation accuracy score: {mean_score} +/- {std_dev} (std) min: {min_score}, max: {max_score}, mean: {mean_score}")
```

```
118
Enhanced RF model cross validation accuracy score: 0.8959 +/- 0.0022 (std) min: 0.893, max: 0.9001, mean:0.8959
119
Enhanced RF model cross validation accuracy score: 0.8959 +/- 0.0023 (std) min: 0.8924, max: 0.8998, mean:0.8959
121
Enhanced RF model cross validation accuracy score: 0.8958 +/- 0.0023 (std) min: 0.8928, max: 0.8999, mean:0.8958
122
Enhanced RF model cross validation accuracy score: 0.8959 +/- 0.0024 (std) min: 0.8929, max: 0.9, mean:0.8959
```

4、特征评分

```
#拟合模型，以便可以访问值：
model_pipe.fit(X, y)

#需要所有（编码）功能的名称。
#从一个热编码中获取列的名称：
onehot_columns = list(model_pipe.named_steps['preprocessor'].
                       named_transformers_['cat'].
                       named_steps['onehot'].
                       get_feature_names(input_features=cat_features))

#为完整列表添加num_功能。
#顺序必须与X的定义相同，其中num_特征是第一个：
feat_imp_list = num_features + onehot_columns

#显示10个最重要的功能，提供功能名称：
feat_imp_df = eli5.formatters.as_dataframe.explain_weights_df(
    model_pipe.named_steps['model'],
    feature_names=feat_imp_list)
print(feat_imp_df.head(10))
```

| | feature | weight | std |
|---|---------------------------|----------|----------|
| 0 | lead_time | 0.146739 | 0.014666 |
| 1 | deposit_type_Non Refund | 0.121359 | 0.107476 |
| 2 | adr | 0.100540 | 0.003240 |
| 3 | deposit_type_No Deposit | 0.083718 | 0.104972 |
| 4 | country_PRT | 0.065081 | 0.029661 |
| 5 | agent | 0.054212 | 0.011421 |
| 6 | total_of_special_requests | 0.050366 | 0.010703 |
| 7 | stays_in_week_nights | 0.040145 | 0.002162 |
| 8 | market_segment_Online TA | 0.036139 | 0.016091 |
| 9 | previous_cancellations | 0.027699 | 0.014852 |

可以看出权重最大的三个特征是提前预定的时长、押金类型（不退回）和人均价格。其中提前预定的时长和人均价格都在前文的探索性分析中被证明与是否取消订单高度相关。