In [572…
```python
# Import standard operational packages.
import numpy as np
import pandas as pd

# Important tools for modeling and evaluation.
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import StandardScaler

# Import visualization packages.
import matplotlib.pyplot as plt
import seaborn as sns
```

In [576…
```python
penguins = pd.read_csv(r'C:\Users\HP\Desktop\Advance Data Analyst\6. The nuts and bolts of ML\3. Module 3\2. Evaluate K means\Files\penguins.csv')
penguins.head(10)
```

Out[576…

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female
5	Adelie	Torgersen	39.3	20.6	190.0	3650.0	male
6	Adelie	Torgersen	38.9	17.8	181.0	3625.0	female
7	Adelie	Torgersen	39.2	19.6	195.0	4675.0	male
8	Adelie	Torgersen	34.1	18.1	193.0	3475.0	NaN
9	Adelie	Torgersen	42.0	20.2	190.0	4250.0	NaN

In [578…
```python
# Review the first 10 rows.

penguins.head(n = 10)
```

Out[578…

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female
5	Adelie	Torgersen	39.3	20.6	190.0	3650.0	male
6	Adelie	Torgersen	38.9	17.8	181.0	3625.0	female
7	Adelie	Torgersen	39.2	19.6	195.0	4675.0	male
8	Adelie	Torgersen	34.1	18.1	193.0	3475.0	NaN
9	Adelie	Torgersen	42.0	20.2	190.0	4250.0	NaN

In [580…
```python
# Find out how many penguin types there are.

penguins['species'].unique()
```

Out[580…
```
array(['Adelie', 'Chinstrap', 'Gentoo'], dtype=object)
```

In [582…
```python
# Find the count of each species type.

penguins['species'].value_counts(dropna = False)
```

Out[582…
```
species
Adelie       152
Gentoo       124
Chinstrap     68
Name: count, dtype: int64
```

In [584…
```python
# Check for missing values.

penguins.isnull().sum()
```

Out[584…
```
species               0
island                0
bill_length_mm        2
bill_depth_mm         2
flipper_length_mm     2
body_mass_g           2
sex                  11
dtype: int64
```

In [586…
```python
# Drop rows with missing values.
# Save DataFrame in variable `penguins_subset`.

penguins_subset = penguins.dropna(axis=0).reset_index(drop = True)
```

In [588…
```python
# Check for missing values.

penguins_subset.isna().sum()
```

Out[588…
```
species              0
island               0
bill_length_mm       0
bill_depth_mm        0
flipper_length_mm    0
body_mass_g          0
sex                  0
dtype: int64
```

In [590…
```python
# View first 10 rows.

penguins_subset.head(10)
```

Out[590…

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female
3	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female
4	Adelie	Torgersen	39.3	20.6	190.0	3650.0	male
5	Adelie	Torgersen	38.9	17.8	181.0	3625.0	female
6	Adelie	Torgersen	39.2	19.6	195.0	4675.0	male
7	Adelie	Torgersen	41.1	17.6	182.0	3200.0	female
8	Adelie	Torgersen	38.6	21.2	191.0	3800.0	male
9	Adelie	Torgersen	34.6	21.1	198.0	4400.0	male

In [592…
```python
penguins_subset['sex'] = penguins_subset['sex'].str.upper()
```

In [594…
```python
# Convert `sex` column from categorical to numeric.

penguins_subset = pd.get_dummies(penguins_subset, drop_first = True, columns=['sex'])
```

In [596…
```python
# Drop the island column.

penguins_subset = penguins_subset.drop(['island'], axis=1)
```

In [598…
```python
# Exclude `species` variable from X

X = penguins_subset.drop(['species'], axis=1)
```

In [600…
```python
#Scale the features.
#Assign the scaled data to variable `X_scaled`.

X_scaled = StandardScaler().fit_transform(X)
```

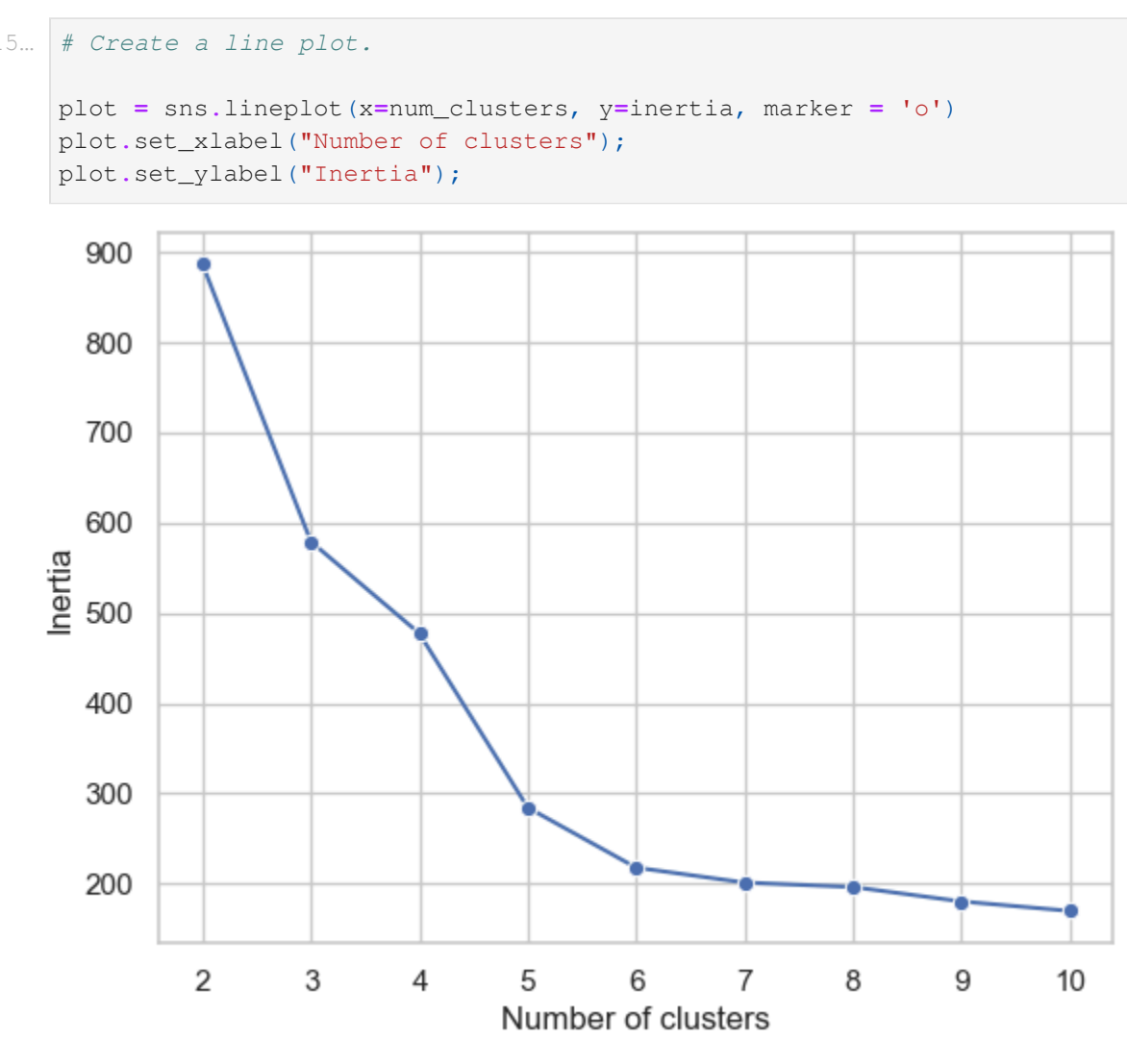
In [602…
```python
# Fit K-means and evaluate inertia for different values of k.

num_clusters = [i for i in range(2, 11)]

def kmeans_inertia(num_clusters, x_vals):
    """
    Accepts as arguments list of ints and data array.
    Fits a KMeans model where k = each value in the list of ints.
    Returns each k-value's inertia appended to a list.
    """
    inertia = []
    for num in num_clusters:
        kms = KMeans(n_clusters=num, random_state=42)
        kms.fit(x_vals)
        inertia.append(kms.inertia_)

    return inertia
```

In [614…
```python
# Return a list of inertia for k=2 to 10.

inertia = kmeans_inertia(num_clusters, X_scaled)
inertia
```

```
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
```

Out[614…
```
[885.622414365225,
 578.8284278107235,
 477.22956735281946,
 284.5464837898288,
 218.05156297830223,
 201.51585567445812,
 196.58908645819537,
 180.65751845628574,
 170.15703589755498]
```

In [615…
```python
# Create a line plot.

plot = sns.lineplot(x=num_clusters, y=inertia, marker = 'o')
plot.set_xlabel("Number of clusters");
plot.set_ylabel("Inertia");
```

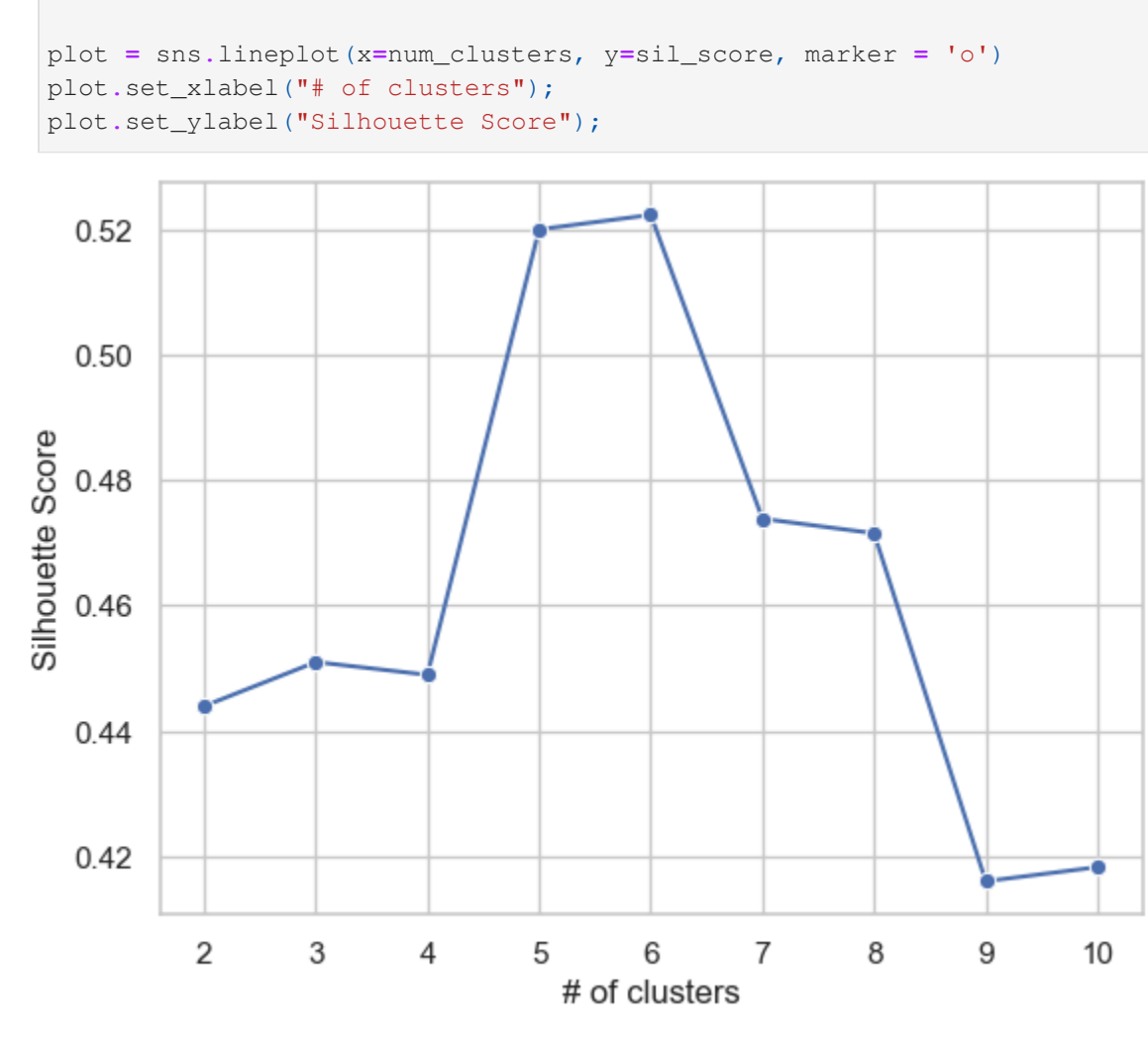


In [630…
```python
# Evaluate silhouette score.
# Write a function to return a list of each k-value's score.

def kmeans_sil(num_clusters, x_vals):
    """
    Accepts as arguments list of ints and data array.
    Fits a KMeans model where k = each value in the list of ints.
    Calculates a silhouette score for each k value.
    Returns each k-value's silhouette score appended to a list.
    """
    sil_score = []
    for num in num_clusters:
        kms = KMeans(n_clusters=num, random_state=42)
        kms.fit(x_vals)
        sil_score.append(silhouette_score(x_vals, kms.labels_))

    return sil_score


sil_score = kmeans_sil(num_clusters, X_scaled)
sil_score
```

```
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
```

Out[630…
```
[0.44398088353055243,
 0.45101024097188364,
 0.4489699212061027,
 0.519998574860868,
 0.5223086008347773,
 0.47386350642293157,
 0.4715443426463867,
 0.4160561489496056,
 0.418306343369105]
```

In [620…
```python
# Create a line plot.

plot = sns.lineplot(x=num_clusters, y=sil_score, marker = 'o')
plot.set_xlabel("# of clusters");
plot.set_ylabel("Silhouette Score");
```



In [632…
```python
# Fit a 6-cluster model.

kmeans6 = KMeans(n_clusters=6, random_state=42)
kmeans6.fit(X_scaled)
```

```
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=2.
  warnings.warn(
```

Out[632…
```
KMeans
KMeans(n_clusters=6, random_state=42)
```

In [634…
```python
# Print unique labels.

print('Unique labels:', np.unique(kmeans6.labels_))
```

```
Unique labels: [0 1 2 3 4 5]
```

In [636…
```python
# Create a new column `cluster`.

penguins_subset['cluster'] = kmeans6.labels_
penguins_subset.head()
```

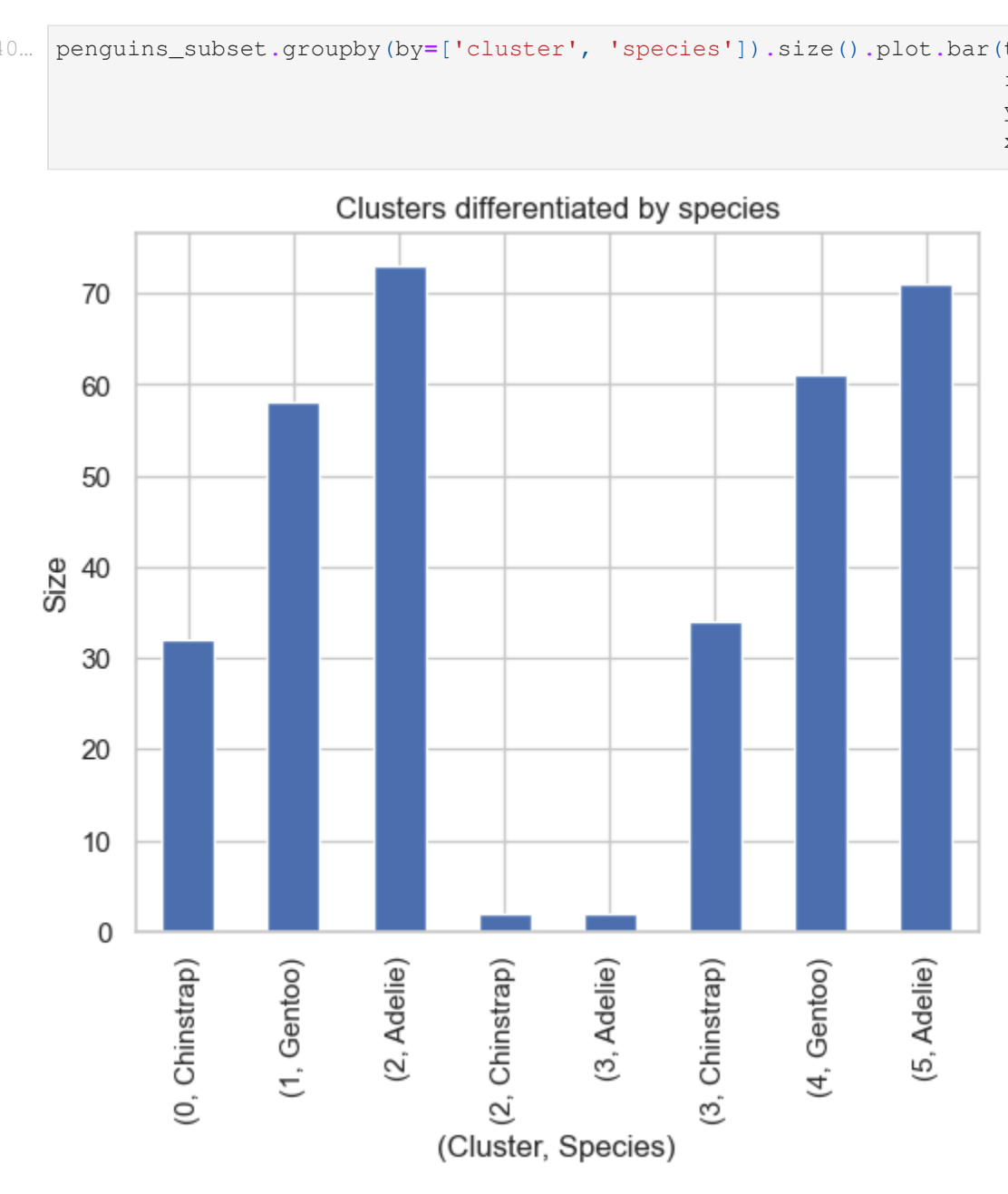
Out[636…

	species	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex_MALE	cluster
0	Adelie	39.1	18.7	181.0	3750.0	True	5
1	Adelie	39.5	17.4	186.0	3800.0	False	2
2	Adelie	40.3	18.0	195.0	3250.0	False	2
3	Adelie	36.7	19.3	193.0	3450.0	False	2
4	Adelie	39.3	20.6	190.0	3650.0	True	5

In [638…
```python
# Verify if any `cluster` can be differentiated by `species`.

penguins_subset.groupby(by=['cluster', 'species']).size()
```

Out[638…
```
cluster  species
0        Chinstrap    32
1        Gentoo       58
2        Adelie       73
         Chinstrap     2
3        Adelie        2
         Chinstrap    34
4        Gentoo       61
5        Adelie       71
dtype: int64
```

In [640…
```python
penguins_subset.groupby(by=['cluster', 'species']).size().plot.bar(title='Clusters differentiated by species',
                                                                   figsize=(6, 5),
                                                                   ylabel='Size',
                                                                   xlabel='(Cluster, Species)');
```

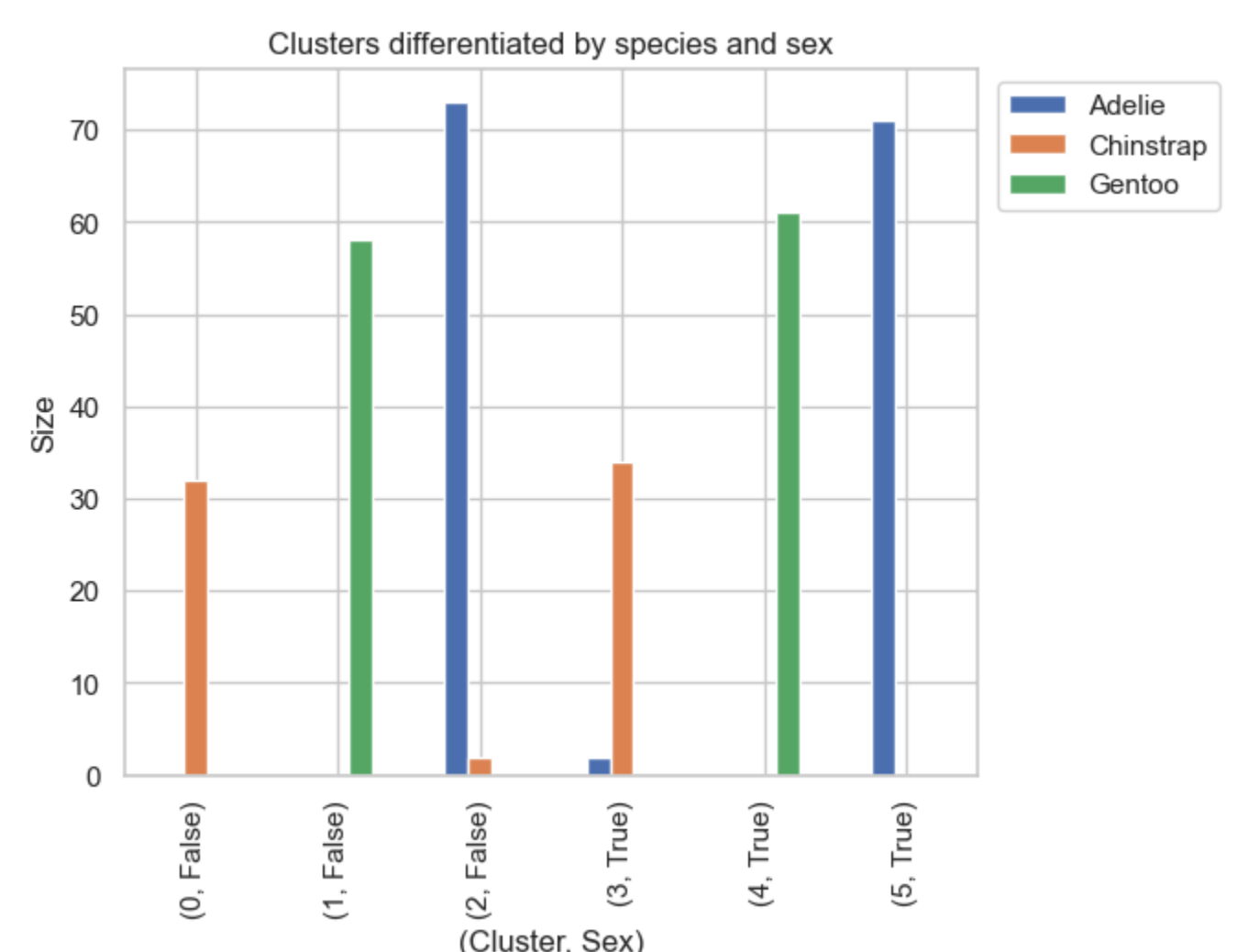


In [642…
```python
# Verify if each `cluster` can be differentiated by `species` AND `sex_MALE`.

penguins_subset.groupby(by=['cluster','species', 'sex_MALE']).size().sort_values(ascending = False)
```

Out[642…
```
cluster  species    sex_MALE
2        Adelie     False       73
5        Adelie     True        71
4        Gentoo     True        61
1        Gentoo     False       58
3        Chinstrap  True        34
0        Chinstrap  False       32
2        Chinstrap  False        2
3        Adelie     True         2
dtype: int64
```

In [644…
```python
penguins_subset.groupby(by=['cluster','species','sex_MALE']).size().unstack(level = 'species', fill_value=0).plot.bar(title='Clusters differentiated by species and sex',
                                                                                                                      figsize=(6, 5),
                                                                                                                      ylabel='Size',
                                                                                                                      xlabel='(Cluster, Sex)')
plt.legend(bbox_to_anchor=(1.3, 1.0))
```

Out[644…
```
<matplotlib.legend.Legend at 0x27bf5a631a0>
```



In [ ]:
```python
# key takeaways

# Many machine learning workflows are about cleaning, encoding, and scaling data.
# Inertia and silhouette score can be used to find the optimal value of clusters.
# Clusters can find natural groupings in data.
# The clusters in this lab are mostly differentiated by species and sex as shown by the groupby results and corresponding graphs.
# The elbow plot and especially the silhouette scores suggests that 6 clusters are optimal for this data.
# Having 6 clusters makes sense because the study suggests that there is sexual dimorphism (differences between the sexes) for each of the three species (2 sexes * 3 different species = 6 clusters).
```

```
# What summary would you provide to stakeholders?
#
# The K-means clustering enabled this data to be effectively grouped. It helped identify patterns that can educate team members about prejudice.
# The success of the cluster results suggests that the organization can apply clustering to other projects and continue augmenting employee education.
```