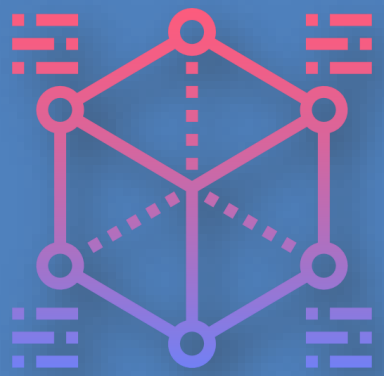
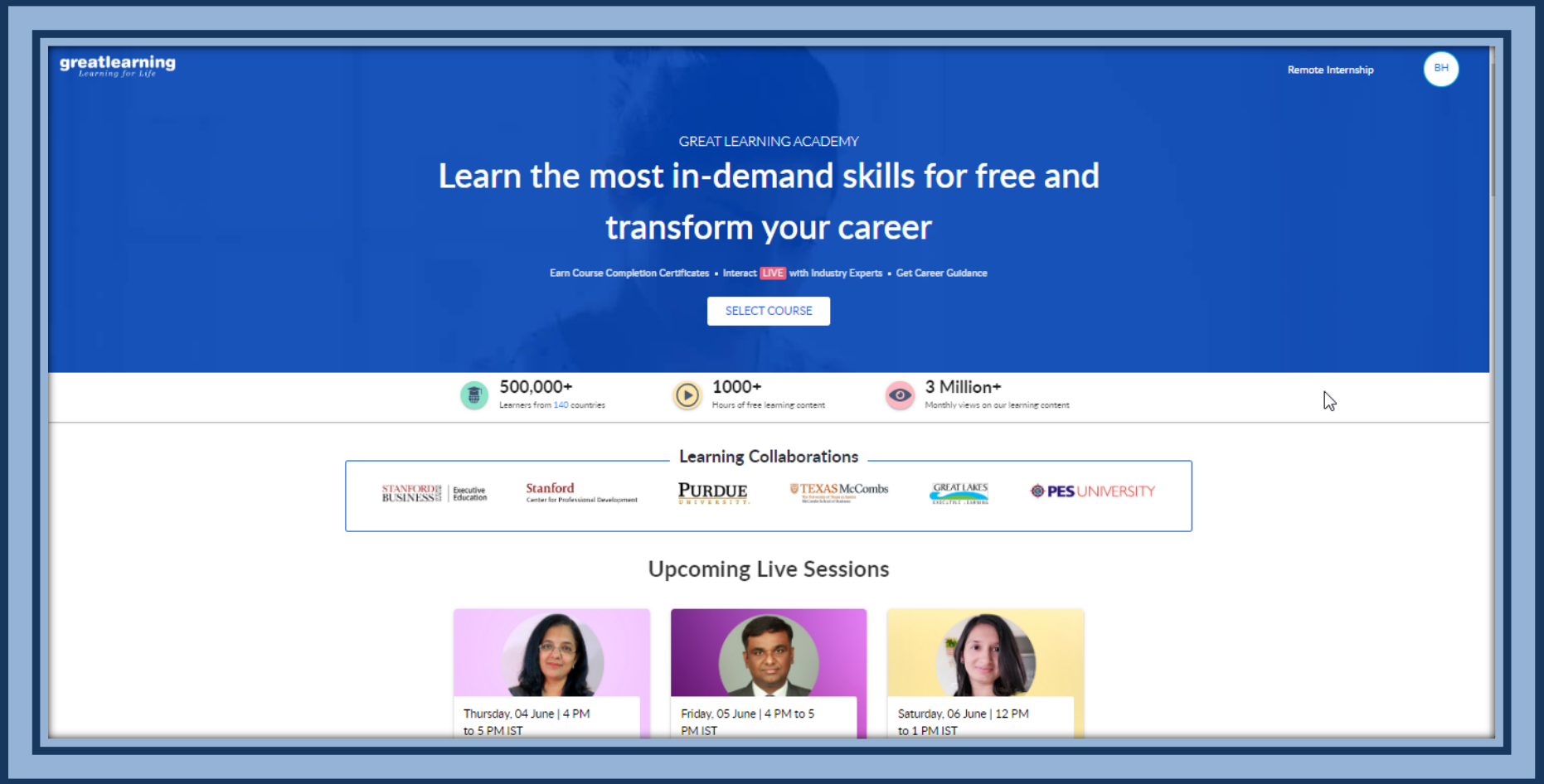
greatlearning
Learning for Life

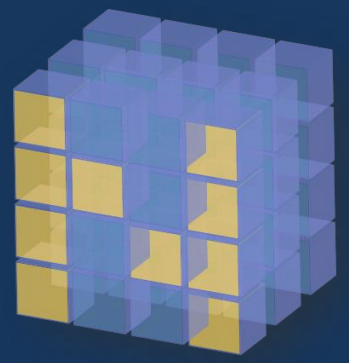




Python NumPy

NumPy stands for Numerical python and is the core library for numeric and scientific computing

It consists of multi-dimensional array objects and a collection of routines for processing those arrays



NumPy

Creating NumPy Array

greatlearning
Learning for Life

Single-dimensional Array

```
In [3]: import numpy as np

        n1=np.array([10,20,30,40])
        n1

Out[3]: array([10, 20, 30, 40])
```

Multi-dimensional Array

```
In [6]: import numpy as np

        n2=np.array([[10,20,30,40],[40,30,20,10]])
        n2

Out[6]: array([[10, 20, 30, 40],
               [40, 30, 20, 10]])
```

Initializing NumPy Array

greatlearning
Learning for Life

Initializing NumPy array with zeros

```
In [30]: import numpy as np
         n1=np.zeros((1,2))
         n1

Out[30]: array([[0., 0.]])
```

```
In [31]: import numpy as np
         n1=np.zeros((5,5))
         n1

Out[31]: array([[0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.]])
```

Initializing NumPy Array

greatlearning
Learning for Life

Initializing NumPy array with same number

```
In [38]: import numpy as np
         n1=np.full((2,2),10)
         n1

Out[38]: array([[10, 10],
                [10, 10]])
```

Initializing NumPy Array

Initializing NumPy array within a range

```
In [34]: import numpy as np
         n1=np.arange(10,20)
         n1

Out[34]: array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19])
```

```
In [35]: import numpy as np
         n1=np.arange(10,50,5)
         n1

Out[35]: array([10, 15, 20, 25, 30, 35, 40, 45])
```

Python Pandas

Pandas stands for Panel Data and is the core library for data manipulation and data analysis

It consists of single and multi-dimensional data-structures for data-manipulation

Pandas

Pandas Data-Structures

greatlearning
Learning for Life

Single-dimensional

Series Object

Multi-dimensional

Data-frame

Pandas Series Object

Series Object is one-dimensional labeled array

```
In [2]: import pandas as pd
        s1=pd.Series([1,2,3,4,5])
        s1

Out[2]: 0    1
        1    2
        2    3
        3    4
        4    5
        dtype: int64
```

```
In [4]: type(s1)

Out[4]: pandas.core.series.Series
```

Changing Index

```
In [2]: import pandas as pd
        s1=pd.Series([1,2,3,4,5])
        s1

Out[2]: 0    1
        1    2
        2    3
        3    4
        4    5
        dtype: int64
```

```
In [5]: import pandas as pd
        s1=pd.Series([1,2,3,4,5],index=['a','b','c','d','e'])
        s1

Out[5]: a    1
        b    2
        c    3
        d    4
        e    5
        dtype: int64
```

Extracting Individual Elements

greatlearning
Learning for Life

Extracting a single element

```
In [15]: s1 = pd.Series([1,2,3,4,5,6,7,8,9])
         s1[3]

Out[15]: 4
```

Extracting elements from back

```
In [17]: s1 = pd.Series([1,2,3,4,5,6,7,8,9])
         s1[-3:]

Out[17]: 6    7
         7    8
         8    9
         dtype: int64
```

Extracting a sequence of elements

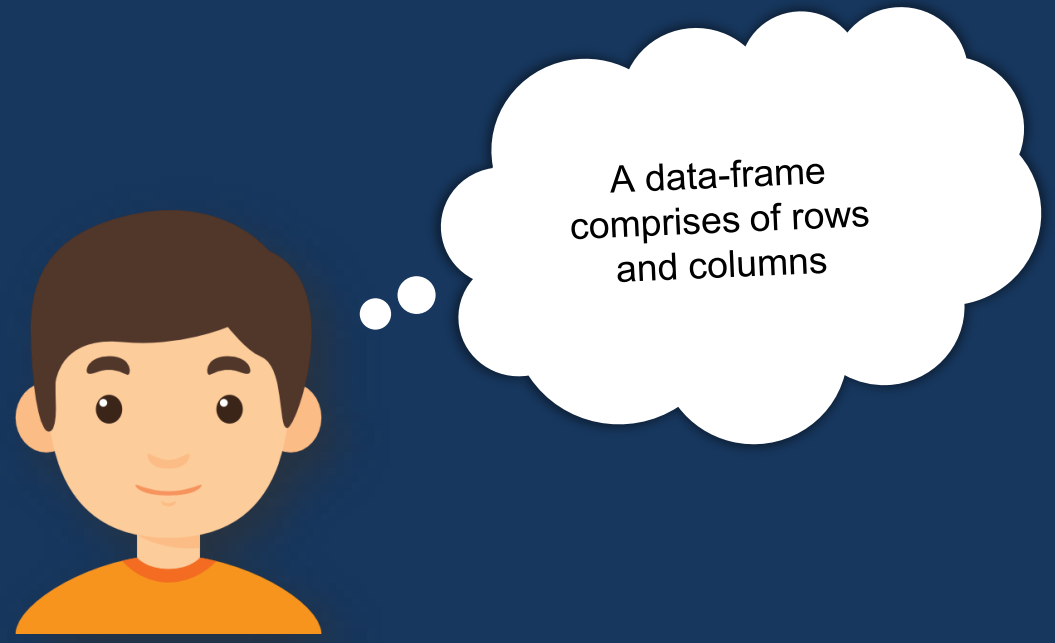
```
In [16]: s1 = pd.Series([1,2,3,4,5,6,7,8,9])
         s1[:4]

Out[16]: 0    1
         1    2
         2    3
         3    4
         dtype: int64
```

Pandas Dataframe

greatlearning
Learning for Life

Dataframe is a 2-dimensional labelled data-structure



Out[9]:

	Name	Marks
0	Bob	76
1	Sam	25
2	Anne	92

Creating a Dataframe

greatlearning
Learning for Life

This is how you can create a data.frame

```
In [9]: import pandas as pd

        pd.DataFrame({"Name":['Bob','Sam','Anne'],"Marks":[76,25,92]})
```

Out[9]:

	Name	Marks
0	Bob	76
1	Sam	25
2	Anne	92

Dataframe In-Built Functions

greatlearning
Learning for Life

head()

shape()

describe()

tail()

SeaBorn Line Plot

greatlearning
Learning for Life

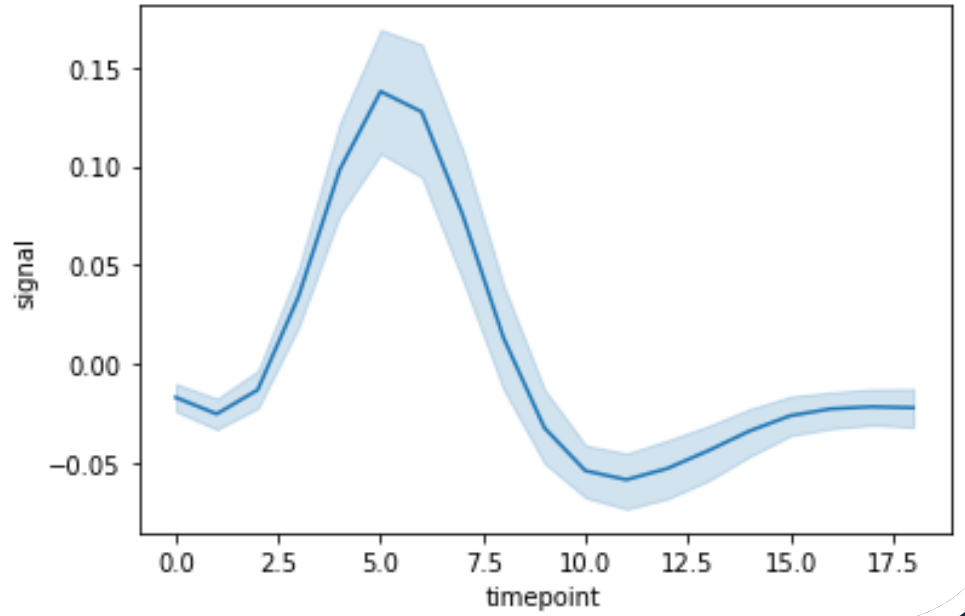
```
In [10]: import seaborn as sns
         from matplotlib import pyplot as plt
```

```
In [18]: fmri = sns.load_dataset("fmri")
         fmri.head()
```

Out[18]:

	subject	timepoint	event	region	signal
0	s13	18	stim	parietal	-0.017552
1	s5	14	stim	parietal	-0.080883
2	s12	18	stim	parietal	-0.081033
3	s11	18	stim	parietal	-0.046134
4	s10	18	stim	parietal	-0.037970

```
In [20]: sns.lineplot(x="timepoint", y="signal", data=fmri)
         plt.show()
```

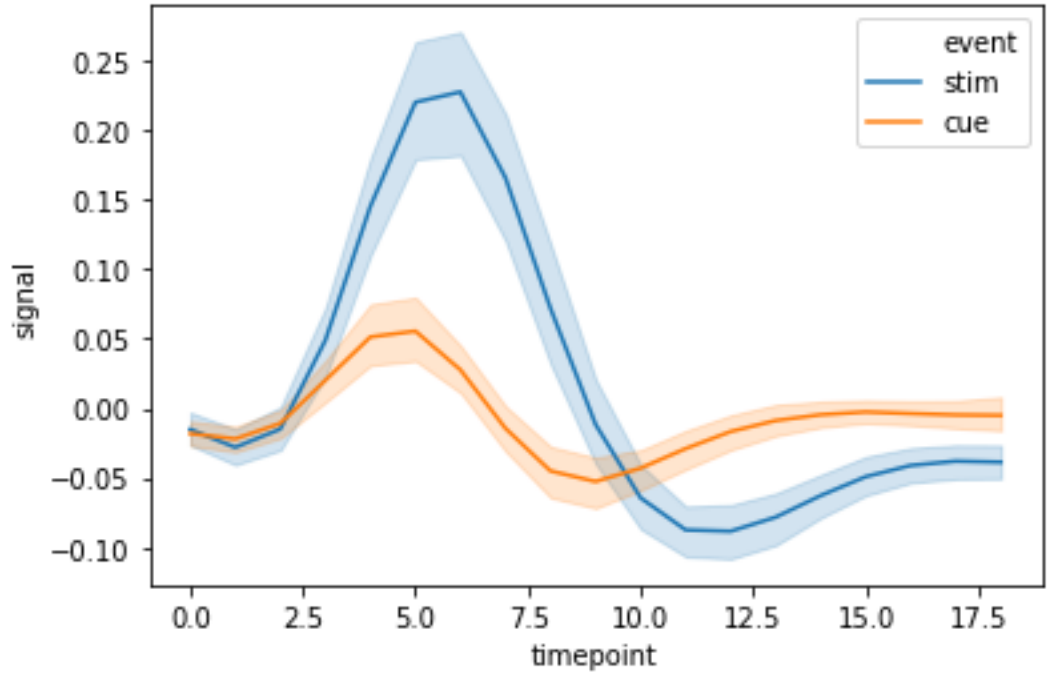


SeaBorn Line Plot

greatlearning
Learning for Life

Grouping data with ‘hue’

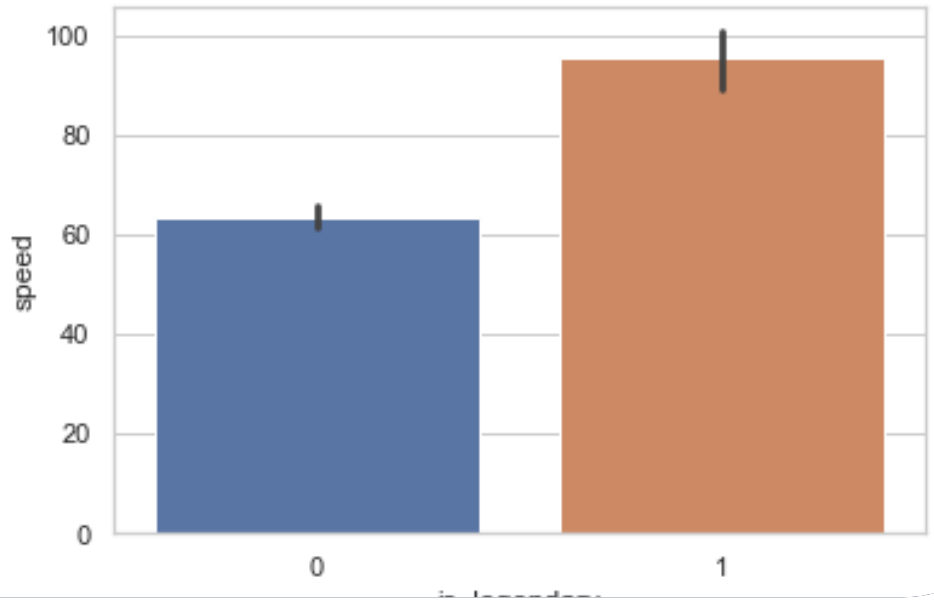
```
In [21]: sns.lineplot(x="timepoint", y="signal", data=fmri,hue="event")
         plt.show()
```



SeaBorn Bar Plot

```
In [29]: import pandas as pd
         sns.set(style="whitegrid")
         pokemon=pd.read_csv('pokemon.csv')
```

```
In [31]: sns.barplot(x="is_legendary", y="speed", data=pokemon)
         plt.show()
```



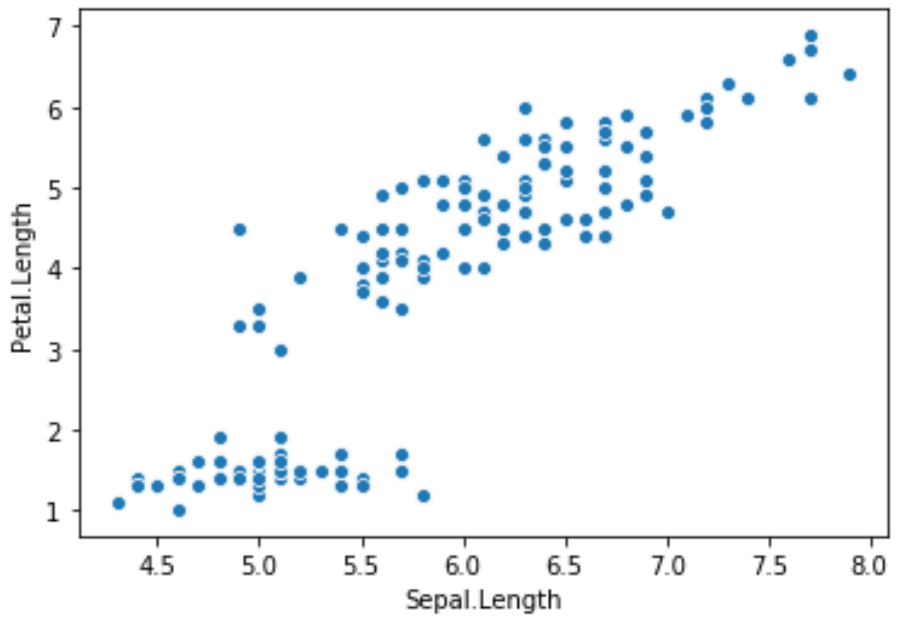
SeaBorn Scatterplot

greatlearning
Learning for Life

```
In [5]: iris = pd.read_csv('iris.csv')
        iris.head()
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

```
sns.scatterplot(x="Sepal.Length", y="Petal.Length", data=iris)
plt.show()
```



Case Study

Case Study

We will have a case study on this census dataset

age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex
90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female
82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female
66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female
54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female
41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female

Case Study

```
import pandas as pd
census=pd.read_csv('census.csv')
census.head()
```

age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex
90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female
82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female
66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female
54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female
41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female

```
In [7]: census.shape
Out[7]: (32561, 15)
```

Case Study

```
census['age'].min()
```

```
17
```

```
census['hours.per.week'].mean()
```

```
40.437455852092995
```

```
census['age'].max()
```

```
90
```

```
census['hours.per.week'].max()
```

```
99
```

Case Study

```
census['race'].value_counts()
```

```
White                 27816
Black                  3124
Asian-Pac-Islander     1039
Amer-Indian-Eskimo      311
Other                   271
Name: race, dtype: int64
```

```
census['sex'].value_counts()
```

```
Male      21790
Female    10771
Name: sex, dtype: int64
```

```
census['income'].value_counts()
```

```
<=50K    24720
>50K      7841
Name: income, dtype: int64
```

```
census['workclass'].value_counts()
```

```
Private             22696
Self-emp-not-inc     2541
Local-gov            2093
?                    1836
State-gov            1298
Self-emp-inc         1116
Federal-gov           960
Without-pay            14
Never-worked            7
Name: workclass, dtype: int64
```

Renaming Columns

```
census.rename(columns={'workclass':"employment_type"},inplace=True)
```

```
census.rename(columns={'hours.per.week':"hours_worked"},inplace=True)
```

Extracting Individual Columns

```
unmarried = census[census['relationship']=='Unmarried']
unmarried.head()
```

```
divorced = census[census['marital.status']=='Divorced']
divorced.head()
```

Case Study

greatlearning
Learning for Life

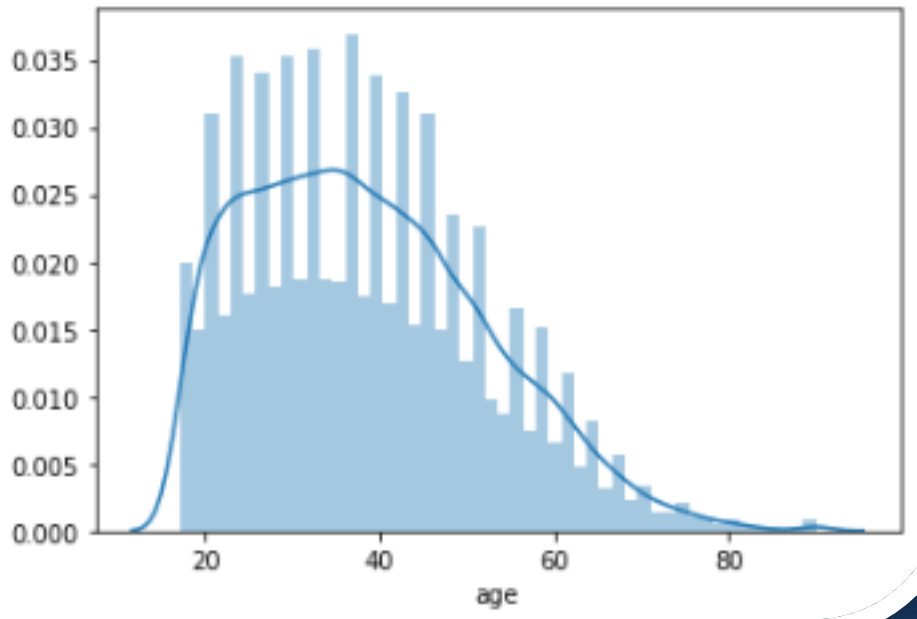
```
old_male = census[(census['age']>50) & (census['sex']=='Male')]
old_male.head()
```

```
white_income = census[(census['race']=='White') & (census['income']=='>50K') ]
white_income.head()
```

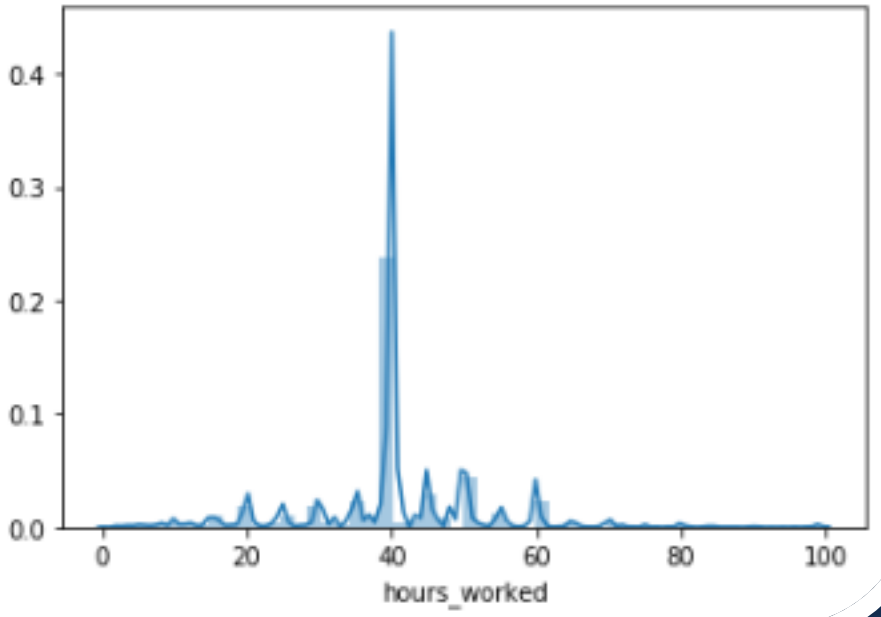
```
master_private= census[(census['education']=='Masters') & (census['employment_type']=='Private') ]
master_private.head()
```

Case Study

```
sns.distplot(census['age'])
plt.show()
```

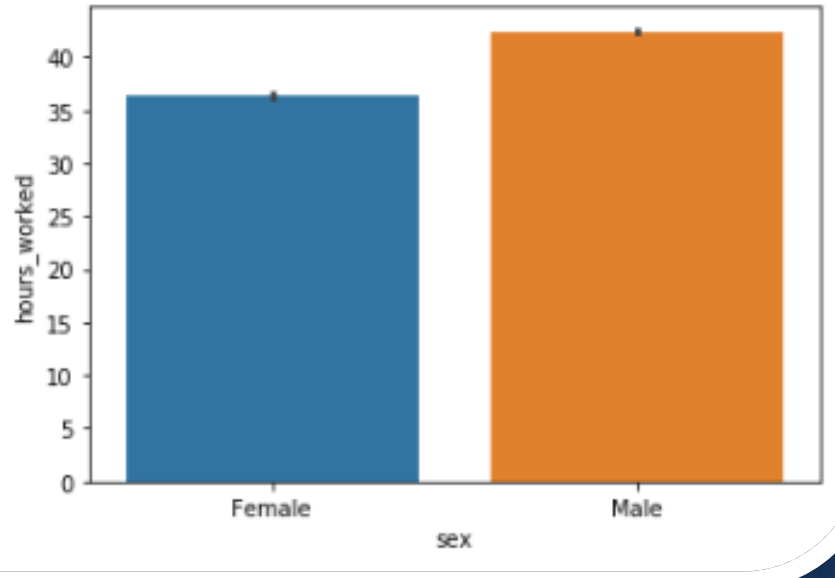



```
sns.distplot(census['hours_worked'])
plt.show()
```

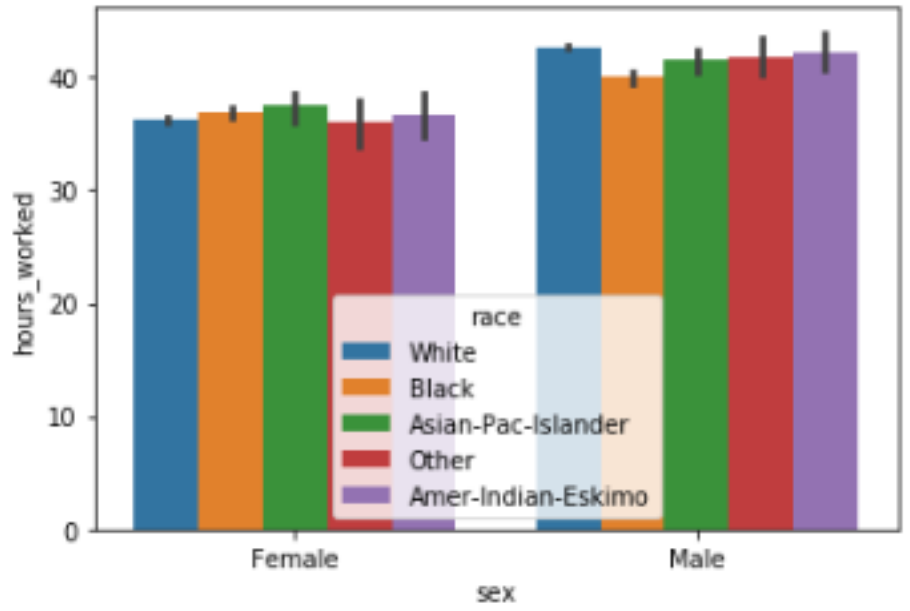


Case Study

```
sns.barplot(x='sex',y='hours_worked',data=census)
plt.show()
```

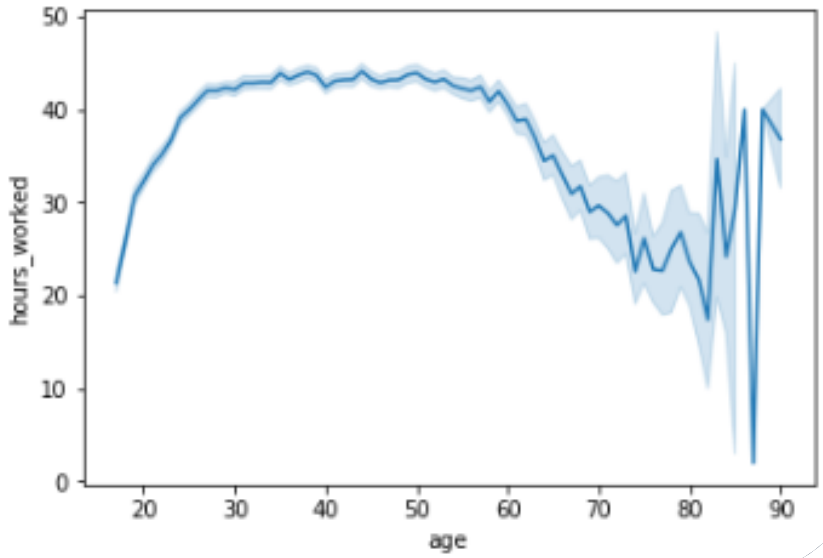



```
sns.barplot(x='sex',y='hours_worked',data=census,hue="race")
plt.show()
```

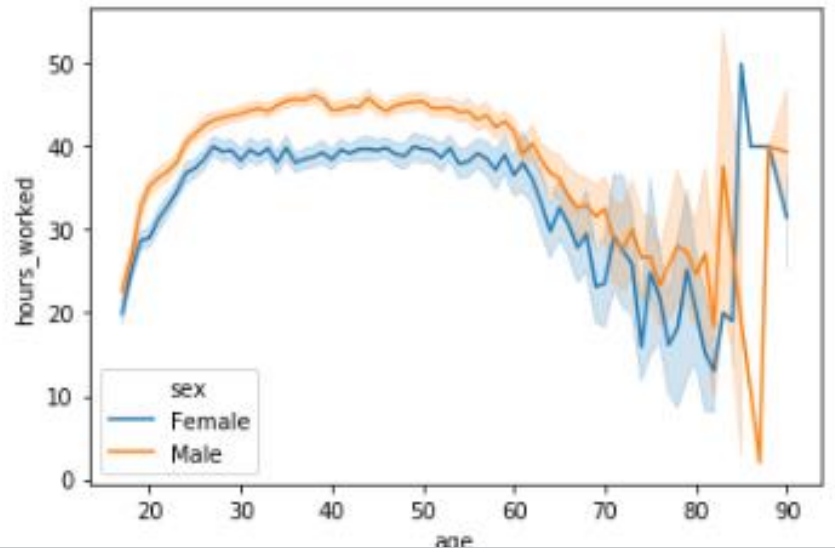


Case Study

```
sns.lineplot(x='age',y='hours_worked',data=census)
plt.show()
```

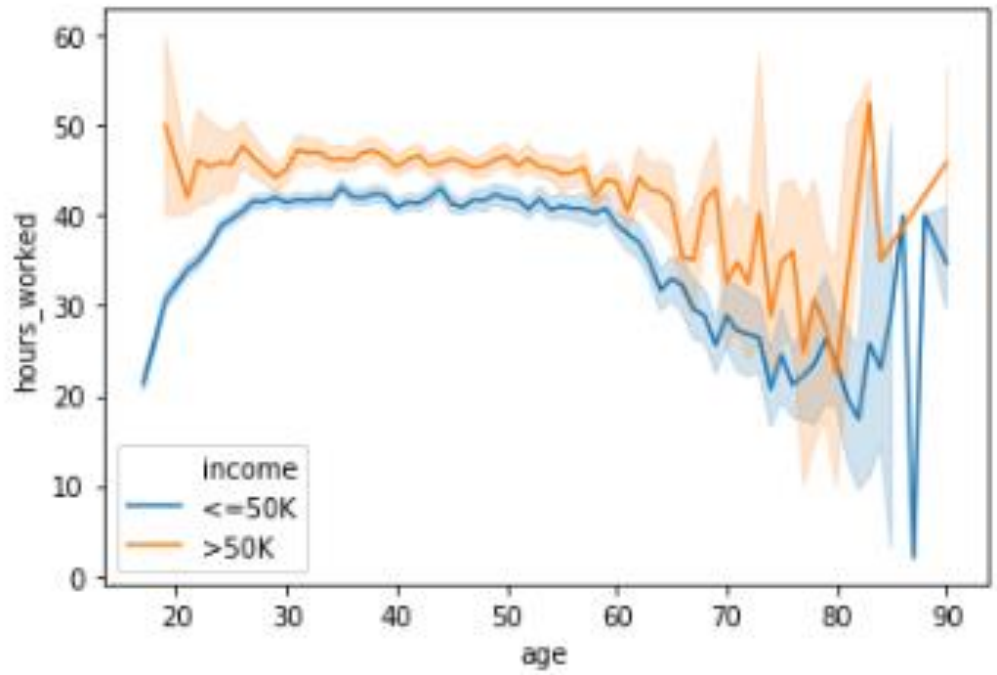



```
sns.lineplot(x='age',y='hours_worked',data=census,hue="sex")
plt.show()
```



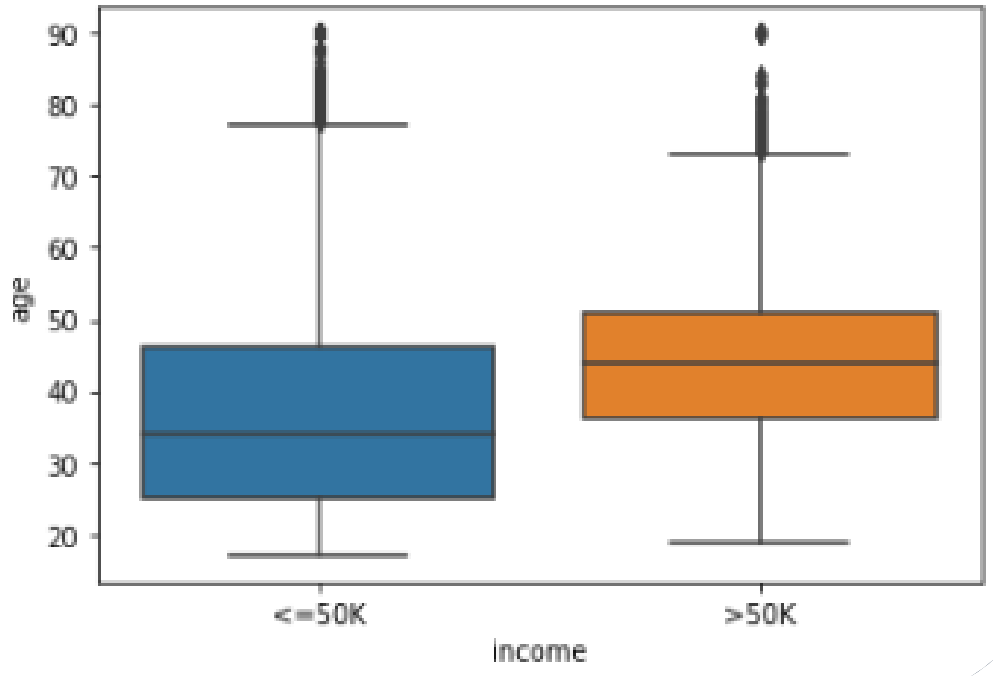
Case Study

```
sns.lineplot(x='age',y='hours_worked',data=census,hue="income")
plt.show()
```



Case Study

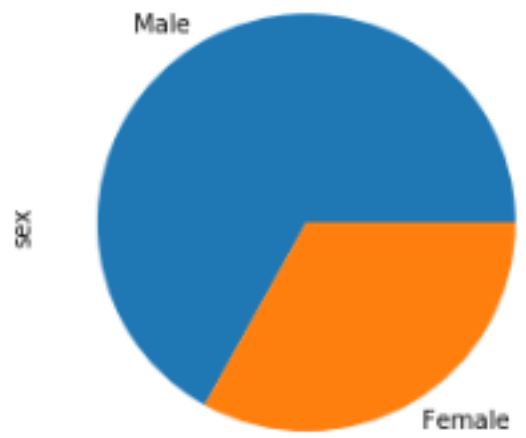
```
sns.boxplot(x='income',y='age',data=census)
plt.show()
```



Case Study

```
census.sex.value_counts().plot(kind='pie')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1fe2e5bfcc8>
```




```
census.income.value_counts().plot(kind='pie')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1fe2b6a4188>
```