Chapter 3: Data Visualisation Numpy-Linear Algebra library binding C librarys Pandas - Built on top of Numpy for fast analysis,data cleaning and preparation with built-in visualisatiom feature Matpotlib- visualising data graphically Seaborn-Statistical plotting library

Python Libraries:

Numpy- Linear Algebra library binding C librarys Used for working with arrays it also has functions for working in domain of fourrier transform and matrices

In [1]:
```
import pandas as pd
df = pd.read_csv('temporal.txt')
#df.head(10)
df
```

Out[1]:

	Mes	data science	machine learning	deep learning	categorical
0	2004-01-01	12	18	4	1
1	2004-02-01	12	21	2	1
2	2004-03-01	9	21	2	1
3	2004-04-01	10	16	4	1
4	2004-05-01	7	14	3	1
...	...	...	...	...	...
189	2019-10-01	90	98	98	0
190	2019-11-01	87	97	96	0
191	2019-12-01	81	89	91	0
192	2020-01-01	94	94	93	1
193	2020-02-01	100	99	99	1

194 rows × 5 columns

In [2]:
```
df.describe
```

Out[2]:
```
<bound method NDFrame.describe of            Mes  data science  machine learning  deep learning  categorical
0    2004-01-01            12                18              4              1
1    2004-02-01            12                21              2              1
2    2004-03-01             9                21              2              1
3    2004-04-01            10                16              4              1
4    2004-05-01             7                14              3              1
..          ...           ...               ...            ...            ...
189  2019-10-01            90                98             98              0
190  2019-11-01            87                97             96              0
191  2019-12-01            81                89             91              0
192  2020-01-01            94                94             93              1
193  2020-02-01           100                99             99              1

[194 rows x 5 columns]>
```

df.describe-

In [ ]:

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 194 entries, 0 to 193
Data columns (total 5 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Mes               194 non-null    object
 1   data science      194 non-null    int64 
 2   machine learning  194 non-null    int64 
 3   deep learning     194 non-null    int64 
 4   categorical       194 non-null    int64 
dtypes: int64(4), object(1)
memory usage: 7.7+ KB
```

In [5]:
```
pd.set_option('display.max_rows',500)
pd.set_option('display.max_columns',500)
pd.set_option('display.width',1000)
```

In [9]:
```
format_dict = {'data science':'${0:,.2f}','Mes':'{:%m-%Y}','machine learning':'{:.2%}'}
#We make sure that the month coloumn has date time format
df['Mes'] = pd.to_datetime(df['Mes'])
```

```
#We apply the style to visualisation
df.head().style.format(format_dict)
```

Out[9]:

	Mes	data science	machine learning	deep learning	categorical
0	01-2004	$12.00	1800.00%	4	1
1	02-2004	$12.00	2100.00%	2	1
2	03-2004	$9.00	2100.00%	2	1
3	04-2004	$10.00	1600.00%	4	1
4	05-2004	$7.00	1400.00%	3	1

In [10]:
```
format_dict = {'Mes':'{:%m-%Y}'}
#Simplified format dictionary with values that do make sense for our data
df.head().style.format(format_dict).highlight_max(color='darkgreen').highlight_min(color='#ff0000')
```

Out[10]:

	Mes	data science	machine learning	deep learning	categorical
0	01-2004	12	18	4	1
1	02-2004	12	21	2	1
2	03-2004	9	21	2	1
3	04-2004	10	16	4	1
4	05-2004	7	14	3	1

In [11]:
```
format_dict = {'Mes':'{:%m-%Y}'}
#Simplified format dictionary with values that do make sense for our data
df.head().style.format(format_dict).highlight_max(color='pink').highlight_min(color='#ffC0CB')
```

Out[11]:

	Mes	data science	machine learning	deep learning	categorical
0	01-2004	12	18	4	1
1	02-2004	12	21	2	1
2	03-2004	9	21	2	1
3	04-2004	10	16	4	1
4	05-2004	7	14	3	1

Mes is the coloumn name intensity of colour decreases/increases- called gradient first 10 records-df.head(10) cmap is a type of graph-BuGn -a color map

In [13]:
```
df.head(10).style.format(format_dict).background_gradient(subset=['data science','machine learning'],cmap = 'BuGn').highlight_ma
```

Out[13]:

	Mes	data science	machine learning	deep learning	categorical
0	01-2004	12	18	4	1
1	02-2004	12	21	2	1
2	03-2004	9	21	2	1
3	04-2004	10	16	4	1
4	05-2004	7	14	3	1
5	06-2004	9	17	3	1
6	07-2004	9	16	3	1
7	08-2004	7	14	3	1
8	09-2004	10	17	4	1
9	10-2004	8	17	4	1

In [14]:
```
df.head().style.format(format_dict).bar(color='red',subset=['data science','deep learning'])
```

Out[14]:

	Mes	data science	machine learning	deep learning	categorical
0	01-2004	12	18	4	1
1	02-2004	12	21	2	1
2	03-2004	9	21	2	1
3	04-2004	10	16	4	1
4	05-2004	7	14	3	1

Pandas-styling pandas.pydata.org minimum 5 different styles-try it out user guide- methods to add styles install anaconda

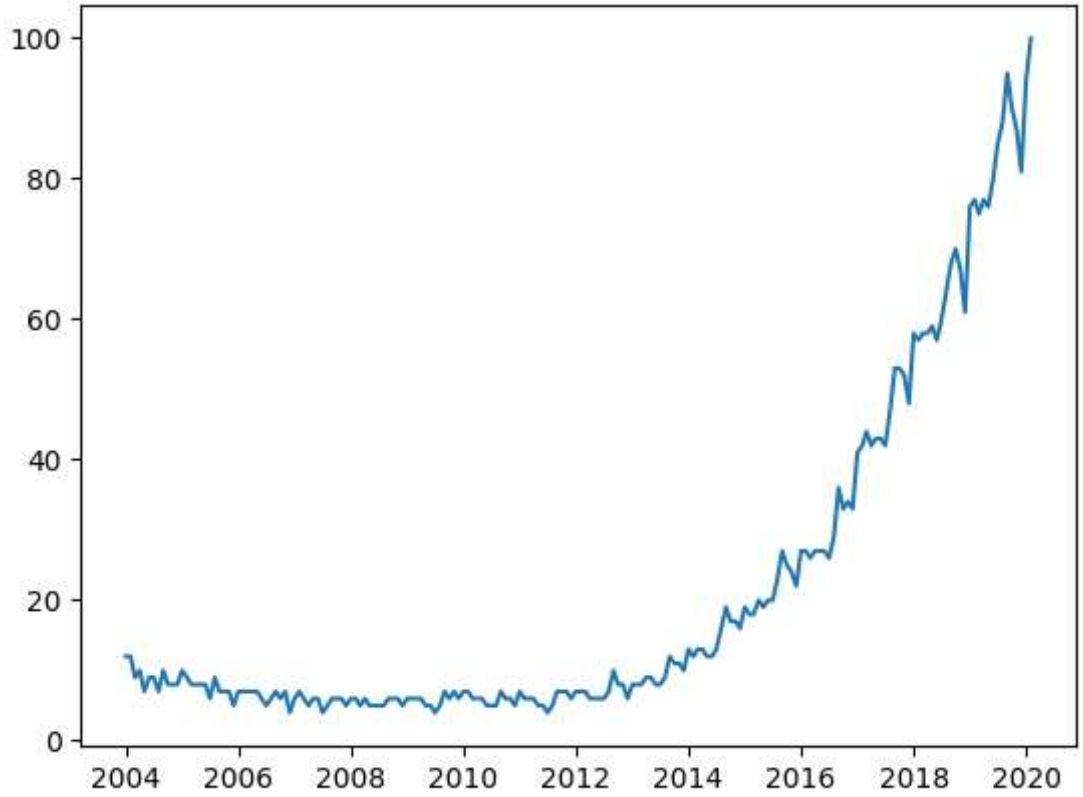
In [15]:
```
df.head(10).style.format(format_dict).background_gradient(subset=['data science','machine learning'], cmap ='BuGn').highlight_ma
```

Out[15]:

	Mes	data science	machine learning	deep learning	categorical
0	01-2004	12	18	4	1
1	02-2004	12	21	2	1
2	03-2004	9	21	2	1
3	04-2004	10	16	4	1
4	05-2004	7	14	3	1
5	06-2004	9	17	3	1
6	07-2004	9	16	3	1
7	08-2004	7	14	3	1
8	09-2004	10	17	4	1
9	10-2004	8	17	4	1

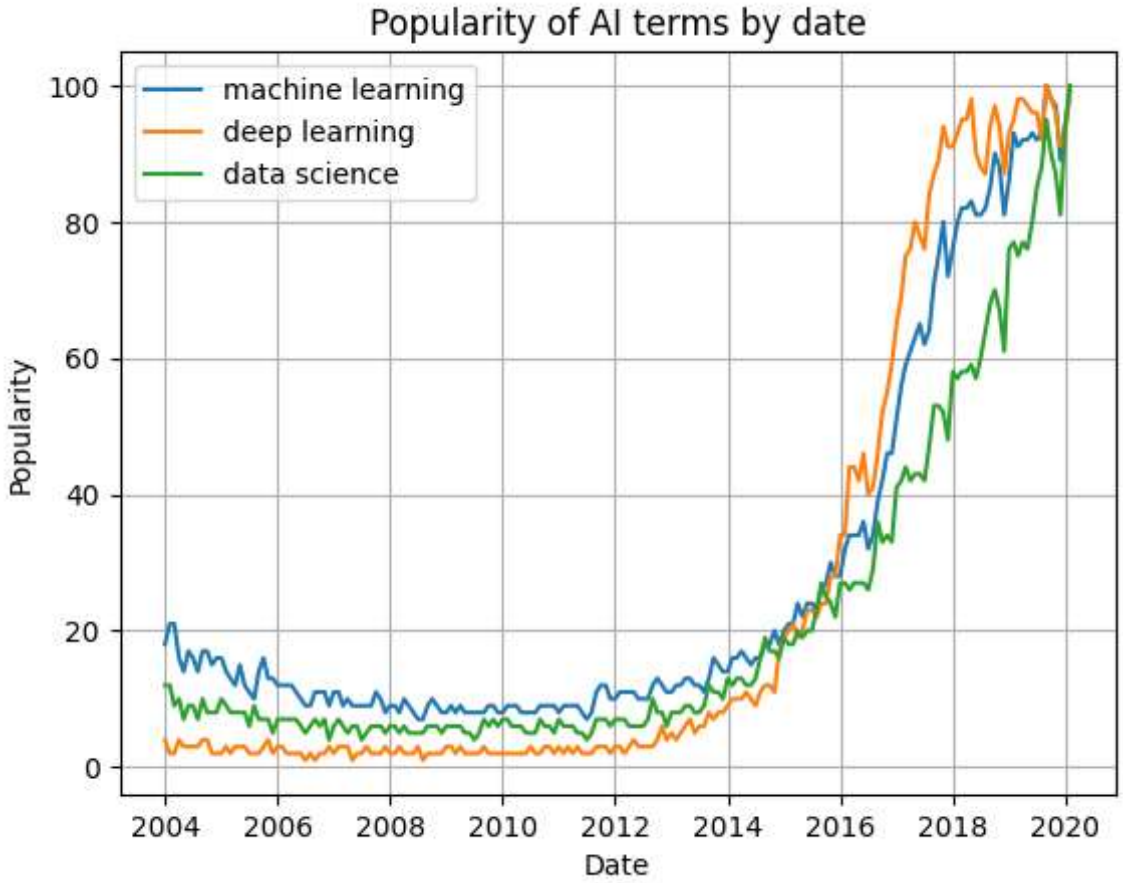
In [20]:
```
import matplotlib.pyplot as plt
plt.plot(df['Mes'],df['data science'],label='data science')
```

Out[20]:
```
[<matplotlib.lines.Line2D at 0x276437c2140>]
```



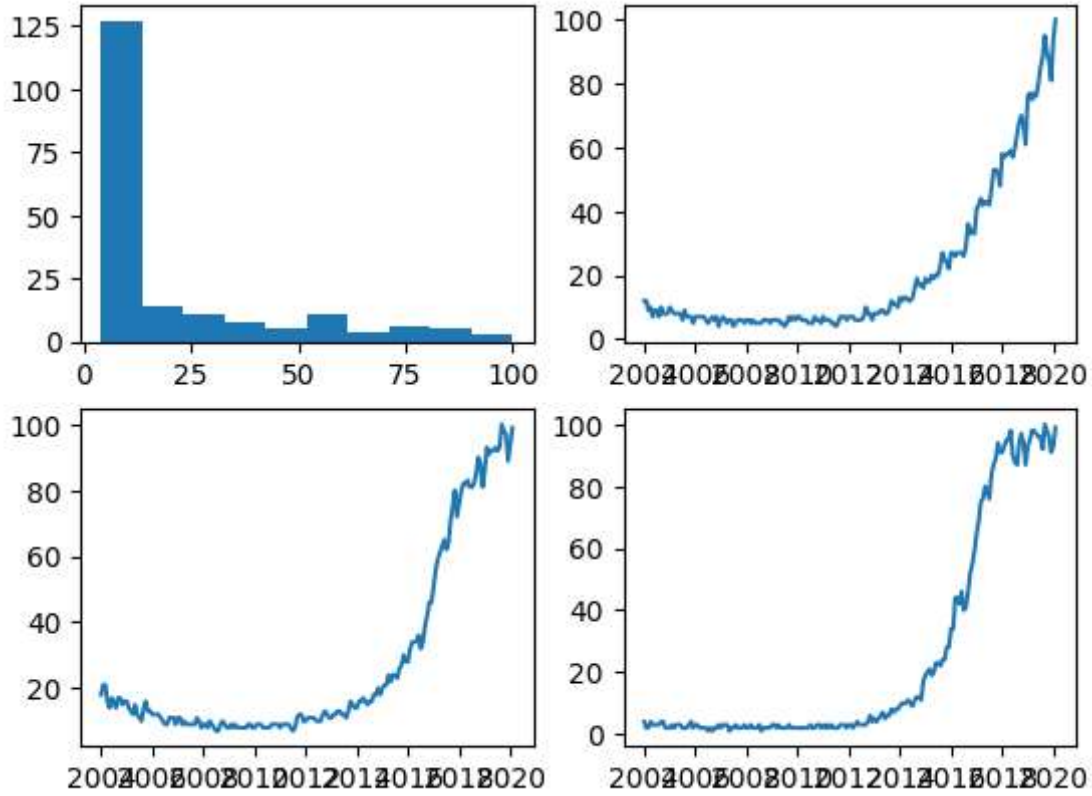
In [21]:
```
plt.plot(df['Mes'],df['machine learning'],label ='machine learning')
plt.plot(df['Mes'],df['deep learning'],label ='deep learning')
plt.plot(df['Mes'],df['data science'],label ='data science')
plt.xlabel('Date')
plt.ylabel('Popularity')
plt.title('Popularity of AI terms by date')
plt.grid(True)
plt.legend()
```

Out[21]:
```
<matplotlib.legend.Legend at 0x27645a5fdf0>
```



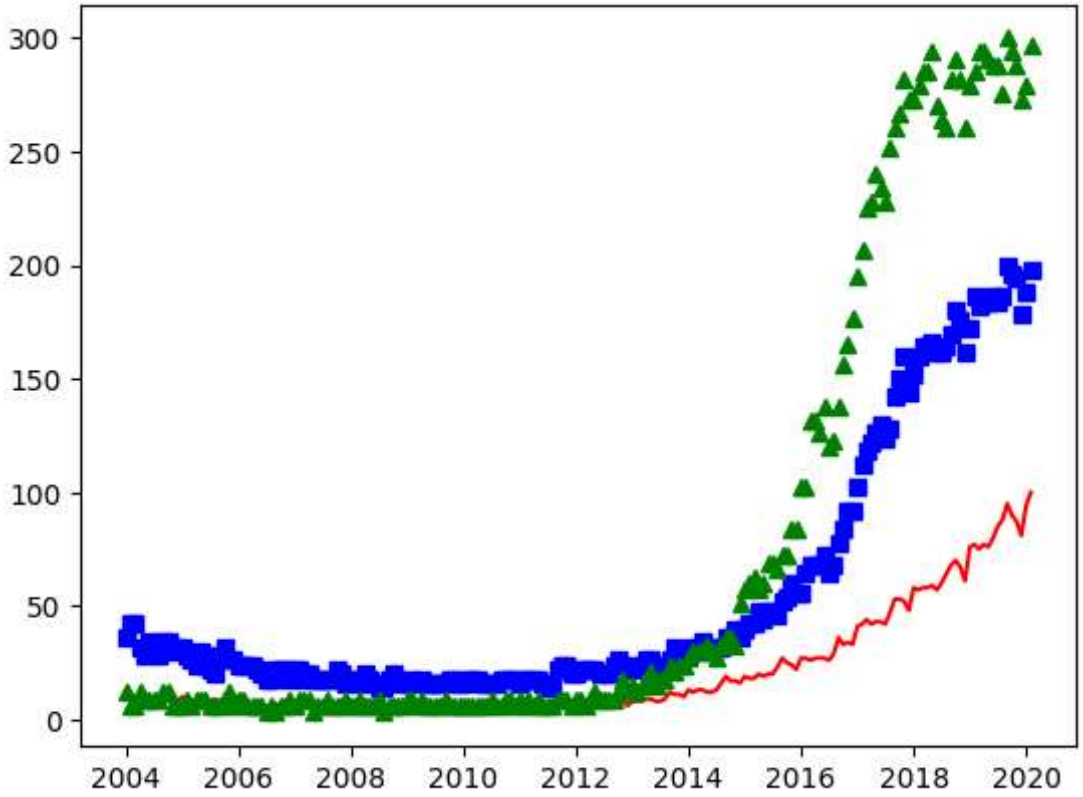
In [22]:
```
fig,axes= plt.subplots(2,2)
axes[0,0].hist(df['data science'])
axes[0,1].plot(df['Mes'],df['data science'])
axes[1,0].plot(df['Mes'],df['machine learning'])
axes[1,1].plot(df['Mes'],df['deep learning'])
```

Out[22]:
```
[<matplotlib.lines.Line2D at 0x27646bb57e0>]
```



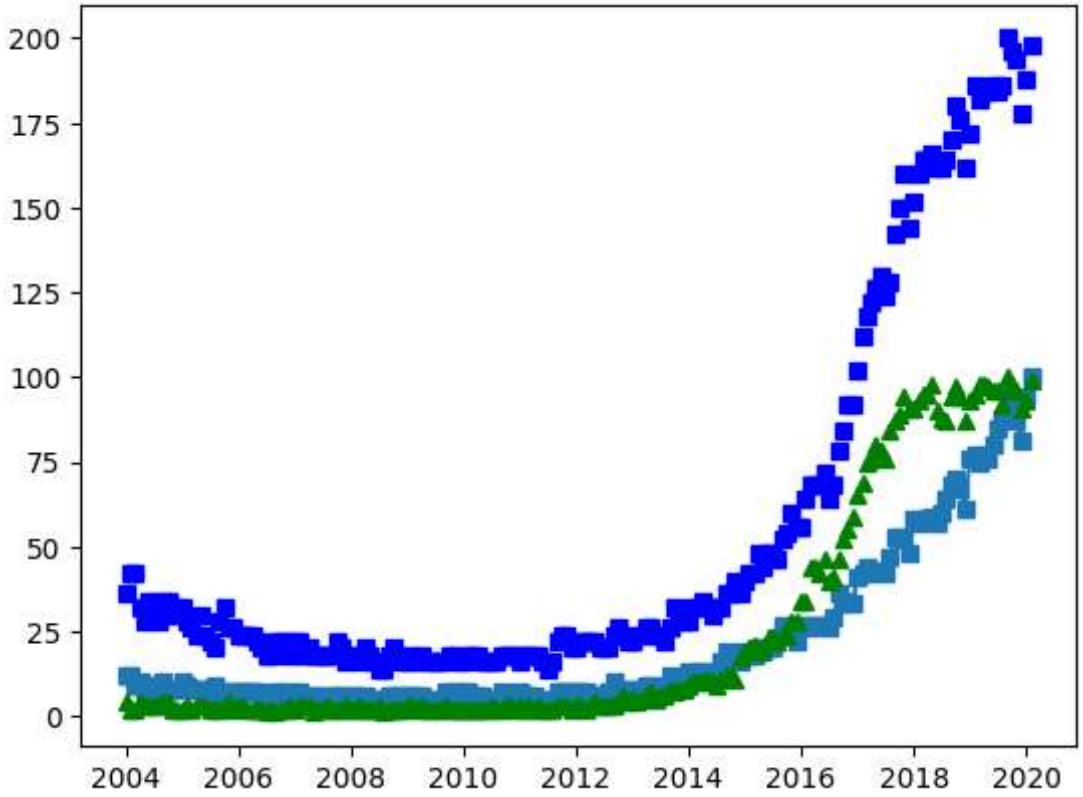
In [25]:
```
plt.plot(df['Mes'],df['data science'], 'r-')
plt.plot(df['Mes'],df['machine learning']*2, 'bs')
plt.plot(df['Mes'],df['deep learning']*3, 'g^')
```

Out[25]:
```
[<matplotlib.lines.Line2D at 0x27646d75a50>]
```



In [28]:
```
plt.plot(df['Mes'],df['data science'], 's-')
plt.plot(df['Mes'],df['machine learning']*2, 'bs') #increasing the thickness by 2 times *2
plt.plot(df['Mes'],df['deep learning']*1, 'g^') #increasing thickness by 3 times
```

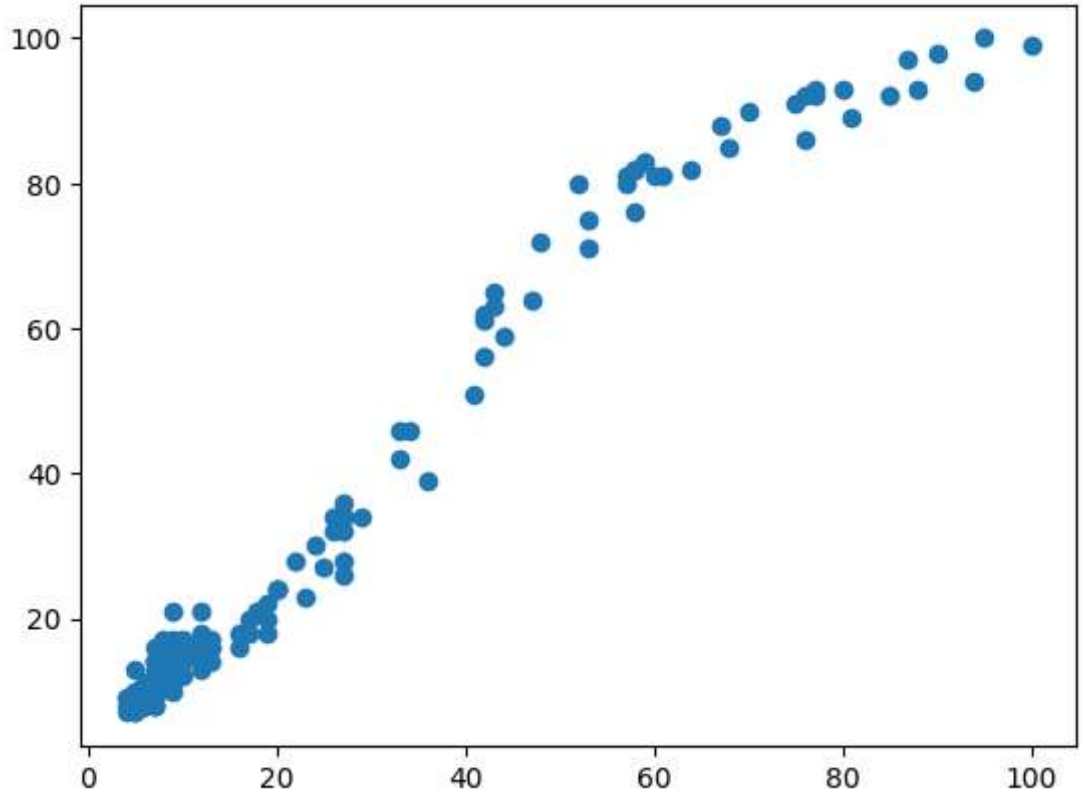
Out[28]:
```
[<matplotlib.lines.Line2D at 0x2764a9c1c00>]
```



scattered plot

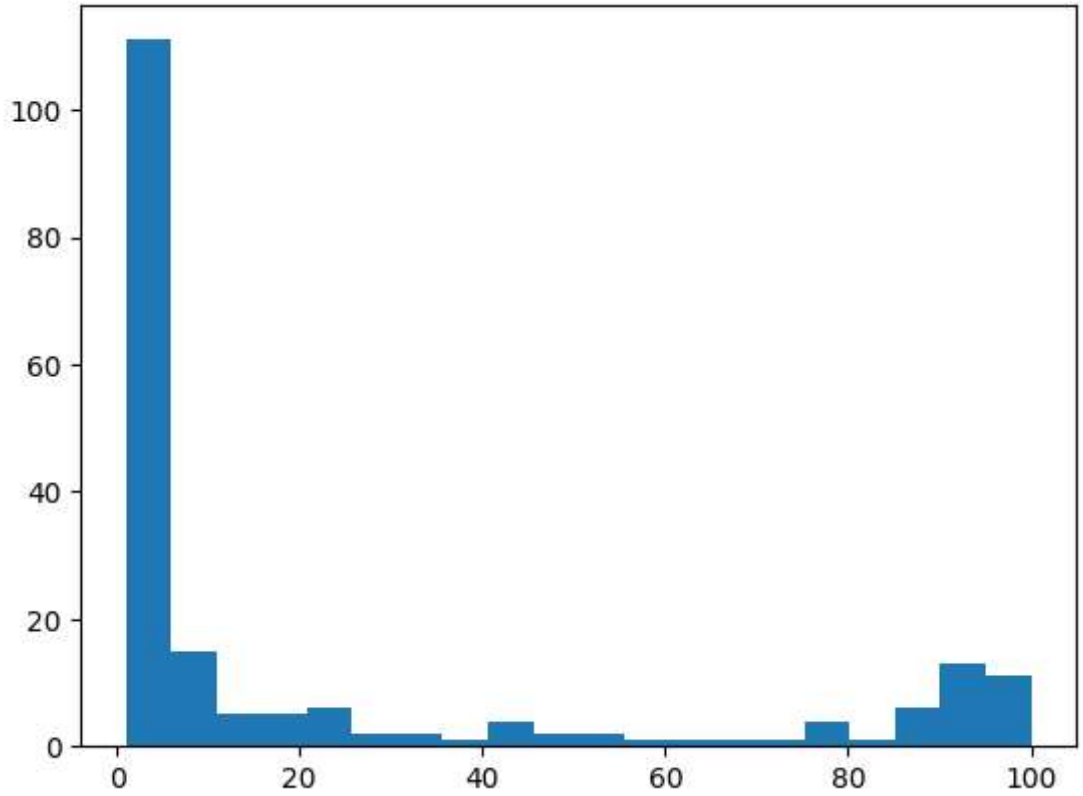
In [29]:
```
plt.scatter(df['data science'],df['machine learning'])
```

Out[29]:
```
<matplotlib.collections.PathCollection at 0x2764aa4df00>
```



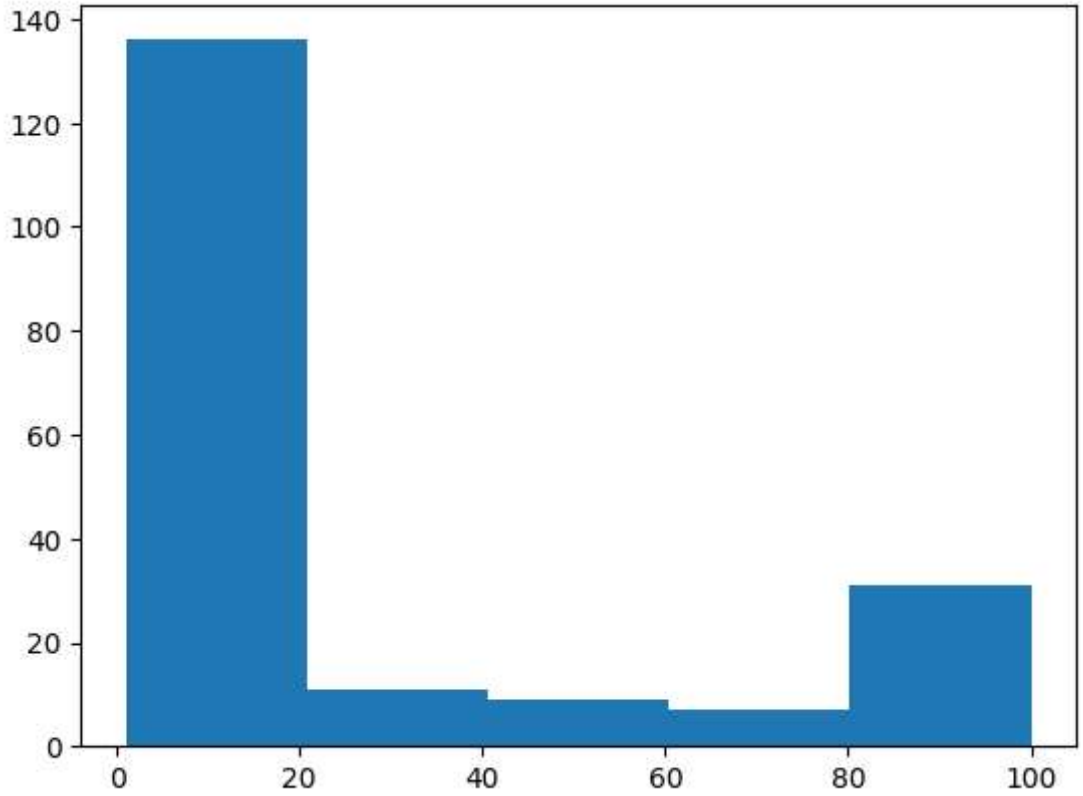
In [33]:
```
plt.hist(df['deep learning'],bins =20)
```

Out[33]:
```
(array([111.,  15.,   5.,   5.,   6.,   2.,   2.,   1.,   4.,   2.,   2.,
          1.,   1.,   1.,   1.,   4.,   1.,   6.,  13.,  11.]),
 array([  1.  ,   5.95,  10.9 ,  15.85,  20.8 ,  25.75,  30.7 ,  35.65,
         40.6 ,  45.55,  50.5 ,  55.45,  60.4 ,  65.35,  70.3 ,  75.25,
         80.2 ,  85.15,  90.1 ,  95.05, 100.  ]),
 <BarContainer object of 20 artists>)
```



In [32]:
```
plt.hist(df['deep learning'],bins =5)
```

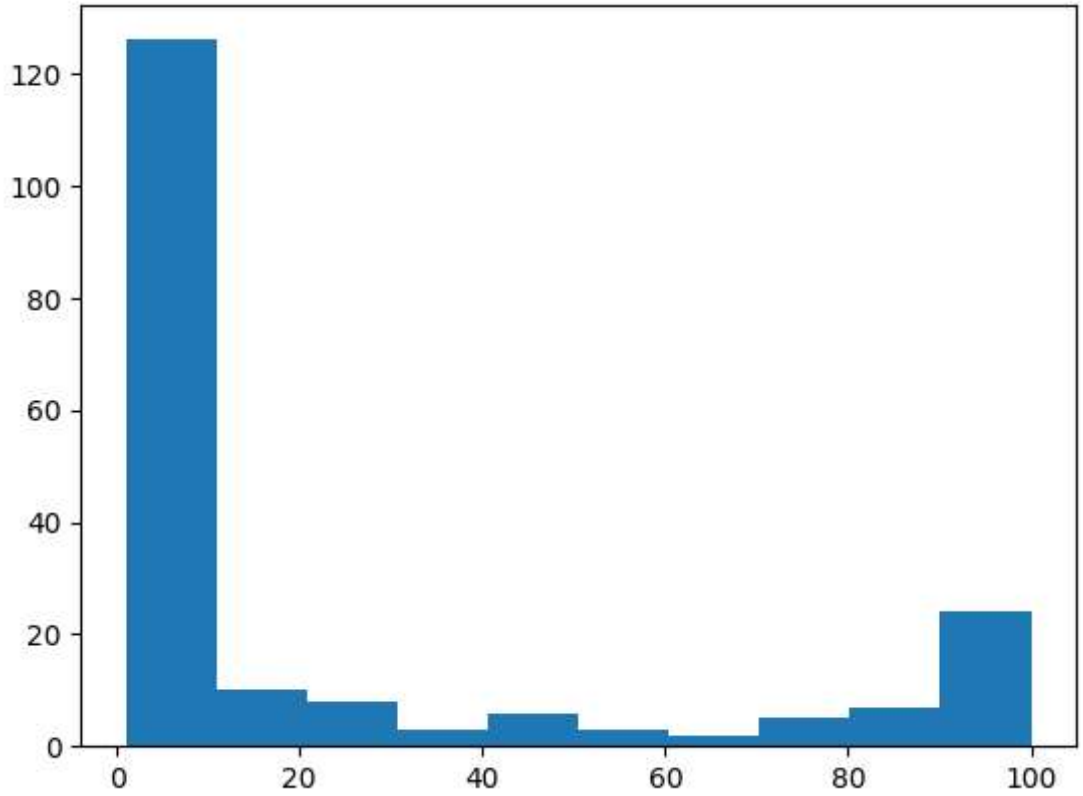
Out[32]:
```
(array([136.,  11.,   9.,   7.,  31.]),
 array([  1. ,  20.8,  40.6,  60.4,  80.2, 100. ]),
 <BarContainer object of 5 artists>)
```



histogram -1D array bins is the interval higher the bin value lower the thickness there are 136 numbers in deep learning- 100/20 = 5 bins total is sum of 1st array

In [34]:
```
plt.hist(df['deep learning'],bins =10)
```

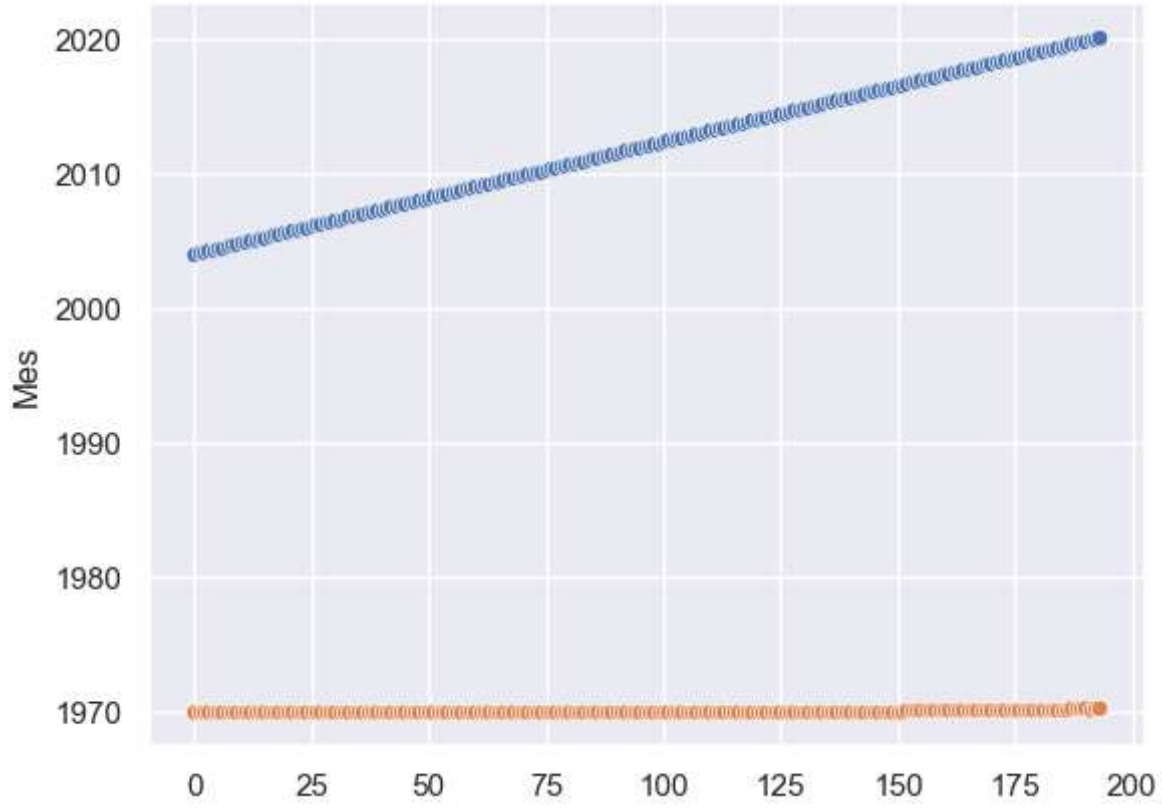
Out[34]:
```
(array([126.,  10.,   8.,   3.,   6.,   3.,   2.,   5.,   7.,  24.]),
 array([  1. ,  10.9,  20.8,  30.7,  40.6,  50.5,  60.4,  70.3,  80.2,
         90.1, 100. ]),
 <BarContainer object of 10 artists>)
```



seaborn

In [38]:
```
import seaborn as sns
sns.set()
sns.scatterplot(df['Mes'])
sns.scatterplot(df['data science'])
```

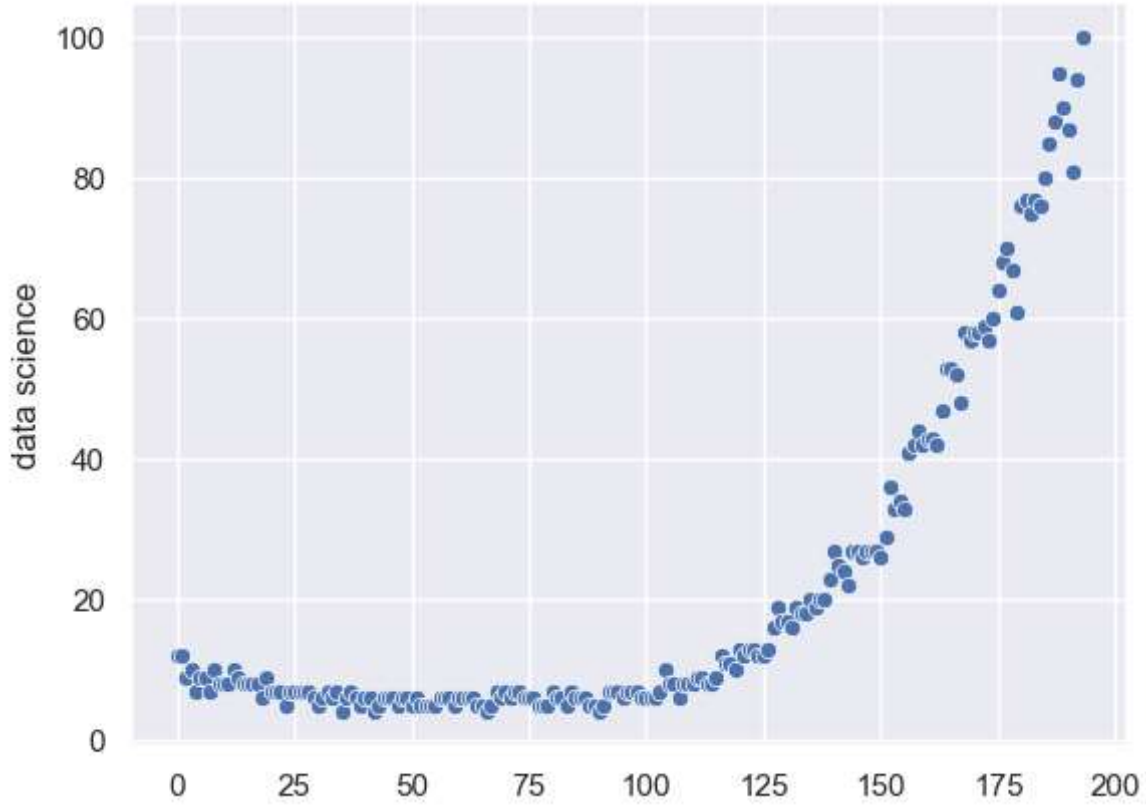
Out[38]:
```
<AxesSubplot: ylabel='Mes'>
```



In [ ]:

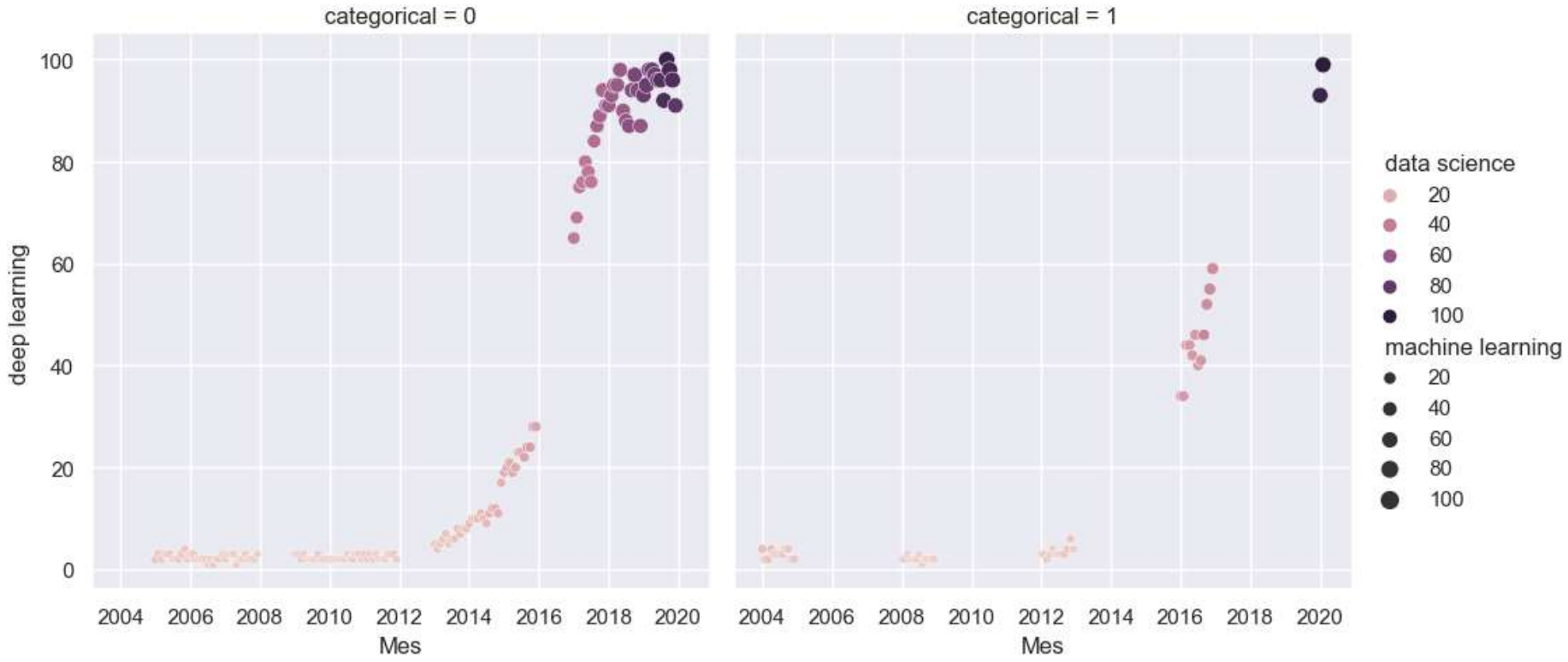
In [39]:
```
sns.scatterplot(df['data science'])
```

Out[39]:
```
<AxesSubplot: ylabel='data science'>
```



In [43]:
```
sns.relplot(x= 'Mes', y='deep learning',hue='data science',size ='machine learning',col='categorical',data=df)
```

Out[43]:
```
<seaborn.axisgrid.FacetGrid at 0x27652edf580>
```

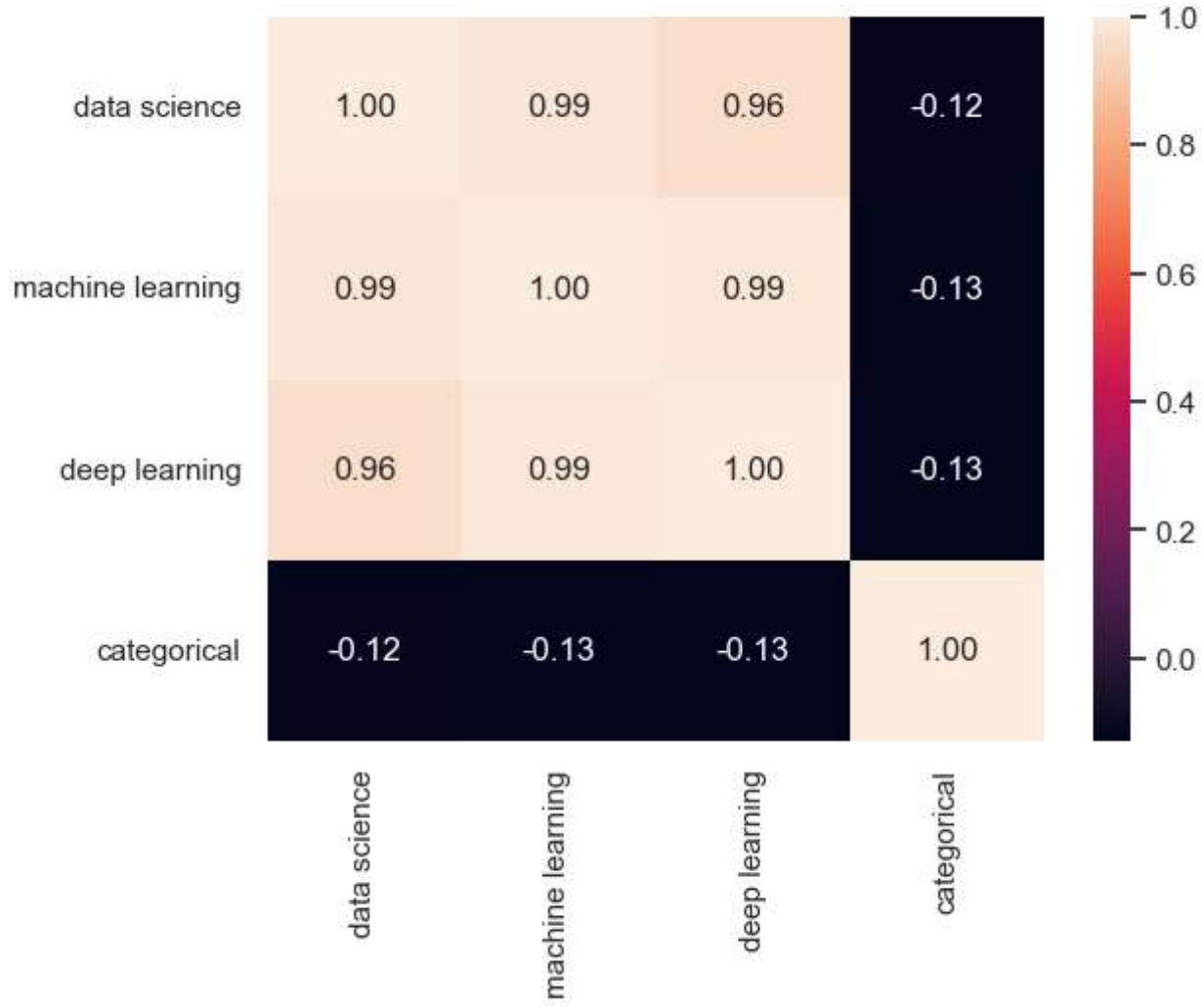


labelled and unlabelled data 0 and 1 supervised learning

In [45]:
```
sns.heatmap(df.corr(), annot=True,fmt='.2f')
```

```
C:\Users\Anusha\AppData\Local\Temp\ipykernel_7732\361305419.py:1: FutureWarning: The default value of numeric_only in DataFrame.
corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_onl
y to silence this warning.
  sns.heatmap(df.corr(), annot=True,fmt='.2f')
```

Out[45]: <AxesSubplot: >

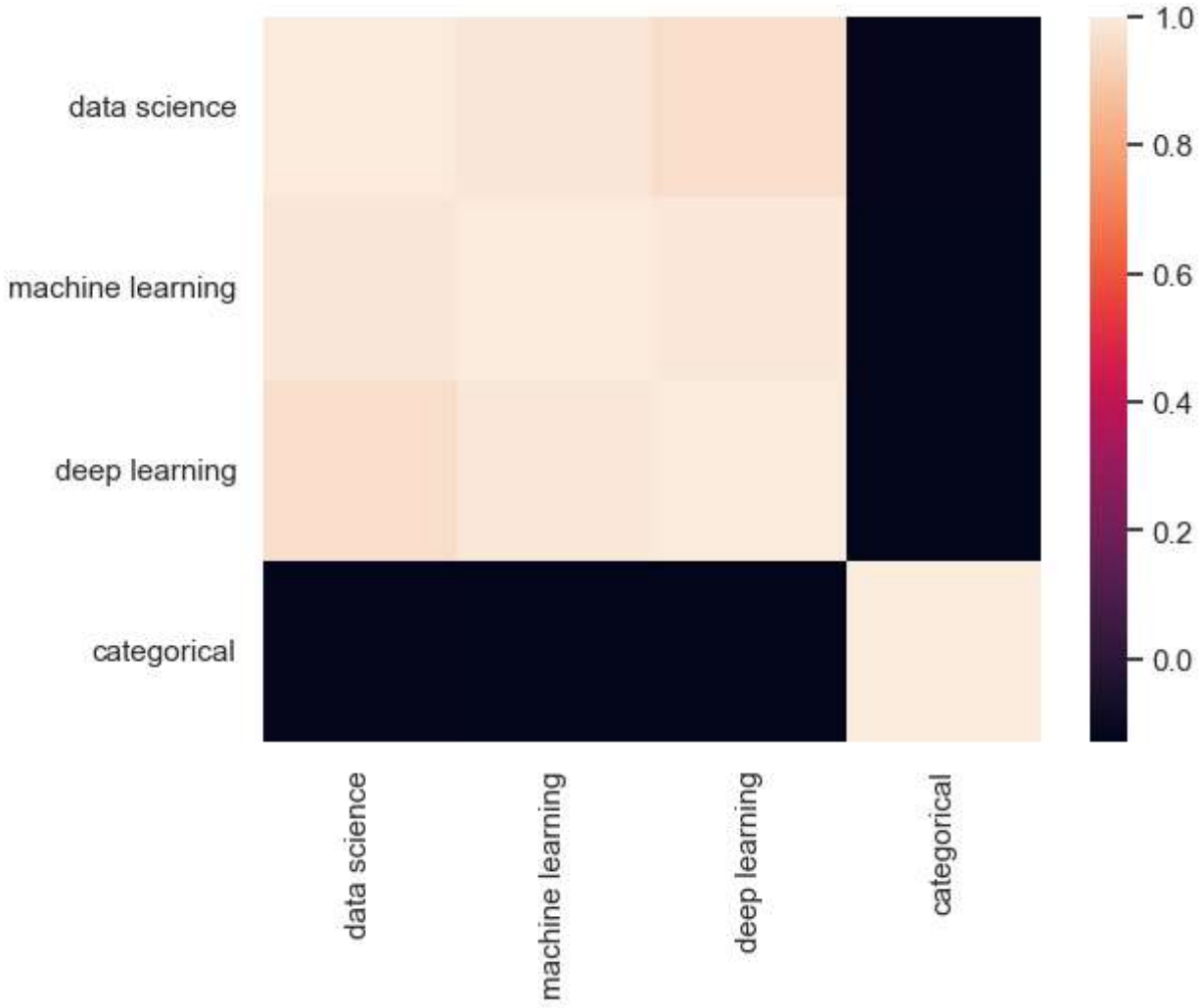


example data science and data science it is equal to 1

In [46]:
```
sns.heatmap(df.corr(), annot=False,fmt='.2f')
```

```
C:\Users\Anusha\AppData\Local\Temp\ipykernel_7732\953803944.py:1: FutureWarning: The default value of numeric_only in DataFrame.
corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_onl
y to silence this warning.
  sns.heatmap(df.corr(), annot=False,fmt='.2f')
```

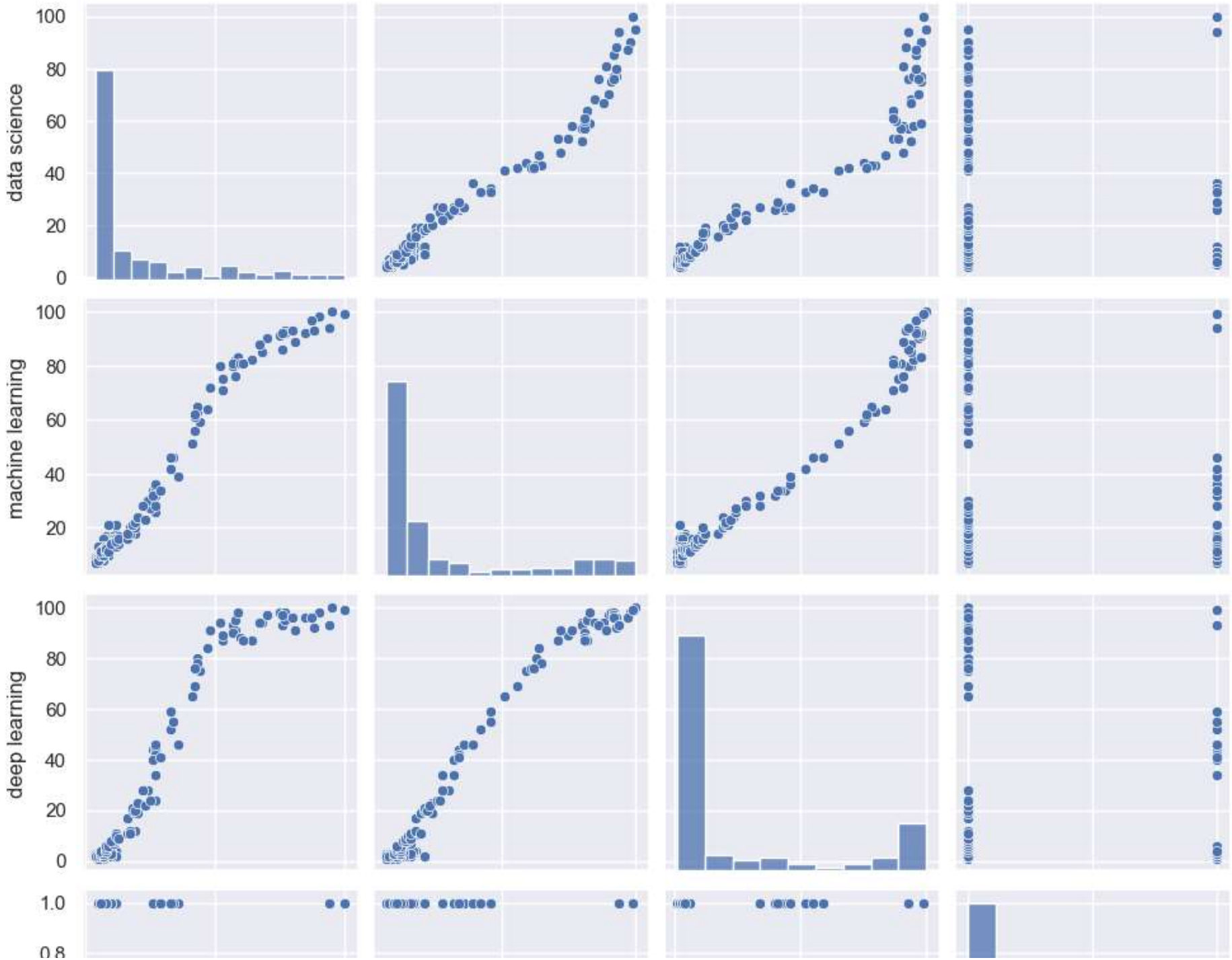
Out[46]: <AxesSubplot: >

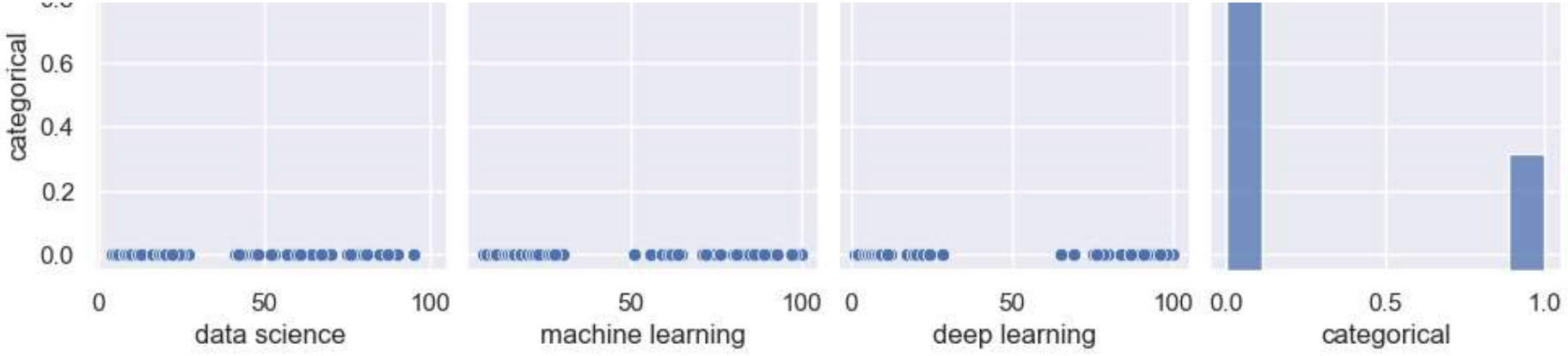


covariance tells how much it is deviating from mean corelation tells square root of covariance 2 data are corelated if its value is near to 1 correlation range is -1 to 1 negative value indicates its negatively correlated.

In [47]:
```
sns.pairplot(df)#for every attribute present in the dataframe there is a graph called a pair plot
```

Out[47]:
```
<seaborn.axisgrid.PairGrid at 0x27646de0970>
```

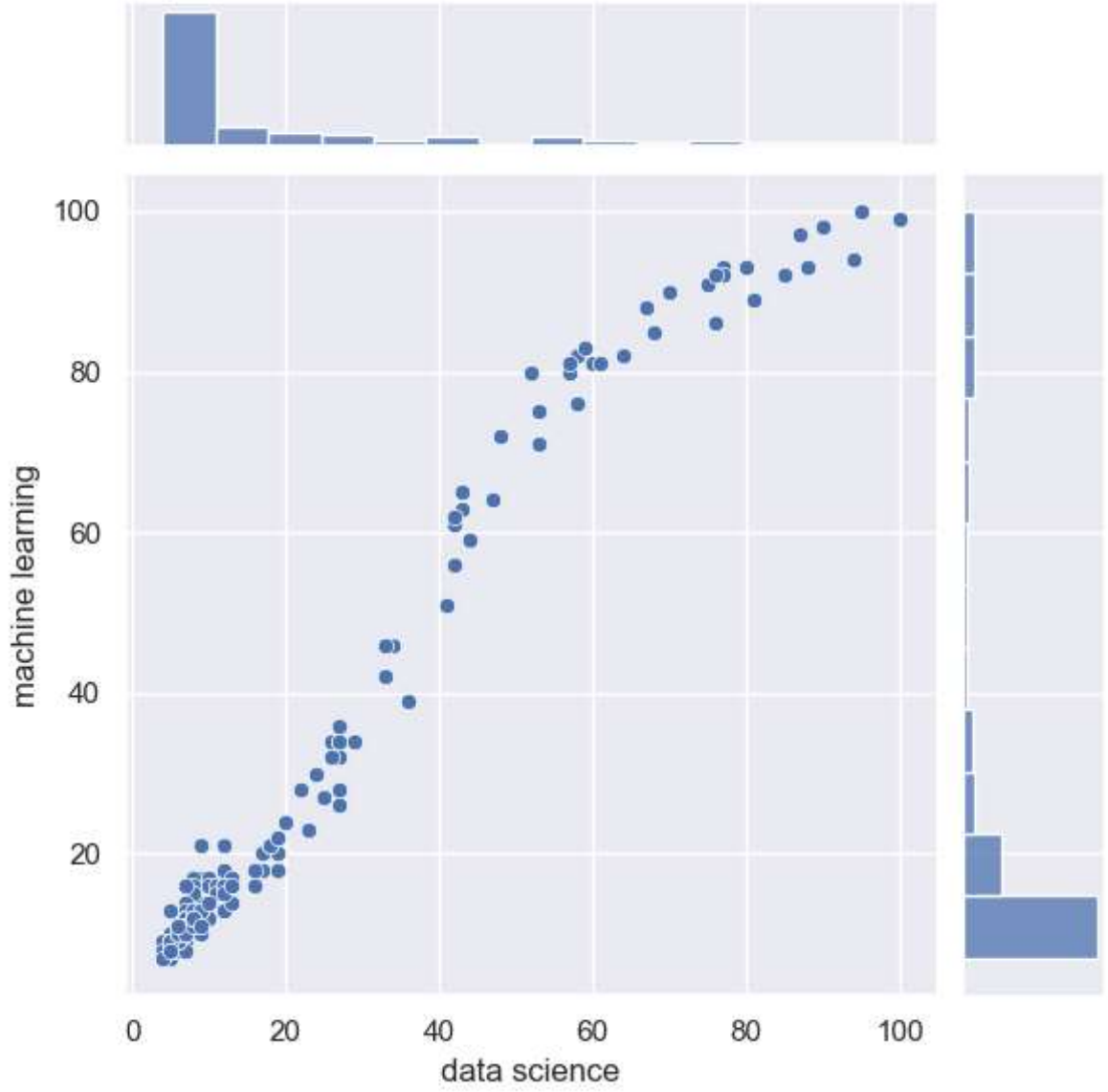

data science
machine learning
deep learning
0
20
40
60
80
100
1.0
0.8



Joint plot

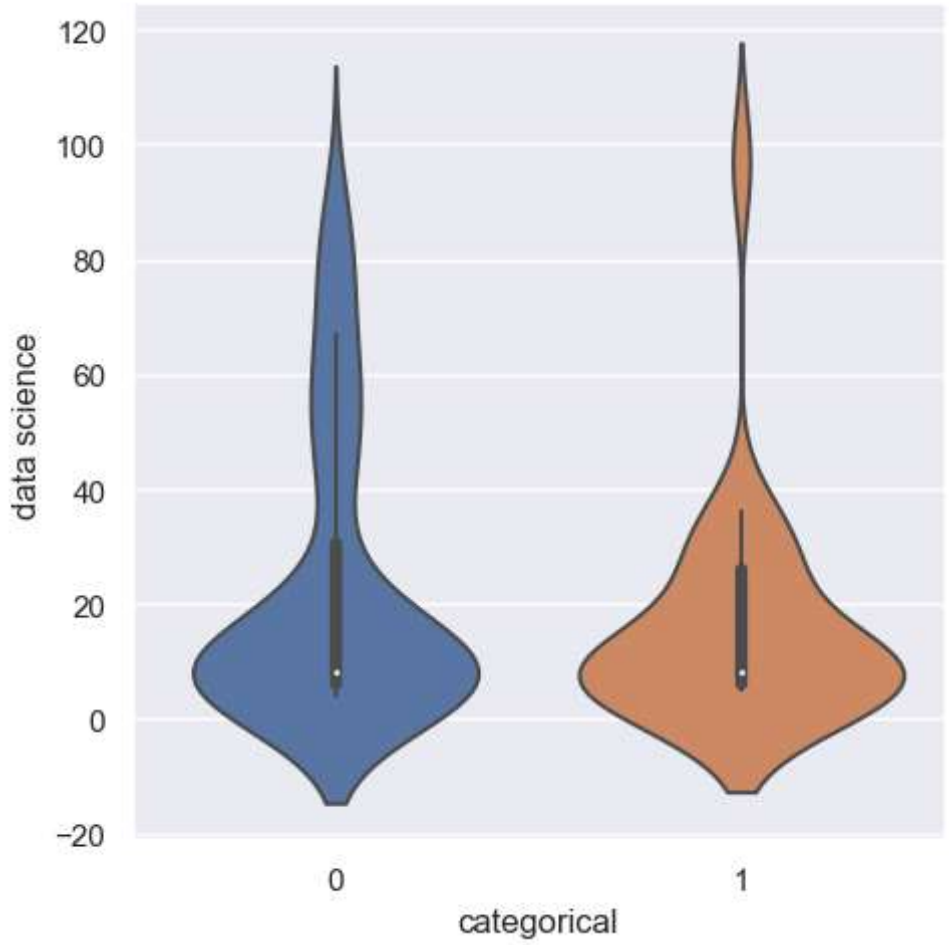
In [48]:
```
sns.jointplot(x='data science',y='machine learning',data =df)
```

Out[48]:
```
<seaborn.axisgrid.JointGrid at 0x27652f4d060>
```



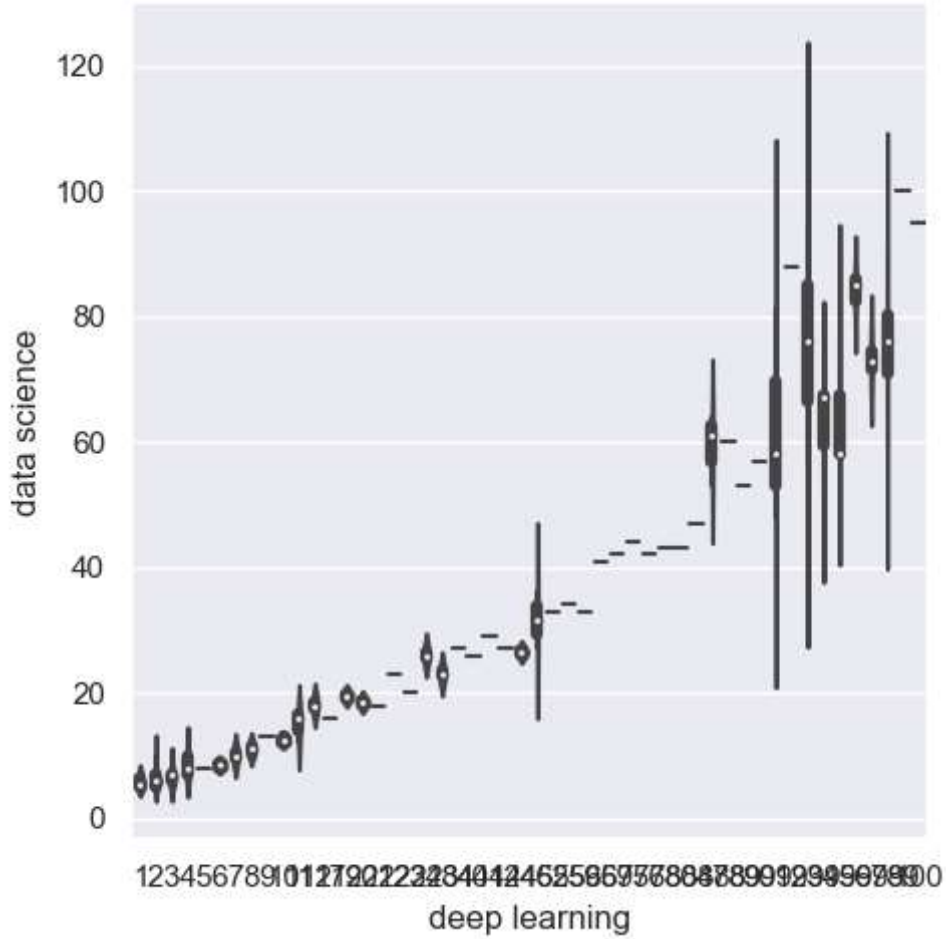
In [49]:
```
sns.catplot(x='categorical',y='data science',kind='violin',data=df) #kind is the shape
```

Out[49]:
```
<seaborn.axisgrid.FacetGrid at 0x27653350fa0>
```



In [54]:
```
sns.catplot(x='deep learning',y='data science',kind='violin',data=df)
```

Out[54]:
```
<seaborn.axisgrid.FacetGrid at 0x27653ec7670>
```

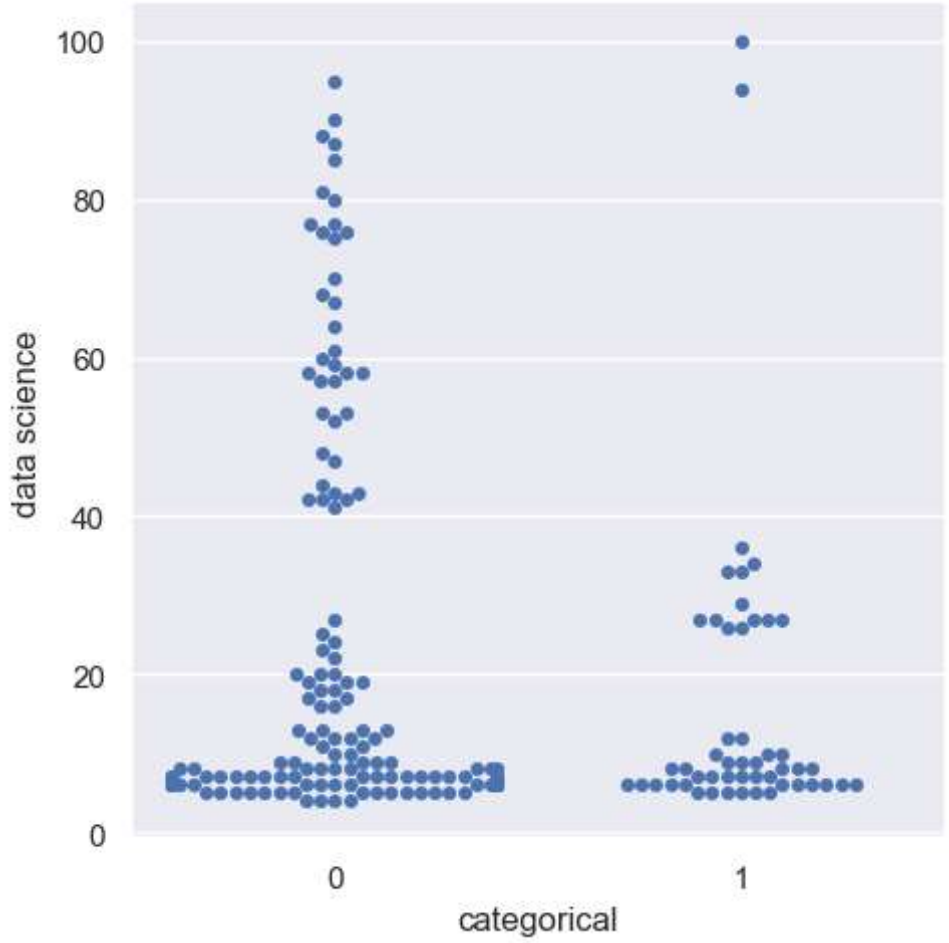


In [ ]:

In [52]:
```
sns.catplot(x='categorical',y='data science',kind='swarm',data=df)
```

Out[52]: <seaborn.axisgrid.FacetGrid at 0x27654f80d00>

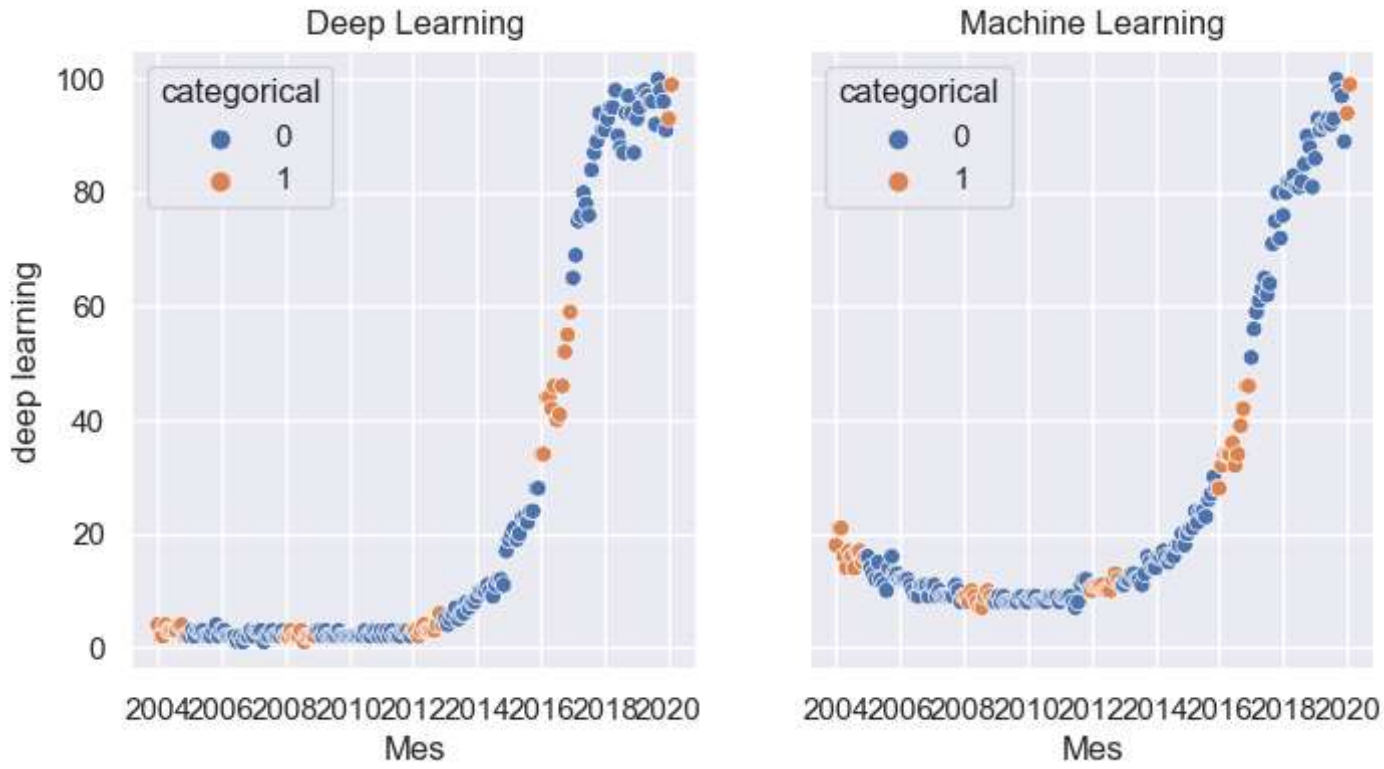
```
C:\Users\Anusha\AppData\Local\Programs\Python\Python310\lib\site-packages\seaborn\categorical.py:3540: UserWarning: 16.7% of the points cannot be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```



subplot in seaborn

In [55]:
```
fig,axes=plt.subplots(1,2, sharey=True,figsize=(8,4))
sns.scatterplot(x='Mes',y='deep learning',hue ='categorical',data=df,ax=axes[0])
axes[0].set_title('Deep Learning')
sns.scatterplot(x='Mes',y='machine learning',hue= 'categorical',data=df,ax=axes[1])
axes[1].set_title('Machine Learning')
```

Out[55]:
```
Text(0.5, 1.0, 'Machine Learning')
```



write description about the graph and what it represents