```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.linear_model import SGDClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt

import seaborn as sns


iris=load_iris()
df=pd.DataFrame(data=iris.data,columns=iris.feature_names)
df['target']=iris.target
print(df.head())
```

```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

   target
0       0
1       0
2       0
3       0
4       0
```

```
X = df.drop('target',axis=1)
X
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

Next steps: Generate code with X | View recommended plots | New interactive sheet

```
y=df['target']
y
```

What can I help you build?

```
target
```

```
X_train, X_test, y_train, y_test=train_test_split(X,y,test_size=0.2,random_state=42)
sgd_clf=SGDClassifier(max_iter=1000,tol=1e-3)
sgd_clf.fit(X_train,y_train)
y_pred=sgd_clf.predict(X_test)
```

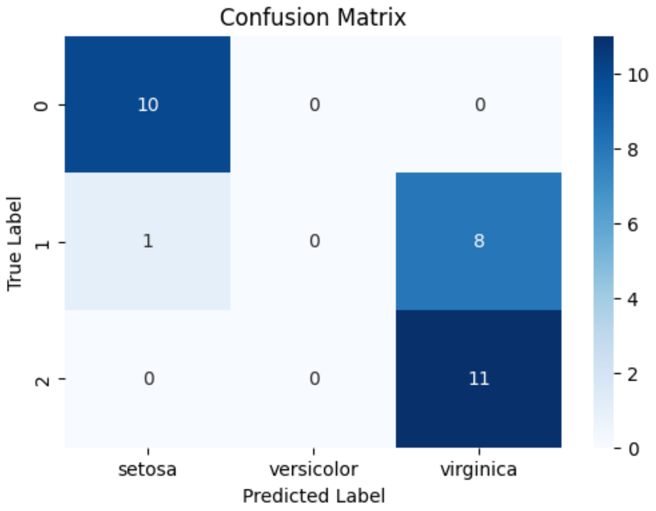
```
accuracy=accuracy_score(y_test,y_pred)
print(f"Accuracy: {accuracy:.3f}")
```

```
Accuracy: 0.700
145    2
```

```
cm=confusion_matrix(y_test,y_pred)
print("Confusion Matrix:")
print(cm)
```

```
Confusion Matrix:
[[10  0  0]
 [ 1  0  8]
 [ 0  0 11]]
149    2
150 rows × 1 columns
dtype: int64
```

```
plt.figure(figsize=(6,4))
sns.heatmap(cm,annot=True,cmap="Blues",fmt='d',xticklabels=iris.target_names)
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.show()
```



```
import pandas as pd
import matplotlib.pyplot as plt
```

```
data=pd.read_csv("/content/Mall_Customers (1).csv")
data.head()
```

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Next steps: Generate code with data View recommended plots New interactive sheet

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   CustomerID              200 non-null    int64 
 1   Gender                  200 non-null    object
 2   Age                     200 non-null    int64 
 3   Annual Income (k$)      200 non-null    int64 
 4   Spending Score (1-100)  200 non-null    int64 
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

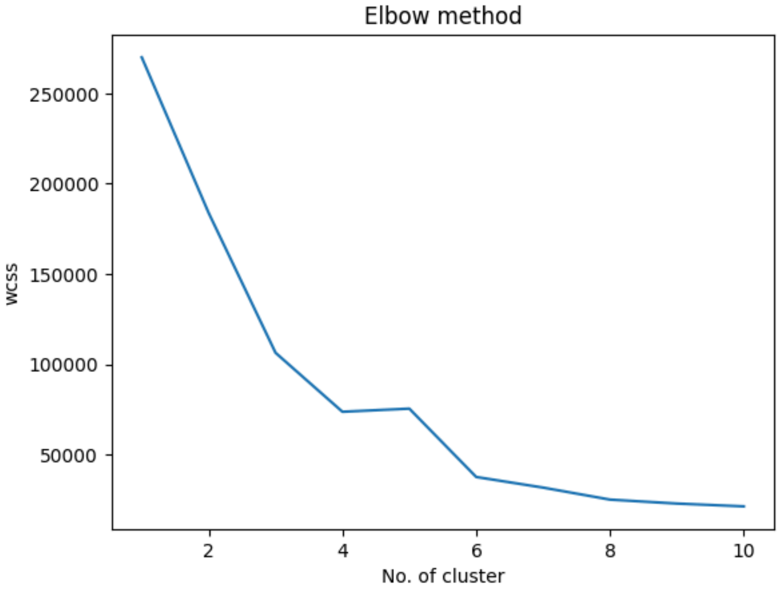
```
data.isnull().sum()
```

	0
CustomerID	0
Gender	0
Age	0
Annual Income (k$)	0
Spending Score (1-100)	0

dtype: int64

```
from sklearn.cluster import KMeans
wcss=[]
for i in range(1,11):
    kmeans=KMeans(n_clusters=i, init="k-means++")
    kmeans.fit(data.iloc[:,3:])
    wcss.append(kmeans.inertia_)
```

```
plt.plot(range(1,11),wcss)
plt.xlabel("No. of cluster")
plt.ylabel("wcss")
plt.title("Elbow method")
plt.show()
```

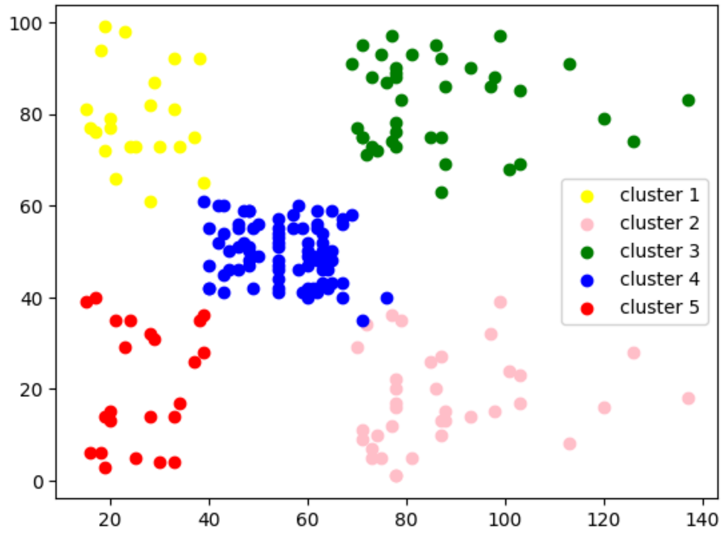


```
km=KMeans(n_clusters=5)
km.fit(data.iloc[:,3:])
y_pred=km.predict(data.iloc[:,3:])
y_pred
```

```
array([4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0,
       4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 3,
       4, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 2, 1, 2, 3, 2, 1, 2, 1, 2,
       1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 3, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2,
       1, 2], dtype=int32)
```

```
data["clusters"]=y_pred
df0=data[data["clusters"]==0]
df1=data[data["clusters"]==1]
df2=data[data["clusters"]==2]
df3=data[data["clusters"]==3]
df4=data[data["clusters"]==4]
```

```
plt.scatter(df0["Annual Income (k$)"],df0["Spending Score (1-100)"],color="yellow",label="cluster 1")
plt.scatter(df1["Annual Income (k$)"],df1["Spending Score (1-100)"],color="pink",label="cluster 2")
plt.scatter(df2["Annual Income (k$)"],df2["Spending Score (1-100)"],color="green",label="cluster 3")
plt.scatter(df3["Annual Income (k$)"],df3["Spending Score (1-100)"],color="blue",label="cluster 4")
plt.scatter(df4["Annual Income (k$)"],df4["Spending Score (1-100)"],color="red",label="cluster 5")
plt.legend()
plt.show()
```


cluster 1
cluster 2
cluster 3
cluster 4
cluster 5