learn-co-curriculum / probability-density-functions-lab Public

View license

1 star 185 forks

Star Watch

Code Issues Pull requests Actions Projects Security Insights

solution

This branch is 6 commits ahead, 15 commits behind master.

lmcm18 fixed learning objectives as well as some typos in code cells ... on Oct 25, 2019 9

View code

README.md

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:

- Plot and interpret density plots and comment on the shape of the plot
- Estimate probabilities for continuous variables by using interpolation

Let's get started

Let's import the necessary libraries for this lab.

```python
# Import required libraries
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('ggplot')
import pandas as pd
```

Import the data, and calculate the mean and the standard deviation

- Import the dataset 'weight-height.csv' as a pandas dataframe.
- Next, calculate the mean and standard deviation for weights and heights for men and women individually. You can simply use the pandas `.mean()` and `.std()` to do so.

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

```python
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
```

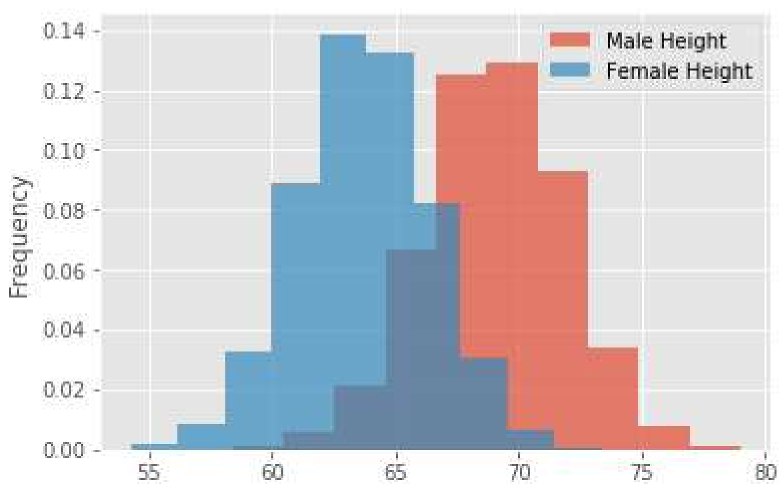
```
Male Height mean: 69.02634590621737
Male Height sd: 2.8633622286606517
Male Weight mean: 187.0206206581929
```

```
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 histograms (with densities on the y-axis) for male and female heights

- Make sure to create overlapping plots
- Use binsize = 10, set alpha level so that overlap can be visualized

```
binsize = 10
male_df.Height.plot.hist(bins = binsize, density = True,  alpha = 0.7, label ="Male
female_df.Height.plot.hist(bins = binsize, density = True, alpha = 0.7, label = 'Fem
plt.legend()
plt.show()
```

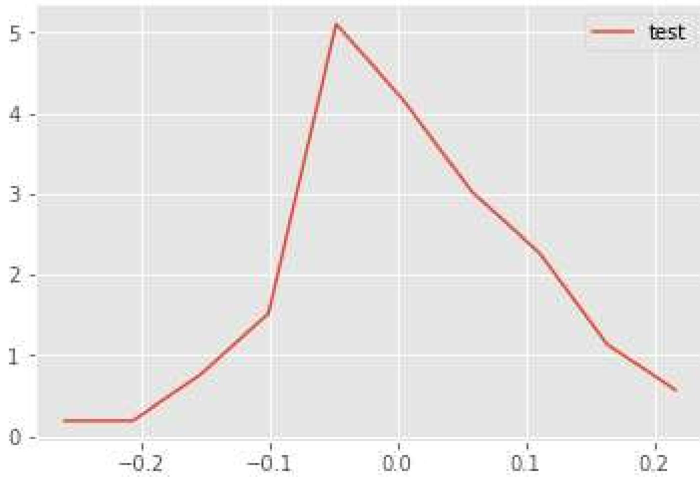



```
# Record your observations - are these inline with your personal observations?

# Men tend to have higher values of heights in general than female
# The most common region for male and female heights is between 65 - 67 inches (abou
# Male heights have a slightly higher spread than female heights, hence the male hei
# Both heights are normally distributed
```

Create a density function using interpolation

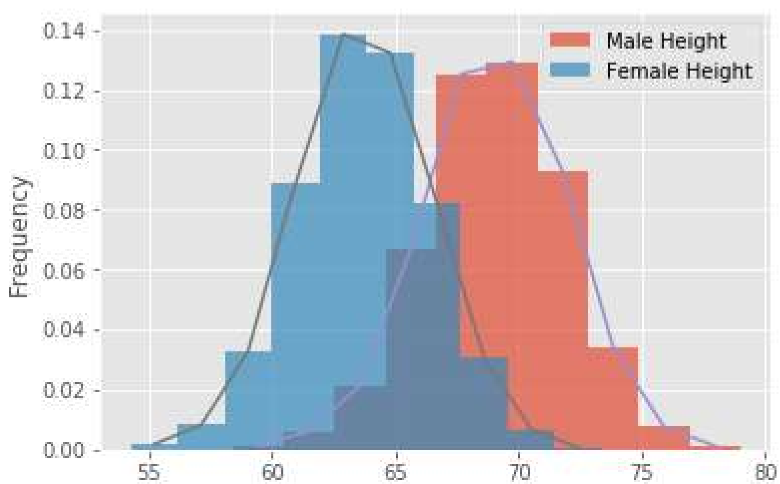
- Write a density function density() that uses interpolation and takes in a random variable
- Use `np.histogram()`
- The function should return two lists carrying x and y coordinates for plotting the density function

```
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 coord
    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()
plt.show()
```



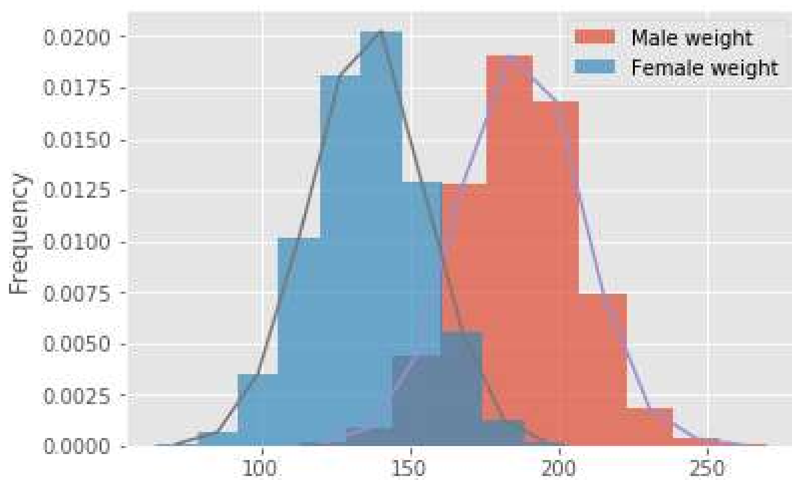
Add overlapping density plots to the histograms plotted earlier

```
male_df.Height.plot.hist(bins = binsize, normed = True,  alpha = 0.7, label ="Male H
female_df.Height.plot.hist(bins = binsize, normed = True, alpha = 0.7, label = 'Fema
plt.legend()
x,y = density(male_df.Height)
plt.plot(x,y)
x,y = density(female_df.Height)
plt.plot(x,y)
plt.show()
```



Repeat the above exercise for male and female weights

```
male_df.Weight.plot.hist(bins = binsize, normed = True,  alpha = 0.7, label ="Male w
female_df.Weight.plot.hist(bins = binsize, normed = True, alpha = 0.7, label = 'Fema
plt.legend()
x,y = density(male_df.Weight)
plt.plot(x,y)
x,y = density(female_df.Weight)
plt.plot(x,y)
plt.show()
```

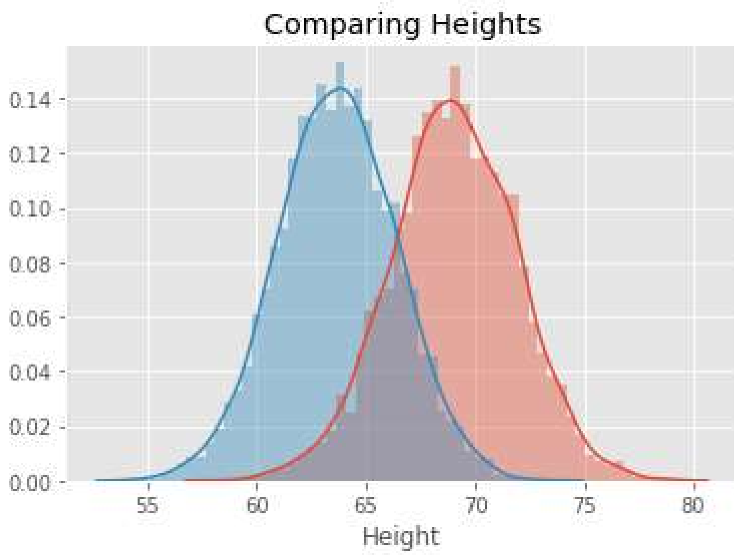


Write your observations in the cell below

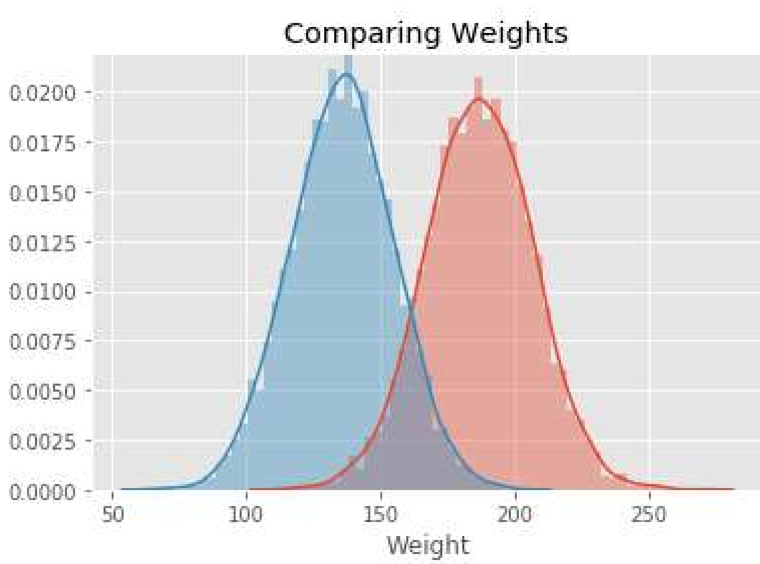
```
# Record your observations - are these inline with your personal observations?

# The patterns and overlap are highly similar to what we see with height distributic
# Men generally are heavier 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 the above experiments in seaborn and compare with your results

```
import seaborn as sns
sns.distplot(male_df.Height)
sns.distplot(female_df.Height)
plt.title('Comparing Heights')
plt.show()
```




```
import seaborn as sns
sns.distplot(male_df.Weight)
sns.distplot(female_df.Weight)
plt.title('Comparing Weights')
plt.show()
```



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

Summary

In this lesson, you learned how to build the probability density curves visually for a given dataset and compare the distributions visually by looking at the spread, center, and overlap. This is a useful EDA technique and can be used to answer some initial questions before embarking on a complex analytics journey.

Releases

No releases published

Packages

No packages published

Contributors 5

Languages

- Jupyter Notebook 100.0%