Python IQ

- **What are the key features of Python?**
  - Python is an interpreted language. That means, unlike other languages like C, Python does not need to be compiled before it is run. Other examples of interpreted languages - PHP and Ruby.
  - Python is dynamically typed, meaning that we do not need to state the 'type' of variable while declaring. We can have a code with two consecutive statements like:

    ```
    X = 111
    X = "I'm a string."
    ```

    This will be correct in Python
  - Python is well suited to object oriented programming in that it allows the definition of classes along with composition and inheritance. Python does not have access specifiers (like 'public' and 'private' in C++).
  - In Python, functions are first-class objects. This means that they can be assigned to variables, returned from other functions and passed into functions. Classes are also first-class objects.
  - Writing Python code is quick but running the code is often slower as compared to compiled languages. Fortunately, Python allows the inclusion of C based extensions so that the bottlenecks can be optimized away. The 'numpy' package is a good example of this, it is really quite quick because a lot of the number crunching that it does, isn't actually done by Python.
  - Python finds usage in many spheres - web applications, automation, scientific modelling, big data applications and many more. It's also often used as a "glue" code to get other languages and components to play nice.

- **What is the difference between deep and shallow copy?**
  - Shallow copy is used when a new instance type gets created and it keeps the values that are copied in a new instance. Shallow copy is used to copy the reference pointers just like it copies the values. These references point to the original objects and the changes made in any member of the class will also affect the original copy of it. Shallow copy allows faster execution of the program and it depends on the size of the data that is used.
  - Deep copy is used to store the values that are already copied. Deep copy doesn't copy the reference pointers to the objects. It makes the reference to an object and the new object that is pointed by some other object gets stored. The changes made in the original copy won't affect any other copy that uses the object. Deep copy makes the execution of the Program slower due to making certain copies for each object that is been called.

- **What is the difference between list and tuples?**
  - Lists are mutable, i.e. they can be edited. Syntax: list_1 = [10, 'Chelsea', 20]
  - Tuples are immutable, i.e. they are lists which cannot be edited. Syntax: tup_1 = (10, 'Chelsea', 20)

- **How is Multithreading achieved in Python?**
  - Python has a multi-threading package but if you want to multithread to speed the code up, Python has a construct called Global Interpreter Lock (GIL).
  - The GIL makes sure that only one of your threads can execute at any one time. A thread acquires a GIL, does a little work, then passes the GIL to the next thread.
  - This happens very quickly and hence to the human eye, it may look like the threads are executing in parallel. But they are really just taking turns using the same CPU core.
  - All this GIL passing adds overhead to the execution. This means that if you want to make your code run faster, then using threading package often isn't a good idea.

- **How can the ternary operators be used in Python?**
  - The ternary operator is the operator that is used to show the conditional statements. This consists of true or false values with a statement that has to be evaluated for it.
  - Syntax: The Ternary operator will be given as:
  - [on_true] if [expression] else [on_false].
  - x, y = 25, 50; big = x if x > y else y

- **How is memory managed in Python?**

- In Python, the memory is managed by the Python private heap space.
- All the Python objects are data structures that are located in a private heap.
- The programmer does not have access to the private heap and the interpreter takes care of this heap.
- The allocation of Python heap space for Python objects is done by the Python memory manager.
- The core API gives access to some tools for the programmer to code.
- Python also has an inbuilt garbage collector, which recycles all the unused memory, frees the memory and makes it available to the heap space.

- **Explain Inheritance in Python with an example.**
  - Inheritance allows one class to gain access to all the members (say attributes and methods) of another class.
  - Inheritance provides code reusability, makes it easier to create and maintain an application.
  - The class from which we are inheriting is called super-class and the class that is inherited is called derived / child class.
  - There are different types of inheritance supported by Python:
    - Single Inheritance - where the derived class acquires the members of a single super class.
    - Multi-level inheritance - a derived class d1 is inherited from base class b1 and d2 is inherited from b2.
    - Hierarchical Inheritance - from one base class you can inherit any number of child classes.
    - Multiple Inheritance - a derived class is inherited from more than one base class.

- **Explain 'Flask' and its benefits.**
  - Flask is a web micro framework for Python based on "Werkzeug, Jinja2 and good intentions" BSD license.
  - Werkzeug and Jinja2 are two of its dependencies.
  - This means it will have little or no dependencies on external libraries.
  - This makes the framework light while there is little dependency to update and less security bugs.
  - A session basically allows you to remember information from one request to another.
  - In a flask, a session uses a signed cookie so the user can look at the session contents and modify.
  - The user can modify the session if only it has the secret key Flask.secret_key.

- **What is the usage of the help() and dir() function in Python?**
  - help() and dir() both the functions are accessible from the Python interpreter and used for viewing a consolidated dump of build-in functions.
  - help() - The help() function is used to display the documentation string and also facilitates you to see the help related to modules, keywords and attributes etc
  - dir() - The dir() function is used to display the defined symbols.

- **Whenever Python exits, why isn't all the memory de-allocated?**
  - Whenever Python exits, especially those Python modules that have circular references to other objects or the objects that are referenced from the global namespace are not always de-allocated or freed.
  - It is impossible to de-allocate those portions of the memory that are reserved by the C library.
  - On exit, because of having its own efficient clean up mechanism, Python would try to de-allocate/destroy every other object.

- **What is a 'dictionary' in Python?**
  - Dictionary is a built-in datatype in Python.
  - It defines one-to-one relationship between keys and values.
  - Dictionaries contain pair of keys and their corresponding values.
  - Dictionaries are indexed by keys.
  - Example:

    ```
    dict = {'Country': 'India', 'Capital': 'Delhi', 'PM': 'Modi'}
    print dict[Country] -> India
    print dict[Capital] -> Delhi
    print dict[PM] -> Modi
    ```

- **What is monkey patching in Python?**
  - In Python, the term monkey patch only refers to dynamic modifications of a class or module at runtime.

- Example:
  ```
  # m.py
  class MyClass:
          def f(self):
                  print "f()"
  ```
- We can run monkey-patch testing as follows:
  ```
  import m
  def moneky_f(self):
          print "monkey_f()"

  m.MyClass.f = monkey_f
  obj = m.MyClass()
  obj.f()
  ```
- The output will be:
  ```
  monkey_f()
  ```
- As we can see, we did make some changes in the behaviour of f() in MyClass using the function we defined, monkey_f(), outside of the module m.

- **What do we mean by: *args, **kwargs in Python? Why would we use it?**
  - We use *args when we are not sure of how many arguments are going to be passed to a function, or if we want to pass a stored list or tuple of arguments to a function.
  - **kwargs is used when we don't know how many keyword arguments will be passed to a function or it can be used to pass the values of a dictionary as keyword arguments.
  - The identifiers 'args' and 'kwargs' are a convention, we can also use *bob and **billy but that would be weird and off standard.

- **Write a one liner that will count the number of capital letters in a file. Your code should work even if the file is too big to fit in the memory.**
  - Starting with a multiple line code first (which will be reduced to line, hence):
    ```
    with open(SOME_LARGE_FILE) as fh:
    count = 0
    text = fh.read()
    for character in text:
            if character.isupper():
                    count+=1
    ```
  - Reducing this to a one-line code:
    ```
    count sum(1 for line in fh for character in line if character.isupper())
    ```

- **What are negative indexes and why are they used?**
  - The sequences in Python are indexed and it consists of the positive as well as negative numbers.
  - The numbers that are positive use '0' as first index and then '1' and so on.
  - The index for the negative numbers starts from '-1' which represents the last index in the sequence.
  - '-2' then represents the penultimate index and so on.

*** How can you randomize the items of a list in place in Python?**

- Example:

```
from random import shuffle
x = ['Keep', 'The', 'Blue', 'Flag', 'Flying', 'High.']
shuffle(x) # This is the function responsible for shuffling the string
print(x)
```

- The output of the code will be:

```
['Flying', 'Keep', 'Blue', 'High.', 'The',  'Flag']
```

*** What is the process of compilation and linking in Python?**

- Compiling allows the new extensions to be compiled properly without any error
- Linking can be done only when it passes the compiled procedure.

- If dynamic loading is used, then it depends on the style that is being provided with the system.
- The python interpreter can be used to provide the dynamic loading of the configuration setup files and will rebuild the interpreter.
- Steps involved:

# Create a file with any name and in any language that is supported by the compiler of your system. Example: .c or .cpp

# Place the file in the Modules/ directory of the distribution which is getting used.

# Add a line in the file Setup.local that is present in the Modules/ directory.

# Run the file using spam file.object.

# After the successful run of this, rebuild the interpreter by using the 'make' command on the top-level directory.

# If the file is changed then run rebuildMakefile by using the command as 'make Makefile'.

* Write a sorting algorithm for a numerical dataset in Python.

- The following code can be used to sort the list in Python:

```
list = ['1' ,'4' ,'0' ,'6' ,'9' ]
list = [int(i) for i in list]
list.sort()
print(list)
```

* Looking at the below code, write the final values of A0, A1, A2 ..... An.

Code:

```
A0 = dict(zip(('a','b','c','d','e'), (1,2,3,4,5)))
A1 = range(10)
A2 = sorted([i for i in A1 if i in A0])
A3 = sorted([A0[s] for s in A0])
A4 = [i for i in A1 if i in A3]
A5 = {i:i*i for i in A1}
A6 = [[i,i*i] for i in A1]
print(A0,A1,A2,A3,A4,A5,A6)
```

- A0 = {'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5}
- A1 = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
- A2 = []
- A3 = [1,2,3,4,5]
- A4 = [1,2,3,4,5]
- A5 = {0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}
- A6 = [[0: 0], [1: 1], [2: 4], [3: 9], [4: 16], [5: 25], [6: 36], [7: 49], [8: 64], [9: 81]]

* What are Python zip() and dict() functions?

- zip():

# Make an iterator that aggregates elements from each of the iterables.

# Returns an iterator of tuples, where the i-th tuple contains the i-th element from each of the argument sequences or iterables.

# The iterator stops when the shortest input iterable is exhausted.

# With a single iterable argument, it returns an iterator of 1-tuples.

# With no arguments, it returns an empty iterator.

# Example: zip('ABCD', 'xy') --> Ax By

# The left-to-right evaluation order of the iterables is guranteed.

# This makes possible an idiom for clustering a data series into n-length groups using zip(*[iter(s)]*n)

# zip() should only be used with unequal length inputs when we do not care about the trailing unmatched values from the longer iterables.

# If those values are important, use itertools.zip_longer() instead.

# zip() in conjunction with * operator can be used to unzip a list:

# Example:

```
                x = [1,2,3]
                y = [4,5,6]
                zipped = zip(x,y)
                list(zipped) # Prints-> [(1,4),(2,5),(3,6)]
                x2, y2 = zip(*zip(x, y))
                x == list(x2) and y == list(y2) # Prints-> True
```

- dict():

```
        # dict() is the constructor for the dictionary type in Python.
        # This can build dictionaries directly from a sequence of key-value pairs:
        # Example:
                dict([('a',1), ('b',2), ('c',3)])
                # Prints-> {'a': 1, 'b': 2, 'c': 3}
```

*** Explain the following methods from the ‘re’ module in Python: split(), sub(), subn()**

- The 're' module in Python provides three methods to utilize regular expression related operations/calculations:

```
        # split() - uses a regex pattern to 'split' a given string into a list.
        # sub() - finds all substrings where the regex pattern matches and then replace them with a different string
        # subn() - it is similar to sub() and also returns the new string along with the no. of replacements.
```

*** How can you generate random numbers in Python?**

- 'random' module is the standard module that is used to generate random numbers.

```
        # import random
        # random.random()
```

- The statement random.random() method returns floating point numbers in the range [0,1).

*** What is the difference between 'range' and 'xrange'?**

- Both xrange and range provide a way to generate a list of integers to use.
- The key difference being - range provides with a Python list object whereas xrange returns an xrange object.
- xrange does not actually generates a static list at runtime like range does.
- It creates the values as we need them via a special technique called yielding.
- This technique is used with a type of object known as generators.
- If we have a really gigantic range, we would like to generate a list for, say one billion, xrange is the function to use.
- This is especially true for memory sensitive systems like cell phones.
- ‘range’ will use much more memory in creating an array of integers / list. It’s a memory hungry beast.

*** What is pickling and unpickling?**

- Pickle module accepts any Python object and converts it into a string representation and dumps it into a file by using dump function.
- This process is called 'pickling'.
- While the process of retrieving original Python objects from the stored string representation is called 'unpickling'.

- **Which of the following statements would create a dictionary? (MCQ)**
    - ○ d = {}
    - ○ **d = {“john”: 40, “peter”: 45}**
    - ○ **d = {40: “john”, 45: “peter”}**
    - ○ **d = {40: “john”, 45: ”50”}**
        - Dictionaries are created by specifying keys and values.

- **Which one of these is floor division?**
    - ○ /
    - ○ **//**

  - %
  - None of the mentioned
    - The floor division operation (//) is useful when both the operands are integers and we are expecting a decimal output. Example: 5/2 = 2; 5//2 = 2.5

- **What is the maximum possible length of an identifier?**
  - 31 characters
  - 63 characters
  - 79 characters
  - **None of the above**
    - Identifiers can be of any length

- **Why are local variable names beginning with an underscore discouraged?**
  - **They are used to indicate a private variables of a class.**
  - They confuse the interpreter
  - They are used to indicate global variables
  - They slow down execution
    - As python has no concept of ‘access specifiers’ that is private or public variables, leading underscores are used to indicate variables that must not be accessed from outside the class.

- **Which of the following is an invalid statement?**
  - abc = 1,000,000
  - **a b c = 1000 2000 3000**
  - a,b,c = 1000, 2000, 3000
  - a_b_c = 1,000,000
    - Python does not allow spaces within the variable names

- **What will be the output of the following code:**

```
try:
    if '1' != 1:
        raise "someError"
    else:
        print("someError has not occured")
except "someError":
    print ("someError has occured")
```

  - someError has occurred
  - someError has not occurred
  - **invalid code**
  - none of the above
    - A new exception class must inherit from a BaseException. There is not such inheritance here

- **Suppose list1 is [2, 33, 222, 14, 25]. What is list1[-1]?**
  - Error
  - None
  - **25**
  - 2
    - 25. Index -1 corresponds to the last element of the list

- **What will be the output of the following code:**

```
f = None

for i in range (5):
    with open("data.txt", "w") as f:
        if i > 2:
            break

print f.closed
```

  - **True**
  - False
  - None
  - Error

        - The WITH statement that is used to ‘open’ the file ensures that the file object is closed when the block exits.

- **When does the ‘else’ part of the try-except-else be executed?**
    - Always
    - When an exception occurs
    - **When no exception occurs**
    - When an exception occurs into the except block
        - For a try-except-else block, only executes when no exception occurs.

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**
    - test

- **Explain how we can**

  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

- **Explain how we can**
  - test

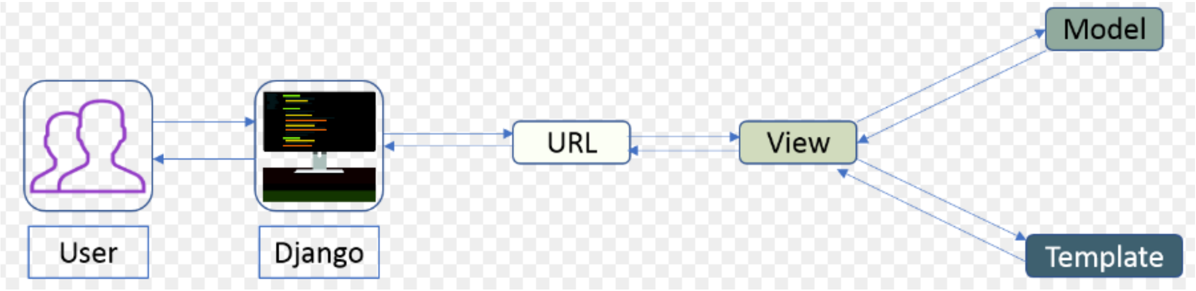
- **Explain how we can**
  - test

Django – Python Interview Questions

- **Mention the differences between Django, Pyramid and Flask.**
  - Flask – It is a ‘microframework’ primarily build for a small application with simpler requirements. In flask, one has to use external libraries. Flask is ready to use.

- Pyramid – It is built for larger applications. It provides flexibility and lets the developer use the right tools for their project. The developer can choose the database, URL structure, templating style and more. Pyramid is heavy configurable.
- Django – It can also be used for larger applications, just like Pyramid. It includes an ORM.

- **Discuss the Django architecture.**
  - Django MVT Pattern:



  - The developer provides with the Model, the view and the template then just maps it to the URL and Django does the magic to serve it to the user.

- **Whenever Python exits,**
  - Test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**

  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

- **Whenever Python exits,**
  - test

Machine Learning IQ

- **What is Machine Learning?**
  - Definition #1:
    - The field of study that gives computers the ability to learn without being explicitly programmed.
  - Definition #2:
    - A computer program is said to learn from an experience 'E' with respect to some task 'T' and some performance measure 'P' if its performance on 'T' as measured by 'P' improves with experience 'E'.

- **What are the different types of Machine Learning algorithms?**
  - The primary types of ML algorithms are:
    - Supervised
    - Unsupervised

        - Others - Reinforcement Learning, Recommender Systems etc.

- **What is Supervised Learning?**
    - Supervised learning is the scenario when there are two sets of data available
    - Set 1 is where we have the data features along with the 'right answers'
    - The algorithm analyses the Set 1 data and creates a prediction model.
    - Set 2 is then fed to the model and the answers are predicted

- **What are the primary sub-categories under Supervised Learning?**
    - REGRESSION
        - Here the 'right answers' need to be chosen from within a continuous set of permissible value range.
        - In regression, we try to predict the results within a continuous output, meaning that we are trying to map the input variables to some continuous function.
        - Example: Predicting the housing prices based on the Set1 provided.
        - Example: Given a picture of a person, we have to predict their age on the basis of the given picture.
    - CLASSIFICATION
        - Here the 'right answers' need to be chosen from a finite discrete set of permissible values (True/False; 0/1/2/3 etc)
        - Example: Predicting the tumour to be Malignant/Benign. Predicting the parts being ‘short’ or ‘not’.
        - Example: Given a patient with a tumour, we have to predict whether the tumour is malignant or benign.

- **What are Machine Learning Data Features?**
    - Features are the key business/source data attributes relevant to the candidate/prediction/output fields.
    - The features can range from as few as just one to as many as infinite.
    - Examples:
        - Housing Prices - Size, # of bedrooms, age, area etc
        - Tumour - size, patient's age, clump thickness, uniformity of cell size, uniformity of cell shapes etc

- **What is Unsupervised Learning?**
    - Unsupervised learning allows us to approach the problem with little or no idea as to what our results would look like.
    - There are no pre-existing structures or right answers to refer to.
    - There is also no feedback mechanism based on the prediction results.
    - Examples:
        - Google News: Group the new articles together based on the context and subject matter
        - Genes: Automatically group the genes into groups that are similar or related based on variables
        - Additional Examples: Organize computing clusters, Market segmentation, Social network analysis, Astronomical data analysis

- **What are the primary sub-categories under Unsupervised Learning?**
    - CLUSTERING
        - Data-set can be divided or categorized into clusters
        - Example: Genes
    - NON-CLUSTERING
        - Data-set cannot be clustered or categorized
        - Example: Distinguishing between overlapping voices recorded during parallel conversations at a party.

- **How do the Supervised Learning Algorithms work?**
    - Step 1: Identify the training dataset.

  - Step2: Identify the input dataset (x).
  - Spet3: Identify the 'Hypothesis' function (h(x))
  - Step4: Pass the input dataset values (x) to the 'Hypothesis' function, h(x) to get the predicted values.
  - Step5: the values generated by h(x) must be as close as possible to the real/actual values (y)
  - Equation:
    - y = h(x)

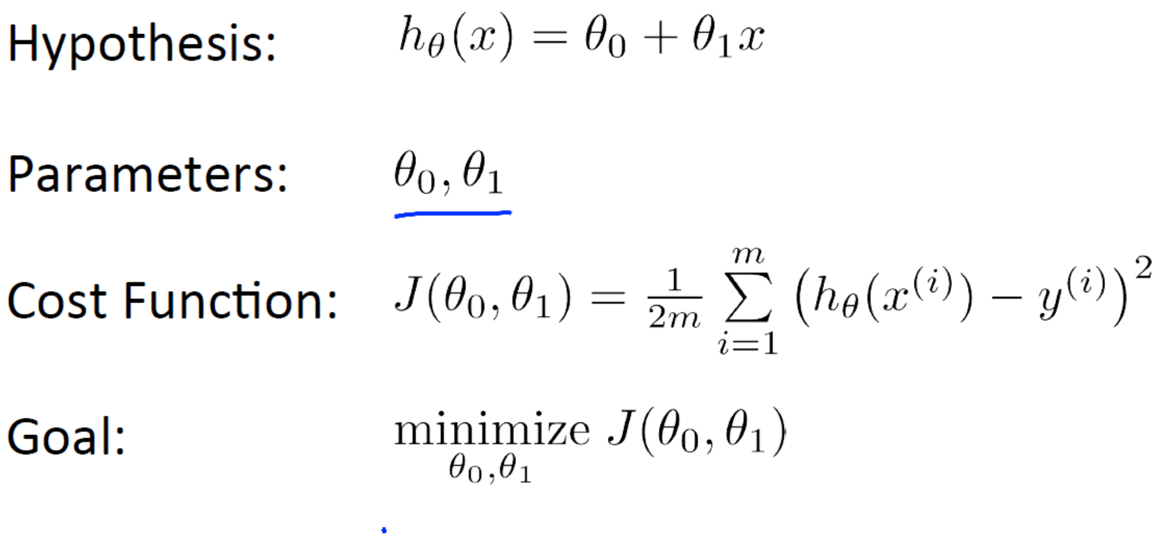
- **How is the hypothesis function represented for 'Linear Regression with One Variable'?**
    - h(x) = theta0 + theta1(x)
  - This is the simple equation for Linear Regression with One Variable or 'Univariate Linear Regression''

- **What is Linear Regression?**
  - A concept where a discrete set of input data is approximately represented via a linear (line equation based) hypothesis function h(x).
  - Example:
    - h(x) = theta0 + theta1(x)

- **What is the 'Cost Function'?**
  - The hypothesis function h(x) that is defined as above, needs to be chosen in such a way that it represents the data in the best possible way. That is, mathematically the data points (h(x)) are as close to the line (y) as possible.
  - In other words, the values of the parameters (theta1, theta2 …. etc) needs to be chosen such that the squared difference:
    - [h(x) - y]^2

    gets the minimum possible value.
  - The Cost Function J(theta) ensures this.
    - J(theta) = (1/2m)Sum[h(x) - y]^2
  - This is also known as the '**Squared Error Function**'

- **Why the 'Cost Function' gets calculated as a 'Squared Error' value?**
  - The Squared Error Function has mathematically proven to work well in regression scenarios
  - It also nullifies the potential difference between deviations of the values whether negative or positive.



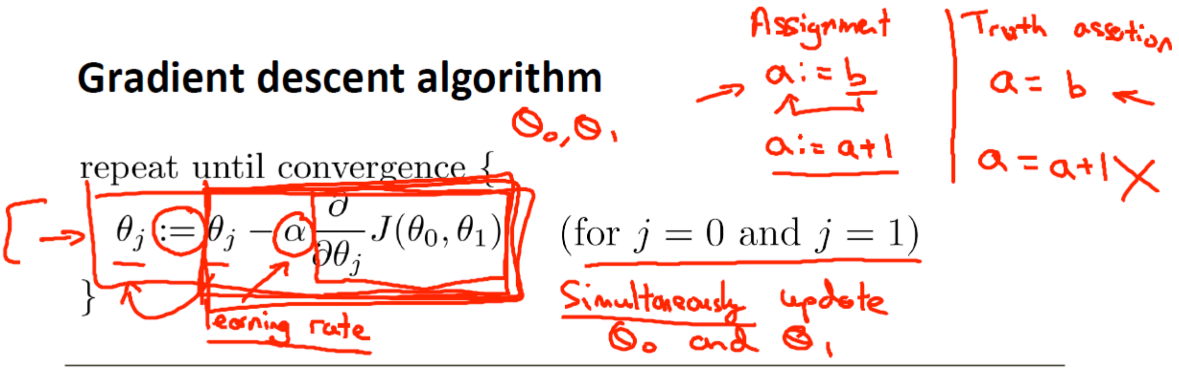
Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters: $\underline{\theta_0, \theta_1}$

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

- **What is Gradient Descend?**
  - Gradient Descend is an algorithm to mathematically calculate the specific values of theta(j)s for which the cost function gets the minimum possible values.



Gradient descent algorithm

repeat until convergence {

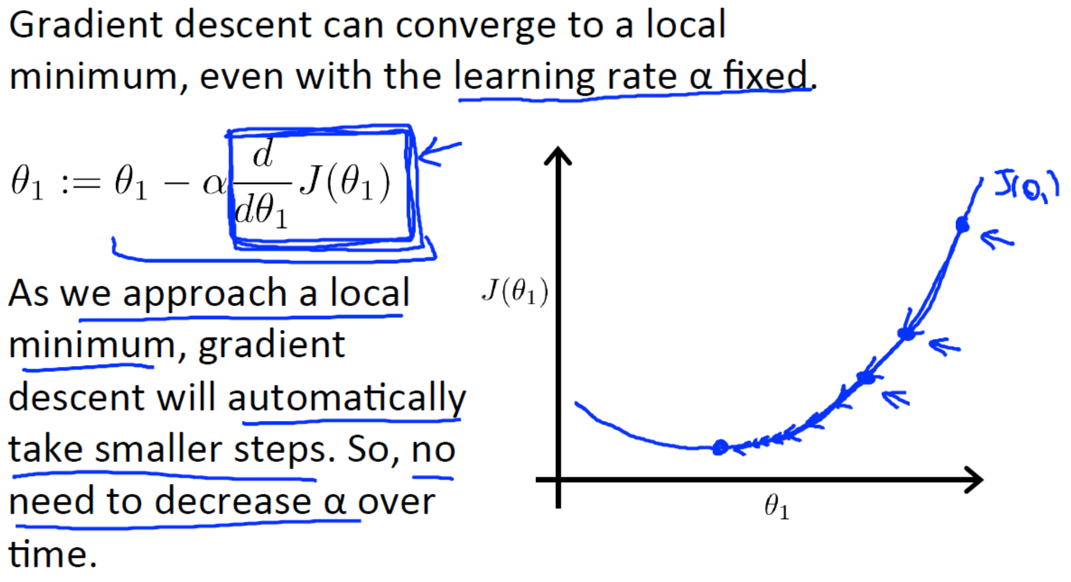
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad (\text{for } j = 0 \text{ and } j = 1)$$

}

- ○ The values for the parameters (theta(j)s) need to be updated simultaneously.

- ○ **What is Learning Rate (alpha)?**
  - ○ The learning rate (alpha) drives the precision – how steep or slow the change in the values for thetas should be.
  - ○ If alpha is too small, the gradient descend will be slow to converge
  - ○ If alpha is too large, the gradient descend may fail to converge and even start diverging

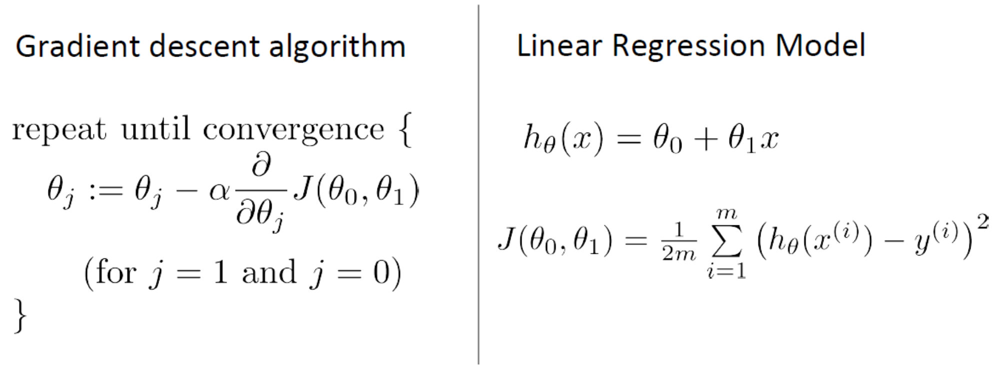
- ○ **Why is it not necessary to alter/decrease/increase the learning rate, alpha over time?**



Gradient descent can converge to a local minimum, even with the learning rate α fixed.

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease α over time.



Gradient descent algorithm

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

$$(\text{for } j = 1 \text{ and } j = 0)$$

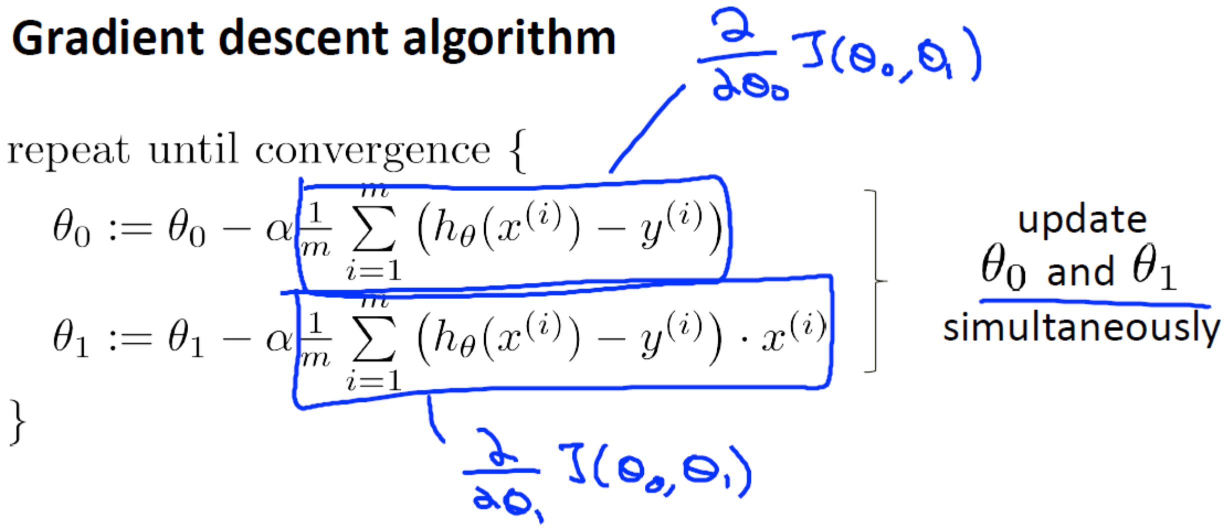
}

Linear Regression Model

$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Simplified Gradient Descend equations:



Gradient descent algorithm

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1)$$

repeat until convergence {

$$\theta_0 := \theta_0 - \alpha\boxed{\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)}$$

$$\theta_1 := \theta_1 - \alpha\boxed{\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)\cdot x^{(i)}}$$

}

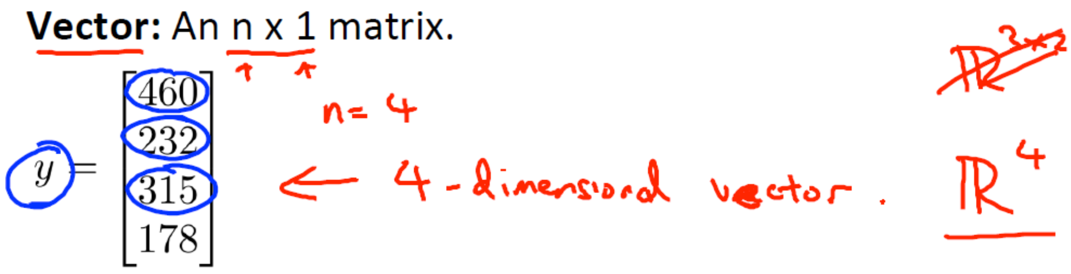
$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1)$$

update θ_0 and θ_1 simultaneously

- **What is Batch Gradient Descent?**
  - In batch Gradient Descent each step of the gradient descent uses all the training examples.

- **Mathematically what is a matrix?**
  - A Matrix is a rectangular array of numbers
  - Dimension of matrix: # of rows X # of columns

- **What is a vector?**
  - A vector is a nX1 matrix where n is the dimension of the vector.



Vector: An n x 1 matrix.

$$y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

n= 4

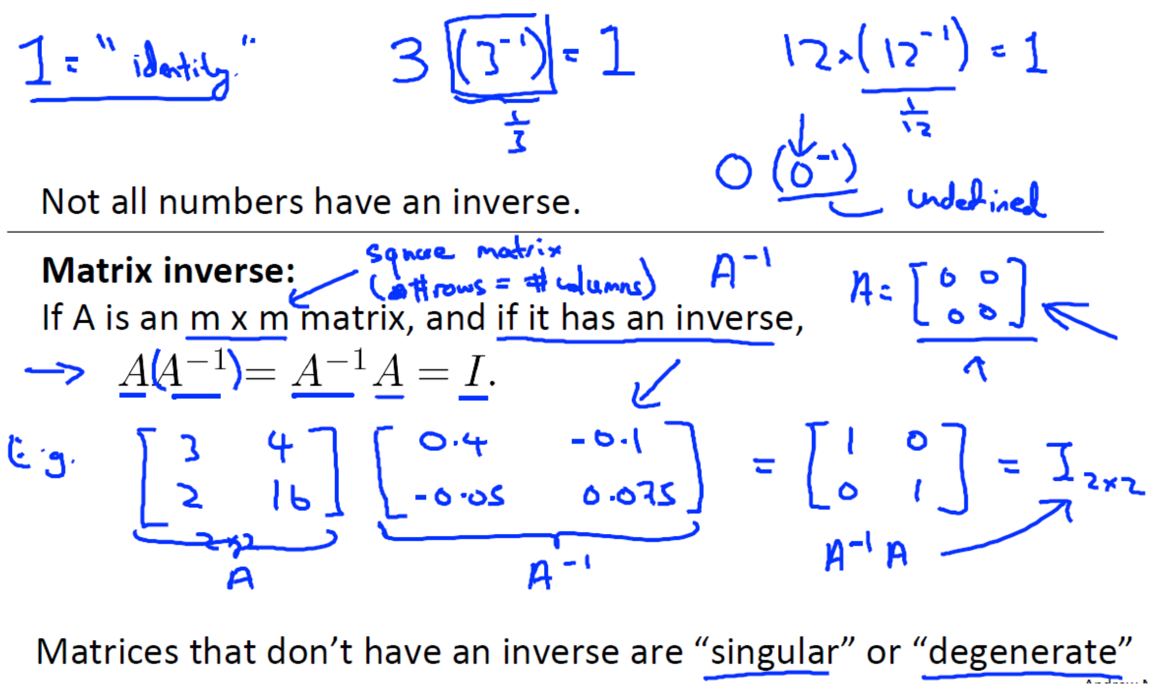
← 4-dimensional vector.

$\cancel{\mathbb{R}^{3\times2}}$

$\mathbb{R}^4$

- **Mathematical properties of Matrics?**
  - A + B = B + A
  - A * B <> B * A → Matrix multiplication is NOT commutative
  - A * (B * C) = (A * B) * C → Matrix multiplication is associative

- **What is an inverse and transpose of a Matrix?**



$1 =$ "identity" $3\,(3^{-1}) = 1$ $\frac{1}{3}$ $12 \times (12^{-1}) = 1$ $\frac{1}{12}$

$0\,(0^{-1})$ undefined

Not all numbers have an inverse.

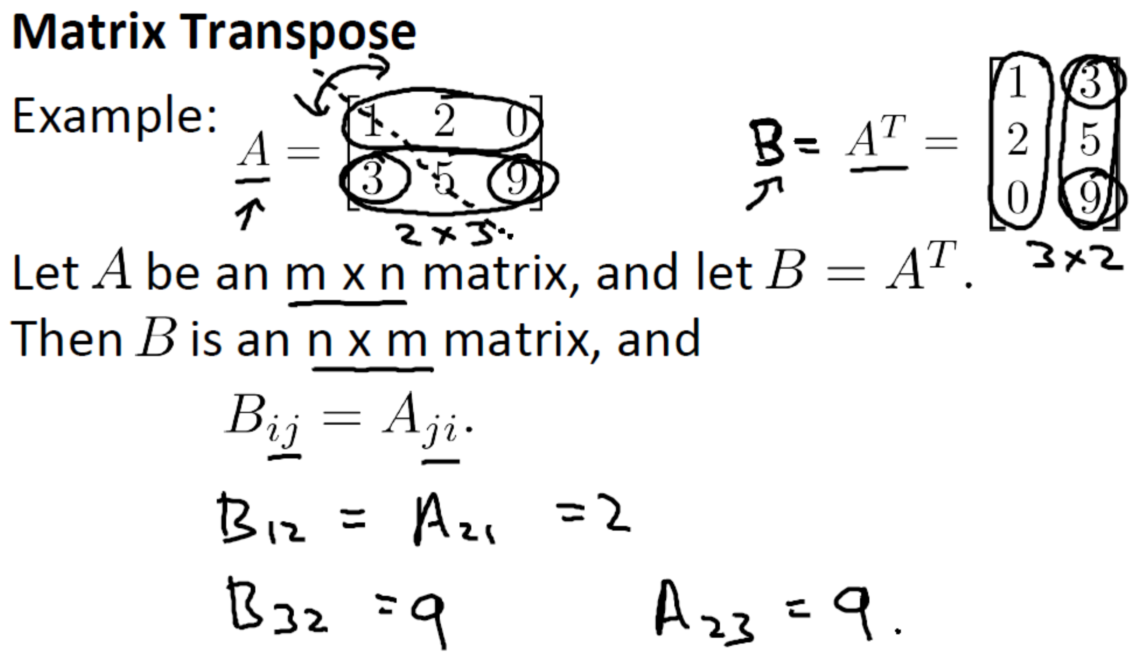
Matrix inverse: square matrix (#rows = #columns) A^{-1} $A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

If A is an m x m matrix, and if it has an inverse,

$$A(A^{-1}) = A^{-1}A = I.$$

E.g. $\underbrace{\begin{bmatrix} 3 & 4 \\ 2 & 16 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} 0.4 & -0.1 \\ -0.05 & 0.075 \end{bmatrix}}_{A^{-1}} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{A^{-1}A} = I_{2\times 2}$

Matrices that don't have an inverse are "singular" or "degenerate"



Matrix Transpose

Example: $A = \underbrace{\begin{bmatrix} 1 & 2 & 0 \\ 3 & 5 & 9 \end{bmatrix}}_{2\times 3}$ $B = A^T = \underbrace{\begin{bmatrix} 1 & 3 \\ 2 & 5 \\ 0 & 9 \end{bmatrix}}_{3\times 2}$

Let A be an m x n matrix, and let $B = A^T$.

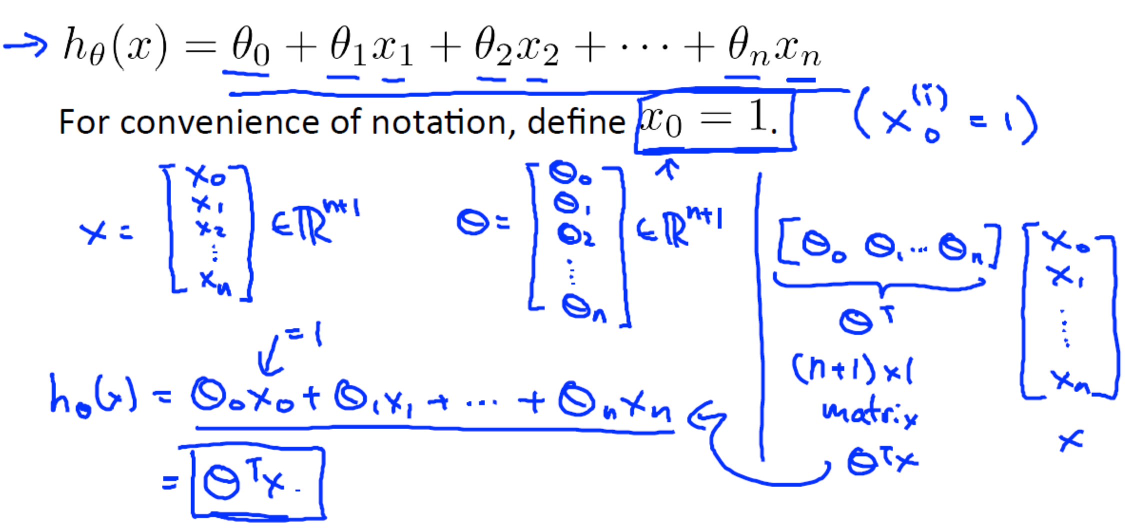
Then B is an n x m matrix, and

$$B_{ij} = A_{ji}.$$

$B_{12} = A_{21} = 2$

$B_{32} = 9$ $A_{23} = 9.$

- **Define Linear Regression with Multiple Variables?**
  - In case of LR with Multiple Variables, the hypothesis function depends on more factors (than just x).
  - It is also known as 'Multivariate Linear Regression'.



$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

For convenience of notation, define $\boxed{x_0 = 1.}$ $(x_0^{(i)} = 1)$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$h_\theta(x) = \theta_0 \overset{=1}{x_0} + \theta_1 x_1 + \cdots + \theta_n x_n$$

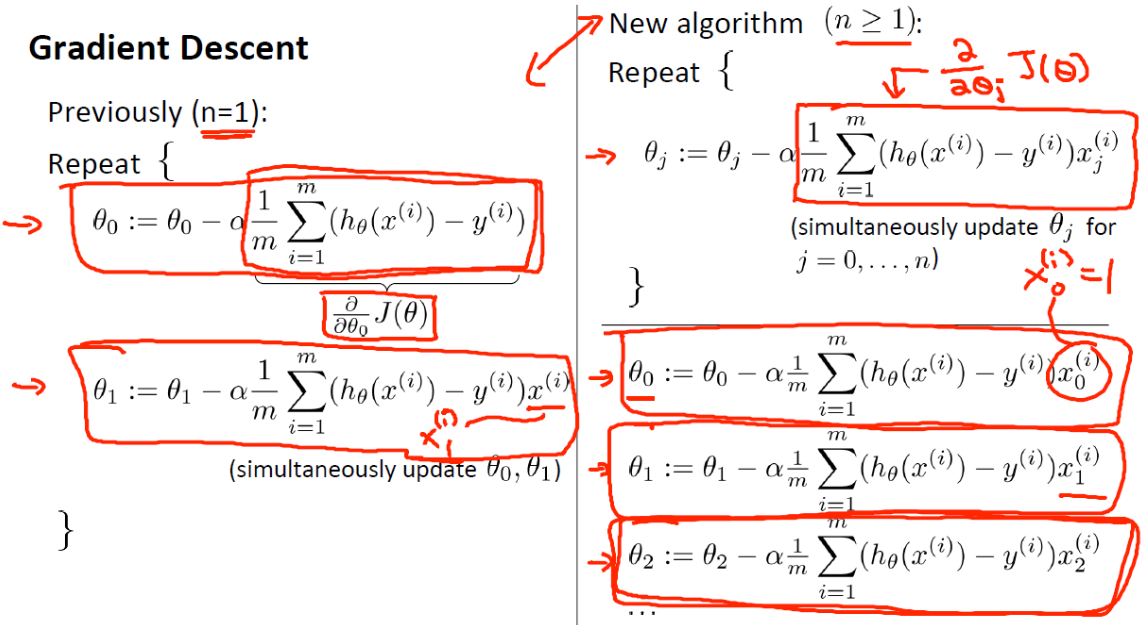
$$= \boxed{\theta^T x.}$$

$$\underbrace{[\theta_0\ \theta_1 \dots \theta_n]}_{\theta^T} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

$(n+1) \times 1$ matrix $\quad x$

$\theta^T x$

- **Define Gradient Descend for Multiple Variables**



Gradient Descent

Previously (n=1):

Repeat {

$$\theta_0 := \theta_0 - \alpha \underbrace{\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})}_{\frac{\partial}{\partial \theta_0} J(\theta)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

$x^{(i)}$

(simultaneously update θ_0, θ_1)

}

New algorithm $(n \geq 1)$:

Repeat {

$\frac{\partial}{\partial \theta_j} J(\theta)$

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(simultaneously update θ_j for $j = 0, \dots, n$)

}

$x_0^{(i)} = 1$

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

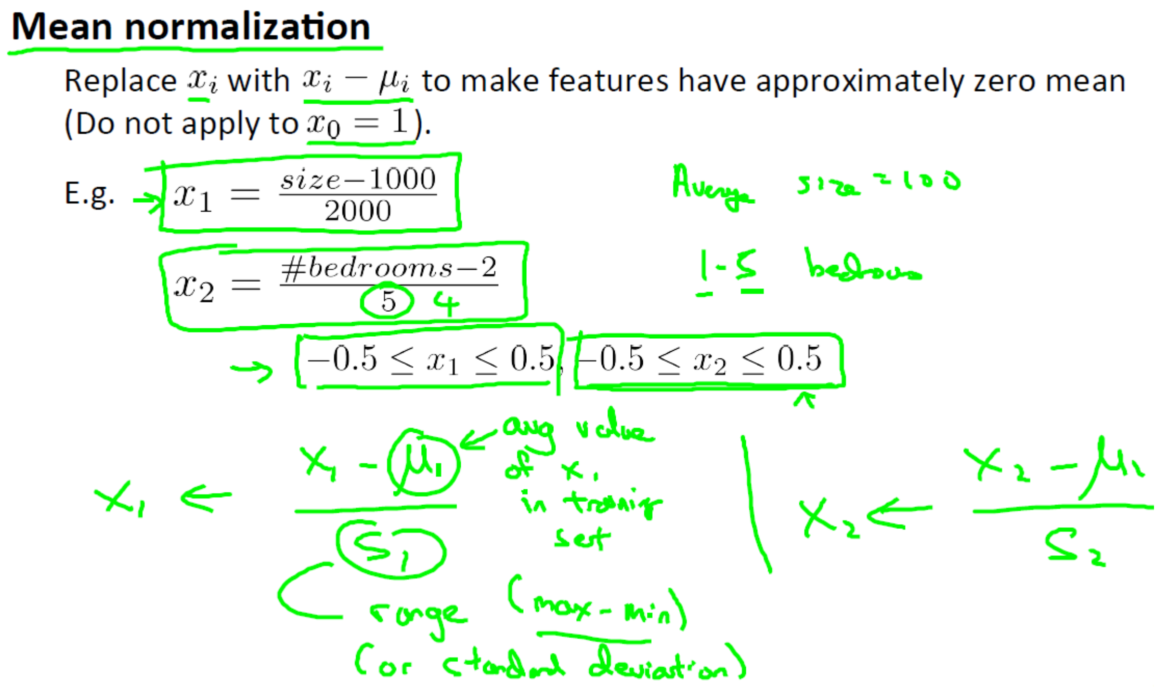
$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

$\cdots$

- **What is meant by Feature Scaling?**
  - Feature scaling is a concept where we need to compare the features and scale them in such a fashion that they are comparable.
  - Thumb rule is to get all the features into approximately the following range: -1 =< x(i) =< 1

- **What is Mean Normalization in context of Feature Scaling?**
  - With mean Normalization, we replace x(i) with x(i) – mu(i) where mu(i) is the mean.
  - This makes the features have approximately zero mean.



Mean normalization

Replace x_i with $x_i - \mu_i$ to make features have approximately zero mean (Do not apply to $x_0 = 1$).

E.g. $x_1 = \frac{size-1000}{2000}$

$x_2 = \frac{\#bedrooms-2}{5 \quad 4}$

$-0.5 \le x_1 \le 0.5$, $-0.5 \le x_2 \le 0.5$

Average size = 100

1-5 bedrooms

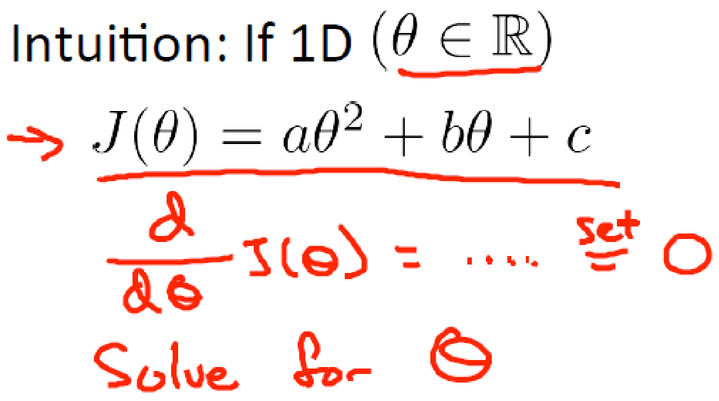
$x_1 \leftarrow \frac{x_1 - \mu_1}{s_1}$ avg value of x_1 in training set

range (max - min) (or standard deviation)

$x_2 \leftarrow \frac{x_2 - \mu_1}{s_2}$

- **What is ‘Polynomial’ regression under LR?**
  - Polynomial regression is fitting a polynomial curve instead of a linear equation.
  - Examples:
    - h(x) = theta(0) + theta(1)x + theta(2)x^2 + theta(3)x^3
    - h(x) = theta(0) + theta(1)x + theta(2)x^(1/2)

- **What is a ‘Normal Equation’?**
  - A normal equation method is an alternate method to solve for theta analytically.
  - Rather than going by the iterative approach (via Gradient Descend) and finding out the values of theta(j)s that generate the lowest values for the cost function J(theta), the Normal Equation can be solved directly to get the most suitable values of thetas directly.

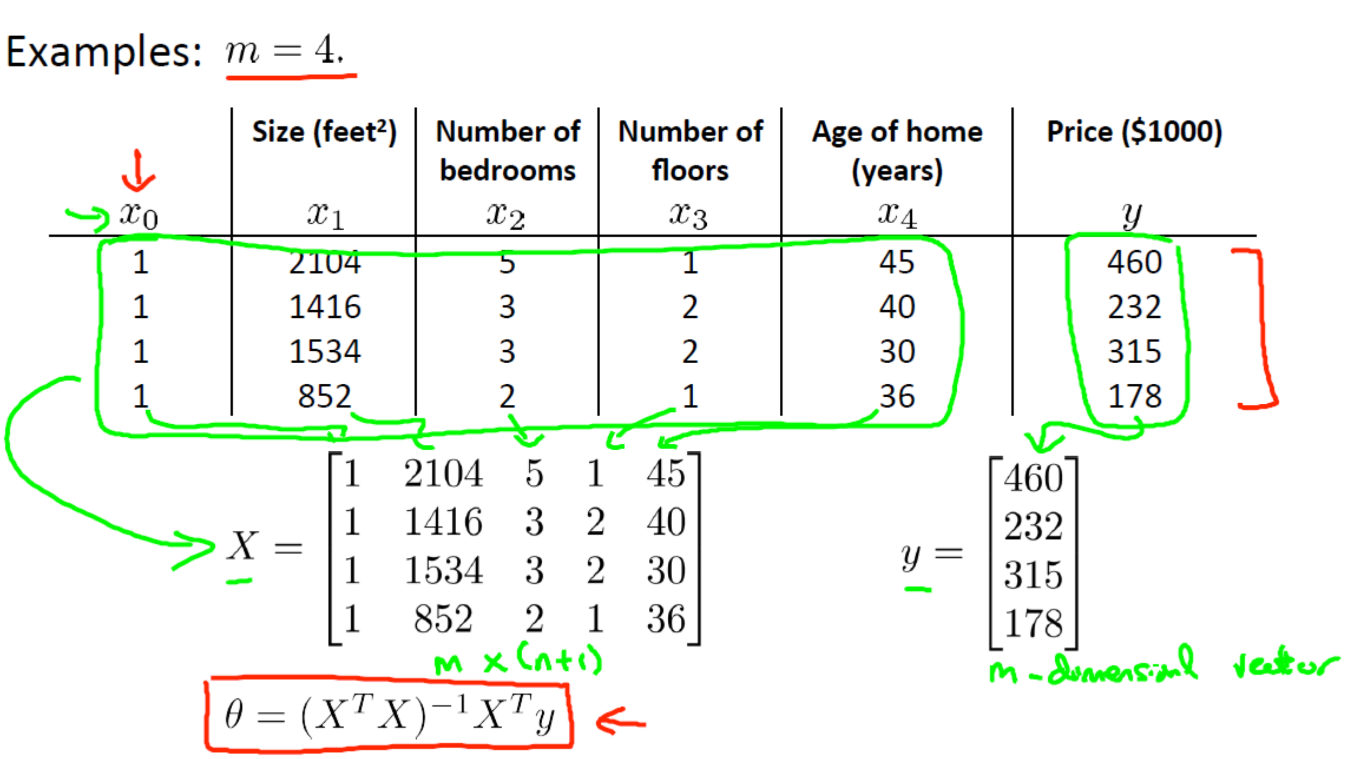


Intuition: If 1D ($\theta \in \mathbb{R}$)

$$J(\theta) = a\theta^2 + b\theta + c$$

$$\frac{d}{d\theta}J(\theta) = \dots \overset{set}{=} 0$$

Solve for θ



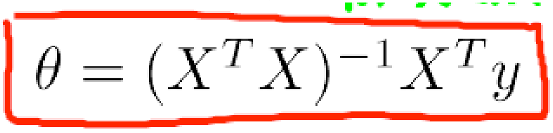
Examples: $m = 4.$

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
x_0	x_1	x_2	x_3	x_4	y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

$$X = \underbrace{\begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}}_{m \times (n+1)} \qquad y = \underbrace{\begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}}_{m\text{-dimensional vector}}$$

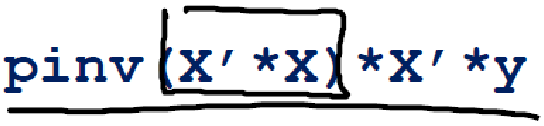
$$\boxed{\theta = (X^T X)^{-1} X^T y}$$

- **What is the mathematical representation for the Normal Equation?**



$$\boxed{\theta = (X^T X)^{-1} X^T y}$$

The octave equivalent is as follows:

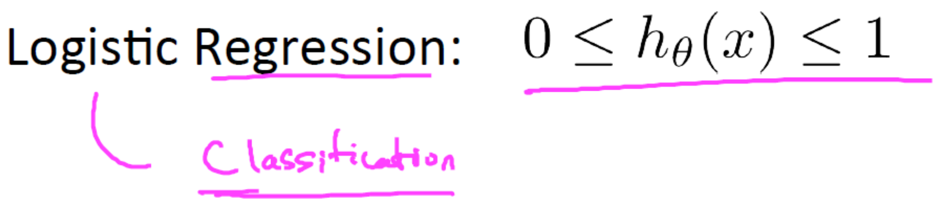


```
pinv(X'*X)*X'*y
```

The ‘pinv’ function calculates the pseudo inverse for a matrix. This takes care of degenerate or singular matrix

- **For the Normal Equation, what happens if X(t)X is non invertible (or degenerate or singular)?**
  - The X(t)X matrix will be non-invertible only when we have:
    - Redundant features – that is features that are linearly dependent.
      - Example: Size in m-sq and size in ft-sq both exist as separate features
    - Too many features – example for a 100 sample record set, we have almost 100, 90 features.
      - This can be fixed by deleting some features or regularization.

- **Define Logistic Regression for Classification.**



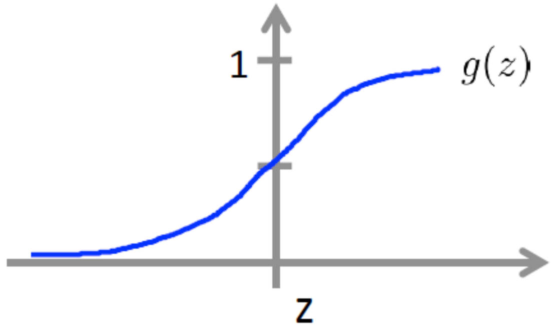
Logistic Regression: $0 \le h_\theta(x) \le 1$

Classification

Logistic regression

$$h_\theta(x) = g(\theta^T x)$$

$$g(z) = \frac{1}{1+e^{-z}}$$



$g(z) \geq 0.5$
when $z \geq 0$

$h_\theta(x) = g(\theta^T x$

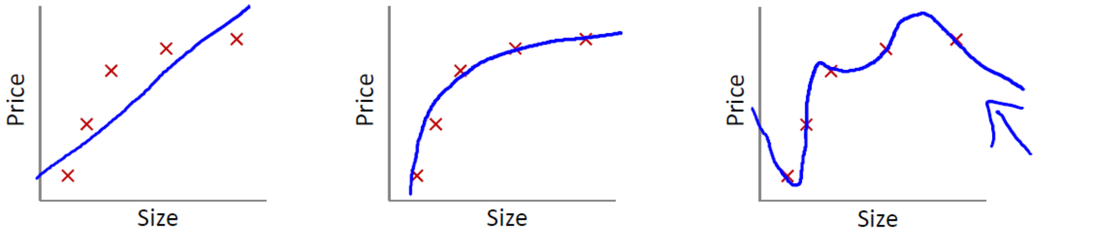
$g(z) < 0.5$
when $z < 0$

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ \boxed{-\log(1 - h_\theta(x))} & \text{if } y = 0 \end{cases}$$

  - <Can cover later on>
  - test

- **What is overfitting and underfitting in the context of LR?**
  - Overfitting: If we have too many features, the learned hypothesis may fit the training set very well but fail to generalize to new examples.
  - Underfitting: On the other hand, with very less number of features the hypothesis may underfit and not be of much use with the new examples, given the lack of accuracy and precision.

Example: Linear regression (housing prices)

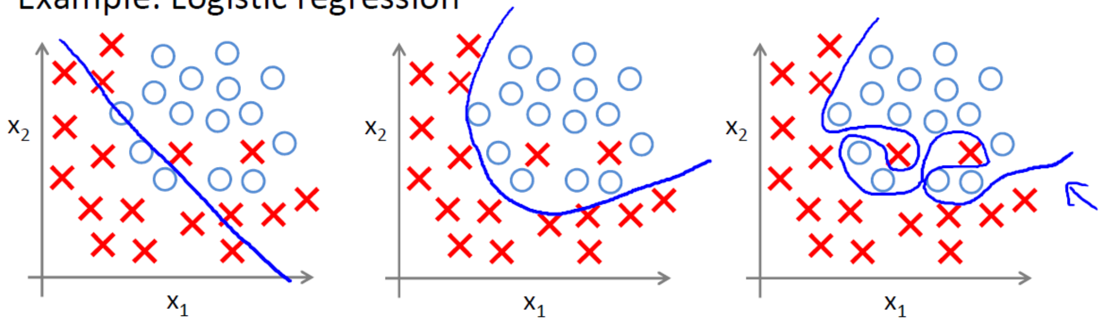


$\rightarrow \theta_0 + \theta_1 x$ "Underfit" "High bias"

$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2$ "Just right"

$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ "Overfit" "High variance"

Example: Logistic regression



$\rightarrow h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

"Underfit"

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots)$

"Overfit"

- **How to address overfitting?**
  - Reduce the number of features
    - Manually select which features to keep.
    - Use model selection algorithms
  - Regularization
    - Keep all the features but reduce magnitude/values of the parameters theta(j)s
    - This works well when we have lot of features, each of which contributes a bit to predicting y.

- **What is Regularization?**
  - The values of the parameters – theta(0), theta(1) , theta(2) , theta(3) , theta(4) … , theta(n) are minimized to insignificant numerical values.
  - Advantages:
    - Simpler hypothesis
    - Less prone to overfitting

Regularization.

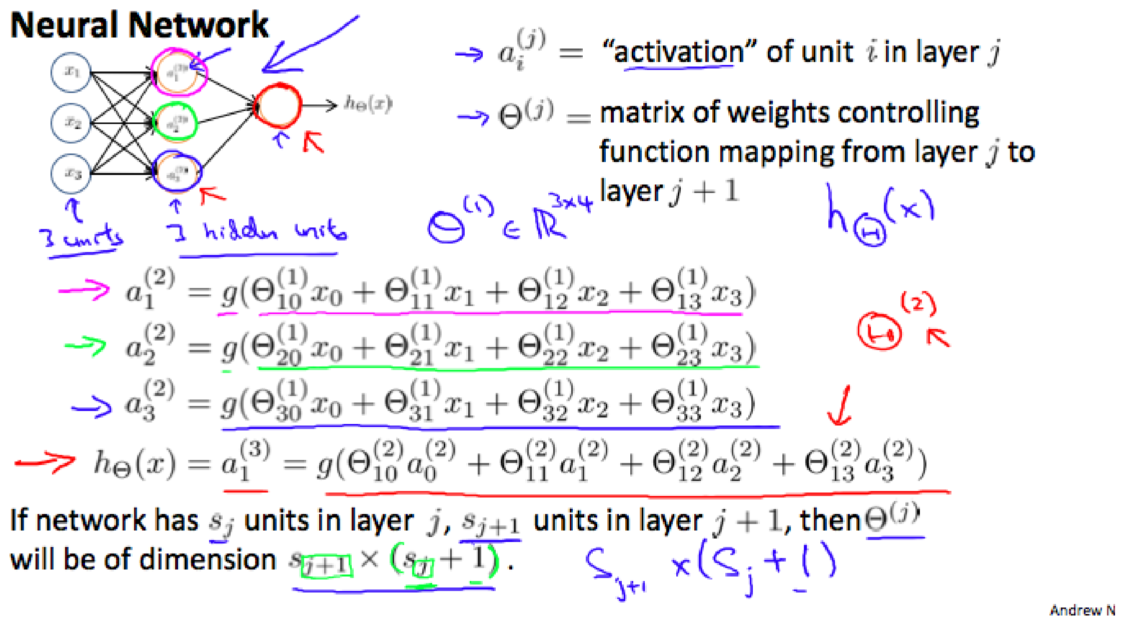
$$\rightarrow J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \theta_j^2\right]$$

regularization parameter

$$\min_\theta J(\theta)$$

- **What is the definition of Neural Networks?**
  - Neural Networks are the algorithms that try to mimic the brain
  - These are extremely simplified versions of how the neural networks within the human brain work

- Neural Network – Intuition:



Neural Network

$a_i^{(j)}$ = "activation" of unit i in layer j

$\Theta^{(j)}$ = matrix of weights controlling function mapping from layer j to layer $j+1$

$\Theta^{(1)} \in \mathbb{R}^{3\times4}$

$h_{\Theta}(x)$

3 units

3 hidden units

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$
$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

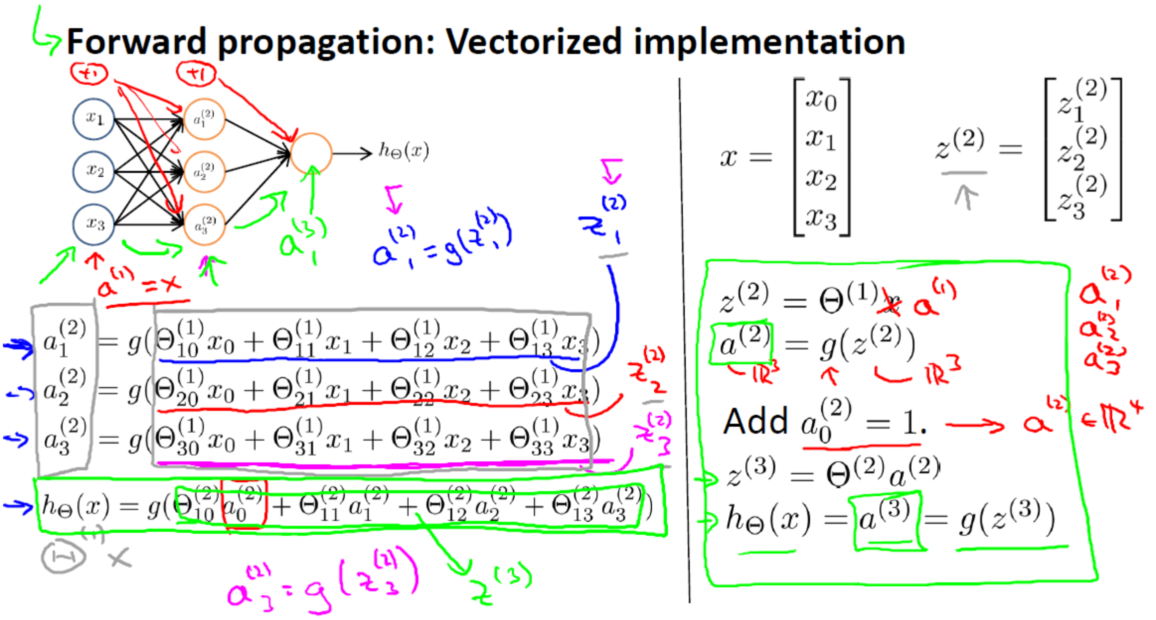
$\Theta^{(2)}$

If network has s_j units in layer j, s_{j+1} units in layer $j+1$, then $\Theta^{(j)}$ will be of dimension $s_{j+1} \times (s_j + 1)$.

$s_{j+1} \times (s_j + 1)$

Andrew N

- **Explain Forward Propagation – vectorised Implementation**



Forward propagation: Vectorized implementation

+1

+1

$a^{(1)} = x$

$a_1^{(3)}$

$a_1^{(2)} = g(z_1^{(2)})$

$z_1^{(2)}$

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$
$$h_\Theta(x) = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

$z_2^{(2)}$ $z_3^{(2)}$

$\Theta^{(1)}$ x

$a_3^{(2)} = g(z_3^{(2)})$

$z^{(3)}$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad z^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{bmatrix}$$

$z^{(2)} = \Theta^{(1)} \cancel{x}\, a^{(1)}$

$a^{(2)} = g(z^{(2)})$ $\mathbb{R}^3$ $\mathbb{R}^3$

Add $a_0^{(2)} = 1$. → $a^{(2)} \in \mathbb{R}^4$

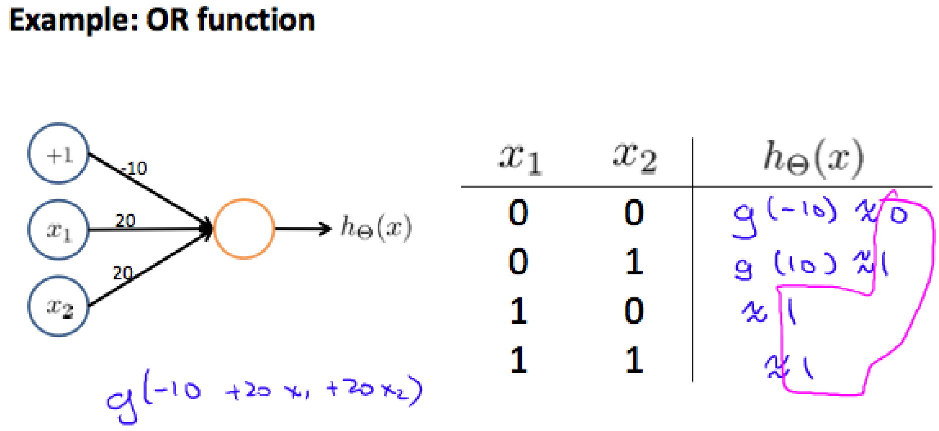
$z^{(3)} = \Theta^{(2)}a^{(2)}$

$h_\Theta(x) = a^{(3)} = g(z^{(3)})$

$a_1^{(2)}$ $a_2^{(2)}$ $a_3^{(2)}$

- test

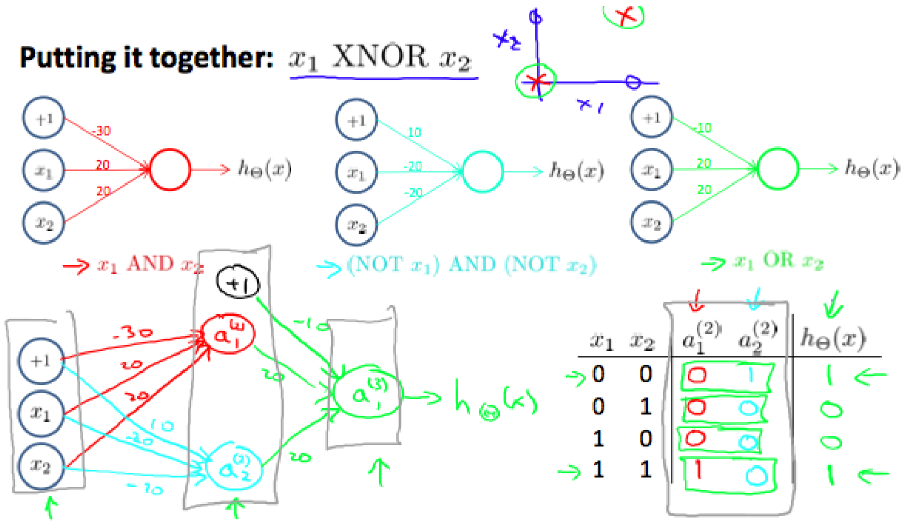
- **Logic Gates – Neural Networks Intuition**



Example: OR function

x_1	x_2	$h_\Theta(x)$
0	0	$g(-10) \approx 0$
0	1	$g(10) \approx 1$
1	0	≈ 1
1	1	≈ 1

$g(-10 + 20x_1 + 20x_2)$



Putting it together: x_1 XNOR x_2

x_1 AND x_2

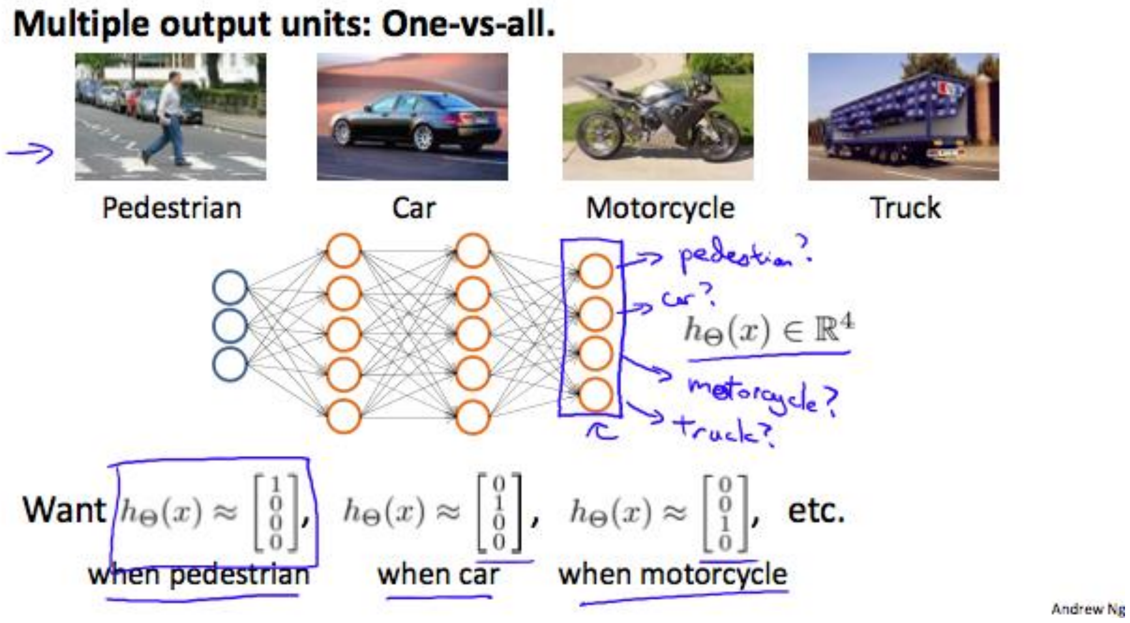
(NOT x_1) AND (NOT x_2)

x_1 OR x_2

x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$h_\Theta(x)$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

- test

- **Multiclass classification using Neural Networks - Intuition**



Multiple output units: One-vs-all.

$h_\Theta(x) \in \mathbb{R}^4$

Want $h_\Theta(x) \approx \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, etc.

when pedestrian when car when motorcycle

Andrew Ng

- test

- **What is Machine Learning?**
  - test

- **What is Machine Learning?**
  - test

- **What is Machine Learning?**
  - test

- **What is Machine Learning?**
  - test

- **What is Machine Learning?**
  - test

- **What is Machine Learning?**
  - test

- **What is Machine Learning?**

    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

- **What is Machine Learning?**
    - test

Machine Learning Online Resources

- **What is the trade-off between bias and variance?**
    - Bias is an error due to erroneous or **overly simplistic assumptions** in the learning algorithm being used. This can lead to the **model under-fitting the data**, making it hard for it to have high predictive accuracy and for you to generalize your knowledge from the training set to testing set.
    - Variance is an error due to **too much complexity** within the learning algorithm. This leads to the algorithm being **highly sensitive to high degree of variation** in the testing data which can lead your model to overfitting the data. We will be carrying too much noise from the training data for the model to be very useful for the test data.

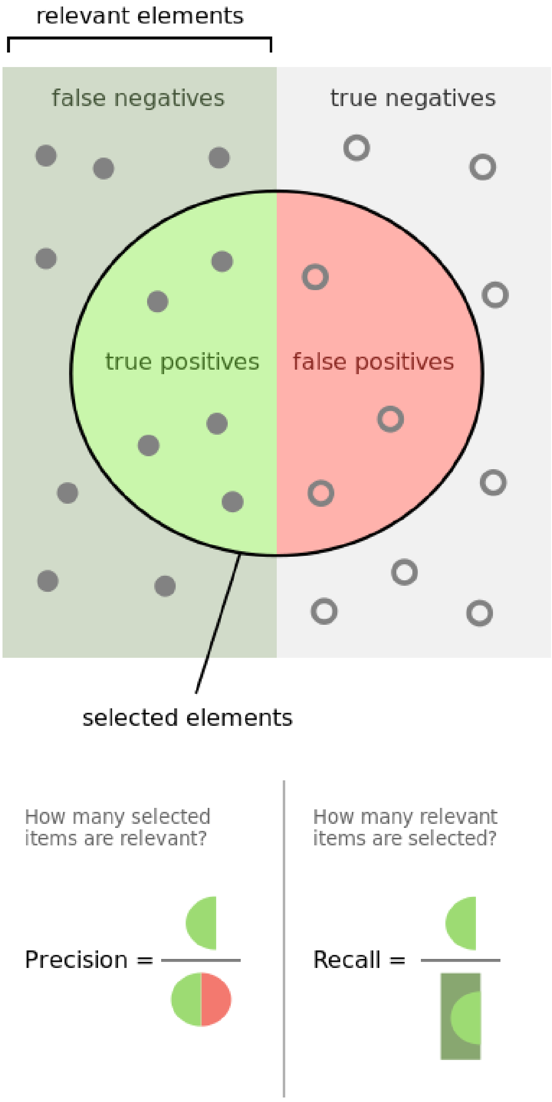
- The bias-variance decomposition essentially decomposes the learning error from any algorithm by adding the bias, the variance and a bit of irreducible error due to noise in the underlying dataset. Essentially, if you make the model more complex and more variables, you'll lose bias but gain some variance – in order to get the optimally reduced amount of error, you'll have to trade-off bias and variance. One would not want either a high variance or high bias within the model.

- **What is the difference between Supervised and Unsupervised Machine Learning?**
  - Supervised learning requires training labelled data.
  - Unsupervised, on the other hand, does not need a labelled data.

- **How is KNN different from k-means clustering?**
  - KNN is a supervised classification algorithm while k-means clustering is an unsupervised clustering algorithm
  - While the mechanisms may seem similar at first, how these are different is that the KNN needs labelled data and the unlabelled data is then classified based on the nearest neighbour (from the labelled set)
  - K-means clustering, on the other hand, only needs a set of unlabelled points and a threshold. The algorithm will take care of unlabelled points and gradually learn how to cluster them into groups by computing the mean of the distance between different points.

- **Explain how ROC (Received Operating Characteristics) curve works.**
  - The ROC curve is a graphical representation of the contrast between true positive rates and the false positive rate at various thresholds.
  - It's often used as a proxy for the trade-off between the sensitivity of the model (true positives) vs the fall-out or the probability it will trigger a false alarm (false positives).

- **Define precision and recall.**
  - Recall: How many relevant items were selected?
    - Recall = (true positives)/(true positives + false negatives)
  - Precision: How many selected items were relevant?
    - Precision = (true positives)/(true positives + false positives)

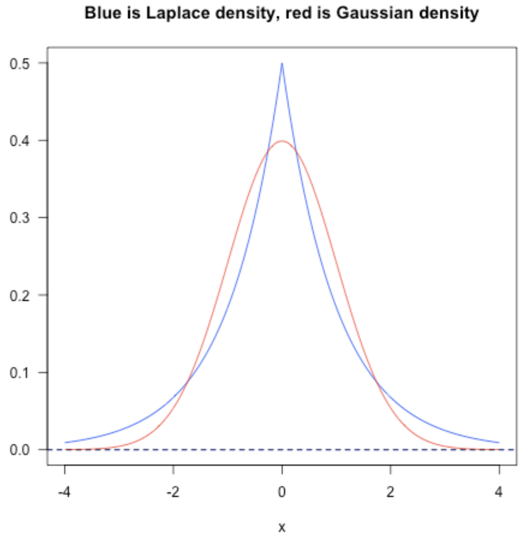


  - Precision and recall have a specific significance with medical diagnosis. A high precision might be critical in scenarios where the diagnosis is a conclusive evidence for a critical following step. Example – a test result decides the severity of treatment or a critical operating procedure.

- **What is Bayes' Theorem? How is it useful in the context of Machine Learning?**
  - Bayes' Theorem gives you the posterior probability of an event give what is known as the prior knowledge.
  - Mathematically, it is expressed as a true positive rate of a condition sample divided by the sum of false positive rate of the population and the true positive rate of the condition.
    - P(A/B) = [P(B/A).P(A)]/P(B)

- **Why is 'Naïve' Bayes naive?**
  - Despite its practical applications, especially in the text mining, Naïve Bayes is considered 'Naïve' because it makes an assumption that is virtually impossible to see in the real-life data: **the conditional probability is calculated as a pure product of individual probabilities of the components**.
  - This implies the absolute independence of features – a condition probably never met in real life.

- **Explain the difference between L1 and L2 regularization.**
  - L2 regularization tends to spread error among all the terms, while L1 is more binary/sparse, with many variables either being assigned a 0 or 1 weighing.
  - L1 corresponding to setting a Laplacean prior on the terms, while L2 corresponds to a Gaussian prior.



- **What is the difference between Type I and Type II error?**
  - Type I – False Positives – claiming that something has happened when it didn't.
  - Type II – False Negatives – claiming that nothing has happened when something has actually.

- **What is a Fourier transform?**
  - A 'Fourier' transform is a generic method to decompose generic functions into a superposition of symmetric functions.
  - A more intuitive analogy is – given a smoothie, how to find out the recipe.

- **What's the difference between probability and likelyhood?**
  The answer depends on whether you are dealing with discrete or continuous random variables. So, I will split my answer accordingly. I will assume that you want some technical details and not necessarily an explanation in plain English. If my assumption is not correct please let me know and I will revise my answer.

  Discrete Random Variables

  Suppose that you have a stochastic process that takes discrete values (e.g., outcomes of tossing a coin 10 times, number of customers who arrive at a store in 10 minutes etc). In such cases, we can calculate the probability of observing a particular set of outcomes by making suitable assumptions about the underlying stochastic process (e.g., probability of coin landing heads is pp and that coin tosses are independent).

  Denote the observed outcomes by OO and the set of parameters that describe the stochastic process as θθ. Thus, when we speak of probability we want to calculate P(O|θ)P(O|θ). In other words, given specific values for θθ, P(O|θ)P(O|θ) is the probability that we would observe the outcomes represented by OO.

  However, when we model a real life stochastic process, we often do not know θθ. We simply observe OO and the goal then is to arrive at an estimate for θθ that would be a plausible choice given the observed outcomes OO. We know that given a value of θθ the probability of observing OO is P(O|θ)P(O|θ). Thus, a 'natural' estimation process is to choose that value of θθ that would maximize the probability that we would actually observe OO. In other words, we find the parameter values θθ that maximize the following function:

  L(θ|O)=P(O|θ)L(θ|O)=P(O|θ)
  L(θ|O)L(θ|O) is called the likelihood function. Notice that by definition the likelihood function is conditioned on the observed OO and that it is a function of the unknown parameters θθ.

  Continuous Random Variables

In the continuous case the situation is similar with one important difference. We can no longer talk about the probability that we observed OO given θθ because in the continuous case P(O|θ)=0P(O|θ)=0. Without getting into technicalities, the basic idea is as follows:

Denote the probability density function (pdf) associated with the outcomes OO as: f(O|θ)f(O|θ). Thus, in the continuous case we estimate θθ given observed outcomes OO by maximizing the following function:

L(θ|O)=f(O|θ)L(θ|O)=f(O|θ)
In this situation, we cannot technically assert that we are finding the parameter value that maximizes the probability that we observe OO as we maximize the PDF associated with the observed outcomes OO.

- **What is deep learning and how does it contrast with other machine learning algorithms?**
    - Deep learning is a subset of Machine Learning that is concerned with Neural networks:
    - How to use ‘backpropogation’ and certain principles from neuroscience to more accurately model large sets of unlabelled or semi-structured data.
    - In that sense, deep learning represents an unsupervised learning algorithm that learns representations of data through the use of neural nets.

- **What is the difference between generative and discriminative model?**
    - A generative model will learn categories of data while a discriminative model will simply learn the distinction between different categories of data.
    - Discriminative models will generally outperform generative models on classification tasks.

- **What cross-validation technique would you use on a time series dataset?**
    - Instead of using the standard k-fold cross-validation, you have to pay attention to the fact that a time series is not randomly distributed data – it is inherently ordered by chronological order.
    - For example, if a pattern emerges in later time periods, the model may still pick up on it, even if that effect doesn’t hold in earlier years.
    - We will have to do something like forward chaining where you’ll be able to model on past data then look at forward-facing data
        - fold 1: training[1], test[2]
        - fold2: training [1 2], test[3]
        - fold3: training[1 2 3], test[4]
        - fold4: training[1 2 3 4], test[5]
        - fold5: training[1 2 3 4 5], test[6]

- **How is the decision tree pruned?**
    - Pruning is what happens in decision trees when branches that have a weak predictive power are removed in order to reduce the complexity of the model and increase the predictive accuracy

- **How is KNN different from k-means clustering?**
    - test

- **How is KNN different from k-means clustering?**
    - test

- **How is KNN different from k-means clustering?**
    - test

- **How is KNN different from k-means clustering?**
    - test

- **How is KNN different from k-means clustering?**
    - test

- **How is KNN different from k-means clustering?**
    - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

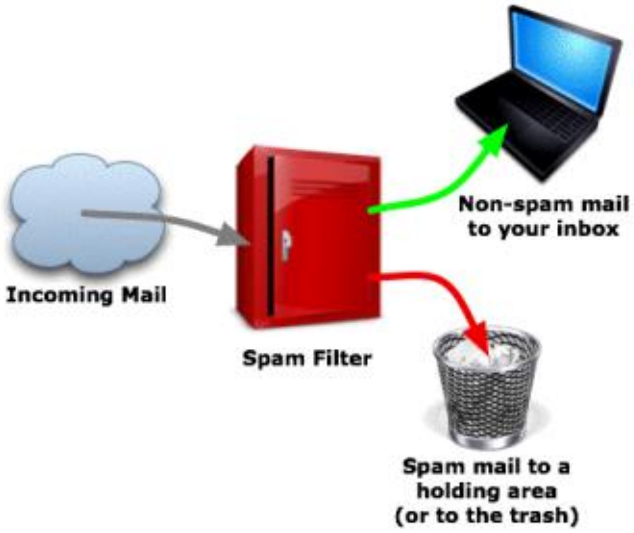
- **How is KNN different from k-means clustering?**
  - test

- **How is KNN different from k-means clustering?**
  - test

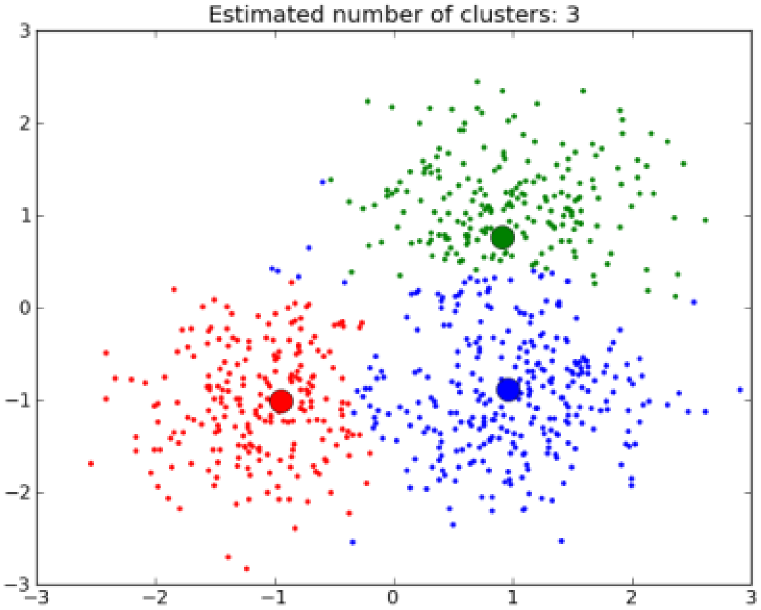
- **How is KNN different from k-means clustering?**
  - test
  -

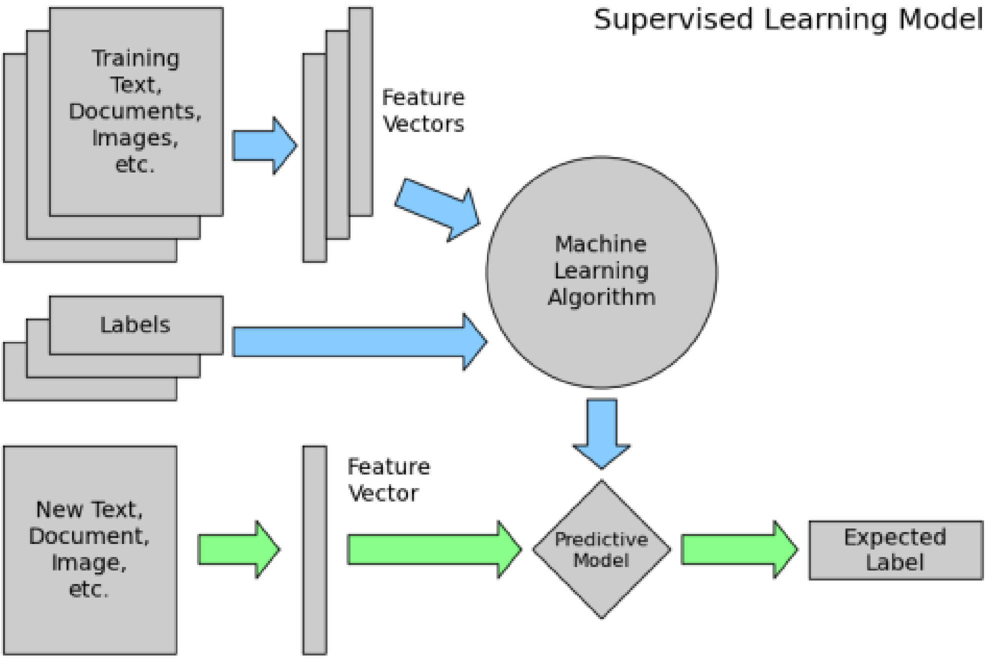
- **What is Machine Learning?**
    - Machine Learning in a 'semi-automated' extraction of knowledge from the data
    - There are three main factors that drive Machine Learning implementations:
        - A business problem that we are looking to get an answer to
        - It involves some amount of automation. An algorithm is fed to the computer which then uses it to process the data to return desired output
        - But, it is not a completely automated process. It needs some intelligent decision making on the part of the data scientist

- **What are the two main categories of Machine Learning?**
    - **Supervised Learning** (aka Predictive Modelling)
        - **Making the predictions based on the data**
        - Example: Is a give email "spam" or "ham"?


Incoming Mail
Spam Filter
Non-spam mail to your inbox
Spam mail to a holding area (or to the trash)

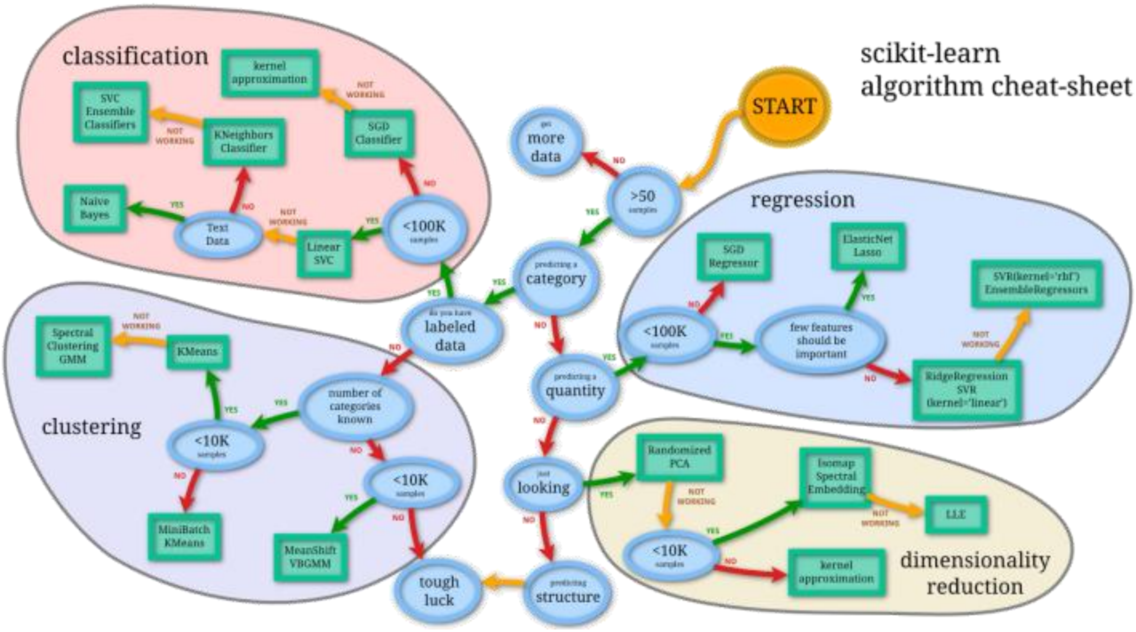
-
    - **Unsupervised Learning**
        - **Extracting some structure from the data**
        - Example: Segment the grocery store customers into clusters that exhibits similar behaviour
        - There is no "right answer"



- **How does the Supervised Machine Learning model works?**
    - Step 1: Train a machine learning model using labelled data
        - The 'labelled data' has been labelled with the outcome
        - The model learns the relationship between the attributes of the data and its outcome
    - Step2: The model then makes predictions on the new data for which the label is unknown
    - 
Supervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
Labels
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Expected Label
    - The primary objective of the supervised machine learning model is to build a model that "generalizes" and accurately predicts the future rather than the past.

- **What the key attributes to consider while finalizing the learning algorithm?**
    - Choose the **right attributes/features** from source data.
    - Choose the **right model** (based on hit and try approach)
    - **Optimize the model** for best performance.
    - Ensure that **the model should generalize to unseen/new data**
    - Estimate **how well the model is likely to perform on unseen data**.

- **Scikit-learn algorithm cheat-sheet.**



- ○

- **What are the key benefits of using scikit-learn for Machine Learning?**
  - Consistent interface to machine learning models
  - Provides with many tuning parameters but with sensible defaults
  - Exceptional documentation
  - Rich set of functionality for companion tasks
  - Active community for development and support

- **What are the drawbacks with using Machine Learning?**
  - Harder than R to get started with Machine Learning
  - Less emphasis (than R) on model interoperability

- **What is the famous iris dataset?**
  - It's a standard example dataset wherein 50 samples of 3 different species of iris (a flower) were collected
  - Total sample set is 50 X 3 = 150 samples
  - Feature Set: sepal length, sepal width, petal length, petal width

- **Code: Display the iris dataset.**

```
from IPython.display import IFrame
IFrame('http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data', width=300, height=200)
```

- **Code: Loading the iris dataset into scikit-learn.**

```
# import load_iris function from datasets module
from sklearn.datasets import load_iris

# save "bunch" object containing iris dataset and its attributes
iris = load_iris()
type(iris)

# print the iris data
print(iris.data)

# print the names of the four features
print(iris.feature_names)
```

```
# print integers representing the species of each observation
print(iris.target)

# print the encoding scheme for species: 0 = setosa, 1 = versicolor, 2 =
virginica
print(iris.target_names)

# check the types of the features and response
print(type(iris.data))
print(type(iris.target))

# check the shape of the features (first dimension = number of observations,
second dimensions = number of features)
print(iris.data.shape)

# check the shape of the response (single dimension matching the number of
observations)
print(iris.target.shape)

# store feature matrix in "X"
X = iris.data
# store response vector in "y"
y = iris.target
```

- **What are the key observations around the iris dataset?**
  - 150 **observations**
  - 4 **features**
  - **Response** variable is the iris species
  - **Classification** problem since response is categorical

- **Define the process steps implementing KNN with scikit-learn?**
  - Pickup a value for k.
  - Search for the k observations in the training dataset that are nearest to the features of the unknown input
  - Use the most popular response value from the k nearest neighbours as the predicted response value.

- **Walkthrough the four step modelling pattern for applying the algorithms (example applying KNN and other models to iris)**
  - Step 1: Import the chass(es) that are planned to be used
    ```
    from sklearn.neighbors import KNeighborsClassifier
    ```
  - Step2: Instantiate the model (or estimator)
    ```
    knn = KNeighborsClassifier(n_neighbors=1)
    print(knn)
    ```
  - Step3: Fit the model with data (aka “model training”)
    ```
    knn.fit(X, y)
    ```
  - Step4: Predict the response for a new observation
    ```
    knn.predict([[3, 5, 4, 2]])
    X_new = [[3, 5, 4, 2], [5, 4, 3, 2]]
    knn.predict(X_new)
    ```
  - Step5: Using a different value for K
    ```
    # instantiate the model (using the value K=5)
    knn = KNeighborsClassifier(n_neighbors=5)

    # fit the model with data
    knn.fit(X, y)
    ```

```
# predict the response for new observations
knn.predict(X_new)
```

  - Step6: Using a different classification model (example: Logistic Regression)

```
# import the class
from sklearn.linear_model import LogisticRegression

# instantiate the model (using the default parameters)
logreg = LogisticRegression()

# fit the model with data
logreg.fit(X, y)

# predict the response for new observations
logreg.predict(X_new)
```

- **What is model evaluation for supervised learning?**
  - For any machine learning scenario with supervised learning, there are some key parameters to consider:
    - Which model to use
    - Best tuning parameters
    - Likely performance of the model

- **Evaluation Procedure #1: Train and test on the entire dataset.**
  - Train the model on the **entire dataset**
  - Test the model on the **same dataset** and evaluate how well we did by comparing the **predicted** response values with the **true** response values
  - **Logistic Regression:**

```
# import the class
from sklearn.linear_model import LogisticRegression

# instantiate the model (using the default parameters)
logreg = LogisticRegression()

# fit the model with data
logreg.fit(X, y)

# predict the response values for the observations in X
logreg.predict(X)

# store the predicted response values
y_pred = logreg.predict(X)

# check how many predictions were generated
len(y_pred)

# compute classification accuracy for the logistic regression model
from sklearn import metrics
print(metrics.accuracy_score(y, y_pred))
```

  - **KNN (K=5):**

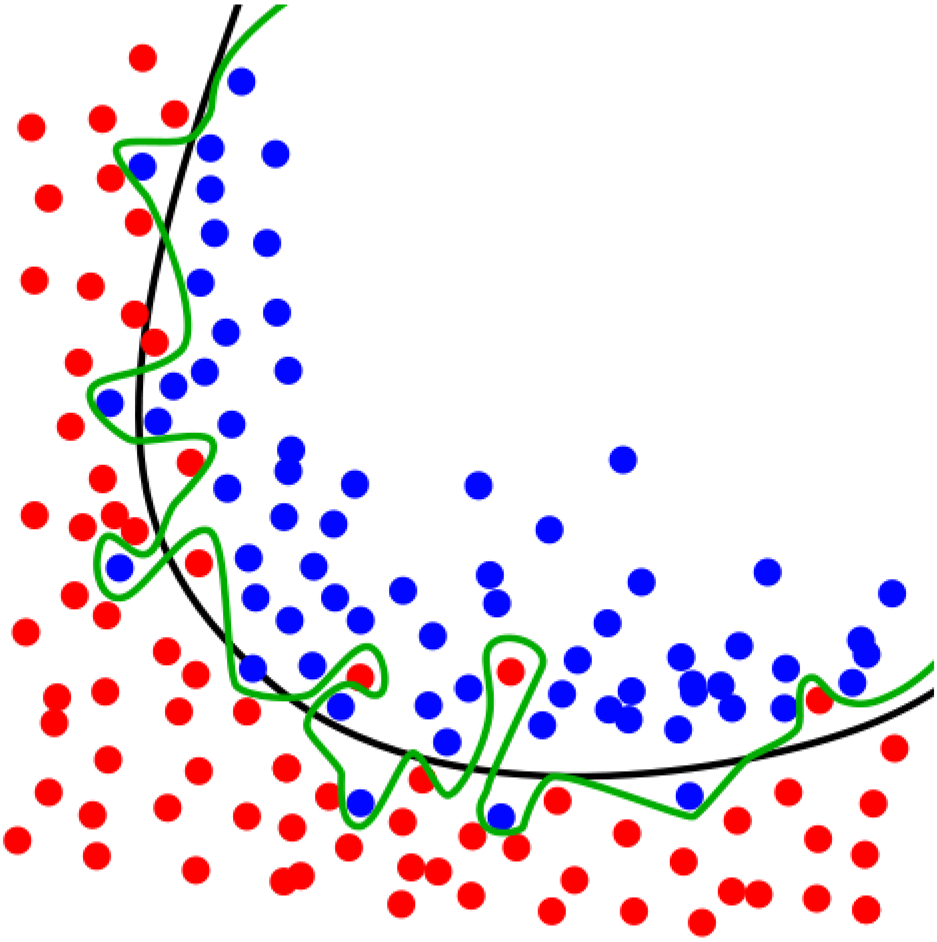
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X, y)
y_pred = knn.predict(X)
print(metrics.accuracy_score(y, y_pred))
```

  - **KNN (K=1):**

```
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X, y)
y_pred = knn.predict(X)
print(metrics.accuracy_score(y, y_pred))
```

    - This is known as training accuracy – when the model is trained and tested on the same dataset

- **What can be the possible issues with training and testing the model on the same dataset?**
    - The true accuracy can be determined only when the model is tested on ‘out-of-sample’ data
    - Training and testing on the same data can cause a high training accuracy, resulting into a model that is overly complex and won’t really generalize
    - The unnecessary complex model will overfit the training data



- **Evaluation Procedure #2: Train/test split**
    - Split the dataset into two pieces – a **training set** and a **testing set**
    - Train the model on the **training set**
    - Test the model on the **testing set.**
    - Evaluate how well it did.

    - **Logistic Regression:**

```
# print the shapes of X and y
print(X.shape)
print(y.shape)

# STEP 1: split X and y into training and testing sets
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=4)

# print the shapes of the new X objects
print(X_train.shape)
print(X_test.shape)

# print the shapes of the new y objects
print(y_train.shape)
print(y_test.shape)
```

```
# STEP 2: train the model on the training set
logreg = LogisticRegression()
logreg.fit(X_train, y_train)

# STEP 3: make predictions on the testing set
y_pred = logreg.predict(X_test)

# compare actual response values (y_test) with predicted response values (
y_pred)
print(metrics.accuracy_score(y_test, y_pred))
```

- **KNN (K=5):**

```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
print(metrics.accuracy_score(y_test, y_pred))
```

- **KNN (K=1):**

```
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
print(metrics.accuracy_score(y_test, y_pred))
```

- Can we locate even better values for k?

```
# try K=1 through K=25 and record testing accuracy
k_range = list(range(1, 26))
scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    scores.append(metrics.accuracy_score(y_test, y_pred))

# import Matplotlib (scientific plotting library)
import matplotlib.pyplot as plt

# allow plots to appear within the notebook
%matplotlib inline

# plot the relationship between K and testing accuracy
plt.plot(k_range, scores)
plt.xlabel('Value of K for KNN')
plt.ylabel('Testing Accuracy')
```

- Observations:
  - Training accuracy rises as the model complexity increases
  - Testing accuracy penalizes models that are too complex or not complex enough
  - For KNN models, complexity is determined by the value of K. Lower value = more complex

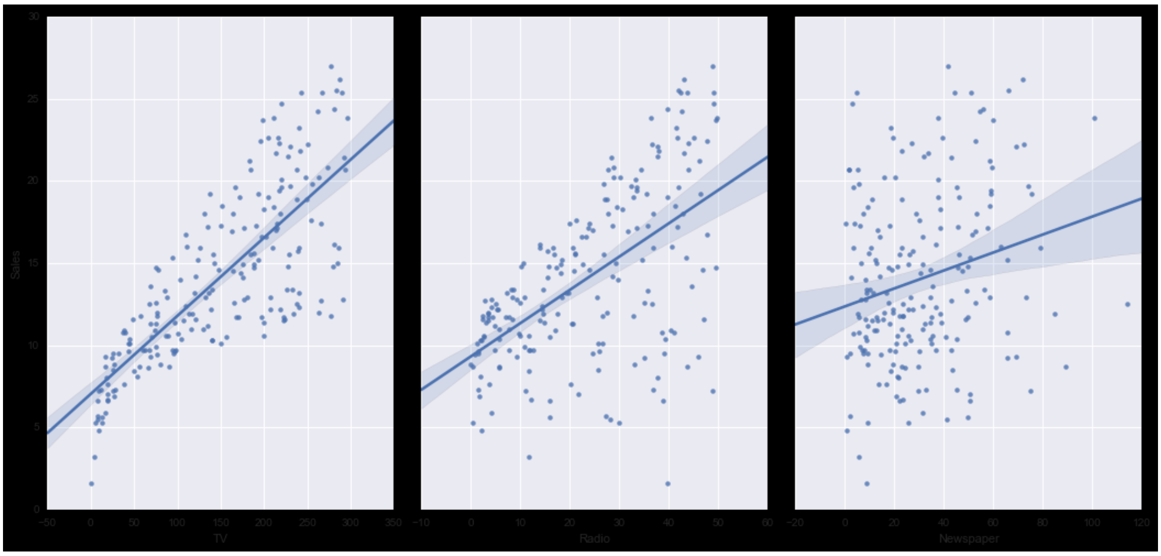
- **What is ‘pandas’?**
  - Pandas is a popular python library for data exploration, manipulation and analysis
  - Sample example of importing and using pandas:

```
# conventional way to import pandas
import pandas as pd

# read CSV file directly from a URL and save the results
data = pd.read_csv('http://www-bcf.usc.edu/~gareth/ISL/Advertising.csv', i
ndex_col=0)
```

```
# display the first 5 rows
data.head()

# display the last 5 rows
data.tail()

# check the shape of the DataFrame (rows, columns)
data.shape
```

- **What is ‘seaborn’?**
  - Seaborn is a python library for statistical data visualizationbuild on top of Matplotlib
  - Sample example of importing and using seaborn:

```
# conventional way to import seaborn
import seaborn as sns

# allow plots to appear within the notebook
%matplotlib inline

# visualize the relationship between the features and the response using s
catterplots
sns.pairplot(data, x_vars=['TV','Radio','Newspaper'], y_vars='Sales', size
=7, aspect=0.7, kind='reg')
```

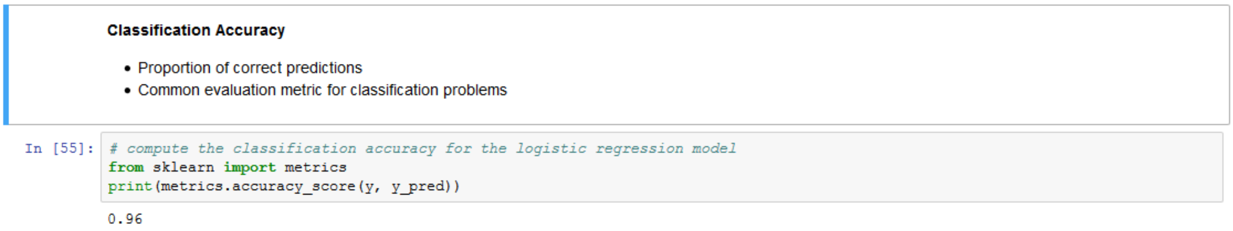


- **What are the key requirements of working with data in scikit-learn?**
  - Features(X) and responses(y) are **separate objects** (X and y)
  - Features and responses should be **numeric**
  - Features and responses should be **NumPy arrays**
  - Features and responses should have **specific shapes**

- **Define the 4-step modelling pattern for scikit-learn.**
  - Step1: Import the class that you plan to use:
  - Step2: Instantiate an object for the class
  - Step3: Fit the model onto the data (features X and response y)
  - Step4: Predict the response for a new ‘out-of-sample’ observation

- **What are the different approaches to evaluate the accuracy of the models**

- ○ **Evaluation procedure #1:** Train and test the model on the entire dataset and check the classification accuracy score. Example:



Classification Accuracy

- Proportion of correct predictions
- Common evaluation metric for classification problems

```
In [55]: # compute the classification accuracy for the logistic regression model
         from sklearn import metrics
         print(metrics.accuracy_score(y, y_pred))

0.96
```

  - Problem with this approach is the complex models overfitting the data and wont generalize
- ○ **Evaluation procedure #2:** Train-test split
  - Split the dataset into two pieces – a training set and a testing set
  - Train the model on the training set
  - Test the model on the testing set and evaluate
- ○
- ○

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

05/07/2018

- **What is a document exploration system?**
  - Document clustering involves the use of descriptors and descriptor extraction.
  - Descriptors are sets of word that describe the contents within the cluster.
  - The application of the document clustering can be categorized into two types – offline and online.
  - Online applications are usually constrained by efficiency problems when compared to offline applications.
  - In general, there are two common algorithms:
    - Hierarchical based algorithm – by aggregating or dividing, the document can be clustered into hierarchical structures suitable for browsing. More accurate, but not efficient.
    - K-mean algorithms and its variants – these are based on the K-mean algorithm. Less accurate, but efficient.
  - Algorithms can be further classified as Hard and Soft.
    - Hard Clustering – Hard clustering computes a hard assignment – each document is assigned to just one cluster.
    - Soft Clustering – The document is distributed across clusters. Example – Dimensionality reduction methods.

- **What is Natural-language processing?**
  - Natural-language processing (NLP) is an area of computer science and AI that is concerned with the interactions between computers and human (natural) languages.
  - In particular, how to program computers to fruitfully process large amount of natural language data.
  - The key challenges around NLP involves – speech recognition, natural-language understanding and natural-language generation.

- **What is Statistical NLP?**
  - It is the integration of machine learning with NLP.
  - The machine learning paradigm calls instead for using statistical inference to automatically learn such rules through the analysis of large corpora (set of documents with human or computer annotations) of typical real-world examples

- **What is Naïve Bayes algorithm?**
  - It is a classification technique based on 'Bayes' Theorem'.
  - It assumes independence among predictors.
  - Naïve Bayes assumes that the presence of one feature in the class is independent of the presence of any other feature.
  - Naïve Bayes is simple, easy to build and useful for very large datasets.

- **How does the Naïve Bayes algorithm works?**
  - Training dataset of weather and corresponding target variable 'Play' (willingness to play)
  - Problem Statement – Classify whether the players will play or not based on weather condition. Steps:
    - Convert the dataset into a frequency table
    - Create likelihood table by finding the probabilities like Overcast probability of overcast weather = 0.29 and probability of playing
  - test

- **What is Deep Learning?**
  - Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks.
  - Deep Learning is a class of machine learning algorithms that:
    - Uses a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input.
    - Learns in supervised (eg. Classification) and/or unsupervised (eg. Pattern analysis) manners.
    - Learns at multiple levels of representations that correspond to different levels of abstractions. The levels form a hierarchy of concepts.
  - Deep learning is a particular kind of Machine Learning that achieves great power and flexibility by learning to represent the world as nested hierarchy of concepts, with each concept defined in relation to simpler concepts and more abstract representations computed in terms of less abstract ones.

- **How can one cluster the documents in an unsupervised way?**
  - test

- **What is TF-IDF?**
  - TF-IDF is short for **term frequency-inverse document frequency** which is a numerical statistic that is intended to reflect how important a word is to a document collection or corpus.
  - It is often used as a weighting factor in searches of information retrieval, text mining and user modelling.
  - The tf-idf value increases proportionally to the number of times a word appears in the document and is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

- **What is LSTM neural network?**
  - LSTM stands for 'Long Short Term Memory'
  - They have proved to be really useful in solving the 'Sequence Prediction' problems.
  - LSTMs have an edge over the conventional feed-forward neural networks and RNN (Recurrent Neural Networks) in many ways.

- **What are Sequence Prediction problems?**
  - Sequence prediction problems are one of the hardest problems to be solved in the data science industry.
  - These include a wide range of problems including – predicting sales, finding patterns in stock market's data, Understanding movie plots, way of speech, language translations, predicting the next word on the iPhone's keyboard.

- **What are word2vec?**
  - Group of related models that are used to produce word embeddings.
  - These are shallow two-layered neural networks that are trained to reconstruct linguistic contexts of words.
  - Word2vec takes as its input, a large corpus of text and produces a vector spaces, with each unique corpus being assigned a corresponding vector in the space.

- **What are Imbalanced Datasets?**
  - Imbalanced data typically refers to a problem with classification problems where the classes are not represented equally.
  - Example:
    - 2-class (binary) classification problem with 100 instances (rows). A total of 80 instances are labelled with Class-1 and remaining 20 instances are labelled with Class-2
    - This is an imbalanced dataset with ratio of Class-1 to Class-2 instances as 4:1.
  - We can have an imbalance problem with both two class and multiclass classifications.

- **What the key tactics in handling imbalanced training data?**
  - Collect more data
  - Try changing the performance metric – Confusion matrix, precision, recall, F1 score, Kappa, ROC curves etc
  - Try resampling the dataset
  - Try generating synthetic samples
  - Try a different algorithm
  - Try penalized models – penalized classification imposes an additional cost on the model for making classification mistakes on the minority class during training.
  - Try a different perspective
  - Try getting creative

- **What is SVM?**
  - SVM or Support Vector Machine is a discriminative classifier formally defined by a separating hyperplane.
  - In other words, given labelled training data (supervised learning), the algorithm outputs an optimal hyperplane which can categorize new examples.
  - In a two dimensional space, this hyperplane is a line dividing a plane into two parts where each class lay in either side.

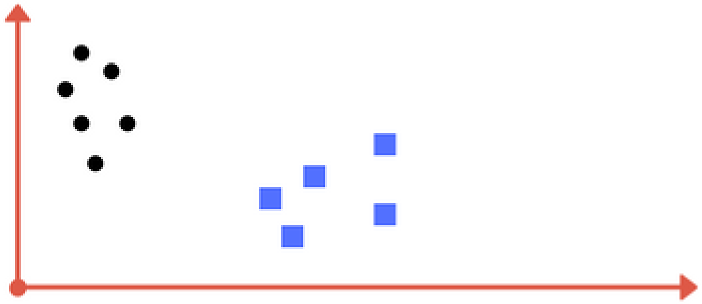


Image A: Draw a line that separates black circles and blue squares.

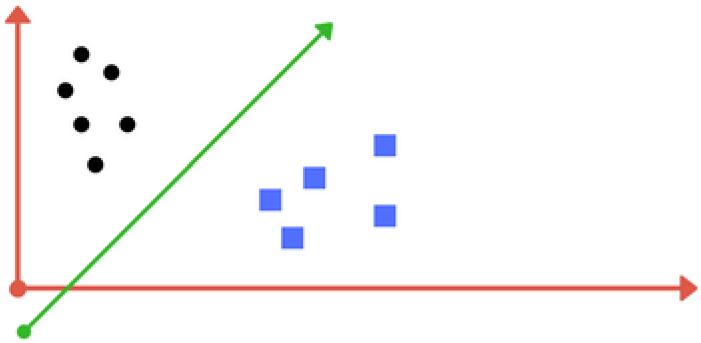
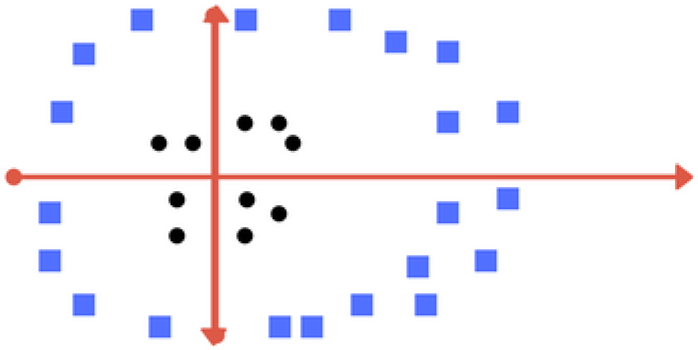
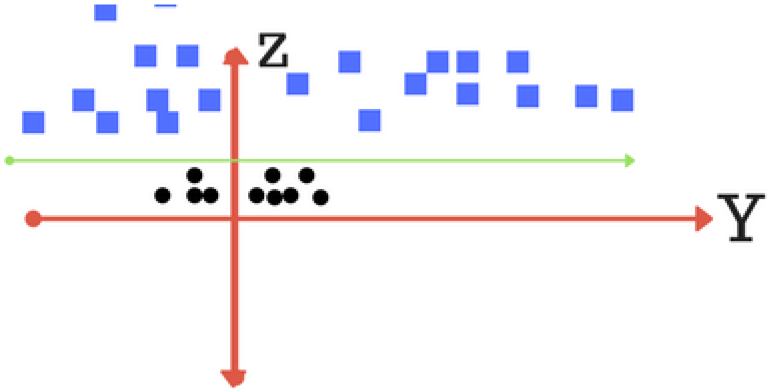


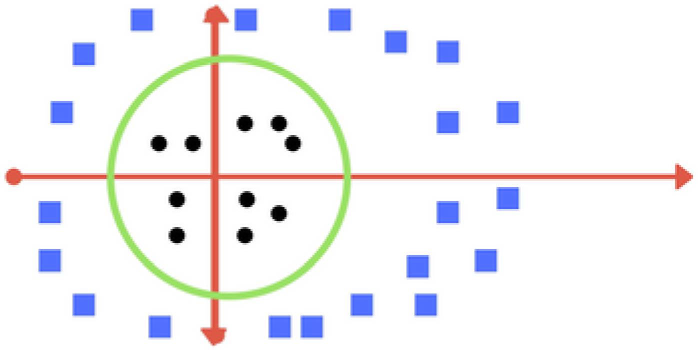
Image B: Sample cut to divide into two classes.



Can you draw a separating line in this plane?



plot of zy axis. A separation can be made here.

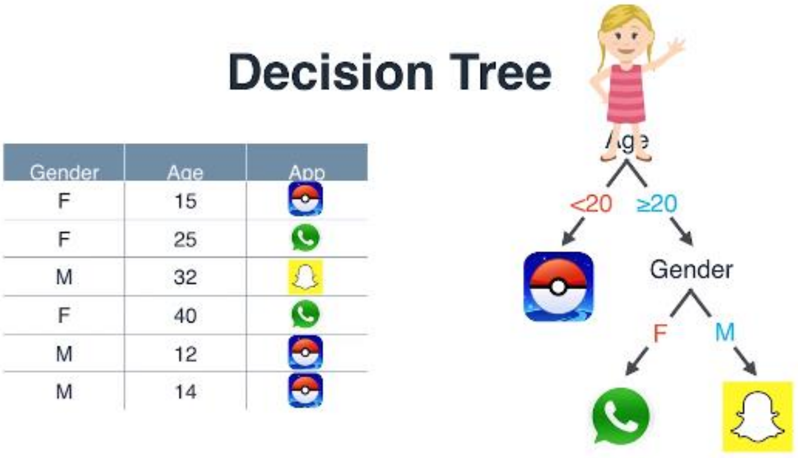


Transforming back to x-y plane, a line transforms to circle.

- **What is an ROC curve? What does the area under the ROC curve signifies?**
  - Received Operating Characteristic (ROC) curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.

05/19/2018

- **What are decision trees?**
  - Decision tree learning is commonly used in data mining.
  - It is a tree like model of decisions and possible consequences, chance event outcomes, resource costs and utility. It is a way to display an algorithm.
  - Decision trees are non-parametric supervised learning method used for classification and regression.
  - Decision Tree Example:



Gender	Age	App
F	15	◓
F	25	📞
M	32	👻
F	40	📞
M	12	◓
M	14	◓

- **What are the types of decision trees?**
  - Classification trees:
    - The predicted outcome is the class to which the data belongs
    - This corresponds to tree models where the target variable can take a finite set of values
  - Regression trees:
    - The predicted outcome can be considered a real number.
    - This corresponds to the tree models where the target variable can take continuous values.

- **What are the pros and cons of decision tree models?**
  - Pros:
    - Easy and simple to understand and interpret
    - Can analyse both numerical and categorical data
  - Cons:
    - Small variation in data can generate a completely different tree.

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**

- test

05/14/2018 – eBook – Hands-on Data Science and Python

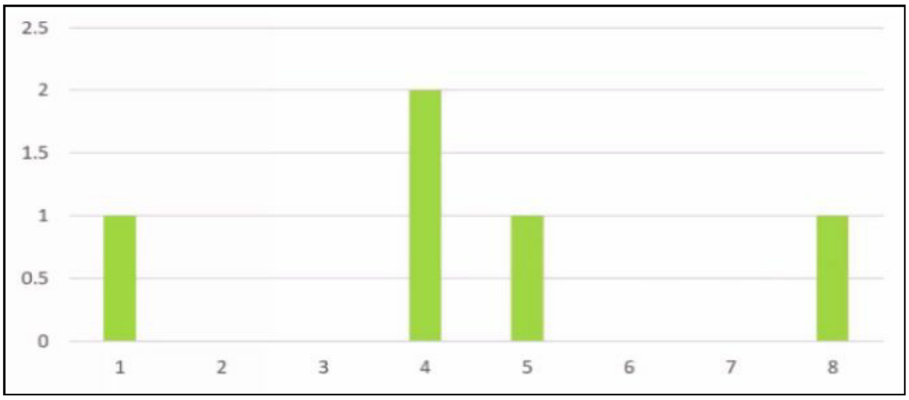
- **What are the different types of 'data'?**
  - Numerical Data – Quantifiable things that can be measured. It is of two types:
    - Discrete – integer based, can only have values in whole numbers. Example: Count of men.
    - Continuous – infinite number of possibilities; can hold fractional values for precision. Example: Height/weights of persons.
  - Categorical Data – Data with no inherent numerical meaning. Categorical data can't be compared with other categories. Example: Gender, States, Product Categories.
  - Ordinal Data – A mixture of numerical and categorical data. Its categorical data that has a mathematical meaning. When order of the data has some importance. Example: Movie ratings on the scale of 1 to 5.

- **Define the following statistical terms – Mean, Median, Mode.**
  - **Mean:** It's just another name for averages.
    - Mean = (Sum of Samples)/(Number of Samples)
  - **Median:** To compute median, sort the dataset and the value that ends up being in the middle will be the median.
  - One line – Take the data, sort is numerically and take the centre point.
  - In case of even number of data points, the median would not be one middle value but two values. Hence the average of the two values will give us the median.
    - Example: 0, 2, 3, 2, 1, 0, 0, 2, 0. Sorted: 0, 0, 0, 0, **1**, 2, 2, 2, 3. Median: 1
  - **Mode:** A mode is the most common value within the dataset.
    - Example: 0, 2, 3, 2, 1, 0, 0, 2, 0. Counts: 0 -> 4, 1 -> 1, 2 -> 3, 3-> 1. Mode: 0
  - Mode is only relevant to 'discrete' data.

- **How is Median less susceptible to outliers than Mean?**
  - The mean value is a mathematical average of all the values within the dataset. Hence, a value which is either significantly large or significantly small, might impact the mean drastically.
  - On the other hand, median value will most likely still remain same or close to previous value with an outlier added as it's just the middle value of the sorted list of input. The outlier most likely will be settling down at the extreme ends of the list.

- **Define the following statistical terms – Variance, Standard Deviation.**
  - **Variance (sigma-square):** It measures, how spread-out the data is.
  - Variance is the average of squared difference from the mean.
  - **Standard Deviation (sigma)**: It is the square root of variance.
  - Example:
    - Dataset: [1, 4, 5, 4, 8]
    - Mean = (1+4+5+4+8)/5 = 4.4
    - Difference from Mean: [-3.4, -0.4, 0.6, -0.4, 3.6]
    - Squared Difference: [11.56, 0.16, 0.36, 0.16, 12.96]
    - Variance = (11.56+0.16+0.36+0.16+12.96)/5 = 5.04
    - Standard Deviation = Square-root(5.04) = 2.24

- **How do we identify outliers using Standard Deviation?**
  - For the same example above, below is the histogram:



  - Here is the frequency of numbers, based on occurrences: 4 -> twice; 1, 5, 8 -> once
  - Anything outside 1 standard deviation from the mean can be considered as an outlier.
  - In this case: SD = 2.24, Mean = 4.4. Hence Range: (4.4 – 2.24) <-> (4.4+2.24) ~=~ 2 <-> 7
  - The values 1 and 8 lie outside the range and hence are outliers

- **What is the difference between Population variance (sigma squared) vs Sample variance (S squared)?**
  - Population variance is calculated for the entire dataset, as a whole.

$$\sigma^2 = \frac{\sum(X-\mu)^2}{N}$$

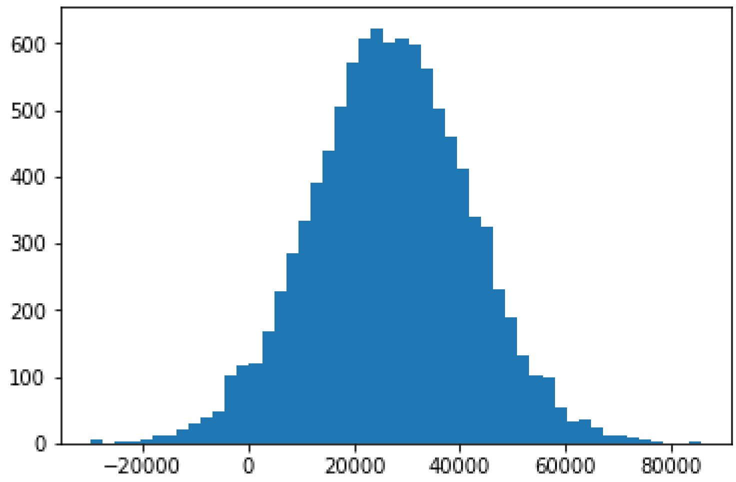
  - Where:
    - X: Each data point
    - mu: the mean
    - N: # of data points
  - Sample variance, on the other hand, is calculated on a subset of data.

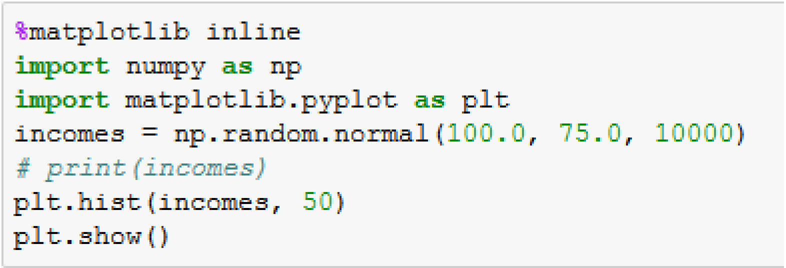
$$s^2 = \frac{\sum(X-M)^2}{N-1}$$

  - Where:
    - X: Each data point
    - mu: the mean
    - N-1: # of data points minus 1

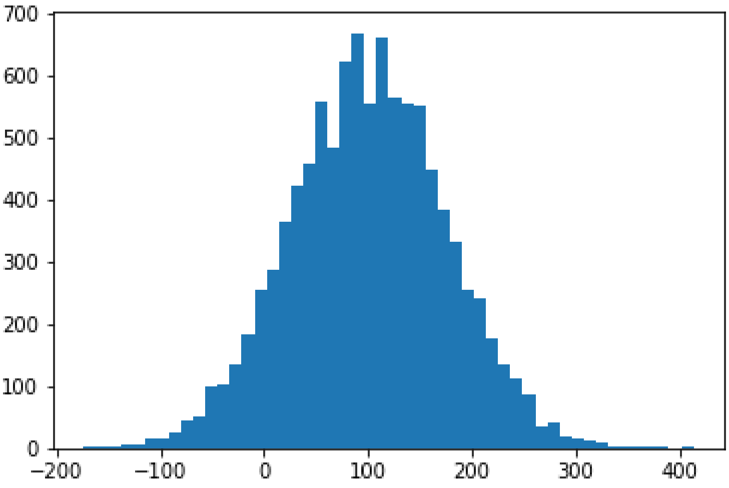
- **Code: Different types of distribution curves**
  - **Normal Probability Density Distribution**

```
%matplotlib inline
import matplotlib.pyplot as plt

incomes = np.random.normal(27000, 15000, 10000)
plt.hist(incomes, 50)
plt.show()
```

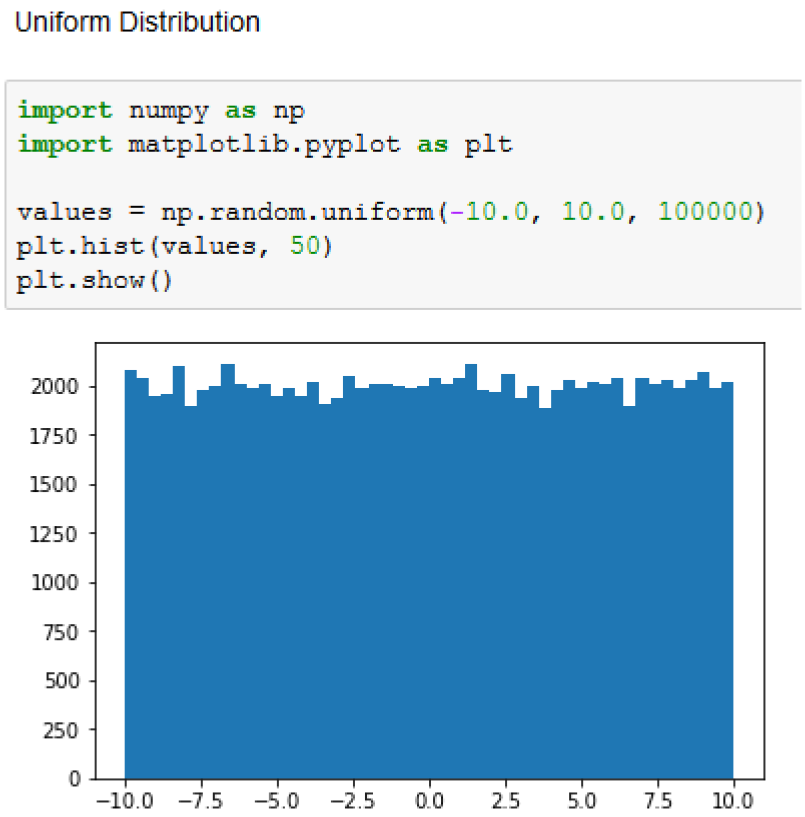




```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
incomes = np.random.normal(100.0, 75.0, 10000)
# print(incomes)
plt.hist(incomes, 50)
plt.show()
```



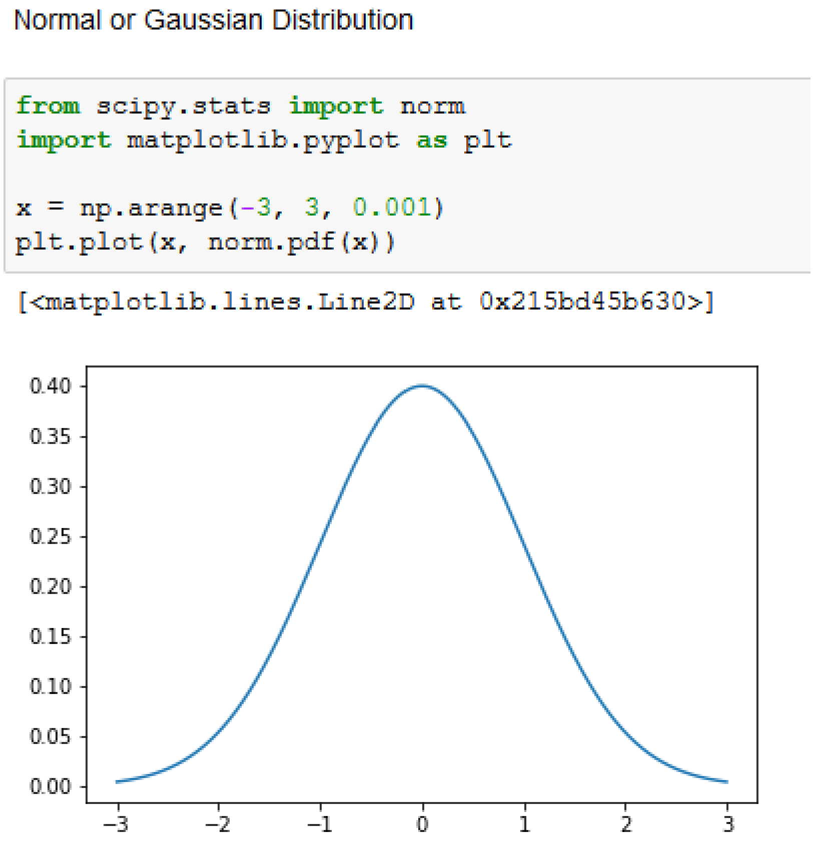
- **Uniform Probability Density Distribution**



Uniform Distribution

```
import numpy as np
import matplotlib.pyplot as plt

values = np.random.uniform(-10.0, 10.0, 100000)
plt.hist(values, 50)
plt.show()
```

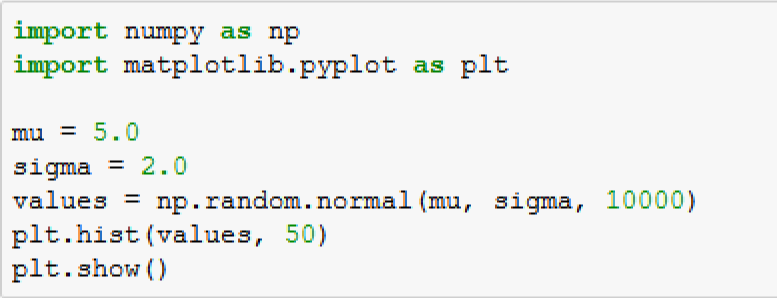
- **Normal or Gaussian Probability Density Distribution**



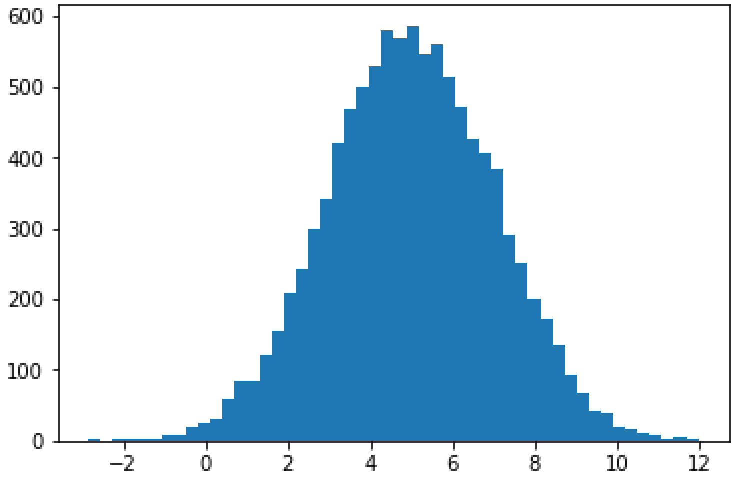
Normal or Gaussian Distribution

```
from scipy.stats import norm
import matplotlib.pyplot as plt

x = np.arange(-3, 3, 0.001)
plt.plot(x, norm.pdf(x))
```

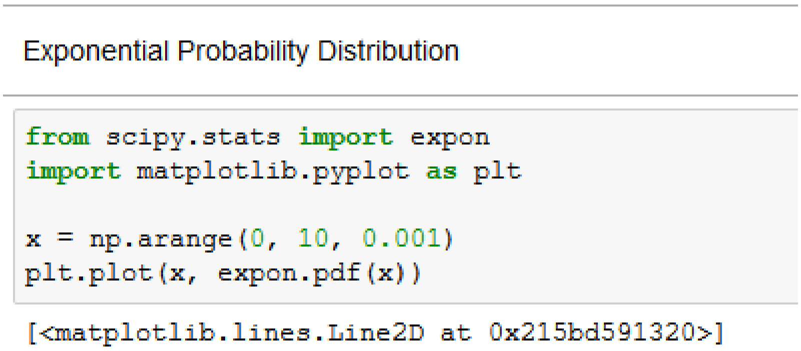
```
[<matplotlib.lines.Line2D at 0x215bd45b630>]
```



```
import numpy as np
import matplotlib.pyplot as plt

mu = 5.0
sigma = 2.0
values = np.random.normal(mu, sigma, 10000)
plt.hist(values, 50)
plt.show()
```



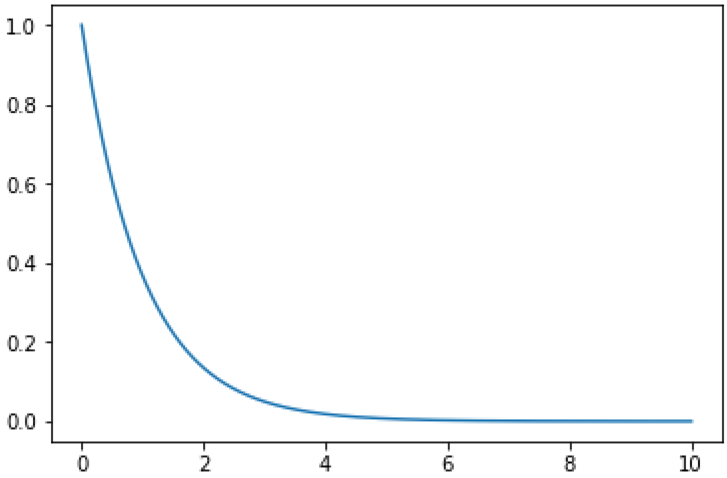
- **Exponential Probability Density Distribution**



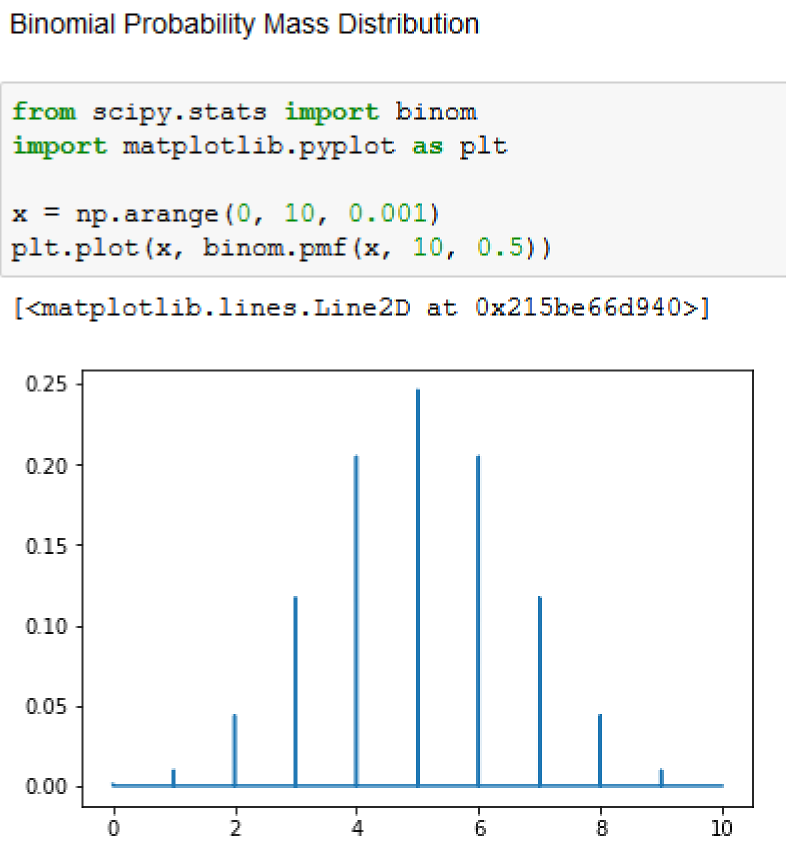
Exponential Probability Distribution

```
from scipy.stats import expon
import matplotlib.pyplot as plt

x = np.arange(0, 10, 0.001)
plt.plot(x, expon.pdf(x))
```

```
[<matplotlib.lines.Line2D at 0x215bd591320>]
```



- **Binomial Probability Mass Distribution**

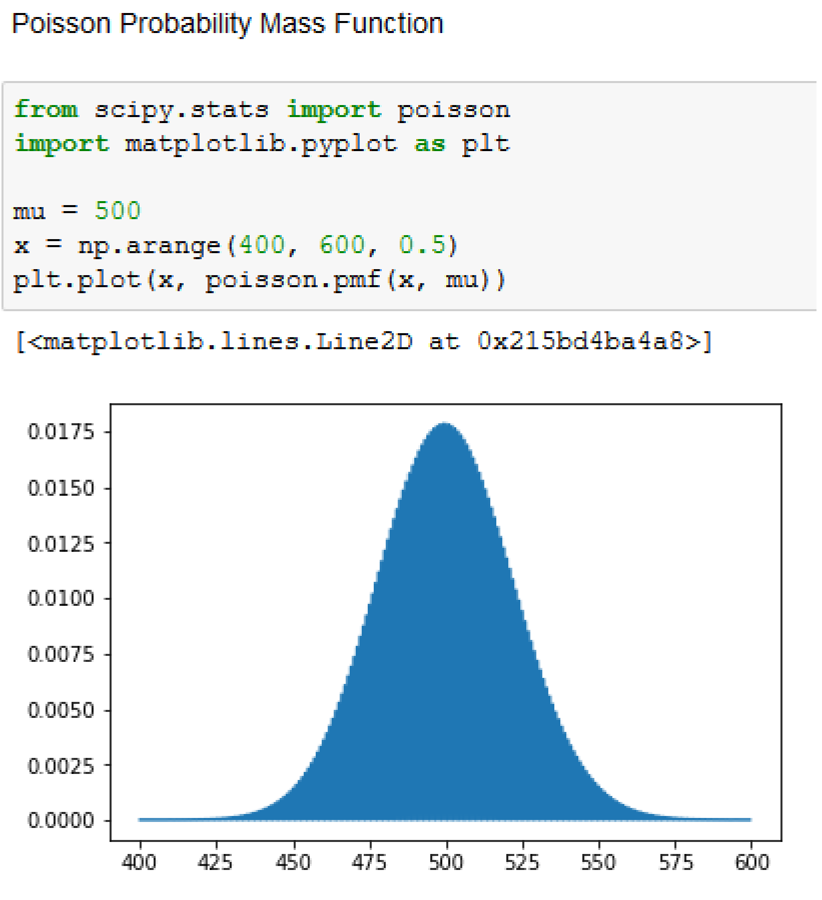


Binomial Probability Mass Distribution

```
from scipy.stats import binom
import matplotlib.pyplot as plt

x = np.arange(0, 10, 0.001)
plt.plot(x, binom.pmf(x, 10, 0.5))
```

```
[<matplotlib.lines.Line2D at 0x215be66d940>]
```

- ○ **Poisson Probability Mass Function**



Poisson Probability Mass Function

```
from scipy.stats import poisson
import matplotlib.pyplot as plt

mu = 500
x = np.arange(400, 600, 0.5)
plt.plot(x, poisson.pmf(x, mu))
```

```
[<matplotlib.lines.Line2D at 0x215bd4ba4a8>]
```

- ○ test

- **How does the Supervised Machine Learning model works?**
  - ○ test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

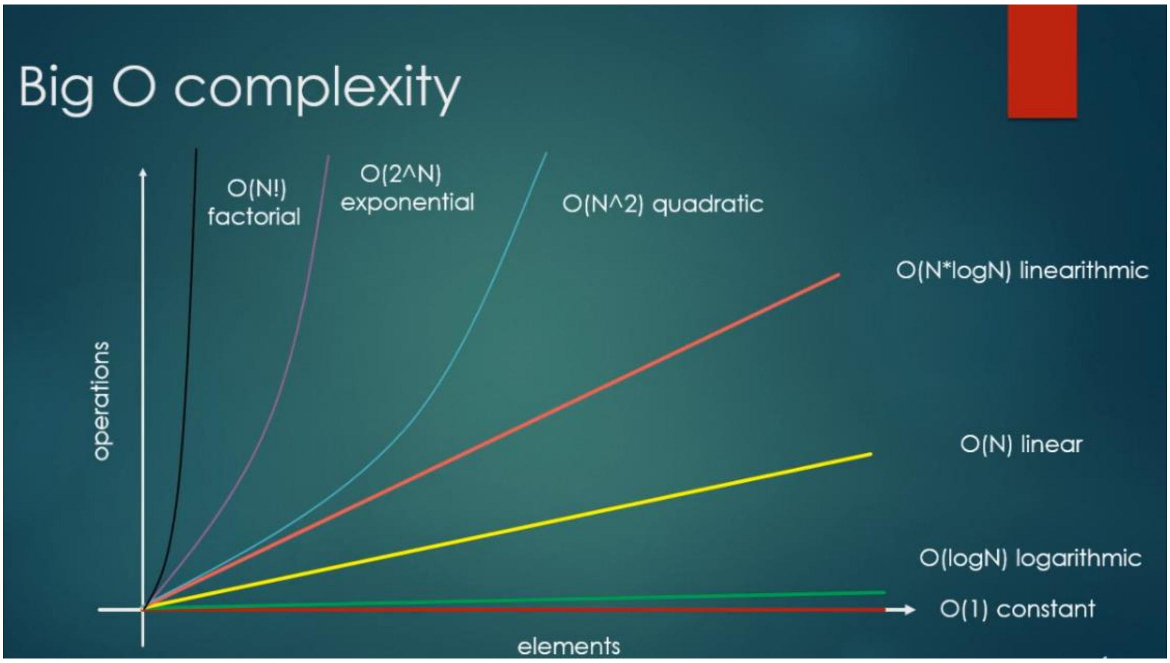
- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

05/20/2018 – Algorithms and Data Structures in Python Videos

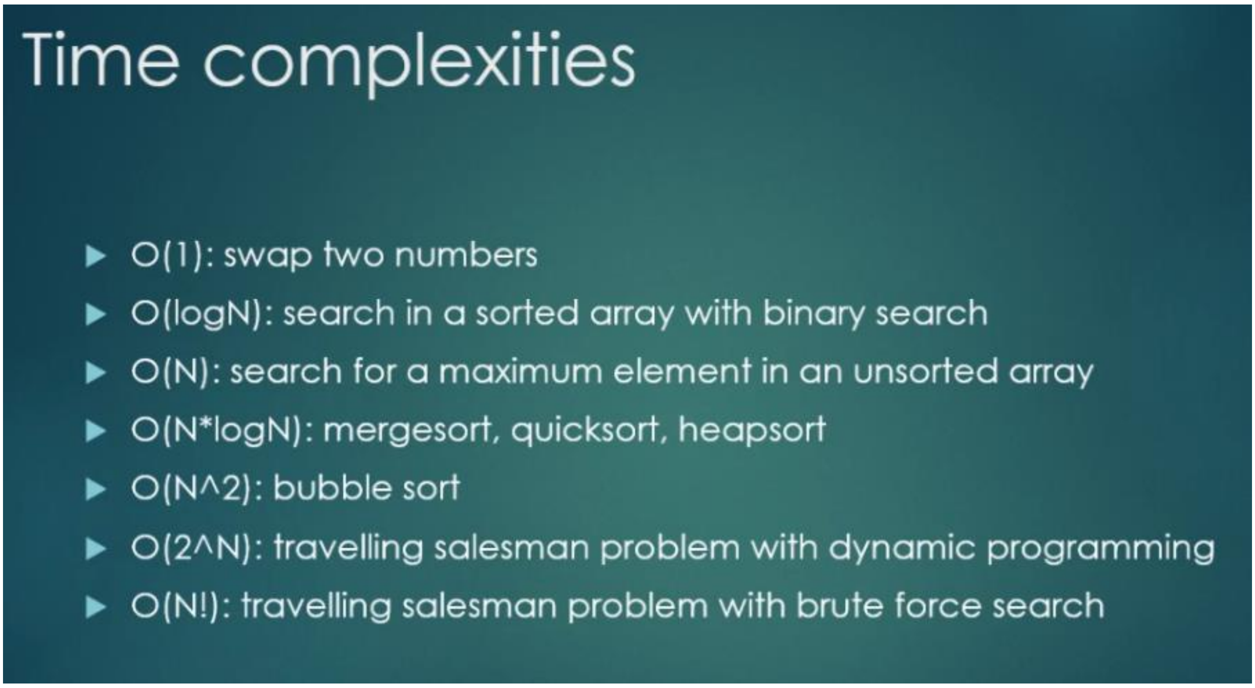
- **What is the Big-O notation?**
  - Also called the Landau notation
  - Describes the limiting behaviour of the function when the argument tends towards infinity
    - f(x) = O(g(x)) as x -> infinity
  - Rules:
    - If f(x) is a sum of several terms, the one with the largest growth rate is kept and all others are omitted.
    - If f(x) is a product of several factors, any constants are omitted.

  - Different types of Big-O complexities



Big O complexity

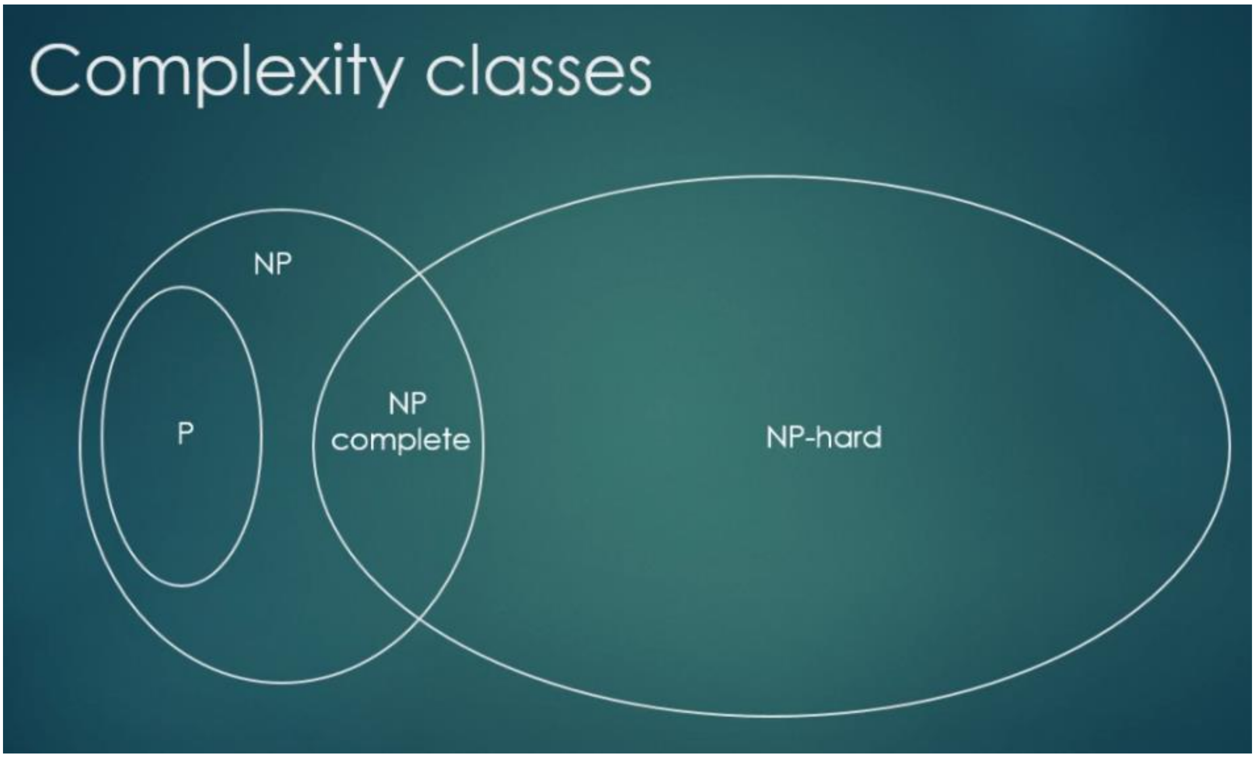
Their examples:



Time complexities

- O(1): swap two numbers
- O(logN): search in a sorted array with binary search
- O(N): search for a maximum element in an unsorted array
- O(N*logN): mergesort, quicksort, heapsort
- O(N^2): bubble sort
- O(2^N): travelling salesman problem with dynamic programming
- O(N!): travelling salesman problem with brute force search

○ Complexity Classes:


Complexity classes

- P (Polynomial):
    - most fundamental complexity class
    - contains all the decision problems that can be solved via a deterministic Turing machine
    - class of computational problems that are efficiently solvable
    - Example: sorting algorithms
- NP (Nondeterministic Polynomial):
    - If we have a solution to a problem, we can verify this solution in polynomial time (by a deterministic Turing machine)
    - Complexity class P is contained in NP
    - Most important question: N = NP. Is it true?
    - Examples: Integer factorization, travelling salesman etc
- NP-complete:
    - A decision problem is NP-complete when it is both in NP and NP-hard
    - Although any given solution to an NP-complete problem can be verified in polynomial time, there is no known efficient way to locate a solution in the first place
    - We usually just look for an approximate solution – heuristics
    - Example: Chinese postman problem, graph colouring, Hamiltonian cycle
- NP-hard:
    - These are at least as hard as the hardest problems in NP
    - A problem H is NP-hard when every problem L in NP can be reduced in polynomial time to H
    - As a consequence, finding a polynomial algorithm to solve any NP-hard problem would give polynomial algorithms for all the problems in NP.
    - Example: Halting problem – it asks ‘given a program and its input, will it run forever?’ That is a yes/no question, so it’s a decision problem.

- **What is a linked list?**
    - It is a type of data structure where in each node is composed of a data and a reference/link to the next node in the sequence
    - A linked list is a base for some other data structures: stacks, queues

- **Compare: Array vs Linked List?**

Attribute	Array	Linked List
Size	Fixed	Dynamic
Operations: Insertion, Deletion	Complex – need to traverse the array	Relatively easy. No relocation or reorganization
Indexing	O(1)	O(n)
Insert at the beginning	O(n)	O(1)
Waste space	0	O(n)
IF inserting/removing elements at the beginning	Do not use arrays	Use linked-list
If the size is changing frequently	Do not use arrays	Use linked-list
Need random access	Use arrays O(1)	Do not use linked-list

- Advantages of linked-list:
    - Linked-list are dynamic data structures (arrays are not !!!)
    - It can allocate the needed memory in run-time
    - Very efficient if we want to manipulate the first elements
    - Easy implementation
    - Can store items with different sizes. An array assumes every element to be exactly the same
    - It's easier for a linked list to grow organically. An array's size needs to be known ahead of time, or re-created when it needs to grow
- Disadvantages of linked-list:
    - Wastes memory because of the references (pointers)
    - Nodes in a linked list must be read in order from the beginning as linked-lists have sequential access
    - Reverse traversing a singly-linked-list is very difficult. A doubly-linked-list is easier to read

- **What is a Binary Search Tree (BST)?**
    - A BST is used to implement lookup tables.
    - Keeps the keys in sorted order so that the lookup and other operations can use the principle of binary search
    - Each comparison allows operations to skip over half of the tree, so that each lookup/insertion/deletion takes time proportional to log (function) of the number of items stored in the tree
    - This is much better than the linear time O(n) required to find items by key in an unsorted array but still slower than the corresponding operations on the hash tables.
    - Insertion: if the key is not equal to that of the root, we search the left or right sub-trees recursively
    - Deletion: soft delete. Does not actually remove the item but just mark it as deleted. Hence, not so efficient

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**

    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**

    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**
    - test

- **How does the Supervised Machine Learning model works?**

  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test

- **How does the Supervised Machine Learning model works?**
  - test