Theodore Koby-Hercsky
DSC530-T302

5.2 Exercises

Exercises

Exercise 5-1: In the BRFSS (see Section 5.4), the distribution of heights is roughly normal with parameters μ = 178 cm and σ = 7.7 cm for men, and μ = 163 cm and σ = 7.3 cm for women.

In order to join Blue Man Group, you have to be male between 5'10" and 6'1" (see http://bluemancasting.com). What percentage of the U.S. male population is in this range? Hint: use `scipy.stats.norm.cdf`.

`scipy.stats` contains objects that represent analytic distributions

In [27]:
```
import scipy.stats
```

For example `scipy.stats.norm` represents a normal distribution.

In [28]:
```
mu = 178
sigma = 7.7
dist = scipy.stats.norm(loc=mu, scale=sigma)
type(dist)
```

Out[28]:
```
scipy.stats._distn_infrastructure.rv_frozen
```

A "frozen random variable" can compute its mean and standard deviation.

In [29]:
```
dist.mean(), dist.std()
```

Out[29]:
```
(178.0, 7.7)
```

It can also evaluate its CDF. How many people are more than one standard deviation below the mean? About 16%

In [30]:
```
dist.cdf(mu-sigma)
```

Out[30]: 0.1586552539314574

How many people are between 5'10" and 6'1"?

In [31]:
```
# To find how many people are between 5'10" or 177.8 cm and
# 6'1" or 185.42 cm we can first calculate the cdf of the distribution
# at both heights then take 6'1" cdf and minus it by the cdf for 5'10".
tall_cdf = dist.cdf(185.42)
short_cdf = dist.cdf(177.8)
people_between = tall_cdf - short_cdf
people_between
```

Out[31]: 0.3427468376314737

```
As seen above we can see that 34.27% fall between 5'10" and 6'1"
meaning that 34.27% of the people would fall into the correct hight to
be part of the Blue Man Group.
```

Exercise 5-2: To get a feel for the Pareto distribution, let's see how different the world would be if the distribution of human height were Pareto. With the parameters xm = 1 m and α = 1.7, we get a distribution with a reasonable minimum, 1 m, and median, 1.5 m.

Plot this distribution. What is the mean human height in Pareto world? What fraction of the population is shorter than the mean? If there are 7 billion people in Pareto world, how many do we expect to be taller than 1 km? How tall do we expect the tallest person to be?

`scipy.stats.pareto` represents a pareto distribution. In Pareto world, the distribution of human heights has parameters alpha=1.7 and xmin=1 meter. So the shortest person is 100 cm and the median is 150.

In [33]:
```
alpha = 1.7
xmin = 1        # meter
dist = scipy.stats.pareto(b=alpha, scale=xmin)
dist.median()
```

Out[33]: 1.5034066538560549

What is the mean height in Pareto world?

In [35]:
```
# First we will need to calculate the mean hight of an individual in
# Pareto world as seen below by using the dist and setting it to the mea
Pareto_mean = dist.mean()
Pareto_mean
```

Out[35]: 2.428571428571429

```
As seen above we see that the mean is 2.43 in Pareto world.
```

What fraction of people are shorter than the mean?

In [36]:
```
# When determining the fraction of the people that are shorter than
# the mean from Pareto world we will calculate the cdf distribution of
# the mean that was just calculated.
Shorterthan_mean = dist.cdf(Pareto_mean)
Shorterthan_mean
```

Out[36]:
```
0.778739697565288
```

```
From the above calculate we see that 77.87% of the people in Pareto
world are shorter than the mean.
```

Out of 7 billion people, how many do we expect to be taller than 1 km? You could use `dist.cdf` or `dist.sf`.

In [42]:
```
# Next I will calculate the amount of people that are expected to be
# taller than 1 km by using the dist.cdf.
expected_taller = 7000000000 * (1-dist.cdf(1000))
expected_taller
```

Out[42]:
```
55602.976430479954
```

```
As seen above we see that 55,602 people that are suspected of being
taller than 1km.
```

How tall do we expect the tallest person to be?

In [43]:
```
# When calculating the expected tallestest person we can use the percent
# point function to calculate the height in meters of the tallest person
tallest_person = dist.ppf(1-1/7000000000)
tallest_person
```

Out[43]:
```
618349.6106759505
```

```
As seen above from the use of the percent point function we see that
the expected hight for the tallest person to be 618,349.61 meters.
```

In [ ]:
```

```

Exercises 6-1

The distribution of income is famously skewed to the right. In this exercise, we'll measure how strong that skew is. The Current Population Survey (CPS) is a joint effort of the Bureau of Labor Statistics and the Census Bureau to study income and related variables. Data collected in 2013 is available from http://www.census.gov/hhes/www/cpstables/032013/hhinc/toc.htm. I downloaded `hinc06.xls`, which is an Excel spreadsheet with information about household income, and converted it to `hinc06.csv`, a CSV file you will find in the repository for this book. You will also find `hinc2.py`, which reads this file and transforms the data.

The dataset is in the form of a series of income ranges and the number of respondents who fell in each range. The lowest range includes respondents who reported annual household income “Under $5000.”Thehighestrangeincludesrespondentswhomade$“250,000 or more.”

To estimate mean and other statistics from these data, we have to make some assumptions about the lower and upper bounds, and how the values are distributed in each range. `hinc2.py` provides `InterpolateSample`, which shows one way to model this data. It takes a `DataFrame` with a column, `income`, that contains the upper bound of each range, and `freq`, which contains the number of respondents in each frame.

It also takes `log_upper`, which is an assumed upper bound on the highest range, expressed in `log10` dollars. The default value, `log_upper=6.0` represents the assumption that the largest income among the respondents is 10^6, or one million dollars.

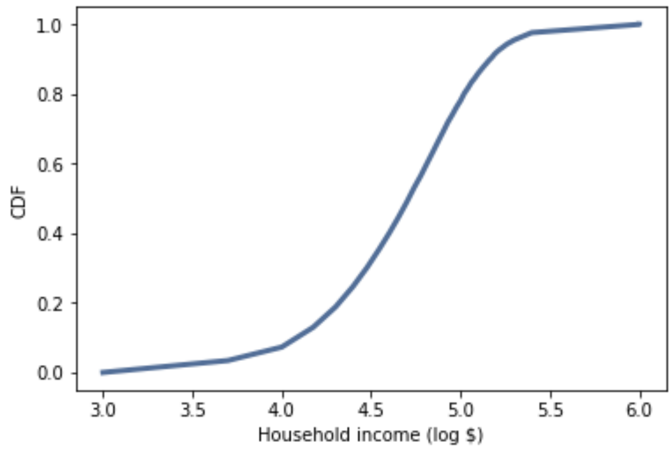
`InterpolateSample` generates a pseudo-sample; that is, a sample of household incomes that yields the same number of respondents in each range as the actual data. It assumes that incomes in each range are equally spaced on a `log10` scale.

In [29]:
```python
def InterpolateSample(df, log_upper=6.0):
    """Makes a sample of log10 household income.

    Assumes that log10 income is uniform in each range.

    df: DataFrame with columns income and freq
    log_upper: log10 of the assumed upper bound for the highest range

    returns: NumPy array of log10 household income
    """
    # compute the log10 of the upper bound for each range
    df['log_upper'] = np.log10(df.income)

    # get the lower bounds by shifting the upper bound and filling in
    # the first element
    df['log_lower'] = df.log_upper.shift(1)
    df.loc[0, 'log_lower'] = 3.0

    # plug in a value for the unknown upper bound of the highest range
    df.loc[41, 'log_upper'] = log_upper

    # use the freq column to generate the right number of values in
    # each range
    arrays = []
    for _, row in df.iterrows():
        vals = np.linspace(row.log_lower, row.log_upper, row.freq)
        arrays.append(vals)

    # collect the arrays into a single sample
    log_sample = np.concatenate(arrays)
    return log_sample
```

In [30]:
```python
import hinc
income_df = hinc.ReadData()
```

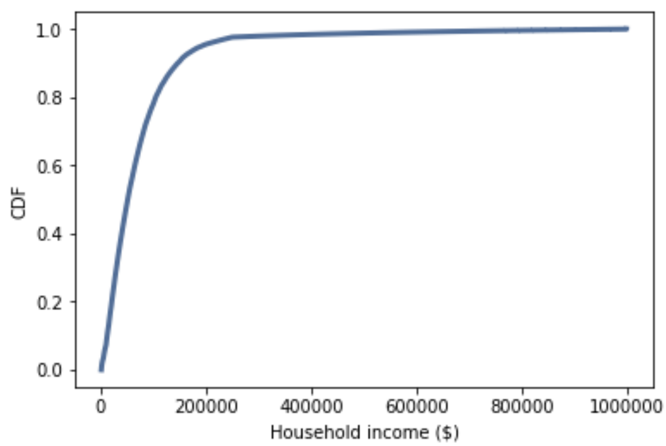
In [31]:
```python
log_sample = InterpolateSample(income_df, log_upper=6.0)
```

In [32]:
```
log_cdf = thinkstats2.Cdf(log_sample)
thinkplot.Cdf(log_cdf)
thinkplot.Config(xlabel='Household income (log $)',
                 ylabel='CDF')
```



In [33]:
```
sample = np.power(10, log_sample)
```

In [34]:
```
cdf = thinkstats2.Cdf(sample)
thinkplot.Cdf(cdf)
thinkplot.Config(xlabel='Household income ($)',
                 ylabel='CDF')
```



Compute the median, mean, skewness and Pearson's skewness of the resulting sample. What fraction of households report a taxable income below the mean? How do the results depend on the assumed upper bound?

In [37]:
```
# The median is calculated as seen below
sample_median = Median(sample)
# The sample median is:
sample_median
```

Out[37]: 163.05664126515046

In [38]:
```
# The mean is calculated as seen below
sample_mean = Mean(sample)
# The sample mean is:
sample_mean
```

Out[38]: 163.09591183695608

In [42]:
```
# The skewness is calculated as seen below
sample_skewness = Skewness(sample)
# The sample skewness is:
sample_skewness
```

Out[42]: 0.07481059952988171

In [43]:
```
# The Pearson's skewness is calculated as seen below
pearson_sample_skewness = PearsonMedianSkewness(sample)
# The sample skewness is:
pearson_sample_skewness
```

Out[43]: 0.016133002719960817

In [44]:
```
# Next I will determine the amount of the household that reports a
# taxable income that is below the mean by using the cdf and probability
household_probability = cdf.Prob(Mean(sample))
household_probability
```

Out[44]: 0.502

```
As seen above we see that 50.20% of households have reported that their
taxable income is below the mean.
```

All of this is based on an assumption that the highest income is one million dollars, but that's certainly not correct. What happens to the skew if the upper bound is 10 million?

Without better information about the top of this distribution, we can't say much about the skewness of the distribution.

In [ ]:
```
```