

BDA_Ch3

2019 年 11 月 6 日

```
In [ ]: # 首先需要引入 pandas 及相关的包
        from pandas import Series

        arr = [1,2,3,4,5]# 创建列表
        series1 = Series(arr)# 通过列表创建 Series
        # 遍历 Series
        for index, val in series1.items():
            print(index, val)
        series1[1] = 'two'# 根据位置下标修改值
        series1.index = ['a','b','c','d','e']# 修改索引标签
        series1['d'] = 'four'# 根据标签下表修改值
        series1.name = 'column_name'
        # 与字典不同, 这里的索引标签可以重复
        series1.index = ['a','b','c','d','c']
        # 使用 rename 方法进行重命名, inplace = True: 不创建新的对象, 直接对原始对象进行修改
        series1.rename('new_name',inplace = True)
        print(series1['c'])# 列出所有索引标签为 'c' 的元素
        print(series1[['a','d']])# 可以列多个标签, 但是要用列表形式
        print(series1[[1,2,4]])# 通过位置来进行检索
        print(series1[1:4])
        # 通过 append() 添加 Series, 注意一定要有赋值操作
        series1 = series1.append(Series([6],index=['f']))
        series1.rename(index={'f':'e'},inplace = True)# 更改行标签
        #drop() 删除索引标签为 'a' 的元素并创建一个新的对象, 所以需要使用 inplace = True
        series1.drop('a', inplace = True)
        del series1['b']#del 命令直接修改原对象
        print(Series({'aa':1,'bb':2}))# 也可以通过字典来创建 Series

In [ ]: from pandas import DataFrame
        import pandas as pd
        import numpy as np

        #np.nan 表示缺失值
        df1 = DataFrame({'From_To': [' LoNDon_paris ', ' MAdrid_miLAN ', ' londON_StockhOlM ',
                                     'Budapest_PaRis', 'Brussels_londOn'],
                          'FlightNumber': [10045, np.nan, 10065, np.nan, 10085],
                          'RecentDelays': [[23, 47], [], [24, 43, 87], [13], [67, 32]],
                          'Airline': ['KLM(!)', '<Air France> (12)', '(British Airways. )',
                                     '12. Air France', '"Swiss Air"']})

        # 按行遍历 DataFrame, 按列遍历则为 iteritems()
        for index, row in df1.iterrows():
            print(row['From_To'])
        #loc 进行数据查询, 其中的参数均为索引
        df1.loc[0,'Airline']
```

```

# 与 loc 不同, iloc 中的参数均为数字, 表示二维表中的位置, -1 表示最后一位
df1.iloc[0,0]
# 切片略有不同
# iloc 中 1:3 不包括 3, loc 中的 1:3 包括 3
df1.iloc[1:3,0]
df1.loc[1:3,'Airline']
# 使用 loc 来进行切片选择一部分数据
# 1:3 表示从行索引为 1 开始到行索引为 3, ('From_To','Airline') 表示选择 From_To 和 Airline 这两列
df1.loc[1:3,('From_To','Airline')]
# 使用 iloc 的话
df1.iloc[1:3,[0,3]]
# 使用 loc 来切片选择 From_To 这一列, 这里 () 可以省略
# df1.From_To 表示 df1 的 From_To 这一列, str.upper() 为所有的字母变为大写
df1.loc[:, 'From_To'] = df1.From_To.str.upper()
df1.loc[:, 'From_To'] = df1.From_To.str.lower()# 改为小写
# 剔除出 Airline 中的噪音数据
df1['Airline'] = df1.Airline.str.extract('([a-zA-Z\s]+)')
# 将出发点和目的地分开成两列
# 注意这里的 expand 参数, 如果为 True 则分割后成为多列
# 如果为 False 则分割后仍然为一列, 用一组列表来表示多个地点
df1.From_To.str.split('_', expand=True)
# 上述语句仅仅是生成一个新的 DataFrame 对象, 并不修改 df1 本身
# 需要通过 join 来拼接两个 DataFrame, 并将拼接后新生成的 DataFrame 赋值给 df1
df1 = df1.join(df1.From_To.str.split('_', expand=True))
# 对列进行重命名
# 如果没有标注 columns, 默认为对行索引进行重命名
df1.rename(columns={0:'From',1:'To'},inplace=True)
# 删除 From_To 这一列, 同样的, 如果没有标注 columns, 默认为删除 'From_To' 这一行
# 当然, 因为没有这样一行, 所以会报错
df1.drop(columns=['From_To'],inplace=True)
# 添加行
# 如果不指定行索引的话, 必须使用 ignore_index=True
df1 = df1.append({'FlightNumber':'MF8169','RecentDelays':[],'Airline':'Xiamen Airline',
                  'From':'Xiamen','To':'Beijing'},ignore_index=True)
# 或者用如下方式指定行索引
# 此处需要先通过 DataFrame() 创建一个 DataFrame, 然后再使用 append
df1 = df1.append(DataFrame([{'FlightNumber':'MF8159','RecentDelays':[],
                             'Airline':'Xiamen Airline','From':'Xiamen',
                             'To':'Beijing'}],index=[8]))

```

```
In [ ]: import pandas as pd
```

```

#header 表示列名所在的行, 如果没有列名则为 None
#sheet_name 表示要读取的 Sheet, 可以是具体名字, 也可以是第几张 Sheet
data = pd.read_excel('test.xlsx', sheet_name = 0, header = 0)
print(data)
# 查看前 5 行数据
print(data.head(5))
# 查看最后 5 行数据
print(data.tail(5))
# 查看简单统计数据
data.describe()
# 数据的筛选

```

```

data[data.日开盘价>10]
data[data.公司简称.str.startswith('万')]
data[data.公司简称.str.contains('A')]
# 列级别的操作
data['new_col1'] = data['日最高价'] - data['日最低价']
data['new_col2'] = data['日最高价'] * data['日最低价']

data.rename(columns={0:'stock_id',1:'company'}, inplace=True)
writer = pd.ExcelWriter('out.xlsx')
data.to_excel(writer, 'sheet1', header = True, index = False)
writer.save()

```

In [5]: # 引入 json 文件解析包

```

import json

# 正常打开文件
f = open('json_test.json')
# 使用 json 包加载文件内容, 加载后就会变成嵌套的字典形式
json_data = json.load(f)
# 在这个文件中, 第一层是字典, 字典里面嵌套了一个列表 (第二层)
sites = json_data['sites']
print(sites)
print('-----')
# 第三层是字典
# 使用 for 循环来进行遍历
for s in sites:
    #s 就是其中一个字典
    # 具体的数据提取方式和字典完全一样
    print('{0}:\t{1}'.format(s['name'],s['url']))
# 关闭文件
f.close()

```

```

[{'name': '菜鸟教程', 'url': 'www.runoob.com'},
{'name': 'google', 'url': 'www.google.com'},
{'name': '微博', 'url': 'www.weibo.com'}]

```

```

-----
菜鸟教程:      www.runoob.com
google:        www.google.com
微博:          www.weibo.com

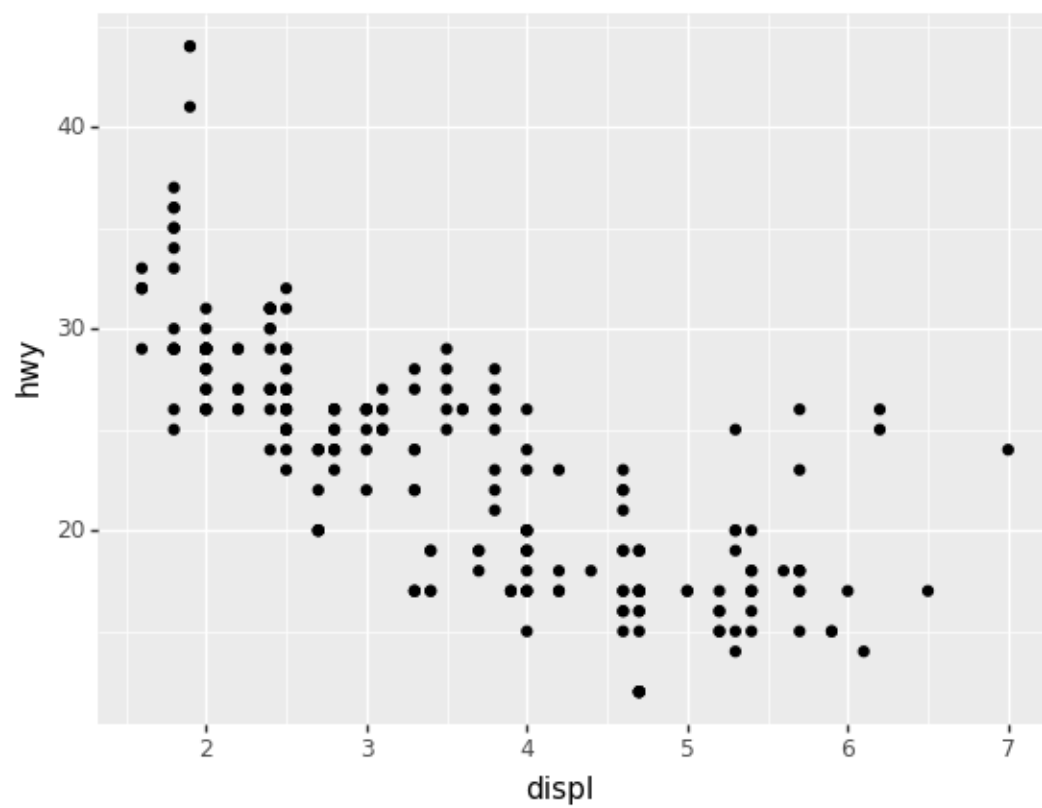
```

In [2]: from plotnine import * # 导入 plotnine 包的所有绘图函数
from plotnine.data import * # 导入 plotnine 包中的所有数据

```

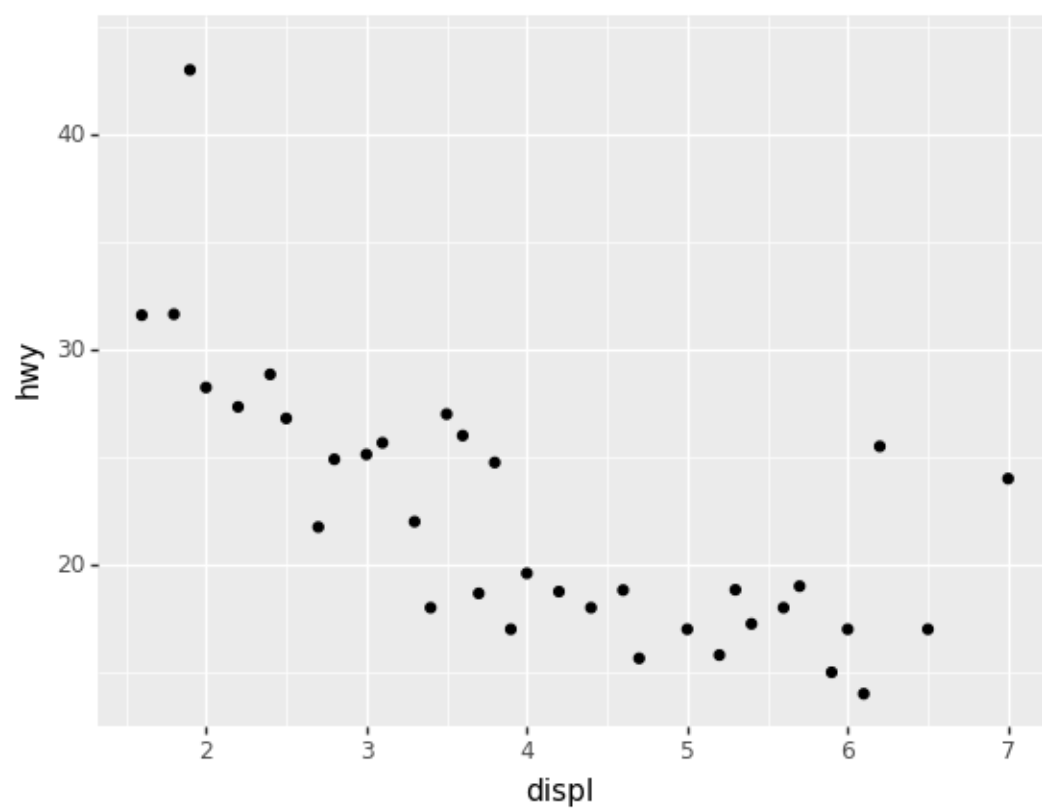
#plotnine 自带的关于汽车油耗数据集 mpg
ggplot(mpg,aes(x='displ',y='hwy'))+geom_point()

```



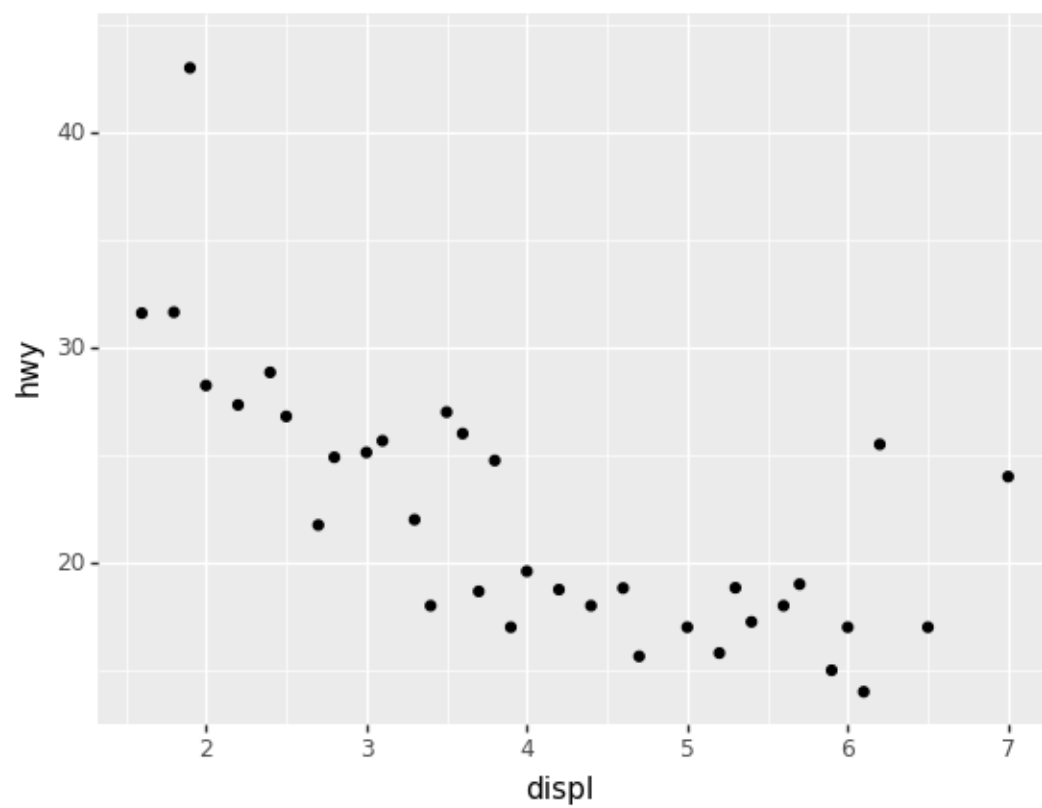
Out[2]: <ggplot: (-9223371842269796513)>

In [3]: `#summary` 默认返回条件均值, $E(y/x)$
 # 即每一个 x 值对应的 y 的均值
`ggplot(mpg,aes(x='displ',y='hwy'))+geom_point(stat='summary')`



Out[3]: <ggplot: (-9223371842269816362)>

In [4]: # 对应的 `stat` 函数
`ggplot(mpg,aes(x='displ',y='hwy'))+stat_summary(geom='point')`

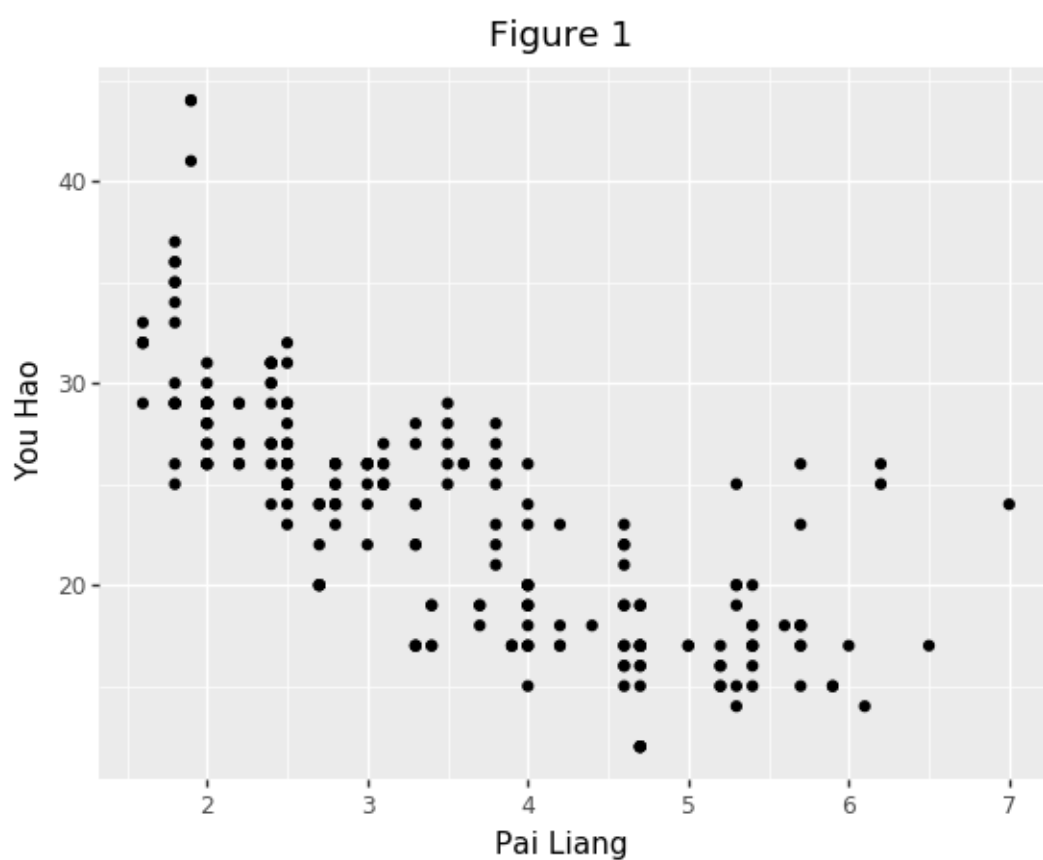


Out[4]: <ggplot: (194585068008)>

In [8]: # 修改轴坐标, 添加图标题

注意整个语句用一对圆括号括了起来, 所以可以换行

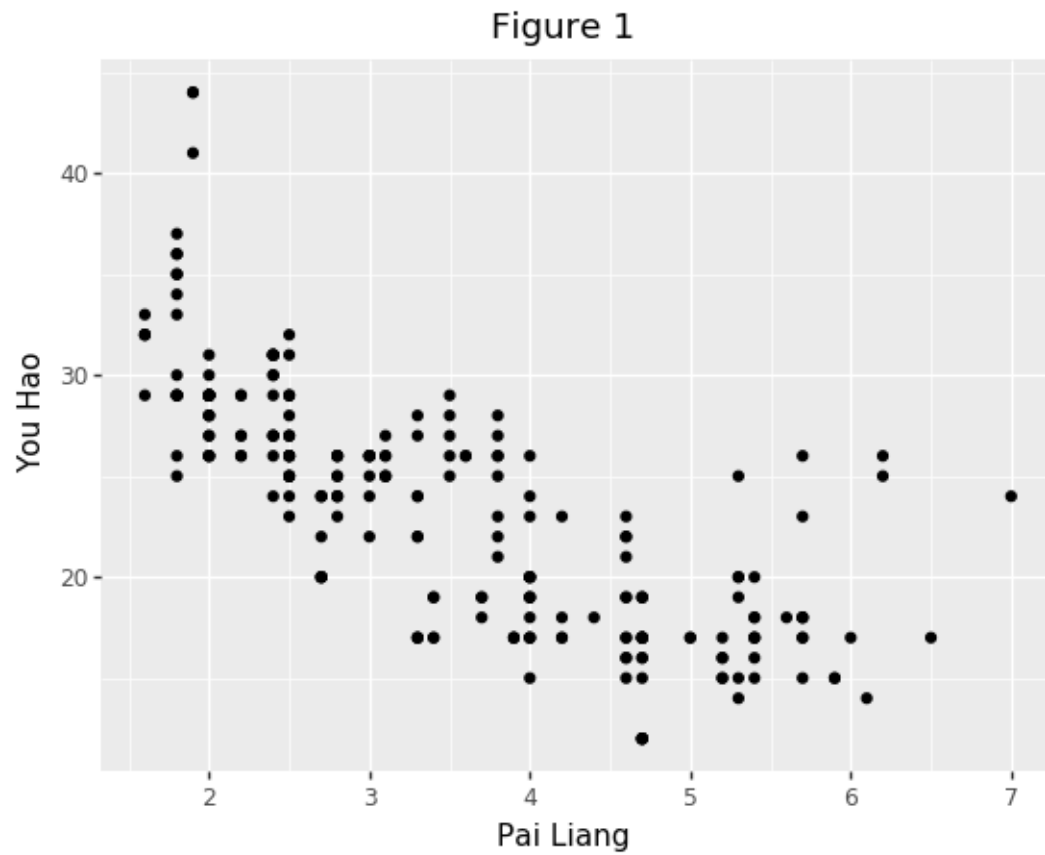
```
(ggplot(mpg,aes(x='displ',y='hwy'))+geom_point()+ggtitle('Figure 1')
+ xlab('Pai Liang')+ylab('You Hao'))
```



Out[8]: <ggplot: (89413311152)>

In [9]: # 另外一种更改标题的方式

```
(ggplot(mpg,aes(x='displ',y='hwy'))+geom_point()  
+labs(title='Figure 1',x='Pai Liang',y='You Hao'))
```



Out[9]: <ggplot: (89400756281)>

In [16]: # 更改坐标轴显示范围 *xlim*, *ylim*

对于连续型变量可以使用 *None* 来设定单侧界限

比如 *ylim(None,20)* 表示最大值为 20, 下限系统自动设置

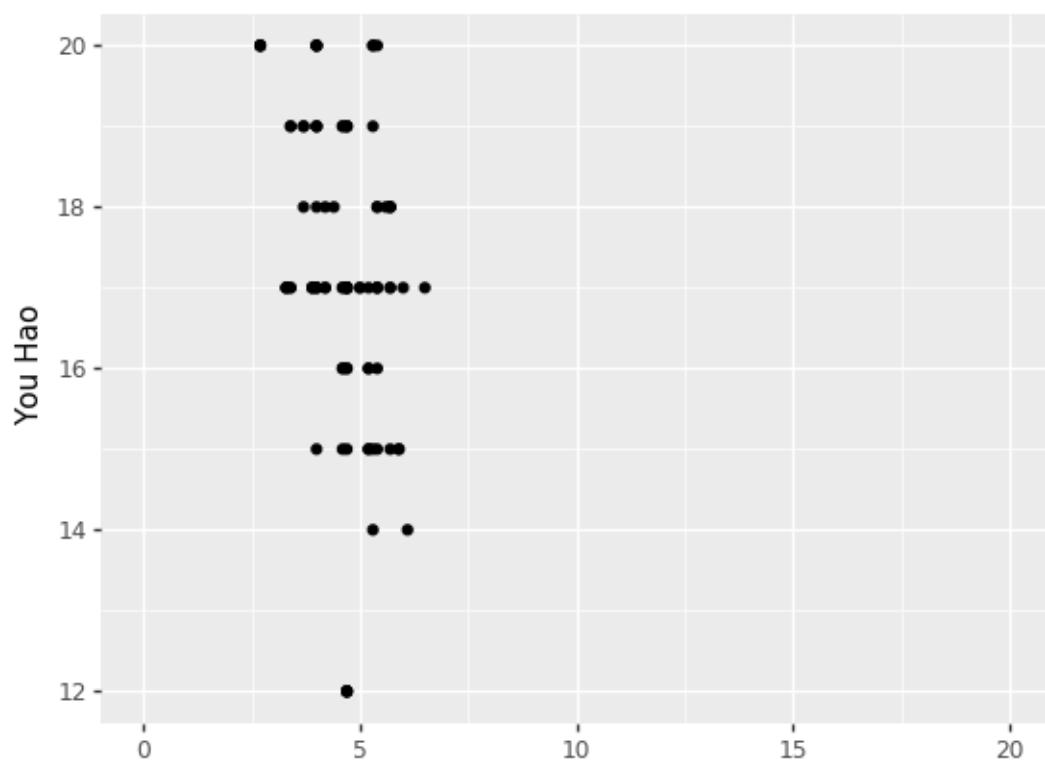
如果 *x* 轴是离散型变量, 则可以使用 *xlim(值 1,..., 值 n)* 来进行筛选

省略 *x* 轴标签

```
(ggplot(mpg,aes(x='displ',y='hwy'))+geom_point()  
+labs(title='Figure 1',x='',y='You Hao')  
+xlim(0,20)+ylim(20,30))
```

c:\users\binfang\appdata\local\programs\python\python37\lib\site-packages\plotnine\layer.py:452: PlotnineWarning: *geom_point()* requires *na.rm=TRUE* to be set. This warning arises because the underlying *ggplot2::geom_point()* function will not automatically remove *NA* values. This warning arises because the underlying *ggplot2::geom_point()* function will not automatically remove *NA* values.
self.data = self.geom.handle_na(self.data)

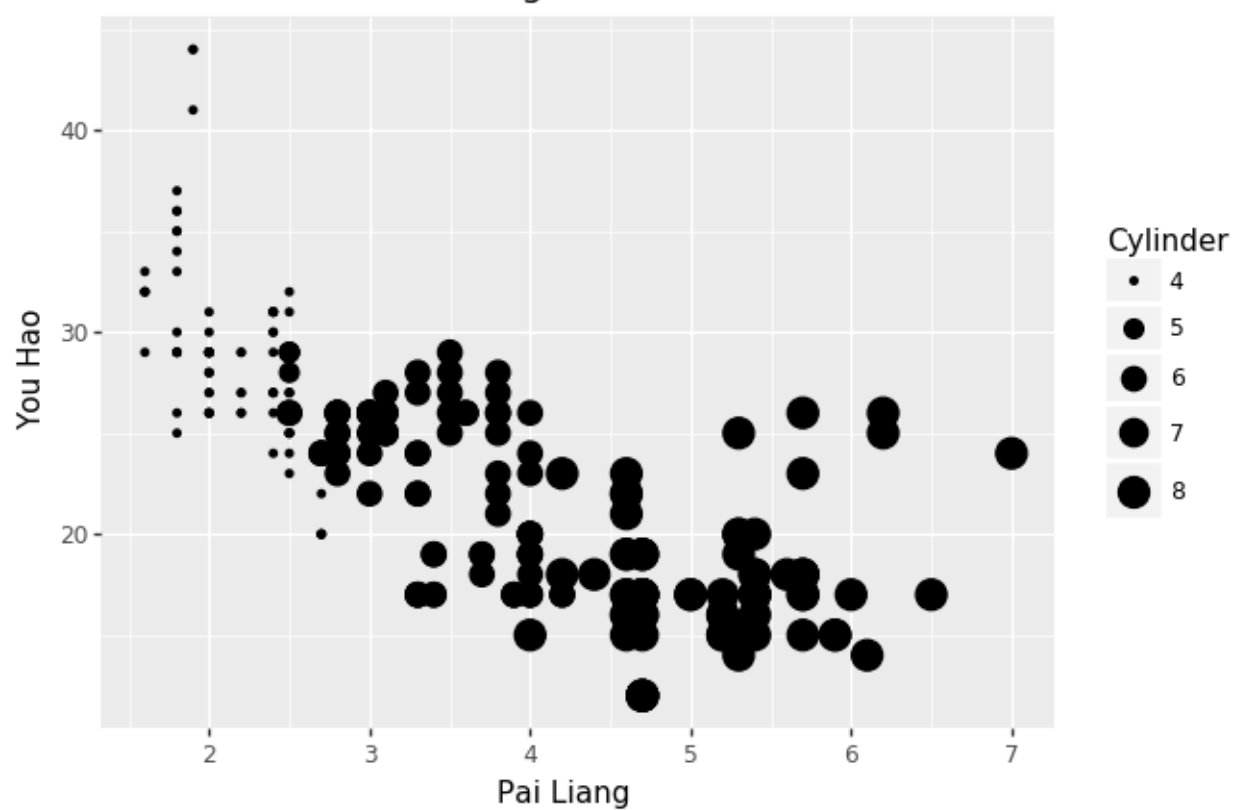
Figure 1



Out[16]: <ggplot: (-9223371908191863457)>

```
In [10]: # 改变点的大小，变成气泡图
# 气泡面积大小就是变量值的大小（相对）
# 用 size 参数指定
# 在 geom_point() 中可以用 color 参数指定颜色
# 比如 geom_point(color='blue')
(ggplot(mpg,aes(x='displ',y='hwy', size='cyl'))+geom_point()
+labs(title='Figure 1',x='Pai Liang',y='You Hao',size='Cylinder'))
```

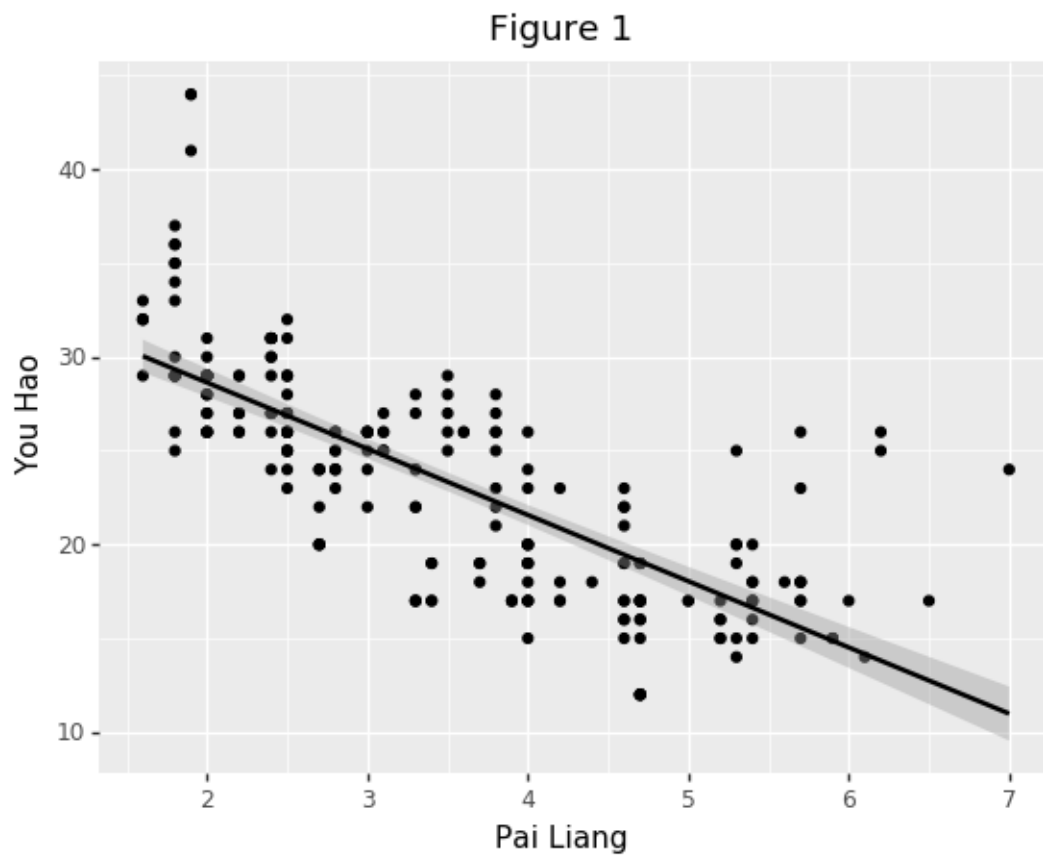
Figure 1



```
Out[10]: <ggplot: (-9223371911203886108)>
```

```
In [10]: # 添加一条趋势线
#method = 'lm'表示使用线性拟合的方式
# 默认会给出置信区间，用阴影表示
(ggplot(mpg,aes(x='displ',y='hwy'))+geom_point()
+labs(title='Figure 1',x='Pai Liang',y='You Hao')+geom_smooth(method='lm'))
```

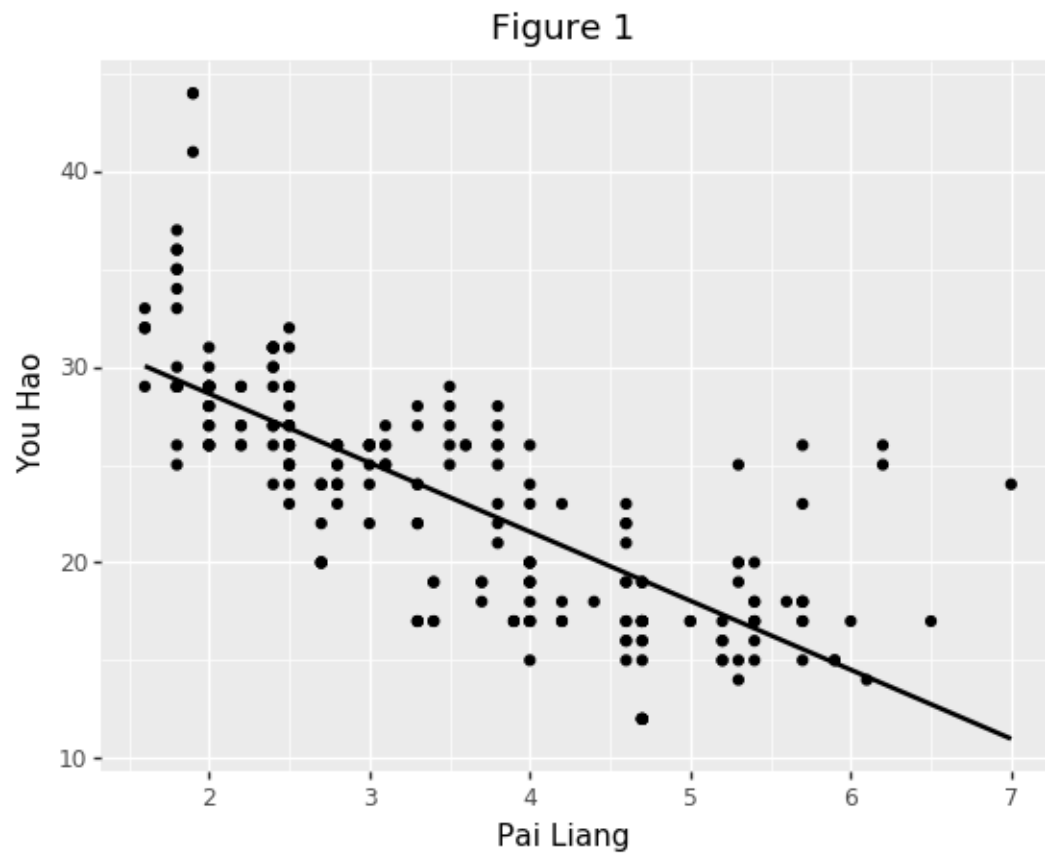
```
c:\users\binfang\appdata\local\programs\python\python37\lib\site-packages\numpy\core\fromnumeric.py:2495: FutureWarning
return ptp(axis=axis, out=out, **kwargs)
```



```
Out[10]: <ggplot: (89413449736)>
```

```
In [11]: # 使用 se=False 选项去掉置信区间
(ggplot(mpg,aes(x='displ',y='hwy'))+geom_point()
+labs(title='Figure 1',x='Pai Liang',y='You Hao')+geom_smooth(method='lm',se=False))
```

```
c:\users\binfang\appdata\local\programs\python\python37\lib\site-packages\numpy\core\fromnumeric.py:2495: FutureWarning
return ptp(axis=axis, out=out, **kwargs)
```

Out[11]: <ggplot: (89413408618)>

In [7]: # 分组展示

以 *drv* 变量 (汽车的驱动方式) 为分组变量

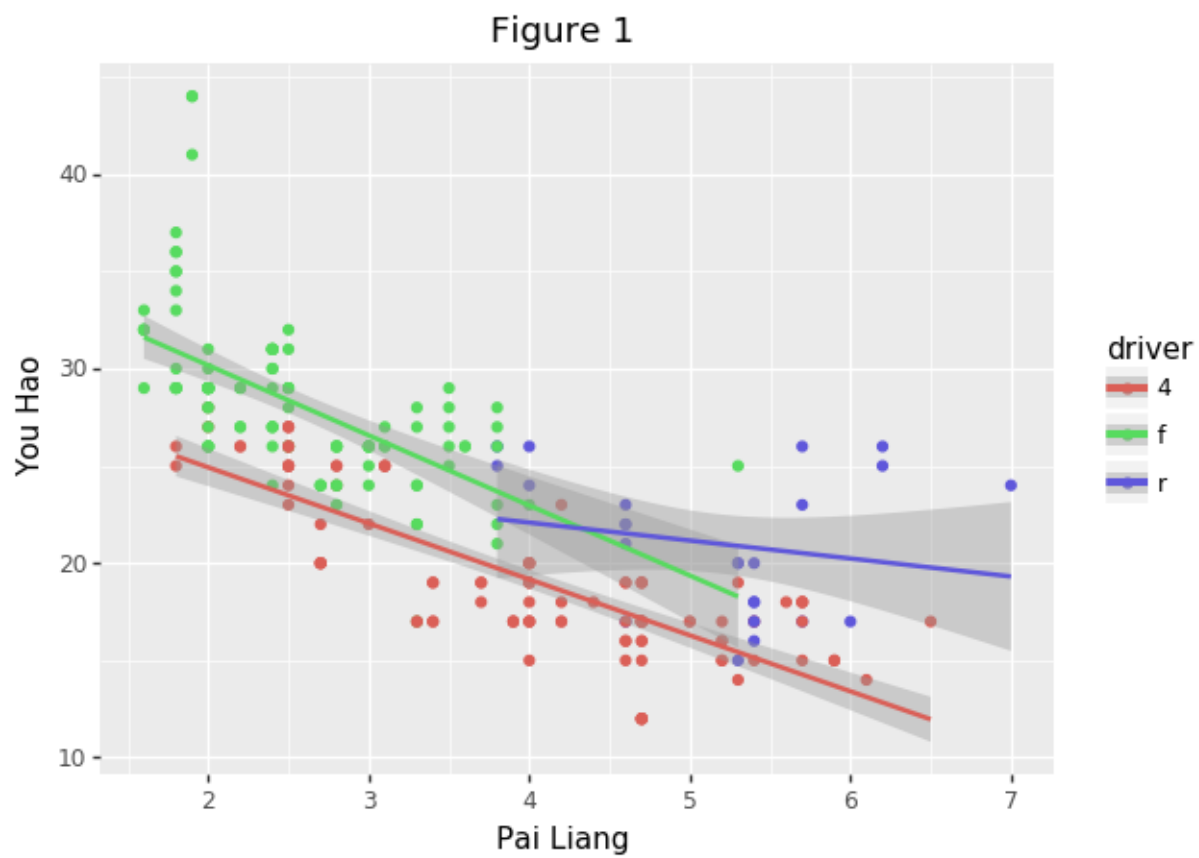
这里使用的是对不同的组别上不同的颜色的方式, 所以用了 *color* 参数

颜色系统自动会进行设置, 当然也可以自行设置

由于是在 *ggplot* 里面进行的分组设置, 因此所有的图层都会分组

```
(ggplot(mpg,aes(x='displ',y='hwy', color='factor(drv)'))+geom_point()
+labs(title='Figure 1',x='Pai Liang',y='You Hao',color='driver')
+geom_smooth(method='lm'))
```

c:\users\binfang\appdata\local\programs\python\python37\lib\site-packages\numpy\core\fromnumeric.py:2495: FutureWarning
return ptp(axis=axis, out=out, **kwargs)

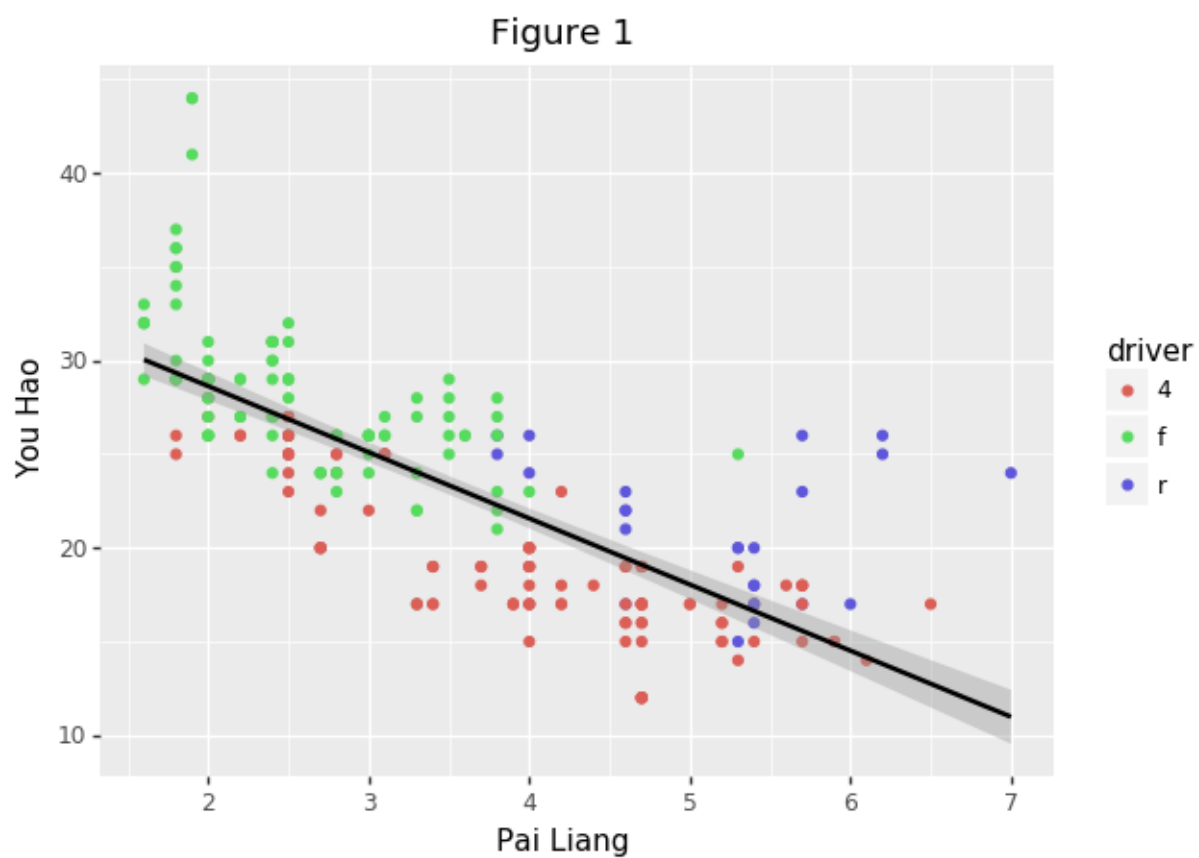


Out[7]: <ggplot: (125647019966)>

In [33]: # 这里仅在散点图图层进行了分组，因此拟合曲线部分不受影响

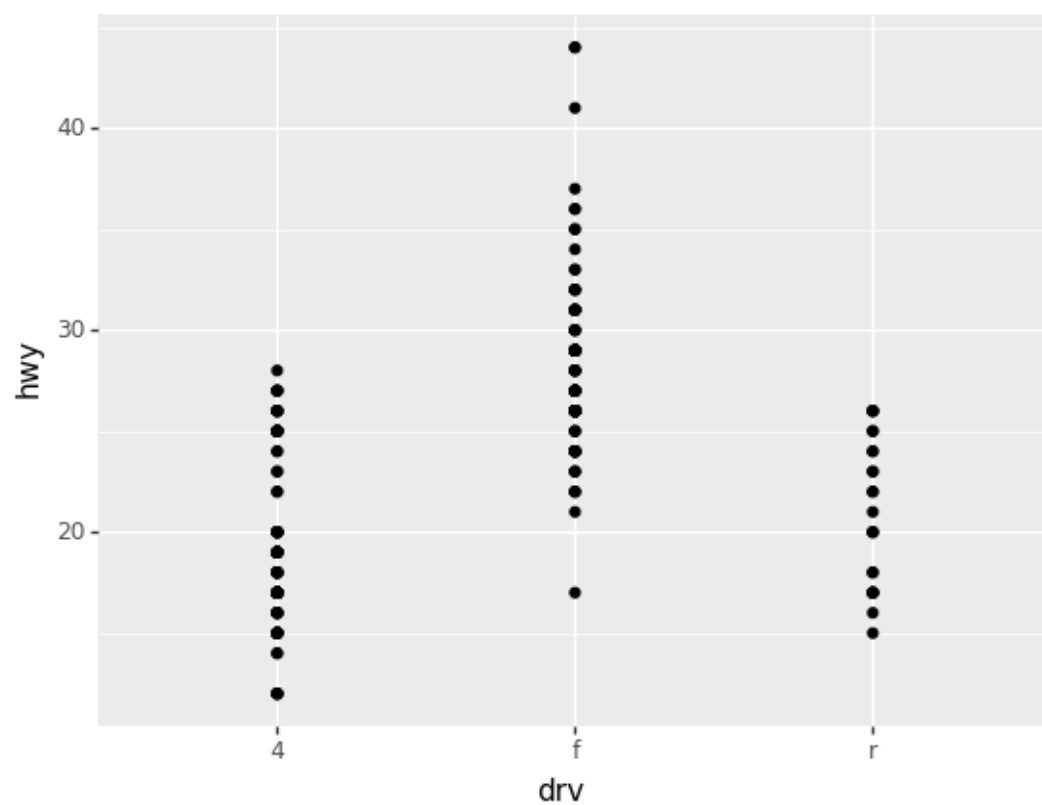
```
(ggplot(mpg, aes(x='displ', y='hwy')) + geom_point(aes(color='factor(drv)')))
+ labs(title='Figure 1', x='Pai Liang', y='You Hao', color='driver')
+ geom_smooth(method='lm'))
```

c:\users\binfang\appdata\local\programs\python\python37\lib\site-packages\numpy\core\fromnumeric.py:2495: FutureWarning
return ptp(axis=axis, out=out, **kwargs)



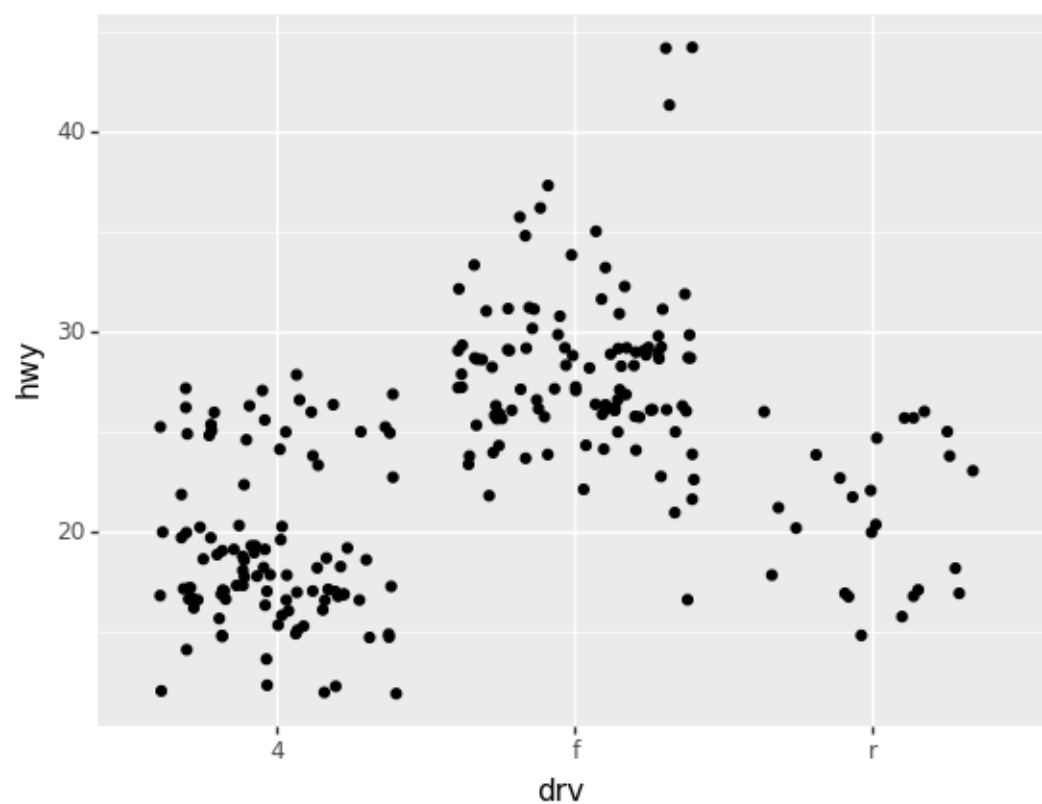
```
Out[33]: <ggplot: (89413356310)>
```

```
In [13]: ggplot(mpg,aes('drv','hwy'))+geom_point()
```



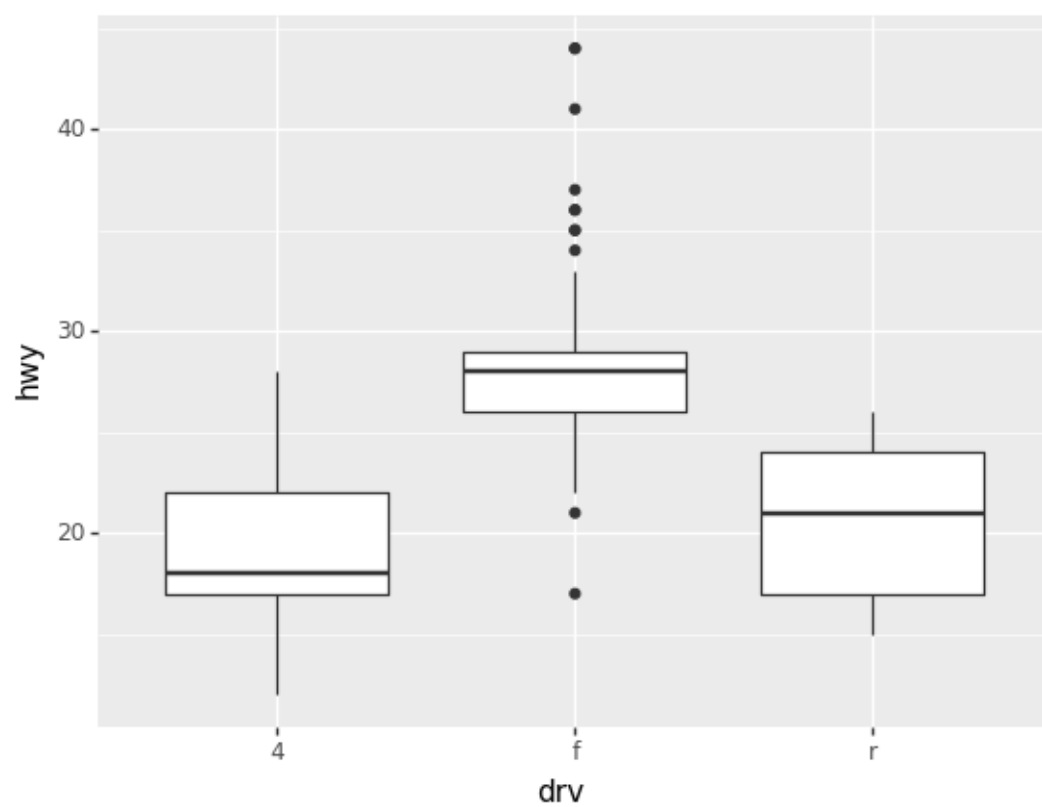
```
Out[13]: <ggplot: (-9223371947454019503)>
```

```
In [14]: # 扰动点图，在数据中加入一些随机噪声  
ggplot(mpg,aes('drv','hwy'))+geom_jitter()
```



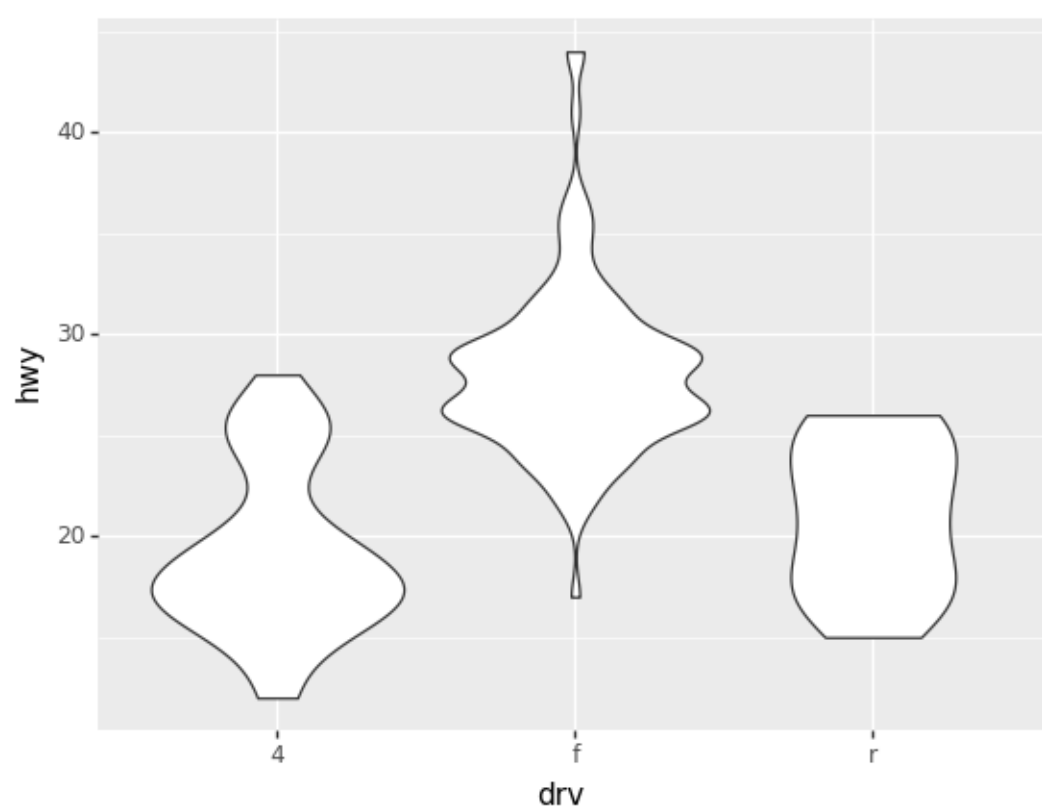
Out[14]: <ggplot: (-9223371947437948363)>

```
In [15]: # 箱线图
# 最大值、75%、中位数、25%、最小值，散点表示异常值
ggplot(mpg,aes('drv','hwy'))+geom_boxplot()
```



Out[15]: <ggplot: (-9223371947437883932)>

```
In [16]: # 小提琴图
# 曲线表示数据的分布
ggplot(mpg,aes('drv','hwy'))+geom_violin()
```

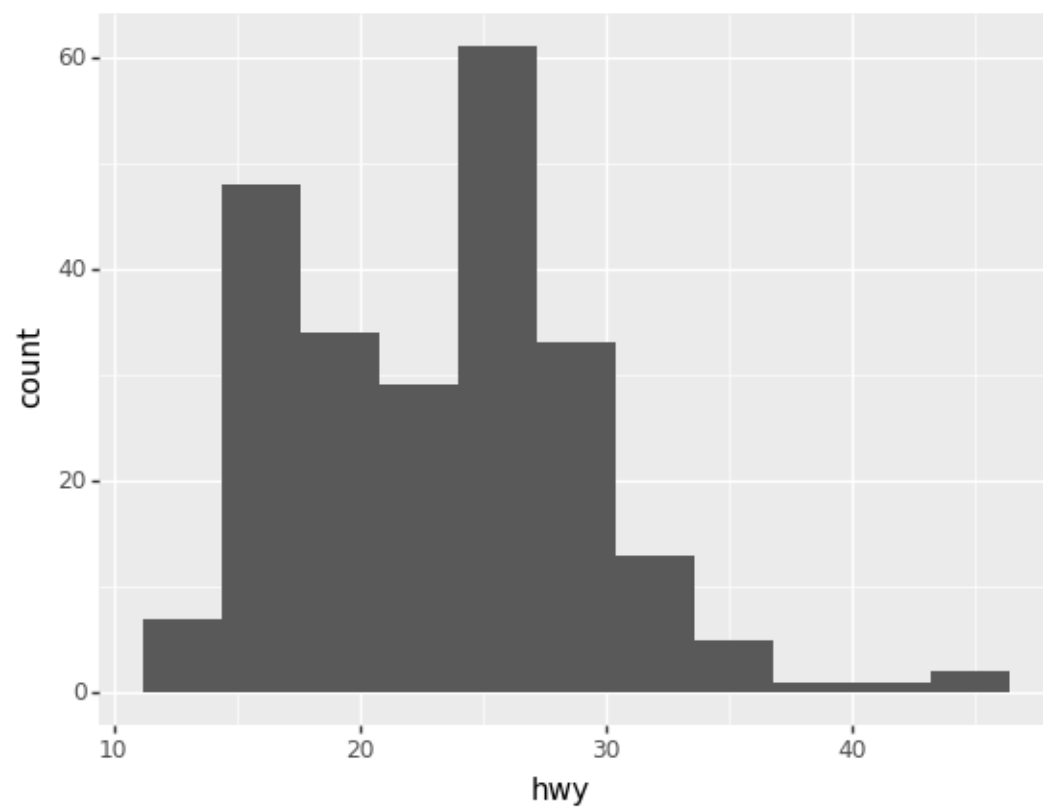


```
Out[16]: <ggplot: (89416828252)>
```

```
In [17]: # 直方图
```

```
ggplot(mpg,aes('hwy'))+geom_histogram()
```

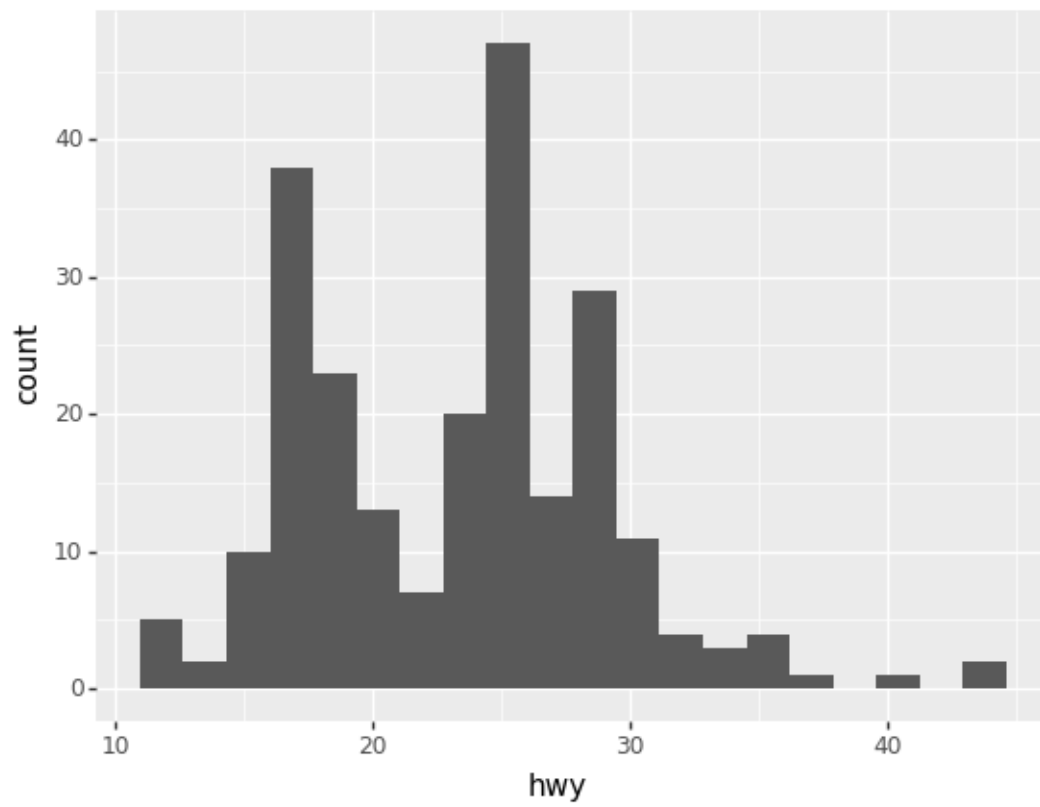
```
c:\users\binfang\appdata\local\programs\python\python37\lib\site-packages\plotnine\stats\stat_bin.py:93: PlotnineWarning: warn(msg.format(params['bins']), PlotnineWarning)
```



```
Out[17]: <ggplot: (-9223371947437831010)>
```

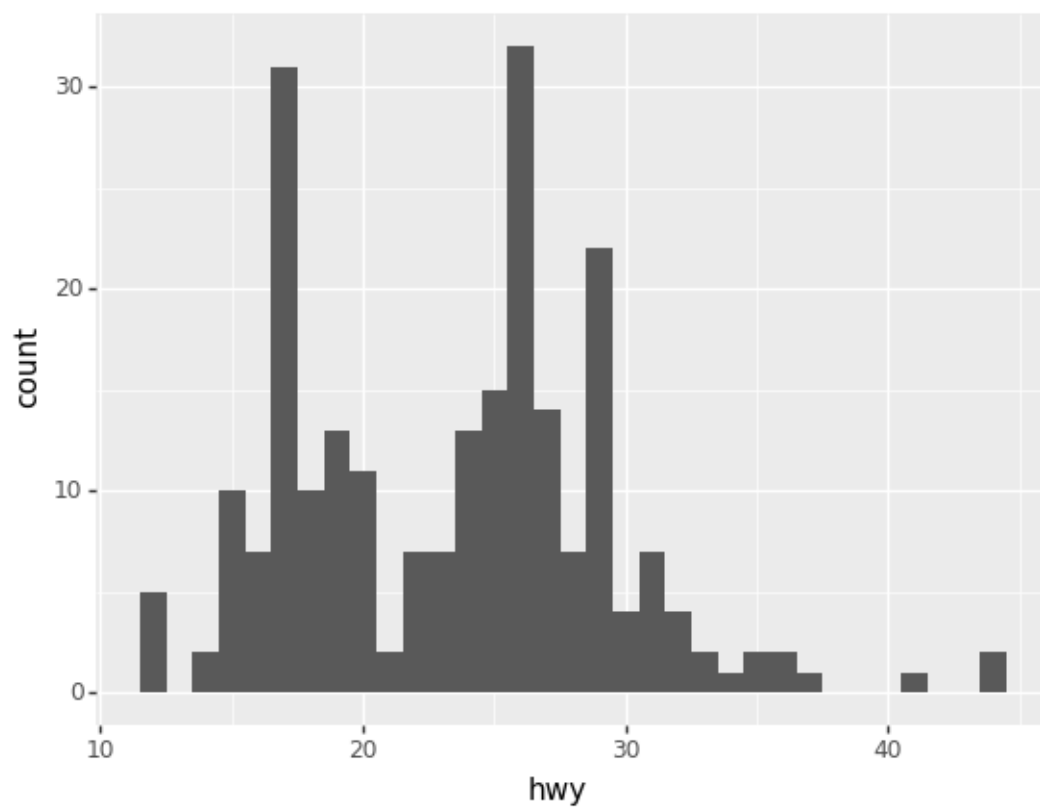
```
In [18]: # 将直方图的组数设置为 20
```

```
ggplot(mpg,aes('hwy'))+geom_histogram(bins=20)
```



Out[18]: <ggplot: (-9223371947437725829)>

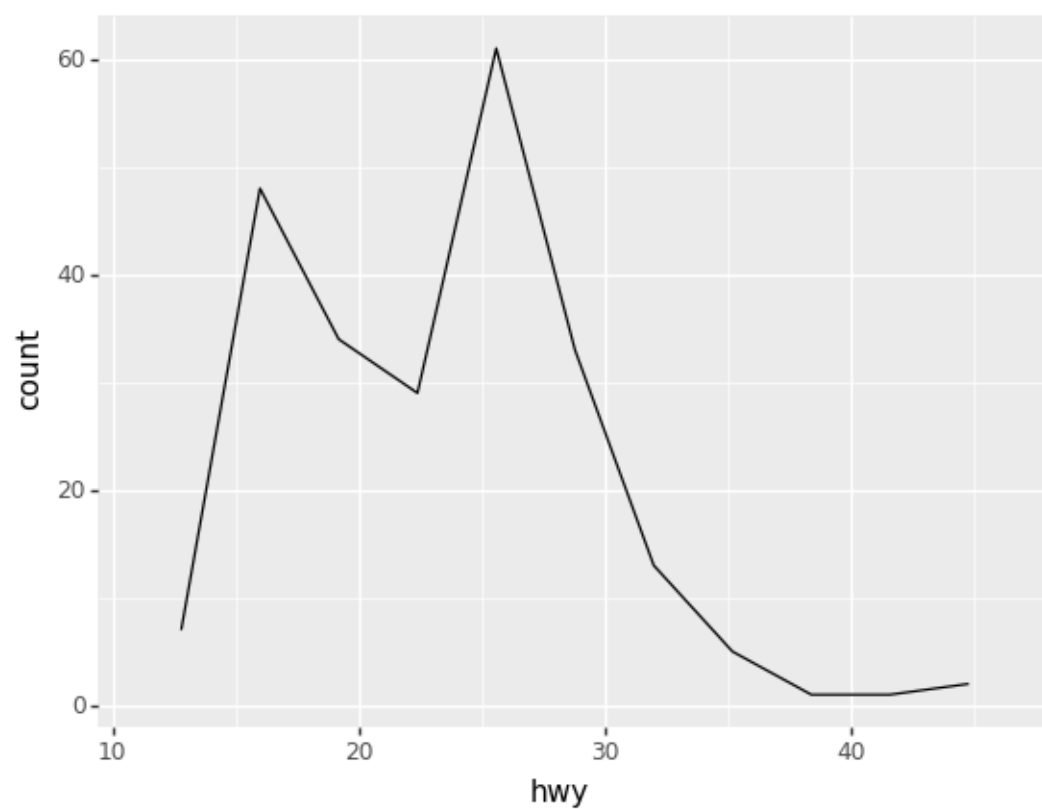
In [20]: # 或者将每组的宽度设置为 1
 ggplot(mpg,aes('hwy'))+geom_histogram(binwidth=1)



Out[20]: <ggplot: (89417049962)>

In [3]: # 频数多边形图
 ggplot(mpg,aes('hwy'))+geom_freqpoly()

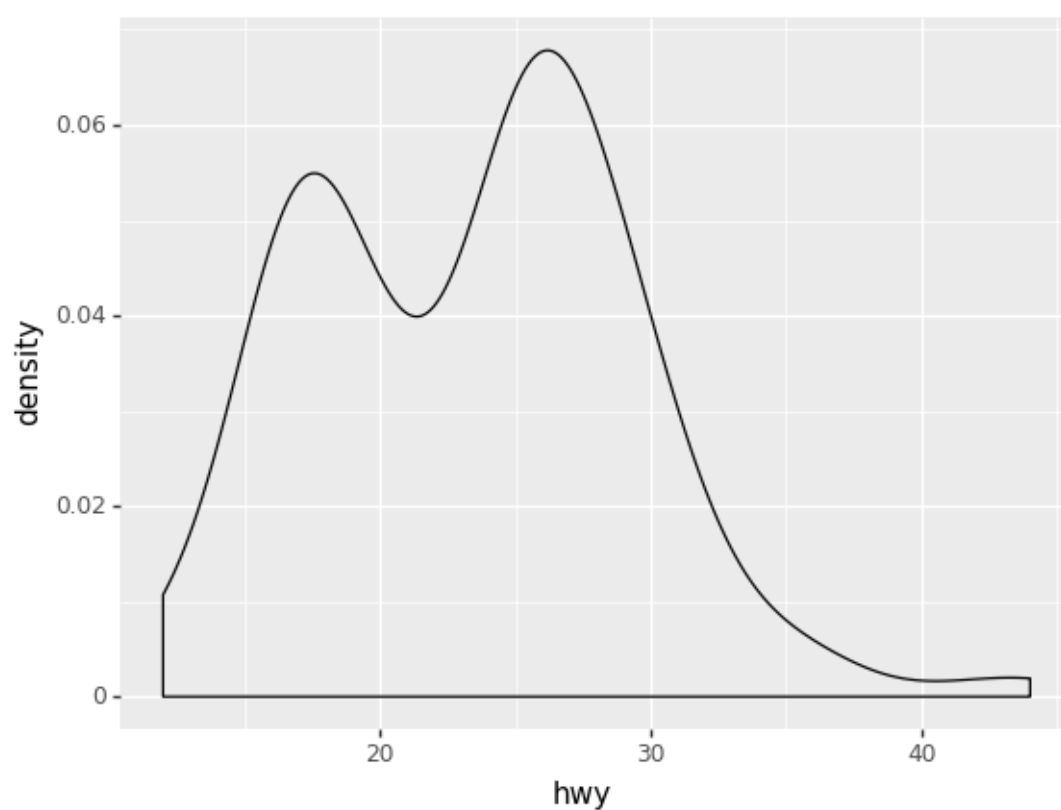
```
c:\users\binfang\appdata\local\programs\python\python37\lib\site-packages\plotnine\stats\stat_bin.py:93: PlotnineWarning: warn(msg.format(params['bins']), PlotnineWarning)
```



```
Out[3]: <ggplot: (128659241792)>
```

```
In [21]: # 密度图
```

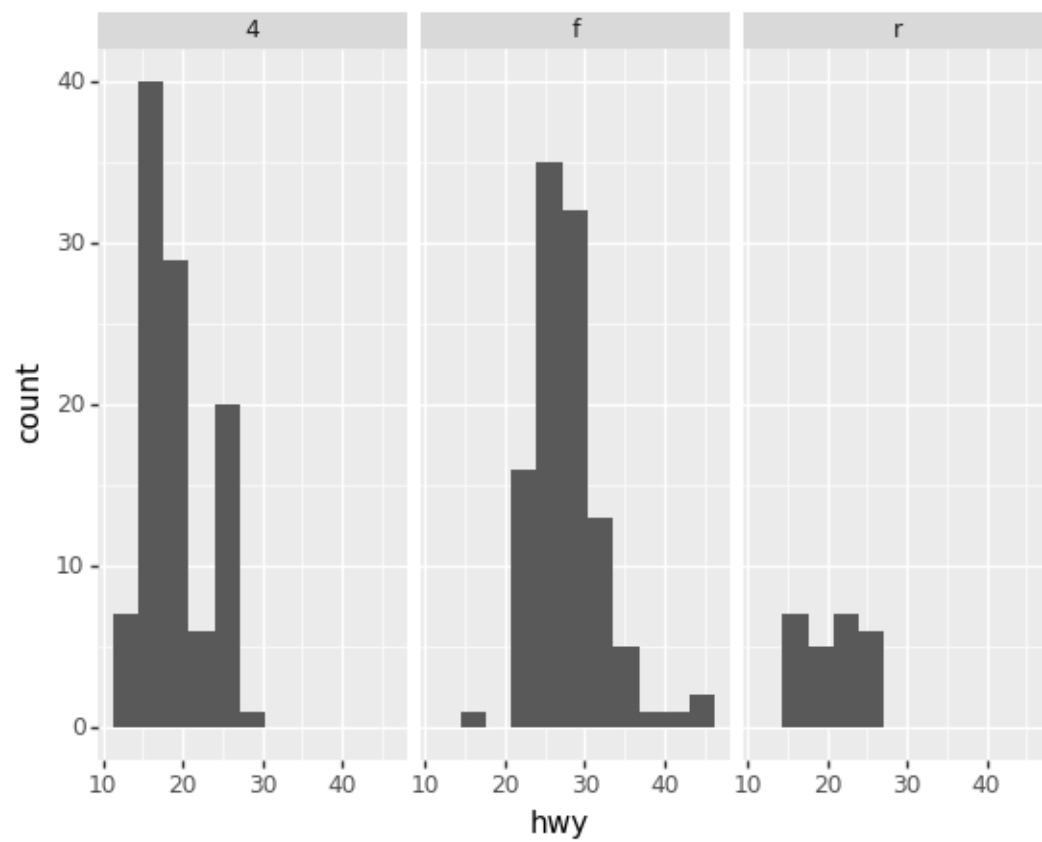
```
ggplot(mpg,aes('hwy'))+geom_density()
```



```
Out[21]: <ggplot: (-9223371947441402827)>
```

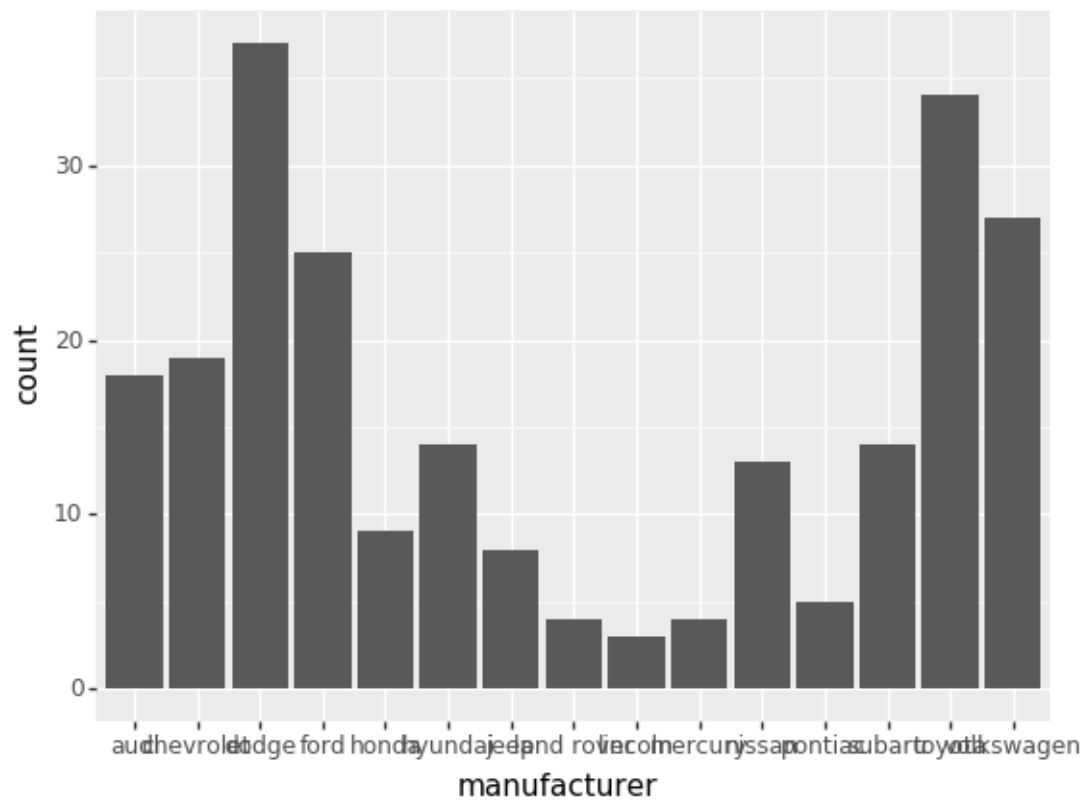
```
In [22]: # 根据变量 drv 分面展示
#facet_wrap 为分面函数
# 注意, 和分组不同, 分面是绘制了三个不同的图, 而不是在一个图上绘制三组数
ggplot(mpg,aes('hwy'))+geom_histogram()+facet_wrap('drv')
```

```
c:\users\binfang\appdata\local\programs\python\python37\lib\site-packages\plotnine\stats\stat_bin.py:93: PlotnineWarning:
warn(msg.format(params['bins']), PlotnineWarning)
```



```
Out[22]: <ggplot: (89413322374)>
```

```
In [23]: # 条形图
ggplot(mpg,aes('manufacturer'))+geom_bar()
```

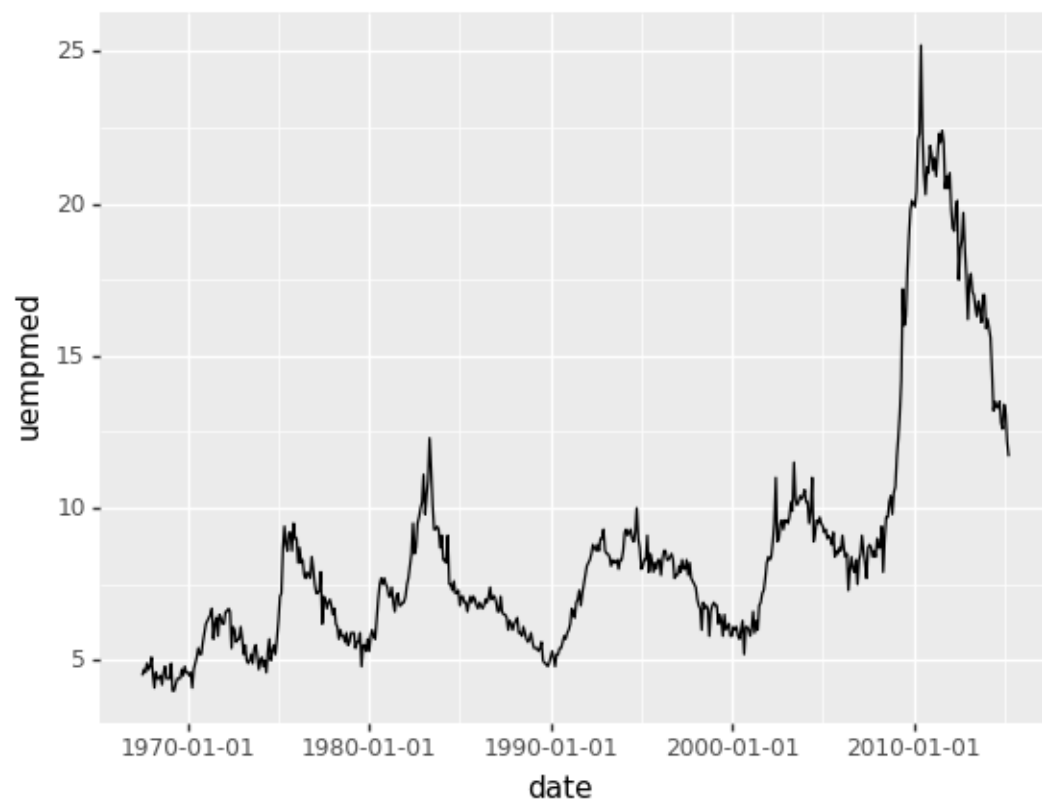



Out[23]: <ggplot: (-9223371947441378188)>

In [24]: # 折线图

年份与失业持续时长的关系图

```
ggplot(economics, aes('date', 'uempmed')) + geom_line()
```



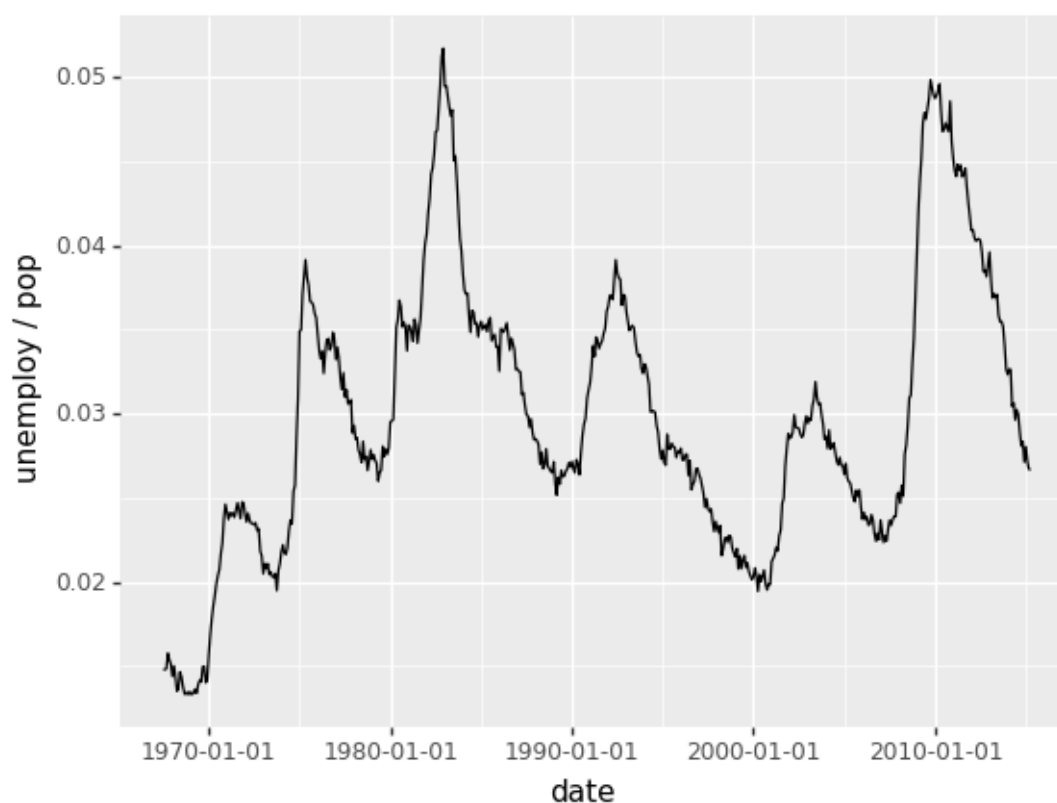
Out[24]: <ggplot: (89413486519)>

In [25]: # 查看 *economics* 这个数据集的信息

```
economics.info()
```

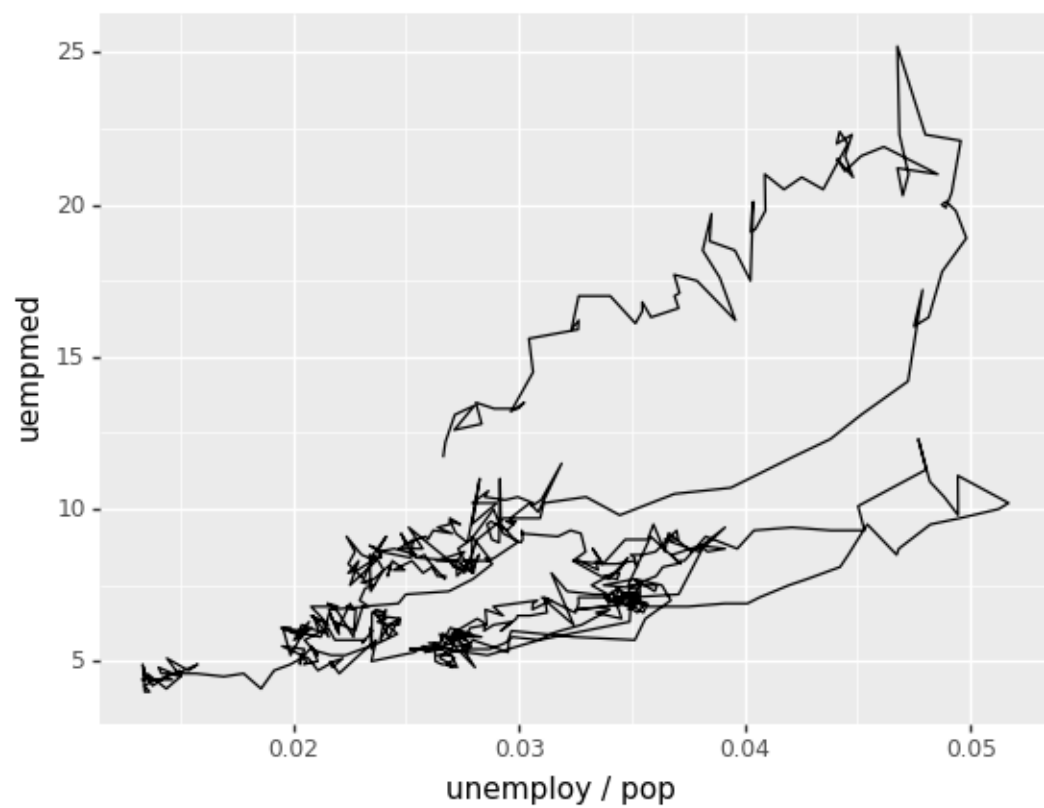
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 574 entries, 0 to 573
Data columns (total 6 columns):
date          574 non-null datetime64[ns]
pce           574 non-null float64
pop           574 non-null int64
psavert       574 non-null float64
uempmed       574 non-null float64
unemploy      574 non-null int64
dtypes: datetime64[ns](1), float64(3), int64(2)
memory usage: 27.0 KB
```

```
In [26]: # 时间与失业率的关系
# 失业率是通过失业人数除以总人口计算出来的
ggplot(economics, aes('date', 'unemploy / pop')) + geom_line()
```



```
Out[26]: <ggplot: (89413392739)>
```

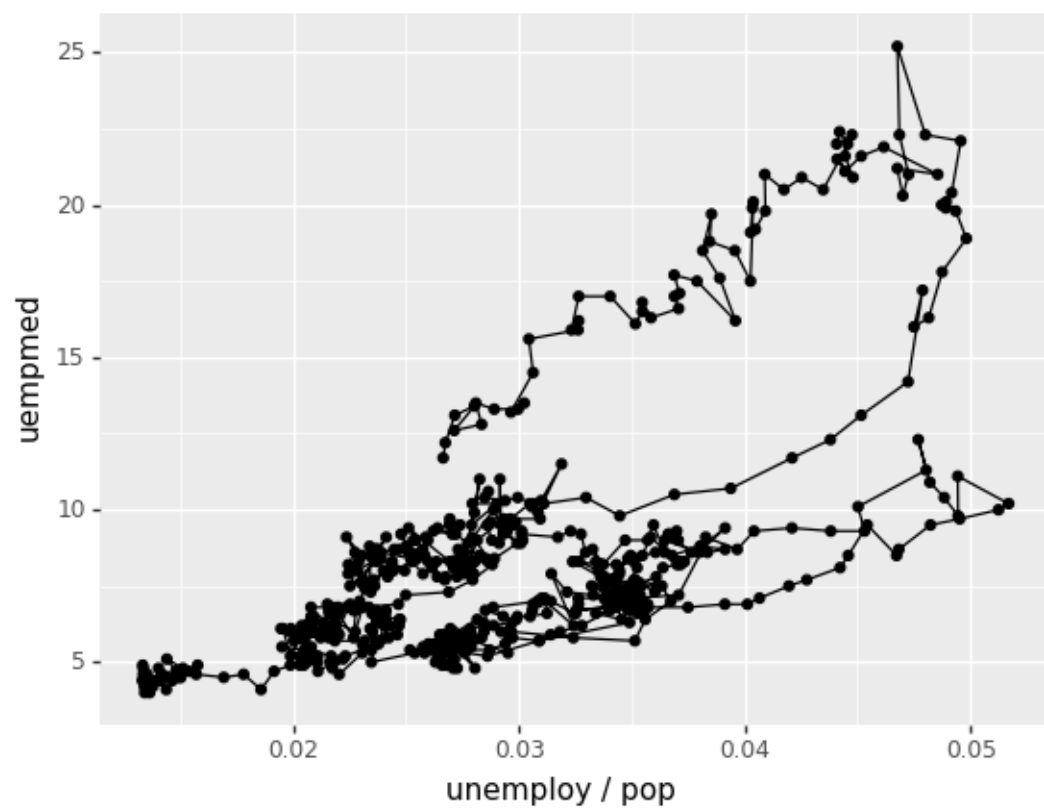
```
In [27]: # 失业率与失业时长的关系
ggplot(economics, aes('unemploy / pop', 'uempmed')) + geom_path()
```



Out [27]: <ggplot: (89416901373)>

In [28]: # 添加点

```
ggplot(economics, aes('unemploy / pop', 'uempmed')) + geom_path()+geom_point()
```

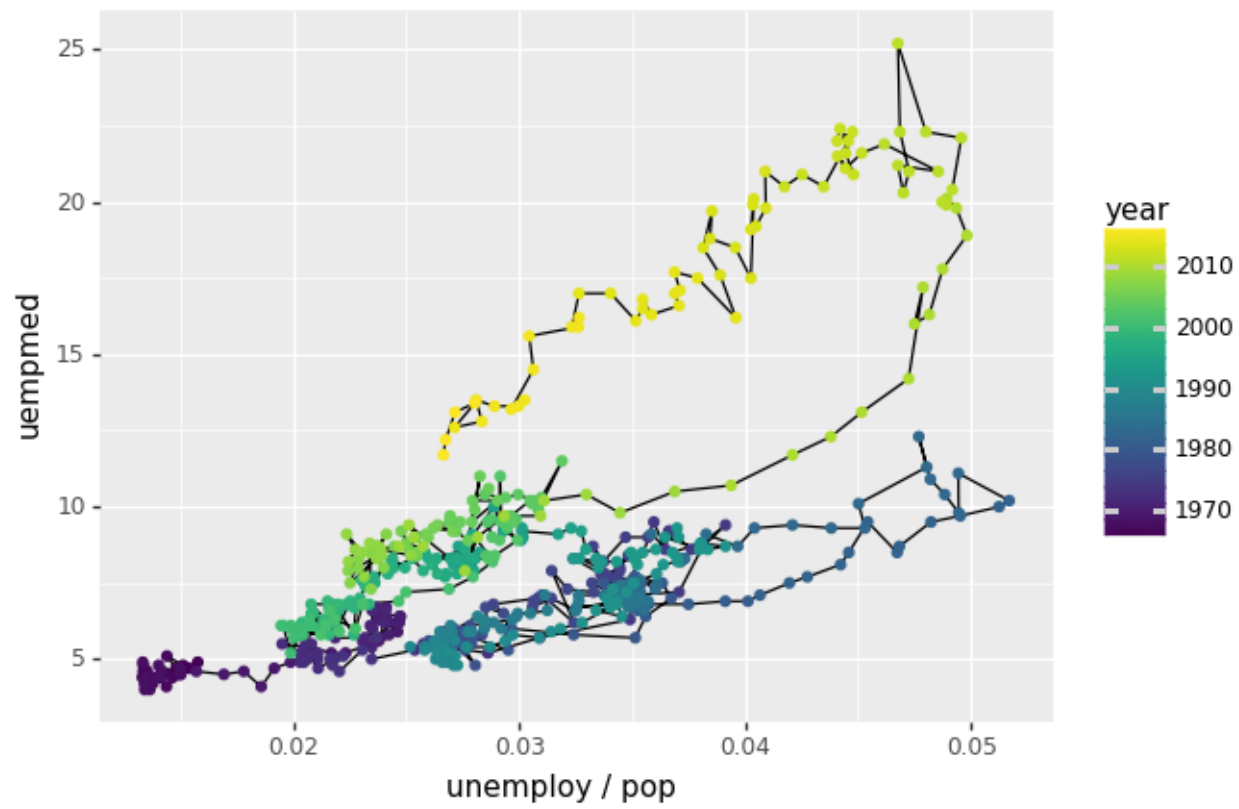


Out [28]: <ggplot: (-9223371947437936331)>

In [29]: # 为不同年份着不同颜色

因为年份是一个连续的变量，所以颜色会有一个渐变的过程，这样更加容易理解图

```
econ2 = economics.join(economics['date'].dt.year.rename('year', inplace = True))
ggplot(econ2, aes('unemploy / pop', 'uempmed')) + geom_path()+geom_point(aes(color='year'))
```



Out[29]: <ggplot: (89413503054)>