# Pandas basic

In [5]:
```
import numpy as np
import pandas as pd
```

In [6]:
```
df = pd.DataFrame(
{"a" : [4, 5, 6, 4],
"b" : [7, 8, 9, 9],
"c" : [10, 11, 12, 10]},
index = [1, 2, 3, 4])
df
```

Out[6]:

| | a | b | c |
|---|---|---|---|
| **1** | 4 | 7 | 10 |
| **2** | 5 | 8 | 11 |
| **3** | 6 | 9 | 12 |
| **4** | 4 | 9 | 10 |

In [3]:
```
#Series 형태로 나옴
df['a']
```

Out[3]:
```
1    4
2    5
3    6
Name: a, dtype: int64
```

In [4]:
```
#Series 형태를 dataFrame으로 출력
#Series 는 1차원 dataFrame은 2차원임
df[['a']]
```

Out[4]:

| | a |
|---|---|
| **1** | 4 |
| **2** | 5 |
| **3** | 6 |

# Subset

In [10]:
```
df['a'] > 4
#이게 Series 형태로 나오니까 밑에줄에서 []한번으로 DataFrame 형태로 출력함
```

Out[10]:
```
1    False
2     True
3     True
4    False
Name: a, dtype: bool
```

In [8]:
```
df[df['a'] > 4]
```

Out[8]:

| | a | b | c |
|---|---|---|---|
| **2** | 5 | 8 | 11 |
| **3** | 6 | 9 | 12 |

In [8]:
```
#두개 이상의 값을 가져오려면 반드시 df형태로 가져와야 한다. 즉 list형태로 감싸줘야 함
df[['a','b']]
```

Out[8]:

| | a | b |
|---|---|---|
| **1** | 4 | 7 |
| **2** | 5 | 8 |
| **3** | 6 | 9 |

## Summarize Data

In [15]:
```
df["a"].value_counts()
```

Out[15]:
```
4    2
5    1
6    1
Name: a, dtype: int64
```

In [ ]:
```

```

## Reshaping

In [18]:
```
df["a"].sort_values()
# a컬럼 기준으로 정렬
```

Out[18]:
```
1    4
4    4
2    5
3    6
Name: a, dtype: int64
```

In [19]:
```
df.sort_values("a")
#전체에서 a값 기준으로 정렬
```

Out[19]:

| | a | b | c |
|---|---|---|---|
| **1** | 4 | 7 | 10 |
| **4** | 4 | 9 | 10 |
| **2** | 5 | 8 | 11 |
| **3** | 6 | 9 | 12 |

In [20]:
```
df.sort_values("a", ascending=False)
#a값 기준인데 역순으로 정렬
```

Out[20]:

| | a | b | c |
|---|---|---|---|
| **3** | 6 | 9 | 12 |
| **2** | 5 | 8 | 11 |
| **1** | 4 | 7 | 10 |
| **4** | 4 | 9 | 10 |

In [30]:

```
df = df.drop(["c"], axis=1)

# tab눌러보면 axis=0 으로 설정되어있는데 0은 행(row)이다. c는 컬럼이기 때문에 1로 바꿔
# 재대입 안해주면 그 순간에만 그렇게 출력돼고 원본은 안바뀜
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-30-348e7b261519> in <module>
----> 1 df = df.drop(["c"], axis=1)
      2
      3 # tab눌러보면 axis=0 으로 설정되어있는데 0은 행(row)이다. c는 컬럼이기
 때문에 1로 바꿔준다.
      4 # 재대입 안해주면 그 순간에만 그렇게 출력돼고 원본은 안바뀜

~Wanaconda3WlibWsite-packagesWpandasWcoreWframe.py in drop(self, labels, axi
s, index, columns, level, inplace, errors)
   4306                 weight  1.0     0.8
   4307         """
-> 4308         return super().drop(
   4309             labels=labels,
   4310             axis=axis,

~Wanaconda3WlibWsite-packagesWpandasWcoreWgeneric.py in drop(self, labels, a
xis, index, columns, level, inplace, errors)
   4151         for axis, labels in axes.items():
   4152             if labels is not None:
-> 4153                 obj = obj._drop_axis(labels, axis, level=level, errors=err
ors)
   4154
   4155         if inplace:

~Wanaconda3WlibWsite-packagesWpandasWcoreWgeneric.py in _drop_axis(self, labe
ls, axis, level, errors)
   4186                 new_axis = axis.drop(labels, level=level, errors=errors)
   4187             else:
-> 4188                 new_axis = axis.drop(labels, errors=errors)
   4189             result = self.reindex(**{axis_name: new_axis})
   4190

~Wanaconda3WlibWsite-packagesWpandasWcoreWindexesWbase.py in drop(self, labe
ls, errors)
   5589         if mask.any():
   5590             if errors != "ignore":
-> 5591                 raise KeyError(f"{labels[mask]} not found in axis")
   5592             indexer = indexer[~mask]
   5593         return self.delete(indexer)

KeyError: "['c'] not found in axis"
```

In [31]:

```
df
```

Out[31]:

| | a | b |
|---|---|---|
| **1** | 4 | 7 |

| | a | b |
|---|---|---|
| **2** | 5 | 8 |
| **3** | 6 | 9 |
| **4** | 4 | 9 |

# Group data

## Groupby, pivot_table

In [22]:
```
df.groupby(["a"])["b"].mean()
#a라는 컬럼값 기준으로 b컬럼의 평균을 구함
#a=4 일때 b= 7,9니까 7+9/2 = 8 이렇게
```

Out[22]:
```
a
4    8
5    8
6    9
Name: b, dtype: int64
```

In [20]:
```
df.groupby(["a"])["b"].mean()
```

Out[20]:
```
a
4    8
5    8
6    9
Name: b, dtype: int64
```

In [25]:
```
df.groupby(["a"])["b"].agg(["mean", "sum", "count"])
#agg 여러개를 나타내고 싶을 때 물론 단일로도 사용 가능. 위에 컬럼이 컬럼명을 쓸 수 있(
```

Out[25]:

| a | mean | sum | count |
|---|---|---|---|
| **4** | 8 | 16 | 2 |
| **5** | 8 | 8 | 1 |
| **6** | 9 | 9 | 1 |

In [26]:
```
df.groupby(["a"])["b"].describe()
```

Out[26]:

| a | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **4** | 2.0 | 8.0 | 1.414214 | 7.0 | 7.5 | 8.0 | 8.5 | 9.0 |
| **5** | 1.0 | 8.0 | NaN | 8.0 | 8.0 | 8.0 | 8.0 | 8.0 |
| **6** | 1.0 | 9.0 | NaN | 9.0 | 9.0 | 9.0 | 9.0 | 9.0 |

In [33]:
```
pd.pivot_table(df, index="a", values="b",aggfunc="sum")
```

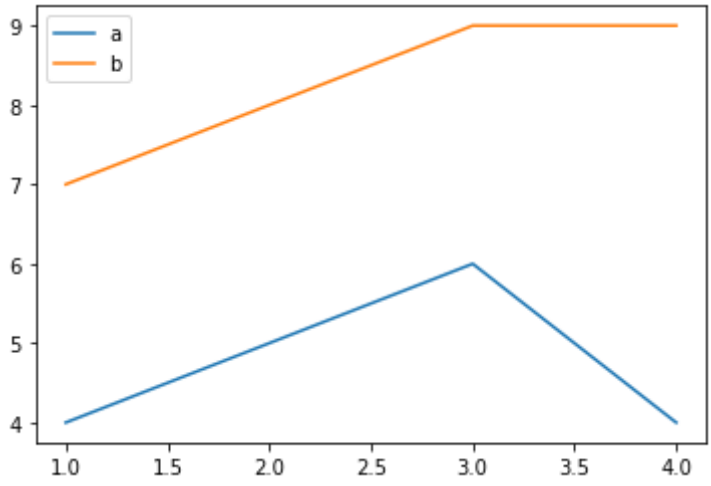
Out[33]:

| a | b |
|---|---|
| 4 | 16 |
| 5 | 8 |
| 6 | 9 |

# Plotting

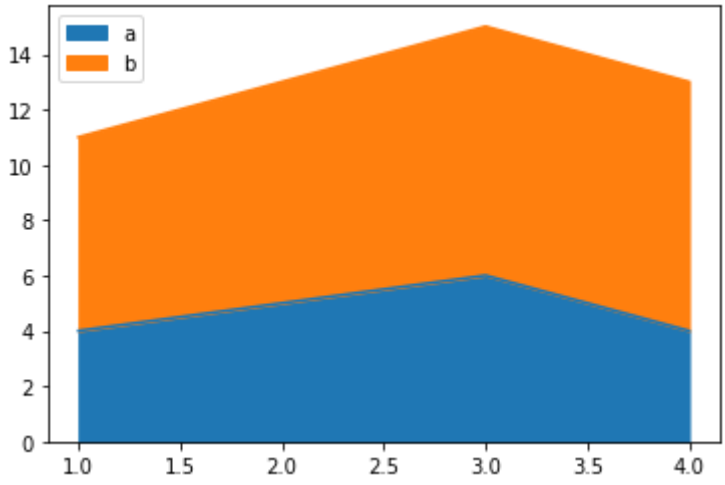
In [34]:
```
df.plot()
```

Out[34]: <AxesSubplot:>



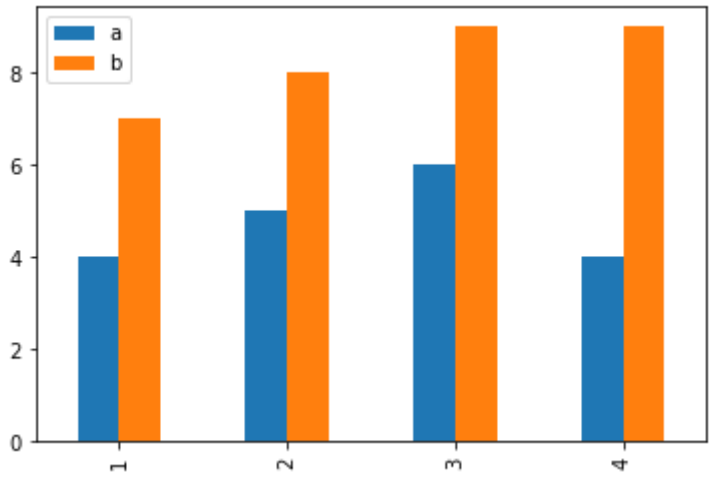

In [35]:
```
df.plot.area()
```

Out[35]: <AxesSubplot:>



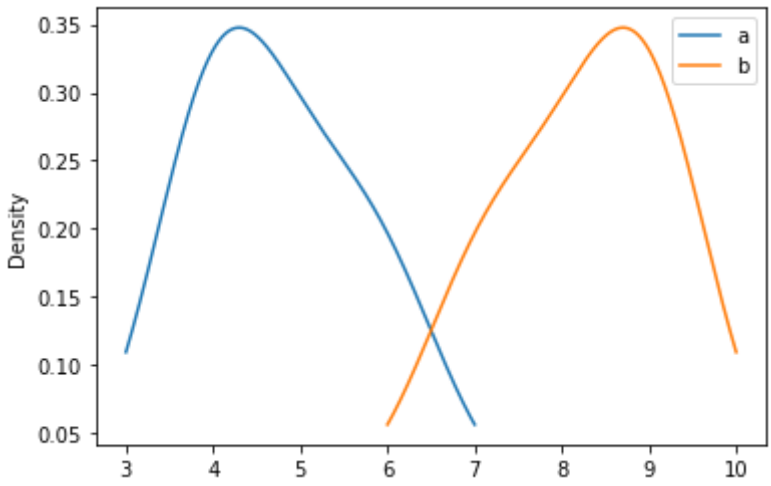

In [36]:
```
df.plot.bar()
```

Out[36]: <AxesSubplot:>




In [37]:
```
df.plot.density()
```

Out[37]: <AxesSubplot:ylabel='Density'>




In [ ]:
```

```