CS170 PROJECT

Kenji Kevin Fulcher

Ivan Gil Mercano

Ryan Harvi Pallarca

BSCS 2 - OL151

DATA GATHERING

In our data gathering project, we all chose to get a data set from the internet which contains not too simple and not too complex data set. Based on the instructions, the data may come from the website Kaggle or from a different source that also has a complete and fair data. As we remembered, there are many datasets in different websites but what we have decided to choose is an airline safety data set that came from a website called GitHub. When we downloaded the file, we saw a sufficient amount of data inside and it includes a lot of rows and columns which is eligible for us to show the data gathering process in our project. Within the next line, we have declared a variable for our csv file called “air_data” so in the next line we can disclose our data by calling the file name in the next line.

In [245...
```
#imports
import pandas as pd
import numpy as np
```

In [246...
```
air_data = pd.read_csv('airline-safety.csv')
```

In [247...
```
air_data
```

Out[247...

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating	Unnamed:3
0	Aer Lingus	320906734	2	0	3	NaN
1	Aeroflot*	1197672318	76	6	1	NaN
2	Aerolineas Argentinas	385803648	6	1	3	NaN
3	Aeromexico*	596871813	3	5	2	NaN
4	Air Canada	1865253802	2	2	1	NaN
5	Air France	3004002661	14	6	1	NaN
6	Air India*	869253552	2	4	1	NaN
7	Air New Zealand*	710174817	3	5	2	NaN
8	Alaska Airlines*	965346773	5	5	1	NaN

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating	Unnamed:3
9	Alitalia	698012498	7	4	2	NaN
10	All Nippon Airways	1841234177	3	7	1	NaN
11	American*	5228357340	21	17	1	NaN
12	Austrian Airlines	358239823	1	1	3	NaN
13	Avianca	396922563	5	0	3	NaN
14	British Airways*	3179760952	4	6	1	NaN
15	Cathay Pacific*	2582459303	0	2	1	NaN
16	China Airlines	813216487	12	2	2	NaN
17	Condor	417982610	2	0	3	NaN
18	COPA	550491507	3	0	2	NaN
19	Delta / 1rthwest*	6525658894	24	24	1	NaN
20	Egyptair	557699891	8	4	2	NaN
21	El Al	335448023	1	1	3	NaN
22	Ethiopian Airlines	488560643	25	5	3	NaN
23	Finnair	506464950	1	0	2	NaN
24	Garuda Indonesia	613356665	10	4	2	NaN
25	Gulf Air	301379762	1	3	3	NaN
26	Hawaiian Airlines	493877795	0	1	3	NaN
27	Iberia	1173203126	4	5	1	NaN
28	Japan Airlines	1574217531	3	0	1	NaN
29	Kenya Airways	277414794	2	2	3	NaN
30	KLM*	1874561773	7	1	1	NaN
31	Korean Air	1734522605	12	1	1	NaN
32	LAN Airlines	1001965891	3	0	1	NaN
33	Lufthansa*	3426529504	6	3	1	NaN
34	Malaysia Airlines	1039171244	3	3	1	NaN
35	Pakistan International	348563137	8	10	3	NaN
36	Philippine Airlines	413007158	7	2	3	NaN
37	Qantas*	1917428984	1	5	1	NaN

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating	Unnamed:3
38	Royal Air Maroc	295705339	5	3	3	NaN
39	SAS*	682971852	5	6	2	NaN
40	Saudi Arabian	859673901	7	11	1	NaN
41	Singapore Airlines	2376857805	2	2	3	NaN
42	South African	651502442	2	1	2	NaN
43	Southwest Airlines	3276525770	1	8	1	NaN
44	Sri Lankan / AirLanka	325582976	2	4	3	NaN
45	SWISS*	792601299	2	3	2	NaN
46	TACA	259373346	3	1	3	NaN
47	TAM	1509195646	8	7	1	NaN
48	TAP - Air Portugal	619130754	0	0	2	NaN
49	Thai Airways	1702802250	8	2	1	NaN
50	Turkish Airlines	1946098294	8	8	1	NaN
51	United / Continental*	7139291291	19	14	1	NaN
52	US Airways / America West*	2455687887	16	11	1	NaN
53	Vietnam Airlines	625084918	7	1	2	NaN
54	Virgin Atlantic	1005248585	1	0	1	NaN
55	Xiamen Airlines	430462962	9	2	3	NaN

Data Cleansing

For the next process in the project discussion, we have to do a data cleansing. This process is dropping an unused column in the table because it would not serve any purpose if it is still included in the next processes. Moreover, it would be more convenient for the programmers to sort and organize the data. In our project, the column “Unnamed:3” does not have a purpose in our data set and it would not affect our other data if we were to drop it. Therefore, we are going to drop it from our data set.

In [248…
```
air_data.columns
```

Out[248…
```
Index(['airline', 'avail_seat_km_per_week', 'incidents_85_99',
       'incidents_00_14', 'safety_rating', 'Unnamed:3'],
```

/

```
dtype='object')
```

In [249…
```
air_data.drop(labels=["Unnamed:3"], axis = 1, inplace = True)
```

In [250…
```
air_data.head()
```

Out[250…

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating
0	Aer Lingus	320906734	2	0	3
1	Aeroflot*	1197672318	76	6	1
2	Aerolineas Argentinas	385803648	6	1	3
3	Aeromexico*	596871813	3	5	2
4	Air Canada	1865253802	2	2	1

Exploratory Data Analysis

After some data from the set has been cleansed, the values of the data were sorted in ascending order in order to find the lowest to the highest data regarding a specific category. The top 5 lowest incident rating from 1985 to 1999 was in the Hawaiian airlines, air Portugal, and Cathay pacific with 0 incidents and the Virgin Atlantic and Finnair with 1 incident. Next the top 5 lowest incident rating from 2000 to 2014 was Aer Lingus, COPA, Condor, Air Portugal, and Virgin Atlantic all having 0 incidents happening within these years. Next is the top 5 lowest availed seats travelled per week measured in kilometers, the lowest being TACA, Kenya Airways, Royal Air Maroc, Gulf Air, and Aer Lingus sorted ascendingly. Lastly are the top 5 lowest airlines for the safety rating, these being Iberia, Qantas, Southwest Airlines, Malaysia Airlines, and Lufthansa, all having a safety rating of 1 which is the lowest while 3 is the highest. The data is then compared through the use of a scatterplot, the compared data are the incidents from 1985 to 1999 against the incidents from 2000 to 2014, availed seats per week (km) against the safety rating, the incidents of 1985 to 1999 against the safety ratings, the incidents of 2000 to 2014 against the safety ratings, the incidents of 2000 to 2014 against the availed seats per week (km), and lastly, the incidents of 1985 to 1999 against the availed seats per week (km).

In [251…
```
import matplotlib.pyplot as plt
import matplotlib as mlp
%matplotlib inline
```

In [252…
```
air_data.shape
```

Out[252…
```
(56, 5)
```

In [253…
```
air_data.describe
```

Out[253…
```
<bound method NDFrame.describe of                        airline  avail_seat_km_per_
week  incidents_85_99  \
0                   Aer Lingus               320906734                2
1                    Aeroflot*              1197672318               76
2        Aerolineas Argentinas               385803648                6
3                  Aeromexico*               596871813                3
4                   Air Canada              1865253802                2
5                   Air France              3004002661               14
6                   Air India*               869253552                2
7             Air New Zealand*               710174817                3
8             Alaska Airlines*               965346773                5
9                     Alitalia               698012498                7
10           All Nippon Airways             1841234177                3
```

11	American*	5228357340	21
12	Austrian Airlines	358239823	1
13	Avianca	396922563	5
14	British Airways*	3179760952	4
15	Cathay Pacific*	2582459303	0
16	China Airlines	813216487	12
17	Condor	417982610	2
18	COPA	550491507	3
19	Delta / 1rthwest*	6525658894	24
20	Egyptair	557699891	8
21	El Al	335448023	1
22	Ethiopian Airlines	488560643	25
23	Finnair	506464950	1
24	Garuda Indonesia	613356665	10
25	Gulf Air	301379762	1
26	Hawaiian Airlines	493877795	0
27	Iberia	1173203126	4
28	Japan Airlines	1574217531	3
29	Kenya Airways	277414794	2
30	KLM*	1874561773	7
31	Korean Air	1734522605	12
32	LAN Airlines	1001965891	3
33	Lufthansa*	3426529504	6
34	Malaysia Airlines	1039171244	3
35	Pakistan International	348563137	8
36	Philippine Airlines	413007158	7
37	Qantas*	1917428984	1
38	Royal Air Maroc	295705339	5
39	SAS*	682971852	5
40	Saudi Arabian	859673901	7
41	Singapore Airlines	2376857805	2
42	South African	651502442	2
43	Southwest Airlines	3276525770	1
44	Sri Lankan / AirLanka	325582976	2
45	SWISS*	792601299	2
46	TACA	259373346	3
47	TAM	1509195646	8
48	TAP - Air Portugal	619130754	0
49	Thai Airways	1702802250	8
50	Turkish Airlines	1946098294	8
51	United / Continental*	7139291291	19
52	US Airways / America West*	2455687887	16
53	Vietnam Airlines	625084918	7
54	Virgin Atlantic	1005248585	1
55	Xiamen Airlines	430462962	9

	incidents_00_14	safety_rating
0	0	3
1	6	1
2	1	3
3	5	2
4	2	1
5	6	1
6	4	1
7	5	2
8	5	1
9	4	2
10	7	1
11	17	1
12	1	3
13	0	3
14	6	1
15	2	1
16	2	2
17	0	3
18	0	2
19	24	1
20	4	2
21	1	3
22	5	3
23	0	2
24	4	2

```
25        3      3
26        1      3
27        5      1
28        0      1
29        2      3
30        1      1
31        1      1
32        0      1
33        3      1
34        3      1
35       10      3
36        2      3
37        5      1
38        3      3
39        6      2
40       11      1
41        2      3
42        1      2
43        8      1
44        4      3
45        3      2
46        1      3
47        7      1
48        0      2
49        2      1
50        8      1
51       14      1
52       11      1
53        1      2
54        0      1
55        2      3  >
```

In [254…

```
incidents_85_99 = air_data.sort_values('incidents_85_99', ascending = True)
incidents_85_99.head()
```

Out[254…

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating
26	Hawaiian Airlines	493877795	0	1	3
48	TAP - Air Portugal	619130754	0	0	2
15	Cathay Pacific*	2582459303	0	2	1
54	Virgin Atlantic	1005248585	1	0	1
23	Finnair	506464950	1	0	2

In [255…

```
incidents_00_14 = air_data.sort_values('incidents_00_14', ascending = True)
incidents_00_14.head()
```

Out[255…

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating
0	Aer Lingus	320906734	2	0	3
18	COPA	550491507	3	0	2
17	Condor	417982610	2	0	3
48	TAP - Air Portugal	619130754	0	0	2
54	Virgin Atlantic	1005248585	1	0	1

In [256…

```
avail_seat_km_per_week = air_data.sort_values('avail_seat_km_per_week', ascending =
avail_seat_km_per_week.head()
```

Out[256…

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating
46	TACA	259373346	3	1	3

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating
29	Kenya Airways	277414794	2	2	3
38	Royal Air Maroc	295705339	5	3	3
25	Gulf Air	301379762	1	3	3
0	Aer Lingus	320906734	2	0	3

In [257…

```
safety_rating = air_data.sort_values('safety_rating', ascending = True)
safety_rating.head()
```

Out[257…

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating
27	Iberia	1173203126	4	5	1
37	Qantas*	1917428984	1	5	1
43	Southwest Airlines	3276525770	1	8	1
34	Malaysia Airlines	1039171244	3	3	1
33	Lufthansa*	3426529504	6	3	1

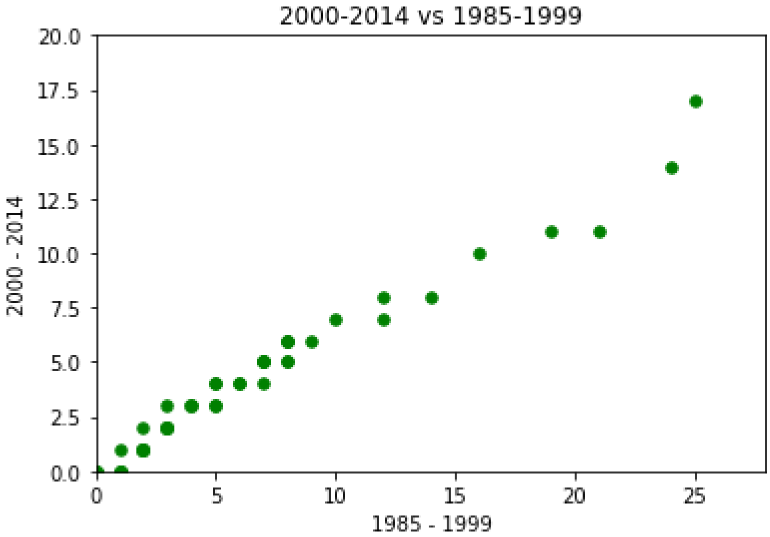
In [258…

```
fx= incidents_85_99['incidents_85_99']
fy = incidents_00_14['incidents_00_14']
```

In [259…

```
plt.scatter(fx, fy, color='g', s=30)
plt.title('2000-2014 vs 1985-1999')
plt.xlabel('1985 - 1999')
plt.ylabel('2000 - 2014')
plt.xlim([0, 28])
plt.ylim([0, 20])
```

Out[259… (0.0, 20.0)



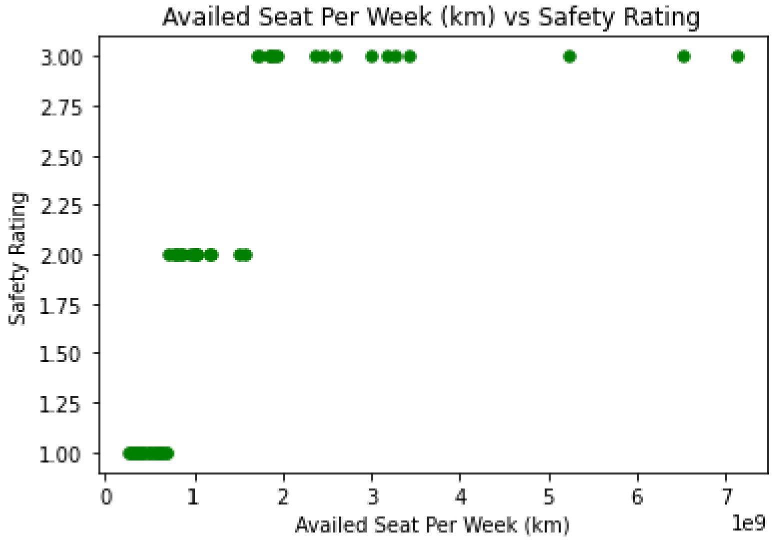
In [260…

```
fx1 = avail_seat_km_per_week['avail_seat_km_per_week']
fy1 = safety_rating['safety_rating']
```

In [261…

```
plt.scatter(fx1, fy1, color='g', s=30)
plt.title('Availed Seat Per Week (km) vs Safety Rating')
plt.xlabel('Availed Seat Per Week (km)')
plt.ylabel('Safety Rating')
```

Out[261… Text(0, 0.5, 'Safety Rating')



In [262…

```
plt.scatter(fx, fy1, color='g', s=30)
plt.title('Incidents in 1985 - 1999 vs Safety Rating')
plt.xlabel('Incidents in 1985 - 1999')
plt.ylabel('Safety Rating')
```

Out[262… Text(0, 0.5, 'Safety Rating')

In [263…

```
plt.scatter(fy, fy1, color='g', s=30)
plt.title('Incidents in 2000 - 2014 vs Safety Rating')
plt.xlabel('Incidents in 2000 - 2014')
plt.ylabel('Safety Rating')
```

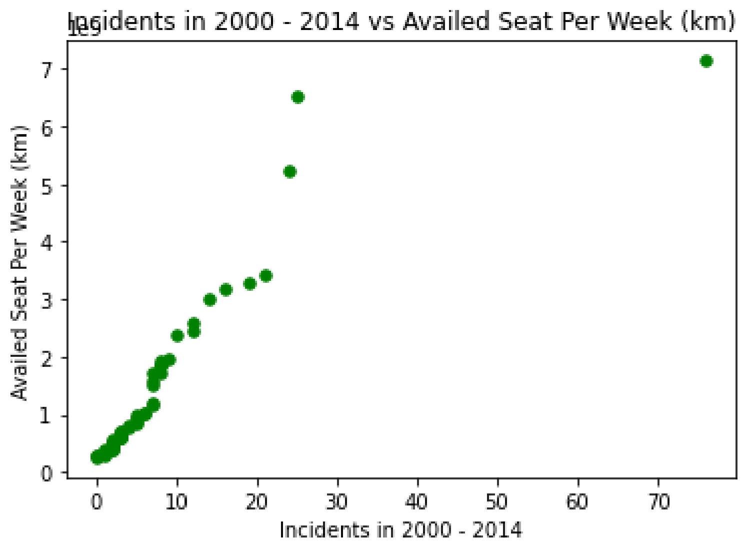
Out[263… Text(0, 0.5, 'Safety Rating')

In [264…

```
plt.scatter(fx, fx1, color='g', s=30)
plt.title('Incidents in 2000 - 2014 vs Availed Seat Per Week (km)')
plt.xlabel('Incidents in 2000 - 2014')
plt.ylabel('Availed Seat Per Week (km)')
```

Out[264…

```
Text(0, 0.5, 'Availed Seat Per Week (km)')
```

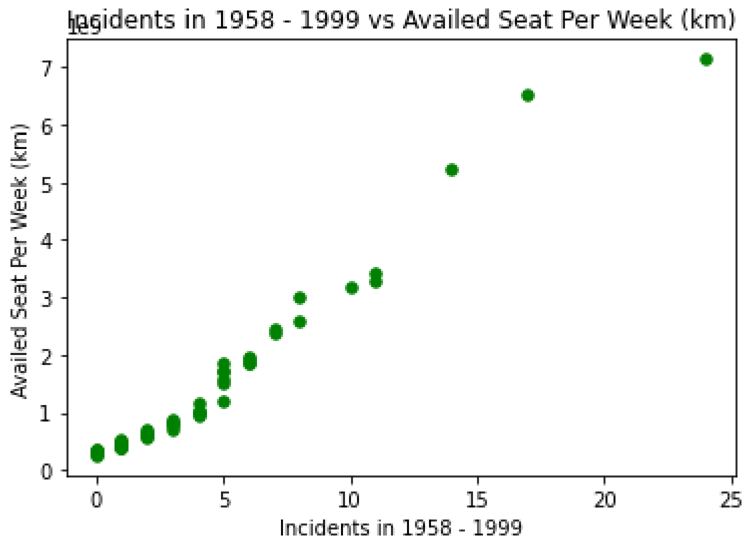


In [265…

```
plt.scatter(fy, fx1, color='g', s=30)
plt.title('Incidents in 1958 - 1999 vs Availed Seat Per Week (km)')
plt.xlabel('Incidents in 1958 - 1999')
plt.ylabel('Availed Seat Per Week (km)')
```

Out[265…

```
Text(0, 0.5, 'Availed Seat Per Week (km)')
```



Data Modelling

Next, we have the data modeling process. The data modeling is responsible for providing an accurate prediction of our data. Using three different models namely the Linear Regression model, Multi-Layer Perceptron model, and the Random Forest model, we will be able to utilize these methods in order to show the comparison in our air data. Each of our data will be called separately from other data and will be assessed individually by using the train and test split method. In our project, the incidents coming from different years will be compared and then it will be given a safety rating with accuracy based on how many incidents happened in those years. In our project, the first step would be importing the said methods and giving them a specific variable name for each. Then, we will declare a variable to our air data specifically the incidents from two different columns along with the safety rating. The next method is to create the train and test method and applying it to our air data and then specifying the random state to 1 since it is necessary for the model to predict the right output.

In [266…

```
from sklearn.linear_model import LinearRegression
from sklearn.neural_network import MLPRegressor
from sklearn.ensemble import RandomForestRegressor

from sklearn.model_selection import train_test_split
```

In [267…

```
air_modelLR = LinearRegression()
air_modelMLPR = MLPRegressor()
air_modelRFP = RandomForestRegressor()
```

In [268…

```
air_data.head()
```

Out[268…

	airline	avail_seat_km_per_week	incidents_85_99	incidents_00_14	safety_rating
0	Aer Lingus	320906734	2	0	3
1	Aeroflot*	1197672318	76	6	1
2	Aerolineas Argentinas	385803648	6	1	3
3	Aeromexico*	596871813	3	5	2
4	Air Canada	1865253802	2	2	1

In [269…

```
X = air_data[['incidents_85_99', 'incidents_00_14']]
y = air_data[['safety_rating']]
```

In [270…

```
X
```

Out[270…

	incidents_85_99	incidents_00_14
0	2	0
1	76	6
2	6	1
3	3	5
4	2	2
5	14	6
6	2	4
7	3	5
8	5	5
9	7	4
10	3	7
11	21	17
12	1	1
13	5	0
14	4	6
15	0	2
16	12	2
17	2	0
18	3	0
19	24	24
20	8	4
21	1	1
22	25	5
23	1	0
24	10	4
25	1	3
26	0	1
27	4	5
28	3	0
29	2	2
30	7	1
31	12	1
32	3	0
33	6	3
34	3	3
35	8	10

	incidents_85_99	incidents_00_14
36	7	2
37	1	5
38	5	3
39	5	6
40	7	11
41	2	2
42	2	1
43	1	8
44	2	4
45	2	3
46	3	1
47	8	7
48	0	0
49	8	2
50	8	8
51	19	14
52	16	11
53	7	1
54	1	0
55	9	2

In [271…

```
y
```

Out[271…

	safety_rating
0	3
1	1
2	3
3	2
4	1
5	1
6	1
7	2
8	1
9	2
10	1
11	1
12	3
13	3

	safety_rating
14	1
15	1
16	2
17	3
18	2
19	1
20	2
21	3
22	3
23	2
24	2
25	3
26	3
27	1
28	1
29	3
30	1
31	1
32	1
33	1
34	1
35	3
36	3
37	1
38	3
39	2
40	1
41	3
42	2
43	1
44	3
45	2
46	3
47	1
48	2
49	1
50	1

	safety_rating
51	1
52	1
53	2
54	1
55	3

In [272…

```
Xtrain, Xtest, Ytrain, Ytest = train_test_split(X,y, test_size = 0.2, random_state =
```

In [273…

```
Xtrain
```

Out[273…

	incidents_85_99	incidents_00_14
38	5	3
41	2	2
10	3	7
3	3	5
24	10	4
52	16	11
35	8	10
26	0	1
45	2	3
54	1	0
27	4	5
34	3	3
13	5	0
22	25	5
47	8	7
30	7	1
17	2	0
51	19	14
31	12	1
23	1	0
4	2	2
14	4	6
29	2	2
28	3	0
50	8	8
40	7	11
18	3	0
55	9	2

	incidents_85_99	incidents_00_14
20	8	4
25	1	3
6	2	4
7	3	5
53	7	1
1	76	6
16	12	2
0	2	0
15	0	2
5	14	6
11	21	17
9	7	4
8	5	5
12	1	1
43	1	8
37	1	5

In [274…

```
Ytrain
```

Out[274…

	safety_rating
38	3
41	3
10	1
3	2
24	2
52	1
35	3
26	3
45	2
54	1
27	1
34	1
13	3
22	3
47	1
30	1
17	3
51	1

	safety_rating
31	1
23	2
4	1
14	1
29	3
28	1
50	1
40	1
18	2
55	3
20	2
25	3
6	1
7	2
53	2
1	1
16	2
0	3
15	1
5	1
11	1
9	2
8	1
12	3
43	1
37	1

In [275…

```
Xtest
```

Out[275…

	incidents_85_99	incidents_00_14
44	2	4
2	6	1
46	3	1
19	24	24
32	3	0
33	6	3
36	7	2
39	5	6

	incidents_85_99	incidents_00_14
49	8	2
42	2	1
48	0	0
21	1	1

In [276...

```
Ytest
```

Out[276...

	safety_rating
44	3
2	3
46	3
19	1
32	1
33	1
36	3
39	2
49	1
42	2
48	2
21	3

In [277...

```
air_modelLR.fit(Xtrain,Ytrain)
air_modelMLPR.fit(Xtrain,Ytrain)
air_modelRFP.fit(Xtrain,Ytrain)
```

```
D:\Iggmercano\lib\site-packages\sklearn\utils\validation.py:72: DataConversionWarnin
g: A column-vector y was passed when a 1d array was expected. Please change the shap
e of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
D:\Iggmercano\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:58
2: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and th
e optimization hasn't converged yet.
  warnings.warn(
<ipython-input-277-0a1fa37a634f>:3: DataConversionWarning: A column-vector y was pas
sed when a 1d array was expected. Please change the shape of y to (n_samples,), for
example using ravel().
  air_modelRFP.fit(Xtrain,Ytrain)
```

Out[277...

```
RandomForestRegressor()
```

Evaluation

In the data evaluation phase, the Linear Regression model, Multi-Layer Perceptron model, and the Random Forest model are then predicted using Xtest and are then assigned to separate variables. By using Ytest, the mean squared error and the mean absolute error of the three models can be presented, the three model scores are then graphed into a bar chart and is plotted to see various data. The first graph presents the R-squared score comparison for each of the data, the Linear Regression model is the lowest at a score of .16, next is the Multi-Layer

Perception model at a score of .24, and lastly is the Random Forest model, which is the highest and has the score of .74. The second graph compares the three models regarding the mean squared error comparison, the data for the models are all nearly similar with one another, the Linear Regression model at .77 percent when rounded up, the Multi-Layer Perception model is at .80, lastly is the Random Forest model which is at a score of .97. The third and last graph compares the three models in terms of the mean absolute error comparison, the scores of the three models for this graph are all close to one another, the Linear Regression model is at .80 when rounded up, the Multi-Layer Perception model is at .83 when rounded up as well, and lastly is the Random Forest model which is at the score of .87 when rounded up.

In [278...

```
print(air_modelLR.score(Xtrain,Ytrain))
print(air_modelMLPR.score(Xtrain,Ytrain))
print(air_modelRFP.score(Xtrain,Ytrain))
```

```
0.16269587299190624
0.24186542044892567
0.7408440949956632
```

In [279...

```
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
```

In [280...

```
air_modelLRpred = air_modelLR.predict(Xtest)
air_modelLMLPRpred = air_modelMLPR.predict(Xtest)
air_modelLRRFpred = air_modelRFP.predict(Xtest)
```

In [281...

```
print(mean_squared_error(air_modelLRpred,Ytest))
print(mean_squared_error(air_modelLMLPRpred,Ytest))
print(mean_squared_error(air_modelLRRFpred,Ytest))
```

```
0.769454040203669
0.8064697169247879
0.9728163026856577
```

In [282...

```
print(mean_absolute_error(air_modelLRpred,Ytest))
print(mean_absolute_error(air_modelLMLPRpred,Ytest))
print(mean_absolute_error(air_modelLRRFpred,Ytest))
```

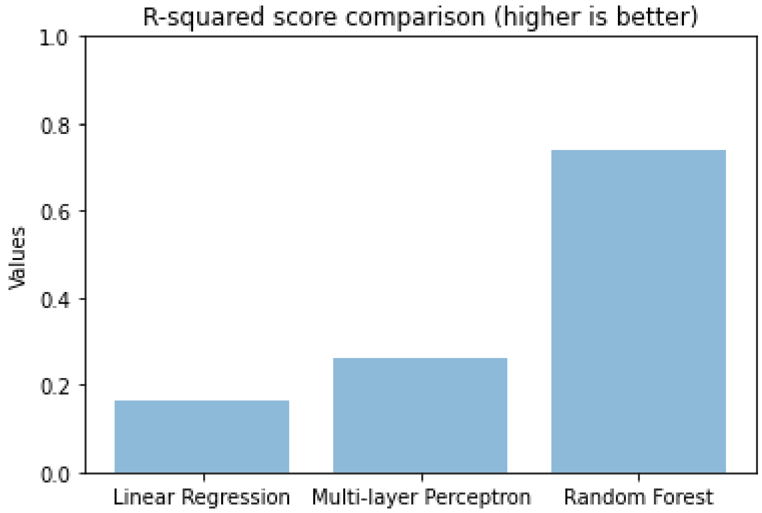
```
0.7984905840783393
0.8288822946774648
0.8688065476190477
```

In [283...

```
Model_score = ('Linear Regression', 'Multi-layer Perceptron', 'Random Forest')
Ypos_score = np.arange(len(Model_score))
Values = [0.16269587299190624, 0.26131569317759185, 0.7383876354290524]
```

In [284...

```
plt.bar(Ypos_score, Values, align='center', alpha=0.5)
plt.xticks(Ypos_score, Model_score)
plt.ylabel('Values')
plt.title('R-squared score comparison (higher is better)')
plt.ylim([0, 1])
plt.show()
```

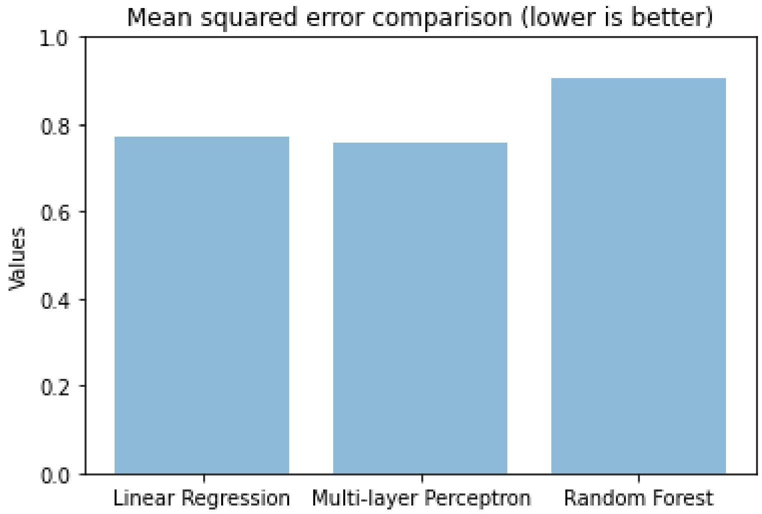


In [285...

```
Model_sq_error = ('Linear Regression', 'Multi-layer Perceptron', 'Random Forest')
Ypos_sq_error = np.arange(len(Model_sq_error))
Values = [0.769454040203669, 0.7594476008076797, 0.9066726928854876]
```

In [286...

```
plt.bar(Ypos_sq_error, Values, align='center', alpha=0.5)
plt.xticks(Ypos_sq_error, Model_sq_error)
plt.ylabel('Values')
plt.title('Mean squared error comparison (lower is better)')
plt.ylim([0, 1])
plt.show()
```

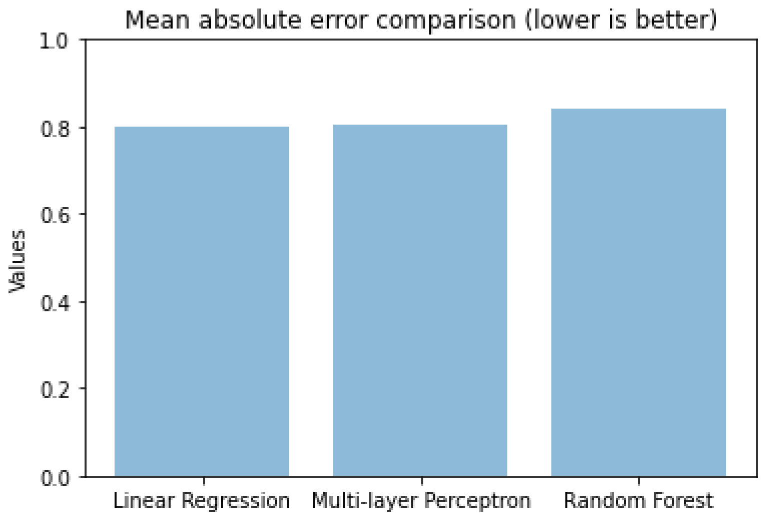


In [287...

```
Model_ab_error = ('Linear Regression', 'Multi-layer Perceptron', 'Random Forest')
Ypos_ab_error = np.arange(len(Model_ab_error))
Values = [0.7984905840783393, 0.8049702599580387, 0.8411805555555555]
```

In [288...

```
plt.bar(Ypos_ab_error, Values, align='center', alpha=0.5)
plt.xticks(Ypos_ab_error, Model_ab_error)
plt.ylabel('Values')
plt.title('Mean absolute error comparison (lower is better)')
plt.ylim([0, 1])
plt.show()
```



Recommendations

The Data Scientist incourage future scientist to use this data science project to use for thier own projects. The Data Scientist recommends to investigate not just the incidents of the airlines, but also the fatalities that have been recorded using the same method that this project have. This may or may not tell yield the same result but this will emphasize and reduce the frequency of error that this project will give. The Data scientist also recommend to use other data modelling that could describe the data even better and could lessen the margin of error. Future scientist can also search for much more broad data so that they can further explore the environment of the data data scinetist used in this porject.

References

https://towardsdatascience.com/how-to-evaluate-your-machine-learning-models-with-python-code-5f8d2d8d945b

https://machinelearningmastery.com/metrics-evaluate-machine-learning-algorithms-python/

https://pythonspot.com/matplotlib-scatterplot/

https://www.youtube.com/watch?v=XDv6T4a0RNc&t=536s

https://www.youtube.com/watch?v=ZjQCPMO7LBE&t=572s

https://www.youtube.com/watch?v=snkkKrek7TU&t=50s

https://www.youtube.com/watch?v=mKSWAlvXSmw

https://youtu.be/dxueNcTYjqI

In [ ]: