Import Required Packages for EDA

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
import plotly.graph_objects as go
import plotly.express as px
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

Read the dataset/s

```
penguin_df=pd.read_csv('/penguins_size.csv')
penguin_df
```

	species	island	culmen_length_mm	culmen_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
...	...	...	...	...	...	...	...
339	Gentoo	Biscoe	NaN	NaN	NaN	NaN	NaN
340	Gentoo	Biscoe	46.8	14.3	215.0	4850.0	FEMALE
341	Gentoo	Biscoe	50.4	15.7	222.0	5750.0	MALE
342	Gentoo	Biscoe	45.2	14.8	212.0	5200.0	FEMALE
343	Gentoo	Biscoe	49.9	16.1	213.0	5400.0	MALE

344 rows × 7 columns

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

Checking description(first 5 and last 5 rows)

```
penguin_df.head()
```

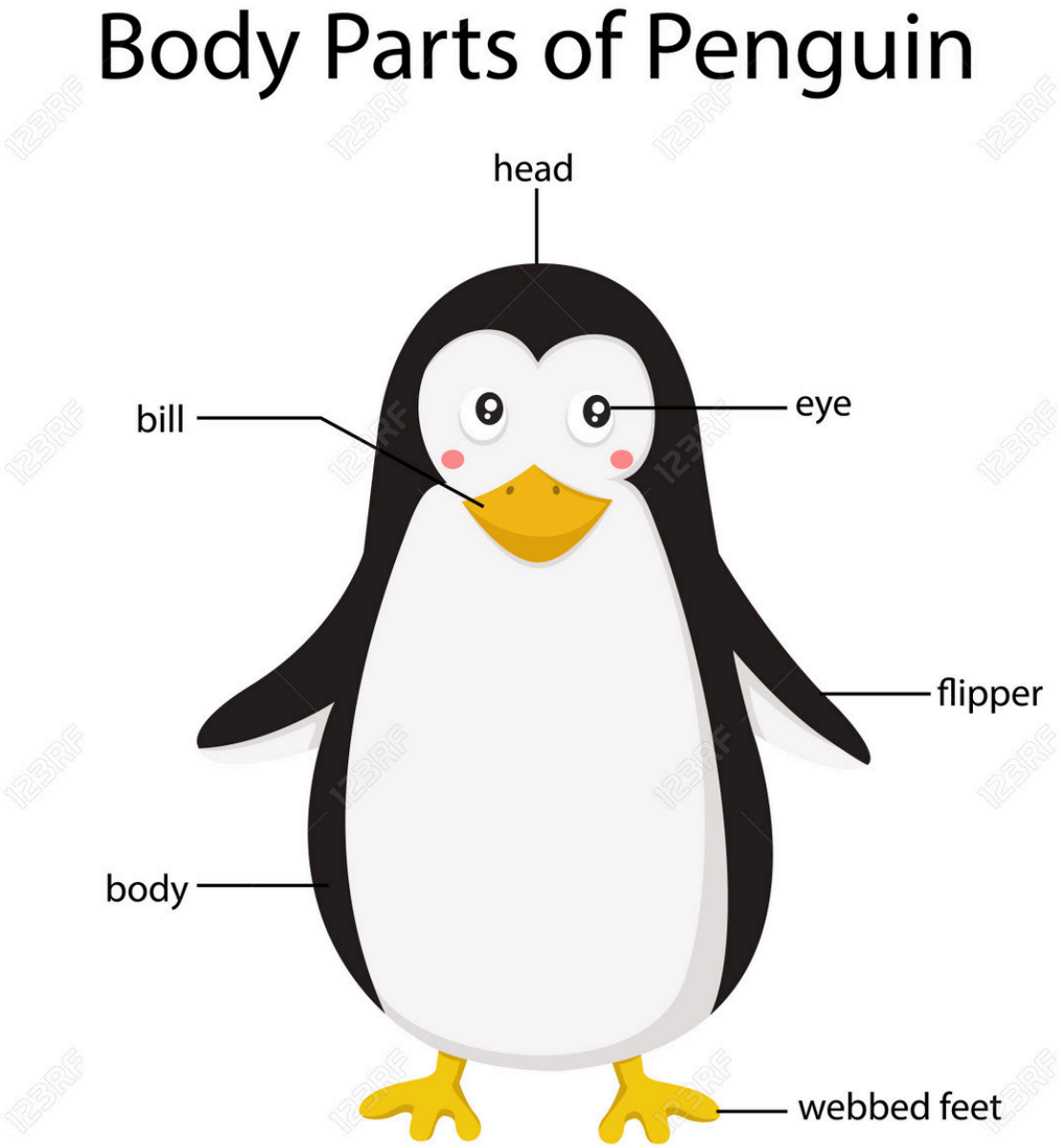
	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	MALE
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	FEMALE
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	FEMALE
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	FEMALE

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

```
penguin_df.shape
```

```
(344, 7)
```

Describing the data



Columns in the dataset

Species: penguin species (Chinstrap, Adélie, or Gentoo)

Island: island name (Dream, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica)

culmen_length_mm: culmen length (mm)

culmen_depth_mm: culmen depth (mm)

flipper_length_mm: flipper length (mm)

body_mass_g: body mass (g)

Sex: penguin sex

What is culmen?

The upper margin of the beak or bill is referred to as the culmen and the measurement is taken using calipers with one jaw at the tip of the upper mandible and the other at base of the skull or the first feathers depending on the standard chosen.

Name of the attributes

```
penguin_df.columns
```

```
Index(['species', 'island', 'culmen_length_mm', 'culmen_depth_mm',
       'flipper_length_mm', 'body_mass_g', 'sex'],
      dtype='object')
```

Unique values for each attribute

```
penguin_df.nunique()
```

	0
species	3
island	3
culmen_length_mm	164
culmen_depth_mm	80
flipper_length_mm	55
body_mass_g	94
sex	3

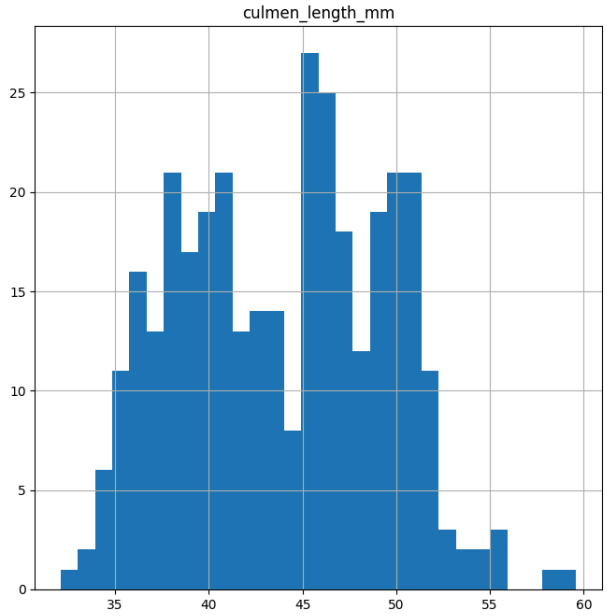
Double-click (or enter) to edit

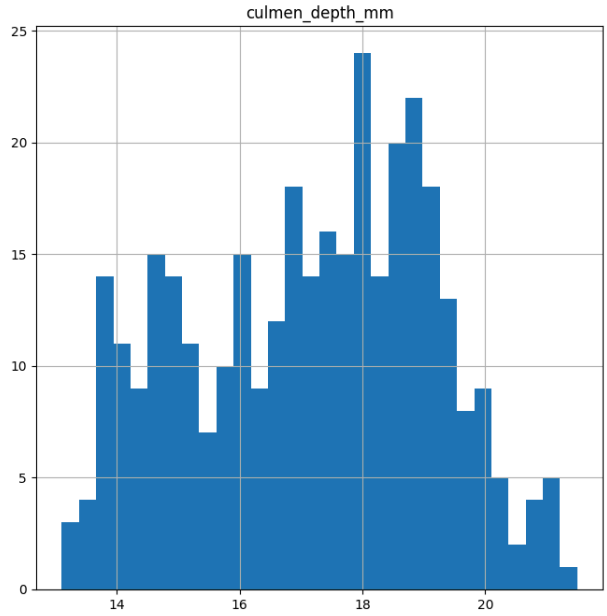
```
penguin_df.info()
```

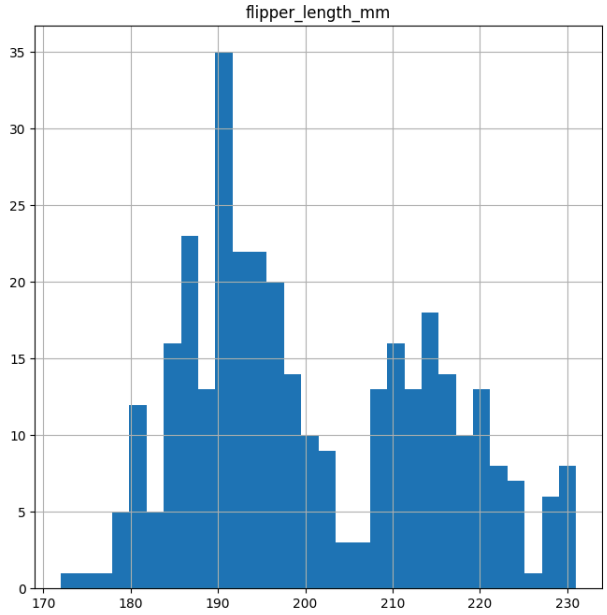
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   species            344 non-null    object
 1   island             344 non-null    object
 2   culmen_length_mm   342 non-null    float64
 3   culmen_depth_mm    342 non-null    float64
 4   flipper_length_mm  342 non-null    float64
 5   body_mass_g        342 non-null    float64
 6   sex                334 non-null    object
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
```

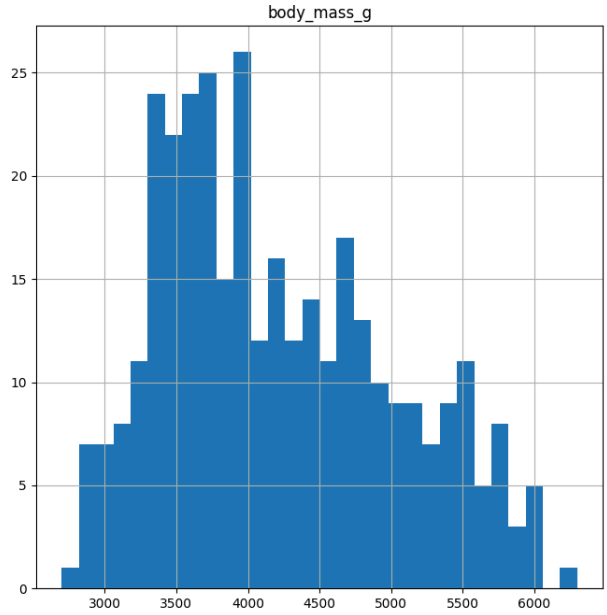
Visualising data distribution in detail

```
fig = plt.figure(figsize =(18,18))
ax=fig.gca()
penguin_df.hist(ax=ax,bins =30)
plt.show()
```





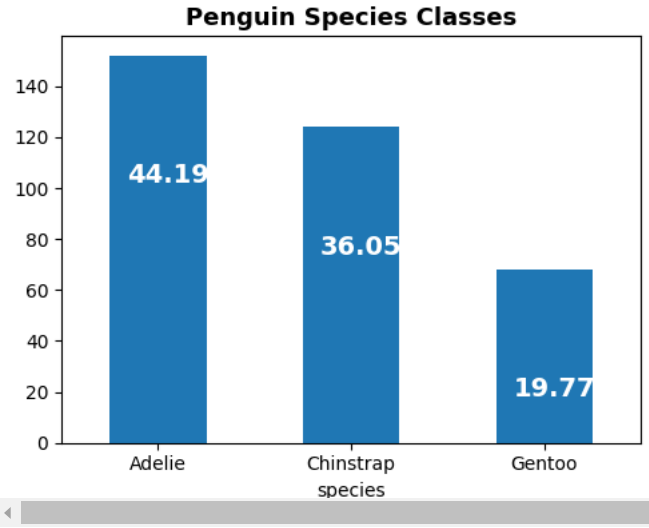




checking target value distribution

```
print(penguin_df.species.value_counts())
fig, ax = plt.subplots(figsize=(5,4))
name = ["Adelie", "Chinstrap", "Gentoo"]
ax = penguin_df.species.value_counts().plot(kind='bar')
ax.set_title("Penguin Species Classes", fontsize = 13, weight = 'bold')
ax.set_xticklabels (name, rotation = 0)

# To calculate the percentage
totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x()+.09, i.get_height()-50, \
            str(round((i.get_height()/total)*100, 2))+'%', fontsize=14,
                color='white', weight = 'bold')

plt.tight_layout()
```

```
species
Adelie       152
Gentoo       124
Chinstrap     68
Name: count, dtype: int64
```




```
!pip install https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
```

```
irement already satisfied: seaborn<0.14,>=0.10.1 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (0.1
ecting multimethod<2,>=1.4 (from ydata-profiling==0.0.dev0)
wnloading multimethod-1.12-py3-none-any.whl.metadata (9.6 kB)
irement already satisfied: statsmodels<1,>=0.13.2 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (0.
irement already satisfied: typeguard<5,>=3 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (4.3.0)
ecting imagehash==4.3.1 (from ydata-profiling==0.0.dev0)
wnloading ImageHash-4.3.1-py2.py3-none-any.whl.metadata (8.0 kB)
irement already satisfied: wordcloud>=1.9.3 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (1.9.3)
ecting dacite>=1.8 (from ydata-profiling==0.0.dev0)
wnloading dacite-1.8.1-py3-none-any.whl.metadata (15 kB)
irement already satisfied: numba<1,>=0.56.0 in /usr/local/lib/python3.10/dist-packages (from ydata-profiling==0.0.dev0) (0.60.0)
ecting PyWavelets (from imagehash==4.3.1->ydata-profiling==0.0.dev0)
wnloading pywavelets-1.7.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.0 kB)
irement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (from imagehash==4.3.1->ydata-profiling==0.0.dev0) (
irement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2<3.2,>=2.11.1->ydata-profiling=
irement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.10,>=3.5->ydata-profili
irement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.10,>=3.5->ydata-profiling==
irement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.10,>=3.5->ydata-profil
irement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.10,>=3.5->ydata-profil
irement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.10,>=3.5->ydata-profilin
irement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.10,>=3.5->ydata-profili
irement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib<3.10,>=3.5->ydata-pro
irement already satisfied: llvmlite<0.44,>=0.43.0dev0 in /usr/local/lib/python3.10/dist-packages (from numba<1,>=0.56.0->ydata-pr
irement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas!=1.4.0,<3,>1.1->ydata-profiling==
irement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas!=1.4.0,<3,>1.1->ydata-profiling
irement already satisfied: joblib>=0.14.1 in /usr/local/lib/python3.10/dist-packages (from phik<0.13,>=0.11.1->ydata-profiling==0
irement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.10/dist-packages (from pydantic>=2->ydata-profiling==
irement already satisfied: pydantic-core==2.23.4 in /usr/local/lib/python3.10/dist-packages (from pydantic>=2->ydata-profiling==0
irement already satisfied: typing-extensions>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from pydantic>=2->ydata-profiling
irement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.24.0->ydata-p
irement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.24.0->ydata-profiling==0.
irement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.24.0->ydata-profili
irement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.24.0->ydata-profili
irement already satisfied: patsy>=0.5.6 in /usr/local/lib/python3.10/dist-packages (from statsmodels<1,>=0.13.2->ydata-profiling=
irement already satisfied: attrs>=19.3.0 in /usr/local/lib/python3.10/dist-packages (from visions<0.7.7,>=0.7.5->visions[type_ima
irement already satisfied: networkx>=2.4 in /usr/local/lib/python3.10/dist-packages (from visions<0.7.7,>=0.7.5->visions[type_ima
irement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.6->statsmodels<1,>=0.13.2->ydata-profi
loading ImageHash-4.3.1-py2.py3-none-any.whl (296 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 296.5/296.5 kB 6.6 MB/s eta 0:00:00
loading dacite-1.8.1-py3-none-any.whl (14 kB)
loading multimethod-1.12-py3-none-any.whl (10 kB)
loading phik-0.12.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (686 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 686.1/686.1 kB 24.7 MB/s eta 0:00:00
loading visions-0.7.6-py3-none-any.whl (104 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 104.8/104.8 kB 7.5 MB/s eta 0:00:00
loading pywavelets-1.7.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.5 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.5/4.5 MB 70.4 MB/s eta 0:00:00
ding wheels for collected packages: ydata-profiling, htmlmin
ilding wheel for ydata-profiling (setup.py) ... done
eated wheel for ydata-profiling: filename=ydata_profiling-0.0.dev0-py2.py3-none-any.whl size=356222 sha256=e3fc42af59f9beccd9099c
ored in directory: /tmp/pip-ephem-wheel-cache-i_0_ptp3/wheels/07/29/61/f533cc7cbd0a97efb2d1b94d3254a3e859a949367ba842577b
ilding wheel for htmlmin (setup.py) ... done
eated wheel for htmlmin: filename=htmlmin-0.1.12-py3-none-any.whl size=27081 sha256=abd35d66373ff0bdf4b1b6050d309ce681ffab45b2e14
ored in directory: /root/.cache/pip/wheels/dd/91/29/a79cecb328d01739e64017b6fb9a1ab9d8cb1853098ec5966d
essfully built ydata-profiling htmlmin
alling collected packages: htmlmin, PyWavelets, multimethod, dacite, imagehash, visions, phik, ydata-profiling
essfully installed PyWavelets-1.7.0 dacite-1.8.1 htmlmin-0.1.12 imagehash-4.3.1 multimethod-1.12 phik-0.12.4 visions-0.7.6 ydata-
```

```
#obtain full profiler report
#restart kernel
#re-run import libraries and data
import pandas as pd
import numpy as np
from pandas_profiling import ProfileReport
profile = ProfileReport(penguin_df,title="Penguin Species EDA",
                        html={'style':{'full_width':True}})
profile.to_notebook_iframe()
```

Summarize dataset: 100% 33/33 [00:07<00:00, 3.38it/s, Completed]

Generate report structure: 100% 1/1 [00:05<00:00, 5.85s/it]

Render HTML: 100% 1/1 [00:00<00:00, 1.08it/s]

13.2	1	0.3%
13.3	1	0.3%
13.4	1	0.3%
13.5	2	0.6%
13.6	1	0.3%
13.7	6	1.7%
13.8	4	1.2%
13.9	4	1.2%
14	2	0.6%

Value	Count	Frequency (%)
21.5	1	0.3%
21.2	2	0.6%
21.1	3	0.9%
20.8	1	0.3%
20.7	3	0.9%
20.6	1	0.3%
20.5	1	0.3%
20.3	3	0.9%
20.2	1	0.3%
20.1	1	0.3%

```
#pre-processing
from sklearn.exceptions import DataDimensionalityWarning
#encode object columns to integers
from sklearn import preprocessing
from sklearn.preprocessing import OrdinalEncoder

for col in penguin_df:
  if penguin_df[col].dtype =='object':
    penguin_df[col]=OrdinalEncoder().fit_transform(penguin_df[col].values.reshape(-1,1))
penguin_df
```

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	0.0	2.0	39.1	18.7	181.0	3750.0	2.0
1	0.0	2.0	39.5	17.4	186.0	3800.0	1.0
2	0.0	2.0	40.3	18.0	195.0	3250.0	1.0
3	0.0	2.0	NaN	NaN	NaN	NaN	NaN
4	0.0	2.0	36.7	19.3	193.0	3450.0	1.0
...	...	...	...	...	...	...	...
339	2.0	0.0	NaN	NaN	NaN	NaN	NaN
340	2.0	0.0	46.8	14.3	215.0	4850.0	1.0
341	2.0	0.0	50.4	15.7	222.0	5750.0	2.0
342	2.0	0.0	45.2	14.8	212.0	5200.0	1.0
343	2.0	0.0	49.9	16.1	213.0	5400.0	2.0

344 rows × 7 columns

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

```
class_label =penguin_df['species']
penguin_df = penguin_df.drop(['species'], axis =1)
penguin_df = (penguin_df-penguin_df.min())/(penguin_df.max()-penguin_df.min())
penguin_df['species']=class_label
penguin_df
```

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex	species
0	1.0	0.254545	0.666667	0.152542	0.291667	1.0	0.0
1	1.0	0.269091	0.511905	0.237288	0.305556	0.5	0.0
2	1.0	0.298182	0.583333	0.389831	0.152778	0.5	0.0
3	1.0	NaN	NaN	NaN	NaN	NaN	0.0
4	1.0	0.167273	0.738095	0.355932	0.208333	0.5	0.0
...	...	...	...	...	...	...	...
339	0.0	NaN	NaN	NaN	NaN	NaN	2.0
340	0.0	0.534545	0.142857	0.728814	0.597222	0.5	2.0
341	0.0	0.665455	0.309524	0.847458	0.847222	1.0	2.0
342	0.0	0.476364	0.202381	0.677966	0.694444	0.5	2.0
343	0.0	0.647273	0.357143	0.694915	0.750000	1.0	2.0

344 rows × 7 columns

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

```
# Pre-processing
penguin_data = penguin_df.copy()
le = preprocessing.LabelEncoder()

# Encoding the categorical and numerical columns
island = le.fit_transform(list(penguin_data["island"]))
gender = le.fit_transform(list(penguin_data["sex"]))
culmen_length = le.fit_transform(list(penguin_data["culmen_length_mm"]))
culmen_depth = le.fit_transform(list(penguin_data["culmen_depth_mm"]))
flipper_length = le.fit_transform(list(penguin_data["flipper_length_mm"]))
body_mass = le.fit_transform(list(penguin_data["body_mass_g"]))
# Note: Remove or update 'year' as the dataset provided doesn't include a 'year' column
#year = le.fit_transform(list(penguin_data["year"]))
species = le.fit_transform(list(penguin_data["species"]))
```

Predictive analytics model development by comparing different Scikit-learn classification algorithms

```
import sklearn
from sklearn.preprocessing import StandardScaler
```

```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn.metrics import accuracy_score
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
```

```
# Pre-processing
penguin_data = penguin_df.copy()
le = preprocessing.LabelEncoder()

# Encoding the categorical and numerical columns
island = le.fit_transform(list(penguin_data["island"]))
gender = le.fit_transform(list(penguin_data["sex"]))
culmen_length = le.fit_transform(list(penguin_data["culmen_length_mm"]))
culmen_depth = le.fit_transform(list(penguin_data["culmen_depth_mm"]))
flipper_length = le.fit_transform(list(penguin_data["flipper_length_mm"]))
body_mass = le.fit_transform(list(penguin_data["body_mass_g"]))
ps = le.fit_transform(list(penguin_data["species"]))

# Create features (X) and labels (y)
x = list(zip(island, gender, culmen_length, culmen_depth, flipper_length, body_mass))
y = list(ps)

# Test options and evaluation metric
num_folds = 5
seed = 7
scoring = 'accuracy'

# Splitting data into training and testing sets
x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(x, y, test_size=0.20, random_state=seed)

# Output the size of the training and testing subsets
np.shape(x_train), np.shape(x_test)
```

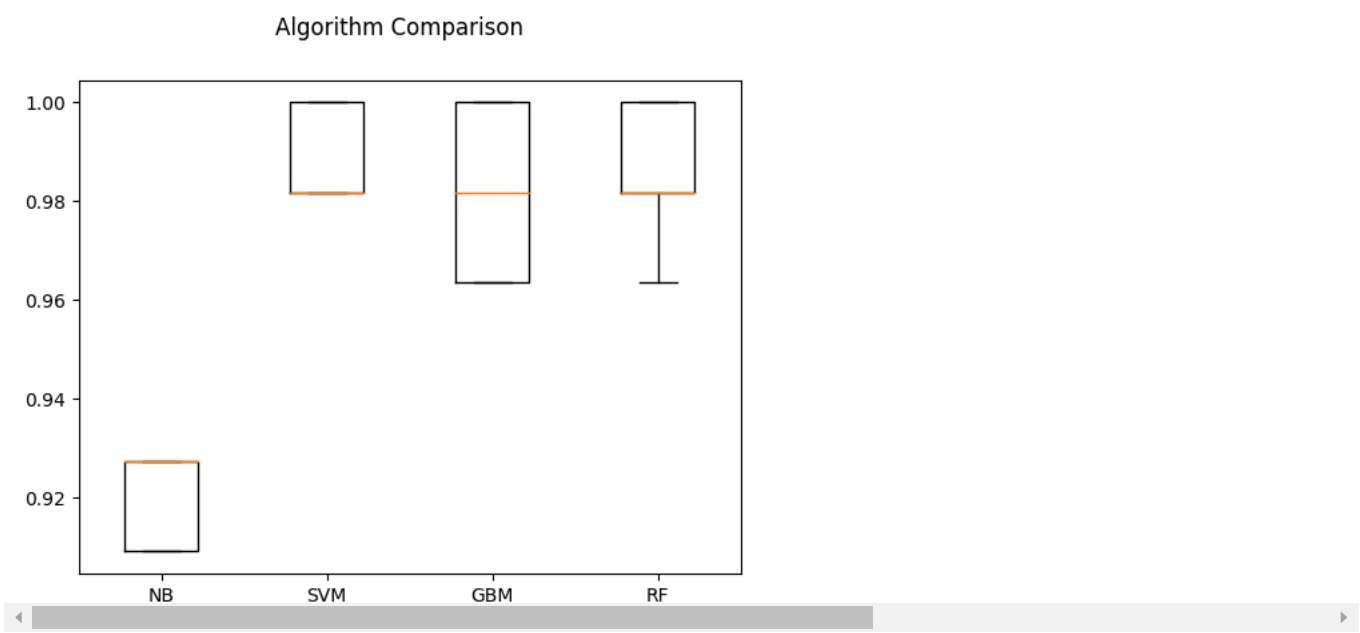
```
((275, 6), (69, 6))
```

```
models = []
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))
models.append(('GBM', GradientBoostingClassifier()))
models.append(('RF', RandomForestClassifier()))
# evaluate each model in turn
results = []
names = []
print("Performance on Training set")
for name, model in models:
  kfold = KFold(n_splits=num_folds,shuffle=True,random_state=seed)
  cv_results = cross_val_score(model, x_train, y_train, cv=kfold, scoring='accuracy')
  results.append(cv_results)
  names.append(name)
  msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
  msg += '\n'
  print(msg)
```

```
Performance on Training set
NB: 0.920000 (0.008907)

SVM: 0.989091 (0.008907)

GBM: 0.981818 (0.016262)

RF: 0.985455 (0.013606)
```

Compare Algorithms' Performance

```
fig = plt.figure()
fig.suptitle('Algorithm Comparison')
ax = fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names)
plt.show()
```



Model Evaluation by testing with independent/external test data set.

```
 # Make predictions on validation/test dataset

svm = SVC()
gb = GradientBoostingClassifier()
rf = RandomForestClassifier()
nb = GaussianNB()

best_model = svm

best_model.fit(x_train, y_train)
y_pred = best_model.predict(x_test)
print("Best Model Accuracy Score on Test Set:", accuracy_score(y_test, y_pred))
```

```
Best Model Accuracy Score on Test Set: 0.9565217391304348
```

Model Performance Evaluation Metric 1 - Classification Report

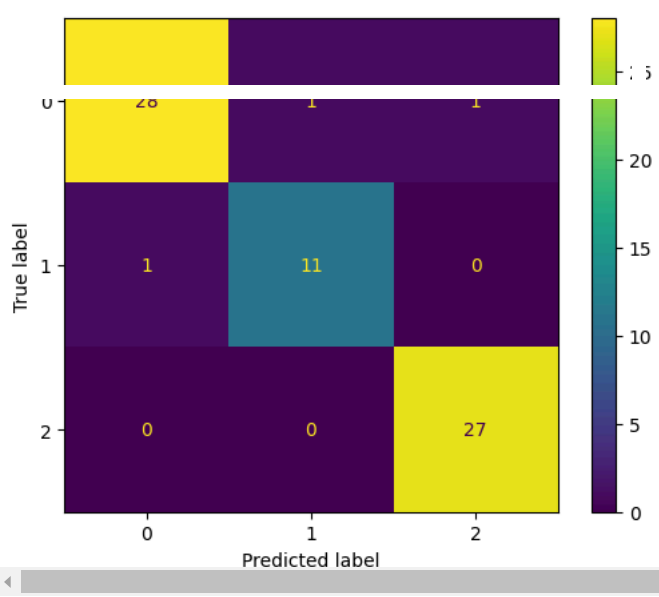
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.93      0.95        30
           1       0.92      0.92      0.92        12
           2       0.96      1.00      0.98        27

    accuracy                           0.96        69
   macro avg       0.95      0.95      0.95        69
weighted avg       0.96      0.96      0.96        69
```

Model Performance Evaluation Metric 2

```
#Confusion matrix
from sklearn.metrics import confusion_matrix, ConfusionMatrixDispla
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
```

```
disp.plot()
plt.show()
```



Model Evaluation Metric 4-prediction report

```
for x in range(len(y_pred)):
  print("Predicted: ", y_pred[x], "Actual: ", y_test[x], "Data: ", x_test[x],)
```

```
Predicted:  1 Actual:  1 Data:  (1, 2, 118, 44, 16, 17)
Predicted:  0 Actual:  0 Data:  (1, 1, 45, 36, 3, 12)
Predicted:  0 Actual:  0 Data:  (0, 2, 72, 52, 21, 42)
Predicted:  2 Actual:  2 Data:  (0, 2, 136, 26, 46, 87)
Predicted:  0 Actual:  0 Data:  (1, 2, 53, 41, 12, 20)
Predicted:  1 Actual:  0 Data:  (2, 2, 95, 58, 22, 44)
Predicted:  2 Actual:  2 Data:  (0, 0, 85, 26, 41, 67)
Predicted:  2 Actual:  2 Data:  (0, 1, 77, 9, 32, 56)
Predicted:  1 Actual:  1 Data:  (1, 1, 89, 35, 16, 12)
Predicted:  0 Actual:  0 Data:  (0, 1, 32, 55, 0, 9)
Predicted:  0 Actual:  0 Data:  (1, 2, 18, 64, 15, 33)
Predicted:  1 Actual:  1 Data:  (1, 2, 151, 57, 22, 19)
Predicted:  0 Actual:  0 Data:  (1, 1, 23, 54, 18, 21)
Predicted:  1 Actual:  1 Data:  (1, 1, 99, 44, 12, 27)
Predicted:  0 Actual:  0 Data:  (2, 2, 53, 59, 24, 40)
Predicted:  2 Actual:  2 Data:  (0, 1, 116, 12, 34, 57)
Predicted:  2 Actual:  2 Data:  (0, 2, 120, 26, 32, 79)
Predicted:  0 Actual:  0 Data:  (2, 2, 83, 49, 34, 40)
Predicted:  2 Actual:  0 Data:  (2, 3, 164, 80, 55, 94)
Predicted:  0 Actual:  0 Data:  (1, 1, 40, 57, 15, 25)
Predicted:  0 Actual:  0 Data:  (2, 1, 50, 39, 2, 19)
Predicted:  2 Actual:  2 Data:  (0, 1, 86, 7, 36, 63)
Predicted:  1 Actual:  1 Data:  (1, 2, 124, 51, 20, 51)
Predicted:  2 Actual:  2 Data:  (0, 2, 132, 21, 42, 86)
Predicted:  2 Actual:  2 Data:  (0, 1, 95, 11, 43, 61)
Predicted:  0 Actual:  0 Data:  (0, 1, 14, 38, 10, 9)
Predicted:  2 Actual:  2 Data:  (0, 1, 86, 2, 37, 74)
Predicted:  1 Actual:  1 Data:  (1, 2, 144, 51, 22, 31)
```