The Probability Density Function - Lab

Introduction

In this lab we will look at building visualizations known as **density plots** to estimate the probability density for a given set of data.

Objectives

You will be able to:

- Calculate the PDF from given dataset containing real valued random variables
- Plot density functions and comment on the shape of the plot
- Plot density functions using seaborn

Let's get started

We shall import all the required libraries for you for this lab.

In [2]:
```
# Import required libraries
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('ggplot')
import pandas as pd
```

Import the dataset 'weight-height.csv' as pandas dataframe . Calculate the mean and standard deviation for weights and heights for male and female individually.

Hint : Use your pandas dataframe subsetting skills like loc(), iloc() and groupby()

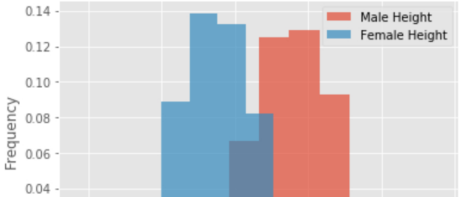
In [4]:
```
data = pd.read_csv('weight-height.csv')
male_df =  data.loc[data['Gender'] == 'Male']
female_df =  data.loc[data['Gender'] == 'Female']

print('Male Height mean:', male_df.Height.mean())
print('Male Height sd:', male_df.Height.std())

print('Male Weight mean:', male_df.Weight.mean())
print('Male Weight sd:' ,male_df.Weight.std())

print('Female Height mean:', female_df.Height.mean())
print('Female Height sd:' ,female_df.Height.std())

print('Female Weight mean:', female_df.Weight.mean())
print('Female Weight sd:' ,female_df.Weight.std())

# Male Height mean: 69.02634590621737
# Male Height sd: 2.8633622286606517
# Male Weight mean: 187.0206206581929
# Male Weight sd: 19.781154516763813
# Female Height mean: 63.708773603424916
# Female Height sd: 2.696284015765056
# Female Weight mean: 135.8600930074687
# Female Weight sd: 19.022467805319007
```

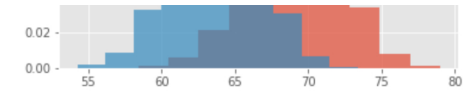
```
Male Height mean: 69.02634590621737
Male Height sd: 2.8633622286606517
Male Weight mean: 187.0206206581929
Male Weight sd: 19.781154516763813
Female Height mean: 63.708773603424916
Female Height sd: 2.696284015765056
Female Weight mean: 135.8600930074687
Female Weight sd: 19.022467805319007
```

Plot overlapping normalized histograms for male and female heights - use binsize = 10, set alpha level so that overlap can be visualized

In [5]:
```
binsize = 10
male_df.Height.plot.hist(bins = binsize, normed = True,  alpha = 0.7, label ="Male Height");
female_df.Height.plot.hist(bins = binsize, normed = True, alpha = 0.7, label = 'Female Height');
plt.legend()
```

Out[5]:
```
<matplotlib.legend.Legend at 0x10a5a38d0>
```



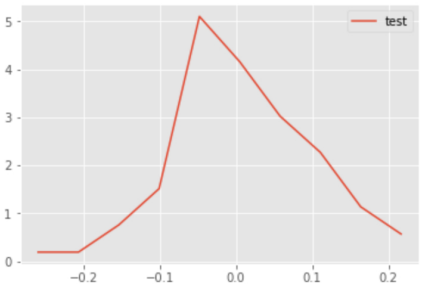


In [ ]:
```
# Record your observations - are these inline with your personal observations?

# Men tend to have higher values of heights in general than female
# The most common region for male and femle heights is between 65 - 67 inches (about 5 and a half feet)
# MAle heights have a slightly higher spread than female heights, hence the male heigh peak is slightly smaller than female heigh
# Both heights are normally distributed
```

Write a function density() that takes in a random variable and calculates the density function using `np.hist` and interpolation. The function should return two lists carrying x and y coordinates for plotting the density functio

In [7]:
```
def density(x):

    n, bins = np.histogram(x, 10, density=1)
    # Initialize numpy arrays with zeros to store interpolated values
    pdfx = np.zeros(n.size)
    pdfy = np.zeros(n.size)

    # Interpolate through histogram bins
    # identify middle point between two neighbouring bins, in terms of x and y coords
    for k in range(n.size):
        pdfx[k] = 0.5*(bins[k]+bins[k+1])
        pdfy[k] = n[k]

    # plot the calculated curve
    return pdfx, pdfy


# Generate test data and test the function
np.random.seed(5)
mu, sigma = 0, 0.1 # mean and standard deviation
s = np.random.normal(mu, sigma, 100)
x,y = density(s)
plt.plot(x,y, label = 'test')
plt.legend()
```

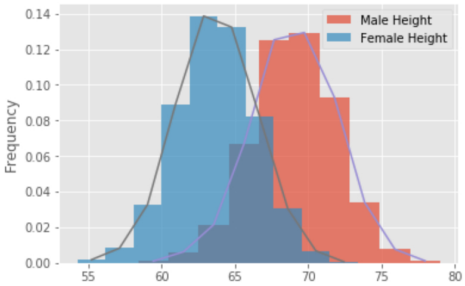
Out[7]:
```
<matplotlib.legend.Legend at 0x10acba668>
```



Add Overlapping density plots for male and female heights to the histograms plotted earlier

In [9]:
```
male_df.Height.plot.hist(bins = binsize, normed = True,  alpha = 0.7, label ="Male Height");
female_df.Height.plot.hist(bins = binsize, normed = True, alpha = 0.7, label = 'Female Height');
plt.legend()
x,y = density(male_df.Height)
plt.plot(x,y)
x,y = density(female_df.Height)
plt.plot(x,y)
```

Out[9]:
```
[<matplotlib.lines.Line2D at 0x10e25c9b0>]
```

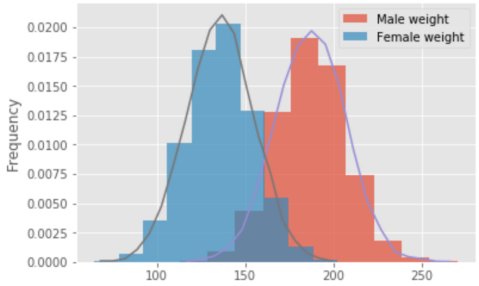


Repeat above exerice for male and female weights

In [70]:
```
male_df.Weight.plot.hist(bins = binsize, normed = True,  alpha = 0.7, label ="Male weight");
female_df.Weight.plot.hist(bins = binsize, normed = True, alpha = 0.7, label = 'Female weight');
plt.legend()
x,y = density_curve(male_df.Weight)
plt.plot(x,y)
x,y = density_curve(female_df.Weight)
```

```
plt.plot(x,y)
```

Out[70]: [<matplotlib.lines.Line2D at 0x115c5fa90>]



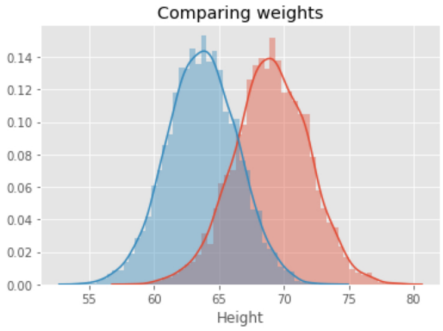
Write your observations in the cell below.

In [68]:
```
# Record your observations - are these inline with your personal observations?

# The patterns and overlap resemble highly with height distributions
# Man generally have more weight than women
# The common region for common weights is around 160 lbs.
# Male weight has slightly higher spread than female weight (i.e. more variation)
# Most females are around 130-140 lbs whereas most men are around 180 pounds.

#Takeaway

# Weight is more suitable to distinguish between males and females than height
```

Repeat Above experiments in seaborn and compare with your results.

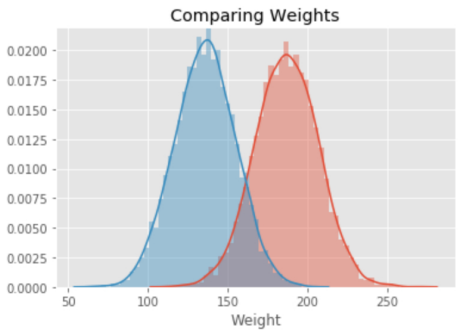
In [13]:
```
import seaborn as sns
sns.distplot(male_df.Height)
sns.distplot(female_df.Height)
plt.title('Comparing Heights')
```

Out[13]: Text(0.5,1,'Comparing weights')



In [14]:
```
import seaborn as sns
sns.distplot(male_df.Weight)
sns.distplot(female_df.Weight)
plt.title('Comparing Weights')
```

Out[14]: Text(0.5,1,'Comparing Weights')



In [ ]:
```
# Your comments on the two approaches here.
# are they similar ? what makes them different if they are ?
```

Summary

In this lesson we saw how to build the probability density curves visually for given datasets and compare on the distribution visually by looking at the spread , center and overlap between data elements. This is a useful EDA technique and can be used to answer some initial questions before embarking on a complex analytics journey.