西南财经大学

Southwestern University of Finance and Economics

课程论文

学年学期：	2022-2023 学年第 1 学期
课程名称：	数据挖掘
论文题目：	基于历史数据的白葡萄酒品质预测
学生学号：	42051089
学生姓名：	廖晨皓
学　　院：	工商管理学院
年级专业：	2020 级市场营销（金融服务与营销）

评语：

得　分：

评阅教师签字：　　年　　月

一、数据概况与实验目的

葡萄酒曾经被视为一种奢侈品，如今已获得越来越多的消费者的喜爱与认可。葡萄牙是十大葡萄酒出口国之一，2005 年其占有的市场份额为 3.17% (FAO, 2008)。从 1997 年到 2007 年，其 Vinho Verde 葡萄酒的出口量增加了 36% (CVRVV, 2008)。为了保证其可持续增长，葡萄酒行业正在投资于葡萄酒制造和销售过程的新技术。在当前这一葡萄酒被日渐受追捧的市场背景下，对葡萄酒进行质量检测认定和品质分级已经越发重要。认证可以防止葡萄酒的非法掺假以保障人类健康，并保证葡萄酒市场的质量。质量评估通常是认证过程的一部分，可以用来改善葡萄酒的制造，并对葡萄酒进行质量分级 (Cortez et al., 2009)。

葡萄酒认证一般通过物理化学和感官测试来评估 (Ebeler, 1999)。常规用于描述葡萄酒特征的理化实验室测试包括测定密度、酒精或 pH 值，而感官测试主要依靠人类专家。应该强调的是，味觉是人类感官中最不被理解的 (Smite & Margolskee, 2006)，因此葡萄酒的分类是一项困难的任务。此外，物理化学和感官分析之间的关系是复杂的，仍然没有完全理解 (Legin et al., 2003)。

本次实验的数据来自加州大学尔湾分校所提供的机器学习数据仓库中所提供的 “Vinho Verde”葡萄酒化学分析数据 (UCI Machine Learning Repository, 1987)。本数据集最初来自于 Cortez et al. (2009) 的论文中。本数据集一共包含数据 4898 条、字段 12 个，其中有以 1-10 分对酒品质量进行评价的量化分数以及 11 个记录酒体不同理化成分的字段。

在本实验中，我将利用这一数据集，通过不同模型对数据进行处理并对结果进行比较，以期通过相应的理化性质对酒体质量进行相对准确的预测之目的。

需要说明的是，本次实验的代码均已上传 Github。为优化版式，正文中可能将仅提供展示思路的“伪代码”而不再提供全部代码。完整可运行的代码请在 Github 仓库中进行下载。

二、数据预处理

在数据挖掘中，数据预处理流程十分重要。通过适当的预处理后的数据才能在应用中呈现更优的质量。

在数据预处理中，数据清洗是其中的很大一部分工作。数据清洗（data cleaning）主要通过填补数据集中的缺失值，平滑噪声数据，处理异常值并解决数据不一致性等问题来处理“脏”数据，以达到“清洗”数据的目的。在数据清洗中常见的操作包括删除无关或重复的数据、平滑噪声、处理缺失、处理异常值等操作。

首先，我们检查数据集是否存在重复或空缺数据。

通过引入 Pandas 库并读取数据集后使用 Pandas 所提供的函数对数据集进行去重、去缺失值的处理。

```
//以下函数来自 Pandas 库
data.drop_duplicates(inplace=True)
data.dropna(inplace=True)#参数 inplace=True 代表直接在原 DataFrame 上操作
```

通过这两个函数的操作后，发现剩余数据有 3954 条，仍是一个较大的数据量，应满足我们接下来分析的需要。因此，为保证模型质量，此处对缺失值直接做删除处理，不再保留或进行插值。

对于异常值分析，此处通过计算每一列的 Z-score，然后对出现超过 3 的行进行整行抛弃的做法来进行。

Z-score 是统计学中的一个无量纲量，其计算公式为$z = \frac{x-\mu}{\sigma}$，其中x为需要被标准化的原始值、μ为总体的平均值、σ为总体的标准差。Z 值的量代表着原始值和总体平均值之间的距离，是以标准差为单位计算。在原始分数低于平均值时 Z 则为负数，反之则为正数。换句话说，Z 值是从感兴趣的点到均值之间有多少个标准差。在此我们认为 Z-score 绝对值大于 3 的值为异常值，应当予以提出。“3 个标准差” (又称为 3-Sigma)这一标准，是在大量文献和实践中被广泛接受的一个标准，在 3 个标准差之内包括了 99.7%的数据。

再次说明，同本实验的其他代码一样，这一部分的代码仍然已上传至 Github，此处仅展示部分核心思路。

```
//Preprocessing.py
# 通过 Z-Score 方法判断异常值
df_zscore = data.copy()  # 复制一个用来存储 Z-score 得分的数据框
```

```
cols = data.columns  #  获得列表框的列名
for col in cols:
    df_col = data[col]  #  得到每一列的值
    z_score = (df_col - df_col.mean()) / df_col.std()  #  计算每一列的
Z-score 得分
    df_zscore[col] = z_score.abs() > 3  # 判断 Z-score 得分是否大于 3，如
果是则是 True，否则为 False
df_drop_outlier = data.copy()
for col in cols:
    df_drop_outlier = data[df_zscore[col] == False] # 丢弃这一列所有 Z-
score>3 的值，通过 for 循环达到“清除每一列”的目的
    print(df_drop_outlier) #观察是否达到清除异常值的目的
```

在经过异常值检测后，Z-score 小于 3 的数据有 3929 条。本次实验接下来的部分将以这一经过预处理后的数据集为基础来进行。

自此，我们便完成了数据清理工作。

由于一些模型对输入的数据有特定的格式要求，因此数据预处理的一个重要步骤就是数据变换。数据变换是将数据从一种格式转换为另一种格式的过程。主要是对数据进行规范化的操作，将数据转换成“适当的”格式，以适用于挖掘任务及算法的需要。

目前，Python 中对于数据变换已有相对成熟的第三方库供使用。

```
//Preprocessing.py
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
```

一般而言，涉及到“空间”、“距离”相关的算法，都需要对数据进行标准化和归一化。因此，我们在此处提前对数据进行这一处理，以备后用。

```
//Preprocessing.py
//此处复用上文中的检测异常值小节曾使用过的代码，只不过是将其用途转化为计算并
保存 Z-score 到本地 CSV 文件
for col in cols:
    df_col = df_drop_outlier[col]
    z_score = (df_col - df_col.mean()) / df_col.std()
    df_zscore[col] = z_score
    df_zscore.to_csv('./z-score.csv')
```

对于完成数据清洗但未进行数据变换的数据集，此处将其存于本地，以备后用

```
//Preprocessing.py
df_drop_outlier.to_csv('after_preprocessing.csv')
```

到此，我们对于数据的预处理步骤就基本完成。

三、数据可视化

在构建模型并对数据进行回归分析之前，我们可以利用数据可视化工具对数据进行探索。数据可视化可以将抽象的数据转化为更为直观的视觉表达，有助于我们厘清研究的方向，同时对于变量选择也大有裨益。

在 Python 中有着大量成熟的第三方库可用于数据的可视化。在此我们选择较为成熟的 Matplotlib 使用。通常，在使用数据可视化包的同时，需要配合 Pandas/Numpy 等库一同使用。

```
// visualize.py
import pandas as pd
import seaborn as sns
import numpy as np
from matplotlib import pyplot as plt
plt.rcParams["font.sans-serif"]=["SimHei"] #设置中文字体，防止图像中的中文变成乱码
plt.rcParams["axes.unicode_minus"]=False #该语句解决图像中的“-”负号的乱码问题
from pandas import read_csv
df = read_csv('./after_preprocessing.csv')
df.drop(df.columns[0], inplace=True, axis=1) #删除原文件中多出来的一行index，下同
df_z = read_csv('./z-score.csv')
df_z.drop(df_z.columns[0], inplace=True, axis=1)
```

经过以上操作，我们便成功的对数据进行了读取，可进行数据可视化。

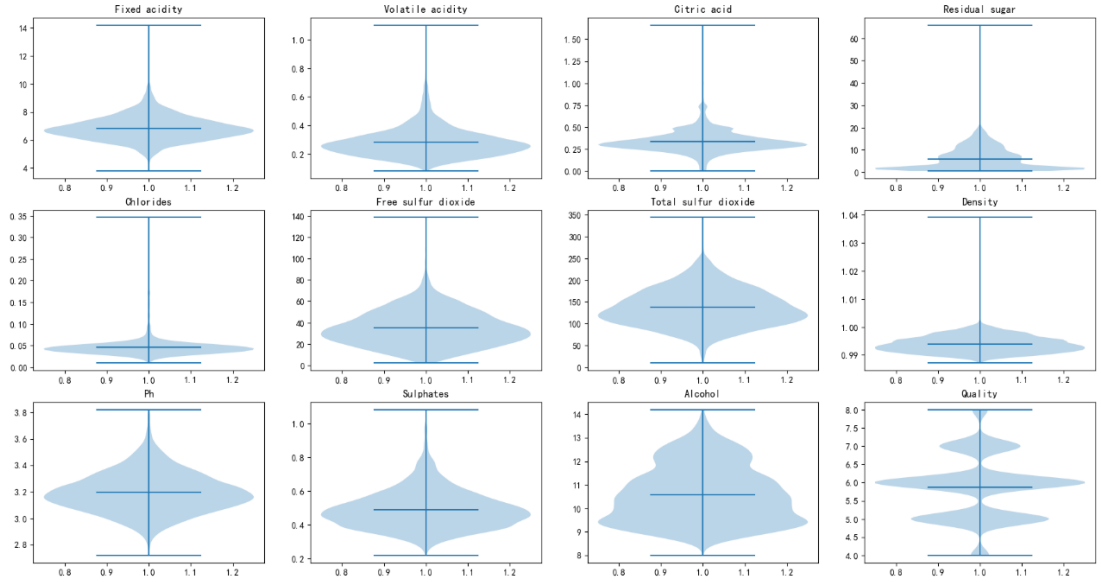


图 1 各变量小提琴图

首先，我在此对各个变量分别绘制了小提琴图。小提琴图用于显示数据的分布及其密度，这种图表结合了箱型图和密度图的特征，显示了数据的分布形状。在图 1 中可看到，大部分数据都聚集在图片的中下部分。但也有例外，如总 SO_2 数值和 pH 值则是主要分布在图表的中央部分，酒精度则是在图表各处均有分布，相对较为均衡。而对于酒体质量则是质量为 6 的酒占最多，而质量分别为 5.0 与 7.0 左右的酒也相对较多。

对于此类在同一画布中存在多个子图的图表，我们可以借助 Matplotlib 来实现。

```
plt.figure(figsize=(10,6)) #首先创建一个新的画布，并指定尺寸为(10, 6)
plt.subplot(341) #通过 subplot 函数，在刚才创建的画布上创建一个子画布，其中的参数指的是：共 3 行 4 列 这是第 1 个子画布，此外还可以用 plt.subplot(3, 4, 1)来指定子画布
```

```
plt.subplot(341)
plt.violinplot(fa, showmeans=True)
plt.title('Fixed acidity')
这一部分便是对这一个子画布进行操作的区域
```

```
plt.subplot(342)#切换到下一个子画布，并进行作图，以此类推…
plt.show() #此时将会把之前用 plt.figure()创建的画布上的所有子画布同时显示出来
```

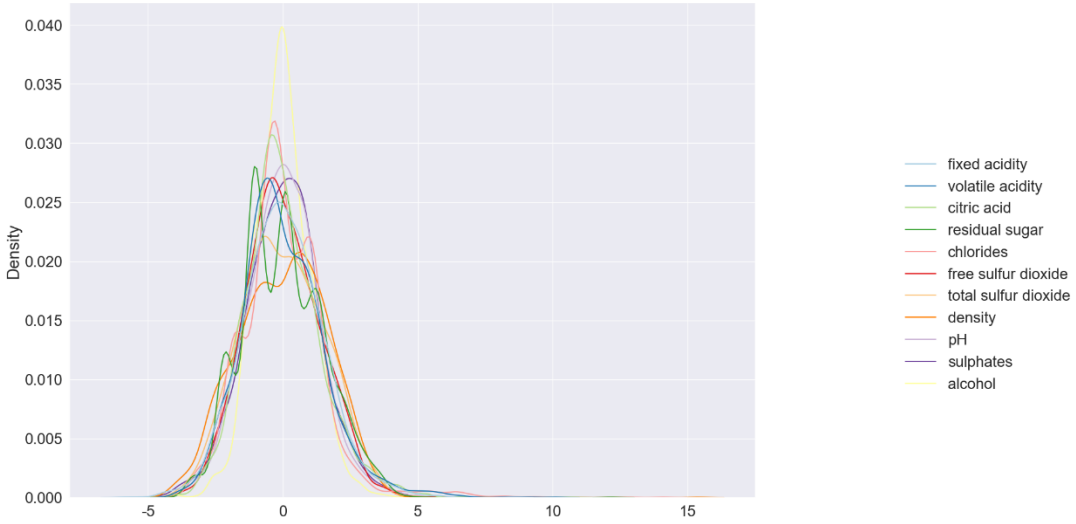


图 2 各变量 Z-score 与质量 Z-score 相减后的分布密度

为了探索哪些变量与质量的关联最大，在此我做了一个简单的可视化实验。我将每一个标准化后的变量与这瓶酒对应的标准化后的质量进行相减。所得到的结果代表了这瓶酒的这一性质和酒的质量偏差的方向与数值。在对所有的数据都进行这一操作之后，将其画为一幅分布密度图。借助 seaborn 包的相关函数，可以更加轻松画出此类图片。

```
// visualize.py
sub = df_z.copy()
for i in range(11):
    data = df_z[cols[i]].values
    for j in range(len(data)):
        k = data[j] - qu_z[j]
        sub.iloc[j, i] = k
sub.drop('quality',axis=1,inplace=True)
sns.set(font_scale=1.8)
sns.displot(data=sub, palette='Paired', kind='kde')
plt.show()
```

在图 2 中可以看到，酒精度、硫酸盐、柠檬酸、氯化物等变量的偏差与质量的偏差的差大量聚集在 0 附近，这代表他们分别与质量的偏差的方向和大小相

近，可能代表这他们之间存在一定的相关关系。不过，此处仅为简单的可视化探索，变量之间实际的关系将在下一节被具体测量。

四、建模与检验

在本节中，我将对本次数据挖掘实验中所采用的指标和方法进行介绍，建模并对其效果进行检验。

4.1 方法与指标介绍

在开始在正式的建模与训练之前，首先我将介绍并引入几个用于衡量模型表现的指标。

通常而言，回归模型的表现可以使用如平均绝对误差 (MAE)这样的算法来进行测量。平均绝对误差是所有单个观测值与算术平均值的偏差的绝对值的平均。平均绝对误差可以避免误差相互抵消的问题，因而可以准确反映实际预测误差的大小，其公式如下：

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|E_i| = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right|$$

其中Y_i与$\hat{Y}_i$分别代表实际值和模型预测值。

MSE（mean squared error，均方误差）是常用的回归模型评估指标。它表示预测值与实际值之差的平方的平均值。

MSE 计算公式如下：

$$MSE = \frac{1}{n} * \sum\left(y_{pred} - y\right)^2$$

其中，n 是样本数量，y_pred 是预测值，y 是实际值。

MSE 常用于评估回归模型的性能。当 MSE 较小时，表示模型的预测精度较高；当 MSE 较大时，表示模型的预测精度较低。

对于模型的评估，还可以采用如识别准确度 (Accuracy)的方法进行评价。

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

换而言之，识别准确度等于：$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$

而精确率 (Precision) 指被分类器判定正例中的正样本的比重，其计算公式为$Precision = \frac{TP}{TP+FP}$

召回率 (Recall) 指的是被预测为正例的占总的正例的比重，其计算公式如下：$Recall = \frac{TP}{TP+FN}$

对于连续型的变量还可以使用如均方根误差（Root Mean Squared Error，RMSE）：是 MSE 的平方根。RMSE 可以反映模型的预测偏差的大小；R^2 分数（R^2 Score）：指回归模型的拟合程度。R^2 分数越高，说明模型越好。

Hold-out 方法将数据随机划分为训练和测试子集。前一个子集用于测试模型(比如有 2/3 的数据)，而后一个子集(比如有剩余的 1/3)用于计算估计。在 python 中，可以使用 scikit-learn 提供的 train_test_split 函数进行这一操作。另一个更好的算法是 k-fold 交叉验证法，其中数据被分为大小相等的 k 个分区。每次测试一个子集，其余的数据用于测试模型。这个过程依次重复，直到所有的子集都被测试完毕。因此，在这个方案下，所有的数据都用于训练和测试。然而，这种方法需要大约 k 倍的计算量，因为有 k 个模型被测试。但 k-fold 方法因为计算量显然更大，因此其性能开销也更大。

4.2 变量选择

使用 scikit-learn 库的 SelectKBest 函数来选择最重要的特征。根据每个特征的 F 值进行排序。这个函数可以同时用于回归和分类任务。

```
//varSelection.py
from sklearn.feature_selection import SelectKBest, f_regression
from pandas import read_csv
df = read_csv('./after_preprocessing.csv')
y_train = df['quality']
x = df.drop('quality', axis=1)
X_train = x
selector = SelectKBest(f_regression, k=5)
X_new = selector.fit_transform(X_train, y_train)
selected_feature_indices = selector.get_support(indices=True)
for i in selected_feature_indices:
    print(df.columns[i])
```

经过计算，得到 5 个 F 值最高的特征：

表 1 变量选择

排名	变量名称	中文名
1	volatile acidity	挥发性酸
2	chlorides	氯化物
3	total sulfur dioxide	总 SO_2
4	density	密度
5	alcohol	酒精度

4.3 支持向量回归(SVR)算法

SVM 算法是针对离散变量，而此处针对连续变量我们需要使用 SVR 算法。根据前一小节中得到的变量，在这一节中，我将尝试不同的核函数组合进行训练并对训练结果进行评估。

要使用 SVR 算法，需要引入 scikit-learn 中的特定函数。并且同样使用 scikit-learn 进行测试集和训练集的划分，以在训练后评价效果。

本次训练共选择了三个核函数：linear、poly 和 rbf 进行测试，并对 MSE、MAE 和 R^2 进行记录，作图如下：

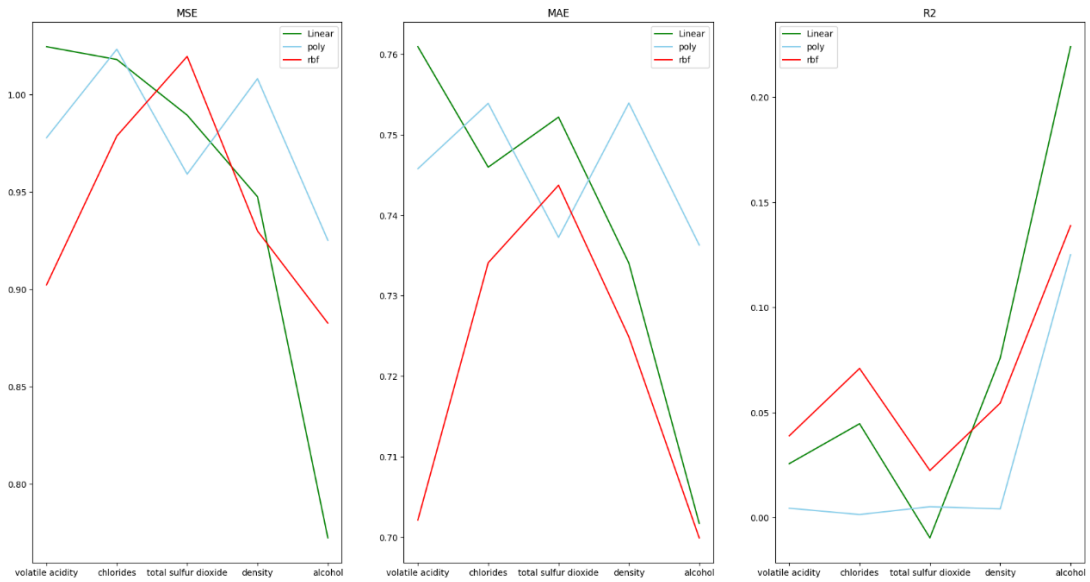


图 3 使用不同核函数下的 SVR 算法

即使不对其具体数字进行展示，也可看出这一方法得出的结果极差，在本次

实验任务中无利用价值。

4.4 支持向量机(SVM)算法

鉴于 SVM 仅可接受离散点，因此需要将连续型变量进行转换。

```
//SVM.py
from sklearn.preprocessing import KBinsDiscretizer
est = KBinsDiscretizer(n_bins=k, encode='ordinal',
strategy='uniform')#其中，n_bins 参数指定离散化后的分段数，encode 参数指定离散化后的输出类型，strategy 参数指定离散化后每一段的分布策略。
X_transformed = est.fit_transform(X)
```

在完成分箱等数据处理后，我使用 linear、poly 和 rbf 三种核函数对数据进行了训练尝试，其训练的准确度如下图所示：

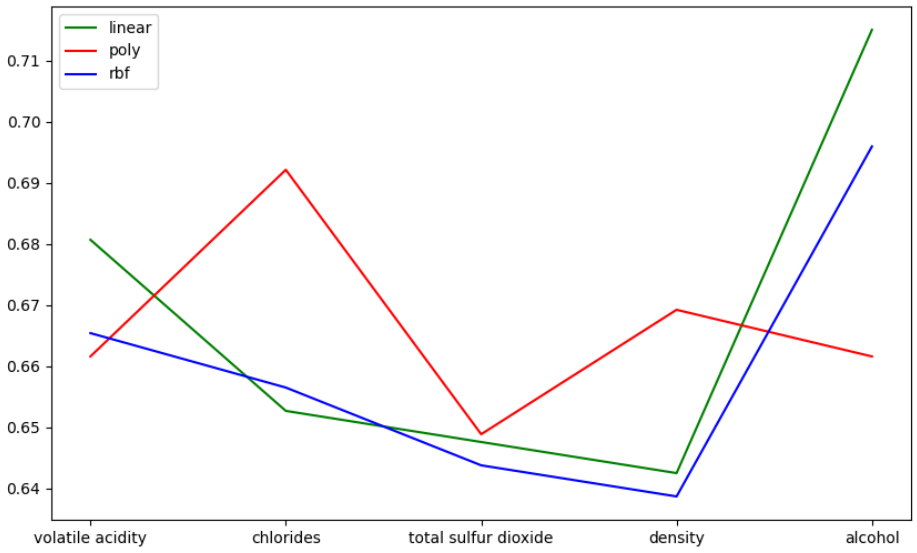


图 4 SVM 模型训练准确度

从图上可明显看出，模型准确度均在 60%以上，对于酒精度变量，使用 linear 和 poly 核函数甚至有接近 70%的准确度。这说明 SVM 在解决当下问题中有较好的效果，可以考虑采用。

4.5 神经网络 (NN)

神经网络是一种机器学习模型，用于从数据中学习模式。它通常由输入层、

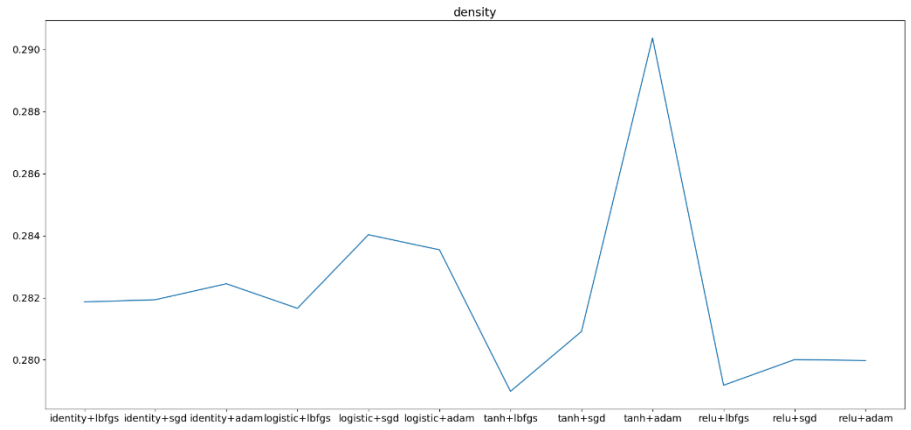
隐藏层和输出层组成。输入层的节点代表输入的特征，输出层的节点代表预测的标签。隐藏层的节点连接输入层和输出层，用于提取特征并传递信息。神经网络的权重是每个节点对输出的贡献程度。神经网络通过计算输入层的权重和再通过激活函数进行转换，得出隐藏层的输出，最终通过隐藏层的输出得出输出层的预测值。在此，我们利用神经网络构建回归模型。

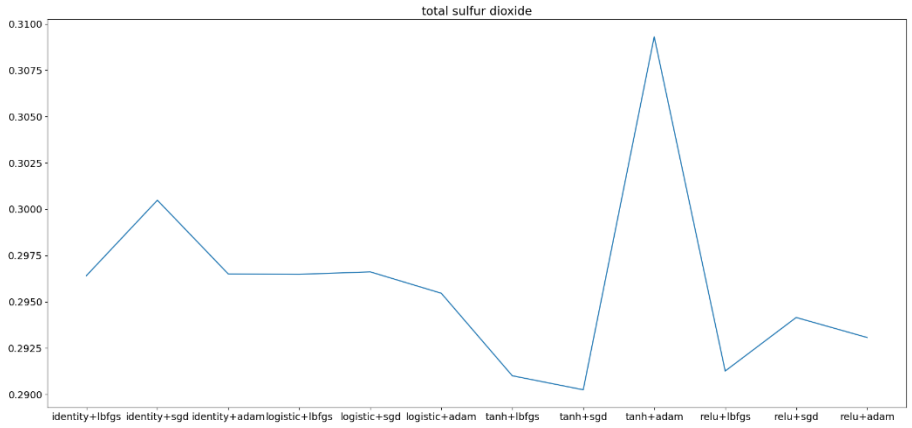
目前有相当多的第三方可以构建并训练神经网络模型，如 TensorFlow、PyTorch 以及常用的 scikit-learn。本研究中选择使用 scikit-learn 进行训练。

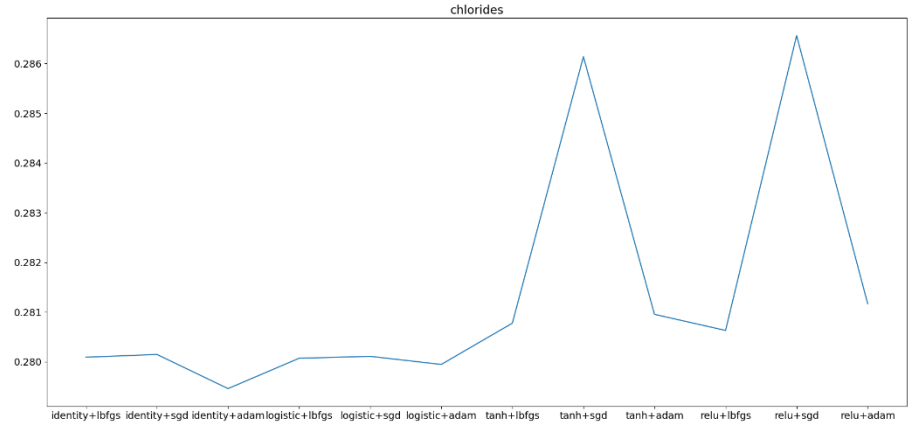
```
#NN.py
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
```

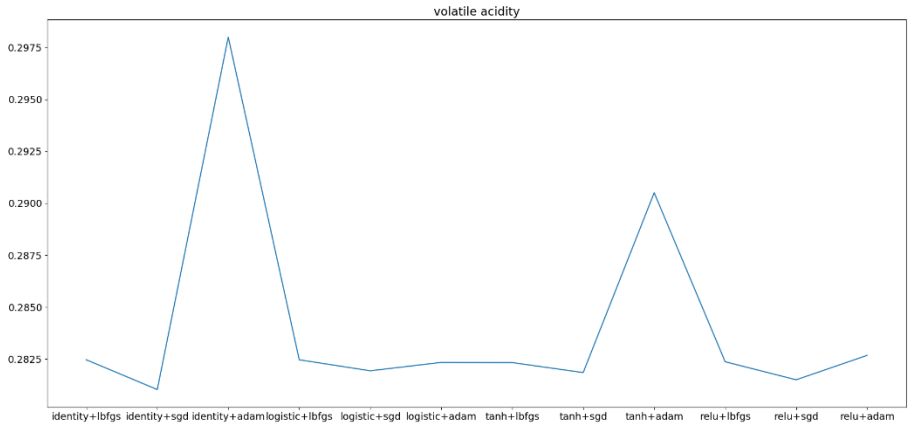
在引入相关库之后，使用数据对其进行训练，并使用均方误差 (MSE)对结果进行评价。

scikit-learn 中对”solver”和”activation”分别提供了 3 种和 4 种不同的选项，即 12 中不同组合。为方便检查不同组合的效果，我将训练结果可视化进行展示。




total sulfur dioxide
0.3100
0.3075
0.3050
0.3025
0.3000
0.2975
0.2950
0.2925
0.2900
identity+lbfgs identity+sgd identity+adam logistic+lbfgs logistic+sgd logistic+adam tanh+lbfgs tanh+sgd tanh+adam relu+lbfgs relu+sgd relu+adam


chlorides
0.286
0.285
0.284
0.283
0.282
0.281
0.280
identity+lbfgs identity+sgd identity+adam logistic+lbfgs logistic+sgd logistic+adam tanh+lbfgs tanh+sgd tanh+adam relu+lbfgs relu+sgd relu+adam


volatile acidity
0.2975
0.2950
0.2925
0.2900
0.2875
0.2850
0.2825
identity+lbfgs identity+sgd identity+adam logistic+lbfgs logistic+sgd logistic+adam tanh+lbfgs tanh+sgd tanh+adam relu+lbfgs relu+sgd relu+adam

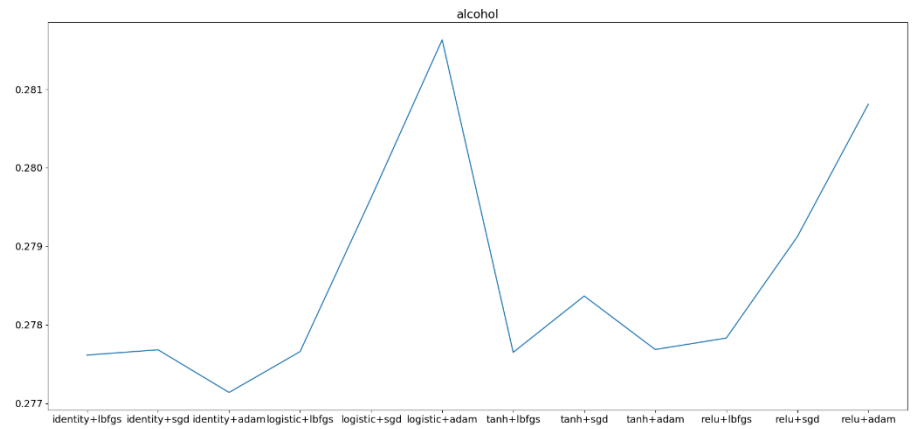

alcohol
0.281
0.280
0.279
0.278
0.277
identity+lbfgs identity+sgd identity+adam logistic+lbfgs logistic+sgd logistic+adam tanh+lbfgs tanh+sgd tanh+adam relu+lbfgs relu+sgd relu+adam

图 5 神经网络训练结果

尽管各组合的 MSE 值各有不同，但总体而言都大约在 0.3 以下表现极为优秀。可以考虑以此作为酒品质量鉴定方法之一。

表 2 神经网络各变量的最优结果

变量	组合	MSE
volatile acidity	identity+sgd	0.2810
chlorides	identity+adam	0.2795
total sulfur dioxide	tanh+sgd	0.2903
density	tanh+lbfgs	0.2790
alcohol	identity+adam	0.2771

在表 2 中，我整理了不同变量下最优的神经网络激活函数和梯度下降算法组合，可供备选。

4.6 线性回归

线性回归是一种机器学习方法，它用于预测连续型输出变量（也称为因变量）的值，其中输入变量（也称为自变量）可以是一个或多个。线性回归假设输入变量与因变量之间存在线性关系，并尝试通过学习输入变量与因变量之间的映射关系来建立一个线性模型。在线性回归中，我们尝试找到一条直线（称为回归线），使得其与训练数据中的所有观测值尽可能接近。在 Python 中进行线性回归有很多种方法。一种方法是使用 scikit-learn 库中的 LinearRegression 类。

```
//linear_reg.py
from sklearn.linear_model import LinearRegression
```

通过对之前所选定的五个变量与质量这一变量进行线性回归，我们得到了五

个线性变量模型下各自的 MSE，如下图所示。

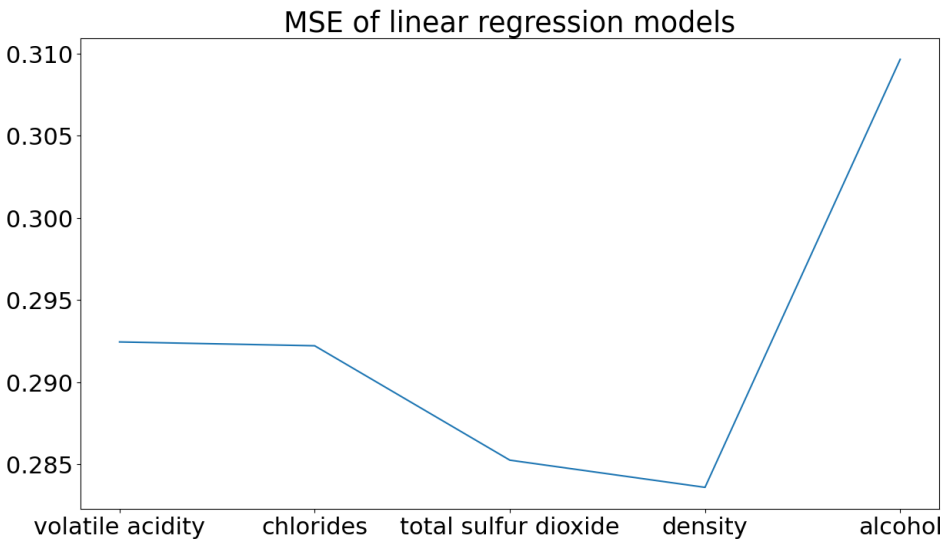


图 6 线性回归模型 MSE 汇总

在线性模型中，密度-质量和总 SO_2-质量两个模型的 MSE 最小，但所有模型的 MSE 都在可接受且效果较好的范围之中。

总结

在本次实验中我们共采用了四种不同的方法进行试验：SVR、SVM、神经网络和线性回归，其中除了 SVR 以外的三种方法都得到了有参考价值的模型，下表是对各种方法中有利用价值的酒品质量确定方法的总结

表 3 模型总结

方法	自变量*
SVM	酒精度 (linear) 氯化物 (poly)
神经网络	酒精度 (identity+adam) 挥发性酸 (identity_sgd) 密度 (tanh+lbfgs)
线性回归	总 SO_2 密度

*：因变量为质量

在表 3 中所列出的自变量均与质量之间有着较好的相关性，从表 3 中可见，酒精度-质量、密度-质量这两对关系的表现较为优异，多次出现在以不同方法进行回归的较优结果中。

根据不同的场景，我将推荐不同的模型来基于酒品的理化指标来对酒品的质量进行预测。

酒精度是酒最为常见、最易获取的指标，在很多情况下，无论是研究人员还是消费者都不需要对酒进行精密的测验就可从外包装上看到明确的酒精度标识。且本次实验中，酒精度表现出了与质量较强的关系。因此，在许多情况下可以用酒精度来对酒的质量进行预测，比如对酒在认证时进行初步分类时、消费者在零售场景中选购酒品时。

密度代表了单位体积物质的质量 (此处的“质量”指代“mass”)，酒的密度越高，单位酒液中含有的物质越多，因此也可能含有更多可以提高口感和质量(指代“quality”)的风味物质。在本次实验中，密度表现出了与质量 (指代“quality”)高度的相关性，加之密度也是一个相对容易测定的物理量。在对酒的质量进行预测时，可以选用密度这一指标。

对于氯化物、挥发性酸和总 SO_2 指标而言，他们也表现出了与质量的相关性。

葡萄酒中的氯化物是指在酒中含有的氯原子的化合物。氯化物在葡萄酒中的主要来源有两种：一种是在酿酒过程中使用的清洁剂，例如氯消毒剂和碳酸氢钠（即常用的苏打水）；另一种是在酿酒过程中产生的天然化合物，例如酒精代谢过程中产生的乙醛和乙酸。

在葡萄酒中，挥发性酸可能来自于葡萄皮、葡萄蒂和发酵过程中产生的酸。酒中的挥发性酸含量通常以毫克乙酸相当物计算，即在等体积的乙酸中含有的挥发性酸的毫克数。挥发性酸也是葡萄酒中的重要成分之一，它与其他成分共同贡献了酒的香气和口感。

硫化物在葡萄酒中主要来源于葡萄本身，在葡萄酒的酿造过程中也会产生一些硫化物。总硫化物在葡萄酒中的含量通常以毫克硫酸盐形式的硫化物计算，即在等体积的硫酸中含有的硫化物的毫克数。

这三者对于葡萄酒风味的贡献都是复杂多样的，对于不同产区、不同庄园、不同葡萄品种、不同工艺的葡萄酒而言，他们有着大相径庭的作用。因此，需要

有经验的酿酒师在实践过程中与科研人员不断沟通，并调整这三大指标在预测葡萄酒质量的模型中的权重。

就算法而言，SVM、神经网络和线性回归都是在该领域非常经典的算法。SVM（支持向量机）、线性回归和神经网络都是常用的机器学习算法，它们各有优劣，在不同的场景下表现不同。

在性能方面，SVM 在许多分类和回归问题中都表现出色，并且能够很好地处理高维数据。线性回归适用于预测连续型输出变量的值，在线性数据上表现较好，但是对于非线性数据可能表现不佳。神经网络在解决复杂的分类和回归问题方面表现出色，但是训练时间可能较长。在开销方面，SVM 的训练时间较长，但是它的预测速度很快。线性回归的训练和预测时间都很短，但是它只能用于线性数据。神经网络的训练时间较长，但是预测速度较快。

因此，在选择算法时，需要根据具体的应用场景和性能要求来确定最合适的算法。

总而言之，本研究认为在指标选择上密度和酒精度与酒的质量为较强的正相关。而挥发性酸、总硫化物、氯化物与酒的质量关系复杂，尽管在本研究所设计的样本数据中表现出与酒的质量呈正相关，但仍然需要酿酒师与研究人员采取更广泛的研究来最终决定其在模型中的地位。至于算法选择，SVM、线性回归和神经网络都是相当经典的算法，可以根据实际情景进行选择或搭配。

参考文献

A. Legin, A. Rudnitskaya, L. Luvova, Y. Vlasov, C. Natale, and A. D'Amico. Evaluation of Italian wine by the electronic tongue: recognition, quantitative analysis and correlation with human sensory perception. Analytica Chimica Acta, 484(1):33-34, 2003.

CVRVV. *Portuguese Wine - Vinho Verde. Comiss~ao de Viticultura da Regi~aodos Vinhos Verdes (CVRVV), http://www.vinhoverde.pt, July 2008.*

D. Smith and R. Margolskee. *Making sense of taste. Scientific American, Special issue, 16(3):84-92, 2006.*

FAO. FAOSTAT - *Food and Agriculture Organization agriculture trade domain statistics. http://faostat.fao.org/site/535/DesktopDefault.aspx?PageID=535, July 2008.*

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. *Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.*

S. Ebeler. *Flavor Chemistry - Thirty Years of Progress, chapter Linking flavour chemistry to sensory analysis of wine, pages 409{422. Kluwer Academic Publishers, 1999.*

UCI Machine Learning Repository: *Wine Data Set*. (1987). http://archive.ics.uci.edu/ml/datasets/Wine

附录

完整源代码地址：https://github.com/LisPerfect/DataMiningExp