红酒评论数据集

1. 数据集加载

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
%matplotlib inline

path=['./dataset/wine_reviews/winemag-data-130k-v2.csv',
      './dataset/wine_reviews/winemag-data_first150k.csv']
data = pd.concat([pd.read_csv(p) for p in path])
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 280901 entries, 0 to 150929
Data columns (total 14 columns):
 #   Column                 Non-Null Count   Dtype  
---  ------                 --------------   -----  
 0   Unnamed: 0             280901 non-null  int64  
 1   country                280833 non-null  object 
 2   description            280901 non-null  object 
 3   designation            197701 non-null  object 
 4   points                 280901 non-null  int64  
 5   price                  258210 non-null  float64
 6   province               280833 non-null  object 
 7   region_1               234594 non-null  object 
 8   region_2               111464 non-null  object 
 9   taster_name            103727 non-null  object 
 10  taster_twitter_handle  98758 non-null   object 
 11  title                  129971 non-null  object 
 12  variety                280900 non-null  object 
 13  winery                 280901 non-null  object 
dtypes: float64(1), int64(2), object(11)
memory usage: 32.1+ MB
```

2. 数据集可视化可摘要

2.1 数据摘要和可视化

- 共13个属性(ID除外),其中:
- 标称属性:
    1. country 国家
    2. designation 葡萄园
    3. province 省份
    4. region_1 葡萄酒生产地
    5. region_2 详细生产地
    6. variety 葡萄酒种类
    7. winery 酿酒厂

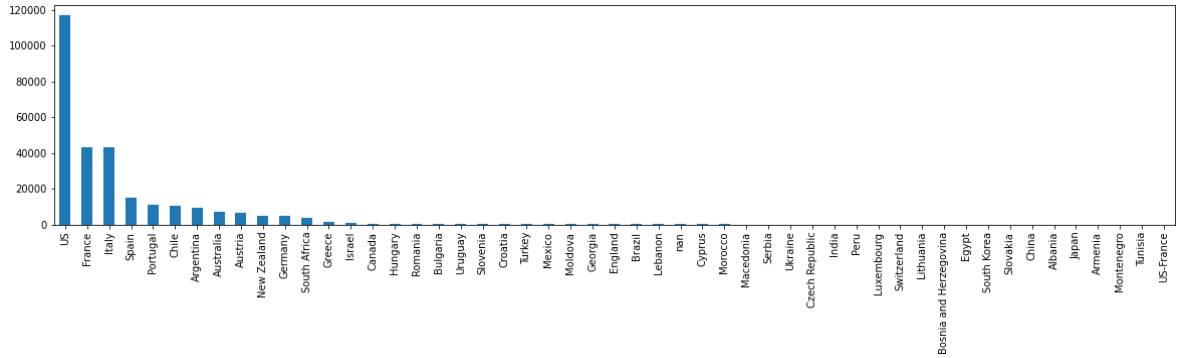
- 数值属性：
    8. points 评分
    9. price 价格
- 其他：
    10. title 评价标题
    11. description 评价内容
    12. taster_name 评价人
    13. taster_twitter_handle 评价人twitter

(1) country属性

```
print(data['country'].value_counts(dropna = False).head(10))
data['country'].value_counts(dropna = False).plot(kind="bar",figsize=(20,4))
```

```
US             116901
France          43191
Italy           43018
Spain           14913
Portugal        11013
Chile           10288
Argentina        9431
Australia        7286
Austria          6402
New Zealand      4739
Name: country, dtype: int64
```

```
<AxesSubplot:>
```

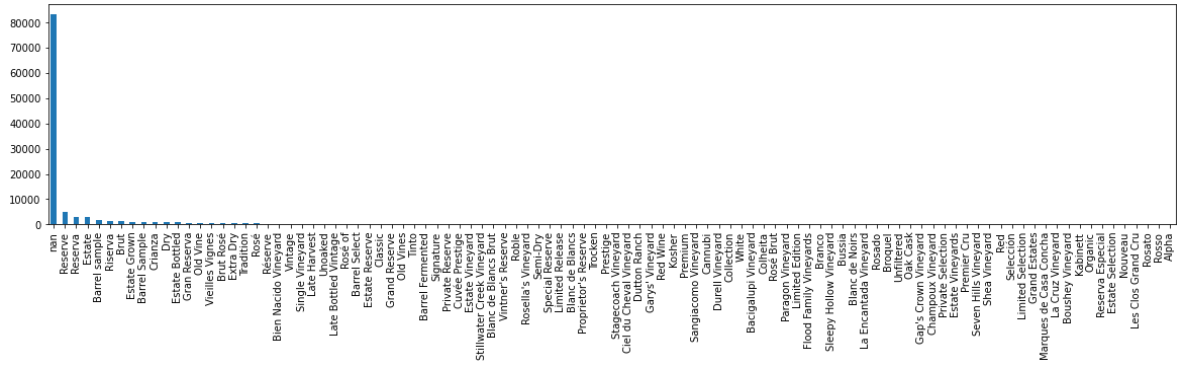


(2) designation 属性

```
print(data['designation'].value_counts(dropna = False).head(10))
data['designation'].value_counts(dropna = False)
[:100].plot(kind="bar",figsize=(20,4))
```

```
NaN                83200
Reserve             4761
Reserva             3069
Estate              2893
Barrel sample       1701
Riserva             1452
Brut                1137
Estate Grown        1070
Barrel Sample        891
Crianza              846
Name: designation, dtype: int64
```

```
<AxesSubplot:>
```

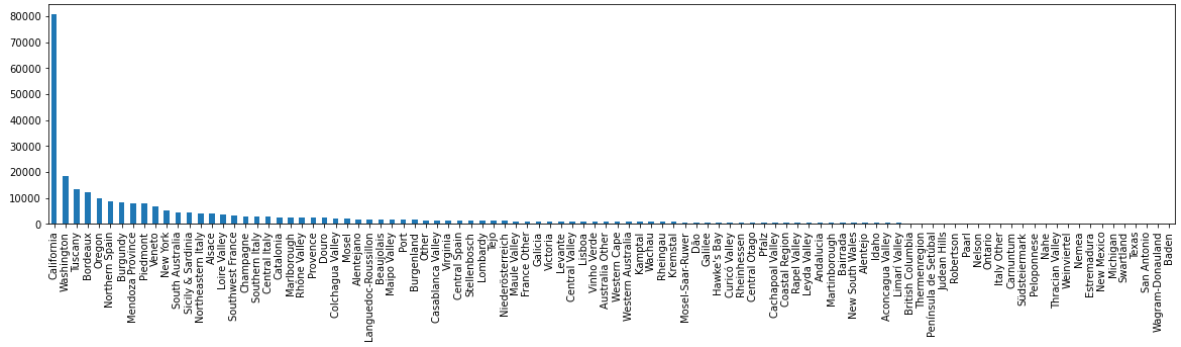


(3) province 属性

```
print(data['province'].value_counts(dropna = False).head(10))
data['province'].value_counts(dropna = False)[:100].plot(kind="bar",figsize=(20,4))
```

```
California          80755
Washington          18389
Tuscany             13178
Bordeaux            12052
Oregon               9962
Northern Spain       8743
Burgundy             8288
Mendoza Province     8006
Piedmont             7822
Veneto               6678
Name: province, dtype: int64
```

```
<AxesSubplot:>
```

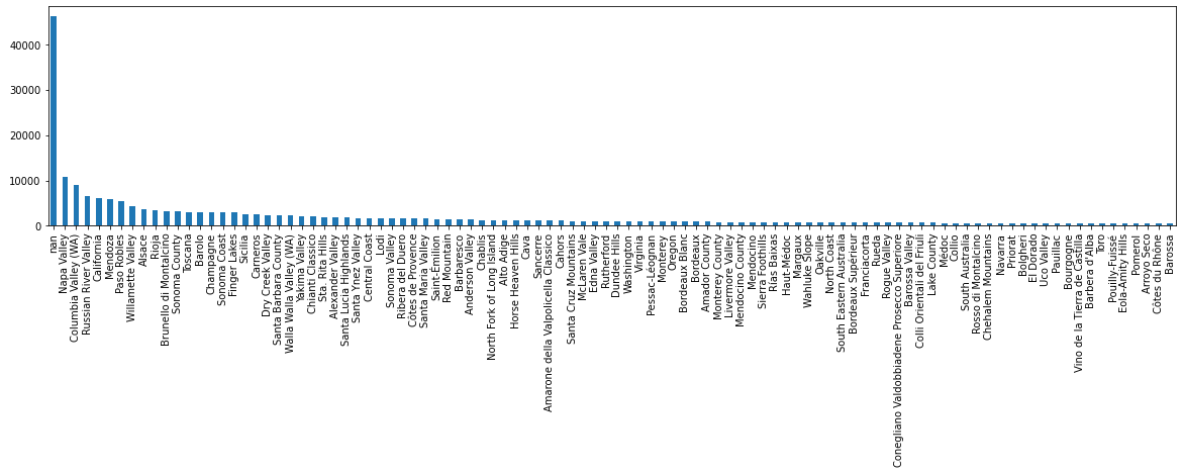


(4) region_1 属性

```
print(data['region_1'].value_counts(dropna = False).head(10))
data['region_1'].value_counts(dropna = False)[:100].plot(kind="bar",figsize=(20,4))
```

```
NaN                   46307
Napa Valley           10689
Columbia Valley (WA)   9099
Russian River Valley   6662
California             6091
Mendoza                5887
Paso Robles            5403
Willamette Valley      4397
Alsace                 3737
Rioja                  3362
Name: region_1, dtype: int64
```

```
<AxesSubplot:>
```

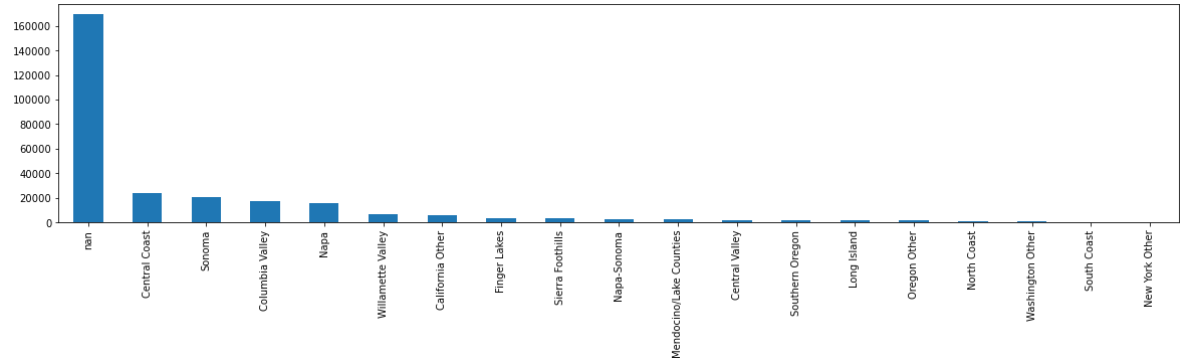


(5) region_2 属性

```
print(data['region_2'].value_counts(dropna = False).head(10))
data['region_2'].value_counts(dropna = False).plot(kind="bar",figsize=(20,4))
```

```
NaN                   169437
Central Coast          24122
Sonoma                 20286
Columbia Valley        17260
Napa                   15615
Willamette Valley       6604
California Other        6179
Finger Lakes            3287
Sierra Foothills        3122
Napa-Sonoma             2814
Name: region_2, dtype: int64
```

```
<AxesSubplot:>
```

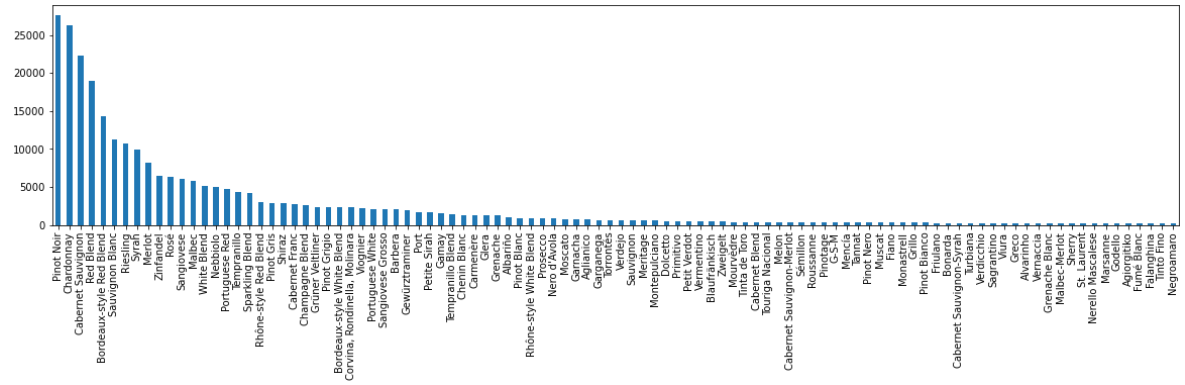


(6) variety 属性

```
print(data['variety'].value_counts(dropna = False).head(10))
data['variety'].value_counts(dropna = False)[:100].plot(kind="bar",figsize=
(20,4))
```

```
Pinot Noir                  27563
Chardonnay                  26235
Cabernet Sauvignon          22272
Red Blend                   19008
Bordeaux-style Red Blend    14262
Sauvignon Blanc             11287
Riesling                    10713
Syrah                        9967
Merlot                       8172
Zinfandel                    6513
Name: variety, dtype: int64
```

```
<AxesSubplot:>
```

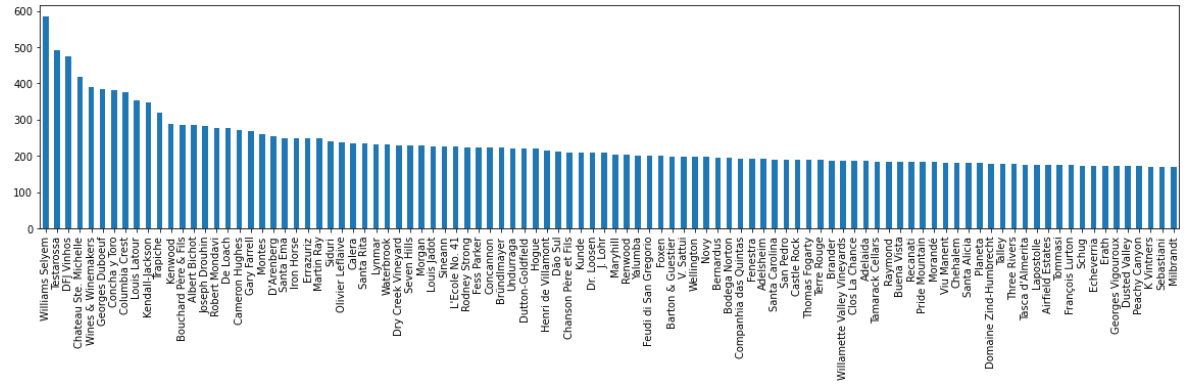


(7) winery 属性

```
print(data['winery'].value_counts(dropna = False).head(10))
data['winery'].value_counts(dropna = False)[:100].plot(kind="bar",figsize=(20,4))
```

```
Williams Selyem          585
Testarossa               492
DFJ Vinhos               473
Chateau Ste. Michelle    419
Wines & Winemakers       391
Georges Duboeuf          384
Concha y Toro            380
Columbia Crest           376
Louis Latour             353
Kendall-Jackson          346
Name: winery, dtype: int64
```

```
<AxesSubplot:>
```



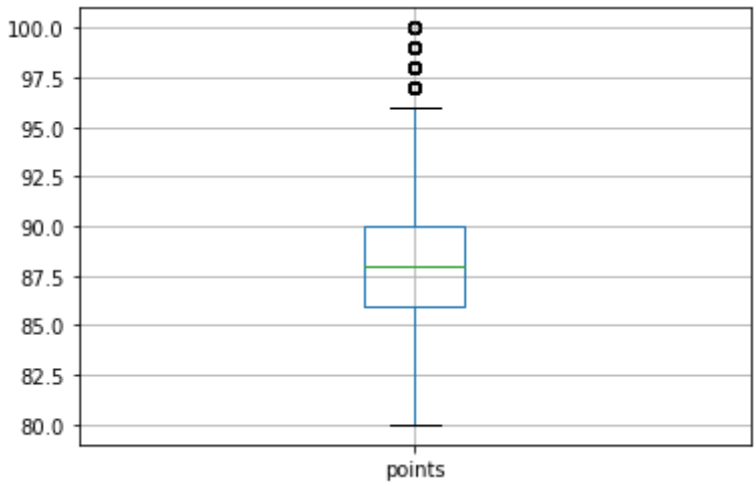
(8) points 属性

```
print('最小值',data['points'].min())
print('第一四分位数',data['points'].quantile(0.25))
print('中位数',data['points'].median())
print('第三四分位数',data['points'].quantile(0.75))
print('最大值',data['points'].max())
print('缺失值个数',data['points'].isnull().sum())
```

```
最小值 80
第一四分位数 86.0
中位数 88.0
第三四分位数 90.0
最大值 100
缺失值个数 0
```

```
data.boxplot(['points'],return_type='dict')
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x2b839845fc8>,
  <matplotlib.lines.Line2D at 0x2b8398614c8>],
 'caps': [<matplotlib.lines.Line2D at 0x2b839868908>,
  <matplotlib.lines.Line2D at 0x2b839868d48>],
 'boxes': [<matplotlib.lines.Line2D at 0x2b839861408>],
 'medians': [<matplotlib.lines.Line2D at 0x2b839868d88>],
 'fliers': [<matplotlib.lines.Line2D at 0x2b839868e48>],
 'means': []}
```




```
def detect_outliers(sr):
    q1 = sr.quantile(0.25)
    q3 = sr.quantile(0.75)
    iqr = q3-q1 #Interquartile range
    fence_low  = q1-1.5*iqr
    fence_high = q3+1.5*iqr
    outliers = sr.loc[(sr < fence_low) | (sr > fence_high)]
    return outliers
detect_outliers(data['points'])

```

```
345       100
346        98
347        97
348        97
349        97
          ...
143626     97
144494     97
144652     97
148047     97
149172     97
Name: points, Length: 928, dtype: int64
```

其中points>97的均视为离群点

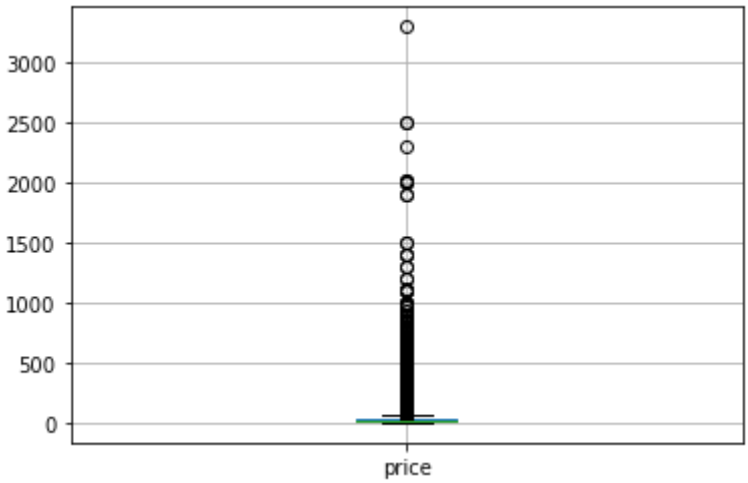
(9) price 属性

```
print('最小值',data['price'].min())
print('第一四分位数',data['price'].quantile(0.25))
print('中位数',data['price'].median())
print('第三四分位数',data['price'].quantile(0.75))
print('最大值',data['price'].max())
print('缺失值个数',data['price'].isnull().sum())
```

```
最小值 4.0
第一四分位数 16.0
中位数 25.0
第三四分位数 40.0
最大值 3300.0
缺失值个数 22691
```

```
data.boxplot(['price'],return_type='dict')
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x1c667b47d08>,
  <matplotlib.lines.Line2D at 0x1c667b36ac8>],
 'caps': [<matplotlib.lines.Line2D at 0x1c667b36a48>,
  <matplotlib.lines.Line2D at 0x1c667b341c8>],
 'boxes': [<matplotlib.lines.Line2D at 0x1c667b44388>],
 'medians': [<matplotlib.lines.Line2D at 0x1c667b34d08>],
 'fliers': [<matplotlib.lines.Line2D at 0x1c667b32308>],
 'means': []}
```




```
def detect_outliers(sr):
    q1 = sr.quantile(0.25)
    q3 = sr.quantile(0.75)
    iqr = q3-q1 #Interquartile range
    fence_low  = q1-1.5*iqr
    fence_high = q3+1.5*iqr
    outliers = sr.loc[(sr < fence_low) | (sr > fence_high)]
    return outliers
detect_outliers(data['price'])
```

```
60        100.0
111        85.0
118        80.0
119        80.0
134        78.0
          ...
150570     90.0
150613     80.0
150727     83.0
150762    100.0
150765     87.0
Name: price, Length: 15128, dtype: float64
```

3. 数据缺失的处理

统计所有属性的数据缺失情况：

```
print(data.isnull().sum(axis=0))
```

```
Unnamed: 0                0
country                  68
description               0
designation           83200
points                    0
```

```
price                    22691
province                    68
region_1                 46307
region_2                169437
taster_name             177174
taster_twitter_handle   182143
title                   150930
variety                      1
winery                       0
dtype: int64
```

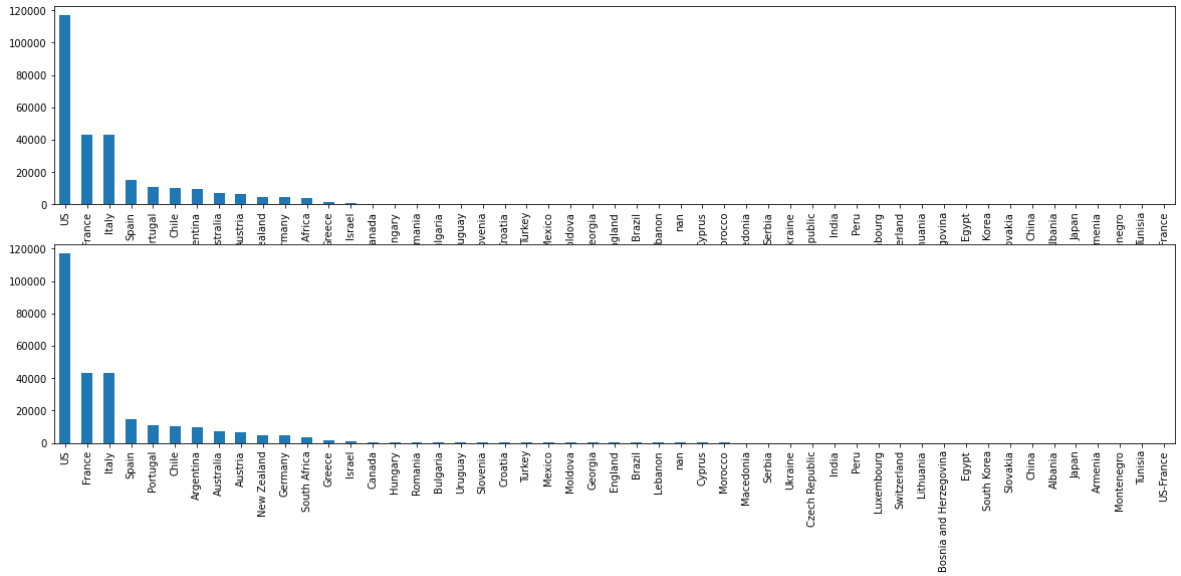
3.1 处理country属性缺失

缺失原因：统计失误，可以通过属性的相关关系来填补缺失值，查找country属性缺失的葡萄园所在国家并进行填充

```
import copy
designation2country = {
    "Askitikos":"Greece",
    "Shah":"US",
    "Piedra Feliz":"Chile",
}
data_country = copy.deepcopy(data)
for index,row in data_country.iterrows():
    if not row['country']:
        row['country'] = designation2country[row['designation']]
```

```
import matplotlib.pyplot as plt
plt.subplot(2,1,1)
data["country"].value_counts(dropna = False).plot(kind='bar',figsize=(20,8))
plt.subplot(2,1,2)
data_country["country"].value_counts(dropna = False).plot(kind='bar',figsize=
(20,8))
```

```
<AxesSubplot:>
```

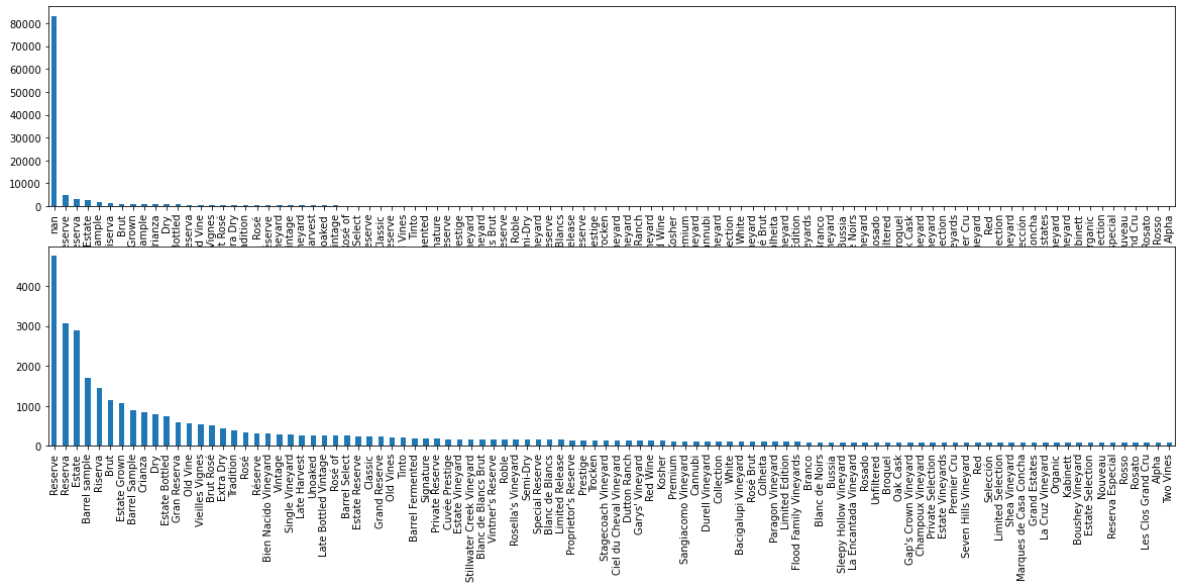


3.2 处理designation属性缺失

缺失原因：葡萄园可能较小或葡萄来源不明，将缺失部分剔除

```
data_designation = data.dropna(subset=['designation'])
plt.subplot(2,1,1)
data["designation"].value_counts(dropna = False)
[:100].plot(kind='bar',figsize=(20,8))
plt.subplot(2,1,2)
data_designation["designation"].value_counts(dropna = False)
[:100].plot(kind='bar',figsize=(20,8))
```

```
<AxesSubplot:>
```



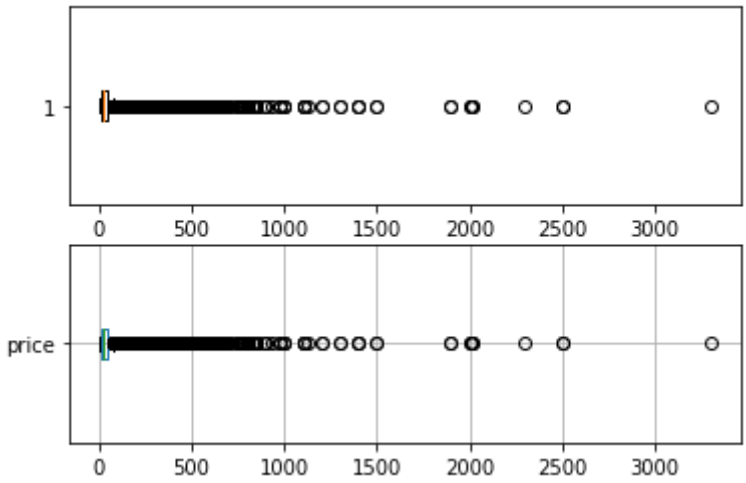
3.3 处理price属性缺失

缺失原因：通过数据平均值来填补缺失值

```
data_price= copy.deepcopy(data)
mean_price = data_price['price'].mean()
data_price = data_price['price'].fillna(mean_price)

plt.subplot(2,1,1)
plt.boxplot(data_price,vert=False)
plt.subplot(2,1,2)
data.boxplot(['price'],return_type='dict',vert=False)
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7feddd199940>,
  <matplotlib.lines.Line2D at 0x7feddd199c10>],
 'caps': [<matplotlib.lines.Line2D at 0x7feddd199ee0>,
  <matplotlib.lines.Line2D at 0x7feddd1371f0>],
 'boxes': [<matplotlib.lines.Line2D at 0x7feddd199670>],
 'medians': [<matplotlib.lines.Line2D at 0x7feddd1374c0>],
 'fliers': [<matplotlib.lines.Line2D at 0x7feddd137790>],
 'means': []}
```



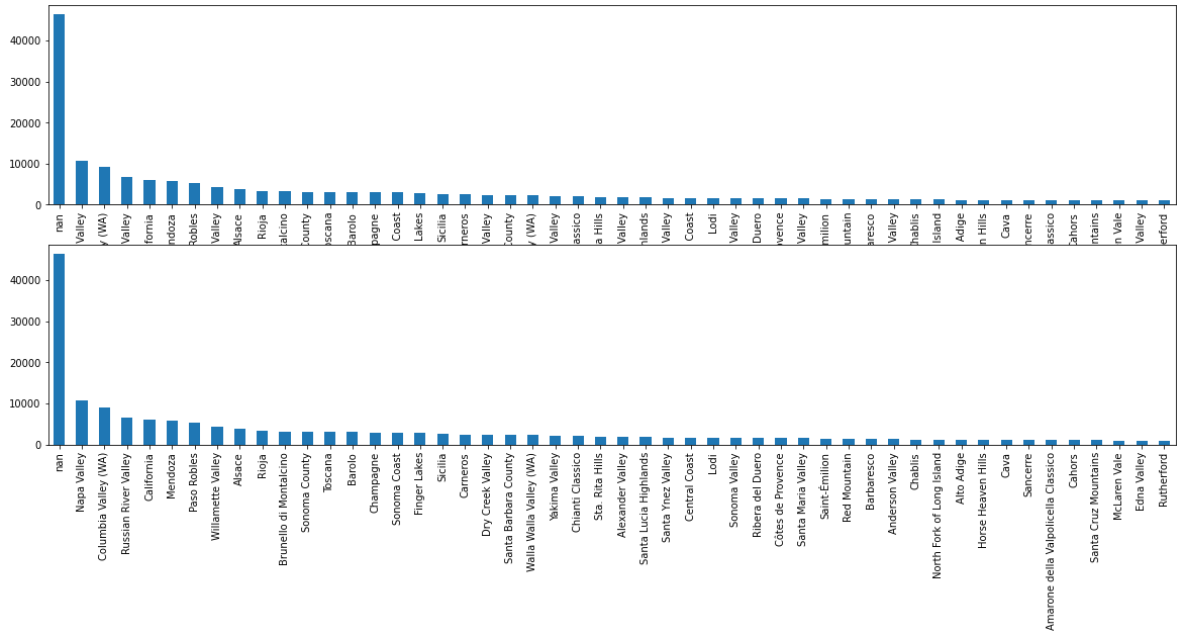
3.4 处理region_1属性缺失

缺失原因：通过相似对象的最高频率来填补缺失值

```
mode = data['region_1'].mode()
data_region1 = data['region_1'].fillna(mode)

plt.subplot(2,1,1)
data['region_1'].value_counts(dropna = False)[:50].plot(kind='bar',figsize=(20,8))
plt.subplot(2,1,2)
data_region1.value_counts(dropna = False)[:50].plot(kind='bar',figsize=(20,8))
```

```
<AxesSubplot:>
```



3.5 处理region_2属性缺失

缺失原因：将缺失部分剔除

```
data_region2 = data.dropna(subset=['region_2'])
plt.subplot(2,1,1)
data["region_2"].value_counts(dropna = False)[:50].plot(kind='bar',figsize=
(20,8))
plt.subplot(2,1,2)
data_region2["region_2"].value_counts(dropna = False)
[:50].plot(kind='bar',figsize=(20,8))
```

```
<AxesSubplot:>
```