```
# -*- coding: utf-8 -*-
"""Penguins Group Project.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/16Fj2ForcPSHX6mi6dTdF_4-ecl6YpC5w

Group Contributions Statement.

Our group members are Cody Lejang, Megan Tieu, and Arely Perez. All three of us wrote the data acquisition and preparation for our data. Arely created the first table with Culmen Lengt
"""

import pandas as pd
import numpy as np
import urllib
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
#importing and accessing all of our tools

url = "https://philchodrow.github.io/PIC16A/datasets/palmer_penguins.csv"
##we imported the data from the link

penguins = pd.read_csv(url)
#reading the data in our url link

train, test = train_test_split(penguins, test_size = 0.3, random_state = 0)
from sklearn import preprocessing
def prep_penguins_data(data):

    df = data.copy()
    le = preprocessing.LabelEncoder()
    df = df[df.Sex != '.']
    df = df.sort_values(by=['Sex'])
    X=penguins[["Culmen Length (mm)","Culmen Depth (mm)"]]
    y=penguins['Species']
    return(X, y)
X_train, y_train = prep_penguins_data(train)
X_test,  y_test  = prep_penguins_data(test)
## we're splitting the data into train and test groups where the test group is 30% of the data and the train group is 70% of the data.

penguins
#printing our table

"""Here we have our penguins data set. Now that we can view it we are able to use specific columns and rows in order to predict the species of a penguin. We also are able to split the

penguins.groupby(["Species","Island"])[["Culmen Length (mm)", "Body Mass (g)" ,"Culmen Depth (mm)"]].mean()
#based on our data table grouping by species island culmen length and body mass
#only looking at the mean of these columns

"""This table shows the penguins grouped by species, and then broken down by island. The columns display the penguins culmen length, body mass, and culmen depth. Itâ  s apparent that t

"""

fig, ax = plt.subplots(1)
#creating an empty figure
ax.set (xlabel= "Culmen Length (mm)", #length vs. depth graph
        ylabel= "Culmen Depth (mm)")
#creating a length vs. depth graph
#setting the x and y axis

species = set(penguins['Species'])
#only looking at our data for each species

for s in species:
  sub = penguins[penguins["Species"]==s]
  ax.scatter(sub['Culmen Length (mm)'], sub ['Culmen Depth (mm)'], label=s.split(' ')[0], alpha=0.5)
#plotting the points for each species
#generate scatterplot color coded by species
#making sure points are a bit lighter

ax.legend()
#creating our legend and labels

"""Explanation: The data shows a slight positive correlation between Culmen Length vs. Culmen Depth in a given species.
The data shows that the Gentoo has the lowest culmen depth and the Adelie and Chinstrap pebguins have equally high culmen depths.
The Adelie penguins have a shorter culmen length in comparison to Gentoo and Chinstrap penguins.

Next steps: Now that we have our first figure we can see the positive correlations between culmen length and depth for each species and can start to notice patterns between each specie
"""

fig,ax=plt.subplots(1)
#creating a figure


def plot_hist(df,colname,alpha):

    """
    function called plot_hist is created
    creating a histogram on the body mass
    for each species

    """


    ax.hist(df[colname],alpha=alpha)
    #creating a histogram

    ax.set(xlabel="Body mass (g)")
    #creating an x axis label

    ax.set(ylabel="species")
    #creating a y axis label

    ax.legend(('Adelie', 'Chinstrap','Gentoo'), loc='upper right');
    #creating a legend on the upper right of our figure


penguins.groupby("Species").apply(plot_hist,'Body Mass (g)',0.5)
#grouping by species and looking at the body mass for each species
#making sure bars are 0.5 lighter

"""Explanation: This figure shows the correlation between all three species and their body mass in grams. Our three species are Gentoo, Adelie, and Chinstrap. Our bar graph has Body ma

Next steps: Now that we can look at two more variables like body mass and species we can continue to notice patterns between our species and choose two more variables to graph so we ca
"""
import seaborn as sns
# box plot of penguins culmen depth grouped by species and seperated by sex
```

```
from matplotlib import pyplot as plt
fig, ax = plt.subplots(figsize = (13,10))
df = penguins.copy()
 #copy penguins dataframe

df = df[df.Sex != '.']
#drop rows where sex is listed as '.'
sns.boxplot(x = "Species", y = "Culmen Depth (mm)",data = df, hue = "Sex")
#setting the x and y axis labels and color coding based on sex

"""Explanation: This figure shows that across all three species, males have a longer Culmen Depth than females.
Gentoo penguins have shorter Culmen Depths than both Adelie and Chinstrap penguins, regardless of if the penguins are male or female.
From these observations, we can conclude that observing Culmen Depth can help determine which species the penguins belong to
because typically male penguins with a culmen depth of greater than 18 mm belong to the adelie or chinstrap, whereas male
penguins with a culmen depth less than 18 mm are more likely to be gentoo. For females, penguins with a culmen depth greater than 16 mm are
likely to be adelie or chinstrap, whereas females who are under 16 mm are likely to be gentoo.

Next Steps: Now we can begin to notice more patterns specifically in regards to the difference between males and females. Given that we have looked at three figures with different pair

**Feature Selection**

Based on the 3 figures above, we decided to choose Culmen Length (mm) and Culmen Depth (mm). We decided these were the best features used to predict the species, because we rated it th
"""

from sklearn import preprocessing

#recode the labels
le=preprocessing.LabelEncoder()
penguins["Species"]=le.fit_transform(penguins["Species"])

penguins=penguins.dropna(subset=["Culmen Length (mm)","Culmen Depth (mm)"])

X=penguins[["Culmen Length (mm)","Culmen Depth (mm)"]]
y=penguins['Species']

def plot_regions(c,X,y):
    """
    function called plot_regions is created
    so that we can

    """
    c.fit(X,y)

    x0=X["Culmen Length (mm)"]
    x1=X["Culmen Depth (mm)"]
    #setting X0 and X1 as new objects
    #each equal to the data corresponing in the culmen length and depth column

    grid_x=np.linspace(x0.min(),x0.max(),501)
    grid_y=np.linspace(x1.min(),x1.max(),501)
    #fixing up the row and column line space

    xx,yy=np.meshgrid(grid_x,grid_y)
    np.shape(xx),np.shape(yy)
    #getting the shape of our x and y axis

    XX=xx.ravel()
    YY=yy.ravel()
    #looking at an array


    p=c.predict(np.c_[XX,YY])
    #is is the new object equal to our predictions

    p=p.reshape(xx.shape)
    #reshaping our new object p

    fig,ax=plt.subplots(1)
    #creating an empty figure with no plots yet


    ax.contourf(xx,yy,p,cmap="jet",alpha=.2)
    #plot the decision regions

    ax.scatter(x0,x1,c=y,cmap="jet")
    #setting a scatter of our points

    ax.set(xlabel="Culmen Length (mm)",ylabel="Culmen Depth (mm)")
    #labeling our x and y axis

    custom = [Line2D([], [], marker='.', color='darkblue', linestyle='None'),
    Line2D([], [], marker='.', color='green', linestyle='None'),
    Line2D([], [], marker='.', color='red', linestyle='None')]
    ax.legend(custom, ['Adelie',
    'Chinstrap',
    'Gentoo'], loc='lower left')
    #legend created with color coded species

from sklearn.linear_model import LogisticRegression
from matplotlib.lines import Line2D
LR=LogisticRegression()
#creating our logistic regression
model=plot_regions(LR,X,y)
#plotting our regions

from sklearn.model_selection import cross_val_score


cv_scores=cross_val_score(LR,X,y,cv=5)
#cross validating our scores
cv_scores.mean()
#getting the mean of every score

"""Our first model is a logistic regression model. It has a cross value score of 0.961 and it compares Culmen Length and Depth for each species. """

from sklearn.ensemble import RandomForestClassifier
RF=RandomForestClassifier()
#creating random forest

plot_regions(RF,X,y)
#plotting our regions

from sklearn.model_selection import cross_val_score
from sklearn import tree

T=tree.DecisionTreeClassifier(max_depth=3)

cv_scores=cross_val_score(RF,X,y,cv=5)
#cross validation scores
cv_scores.mean()
```

```
#mean of all of the cross validation scores

"""Our second model is a random forest model. It has a cross value score of 0.97 and it compares Culmen Length and Depth for each species."""

from sklearn import svm

SVM=svm.SVC()
plot_regions(SVM,X,y)
from sklearn.model_selection import cross_val_score
cv_scores=cross_val_score(SVM,X,y,cv=5)
cv_scores.mean()

"""Our final model is a support vector machine. It has a cross value score of 0.953 and it compares Culmen Length and Depth for each species.

**Discussion. Describe the overall performance of your models, state which combination of model and features (measurements) you recommend. Discuss how the model could be improved if mo
**

#Model Performance
The cross value scores of the Logistic Regression model is roughly .961. The Random Forest classifier model's cross value score is approximately .97. The SVM model's score is lowest at

#Model/Feature Recommendations
We decided to use the Culmen Length vs. Culmen Depth relationship to generate a species predicting model. The features were selected through analysis of measures of center and overall

All machine learning models have a high cross value score and therefore would be accurate predictors for species based on the two quantitative variables/features we selected.
#Introduction of more data
The introduction of more data would bolster the accuracy of the model. With additional data, there is mroe training data, leading to a greater amount of accuracy. If different data was
"""
```