Exploratory Data Analysis (EDA) on a Public Dataset

DESIGN DOCUMENT

By Michael Butler

Wine Reviews

130k wine reviews with variety, location, winery, price, and description

Overview

Skills Learned: Data cleaning, visualization, basic statistics
Objective: Take a public dataset (e.g., from Kaggle or UCI Machine Learning Repository) and perform exploratory data analytics.
Steps:

1. Download a dataset (e.g., Titanic, Iris, or any other simple dataset)
2. Clean the data by handling missing values, duplicates, and formatting issues
3. Perform basic descriptive statistics (mean, median, standard deviation)
4. Create visualizations (scatter plots, bar charts, histograms, box plots)
5. Summarize key insights from the data

Tools: Python (Pandas, Matplotlib, Seaborn)

Introduction

Objective

The goal of this project was to explore and understand the basics of data science as well as learn fundamental data analytical tools in python. Since these tools and data analysis are relatively new, this project was created to get familiar with the concepts of data analysis as well as introduction to new python tools. This project will cover pandas, matplotlib, and seaborn to gather and analyze data.

Dataset Description

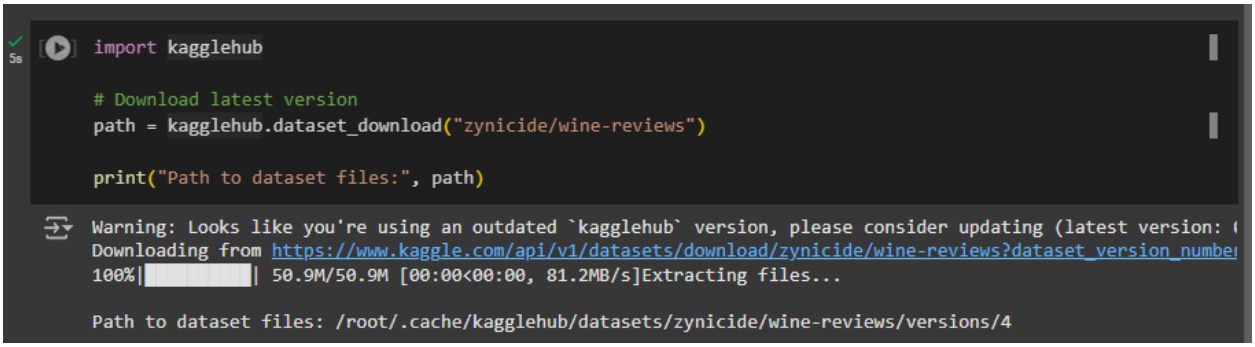
For this project a free dataset from Kaggle was used. This dataset consisted of wine reviews. The data contained information such as a rating, price, province, which was used for the majority of this project. Other fields contained country, description, regions 1 and 2, tasters name and twitter account, title of the wine, variety, and winery.

Source: https://www.kaggle.com/datasets/zynicide/wine-reviews?select=winemag-data-130k-v2.csv

Data Preprocessing and Cleaning

Data Loading

The dataset was downloaded with the use of **kagglehub**. From the download section of the website, copied and pasted to the Jupyter Notebook service.



Next step was to import **pandas** and read in the appropriate file and setting to **df**. Then checking if the data was displayed appropriately.

```
[62] import pandas as pd

     file_path = path + "/winemag-data-130k-v2.csv"

     df = pd.read_csv(file_path)
     df
```

	Unnamed: 0	country	description	designation	points	price	province	region_1	region_2	taster_
0	0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	NaN	O'K
1	1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN	Roger
2	2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willamette Valley	Paul Gr
3	3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	NaN	Alexa Pea

Handling Missing Data

Checking for any missing data using **null()**. In the screenshot below if there is a value listed as **True** then this indicates that the field is missing data.

```
[5] missing_values = df.isnull()
    missing_values
```

	Unnamed: 0	country	description	designation	points	price	province	region_1	region_2	taster_name
0	False	False	False	False	False	True	False	False	True	False
1	False	False	False	False	False	False	False	True	True	False
2	False	False	False	True	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	True	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
129966	False	False	False	False	False	False	False	True	True	False
129967	False	False	False	True	False	False	False	False	False	False
129968	False	False	False	False	False	False	False	False	True	False
129969	False	False	False	True	False	False	False	False	True	False
129970	False	False	False	False	False	False	False	False	True	False

129971 rows × 14 columns

For this project, the data was cleaned using the **dropna()** method. This will remove any rows that contain a missing value. After dropping the rows with missing information there is still a large amount of data to use for our analysis.

```
clean_df = df.dropna()
clean_df
```

	Unnamed: 0	country	description	designation	points	price	province	region_1	region_2	taste
4	4	US	Much like the regular bottling from 2012, this...	Vintner's Reserve Wild Child Block	87	65.0	Oregon	Willamette Valley	Willamette Valley	Paul
10	10	US	Soft, supple plum envelopes an oaky structure ...	Mountain Cuvée	87	19.0	California	Napa Valley	Napa	
23	23	US	This wine from the Geneseo district offers aro...	Signature Selection	87	22.0	California	Paso Robles	Central Coast	Ke
25	25	US	Oak and earth intermingle around robust aromas...	King Ridge Vineyard	87	69.0	California	Sonoma Coast	Sonoma	
35	35	US	As with many of the Erath 2010 vineyard design...	Hyland	86	50.0	Oregon	McMinnville	Willamette Valley	Paul
...	...	...	...	...	...	...	...	...	...	

Removing Duplicates

After checking for duplicates using **duplicated()**, there were no duplicates as a result.

```
duplicate_rows = df[df.duplicated()]

print(duplicate_rows)
```

```
Empty DataFrame
Columns: [Unnamed: 0, country, description, designation, points, price, province, region_1, region_2, taster
Index: []
```

Exploratory Data Analysis (EDA)

This project was based on the tutorial videos from **Greg Hogg** on Youtube. However the data used in this project is not the same data.

Source: https://www.youtube.com/playlist?list=PLKYEe2WisBTECZ8mZCfFxzrBBuGrS1Gfu

Since this project was based off of a series of tutorials there was no initial investigation or hypothesis regarding this data. The process was to get familiar with the python tools and observe the outcomes listed.

However, the choice of the wine reviews dataset was to view the correlations between the ratings of wine and the locations as well as price.

Insights and Findings

The ratings of wine compared to the price varied. There were some extreme outliers in terms of price. Most of the higher rankings seem to come from the California province, which is not unusual as California is known to have some of the most prestigious wineries.

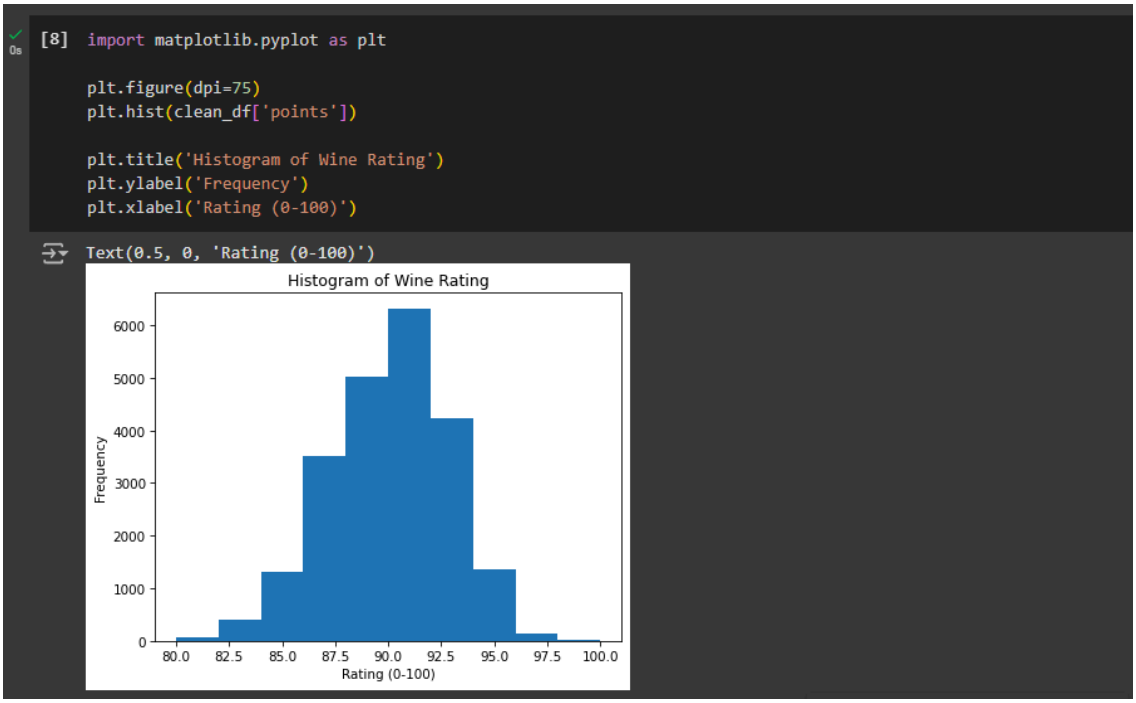
Throughout the tutorial, there was a challenge with using some of the fields in the dataset. When running the python scripts, the `Country` field was not projecting all the countries listed in the database. This caused some frustration and resulted in using `Province` for most of the data.

The extreme outliers also caused some of the graphs to become distorted, making some of the graphs difficult to read. One of these

outliers was a bottle of wine that was priced at over $2000 while most stayed within the 50-200 range.

Visualizations

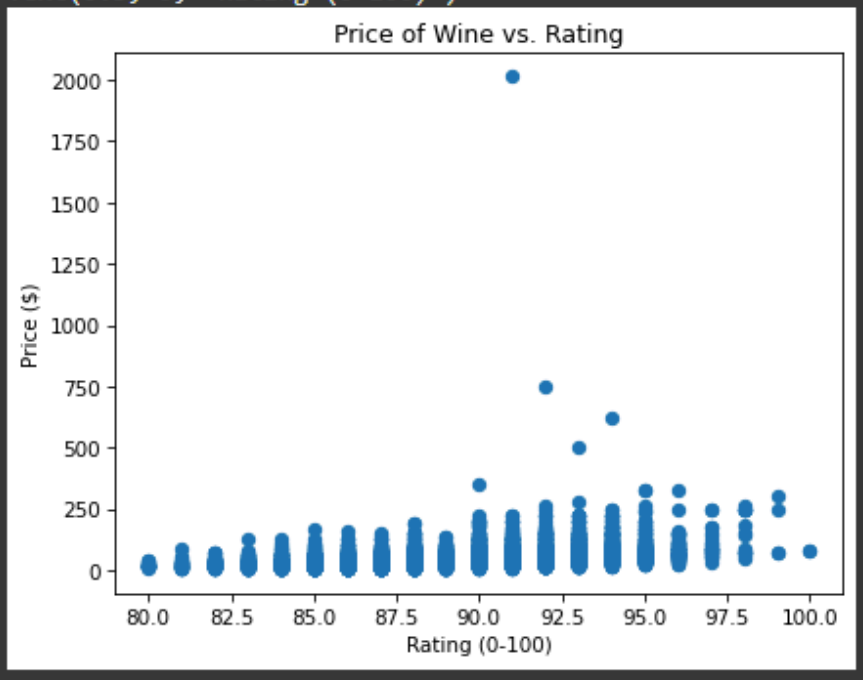
Below are the results of most of the tutorial examples, some of which are not helpful.



Histogram plot of the ratings for each wine. Most of the wine ratings fall into the 90-92.5 range. The lowest rating at 80 shows that the wine listed in the database are of good quality.

```
plt.figure(dpi=75)
plt.scatter(clean_df['points'], clean_df['price'])

plt.title('Price of Wine vs. Rating')
plt.ylabel('Price ($)')
plt.xlabel('Rating (0-100)')
```

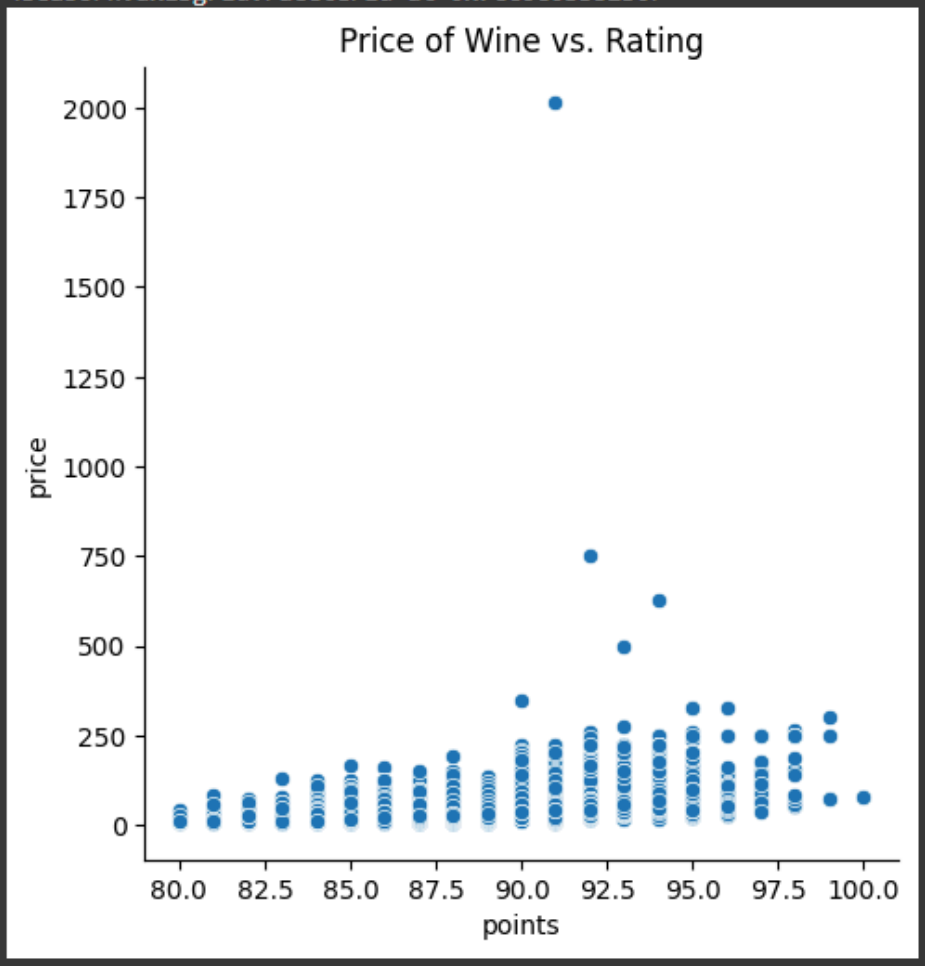
```
Text(0.5, 0, 'Rating (0-100)')
```



Comparing the price of wine to the rating in a scatterplot. Because there is an outlier wine priced significantly higher than the rest of the wine, the scatterplot was condensed into the bottom of the graph. This is an occurrence for most of the graphs.

```
import seaborn as sns

g = sns.relplot(kind='scatter', data=clean_df, x='points', y='price')
g.set(title='Price of Wine vs. Rating')
```

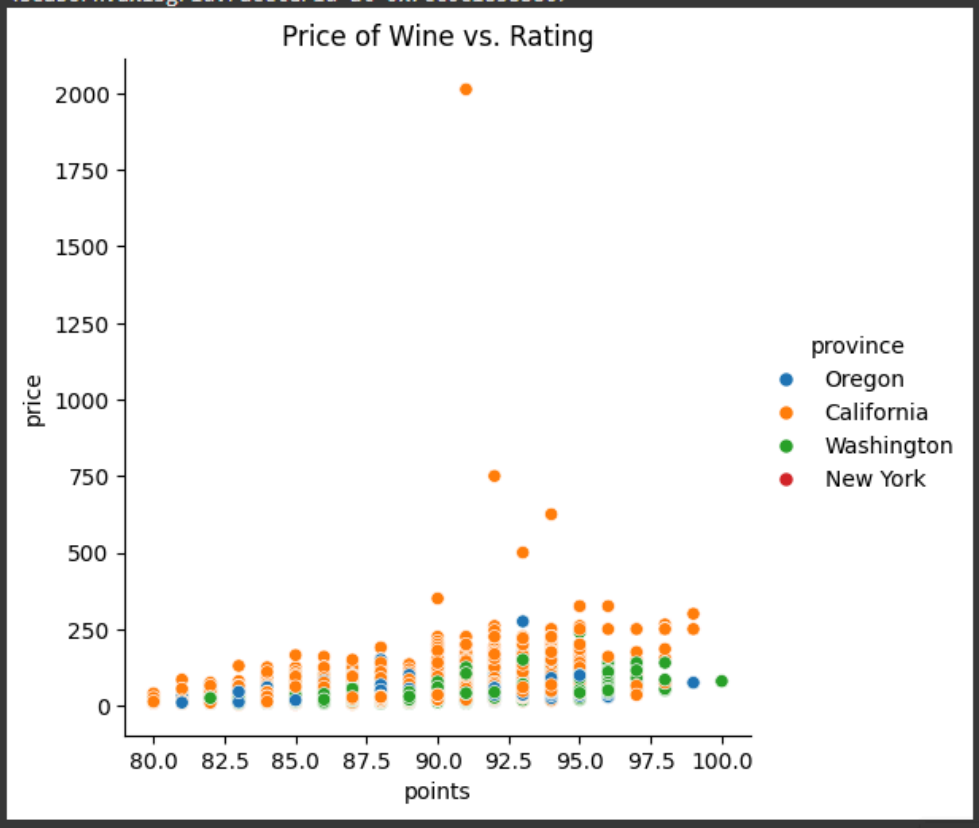
```
<seaborn.axisgrid.FacetGrid at 0x7cc0c0bb82b0>
```



This is the same scatter plot but using seaborn.

```
[26] g = sns.relplot(kind='scatter', data=clean_df, x='points', y='price', hue='province')
     g.set(title='Price of Wine vs. Rating')
```

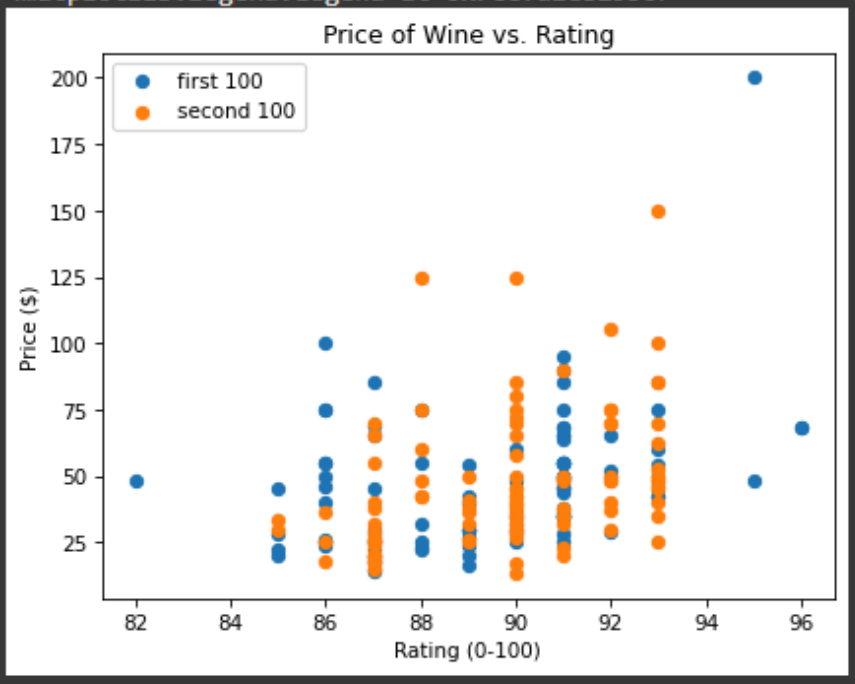
```
<seaborn.axisgrid.FacetGrid at 0x7cc0c1838b80>
```



With seaborn, colors are added to represent each province in the United States. In this scatterplot, California wines are shown the most, while New York is not visible due to the distorted scatterplot.

```
plt.figure(dpi=75)
plt.scatter(clean_df['points'][:100], clean_df['price'][:100])
plt.scatter(clean_df['points'][100:200], clean_df['price'][100:200])
plt.title('Price of Wine vs. Rating')
plt.ylabel('Price ($)')
plt.xlabel('Rating (0-100)')

plt.legend(['first 100', 'second 100'])
```

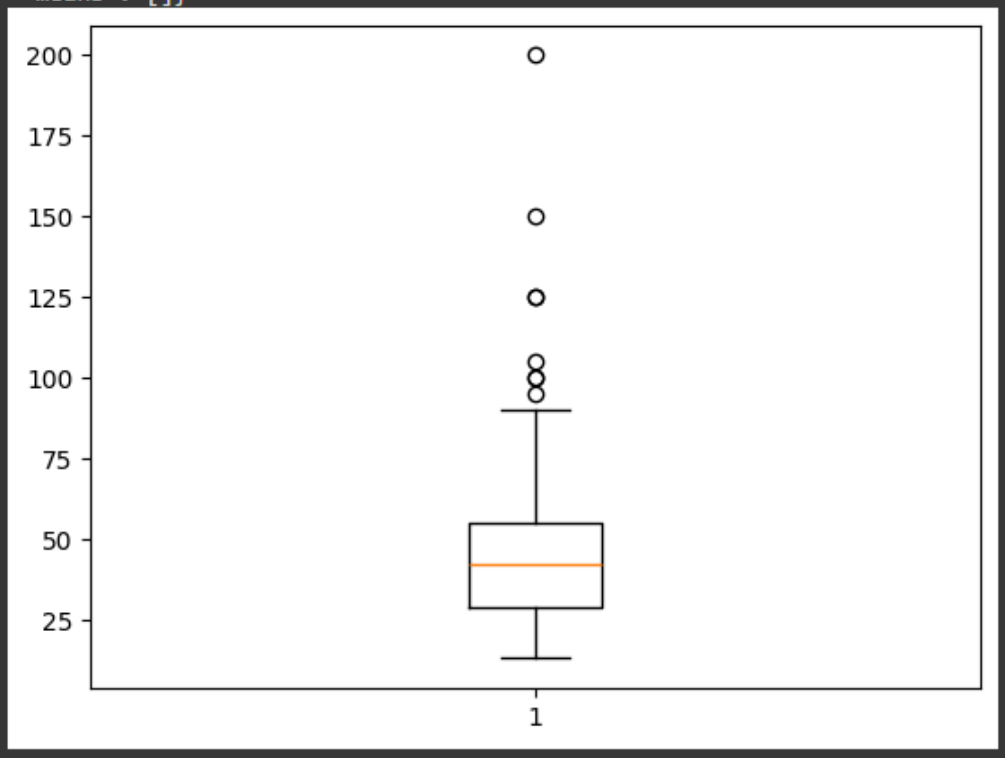
```
<matplotlib.legend.Legend at 0x7ccfd2e82950>
```



This was used to introduce the concept of splitting the data into two separate pieces of data. This could have been used to get a better visualization between different wine provinces or countries.

```
plt.boxplot(clean_df['price'][:200])
# plt.boxplot(clean_df['points'])
```

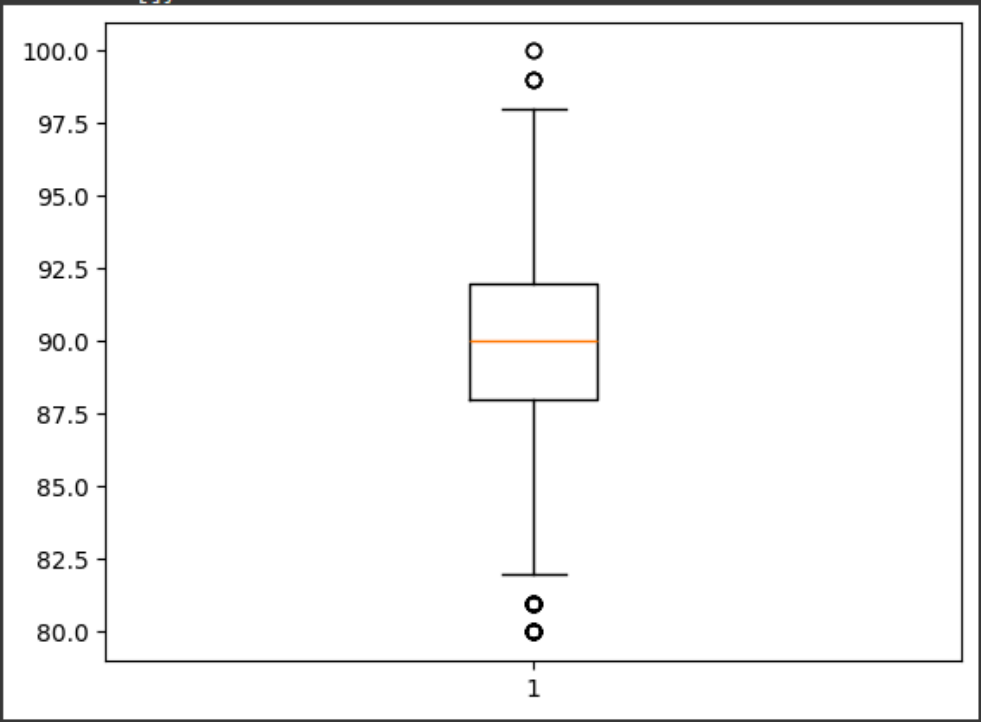
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7ccfd295dae0>,
  <matplotlib.lines.Line2D at 0x7ccfd295dd80>],
 'caps': [<matplotlib.lines.Line2D at 0x7ccfd295e020>,
  <matplotlib.lines.Line2D at 0x7ccfd295e2c0>],
 'boxes': [<matplotlib.lines.Line2D at 0x7ccfd295d840>],
 'medians': [<matplotlib.lines.Line2D at 0x7ccfd295e560>],
 'fliers': [<matplotlib.lines.Line2D at 0x7ccfd295e800>],
 'means': []}
```



This graph shows the extreme outliers in the data from only the first 200 wines. As we see the majority of the wine price is in the 25 to 50 range. However we have outliers that reach into 200 or higher. Which was one of the main reasons why some of the graphs created distorted.

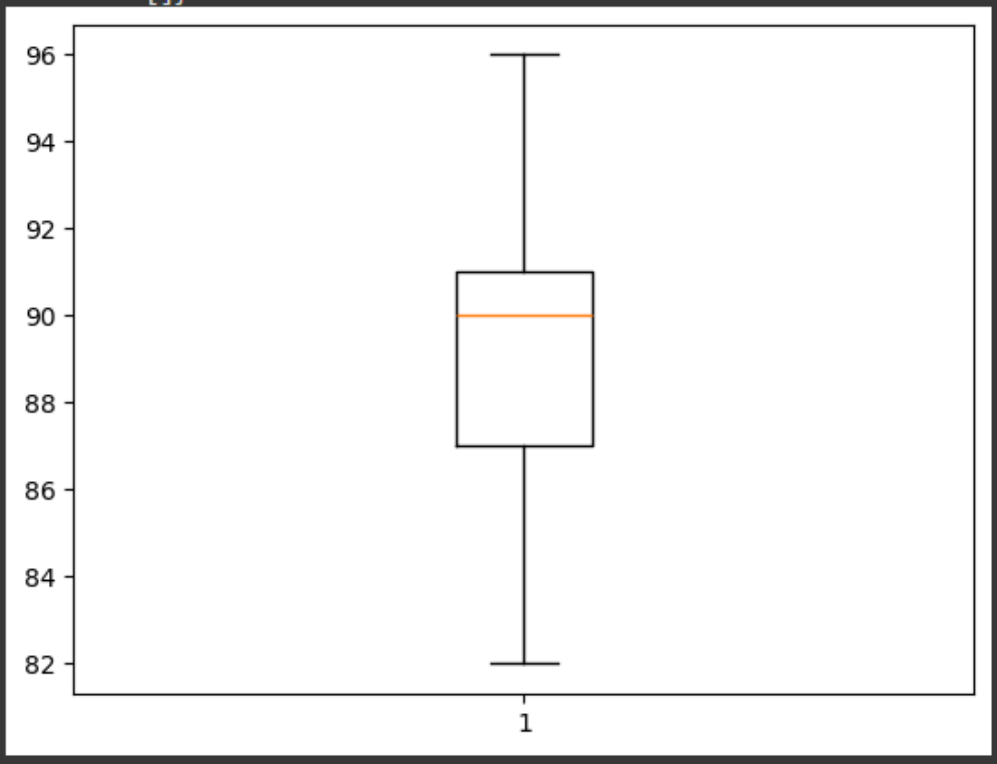
```
# plt.boxplot(clean_df['price'][:200])
plt.boxplot(clean_df['points'])
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7ccfd2ac9600>,
  <matplotlib.lines.Line2D at 0x7ccfd2ac98a0>],
 'caps': [<matplotlib.lines.Line2D at 0x7ccfd2ac9b40>,
  <matplotlib.lines.Line2D at 0x7ccfd2ac9de0>],
 'boxes': [<matplotlib.lines.Line2D at 0x7ccfd2ac9360>],
 'medians': [<matplotlib.lines.Line2D at 0x7ccfd2aca080>],
 'fliers': [<matplotlib.lines.Line2D at 0x7ccfd2aca320>],
 'means': []}
```



```
# plt.boxplot(clean_df['price'][:200])
plt.boxplot(clean_df['points'][:200])
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7ccfd2a423b0>,
  <matplotlib.lines.Line2D at 0x7ccfd2a42650>],
 'caps': [<matplotlib.lines.Line2D at 0x7ccfd2a428f0>,
  <matplotlib.lines.Line2D at 0x7ccfd2a42b90>],
 'boxes': [<matplotlib.lines.Line2D at 0x7ccfd2a42110>],
 'medians': [<matplotlib.lines.Line2D at 0x7ccfd2a42e30>],
 'fliers': [<matplotlib.lines.Line2D at 0x7ccfd2a430d0>],
 'means': []}
```



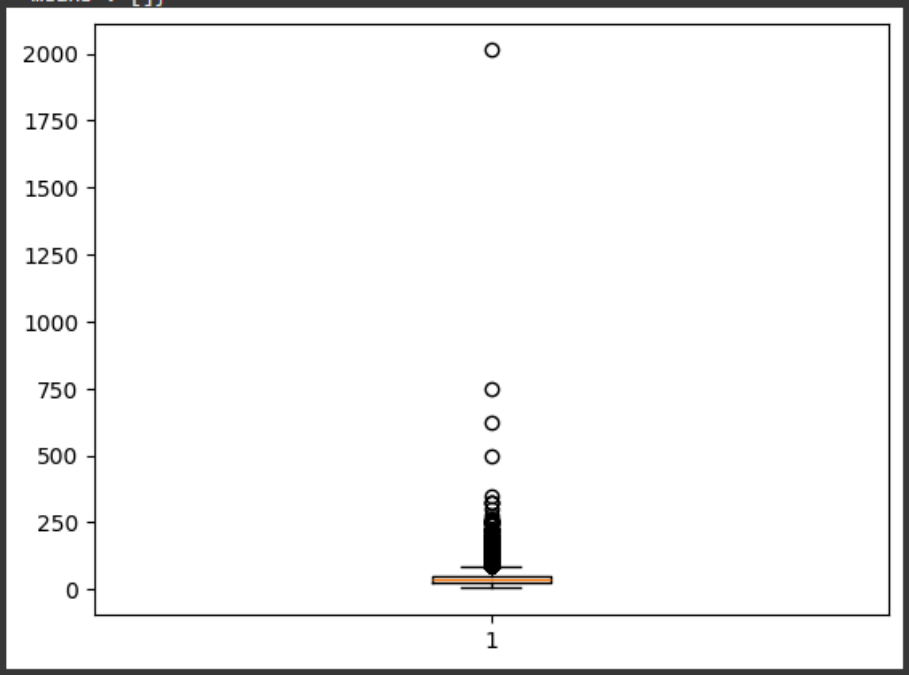
```
[57] bins_series = pd.cut(clean_df['price'][:200], bins=4).value_counts()
     bins_series
```

price	count
(12.813, 59.75]	153
(59.75, 106.5]	43
(106.5, 153.25]	3
(153.25, 200.0]	1

dtype: int64

```
[53] plt.boxplot(clean_df['price'])
     # plt.boxplot(clean_df['points'])
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7ccfd29b91b0>,
  <matplotlib.lines.Line2D at 0x7ccfd29b9450>],
 'caps': [<matplotlib.lines.Line2D at 0x7ccfd29b96f0>,
  <matplotlib.lines.Line2D at 0x7ccfd29b9990>],
 'boxes': [<matplotlib.lines.Line2D at 0x7ccfd29b8f10>],
 'medians': [<matplotlib.lines.Line2D at 0x7ccfd29b9c30>],
 'fliers': [<matplotlib.lines.Line2D at 0x7ccfd29b9ed0>],
 'means': []}
```



```
[57] bins_series = pd.cut(clean_df['price'][:200], bins=4).value_counts()
     bins_series
```

price	count
(12.813, 59.75]	153
(59.75, 106.5]	43
(106.5, 153.25]	3
(153.25, 200.0]	1

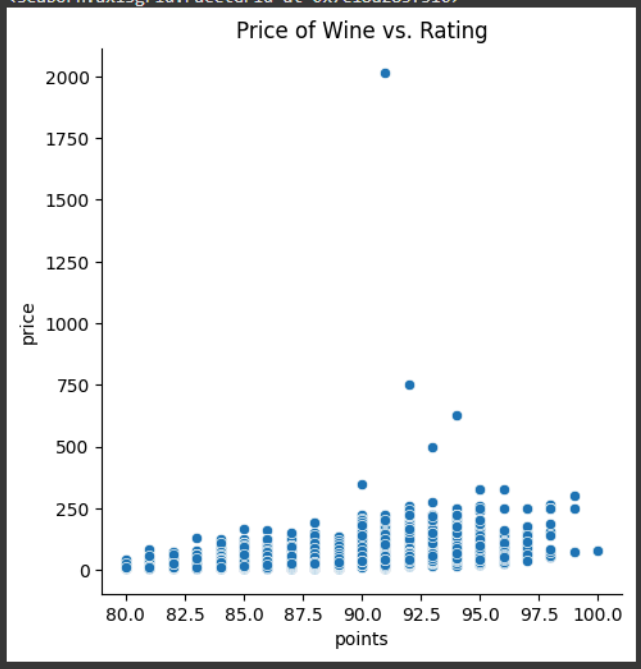
dtype: int64

```
values = bins_series.to_list()
labels = [str(x) for x in bins_series.index.to_list()]
labels
```

```
['(12.813, 59.75]', '(59.75, 106.5]', '(106.5, 153.25]', '(153.25, 200.0]']
```

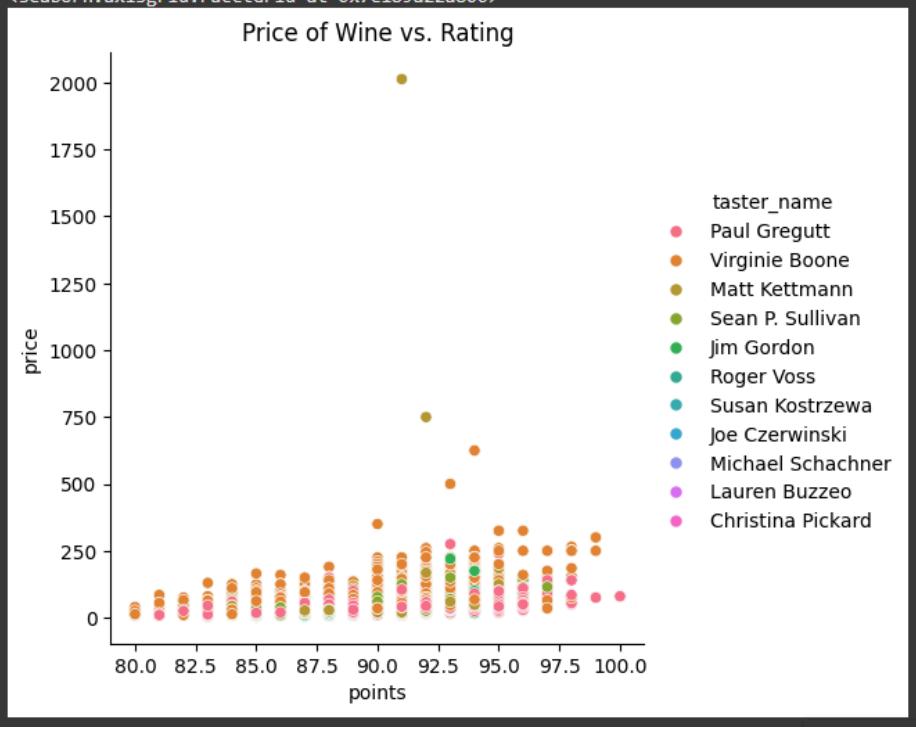
```
[16] import seaborn as sns

     g = sns.relplot(kind='scatter', data=clean_df, x='points', y='price')
     g.set(title='Price of Wine vs. Rating')
```

```
<seaborn.axisgrid.FacetGrid at 0x7e18a283f310>
```



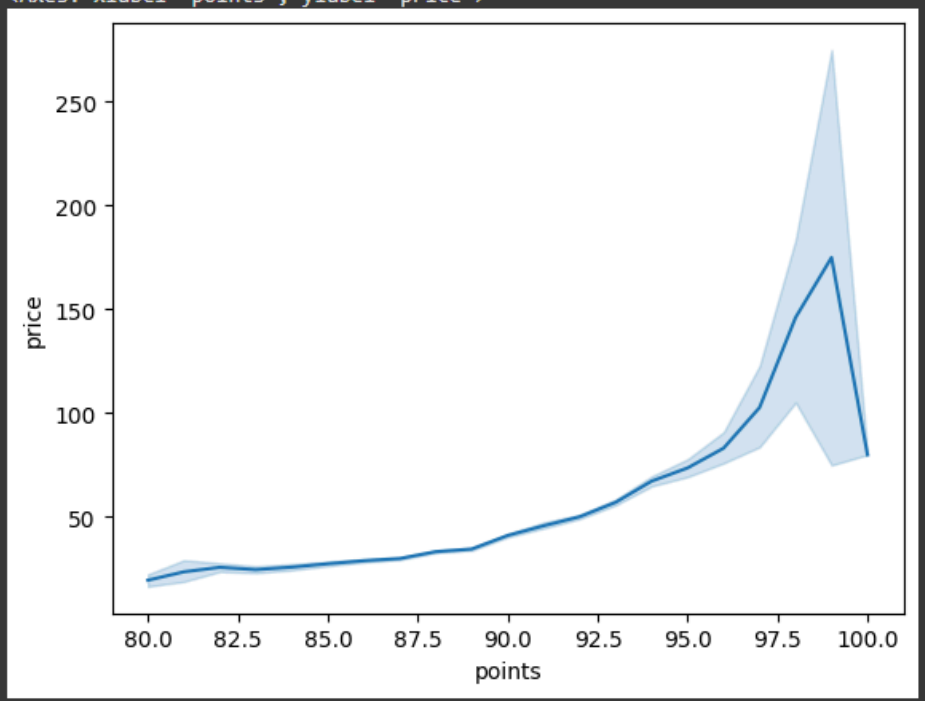
```
[20] g = sns.relplot(kind='scatter', data=clean_df, x='points', y='price', hue='taster_name')
     g.set(title='Price of Wine vs. Rating')
```

```
<seaborn.axisgrid.FacetGrid at 0x7e189a22a800>
```



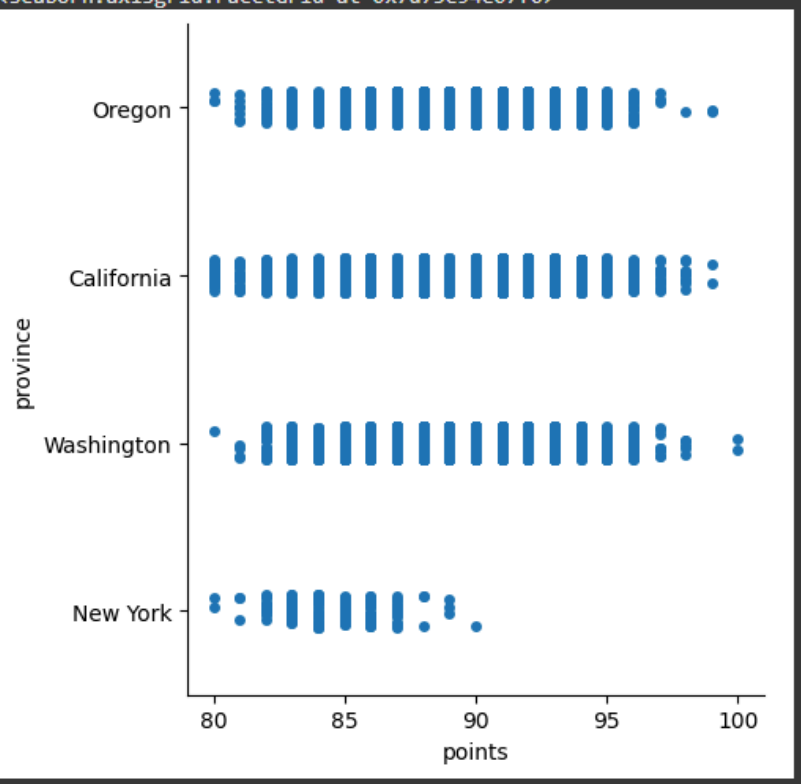

```
sns.lineplot(x='points', y='price', data=clean_df)
```

```
<Axes: xlabel='points', ylabel='price'>
```



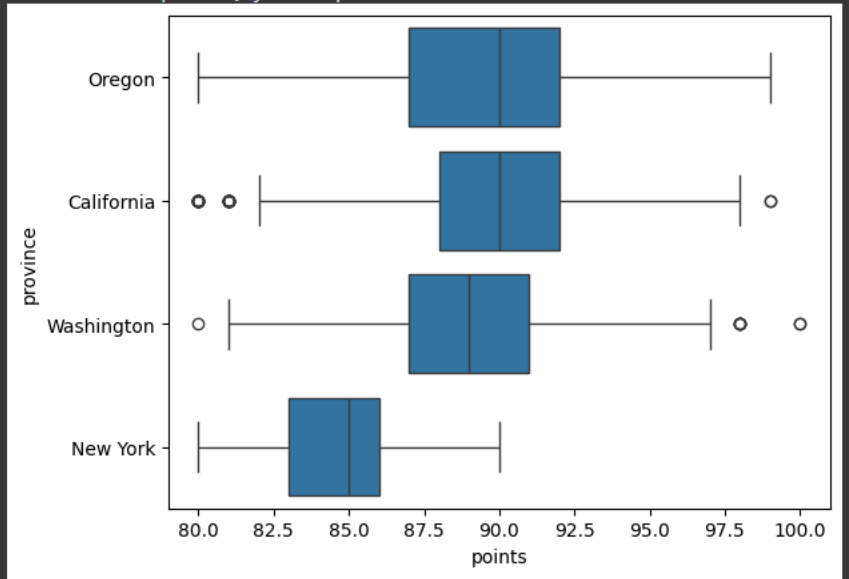
```
sns.catplot(x='points', y='province', data=clean_df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7d75e94e07f0>
```



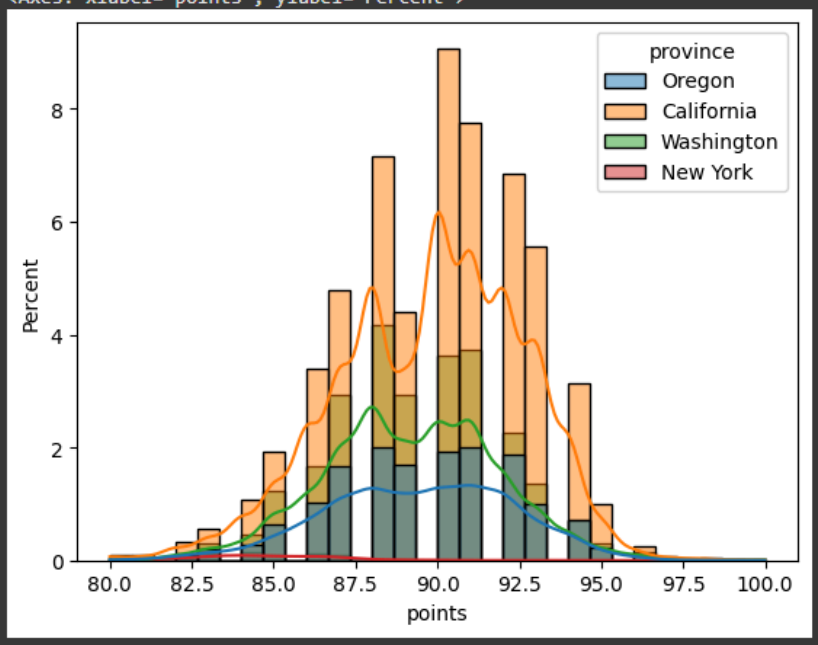

```
sns.boxplot(x='points', y='province', data=clean_df)
```

```
<Axes: xlabel='points', ylabel='province'>
```



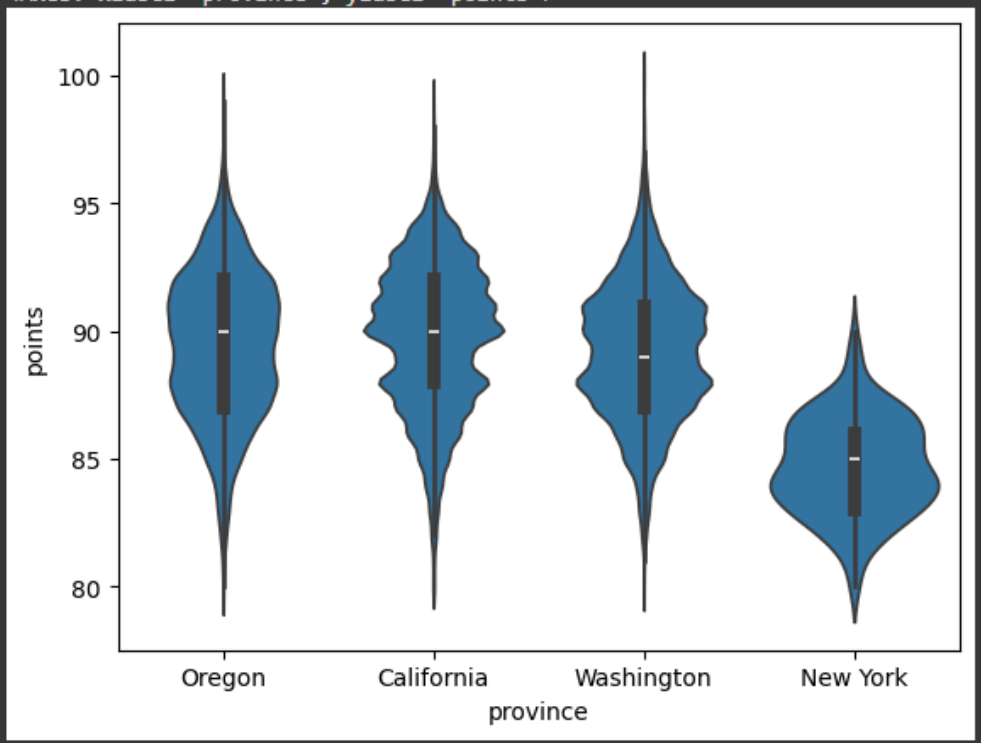
```
[64] sns.histplot(data=clean_df, x='points', bins=30, kde=True, stat='percent', hue='province')
```

```
<Axes: xlabel='points', ylabel='Percent'>
```




```
sns.violinplot(y='points', x='province', data=clean_df)
```

```
<Axes: xlabel='province', ylabel='points'>
```



Conclusion

From the data gathered it would seem that the better quality of wine is located in California. Although, the data was limited to the US province due to complications with the use of the data. This could be from removing too much of the information during the data cleaning process.

This project however was simply designed to introduce data science foundations in python. As well as getting a better understanding with a dataset that was not provided from the listed tutorials.

Future Work

This project was a good start to get familiar with data usage in python. While this is the end of this project. Other projects that may use graphs could potentially arise such as tracking the prices of groceries. The main future work will be diving deeper into understanding data science and potentially machine learning.

Appendices

Dataset:

https://www.kaggle.com/datasets/zynicide/wine-reviews?select=winemag-data-130k-v2.csv

Gregg Hogg Listed Tutorial (beginning to seaborn):

https://www.youtube.com/playlist?list=PLKYEe2WisBTECZ8mZCfFxzrBBuGrS1Gfu