自行选择两个数据集进行探索性分析

分析报告一

一、数据

1.1 数据集选择

Wine Reviews：winemag-data_first150k.csv
Wine Reviews：winemag-data-130k-v2.csv(用于对比数据缺失处理的原始数据集使用)

1.2 编程语言：python

1.3 导入所需各类依赖包

In [ ]:
```
import pandas as pd
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
```

二、数据分析要求

2.1 数据可视化及摘要

* 数据摘要

2.1.1 标称属性，给出每个可能取值的频数

该数据集中标称属性有：country、disignation、province、region_1、region_2、variety、winery

由于属性值较多，这里我们以country为例作展示，其余标称属性可能取值的频数运行代码后可查看

Name: country, dtype: int64

In [ ]:
```
wine = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv'))
print(wine['country'].value_counts())
```

country	频数	country	频数
US	62397	Lebanon	37
Italy	23478	Cyprus	31
France	21098	Brazil	25
Spain	8268	Macedonia	16
Chile	5816	Serbia	14
Argentina	5631	Morocco	12
Portugal	5322	England	9
Australia	4957	Luxembourg	9
New Zealand	3320	Lithuania	8
Austria	3057	India	8
Germany	2452	Czech Republic	6
South Africa	2258	Ukraine	5
Greece	884	Switzerland	4
Israel	630	Bosnia and Herzegovina	4
Hungary	231	South Korea	4
Canada	196	Egypt	3
Romania	139	China	3
Slovenia	94	Slovakia	3
Uruguay	92	Albania	2
Croatia	89	Montenegro	2
Bulgaria	77	Tunisia	2
Moldova	71	Japan	2
Mexico	63	US-France	1
Turkey	52		
Georgia	43		

2.1.2 数值属性，给出数值属性的五数概括及缺失值的个数

该数据集中数值属性有：price、points

Name: price, dtype: float64

In [ ]:
```
wine = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv'))
print(wine['price'].describe())
```

price	
count	137235.0000
mean	33.1315
std	36.3225
min	4.0000
25%	16.0000
50%	24.0000
75%	40.0000
max	2300.0000

Name: points, dtype: float64

In [ ]:
```
wine = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv'))
print(wine['points'].describe())
```

points	
count	150930.0000
mean	87.8884
std	3.2224
min	80.0000
25%	86.0000
50%	88.0000
75%	90.0000
max	100.0000

该数据集的缺省值情况为

In [ ]:
```
print(wine.isna().sum())
```

	缺失值
country	5
description	0
designation	45735
points	0
price	13695
province	5
region_1	25060
region_2	89977
variety	0
winery	0

* 数据可视化

2.1.3 使用直方图、盒图等检查数据分布及离群点

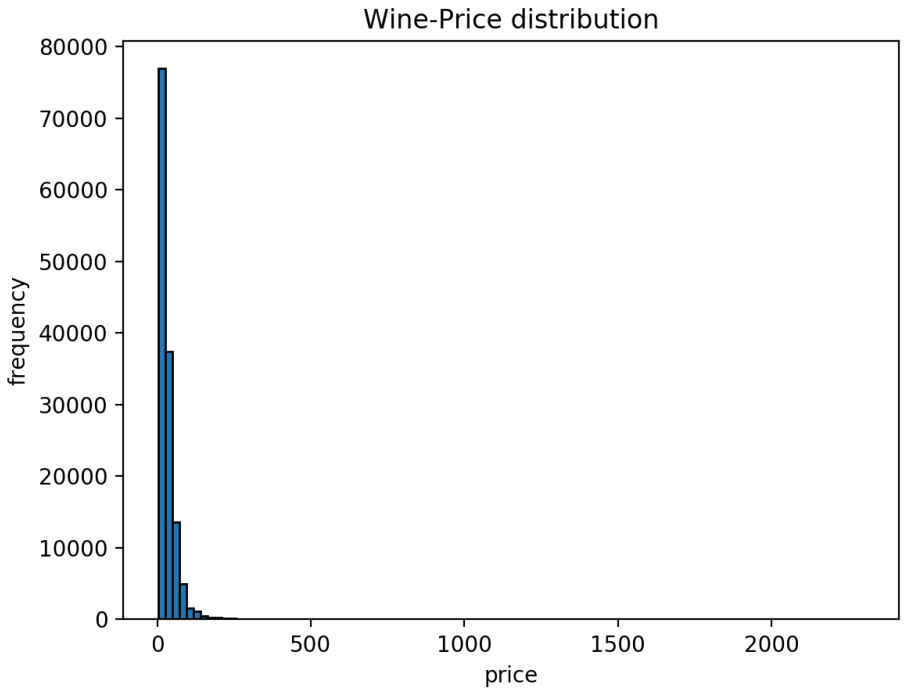
（这里给出price和points属性的可视化展示）

(1)、 **price属性直方图**

In [ ]:
```
#price属性直方图
plt.hist(x=wine['price'], bins=100, edgecolor='black')
# 添加x轴和y轴标签
plt.xlabel('price')
plt.ylabel('frequency')
# 添加标题
plt.title('Wine-Price distribution')
plt.savefig('./wineResult/price_distribution_hist.png')
plt.show()
```

In [11]:
```
from IPython.display import Image
Image(filename = 'price_distribution_hist.png', width=500, height=500)
```

Out[11]:

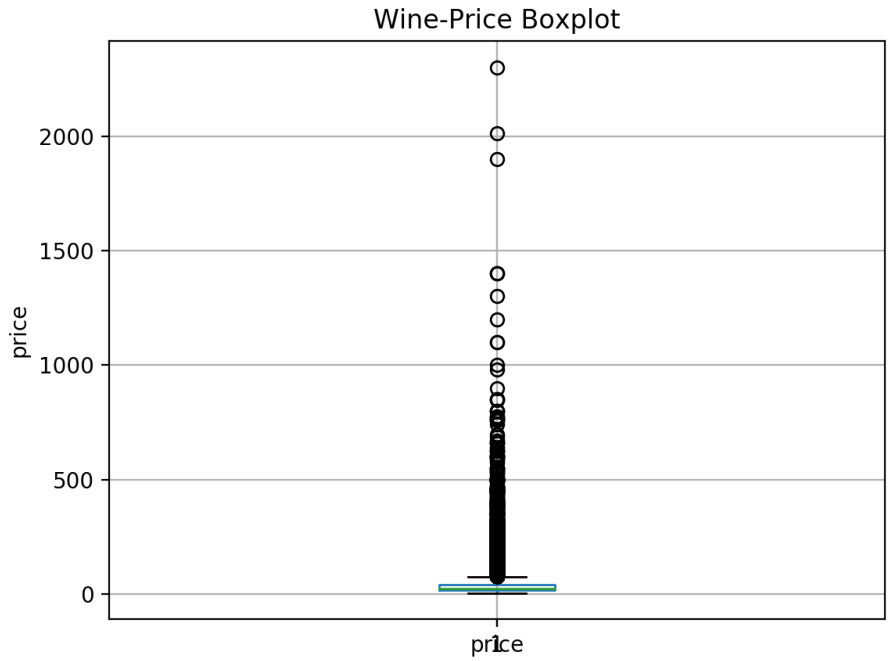


price属性盒图

In [ ]:
```
#price属性盒图(不丢弃缺失值情况)
priceNa = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv').price)
priceNa.boxplot(sym='o')
plt.boxplot(wine['price'], sym='o')
plt.ylabel('price')
plt.title('Wine-Price Boxplot')
#plt.legend()
plt.savefig('./wineResult/price_box.png')
plt.show()
```

In [8]:
```
from IPython.display import Image
Image(filename = 'price_box.png', width=500, height=500)
```

Out[8]:

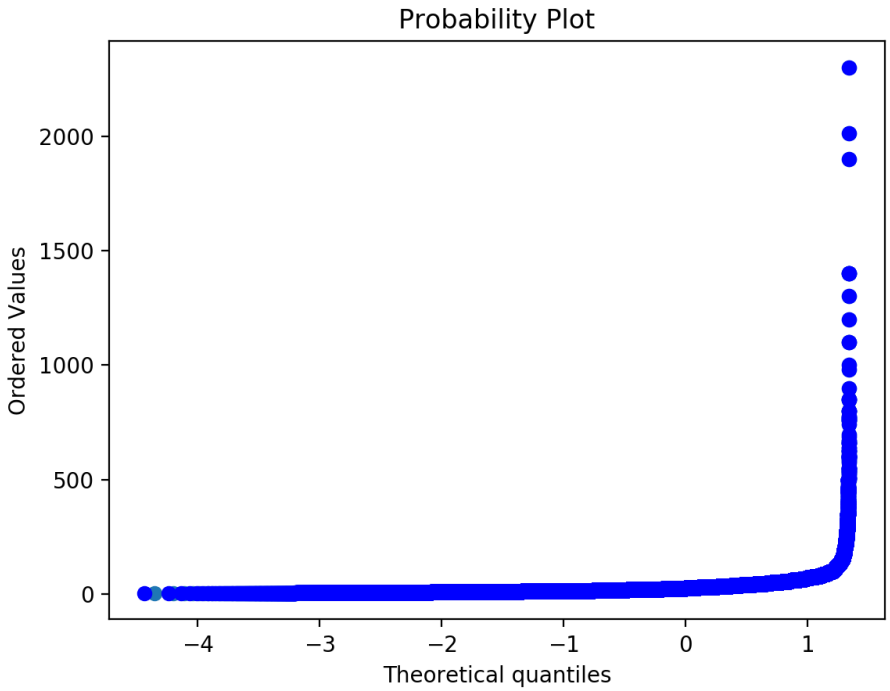


price属性Q-Q图

In [ ]:
```
#price属性QQ图(不丢弃缺失值)
sorted_ = np.sort(wine['price'])
yvals = np.arange(len(sorted_))/float(len(sorted_))
x_label = stats.norm.ppf(yvals)
plt.scatter(x_label, sorted_)
stats.probplot(wine['price'], dist="norm", plot=plt)
plt.savefig('./wineResult/price_qq.png')
plt.show()
```

In [7]:
```
from IPython.display import Image
Image(filename = 'price_qq.png', width=500, height=500)
```

Out[7]:

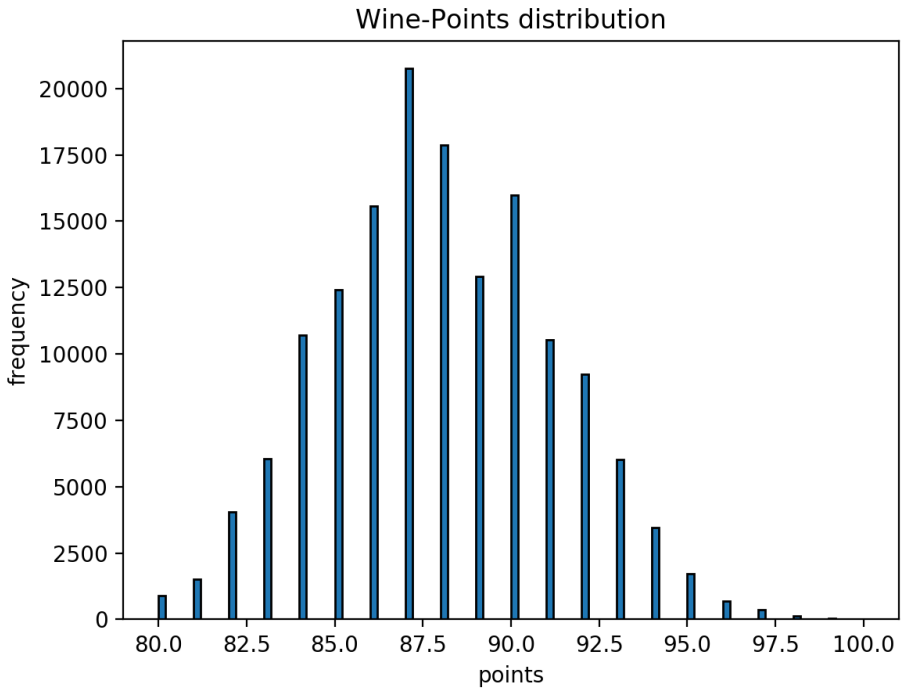


(2)、 **points属性直方图**

In [ ]:
```
#points属性直方图
plt.hist(x=wine['points'], bins=100, edgecolor='black')
# 添加x轴和y轴标签
plt.xlabel('points')
plt.ylabel('frequency')
# 添加标题
plt.title('Wine-Points distribution')
plt.savefig('./wineResult/points_distribution_hist.png')
plt.show()
```

In [6]:
```
from IPython.display import Image
Image(filename = 'points_distribution_hist.png', width=500, height=500)
```

Out[6]:

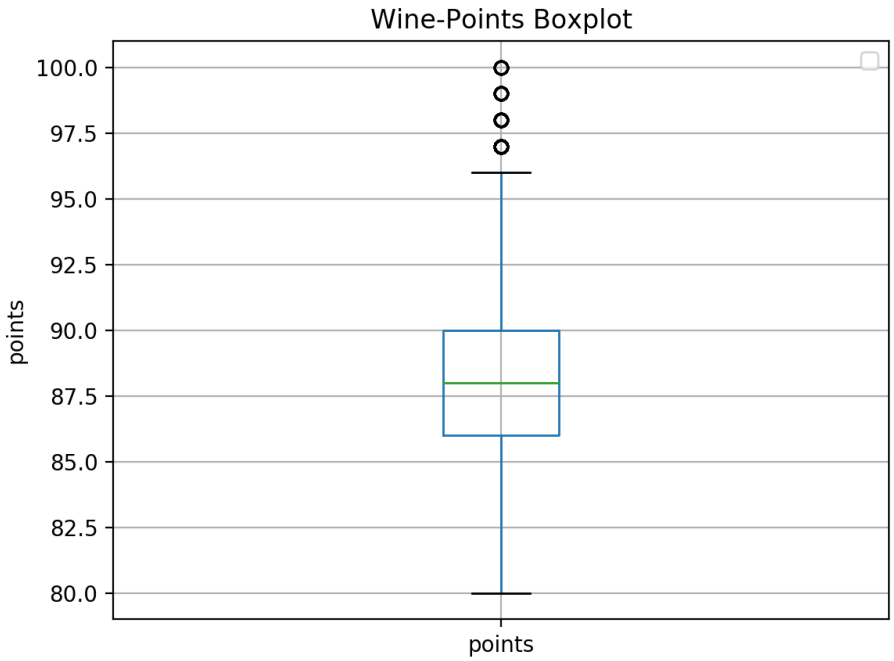


points属性盒图

In [ ]:
```
#points属性盒图(不丢弃缺失值情况)
priceNa = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv').points)
priceNa.boxplot(sym='o')
plt.ylabel('points')
plt.title('Wine-Points Boxplot')
plt.savefig('./wineResult/points_box.png')
plt.show()
```

In [5]:
```
from IPython.display import Image
Image(filename = 'points_box.png', width=500, height=500)
```

Out[5]:

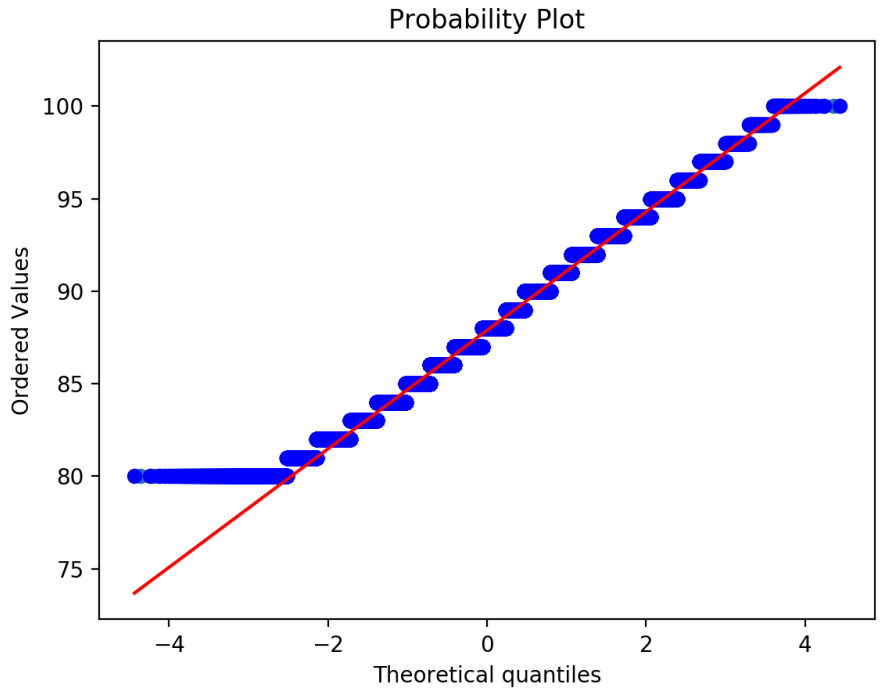


In [ ]:
```
#points属性QQ图(不丢弃缺失值)
sorted_ = np.sort(wine['points'])
yvals = np.arange(len(sorted_))/float(len(sorted_))
x_label = stats.norm.ppf(yvals)
plt.scatter(x_label, sorted_)
stats.probplot(wine['points'], dist="norm", plot=plt)
plt.savefig('./wineResult/points_qq.png')
plt.show()
```

points属性Q-Q图

In [4]:
```
from IPython.display import Image
Image(filename = 'points_qq.png', width=500, height=500)
```

Out[4]:



2.2 数据缺失的处理

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:

由于属性值较多，这里我们以price数值属性为例，price属性值缺失的原因可能为：红酒数据收集是数据缺失

2.2.1 将缺失部分剔除(这里直接展示剔除缺失值之后与原数据集的对比可视化)

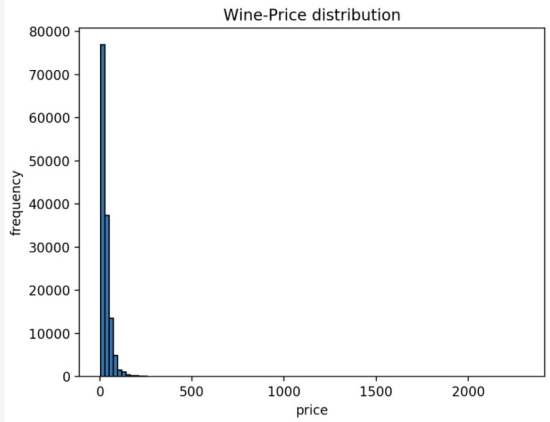
price直方图 (左为丢弃数据后直方图，右为原始数据直方图)

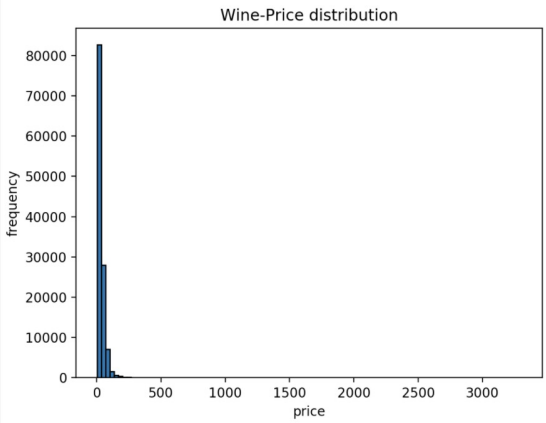
In [ ]:
```
#原始数据集（去重处理后）
wineV2 = pd.DataFrame(pd.read_csv('winemag-data-130k-v2.csv'))

#删除
#直方图
plt.hist(wine['price'].dropna(), bins=100, edgecolor='black')
# 添加x轴和y轴标签
plt.xlabel('price')
plt.ylabel('frequency')
# 添加标题
plt.title('Wine-Price distribution')
plt.savefig('./wineResult/price_delete_hist.png')
plt.show()
#原始直方图
plt.hist(wineV2['price'], bins=100, edgecolor='black')
# 添加x轴和y轴标签
plt.xlabel('price')
plt.ylabel('frequency')
# 添加标题
plt.title('Wine-Price distribution')
plt.savefig('./wineResult/priceCom_hist.png')
plt.show()
```

In [3]:
```
from IPython.display import Image
Image(filename = 'price_hist.png')
```

Out[3]:





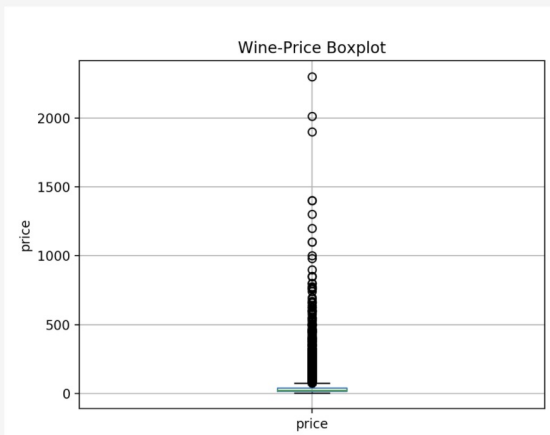
price盒图 (左为丢弃数据后盒图，右为原始数据盒图)

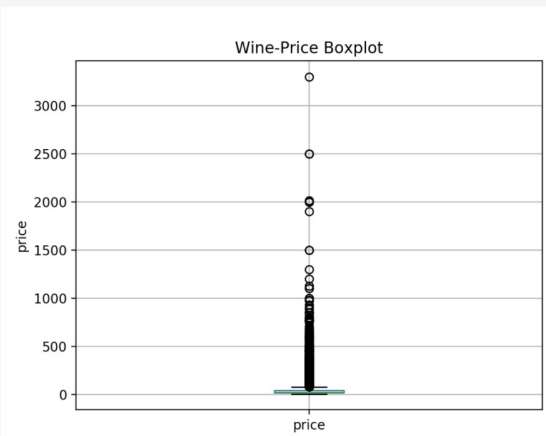
In [ ]:
```
#盒图
priceNa = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv').price).dr
opna()
priceNa.boxplot(sym='o')
plt.ylabel('price')
plt.title('Wine-Price Boxplot')
plt.savefig('./wineResult/price_delete_box.png')
plt.show()

#原始数据盒图
priceNa = pd.DataFrame(pd.read_csv('winemag-data-130k-v2.csv').price)
priceNa.boxplot(sym='o')
plt.ylabel('price')
plt.title('Wine-Price Boxplot')
plt.savefig('./wineResult/priceCom_box.png')
plt.show()
```

In [4]:
```
from IPython.display import Image
Image(filename = 'price_box2.png')
```

Out[4]:



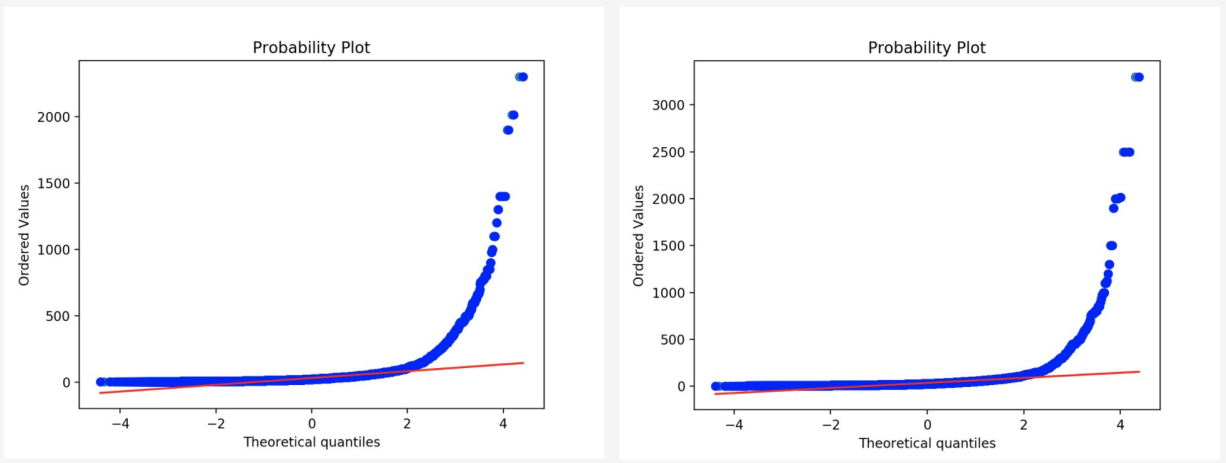


priceQ-Q图 (左为丢弃数据后Q-Q图，右为原始数据Q-Q图)

In [ ]:
```
#Q-Q图
sorted_ = np.sort(wine['price'].dropna())
yvals = np.arange(len(sorted_))/float(len(sorted_))
x_label = stats.norm.ppf(yvals)
plt.scatter(x_label, sorted_)
stats.probplot(wine['price'].dropna(), dist="norm", plot=plt)
plt.savefig('./wineResult/price_delete_qq.png')
plt.show()
#原始数据Q-Q图
sorted_ = np.sort(wineV2['price'].dropna())
yvals = np.arange(len(sorted_))/float(len(sorted_))
x_label = stats.norm.ppf(yvals)
plt.scatter(x_label, sorted_)
stats.probplot(wineV2['price'].dropna(), dist="norm", plot=plt)
plt.savefig('./wineResult/priceCom_qq.png')
plt.show()
```

In [5]:
```
from IPython.display import Image
Image(filename = 'price_delete_qq.png')
```

Out[5]:



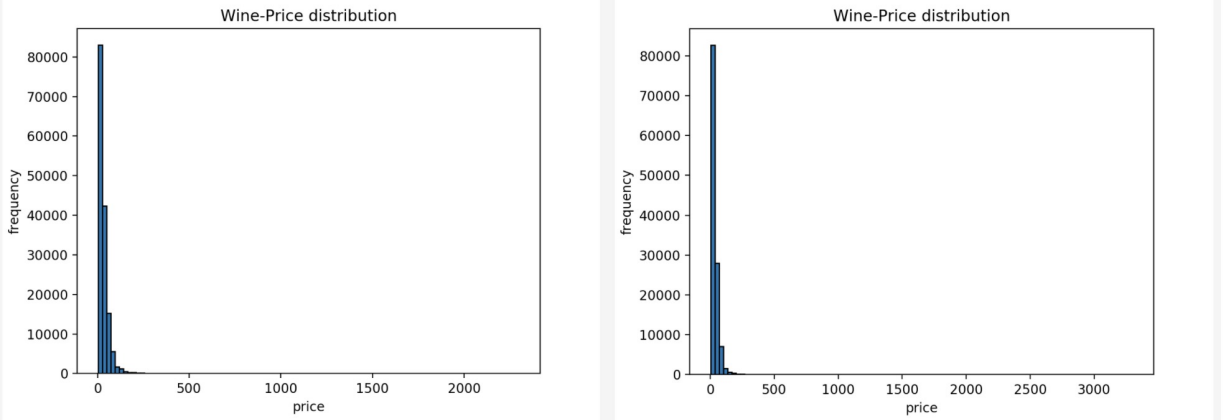
2.2.2 用最高频率值来填补缺失值

price直方图 (左为利用众数填充缺失值后直方图，右为原始数据直方图)

In [ ]:
```
plt.hist(wine['price'].fillna(wine['price'].interpolate(missing_values='NaN', strategy='mode', axis=0, verbose=0, copy=True)),
         bins=100, edgecolor='black')
# 添加x轴和y轴标签
plt.xlabel('price')
plt.ylabel('frequency')
# 添加标题
plt.title('Wine-Price distribution')
plt.savefig('./wineResult/price_mode_hist.png')
plt.show()
```

In [10]:
```
from IPython.display import Image
Image(filename = 'price_mode_hist.png')
```

Out[10]:

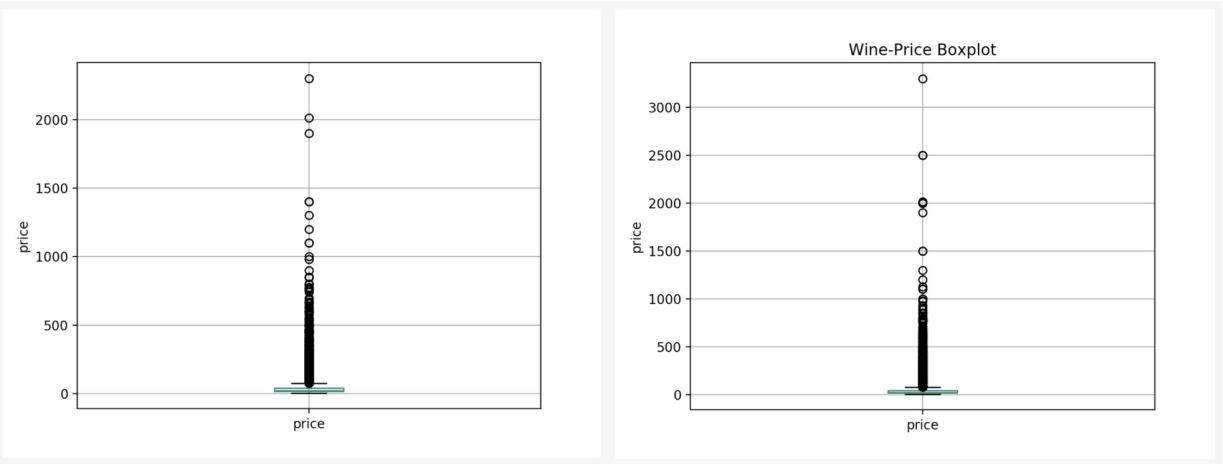


price盒图 (左为利用众数填充缺失值后盒图，右为原始数据盒图)

In [ ]:
```
#盒图
priceNa = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv').price).fi
llna(wine['price'].interpolate(missing_values='NaN', strategy='mode',
                                                    axis=0, verbose
=0, copy=True))
priceNa.boxplot(sym='o')
plt.ylabel('price')
plt.savefig('./wineResult/price_mode_box.png')
plt.show()
```

In [11]:
```
from IPython.display import Image, display, HTML
Image(filename = 'price_mode_box.png')
```

Out[11]:

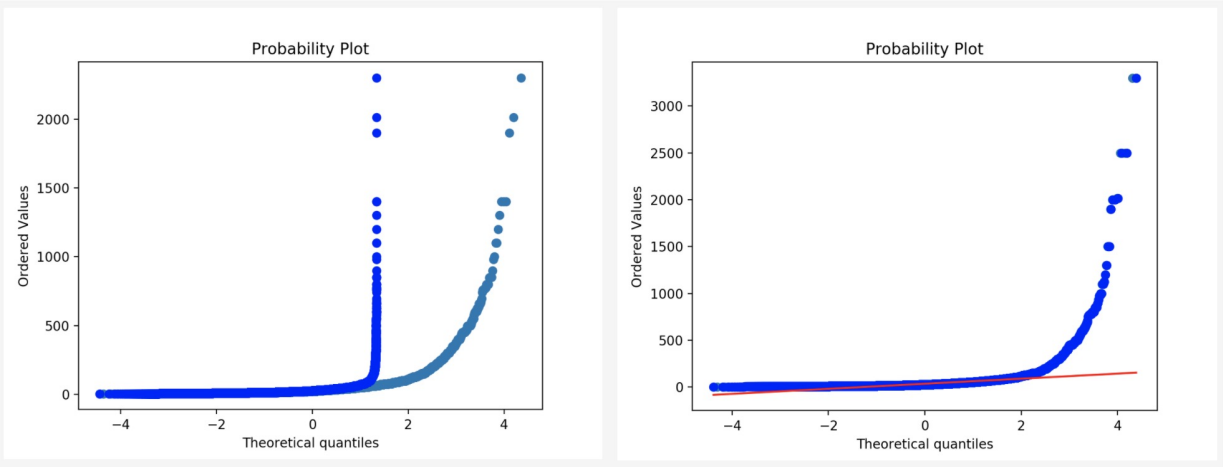


priceQ-Q图 (左为利用众数填充缺失值后Q-Q图，右为原始数据Q-Q图)

In [ ]:
```
#Q-Q图
sorted_ = np.sort(wine['price'].fillna(wine['price'].interpolate(missing_v
alues='NaN', strategy='mode', axis=0, verbose=0, copy=True)))
yvals = np.arange(len(sorted_))/float(len(sorted_))
x_label = stats.norm.ppf(yvals)
plt.scatter(x_label, sorted_)
stats.probplot(wine['price'], dist="norm", plot=plt)
plt.savefig('./wineResult/price_mode_qq.png')
plt.show()
```

In [13]:
```
from IPython.display import Image
Image(filename = 'price_mode_qq.png')
```

Out[13]:



2.2.3 通过属性的相关关系来填补缺失值

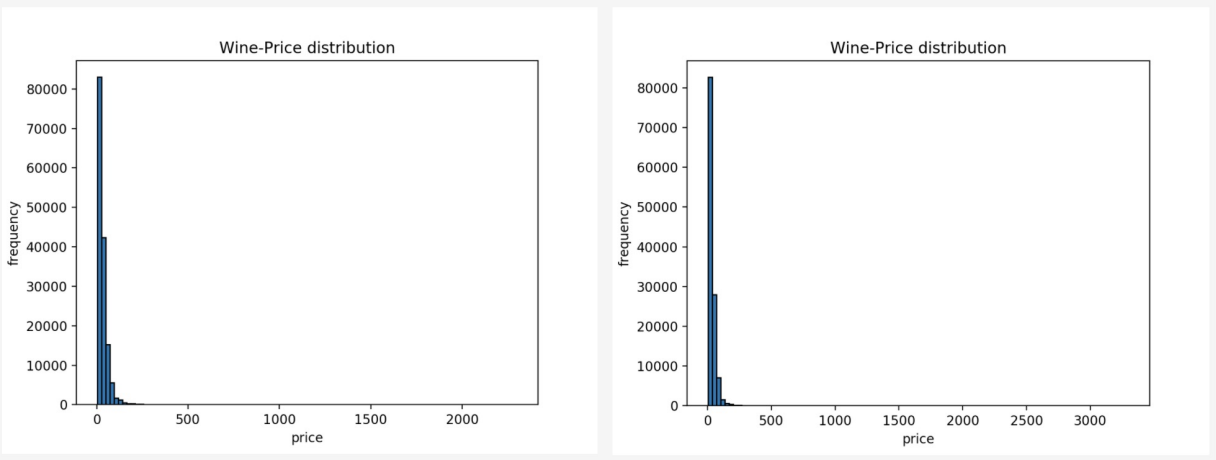
这部分缺失值本来打算用KNN（K近邻算法）来实现，即距离越近关系越好，但是代码实现有点问题就暂时用了中位数插值

price直方图 (左为利用中位数填充缺失值后直方图，右为原始数据直方图)

In [ ]:
```
#通过属性的相关关系来填补缺失值
wine = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv'))
#直方图
plt.hist(wine['price'].interpolate(missing_values='NaN', strategy='median'
, axis=0, verbose=0, copy=True),
         bins=100, edgecolor='black')
# 添加x轴和y轴标签
plt.xlabel('price')
plt.ylabel('frequency')
# 添加标题
plt.title('Wine-Price distribution')
plt.savefig('./wineResult/price_means_hist.png')
plt.show()
```

In [15]:
```
from IPython.display import Image
Image(filename = 'price_median_hist.png')
```

Out[15]:



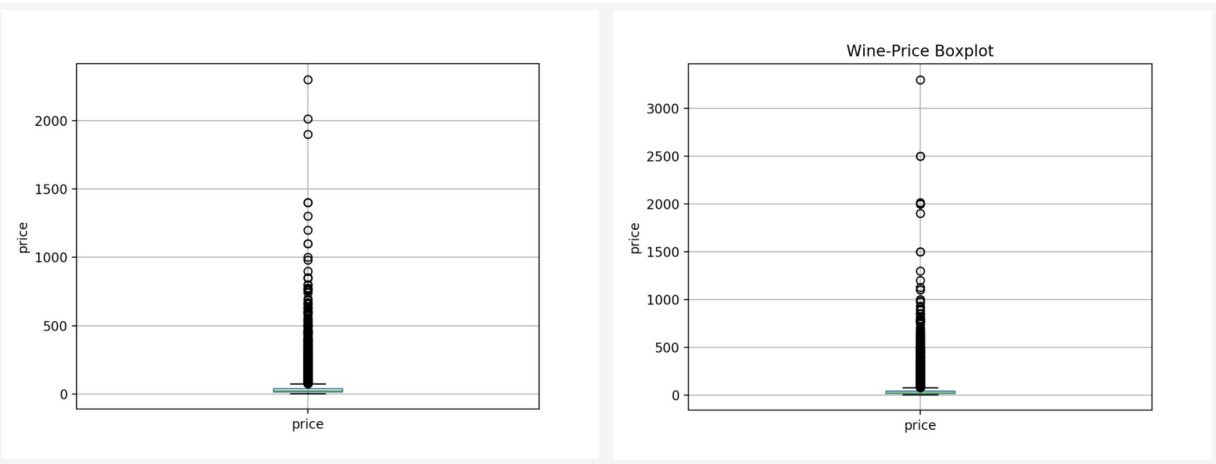
price盒图 (左为利用中位数填充缺失值后盒图，右为原始数据盒图)

In [ ]:
```
#盒图
priceNa = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv').price).fi
llna(wine['price'].interpolate(missing_values='NaN', strategy='median',

axis=0, verbose=0, copy=True))
priceNa.boxplot(sym='o')
plt.ylabel('price')
plt.savefig('./wineResult/price_median_box.png')
plt.show()
```

In [17]:
```
from IPython.display import Image
Image(filename = 'price_median_box.png')
```

Out[17]:

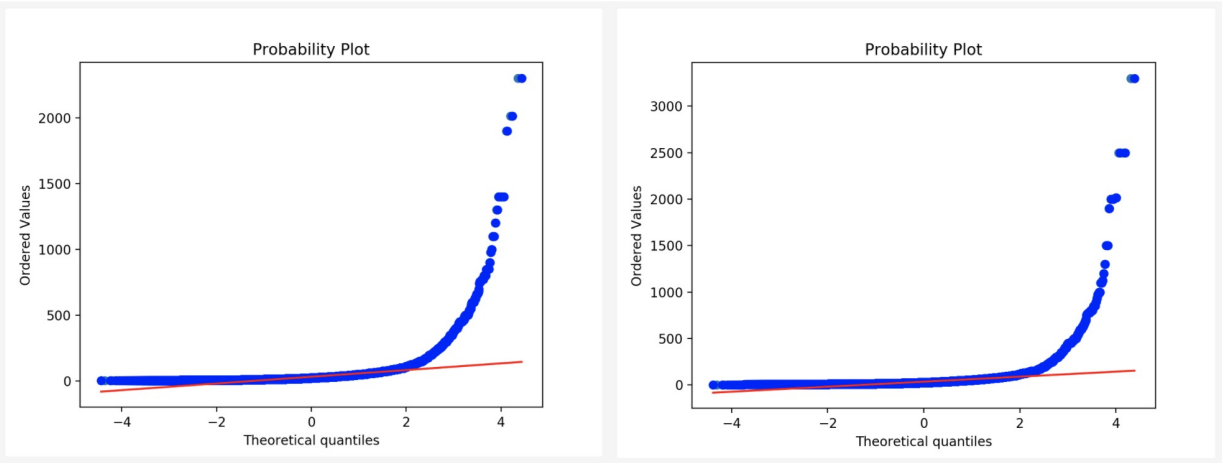


priceQ-Q图 (左为利用中位数填充缺失值后Q-Q图，右为原始数据Q-Q图)

In [ ]:
```
#Q-Q图
sorted_ = np.sort(wine['price'].interpolate(missing_values='NaN', strategy
='median', axis=0, verbose=0, copy=True))
yvals = np.arange(len(sorted_))/float(len(sorted_))
x_label = stats.norm.ppf(yvals)
plt.scatter(x_label, sorted_)
stats.probplot(wine['price'].interpolate(missing_values = 'NaN', strategy=
'median', axis=0, verbose=0, copy=True), dist="norm", plot=plt)
plt.savefig('./wineResult/price_median_qq.png')
plt.show()
```

In [20]:
```
from IPython.display import Image
Image(filename = 'price_median_qq.png')
```

Out[20]:



2.2.4 通过数据对象之间的相似性来填补缺失值

利用随机森林预测值来填充缺失值

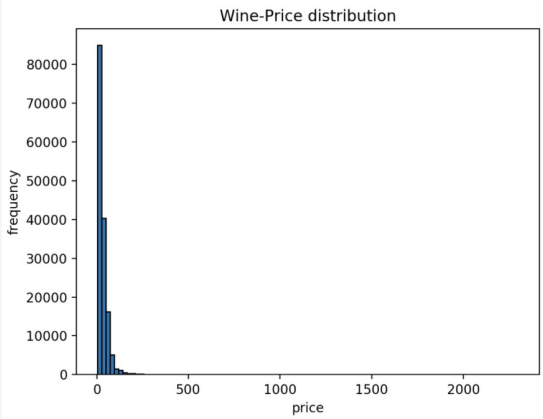
price直方图 (左为利用随机森林预测值填充缺失值后直方图，右为原始数据直方图)

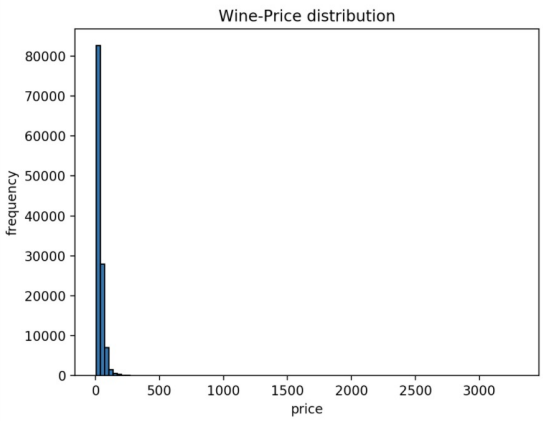
In [ ]:
```
#通过数据对象之间的相似性来填补缺失值
wine = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv'))
known_price = wine[wine['price'].notnull()]
unknown_price = wine[wine['price'].isnull()]
x = known_price[['points']]
y = known_price[['price']]
t_x = unknown_price[['points']]
fc = RandomForestClassifier()
fc.fit(x, y.values.ravel())
pr = fc.predict(t_x)
wine.loc[wine.price.isnull(), 'price'] = pr

#直方图
plt.hist(wine['price'], bins=100, edgecolor='black')
# 添加x轴和y轴标签
plt.xlabel('price')
plt.ylabel('frequency')
# 添加标题
plt.title('Wine-Price distribution')
plt.savefig('./wineResult/price_relative_hist.png')
plt.show()
```

In [22]:
```
from IPython.display import Image
Image(filename = 'price_relative_hist.png')
```

Out[22]:



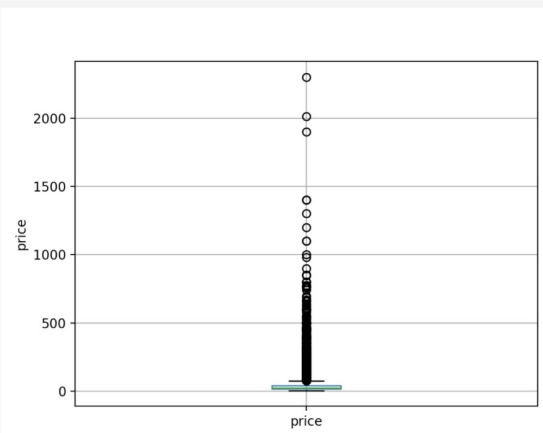


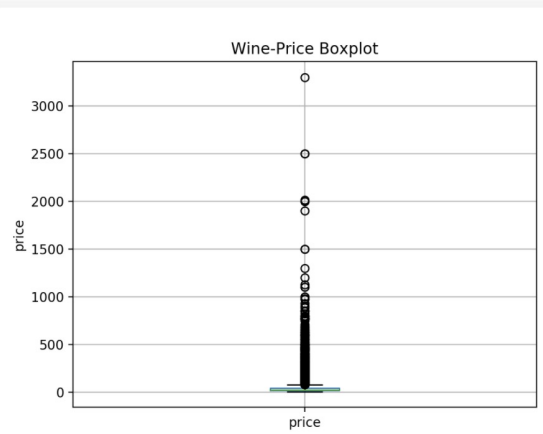
price盒图 (左为利用随机森林预测值填充缺失值后盒图，右为原始数据盒图)

In [ ]:
```
#盒图
priceNa = pd.DataFrame(pd.read_csv('winemag-data_first150k.csv').price)
priceNa.boxplot(sym='o')
plt.ylabel('price')
plt.savefig('./wineResult/price_relative_box.png')
plt.show()
```

In [24]:
```
from IPython.display import Image
Image(filename = 'price_relative_box.png')
```

Out[24]:





priceQ-Q图 (左为利用随机森林预测值填充缺失值后Q-Q图，右为原始数据Q-Q图)

In [ ]:
```
#Q-Q图
sorted_ = np.sort(wine['price'])
yvals = np.arange(len(sorted_))/float(len(sorted_))
x_label = stats.norm.ppf(yvals)
plt.scatter(x_label, sorted_)
stats.probplot(wine['price'])
plt.savefig('./wineResult/price_relative_qq.png')
plt.show()
```

In [26]:
```
from IPython.display import Image
Image(filename = 'price_relative_qq.png')
```

Out[26]: