**目录**

[一、 设计目的和任务 1](#_Toc7502)

[二、 开发环境 2](#_Toc6653)

[2.1硬件环境 2](#_Toc7547)

[2.2软件环境 2](#_Toc2061)

[三、 设计题目 3](#_Toc1065)

[3.1题目名称 3](#_Toc19341)

[3.2选题创新性阐述（市场价值、用户需求、服务提升、技术创新等方面） 3](#_Toc13475)

[3.3赛题建模 4](#_Toc9291)

[3.4数据探索 6](#_Toc23178)

[(1) 探索目标 6](#_Toc22106)

[(2) 内容介绍 6](#_Toc15536)

[(3) 代码示例 7](#_Toc7185)

[四、 特征工程 13](#_Toc10162)

[4.1 特征工程目标 13](#_Toc9392)

[4.2 内容介绍 13](#_Toc13959)

[4.3 代码示例 14](#_Toc27964)

[(1) 导入数据 14](#_Toc12382)

[(2) 删除异常值 14](#_Toc28006)

[(3) 特征构造 15](#_Toc11895)

[(4) 特征筛选 15](#_Toc8305)

[五、 模型选择和训练 17](#_Toc4589)

[5.1 学习目标 17](#_Toc29295)

[5.2 内容介绍 17](#_Toc20602)

[5.3 代码示例 18](#_Toc7252)

[(1) 读取数据 18](#_Toc6340)

[(2) 线性回归&五折交叉验证&模拟真实业务情况 18](#_Toc3500)

[(3) 多种模型对比 20](#_Toc4536)

[(4) 模型调参 21](#_Toc14586)

[(5) 小结 21](#_Toc4)

[六、 模型融合和优化 22](#_Toc4941)

[6.1 模型融合目标 22](#_Toc10144)

[6.2 内容介绍 22](#_Toc14352)

[6.3 Stacking相关理论介绍 22](#_Toc14007)

[6.4 代码示例 23](#_Toc8780)

[(1) 回归、分类概率-融合： 23](#_Toc669)

[(2) 分类模型融合 24](#_Toc11115)

[(3) 一些其它方法 25](#_Toc31471)

[(4) 本赛题示例 25](#_Toc30980)

[七、 总结 26](#_Toc1382)

[八、 参考文献 27](#_Toc16281)

[九、 附录：见源程序电子代码 27](#_Toc9602)

# 设计目的和任务

1. **设计目的**

本次课程设计《影响二手车交易价格的关键因素与预测模型构建》旨在通过实际操作和深入研究，巩固和加深自身对《素质拓展特色项目》课程中相关知识的理解和掌握。通过构建一个能够预测二手车交易价格的模型，使自我能够将所学的理论知识与实际问题相结合，提升解决实际问题的能力。同时，通过本次设计，培养学习方式、数据分析和创新思维等综合素质，为自我未来的职业发展奠定坚实的基础。

本次设计的目的包括：

1. 知识应用与巩固：通过构建预测模型，使自我能够将课程中所学的数据分析、机器学习、人工智能等技术知识应用到实际问题中，从而加深对理论知识的理解。
2. 技能提升：通过数据收集、处理、分析和模型构建等过程，提升个人的数据处理能力、模型构建能力和创新思维。
3. 综合素质培养：通过项目管理和设计报告撰写等环节，培养个人的协作精神、项目管理能力和书面表达能力。
4. **设计任务**

本次设计任务要求学生综合运用课程相关知识，完成二手车交易预测模型的构建，并撰写设计报告。具体任务包括：

1. 数据收集与处理：收集二手车交易的相关数据，包括车辆信息、交易价格、市场状况等，并对数据进行清洗、整理和预处理，确保数据的质量和可用性。
2. 关键因素分析：运用统计分析和数据挖掘技术，分析影响二手车交易价格的关键因素，并构建相应的特征集。
3. 预测模型构建：基于关键因素分析的结果，选择合适的机器学习算法（如线性回归、决策树、随机森林、神经网络等），构建二手车交易价格预测模型，并进行模型训练和验证。
4. 模型优化与评估：通过调整模型参数、优化算法选择等手段，提高模型的预测精度和泛化能力，并使用合适的评估指标（如均方误差、准确率等）对模型进行评估。
5. 设计报告撰写：根据设计过程和结果，撰写详细的设计报告，包括数据收集与处理、关键因素分析、预测模型构建、模型优化与评估等内容，并展示设计成果和创新点。

# 开发环境

## 2.1硬件环境

处理器: 12th Gen Intel(R) Core(TM) i7-12650H 2.30 GHz

机带 RAM: 16.0 GB (15.7 GB 可用)

系统类型: 64 位操作系统, 基于 x64 的处理器

## 2.2软件环境

操作系统: windows 11

集成开发环境: Anaconda3

基础库或工具:

|  |
| --- |
| 1. numpy:   NumPy 是 Python 中用于数值计算的库。   1. pandas:   Pandas 是一个用于数据分析和数据处理的库，提供了 DataFrame 和 Series 两种主要数据结构。   1. warnings:   Python 的内置模块，用于处理警告消息。你可以用它来忽略、显示或触发警告。   1. matplotlib:   Matplotlib 是一个用于绘制图形的 Python 库。它提供了一个类似于 MATLAB 的界面，用于生成静态、动态、交互式的可视化图表。   1. matplotlib.pyplot as plt:   Matplotlib 的 pyplot 模块提供了一个类似于 MATLAB 的绘图框架。通过它，你可以使用简单的函数来创建图形。   1. seaborn:   Seaborn 是一个基于 Matplotlib 的数据可视化库。   1. scipy.special.jn:   SciPy 是一个用于数学、科学和工程的开源 Python 库。   1. IPython.display:   IPython 是一个增强的 Python 交互式 shell。   1. time:   Python 的内置模块，用于时间相关的函数。你可以用它来获取当前时间、测量代码的执行时间等。 |

# 设计题目

## 3.1题目名称

影响二手车交易价格的关键因素分析与预测模型构建

## 3.2选题创新性阐述（市场价值、用户需求、服务提升、技术创新等方面）

1. 市场价值

随着汽车市场的日益成熟和消费者购车观念的转变，二手车市场正逐渐成为汽车行业的重要组成部分。本选题紧扣二手车市场的发展脉搏，通过深入分析影响二手车交易的关键因素，并构建预测模型，为市场参与者提供更为精准的决策依据。这不仅有助于提升二手车交易的效率和透明度，还能促进市场资源的优化配置，从而推动整个二手车市场的健康、持续发展。因此，本选题在市场价值方面具有显著的创新性和实用性。

1. 用户需求

当前，消费者对二手车交易的需求日益多样化，他们不仅关注车辆的性能和价格，还重视购车的便捷性和安全性。本选题通过构建预测模型，能够更准确地把握消费者的购车偏好和需求变化，从而为二手车交易平台提供更个性化的服务方案。这种以用户需求为中心的研究思路，有助于提升消费者的购车体验，满足他们对高效、安全、便捷交易的期待。

1. 服务提升

二手车市场的服务质量和水平直接影响到消费者的购车意愿和市场的整体竞争力。本选题通过分析和预测二手车交易的关键因素，有助于二手车交易平台优化服务流程、提升服务质量。例如，通过模型预测，平台可以提前了解消费者的购车需求和偏好，从而为他们提供更合适的车源推荐和购车方案。这种服务模式的创新，不仅能够提高消费者的满意度和忠诚度，还能为平台带来更多的商业机会。

1. 技术创新

在技术创新方面，本选题运用先进的数据分析和机器学习技术，构建二手车交易预测模型。这种技术创新不仅提高了数据处理的效率和准确性，还为二手车市场引入了新的决策支持工具。通过模型的智能预测和分析，市场参与者能够更科学地评估二手车价值、预测市场走势，从而做出更明智的决策。这种技术创新为二手车市场的智能化发展奠定了坚实基础。

## 3.3赛题建模

1. 赛题背景

赛题以二手车市场为背景，要求选手预测二手汽车的交易价格，这是一个典型的回归问题。通过这道赛题来引导大家走进AI数据竞赛的世界，主要针对于于竞赛新人进行自我练习、自我提高。

1. 赛题任务和目标

比赛要求参赛选手根据给定的数据集，建立模型，二手汽车的交易价格。

赛题以预测二手车的交易价格为任务，数据集报名后可见并可下载，该数据来自某交易平台的二手车交易记录，总数据量超过40w，包含31列变量信息，其中15列为匿名变量。为了保证比赛的公平性，将会从中抽取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。

1. 数据介绍

一般而言，对于数据在比赛界面都有对应的数据概况介绍（匿名特征除外），说明列的性质特征。了解列的性质会有助于我们对于数据的理解和后续分析。 匿名特征，就是未告知数据列所属的性质的特征列。

|  |
| --- |
| train.csv   * name - 汽车编码 * regDate - 汽车注册时间 * model - 车型编码 * brand - 品牌 * bodyType - 车身类型 * fuelType - 燃油类型 * gearbox - 变速箱 * power - 汽车功率 * kilometer - 汽车行驶公里 * notRepairedDamage - 汽车有尚未修复的损坏 * regionCode - 看车地区编码 * seller - 销售方 * offerType - 报价类型 * creatDate - 广告发布时间 * price - 汽车价格 * v\_0', 'v\_1', 'v\_2', 'v\_3', 'v\_4', 'v\_5', 'v\_6', 'v\_7', 'v\_8', 'v\_9', 'v\_10', 'v\_11', 'v\_12', 'v\_13','v\_14'（根据汽车的评论、标签等大量信息得到的embedding向量）【人工构造 匿名特征】   数字全都脱敏处理，都为label encoding形式，即数字形式 |

v\_0', v\_1, v\_2, ..., v\_14 这些变量代表从汽车的评论、标签等大量信息中通过某种嵌入（embedding）技术得到的向量元素。这种嵌入向量通常用于将非数值型数据（如文本）转换为数值型数据，以便机器学习模型能够处理。这些向量中的每个元素（如v\_0', v\_1等）都捕捉了原始数据中的某些特征或信息，但具体含义是隐式的，因为它们是通过算法自动学习得到的，而不是人为直接定义的。

嵌入向量的作用 特征降维：嵌入向量可以将高维的文本数据（如单词、句子或更复杂的文本）映射到低维的连续向量空间中，从而减少计算复杂性和提高模型效率。 捕捉语义信息：在向量空间中，相似的文本（在语义上）会被映射到相近的位置，这使得模型能够理解和利用文本之间的语义关系。 支持复杂的机器学习模型：嵌入向量作为输入特征，可以支持深度学习等复杂模型，从而在汽车评论分析、情感分析、推荐系统等任务中取得更好的性能。

匿名特征 “人工构造匿名特征”可能指的是在特征工程中，通过某种方式（如聚合、转换、编码等）从原始数据中提取的、不包含或隐藏了直接识别信息的特征。然而，在您给出的嵌入向量场景中，这些向量本身就是从原始文本数据中自动学习得到的，它们已经隐式地包含了原始数据的信息，但具体的信息内容（如具体的单词或句子）是不直接可见的，因此可以说它们在一定程度上是“匿名”的。

1. 评测指标

本赛题的评价标准为**MAE(Mean Absolute Error):**

MAE（Mean Absolute Error，平均绝对误差）是一个用于衡量预测模型性能的指标，特别是在回归问题中。它计算的是预测值（）与真实值（）之间绝对误差的平均值。这里的“绝对误差”指的是预测值与真实值之差的绝对值，这样可以避免正负误差相互抵消的问题。

具体来说，MAE 的计算公式为：

MAE=

其中：

* 是第*i*个样本的真实值。
* 是第*i*个样本的预测值。
* *n* 是样本的总数。

这个公式首先计算每个样本的预测值与真实值之差的绝对值（即绝对误差），然后将所有样本的绝对误差相加，最后除以样本总数*n*，得到平均绝对误差 MAE。

**MAE 的值越小，说明预测模型的性能越好，因为它意味着预测值更接近真实值。**

1. 解题思路
2. 此题为传统的数据挖掘问题，通过数据科学以及机器学习深度学习的办法来进行建模得到结果。
3. 此题是一个典型的回归问题。
4. 主要应用xgb、lgb、catboost，以及pandas、numpy、matplotlib、seabon、sklearn、keras等等数据挖掘常用库或者框架来进行数据挖掘任务。
5. 通过EDA来挖掘数据的联系和自我熟悉数据。

## 3.4数据探索

1. 探索目标

数据探索的价值主要在于熟悉数据集，了解数据集，对数据集进行验证来确定所获得数据集可以用于接下来的机器学习或者深度学习使用。当了解了数据集之后我们下一步就是要去了解变量间的相互关系以及变量与预测值之间的存在关系。引导数据科学从业者进行数据处理以及特征工程的步骤,使数据集的结构和特征集让接下来的预测问题更加可靠。完成对于数据的探索性分析，并对于数据进行一些图表或者文字总结并打卡。

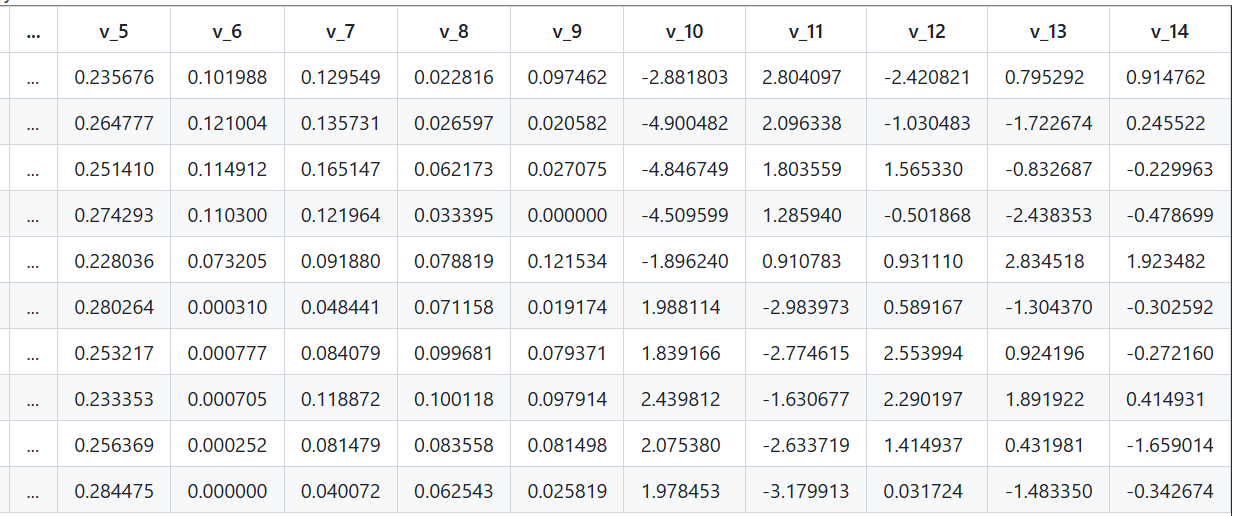
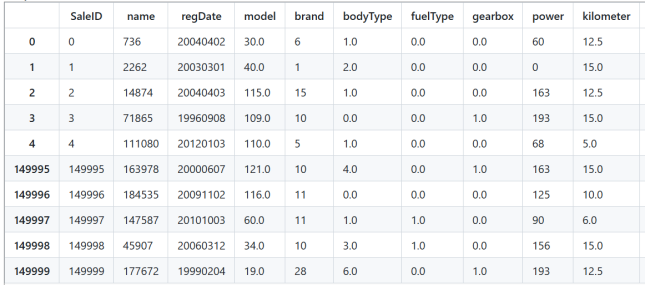
1. 内容介绍

|  |
| --- |
| 1. 载入各种数据科学以及可视化库  * 数据科学库 pandas、numpy、scipy； * 可视化库 matplotlib、seabon； * 其他；  1. 载入数据  * 载入训练集和测试集； * 简略观察数据(head()+shape)；  1. 数据总览  * 通过describe()来熟悉数据的相关统计量 * 通过info()来熟悉数据类型  1. 判断数据缺失和异常  * 查看每列的存在nan情况 * 异常值检测  1. 了解预测值的分布  * 总体缝补概况（无界约翰逊分布等） * 查看skewness and kurtosis * 查看预测值的具体频数  1. 特征分为类别特征和数字特征，并对类别特征查看unique分布 2. 数字特征分析  * 相关性分析 * 查看几个特征得 偏度和峰值 * 每个数字特征得分布可视化 * 数字特征相互之间的关系可视化 * 多变量互相回归关系可视化  1. 类型特征分析  * unique分布 * 类别特征箱形图可视化 * 类别特征的小提琴图可视化 * 类别特征的柱形图可视化类别 * 特征的每个类别频数可视化(count\_plot) |

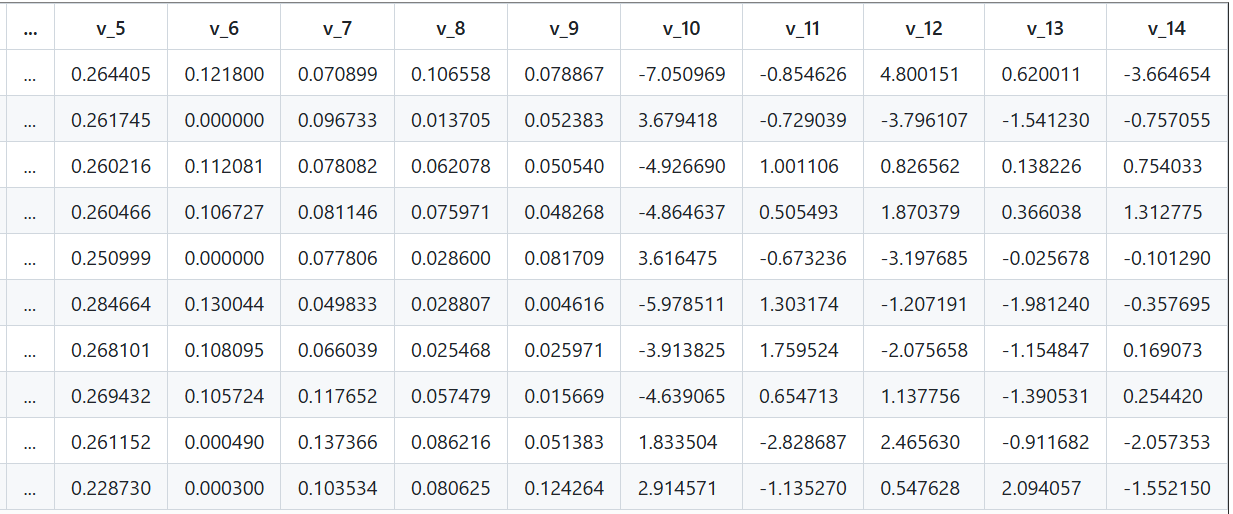
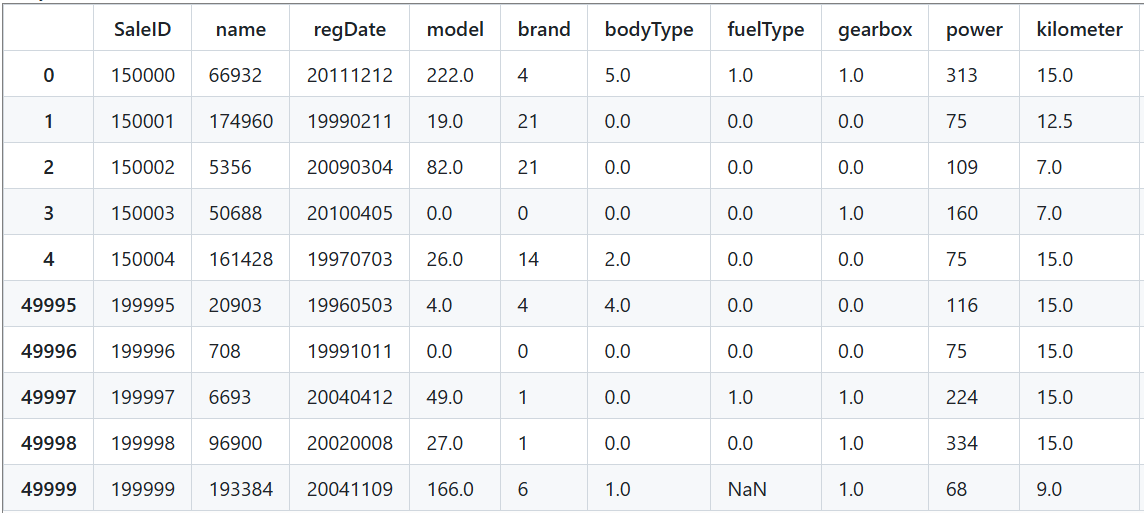
1. 代码示例
2. 载入各种数据科学及可视化库
3. 载入数据

简略观察数据

(head()+shape)Train\_data.head().append(Train\_data.tail())



Test\_data.head().append(Test\_data.tail())



1. 总览数据概况

## 1) 通过describe()来熟悉数据的相关统计量

Train\_data.describe()

Test\_data.describe()

## 2) 通过info()来熟悉数据类型

Train\_data.info()

Test\_data.info()

1. 判断数据缺失和异常

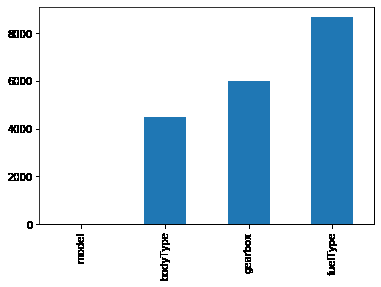
# nan可视化

missing = Train\_data.isnull().sum()

missing = missing[missing > 0]

missing.sort\_values(inplace=True)

missing.plot.bar()



通过以上两句可以很直观的了解哪些列存在 “nan”, 并可以把nan的个数打印，主要的目的在于 nan存在的个数是否真的很大，如果很小一般选择填充，如果使用lgb等树模型可以直接空缺，让树自己去优化，但如果nan存在的过多、可以考虑删掉。

# 可视化看下缺省值

msno.matrix(Train\_data.sample(250))

msno.bar(Train\_data.sample(1000))

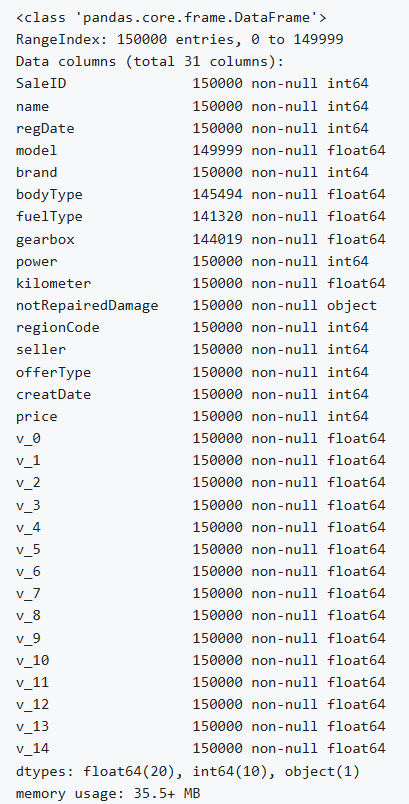
# 可视化看下缺省值

msno.matrix(Test\_data.sample(250))

msno.bar(Test\_data.sample(1000))

## 2) 查看异常值检测

Train\_data.info()



可以发现除了notRepairedDamage 为object类型其他都为数字，这里我们把他的几个不同的值都进行显示就知道了。

1. 了解预测值的分布

约翰逊SU分布（Johnson SU Distribution）： 约翰逊SU分布是一种四参数变换，用于将任意有界或无界的连续数据转换为正态分布。这种变换特别有用，因为它可以处理那些不能直接通过简单的线性变换转换为正态分布的数据。约翰逊SU分布由四个参数定义：γ（gamma）、ξ（xi）、λ（lambda）和ε（epsilon），其中γ和ξ用于位置和平移变换，λ和ε用于形状变换。尽管它很灵活，但计算可能相对复杂，且参数估计可能需要专门的软件或算法。

正态分布（Normal Distribution）： 正态分布，也称为高斯分布，是一种在自然界和社会科学中广泛存在的连续概率分布。它由一个均值（μ）和一个标准差（σ）完全定义。正态分布曲线是关于其均值对称的，且其形状由标准差决定：标准差越大，曲线越扁平；标准差越小，曲线越陡峭。正态分布因其简洁性和对许多自然现象的准确描述而备受青睐。 对数正态分布（Log-Normal Distribution）：

对数正态分布是当随机变量的对数服从正态分布时，该随机变量本身所服从的分布。换句话说，如果X是一个对数正态分布的随机变量，那么ln(X)是一个正态分布的随机变量。对数正态分布通常用于描述那些增长过程（如股票价格、人口增长等），其中增长速率本身可能具有正态分布的特性。对数正态分布的偏斜度总是正的，即其分布曲线向右偏斜，这反映了增长过程中的正偏态特性。

## 1) 总体分布概况（无界约翰逊分布等）

import scipy.stats as st

y = Train\_data['price']

plt.figure(1); plt.title('Johnson SU')

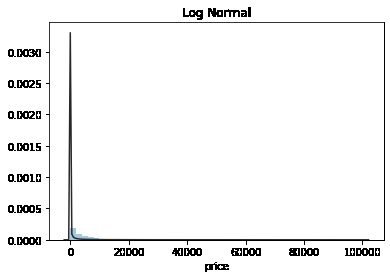
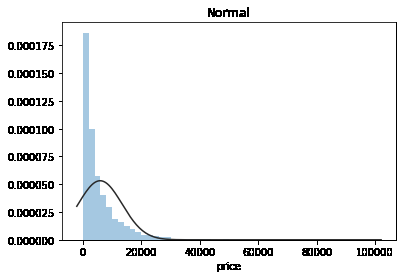
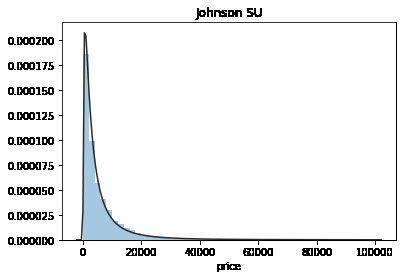
sns.distplot(y, kde=False, fit=st.johnsonsu)

plt.figure(2); plt.title('Normal')

sns.distplot(y, kde=False, fit=st.norm)

plt.figure(3); plt.title('Log Normal')

sns.distplot(y, kde=False, fit=st.lognorm)



价格不服从正态分布，所以在进行回归之前，它必须进行转换。虽然对数变换做得很好，但最佳拟合是无界约翰逊分布。

## 2) 查看skewness and kurtosis

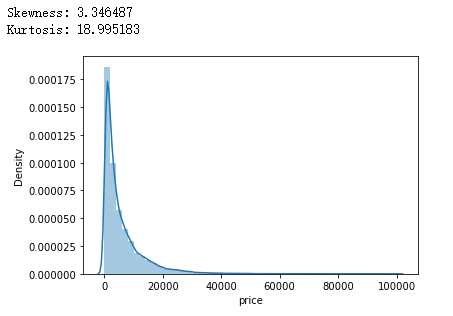
sns.distplot(Train\_data['price']);

print("Skewness: %f" % Train\_data['price'].skew())

print("Kurtosis: %f" % Train\_data['price'].kurt())

偏度（Skewness）衡量了数据分布的偏斜方向和程度。如果偏度为0，则数据分布是对称的；如果偏度大于0，则分布向右偏斜（正偏态），意味着数据的右尾更长；如果偏度小于0，则分布向左偏斜（负偏态），意味着数据的左尾更长。

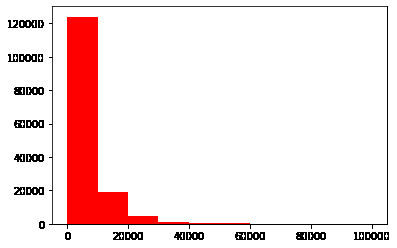
峰度（Kurtosis）衡量了数据分布的尖锐程度或峰态。正态分布的峰度为0（在某些定义中，为3减去峰度的实际值以得到与正态分布比较的峰度）。如果峰度大于0，则分布比正态分布更尖锐，有更多的极端值；如果峰度小于0，则分布比正态分布更平坦，极端值较少。



## 3) 查看预测值的具体频数

plt.hist(Train\_data['price'], orientation = 'vertical',histtype = 'bar', color ='red')

plt.show()

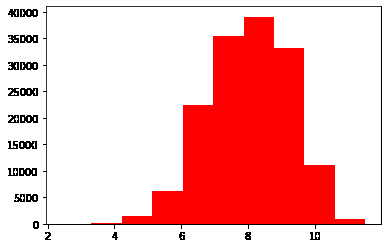


查看频数, 大于20000得值极少，其实这里也可以把这些当作特殊得值（异常值）直接用填充或者删掉，再前面进行。

# log变换 z之后的分布较均匀，可以进行log变换进行预测，这也是预测问题常用的

trickplt.hist(np.log(Train\_data['price']), orientation = 'vertical',histtype = 'bar', color ='red')

plt.show()



1. 特征分为类别特征和数字特征，并对类别特征查看unique分布。

# 特征nunique分布

for cat\_fea in categorical\_features:

print(cat\_fea + "的特征分布如下：")

print("{}特征有个{}不同的值".format(cat\_fea, Train\_data[cat\_fea].nunique()))

print(Train\_data[cat\_fea].value\_counts())

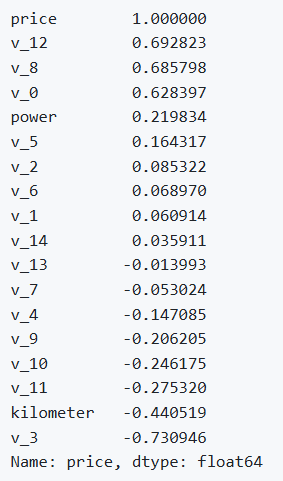
1. 数字特征分析

## 1) 相关性分析

price\_numeric = Train\_data[numeric\_features]

correlation = price\_numeric.corr(

print(correlation['price'].sort\_values(ascending = False),'\n')



## 2) 查看几个特征得 偏度和峰值

for col in numeric\_features:

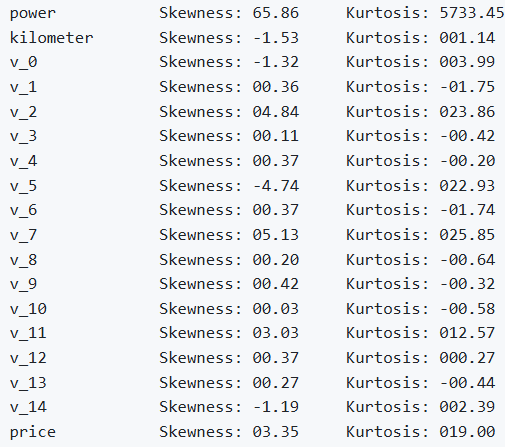
print('{:15}'.format(col),

'Skewness: {:05.2f}'.format(Train\_data[col].skew()) ,

' ' ,

'Kurtosis: {:06.2f}'.format(Train\_data[col].kurt())

)



## 3) 每个数字特征得分布可视化

f = pd.melt(Train\_data, value\_vars=numeric\_features)

g = sns.FacetGrid(f, col="variable", col\_wrap=2, sharex=False, sharey=False)

g = g.map(sns.distplot, "value")

## 4) 数字特征相互之间的关系可视化

sns.set()

columns = ['price', 'v\_12', 'v\_8' , 'v\_0', 'power', 'v\_5', 'v\_2', 'v\_6','v\_1','v\_14']

sns.pairplot(Train\_data[columns],size = 2 ,kind ='scatter',diag\_kind='kde')

plt.show()

## 5) 多变量互相回归关系可视化

fig, ((ax1, ax2), (ax3, ax4), (ax5, ax6), (ax7, ax8), (ax9, ax10)) = plt.subplots(nrows=5, ncols=2, figsize=(24, 20))# ['v\_12', 'v\_8' , 'v\_0', 'power', 'v\_5', 'v\_2', 'v\_6', 'v\_1', 'v\_14']

1. 类别特征分析

## 1) unique分布

for fea in categorical\_features:

print(Train\_data[fea].nunique())

## 2) 类别特征箱形图可视化

# 因为 name和 regionCode的类别太稀疏了，这里我们把不稀疏的几类画一下

categorical\_features = ['model',

'brand',

'bodyType',

'fuelType',

'gearbox',

'notRepairedDamage']

for c in categorical\_features:

Train\_data[c] = Train\_data[c].astype('category')

if Train\_data[c].isnull().any():

Train\_data[c] = Train\_data[c].cat.add\_categories(['MISSING'])

Train\_data[c] = Train\_data[c].fillna('MISSING')

def boxplot(x, y, \*\*kwargs):

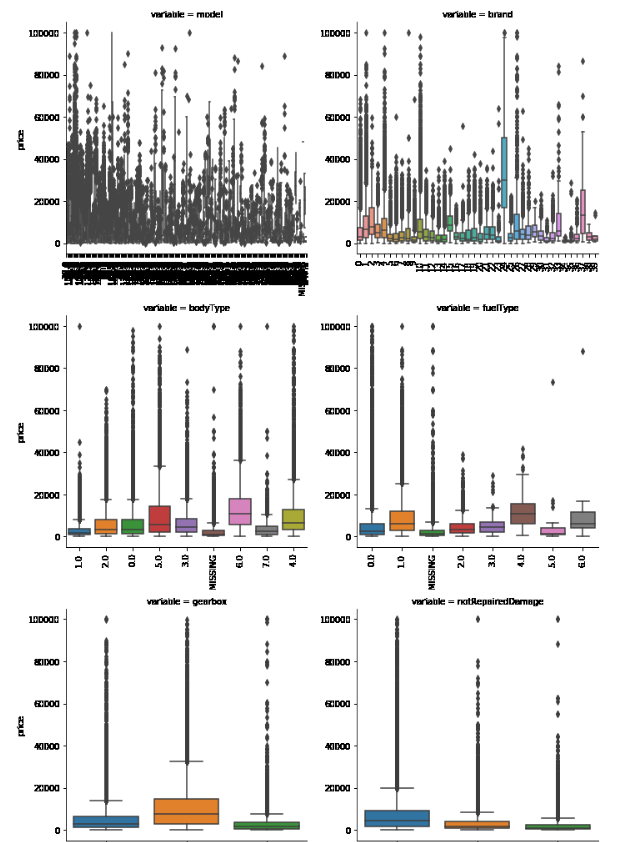
sns.boxplot(x=x, y=y)

x=plt.xticks(rotation=90)

f = pd.melt(Train\_data, id\_vars=['price'], value\_vars=categorical\_features)

g = sns.FacetGrid(f, col="variable", col\_wrap=2, sharex=False, sharey=False, size=5)

g = g.map(boxplot, "value", "price")



# 特征工程

## 4.1 特征工程目标

1. 对于特征进行进一步分析，并对于数据进行处理。
2. 完成对于特征工程的分析，并对于数据进行一些图表或者文字总结并打卡。

4.2 内容介绍

常见的特征工程包括：

|  |
| --- |
| 1. 异常处理：  * 通过箱线图（或 3-Sigma）分析删除异常值；  1. 特征归一化/标准化：  * 归一化（抓换到 [0,1] 区间）；  1. 特征筛选  * 过滤式（filter）：先对数据进行特征选择，然后在训练学习器，常见的方法有 Relief/ 方差选择发/相关系数法/卡方检验法/互信息法； |

## 4.3 代码示例

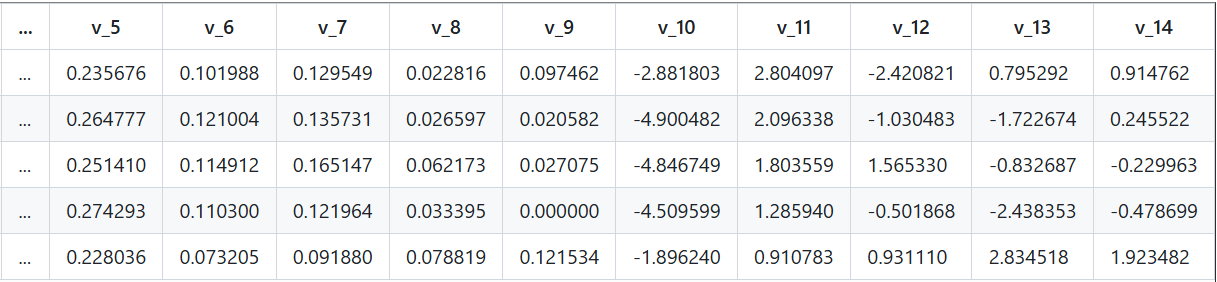
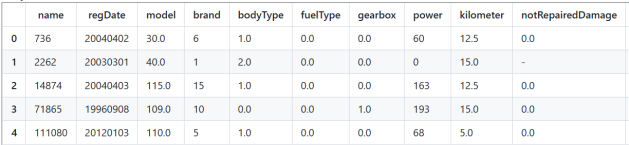
1. 导入数据

train = pd.read\_csv('train.csv', sep=' ')

test = pd.read\_csv('testA.csv', sep=' ')

print(train.shape)

print(test.shape)



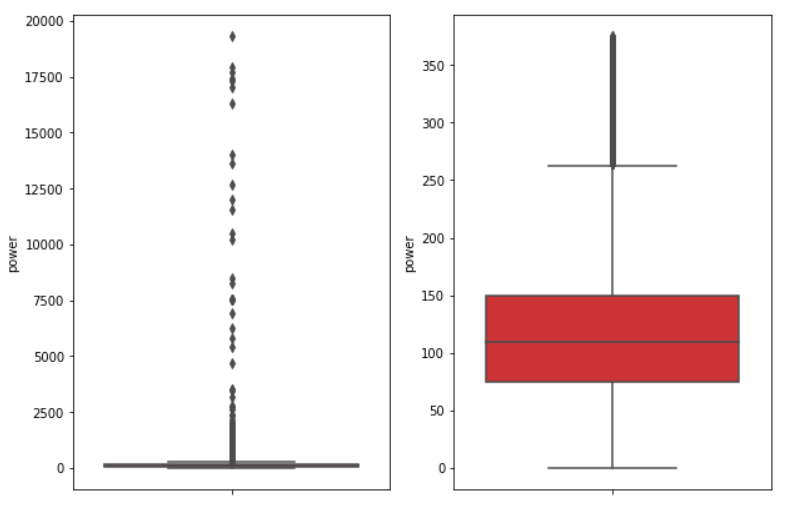
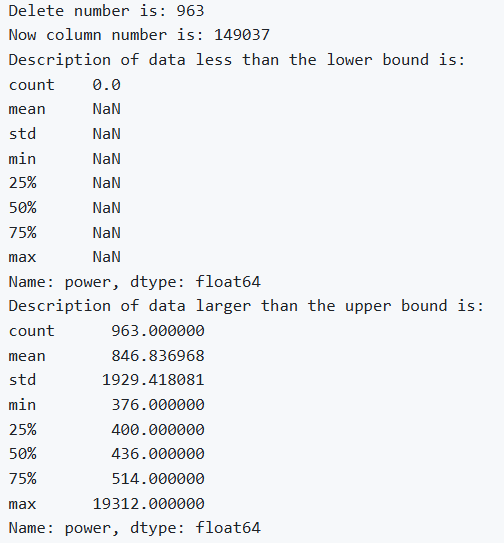
1. 删除异常值

# 我们可以删掉一些异常数据，以 power 为例。

# 这里删不删同学可以自行判断

# 但是要注意 test 的数据不能删

train = outliers\_proc(train, 'power', scale=3)



count: 963.000000 表示有963个数据点高于设定的上界。

mean: 846.836968 表示这些高于上界的数据点的平均值为846.84。

std: 1929.418081 表示这些数据的标准差很大，说明数据点之间的离散程度很高。

min: 376.000000 表示高于上界的数据点中的最小值是376。

25%, 50%, 75% 分别表示第一四分位数（400）、中位数（436）和第三四分位数（514），这些数据提供了数据分布的更多细节。

max: 19312.000000 表示高于上界的数据点中的最大值是19312，这是一个非常大的值，可能对整体数据的均值和标准差产生显著影响。

1. 特征构造

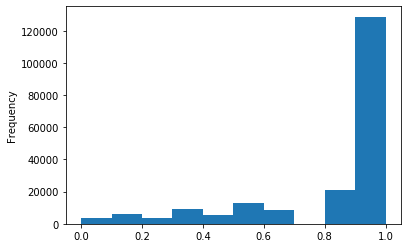
归一化是一种将数据的数值范围调整到特定区间（通常是0到1之间）的过程。这个过程通过重新缩放特征值来实现，使得所有特征值都落在相同的数值范围内。归一化对于许多基于距离的算法（如K-近邻算法、K-均值聚类）和需要计算权重的算法（如神经网络）特别重要，因为它可以帮助避免某些特征对结果产生过大的影响。

# 直接做归一化

data['kilometer'] = ((data['kilometer'] - np.min(data['kilometer'])) /

(np.max(data['kilometer']) - np.min(data['kilometer'])))

data['kilometer'].plot.hist()



1. 特征筛选

# 直接看图

data\_numeric = data[['power', 'kilometer', 'brand\_amount', 'brand\_price\_average',

'brand\_price\_max', 'brand\_price\_median']]

correlation = data\_numeric.corr()

f , ax = plt.subplots(figsize = (7, 7))

plt.title('Correlation of Numeric Features with Price',y=1,size=16)

sns.heatmap(correlation,square = True, vmax=0.8)

1. 解读特征之间的相关性：

·正相关：如果两个特征之间的颜色偏向红色，并且颜色较深，那么这表示这两个特征之间存在正相关关系。即，当一个特征的值增加时，另一个特征的值也倾向于增加。

·负相关：如果两个特征之间的颜色偏向蓝色（或绿色，取决于颜色方案），并且颜色较深，那么这表示这两个特征之间存在负相关关系。即，当一个特征的值增加时，另一个特征的值倾向于减少。

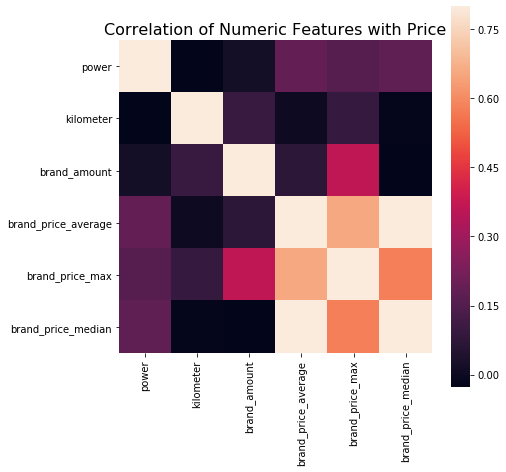
·无相关：如果两个特征之间的颜色接近热图的背景色（通常是白色或浅灰色），或者颜色非常浅，那么这表示这两个特征之间没有明显的相关性。

1. 注意相关性系数的范围：

在你的代码中，vmax=0.8 表示热图的颜色渐变将只反映相关性系数的绝对值在0到0.8之间的变化。这意味着即使有些特征之间的相关性系数超过0.8，它们在热图上的颜色深浅也会受到限制，以避免过于强烈的颜色对比。

1. 分析特征与价格的关系：

由于你的热图是基于与价格（price）的相关性生成的，因此你可以特别关注每一行或列中与price的关系。这些将直接告诉你哪些特征对价格有显著的正向或负向影响。



# 模型选择和训练

## 学习目标

了解常用的机器学习模型，并掌握机器学习模型的建模与调参流程

## 内容介绍

|  |
| --- |
| 1. 读取数据 2. 线性回归&五折交叉验证&模拟真实业务情况  * 简单建模； * 交叉验证方法； * 模拟真实业务情况； * 绘制验证曲线；  1. 模型对比  * 常用线性模型； * 常用非线性模型；  1. 模型调参  * 贪心调参方法；  1. 小结 |

## 5.3 代码示例

1. 读取数据
2. 线性回归&五折交叉验证&模拟真实业务情况

sample\_feature = sample\_feature.dropna().replace('-', 0).reset\_index(drop=True)

sample\_feature['notRepairedDamage'] = sample\_feature['notRepairedDamage'].astype(np.float32)

train = sample\_feature[continuous\_feature\_names + ['price']]

train\_X = train[continuous\_feature\_names]

train\_y = train['price']

1. 简单建模

import seaborn as sns

print('It is clear to see the price shows a typical exponential distribution')

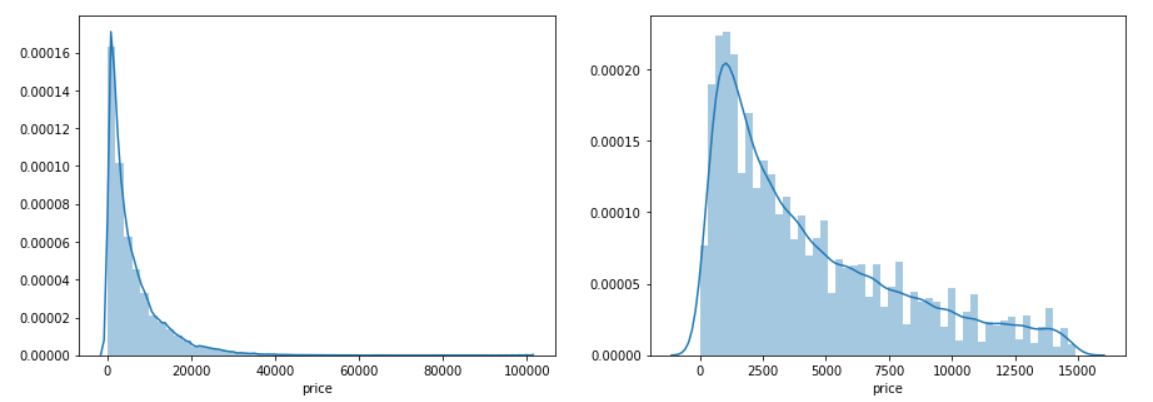
plt.figure(figsize=(15,5))

plt.subplot(1,2,1)

sns.distplot(train\_y)

plt.subplot(1,2,2)

sns.distplot(train\_y[train\_y < np.quantile(train\_y, 0.9)])



在这里我们对标签进行了 变换，使标签贴近于正态分布。

再次进行可视化，发现预测结果与真实值较为接近，且未出现异常状况。

plt.scatter(train\_X['v\_9'][subsample\_index], train\_y[subsample\_index], color='black')

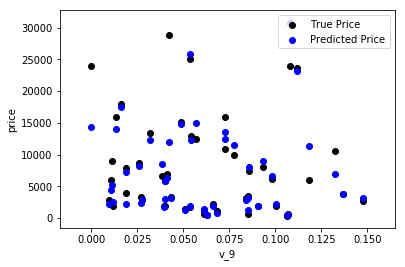
plt.scatter(train\_X['v\_9'][subsample\_index], np.exp(model.predict(train\_X.loc[subsample\_index])), color='blue')

plt.xlabel('v\_9')

plt.ylabel('price')plt.legend(['True Price','Predicted Price'],loc='upper right')

print('The predicted price seems normal after np.log transforming')

plt.show()



1. 五折交叉验证

在使用训练集对参数进行训练的时候，经常会发现人们通常会将一整个训练集分为三个部分（比如mnist手写训练集）。一般分为：训练集（train\_set），评估集（valid\_set），测试集（test\_set）这三个部分。这其实是为了保证训练效果而特意设置的。其中测试集很好理解，其实就是完全不参与训练的数据，仅仅用来观测测试效果的数据。而训练集和评估集则牵涉到下面的知识了。

因为在实际的训练中，训练的结果对于训练集的拟合程度通常还是挺好的（初始条件敏感），但是对于训练集之外的数据的拟合程度通常就不那么令人满意了。因此我们通常并不会把所有的数据集都拿来训练，而是分出一部分来（这一部分不参加训练）对训练集生成的参数进行测试，相对客观的判断这些参数对训练集之外的数据的符合程度。这种思想就称为交叉验证（Cross Validation）。

五折交叉验证（5-fold cross-validation）是一种评估机器学习模型性能的方法，它将数据集分成五个大小相等的子集（如果数据集大小不能被5整除，则可能需要稍微调整某些子集的大小以确保所有子集尽可能接近）。在五折交叉验证中，模型会被训练五次，每次都会使用四个子集作为训练集，剩下的一个子集作为验证集来评估模型的性能。这个过程会重复五次，每次使用不同的子集作为验证集，最终会得到五个性能评估结果。

scores = pd.DataFrame(scores.reshape(1,-1))

scores.columns = ['cv' + str(x) for x in range(1, 6)]

scores.index = ['MAE']

scores

|  | **Cv1** | **Cv2** | **Cv3** | **Cv4** | **Cv5** |
| --- | --- | --- | --- | --- | --- |
| **MAE** | 0.191642 | 0.194986 | 0.192737 | 0.195329 | 0.19445 |

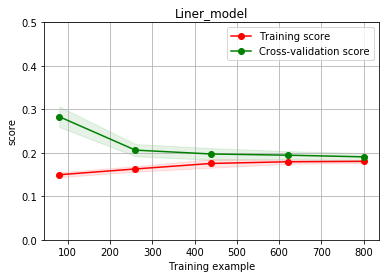
1. 模拟真实业务情况

但在事实上，由于我们并不具有预知未来的能力，五折交叉验证在某些与时间相关的数据集上反而反映了不真实的情况。通过2018年的二手车价格预测2017年的二手车价格，这显然是不合理的，因此我们还可以采用时间顺序对数据集进行分隔。在本例中，我们选用靠前时间的4/5样本当作训练集，靠后时间的1/5当作验证集，最终结果与五折交叉验证差距不大。

mean\_absolute\_error(val\_y\_ln, model.predict(val\_X))=0.19443858353490887

1. 绘制学习率曲线与验证曲线

plot\_learning\_curve(LinearRegression(), 'Liner\_model', train\_X[:1000], train\_y\_ln[:1000], ylim=(0.0, 0.5), cv=5, n\_jobs=1)



1. 多种模型对比

train = sample\_feature[continuous\_feature\_names + ['price']].dropna()

train\_X = train[continuous\_feature\_names]

train\_y = train['price']

train\_y\_ln = np.log(train\_y + 1)

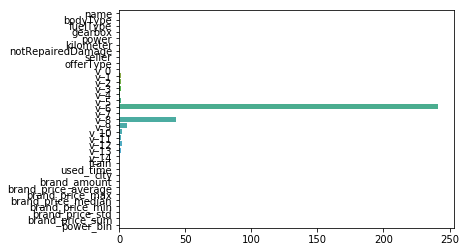
1. 线性模型&嵌入式特征选择

在过滤式和包裹式特征选择方法中，特征选择过程与学习器训练过程有明显的分别。而嵌入式特征选择在学习器训练过程中自动地进行特征选择。嵌入式选择最常用的是L1正则化与L2正则化。在对线性回归模型加入两种正则化方法后，他们分别变成了岭回归与Lasso回归。

model = LinearRegression().fit(train\_X, train\_y\_ln)

print('intercept:'+ str(model.intercept\_))

sns.barplot(abs(model.coef\_), continuous\_feature\_names)



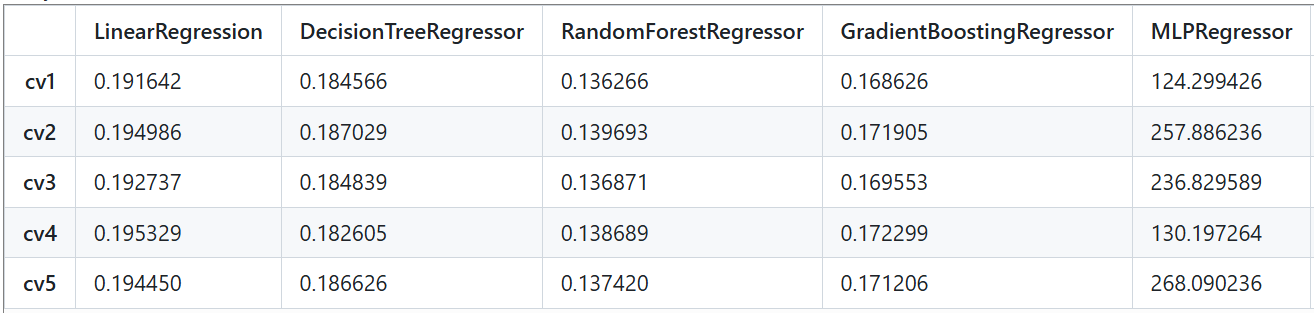
L2正则化在拟合过程中通常都倾向于让权值尽可能小，最后构造一个所有参数都比较小的模型。因为一般认为参数值小的模型比较简单，能适应不同的数据集，也在一定程度上避免了过拟合现象。可以设想一下对于一个线性回归方程，若参数很大，那么只要数据偏移一点点，就会对结果造成很大的影响；但如果参数足够小，数据偏移得多一点也不会对结果造成什么影响，专业一点的说法是『抗扰动能力强』。

1. 非线性模型

result = pd.DataFrame(result)

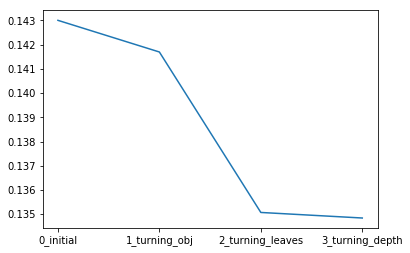
result.index = ['cv' + str(x) for x in range(1, 6)]

result



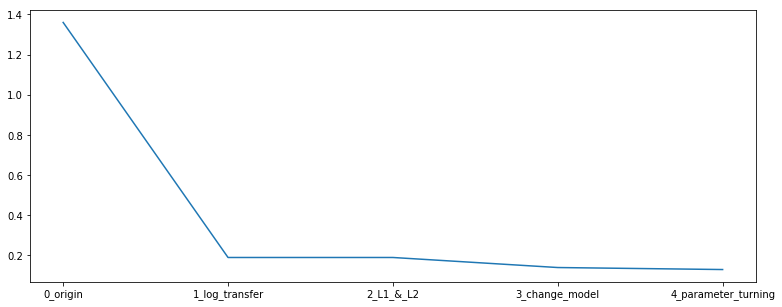
1. 贪心调参

sns.lineplot(x=['0\_initial','1\_turning\_obj','2\_turning\_leaves','3\_turning\_depth'], y=[0.143 ,min(best\_obj.values()), min(best\_leaves.values()), min(best\_depth.values())])



1. 小结

在本章中，我们完成了建模与调参的工作，并对我们的模型进行了验证。此外，我们还采用了一些基本方法来提高预测的精度，提升如下图所示。



# 模型融合和优化

## 6.1 模型融合目标

对于多种调参完成的模型进行模型融合。

## 6.2 内容介绍

|  |
| --- |
| 1. 简单加权融合  * 回归（分类概率）：算术平均融合（Arithmetic mean），几何平均融合（Geometric mean）； * 分类：投票（Voting) * 综合：排序融合(Rank averaging)，log融合  1. stacking/blending  * 构建多层模型，并利用预测结果再拟合预测。  1. boosting/bagging（在xgboost，Adaboost,GBDT中已经用到）  * 多树的提升方法 |

## 6.3 Stacking相关理论介绍

一、基本概念

定义：Stacking是指训练一个模型用于组合其他各个模型的预测结果，即首先训练多个不同的基学习器（base learners），然后将这些基学习器的输出作为输入来训练一个元学习器（meta-learner），以得到一个最终的输出。

结构：Stacking模型通常由两层组成，第一层是多个基学习器，第二层是用于组合基学习器输出的元学习器。

二、工作流程

数据划分：将原始数据集划分为训练集和测试集（或验证集），通常使用交叉验证等方法来更充分地利用训练数据。

训练基学习器：在训练集上独立训练多个不同的基学习器，这些基学习器可以是不同类型的模型，如决策树、神经网络、支持向量机等。

生成一级预测结果：使用训练好的基学习器对训练集（或另一个独立的数据集）进行预测，得到每个基学习器的预测结果。

构造次级学习器：将基学习器的预测结果作为新的特征输入到次级学习器中，次级学习器的目标是结合这些预测结果来生成最终的预测。

训练次级学习器：使用基学习器的预测结果和原始数据的标签来训练次级学习器。

最终预测：使用训练好的次级学习器对测试集进行预测，得到最终的预测结果。

## 6.4 代码示例

1. 回归、分类概率-融合
2. 简单加权平均，结果直接融合

## 根据加权计算

MAEw = [0.3,0.4,0.3] # 定义比重权值

Weighted\_pre = Weighted\_method(test\_pre1,test\_pre2,test\_pre3,w)

print('Weighted\_pre MAE:',metrics.mean\_absolute\_error(y\_test\_true, Weighted\_pre))

Weighted\_pre MAE: 0.0575

可以发现加权结果相对于之前的结果是有提升的，这种我们称其为简单的加权平均。

1. Stacking融合（回归）

model\_L2= linear\_model.LinearRegression()

Stacking\_pre = Stacking\_method(train\_reg1,train\_reg2,train\_reg3,y\_train\_true,

test\_pre1,test\_pre2,test\_pre3,model\_L2)

print('Stacking\_pre MAE:',metrics.mean\_absolute\_error(y\_test\_true, Stacking\_pre))

Stacking\_pre MAE: 0.0421348314607

可以发现模型结果相对于之前有进一步的提升，这是我们需要注意的一点是，对于第二层Stacking的模型不宜选取的过于复杂，这样会导致模型在训练集上过拟合，从而使得在测试集上并不能达到很好的效果。

1. 分类模型融合
2. voting投票机制

Voting即投票机制，分为软投票和硬投票两种，其原理采用少数服从多数的思想。

硬投票（Hard Voting） 定义： 硬投票是一种基于多数表决的投票策略。在硬投票中，每个基本学习器（或称为基本模型）都对样本进行预测，并投票选择出现次数最多的类别作为最终的预测结果。如果存在平局，则通常选择其中一个类别作为结果（具体选择哪个类别可能取决于实现方式或额外规则）。

软投票（Soft Voting） 定义： 软投票是一种基于概率的投票策略。在软投票中，每个基本学习器都会给出每个类别的概率或置信度估计，然后对这些概率进行平均或加权平均，以产生最终的预测结果。最终的类别是概率平均值最高的那个类别。

'''硬投票：对多个模型直接进行投票，不区分模型结果的相对重要度，最终投票数最多的类为最终被预测的类。'''

Accuracy: 0.97 (+/- 0.02) [XGBBoosting]

Accuracy: 0.33 (+/- 0.00) [Random Forest]

Accuracy: 0.95 (+/- 0.03) [SVM]

Accuracy: 0.94 (+/- 0.04) [Ensemble]

'''软投票：和硬投票原理相同，增加了设置权重的功能，可以为不同模型设置不同权重，进而区别模型不同的重要度。'''

Accuracy: 0.96 (+/- 0.02) [XGBBoosting]

Accuracy: 0.33 (+/- 0.00) [Random Forest]

Accuracy: 0.95 (+/- 0.03) [SVM]

Accuracy: 0.96 (+/- 0.02) [Ensemble]

1. 分类的Stacking\Blending融合

stacking是一种分层模型集成框架。

以两层为例，第一层由多个基学习器组成，其输入为原始训练集，第二层的模型则是以第一层基学习器的输出作为训练集进行再训练，从而得到完整的stacking模型, stacking两层模型都使用了全部的训练数据。

val auc Score: 1.000000

val auc Score: 0.500000

val auc Score: 0.500000

val auc Score: 0.500000

val auc Score: 0.500000

Val auc Score of Stacking: 1.000000

Blending，其实和Stacking是一种类似的多层模型融合的形式

其主要思路是把原始的训练集先分成两部分，比如70%的数据作为新的训练集，剩下30%的数据作为测试集。

在第一层，我们在这70%的数据上训练多个模型，然后去预测那30%数据的label，同时也预测test集的label。

在第二层，我们就直接用这30%数据在第一层预测的结果做为新特征继续训练，然后用test集第一层预测的label做特征，用第二层训练的模型做进一步预测

其优点在于：

* 比stacking简单（因为不用进行k次的交叉验证来获得stacker feature）
* 避开了一个信息泄露问题：generlizers和stacker使用了不一样的数据集

缺点在于：

* 使用了很少的数据（第二阶段的blender只使用training set10%的量）
* blender可能会过拟合
* stacking使用多次的交叉验证会比较稳健

1. 分类的Stacking融合(利用mlxtend)：

可以发现 基模型 用 'KNN', 'Random Forest', 'Naive Bayes' 然后再这基础上 次级模型加一个 'LogisticRegression'，模型测试效果有着很好的提升。

1. 一些其它方法

将特征放进模型中预测，并将预测结果变换并作为新的特征加入原有特征中再经过模型预测结果 （Stacking变化）。

# 总结

目标：构建二手车交易价格预测模型，分析品牌、型号、年份、里程数等因素对价格的影响。

过程：

1. 数据预处理：填充缺失值，处理异常值，标准化数据。
2. 数据分析：可视化各因素与价格关系，计算相关系数。
3. 特征工程：筛选组合特征，如车辆年龄、品牌型号分类编码、地域区域划分。
4. 建模调参：尝试多种算法，优化参数，梯度提升树表现最佳。
5. 模型融合：采用加权平均融合多模型预测结果，提升预测准确度。

总结：

本项目通过数据分析、处理异常值、标准化数据、特征值分析筛选等处理后，通过建模融合，得出预测结果。可以说就是对数据的总结归纳，然后训练模型得出与原先近似的值。刚开始从特征筛选这一项目了解到价格与各个特征有密不可分的关系，一一对应预反而繁琐了预测的过程；再从建模时查看特征值的权重发现匿名特征比一般特征的权重占比更大，而且在用匿名特征来预测价格后通过五折交叉验证得出的MAE值体现这些数据参与预测得出的结果接近真实值。

但在事实上，单单通过数据并不具有预知未来的能力，五折交叉验证在某些与时间相关的数据集上反而反映了不真实的情况。我们只能通过预测的结果推断出未来的价格走向。从简单建模这一项目可以看出低价格的二手车预测值与实际值来去不大，而高价格有着很大的出入，说明高端市场的行情是飘忽不定的，随着时间的流失，品牌影响力可能因科技发展、全球经济形势、政策环境、文化潮流变得不再是消费者眼中的关注点；低端市场因为有着庞大的消费群体，反而是一种优势，短时间内大众不会因某些影响猛然提升自己的消费水平。

展望：

通过本次实验，我们成功地构建了一个用于预测二手车交易价格的机器学习模型。该模型能够准确地估计二手车的交易价格，并为市场参与者提供有价值的参考。在实验过程中，我们深入理解了影响二手车交易价格的关键因素，并通过数据分析、特征工程、建模调参和模型融合等技术手段，不断优化模型的性能。 展望未来，我们可以进一步探索更多的影响因素和特征工程方法，以提高模型的预测能力。此外，我们还可以尝试将深度学习技术应用于二手车交易价格预测领域，以挖掘更多的潜在信息和模式。同时，我们也将关注二手车市场的动态变化，及时调整模型以适应市场的变化需求。

# 参考文献

[1] 灵魂拷问:你看过Xgboost原文吗？[EB/OL].[2020-04-09].

<https://blog.csdn.net/hzbooks/article/details/108216176>

[2] 回归分析的五个基本假设 [EB/OL].[2017-07-25].

<https://blog.csdn.net/Noob_daniel/article/details/76087829>

[3] 【机器学习】集成学习之Stacking（堆叠泛化）[EB/OL].[2024-03-31].

<https://blog.csdn.net/ueke1/article/details/137190677>

# 附录：见源程序电子代码

<task1赛题理解.ipynb>

<task2数据分析.ipynb>

<task3特征工程.ipynb>

<task4模型调参.ipynb>

<task5模型融合.ipynb>