银行客户定期存款意向二分类预测

> 1452458 叶承玮

- 任务：利用提供的银行客户信息对客户是否会定期存款进行二分类预测。
- 历时：约一个月
- 使用语言：Python
- 详细说明：https://www.kaggle.com/c/tonhjiprmidterm
- 尝试使用分类器：sklearn中的LogisticRegression()、LinearRegression()、RandomForestClassifier()

以下是本次任务的过程：

数据导入及预处理

需要包含的库：

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
```

读取MT_Train.csv和MT_Test.csv

```
train_data = pd.read_csv('MT_Train.csv')
pred_data = pd.read_csv('MT_Test.csv')
```

提取每一列数据并打上标记

```
# do preprocessing
strIndex = train_data.columns[train_data.dtypes == object]
for i in strIndex:
    le = LabelEncoder()
    le.fit(train_data[i])
    train_data[i] = le.transform(train_data[i])
    try:
        pred_data[i] = le.transform(pred_data[i])
    except:
        pass
```

选取输入的属性X和输出的值y

```
X = train_data.iloc[:, :-1]
y = train_data.iloc[:, -1]
X_train, X_test, y_train, y_test = train_test_split(X, y)

# prepare test data
X_pred = pred_data.iloc[:, 1:]
```

对不同分类器的尝试

1.用LinearRegression()预测

需要包含的库：

```
from sklearn.linear_model import LinearRegression
```

定义LinearRegression的缩写

```
mx = LinearRegression()
mx.fit(x_train.values, y_train.values)
```

用.predict方法进行预测（得到0或1的值）

```
y_pred = {}
y_predY=[]
y_predS=[]
y_predNum = mx.predict(X_pred)
y_predNum = y_predNum.round().astype(int)
```

把0记为no，1记为yes，存入列表，转换为DataFrame，并输出

```
for i in y_predNum:
    if i==0:
        y_predY.append('no')
    else:
        y_predY.append('yes')
leng = len(y_predNum)
for i in range(0,leng):
    y_predS.append(i)
y_pred['sampleId'] = y_predS
y_pred['y'] = y_predY
data = pd.DataFrame(y_pred)
data.to_csv('MT_y_linearpred.csv', index=False)
```

2.用LogisticRegression()预测

需要包含的库：

```
from sklearn.linear_model import LogisticRegression
```

定义LogisticRegression的缩写

```
mx = LogisticRegression()
mx.fit(x_train.values, y_train.values)
```

用.predict方法进行预测（得到0或1的值）

```
y_pred = {}
y_predY=[]
y_predS=[]
y_predNum = mx.predict(X_pred)
y_predNum = y_predNum.round().astype(int)
```

把0记为no，1记为yes，存入列表，转换为DataFrame，并输出

```
for i in y_predNum:
    if i==0:
        y_predY.append('no')
    else:
        y_predY.append('yes')
leng = len(y_predNum)
for i in range(0,leng):
    y_predS.append(i)
y_pred['sampleId'] = y_predS
y_pred['y'] = y_predY
data = pd.DataFrame(y_pred)
data.to_csv('MT_y_logisticpred.csv', index=False)
```

3.用RandomForestClassifier()预测

随机森林(Random Forest)在以决策树为基学习器构建Bagging集成的基础上，进一步在决策树的训练过程中引入了随机属性选择。随机森林简单、容易实现、计算开销小，在很多现实任务中展现出强大的性能。随机森林中基学习器的多样性不仅来自样本扰动，还来自属性扰动，这就使最终集成的泛化性能可通过个体学习器之间差异度的增加而进一步提升。随机森林的起始性能往往相对较差，但随着个体学习器数目的增加，通常会收敛到更低的泛化误差。 需要包含的库：

```
from sklearn.ensemble import RandomForestClassifier
```

定义RandomForestClassifier的缩写

```
clf = RandomForestClassifier(n_estimators=12000, n_jobs=-1, class_weight={0:5})
clf.fit(X, y)
```

用.predict方法进行预测（得到0或1的值）

```
y_pred = {}
y_predY=[]
y_predS=[]
y_predNum = clf.predict(X_pred)
y_predNum = y_predNum.round().astype(int)
```

把0记为no，1记为yes，存入列表，转换为DataFrame，并输出

```
for i in y_predNum:
    if i==0:
        y_predY.append('no')
    else:
        y_predY.append('yes')
leng = len(y_predNum)
for i in range(0,leng):
    y_predS.append(i)
y_pred['sampleId'] = y_predS
y_pred['y'] = y_predY
data = pd.DataFrame(y_pred)
data.to_csv('RandomForest.csv', index=False)
```

第一次尝试之后的思考

尝试提交三种方法得出的预测结果之后，RandomForest的F1 Score是最好的，LogisticRegression次之，LinearRegression最低，大致都在0.88左右，需要改进。

在之前的算法中，我是直接把.predict方法输出的预测概率作四舍五入，然后把0设定成no，1设定成yes。一般来说，不同的分类器预测某一个样本的概率不应该有太大出入，可能会在很小的范围内波动。当这个概率在远离0.5的地方波动，由于都要作四舍五入的处理，各分类器的最终结果不会有区别。但是，当这个概率在0.5附近波动时，可能有的分类器预测概率是0.49，另一些分类器预测概率是0.51，在作四舍五入的时候就会出现偏差。因此，我想采用修改阈值的方法使算法的预测更加准确。

1、对LinearRegression()的修正

使用.predict_proba方法，产生预测为yes的概率值（此处.predict_proba和.predict都是产生实际概率值。为了和另两种方法的代码保持一致，使用前者），然后在所有概率值上加一个数（相当于阈值降低这个数值）。经过多次试验，+0.1125是能使LinearRegression取得最大F1 Score的参数，在public子集中能取得0.88773的分数。

```
y_pred = {}
y_predY=[]
y_predS=[]
y_predNum = mx.predict_proba(X_pred)
probability = []
length = len(y_predNum)
for i in range(0,length):
    y = y_predNum[i]
    prob = y[-1]+0.1125
    prob = prob.round().astype(int)
    probability.append(prob)

for i in probability:
if i==0:
    y_predY.append('no')
else:
    y_predY.append('yes')
```

2、对LogisticRegression()的修正

LogisticRegression.predict产生值本身就是近似过后的0和1，为了使用原始概率，采用.predict_proba方法。原理同上。经多次试验，+0.15是能使LogisticRegression取得最大F1 Score的参数，在public子集中能取得0.89051的分数。以上两种方法的修正之后在private子集中的最好成绩在0.87940左右。

```
mx = LogisticRegression()
mx.fit(x_train.values, y_train.values)
y_pred = {}
y_predY=[]
y_predS=[]
y_predNum = mx.predict_proba(X_pred)
probability = []
length = len(y_predNum)
for i in range(0,length):
    y = y_predNum[i]
    prob = y[-1]+0.15
    prob = prob.round().astype(int)
    probability.append(prob)

for i in probability:
if i==0:
    y_predY.append('no')
else:
    y_predY.append('yes')
```

3、对RandomForestClassifier()的修正

RandomForestClassifier()产生的结果本身带有随机性，同样的算法每次运行的结果有细微的偏差，我试图通过调整阈值的方法使预测结果达到最优。原理同上。修正后在public子集中最高能达到0.89922,在private子集中最高能达到0.89063的分数（两次最高分不是同一个算法）。

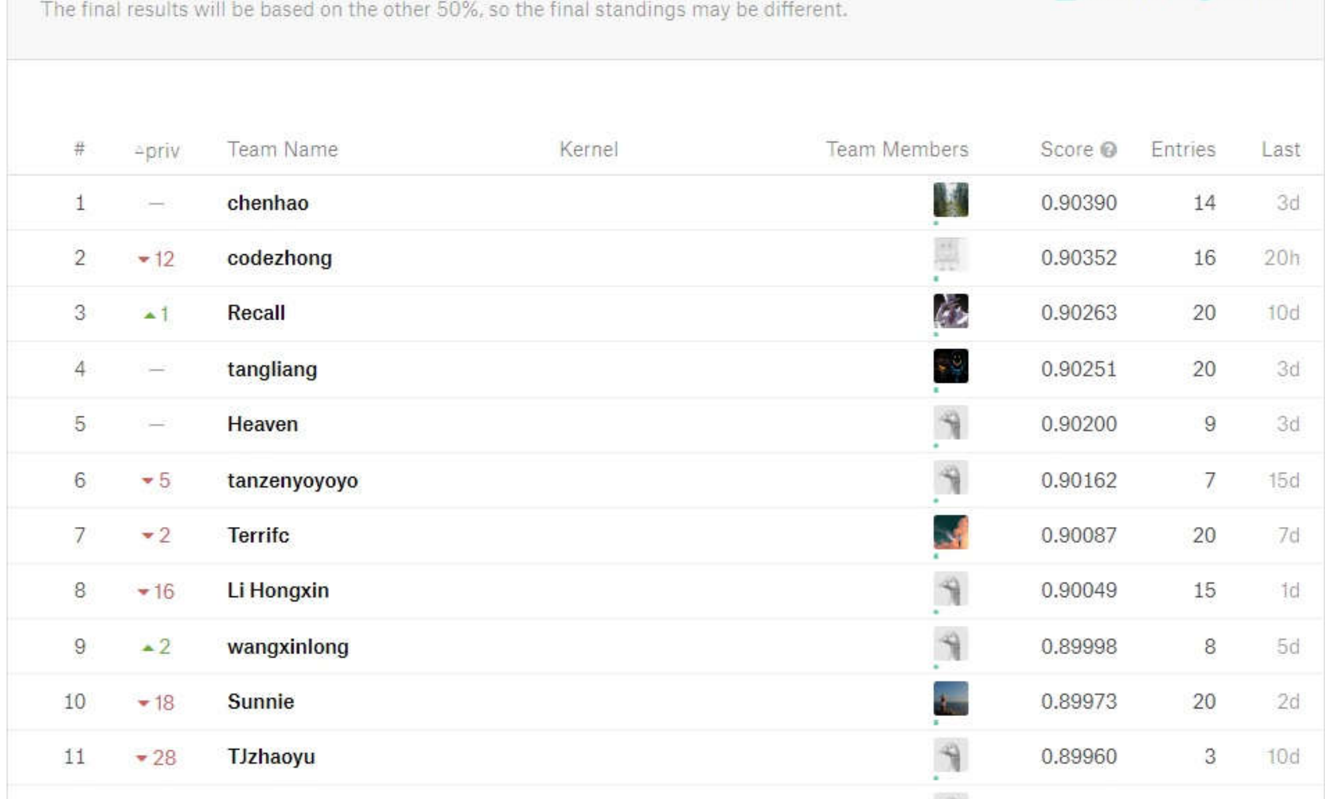
```
clf = RandomForestClassifier(n_estimators=12000, n_jobs=-1, class_weight={0:5})
clf.fit(X, y)
y_pred = clf.predict_proba(X_pred)
probability = []
length = len(y_pred)
for i in range(0,length):
    y = y_pred[i]
    prob = y[-1]+0.025
    prob = prob.round().astype(int)
    probability.append(prob)
```

任务结果与思考

本次任务我尝试了三种不同的分类器来对客户是否会定期存款进行二分类预测。在反复试验中，RandomForestClassifier总体表现最好，LogisticRegression次之，LinearRegression最差。通过修改概率阈值，可以提高预测的F1 Score，但提高幅度不大。

以下是本次竞赛的结果：

Public Leader Board:



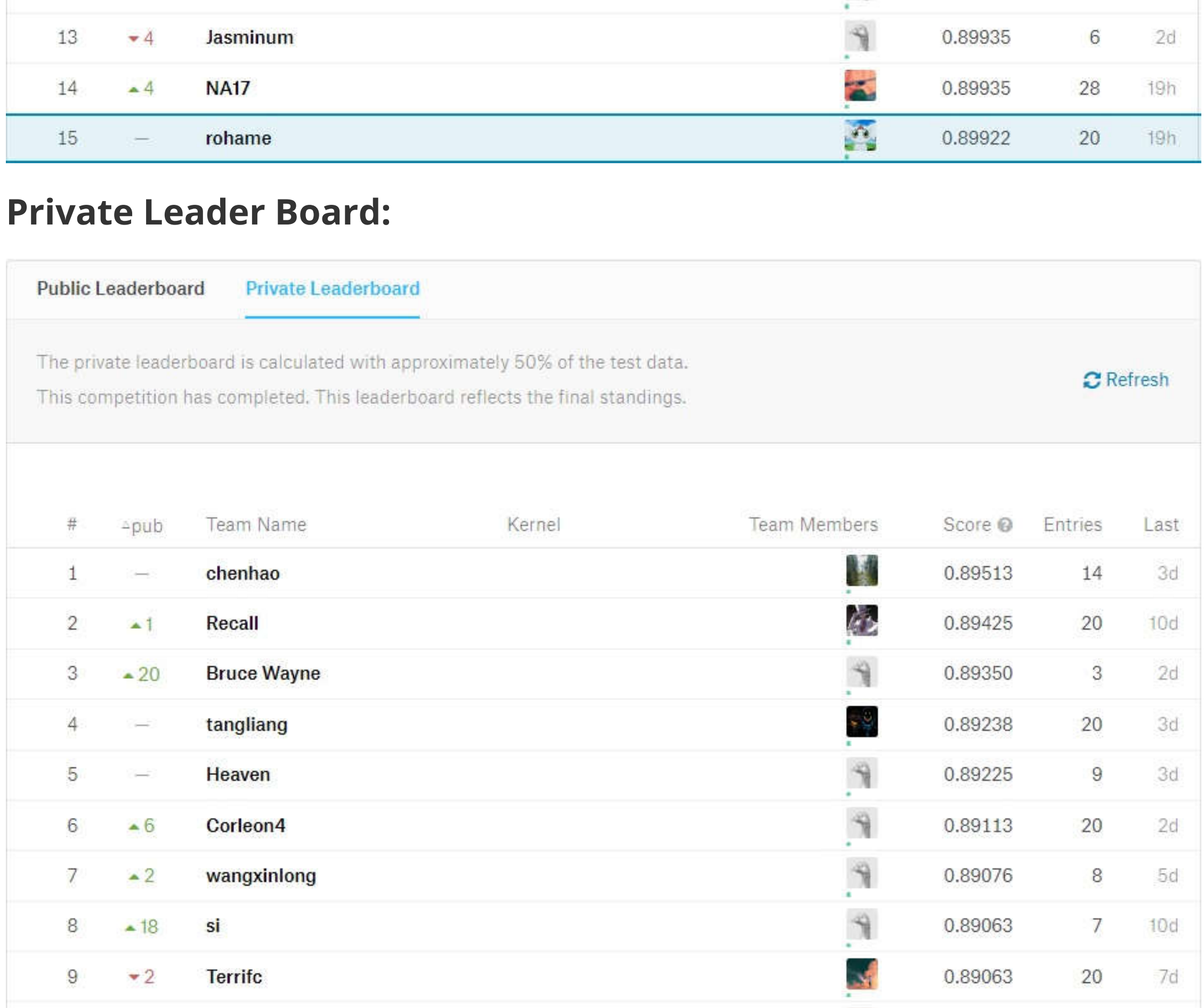
Public Leaderboard Private Leaderboard

This leaderboard is calculated with approximately 50% of the test data.
The final results will be based on the other 50%, so the final standings may be different.

Raw Data Refresh

#	Δpriv	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	chenhao			0.90390	14	3d
2	▼12	codezhong			0.90352	16	20h
3	▲1	Recall			0.90263	20	10d
4	—	tangliang			0.90251	20	3d
5	—	Heaven			0.90200	9	3d
6	▼5	tanzenyoyoyo			0.90162	7	15d
7	▼2	Terrifc			0.90087	20	7d
8	▼16	Li Hongxin			0.90049	15	1d
9	▲2	wangxinlong			0.89998	8	5d
10	▼18	Sunnie			0.89973	20	2d
11	▼28	TJzhaoyu			0.89960	3	10d
12	▲6	Corleon4			0.89948	20	2d
13	▼4	Jasminum			0.89935	6	2d
14	▲4	NA17			0.89935	28	19h
15	—	rohame			0.89922	20	19h

Private Leader Board:



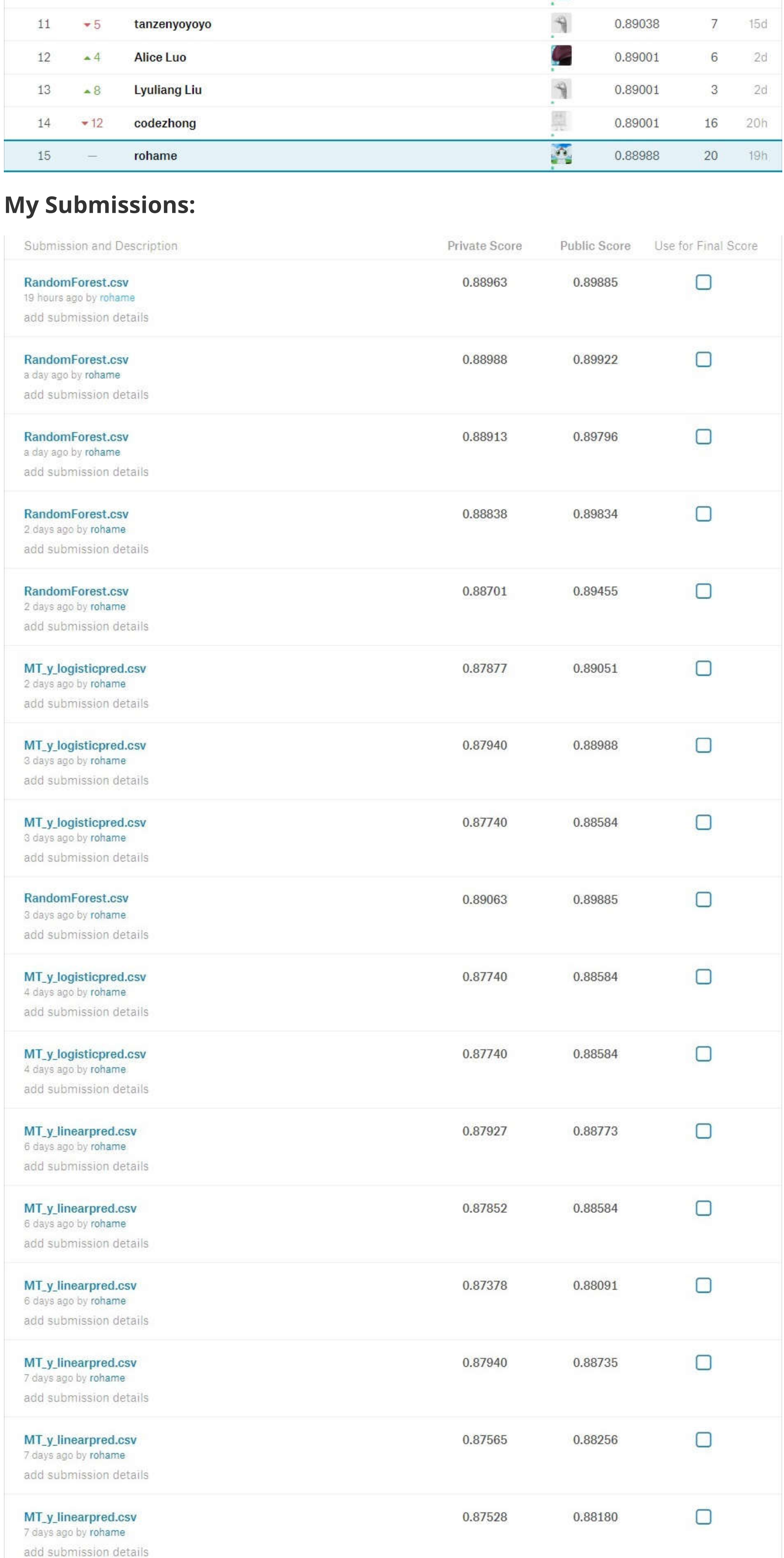
Public Leaderboard Private Leaderboard

The private leaderboard is calculated with approximately 50% of the test data.
This competition has completed. This leaderboard reflects the final standings.

Refresh

#	Δpub	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	chenhao			0.89513	14	3d
2	▲1	Recall			0.89425	20	10d
3	▲20	Bruce Wayne			0.89350	3	2d
4	—	tangliang			0.89238	20	3d
5	—	Heaven			0.89225	9	3d
6	▲6	Corleon4			0.89113	20	2d
7	▲2	wangxinlong			0.89076	8	5d
8	▲18	si			0.89063	7	10d
9	▼2	Terrifc			0.89063	20	7d
10	▲4	NA17			0.89051	28	19h
11	▼5	tanzenyoyoyo			0.89038	7	15d
12	▲4	Alice Luo			0.89001	6	2d
13	▲8	Lyuliang Liu			0.89001	3	2d
14	▼12	codezhong			0.89001	16	20h
15	—	rohame			0.88988	20	19h

My Submissions:



Submission and Description	Private Score	Public Score	Use for Final Score
RandomForest.csv 19 hours ago by rohame add submission details	0.88963	0.89885	☐
RandomForest.csv a day ago by rohame add submission details	0.88988	0.89922	☐
RandomForest.csv a day ago by rohame add submission details	0.88913	0.89796	☐
RandomForest.csv 2 days ago by rohame add submission details	0.88838	0.89834	☐
RandomForest.csv 2 days ago by rohame add submission details	0.88701	0.89455	☐
MT_y_logisticpred.csv 2 days ago by rohame add submission details	0.87877	0.89051	☐
MT_y_logisticpred.csv 3 days ago by rohame add submission details	0.87940	0.88988	☐
MT_y_logisticpred.csv 3 days ago by rohame add submission details	0.87740	0.88584	☐
RandomForest.csv 3 days ago by rohame add submission details	0.89063	0.89885	☐
MT_y_logisticpred.csv 4 days ago by rohame add submission details	0.87740	0.88584	☐
MT_y_logisticpred.csv 4 days ago by rohame add submission details	0.87740	0.88584	☐
MT_y_linearpred.csv 6 days ago by rohame add submission details	0.87927	0.88773	☐
MT_y_linearpred.csv 6 days ago by rohame add submission details	0.87852	0.88584	☐
MT_y_linearpred.csv 6 days ago by rohame add submission details	0.87378	0.88091	☐
MT_y_linearpred.csv 7 days ago by rohame add submission details	0.87940	0.88735	☐
MT_y_linearpred.csv 7 days ago by rohame add submission details	0.87565	0.88256	☐
MT_y_linearpred.csv 7 days ago by rohame add submission details	0.87528	0.88180	☐
linear+0.02.py 7 days ago by rohame add submission details	NULL	Error	☐
MT_y_linearpred.csv 9 days ago by rohame add submission details	0.86953	0.87296	☐
MT_y_linearpred.csv 9 days ago by rohame add submission details	0.87440	0.87713	☐
MT_y_linearpred.csv 9 days ago by rohame add submission details	0.87415	0.87788	☐

No more submissions to show

附：最终Python代码

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier

# load raw data
train_data = pd.read_csv('MT_Train.csv')
pred_data = pd.read_csv('MT_Test.csv')

# do preprocessing
strIndex = train_data.columns[train_data.dtypes == object]
for i in strIndex:
    le = LabelEncoder()
    le.fit(train_data[i])
    train_data[i] = le.transform(train_data[i])
    try:
        pred_data[i] = le.transform(pred_data[i])
    except:
        pass

# prepare train data
X = train_data.iloc[:, :-1]
y = train_data.iloc[:, -1]

# prepare test data
X_pred = pred_data.iloc[:, 1:]

clf = RandomForestClassifier(n_estimators=12000, n_jobs=-1, class_weight={0:5})
clf.fit(X, y)

y_pred = clf.predict_proba(X_pred)
probability = []
length = len(y_pred)
for i in range(0,length):
    y = y_pred[i]
    prob = y[-1]+0.025
    prob = prob.round().astype(int)
    probability.append(prob)

le = LabelEncoder()
le.fit(["yes", "no"])
probability = le.inverse_transform(probability)

data = pd.DataFrame({'sampleId':range(0,len(probability)),'y':probability})
data.to_csv('RandomForest.csv', index=False)
```