



# 《数据挖掘》 实验报告

Spaceship Titanic | Kaggle

姓 名：段欣然

学 号：202011081033

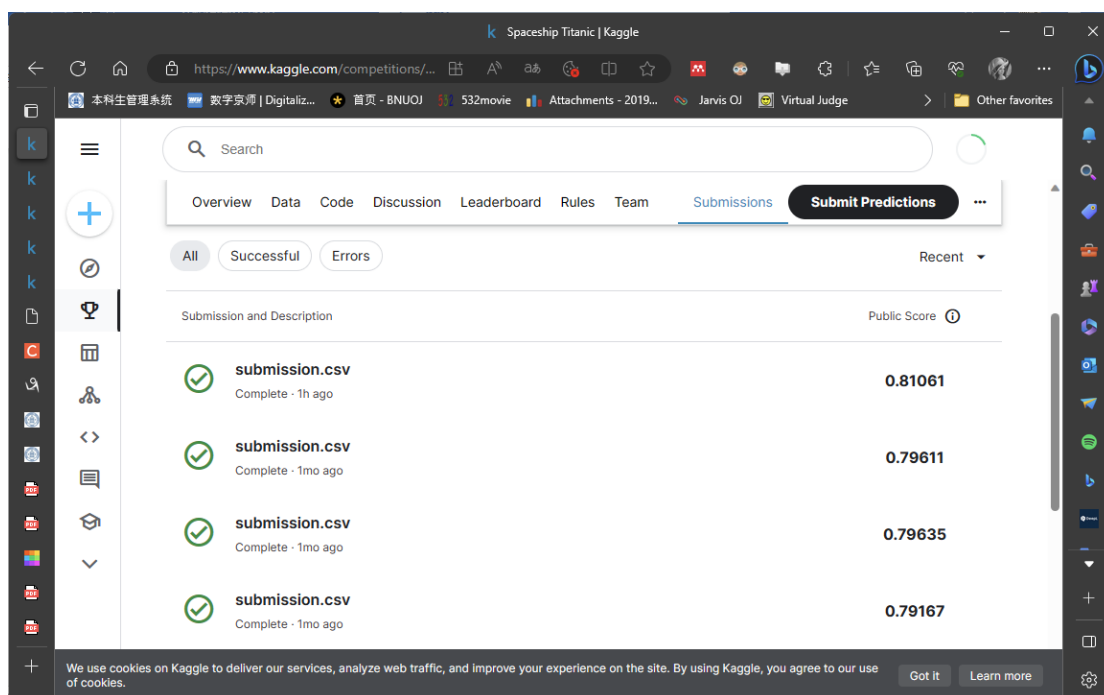
专 业：计算机科学与技术

年 级：2020

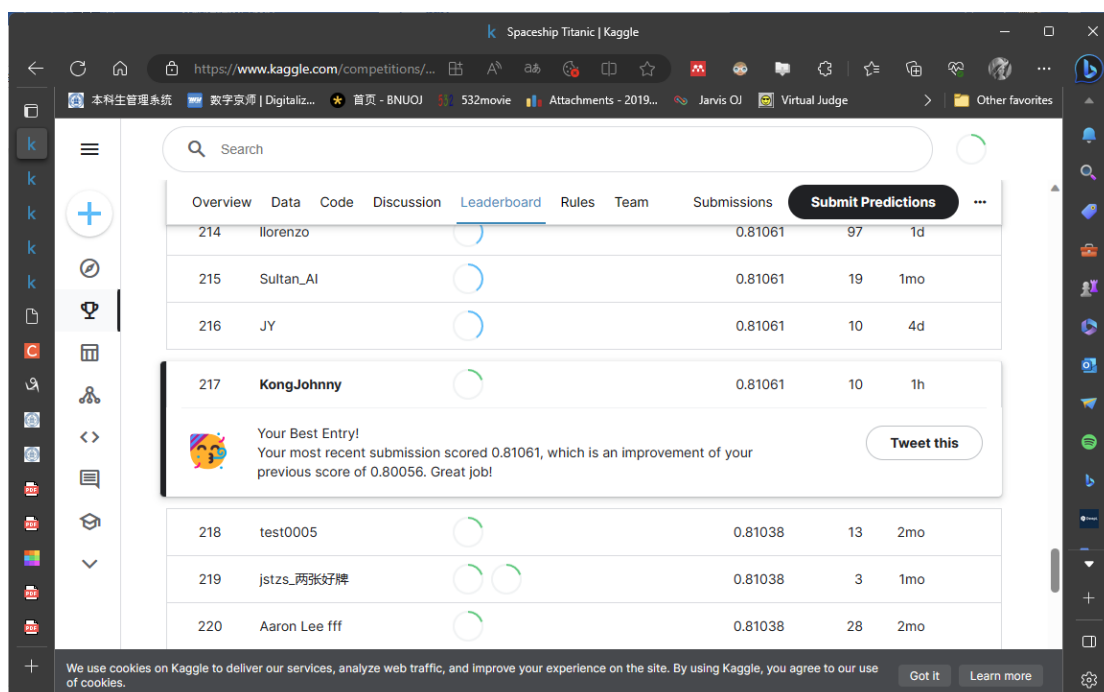
完成日期： 2022 年 6 月

## 一、竞赛并提交的记录

- (1) 账号: [KongJohnny](#)
- (2) 提交的 notebook 网址: [notebook35d36512bc | Kaggle](#)
- (3) 提交记录的截图:



- (4) Leaderboard 分数的截图, 并标注。



## 二、数据挖掘方案

### (1) 问题定义

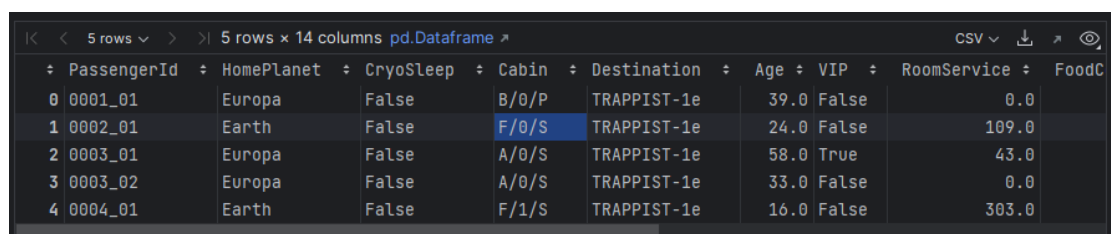
本次实验的问题是根据给定的数据集，构建一个模型来预测乘客是否会被运送到目的地。

### (2) 数据 EDA 与预处理

在这一部分，对原始数据进行了探索性数据分析（EDA）和预处理的步骤。

首先，使用 pandas 库读取了训练数据和测试数据，并使用 head()方法查看了训练数据的前几行。

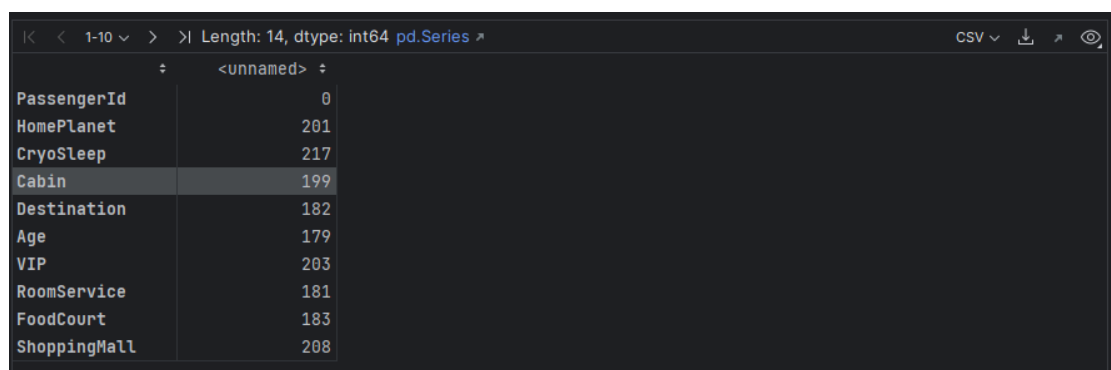
```
1. import pandas as pd
2. #%%
3. train_data = pd.read_csv('data/train.csv')
4. test_data = pd.read_csv('data/test.csv')
5. train_data.head()
```



	PassengerId	HomePlanet	CryoSleep	Cabin	Destination	Age	VIP	RoomService	FoodCourt
0	0001_01	Europa	False	B/0/P	TRAPPIST-1e	39.0	False	0.0	0.0
1	0002_01	Earth	False	F/0/S	TRAPPIST-1e	24.0	False	109.0	0.0
2	0003_01	Europa	False	A/0/S	TRAPPIST-1e	58.0	True	43.0	0.0
3	0003_02	Europa	False	A/0/S	TRAPPIST-1e	33.0	False	0.0	0.0
4	0004_01	Earth	False	F/1/S	TRAPPIST-1e	16.0	False	303.0	0.0

然后，使用 isnull().sum()方法计算了训练数据中的缺失值数量。

```
1. train_data.isnull().sum()
```



	<unnamed>
PassengerId	0
HomePlanet	201
CryoSleep	217
Cabin	199
Destination	182
Age	179
VIP	203
RoomService	181
FoodCourt	183
ShoppingMall	208

接下来，使用 `sklearn.impute.KNNImputer` 类对训练数据和测试数据中的缺失值进行了处理。具体地，将 `Transported` 和 `CryoSleep` 列中的数据转换为数值类型，并使用 KNN 插补方法对其他特征的缺失值进行填充。同时，对 `HomePlanet`、`Destination`、`Deck` 和 `Side` 列的缺失值进行了填充。

```
1. from sklearn.impute import KNNImputer
2. train_data['Transported'] = train_data['Transported']*1
3. train_data['CryoSleep'] = train_data['CryoSleep']*1
4. test_data['CryoSleep'] = test_data['CryoSleep']*1
5.
6. imputer = KNNImputer(n_neighbors=5, weights='uniform', metric='na
   n_euclidean')
7.
8. train_data[['Deck', 'Num', 'Side']] = train_data['Cabin'].str.spl
   it('/', expand=True)
9. train_data[['Age', 'RoomService', 'FoodCourt', 'ShoppingMall', 'S
   pa', 'VRDeck', 'CryoSleep', 'Num']] = imputer.fit_transform(train
   _data[['Age', 'RoomService', 'FoodCourt', 'ShoppingMall', 'Spa',
   'VRDeck', 'CryoSleep', 'Num']])
10. train_data['HomePlanet'].fillna('Earth', inplace=True)
11. train_data['Destination'].fillna('TRAPPIST-1e', inplace=True)
12. train_data['Deck'].fillna('F', inplace=True)
13. train_data['Side'].fillna('P', inplace=True)
14.
15. test_data[['Deck', 'Num', 'Side']] = test_data['Cabin'].str.split
   ('/', expand=True)
16. test_data[['Age', 'RoomService', 'FoodCourt', 'ShoppingMall', 'Sp
   a', 'VRDeck', 'CryoSleep', 'Num']] = imputer.fit_transform(test_d
   ata[['Age', 'RoomService', 'FoodCourt', 'ShoppingMall', 'Spa', 'V
   RDeck', 'CryoSleep', 'Num']])
17. test_data['HomePlanet'].fillna('Earth', inplace=True)
18. test_data['Destination'].fillna('TRAPPIST-1e', inplace=True)
19. test_data['Deck'].fillna('F', inplace=True)
20. test_data['Side'].fillna('P', inplace=True)
```

随后，使用 `sklearn.preprocessing.LabelEncoder` 类对训练数据和测试数据中的 `Deck`、`Num` 和 `Side` 列进行了标签编码。

```
1. from sklearn.preprocessing import LabelEncoder
```

```

2. encoder = LabelEncoder()
3. train_data['Deck'] = encoder.fit_transform(train_data['Deck'])
4.
5. encoder = LabelEncoder()
6. train_data['Num'] = encoder.fit_transform(train_data['Num'])
7.
8. encoder = LabelEncoder()
9. train_data['Side'] = encoder.fit_transform(train_data['Side'])
10.
11. encoder = LabelEncoder()
12. test_data['Deck'] = encoder.fit_transform(test_data['Deck'])
13.
14. encoder = LabelEncoder()
15. test_data['Num'] = encoder.fit_transform(test_data['Num'])
16.
17. encoder = LabelEncoder()
18. test_data['Side'] = encoder.fit_transform(test_data['Side'])

```

为了进一步进行特征工程，创建了一个名为 AllSpending 的新特征，该特征表示乘客在 RoomService、FoodCourt、ShoppingMall、Spa 和 VRDeck 上的总消费。

```

1. train_data['AllSpending'] = train_data['RoomService'] + train_data['FoodCourt'] + train_data['ShoppingMall'] + train_data['Spa'] + train_data['VRDeck']
2. test_data['AllSpending'] = test_data['RoomService'] + test_data['FoodCourt'] + test_data['ShoppingMall'] + test_data['Spa'] + test_data['VRDeck']

```

最后，使用 sklearn.preprocessing.StandardScaler 对训练数据和测试数据中的 RoomService、Spa、VRDeck 和 AllSpending 列进行了特征缩放，并使用 pd.get\_dummies 方法对特征进行了独热编码。

```

1. from sklearn.preprocessing import StandardScaler
2. train_label = train_data['Transported']
3. features = ['CryoSleep', 'RoomService', 'Spa', 'VRDeck', 'Deck', 'Side', 'AllSpending']
4.
5. cols_to_norm = ['RoomService', 'Spa', 'VRDeck', 'AllSpending']
6. scaler = StandardScaler()
7.

```

```
8. train_data[cols_to_norm] = scaler.fit_transform(train_data[cols_to_norm])
9. test_data[cols_to_norm] = scaler.transform(test_data[cols_to_norm])
10.
11. train_input = pd.get_dummies(train_data[features])
12. test_input = pd.get_dummies(test_data[features])
```

### (3) 特征工程

在特征工程阶段，进行了以下操作：

- 对 Deck、Num 和 Side 列使用了标签编码，将其转换为数值形式以便于模型训练；
- 创建了新的特征 AllSpending，表示乘客在不同消费项目上的总消费。

相应代码已在上文给出。

### (4) 模型构建

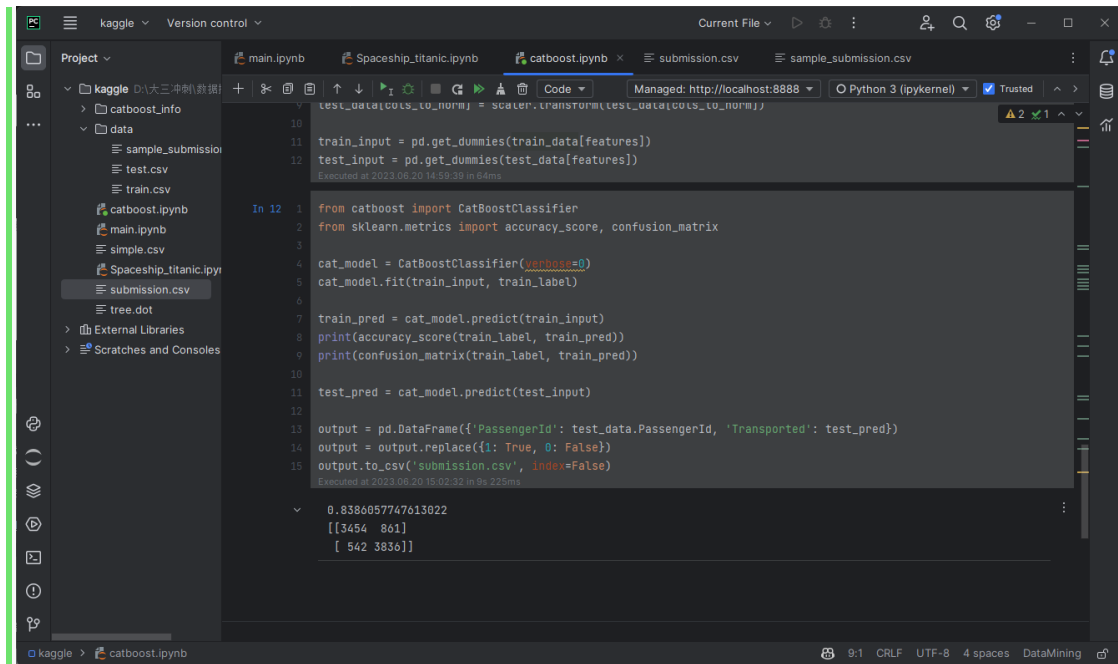
在模型构建阶段，使用了 CatBoostClassifier 模型。首先，导入了 CatBoostClassifier 类和 accuracy\_score、confusion\_matrix 指标。然后，实例化了 CatBoostClassifier 模型，并使用训练数据进行了模型训练。接着，对训练数据进行了预测，并使用 accuracy\_score 和 confusion\_matrix 计算了训练集上的准确率和混淆矩阵。最后，对测试数据进行了预测，并将结果保存为 submission.csv 文件。

```
1. from catboost import CatBoostClassifier
2. from sklearn.metrics import accuracy_score, confusion_matrix
3.
4. cat_model = CatBoostClassifier(verbose=0)
5. cat_model.fit(train_input, train_label)
6.
7. train_pred = cat_model.predict(train_input)
8. print(accuracy_score(train_label, train_pred))
9. print(confusion_matrix(train_label, train_pred))
10.
```

```

11.test_pred = cat_model.predict(test_input)
12.
13.output = pd.DataFrame({'PassengerId': test_data.PassengerId, 'Transported': test_pred})
14.output = output.replace({1: True, 0: False})
15.output.to_csv('submission.csv', index=False)

```



### 三、问题回答

(1) 必答题：你的方案里的哪一项技术（数据处理、模型改进）对提高分数产生了显著作用？结合理论知识对原因进行解释

1. 缺失值处理：在数据预处理阶段，使用 `sklearn.impute.KNNImputer` 对缺失值进行了处理。KNN 插补是一种基于相似性的方法，它使用邻近样本的信息来估计缺失值。通过填充缺失值，我们避免了对带有缺失值的样本进行删除或简单填充的情况。这有助于保留更多有用的信息，可能提高了模型的性能。
2. 特征工程：在特征工程阶段，创建了新的特征 `AllSpending`，表示乘客在不同消费项目上的总消费。这个新特征捕捉了乘客在多个消费项目上的总体消费情况。这可能有助于模型更好地理解乘客的消费行为，并提供更丰富的信息，从而提高了预测的准确性。
3. 模型选择：在模型构建阶段，选择了 `CatBoostClassifier` 作为分类器模

型。CatBoost 是一种基于梯度提升决策树 (GBDT) 的强大机器学习算法，具有处理类别特征、自动特征缩放和更好的泛化能力等优点。相比传统的 GBDT 算法，CatBoost 能够更好地处理类别特征，无需额外的特征编码。因此，模型选择可能对提高分数产生显著作用。

(2) 选答问题：你的方案的改进或者创新点是什么，并详细说明。

1. 标签编码的应用：在处理 Deck、Num 和 Side 列时，使用了 `sklearn.preprocessing.LabelEncoder` 进行标签编码。标签编码将类别特征转换为数值形式，使得模型能够处理这些特征。这种方法在处理有序类别特征时非常有用，可以将类别之间的顺序关系考虑到模型中。在这种情况下，通过标签编码将类别特征转换为数值特征，可能对模型的性能产生积极影响。
2. 特征缩放的应用：在特征工程的最后阶段，使用了 `sklearn.preprocessing.StandardScaler` 对 RoomService、Spa、VRDeck 和 AllSpending 等列进行了特征缩放。特征缩放对于基于距离或梯度的算法（如 KNN、支持向量机和神经网络）尤其重要，它可以确保不同特征具有相似的尺度。通过特征缩放，可以避免某些特征对模型的训练产生过大的影响，提高了模型的稳定性和收敛速度。
3. 使用 CatBoost 模型进行训练：在模型构建阶段，选择了 `CatBoostClassifier` 作为分类器模型。CatBoost 在梯度提升决策树 (GBDT) 的基础上，通过处理类别特征、自动特征缩放和更好的泛化能力等方面进行了改进。相比传统的 GBDT 算法，CatBoost 能够更好地处理类别特征，无需额外的特征编码。这种模型选择可能对最终的预测性能和模型的鲁棒性有积极的影响。