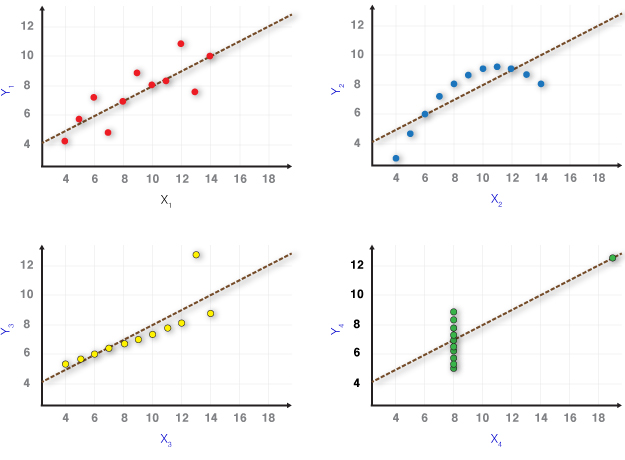
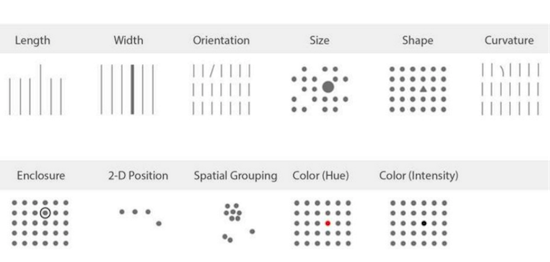
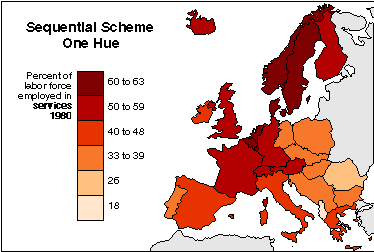
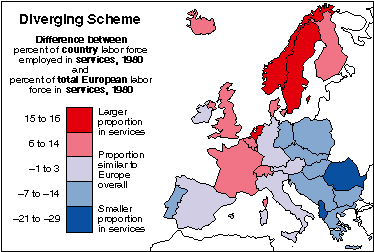
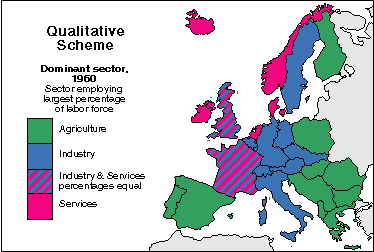
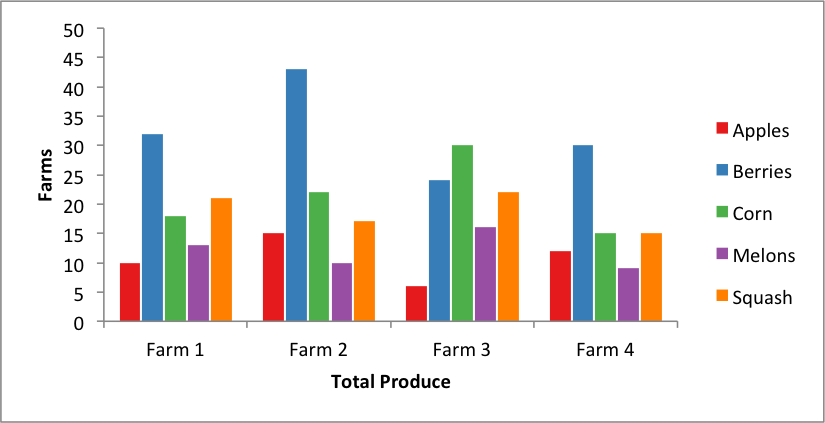
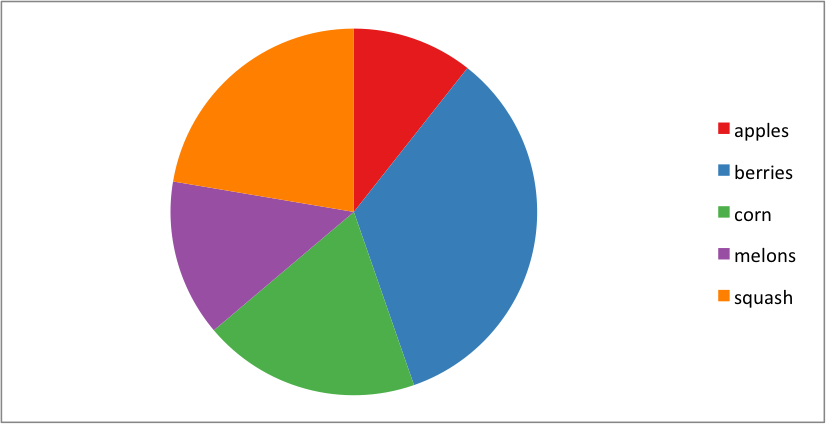
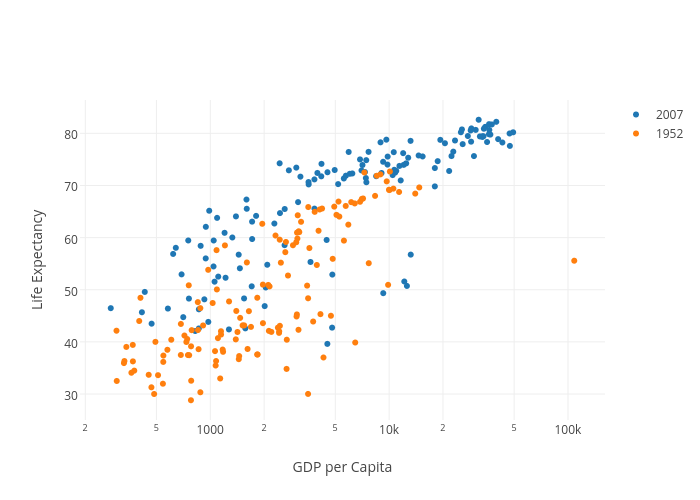
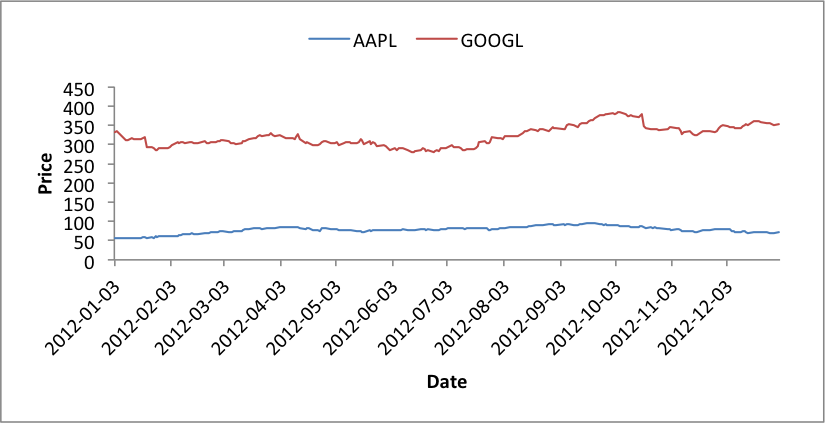
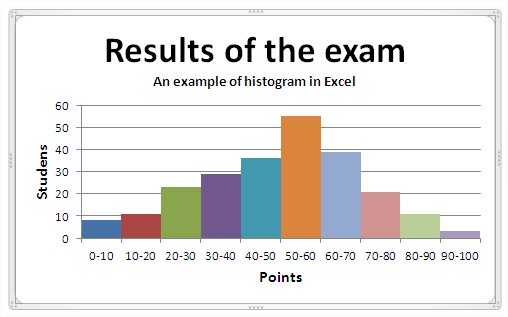
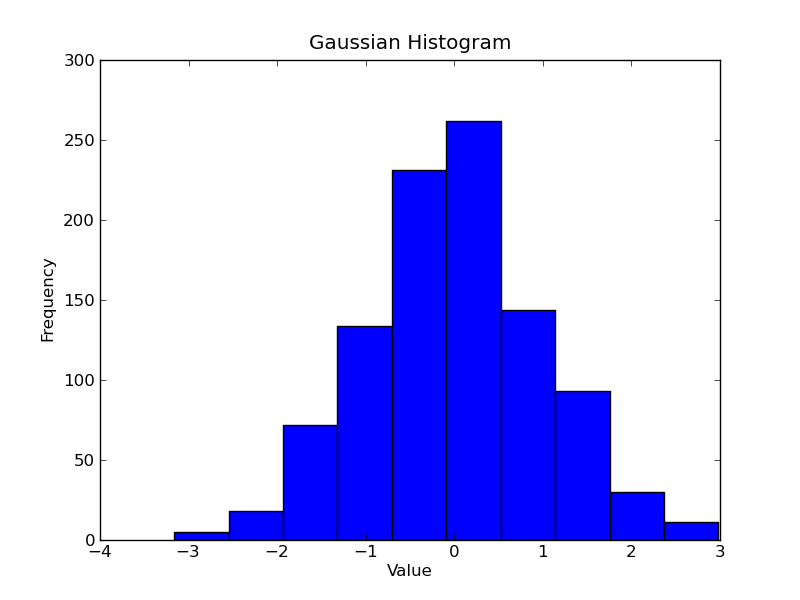
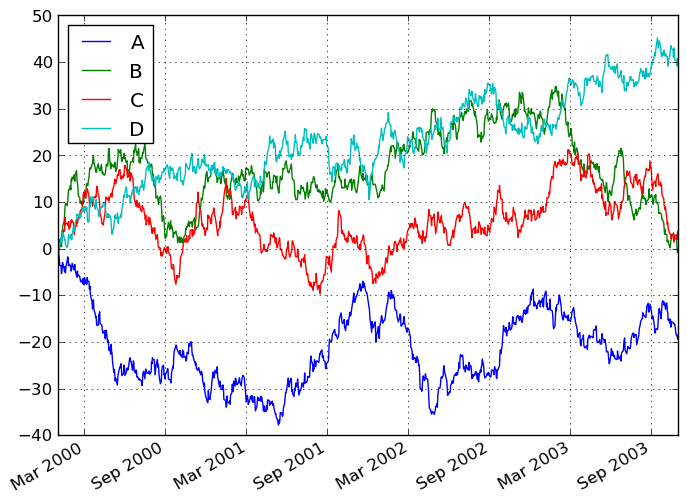
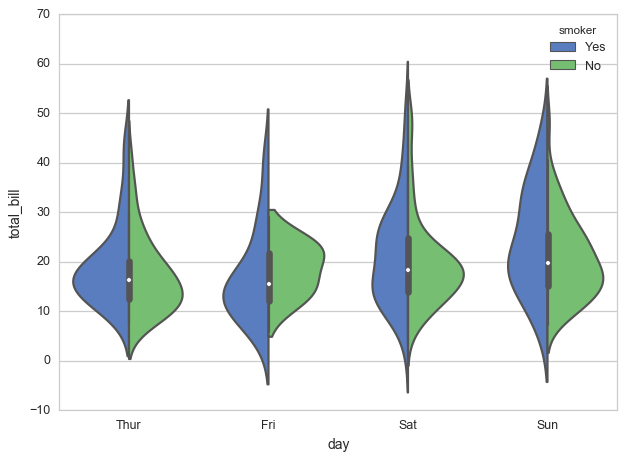
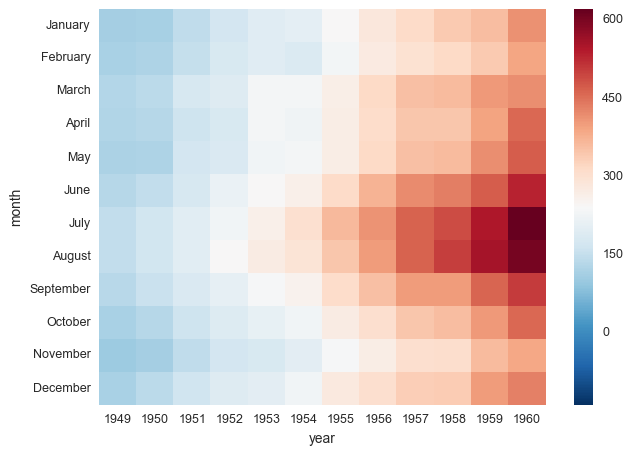
Unit 5-1: Principles of Data Visualizations

* Why Use Data Visualization?
  + Let's start by exploring the classic Anscombe's Quartet data set.
  + Look at this table, which represents four sets of x-y pairs. Do you observe any patterns?
  + I - II - III - IV -
  + x y x y x y x y
  + 10.0 8.04 10.0 9.14 10.0 7.46 8.0 6.58
  + 8.0 6.95 8.0 8.14 8.0 6.77 8.0 5.76
  + 13.0 7.58 13.0 8.74 13.0 12.74 8.0 7.71
  + 9.0 8.81 9.0 8.77 9.0 7.11 8.0 8.84
  + 11.0 8.33 11.0 9.26 11.0 7.81 8.0 8.47
  + 14.0 9.96 14.0 8.110 14.0 8.84 8.0 7.04
  + 6.0 7.24 6.0 6.13 6.0 6.08 8.0 5.25
  + 4.0 4.26 4.0 3.10 4.0 5.39 19.0 12.5
  + 12.0 10.84 12.0 9.13 12.0 8.15 8.0 5.56
  + 7.0 4.82 7.0 7.26 7.0 6.42 8.0 7.91
  + 5.0 5.68 5.0 4.74 5.0 5.73 8.0 6.89
  + Probably not. Because of the way the human brain processes information, charts or graphs that visualize large amounts of complex data are easier to understand than spreadsheets or reports.
  + Many people use summary statistics to deal with this, but can you think of any issues that may arise?
  + Here are the summary statistics for our data set:
  + Set sum X sum Y avg X avg Y stdev X stdev Y
  + I 99.0 82.5 9.0 7.5 3.32 2.03
  + II 99.0 82.5 9.0 7.5 3.32 2.03
  + III 99.0 82.5 9.0 7.5 3.32 2.03
  + IV 99.0 82.5 9.0 7.5 3.32 2.03
  + Although the data in our tables vary greatly across the sets, each of them yield the exact same summary statistics.
  + Because it's hard to make sense of the raw data and the summary statistics mask much of the data’s details, we need a more comprehensive way of looking at the set. This is when we turn to visualization.
  + Here are the data sets represented visually on a scatterplot:
  + We can immediately see how the data are arranged and get a sense of their distribution, scale, center, etc.
  + Data visualization is a quick, easy way to convey concepts in a universal manner — and you can experiment with different scenarios by making slight adjustments.
  + A side note: Statistician Francis Anscombe constructed this data set in 1973 to demonstrate the failures of summary statistics, highlight the effect of outliers on statistical properties, and attack the impression among statisticians that "numerical calculations are exact, but graphs are rough." You can read more about it here.
* Key Attributes of Good Visualization
  + Becoming familiar with some key design principles will help us create more effective visualizations.
  + To begin, look at this example of three different ways to visualize the same data. Which one gives you the most immediate information? Why?
    - 
    - 
    - 
  + The last visualization on the previous slide was probably your pick for the most informative. To help us understand why, let's look at this set of visualizations from Jeffrey Shaffer, which illustrates several different ways of organizing, emphasizing, and presenting data:
  + 
  + In these examples, we can immediately see that our brains process some attributes more readily than others. The ones we tend to focus on most are (in order): position, color, and size.
* A Note on Color
  + While position and size are somewhat intuitive (and often an inherent part of the visualization itself), color should be used a bit more deliberately. Generally, there are three ways to use color:
  + 1) Sequential: Best for values ordered from low to high.
  + 
  + 2) Divergent: Best for values that have a critical midpoint, such as an average or zero.
  + 
  + 3) Categorical: Best for values that fall into distinct groups (often for qualitative data).
  + 
* Choosing Your Visualization
  + Now that you have an idea of some guiding principles, let's look at the most important part of creating visualizations: choosing the right one.
  + Some of the most common types of data visualizations are:
    - Bar charts.
    - Pie charts.
    - Scatterplots.
    - Line graphs.
    - Histograms.
  + As you work with data, you’ll come across an endless variety of visualizations, including heat maps and box-and-whisker plots. Check out this extensive gallery to see more.
  + The next slides will explore each of the visualizations mentioned above.
  + Many of the examples to follow came from this website, which offers a great overview of charts and includes some examples of code.
* Bar Charts
  + Bar charts are one of the most common ways of visualizing data. They illustrate a summary statistic (e.g., count, sum, etc.) for each data point in a set of categories.
  + It's good practice to leave a small gap between the bars in your bar chart in order to show that the categories are distinct.
  + Use bar charts when:
    - Your data split neatly into categories.
    - Your data are numerical (especially counts).
    - You want to quickly compare categories to find highs and lows.
  + 
* Pie Charts
  + Pie charts are the most commonly misused chart type, so be careful with them. They show the proportion of the whole that's represented by each category.
  + Use pie charts when:
    - You are showing relative proportions or percentages.
    - Your data exist in distinct categories.
  + Pie charts can be hard to fully comprehend. Try to estimate the percentage that apples represent in the chart below. It's difficult to gauge an exact proportion for any slice using only the naked eye. If you want to compare data, use bar or stacked bar charts. If your viewer has to work to translate pie wedges into relevant data or compare pie charts to one another, the key points you're trying to convey might go unnoticed.
* 
* Scatterplots
  + Scatterplots are helpful for providing a sense of trends, concentrations, and outliers. This knowledge will provide you gain a clear idea of what you should investigate further.
  + Use scatterplots when:
    - You have two-dimensional data and want to spot trends or relationships.
    - You want to compare two specific attributes of each data point.
    - Your data lie along a continuous, numerical scale.
  + 
  + This example is actually using three specific attributes, but "Life Expectancy" and "GDP per Capita" are the main features being compared. Having the year represented as a color (optional) adds another dimension to the visualization; in this case, an additional layer of information.
* Line Graphs
  + Line graphs are used to indicate trends — most often ones that occur over time. Before using a line graph, consider why you are connecting your data points. Does the relationship between two neighboring data points matter? Does one influence the other? Do your data points have a meaningful order?
  + Use line graphs when:
    - You are showing a progression.
    - You are representing change over time.
    - Your data represent a continuous trend, such as a steady increase in the independent variable.
    - The relationship between each data point matters and you want to see increases and decreases between them.
* 
* Histograms
  + Histograms are useful for exploring the distribution of your data, especially when checking for normal distribution (like a bell curve).
  + They are similar to bar charts, but a key difference is that the x axis of a histogram represents a continuous variable, rather than distinct categories. Unlike bar charts, histograms shouldn't have a gap between bars. Why? Because you're showing a continuous and connected data set.
  + Use histograms when:
    - You want to compare a summary statistic within "buckets" along a continuous scale.
    - Your data can easily be collected into groups (but aren't categorical).
    - You only want to examine a single, specific attribute of the data.
* 
* Matplotlib
* 
  + Matplotlib is the core visualization library in Python. Many other libraries are built on top of it, so understanding how to manipulate its key features, such as plot size and axis, is useful no matter which tool you choose. It is extremely powerful, but this complexity creates a steep learning curve.
  + There's a lot you can accomplish with this library and, in future lessons, we'll get to know some of the main methods involved. Matplotlib graphs start out simply, and — while they are highly customizable — they aren't always the best looking out of the box. Still, Matplotlib is an invaluable library that's crucial to creating data visualizations in Python.
  + Matplotlib Histogram
  + 
  + Matplotlib Line Graph
  + 
  + The general sequence of using Matplotlib is:
    - Collecting and cleaning your data.
    - Creating your plot and its axes.
    - Customizing these to your heart's content.
    - Plotting your data on the graph.
    - Continuing to add plots to the graph as desired (e.g., a line chart and bar chart on the same graph, or layered transparent graphs).
    - Displaying the graph.
* Seaborn
  + Seaborn was created to make Matplotlib a little more user friendly. While still based on Matplotlib's foundation, it tends to make more visually appealing graphs right away and tries to simplify the visualization process. It comes with established styles that are easy to apply and includes some plot types — including violin plots, pair plots, and heat maps — that can be created in Matplotlib but are more complex to execute.
  + Seaborn Violin Plot 
  + Seaborn Heat Map
  + 
  + The sequence of creating these graphs in Seaborn is similar to Matplotlib, and we will explore that methodology and the differences between the two in future lessons.
* Pandas
* 
  + Pandas — the Python library for organizing, cleaning, and otherwise manipulating your data — takes a different approach to visualization. Instead of creating the plot and then inserting the data, it creates the graph directly from the data set itself.
  + The visualizations created by Pandas can all be customized in the same way as Matplotlib, using the same exact process and commands. And, like many of these libraries, Pandas' visualization capabilities are built off of Matplotlib. However, Pandas' DataFrame and Series objects also have built-in plotting methods, allowing for quick and easy visualizations without too much extra work.
  + This is especially useful in exploratory data analysis, when you only need a cursory glance at the data to find patterns.
  + Again, we'll go into the methodology and code for this process in future lessons.
  + Other Visualization Libraries
    - Plotly
      * A popular library for dynamic and interactive data visualization. It's primarily used through an online interface, so you need to sign up for an account. It does have an interface that can be used in Jupyter Notebooks, however — you just have to connect to your online account. In addition to creating interactive charts, Plotly also includes chart types that are sometimes omitted or difficult to create in other libraries. An example is a 3-D chart that a user can turn and navigate to see different angles.
    - Bokeh
      * This library is useful for creating visualizations for the web, as its outputs can take the form of JSON, HTML, or other web applications. Additionally, visualizations created in Bokeh, like Plotly, are interactive and dynamic. Bokeh has different levels of complexity in its interfaces; the middle one having the same level of customizable complexity as Matplotlib.
    - Pygal
      * Another library for interactive visualizations, the main benefit of Pygal is the ability to output its plots as SVG files, which is important for designers. Pygal may have trouble with large data sets, but it's useful for creating visually appealing, dynamic charts with fairly simple code.
    - ggplot
      * Another library built on Matplotlib, ggplot is less customizable than others, but its main value is its ability to recreate a similar library in R (ggplot2) for use within Python. Because it was inspired by R, ggplot can be difficult for Python coders to get used to, but it allows users to create complicated graphs that are familiar to (and required for) many statistics applications.