DEPARTMENT OF COMPUTER SCIENCE
SHAHEED SUKHDEV COLLEGE OF BUSINESS STUDIES
(UNIVERSITY OF DELHI)

CLASSIFICATION OF PALMER PENGUINS DATASET
(DATA MINING PROJECT REPORT)

SUBMITTED BY:
Shefalika Ghosh (19544)
Niti Tyagi (19522)

PROJECT SUPERVISOR:
Dr. Anamika Gupta, PhD
(Assistant Professor)

DECLARATION

It is hereby certified that the work being presented in the Data Mining Project Report entitled **“Palmer Archipelago (Antarctica) Penguins"** has been successfully completed under the supervision of Dr. Anamika Gupta, Ph.D. (Assistant Professor, Shaheed Sukhdev College of Business Studies, affiliated to University of Delhi) and is an authentic record of my own work carried out during the academic year 2021-2022.

Shefalika Ghosh

(Roll No: 19544)

This is to certify that the above statement made by the student is correct to the best of my knowledge.

Dr. Anamika Gupta, Ph.D.

(Assistant Professor)

(Project Supervisor)

ACKNOWLEDGEMENT

A perfect finish to any project requires guidance and I was lucky to have that support, bearing, and supervision in every perspective from my instructor. I am using this opportunity to express my gratitude to my professor **Dr. Anamika Gupta** who supported me throughout the course of this Data Mining project. Her aspiring guidance, support, encouragement and enthusiasm during the project work helped me in widening my horizons of knowledge. I am sincerely grateful to her for sharing her honest and illuminating views and experience on a number of issues related to the project.

I would also like to extend my sincere thanks to my project partner **Ms. Niti Tyagi**. This project would not have been possible without her kind support, help and incredible contribution every step of the way.

This acknowledgement will remain incomplete if I fail to express my deep sense of obligation to my parents for their consistent support and encouragement.

ABSTRACT

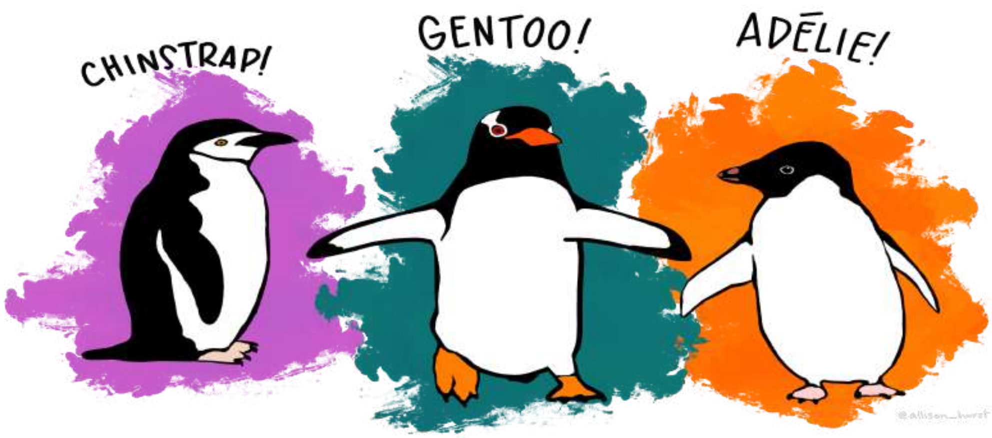
Data mining is the science of discovering correlations, hidden patterns, trends or relationships by searching through a large amount of data. It is also called knowledge discovery. Using a combination of machine learning, statistical analysis, modeling techniques and database technology, data mining finds patterns and subtle relationships in data and infers rules that allow the prediction of future.

This project acts as a platform to give us an introductory understanding to the vast field of data mining. For our project we have chosen the **Palmer Archipelago (Antarctica) penguins dataset** which describes size measurements collected from 2007 - 2009 for 3 penguin species in the islands of Palmer Archipelago, Antarctica. This project aims to demonstrate and teach classical data exploration, visualization, and classification techniques.

The objective of this project is to build a classification model to predict the species of the penguin from its size measurements and evaluating the performance of that model. This document contains the full Python code used to perform classification on the dataset.

Table of contents

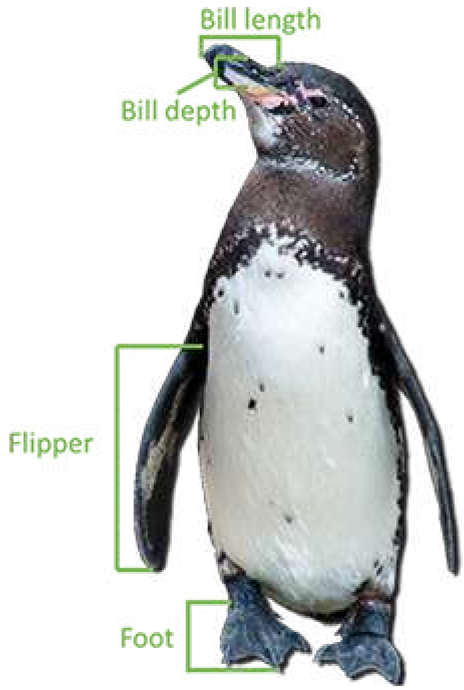
Dataset Description



Palmer Archipelago (Anatarctica) Penguins dataset contains size measurements collected from 2007-2009 for 3 penguin species namely - **'Adélie', 'Chinstrap', 'Gentoo'** in the islands of Palmer Archipelago, Antarctica. The dataset is composed of 344 observations with 8 variables, 5 of which are numeric and 3 are qualitative. The dataset is mostly complete with just a few observations with missing values that will need to be handled.

Penguins Data Column Definition

- **Species:** penguin species (Chinstrap, Adélie, or Gentoo)
- **Island:** island name (Dream, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica)
- **bill_length_mm:** culmen length (mm)
- **bill_depth_mm:** culmen depth (mm)
- **flipper_length_mm:** flipper length (mm)
- **body_mass_g:** body mass (g)
- **Sex:** penguin sex (male or female)
- **Year:** year in which data is recorded



What is bill?

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.

What are flippers?

Penguins' wings are called flippers. They are flat, thin, and broad with a long, tapered shape and a blunt, rounded tip.

Penguin species

1. Adélie

The Adélie penguin has a black head and distinctive white eye rings

2. Gentoo

The gentoo has a black head with white eyelids, and a distinct triangular white patch above each eye, usually extending over the head

3. Chinstrap

The top of a chinstrap's head is black and the face is white, with a stripe of black extending under the chin.

Python Libraries

1. Pandas
2. Numpy
3. Matplotlib
4. Seaborn
5. scikit-learn

Palmer Penguins - Data Exploration and Visualization

Niti Tyagi(19522) | Shefalika Ghosh(19544)

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:

```
data = pd.read_csv("palmerpenguins.csv")
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   species            344 non-null    object 
 1   island             344 non-null    object 
 2   bill_length_mm     342 non-null    float64
 3   bill_depth_mm      342 non-null    float64
 4   flipper_length_mm  342 non-null    float64
 5   body_mass_g        342 non-null    float64
 6   sex                333 non-null    object 
 7   year               344 non-null    int64  
dtypes: float64(4), int64(1), object(3)
memory usage: 21.6+ KB
```

In [3]:

```
data.head()
```

Out[3]:

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	year
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male	2007
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female	2007
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female	2007
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN	2007
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female	2007

In [4]:

```
data.describe()
```

Out[4]:

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	year
count	342.000000	342.000000	342.000000	342.000000	344.000000
mean	43.921930	17.151170	200.915205	4201.754386	2008.029070
std	5.459584	1.974793	14.061714	801.954536	0.818356
min	32.100000	13.100000	172.000000	2700.000000	2007.000000
25%	39.225000	15.600000	190.000000	3550.000000	2007.000000
50%	44.450000	17.300000	197.000000	4050.000000	2008.000000
75%	48.500000	18.700000	213.000000	4750.000000	2009.000000

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	year
max	59.600000	21.500000	231.000000	6300.000000	2009.000000

In [8]:

```
print('Correlation:')
data.corr()
```

```
Correlation:
```

Out[8]:

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	year
bill_length_mm	1.000000	-0.235053	0.656181	0.595110	0.054545
bill_depth_mm	-0.235053	1.000000	-0.583851	-0.471916	-0.060354
flipper_length_mm	0.656181	-0.583851	1.000000	0.871202	0.169675
body_mass_g	0.595110	-0.471916	0.871202	1.000000	0.042209
year	0.054545	-0.060354	0.169675	0.042209	1.000000

We can see that flipper length is highly positively correlated with body mass which makes sense given that larger penguins should have larger flippers.

Let's now take a look at the different penguin species in our dataset

In [5]:

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

Out[5]:

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

In [7]:

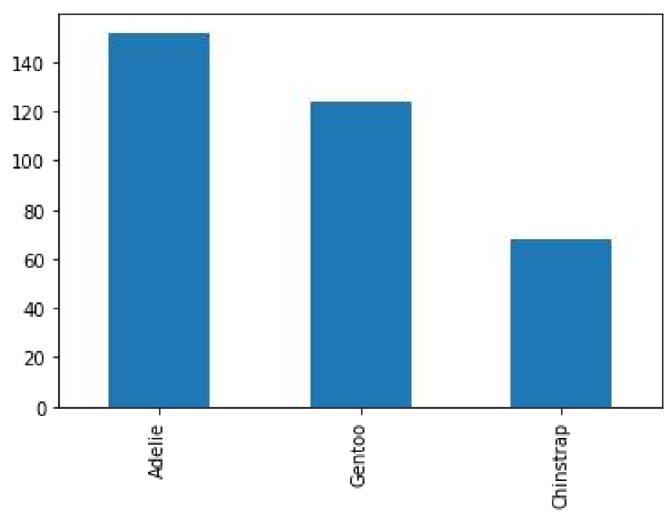
```
data['species'].value_counts()
```

Out[7]:

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

In [6]:

```
data['species'].value_counts().plot(kind='bar')
plt.show()
```



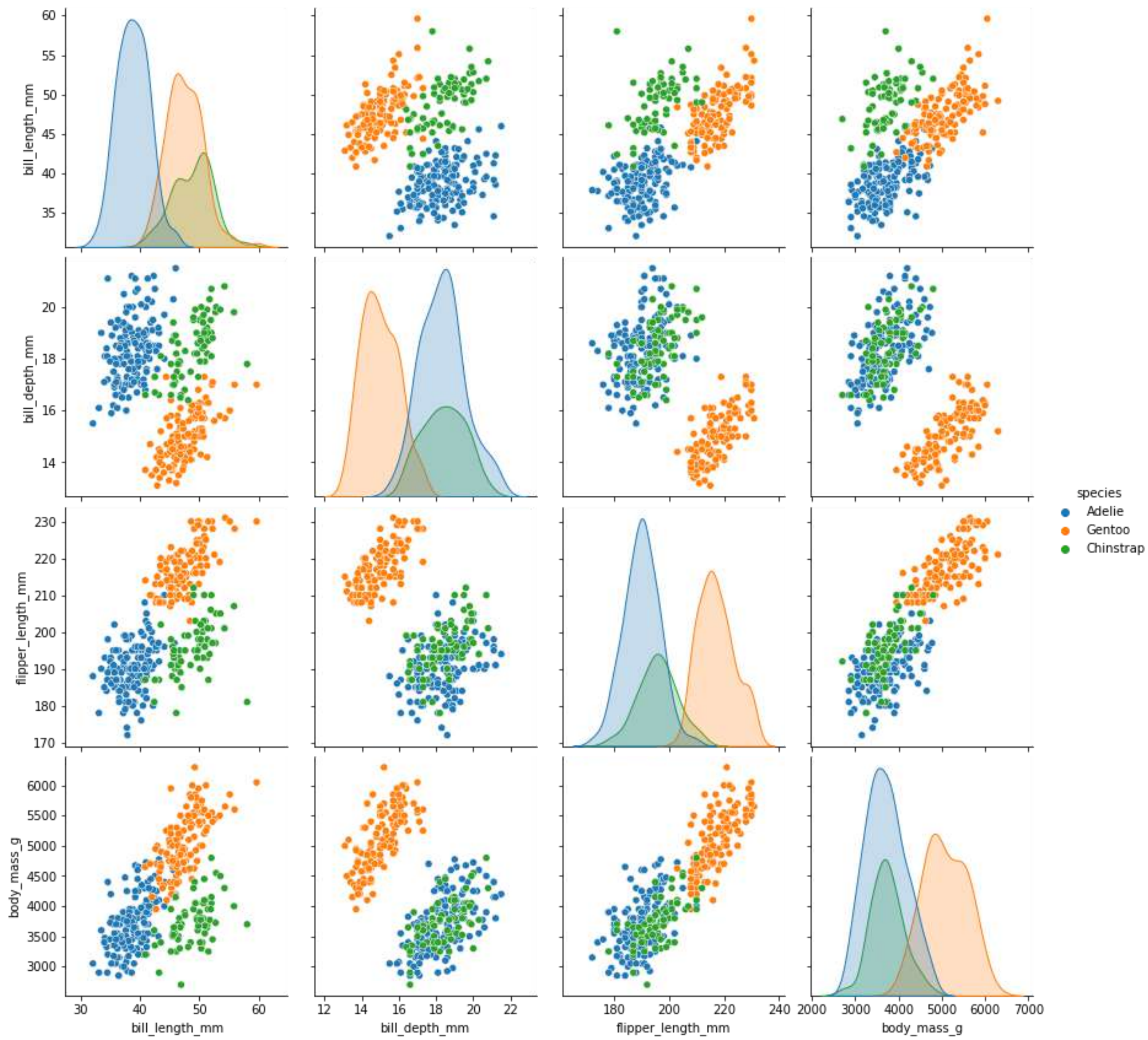
There are 3 penguin species namely- Adelie, Gentoo and Chinstrap.

It can clearly be seen that while the 'Adelie' species dominates the dataset, the 'Chinstrap' species has the lowest no. of instances

Let's examine the relationsip among the features of the different penguin species

In [13]:

```
sns.pairplot(data=data[['species','bill_length_mm','bill_depth_mm','flipper_length_mm', 'body_mass_g
             hue="species", height=3, diag_kind="kde")
plt.show()
```



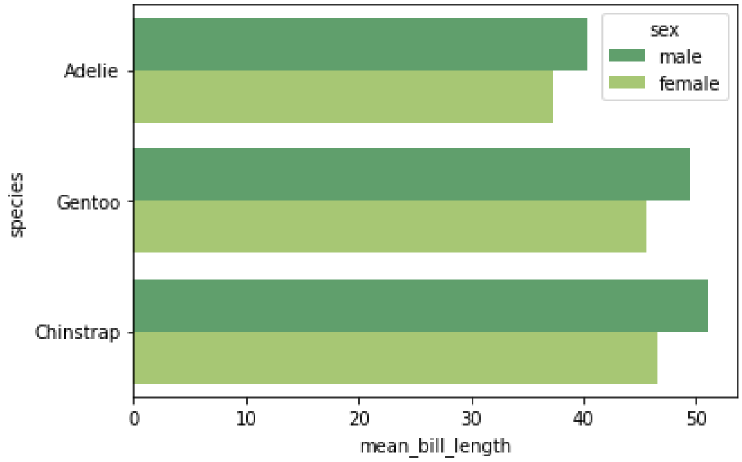
- When looking at body measurements we see that the features of Adelie and Chinstrap penguins largely overlap except for bill_length. This suggests that we might be able to use bill_depth, body_mass and flipper_length to differentiate the Gentoo penguins from the other species.
- Another thing to be noted is that the Adelie penguin stands out from the others in bill_length.

-> Examining bill length

In [14]:

```
df = data.loc[:,['species','bill_length_mm','sex']]
df['mean_bill_length'] = df.groupby(['species','sex'])['bill_length_mm'].transform('mean')
df = df.drop('bill_length_mm', axis = 1).drop_duplicates()
```

In [15]:
```
sns.barplot(data=df, x='mean_bill_length', y='species', hue='sex',palette='summer')
plt.show()
```



The Chinstrap species of penguins (both male and female) have the longest bills

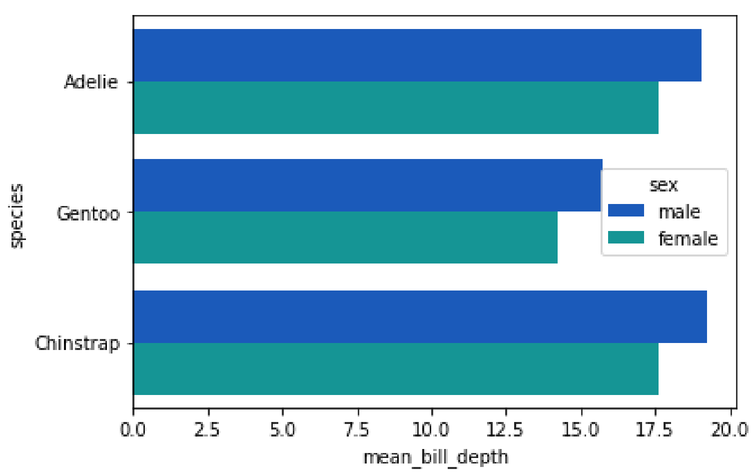
-> Examining bill_depth

In [22]:
```
dfd = data.loc[:,['species','bill_depth_mm','sex']]
dfd['mean_bill_depth'] = dfd.groupby(['species','sex'])['bill_depth_mm'].transform('mean')
dfd = dfd.drop('bill_depth_mm', axis=1).drop_duplicates()
dfd.dropna()
```

Out[22]:

	species	sex	mean_bill_depth
0	Adelie	male	19.072603
1	Adelie	female	17.621918
152	Gentoo	female	14.237931
153	Gentoo	male	15.718033
276	Chinstrap	female	17.588235
277	Chinstrap	male	19.252941

In [23]:
```
sns.barplot(data=dfd, x='mean_bill_depth', y='species', hue='sex',palette="winter")
plt.show()
```



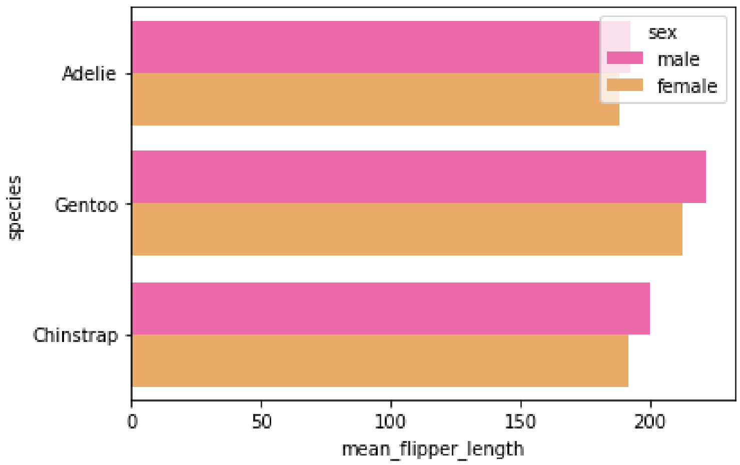
Among males, Chinstrap penguins have bills with the most depth.

Among females, Adelie penguins have bills with the most depth

-> Examining flipper_length

In [19]:

```
df2 = data.loc[:,['species','flipper_length_mm','sex']]
df2['mean_flipper_length'] = df2.groupby(['species','sex'])['flipper_length_mm'].transform('mean')
df2 = df2.drop('flipper_length_mm', axis=1).drop_duplicates()

sns.barplot(data=df2, x='mean_flipper_length', y='species', hue='sex', palette="spring")
plt.show()
```

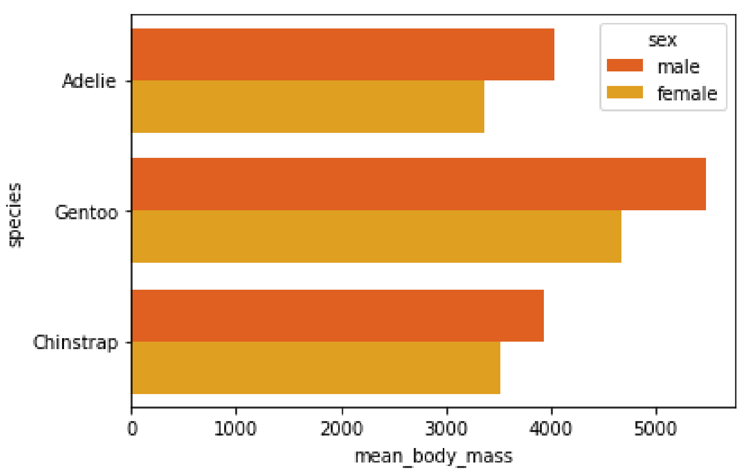


Gentoo species of penguins (both male and female) have the longest flipper lengths

-> Examining body_mass

In [27]:

```
df3 = data.loc[:,['species','body_mass_g','sex']]
df3['mean_body_mass'] = df3.groupby(['species','sex'])['body_mass_g'].transform('mean')
df3 = df3.drop('body_mass_g', axis=1).drop_duplicates()

sns.barplot(data=df3, x='mean_body_mass', y='species', hue='sex', palette = 'autumn')
plt.show()
```



Gentoo species of penguins (both male and female) have the highest body mass

We now look into the island attribute of our data

In [28]:
```
frame=data['island'].value_counts()
frame
```

Out[28]:
```
Biscoe       168
Dream        124
Torgersen     52
Name: island, dtype: int64
```

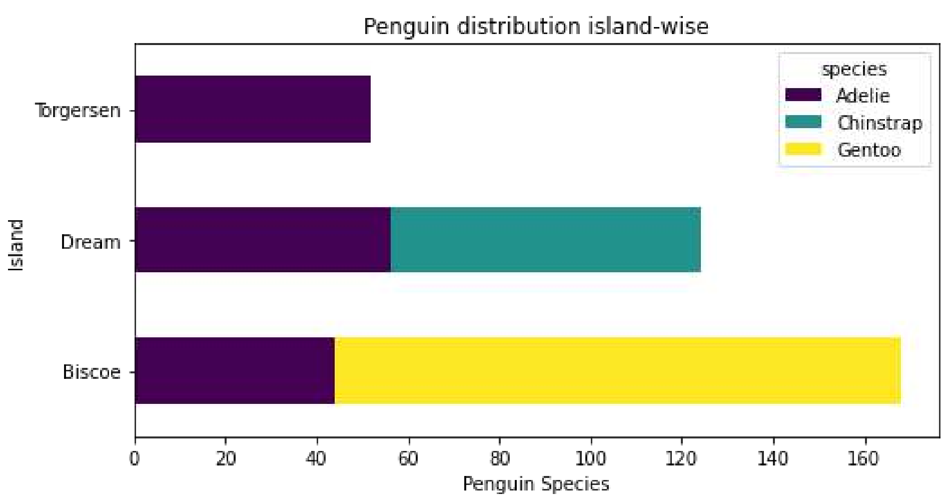
In [48]:
```
cross_tab = pd.crosstab(index=data['island'],
                        columns=data['species'])
cross_tab
```

Out[48]:

species island	Adelie	Chinstrap	Gentoo
Biscoe	44	0	124
Dream	56	68	0
Torgersen	52	0	0

In [70]:
```
cross_tab.plot(kind='barh',
            stacked=True,
            colormap='viridis',
            figsize=(8, 4))

plt.ylabel("Island")
plt.xlabel("Penguin Species")
plt.title("Penguin distribution island-wise")
```

Out[70]:
```
Text(0.5, 1.0, 'Penguin distribution island-wise')
```



1. There are 3 islands in our dataset-

- Biscoe
- Dream
- Torgersen

2. Biscoe island has the largest population of penguins.

3. Adelie penguins inhabit all 3 islands while Gentoo penguin species are found only on Biscoe and Chinstrap are found only on Dream island.

Palmer Penguins - Classification

Shefalika Ghosh(19544) | Niti Tyagi(19522)

In [1]:

```
import numpy as np
import pandas as pd
```

In [2]:

```
data = pd.read_csv("palmerpenguins.csv")
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   species            344 non-null    object
 1   island             344 non-null    object
 2   bill_length_mm     342 non-null    float64
 3   bill_depth_mm      342 non-null    float64
 4   flipper_length_mm  342 non-null    float64
 5   body_mass_g        342 non-null    float64
 6   sex                333 non-null    object
 7   year               344 non-null    int64
dtypes: float64(4), int64(1), object(3)
memory usage: 21.6+ KB
```

In [53]:

```
data.head()
```

Out[53]:

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	year
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male	2007
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female	2007
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female	2007
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN	2007
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female	2007

1. Feature Selection

In [3]:

```
df = data.drop(["island","sex"], axis=1)
df = df.dropna(subset=['bill_length_mm', 'bill_depth_mm','flipper_length_mm','body_mass_g'],how="all
df.head()
```

Out[3]:

	species	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	year
0	Adelie	39.1	18.7	181.0	3750.0	2007
1	Adelie	39.5	17.4	186.0	3800.0	2007
2	Adelie	40.3	18.0	195.0	3250.0	2007
4	Adelie	36.7	19.3	193.0	3450.0	2007
5	Adelie	39.3	20.6	190.0	3650.0	2007

In [4]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Classification using : Decision Tree, Naive Bayes and KNN classifiers

Train set = 75%

Test set = 25%

In [6]:
```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix
```

In [49]:
```
from sklearn.metrics import classification_report
```

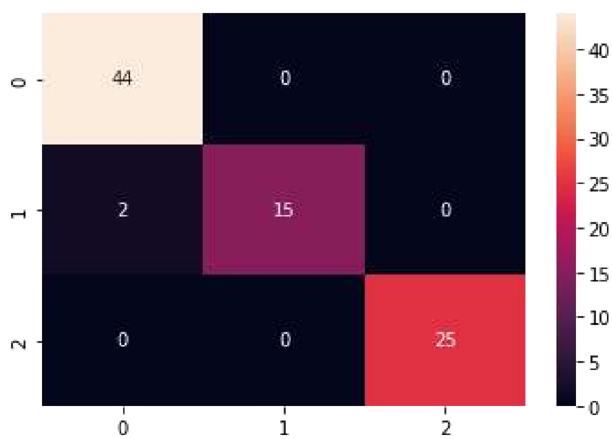
In [8]:
```
X = df.drop(["species"], axis=1)
Y = df.species
```

We will build models to correctly classify penguin species based on their bill length, bill depth, flipper length, body mass

In [9]:
```
X_Train, X_Test, y_train, y_test = train_test_split(X, Y, test_size = 0.25,random_state=42)
```

Decision Tree

In [30]:
```
decision_tree = DecisionTreeClassifier()
decision_tree.fit(X_Train,y_train)

y_pred = decision_tree.predict(X_Test)
sns.heatmap(confusion_matrix(y_test,y_pred), annot = True, fmt = 'd')
plt.show()
```



In [31]:
```
dtree_score = decision_tree.score(X_Test,y_test)
dtree_acc = float("%.4f" % round(dtree_score,4))*100
print("Accuracy of the Decision Tree classifier is: ",dtree_acc,"%")
```

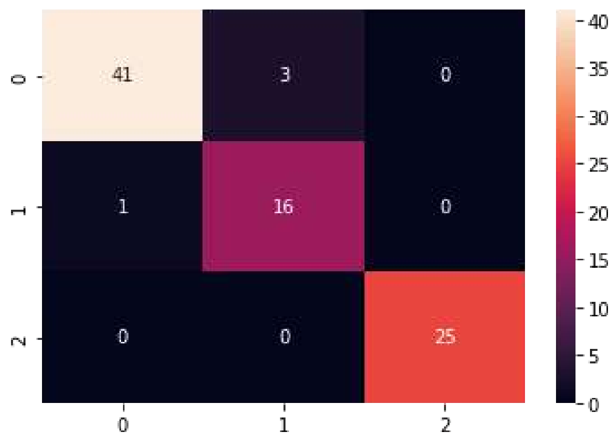
```
Accuracy of the Decision Tree classifier is:  97.67 %
```

In [53]:
```
print("Decision tree")
print(classification_report(y_test, y_pred))
```

```
Decision tree
              precision    recall  f1-score   support

      Adelie       0.96      1.00      0.98        44
   Chinstrap       1.00      0.88      0.94        17
      Gentoo       1.00      1.00      1.00        25

    accuracy                           0.98        86
   macro avg       0.99      0.96      0.97        86
weighted avg       0.98      0.98      0.98        86
```

Naive Bayes

In [14]:
```
gnb = GaussianNB()
gnb.fit(X_Train,y_train)

y_predg = gnb.predict(X_Test)
sns.heatmap(confusion_matrix(y_test,y_predg), annot = True, fmt = 'd')
plt.show()
```



In [15]:
```
gnb_score = gnb.score(X_Test,y_test)
gnb_acc = float("%.4f" % round(gnb_score,4))*100
print("Accuracy of Gaussian Naive Bayes classifier is: ",gnb_acc,"%")
```

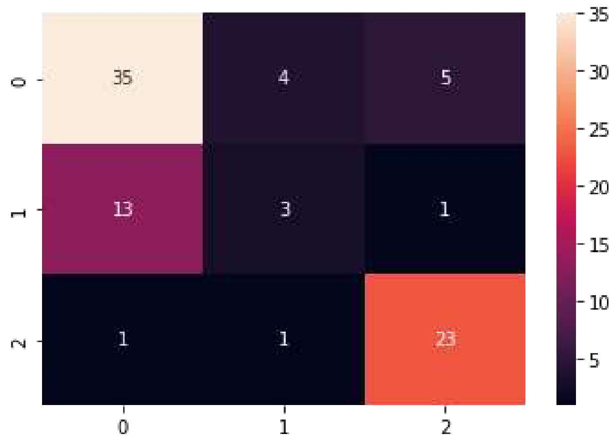
```
Accuracy of Gaussian Naive Bayes classifier is:  95.35 %
```

In [52]:
```
print("Gaussian Naive Bayes")
print(classification_report(y_test, y_predg))
```

```
Gaussian Naive Bayes
              precision    recall  f1-score   support

      Adelie       0.98      0.93      0.95        44
   Chinstrap       0.84      0.94      0.89        17
      Gentoo       1.00      1.00      1.00        25

    accuracy                           0.95        86
   macro avg       0.94      0.96      0.95        86
weighted avg       0.96      0.95      0.95        86
```

K-nearest neighbours

In [16]:
```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_Train,y_train)

y_predk = knn.predict(X_Test)
sns.heatmap(confusion_matrix(y_test,y_predk), annot = True, fmt = 'd')
plt.show()
```



In [17]:
```
knn_score = knn.score(X_Test,y_test)
knn_acc = float("%.4f" % round(knn_score,4))*100
print("Accuracy of the K-nearest neighbours classifier is: ",knn_acc,"%")
```

```
Accuracy of the K-nearest neighbours classifier is:  70.93 %
```

In [51]:
```
print("KNN")
print(classification_report(y_test, y_predk))
```

```
KNN
              precision    recall  f1-score   support

      Adelie       0.71      0.80      0.75        44
   Chinstrap       0.38      0.18      0.24        17
      Gentoo       0.79      0.92      0.85        25

    accuracy                           0.71        86
   macro avg       0.63      0.63      0.61        86
weighted avg       0.67      0.71      0.68        86
```

Decision Tree performs best with an accuracy of 97.67% while KNN performs the least well with an accuracy of 70.93%

2. Feature Scaling

In [77]:
```
df.describe()
```

Out[77]:

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	year
count	342.000000	342.000000	342.000000	342.000000	342.000000
mean	43.921930	17.151170	200.915205	4201.754386	2008.029240
std	5.459584	1.974793	14.061714	801.954536	0.817168
min	32.100000	13.100000	172.000000	2700.000000	2007.000000
25%	39.225000	15.600000	190.000000	3550.000000	2007.000000

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	year
50%	44.450000	17.300000	197.000000	4050.000000	2008.000000
75%	48.500000	18.700000	213.000000	4750.000000	2009.000000
max	59.600000	21.500000	231.000000	6300.000000	2009.000000

We can clearly see that our data needs to be scaled else 'flipper_length' and 'body_mass' will play a more decisive role than the other attributes in algorithms that calculate distances between data (such as in KNN)

Applying classifiers on data scaled to standard format

Since information based algorithms (Decision Trees, Random Forests) and probability based algorithms (Naive Bayes, Bayesian Networks) don't require normalization because they are immune to the feature magnitude, we'll normalize the data and apply KNN classifier to see if it improves the performance of the K-nearest neighbours classifier

In [5]:

```
from sklearn import preprocessing
```

In [10]:

```
min_max_scaler = preprocessing.MinMaxScaler()
X_train_minmax = min_max_scaler.fit_transform(X_Train)
X_test_minmax = min_max_scaler.transform(X_Test)
```

KNN CLASSIFIER

In [18]:

```
knn.fit(X_train_minmax,y_train)
kscore = knn.score(X_test_minmax,y_test)

knn_acc2 = float("%.4f" % round(kscore,4))*100
print("Accuracy of the K-nearest neighbours classifier when applied on normalized data: ",knn_acc2,'
```

```
Accuracy of the K-nearest neighbours classifier when applied on normalized data:  98.83999999999999
%
```

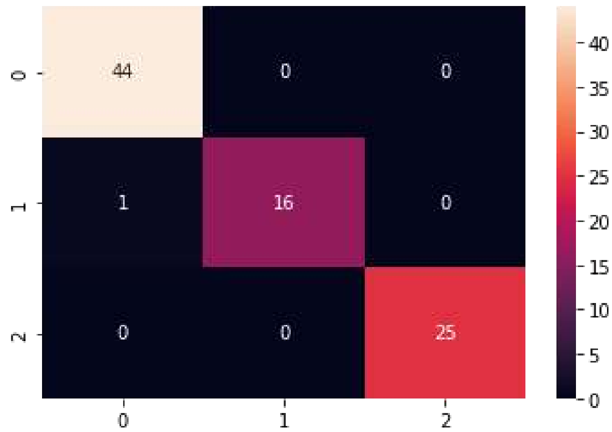
In [55]:

```
y_predk2 = knn.predict(X_test_minmax)

print("KNN performance on normalized data")
print(classification_report(y_test, y_predk2))
```

```
KNN performance on normalized data
              precision    recall  f1-score   support

      Adelie       0.98      1.00      0.99        44
   Chinstrap       1.00      0.94      0.97        17
      Gentoo       1.00      1.00      1.00        25

    accuracy                           0.99        86
   macro avg       0.99      0.98      0.99        86
weighted avg       0.99      0.99      0.99        86
```

In [78]:

```
sns.heatmap(confusion_matrix(y_test,y_predk2), annot = True, fmt = 'd')
plt.show()
```



GAUSSIAN NAIVE BAYES CLASSIFIER

In [19]:

```
gnb.fit(X_train_minmax,y_train)
nbscore = gnb.score(X_test_minmax,y_test)

gnb_acc2 = float("%.4f" % round(nbscore,4))*100
print("Accuracy of the Naive Bayes classifier when applied on normalized data: ",gnb_acc2,"%")
```

```
Accuracy of the Naive Bayes classifier when applied on normalized data:  95.35 %
```

DECISION TREE CLASSIFIER

In [22]:

```
decision_tree.fit(X_train_minmax,y_train)
dtscore = decision_tree.score(X_test_minmax,y_test)

dt_acc2 = float("%.4f" % round(dtscore,4))*100
print("Accuracy of the Decision Tree classifier when applied on normalized data: ",dt_acc2,"%")
```

```
Accuracy of the Decision Tree classifier when applied on normalized data:  97.67 %
```

As we can see, on normalized data, KNN outperforms Decision tree and Naive Bayes classifiers

And as mentioned before, normalization doesn't affect the accuracy of Decision Tree and Naive Bayes classifiers

In [73]:

```
data2 = {'Not normalized (Accuracy in %)': [dtree_acc, gnb_acc, knn_acc],
        'Normalized data(Accuracy in %)': [dt_acc2, gnb_acc2, knn_acc2]}

norm_results = pd.DataFrame(data2, index =['Decision Tree', 'Gaussian Naive Bayes', 'KNN'])

norm_results
```

Out[73]:

	Not normalized (Accuracy in %)	Normalized data(Accuracy in %)
Decision Tree	97.67	97.67
Gaussian Naive Bayes	95.35	95.35
KNN	70.93	98.84

We can see that KNN classifier's performance improves significantly after normalizing the data

3. Cross-Validation

We'll implement 3 techniques:-

1. Random Subsampling
2. K-fold cross validation
3. Stratified K-fold cross validation

In [32]:

```
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
```

3.1) Random subsampling

KNN CLASSIFIER

In [34]:

```
Norm_X = min_max_scaler.fit_transform(X)
```

In [35]:

```
model_knn = KNeighborsClassifier(n_neighbors=5)
shuffle_split=ShuffleSplit(test_size=0.3,train_size=0.5,n_splits=10)

scores_knn=cross_val_score(model_knn,Norm_X,Y,cv=shuffle_split)
print("The accuracy obtained in each iteration: \n",scores_knn)
print("\nKNN classifier Accuracy:",(scores_knn.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [0.94174757 0.97087379 0.98058252 0.97087379 0.96116505 0.95145631
 0.98058252 0.99029126 0.98058252 0.98058252]

KNN classifier Accuracy: 97.08737864077669 %
```

DECISION TREE CLASSIFIER

In [42]:

```
model_dtree = DecisionTreeClassifier()
scores_dtree=cross_val_score(model_dtree,X,Y,cv=shuffle_split)
print("The accuracy obtained in each iteration: \n",scores_dtree)
print("\nDecision Tree classifier Accuracy:",(scores_dtree.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [0.96116505 0.97087379 0.95145631 0.94174757 0.94174757 0.95145631
 0.95145631 1.         0.97087379 0.9223301 ]

Decision Tree classifier Accuracy: 95.63106796116504 %
```

GAUSSIAN NAIVE BAYES CLASSIFIER

In [43]:

```
model_gnb = GaussianNB()
scores_gnb=cross_val_score(model_gnb,X,Y,cv=shuffle_split)
print("The accuracy obtained in each iteration: \n",scores_gnb)
print("\nGaussian Naive Bayes classifier Accuracy:",(scores_gnb.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [0.98058252 0.95145631 0.97087379 0.97087379 0.96116505 0.98058252
 0.98058252 1.         0.98058252 0.97087379]

Gaussian Naive Bayes classifier Accuracy: 97.47572815533981 %
```

With random subsampling, Gaussian Naive Bayes classifier gives the highest mean accuracy

3.2) K-fold cross validation

In [56]:

```
kfolds = KFold(n_splits=6)
```

KNN CLASSIFIER

In [60]:

```
model_1 = KNeighborsClassifier(n_neighbors=5)
score_knn = cross_val_score(model_1, Norm_X, Y, cv=kfolds)
print("The accuracy obtained in each iteration: \n",score_knn)
print("\nKNN classifier Accuracy:",(score_knn.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [0.14035088 0.96491228 0.94736842 0.89473684 0.80701754 0.22807018]

KNN classifier Accuracy: 66.37426900584794 %
```

DECISION TREE CLASSIFIER

In [61]:

```
model_2 = DecisionTreeClassifier()
score_dtree = cross_val_score(model_2, X, Y, cv=kfolds)
print("The accuracy obtained in each iteration: \n",score_dtree)
print("\nDecision Tree classifier Accuracy:",(score_dtree.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [0.15789474 0.89473684 0.98245614 0.98245614 0.94736842 0.71929825]

Decision Tree classifier Accuracy: 78.0701754385965 %
```

NAIVE BAYES CLASSIFIER

In [62]:

```
model_3 = GaussianNB()
score_gnb = cross_val_score(model_3, X, Y, cv=kfolds)
print("The accuracy obtained in each iteration: \n",score_gnb)
print("\nNaive Bayes classifier Accuracy:",(score_gnb.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [0.84210526 0.94736842 0.94736842 1.         1.         0.22807018]

Naive Bayes classifier Accuracy: 82.7485380116959 %
```

With K-fold cross validation (with K=6), Gaussian Naive Bayes classifier again gives the highest mean accuracy

3.3) Stratified K-fold cross validation

(The folds are made by preserving the percentage of samples for each class.)

In [64]:

```
from sklearn.model_selection import StratifiedKFold
skf = StratifiedKFold(n_splits=6, shuffle=True, random_state=1)
```

KNN CLASSIFIER

In [65]:

```
score_knn2 = cross_val_score(model_1, Norm_X, Y, cv=skf)
print("The accuracy obtained in each iteration: \n",score_knn2)
print("\nKNN classifier Accuracy:",(score_knn2.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [1.         0.98245614 1.         0.96491228 0.94736842 1.        ]
```

```
KNN classifier Accuracy: 98.24561403508771 %
```

DECISION TREE CLASSIFIER

In [66]:
```
score_dtree2 = cross_val_score(model_2, X, Y, cv=skf)
print("The accuracy obtained in each iteration: \n",score_dtree2)
print("\nDecision Tree classifier Accuracy:",(score_dtree2.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [0.94736842 0.96491228 1.         0.94736842 0.94736842 0.96491228]

Decision Tree classifier Accuracy: 96.19883040935672 %
```

NAIVE BAYES CLASSIFIER

In [67]:
```
score_gnb2 = cross_val_score(model_3, X, Y, cv=skf)
print("The accuracy obtained in each iteration: \n",score_gnb2)
print("\nNaive Bayes classifier Accuracy:",(score_gnb2.mean()*100),"%")
```

```
The accuracy obtained in each iteration:
 [0.98245614 0.96491228 1.         0.96491228 0.92982456 0.98245614]

Naive Bayes classifier Accuracy: 97.07602339181287 %
```

With Stratified K-fold cross validation (with K=6), KNN classifier gives the highest mean accuracy

Summarizing the results of cross-validation

In [68]:
```
rk = scores_knn.mean()*100
rd = scores_dtree.mean()*100
rb = scores_gnb.mean()*100

kk = score_knn.mean()*100
kd = score_dtree.mean()*100
kb = score_gnb.mean()*100

skk = score_knn2.mean()*100
skd = score_dtree2.mean()*100
skb = score_gnb2.mean()*100
```

In [71]:
```
data_acc = {'Random Subsampling (mean accuracy %)':[rk,rd,rb],
        'K-fold <k=6> (mean accuracy %)': [kk,kd,kb],
        'Stratified K-fold <k=6> (mean accuracy %)':[skk,skd,skb]}

crossval_results = pd.DataFrame(data_acc, index =['KNN', 'Decision Tree', 'Gaussian Naive Bayes'])

crossval_results
```

Out[71]:

	Random Subsampling (mean accuracy %)	K-fold <k=6> (mean accuracy %)	Stratified K-fold <k=6> (mean accuracy %)
KNN	97.087379	66.374269	98.245614
Decision Tree	95.631068	78.070175	96.198830
Gaussian Naive Bayes	97.475728	82.748538	97.076023

We can see that Stratified K-fold cross validation gives highest accuracy. It should also be noted that KNN (on normalized data) performs best overall

SUMMARY OF RESULTS

Data Analysis and Visualization

Flipper length is highly positively correlated with body mass (which makes sense given that larger penguins should have larger flippers)

There are 3 penguin species namely- Adelie, Gentoo and Chinstrap. It is observed that while the 'Adelie' species dominates the dataset, the 'Chinstrap' species has the lowest no. of instances.

Examining the relationship among the features of the different penguin species:-

- When looking at body measurements we see that the features of Adelie and Chinstrap penguins largely overlap except for bill_length. This suggests that we might be able to use bill_depth, body_mass and flipper_length to differentiate the Gentoo penguins from the other species.
- Another thing to be noted is that the Adelie penguin stands out from the others in bill_length.
- The Chinstrap species of penguins (both male and female) have the longest bills.
- Among males, Chinstrap penguins have bills with the most depth. Among females, Adelie penguins have bills with the most depth.
- Gentoo species of penguins (both male and female) have the longest flipper lengths.
- Gentoo species of penguins (both male and female) have the highest body mass.

Examining the 'island' attribute:-

- There are 3 islands in our dataset:-
  1. Biscoe
  2. Dream
  3. Torgersen

Biscoe island has the largest population of penguins.

Adelie penguins inhabit all 3 islands while Gentoo penguin species are found only on Biscoe and Chinstrap are found only on Dream island.

(P.T.O)

Classification

We experiment with 3 classification techniques (Decision Tree, Gaussian Naïve Bayes, KNN) to predict penguin species from their body part measurements.

After data cleaning we perform the following:-

1. **Feature selection** –

   We select only the relevant features (bill_length_mm bill_depth_mm, flipper_length_mm, body_mass_g) for building our classification models.

2. **Feature Scaling and Modeling**–

   We implemented 3 classifiers on our dataset and evaluated their performance measures and found that Decision Tree performs best with an accuracy of 97.67% while KNN performs the least well with an accuracy of 70.93%.

   We notice that the data needs to be scaled else attributes 'flipper_length' and 'body_mass' play a more decisive role than the other attributes in algorithms that calculate distances between data (KNN).

   After feature scaling, on normalized data, KNN outperforms Decision tree and Naive Bayes classifiers.

   The results for the same are summarized in the table below:-

	Not normalized (Accuracy in %)	Normalized data(Accuracy in %)
Decision Tree	97.67	97.67
Gaussian Naive Bayes	95.35	95.35
KNN	70.93	98.84

3. **Cross – Validation** –

   We cross-validate our dataset using 3 techniques- Random Subsampling, K-fold, Stratified K-fold on 3 classification models and evaluate their performance.

   We observe that Stratified K-fold cross validation gives highest accuracy. It was also noted that KNN (on normalized data) performs best overall.

   The results for the same are summarized in the table below:-

	Random Subsampling (mean accuracy %)	K-fold <k=6> (mean accuracy %)	Stratified K-fold <k=6> (mean accuracy %)
KNN	97.087379	66.374269	98.245614
Decision Tree	95.631068	78.070175	96.198830
Gaussian Naive Bayes	97.475728	82.748538	97.076023

Bibliography

- https://www.kaggle.com/code/parulpandey/penguin-dataset-the-new-iris/data
- https://www.geeksforgeeks.org/
- https://www.google.com/
- https://medium.com/
- https://towardsdatascience.com