import pandas as pd

path="dataset.csv"

data = pd.read\_csv(path ,encoding='latin-1')

print(data.info())

#Data analysis

data= data.drop(['Secondary effects of the initial incident','General Human and organisational factors','Human and organisational factors based on incident type','Wind Direction','Wind Speed','Sea State','Air Temperature','Water Temperature','Raw','Economic impact damage on vessel','Economic impact damage on facilities'],'columns')

print(data.info())

print(data)

#Total no of deaths

Noofdeaths = data["Deaths"].sum()

print("Total no of deaths ",Noofdeaths)

#Total no of injuries

Noofinjuries = data["Injuries"].sum()

print("Total no of Injuries ",Noofinjuries)

#Injuries based on ship type

import plotly.express as px

position = data["Ship Type"]

runcounts = data["Injuries"]

fig = px.pie(data, values=runcounts,title='No of Injuries Based on Ship Type', names=position,hole = 0.2)

fig.show()

#deaths based on ship type

import plotly.express as px

position = data["Ship Type"]

runcounts = data["Deaths"]

fig = px.pie(data, values=runcounts,title='No of Deaths Based on Ship Type', names=position,hole = 0.2)

fig.show()

#Injuries based on Accident type

import plotly.express as px

position = data["Accident Type"]

runcounts = data["Injuries"]

fig = px.pie(data, values=runcounts,title='No of Injuries Based on Accident Type', names=position,hole = 0.2)

fig.show()

#Deaths based on Accident type

import plotly.express as px

position = data["Accident Type"]

runcounts = data["Deaths"]

fig = px.pie(data, values=runcounts,title='No of Deaths Based on Accident Type', names=position,hole = 0.2)

fig.show()

#No of passengers based on ship type

import plotly.express as px

position = data["Ship Type"]

runcounts = data["Passengers"]

fig = px.pie(data, values=runcounts, names=position,title='No of passengers Based on Ship Type',hole = 0.2)

fig.show()

#No of Crew Members based on ship type

import plotly.express as px

position = data["Ship Type"]

runcounts = data["Crew Members"]

fig = px.pie(data, values=runcounts,title='No of Crew members Based on Ship Type', names=position,hole = 0.2)

fig.show()

#Linear Regression to predict no of injuries

import sklearn

from sklearn.linear\_model import LinearRegression

from sklearn import preprocessing

# label\_encoder object knows how to understand word labels.

label\_encoder = preprocessing.LabelEncoder()

# Encode labels in column 'species'.

data['Accident Type\_old']= data['Accident Type']

data['Accident Type\_encodedvalues']= label\_encoder.fit\_transform(data['Accident Type'])

#print(data['Accident Type\_old'])

#print(data['Accident Type\_encodedvalues'])

print(data)

data['Accident Type']= label\_encoder.fit\_transform(data['Accident Type'])

data['Ship Type']= label\_encoder.fit\_transform(data['Ship Type']) #,'Visibility'

print(pd.isnull(data).sum())

inputs = data[['Accident Type','Ship Type','Deaths','Rain','Ship Length (m)','Persons on board','Crew Members','Passengers','Successful evacuation','Location Type','Environmental Pollution','lon','lat']]

output = data['Injuries']

model = LinearRegression()

model.fit(inputs,output)

acc = model.score(inputs,output)

print(acc)

#Logistic Regression to predict Accident Type

from sklearn.linear\_model import LogisticRegression

logisticRegr = LogisticRegression()

inputs = data[['Injuries','Ship Type','Deaths','Rain','Ship Length (m)','Persons on board','Crew Members','Passengers','Successful evacuation','Location Type','Environmental Pollution']]

output = data['Accident Type']

print(inputs)

print(output)

logisticRegr.fit(inputs,output)

score = logisticRegr.score(inputs, output)

print("Accuracy Using Logistic regression is : ",logisticRegr.score(inputs, output)\*100)

#Support Vector machine to predict Accident Type

# Fitting SVM to the Training set

from sklearn.svm import SVC

from sklearn import \*

#preprocessing for standar scaling

from sklearn.preprocessing import StandardScaler

model = SVC(random\_state = 0)

inputs = data[['Injuries','Ship Type','Deaths','Rain','Ship Length (m)','Persons on board','Crew Members','Passengers','Successful evacuation','Location Type','Environmental Pollution']]

output = data[['Accident Type']]

sc = StandardScaler()

inputs= sc.fit\_transform(inputs)

#output= sc1.fit\_transform(output)

print(output)

model.fit(inputs, output)

print("Accuracy Using SVM is : ",model.score(inputs, output)\*100)

#KNN - K nearest Neighbor to predict Accident Type

from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier(n\_neighbors = 9)

inputs = data[['Injuries','Ship Type','Deaths','Rain','Ship Length (m)','Persons on board','Crew Members','Passengers','Successful evacuation','Location Type','Environmental Pollution']]

output = data[['Accident Type']]

model.fit(inputs, output)

print("Accuracy Using KNN is : ",model.score(inputs, output)\*100)

# Decision tree to predict Accident Type

from sklearn import tree

model = tree.DecisionTreeClassifier()

inputs = data[['Injuries','Ship Type','Deaths','Rain','Ship Length (m)','Persons on board','Crew Members','Passengers','Successful evacuation','Location Type','Environmental Pollution']]

output = data[['Accident Type']]

model.fit(inputs, output)

print("Accuracy Using Decision tree is : ",model.score(inputs, output)\*100)

result = model.predict([[4,1,2,0,441,731,201,530,0,1,0]])

print(result)