Capstone Project 1 Final Report

1. Updated Problem Statement:

Aerospace is the one of the biggest industries in the world. Passengers safety perception is extremely effective on the investment's fate. In order to protect the customers' health and comfort, and the companies' investments, flight safety is always priority in the market. The statistical researches on this topic was not analysed through the view of a data scientists lately. In the researches, simpler models used with basic statistical approach. Searching for evidence and identifying the real issues is not that easy if we are talking about flight safety. It is a culture for the aviators. So, it is very similar to analyze texts with NLP to investigate a real safety issue. My original goal at the beginning was to contribute to aircraft and aviation safety management systems by analyzing and predicting one of the following questions precisely:

- By providing the conditions (such as pilot experience, weather, aircraft and mission type, phase of flight, location) under the classification of consequences and factors of the issue, can we predict how likely to have a flight safety issue?
- Which parts of the flight have more risk in the terms of safety issue?
- Is there a seasonal pattern on safety incidents depending on date, location or weather?

The results can contribute to pilot training, aircraft design, risk assessment of operations in airline companies. A proactive approach can be developed for air traffic control system as well as a reasonable allocation of resources can be conducted for safety purposes. But after having further analysis; it turns out that the dataset I have is not the best fit for that questions. The first step should be to understand what are the basic categories for safety issues and what are the root causes to that undesired safety incidents. Having said that, the problem statement after data cleaning and data wrangling processes will be modified to the following statement:

- Can we predict the reason of air conflict safety issues by analyzing crew statements?

And the updated title of the project will be;

- A classification model on causes of air conflicts by air crew narratives.

2. Obtaining Data:

Aviation Safety Network is providing up-to-date, complete and reliable authoritative information on airliner incidents and safety issues to everyone. The Aviation Safety Reporting System (ASRS) database is the world's largest repository of voluntary, confidential safety

information provided by aviation's frontline personnel, including pilots, controllers, mechanics, flight attendants, and dispatchers. The database provides a foundation for specific products and subsequent research addressing a variety of aviation safety issues.

ASRS's database includes the narratives submitted by reporters (after they have been sanitized for identifying details). These narratives provide an exceptionally rich source of information for research. The database also contains coded information by expert analysts from the original report which is used for data retrieval and analyses.

https://asrs.arc.nasa.gov/overview/database.html

https://asrs.arc.nasa.gov/search/database.html

Following variables are listed in the database:

Report number (ACN), Month of incident, Location, State, Flight Conditions, Weather, Reporter Organization, Reporter Function, Flight Plan, Mission, Flight Phase, Aircraft Model, Airspace Class, Event Type, Anomaly Issue, Anomaly Procedure, Detector, Primary Problem, Contributing Factors, Human Factors, Result, Narrative, Synopsis.

One missing but pretty important variable would be the aging of aircraft and engine. Due to confidentially the data set is not including these variables. But partially maintenance condition is included.

I downloaded a data with approximately 200.000 observations in 100 columns in which has texts in some of them. I am following the explained steps below:

- The website had a limitation of 5000 observation for each download. In the query page, I chose date criteria for all data to be downloaded. The first file was 2019. Then I chose January-June in the first chunk, and July-December in the second chunk each year. As a result, except 2019, 1999, 1995 and 1988, all years have two files with approximately 3500 observations. 1999 and 1995 have three files. In total 64 csv files obtained with exact same formatting.
- All files were combined by using following commands with a for loop and can be executed like;

Read from 64 csv files & Form a Dataframe

```
In [122]:
filenames=glob('database/*.csv')
data=[]
for f in filenames:
    df=pd.read_csv(f, header=[0,1], delimiter=',')
    data.append(df)

In [123]:
final_df = pd.concat(data, axis=0, ignore_index=True)
final_df.shape

Out[123]: (202525, 97)
```

3. Data Cleaning & Data Wrangling:

- I got rid of the duplicates and unnecessary columns with the following command;

```
df_nodup=final_df.drop_duplicates()
df_nodup.shape
df1=df_nodup.drop(columns=['Work Environment Factor','RVR.Single Value','Aircraft Zone', 'Maintenance Status.Maintenance Deferred','Maintenance Status.Records Complete', 'Maintenance Status.Released For Service','Maintenance Status.Required / Correct Doc On Board','Maintenance Status.Maintenance Type','Maintenance Status.Maintenance Items Involved', 'Cabin Lighting','Crew Size Flight Attendant.Number Of Crew','Callback','When Detected'], level=1)

df1=df_nodup.drop(columns=['Report 2','Unnamed'], level=0)
```

- In order to classify the data frame according to “Events” column, I created multiple data frames. The classification of the data is as follows:
    a. Airspace conflict issues,
    b. Abnormal equipment/activity due to emergency situation,
    c. Flight cabin event,
    d. Ground issues.

  In events column 55 different explanation word taxonomy used. Each taxonomy with different combination falls under one of the classifications above. I wrote a function for this categorization as below and applied it for Anomaly column:

```
Types=['Ground','Emergency','Conflict','Cabin']
def categorize(dataframe,column_name,new_column='Type'):
    dataframe[new_column]='Not Categorized'
    dataframe.index.fillna('Empty')
    dataframe.fillna('Empty')
    for index, row in dataframe.iterrows():
        if 'Equipment' in dataframe[column_name][index]:
            dataframe[new_column][index]='Emergency'
        elif 'Ground' in dataframe[column_name][index]:
            dataframe[new_column][index]='Ground'
        elif 'Clearence' in dataframe[column_name][index]:
            dataframe[new_column][index]='Conflict'
        elif 'Conflict' in dataframe[column_name][index]:
            dataframe[new_column][index]='Conflict'
        elif 'Deviation' in dataframe[column_name][index]:
            dataframe[new_column][index]='Conflict'
        elif 'Cabin' in dataframe[column_name][index]:
            dataframe[new_column][index]='Cabin'
        else:
            dataframe[new_column][index]='Not Categorized'
```

- I created a new dataframe with only “conflict” kind of safety issues which are including air traffic conflicts.

Conflict	91669
Emergency	65320
Ground	23084
Not Categorized	20544
Cabin	1908

- In order to get rid of two-level column names and get more useful column names, I applied the following code:

```
column_names=list(df_c1.columns)
del column_names[-1]
del column_names[-1]
df_new={}
for l1,l2 in column_names:
    if df_c1[l1,l2].dtype=='object':
        df_t=df_c1[l1,l2].fillna('None')
        column_t=str(l1)+'_'+str(l2)
        df_new[column_t]=df_t
    if df_c1[l1,l2].dtype=='float64' or df_c1[l1,l2].dtype=='int64':
        df_t=df_c1[l1,l2].fillna(0)
        column_t=str(l1)+'_'+str(l2)
        df_new[column_t]=df_t
len(df_new)
column_new_names=['_No','Month', 'Time', 'Place_refer', 'State', 'Radial', 'Distance', 'AGL',
'MSL', 'MCondition', 'Visibility','Light', 'Ceiling', 'AC1_ATC', 'AC1_Operator', 'AC1_Model',
'AC1_Crew', 'AC1_Rule', 'AC1_FP', 'AC1_Mission', 'AC1_Nav', 'AC1_Phase', 'AC1_Route',
'AC1_Airspace', 'AC1_Seats','AC1_Passengers','AC2_ATC','AC2_Operator', 'AC2_Model',
'AC2_Crew', 'AC2_Rule', 'AC2_FP', 'AC2_Mission', 'AC2_Nav', 'AC2_Phase',
'AC2_Route','AC2_Airspace','AC2_Seats', 'AC2_Passengers', 'P1_Loc', 'P1_Org', 'P1_Func',
'P1_Qual','P1_Experience','P1_HumaFactor','P2_Loc','P2_Org', 'P2_Func', 'P2_Qual',
'P2_Experience', 'Anomaly', 'Detector', 'Result', 'Other_Factors', 'Pri_Problem', 'Narrative',
'Synopsis']
data_clean.columns=column_new_names
data_clean.head()
```

- For missing values; I used the “categorize” function that I defined in 6th bullet. If the Dtype of the column is object then the empty cells will be filled with “None”. It the column is float or integer the empty cells are filled with 0. The concerning lines are as follows:

```
if df_c1[l1,l2].dtype=='object':
        df_t=df_c1[l1,l2].fillna('None')
        column_t=str(l1)+'_'+str(l2)
        df_new[column_t]=df_t
    if df_c1[l1,l2].dtype=='float64' or df_c1[l1,l2].dtype=='int64':
        df_t=df_c1[l1,l2].fillna(0)
        column_t=str(l1)+'_'+str(l2)
        df_new[column_t]=df_t
```

- For each column I applied different codes to clean data. Outliers of each column treated differently. If there is an error, it needs to be checked individually. Examples of some columns are as follows:

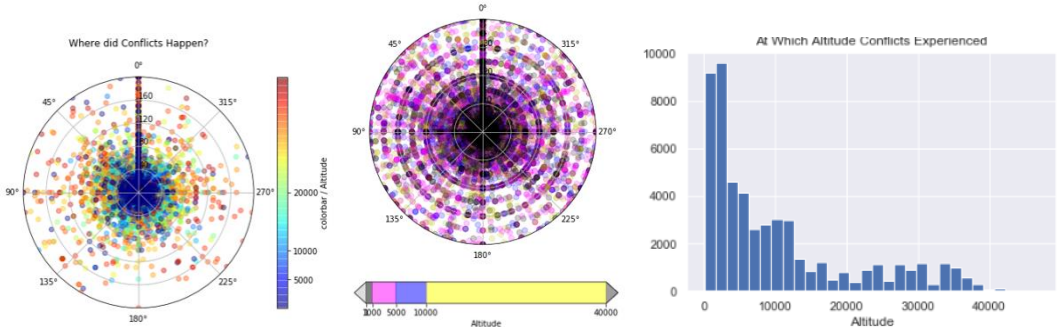
```
empty_time=['None','ZZZ']
for item in empty_time:
   for index,row in df.iterrows():
      if df.Time[index]==item:
         if df.Light[index]=='Daylight':
            df.Time[index]=='0601-1200'
         elif df.Light[index]=='Night':
            df.Time[index]=='0001-0600'
……..
df.AGL=pd.to_numeric(df.AGL,errors='coerce')
df.MSL=pd.to_numeric(df.MSL,errors='coerce')
……..
df.State=df.State.str.upper()
df[df.State=='US.AIRPORT']['State']
df.loc[40984, 'State']='US'
……..
position=df[~((df.Distance==0) & (df.Radial==0))][['Distance','Radial','Altitude']]
……..
Phase_df=df.AC1_Phase.str.split(';')
Phase_df=Phase_df.str.get(0)
df['AC1_Phase']=Phase_df
df.AC1_Phase.value_counts().head(11)
……..
Airspace=df[~(df.AC1_Airspace=='None')]['AC1_Airspace'].value_counts()
#Airspace.index=Airspace.index.str.split('')
Airspace.index=Airspace.index.str.get(6)
Airspace=Airspace.groupby(Airspace.index).sum().sort_values(ascending=False).head(6)
Airspace.index='Class '+Airspace.index

    Airspace= Airspace.sort_index()
```

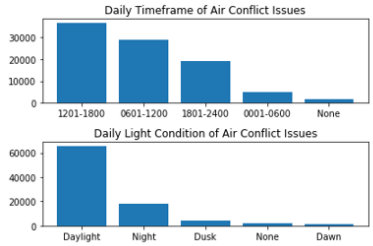
4. Initial Findings:

- After classifying the safety incidents in four categories, air conflict issues were being chosen and the columns which has less than 10% information dropped. As a result, the new data set shape is 91670 x 58.
- Then I started to research on altitude, location (radial, distance columns), time, light, phase, airspace class, month, year, state, primary cause and narrative (synopsis) variables.
- Between all those variables there were inconsistencies because of having empty cells in each questionnaire. The tendency of the crew was using narrative and synopsis sections but arbitrary inputs for the remaining cells. For that reason, most of the variables has a big chunk of “None”, “0” or “NaN” inputs.
- Radial, distance and altitude show us where the exact location of conflict issue. At least one of this three variables might have a “0” value since it was empty in original input file.

I tried to visualize those three on a polar chart. “0” radial, “0” distance or “0” altitude has more data then the others. It creates a kind of unreliability on those data.

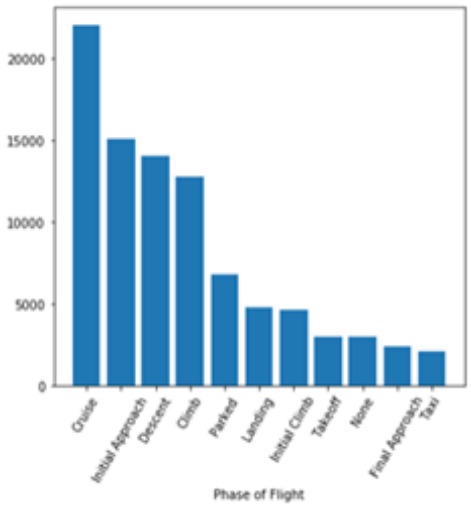


- One other interaction analysis between the variables was time and light variables. Those two has similar characteristics by nature. The effects on the conflict issues should be same and this makes it redundant.



- One big problem for this dataset is all information belongs to conflict issues. And we do not have the information of other flights which has no records of safety issues. So, for dependant column all records should be “1” and obviously this is not an appropriate target variable. To predict the safety issues won’t be possible with the current database. Then, I focused on “Phase”, “Anomaly” and “Pri_Problem” columns as the dependant variables.

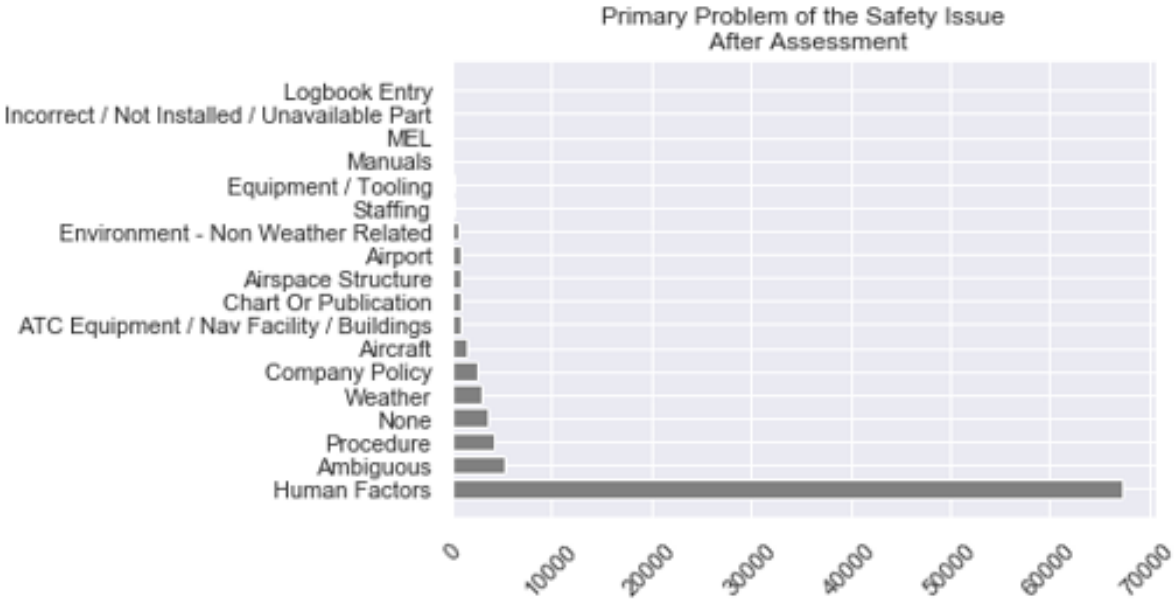
    a. Phases: This variable represents the phase of flight such as “take-off”, “taxi”, “cruise” or “landing” and is highly inter-related with the other location variables and has no relationship with time, light, meteorology… etc. variables. Additionally, to predict this variable by location variables won’t provide any kind of value-added to the customer. Because location variables are showing the parameters relative to airports. It is a different way to define phase.



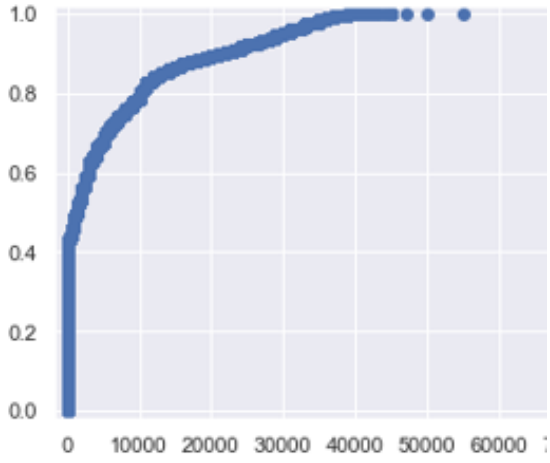
    b. Anomaly: During the data wrangling part, the anomaly variable was used to categorize the data and between four categories “air

conflict" issues were selected. And now the target column become more homogenous. Also, other groups of variables are not strongly related with "Anomaly" column.

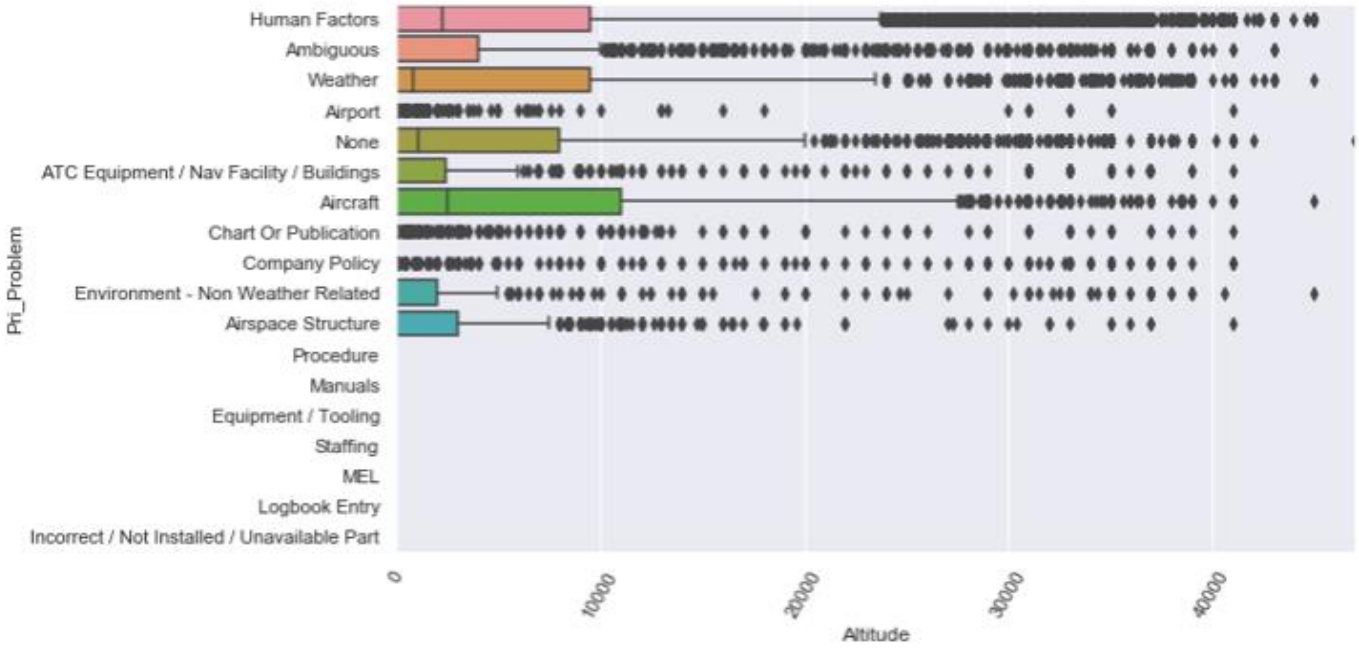
c. Pri_Problem: This variable represents the root cause of the conflict issue. The biggest chunk of data is Human Factors. This variable is the best candidate for target column. Because, firstly there is a huge value-added in this research to understand the safety issue's reason. Second, the data in this column is pretty consistent and pure analyzed through human experiences. The next step should be to understand the relations with other variables.

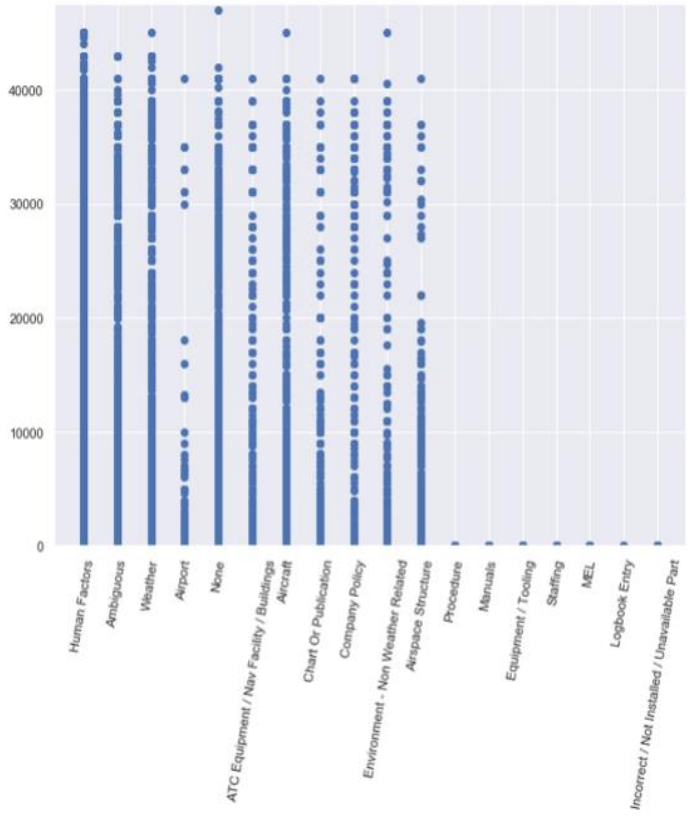


- If we analyse the relationship between independent variables and Pri_Problem column, the findings were not so strong. As an example, I will show "Altitude" variable distribution on target column.
- Emprical Cumulative Distribution Function of "Altitude" column is depicted below:



- The boxplot and scatterplot of Altitude and Pri_Problem column is as follows:





As a result; the most suitable model for this project would provide the root cause analysis for aviation safety issues. Pri_Problem column will be the target (dependant) variable. But still there is not a very powerful connection with independent variable and dependant variable. Further research needed for a better machine learning model. Other options can be “Narrative” and “Synopsis” columns as the independent variables. But those columns are not numerical and the data is formed by texts. My decision will be going on those variables to find a decoding as numerical and formulate a relationship with the target column.

The further stage for this study will be NLP model on Narrative and Synopsis columns to predict primary cause of Air Conflict Safety Issues.

5.NLP Model:

The target (dependent) variable will be “Pri_Problem” column. This column will be classified by using text columns. There are 17 different reason for air conflict issues. The classification of the variable and value counts as below:

```
Human Factors                                67311
Ambiguous                                     5259
Procedure                                     4122
None                                          3598
Weather                                       3017
Company Policy                                2582
Aircraft                                      1428
ATC Equipment / Nav Facility / Buildings       910
Chart Or Publication                           894
Airspace Structure                             881
Airport                                        799
Environment - Non Weather Related              553
Staffing                                       109
Equipment / Tooling                            104
Manuals                                         80
MEL                                             12
Incorrect / Not Installed / Unavailable Part     8
Logbook Entry                                    2
Name: Pri_Problem, dtype: int64
```

For the model development; there are three questions to be answered. I will show the questions below and will give the answers by explaining the progressive parts of the project.

Question 1: There are two text columns for the air conflict narrative. One is “Synopsis” and the other one is “Narrative”. Narrative column is the exact text from each report filled by the crew. In general, this column is longer than the Synopsis column. Synopsis is a quality summary of Narrative text. This column is focusing the most critical information of the incident and represents the version of Narrative column in which story part was chopped off. Now the first question is which column should be used for a better NLP model for the classification purposes.

Question 2: There are 17 categories in Pri_Problem column. The classification model can be run either by each category or a multiclass model. As it is seen above the amount of each class is either very low or very high. When we compare “Human Factors” with “Logbook Entry” won’t be reasonable. I will try binomial models for each individual category and try a top 3 and top 6 multiclass model.

Question 3: Which classifier should be use for best performance? The classifiers which I will use for multiclass will be MNB, TFIDF, Logistic Regression, Decision Tree, Random Forest, Gradient Boosting Model. The best performer model will run with hyper-tuning with parameters and bag of words vectorizer.

6. Findings:

Firstly, I create a copy of the dataframe and run a Multinomial Naïve Bayes model for only "Human Factors" values in the target column and tried to predict it in a model. In first run I used Synopsis column and in the second run I tried Narrative column. The accuracy scores for both training set and test set was approximately 80% for both columns. However, Synopsis column resulted better almost 1.5%. Most likely for Narrative column there were more noise than the Synopsis column. The results are as below for Synopsis and Narrative column respectively:

```
Accuracy Scores for the Training set : 0.8058524874693831  and Test Set : 0.79218075845886
True
```

```
Accuracy Scores for the Training set : 0.790215575272916  and Test Set : 0.7801074861857543
True
```

As a result, I decided to use Synopsis column for the remaining parts of the model.

As the next step I run MNB models for each category. The results as accuracy score was very high, however the amount of true positive was very low for the categories other than top three. As an example, I will depict the results for "Environment – Non Weather Related" category below:

```
Accuracy Scores for the Training set : 0.9936414215964574  and Test Set : 0.9929982590265688
True
[[26237    22]
 [  163     0]]
```

So, the answer for the second question will be Multiclass classification. I will run that for both top 3 and top 6 categories.

Up to this step, we decided to use "Synopsis" column with a multiclass classifier for the air conflict issues dataframe. In the progressive steps, for top 3 categories I will run MNB, TFIDF, Logistic Regression, Decision Tree, Random Forest, Gradient Boosting Model, will find the best performer and I will compare the model with top 6 category model and choose one. And lastly I will hypertune the model with bag of words and other parameters.

In order to restructure the dataframe the steps will be executed as it is exhibited in the below picture.

TOP 3 Classification

In [13]:
```
top3=['Human Factors', 'Ambiguous','Procedure']
df_top3=df[df.Pri_Problem.isin(top3)]
```

In [14]:
```
df_top3.shape
```

Out[14]: (76692, 3)

In [15]:
```
df_top3.head()
```

Out[15]:

idx	Pri_Problem	Synopsis	Narrative
1	Human Factors	SMA PENETRATED TCA ON CLIMB OUT.	THIS WAS MY FIRST DEP FROM BFI ON 31L. MY TURN...
2	Human Factors	SMT PLT DESCENDED TO ARPT UNDERLYING TCA; ACCU...	A VFR FLT; BEING CONDUCTED UNDER FAR PART 91; ...
6	Human Factors	LESS THAN STANDARD SEPARATION BETWEEN TWO ACR ...	ACR Y CLIMBING TO FL210 WAS STOPPED AT 160 FOR...
7	Human Factors	ACR LTT LANDED AT THE WRONG ARPT; DESTINATION ...	SOME TIME HAD PASSED AFTER WE HAD PASSED THE R...
8	Human Factors	LESS THAN STANDARD SEPARATION BETWEEN FLT OF 2...	ACFT X ON FINAL FOR RWY 30L (FLT OF 2) MISSED ...

In [16]:
```
df_top3.Pri_Problem.unique()
```

Out[16]: array(['Human Factors', 'Ambiguous', 'Procedure'], dtype=object)

Results for MNB;

```
Accuracy Scores for the Training set : 0.8250689218389091  and Test Set : 0.8140212100139083
True
[[  467   862   253]
 [ 1427 17276  1502]
 [  217    18   986]]

Classification Report
              precision    recall  f1-score   support

    Ambiguous       0.22      0.30      0.25      1582
Human Factors       0.95      0.86      0.90     20205
    Procedure       0.36      0.81      0.50      1221

    micro avg       0.81      0.81      0.81     23008
    macro avg       0.51      0.65      0.55     23008
 weighted avg       0.87      0.81      0.83     23008
```

Results for TFIDF;

```
Accuracy Scores for the Training set : 0.8832054243349974  and Test Set : 0.8786509040333796
True
[[    2  1577     6]
 [    6 20143    34]
 [    2  1167    71]]

Classification Report
              precision    recall  f1-score   support

    Ambiguous       0.20      0.00      0.00      1585
Human Factors       0.88      1.00      0.94     20183
    Procedure       0.64      0.06      0.11      1240

    micro avg       0.88      0.88      0.88     23008
    macro avg       0.57      0.35      0.35     23008
 weighted avg       0.82      0.88      0.83     23008
```

Results for Logistic Regression;

```
Accuracy Scores for the Training set : 0.9219506743163699  and Test Set : 0.885778859527121
True
[[  207  1287    91]
 [  250 19642   291]
 [  112   597   531]]

Classification Report
               precision    recall  f1-score   support

    Ambiguous       0.36      0.13      0.19      1585
Human Factors       0.91      0.97      0.94     20183
    Procedure       0.58      0.43      0.49      1240

    micro avg       0.89      0.89      0.89     23008
    macro avg       0.62      0.51      0.54     23008
 weighted avg       0.86      0.89      0.87     23008
```

Results for Decision Tree;

```
Accuracy Scores for the Training set : 0.8861113180836003  and Test Set : 0.882519123783032
True
[[   24  1535    26]
 [   13 19978   192]
 [   26   911   303]]

Classification Report
               precision    recall  f1-score   support

    Ambiguous       0.38      0.02      0.03      1585
Human Factors       0.89      0.99      0.94     20183
    Procedure       0.58      0.24      0.34      1240

    micro avg       0.88      0.88      0.88     23008
    macro avg       0.62      0.42      0.44     23008
 weighted avg       0.84      0.88      0.84     23008
```

Results for Random forest and GBM;

```
Accuracy Scores for the Training set : 0.9988637210342002  and Test Set : 0.8858657858136301
True
[[   33  1549     3]
 [   16 20105    62]
 [   17   979   244]]

Classification Report
               precision    recall  f1-score   support

    Ambiguous       0.50      0.02      0.04      1585
Human Factors       0.89      1.00      0.94     20183
    Procedure       0.79      0.20      0.32      1240

    micro avg       0.89      0.89      0.89     23008
    macro avg       0.73      0.40      0.43     23008
 weighted avg       0.86      0.89      0.84     23008


Accuracy Scores for the Training set : 0.8949221369495567  and Test Set : 0.8867350486787204
True
[[   80  1483    22]
 [   55 19939   189]
 [   45   812   383]]

Classification Report
               precision    recall  f1-score   support

    Ambiguous       0.44      0.05      0.09      1585
Human Factors       0.90      0.99      0.94     20183
    Procedure       0.64      0.31      0.42      1240

    micro avg       0.89      0.89      0.89     23008
    macro avg       0.66      0.45      0.48     23008
 weighted avg       0.85      0.89      0.85     23008
```

Random Forest and GBM are chosen as the best performer models. The accuracy score of the test set for the Top 6 category model decreased 6%.

3. Conclusion:

Eventually, our model will be formed by the following criteria:

- NLP text will be gathered from “Synopsis” column.
- A multiclass classifier model will be used.
- Random Forest classifier and Gradient Boosting classifier will be used.
- Top three categories will be used for the classification problem.

As the last step by using grid search random forest and GBM hyper tuned for the following parameters:

```
pipeline = Pipeline([
    ('vect', CountVectorizer()),
    #('gbm', GradientBoostingClassifier()),
    ('rfc', RandomForestClassifier()),
])

parameters = {
    'vect__ngram_range': [(1,1), (1, 2)],
    #'gbm__n_estimators': [100,200,300],
    #'gbm__max_depth': [15,30,None],
    #'gbm__max_features': ['sqrt','log2'],
    'rfc__n_estimators': [100,200,300],
    'rfc__max_depth': [15,30,None],
    'rfc__max_features': ['sqrt','log2'],
    'rfc__class_weight': ['balanced', None]
}
```

Results for random forest classifier:

```
Best score: 0.885
Best parameters set:
        rfc__class_weight: None
        rfc__max_depth: None
        rfc__max_features: 'sqrt'
        rfc__n_estimators: 200
        vect__ngram_range: (1, 1)
Accuracy on training data: 0.998864
Accuracy on test data:     0.883736
[[   26  1557     2]
 [   11 20128    44]
 [   15  1046   179]]

Classification Report
               precision    recall  f1-score   support

    Ambiguous       0.50      0.02      0.03      1585
Human Factors       0.89      1.00      0.94     20183
    Procedure       0.80      0.14      0.24      1240

    micro avg       0.88      0.88      0.88     23008
    macro avg       0.73      0.39      0.40     23008
 weighted avg       0.85      0.88      0.84     23008
```

Results for gradient boosting classifier:

```
Best score: 0.890
Best parameters set:
        gbm__max_depth: 30
        gbm__max_features: 'sqrt'
        gbm__n_estimators: 200
        vect__ngram_range: (1, 2)
Accuracy on training data: 0.968873
Accuracy on test data:     0.887170
[[   81  1474    30]
 [   51 19960   172]
 [   39   830   371]]

Classification Report
               precision    recall  f1-score   support

    Ambiguous       0.47      0.05      0.09      1585
Human Factors       0.90      0.99      0.94     20183
    Procedure       0.65      0.30      0.41      1240

    micro avg       0.89      0.89      0.89     23008
    macro avg       0.67      0.45      0.48     23008
 weighted avg       0.85      0.89      0.85     23008
```