homework-01

August 23, 2023

1 Homework 1

1.1 References

- Lectures 1 through 4 (inclusive).

1.2 Instructions

- Type your name and email in the "Student details" section below.
- Develop the code and generate the figures you need to solve the problems using this notebook.
- For the answers that require a mathematical proof or derivation you should type them using latex. If you have never written latex before and you find it exceedingly difficult, we will likely accept handwritten solutions.
- The total homework points are 100. Please note that the problems are not weighed equally.

[ ]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import matplotlib_inline
matplotlib_inline.backend_inline.set_matplotlib_formats('svg')
import seaborn as sns
sns.set_context("paper")
sns.set_style("ticks")

import numpy as np
import scipy
import scipy.stats as st
import urllib.request
import os

def download(
    url : str,
    local_filename : str = None
):
    """Download a file from a url.

    Arguments
    url            -- The url we want to download.
    local_filename -- The filemame to write on. If not
                      specified
```

```
    """
    if local_filename is None:
        local_filename = os.path.basename(url)
    urllib.request.urlretrieve(url, local_filename)
```

```
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy
version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected
version 1.25.2
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

1.3 Student details

- **First Name:** Stav
- **Last Name:** Zeliger
- **Email:** szeliger@purdue.edu
- **Used generative AI to complete this assignment (Yes/No):** No
- **Which generative AI tool did you use (if applicable)?:** N/A

1.4 Problem 1

Disclaimer: This example is a modified version of the one found in a 2013 lecture on Bayesian Scientific Computing taught by Prof. Nicholas Zabaras. I am not sure where the original problem is coming from.

We are tasked with assessing the usefulness of a tuberculosis test. The prior information I is:

> The percentage of the population infected by tuberculosis is 0.4%. We have run several experiments and determined that: + If a tested patient has the disease, then 80% of the time the test comes out positive. + If a tested patient does not have the disease, then 90% of the time the test comes out negative.

To facilitate your analysis, consider the following logical sentences concerning a patient:

> A: The patient is tested and the test is positive. B: The patient has tuberculosis.

A. Find the probability that the patient has tuberculosis (before looking at the result of the test), i.e., $p(B|I)$. This is known as the base rate or the prior probability. **Answer:** $p(B|I) = \frac{0.4\%}{100\%} = 0.004$

B. Find the probability that the test is positive given that the patient has tuberculosis, i.e., $p(A|B,I)$. **Answer:** $p(A|B,I) = \frac{80\%}{100\%} = 0.8$

C. Find the probability that the test is positive given that the patient does not have tuberculosis, i.e., $p(A|\neg B,I)$. **Answer:** $p(A|\neg B,I) = 1 - \frac{90\%}{100\%} = 0.1$

D. Find the probability that a patient that tested positive has tuberculosis, i.e., $p(B|A,I)$. **Answer:** $P(B|A,I) = \frac{P(A|B,I)P(B,I)}{P(A|I)} = \frac{(0.8)(0.004)}{P(A|B,I)P(B|I)+P(A|\neg B,I)P(\neg B|I)} = \frac{0.0032}{0.8(0.004)+0.1(1-0.004)} \approx 0.031$

E. Find the probability that a patient that tested negative has tuberculosis, i.e., $p(B|\neg A,I)$. Does the test change our prior state of knowledge about about the patient? Is the test useful? **Answer:** $P(B|\neg A,I) = \frac{P(\neg A|B,I)P(B|I)}{P(\neg A|B,I)P(B|I)+P(\neg A|\neg B,I)P(\neg B|I)} = \frac{(1-0.8)(0.004)}{(1-0.8)(0.004)+(1-0.1)(1-0.004)} \approx 0.00089$ This is useful information, since it means that if the test is negative, the patient almost certainly does not

have tuberculosis. If we were to do multiple tests, if any of them came back negative, we can be fairly certain the patient is healthy.

F. What would a good test look like? Find values for

$$p(A|B, I) = p(\text{test is positive}|\text{has tuberculosis}, I),$$

and

$$p(A|\neg B, I) = p(\text{test is positive}|\text{does not have tuberculosis}, I),$$

so that

$$p(B|A, I) = p(\text{has tuberculosis}|\text{test is positive}, I) = 0.99.$$

There are more than one solutions. How would you pick a good one? Thinking in this way can help you set goals if you work in R&D. If you have time, try to figure out whether or not there exists such an accurate test for tuberculosis **Answer:** One possible solution to drive $p(B|A, I) \to 0.99$ is to set $p(A|\neg B, I) \approx 0.00004 = 0.004\%$ and $p(A|B, I) \approx 0.99 = 99\%$. This means that the test is exceedingly good at avoiding false positives. This value has the largest effect on the overall accuracy, since (assuming no screening for symptoms), the test is being administered to many more people who do are healthy. This can help set the goal for the test, with the highest priority being reducing the false positive rate.

1.5 Problem 2 - Practice with discrete random variables

Consider the Categorical random variable:

$$X \sim \text{Categorical}(0.3, 0.1, 0.2, 0.4),$$

taking values in $\{0, 1, 2, 3\}$. Find the following (you may use `scipy.stats.rv_discrete` or do it by hand):

A. The expectation $\mathbb{E}[X]$.

Answer:

[ ]:
```
# You can also answer with code here:
x = [0,1,2,3]
p = [0.3,0.1,0.2,0.4]
rv = st.rv_discrete(values=(x,p))
E = rv.expect()
print(E)
```

```
1.7000000000000002
```

B. The variance $\mathbb{V}[X]$.

Answer:

[ ]:
```
# You can also answer with code here:
x = [0,1,2,3]
p = [0.3,0.1,0.2,0.4]
rv = st.rv_discrete(values=(x,p))
v = rv.var()
print(v)
```

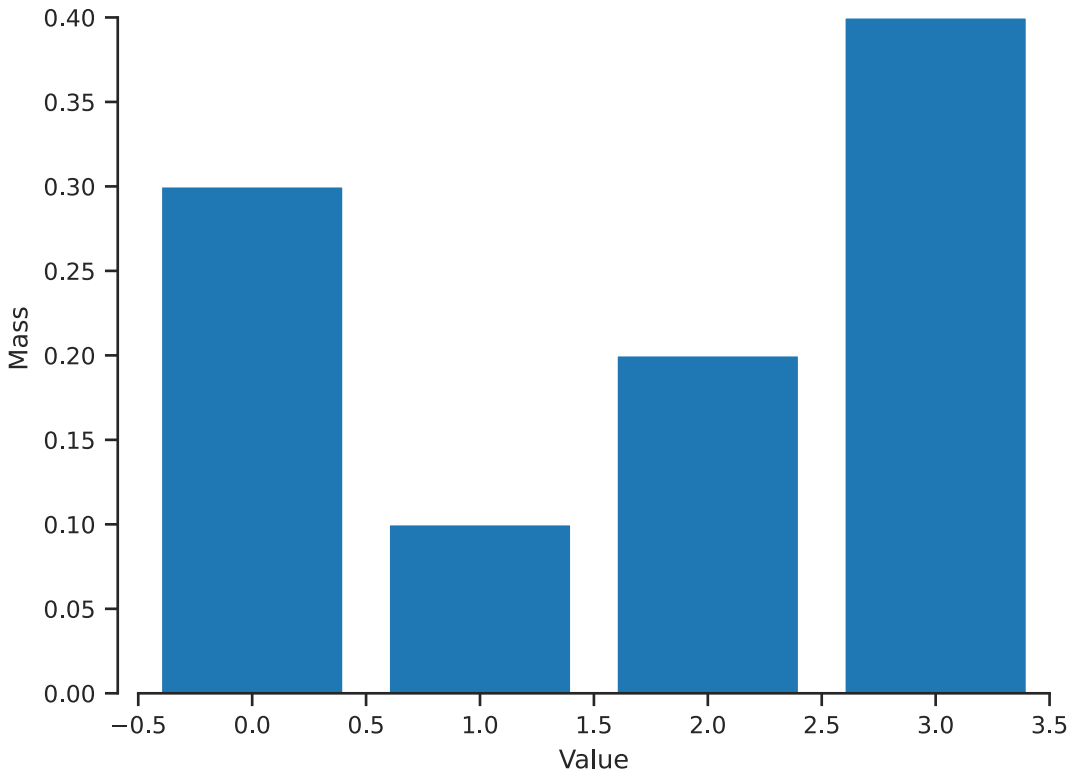
```
1.6099999999999994
```

C. Plot the probability mass function of X.

[ ]:
```
# Your code here. Hint: use a bar plot
x = [0,1,2,3]
p = [0.3,0.1,0.2,0.4]

rv = st.rv_discrete(values=(x,p))
pmf = rv.pmf(x)

fig, ax = plt.subplots(dpi=150)
ax.bar(x, pmf)
ax.set_xlabel('Value')
ax.set_ylabel('Mass')
sns.despine(trim=True);
```



D. Find the probability that X is in $\{0, 2\}$.

Answer:

[ ]:
```
# You can also answer with code here:
x = [0,1,2,3]
p = [0.3,0.1,0.2,0.4]

rv = st.rv_discrete(values=(x,p))
p = rv.cdf(2)
print(p)
```

```
0.6000000000000001
```

E. Find $\mathbb{E}[4X + 3]$.

Answer:

[ ]:
```
# You can also answer with code here:
x = [0,1,2,3]
p = [0.3,0.1,0.2,0.4]
rv = st.rv_discrete(values=(x,p))
E = 4*rv.expect() + 3
print(E)
```

F. Find $\mathbb{V}[4X + 3]$.

Answer:

[ ]:
```
# You can also answer with code here:
x = [0,1,2,3]
p = [0.3,0.1,0.2,0.4]
rv = st.rv_discrete(values=(x,p))
V = 4**2*rv.var()
print(V)
```

```
25.75999999999999
```

1.6 Problem 3 - Predicting the probability of major earthquakes in Southern California

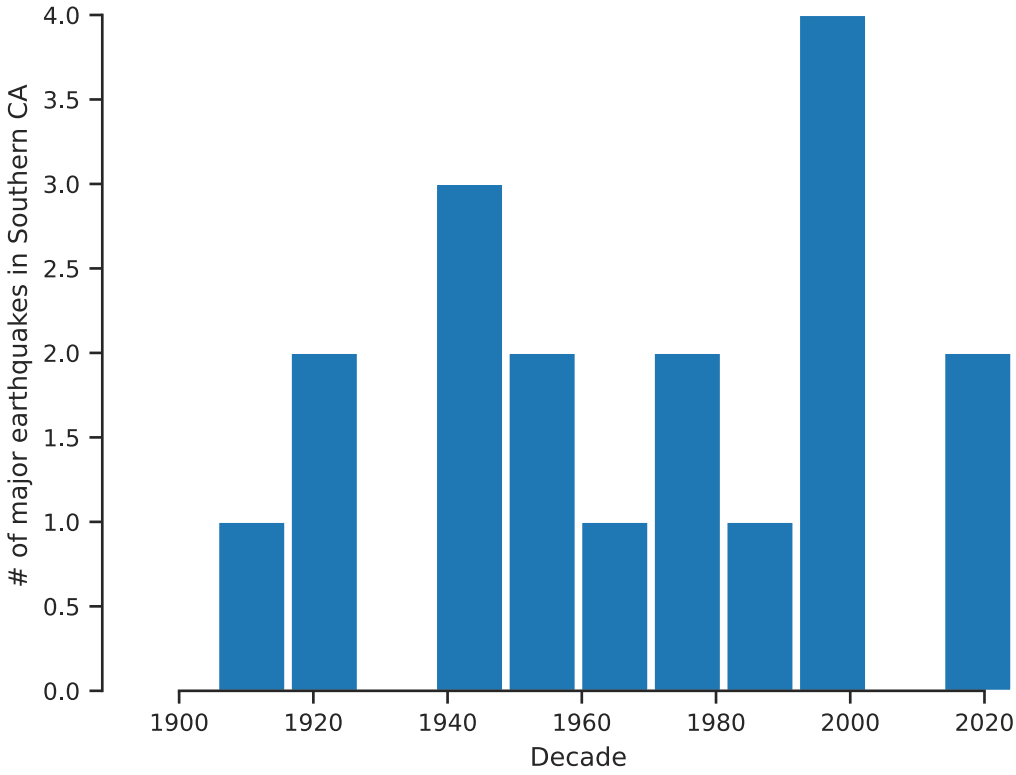
The San Andreas fault extends through California forming the boundary between the Pacific and the North American tectonic plates. It has caused some of the major earthquakes on Earth. We are going to focus on Southern California and we would like to assess the probability of a major earthquake, defined as an earthquake of magnitude 6.5 or greater, during the next ten years.

A. The first thing we are going to do is go over a database of past earthquakes that have occured in Southern California and collect the relevant data. We are going to start at 1900 because data before that time may are unreliable. Go over each decade and count the occurence of a major

earthquake (i.e., count the number of organge and red colors in each decade). We have done this for you.

[ ]:
```
eq_data = np.array([
    0, # 1900-1909
    1, # 1910-1919
    2, # 1920-1929
    0, # 1930-1939
    3, # 1940-1949
    2, # 1950-1959
    1, # 1960-1969
    2, # 1970-1979
    1, # 1980-1989
    4, # 1990-1999
    0, # 2000-2009
    2 # 2010-2019
])
fig, ax = plt.subplots(dpi=150)
ax.bar(np.linspace(1900, 2019, eq_data.shape[0]), eq_data, width=10)
ax.set_xlabel('Decade')
ax.set_ylabel('# of major earthquakes in Southern CA')
plt.legend(loc="best", frameon=False)
sns.despine(trim=True);
```

```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



B. The Poisson distribution is a discrete distribution with values $\{0, 1, 2, \dots\}$ which is commonly used to model the number of events occuring in a certain time period. It is the right choice when these events are happening independently and the probability of any event happening over a small period of time is constant. Let's use the Poisson to model the number of earthquakes X occuring in a decade. We write:

$$X \sim \text{Poisson}(r),$$

where r is the *rate parameter* of Poisson. The rate is the number of events per time period. Here, r is the number of earthquakes per decade. Using the data above, we can set the rate as the empirical average of the observed number of earthquakes per decade:

[ ]:
```
r = np.mean(eq_data)
print('r = {0:1.2f} major earthquakes per decade'.format(r))
```

```
r = 1.50 major earthquakes per decade
```

Strictly speaking, **this is not how you should be calibrating models!!!** We will learn about the **right** way (which uses Bayes' rule) in the subsequent lectures. But it will do for now as the answer you would get using the **right** way is, for this problem, almost the same. Let's define a Poisson distribution using `scipy.stats.poisson` (see documentation here):

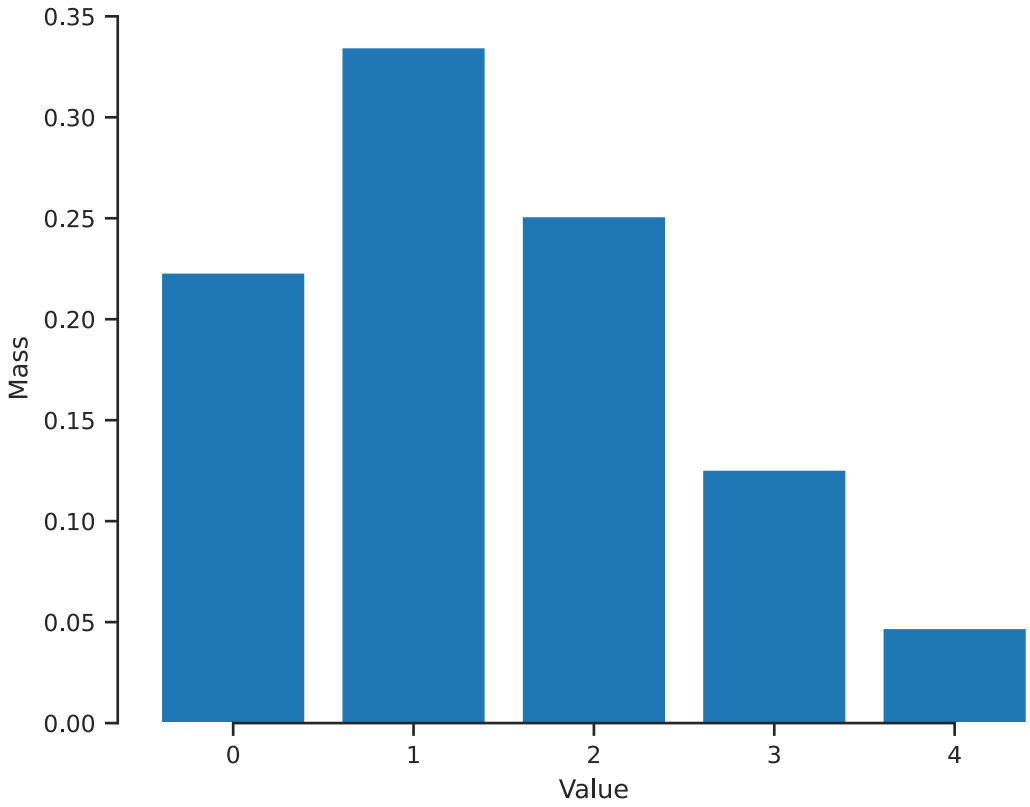
[ ]:
```
X = st.poisson(r)
```

A. Plot the probability mass function of `X`.

[ ]:
```
# Your code here
pmf = X.pmf(eq_data)

fig, ax = plt.subplots(dpi=150)
ax.bar(eq_data, pmf)
ax.set_xlabel('Value')
ax.set_ylabel('Mass')

sns.despine(trim=True)
```



B. What is the probability that at least one major earthquake will occur during the next decade?
Answer:

[ ]:
```
# You can also answer with code here:
P = 1 - X.cdf(0)
print(P)
```

```
0.7768698398515702
```

C. What is the probability that at least one major earthquake will occur during the next two decades? Hint: Consider two independent and identical copies of X, say X_1 and X_2. And consider their sum $Y = X_1 + X_2$. Read this about the sum of two independent Poisson distributions. **Answer:**

[ ]:
```
# You can also answer with code here:
X_sum = st.poisson(r+r)
P = 1 - X_sum.cdf(0)
print(P)
```

```
0.950212931632136
```

D. What is the probability that at least one major earthquake will occur during the next five decades? **Answer:**

[ ]:
```
# You can also answer with code here:
X_sum = st.poisson(5*r)
P = 1 - X_sum.cdf(0)
print(P)
```

```
0.9994469156298522
```

1.7 Problem 4 - Failure of a mechanical component

Assume that you designing a gear for a mechanical system. Under normal operating conditions the gear is expected to fail at a random time. Let T be a random variable capturing the time the gear fails. What should the probability density of T look like?

Here are some hypothetical data to work with. Suppose that we took ten gears and we worked them until failure. The failure times (say in years) are as follows:

[ ]:
```
time_to_fail_data = np.array(
    [
        10.5,
        7.5,
        8.1,
        8.4,
        11.2,
        9.3,
        8.9,
        12.4
    ]
)
```

Why does each gear fail at different times? There are several sources of uncertainty. The most important are:

- Manufacturing imperfections.
- Different loading conditions.

If this was a controlled fatigue experiment, then we could eliminate the second source of uncertainty by using exactly the same loading conditions.

Now, we are going to fit a probability density function to these data. Which one should we use? Well, new gears do not fail easily. So, the probability density function of T should be close to zero for small T. As time goes by, the probability density should increase because various things start happening to the material, e.g., crack formation, fatigue, etc. Finally, the probability density must again start going to zero as time further increases because nothing lasts forever... A probability distribution that is commonly used to model this situation is the Weibull. We are going to fit some fail time data to a Weibull distribution and then you will have to answer a few questions about failing times.

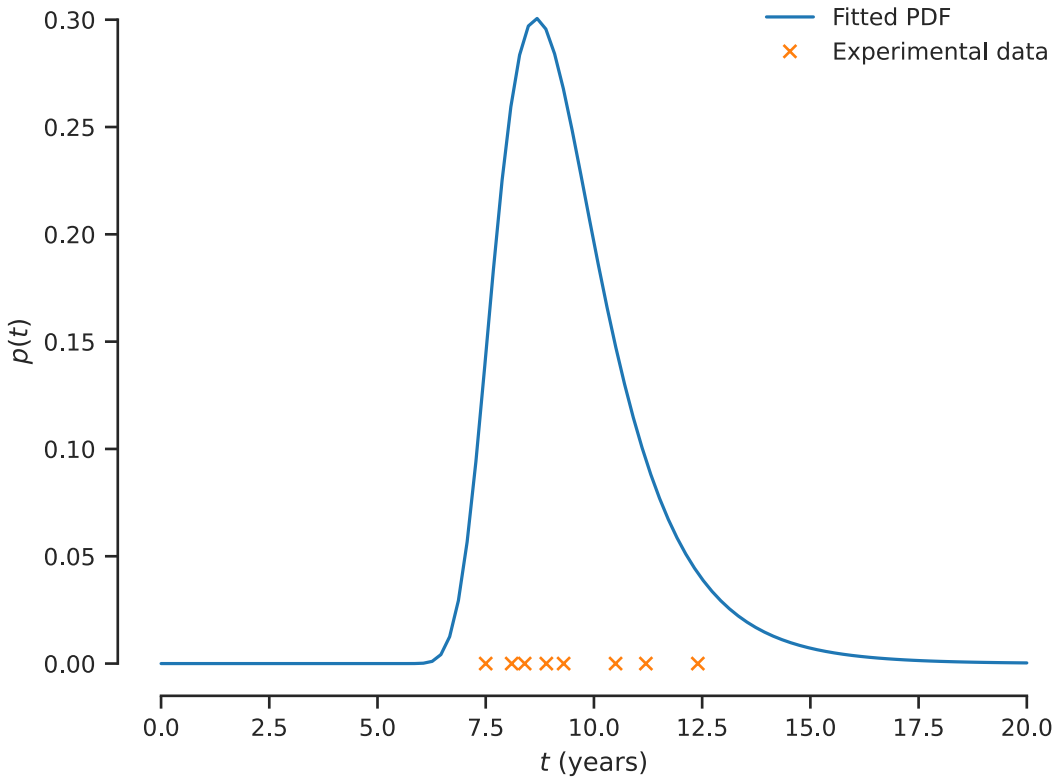
The Weibull has parameters and we are going to fit them to the available data. The method we are going to use is called the *maximum likelihood method*. We haven't really talked about this, and it is not important to know what it is to do this homework problem. We will learn about maximum likelihood in later lectures. Here is how we fit the parameters using `scipy.stats`:

[ ]:
```
fitted_params = st.exponweib.fit(time_to_fail_data, loc=0)
T = st.exponweib(*fitted_params)
print(f"Fitted parameters: {fitted_params}")
```

```
Fitted parameters: (448.066965711728, 0.7099665338918923, 3.4218808260575804,
0.41627831297126994)
```

Let's plot the fitted Weibul PDF and the data we used:

[ ]:
```
fig, ax = plt.subplots()
ts = np.linspace(0.0, 20.0, 100)
ax.plot(
    ts,
    T.pdf(ts),
    label="Fitted PDF"
)
ax.plot(
    time_to_fail_data,
    np.zeros_like(time_to_fail_data),
    "x",
    label="Experimental data"
)
ax.set_xlabel(r"$t$ (years)")
ax.set_ylabel(r"$p(t)$")
plt.legend(loc="best", frameon=False)
sns.despine(trim=True);
```



Now you have to answer a series of questions about the random variable T that we just fitted.

A. Find the mean fail time and its variance. Hint: Do not integrate anything by hand. Just use the functionality of `scipy.stats`.

[ ]:
```
# Your code here
t_mean = T.mean() # Change me
t_var = T.var() # Change me
print(f"E[T] = {t_mean:.2f}")
print(f"V[T] = {t_var:.2f}")
```

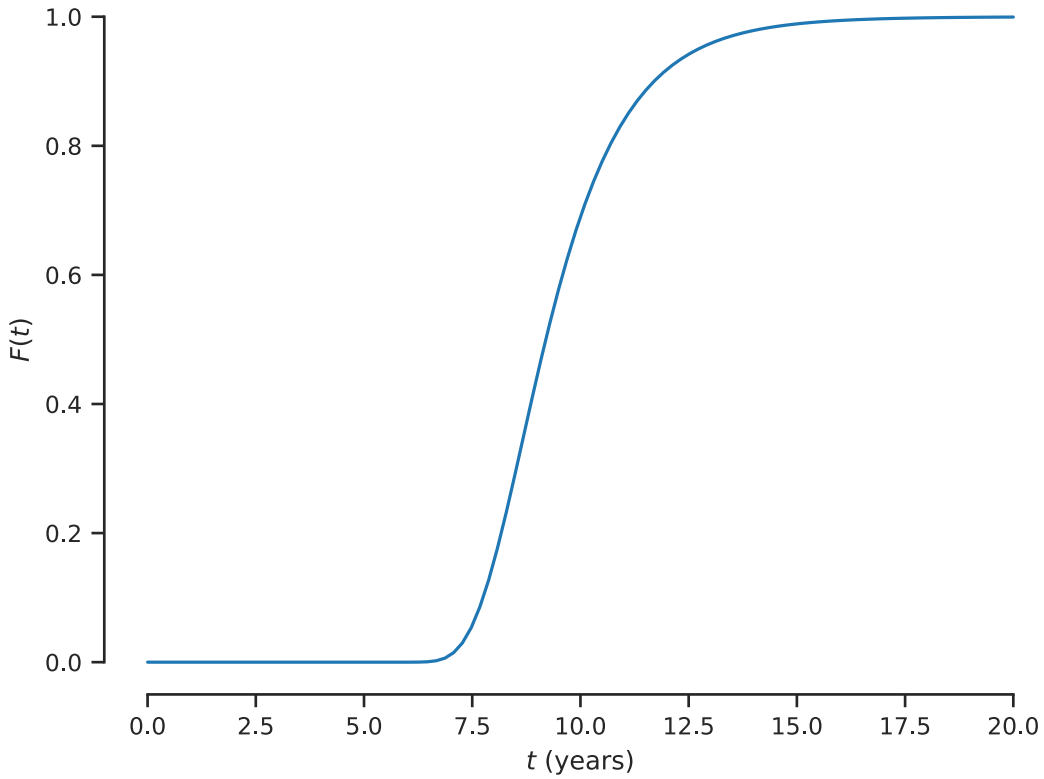
```
E[T] = 9.53
V[T] = 2.88
```

B. Plot the cumulative distribution function $F(t) = P(T \leq t)$ of T.

[ ]:
```
# Your code here
fig, ax = plt.subplots()
ax.plot(
    ts,
    T.cdf(ts)
)
```

```
ax.set_xlabel(r"$t$ (years)")
ax.set_ylabel(r"$F(t)$")
plt.legend(loc="best", frameon=False)
sns.despine(trim=True);
```

```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



C. Plot the probability that gear survives for more than t as a function of t. That is, plot the function:

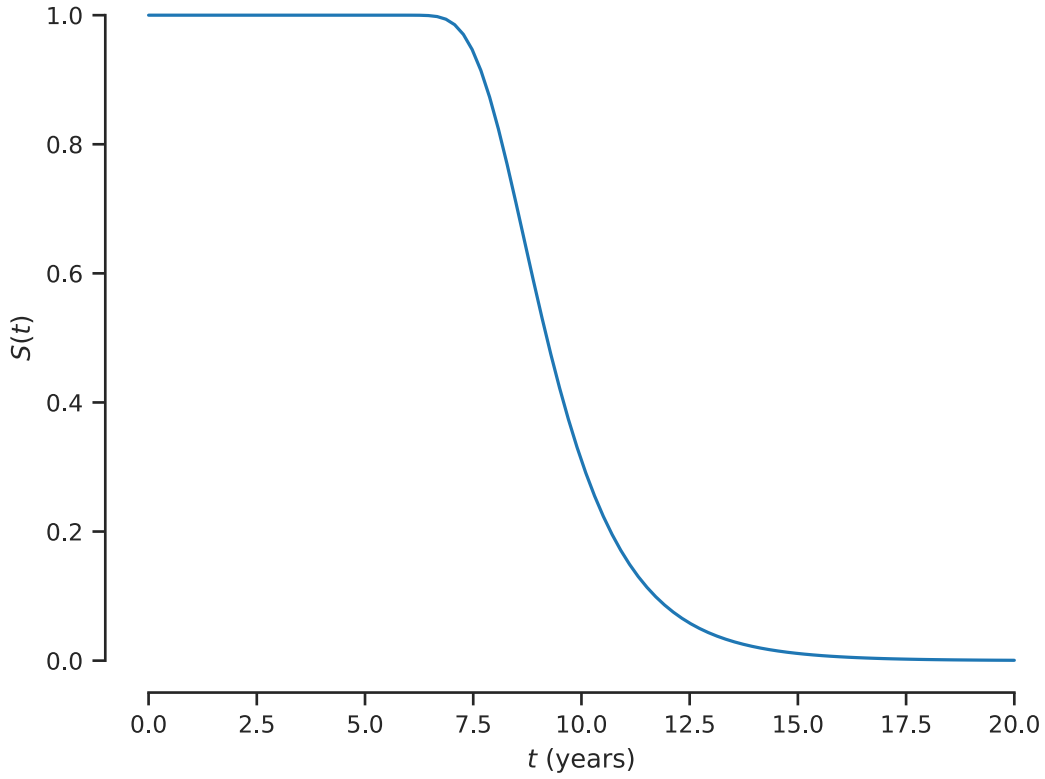
$$S(t) = p(T > t).$$

Hint: First connect $S(t)$ to the cumulative distribution function $F(t)$ of T.

[ ]:
```
# Your code here
fig, ax = plt.subplots()
ax.plot(
    ts,
    1-T.cdf(ts)
)
```

```
ax.set_xlabel(r"$t$ (years)")
ax.set_ylabel(r"$S(t)$")
plt.legend(loc="best", frameon=False)
sns.despine(trim=True);
```

```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



D. Find the probability that the gear lasts anywhere between 8 and 10 years.

[ ]:
```
# Your code here
middleProb = T.cdf(10) - T.cdf(8)
print(middleProb)
```

```
0.5342898192604528
```

E. Find the time t^* such that the probability that the gear fails before t^* is 0.01.

[ ]:
```
tStarPot = np.linspace(0.0, 10.0, 10000001)
tStarPos = np.argmin(np.absolute(T.cdf(tStarPot)- 0.01) )
print("Time: ", tStarPot[tStarPos])
```

```
print("P(t*): ", T.cdf(tStarPot[tStarPos]))
```

```
Time:  6.975009999999999
P(t*):  0.010000000670348695
```