Plotting Simple Graphs

Using the *matplotlib* Module

Plotting Simple Graphs

- The **matplotlib** is a comprehensive library tools for creating static, animated, and interactive visualizations. The **pyplot** submodule provides simple functions for creating and displaying graphs:
  - `from matplotlib import pyplot`
  - ```
    pyplot.bar('Q1', 99.1)
    pyplot.bar('Q2', 10.0)
    pyplot.bar('Q3', 25.4)
    ```
  - `pyplot.xlabel('Quarter')`
  - `pyplot.ylabel('Revenue')`
  - `pyplot.show()`
    ```

Google's Colaboratory

- https://colab.research.google.com
- Allows you to write and execute Python in your browser, combining **executable code** and **rich text** in a single document, along with **images**, **HTML**, etc.
  - Zero configuration required
  - Free access to GPUs
  - Easy sharing
  - Colab is used extensively in the machine learning community

Plotting Functions

Operation	Description
pyplot.bar(*x_value*, *y_value*) pyplot.bar([*x_values*], [*y_values*])	Plots a single bar on the graph or multiple bars when the x_values and y_values are provided as lists
pyplot.plot([*x_coords*], [*y_coords*]) pyplot.plot([*x_coords*], [*y_coords*], *format*)	Plots a line graph. The color and style of the line can be specified with a format string
pyplot.grid('on')	Adds a grid to the graph
pyplot.xlim (*min*, *max*) pyplot.ylim (*min*, *max*)	Sets the range of x_values or y_values shown on the graph
pyplot.title (*text*)	Adds a title to the graph

Plotting Functions, continued

Operation	Description
pyplot.xlabel (*text*) pyplot.ylabel (*text*)	Adds a label below the x-axis or to the left of the y-axis
pyplot.legend ([$label_1$, $label_2$, …])	Adds a legend for multiple lines
pyplot.xticks([x_coord_1, x_coord_2, …], [$label_1$, $label_2$, …])	Adds labels below the tick marks along the x axis
pyplot.yticks([x_coord_1, x_coord_2, …], [$label_1$, $label_2$, …])	Adds labels to the left of the tick marks along the y axis
pyplot.show()	Displays the plot

Plotting Format Options

Character	Color
b	Blue
g	Green
r	Red
c	Cyan
m	Magenta
y	Yellow
k	Black
w	White

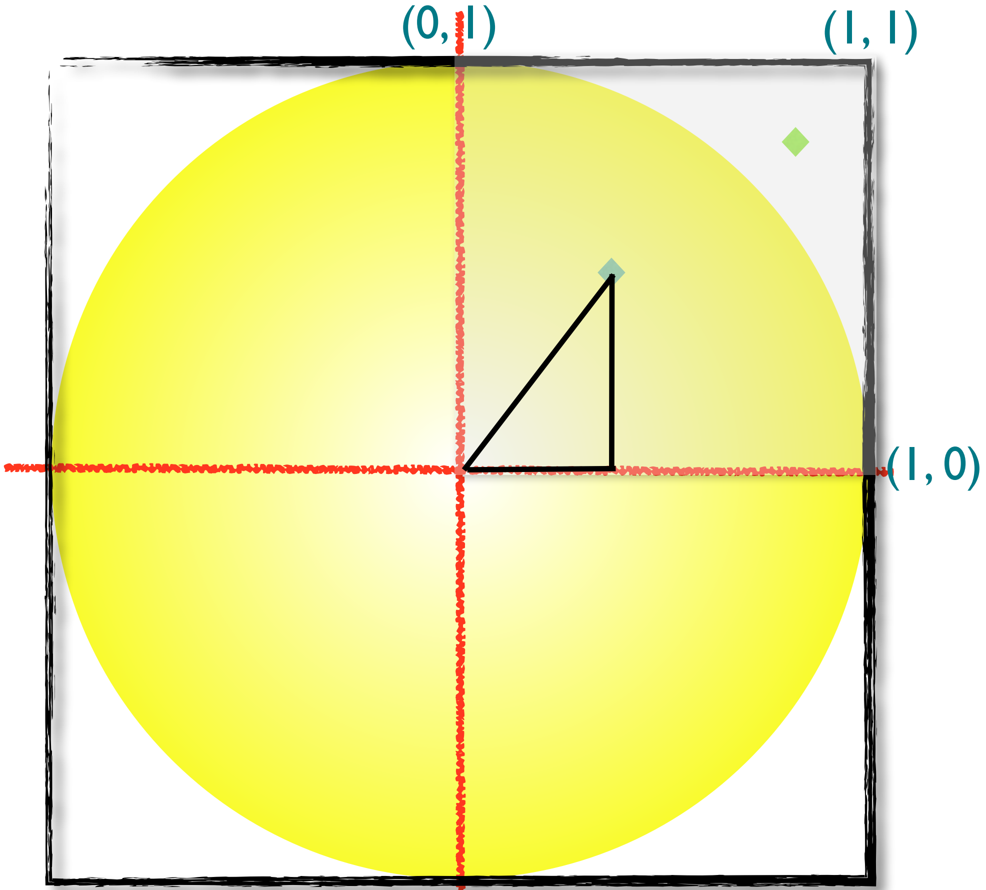
Character	Line Style
-	Solid
--	Dashed
:	Dotted
-.	Alternating dashes and dots

Character	Marker Style
.	Point
o	Circle
v	Triangle down
^	Triangle up
s	Square
*	Star
D	Diamond

Monte Carlo Methods

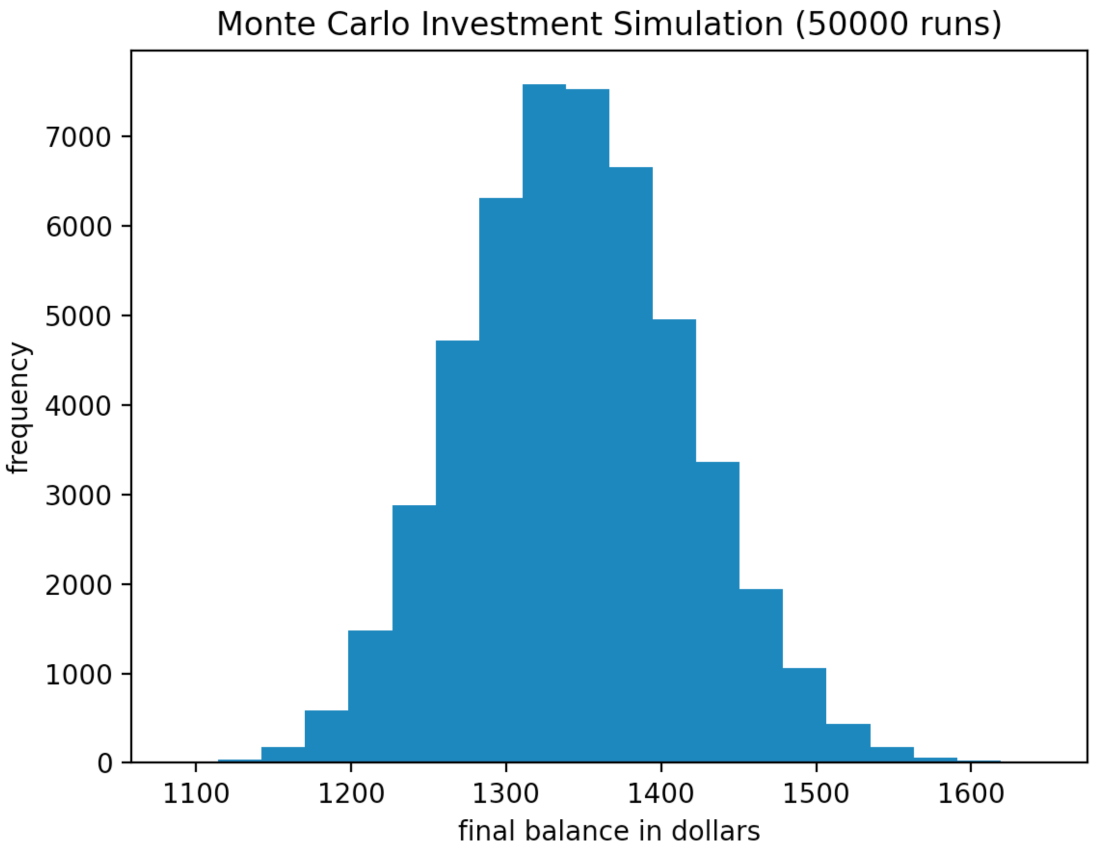
- Estimating the value of an unknown quantity using principles of inferential statistics. It can involve simulation experiments using random numbers.

Experiment to Estimate π



Investment Simulation

- Say we invest $1000 over 10 years at a rate of interest chosen randomly between 0% and 6%. We can perform this experiment 50,000 times and plot the result as a histogram.



Space / Time Tradeoffs

Indexing

A Space-Time Tradeoff: *Indexing*

RowIDs

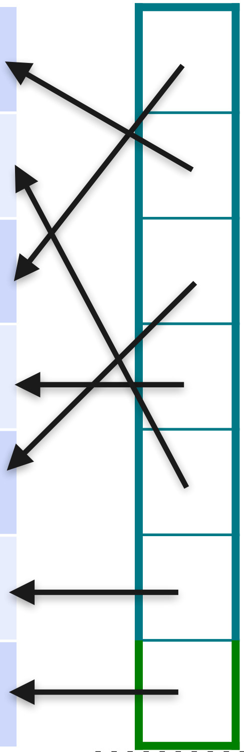
1	Edwards	Nancy	Sales Manager	8-Dec-58
2	Mitchell	Michael	IT Manager	1-Jul-73
3	Callahan	Laura	IT Staff	9-Jan-68
4	King	Robert	IT Staff	29-May-70
5	Johnson	Steve	Sales Support Agent	3-Mar-65
6	Park	Margaret	Sales Support Agent	19-Sep-47
7	Peacock	Jane	Sales Support Agent	29-Aug-73

A Space-Time Tradeoff: *Indexing*

RowIDs

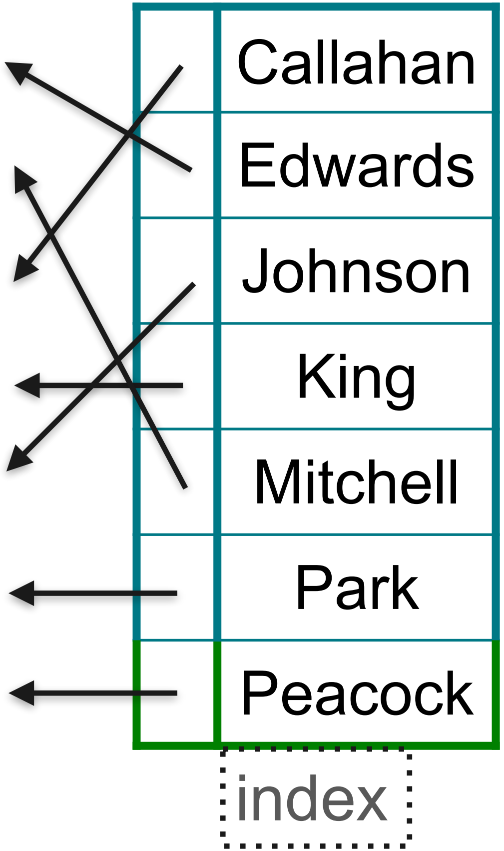
1	Edwards	Nancy	Sales Manager	8-Dec-58
2	Mitchell	Michael	IT Manager	1-Jul-73
3	Callahan	Laura	IT Staff	9-Jan-68
4	King	Robert	IT Staff	29-May-70
5	Johnson	Steve	Sales Support Agent	3-Mar-65
6	Park	Margaret	Sales Support Agent	19-Sep-47
7	Peacock	Jane	Sales Support Agent	29-Aug-73

employees.csv file

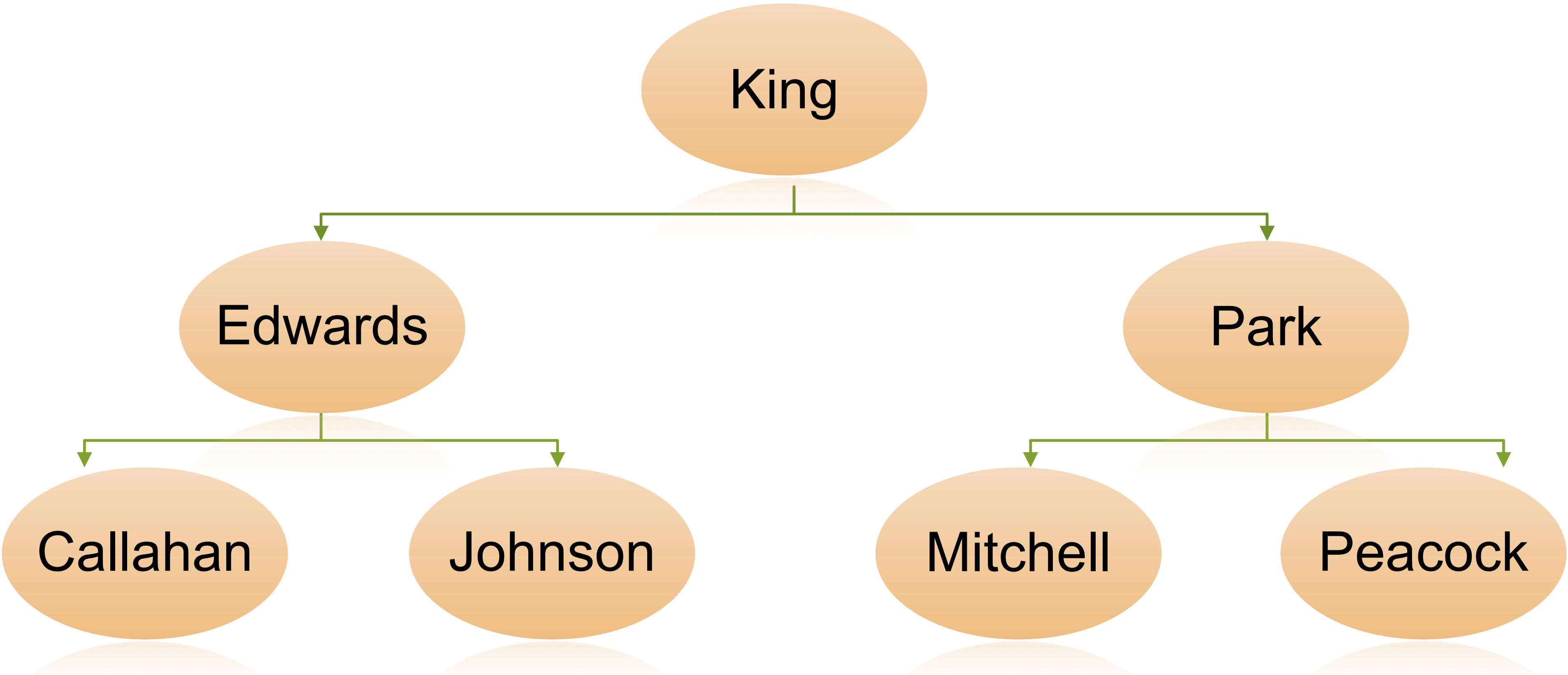


index

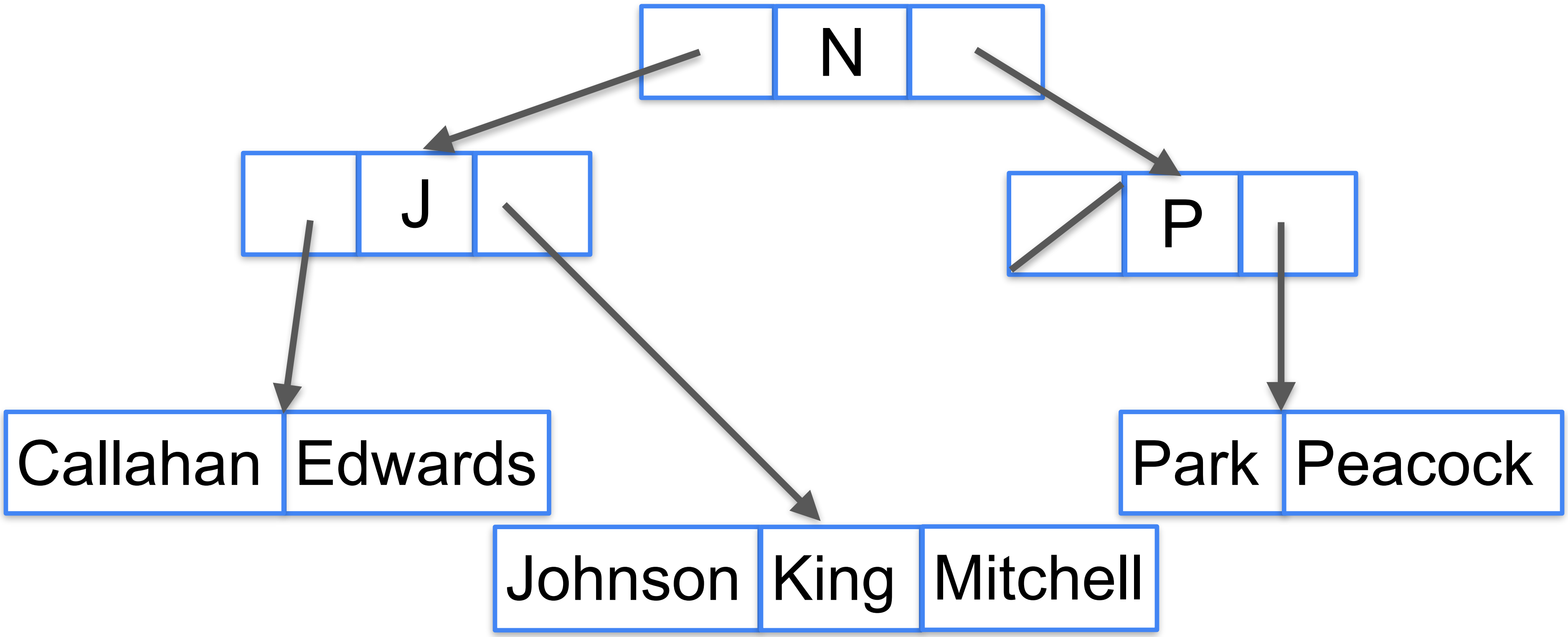
Index Structures: Binary Search



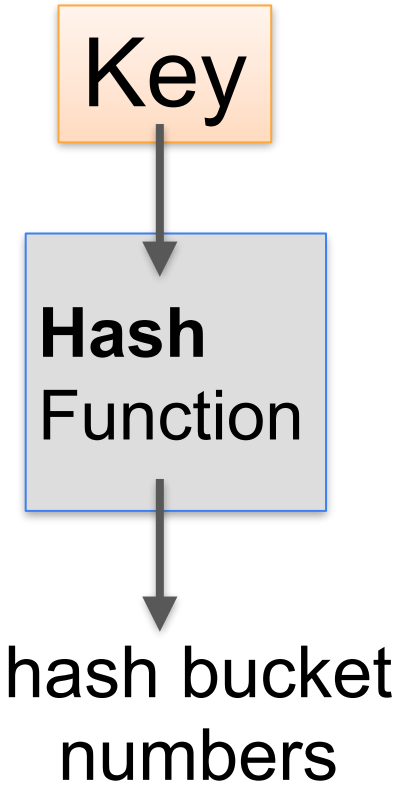
Index Structures: Binary Tree Example



Index Structures: B-Tree Example



Index Structures: Hash Tables



1	Edwards	Nancy	Sales Manager	8-Dec-58
2	Mitchell	Michael	IT Manager	1-Jul-73
3	Callahan	Laura	IT Staff	9-Jan-68
4	King	Robert	IT Staff	29-May-70
5	Johnson	Steve	Sales Support Agent	3-Mar-65
6	Park	Margaret	Sales Support Agent	19-Sep-47
7	Peacock	Jane	Sales Support Agent	29-Aug-73

Other Time/Space Trade-offs

- caching
- compression
- multi-resolution images
- storing derived data vs. recomputing it

Caching

- A *cache* is a place to store data so that it be accessed more easily/ rapidly
    - Hardware caches: L1, L2, L3 (registers are a form of cache)
    - Main memory is frequently a cache for persistent data
    - Database: persistent data may reside on SSD or spinning disk, but subsets are held in memory while they are being accessed
- The trade-off:
    - The cache consumes space ...
    - ... but provides quicker access to data

Compression

- Large data is frequently compressed to consume less space
    - Once compressed, computing on it and/or visualizing it typically requires computation
- The trade-off:
    - Keep data in a compressed form, saving on storage …
    - ... but uncompressing it requires computation (i.e., slower)

Compression Example: Image Storage

- High resolution images are beautiful
  - But they are large, and consume bandwidth to transmit
  - Devices with small displays may not be able to make use of the high resolution
- The trade-off:
  - Store many different resolutions for different scenarios (saving space)
  - Create reduced resolution copies as needed (consuming time)

Storage versus Recomputation

- Any time you create a derived data product, you face a space/time trade-off
  - Compute monthly sales numbers
  - Generate reports

- Do you keep the derived data or recompute it the next time you need it?

Machine Learning

Defining Machine Learning (ML)

- Older, informal definition from Arthur Samuel:
  - "a field of study that gives computers the ability to learn without being explicitly programmed."
- From IBM:
  - "a branch of artificial intelligence (AI) that focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy."
- Formal definition from Tom Mitchell at CMU:
  - "a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Applications of Machine Learning

- Examples we often experience:
  - Google or Bing search (results are ranked)
  - Facial recognition in Facebook or Apple Photos
  - Netflix/Amazon/iTunes recommendations
  - Email spam filter
  - Handwriting recognition (check deposits/mail routing)

- Research examples:
  - Predict whether a cancer cell is malignant or benign
  - Estimate the CO_2 emissions of a hypothetical auto

Machine Learning

Useful Python Libraries

Numpy

- Used to create multi-dimensional arrays
  - import numpy as np
  - arr = np.array([1, 2, 3, 4, 5])
  - arr2 = np.array([[1, 2, 3], [4, 5, 6]])
  - arr3 = np.array([[[1, 2, 3], [4, 5, 6]],
    [[1, 2, 3], [4, 5, 6]]])

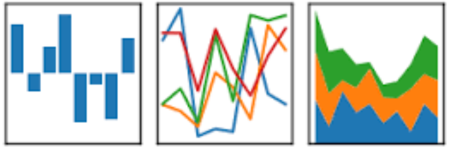
- the **ndim** attribute returns an integer with the # of dimensions the array has.

SciPy

- *SciPy* is a scientific computation library
  - **from scipy import stats**
  - **data = [99, 86, 87, 88,103, 87, 94, 86, 78, 77, 85, 86]**
  - **x = stats.mode(data)**

- *Optimizers* are a set of procedures defined in *SciPy* that either find the minimum value of a function, or the root of an equation.
  - While *NumPy* is capable of finding roots for polynomials and linear equations, it can not find roots for *non* linear equations

Pandas



- A *DataFrame* is a 2 dimensional data structure, like a 2D array, or a table with rows and columns. For example,

```
import pandas as pd
data = {
          "radius": [17.99, 20.57, 19.69],
          "texture": [10.38, 17.77, 21.25]
       }
df = pd.DataFrame(data, index = ["p1", "p2", "p3"])
print(df)
```

- CSV files (as well as a JSON files) can be loaded into a *DataFrame*. For example,

```
import pandas as pd
df = pd.read_csv('cancer_data.csv')
```

Scikit-Learn

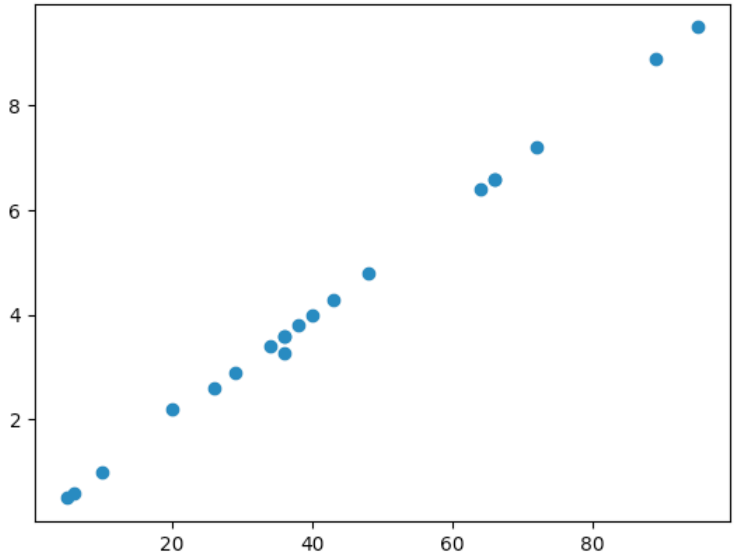
- SciKit-Learn is a collection of algorithms (clustering, regression, classification) and tools for machine learning
- Works well with numpy and scipy
- A machine learning task can be done simply in a few lines of code using SciKit-Learn

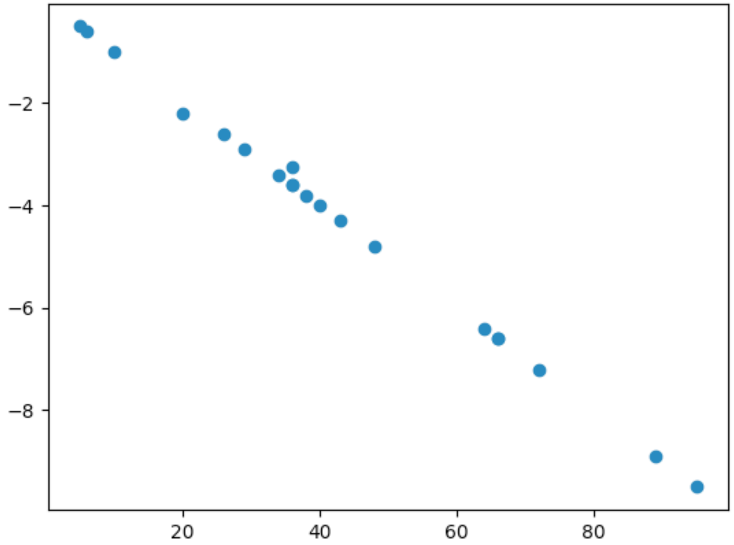
Machine Learning

Supervised Learning via Regression

Supervised Learning using Regression

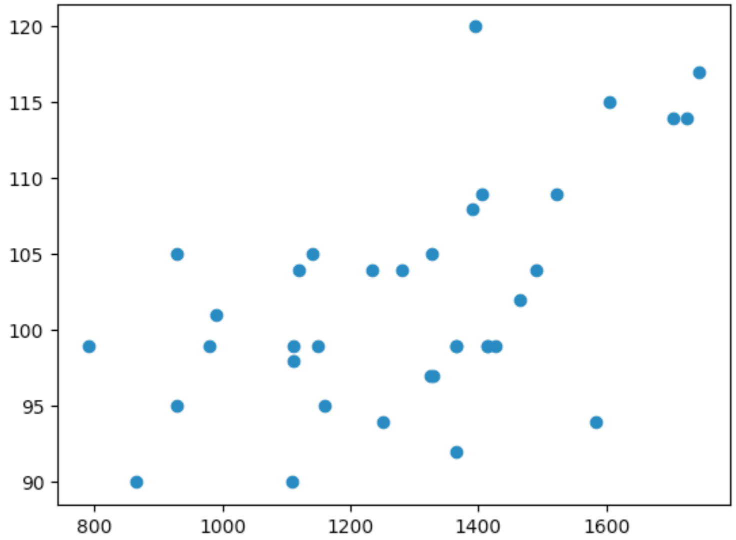
- *Linear Regression* uses the relationship between data points to draw a straight line through all them. This line can be used to predict future values.

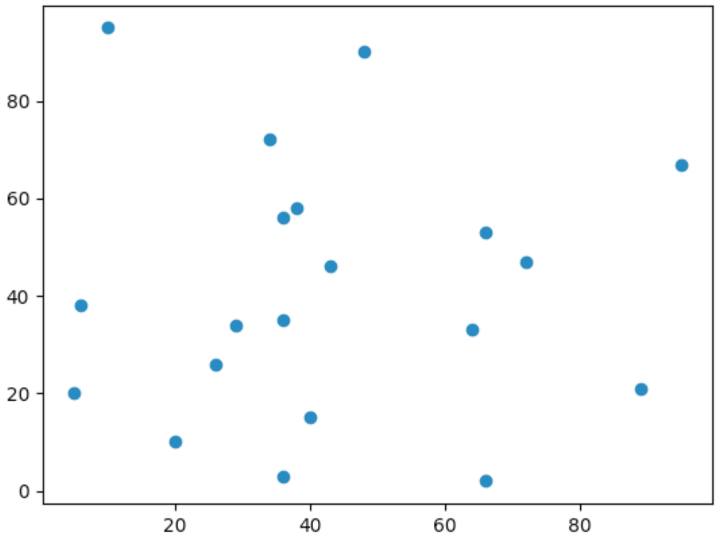




Coefficient of Correlation, r

- To determine how well you data fits a linear regression, compute **r**. The values range from -1 to 1, where 0 means no relationship.





Supervised Learning using Regression

- For example, can the size of a car's engine (the independent variable) predict the car's CO_2 emissions (the dependent variable)?
  - See: https://www.kaggle.com/debajyotipodder/co2-emission-by-vehicles
  - Program linearRegression.py creates a scatterplot of a subset of this vehicle data and then uses the linregress function in the *stats* module of the scipy library
- To improve accuracy, we can use more than one independent variable, such as engine size **and** the number of cylinders. See multipleRegression.py

Non-Linear Regression

- Sometimes linear regression is not be the best method to predict future values. For example,
  - x = [89, 43,36, 36, 95,10, 66, 34,38, 20, 26, 29, 48, 64, 6, 5,36, 66,72,40]
  - y = [21, 46, 3, 35, 67, 95, 53, 72, 58,10,26, 34, 90,33, 38, 20,5 6,2,47,15]
  - see programs badFit.py and goodFit.py

HARVARD BUSINESS ANALYTICS PROGRAM