Parameters

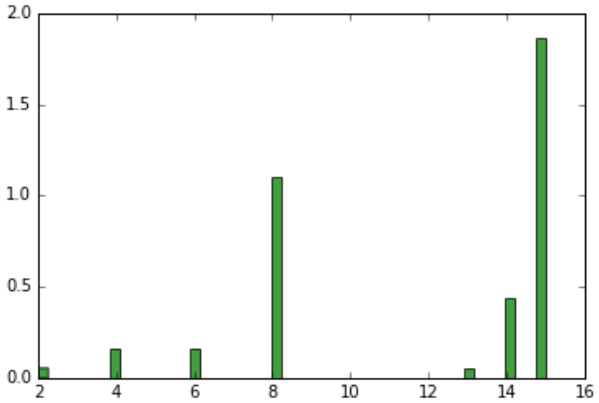
Introduction to Data Science Algorithms

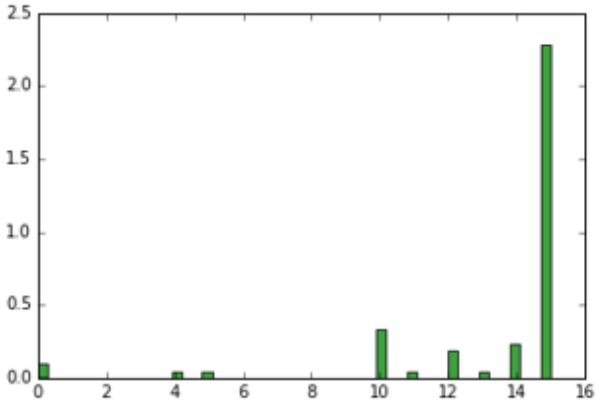
Jordan Boyd-Graber and Michael Paul

SEPTEMBER 27, 2016

Administrivia

- HW1 Graded, posted soon
- Python help session next Thursday
- Don’t leave HW2 until last minute





True / False Test

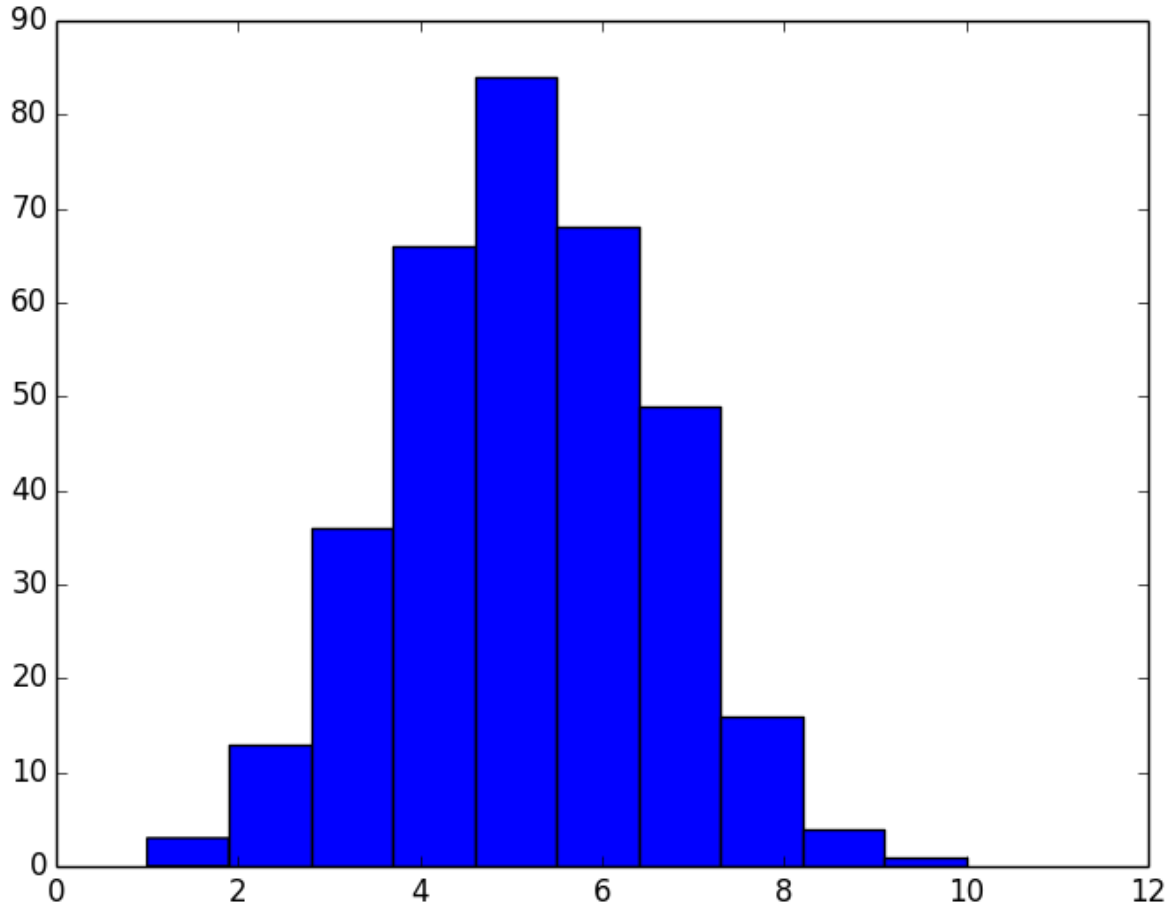
Baumgartner, Prosser, and Crowell are grading a calculus exam. There is a true-false question with ten parts. Baumgartner notices that one student has only two out of the ten correct and remarks, “The student was not even bright enough to have flipped a coin to determine his answers.” “Not so clear,” says Prosser. “With 340 students I bet that if they all flipped coins to determine their answers there would be at least one exam with two or fewer answers correct.” Crowell says, “I’m with Prosser. In fact, I bet that we should expect at least one exam in which no answer is correct if everyone is just guessing.” Who is right in all of this?

True / False Test

Baumgartner, Prosser, and Crowell are grading a calculus exam. There is a true-false question with ten parts. Baumgartner notices that one student has only two out of the ten correct and remarks, “The student was not even bright enough to have flipped a coin to determine his answers.” “Not so clear,” says Prosser. “With 340 students I bet that if they all flipped coins to determine their answers there would be at least one exam with two or fewer answers correct.” Crowell says, “I’m with Prosser. In fact, I bet that we should expect at least one exam in which no answer is correct if everyone is just guessing.” Who is right in all of this?

Hint: Simulate a draw from the appropriate binomial distribution
`numpy.random.binomial`

One Draw



How to do a Draw

```
>>> from numpy.random import binomial
>>> res = binomial(10, .5, 340)
>>> from collections import Counter
>>> Counter(res)
Counter({5: 83, 4: 75, 6: 74, 7: 48, 3: 31, 8: 14, 2: 11, 1
```

Many Experiments

```
>>> res = binomial(10, .5, (1000, 340))
>>> two_or_less = sum(1 for experiment in res if
        any(y<=2 for y in experiment))
>>> two_or_less
1000
>>> zeros = sum(1 for experiment in res if
        any(y==0 for y in experiment))
>>> zeros
302
>>> import matplotlib.pyplot as plt
>>> plt.hist(res[0])
(array([  1.,   4.,  16.,  50.,  73.,  71.,  67.,  39.,  16
>>> plt.show()
```

Peak of Poisson Distribution

Show that a most probable outcome of a Poisson with parameter λ is the integer m s.t. $\lambda - 1 \leq m \leq \lambda$. When will there be two most probable values?

Peak of Poisson Distribution

Show that a most probable outcome of a Poisson with parameter λ is the integer m s.t. $\lambda - 1 \leq m \leq \lambda$. When will there be two most probable values?

- Consider the mass function $\frac{\lambda^x e^{-\lambda}}{x!}$
- Compare x and $x+1$ as a ratio of probabilites

Ratio of probabilities

$$\frac{p(X=t+1)}{p(X=t)} = \frac{\lambda^{t+1}e^{-\lambda}}{(t+1)!}\frac{t!}{\lambda^{t}e^{-\lambda}} \tag{1}$$

$$= \frac{\lambda}{t+1} \tag{2}$$

Ratio of probabilities

$$\frac{p(X=t+1)}{p(X=t)} = \frac{\lambda^{t+1}e^{-\lambda}}{(t+1)!}\frac{t!}{\lambda^{t}e^{-\lambda}} \tag{1}$$

$$= \frac{\lambda}{t+1} \tag{2}$$

Increasing so long as $t < \lambda - 1$. If $t = \lambda - 1$, then constant (i.e., two most probable values). After that, decreasing.

Ratio of probabilities

$$\frac{p(X=t+1)}{p(X=t)} = \frac{\lambda^{t+1}e^{-\lambda}}{(t+1)!}\frac{t!}{\lambda^{t}e^{-\lambda}} \tag{1}$$

$$= \frac{\lambda}{t+1} \tag{2}$$

Increasing so long as $t < \lambda - 1$. If $t = \lambda - 1$, then constant (i.e., two most probable values). After that, decreasing.

$$p(X=t) \begin{cases} > p(X=t+1) & \text{if } \lambda > t+1 \\ = p(X=t+1) & \text{if } \lambda = t+1 \\ < p(X=t+1) & \text{if } \lambda < t+1 \end{cases} \tag{3}$$

Poisson Plot Code

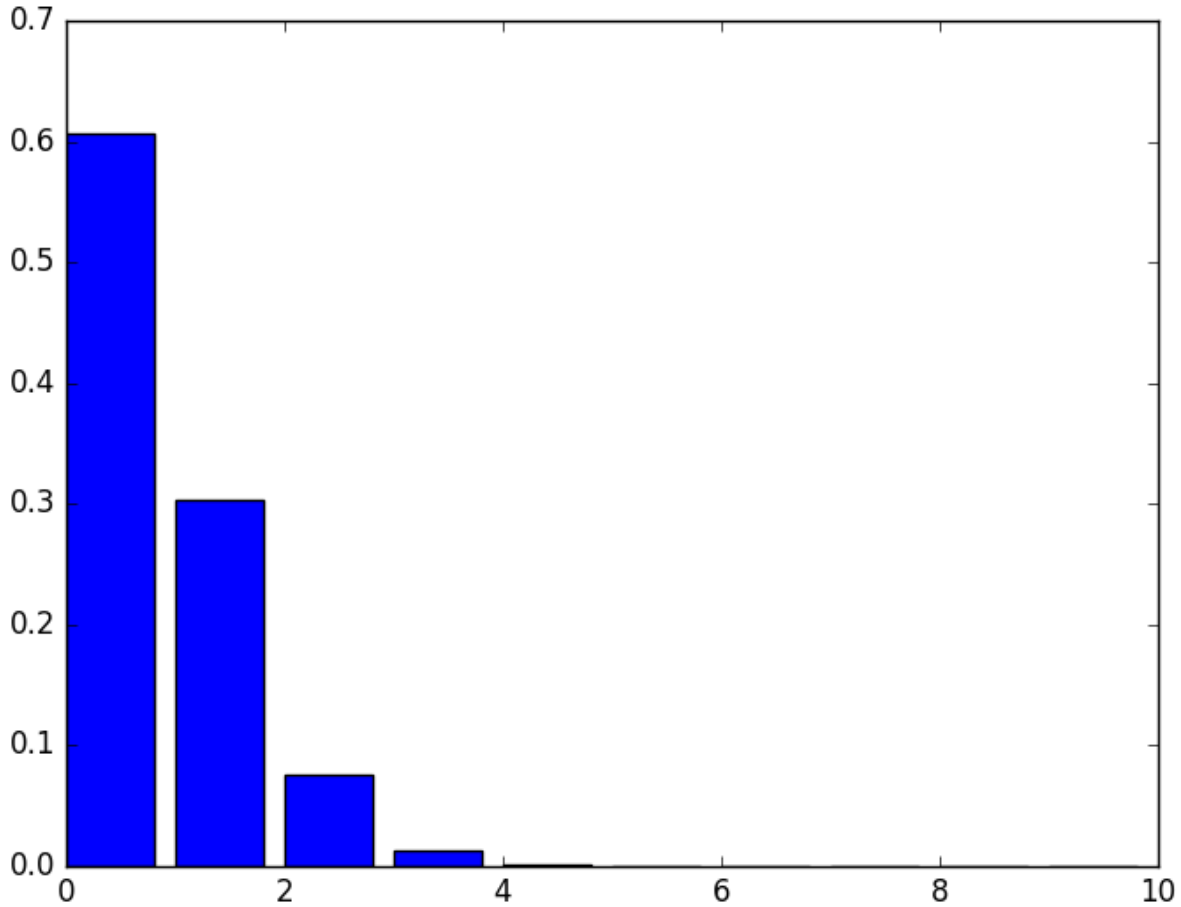
```
def poisson_density(mean):
    return lambda x: mean**x * exp(-mean) / factorial(x)

if __name__ == "__main__":
    mm = float(sys.argv[1])
    x_pos = range(10)
    y_pos = [poisson_density(mm)(x) for x in x_pos]
    plt.bar(x_pos, y_pos)
    plt.savefig("poisson-" + str(mm) + ".png")
```

Poisson Plots

$$\lambda = 0.5$$

Poisson Plots

$$\lambda = 0.5$$

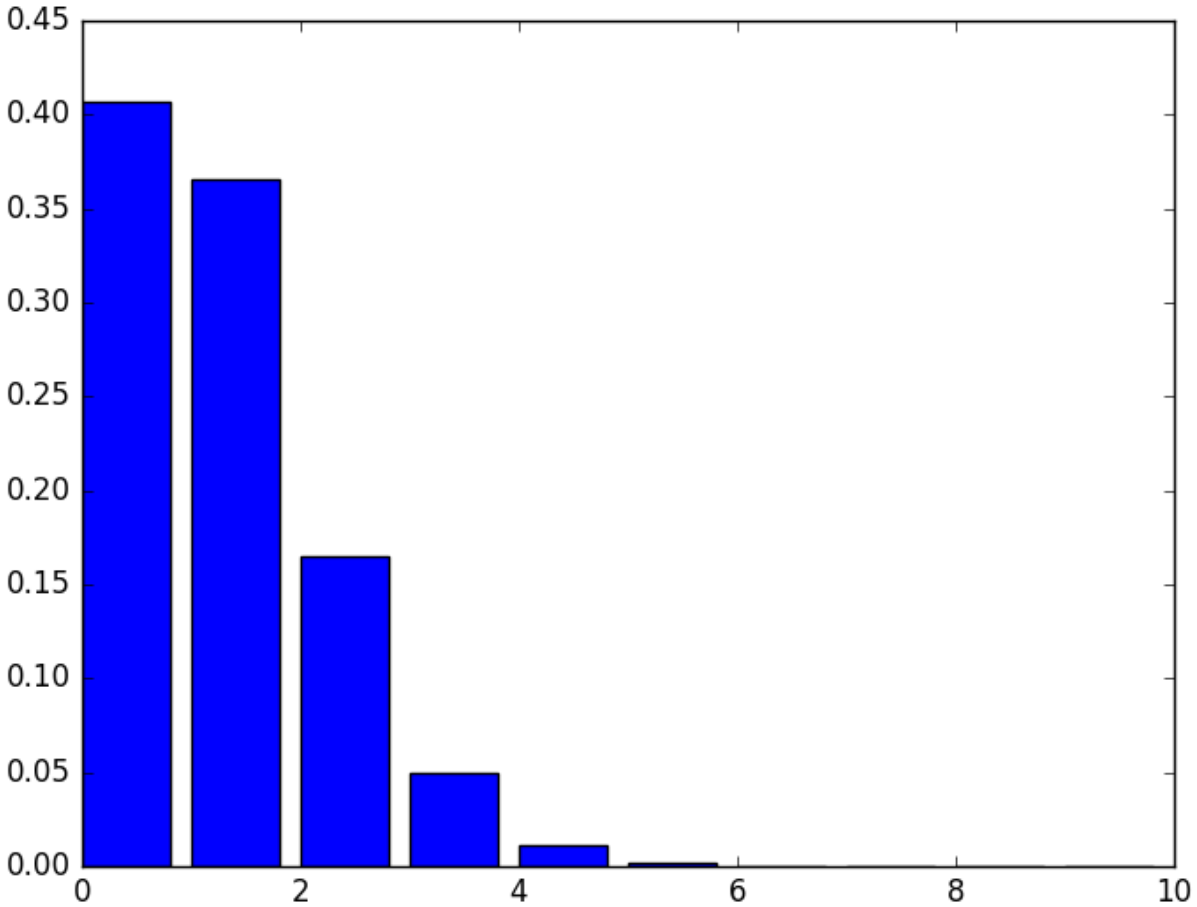


Poisson Plots

$$\lambda = 0.9$$

Poisson Plots

$$\lambda = 0.9$$

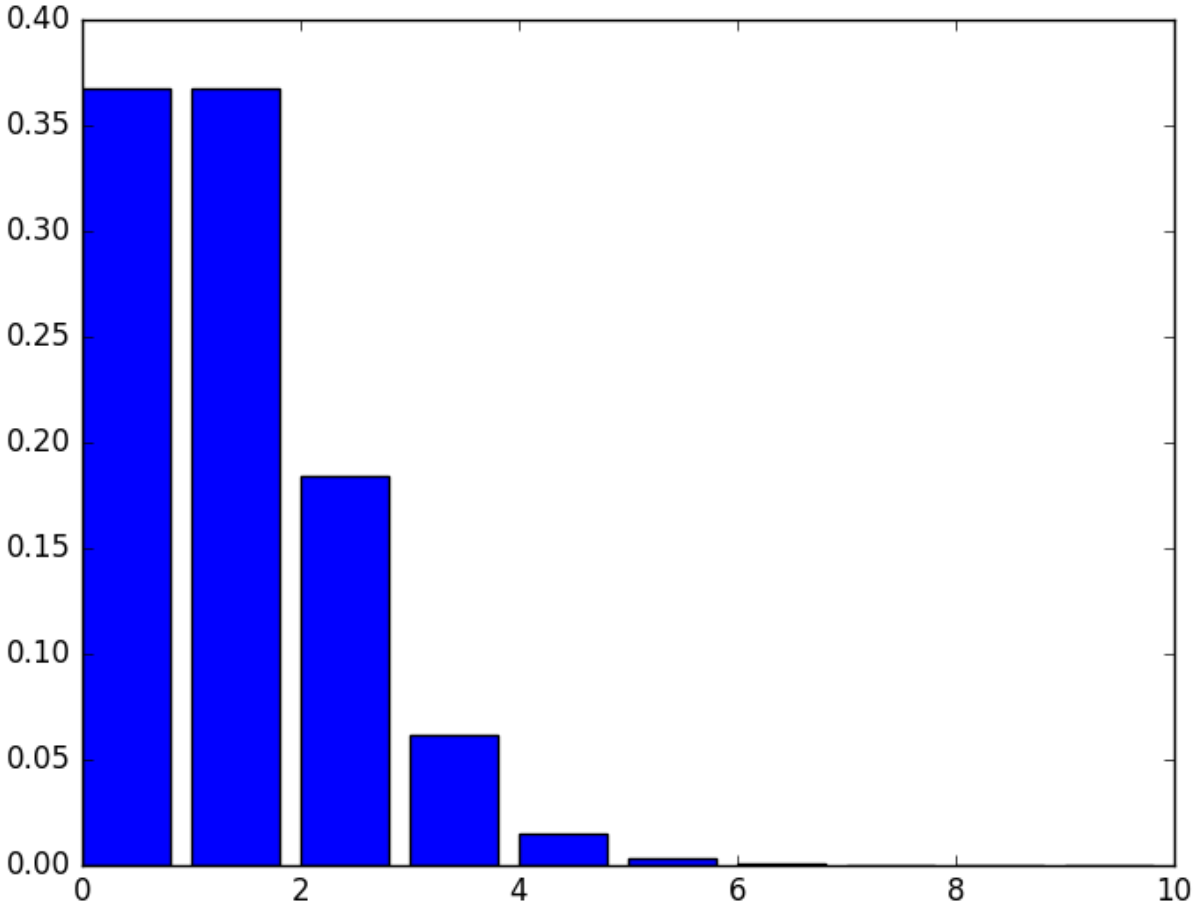


Poisson Plots

$$\lambda = 1.0$$

Poisson Plots

$$\lambda = 1.0$$

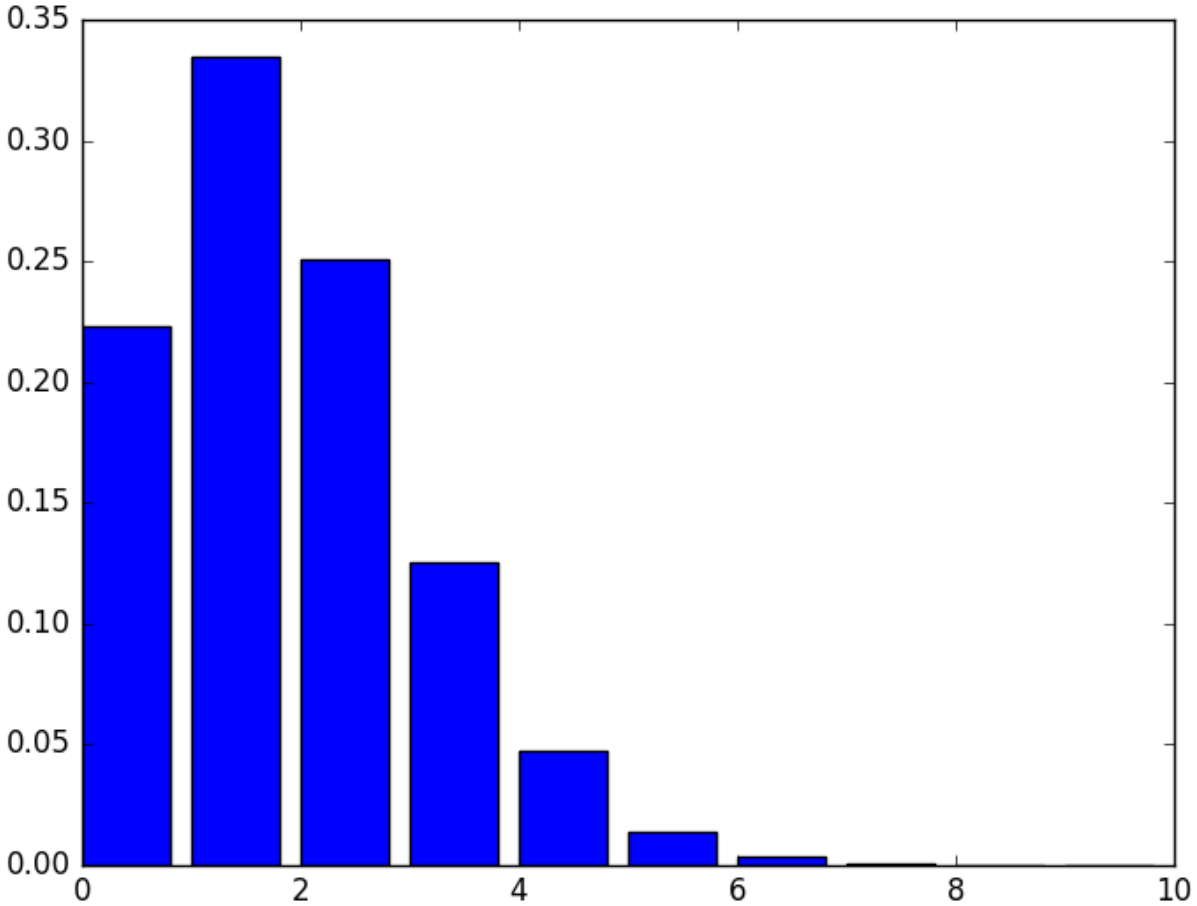


Poisson Plots

$$\lambda = 1.5$$

Poisson Plots

$$\lambda = 1.5$$

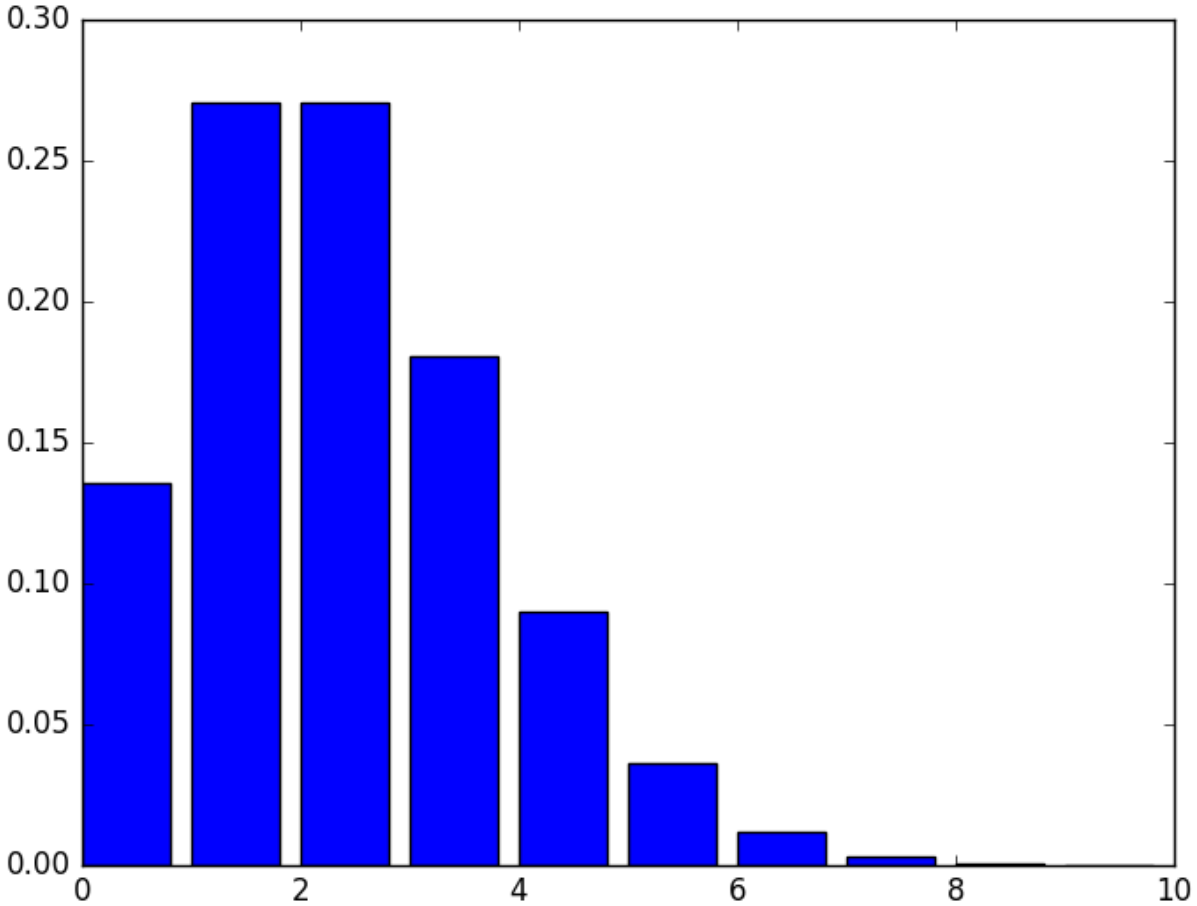


Poisson Plots

$$\lambda = 2.0$$

Poisson Plots

$$\lambda = 2.0$$



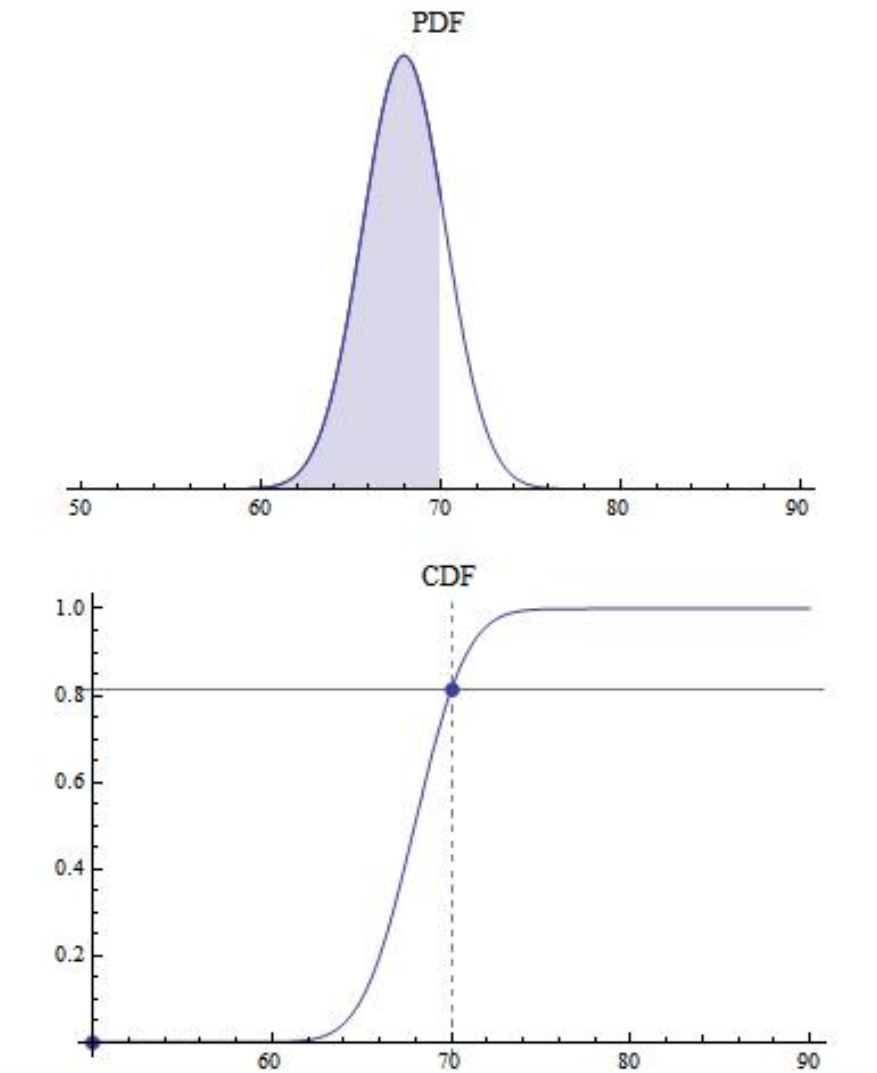
Distribution of Bearings

Bridies' Bearing Works manufactures bearing shafts whose diameters are normally distributed with parameters $\mu = 1$, $= .002$. The buyer's specifications require these diameters to be $1.000 \pm .003$ cm. What fraction of the manufacturer's shafts are likely to be rejected? If the manufacturer improves her quality control, she can reduce the value of σ. What value of σ will ensure that no more than 1 percent of her shafts are likely to be rejected?

Distribution of Bearings

Bridies' Bearing Works manufactures bearing shafts whose diameters are normally distributed with parameters $\mu = 1$, $= .002$. The buyer's specifications require these diameters to be $1.000 \pm .003$ cm. What fraction of the manufacturer's shafts are likely to be rejected? If the manufacturer improves her quality control, she can reduce the value of σ. What value of σ will ensure that no more than 1 percent of her shafts are likely to be rejected?
Hint: `scipy.stats.norm` has `cdf`, it uses standard deviation as the scale.

Cummulative Distribution Function



What is rejected?

```
>>> from scipy.stats import norm
>>> rej = norm.cdf(.997, loc=1, scale=0.002)
>>> rej += 1 - norm.cdf(1.003, loc=1, scale=0.002)
>>> rej
0.13361440253772297
```

What does σ have to be?

```
>>> def rej(sd):
...     val = norm.cdf(.997, loc=1, scale=sd)
...     return val + 1 - norm.cdf(1.003, loc=1, scale=sd)
>>> for x in range(100):
...     val = 0.001 + x/100000.
...     print("%0.04f\t%f" % (val, rej(val)))
0.00115         0.009089
0.00116         0.009704
0.00117         0.010344
0.00118         0.011010
```

Waiting Time

Suppose that the time (in hours) required to repair a car is an exponentially distributed random variable with rate parameter $\lambda = 1/2$. What is the probability that the repair time exceeds 4 hours? If it exceeds 4 hours what is the probability that it exceeds 8 hours?

Hint: Either do an integral or use Python's `scipy.stats.expon.cdf` function. [Note! scale = $1/\lambda$]

Math Solution

$$P(X \geq x) = \int_x^\infty \lambda e^{-\lambda t} dt = -e^{-\lambda t}\Big|_x^\infty = e^{-\lambda x} \tag{4}$$

Math Solution

$$P(X \geq x) = \int_x^\infty \lambda e^{-\lambda t} dt = -e^{-\lambda t}\Big|_x^\infty = e^{-\lambda x} \tag{4}$$

$$P(X \geq 4 \,|\, \lambda = .5) = e^{-2} = 0.135 \tag{5}$$

$$\tag{6}$$

Math Solution

$$P(X \geq x) = \int_x^\infty \lambda e^{-\lambda t} dt = -e^{-\lambda t}\Big|_x^\infty = e^{-\lambda x} \tag{4}$$

$$P(X \geq 4 \,|\, \lambda = .5) = e^{-2} = 0.135 \tag{5}$$

$$P(X \geq 8 \,|\, \lambda = .5, X \geq 4) = \frac{P(X \geq 8 \,|\, \lambda = .5)}{P(X \geq 4 \,|\, \lambda = .5)} = \frac{e^{-4}}{e^{-2}} = e^{-2} = 0.135 \tag{6}$$

Python Solution

```
>>> from scipy.stats import expon
>>> 1.0 - expon.cdf(4, scale=2)
0.13533528323661298
```

Python Solution

```
>>> from scipy.stats import expon
>>> 1.0 - expon.cdf(4, scale=2)
0.13533528323661298

>>> a = (1.0 - expon.cdf(8, scale=2))
>>> a / (1.0 - expon.cdf(4, scale=2))
0.13533528323661298
```

Wrapup

- Understand difference between pmf and cdf
- Know how to call distributions
- Understand what parameters are