|  |
| --- |
| **机器学习** |
| **Mercari Price Suggestion Challenge** |
| 主 研 人：岳天昕  参 研 人： |
| 审 核 人：  方 向：工业算法案例研究  版 本 号：A |
|  |

**版本更新**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **更新描述** | **作者** |
| 2018/10/31 | A | 初稿 |  |

**选题表格**

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 竞赛名 | 竞赛背景描述（50字以内） | 类型（分类/回归） |
| 2018/10/31 | Mercari Price Suggestion Challenge | 产品的细节不同可能意味着定价的巨大差异，Mercari购物公司希望通过用户输入的产品类别、品牌和商品条件等详细信息，构建一种自动推荐产品价格的算法。 | 回归 |

目录

[1. 背景描述 4](#_Toc25575_WPSOffice_Level1)

[1.1 竞赛赛题描述 4](#_Toc5940_WPSOffice_Level2)

[1.2 评估指标描述 5](#_Toc12357_WPSOffice_Level2)

[2. 数据来源及描述性统计分析 5](#_Toc5940_WPSOffice_Level1)

[2.1 大赛数据来源 5](#_Toc22683_WPSOffice_Level2)

[2.2 数据的描述性统计 5](#_Toc16174_WPSOffice_Level2)

[2.2.1 数据基本情况描述： 5](#_Toc5940_WPSOffice_Level3)

[2.2.2 数据字段介绍： 5](#_Toc12357_WPSOffice_Level3)

[2.2.3 数据描述性统计 6](#_Toc22683_WPSOffice_Level3)

[3. 优秀算法思路 6](#_Toc12357_WPSOffice_Level1)

[3.1 方案一 6](#_Toc32293_WPSOffice_Level2)

[3.1.1 方案一数据预处理及特征工程部分方案 6](#_Toc16174_WPSOffice_Level3)

[3.1.2 方案一模型设计、建立部分方案 6](#_Toc32293_WPSOffice_Level3)

[3.1.3 方案一结果、排名等 6](#_Toc29922_WPSOffice_Level3)

[3.1.4 方案一算法流程图 6](#_Toc4787_WPSOffice_Level3)

[3.2 方案二 7](#_Toc29922_WPSOffice_Level2)

[… 8](#_Toc4787_WPSOffice_Level2)

[4. 算法比较 8](#_Toc22683_WPSOffice_Level1)

[表4-1 算法比较 8](#_Toc7253_WPSOffice_Level2)

[5. 总结与展望 8](#_Toc16174_WPSOffice_Level1)

[5.1 总结 8](#_Toc23198_WPSOffice_Level2)

[5.2 建模思路 8](#_Toc16619_WPSOffice_Level2)

1. 背景描述

有时候我们很难知道一些东西到底值多少钱，一些小细节的不同可能意味着定价的巨大差异。 例如，其中一件毛衣售价335美元，另一件售价9.99美元。



考虑到各类产品在线上销售的规模巨大，产品定价就变得更加困难。比如服装具有强烈的季节性定价趋势，受品牌影响也很大，而电子产品的价格则根据产品规格而波动。  
  
Mercari是日本最大的社会性购物应用程序（类似于中国闲鱼），他们想向卖家提供定价建议，但这很难，因为他们的卖家可以在Mercari的市场上放置任何东西。

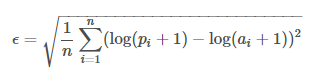
**1.1 竞赛赛题描述**

在此次竞赛中，Mercari购物公司希望通过用户输入的产品类别、品牌和商品条件等详细信息，构建一种自动推荐产品价格的算法，为卖家提供商品的最佳定价建议。

**1.2 评估指标描述**

竞赛中评估模型优劣的指标。

选手提交结果与实际产品销售价格进行对比，以均方根对数误差（RMSLE）为评价指标，结果越小越好。均方根对数误差计算公式如下：



其中：

ε是RMSLE值（得分  
n是数据集中的观察总数  
pi是你对价格的预测  
ai是i的实际销售价格  
log（x）是x的自然对数

1. 数据来源及描述性统计分析

**2.1 大赛数据来源**

Mercari公司提供用户及产品的数据信息。

这里是数据的超链接：<mercari-price-suggestion-challenge>

**2.2 数据的描述性统计**

**2.2.1 数据基本情况描述：**

本比赛使用的数据由train.tsv和test.tsv两个数据文件组成，数据文件中的内容由tab符分隔开来。

**2.2.2 数据字段介绍：**：

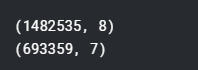
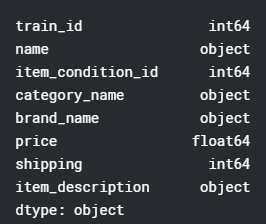
**表2-1 train.tsv数据表字段介绍**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 变量含义 | 变量类型 | 缺失率 |
| **train\_id** | **训练集条目编号** | **离散** | 10% |
| **teat\_id** | **测试集条目编号** | **离散** | 20% |
| **item\_condition\_id** | **卖家提供的商品的状态** | **离散** | 0% |
| **category\_name** | **商品类别** | 离散 | 0% |
| **brand\_name** | **品牌名称** |  | 0% |
| **price** | **商品售价（预测目标）** | 连续 | 0% |
| **shipping** | **运费** | 离散 | 0% |
| **item\_description** | **对商品的文字描述** |  | 0% |

**表2-2 test.tsv数据表字段介绍**

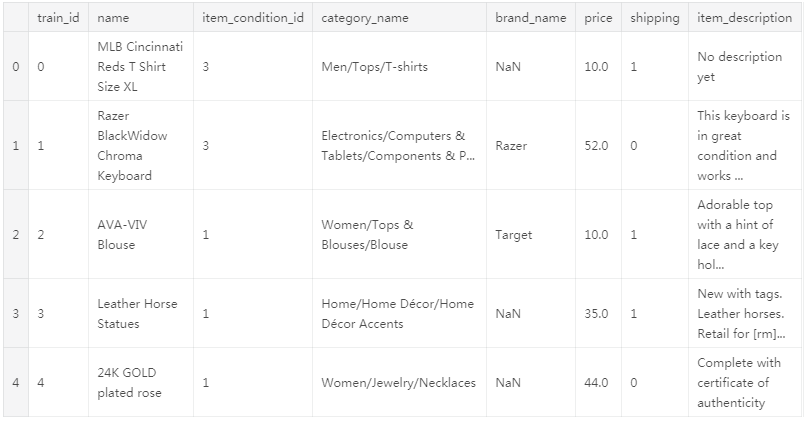
|  |  |  |  |
| --- | --- | --- | --- |
|  | 变量含义 | 变量类型 | 缺失率 |
| **train\_id** | **训练集条目编号** | **离散** | 10% |
| **teat\_id** | **测试集条目编号** | **离散** | 20% |
| **item\_condition\_id** | **卖家提供的商品的状态** | **离散** | 0% |
| **category\_name** | **商品类别** | 离散 | 0% |
| **brand\_name** | **品牌名称** |  | 0% |
| **shipping** | **运费** | 离散 | 0% |
| **item\_description** | **对商品的文字描述** |  | 0% |

**2.2.3 数据描述性统计**

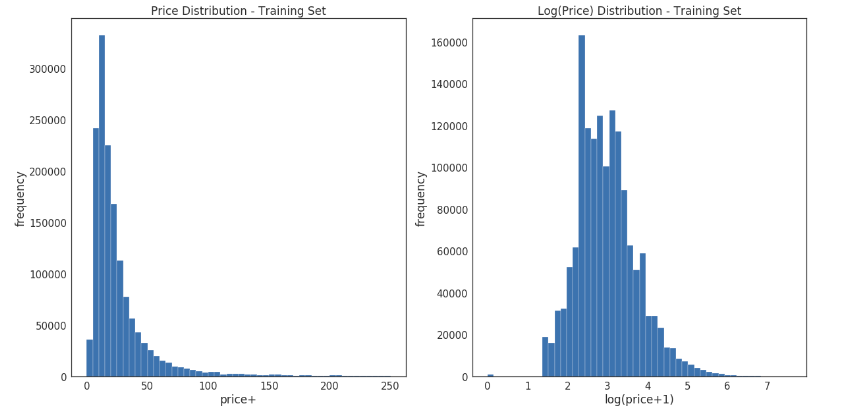
 

数据集和测试集大小 数据类型

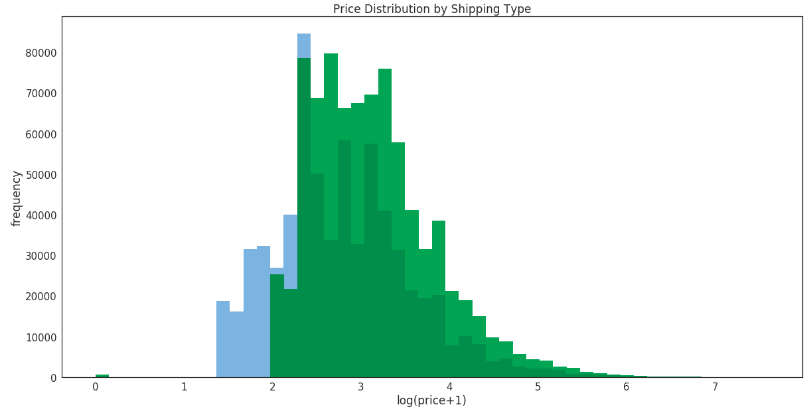
数据预览：



目标值（Price）分布情况：



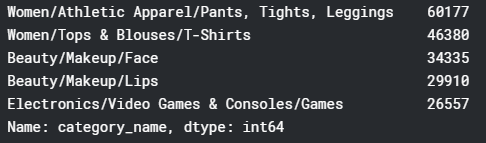
运费与目标值的关系图：数据中卖家承担运费则运费值为1，其余为0



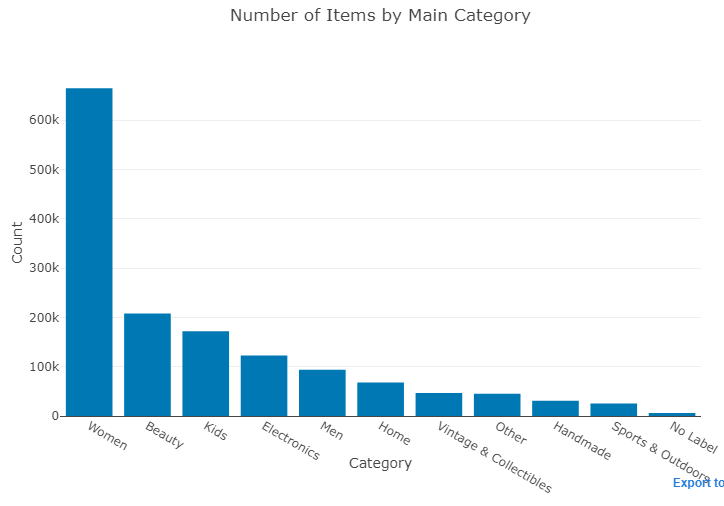
商品类别：

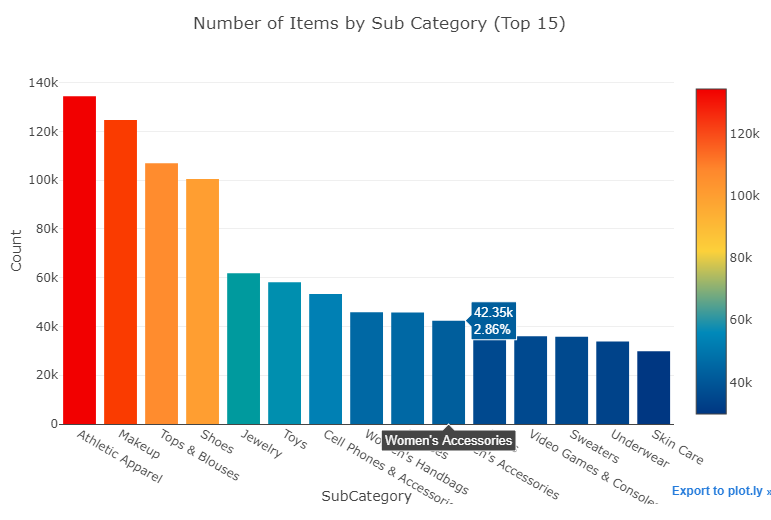
数据中有大约1287种商品类别，我们先看大概的分类，再根据大类分出子类。除此以外，数据中约有6327个商品没有分类。我们将类别拆分成三个不同的列。如何处理品牌名称列中的缺失信息对模型的预测有较重要的影响。

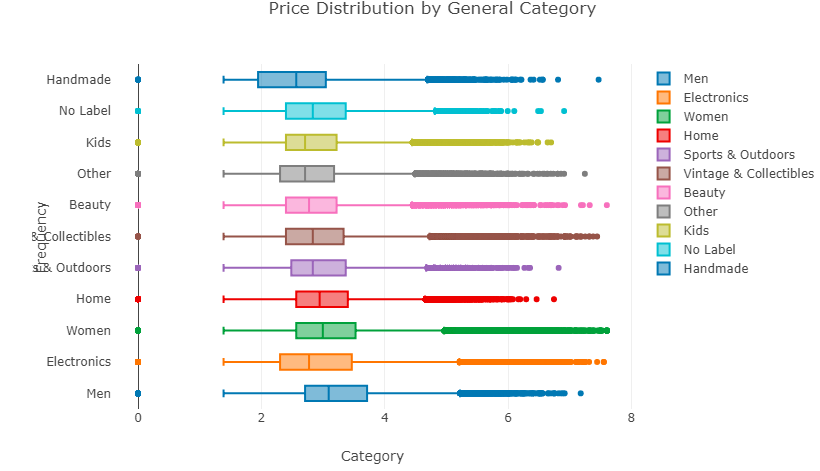
前五大类商品类别预览：



总体而言，我们有7个主要类别（第一个子类别有114种，第二个子类别有871个中）：女性服饰和美妆是两个最常见的类别（超过50％），其次是儿童用品和电子产品。





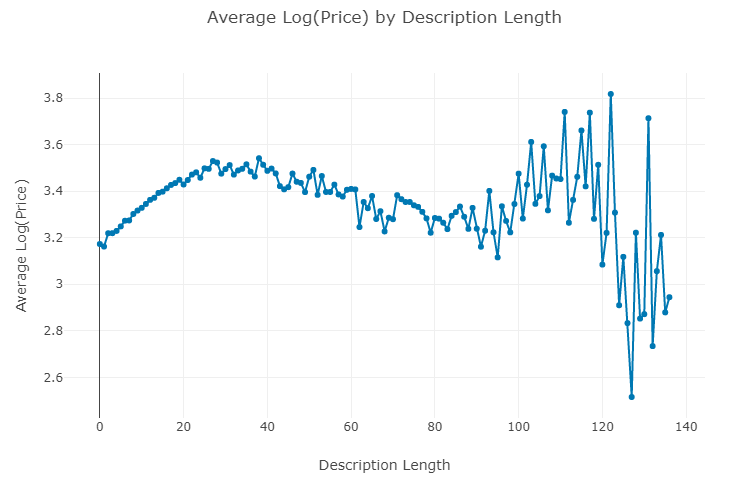


品牌名称：

训练集中共有4809种不同的品牌名称

商品描述：

解析这个特定项目将更具挑战性，因为它是非结构化数据。这是否意味着更详细和冗长的描述会导致更高的出价？我们将删除所有标点符号，删除一些英语词（如“a”，“the”等）和任何其他长度小于3的词



如果我们按类别查看最常见的单词，我们可以发现，“大小尺寸”，“免费”和“运费”是卖家非常常用的，可能是为了吸引顾客，这与我们之前发现价格和运费这两个变量之间的相关性很小是有矛盾的。“品牌名称”也扮演着非常重要的角色 - 它是这四个类别中最常见的用词之一。

其中一个特征“商品描述”的文字处理：

Pre-precessing:tokenization

大多数情况下，自然语言处理的第一步是“标记化”你的文档，其主要目的是规范化我们的文本。三个基本阶段通常包括：

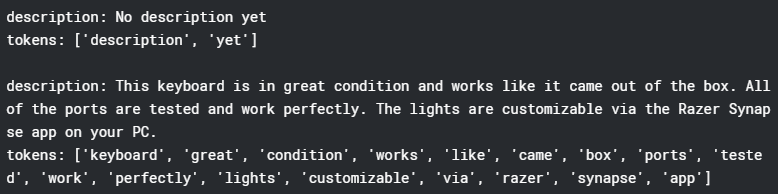
将描述分解为句子，然后将句子分解为tokens

删除标点符号并停用单词

小写所有tokens

(在此题中，我们也只考虑长度大于等于3个字符的单词)

例子：



可以使用WordCloud包轻松查看每个类别中具有最高频率的单词：



Pre-processing:tf-idf:

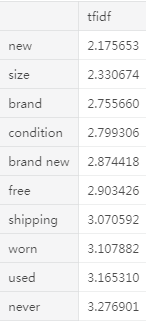
tf-idf是Term Frequency-inverse Document Frequency的首字母缩写。它量化了特定单词相对于文档或语料库集合的词汇表的重要性。该指标取决于两个因素：

术语频率：给定文档中单词的出现

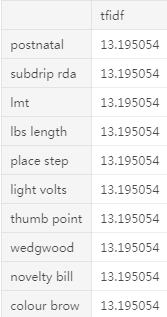
反向文档频率：文档语料库中单词出现的倒数

注：如果在所有文档中广泛使用该词，则它在特定文档中的存在将无法向我们提供有关该文档本身的更多具体信息。可以将“a”，“the”，“and”等词组组合成第二类词组。因此，tf-idf可以被视为特定文档中词语相关性的加权方案。

下面是具有最低tfidf分数的10个词组，是无法用来区分一个描述与另一个描述的非常通用的词：



下面是具有最高tfidf分数的10个词组，其中包含非常具体的单词，通过查看它们，可以猜出它们属于的类别：



鉴于tfidf矩阵的高维度，需要使用奇异值分解（SVD）技术来减小它们的维度。为了使词汇量可视化，接下来可以使用t-SNE将尺寸从50减小到2.。t-SNE更适合将维数降低到2或3。

t-SNE是一种降维的技术，特别适用于高维数据集的可视化。目标是在高维空间中获取一组点，并在较低维空间（通常为2D平面）中找到这些点的表示。它基于概率分布，在邻域图上随机游走，以找到数据中的结构。但由于t-SNE复杂度非常高，通常在应用t-SNE之前会使用其他高维减少技术。

首先，从训练和测试项目的描述中获取样本，因为t-SNE可能需要很长时间才能执行。然后，可以使用SVD将每个矢量的维数减小到n\_components（50）。接着就可以可视化数据点了

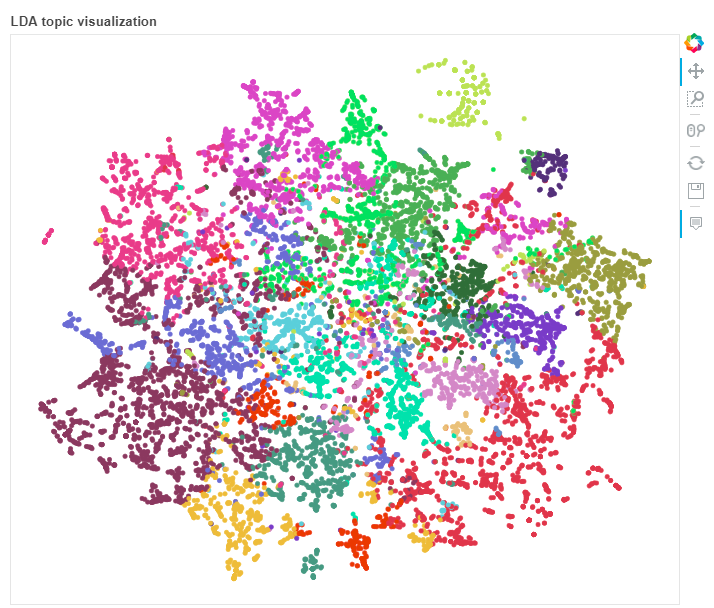
## ****K-Means Clustering****

K-means聚集目标是最小化文档/描述与其聚类质心间的平均欧氏距离。

为了绘制这些聚类，首先需要使用tsne将距离的维度减少到2

## ****Latent Dirichlet Allocation****

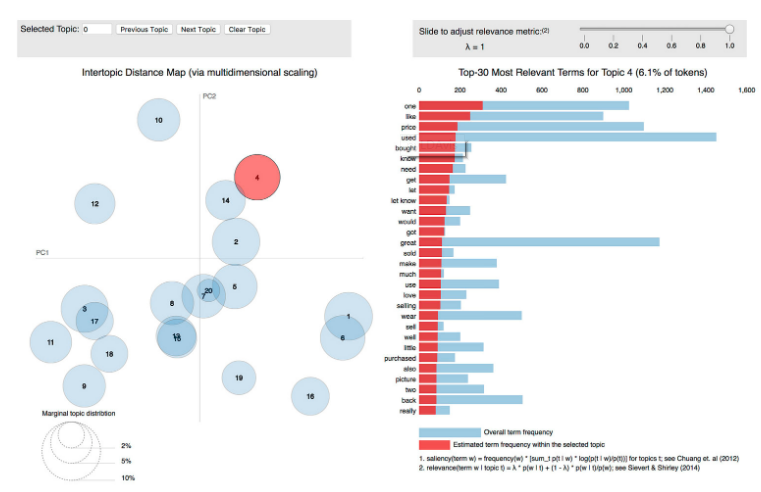
Latent Dirichlet Allocation（LDA）是一种用于发现语料库中存在的主题的算法。 LDA从固定数量的主题开始。每个主题都表示为单词分布，然后每个文档表示为主题分布。虽然词组本身没有意义，但主题提供的词语概率分布提供了文档中包含的不同想法。  
它的输入是一个单词包，即每个文档表示为一行，每列包含语料库中的单词计数。我们将使用一个名为pyLDAvis的强大工具，为LDA提供交互式可视化。



某一数据点事例：



pyLDAvi包：



3. 优秀算法思路

3.1 方案一（TOP1）

此方案由预赛第一第二的Konstantin和Pawel组队完成

Pawel在组队之前的解决方案（RMSLE大约为0.3950）：

Pawel在组队前曾设计了一个比较复杂的模型，由三个部分组成：

1.针对每个类别的商品建立一个模型：Pawel先针对每个商品类别结合运费和商品状况建立了3级的岭回归模型，这个算法在20分钟内得到了0.4050的成绩。

2.多层感知器-Pawel基于1中的模型在稀疏输入（神经网络）上训练下一个模型。目的是得到预测值和真实值之间的差值。

3.LightGBM模型-和2中目的相同

虽然每个类别的训练模型的概念看起来效果很好但实际上并非如此。像这样的训练模型看起来不错，但它低估了正在使用的这个模型。你或许认为神经网络无法通过训练来理解商品类别、商品描述，品牌等之间的相互作用。这是一个错误的假设。我们的例子表明，单个调谐好的神经网络可以学到它想要的东西。

Konstantin在组队之前的解决方案（RMSLE大约为0.3920）：

在正式比赛前，Konstantin建立了两个基本模型

1.Sparse-MLP在tensorflow中的实现

2. 带有conv1d的CNN模型，模型本身并不是很好，但它与MLP模型非常不同，所以给了一个很好的推动。

正式比赛

Pawel与Konstantin决定联手想出一个解决方法而不是简单的结合两个人之前的模型，比赛时他们也更多地关注内存并能够同时运行4个神经网络。

他们将两个人的预处理方案结合，不同的数据集处理方案让他们的成绩提高了0.01

最后他们在每个数据集中使用了3个数据集和4种模型，并且尝试将模型多样化。

***3.1.1*** *方案一数据预处理及特征工程部分方案*

## Build system

如果不能找到一种方法将解决方法分解到模型中，这个任务将很难完成，他们为他们的代码构建了一个独立的系统。最终使用了这个系统创建了python包。  
在3个顺序进程中运行模型非常重要，因为Python在某些操作（特别是数据预处理）后不倾向于清理内存。

## Feature preprocessing

一些技巧：

name chargrams - 虽然不明白具体原因，但是使用n-gram能提高分数。也许是因为它产生了相对密集的特征

词干 – 使用了标准的PorterStemmer

数值向量化 –一个非常大的错误来源是包含以下描述的捆绑项目：“10个数据5个科学家”被矢量化为数据= 10，科学家= 5。仅在1个数据集中应用的该矢量化器将整体改进了0.001。

文本连接 - 通过将文本字段连接在一起来减少文本字段的维度 –最后测试了所有配置{name，item\_description，category，brand}

*3.1.2 方案一模型设计、建立部分方案*

## Models

Pawel发现，通过适当调整的Sparse-MLP模型，与同一数据集上的不同模型相比，最好通过在不同数据集上训练相同模​​型在集合中实现多样性。所以他们决定坚持使用一个真正简化了解决方案的模型。基本模型是具有稀疏输入的多层前馈神经网络：根本没有任何修饰，所以令人惊讶的是它工作十分有效。他们认为其中一个原因是：数据集很大，因此模型必须具有大容量并能够捕获特征交互。但是，当转换CNN可以在嵌入大小为32的4个核心上进行训练时，可以训练4个隐藏大小为256的MLP模型。

仔细的训练调整至关重要：最重要的是，在每个时期之后将批量大小翻倍，这使得模型在每个时期之后更快地运行，并且还得到了更好的最终性能。除批量增加外，他们还降低了学习率。在第二个阶段之后，一切都被调整为获得最佳验证分数，然后在第三个阶段之后过度拟合：这也使得模型在整体中变得更加强大。

他们使用了两种模型变体：一种是使用Huber损失训练（这里更好，因为它对异常值不太敏感），另一种是分类而不是回归。对于分类，将所有价格分成64个桶，并创建一个预测的软目标：首先计算与桶中心的L2距离，然后将softmax应用于此。这种分类模型由于减少了过度拟合而获得了更好的分数，并且还增加了多样性。

对于每个数据集的4个模型中的2个，通过将所有非零值设置为1来在训练和预测期间对输入数据进行二值化。最后还尝试使用非零阈值进行二值化，但这并没有带来太大的改进。

L2正则化有所帮助（他们仅将其应用于第一层），Tensorflow模型也更好地用PRELU激活而不是RELU。

模型实现和优化：由于这是一个对这个数据集大小具有非常严格约束的比赛，因此最大限度地提高训练效率非常重要：通过使第一个隐藏层更大，可以提高分数，除了准备数据集还有12个模型用于训练。最后，设法训练了12个模型，隐藏尺寸256，大约200k特征。具体操作：

TF可以使用多个内核，但它与线性扩展相距甚远，特别是对于稀疏模型。因此，最好为每个模型使用一个核并并行训练4个模型：将TF限制为一个核，OMPNUMTHREADS = 1和通常的TF配置变量，然后允许TF使用多个核，使用未记录的usepersession\_threads = 1进行线程化：这意味着不需要启动多个进程并使用更少的内存。

事实证明，MXNet允许稀疏MLP的更高效的CPU实现，因为它支持稀疏更新。 Pawel编写了一个初始MXNet版本，速度提高了约2倍，然后添加了TF模型的所有功能，使其工作得更好。问题是MXNet执行程序引擎不是线程安全的，如果尝试使用线程进行并行化，则可能得到1个用过的核或段错误。所以不得不转向多处理并编写自己的数据生成器，因为MXNet制作了大量副本并使用了太多内存。这里还有一个将数据放入共享内存的版本，但它太接近磁盘空间限制了，所以不得不划掉它。

整体MXNet解决方案更快，且允许使用较小的初始批量而不会降低速度，但使用更多内存并且看起来不太可靠。所以最后使用相同的数据集进行了两次提交，一次使用MXNet（0.37758私有LB / 0.37665公开），另一次使用TF（0.38006私人/ 0.37920公共）。

最后，共有12个预测，需要合并它们。平均效果很好，但最好调整混合权重，因此使用1％的数据集进行验证（本地5％）并使用Lasso模型调整权重。

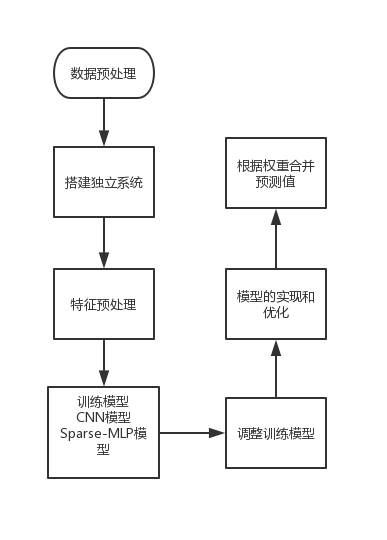
*3.1.3 方案一结果、排名等*

RMSLE = 0.37758

TOP1

**3.1.4** 方案一算法流程图

（这个部分需要自己整理一下绘制，流程图可以用processon画）



**图3-1**

**3.2 方案二(TOP2)**

*3.2.1 方案二数据预处理及特征工程部分方案*

作为预处理步骤，作者对所有名称和描述保留最多60个单词，并对所有文本应用相同的清理过程，他们标记了所有品牌和类别的编码，并将编码值与特征名称连接起来。

***3.2.2*** *方案二模型设计、建立部分方案*

作者制作了4个模型：

1）Ridge模型：训练1-ngrams和custom bigrams（自定义双字母）。

训练1-ngrams时同时使用商品名称和商品描述。自定义双字节组的训练方式如下：将商品名称和商品描述中的前5个单词连接起来，然后在单词列表中应用np.unique()来对它们进行排序并删除重复项，然后创建所有双向可能的单词组合。

ridge模型能够在public LB上得分0.418

2）使用ngram\_range =（1,2）在CountVectorizer上训练的稀疏NN模型。 NN适合来自商品名称和商品特征描述的稀疏数据

3）具有用于商品名称和描述特征的shared\_embedding层的fastText NN模型

4）使用商品名称和描述特征的另一个稀疏NN模型与字符Ngrams相匹配。

作者浪费了一半的时间来构建一个LGB模型，即使它曾经是作者训练过的最好的模型但最后也没有帮助。作者发现使用具有相同数据表示的不同模型是行不通的。所以最后选择使用不同的NN模型和不同的数据输入。

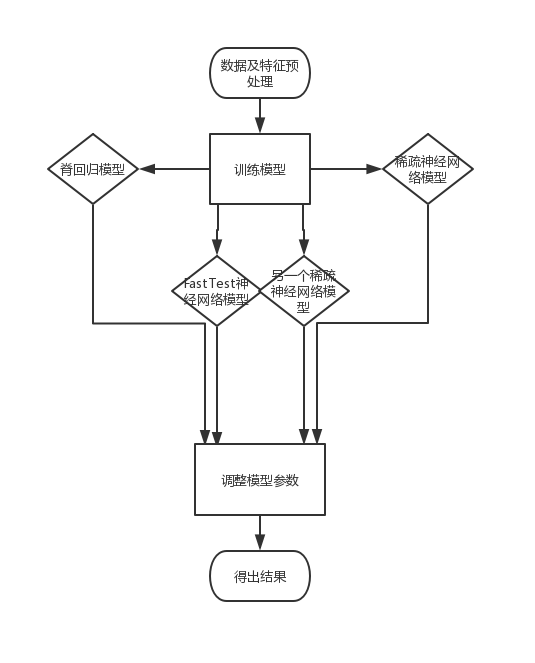
为了使NN运行快速，我们在每个纪元后加倍批量。这非常类似于在每个时期之后用降低学习速率来增快速度。

***3.2.3*** *方案二结果、排名等*

RMSLE =0.38875

TOP2

***3.2.4*** *方案二算法流程图*



4. 算法比较

**表4-1 算法比较**

|  | **评估指标** | **特征工程** | **基础算法** | **基本库** |
| --- | --- | --- | --- | --- |
| 算法一 | **RMSLE** | 特征拆分细化、K-means聚集、数据集降维、tf-idf算法、LDA | **多层感知器模型、CNN卷积神经网络** | **Sklearn**  **Tensorfow** |
| 算法二 | **RMSLE** | 特征拆分细化、LDA等 | **脊回归模型、**  **三种神经网络模型** | **Sklearn**  **Tensorflow**  **Keras** |

5. 总结与展望

**5.1 总结**

本题通过对商品的各项特征进行分析以推算出商品卖价。我认为本题特征处理的重点在于如何将商品的文字描述定量分析而用于模型训练中。还有就是当一种特征过于多且不易区分时，有必要将特征拆分开来分成更小类。

在模型的选择上，两个方案都运用了神经网络模型，且都十分注重模型的多样化。在第一第二的方案中，他们都认为应该对不同模型进行不同的数据输入以达到最好的训练效果。后期的模型优化也十分重要，因为这是一个对数据集大小约束严格的比赛，好的模型优化能大限度地提高训练效率。

**5.2 建模思路**

1.数据预处理：清洗数据并填补缺失值。

2.特征工程：使用heatmap等工具查看各项特征与目标值的关联强弱，选择关联最强的几个特征训练。对于文字描述的处理我没有经验，在本题中学习了wordcloud，tf-idf，会运用在以后的文字处理中。

3.模型选择：选择多个模型分别对数据集进行训练，再根据权重整合结果，得出最优解。