Data 115: Week 7 Scipy Stats Examples

Daryl DeFord

October 26, 2020

This document provides some description of the four scipy.stats functions that we used to analyze normal distributions, as well as the corresponding versions from R that you may have seen on the slides.

- **Preliminaries:** To make use of these functions, we need to import two packages:
  import numpy as np from scipy.stats import norm
- **Random Seed:** If we want to set a random seed for our session, we can use:
  np.random.seed(seed='your favorite number goes here').
- **Binomial Distribution:** To draw k samples from a binomial distribution with a fixed number of trials n and a fixed success probability p, we can use:
  from scipy.stats import binom
  binom.rsv(n,p,k)

  For example, entering binom.rsv(20,.3,50) will return an array with 50 integers, where each sample reports the number of successes out of 20 trials, where each trial has a 30% chance of succeeding. This is equivalent to flipping a weighted coin 20 times, recording the number of heads, and then repeating the process 49 more times, recording the number of heads that occurred in each collection of 20 flips.
- **Normal Distributions:** To make a normal distribution with fixed mean and standard deviation we use the norm function we imported: my_normal = norm(loc=10, scale = 4). Once we have the distribution created, we can access its various parameters using the . notation:
  - **rvs** To generate sample draws from the distribution we can use: my_normal.rvs(100)
  - **pdf** To get the pdf value that corresponds to a particular x value: my_normal.pdf(1.2)
  - **cdf** To get the probability that a draw is less than a given x value we use the cdf: my_normal.cdf(2.4)
  - **ppf** To find the x value that corresponds to a particular percentile of the distribution we use ppf: my_normal.ppf(.99)
- **Plotting:**
  - **pdf** We can use the .pdf function to make line plots of the pdf to compare to histograms. In the lecture, we used something like:
    plt.plot(np.linspace(my_normal.ppf(.01),my_normal.ppf(.99),100),
    my_normal.pdf(np.linspace(my_normal.ppf(.01),my_normal.ppf(.99),100)), color='lime', lw = 4)
  - **QQplot** To make QQ plots in python we can import the probplot function from scipy.stats and then pass it the column of data we want to analyze:

    from scipy.stats import probplot import matplotli.pyplot as plt

    probplot(df['height'],plot=plt)
- **Comparison with R:** These commands all have analogues in the R language that were shown briefly on the slides:

R function	Python Function	Description
dnorm()	norm.pdf()	Probability density function
pnorm()	norm.cdf()	Cumulative density function
qnorm()	norm.ppf()	Percentile point function
rnorm()	norm.rvs()	Random variates (samples)