In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
data = pd.read_csv("ICAO_accidents.csv")
```

In [3]:

```
data.head()
```

Out[3]:

	Unnamed: 0	Date	StateOfOccurrence	Location	Model	Registration	Operato
0	0	"2008-01-02T00:00:00.000Z"	PHL	Masbate Airport (MBT)	NAMC YS11 A	RP-C3592	Philippine Asia Spir
1	1	"2008-01-02T00:00:00.000Z"	IRN	Tehran-Mehrabad Airport (THR)	FOKKER F27 100	EP-IDB	Irar Islami Republi Of Ira Nationa Airlin.
2	2	"2008-01-03T00:00:00.000Z"	USA	Oklahoma City	PILATUS PC12	N398J	Nal
3	3	"2008-01-04T00:00:00.000Z"	VEN	A 20 NM del VOR del Gran Roque	LET L410 UVP	YV2081	Venezuel
4	4	"2008-01-05T00:00:00.000Z"	USA	Kodiak	PIPER PA31P 350	N509FN	Nal

5 rows × 25 columns

In [4]:

```
data.tail()
```

Out[4]:

	Unnamed: 0	Date	StateOfOccurrence	Location	Model	Registration	
6104	39	"2022-05-06T00:00:00.000Z"	ITA	Napoli-Capodichino Airport (NAP)	Boeing 737-82R (WL)	YR-BMM	
6105	40	"2022-05-06T00:00:00.000Z"	CHL	Santiago-Arturo Merino Benitez Airport (SCL)	Beechcraft 200 Super King Air	CC-CDY	
6106	41	"2022-05-11T00:00:00.000Z"	BRA	Boituva, SP	Cessna 208 Caravan I	PT-OQR	Sl Es
6107	42	"2022-05-11T00:00:00.000Z"	CMR	near Nanga Eboko	Viking DHC-6 Twin Otter 400	TJ-TIM	
6108	43	"2022-05-12T00:00:00.000Z"	CHN	Chongqing-Jiangbei International Airport (CKG)	Airbus A319-115 (WL)	B-6425	T

5 rows × 25 columns

In [5]:

```
data.shape
```

Out[5]:

```
(6109, 25)
```

In [6]:

```
data.columns
```

Out[6]:

```
Index(['Unnamed: 0', 'Date', 'StateOfOccurrence', 'Location', 'Model',
       'Registration', 'Operator', 'StateOfOperator', 'StateOfRegistry',
       'FlightPhase', 'Class', 'Fatalities', 'Over2250', 'Over5700',
       'ScheduledCommercial', 'InjuryLevel', 'TypeDesignator', 'Helicopter',
       'Airplane', 'Engines', 'EngineType', 'Official', 'OccCats', 'Risk',
       'Year'],
      dtype='object')
```

In [7]:

```
data.duplicated()
```

Out[7]:

```
0       False
1       False
2       False
3       False
4       False
        ...
6104    False
6105    False
6106    False
6107    False
6108    False
Length: 6109, dtype: bool
```

In [8]:

```
data.duplicated().sum()
```

Out[8]:

```
0
```

In [9]:

```
data.isnull().sum()
```

Out[9]:

```
Unnamed: 0               0
Date                     0
StateOfOccurrence      802
Location               351
Model                  243
Registration             0
Operator              1925
StateOfOperator       4718
StateOfRegistry          2
FlightPhase            934
Class                    0
Fatalities            1141
Over2250                 0
Over5700                24
ScheduledCommercial   2832
InjuryLevel           1955
TypeDesignator           0
Helicopter            1228
Airplane                 0
Engines                  0
EngineType               0
Official              4765
OccCats                  0
Risk                   750
Year                     0
dtype: int64
```

In [10]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6109 entries, 0 to 6108
Data columns (total 25 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Unnamed: 0           6109 non-null   int64
 1   Date                 6109 non-null   object
 2   StateOfOccurrence    5307 non-null   object
 3   Location             5758 non-null   object
 4   Model                5866 non-null   object
 5   Registration         6109 non-null   object
 6   Operator             4184 non-null   object
 7   StateOfOperator      1391 non-null   object
 8   StateOfRegistry      6107 non-null   object
 9   FlightPhase          5175 non-null   object
 10  Class                6109 non-null   object
 11  Fatalities           4968 non-null   float64
 12  Over2250             6109 non-null   bool
 13  Over5700             6085 non-null   object
 14  ScheduledCommercial  3277 non-null   object
 15  InjuryLevel          4154 non-null   object
 16  TypeDesignator       6109 non-null   object
 17  Helicopter           4881 non-null   object
 18  Airplane             6109 non-null   bool
 19  Engines              6109 non-null   int64
 20  EngineType           6109 non-null   object
 21  Official             1344 non-null   object
 22  OccCats              6109 non-null   object
 23  Risk                 5359 non-null   object
 24  Year                 6109 non-null   int64
dtypes: bool(2), float64(1), int64(3), object(19)
memory usage: 1.1+ MB
```

In [11]:

```
data.describe()
```

Out[11]:

	Unnamed: 0	Fatalities	Engines	Year
count	6109.000000	4968.000000	6109.000000	6109.000000
mean	268.898183	2.037641	1.754624	2012.993616
std	210.107085	12.957411	0.685855	3.918512
min	0.000000	0.000000	1.000000	2008.000000
25%	105.000000	0.000000	1.000000	2010.000000
50%	221.000000	0.000000	2.000000	2012.000000
75%	376.000000	0.000000	2.000000	2016.000000
max	931.000000	298.000000	6.000000	2022.000000

In [12]:

```
data = data[(data['ScheduledCommercial']==True) & (data['Airplane'] == True) & (data['Engin
```

In [13]:

```
data = data.drop(columns= ['Unnamed: 0', 'Date','StateOfOccurrence','Location','Model','Reg
                           'Operator','StateOfOperator','StateOfRegistry','Over2250',
                           'Over5700','Class','ScheduledCommercial','TypeDesignator',
                           'Helicopter','Airplane','Engines','EngineType','Official','OccCa
```

In [14]:

```
data.nunique()
```

Out[14]:

```
FlightPhase     9
Fatalities     68
InjuryLevel     5
Risk           47
Year           15
dtype: int64
```

In [15]:

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

Out[15]:

```
array([2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018,
       2019, 2020, 2021, 2022], dtype=int64)
```

In [16]:

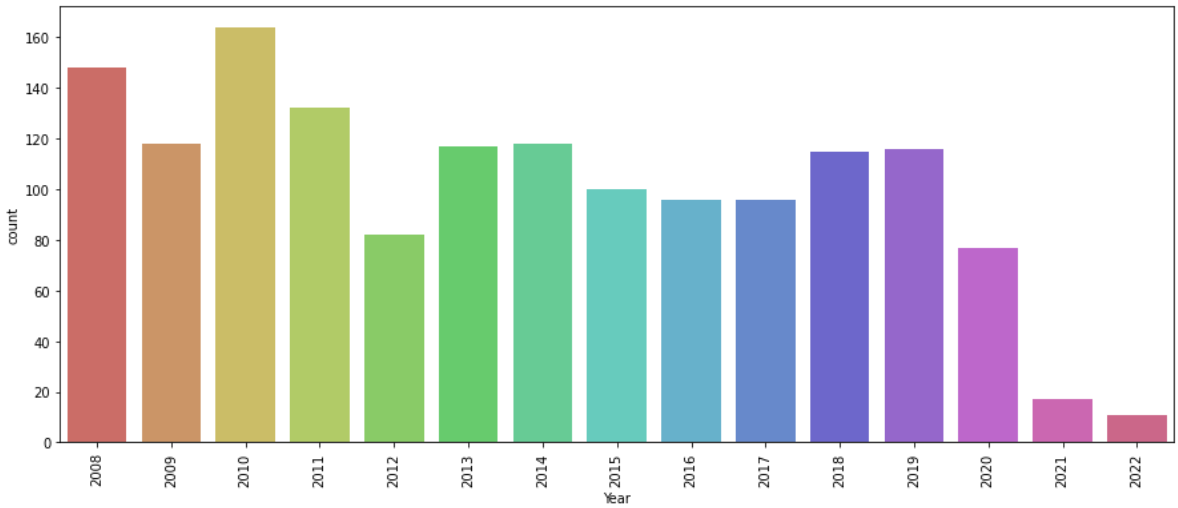
```
data['Year'].value_counts()
```

Out[16]:

```
2010    164
2008    148
2011    132
2009    118
2014    118
2013    117
2019    116
2018    115
2015    100
2016     96
2017     96
2012     82
2020     77
2021     17
2022     11
Name: Year, dtype: int64
```

In [17]:

```
plt.figure(figsize=(15,6))
sns.countplot('Year', data=data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [18]:

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

Out[18]:

```
array(['None', 'Fatal', 'Serious', 'Minor', 'Unknown', nan], dtype=object)
```

In [19]:

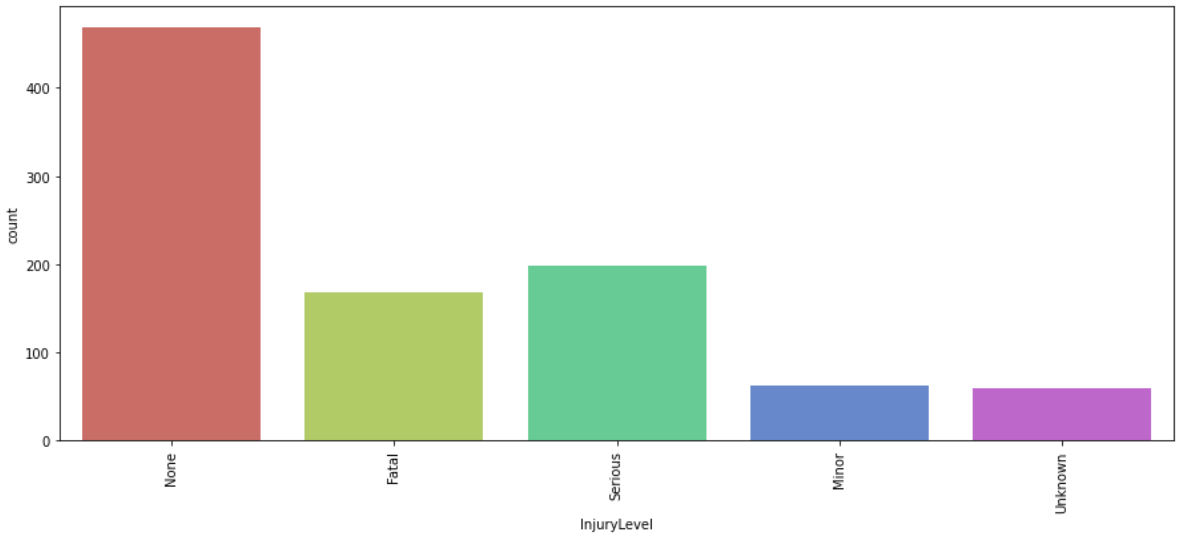
```
data['InjuryLevel'].value_counts()
```

Out[19]:

```
None       469
Serious    198
Fatal      168
Minor       62
Unknown     59
Name: InjuryLevel, dtype: int64
```

In [20]:

```
plt.figure(figsize=(15,6))
sns.countplot('InjuryLevel' , data=data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [21]:

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

Out[21]:

```
array(['Landing', 'Take-off', 'En route', 'Approach', 'Standing', 'Taxi',
       'Unknown', nan, 'Manoeuvring', 'Initial Climb'], dtype=object)
```

In [22]:

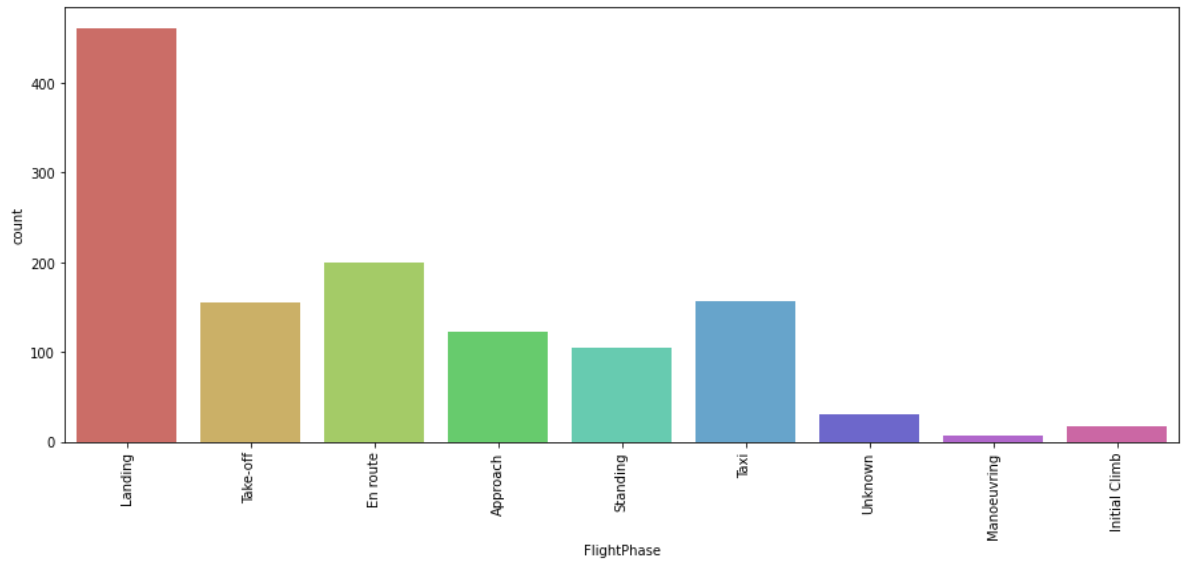
```
data['FlightPhase'].value_counts()
```

Out[22]:

```
Landing          461
En route         199
Taxi             156
Take-off         155
Approach         122
Standing         105
Unknown           30
Initial Climb     17
Manoeuvring        7
Name: FlightPhase, dtype: int64
```

In [23]:

```
plt.figure(figsize=(15,6))
sns.countplot('FlightPhase' , data=data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [24]:

```
data_avg = []
for x in data['Year']:
    if x == 2008:
        avg = round((148 * 1000000)/ 26500000,2)
    elif x == 2009:
        avg = round((118 * 1000000)/ 25900000,2)
    elif x == 2010:
        avg = round((164 *1000000)/ 27800000,2)
    elif x == 2011:
        avg = round((132 *1000000)/ 30100000,2)
    elif x == 2012:
        avg = round((82 *1000000)/ 31200000,2)
    elif x == 2013:
        avg = round((117 * 1000000)/ 32000000,2)
    elif x == 2014:
        avg = round((118 * 1000000) / 33000000,2)
    elif x == 2015:
        avg = round((100 * 1000000)/ 340000000,2)
    elif x == 2016:
        avg = round((96 *1000000) / 35200000,2)
    elif x == 2017:
        avg = round((96 * 1000000) / 36400000,2)
    elif x == 2018:
        avg = round((115 * 1000000)/ 38100000,2)
    elif x == 2019:
        avg = round((116 * 1000000)/ 38900000,2)
    elif x == 2020:
        avg = round((77 * 1000000)/ 16900000,2)
    elif x == 2021:
        avg = round((17 * 1000000)/ 19300000,2)
    else:
        avg = round((11 * 1000000)/10750000,2)
    data_avg.append(avg)
```

In [25]:

```
data_avg
```

Out[25]:

```
[5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58,
 5.58.
```

In [26]:

```
data['acc/1MillionFlights'] = data_avg
```

In [27]:

```
data['InjuryLevel'].isnull().sum()
```

Out[27]:

```
551
```

In [28]:

```
data['InjuryLevel'].fillna('unknown',inplace=True)
```

In [29]:

```
il_update = []
for x in data['InjuryLevel']:
    if x in ['Serious','Fatal']:
        x = 'Serious/Fatal'
    else:
        x = 'Minor/None'
    il_update.append(x)
```

In [30]:

```
data['InjuryLevel'] = il_update
```

In [31]:

```
data['InjuryLevel']=data['InjuryLevel'].apply(lambda x: 1 if x== 'Serious/Fatal' else 0)
```

In [32]:

```
data['Serious/Fatal'] = data['InjuryLevel']
```

In [33]:

```
data.drop(columns= ['InjuryLevel'], inplace=True)
```

In [34]:

```
data.head()
```

Out[34]:

	FlightPhase	Fatalities	Risk	Year	acc/1MillionFlights	Serious/Fatal
0	Landing	0.0	RS	2008	5.58	0
1	Take-off	0.0	OTH	2008	5.58	0
3	En route	14.0	SCF	2008	5.58	1
7	Approach	0.0	SCF	2008	5.58	0
8	Standing	0.0	RS	2008	5.58	0

In [35]:

```
data.drop(columns=['Fatalities'],inplace=True)
```

In [36]:

```
data['FlightPhase'].fillna('Unknown',inplace=True)
```

In [37]:

```
data['FlightPhase'].isnull().sum()
```

Out[37]:

```
0
```

In [38]:

```
data['FlightPhase'].value_counts().head()
```

Out[38]:

```
Landing     461
Unknown     285
En route    199
Taxi        156
Take-off    155
Name: FlightPhase, dtype: int64
```

In [39]:

```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix
```

In [40]:

```
labencoder = LabelEncoder()
labencoder.fit(data['FlightPhase'])
```

Out[40]:

```
▾ LabelEncoder

LabelEncoder()
```

In [41]:

```
data['FlightPhase'] = labencoder.fit_transform(data['FlightPhase'])
```

In [42]:

```
labencoder.classes_
```

Out[42]:

```
array(['Approach', 'En route', 'Initial Climb', 'Landing', 'Manoeuvring',
       'Standing', 'Take-off', 'Taxi', 'Unknown'], dtype=object)
```

In [43]:

```
labencoder = LabelEncoder()
labencoder.fit(data['Risk'])
```

Out[43]:

```
▾ LabelEncoder

LabelEncoder()
```

In [44]:

```
data['Risk'] = labencoder.fit_transform(data['Risk'])
```

In [45]:

```
col = ['Year','FlightPhase','Risk','Serious/Fatal','acc/1MillionFlights']
```

In [46]:

```
data = data.loc[:,col]
```

In [47]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1507 entries, 0 to 6108
Data columns (total 5 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Year                 1507 non-null   int64  
 1   FlightPhase          1507 non-null   int32  
 2   Risk                 1507 non-null   int32  
 3   Serious/Fatal        1507 non-null   int64  
 4   acc/1MillionFlights  1507 non-null   float64
dtypes: float64(1), int32(2), int64(2)
memory usage: 58.9 KB
```

In [48]:

```
data.isnull().sum()
```

Out[48]:

```
Year                   0
FlightPhase            0
Risk                   0
Serious/Fatal          0
acc/1MillionFlights    0
dtype: int64
```

In [49]:

```
X = data.drop(columns=['Serious/Fatal'])
y = data['Serious/Fatal']
```

In [50]:

```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=42)
```

In [51]:

```
X_train.shape
```

Out[51]:

```
(1205, 4)
```

In [52]:

```
y_train.shape
```

Out[52]:

```
(1205,)
```

In [53]:

```
X_train.head()
```

Out[53]:

	Year	FlightPhase	Risk	acc/1MillionFlights
3071	2012	1	31	2.63
4032	2015	7	30	0.29
3083	2012	4	30	2.63
4087	2015	7	6	0.29
4891	2017	1	30	2.64

In [54]:

```
y_train
```

Out[54]:

```
3071    0
4032    0
3083    0
4087    0
4891    0
       ..
4919    0
5392    0
3906    0
5840    0
4912    0
Name: Serious/Fatal, Length: 1205, dtype: int64
```

In [55]:

```
st_x= StandardScaler()
x_train= st_x.fit_transform(X_train)
x_test= st_x.transform(X_test)
```

In [56]:

```
classifier= LogisticRegression(random_state=0)
classifier.fit(x_train, y_train)
```

Out[56]:

```
LogisticRegression
LogisticRegression(random_state=0)
```

In [57]:

```
y_pred = classifier.predict(x_test)
```

In [58]:

```
print(classifier.score(x_train, y_train))
```

```
0.7858921161825726
```

In [59]:

```
print(classifier.score(x_test, y_test))
```

```
0.7748344370860927
```

In [60]:

```
print(classifier.score(x_test, y_pred))
```

```
1.0
```

In [61]:

```
cm= confusion_matrix(y_test, y_pred)
```

In [62]:

```
print(cm)
```

```
[[209  10]
 [ 58  25]]
```

In [63]:

```
classifier_dt= DecisionTreeClassifier(criterion='entropy', random_state=0)
classifier_dt.fit(x_train, y_train)
```

Out[63]:

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', random_state=0)
```

In [64]:

```
y_pred= classifier_dt.predict(x_test)
```

In [65]:

```
print(classifier_dt.score(x_train, y_train))
```

```
0.9236514522821577
```

In [66]:

```
print(classifier_dt.score(x_test, y_test))
```

```
0.8443708609271523
```

In [67]:

```
print(classifier_dt.score(x_test, y_pred))
```

```
1.0
```

In [68]:

```
cm= confusion_matrix(y_test, y_pred)
```

In [69]:

```
print(cm)
```

```
[[204  15]
 [ 32  51]]
```