Probability Distributions (1/30)

Note: we depart from the schedule from the Syllabus for this class to catch up on probability distributions. We will be back on track next session when we combine a discussion on simulation modeling with an application on stochastic models of demand.

Learning Objectives:

- Describe common probability distributions and examples of uncertainty each distribution is suitable for modeling. (Model)
- Apply probability distributions to analyze business problems. (Analyze)
- Write Python code to graph histograms and probability distributions and to perform simple calculations involving the PMF, PDF and CDF. (Code)

Textbook Readings: DMD 2.6-7, 3.1-8
Important concepts from the readings:

- Binomial distribution: description and application.
- Uniform distribution: description and application.
- Normal distribution: description and application.
- Computing the mean, standard deviation, and variance of common probability distributions.
- The Probability Density Function (PDF) and Cumulative Distribution Function (CDF) for continuous random variables.
- Sums of normal random variables. (Note similarity to generic formula for mean and standard deviation in DMD 2.11)
- Central limit theorem (CLT) and the Normal approximation. (We will go over CLT in next class.)

Supplementary Concepts from the Lecture:
Please refer to the video recording of the class session for details.

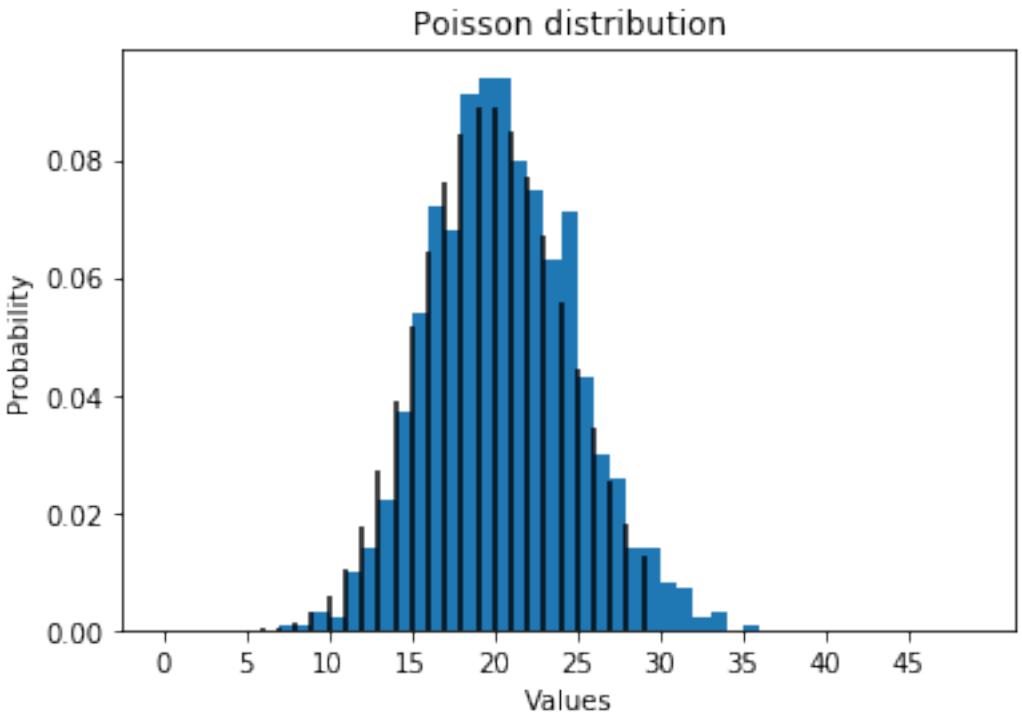
- Bernoulli distribution: description and application.
- Geometric distribution: description and application.
- Poisson distribution: description and application.
- Calculations involving PDFs and CDFs using Python.
- Plotting histograms and probability distributions in Python.

Plotting Histograms and Distribution Functions

The following is code to plot the Poisson distribution with mean 20. This models how many people visit the bank between 1-2pm, assuming the average number is 20. The Poisson distribution in general models the number of events that occur within a given time interval, assuming that events occur at a uniform rate over time. The red dashed curve shows the actual pmf (which is the exact probabilities). The histogram shows proportion of the 1000 samples that are equal to each value. When the sample size is very large, the two will match. However, with smaller samples, there may be random fluctuations so the histogram may not match exactly the PMF.

[83]:
```
from scipy.stats import poisson
import matplotlib.pyplot as plt
dist=poisson(mu=20)
data=dist.rvs(size=1000)
```

```
plt.hist(data,bins=range(50),density=True)
plt.xticks(range(0,50,5))
values=range(0,30)
plt.vlines(values,0,dist.pmf(values))        #Create vertical lines (vlines) at each va
plt.xlabel('Values')
plt.ylabel('Probability')
plt.title('Poisson distribution')
plt.show()
```



Notes:

- In certain Python versions, the code `density=True` will return an error. In those cases, simply change it to `normed=True`. This is a change in the developer of Python on how they want to name the variables. The latest version uses `density`, the old version uses `normed`.
- The variables `dist` is declared in the third line to be a Poisson distribution with mean 20. To analyze another distribution, simply change the definition of `dist`.

Calculations involving PMF and CDF

To calculate the proportion of values between 15 and 25, as well as the probability of obtaining exactly 20, we can use the cdf and pmf functions provided by scipy.stats. You can Google `scipy.stats` to get a list of all distributions and functions.

```
[84]: print('Chance that between 15 and 25 people came is',dist.cdf(25)-dist.cdf(14))
      print('Chance exactly 20 people came is',dist.pmf(20))
```

```
Chance that between 15 and 25 people came is 0.782950746174
Chance exactly 20 people came is 0.0888353173921
```

In general, calculating the probability of a random variable being inside an interval is simply a difference of CDFs. This is true for both discrete and continuous random variables. However, for discrete random variables one has to pay attention to whether the end points are included, as the CDF is the probabiliy that the random variable is smaller than or equal to a certain value, so it includes the endpoint. For discrete, an alternative way is simply to sum up the PMF within the range of interest.

Plotting Continuous Distributions

The following shows similar plotting code for the normal distribution. Note that instead of `plt.vlines`, we use `plt.plot` to show a curve. (We could have done this with pmf as well but for discrete random variables vline may be easier to read. This is personal preference.) Furthermore, we use `np.linspace` to generate the x values for the PDF, instead of `range`, so we can have more values in between. For the Poisson, `range` is more natural because it generates integers, and the Poisson values are always at integers. For the Normal, the PDF is defined for any value so we don't need to stay in the integers, so it's more convenient to interpolate with `np.linspace`. Note further than in drawing the histogram, we use 100 equidistant bins, instead of specifying the end points of the bins via `range`.

[85]:
```
from scipy.stats import norm
import matplotlib.pyplot as plt
import numpy as np
X=norm(loc=100,scale=30)
data=X.rvs(size=10000)
plt.hist(data,bins=100,density=True)
values=np.linspace(0,200,100)
plt.plot(values,X.pdf(values),'r--')
plt.xlabel('Values')
plt.ylabel('Density')
plt.title('Normal distribution')
plt.show()
```