Plane-Crash Analysis

Shivam Choudhary (150040082)

Aditya Pandey (150040117)

Introduction:

We tried to analyze the “Airplane Crashes and Fatalities Data since 1908” and draw-out some meaningful conclusions from them.

We tried to address the following questions:

Highest number of crash by Operators?

Causes of Crash?

Chances to survive a crash over the century?

We employed K-means Clustering Algorithm (Unsupervised Learning) and text processing in this project.

The entire code is made and compiled in Python 3.6.2 and run in Windows environment and the required packages (numpy, pandas, matplotlib, seaborn, etc.) of python are installed via Anaconda.

Github link : https://github.com/shivam19j/Plane-Crash-Analysis

Source of Data:

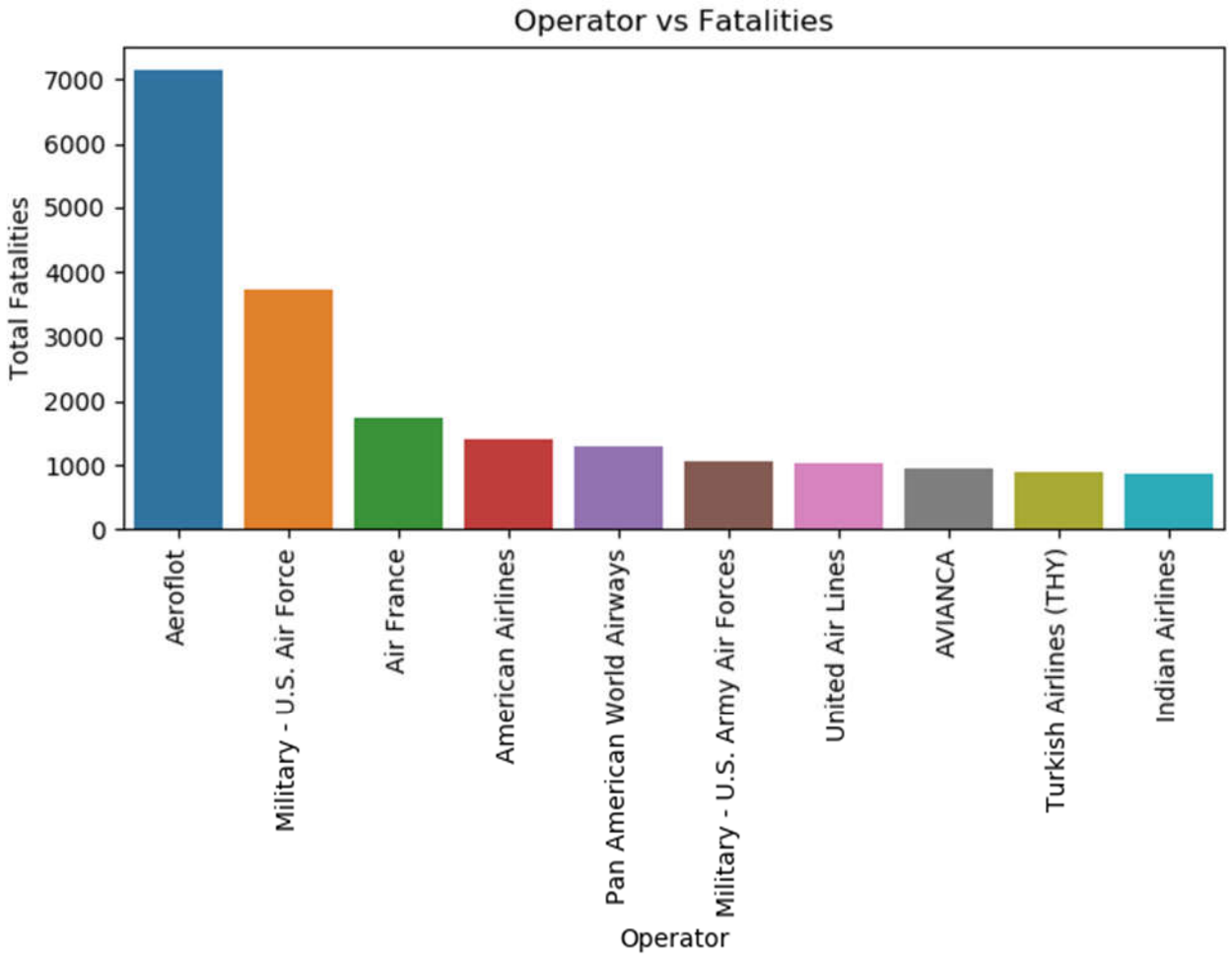
Public dataset: "Airplane Crashes and Fatalities Since 1908" (Full history of airplane crashes throughout the world, from 1908-present) hosted by Open Data by Socrata available at:

https://opendata.socrata.com/Government/Airplane-Crashes-and-Fatalities-Since-1908/q2te-8cvq

Operator vs Fatalities:

We first tried to find tried to see the number of fatalities corresponding to the operators.

```
operator_fatal = df[['Operator','Fatalities']].groupby(['Operator']).sum()
operator_fatal = operator_fatal['Fatalities'].sort_values(ascending=False)[:10]
operator_fatal_keys = operator_fatal.index
operator_fatal_val = operator_fatal.values
fig,ax = plt.subplots(figsize=(8,6))
sns.barplot(x = operator_fatal_keys,y =operator_fatal_val)
plt.title('Operator vs Fatalities')
plt.xlabel('Operator')
plt.ylabel('Total Fatalities')
ticks = plt.setp(ax.get_xticklabels(),rotation=90)
plt.show()
```



Clearly, Aeroflot suffered the maximum fatalities in the past century. This might be because of their higher capacity and higher frequency of usage. This is just a relative comparison.

Causes of Crash:

For investigating the cause of crash, we did some simple text processing on the ‘Summary’ field. We basically looked for the following causes: Weather, lost control, crash at landing, crash at taking off, air collision, shot down, mechanical failure, navigation error, human error, and terract.

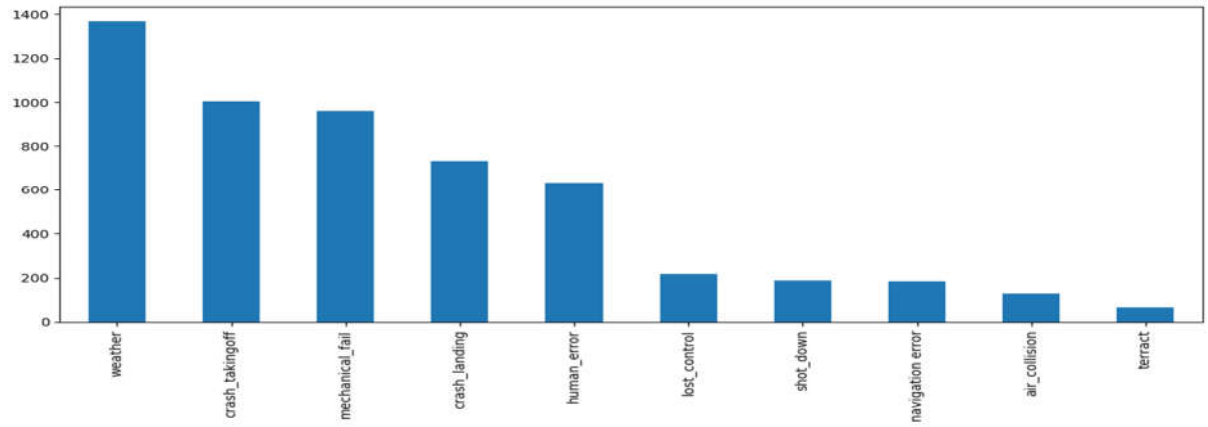
```
def investigate(s):
    data = {
        'weather': [
            'fog', ' rain', 'unlighted', 'thunder', 'turbulence', 'air pocket', 'adverse weather',
            'mist', 'weather conditions', 'storm', 'typhoon', 'icing', 'bad weather', 'poor weather',
            'meteorological conditions', 'head wind', 'lightning', 'weather was poor', 'snow',
            'weather related', ' ice'
        ],
        'lost_control': ['disorientation', 'low altitude', 'loss of control'],
        'crash_landing': ['short of the runway', 'attempting to land', 'on approach', 'final approach'],
        'crash_takingoff': ['taking off', 'takeoff', 'take off'],
        'air_collision': ['mid-air', 'in-flight collision', 'midair', 'planes collided'],
        'shot_down': ['shot down', 'missile', 'rebel', 'fighter'],
        'mechanical_fail': [
            'engine', 'propeller', 'mechanical failure', 'rotor', 'out of fuel', 'system failure', 'component failure',
            'fatigue'
        ],
        'navigation error': [
            'navigational error', 'disoriented', 'altimiter', 'poor visibility', 'altimeter',
            'compass', 'gyros', 'navigational equipment', 'erroneous navigation'
        ],
        'human_error': [
            'failure of the crew', 'pilot error', 'did not follow', 'crew ignored', 'failure to',
            'delayed landing', 'overloaded', 'misinterpretation', 'misjudge', 'failed to', 'lost control',
            'inadequate risk', 'improper use', 'midjudge', 'poor crew'
        ],
        'terract': ['bomb', 'hijacker']
    }

    res = []
    for el, words in data.items():
        res += [el for word in words if word in s]

    return list(set(res))

df['Summary'].fillna('', inplace=True)
all_values = []
for s in df['Summary']:
    all_values += investigate(s.lower())

plt.figure(figsize=(20, 6))
pd.DataFrame(all_values)[0].value_counts().plot('bar')
plt.show()
```

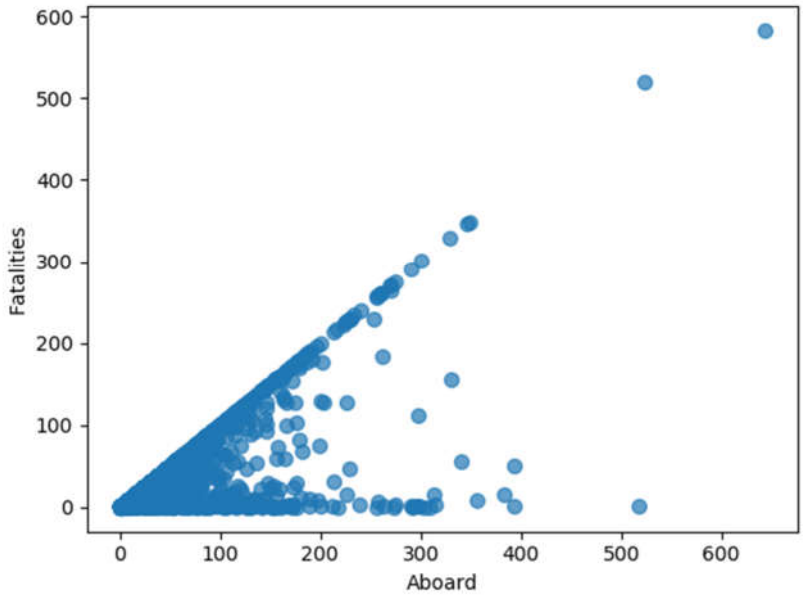


From the graph obtained it is quite clear that weather conditions are most likely to cause a crash and terract(bomb, hijacker) the most unlikely.

Likelihood of Surviving a Crash (over the century):

We first considered the number of fatalities for each crash vs the number of passengers.

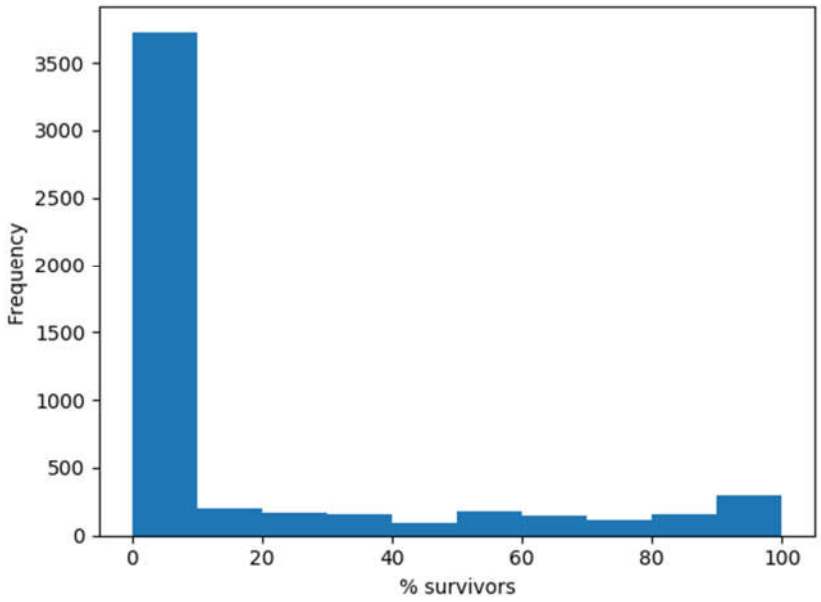
```
7   plt.scatter(df['Aboard'], df['Fatalities'], alpha=0.7, s = 50)
8   plt.xlabel('Aboard')
9   plt.ylabel('Fatalities')
10  plt.show()
```



It is quite clear from the plot that either all or no one survive in a crash, just as we would expect in a plane crash.

Now, we check for the fraction of survivors for the crashes.

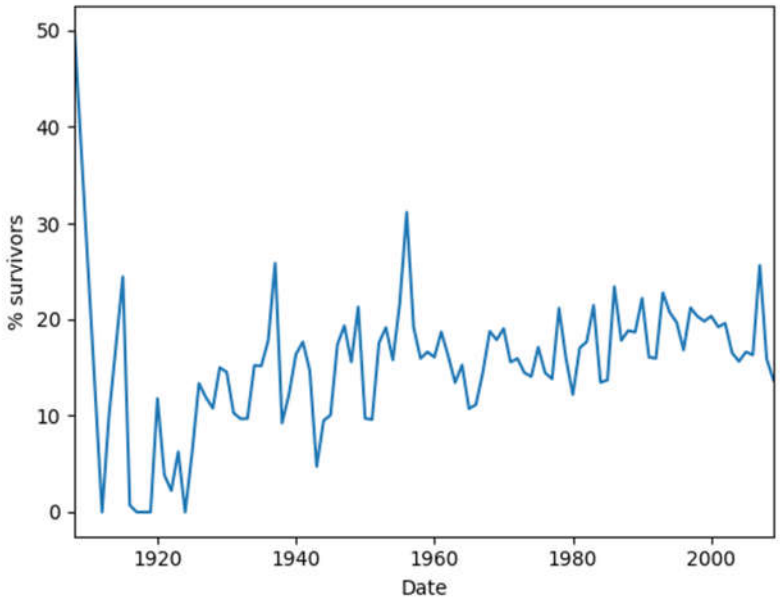
```
12  df['survivors'] = 100*(df.Aboard - df.Fatalities)/df.Aboard
13  df['survivors'].dropna().plot(kind='hist')
14  plt.xlabel('% survivors')
15  plt.show()
```



The obtained histogram clearly shows that nobody survives a crash.

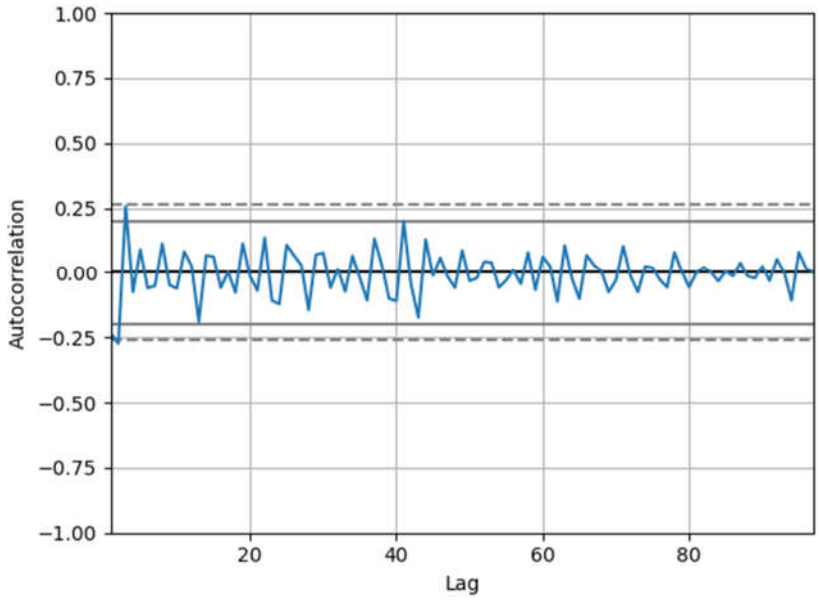
Now, we check whether the fraction of survivors increased over time. We considered the mean fraction of survivors per year, and analyzed the corresponding series.

```
17  df['Date'] = pd.to_datetime(df['Date'])
18  survivors_series = df.groupby(df['Date'].dt.year)['survivors'].mean()
19  survivors_series = pd.Series(survivors_series, index=survivors_series.index)
20  survivors_series.dropna().plot()
21  plt.ylabel('% survivors')
22  plt.show()
```



There seems to be a trend towards increasing fraction of survivors, but it is not clear whether it is a deterministic or stochastic trend. To verify this, we looked at the correlogram of the differenced series.

```
24  from pandas.tools.plotting import autocorrelation_plot
25  autocorrelation_plot(survivors_series.diff().dropna())
26  plt.show()
27
```



The correlogram is consistent with white noise. Thus, the original series is well approximated by a random walk.

The time series of the mean fraction of survivors per year is consistent with a random walk, suggesting that the chances to survive an airplane crash have not substantially increased over the last century.

Outliers:

We performed K-means Clustering over the dataset and choose cluster size to be 3. We first converted the data into the required form and the fed it into the K-means Clustering Algorithm via a Pipeline which ensures the automatic flow of data between different steps.

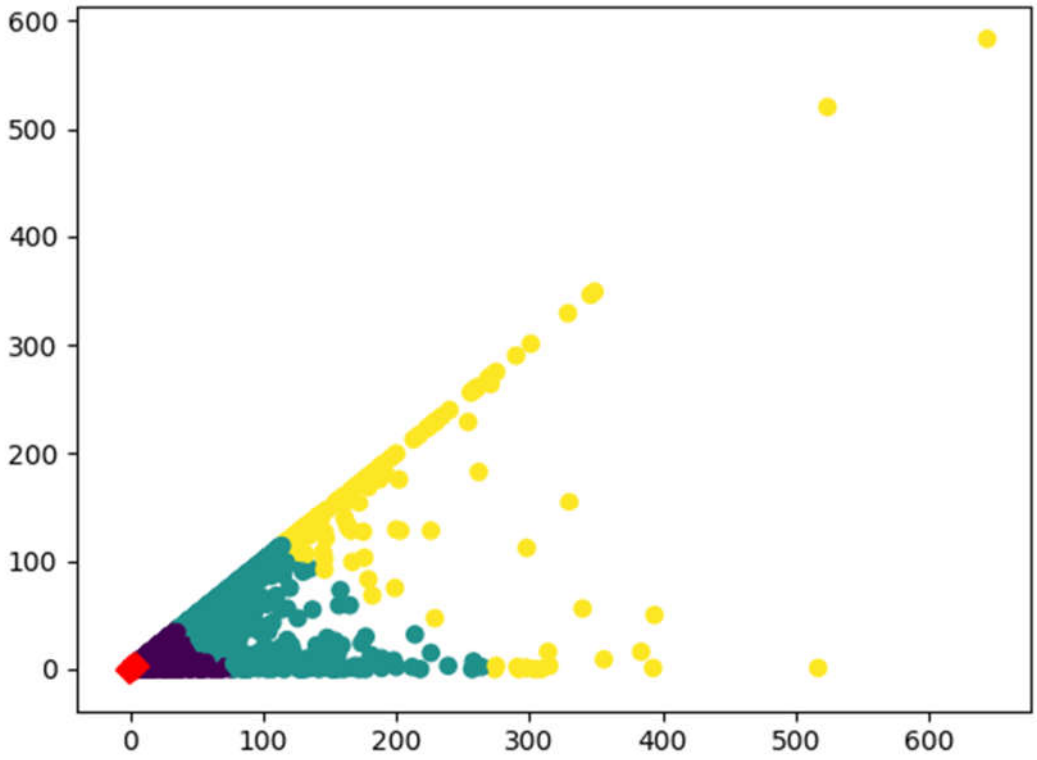
K-means Clustering Algorithm - The algorithm iterates between two steps:

1. Data Assignment Step: Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance.

2. Centroid update step: In this step centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster

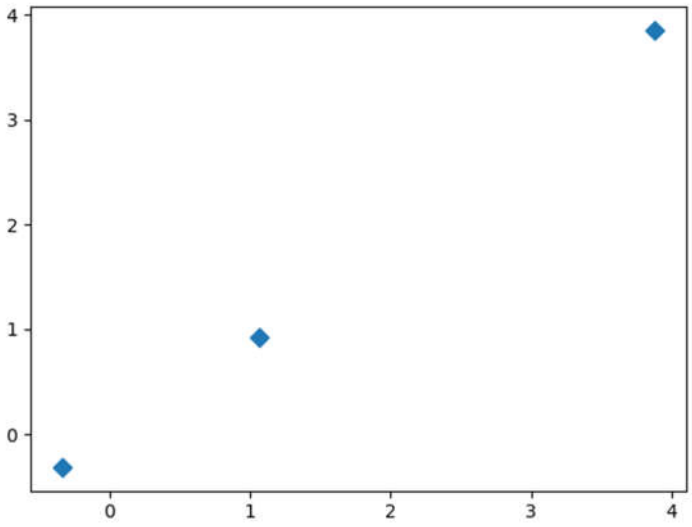
The algorithm iterates between steps 1 and 2 until a stopping criteria is met (i.e. no data points change clusters).

```
scaler = StandardScaler()
kmeans = KMeans(n_clusters=3)
model = make_pipeline(scaler, kmeans)
model.fit(samples)
labels = model.predict(samples)
xs = samples[:,0]
ys = samples[:,1]
plt.scatter(xs, ys, c=labels)
centroids = kmeans.cluster_centers_
centroids_x = centroids[:,0]
centroids_y = centroids[:,1]
plt.scatter(centroids_x,centroids_y,color='Red',marker='D',s=50)
print(kmeans.inertia_)
plt.show()
```



Now, plotting the Cluster centers.

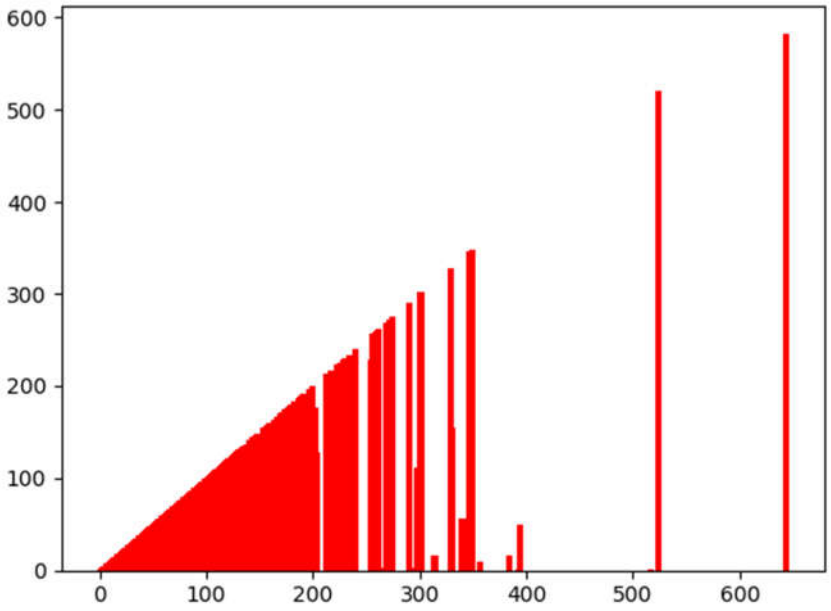
```
centroids = kmeans.cluster_centers_
centroids_x = centroids[:,0]
centroids_y = centroids[:,1]
plt.scatter(centroids_x,centroids_y,marker='D',s=50)
plt.show()
```



Inertia value = 3258.7557904

Inertia value gives us the sum of the distance of each value from the center of it's cluster.

```
60    fig,ax = plt.subplots()
61    ax.bar(xs, ys, width=6, color='r')
62    plt.show()
```



It's quite clear that there are many outliers in the data. These are the miracle cases when some of the passengers managed to survive a crash. These are the same crashes that lie in cluster number 3 in the K-means Cluster indicating a outlier trend.

Conclusion:

It’s quite clear that the chances of survival hasn’t changed much with the advancements of technology which is of great concern and weather conditions have been the major cause of crashes ever since the first flight. We need to find a way to deal with them in order to minimize human losses.