



STATISTICAL THINKING IN PYTHON I

# Probability density functions

# Continuous variables

- Quantities that can take any value, not just discrete values

# Michelson's speed of light experiment



measured speed of light (1000 km/s)

299.85	299.74	299.90	300.07	299.93
299.85	299.95	299.98	299.98	299.88
300.00	299.98	299.93	299.65	299.76
299.81	300.00	300.00	299.96	299.96
299.96	299.94	299.96	299.94	299.88
299.80	299.85	299.88	299.90	299.84
299.83	299.79	299.81	299.88	299.88
299.83	299.80	299.79	299.76	299.80
299.88	299.88	299.88	299.86	299.72
299.72	299.62	299.86	299.97	299.95
299.88	299.91	299.85	299.87	299.84
299.84	299.85	299.84	299.84	299.84
299.89	299.81	299.81	299.82	299.80
299.77	299.76	299.74	299.75	299.76
299.91	299.92	299.89	299.86	299.88
299.72	299.84	299.85	299.85	299.78
299.89	299.84	299.78	299.81	299.76
299.81	299.79	299.81	299.82	299.85
299.87	299.87	299.81	299.74	299.81
299.94	299.95	299.80	299.81	299.87

Image: public domain, Smithsonian

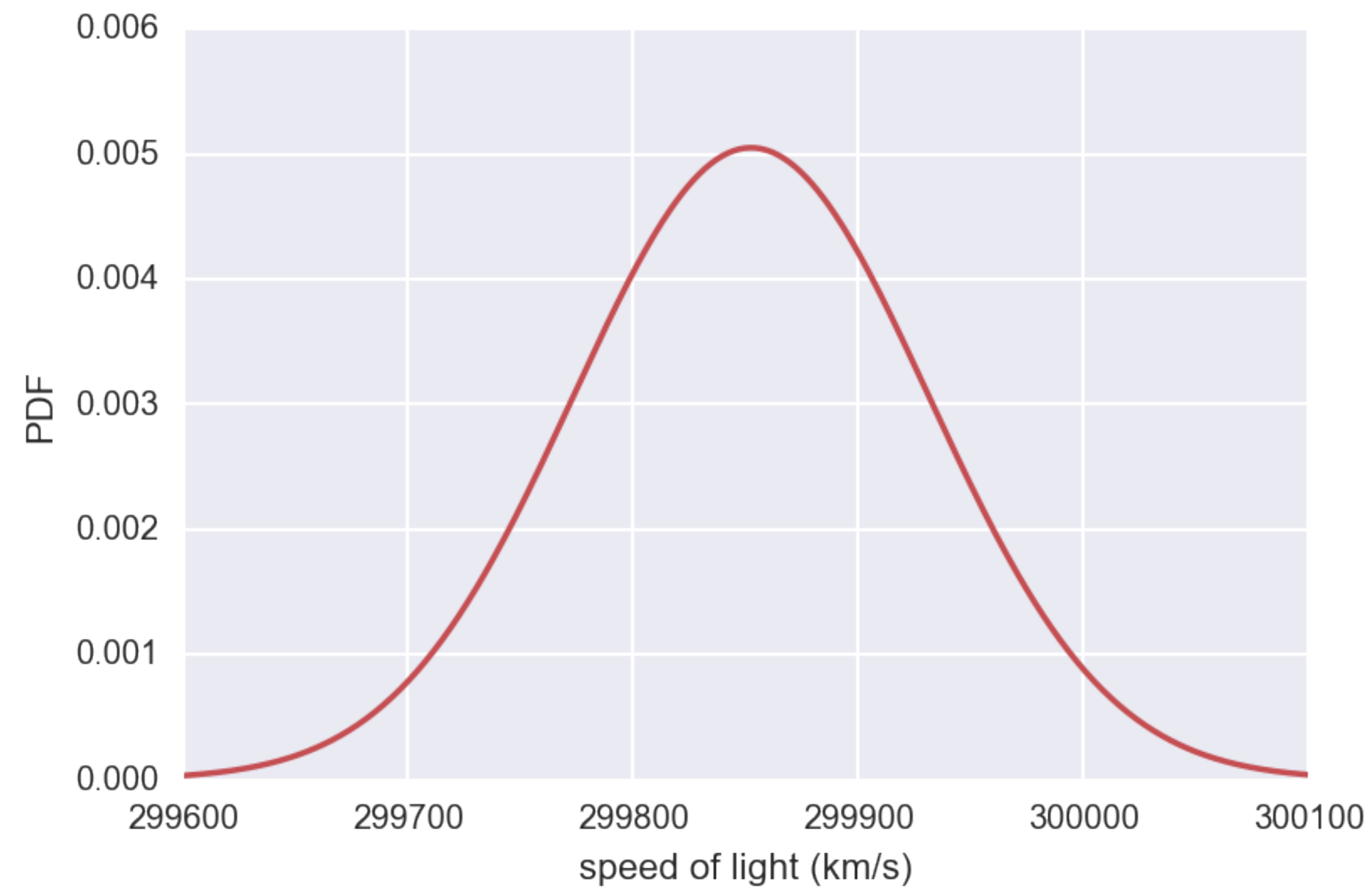
Data: Michelson, 1880

# Probability density function (PDF)

- Continuous analog to the PMF
- Mathematical description of the relative likelihood of observing a value of a continuous variable

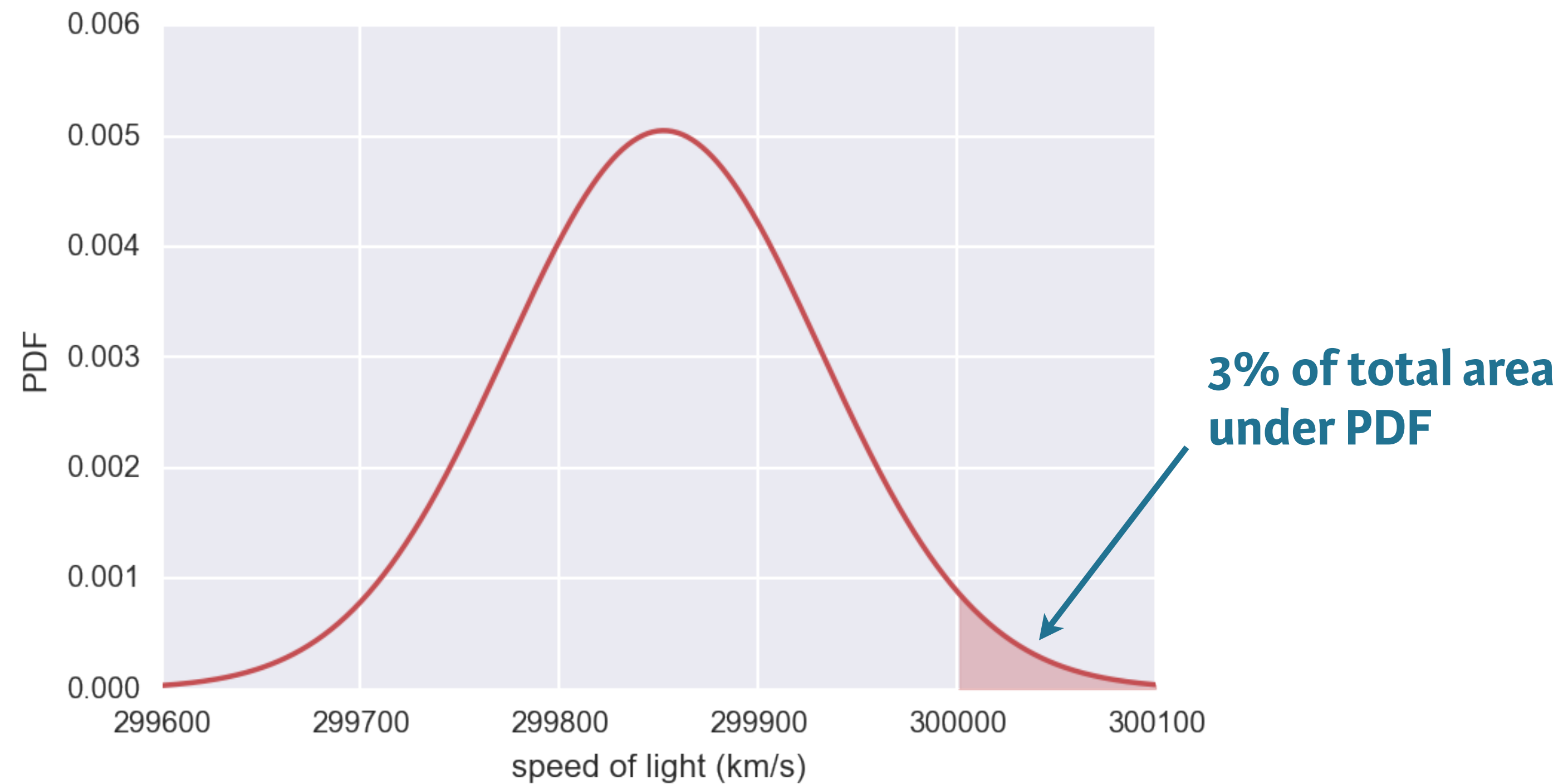


# Normal PDF



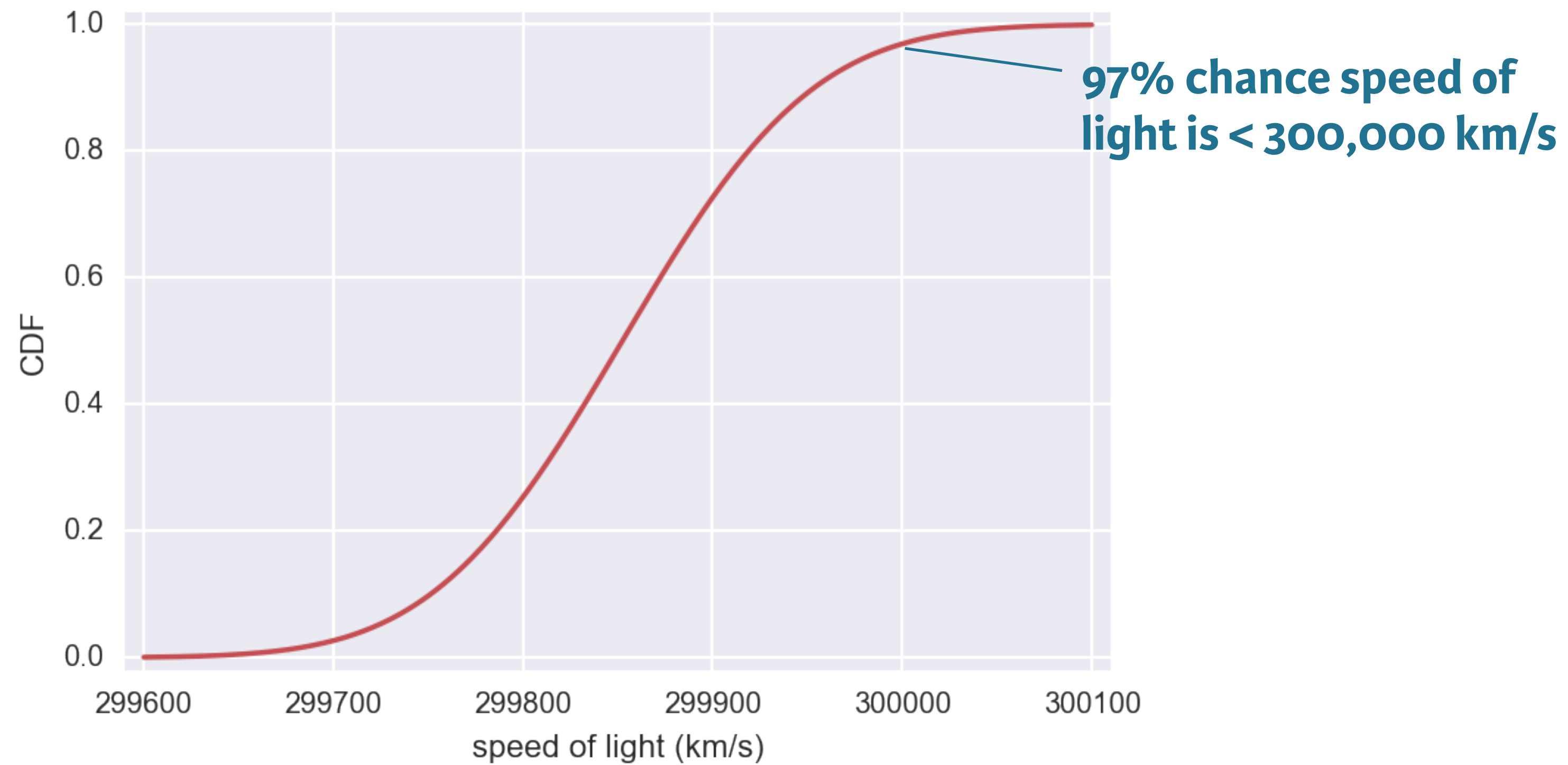


# Normal PDF





# Normal CDF





STATISTICAL THINKING IN PYTHON I

**Let's practice!**





STATISTICAL THINKING IN PYTHON I

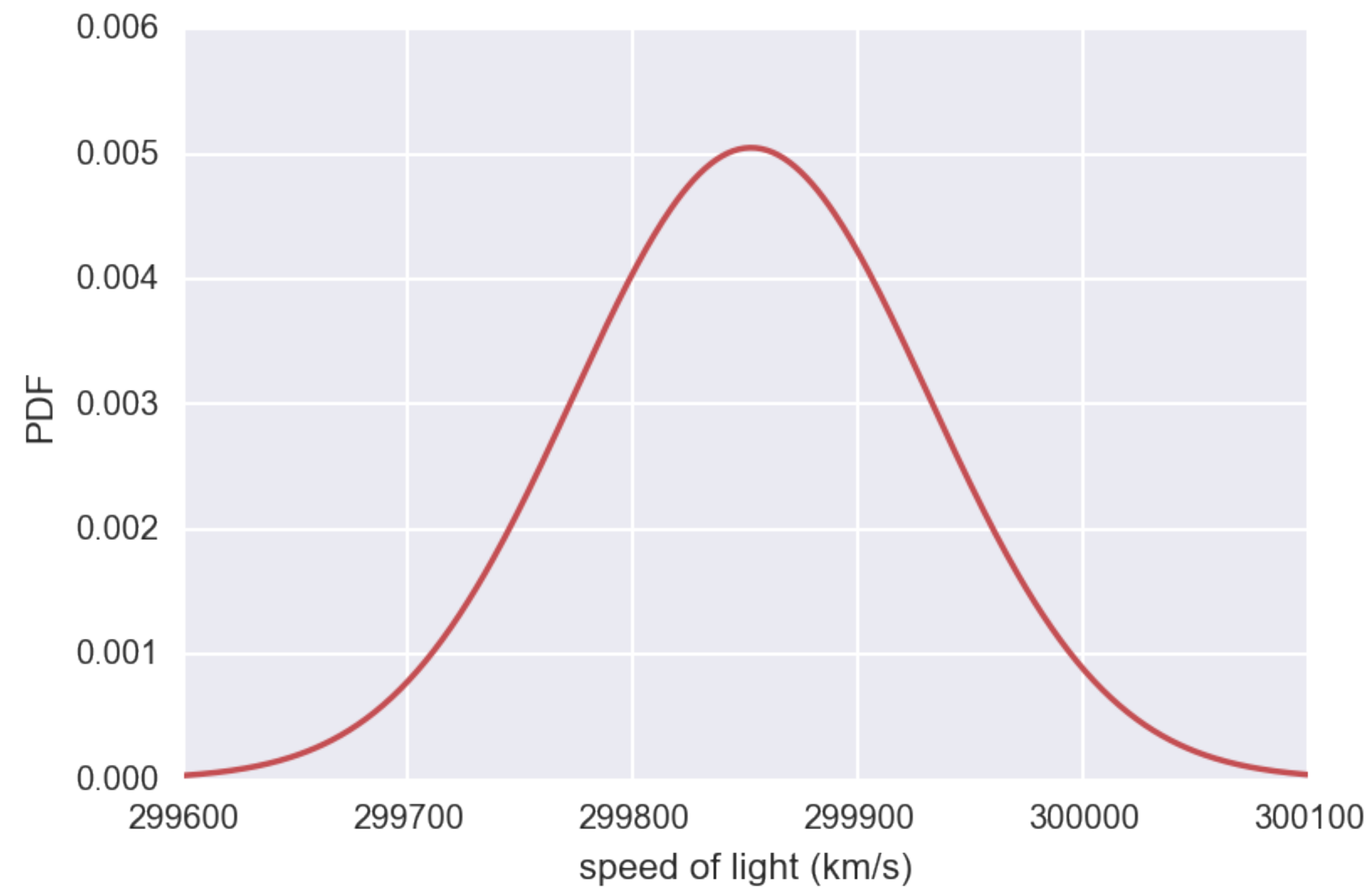
# **Introduction to the Normal distribution**

# Normal distribution

- Describes a continuous variable whose PDF has a single symmetric peak.

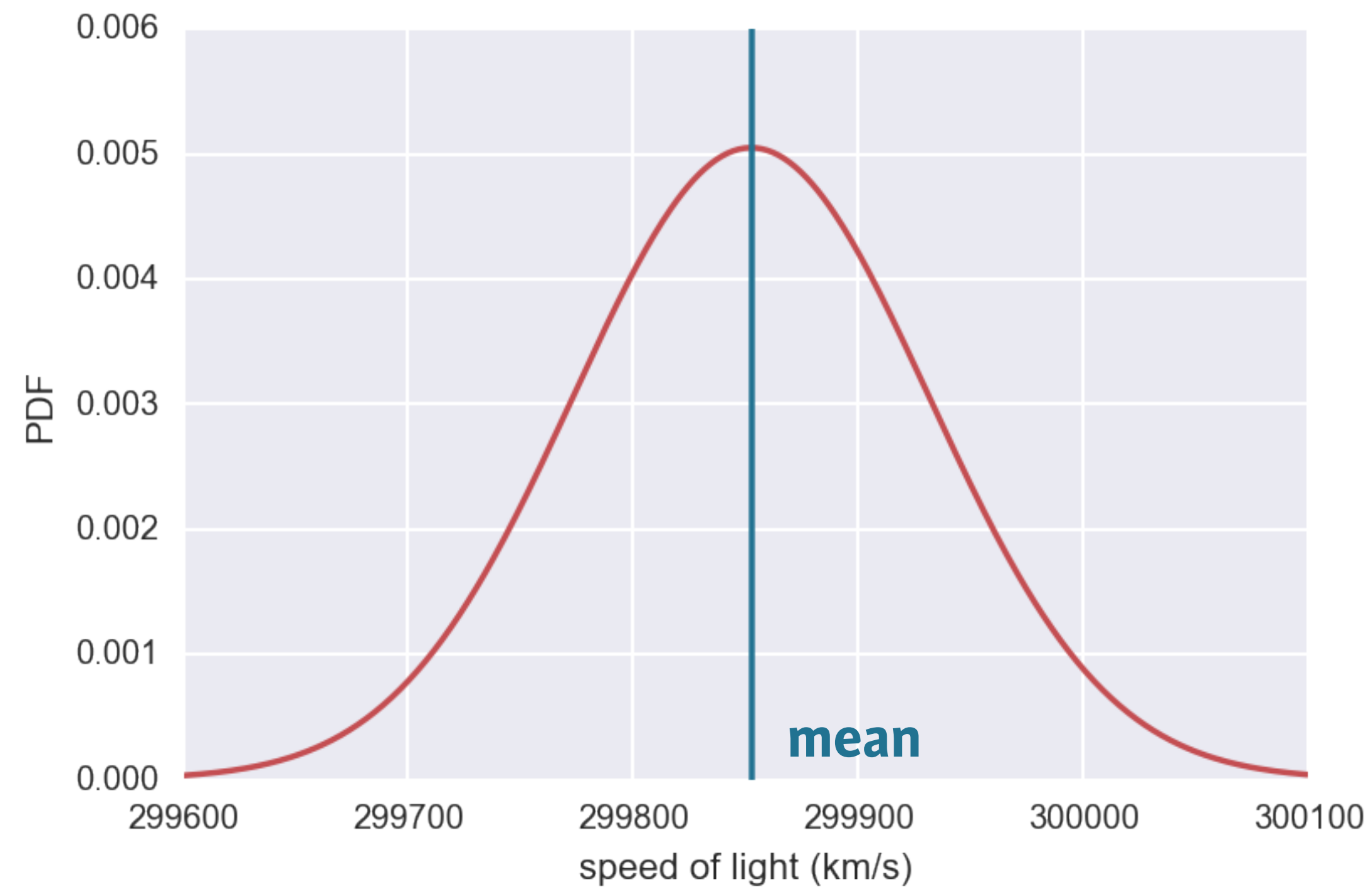


# Normal distribution



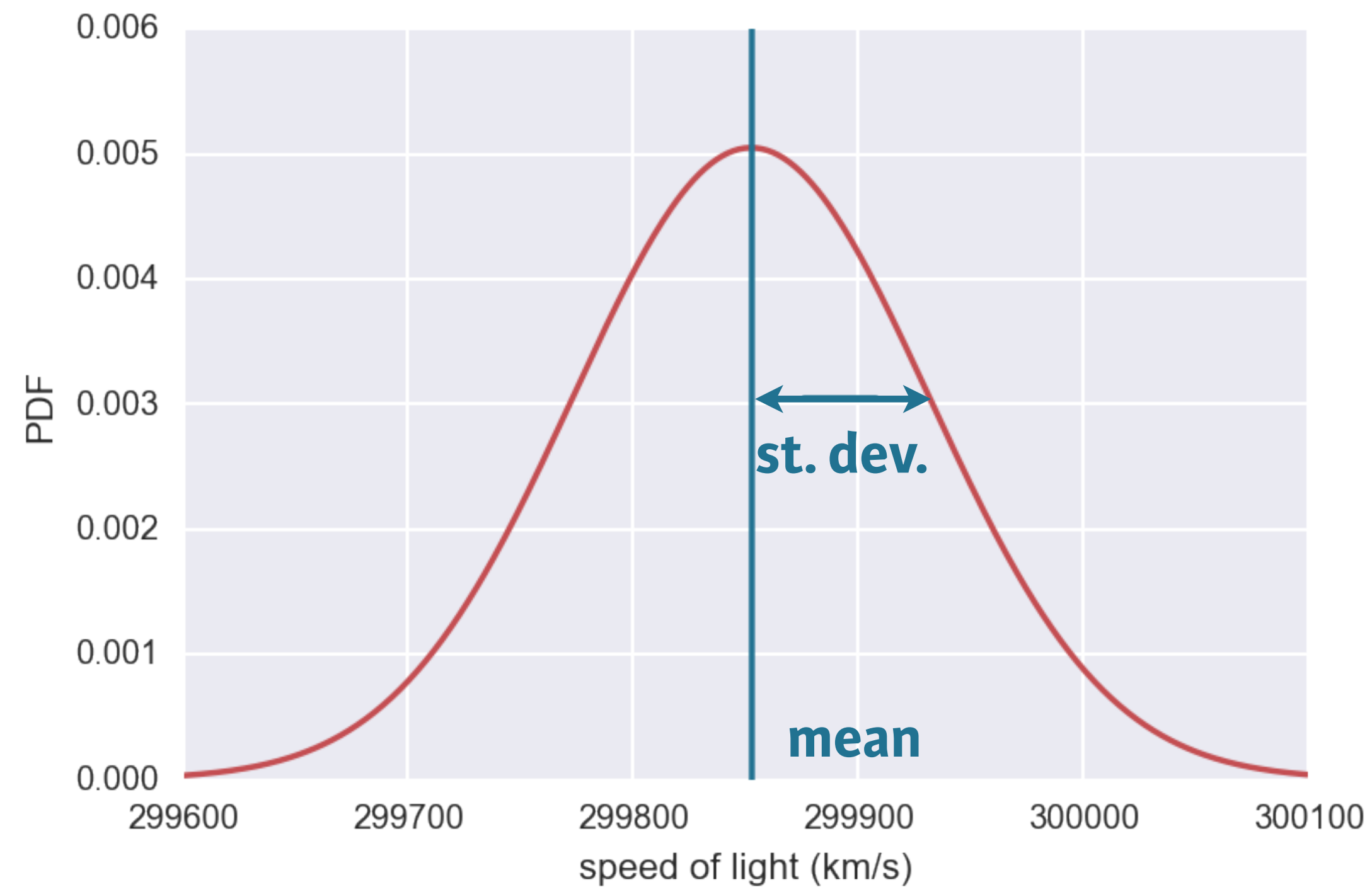


# Normal distribution





# Normal distribution





## Parameter

mean of a  
Normal distribution

st. dev. of a  
Normal distribution

## Calculated from data

mean computed  
from data

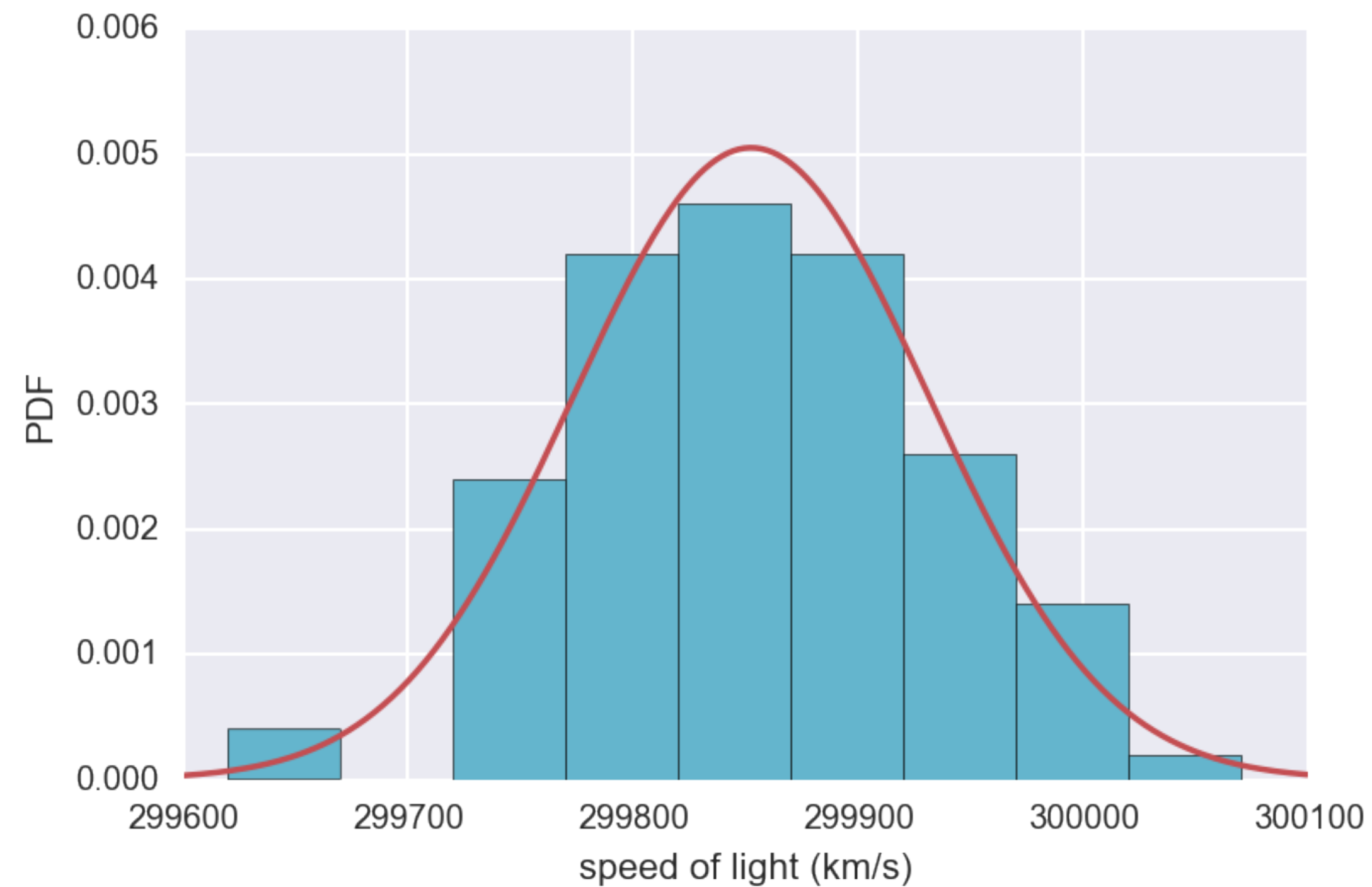
standard deviation  
computed from data

≠

≠



# Comparing data to a Normal PDF





# Checking Normality of Michelson data

```
In [1]: import numpy as np

In [2]: mean = np.mean(michelson_speed_of_light)

In [3]: std = np.std(michelson_speed_of_light)

In [4]: samples = np.random.normal(mean, std, size=10000)

In [5]: x, y = ecdf(michelson_speed_of_light)

In [6]: x_theor, y_theor = ecdf(samples)
```





# Checking Normality of Michelson data

```
In [1]: import matplotlib.pyplot as plt
```

```
In [2]: import seaborn as sns
```

```
In [3]: sns.set()
```

```
In [4]: _ = plt.plot(x_theor, y_theor)
```

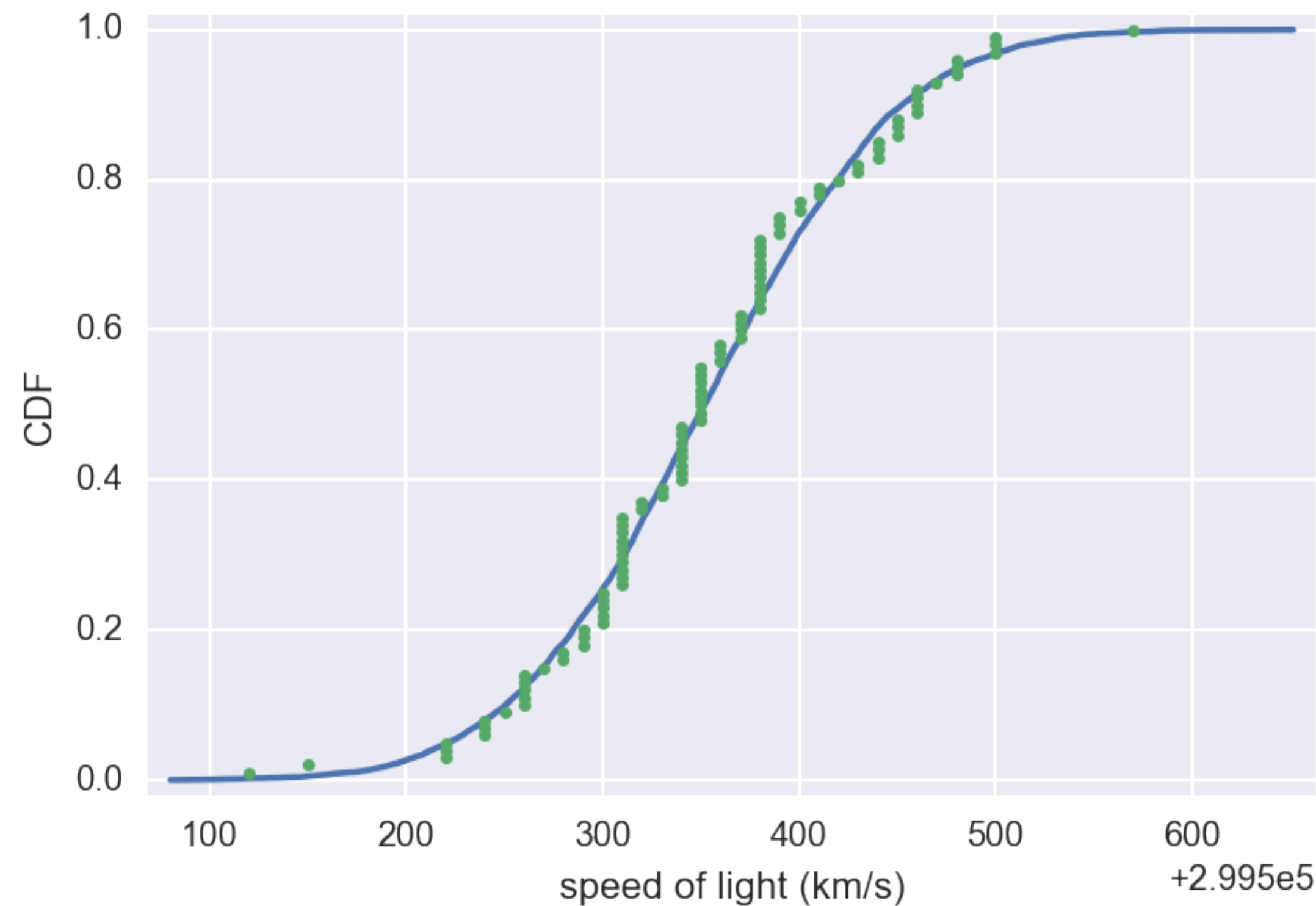
```
In [5]: _ = plt.plot(x, y, marker='.', linestyle='none')
```

```
In [6]: _ = plt.xlabel('speed of light (km/s)')
```

```
In [7]: _ = plt.ylabel('CDF')
```

```
In [8]: plt.show()
```

# Checking Normality of Michelson data





STATISTICAL THINKING IN PYTHON I

**Let's practice!**



STATISTICAL THINKING IN PYTHON I

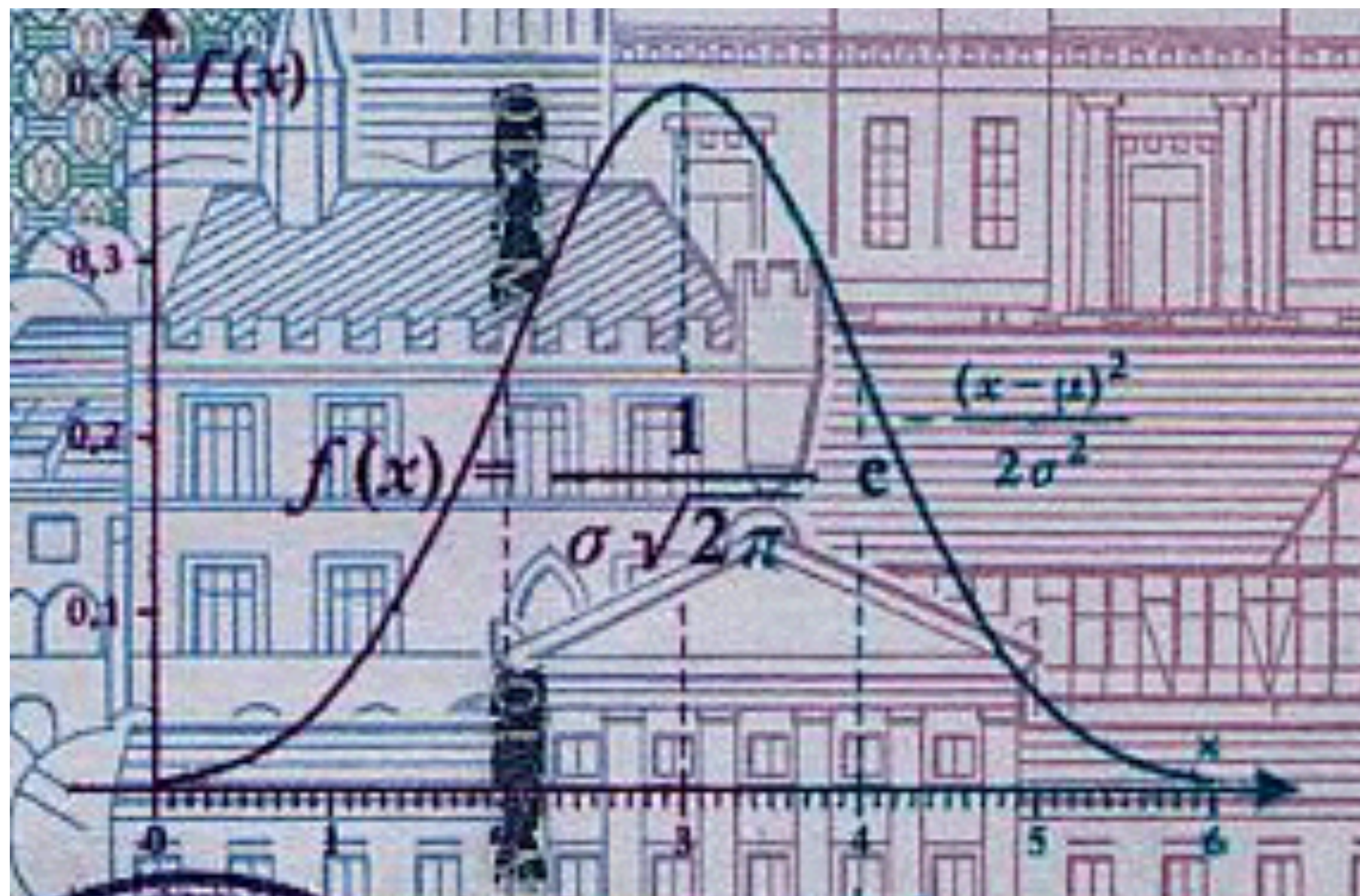
# **The Normal distribution: Properties and warnings**





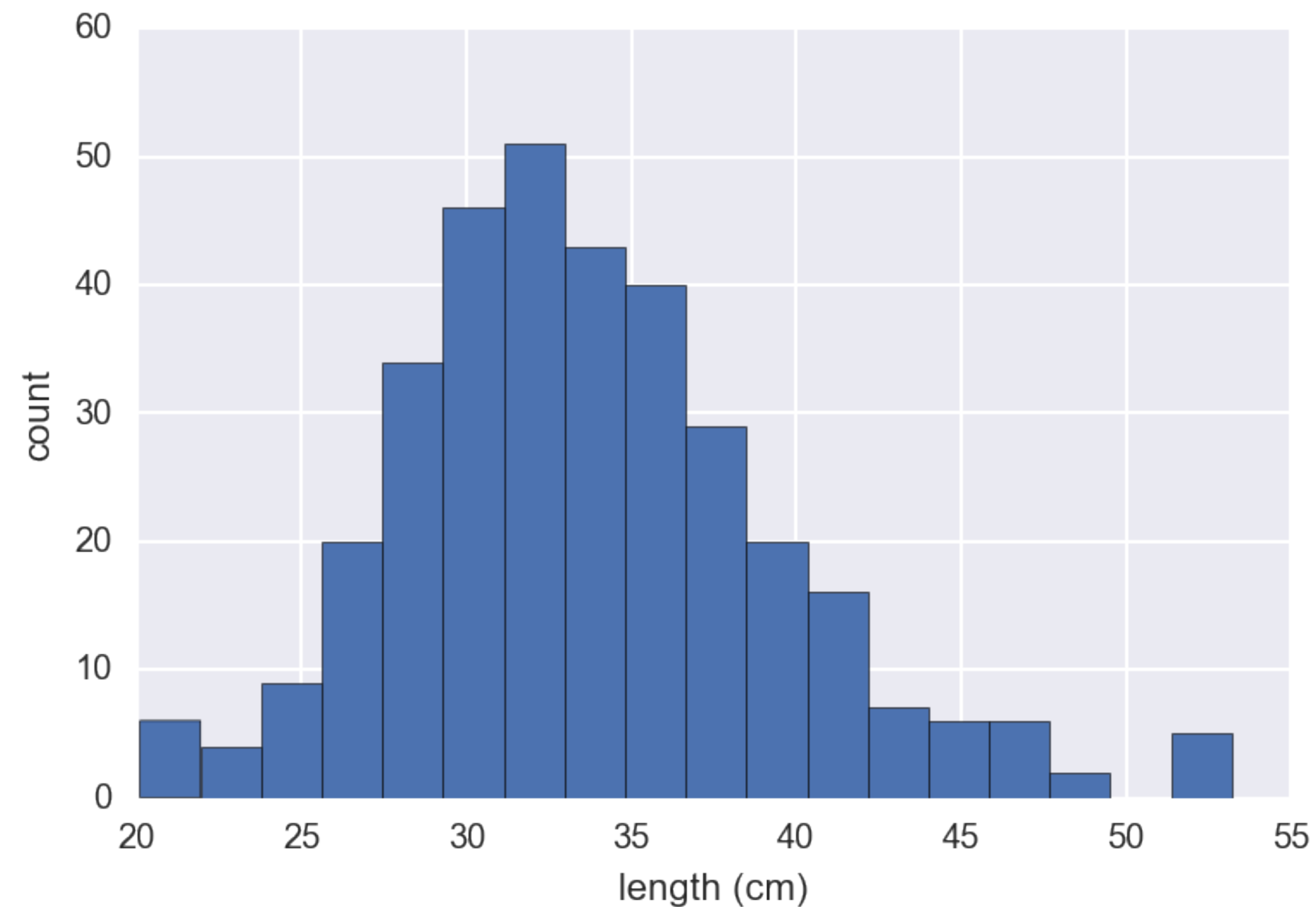


# The Gaussian distribution

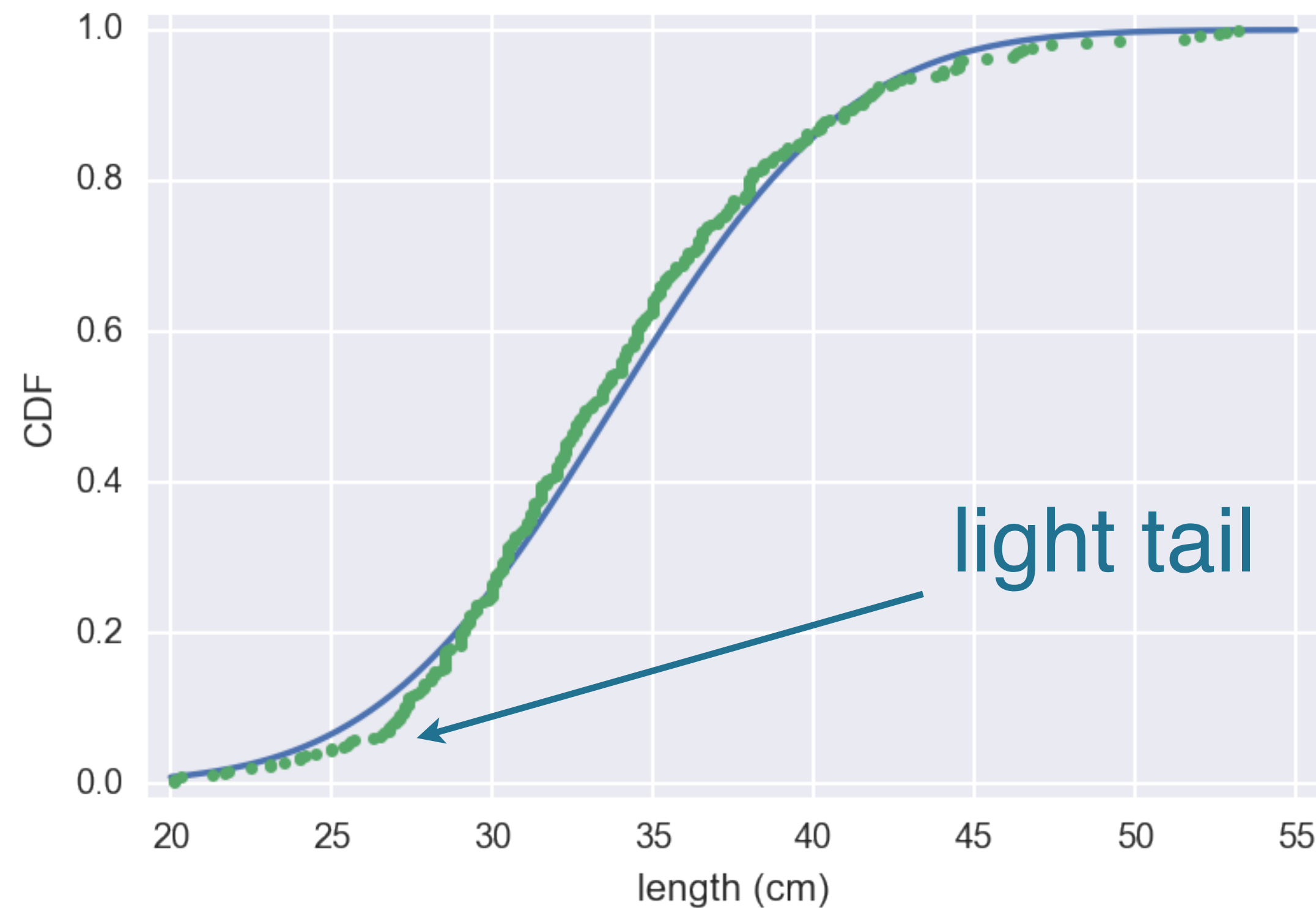




# Length of MA large mouth bass



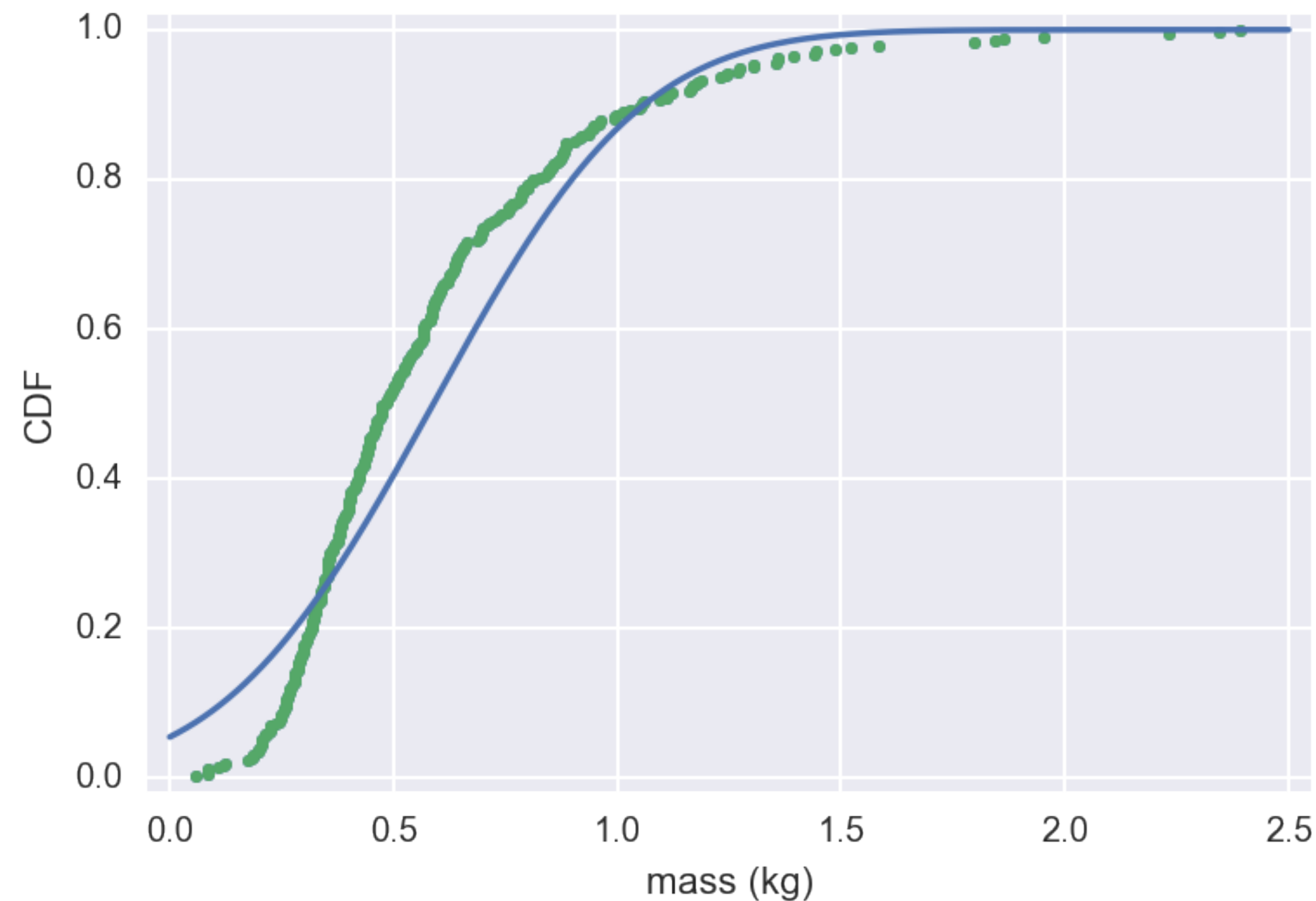
# Length of MA large mouth bass





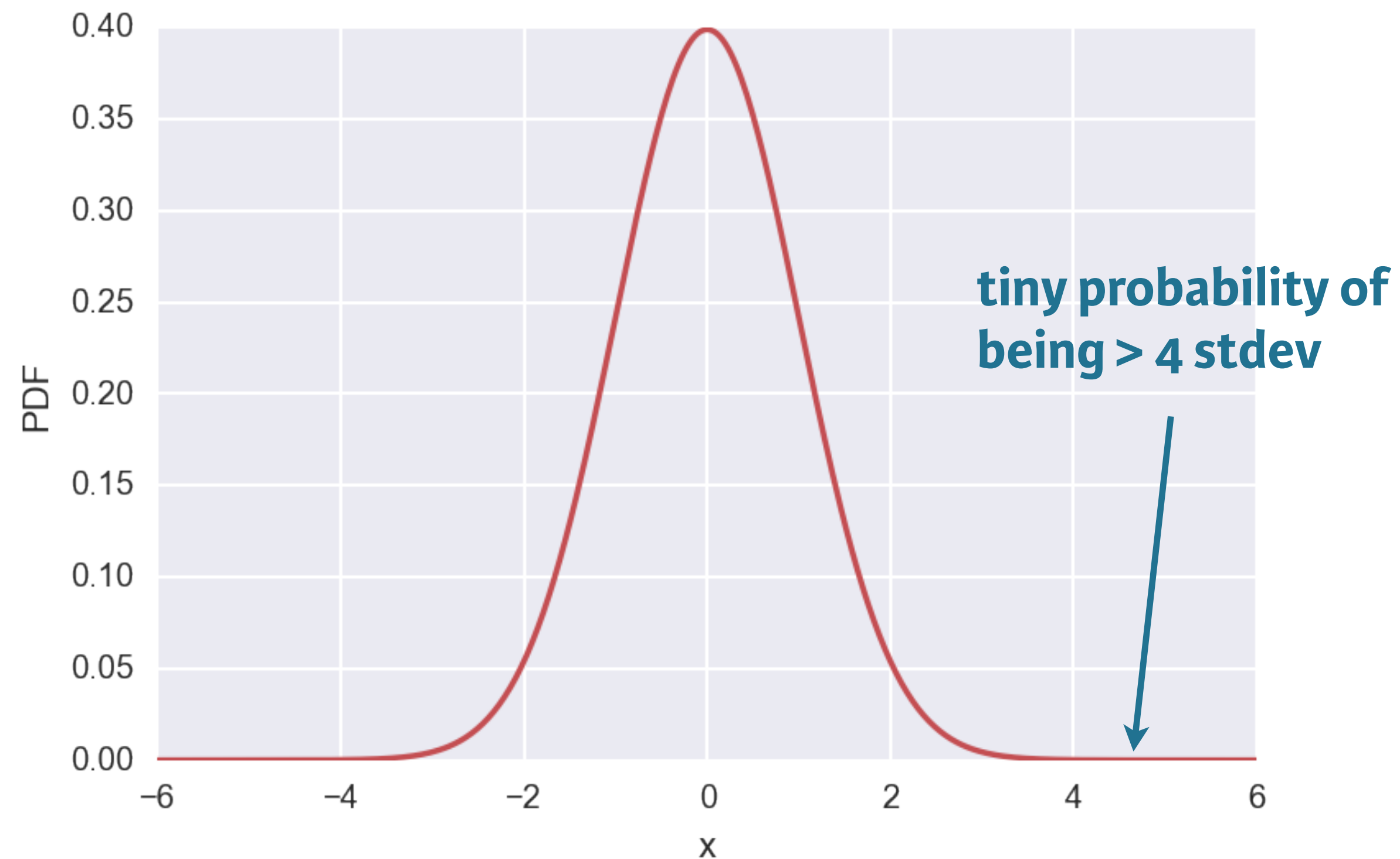


# Mass of MA large mouth bass





# Light tails of the Normal distribution





STATISTICAL THINKING IN PYTHON I

**Let's practice!**



STATISTICAL THINKING IN PYTHON I

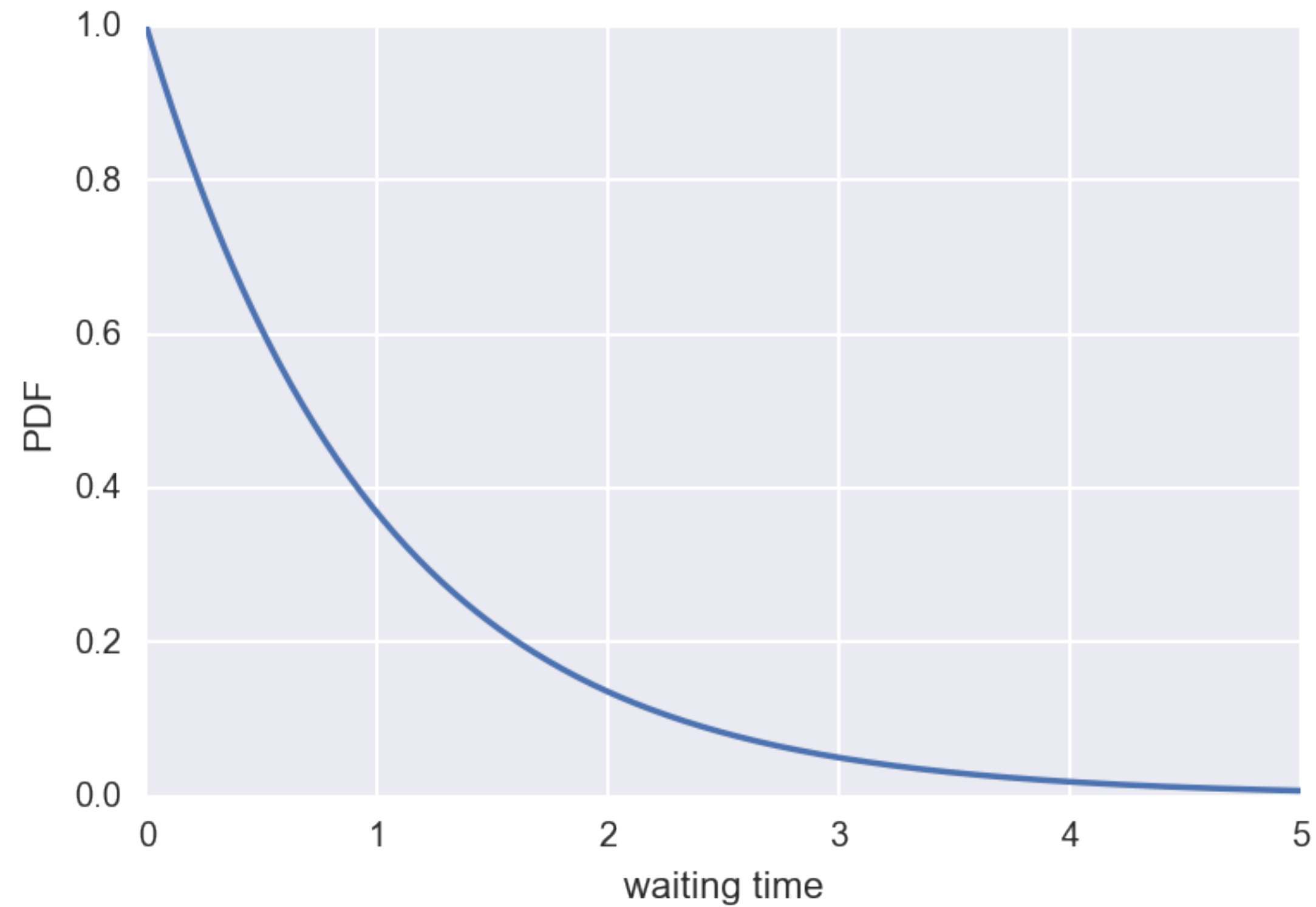
# The Exponential distribution

# The Exponential distribution

- The waiting time between arrivals of a Poisson process is Exponentially distributed



# The Exponential PDF



# Possible Poisson process

- Nuclear incidents:
  - Timing of one is independent of all others



# Exponential inter-incident times

```
In [1]: mean = np.mean(inter_times)

In [2]: samples = np.random.exponential(mean, size=10000)

In [3]: x, y = ecdf(inter_times)

In [4]: x_theor, y_theor = ecdf(samples)

In [5]: _ = plt.plot(x_theor, y_theor)

In [6]: _ = plt.plot(x, y, marker='.', linestyle='none')

In [7]: _ = plt.xlabel('time (days)')

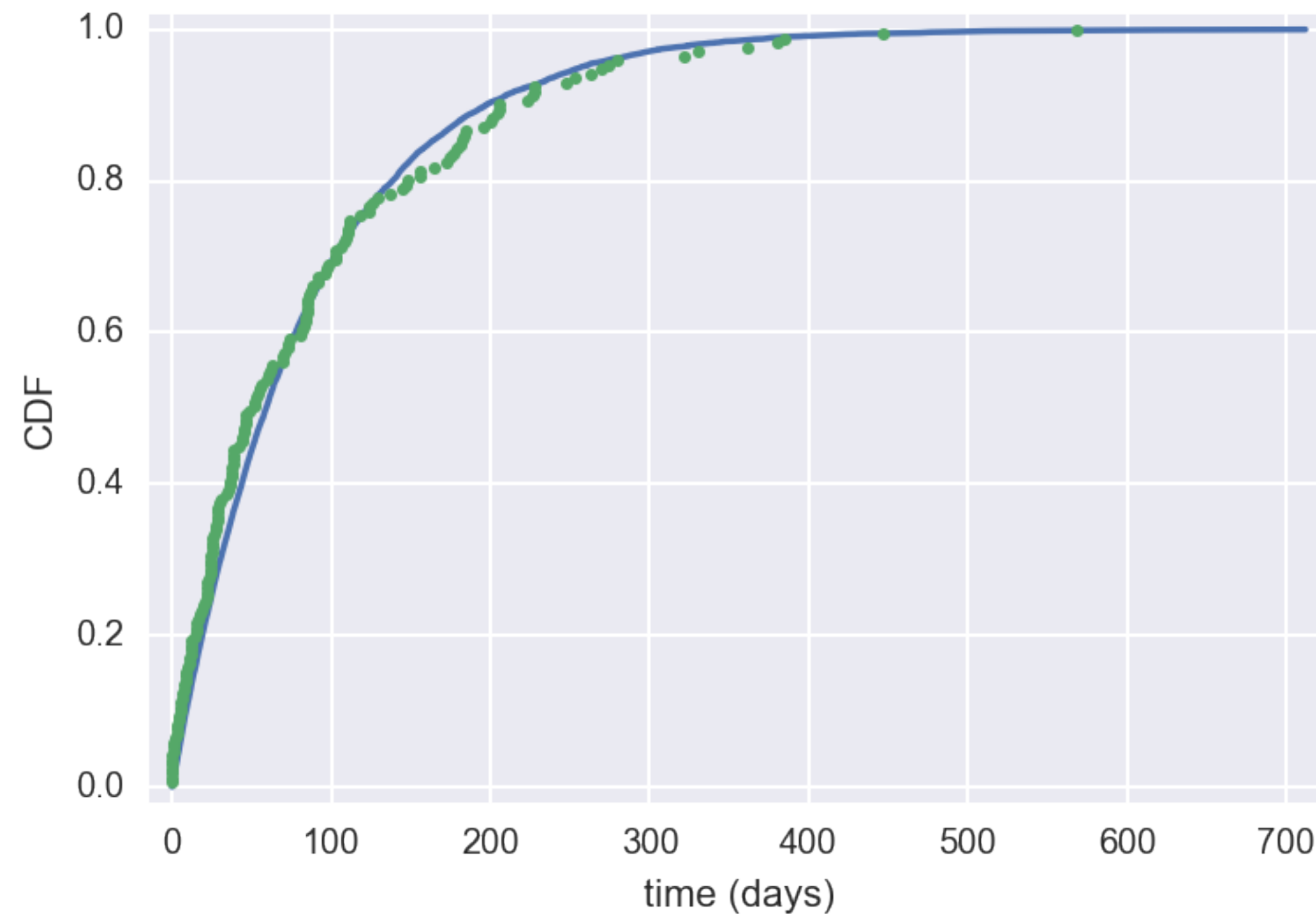
In [8]: _ = plt.ylabel('CDF')

In [9]: plt.show()
```





# Exponential inter-incident times





STATISTICAL THINKING IN PYTHON I

**Let's practice!**



STATISTICAL THINKING IN PYTHON I

# Final thoughts

# You now can...

- Construct (beautiful) instructive plots
- Compute informative summary statistics
- Use hacker statistics
- Think probabilistically

# In the sequel, you will...

- Estimate parameter values
- Perform linear regressions
- Compute confidence intervals
- Perform hypothesis tests



STATISTICAL THINKING IN PYTHON I

**See you in the sequel!**