In [347…
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [348…
```
df = pd.read_excel('animals.xlsx')
```

In [349…
```
df.head()
```

Out[349…

	Name	Blood_Temperature	Give_Birth	Can_Fly	Live_In_Water	Have_Le
0	Human	Warm	Yes	No	No	Y
1	Python	Cold	No	No	No	
2	Bat	Warm	Yes	Yes	No	Y
3	Frog	Cold	No	No	Sometimes	Y
4	Salmon	Cold	No	No	Yes	

In [350…
```
df['Live_In_Water'].unique()
```

Out[350…
```
array(['No', 'Sometimes', 'Yes'], dtype=object)
```

In [351…
```
from sklearn.preprocessing import OrdinalEncoder
order = ['No', 'Sometimes', 'Yes']
OE = OrdinalEncoder( categories = [order])
```

In [352…
```
df['Live_In_Water'] = OE.fit_transform(df[['Live_In_Water']])
```

In [353…
```
df.head()
```

Out[353…

	Name	Blood_Temperature	Give_Birth	Can_Fly	Live_In_Water	Have_Le
0	Human	Warm	Yes	No	0.0	Y
1	Python	Cold	No	No	0.0	
2	Bat	Warm	Yes	Yes	0.0	Y
3	Frog	Cold	No	No	1.0	Y
4	Salmon	Cold	No	No	2.0	

In [354…
```
order = ['No', 'Yes']
OE = OrdinalEncoder( categories = [order])
df['Can_Fly'] = OE.fit_transform(df[['Can_Fly']])
```

In [355…
```
order = ['No', 'Yes']
OE = OrdinalEncoder( categories = [order])
df['Give_Birth'] = OE.fit_transform(df[['Give_Birth']])
```

In [356…
```
df.head()
```

Out[356…

	Name	Blood_Temperature	Give_Birth	Can_Fly	Live_In_Water	Have_Le
0	Human	Warm	1.0	0.0	0.0	Y
1	Python	Cold	0.0	0.0	0.0	
2	Bat	Warm	1.0	1.0	0.0	Y
3	Frog	Cold	0.0	0.0	1.0	Y
4	Salmon	Cold	0.0	0.0	2.0	

In [357…
```
order = ['No', 'Yes']
OE = OrdinalEncoder( categories = [order])
df['Have_Legs'] = OE.fit_transform(df[['Have_Legs']])
df.head()
```

Out[357…

	Name	Blood_Temperature	Give_Birth	Can_Fly	Live_In_Water	Have_Le
0	Human	Warm	1.0	0.0	0.0	
1	Python	Cold	0.0	0.0	0.0	(
2	Bat	Warm	1.0	1.0	0.0	
3	Frog	Cold	0.0	0.0	1.0	
4	Salmon	Cold	0.0	0.0	2.0	(

In [358…
```
from sklearn.preprocessing import OneHotEncoder
```

In [359…
```
OHE = OneHotEncoder(handle_unknown = 'ignore', sparse_output = Fals
OHE_df = OHE.fit_transform(df[['Species']])
```

In [360…
```
df2 = pd.concat([df,OHE_df] ,axis = 1).drop('Species',axis = 1)
df2.head()
```

Out[360…

	Name	Blood_Temperature	Give_Birth	Can_Fly	Live_In_Water	Have_Le
0	Human	Warm	1.0	0.0	0.0	
1	Python	Cold	0.0	0.0	0.0	(
2	Bat	Warm	1.0	1.0	0.0	
3	Frog	Cold	0.0	0.0	1.0	
4	Salmon	Cold	0.0	0.0	2.0	(

In [361…
```
df = pd.read_csv('Superstore.csv',encoding = 'iso8859-1')
```

In [362…
```
df['Sales'].max()
```

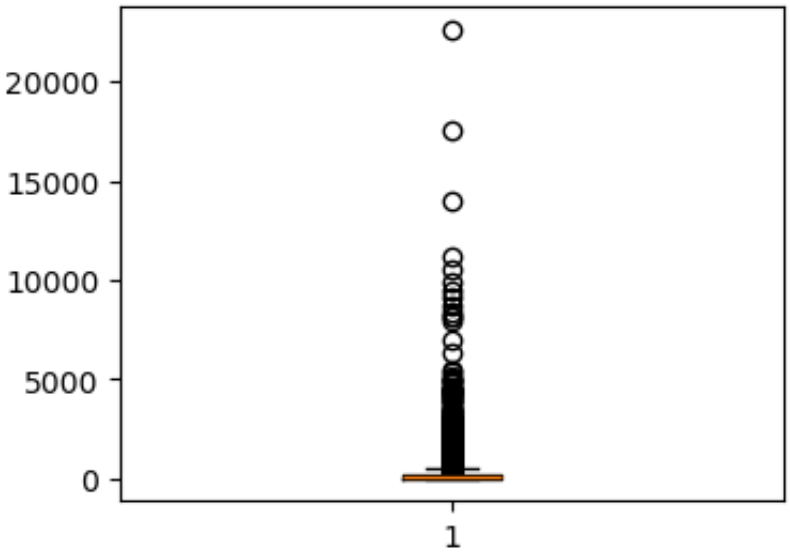
Out[362… 22638.48

In [363…
```
df['Sales'].mean()
```

Out[363…
```
229.85800083049833
```

In [364…
```
plt.figure(figsize = [4,3])
plt.boxplot(df['Sales'])
plt.show
```

Out[364…
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



z score = (x-xb) / sd

In [366…
```
"""
IQR = InterQuartile Range
Q1 = 0.25
Q3 = 0.75
IQE = Q3-Q1

"""
Q1 = df['Sales'].quantile(0.25)
Q3 = df['Sales'].quantile(0.75)
IQR = Q3-Q1
```

In [367…
```
lower_b = Q1 - IQR*2
upper_b = Q3 + IQR*2
```

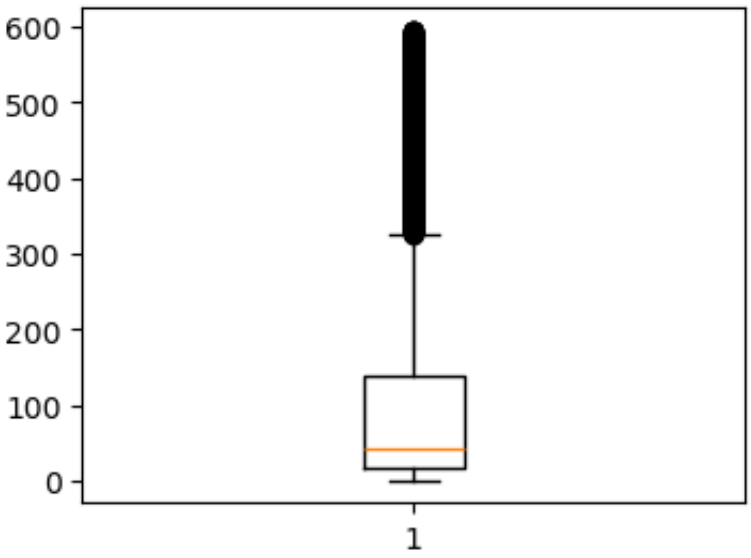
In [368…
```
df2 = df[(df['Sales']>=lower_b)&(df['Sales']<=upper_b)]
```

In [369…
```
df2['Sales'].mean()
```

Out[369…
```
103.39223432555052
```

In [370…
```
plt.figure(figsize = [4,3])
plt.boxplot(df2['Sales'])
plt.show
```

Out[370… <function matplotlib.pyplot.show(close=None, block=None)>



Classification

In [372…
```
df = pd.read_excel('shirt_size.xlsx')
df.head()
```

Out[372…

	Height_cms	Weight_kgs	T_Shirt_Size
0	158	58	M
1	158	59	M
2	160	64	L
3	163	64	L
4	165	61	L

In [373…
```
df['T_Shirt_Size'].unique()
```

Out[373… array(['M', 'L'], dtype=object)

In [374…
```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors = 5)
```

In [375…
```
# X = Att
# Y = Class
x = df[['Height_cms', 'Weight_kgs']]
y = df['T_Shirt_Size']
model.fit(x,y)
```

Out[375… KNeighborsClassifier

KNeighborsClassifier()

In [376…
```
y_pred = model.predict(X)
```

In [377…
```
pd.DataFrame({ 'Class':y.values , 'Predict' :y_pred })
```

Out[377…

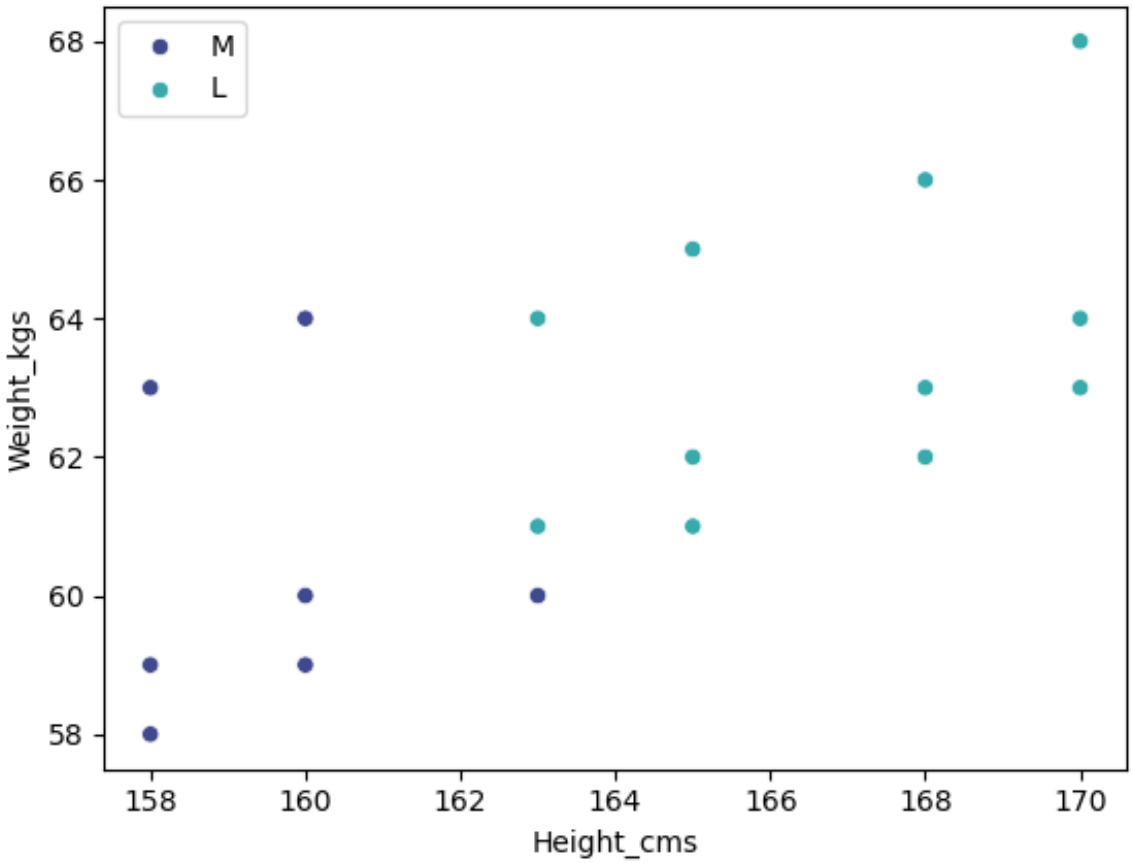
	Class	Predict
0	M	M
1	M	M
2	L	M
3	L	L
4	L	L
5	M	M
6	M	M
7	L	L
8	L	L
9	M	M
10	M	M
11	L	L
12	L	L
13	L	L
14	L	L
15	M	L
16	L	L
17	L	L

In [378…
```
from sklearn.metrics import accuracy_score
accuracy_score(y,y_pred)
```

Out[378… 0.8888888888888888

In [379…
```
sns.scatterplot(data = df, x = 'Height_cms', y = 'Weight_kgs', hue
```

Out[379… <Axes: xlabel='Height_cms', ylabel='Weight_kgs'>



In [380…
```
df = pd.read_excel('customer.xlsx')
```

In [381…
```
df.head(3)
```

Out[381…

	gender	age	salary	purchased
0	Male	19	19000	0
1	Male	35	20000	0
2	Female	26	43000	0

In [382…
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   gender     400 non-null    object
 1   age        400 non-null    int64
 2   salary     400 non-null    int64
 3   purchased  400 non-null    int64
dtypes: int64(3), object(1)
memory usage: 12.6+ KB
```

In [383…
```
df['purchased'].unique()
```

Out[383… array([0, 1])

In [384…
```
df['purchased'].value_counts()
```

Out[384…
```
purchased
0    257
1    143
Name: count, dtype: int64
```

In [385…
```
from sklearn.preprocessing import OrdinalEncoder
order = ['Male', 'Female']
OE = OrdinalEncoder( categories = [order] )
df['gender'] = OE.fit_transform(df[['gender']])
```

In [386…
```
df.head()
```

Out[386…

	gender	age	salary	purchased
0	0.0	19	19000	0
1	0.0	35	20000	0
2	1.0	26	43000	0
3	1.0	27	57000	0
4	0.0	19	76000	0

In [396…
```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors = 5)

from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
df2 = df.copy()
df2[['gender','age','salary']] = scaler.fit_transform(df2[['gender'
x = df2[['gender','age','salary']]
y = df2['purchased']
```

In [398…
```
df2.head()
```

Out[398…

	gender	age	salary	purchased
0	-1.020204	-1.781797	-1.490046	0
1	-1.020204	-0.253587	-1.460681	0
2	0.980196	-1.113206	-0.785290	0
3	0.980196	-1.017692	-0.374182	0
4	-1.020204	-1.781797	0.183751	0

In [400…
```
from sklearn.model_selection import train_test_split
```

In [402…
```
x_train, x_test, y_train, y_test = train_test_split(x,y,train_size
model.fit(x_train,y_train)
y_pred = model.predict(x_test)
```

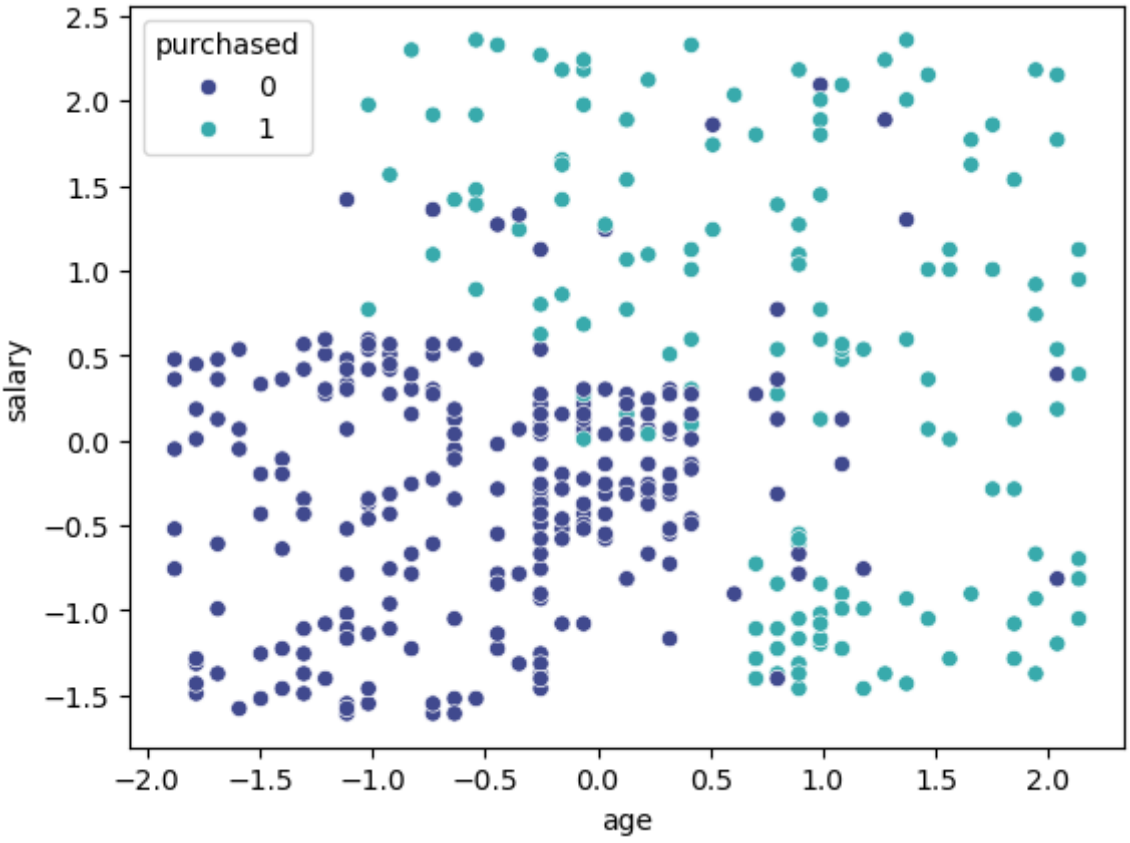
In [404…
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[404…
```
0.93
```

In [ ]:
```
print(pd.DataFrame({'Class':y_test.values , 'Predict': y_pred }).to
```

In [408…
```
sns.scatterplot(data = df2, x = 'age', y = 'salary', hue = y, palet
```

Out[408…
```
<Axes: xlabel='age', ylabel='salary'>
```



In [ ]: