|  |
| --- |
| 修改后计算机学院 |
| **南华大学计算机学院** |

课程设计报告

（ 2023学年春季学期 ）

|  |  |
| --- | --- |
| 课程名称 | 机器学习 |
| **项目名称** | **天池--二手车价格预测** |

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | 付翔 | 学号 | 20214352118 |
| 专业 | 人工智能 | 教师 | 熊东平 |

# 一、项目简介

本项目是阿里云天池上的一个比赛，旨在通过对二手车数据进行建模，预测二手车的交易价格。该比赛使用的数据集来自某交易平台的二手车交易记录，包含31列变量信息，其中15列为匿名变量。为保证比赛的公平性，将从中抽取15万条作为训练集，5万条作为测试集A，5万条作为测试集B，同时会对name、model、brand和regionCode等信息进行脱敏。

比赛链接：

<https://tianchi.aliyun.com/competition/entrance/231784/information>

本项目的任务是使用数据集中的变量，建立模型，预测二手车的交易价格。在模型建立过程中，需要进行数据预处理、特征选择和模型调优等操作，并通过交叉验证等方法评估所建模型的性能。

# 二、数据预处理

本项目中，需要进行的数据预处理主要包括缺失值填充、异常值处理和日期数据处理等。

| **Field** | **Description** |
| --- | --- |
| SaleID | 交易ID，唯一编码 |
| name | 汽车交易名称，已脱敏 |
| regDate | 汽车注册日期，例如20160101，2016年01月01日 |
| model | 车型编码，已脱敏 |
| brand | 汽车品牌，已脱敏 |
| bodyType | 车身类型：豪华轿车：0，微型车：1，厢型车：2，大巴车：3，敞篷车：4，双门汽车：5，商务车：6，搅拌车：7 |
| fuelType | 燃油类型：汽油：0，柴油：1，液化石油气：2，天然气：3，混合动力：4，其他：5，电动：6 |
| gearbox | 变速箱：手动：0，自动：1 |
| power | 发动机功率：范围 [ 0, 600 ] |
| kilometer | 汽车已行驶公里，单位万km |
| notRepairedDamage | 汽车有尚未修复的损坏：是：0，否：1 |
| regionCode | 地区编码，已脱敏 |
| seller | 销售方：个体：0，非个体：1 |
| offerType | 报价类型：提供：0，请求：1 |
| creatDate | 汽车上线时间，即开始售卖时间 |
| price | 二手车交易价格（预测目标） |
| v系列特征 | 匿名特征，包含v0-14在内15个匿名特征 |

在进行数据预处理之前，首先需要读取训练数据和测试数据，并使用 Pandas 库将它们合并到一起。合并后的数据集总共有 20 万条数据，并包含 31 列变量信息，其中 15 列为匿名变量。

* 读取训练数据和测试数据。
* 使用 Pandas 库将训练数据和测试数据合并到一起。
* 对 'price' 做对数变换。
* 用众数填充缺失值。
* 处理异常值:将功率大于 600 的标为 600。
* 将 ‘notRepairedDamage’ 中的缺失值替换为 None。
* 对可分类的连续特征进行分桶。
* 对日期数据进行处理，提取年、月、日等信息和计算二手车使用时间。

# 三、特征工程

在数据预处理之后，需要进行特征工程，以提取更多有用的特征。本项目中包括对数据进行缺失值填充、数据统计和特征组合等操作。

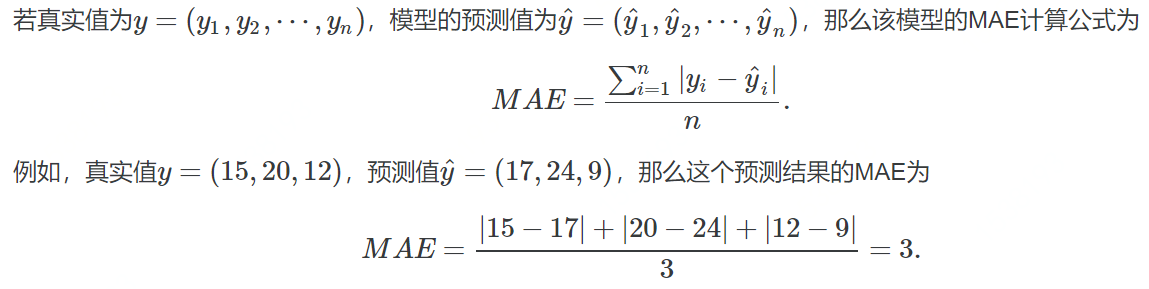
具体的操作包括：

1. 对数据进行缺失值填充。
2. 对行驶路程和功率数据进行统计，例如计算行驶路程与功率的最大值、最小值、中位数和均值等。
3. 为部分属性列的数据生成新的特征，主要是通过对 V0、V3、V8 和 V12 四个特征进行组合生成新的二元和三元特征。

# 四、模型训练与预测

在完成特征工程之后，需要使用机器学习模型对数据进行训练，并预测测试集中二手车的交易价格。本项目中使用的模型是 LightGBMRegressor，使用了四个参数（'n\_estimators'、'learning\_rate'、'num\_leaves'和'lambda\_l2'）进行调参。在训练模型的过程中，使用了 K-Fold 交叉验证，最终得到的结果是五折交叉验证的平均结果。

## 五、结果评估

在使用模型对测试集进行预测之后，需要对结果进行评估。本项目中使用的评估指标是 MAE（Mean Absolute Error），即预测值与实际值之间的平均绝对误差。  
  
MAE越小，说明模型预测得越准确。

结果提交：

提交前请确保预测结果的格式与sample\_submit.csv中的格式一致，以及提交文件后缀名为csv。

形式如下：

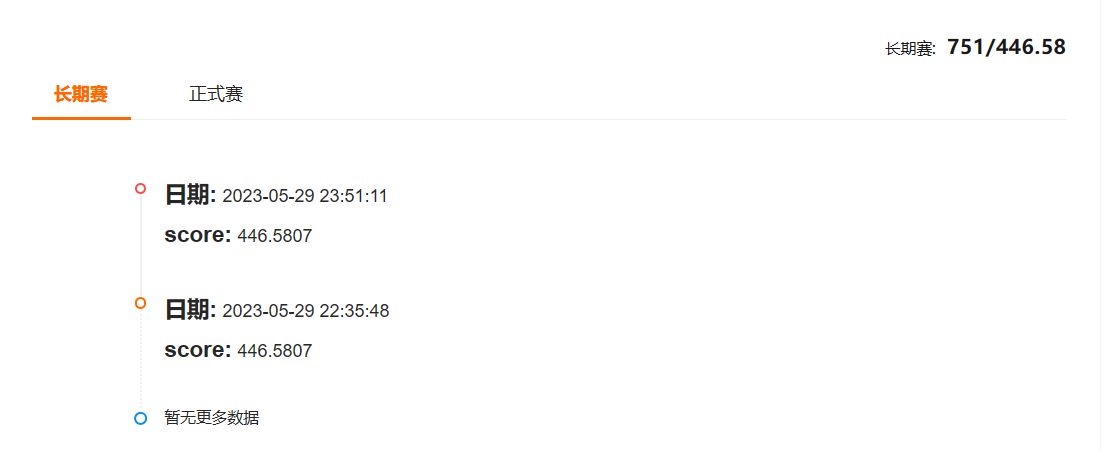
SaleID,price

150000,687

150001,1250

150002,2580

150003,1178



## 六、总结

本项目通过对二手车数据进行建模，预测了二手车的交易价格。在完成项目的过程中，主要涉及到数据预处理、特征工程、模型训练与评估等多个方面。其中特征工程的部分使用了数据的统计和组合等方法，提取了更多有用的特征，从而提升了模型的准确性和稳定性。

在实际应用中，可以通过对数据集进行更加深入的探索和分析，以进一步提高建模的准确性和稳定性，并应用更加高级的机器学习模型和技术来进行建模和预测。

### 附件：

### Readme.md:



### **部分代码：**

# LightGBMRegressor  
# 交叉验证 MAE: 473.3349762655072  
import pandas as pd  
import numpy as np  
from sklearn.model\_selection import KFold  
from lightgbm.sklearn import LGBMRegressor  
from sklearn.metrics import mean\_absolute\_error  
  
# 首先使用 Pandas 库读入训练数据和测试数据，保存到 Train\_data 和 Test\_data 变量中。  
Train\_data = pd.read\_csv('../data/used\_car\_train\_20200313.csv',  
 sep=' ') # handle\_used\_car\_train.csv  
Test\_data = pd.read\_csv('../data/used\_car\_testB\_20200421.csv', sep=' ')  
  
# 使用 pd.concat() 函数将训练数据和测试数据合并，并保存到 df 变量中。  
df = pd.concat([Train\_data, Test\_data], ignore\_index=True)

# 对 'price' 做对数变换，使用 np.log1p() 函数。  
df['price'] = np.log1p(df['price'])

# 用众数填充缺失值  
df['fuelType'] = df['fuelType'].fillna(0)  
df['gearbox'] = df['gearbox'].fillna(0)  
df['bodyType'] = df['bodyType'].fillna(0)  
df['model'] = df['model'].fillna(0)  
  
# 处理异常值，主要是将功率大于 600 的标为 600  
df['power'] = df['power'].map(lambda x: 600 if x > 600 else x) # 赛题限定power<=600  
# 将 ‘notRepairedDamage’ 中的缺失值替换为 None  
df['notRepairedDamage'] = df['notRepairedDamage'].astype('str').apply(lambda x: x if x != '-' else None).astype(  
 'float32')  
  
# 对可分类的连续特征进行分桶，例如将功率（power）分为 31 组，车型（model）分为 24 组。  
bin = [i \* 10 for i in range(31)]  
df['power\_bin'] = pd.cut(df['power'], bin, labels=False)  
  
bin = [i \* 10 for i in range(24)]  
df['model\_bin'] = pd.cut(df['model'], bin, labels=False)

# 对日期数据进行处理，主要是提取年，月，日等信息和计算二手车使用时间。  
from datetime import datetime  
  
  
def date\_process(x):  
 year = int(str(x)[:4])  
 month = int(str(x)[4:6])  
 day = int(str(x)[6:8])  
  
 if month < 1:  
 month = 1  
  
 date = datetime(year, month, day)  
 return date  
  
  
df['regDate'] = df['regDate'].apply(date\_process)  
df['creatDate'] = df['creatDate'].apply(date\_process)  
df['regDate\_year'] = df['regDate'].dt.year  
df['regDate\_month'] = df['regDate'].dt.month  
df['regDate\_day'] = df['regDate'].dt.day  
df['creatDate\_year'] = df['creatDate'].dt.year  
df['creatDate\_month'] = df['creatDate'].dt.month  
df['creatDate\_day'] = df['creatDate'].dt.day  
  
# 使用天数  
df['car\_age\_day'] = (df['creatDate'] - df['regDate']).dt.days  
# 使用年数  
df['car\_age\_year'] = round(df['car\_age\_day'] / 365, 1)  
  
# 对行驶路程和功率数据进行统计，例如：计算行驶路程与功率的最大值、最小值、中位数和均值等。  
kk = ['kilometer', 'power']  
t1 = Train\_data.groupby(kk[0], as\_index=False)[kk[1]].agg(  
 {kk[0] + '\_' + kk[1] + '\_count': 'count', kk[0] + '\_' + kk[1] + '\_max': 'max',  
 kk[0] + '\_' + kk[1] + '\_median': 'median',  
 kk[0] + '\_' + kk[1] + '\_min': 'min', kk[0] + '\_' + kk[1] + '\_sum': 'sum', kk[0] + '\_' + kk[1] + '\_std': 'std',  
 kk[0] + '\_' + kk[1] + '\_mean': 'mean'})  
df = pd.merge(df, t1, on=kk[0], how='left')

# 为部分属性列的数据生成新的特征，主要是通过对 V0、V3、V8 和 V12 四个特征进行组合生成新的二元和三元特征。  
num\_cols = [0, 3, 8, 12]  
for i in num\_cols:  
 for j in num\_cols:  
 df['new' + str(i) + '\*' + str(j)] = df['v\_' + str(i)] \* df['v\_' + str(j)]  
  
for i in num\_cols:  
 for j in num\_cols:  
 df['new' + str(i) + '+' + str(j)] = df['v\_' + str(i)] + df['v\_' + str(j)]  
  
for i in num\_cols:  
 for j in num\_cols:  
 df['new' + str(i) + '-' + str(j)] = df['v\_' + str(i)] - df['v\_' + str(j)]  
  
for i in range(15):  
 df['new' + str(i) + '\*year'] = df['v\_' + str(i)] \* df['car\_age\_year']

# 使用 LightGBMRegressor 作为模型，对数据进行训练和预测。  
# 对数据进行五折交叉检验，最后通过将五次模型训练得到的结果平均作为最终预测结果，并将结果保存到文件中供提交。  
df1 = df.copy()  
test = df1[df1['price'].isnull()]  
X\_train = df1[df1['price'].notnull()].drop(['price', 'regDate', 'creatDate', 'SaleID', 'regionCode'], axis=1)  
Y\_train = df1[df1['price'].notnull()]['price']  
X\_test = df1[df1['price'].isnull()].drop(['price', 'regDate', 'creatDate', 'SaleID', 'regionCode'], axis=1)  
# 五折交叉检验  
cols = list(X\_train)  
oof = np.zeros(X\_train.shape[0])  
sub = test[['SaleID']].copy()  
sub['price'] = 0  
feat\_df = pd.DataFrame({'feat': cols, 'imp': 0})  
skf = KFold(n\_splits=4, shuffle=True, random\_state=2020)  
  
clf = LGBMRegressor(  
 n\_estimators=10000,  
 learning\_rate=0.07, # 0.02,  
 boosting\_type='gbdt',  
 objective='regression\_l1',  
 max\_depth=-1,  
 num\_leaves=31,  
 min\_child\_samples=20,  
 feature\_fraction=0.8,  
 bagging\_freq=1,  
 bagging\_fraction=0.8,  
 lambda\_l2=2,  
 random\_state=2020,  
 metric='mae'  
)  
  
mae = 0  
for i, (trn\_idx, val\_idx) in enumerate(skf.split(X\_train, Y\_train)):  
 print('--------------------- 第 {} 折 ---------------------'.format(i + 1))  
 trn\_x, trn\_y = X\_train.iloc[trn\_idx].reset\_index(drop=True), Y\_train[trn\_idx]  
 val\_x, val\_y = X\_train.iloc[val\_idx].reset\_index(drop=True), Y\_train[val\_idx]  
 clf.fit(  
 trn\_x, trn\_y,  
 eval\_set=[(val\_x, val\_y)],  
 eval\_metric='mae',  
 early\_stopping\_rounds=300,  
 verbose=300  
 )  
  
 sub['price'] += np.expm1(clf.predict(X\_test)) / skf.n\_splits  
 oof[val\_idx] = clf.predict(val\_x)  
 mae += mean\_absolute\_error(np.expm1(val\_y), np.expm1(oof[val\_idx])) / skf.n\_splits  
  
print('交叉验证 MAE:', mae)

# 生成提交文件  
sub.to\_csv('submit.csv', index=False)