Datacamp Python

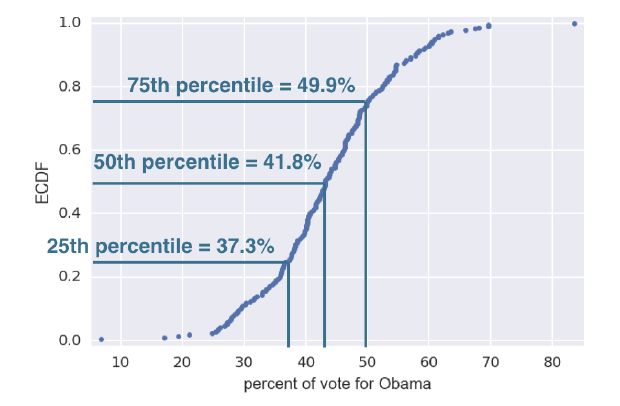
# Statistical Thinking in Python Part 1

## Exploratory Data Analysis

### ECDF

Empirical Cumulative Distribution Function is the distribution function associated with the empirical measure of a sample. The x axis are the observations, sorted from the smallest to the largest. The y axis is the percentage of observation smaller than or equal to the specified value.

For example



25% of observations has value less than or equal to 37.3

50% of observations has value less than or equal to 41.8

75% of observation has value less than or equal to 49.9

## Quantitative exploratory data analysis

Mean and median

Percentiles, outliers and box plots

Variance and standard deviation

Covariance and the Pearson correlation coefficient

## Thinking probabilistically – Discrete variables

**Probability** allows us to describe **uncertainty**.

When we measure 50 flowers for their pedal length, we get a mean value. But if I measure another 50 flowers of the same species, what would the mean pedal length be?

This is the heart of **statical inference**. It is the process by which we go from measured data to probabilistic conclusion about what we might expect if we collected the same data again.

### Binomial Random Variable

In order for a variable to be a binomial random variable,

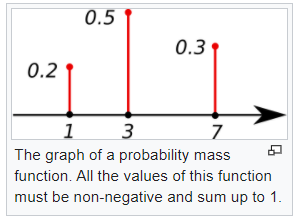
* Each trail must be independent
* Each trail can be called a “success” or a “failure”
* There area a fixed number of trials
* The probability of success on each trial is constant

### Bernoulli trial

An experiment that has two options, “success” and “failure”.

### Probability Mass Function (PMF)

A probability mass function is a function that gives the probability that a discrete random variable is exactly equal to some value.



P(X = 1) = 0.2

P(X = 3) = 0.5

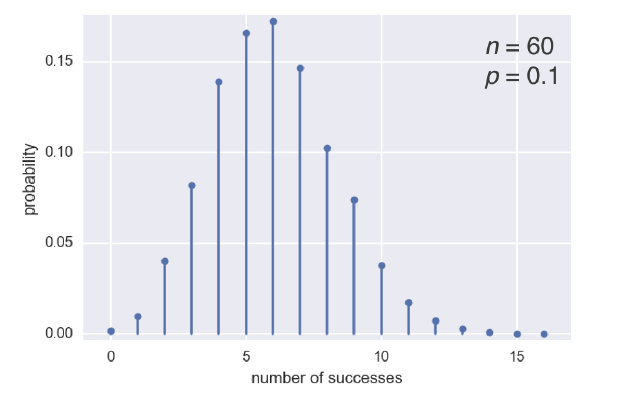
P(X = 7) = 0.3

The discrete random variable X has 0.5 probability to have the value 3.

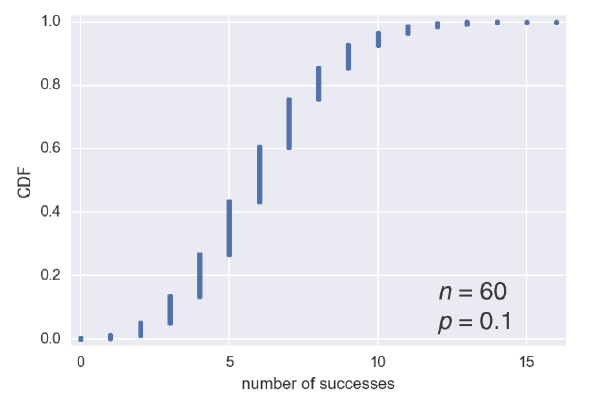
### Binomial Distribution

The number r of success in n Bernoulli trials with probability of p of success, is Binomially distributed.

##### The following is the Binomial PMF



##### And the Binomial CDF



### The Poisson process

A Poison process describes the number of times an event occurs in a period of time, or in a particular area, or over some distance or within any other kind of measurement.

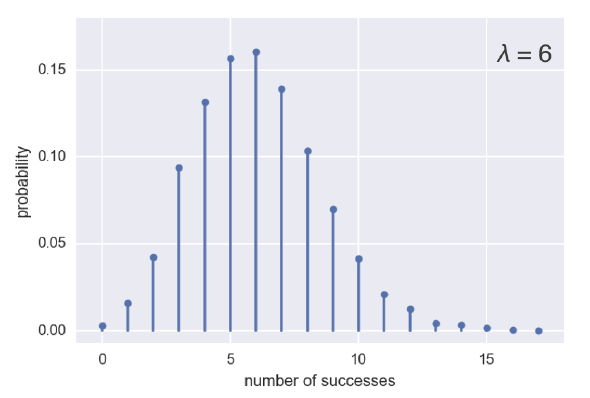
* The experiment counts the number of occurrences of an event over some other measurement.
* The mean is the same for each interval
* The count of events in each interval is independent of the other intervals
* The intervals don’t overlap

### Poisson distribution

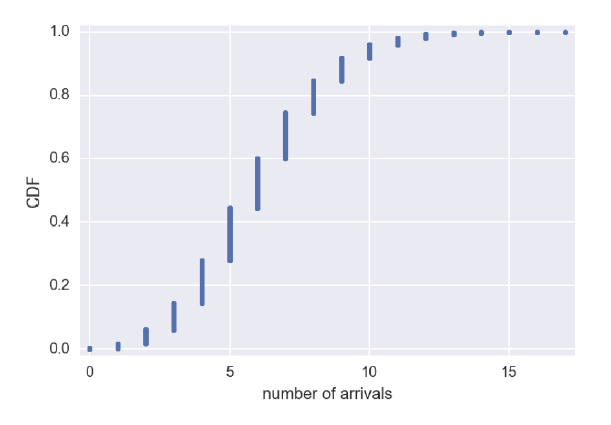
“In his great book on information theory, statistical inference, and machine learning, David MacKay described a town called Poissonville”

The number r of arrivals of a Poisson process in a given time interval with average rate of ? arrivals per interval is Poisson distributed.

#### Poisson PMF



#### Poisson CDF



Poisson distribution is an approximation of the binomial distribution when

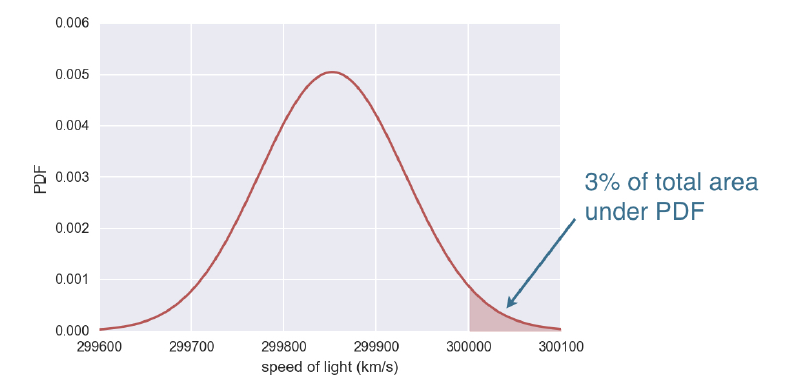
* the number of trials is at least 20
* the probability of success is less than 0.05

i.e., for rare events

## Thinking probabilistically – Continuous variables

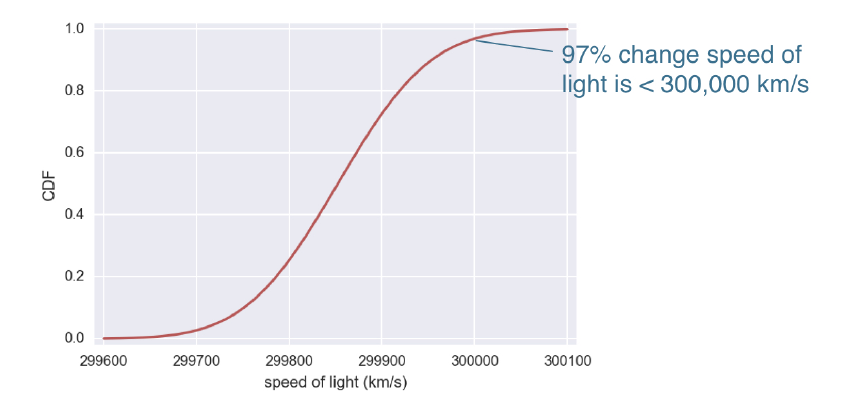
### Probability Density Function (PDF)

Continuous analog to the PMF. It describes the probability of observing a value of a continuous variable.



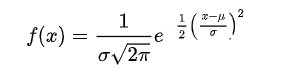
The probability of observing the light whose speed is greater than 300,000 km/s is 3%, the total area under the PDF

On CDF



### The Normal Distribution

The normal (or Gaussian) distribution is a type of continuous probability distribution for a real-valued random variable. The general form of its probability density function is

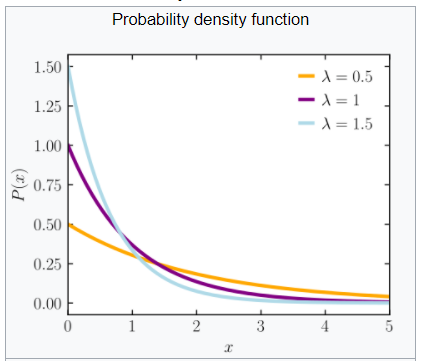


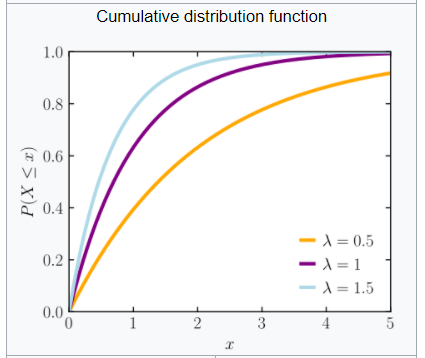
Where mu is the mean and sigma is the standard deviation

### The Exponential Distribution

The exponential distribution is the probability of the time between events in a Poisson point process.

The waiting time between arrivals of a Poison process is exponentially distributed.





# Statistical Thinking in Python Part 2

## Parameter estimation by optimization

Optimal parameters bring the model closest agreement with the data, given that the model we choose is the correct model.

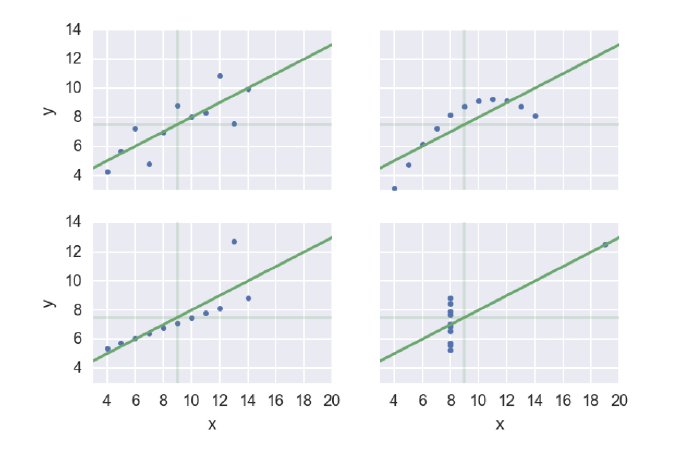
### Linear Regression

Linear regression is the process of using a linear model with point-intercept form to approximate the data.

Least-squares is the process of finding the parameters for which the sum of the squares of the residuals is minimal.

numpy.polyfit(x, y, degree = 1) can be used for linear regression

### Anscombe’s quarter



The four data sets have the same mean value of x, mean value of y, residual and line-slope formulas.

Therefore, perform EDA first.

## Bootstrap confidence intervals

**Resampling** is the process of random selecting observations from the data set (with replacements) as if we are re-conducting the experiment.

**Bootstrapping** is the use of resampled data to perform statistical inference.

A resampled array of data is a **bootstrap sample**.

A statistic computed from a resampled array is called **bootstrap replicate**.

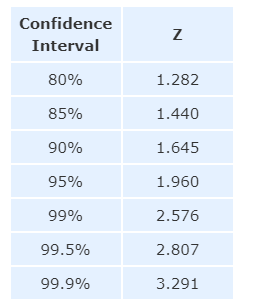
### Confidence interval of a statistic

If we repeat the measurements over and over again, p% of the observed values would lie within the p% confidence level.

**Example in theory**

The average height of men is 176cm, with a standard deviation of 20cm. What is the 95% confidence interval?

We find the z score of 2.5% and 97.5%



Plug in the z-value into the formula



And we have 175 +/- 6.2

Therefore, the interval [168.8, 181.2] includes 95% of the observations.

**Example in simulation**

****

We draw bootstrap replicas from the sample data

Take the percentiles to get the confidence interval.

### Pair bootstrap

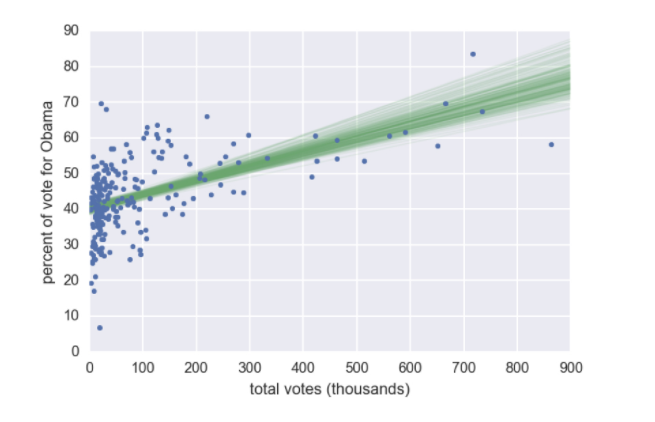
So far, we have been resampling with no assumption about the model or probability distribution of the underlying data, a.k.a, the resampling have been make on the data along.

If we want to perform statistical inference of a linear regression model, we have to consider two parameters, the slope and the intercept.

### Pairs bootstrap for linear regression

* Resample data in pairs (using indices)
* Computer slope and intercept from resampled data
* Each slope and intercept is a bootstrap replicate
* Compute confidence intervals from percentiles of bootstrap replicates.

The results look like this



## Introduction to hypothesis testing

<https://www.statisticshowto.com/probability-and-statistics/hypothesis-testing/>

Example 1:

Percentage of vote to democratic party in counties among Ohio and Pennsylvania are similar.

* Use permutation to randomly reorder two arrays as if they are the same.

### P value

P value is used in the hypothesis testing to help you support or reject the null hypothesis: The p value is the evidence against a null hypothesis. The smaller the p-value, the stronger the evidence that you should reject the null hypothesis.

P-value is the probability of observing a test statistic equally or more extreme than the one you observed, given that the null hypothesis is true.

### The Alpha value

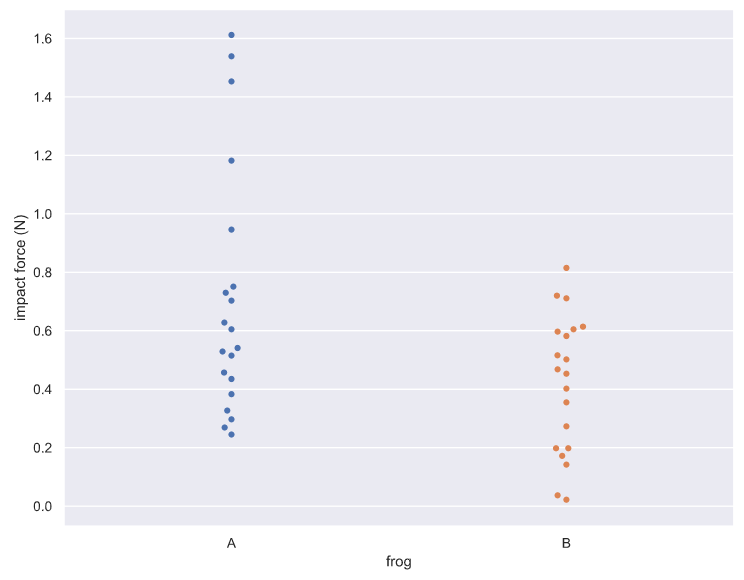
The alpha value is controller by the research and it relates to the confidence level. Depends on if you are running a one-tail test or a two-tail test, a = 1 – 10%, a = 1 – 10%/2

If the p value is extremely small or large and it falls into the area of reject, then we can reject the null hypothesis and approve the alternative hypothesis.

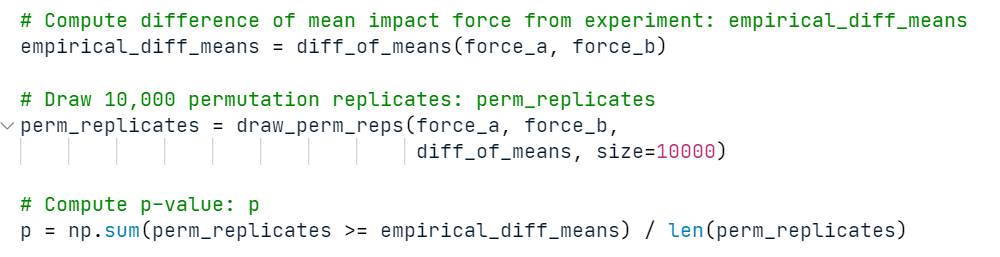
Example 1

Two frogs, one adult and one juvenile hit a wall with their tongue with a force. We are testing the hypothesis that the two frogs the same distribution of impact force.

First, use visual EDA



Then, draw permutations and calculate permutation replicates



Compute the p value, the percentage of permutation replicates that the difference of impact force is greater than the empirical difference

P = 0.0063.

The p value tells us that there is about a 0.6% chance that we would get the difference of means observed in the experiment if frogs were exactly the same.

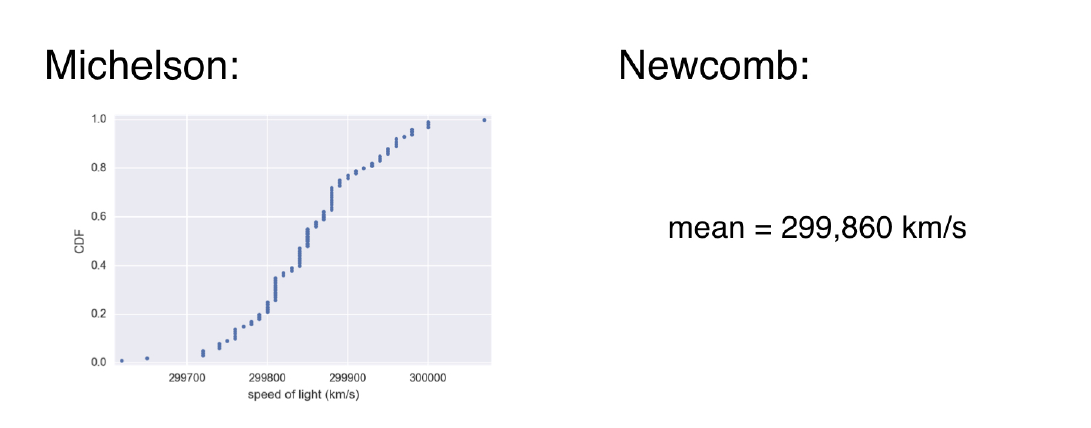
### Pipeline for hypothesis testing

* Clearly state the null hypothesis
* Define your test statistics (mean, property, difference etc / equal to, larger than)
* Generate many sets of simulated data assuming the null hypothesis is true
* Compute the test statistic for each simulated data set
* The p-value is the fraction of your simulated data sets for which the test statistic is at least as extreme as for the real data.

### Michelson and Newcomb

Michelson measure the speed of lights 100 times, Newcomb only have the mean.

Question: Could Michelson got the dataset he had from the experiments, if the true mean of the light is of Newcomb’s



**Null hypothesis**

The true mean speed of light in Michelson’s experiments was actually Newcomb’s reported value.

## Hypothesis test examples

## Putting it all together

# Introduction to Linear Modeling in Python

## Exploring Linear Trends

A linear **model** can be used to descript the distance travelled over time for a vehicle. The model can be used to predict values in between the sample data points (**interpolation**) or values outside the domain / range of the data points (**extrapolation**)

### Visualizing linear relationship

In matplotlib.pyplot use plot to plots scatter plots or a linear function

plt.plot(xs, ys, \*\*options)

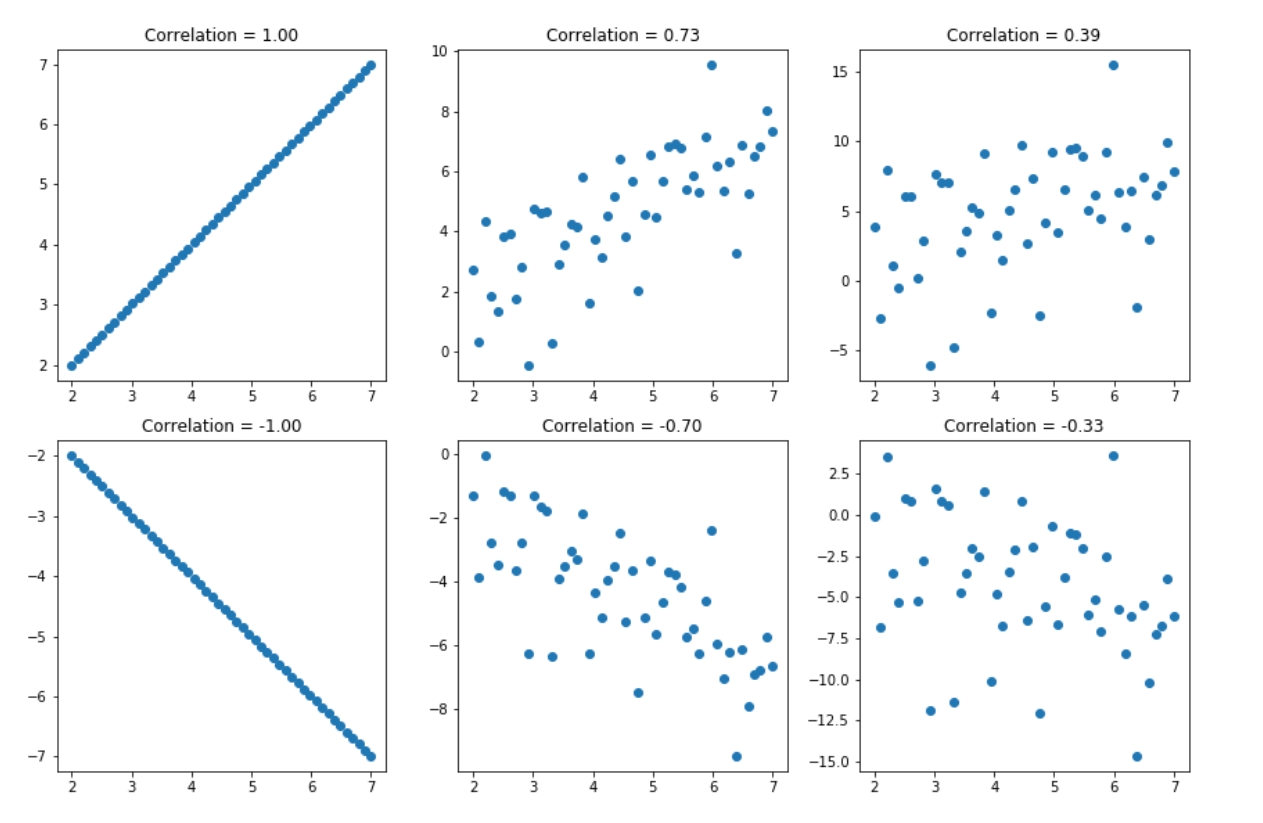
### Quantifying Linear Relationships

Mean, deviation, variance and standard deviation describes the central tendency and the spread the data.

**Covariance** measure how two variables vary together.

**Correlation** is covariance normalized by the standard deviation of x and y. The normalization removes the weight bias of certain variable.

Correlation’s magnitude vs direction



## Build Linear Models

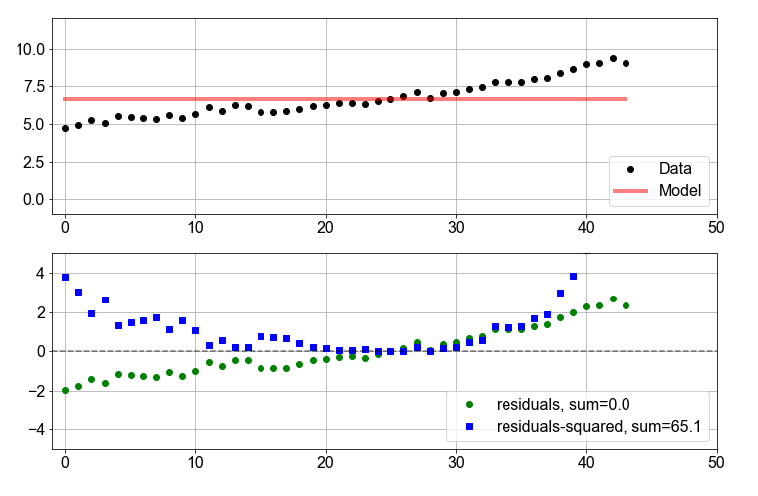
A linear is a Taylor series with two degrees,

y = a0 + a1 \* x

where as a0 is the intercept and a1 is the slope.

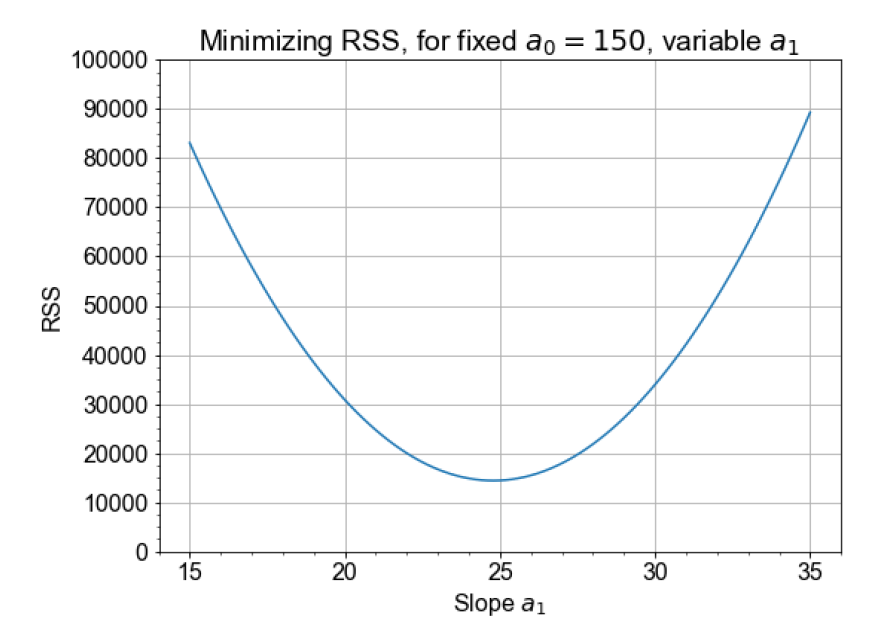
### Residuals

In the graph, the green dots are the residuals. There are both negative and positive values and they could cancel each other out. The blue dots are the squared residuals, they penalize values that are far from the mean.



**RSS**, **R**esidual **S**um of **S**quares is therefore calculated.

For a linear model, we can set a0 to a fixed value and plot possible values of RSS for a1 in a certain range



And we pick the minimum which gives us the least RSS.

### Least-Squares Optimization

Setting RSS slope = 0 and some calculus, yields:

a1 = covariance (x, y) / variance(x)

a0 = mean(y) – a1 \* mean(x)

## Making Model Predictions

### Modeling Real Data

ScikitLearn as a LinearRegression class with yields optimal parameters of a linear model. It taks two numpy arrays.

StatsModel has an OLS (ordinary least square algorithm) for linear model prediction, it accepts a pandas array. It gives error on the estimated parameters.

### The Limits of Prediction

Limits in interpolation: use SPX500 monthly price to build a linear model, but use the model to predict daily price.

Limits in extrapolating. Understand the problem and apply the model only to a reasonable range.

### Goodness-of-Fit

#### Three different R’s

**RSS** is used to help you find the optimal values for model parameters, and thus, the best model. But even the best model will still have non-zero residuals, so how “good” is the best model?

There are two common ways to quantify the goodness-of-fit for a linear model: **RMSE** and **R-squared**.

#### RMSE

RMSE is the root mean squared of the residuals. It tells us how much the model deviates from the data.

#### R-squared

R-squared (or the coefficient of determination) is a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variable.

measures how much variation in the data is due to the linear trend.

### Standard Error

You have computed quantitative measures of variation and "goodness" of the model \*predictions\*, but what about the variation or errors in the model \*parameters\*? How accurate are the model PARAMETERS, are there variations in those parameters, and how much of the variation is due to deterministic trends versus inherent randomness? In this lesson, instead of using a single value like RMSE that summarizes the entire model prediction, we will compute the standard error of each of the model parameters separately.

## Estimating Model Parameters

### Inferential Statistics Concepts

Previously, we found the single best value of each model parameter and used them to build a model. In this Chapter, we’ll treat a model parameter, like slope, not a single value but as a “distribution” of values, whose mean gives our “best” value.

### Model Estimation and Likelihood

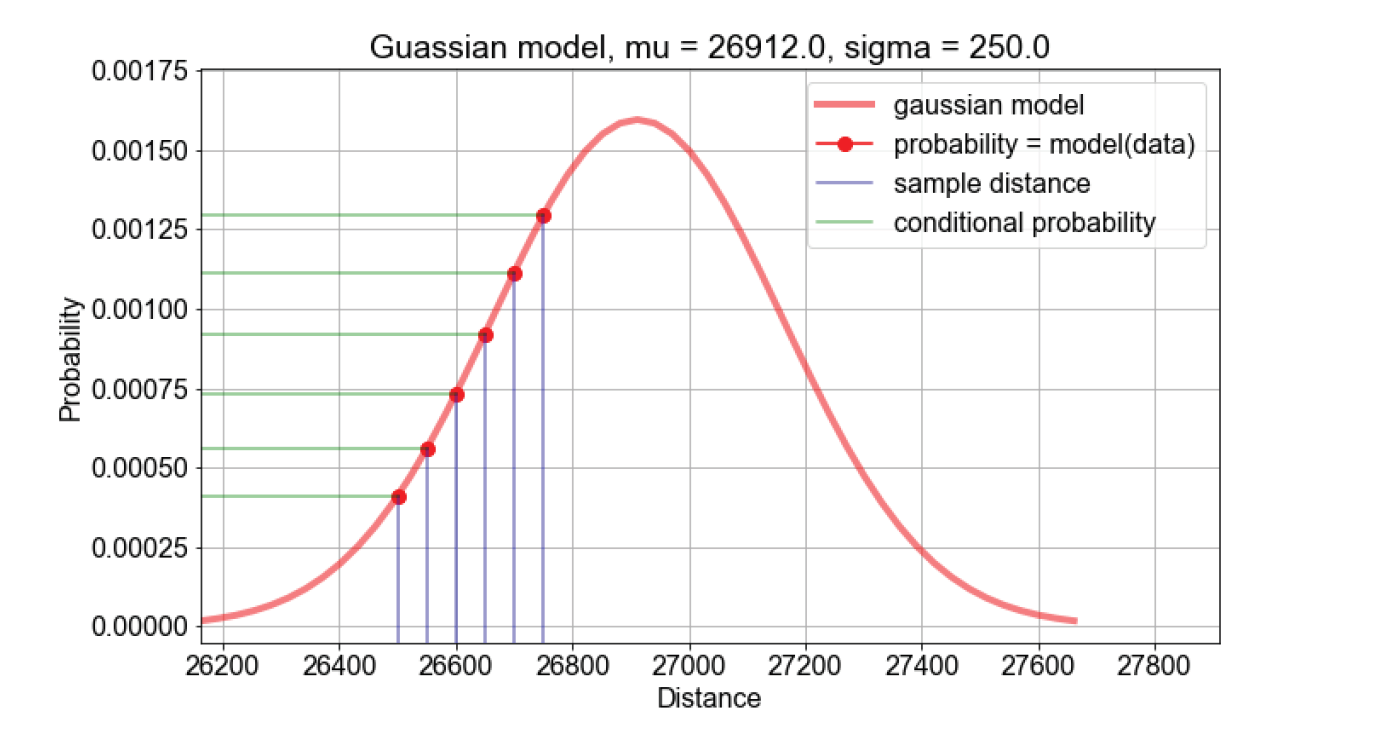
We use estimation to build models of population distribution from sample statistics.

A condition probability is stated as a question: what is the probability that A occurs, “given the condition” that B has already occurred. In the context of data and models, there is a naming convention for conditionals.

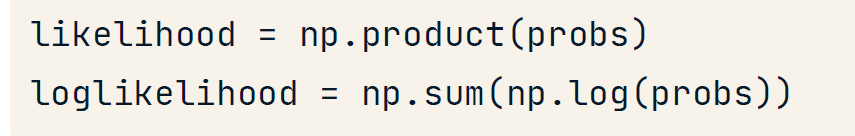
If the **model is given**, what is the **probability** that it output any particular data point.

If the **data is given**, what is the **likelihood** that a candidate model could output the particular data we have.

If we have two candidate models, we would like to choose the one that has a greater likelihood to output a given data.



For a certain value of mu and sigma, calculate the probability of all the sample data points. Take the products of all the probabilities, and take the log to compute the loglikelihood.



Compute log likelihoods for a range of estimations, find the best guess.

### Model Uncertainty and Sample Distributions

Previously, to estimate a model parameter, we assumed a shaped of the parameter distribution. Least-squares assumes a gaussian; maximum likelihood estimation requires us to chose a shape, so we chose gaussian.

But there are situations where the distribution shape is unknown.

Recall how the shape of the sampled temperature data resembled the population shape? What if we used the sample as the model of the population? If we compute the mean of the single sample, it gives us a guess, but no knowledge of the uncertainty in this guess. If we resample the samples, it gives a distribution sample statistic. We can use this to make predictions.

### Model Errors and Randomness

We’ve seen linear model parameters as distributions, spread about some central peak. Now we’ll relate the parameter distribution to the “standard error” of linear model parameters, and check whether our parameter estimates are effected by randomness.

#### Type of Errors

* Measurement error
* Sampling Bias
* Randomness

#### Null Hypothesis

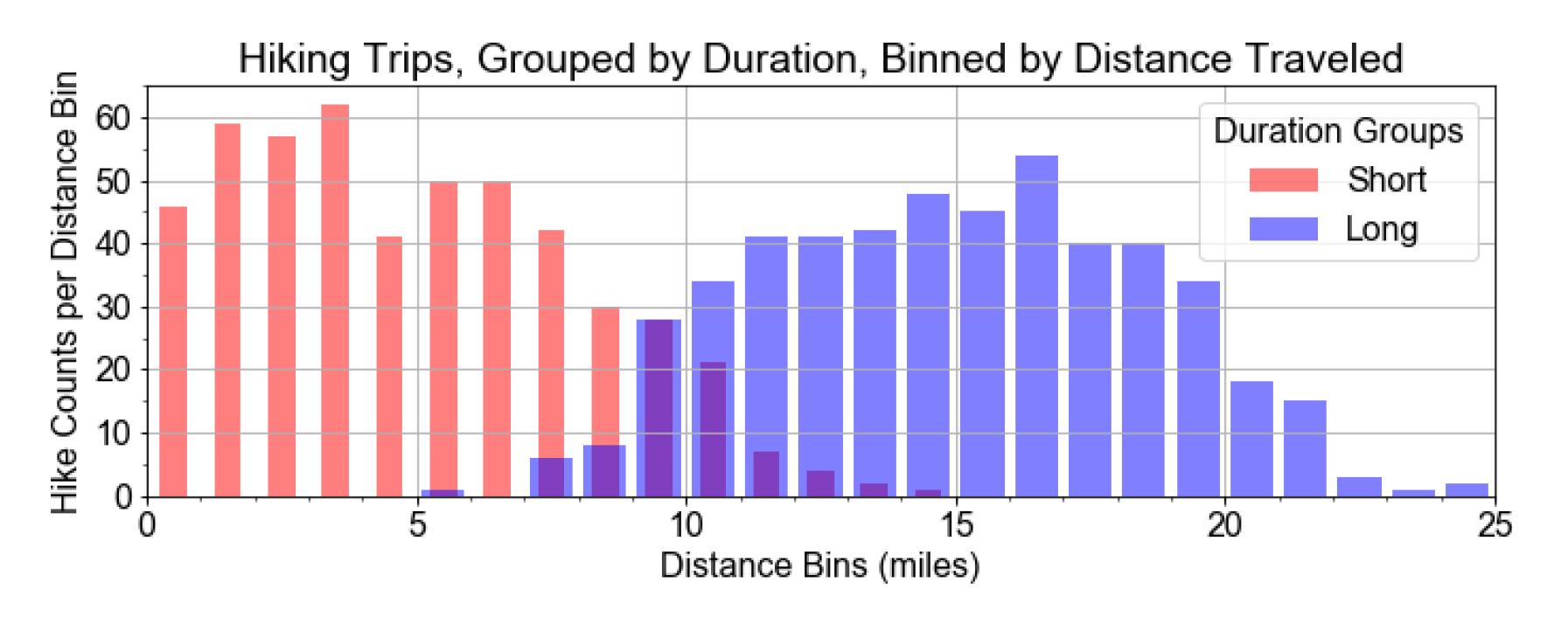
We restate the question:

“Is our effect due to a relationship or due to random chance?”

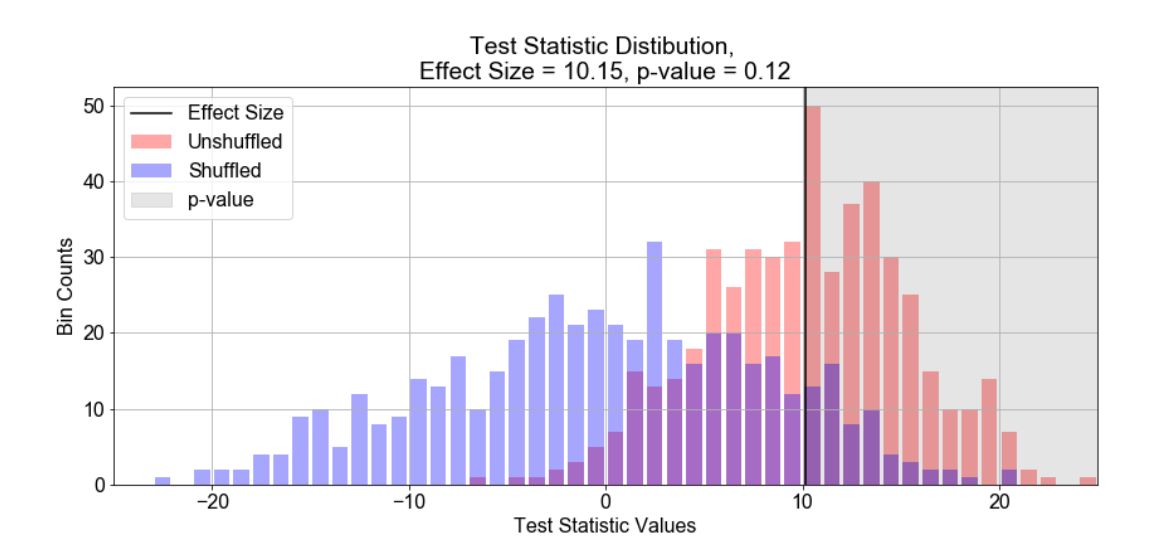
“Does the ordering or grouping of the data cause an effect larger than what could be produced by randomly shuffled data?”

#### Testing null hypothesis

We break the samples into two groups, short and long distance. The difference between those groups, are 10. We call it the **effect size** = 10



We shuffled the data and see the percentage of observations in the shuffled data that exceed the effect size. We call the percentage the p-value.



From the graph, we can see that it is 12 percent chance to get a speed of 10 or more just from random chance. If it is 0.01% chance to get a speed of 10 or more just from random chance, then we know that it is quite certain that the average speed is 10. Here, 12% is kind of in the middle.

# Statistical Simulation in Python

## Basics of randomness & simulation

Simulation Steps

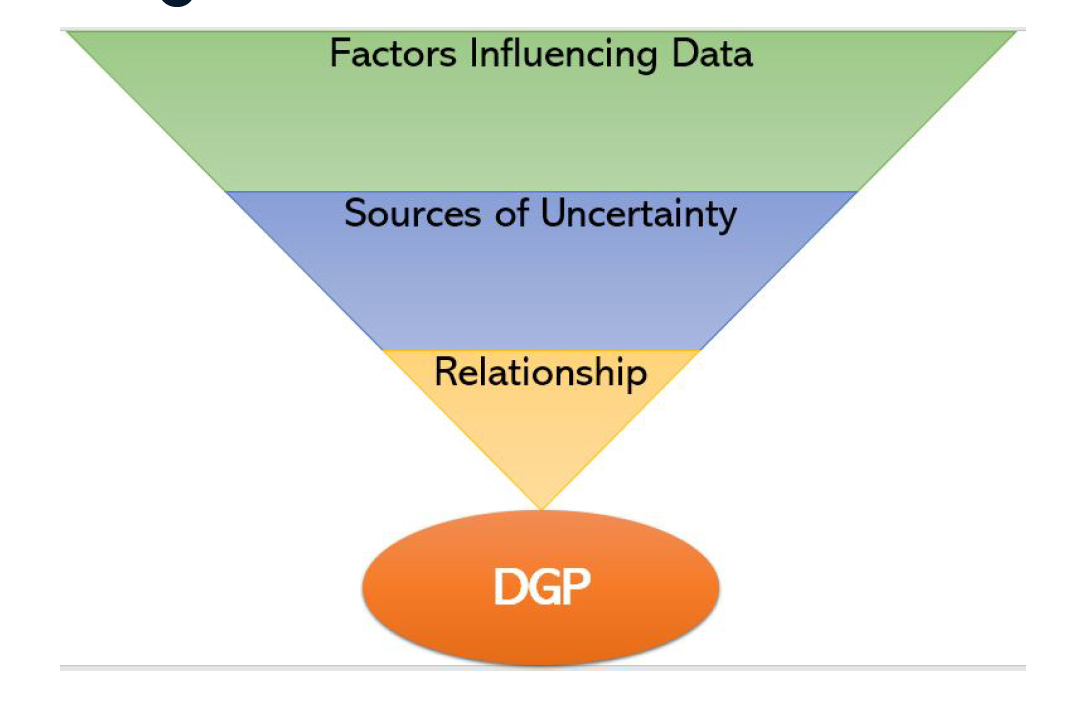
1. Define possible outcomes for random variables
2. Assign probabilities
3. Define relationships between random variables
4. Get multiple outcomes by repeated random sampling
5. Analyze sample outcomes

## Probability & data generation process

### Steps for Estimating Probability:

1. Construct sample space or population
2. Determine how to simulate one outcome
3. Determine rule for success
4. Sample repeatedly and count successes.
5. Calculate frequency of successes as an estimate of probability.

### Data Generating Process



e.g.

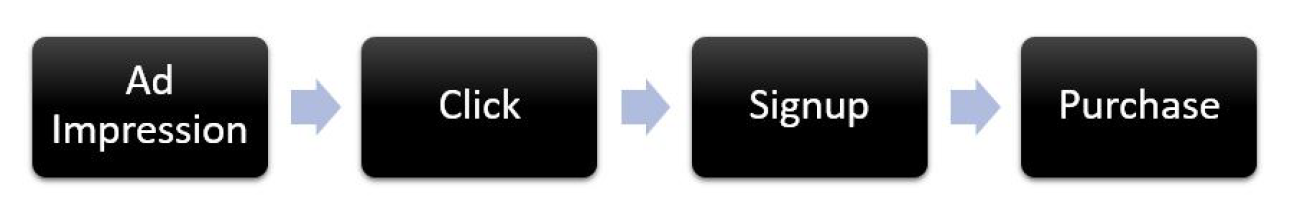
Factors influencing data: How many steps you take on a day

Source of uncertainty: Mood and motivation to go to the gym, 40% of the time

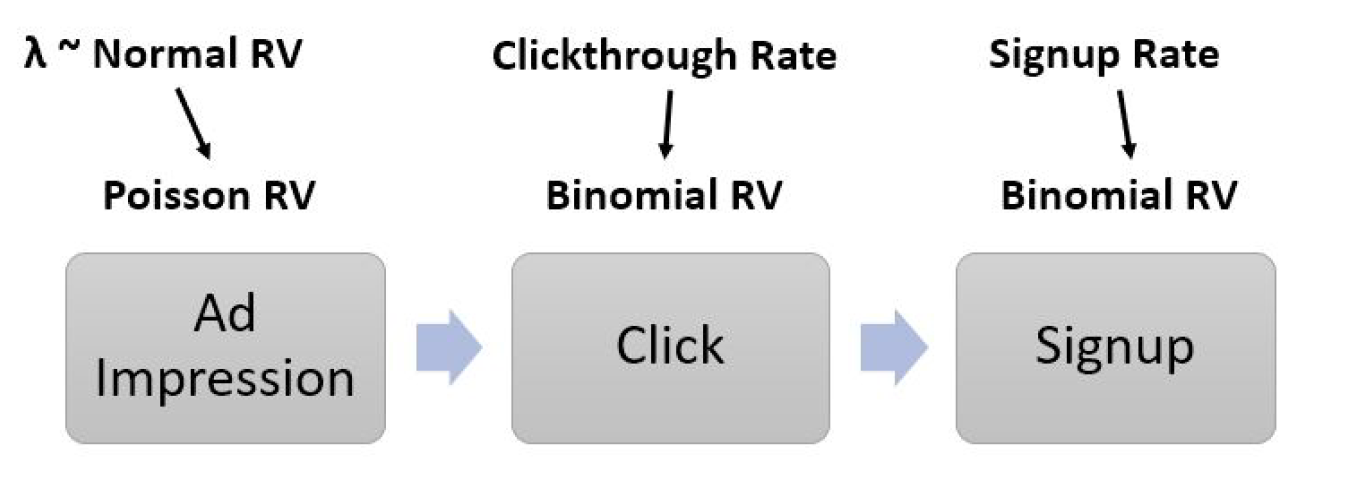
Relationship: If steps > 10000, 80% chance lose one pound, otherwise 20% chance gain one pound.

### eCommerce Ad Simulation

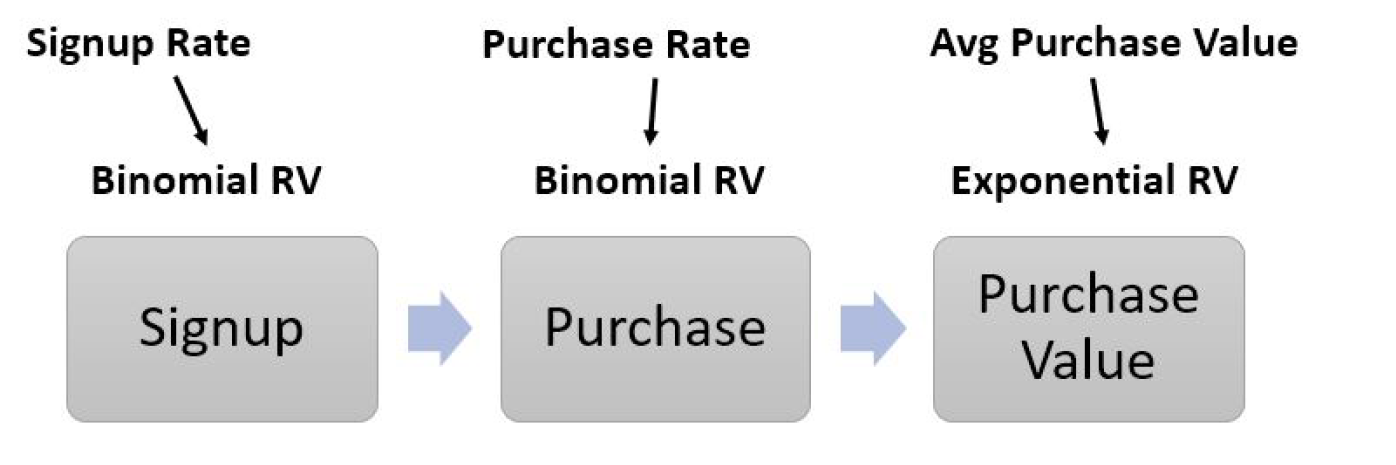
A general eCommerce flow



We model the first three steps using Poisson random variable and binomial random variables.



And we model the purchase flows using binomial and exponential random variables



## Resampling methods

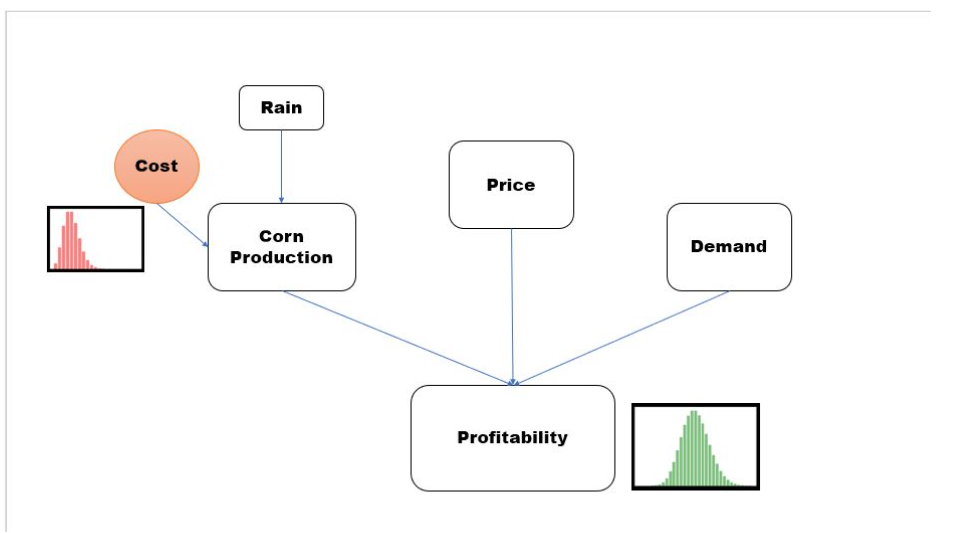
Different resampling methods

* Bootstrap resampling
* Jackknife resampling
* Permutation

## Advanced Applications of Simulation

### Simulation for Business Planning

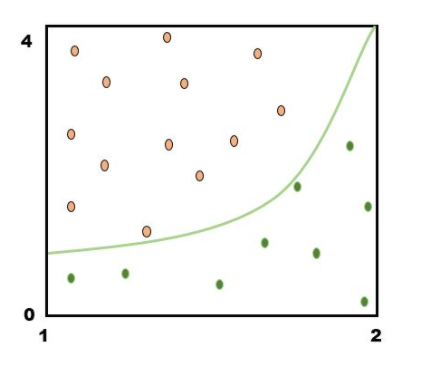
Simulation for Corn production



### Monte Carlo Integration

Steps

* Calculate overall area
* Randomly sample points in the area
* Multiply the fraction of the points below the curve by overall area.



### Simulation for Power Analysis

### Applications in Finance

# Supervised Learning with scikit-learn

## What is Machine Learning

### Supervised learning

Uses labeled data. Train the model with labeled dataset. Then predict whether an email is a spam or not.

### Unsupervised learning:

Unsupervised learning uses unlabeled data. It uncovers patterns from unlabeled data. For example, grouping customers into distinct categories (**clustering**, without having any presumptions of the groups)

### Reinforcement learning

Software agents interacts with an environment.

* Learn how to optimize their behavior
* Given a system of rewards and punishments.

## Classification

In supervised learning, datasets can be divided into features and target variables

The aim is to predict the target variable, given features.

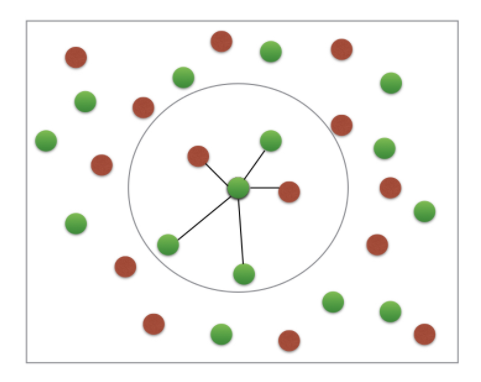
* Classification: target variable consists of categories
* Regression: target variable is continuous

### k-Nearest Neighbors

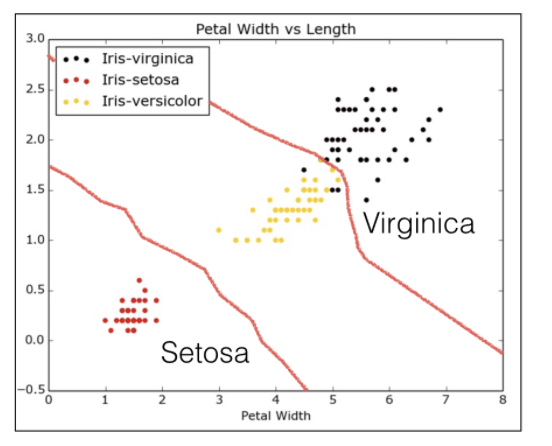
Basic idea: predict the label of a data point by

* Looking at the k closest labeled data points
* Taking a majority vote

K-Nearest neighbors in 2-dimensions, if K = 5, the point in the center is classified as green because 3 out of 5 nodes are green.

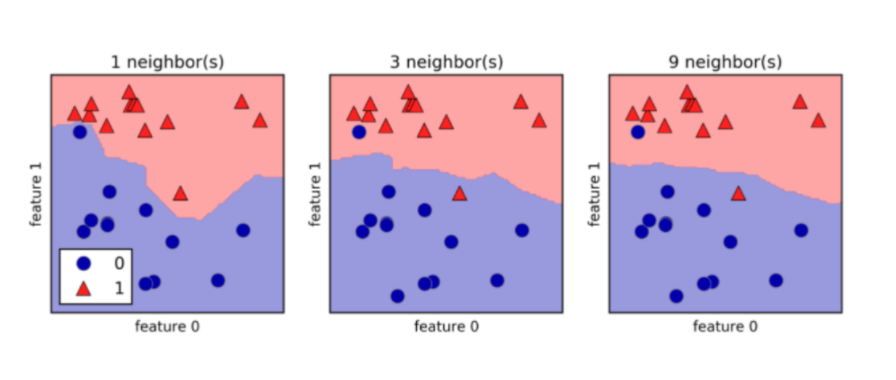


When fitting a KNN model, decision boundaries are created

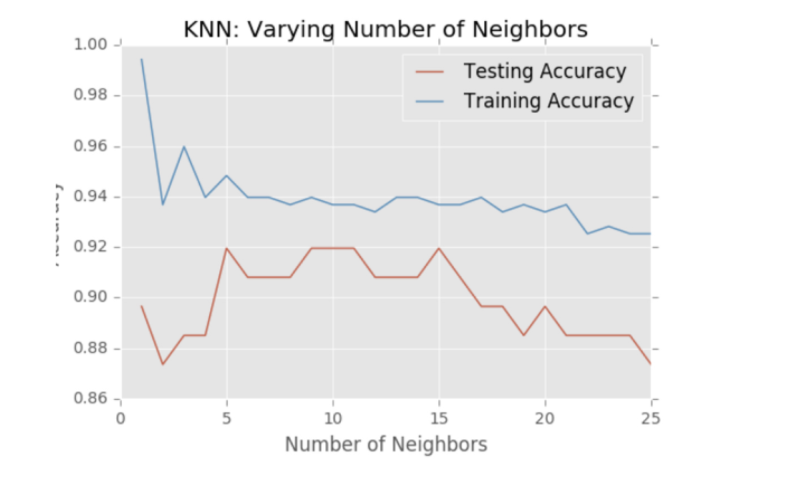


### Measuring performance

For categorization, we measure the performance in the percentage of correctly predicted test data. We split the dataset into training dataset and test dataset.



When k increases, the decision boundary get smoothers and less curvy. It is considered as a less complex model. Complex models run the risk of being sensitive to noise in the specific data you have, rather than reflecting general trends in the data. This is known as **overfitting**.



If we increase K even more, it performs less well in both training and test data set. This is called **underfitting**.

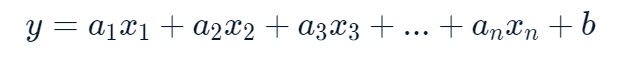
## Regression

### Regression mechanics

* y = a \* x + b
* y is the target
* x is a single feature
* a and b are parameters of the model
* How do we choose a and b?
* Define an error function and minimize the error

Ordinary least squares (**OLS**) minimize the sum of the squares of the residuals.

### Linear regression in higher dimensions

  
Cross validation splits data into n folds, runs regression n times, each time, split fold n as test data and the rest of the fold as training data.

### Ridge regression

Loss function = OLS loss function +

It penalizes models with large coefficients (both large positive and large negative)

Alpha is the parameter we need to choose. It is a hyper parameter, similar to picking k in kNN

Alpha controls model complexity

* Alpha = 0, we get back to OLS, (can lead to overfitting)
* Alpha is high, can lead to underfitting

### Lasso regression

Loss function = OLS loss function +

IT can be used to select important features of a dataset.

Shrinks the coefficient of less important features to exactly 0,

Features that are not shrunk to zero are selected.

## Fine-tuning your model

### How good is your model?

Confusion matrix

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Predicted | |
| Spam | Real |
| Actual | Spam | True Positive | False Negative |
| Real | False Positive | True Negative |

Accuracy = (tp + tn) / (tp + tn + fp + fn)

**Precision** = tp / (tp + fp)

Precision is also called positive predictive value or PPV. High precision means that our classifier had a low false positive rate, that is, not many real emails were predicted as being spam.

**Recall** = tp / (tp + fn)

Recall is also called sensitivity, hit rate or true positive rate. High recall means that our classifier predicted most positive or spam email correctly.

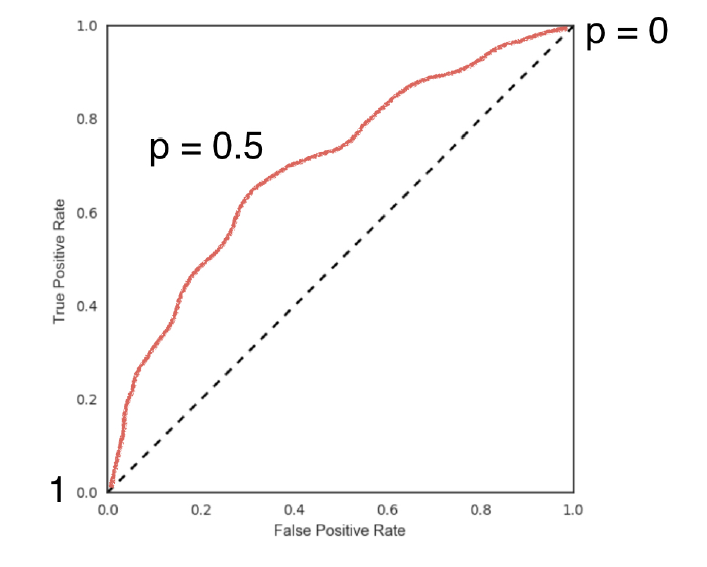
F1-score = 2 \* (precision \* recall ) / (precision + recall)

### Logistic Regression and the ROC curve

Logistic regression is a classification algorithm. Given one feature variable, it will output a probability p,

If P> 0.5, label as 1, If P < 0.5, label as 0.

The probability p is a threshold set to 0.5 by default, we when set p to all possible values, we can plot out the receiver characteristics curve (**ROC**) curve.



#### Area under the ROC curve

The large the area under the ROC curve, the better the model is.

### Hyperparameter Tuning

## Preprocessing and pipelines

# Unsupervised Learning in Python

# Linear Classifiers in Python

# Case Study: School Budgeting with Machine Learning in Python

# Introduction to Deep Learning in Python