数据挖掘大作业一报告

数据探索性分析与数据预处理

姓名：郑安庆 学号：3120181078

一、 问题描述：

本次作业中，将选择 2 个数据集进行探索性分析与预处理。

- 在数据摘要任务中，对于数据集中的标称属性，给出每个可能取值的频数；对于数值属性，给出最大、最小、均值、中位数、四分位数及缺失值的个数。
- 在数据可视化任务中，对数值属性绘制直方图，用 qq 图检验其分布是否为正态分布；对数值属性绘制盒图，对离群值进行识别。
- 在数据缺失的处理任务中，观察数据集中缺失数据，分析其缺失的原因。分别使用下列四种策略对缺失值进行处理：（1）将缺失部分剔除（2）用最高频率值来填补缺失值（3）通过属性的相关关系来填补缺失值（4）通过数据对象之间的相似性来填补缺失值。处理后，可视化地对比新旧数据集。

二、 数据说明：

- **数据集 1：Wine Reviews**

文件名	数据记录个数	属性条数
winemag-data_first150k.csv	150930	11
winemag-data-130k-v2.csv	129971	14

- **数据集 2：Oakland Crime Statistics 2011 to 2016**

文件名	数据记录个数	属性条数
records-for-2011.csv	180016	10
records-for-2012.csv	187431	10
records-for-2013.csv	188052	10
records-for-2014.csv	187480	10
records-for-2015.csv	192581	10
records-for-2016.csv	110828	10

三、 数据分析过程：

3.1 数据可视化和摘要

◆ **数据摘要：**

通过使用 python 语言的 pandas 包的 read_csv 函数将文件数据读入成 DataFrame 格式数据，通过判断数据类型为 int64 或者 float64 判断列的属性是否为数值属性，反之则是标称属性。

对于数据集 Wine Reviews 的 winemag-data_first150k.csv 文件，其包含 8 个标称属性，分别是'country', 'description', 'designation', 'province', 'region_1', 'region_2', 'variety', 'winery'；同时包含 2 个数值属性，分别是'points', 'price'。

对于数据集 Wine Reviews 的 winemag-data-130k-v2.csv 文件，其包含 11 个标称属性，分别是'country', 'description', 'designation', 'province', 'region_1', 'region_2', 'taster_name', 'taster_twitter_handle', 'title', 'variety', 'winery'；同时包含 2 个数值属性，分别是'points', 'price'。

对于数据集 Oakland Crime Statistics 2011 to 2016 的 records-for-2011.csv 和 records-for-2013.csv 文件，其包含 8 个标称属性，分别是'Agency', 'Create Time', 'Location', 'Beat', 'Incident Type Id', 'Incident Type Description', 'Event Number', 'Closed Time'；同时包含 2 个数值属性，分别是'Area Id', 'Priority'。

对于数据集 Oakland Crime Statistics 2011 to 2016 的 records-for-2012.csv 和 records-for-2014.csv 文件，其包含 8 个标称属性，分别是'Agency', 'Create Time', 'Beat', 'Incident Type Id', 'Incident Type Description', 'Event Number', 'Closed Time', 'Location 1'；同时包含 2 个数值属性，分别是'Area Id', 'Priority'。

对于数据集 Oakland Crime Statistics 2011 to 2016 的 records-for-2015.csv 和 records-for-2016.csv 文件，其包含 9 个标称属性，分别是'Agency', 'Create Time', 'Location', 'Area Id', 'Beat', 'Incident Type Id', 'Incident Type

Description', 'Event Number', 'Closed Time'；同时包含 1 个数值属性，是'Priority'。

通过在 Data 类的 process_features()函数判断列的属性后分别使用 process_nom_features()函数和 process_num_features()函数来对列的标称属性和数值属性进行处理。

```
for title in content.columns.values:
    if title == "Unnamed: 0":
        continue
    if content[title].dtypes == "int64" or content[title].dtypes == "float64":
        self.process_num_features(content, title, write_data_path)
    else:
        self.process_nom_features(content, title, write_data_path)
```

对于标称属性，process_nom_features()函数使用 dict 返回该属性对应数据的各个取值及其频数，获得的对应结果位于"./result/[数据集名]/[[文件名]/nominal_attribute"目录下，文件格式为：

country.txt：　　#结果文件名

Feature Name: country　　#属性名称

Value Num: 44　　#不同的取值个数

Canada,257　　#取值 1，个数

Italy,19540　　#取值 2，个数

Czech Republic,12　　#.....

对于数值属性，process_num_features()函数使用 DataFrame 自带的函数返回该属性的最大、最小、均值、中位数、四分位数及缺失值的个数，获得的对应结果位于"./result/[数据集名]/[[文件名]/ numeric_attribute"目录下，文件格式为：

points.txt：　　#结果文件名

Feature Name: points　　#属性名称

Max Num: 100　　#最大值

Min Num: 80　　#最小值

Mean Num: 88.4471382078　　#平均数

Median Num: 88.0 #中位数

Quartile Num: 86.0, 91.0 #下、上四分位数

Missing Num: 0 #缺失值个数

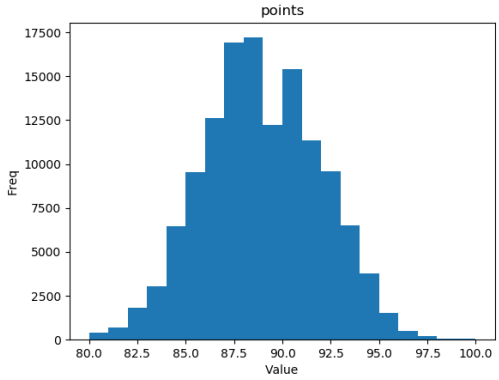
◆ **数据可视化：**

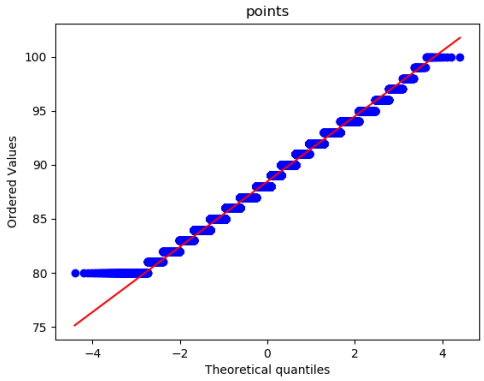
对于两个数据集的全部属性，在 draw_figure()函数中分别调用 draw_histogram()、draw_qq()和 draw_box()函数绘制直方图、qq 图和盒图，调用方法如下：

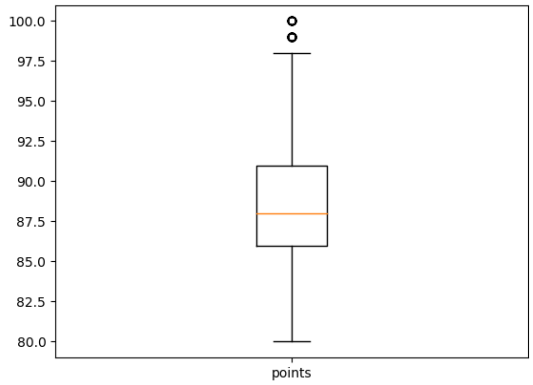
```
# draw histogram
figure_path = os.path.join(write_figure_path, title + "_histogram.png")
self.draw_histogram(content, title, figure_path)
# draw qq
figure_path = os.path.join(write_figure_path, title + "_qq.png")
self.draw_qq(content, title, figure_path)
# draw box
figure_path = os.path.join(write_figure_path, title + "_box.png")
self.draw_box(content, title, figure_path)
```

绘图结果分别位于："./result/[数据集名]/[[文件名]/figure/[列名]_histogram.png"、"./result/[数据集名]/[[文件名]/figure/[列名]_qq.png"和"./result/[数据集名]/[[文件名]/figure/[列名] _box.png"。

以数据集 Wine Reviews 的 winemag-data-130k-v2.csv 文件的'points'属性举例，三类图结果为：







对于数据集 Wine Reviews 的 winemag-data_first150k.csv 和 winemag-data-130k-v2.csv 文件，数值属性'points'符合正态分布，数值属性'price'不符合正态分布。

对于数据集 Oakland Crime Statistics 2011 to 2016 的 records-for-2011.csv 、 records-for-2012.csv 、 records-for-2013.csv 和 records-for-2014.csv 文件，数值属性'Area Id'和'Priority'都不符合正态分布。

对于数据集 Oakland Crime Statistics 2011 to 2016 的 records-for-2015.csv 和文件 records-for-2016.csv，数值属性'Priority'不符合正态分布。

3.2 数据缺失的处理

在此小节中，使用四种策略对数值属性的缺失值进行处理，数据缺失的原因可能为：

- 记录数据时遗漏
- 因为某些属性关联依赖导致的该属性无对应值

分别使用下列四种策略对缺失值进行处理：（1）将缺失部分剔除（2）用最高频率值来填补缺失值（3）通过属性的相关关系来填补缺失值（4）通过数据对象之间的相似性来填补缺失值。

通过使用 data 类的 filling()函数来对缺失值进行处理。获得的对应结果位于"./result/[数据集名]/[[文件名]/ strategy_*.csv(*=1,2,3,4)"目录下。

3.2.1 将缺失部分剔除

使用 python 语言的 pandas 包的 dropna()函数可直接将包含 nan 的数据

删掉，具体实现如下：

```
strategy_path = os.path.join(write_data_path, "strategy_1")
makedir(strategy_path)
with open(os.path.join(strategy_path, file_name), 'w', encoding = 'utf-8') as fp1:
    strategy1_content = content
    for title in content.columns.values:
        if title == "Unnamed: 0":
            strategy1_content = strategy1_content.drop(columns = [title])
        elif strategy1_content[title].dtypes == "int64" or strategy1_content[title].dtypes == "float64":
            strategy1_content = strategy1_content.dropna(subset = [title])
            self.draw_figure(strategy1_content, title, strategy_path)
    strategy1_content.to_csv(fp1)
```

3.2.2 用最高频率值来填补缺失值

通过调用 get_feature_value()函数得到该属性所有可能的取值及其频数，取其最大值后使用 python 语言的 pandas 包的 fillna()函数来填补缺失值，具体实现如下：

```
strategy_path = os.path.join(write_data_path, "strategy_2")
makedir(strategy_path)
with open(os.path.join(strategy_path, file_name), 'w', encoding = 'utf-8') as fp2:
    strategy2_content = content
    for title in content.columns.values:
        if title == "Unnamed: 0":
            strategy2_content = strategy2_content.drop(columns = [title])
        elif strategy2_content[title].dtypes == "int64" or strategy2_content[title].dtypes == "float64":
            value_dict = self.get_feature_value(strategy2_content, title)
            filling_data = max(value_dict, key = value_dict.get)
            strategy2_content = strategy2_content.fillna({title:filling_data})
            self.draw_figure(strategy2_content, title, strategy_path)
    strategy2_content.to_csv(fp2)
```

3.2.3 通过属性的相关关系来填补缺失值

使用属性的相关关系来填补缺失值，考虑的是其他属性对于某特定属性的影响，因此我使用插值法来实现，使用 python 语言的 pandas 包的 interpolate()函数来填补缺失值，具体实现如下：

```
strategy_path = os.path.join(write_data_path, "strategy_3")
makedir(strategy_path)
with open(os.path.join(strategy_path, file_name), 'w', encoding = 'utf-8') as fp3:
    strategy3_content = content
    strategy3_content = strategy3_content.interpolate(kind='nearest')
    strategy3_content.to_csv(fp3)
    for title in content.columns.values:
        if title == "Unnamed: 0":
            continue
        elif strategy3_content[title].dtypes == "int64" or strategy3_content[title].dtypes == "float64":
            self.draw_figure(strategy3_content, title, strategy_path)
```

3.2.4 通过数据对象之间的相似性来填补缺失值

使用数据对象之间的相似性来填补缺失值，考虑的是数据对象之间的相关性，因此我使用 KNN 的方法来实现，不同于之前的方法，对于某个数据对象如果其

数值属性均为 nan 的话，我将使用平均值对其进行填补，其他的使用 python 语言的 sklearn 包的 KNeighborsClassifier 和 KNeighborsRegressor 函数来实现，具体实现如下：

```
strategy_path = os.path.join(write_data_path, "strategy_4")
makedir(strategy_path)
with open(os.path.join(strategy_path, file_name), 'w', encoding = 'utf-8') as fp4:
    strategy4_content = content
    nonan_content = pd.DataFrame()
    num_list = []
    for title in strategy4_content.columns.values:
        if title == "Unnamed: 0":
            strategy4_content = strategy4_content.drop(title,1)
        elif strategy4_content[title].dtypes == "int64" or strategy4_content[title].dtypes == "float64":
            num_list.append(title)
            nonan_content = pd.concat([nonan_content, strategy4_content[title]],axis=1)
    nonan_content.dropna(axis=0, how='any', inplace=True)
    mean_val = [ nonan_content[title].mean() for title in nonan_content.columns.values]

    if len([title for title in num_list if strategy4_content[title].isnull().any() == True]) == len(num_list):
        for i in range(len(strategy4_content)):
            if strategy4_content.loc[i][num_list].isnull().all():
                for j in range(len(num_list)):
                    strategy4_content.loc[i,num_list[j]] = mean_val[j]
                print(strategy4_content.loc[i])

    for title in num_list:
        if strategy4_content[title].isnull().any():
            train_y1 = nonan_content[title]
            train_x1 = nonan_content.loc[:, [other for other in num_list if other!=title]]
            test_x1 = strategy4_content[pd.isna(strategy4_content[title])].loc[:, [other for other in num_list if other!=title]]
            index, pred = self.knn_missing_filled(train_x1, train_y1, test_x1)
            strategy4_content.loc[index, title] = pred
        self.draw_figure(strategy4_content, title, strategy_path)
    strategy4_content.to_csv(fp4)
```

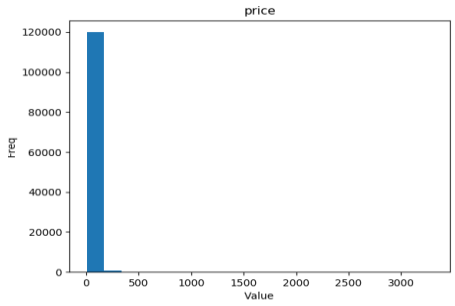
```
def knn_missing_filled(self, x_train, y_train, test, k = 3, dispersed = True):
    if dispersed:
        clf = KNeighborsClassifier(n_neighbors = k, weights = "distance")
    else:
        clf = KNeighborsRegressor(n_neighbors = k, weights = "distance")

    clf.fit(x_train, y_train)
    return test.index, clf.predict(test)
```

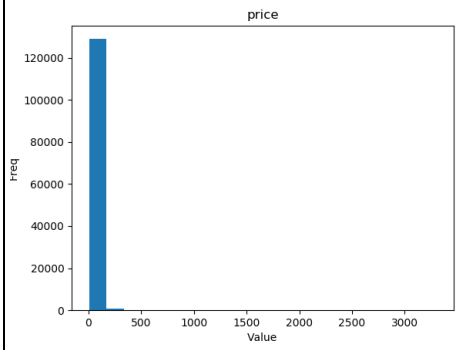
3.2.5 可视化对比

为经过填补的数据分别绘制直方图、qq 图和盒图，并将绘图结果与原数据进行对比。在 draw_figure()函数中分别调用 draw_histogram()、draw_qq()和 draw_box()函数绘制直方图、qq 图和盒图，绘图结果分别位于 strategy_*.csv(*=1,2,3,4)”目录下。

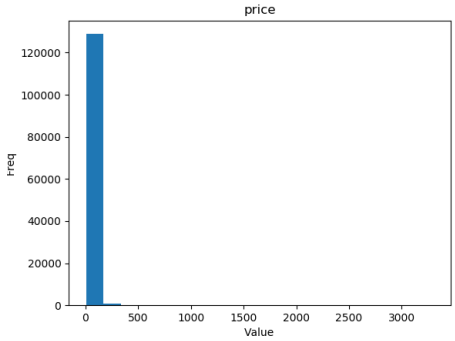
以数据集 Wine Reviews 的 winemag-data-130k-v2.csv 文件的’price’属性（含有 8996 条缺失值）举例，可视化对比结果为：



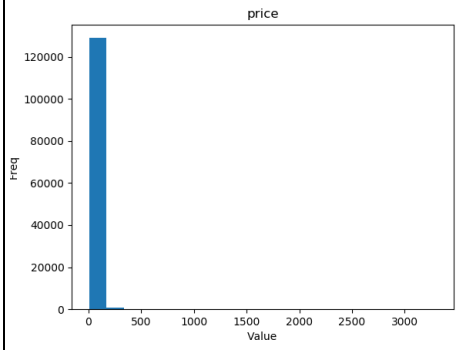
原始直方图、剔除缺失值直方图



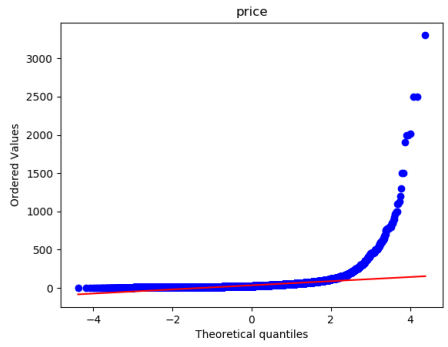
最高频率值填补直方图



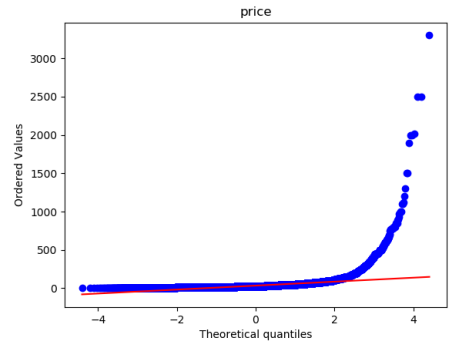
属性相关关系填补直方图



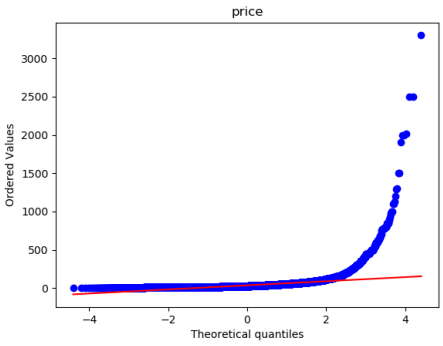
数据对象相似性填补直方图



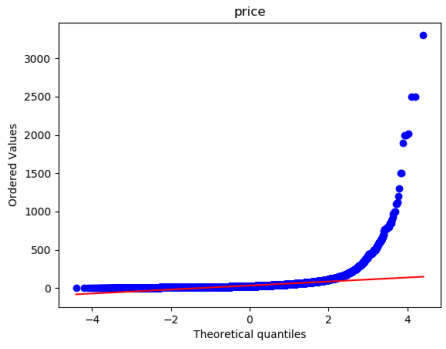
原始 qq 图、剔除缺失值 qq 图



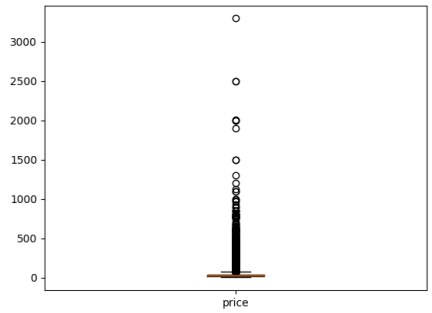
最高频率值填补 qq 图

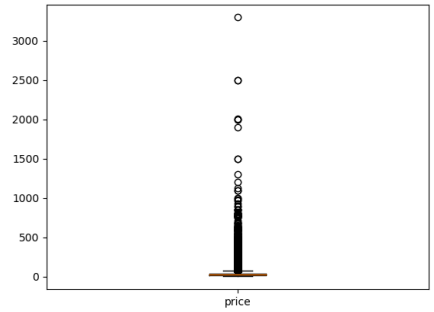


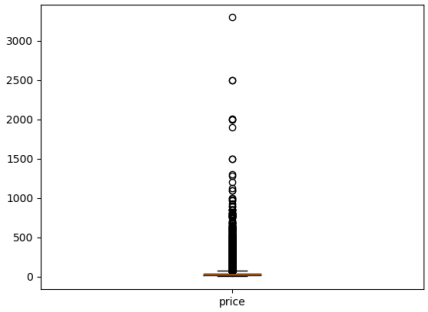
属性相关关系填补 qq 图

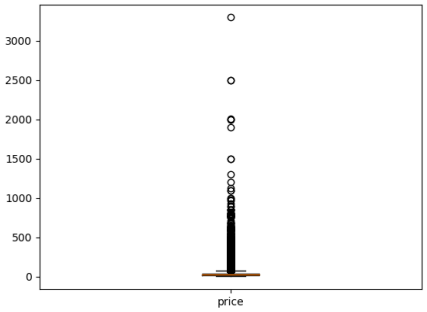


数据对象相似性填补 qq 图

<table>
<tr>
<td> 

原始盒图、剔除缺失值盒图</td>
<td> 

最高频率值填补盒图</td>
</tr>
<tr>
<td> 

属性相关关系填补盒图</td>
<td> 

数据对象相似性填补盒图</td>
</tr>
</table>