Visualizing_Data

May 19, 2022

1 Data Visualization

Estimated time needed: **30** minutes

In this lab, you will learn how to visualize and interpret data

1.1 Objectives

- Import Libraries
- Lab Exercises
  - Identifying duplicates
  - Plotting Scatterplots
  - Plotting Boxplots

1.2 Import Libraries

All Libraries required for this lab are listed below. The libraries pre-installed on Skills Network Labs are commented. If you run this notebook in a different environment, e.g. your desktop, you may need to uncomment and install certain libraries.

[ ]:
```
#install specific version of libraries used in lab
#! mamba install pandas==1.3.3
#! mamba install numpy=1.21.2
#! mamba install scipy=1.7.1-y
#!  mamba install seaborn=0.9.0-y
#!  mamba install matplotlib=3.4.3-y
```

Import the libraries we need for the lab

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Read in the csv file from the url using the request library

[2]:
```

```

```
ratings_url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.
 ↪cloud/IBMDeveloperSkillsNetwork-ST0151EN-SkillsNetwork/labs/teachingratings.
 ↪csv'
ratings_df = pd.read_csv(ratings_url)
```

1.3 Lab Exercises

1.3.1 Identify all duplicate cases using prof. Using all observations, find the average and standard deviation for age. Repeat the analysis by first filtering the data set to include one observation for each instructor with a total number of observations restricted to 94.

Identify all duplicate cases using prof variable - find the unique values of the prof variables

[3]:
```
ratings_df.prof.unique()
```

[3]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 23, 24, 25, 26, 27, 28, 29, 31, 32, 33, 34, 35, 36,
       37, 38, 39, 41, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 53, 54, 55,
       56, 57, 58, 59, 60, 63, 64, 65, 66, 67, 68, 70, 71, 72, 73, 74, 75,
       76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92,
       93, 94, 22, 30, 40, 47, 61, 62, 69])
```

Print out the number of unique values in the prof variable

[4]:
```
ratings_df.prof.nunique()
```

[4]:
```
94
```

Using all observations, Find the average and standard deviation for age

[5]:
```
ratings_df['age'].mean()
```

[5]:
```
48.365010799136066
```

[6]:
```
ratings_df['age'].std()
```

[6]:
```
9.802742037864821
```

Repeat the analysis by first filtering the data set to include one observation for each instructor with a total number of observations restricted to 94.

first we drop duplicates using prof as a subset and assign it a new dataframe name called no_duplicates_ratings_df

[7]:
```
no_duplicates_ratings_df = ratings_df.drop_duplicates(subset =['prof'])
no_duplicates_ratings_df.head()
```

[7]:
```
    minority  age  gender credits    beauty  eval division native tenure  \
0        yes   36  female    more  0.289916   4.3    upper    yes    yes
4         no   59    male    more -0.737732   4.5    upper    yes    yes
7         no   51    male    more -0.571984   3.7    upper    yes    yes
9         no   40  female    more -0.677963   4.3    upper    yes    yes
17        no   31  female    more  1.509794   4.4    upper    yes    yes

     students  allstudents  prof  PrimaryLast  vismin  female  single_credit  \
0          24           43     1            0       1       1              0
4          17           20     2            0       0       0              0
7          55           55     3            0       0       0              0
9          40           46     4            0       0       1              0
17         42           48     5            0       0       1              0

     upper_division  English_speaker  tenured_prof
0                 1                1             1
4                 1                1             1
7                 1                1             1
9                 1                1             1
17                1                1             1
```

Use the new dataset to get the mean of age

[8]:
```
no_duplicates_ratings_df['age'].mean()
```

[8]:
```
47.5531914893617
```

[9]:
```
no_duplicates_ratings_df['age'].std()
```

[9]:
```
10.25651329515495
```

1.3.2 Using a bar chart, demonstrate if instructors teaching lower-division courses receive higher average teaching evaluations.

[10]:
```
ratings_df.head()
```

[10]:
```
   minority  age  gender credits    beauty  eval division native tenure  \
0       yes   36  female    more  0.289916   4.3    upper    yes    yes
1       yes   36  female    more  0.289916   3.7    upper    yes    yes
2       yes   36  female    more  0.289916   3.6    upper    yes    yes
3       yes   36  female    more  0.289916   4.4    upper    yes    yes
4        no   59    male    more -0.737732   4.5    upper    yes    yes

    students  allstudents  prof  PrimaryLast  vismin  female  single_credit  \
0         24           43     1            0       1       1              0
1         86          125     1            0       1       1              0
2         76          125     1            0       1       1              0
```

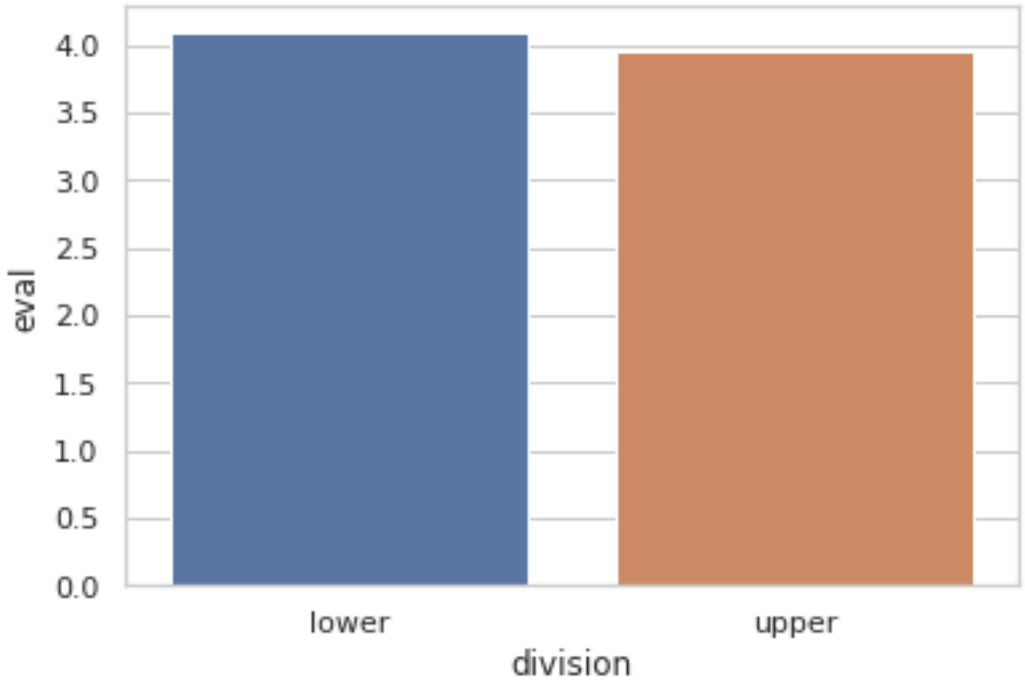
```
3        77          123     1             1       1       1              0
4        17           20     2             0       0       0              0

   upper_division  English_speaker  tenured_prof
0               1                1             1
1               1                1             1
2               1                1             1
3               1                1             1
4               1                1             1
```

Find the average teaching evaluation in both groups of upper and lower-division

[11]:
```
division_eval = ratings_df.groupby('division')[['eval']].mean().reset_index()
```

Plot the barplot using the seaborn library

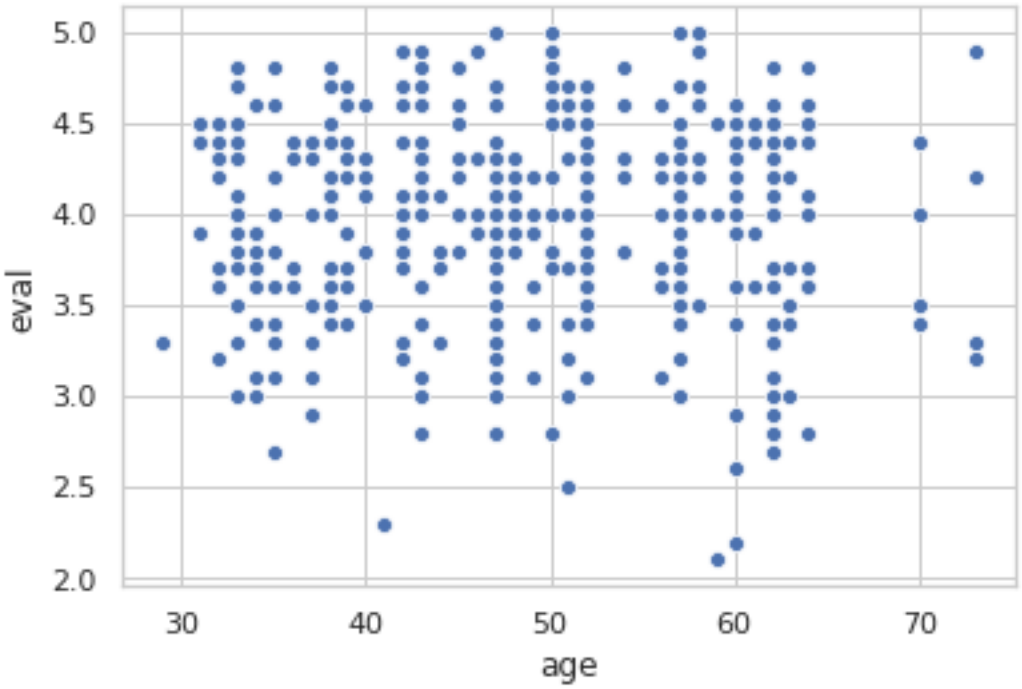
[12]:
```
sns.set(style="whitegrid")
ax = sns.barplot(x="division", y="eval", data=division_eval)
```



1.3.3 Plot the relationship between age and teaching evaluation scores.

Create a scatterplot with the scatterplot function in the seaborn library

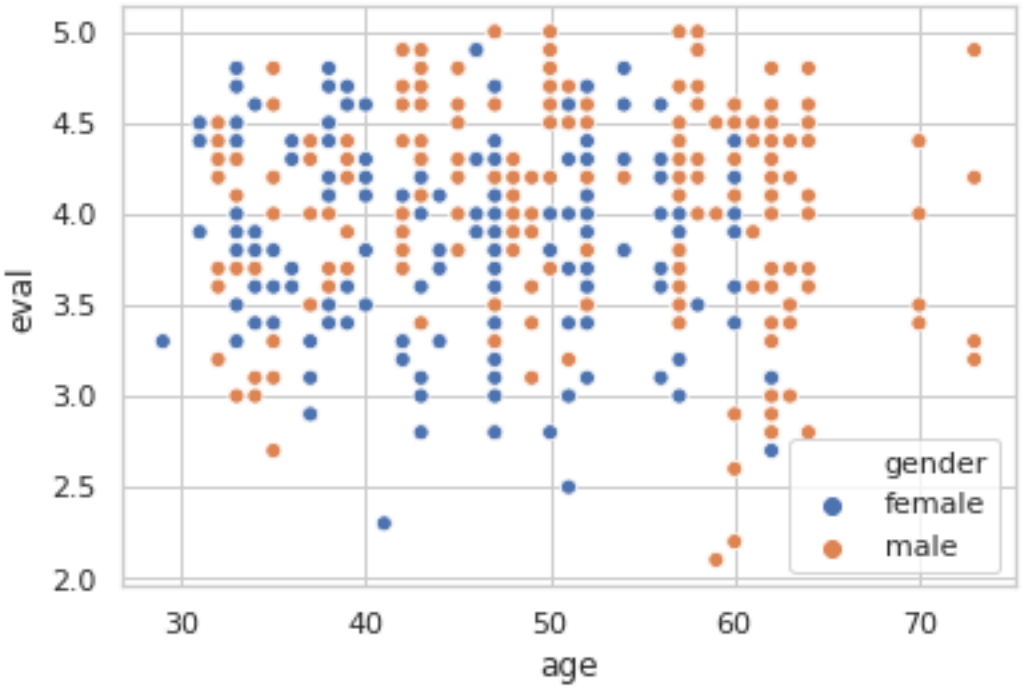
[13]:
```
ax = sns.scatterplot(x='age', y='eval', data=ratings_df)
```



1.3.4 Using gender-differentiated scatter plots, plot the relationship between age and teaching evaluation scores.

Create a scatterplot with the scatterplot function in the seaborn library this time add the hue argument

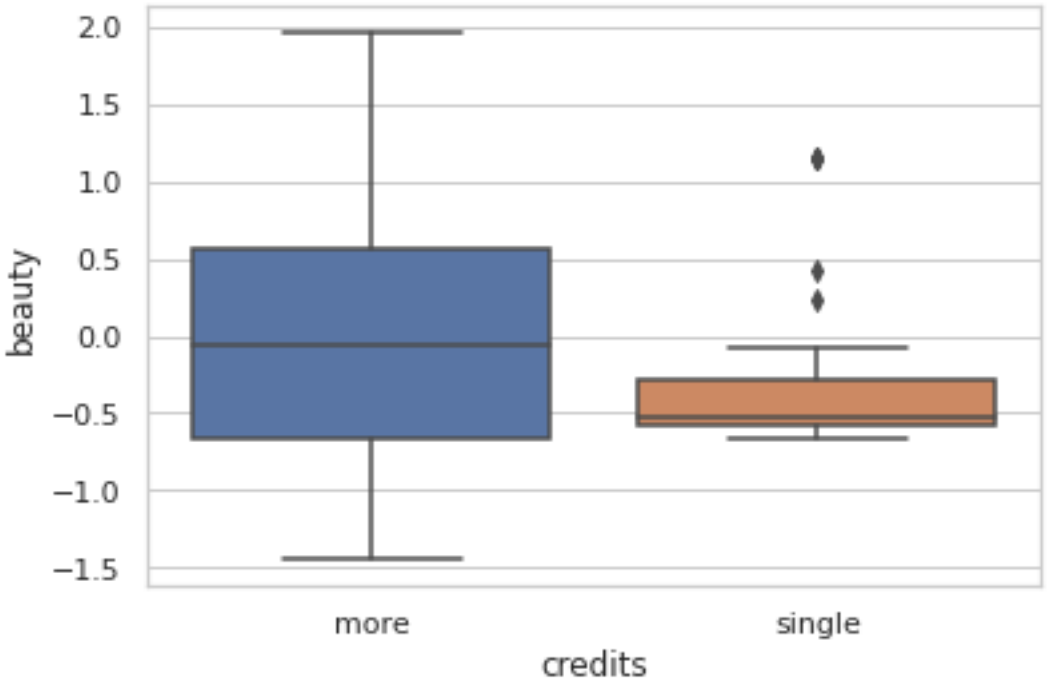
[14]:
```
ax = sns.scatterplot(x='age', y='eval', hue='gender',
                     data=ratings_df)
```



1.3.5 Create a box plot for beauty scores differentiated by credits.

We use the boxplot() function from the seaborn library

[15]:
```
ax = sns.boxplot(x='credits', y='beauty', data=ratings_df)
```

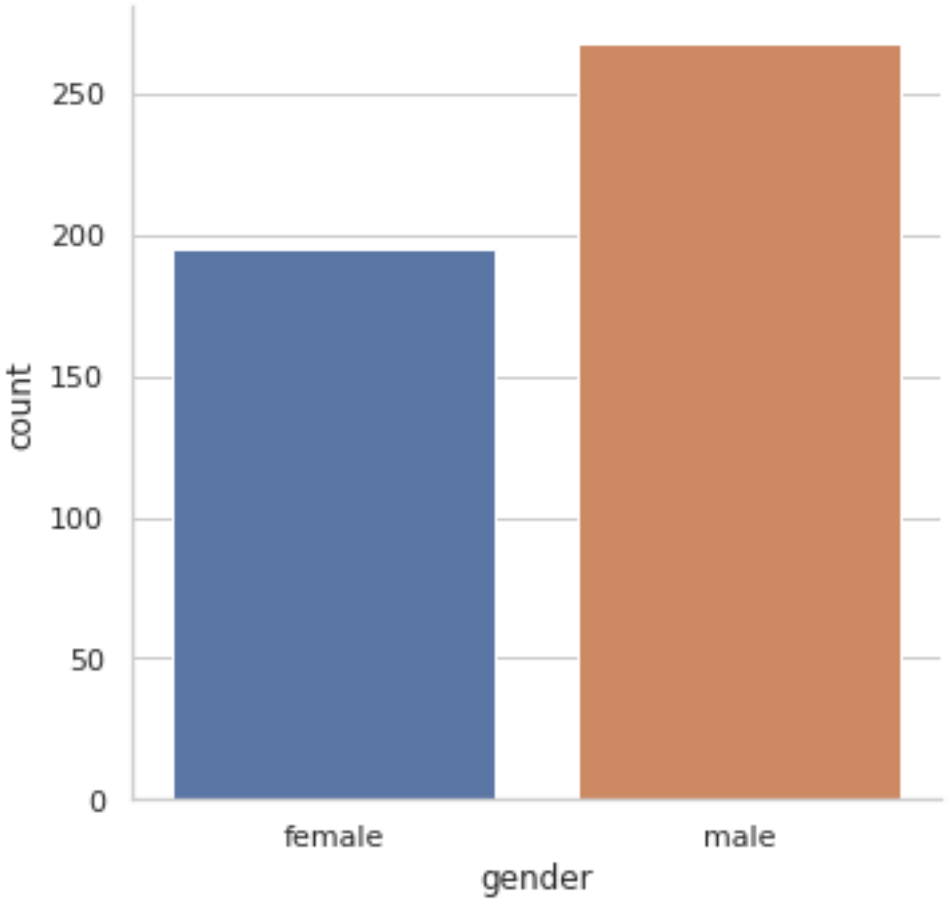


1.3.6 What is the number of courses taught by gender?

We use the catplot() function from the seaborn library

[16]:
```
sns.catplot(x='gender', kind='count', data=ratings_df)
```

[16]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdb840439d0>
```

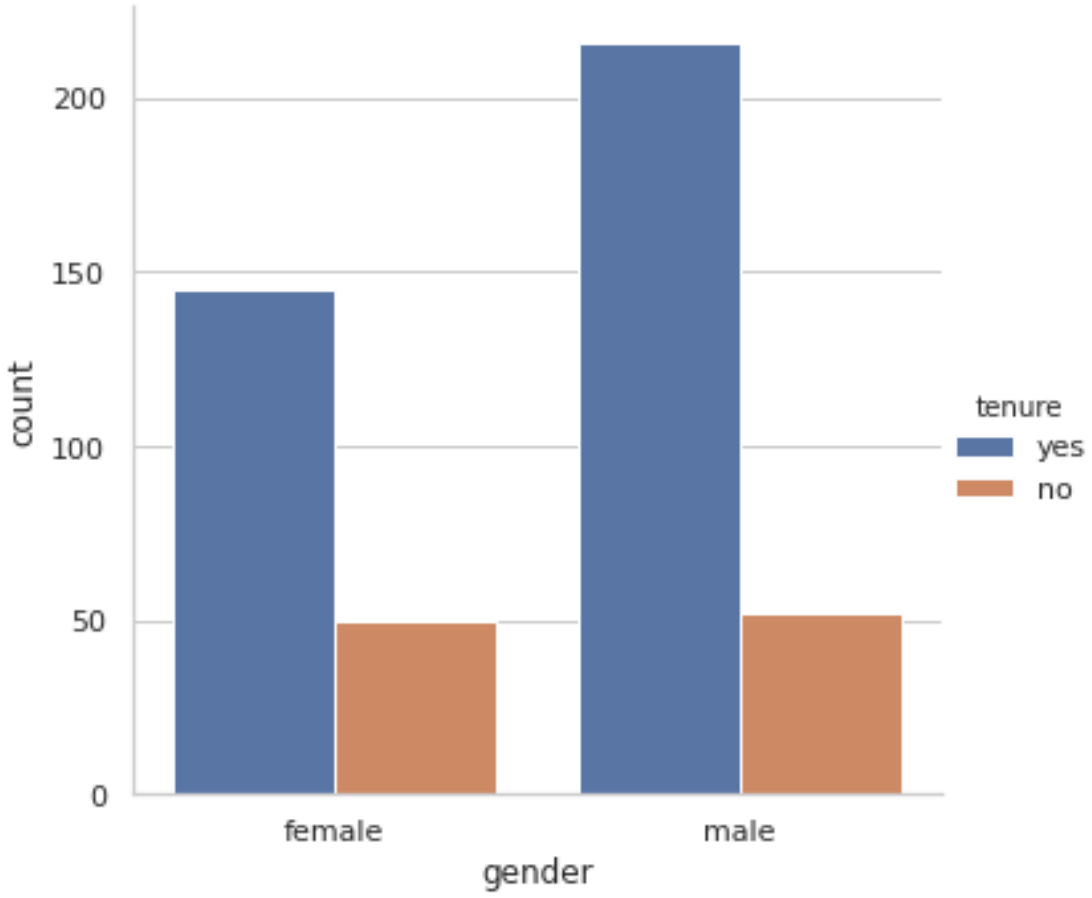


1.3.7 Create a group histogram of taught by gender and tenure

We will add the hue = Tenure argument

[17]:
```
sns.catplot(x='gender', hue = 'tenure', kind='count', data=ratings_df)
```

[17]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdb7ff991d0>
```

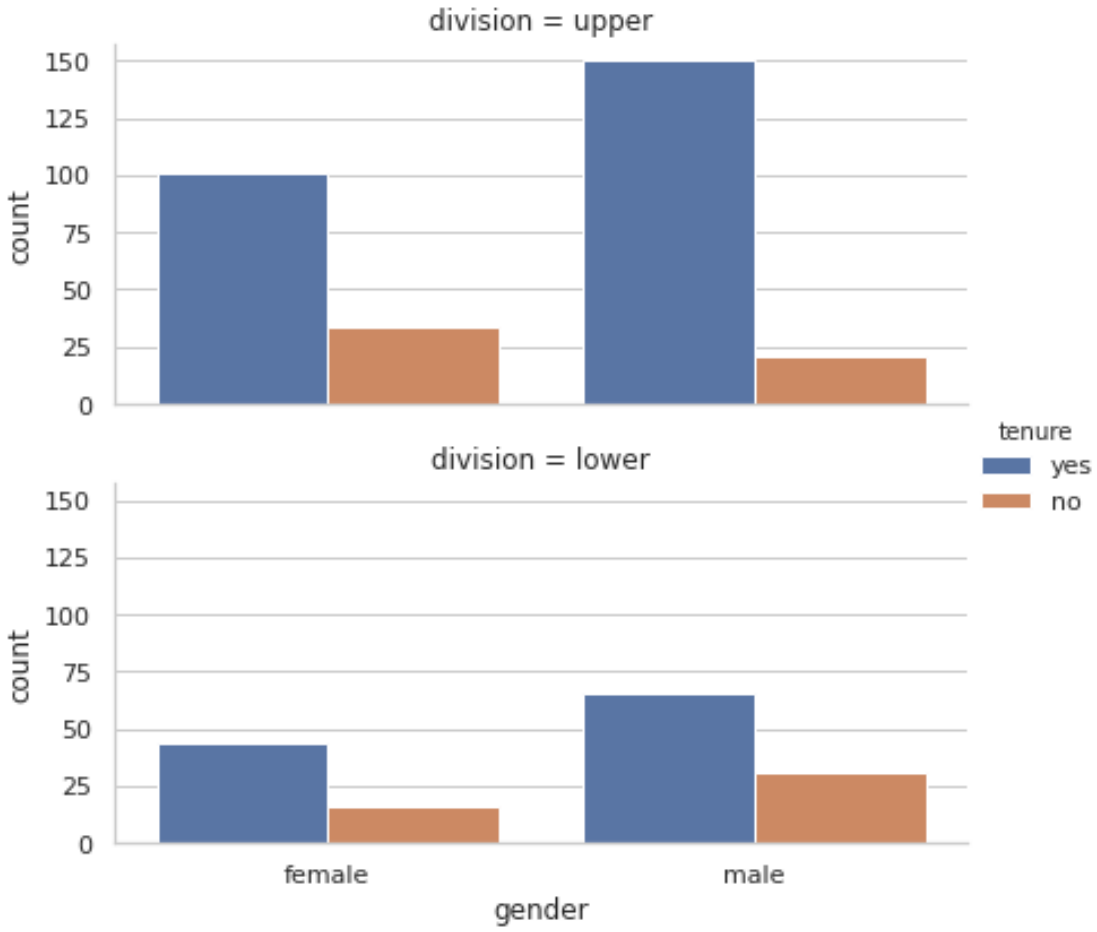


1.3.8 Add division as another factor to the above histogram

We add another argument named row and use the division variable as the row

[18]:
```
sns.catplot(x='gender', hue = 'tenure', row = 'division',
            kind='count', data=ratings_df,
            height = 3, aspect = 2)
```

[18]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdb7fe88290>
```

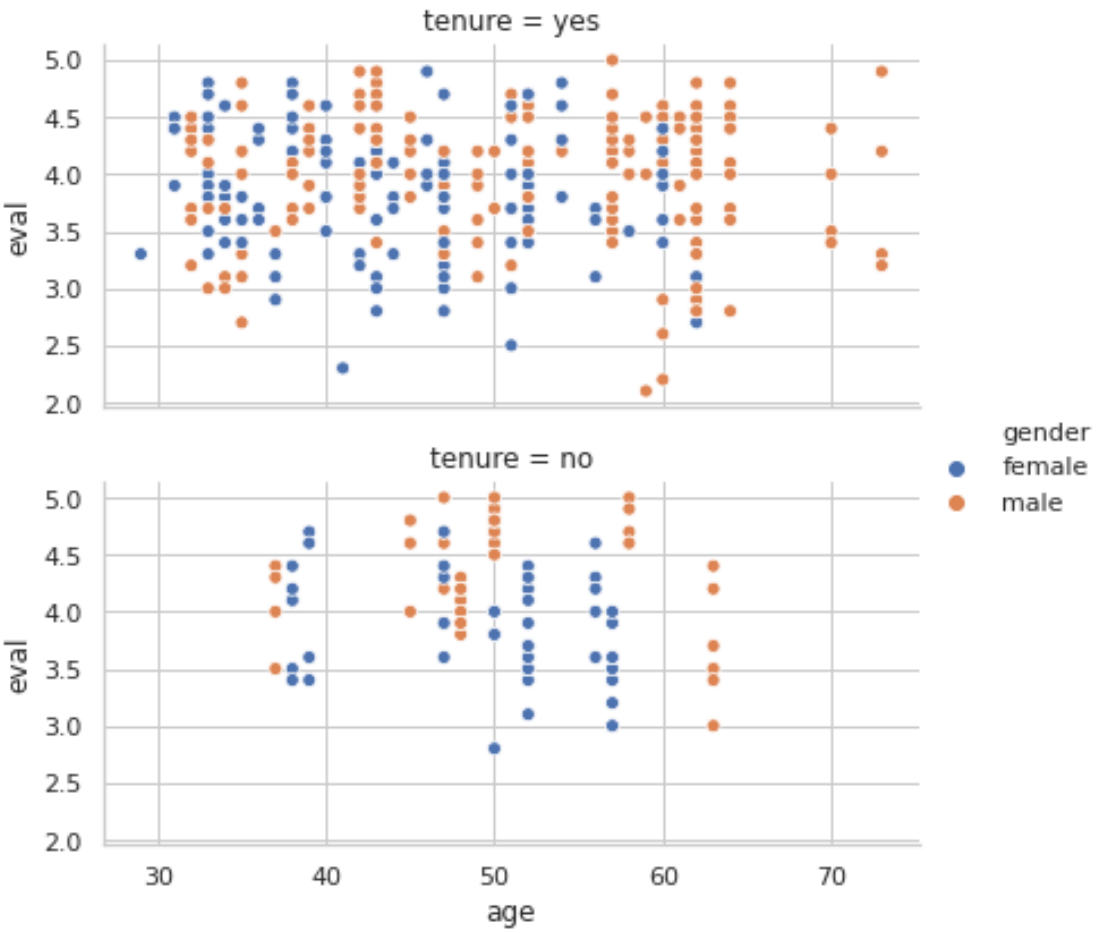


1.3.9 Create a scatterplot of age and evaluation scores, differentiated by gender and tenure

Use the relplot() function for complex scatter plots

[19]:
```
sns.relplot(x="age", y="eval", hue="gender",
            row="tenure",
            data=ratings_df, height = 3, aspect = 2)
```

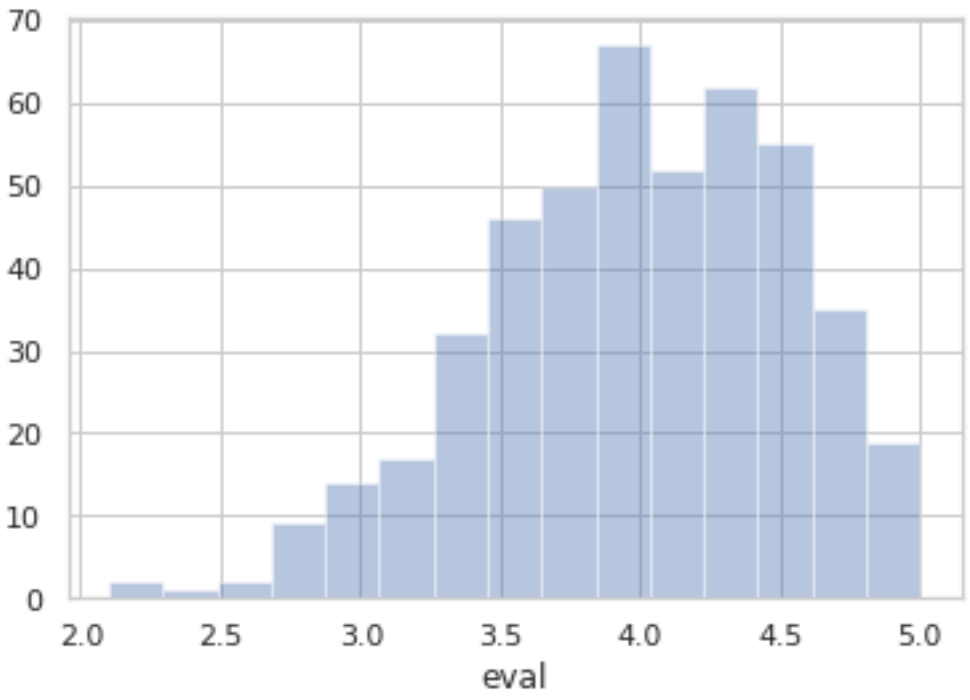
[19]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdb7fd9c790>
```



1.3.10 Create a distribution plot of teaching evaluation scores

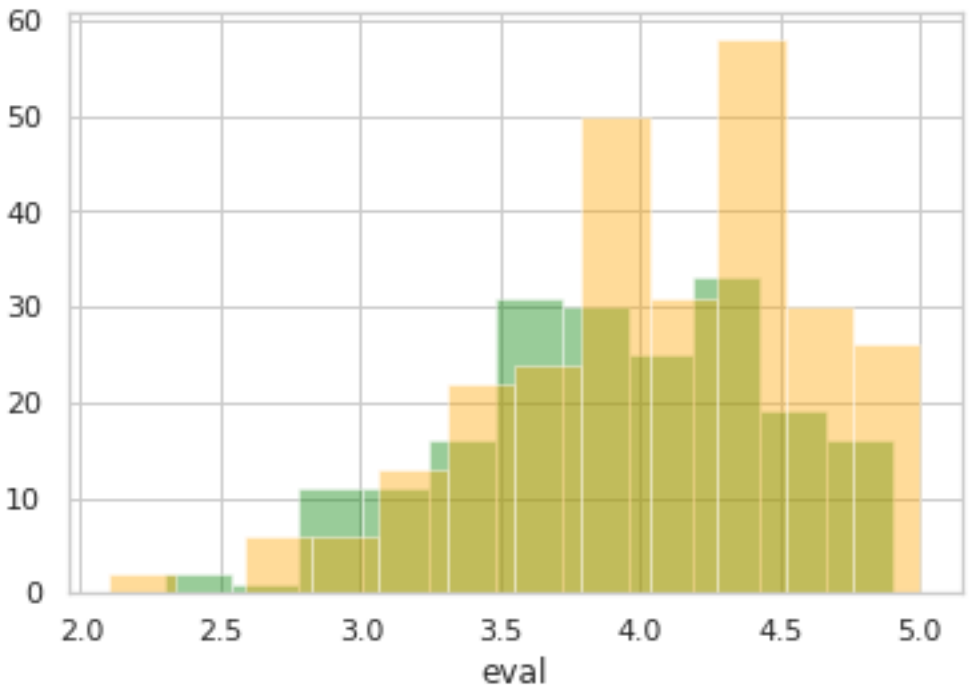
We use the distplot() function from the seaborn library, set kde = false because we don'e need the curve

[20]:
```
ax = sns.distplot(ratings_df['eval'], kde = False)
```



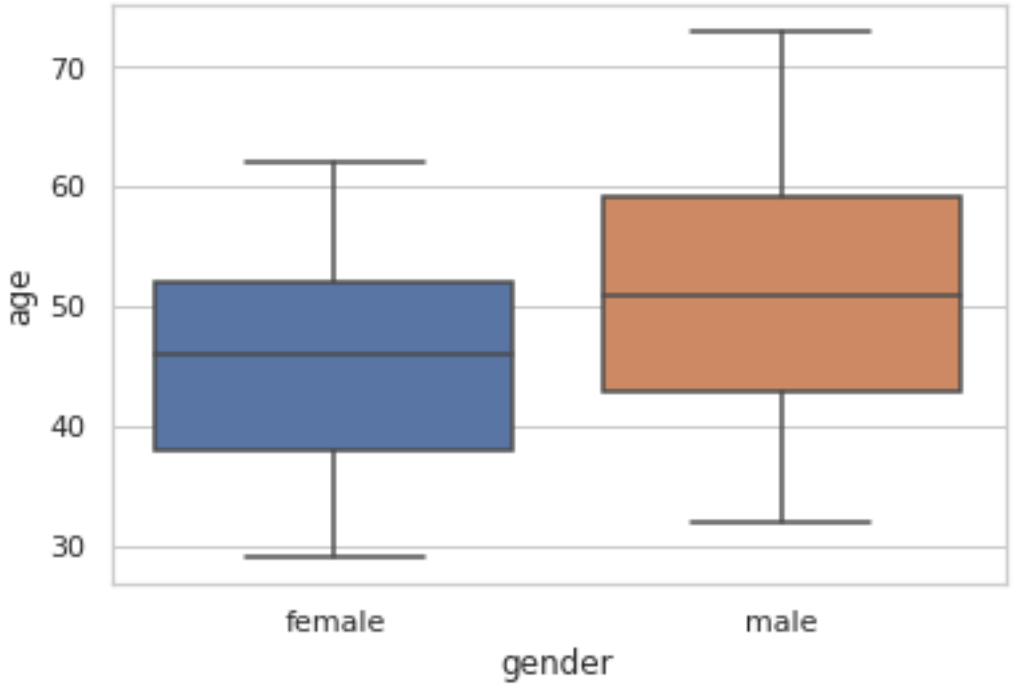
1.3.11 Create a distribution plot of teaching evaluation score with gender as a factor

[21]:
```
## use the distplot function from the seaborn library
sns.distplot(ratings_df[ratings_df['gender'] == 'female']['eval'],␣
 ↪color='green', kde=False)
sns.distplot(ratings_df[ratings_df['gender'] == 'male']['eval'],␣
 ↪color="orange", kde=False)
plt.show()
```



1.3.12 Create a box plot - age of the instructor by gender

[22]:
```
ax = sns.boxplot(x="gender", y="age", data=ratings_df)
```



1.3.13 Compare age along with tenure and gender

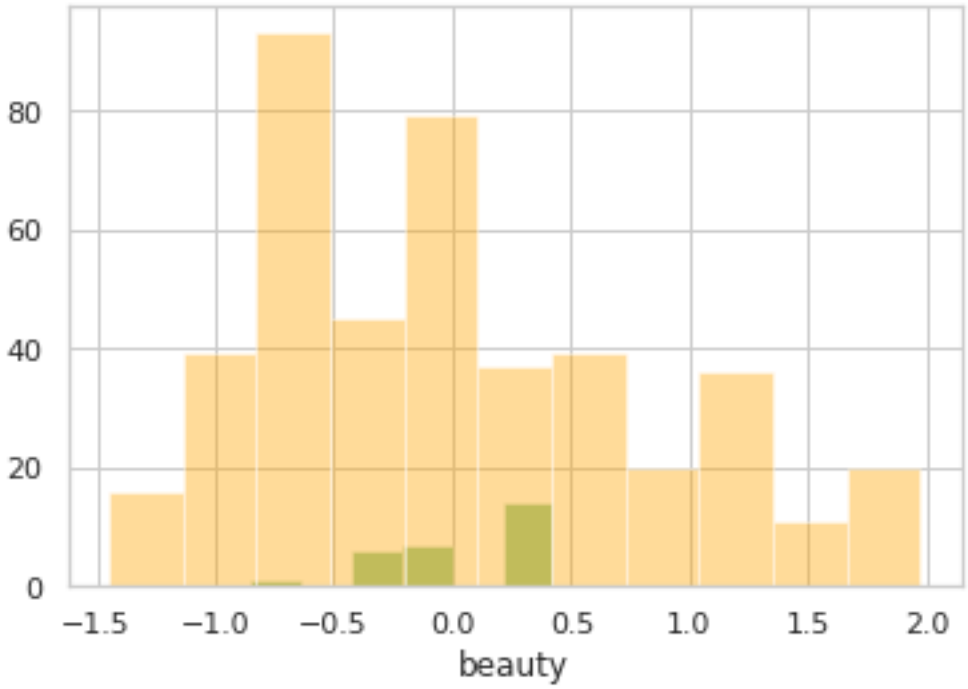
[ ]:
```
ax = sns.boxplot(x="tenure", y="age", hue="gender",
                 data=ratings_df)
```

1.4 Practice Questions

1.4.1 Question 1: Create a distribution plot of beauty scores with Native English speaker as a factor

- Make the color of the native English speakers plot - orange and non - native English speakers - blue

[25]:
```
## insert code
sns.distplot(ratings_df[ratings_df['native'] == 'no']['beauty'], color="green",␣
 ↪kde=False)
sns.distplot(ratings_df[ratings_df['native'] == 'yes']['beauty'],␣
 ↪color="orange", kde=False)
plt.show()
```

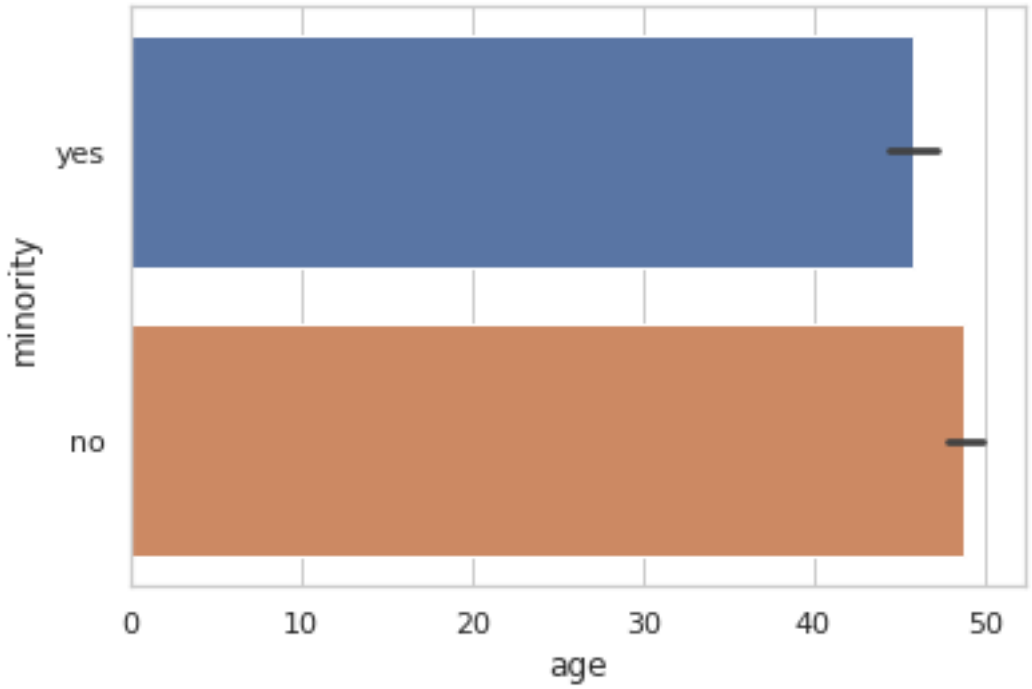


Double-click **here** for the solution.

1.4.2 Question 2: Create a Horizontal box plot of the age of the instructors by visible minority

[30]:
```
## insert code
sns.barplot(x='age', y='minority', data=ratings_df)
```

[30]:
```
<AxesSubplot:xlabel='age', ylabel='minority'>
```



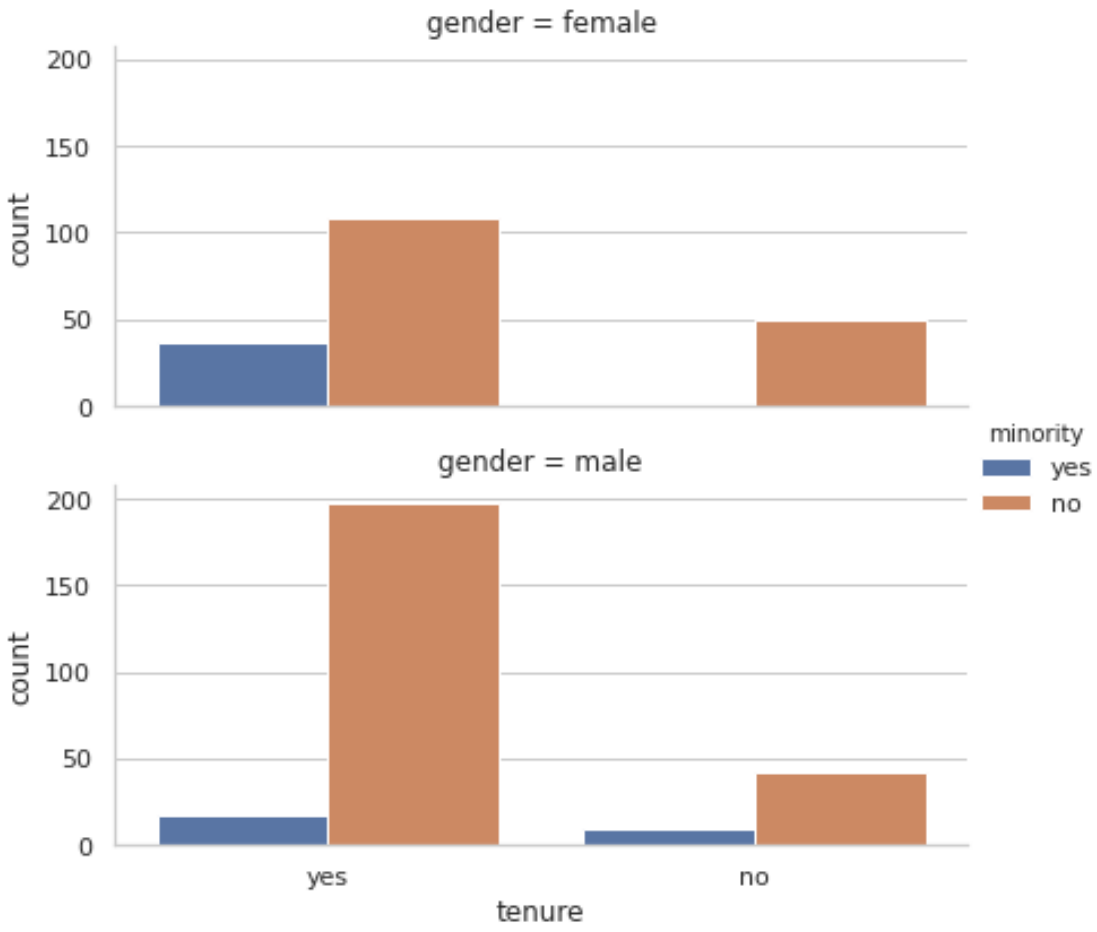
Double-click **here** for a hint.

Double-click **here** for the solution.

1.4.3 Question 3: Create a group histogram of tenure by minority and add the gender factor

[31]:
```
## insert code
sns.catplot(x='tenure', hue = 'minority', row = 'gender',
            kind='count', data=ratings_df,
            height = 3, aspect = 2)
```

[31]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdb7e9c37d0>
```

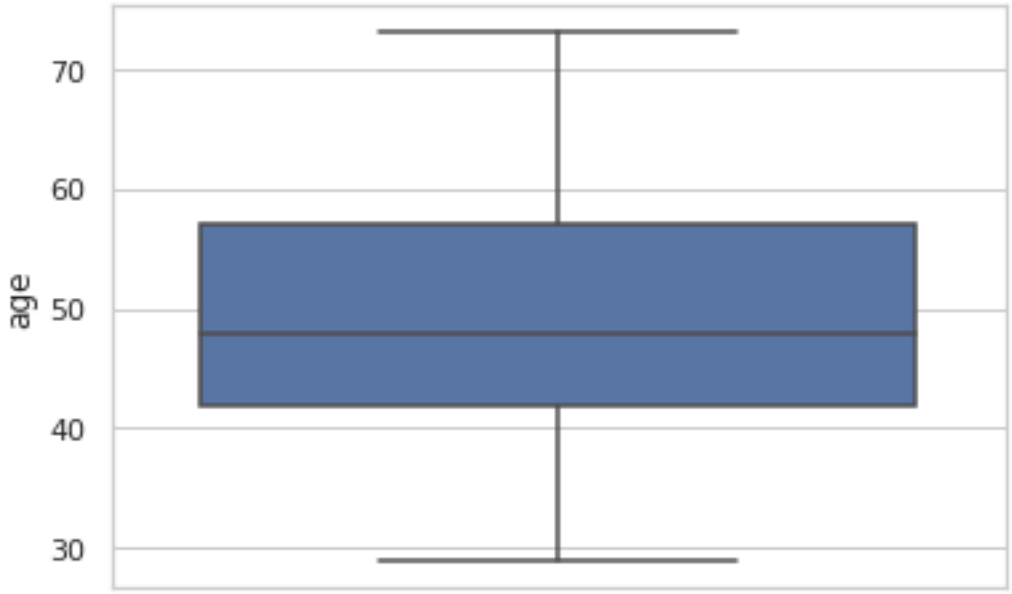


Double-click **here** for the solution.

1.4.4 Question 4: Create a boxplot of the age variable

[34]:
```
## insert code
sns.boxplot(y='age', data=ratings_df)
```

[34]:
```
<AxesSubplot:ylabel='age'>
```



Double-click **here** for the solution.

1.5 Authors

Aije Egwaikhide is a Data Scientist at IBM who holds a degree in Economics and Statistics from the University of Manitoba and a Post-grad in Business Analytics from St. Lawrence College, Kingston. She is a current employee of IBM where she started as a Junior Data Scientist at the Global Business Services (GBS) in 2018. Her main role was making meaning out of data for their Oil and Gas clients through basic statistics and advanced Machine Learning algorithms. The highlight of her time in GBS was creating a customized end-to-end Machine learning and Statistics solution on optimizing operations in the Oil and Gas wells. She moved to the Cognitive Systems Group as a Senior Data Scientist where she will be providing the team with actionable insights using Data Science techniques and further improve processes through building machine learning solutions. She recently joined the IBM Developer Skills Network group where she brings her real-world experience to the courses she creates.

1.6 Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-08-14	0.1	Aije Egwaikhide	Created the initial version of the lab