

Aggregate your data by category

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON



Stefan Jansen
Instructor

Summarize numeric data by category

- So far: Summarize individual variables
- Compute descriptive statistic like mean, quantiles
- Split data into groups, then summarize groups
- Examples:
 - Largest company by exchange
 - Median market capitalization per IPO year
 - Average market capitalization per sector

Group your data by sector

```
nasdaq.info()
```

```
RangeIndex: 3167 entries, 0 to 3166  
Data columns (total 7 columns):  
Stock Symbol      3167 non-null object  
Company Name      3167 non-null object  
Last Sale         3165 non-null float64  
Market Capitalization  3167 non-null float64  
IPO Year          1386 non-null float64  
Sector            2767 non-null object  
Industry          2767 non-null object  
dtypes: float64(3), object(4)  
memory usage: 173.3+ KB
```

Group your data by sector

```
nasdaq['market_cap_m'] = nasdaq['Market Capitalization'].div(1e6)
nasdaq = nasdaq.drop('Market Capitalization', axis=1) # Drop column
nasdaq_by_sector = nasdaq.groupby('Sector') # Create groupby object
for sector, data in nasdaq_by_sector:
    print(sector, data.market_cap_m.mean())
```

```
Basic Industries 724.899933858
Capital Goods 1511.23737278
Consumer Durables 839.802606627
Consumer Non-Durables 3104.05120552
...
Public Utilities 2357.86531507
Technology 10883.4342135
Transportation 2869.66000673
```

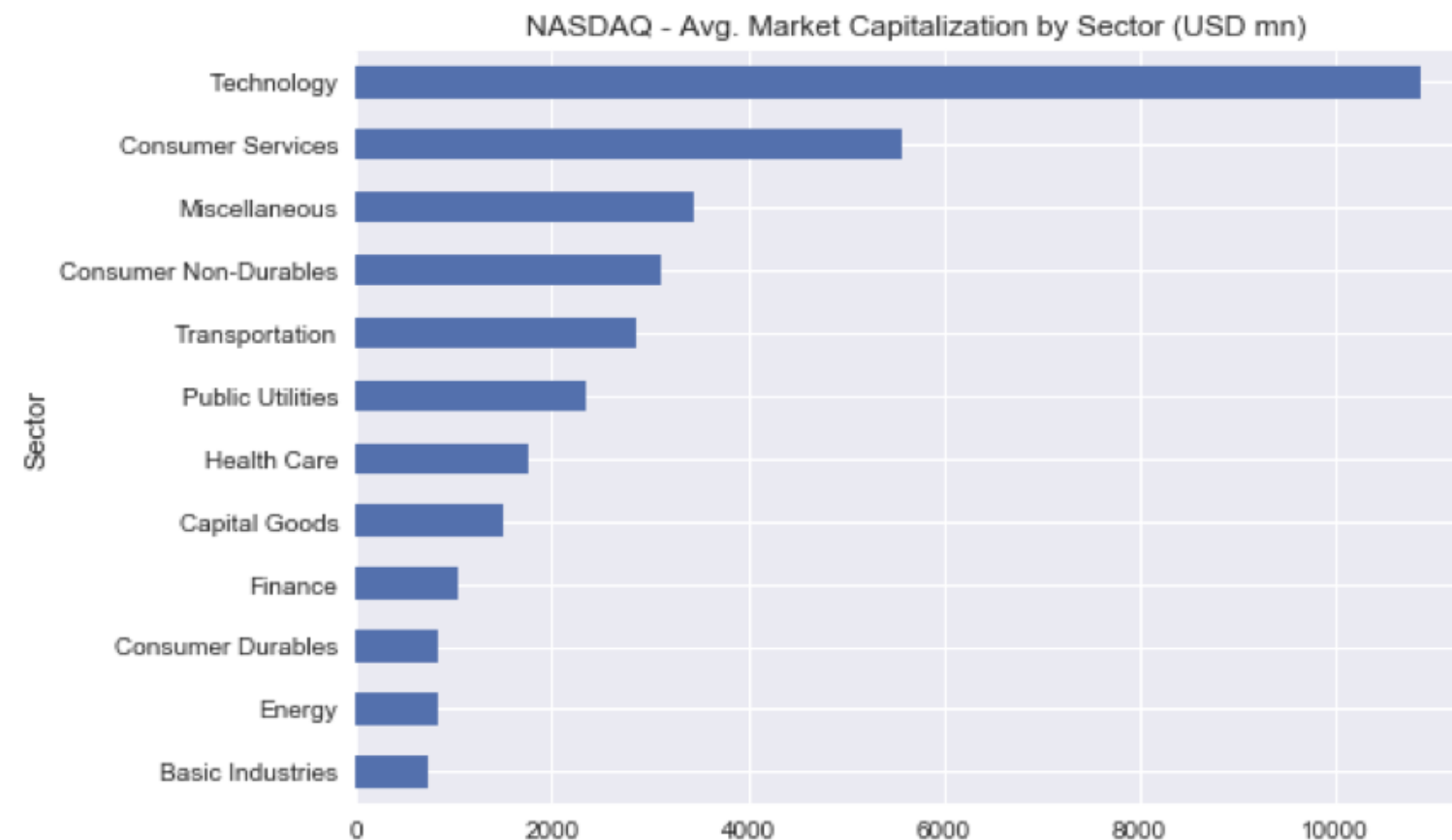
Keep it simple and skip the loop

```
mcap_by_sector = nasdaq_by_sector.market_cap_m.mean()  
mcap_by_sector
```

```
Sector  
Basic Industries          724.899934  
Capital Goods            1511.237373  
Consumer Durables        839.802607  
Consumer Non-Durables    3104.051206  
Consumer Services        5582.344175  
Energy                   826.607608  
Finance                  1044.090205  
Health Care              1758.709197  
...
```

Visualize category summaries

```
title = 'NASDAQ = Avg. Market Cap by Sector'  
mcap_by_sector.plot(kind='barh', title=title)  
plt.xlabel('USD mn')
```



Aggregate summary for all numeric columns

```
nasdaq_by_sector.mean()
```

| | Last Sale | IPO Year | market_cap_m |
|-----------------------|-----------|-------------|--------------|
| Sector | | | |
| Basic Industries | 21.597679 | 2000.766667 | 724.899934 |
| Capital Goods | 26.188681 | 2001.324675 | 1511.237373 |
| Consumer Durables | 24.363391 | 2003.222222 | 839.802607 |
| Consumer Non-Durables | 25.749565 | 2000.609756 | 3104.051206 |
| Consumer Services | 34.917318 | 2004.104575 | 5582.344175 |
| Energy | 15.496834 | 2008.034483 | 826.607608 |
| Finance | 29.644242 | 2010.321101 | 1044.090205 |
| Health Care | 19.462531 | 2009.240409 | 1758.709197 |
| Miscellaneous | 46.094369 | 2004.333333 | 3445.655935 |
| Public Utilities | 18.643705 | 2006.040000 | 2357.865315 |
| ... | | | |

Let's practice!

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON

More ways to aggregate your data

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON



Stefan Jansen
Instructor

Many ways to aggregate

- Last segment: Group by one variable and aggregate
- More detailed ways to summarize your data:
 - Group by two or more variables
 - Apply multiple aggregations
- Examples
 - Median market cap by sector and IPO year
 - Mean & standard deviation of stock price by year

Several aggregations by category

```
nasdaq['market_cap_m'] = nasdaq['Market Capitalization'].div(1e6)
by_sector = nasdaq.groupby('Sector')
by_sector.market_cap_m.agg(['size', 'mean']).sort_values('size')
```

| Sector | size | mean |
|-------------------|------|--------------|
| Transportation | 52 | 2869.660007 |
| Energy | 66 | 826.607608 |
| Public Utilities | 66 | 2357.865315 |
| Basic Industries | 78 | 724.899934 |
| ... | | |
| Consumer Services | 348 | 5582.344175 |
| Technology | 433 | 10883.434214 |
| Finance | 627 | 1044.090205 |
| Health Care | 645 | 1758.709197 |

Several aggregations plus new labels

```
by_sector.market_cap_m.agg({'#0bs': 'size', 'Average': 'mean'})
```

| Sector | #0bs | Average |
|-----------------------|------|--------------|
| Basic Industries | 78 | 724.899934 |
| Capital Goods | 172 | 1511.237373 |
| Consumer Durables | 88 | 839.802607 |
| Consumer Non-Durables | 103 | 3104.051206 |
| Consumer Services | 348 | 5582.344175 |
| ... | | |
| Health Care | 645 | 1758.709197 |
| Miscellaneous | 89 | 3445.655935 |
| Public Utilities | 66 | 2357.865315 |
| Technology | 433 | 10883.434214 |
| Transportation | 52 | 2869.660007 |

Different statistics by column

```
by_sector.agg({'market_cap_m': 'size', 'IPO Year': 'median'})
```

| Sector | market_cap_m | IPO Year |
|-----------------------|--------------|----------|
| Basic Industries | 78 | 1972.0 |
| Capital Goods | 172 | 1972.0 |
| Consumer Durables | 88 | 1983.0 |
| Consumer Non-Durables | 103 | 1972.0 |
| Consumer Services | 348 | 1981.0 |
| ... | | |
| Health Care | 645 | 1981.0 |
| Miscellaneous | 89 | 1987.0 |
| Public Utilities | 66 | 1981.0 |
| Technology | 433 | 1972.0 |
| Transportation | 52 | 1986.0 |

Aggregate by two categories

```
by_sector_year = nasdaq.groupby(['Sector', 'IPO Year'])  
by_sector_year.market_cap_m.mean()
```

| Sector | IPO Year | |
|------------------|----------|-------------|
| Basic Industries | 1972.0 | 877.240005 |
| | 1973.0 | 1445.697371 |
| | 1986.0 | 1396.817381 |
| Transportation | ... | |
| | 1986.0 | 1176.179710 |
| | 1991.0 | 6646.778622 |
| | 1992.0 | 56.074572 |
| | ... | |
| | 2009.0 | 552.445919 |
| | 2011.0 | 3711.638317 |
| | 2013.0 | 125.740421 |

Select from MultiIndex

```
mcap_sector_year = by_sector_year.market_cap_m.mean()  
mcap_sect_year.loc['Basic Industries']
```

```
IPO Year  
1972.0      877.240005  
1973.0     1445.697371  
1986.0     1396.817381  
1988.0       24.847526  
...  
2012.0      381.796074  
2013.0       22.661533  
2015.0      260.075564  
2016.0       81.288336  
Name: market_cap_m, dtype: float64
```

Select from MultiIndex

```
mcap_sect_year.loc[['Basic Industries', 'Transportation']]
```

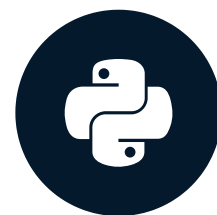
| Sector | IPO Year | |
|------------------|----------|-------------|
| Basic Industries | 1972.0 | 877.240005 |
| | 1973.0 | 1445.697371 |
| | 1986.0 | 1396.817381 |
| | ... | |
| Transportation | 1986.0 | 1176.179710 |
| | 1991.0 | 6646.778622 |
| | 1992.0 | 56.074572 |
| | ... | |

Let's practice!

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON

Summary statistics by category with seaborn

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON



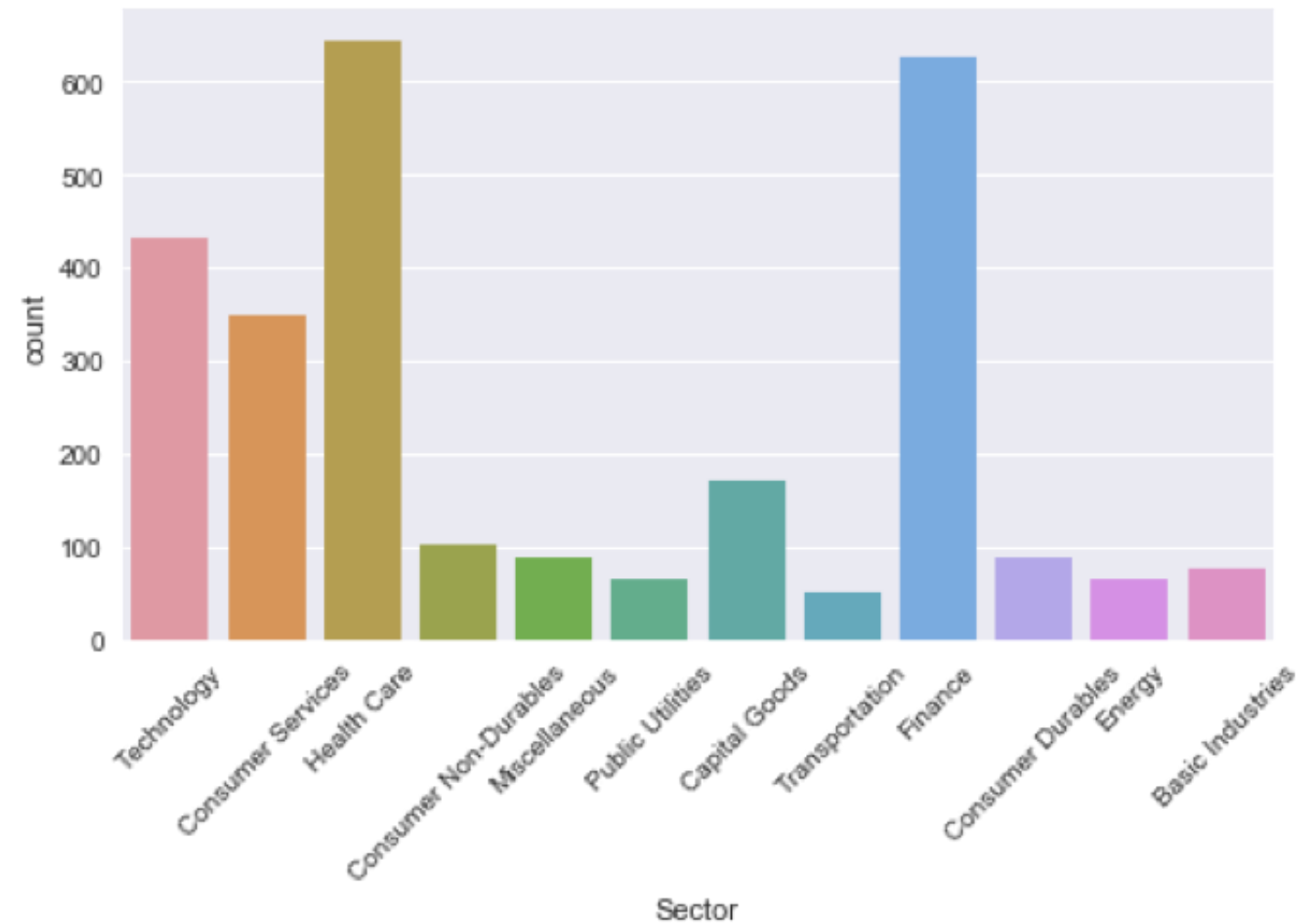
Stefan Jansen
Instructor

Categorical plots with seaborn

- Specialized ways to plot combinations of categorical and numerical variables
- Visualize estimates of summary statistics per category
- Understand how categories impact numerical variables
- Compare using key metrics of distributional characteristics
- Example: Mean Market Cap per Sector or IPO Year with indication of dispersion

The basics: countplot

```
sns.countplot(x='Sector', data=nasdaq)  
plt.xticks(rotation=45)
```



countplot, sorted

```
sector_size = nasdaq.groupby('Sector').size()
order = sector_size.sort_values(ascending=False)
order.head()
```

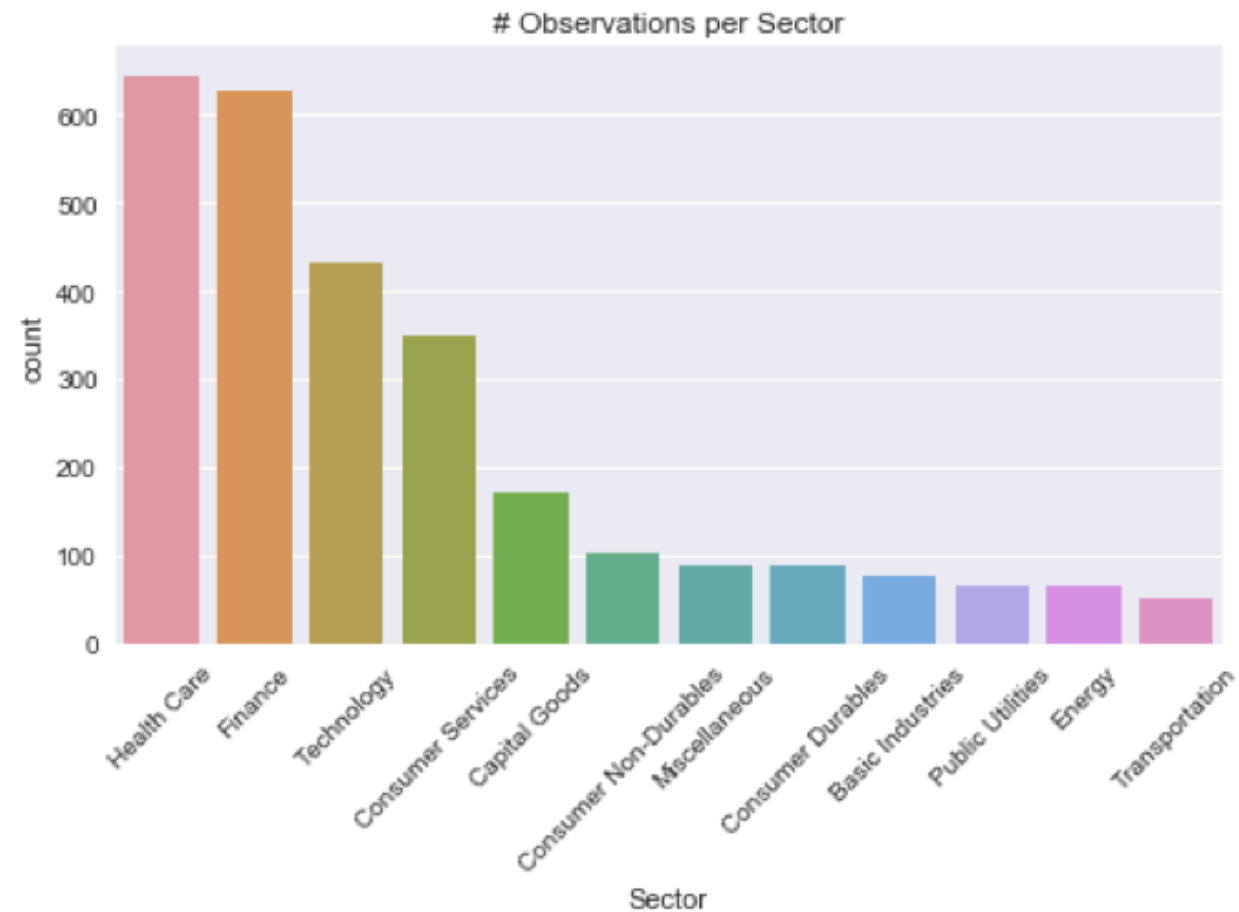
```
Sector
Health Care      645
Finance          627
Technology       433
...
```

```
order = order.index.tolist()
```

```
['Health Care', 'Finance', ..., 'Energy', 'Transportation']
```

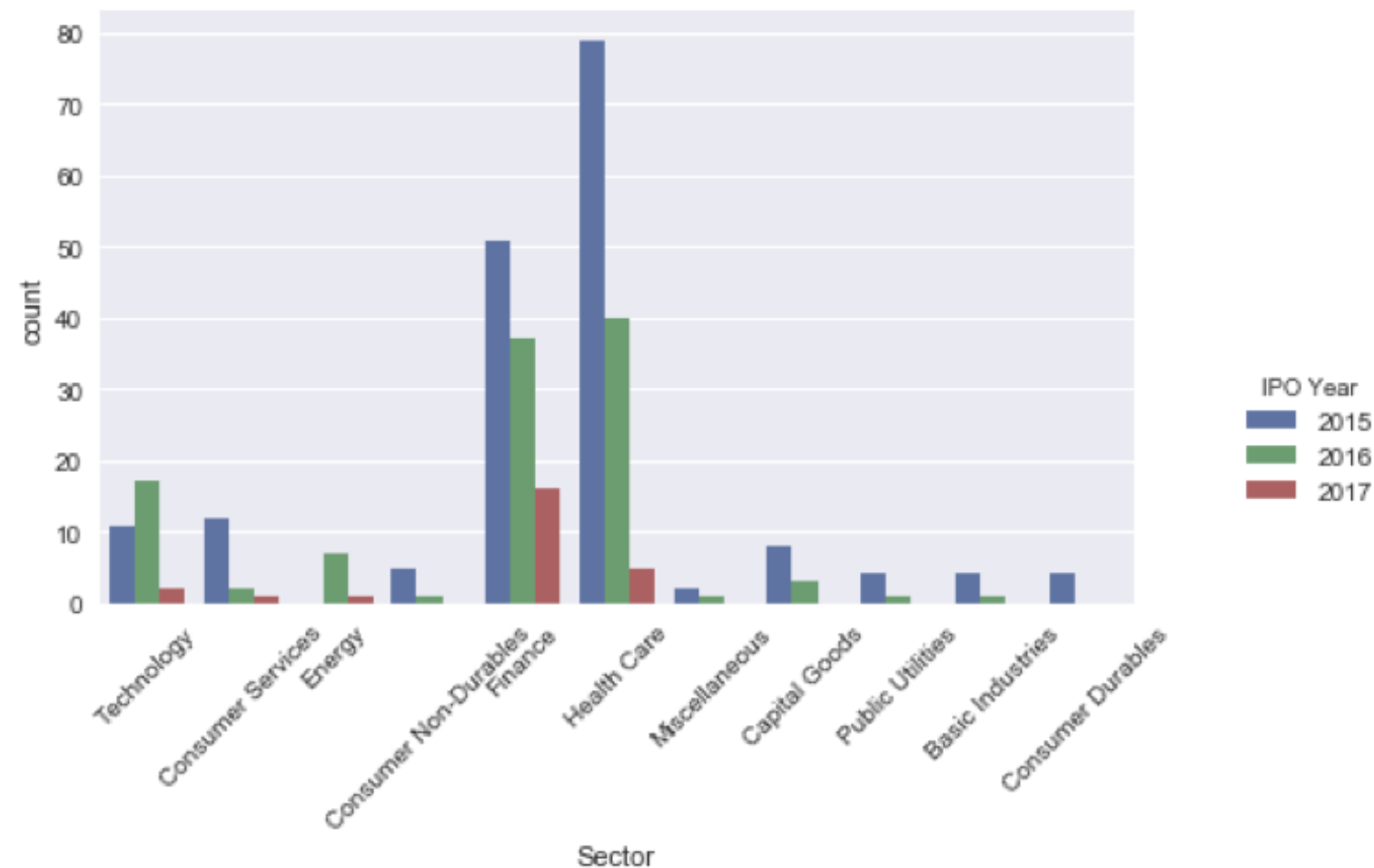
countplot, sorted

```
sns.countplot(x='Sector', data=nasdaq, order=order)
plt.xticks(rotation=45)
plt.title('# Observations per Sector')
```



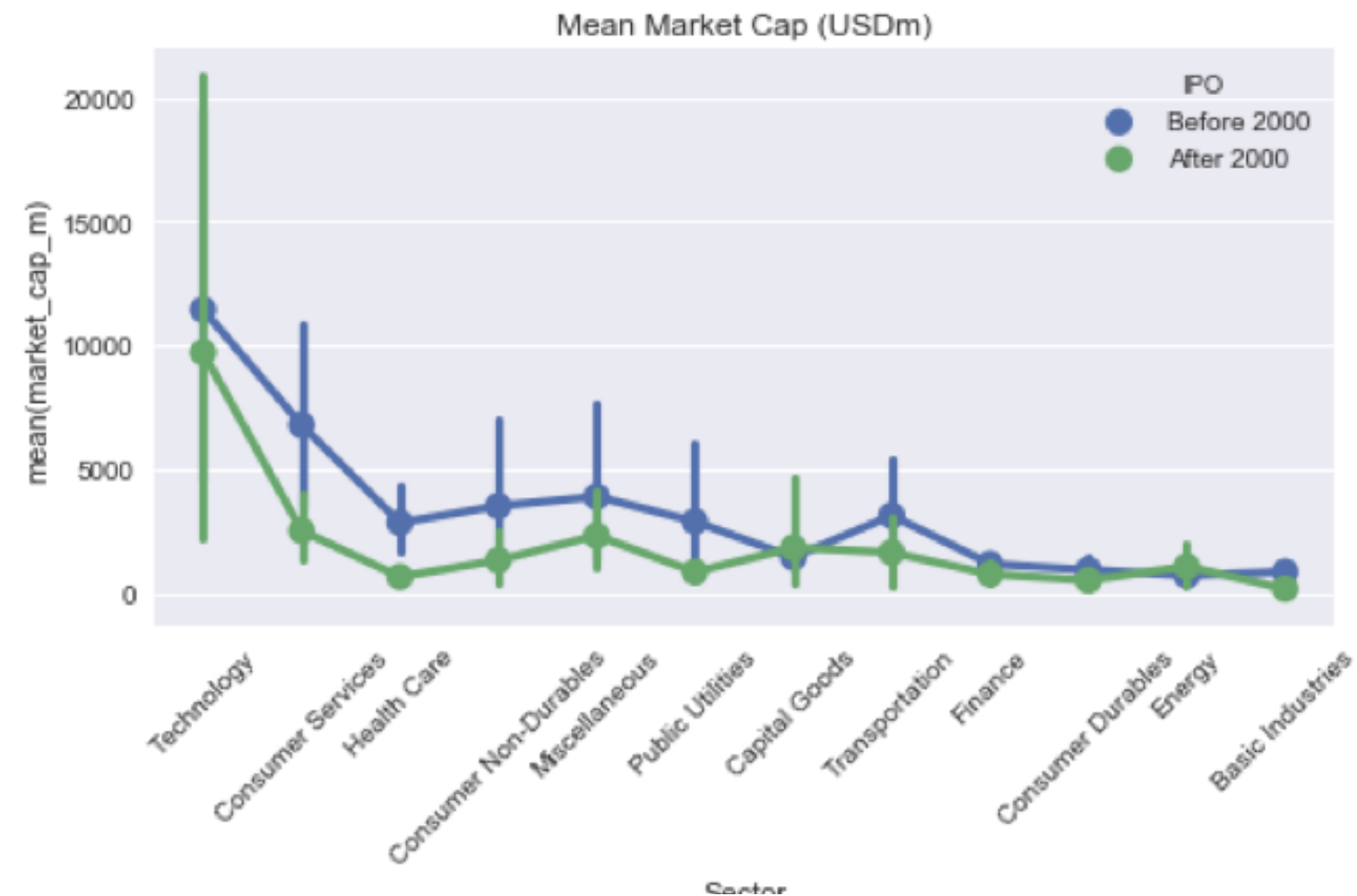
countplot, multiple categories

```
recent_ipos = nasdaq[nasdaq['IPO Year'] > 2014]
recent_ipos['IPO Year'] = recent_ipos['IPO Year'].astype(int)
sns.countplot(x='Sector', hue='IPO Year', data=recent_ipos)
```



Compare stats with PointPlot

```
nasdaq['IPO'] = nasdaq['IPO Year'].apply(lambda x: 'After 2000' if x > 2000 else 'Before 2000')
sns.pointplot(x='Sector', y='market_cap_m', hue='IPO', data=nasdaq)
plt.xticks(rotation=45); plt.title('Mean Market Cap')
```



Let's practice!

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON

Distributions by category with seaborn

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON



Stefan Jansen
Instructor

Distributions by category

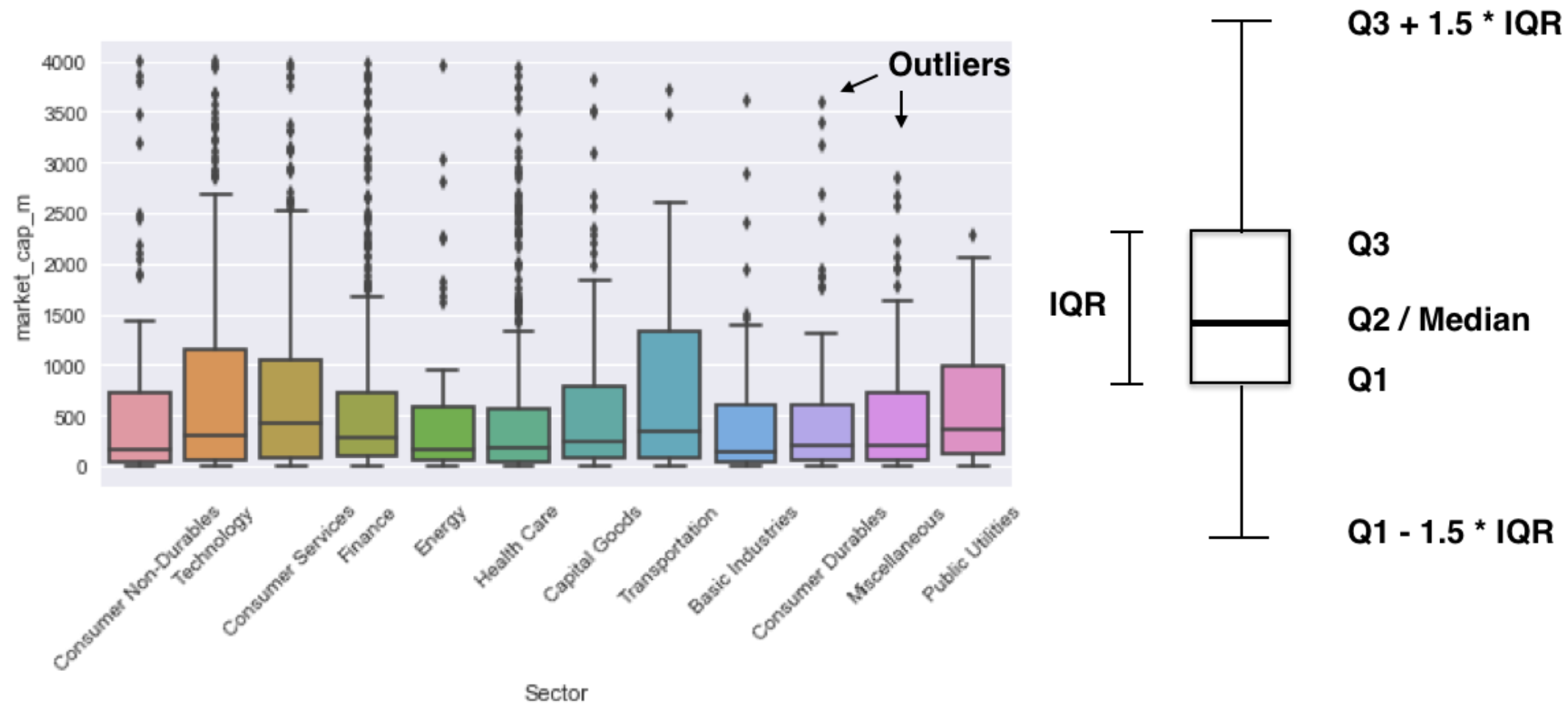
- Last segment: Summary statistics
- Number of observations, mean per category
- Now: Visualize distribution of a variable by levels of a categorical variable to facilitate comparison
- Example: Distribution of Market Cap by Sector or IPO Year
- More detail than summary stats

Clean data: removing outliers

```
nasdaq = pd.read_excel('listings.xlsx', sheetname='nasdaq',  
                      na_values='n/a')  
nasdaq['market_cap_m'] = nasdaq['Market Capitalization'].div(1e6)  
nasdaq = nasdaq[nasdaq.market_cap_m > 0] # Active companies only  
outliers = nasdaq.market_cap_m.quantile(.9) # Outlier threshold  
nasdaq = nasdaq[nasdaq.market_cap_m < outliers] # Remove outliers
```

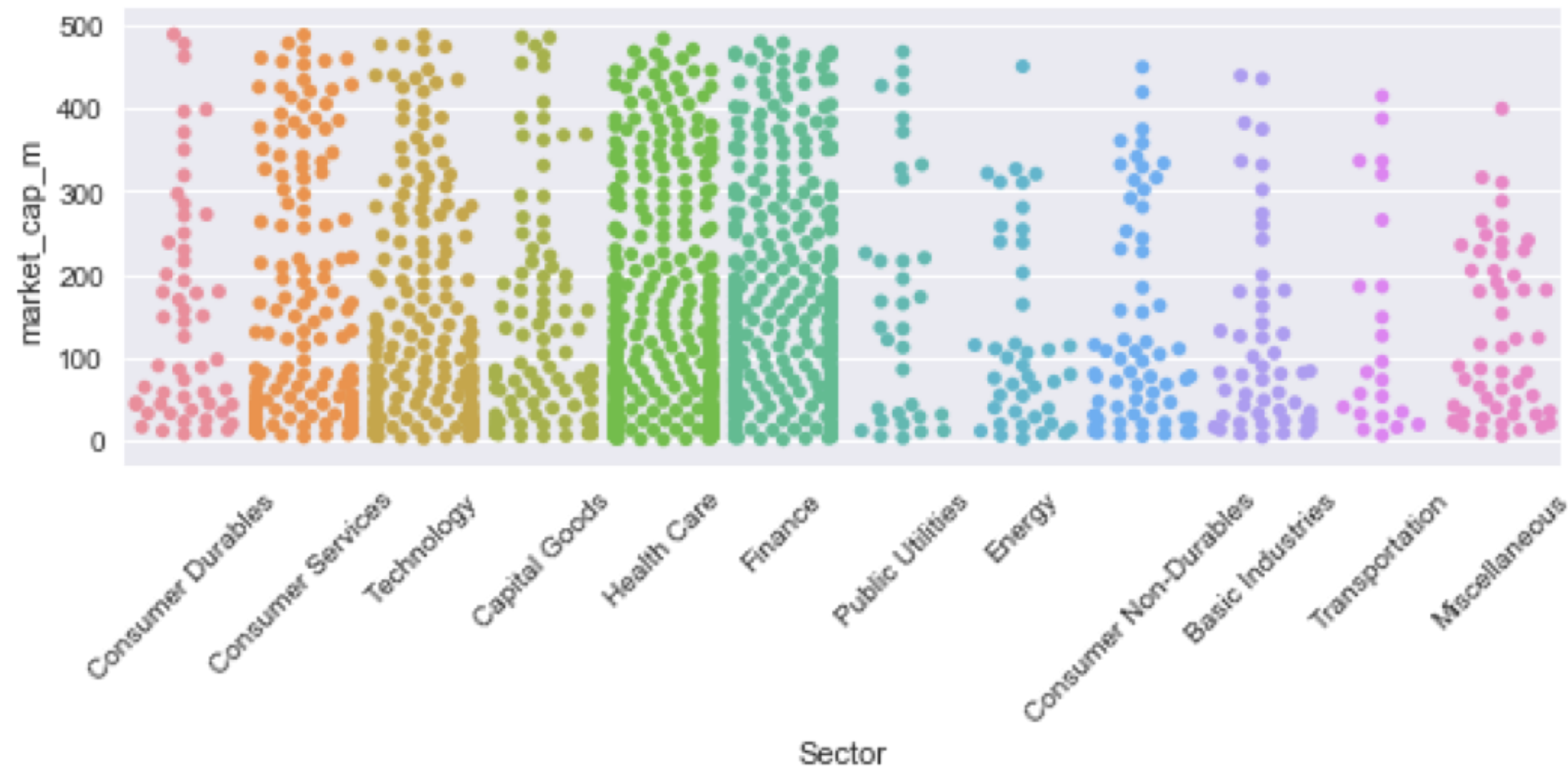
Boxplot: quartiles and outliers

```
import seaborn as sns
sns.boxplot(x='Sector', y='market_cap_m', data=nasdaq)
plt.xticks(rotation=75);
```



A variation: SwarmPlot

```
sns.swarmplot(x='Sector', y='market_cap_m', data=nasdaq)
plt.xticks(rotation=75)
plt.show()
```



Let's practice!

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON

Congratulations!

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON



Stefan Jansen
Instructor

What you learned

- Import data from Excel and online sources
- Combine datasets
- Summarize and aggregate data

Keep learning!

IMPORTING AND MANAGING FINANCIAL DATA IN PYTHON