Data Mining – Assignment1

Jingyi Wu(jingyiw2)

A.

dataset1: Flight Fare Prediction

link:https://www.kaggle.com/datasets/nikhilmittal/flight-fare-prediction-mh

This dataset contains information about flights and corresponding flight ticket price. We can use these features to make predictions about the flight ticket's price. This test dataset consists of 10683 records with 13 columns about flights in India by some Indian and foreign Airlines in 2019. The test dataset contains 2671 records. This is a regression problem.

The independent variables are as follow.

'Airline': The name of the airline.<nominal variable>

'Date_of_Journey': The date of the journey. <time series variable>

'Source': The source from which the service begins.<nominal variable>

'Destination': The destination where the service ends.<nominal variable>

'Route': The route taken by the flight to reach the destination.<nominal variable>

'Dep_Time': The time when the journey starts from the source. <datetimeformat>

'Arrival_Time': Time of arrival at the destination.<datetimeformat>

'Duration': Total duration of the flight.<string, nominal variable>

'Total_Stops': Total stops between the source and destination.<string, ordinal>

'Additional_Info': Additional information about the flight

Dependent variable:

'Price': The price of the ticket. <numeric, int>

Challenges in mining this dataset:

1) Many features in the dataset are about date and time, for example, 'date_of_journey', 'dep time', 'arrival time', 'duration'. And they need feature engineering before being put into models. For the first three, we need to split them into several columns of int type to be ready for machine learning models.
2) There are many duplicate records in the given dataset, which may affect the accuracy of machine learning models.
3) The value distribution of 'additional_info' column is extremely unbalanced, so combining some minority value may improve the model.
4) The types of features are mixed, so we need to transform variables in some specific models.

This is a regression model about prediction, so most regression algorithms can be applied, like polynomial regression, decision tree/random forest regressor. The predicted result would be predicted flight fare for a specific flight that have known features.

Dataset2: Rain in Australia
Link:https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package

This dataset is used to predict next-day rain based on several features. This dataset consists of 142193 records and 23 variables. It contains about 10 years of daily weather observations from numerous Australian weather stations. This is a classification problem.

Variables X:
Date: The date of observation. Dateformat.
Location: The common name of the location of the weather station
MinTemp: The minimum temperature in degrees Celsius.
MaxTemp: The maximum temperature in degrees Celsius.
Rainfall: The amount of rainfall recorded for the day in mm
Evaporation: The so-called Class A pan evaporation (mm) in the 24 hours to 9am
Sunshine: The number of hours of bright sunshine in the day.
WindGustDir: The direction of the strongest wind gust in the 24 hours to midnight
WindGustSpeed: The speed (km/h) of the strongest wind gust in the 24 hours to midnight
WindDir9am: Direction of the wind at 9am
WindDir3pm: Direction of the wind at 3pm
WindSpeed9am: Wind speed (km/hr) averaged over 10 minutes prior to 9am
WindSpeed3pm: Wind speed (km/hr) averaged over 10 minutes prior to 3pm
Humidity9am: Humidity (percent) at 9am
Humidity3pm: Humidity (percent) at 3pm
Pressure9am: Atmospheric pressure (hpa) reduced to mean sea level at 9am
Pressure3pm: Atmospheric pressure (hpa) reduced to mean sea level at 3p
Cloud9am: Fraction of sky obscured by cloud at 9am. This is measured in "oktas", which are a unit of eigths. It records how many eigths of the sky are obscured by cloud. A 0 measure indicates completely clear sky whilst an 8 indicates that it is completely overcast.
Cloud3pm: Fraction of sky obscured by cloud (in "oktas": eighths) at 3pm. See Cload9am for a description of the values
Temp9am: Temperature (degrees C) at 9am
Temp3pm: Temperature (degrees C) at 3pm

RainToday: Boolean: 1 if precipitation (mm) in the 24 hours to 9am exceeds 1mm, otherwise 0

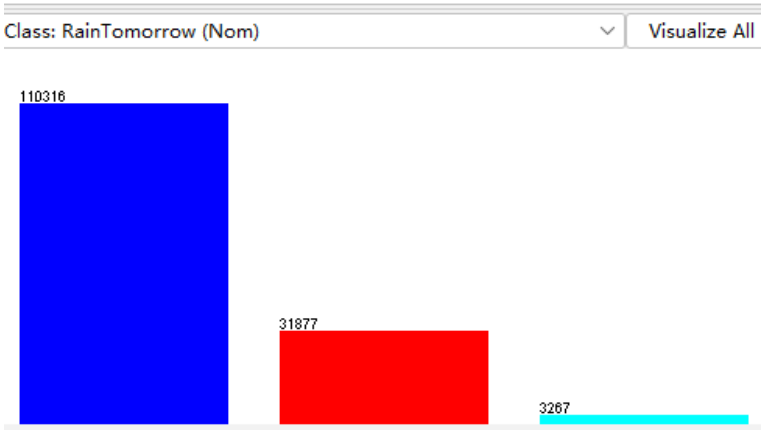
Variable Y:

RainTomorrow: will it rain the next day, yes or no? It is yes if the rain for that day was 1mm or more. A kind of measure of the "risk".

Challenges in mining this dataset:

1) Dataset contains mixture of categorical and numerical variables. When applying certain models, we need to transform categorical features.
2) There are outliers for numeric variables(25% quantile-3*IQR~75% quantile+3*IQR) and we need to drop them.
3) Many variables care strongly correlated to others (with correlation coefficient>0.95). Maybe we can select some of them to fit the model to improve performance.
4) Some data are missing in the dataset. We can use mean/median imputation or random sample imputation to fill in the missing values.
5) For some classification models, we also need feature scaling.
6) The response feature for this dataset is quite unbalanced. To get more accurate prediction, we may need to do over-sampling or subsampling to create balanced data on our own.

No.	Label	Count	Weight
1	No	110316	110316
2	Yes	31877	31877
3	NA	3267	3267



This is a classification problem, so we can use logistic regression and decision tree. The potential mining result is to get a predicted Boolean result about whether it will rain tomorrow.

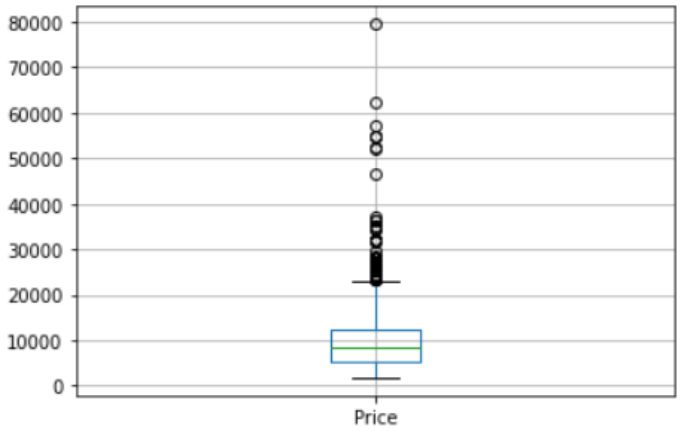
B.

Since the dataset in Kaggle are quite large, it is more efficient to use Kaggle notebook in Python. I will first use Kaggle notebook and then weka for some questions.

Dataset1:

Except for response variable price, other variables are all categorical. For price, basic statistics are as follow.

	Price
count	10683.000000
mean	9087.064121
std	4611.359167
min	1759.000000
25%	5277.000000
50%	8372.000000
75%	12373.000000
max	79512.000000



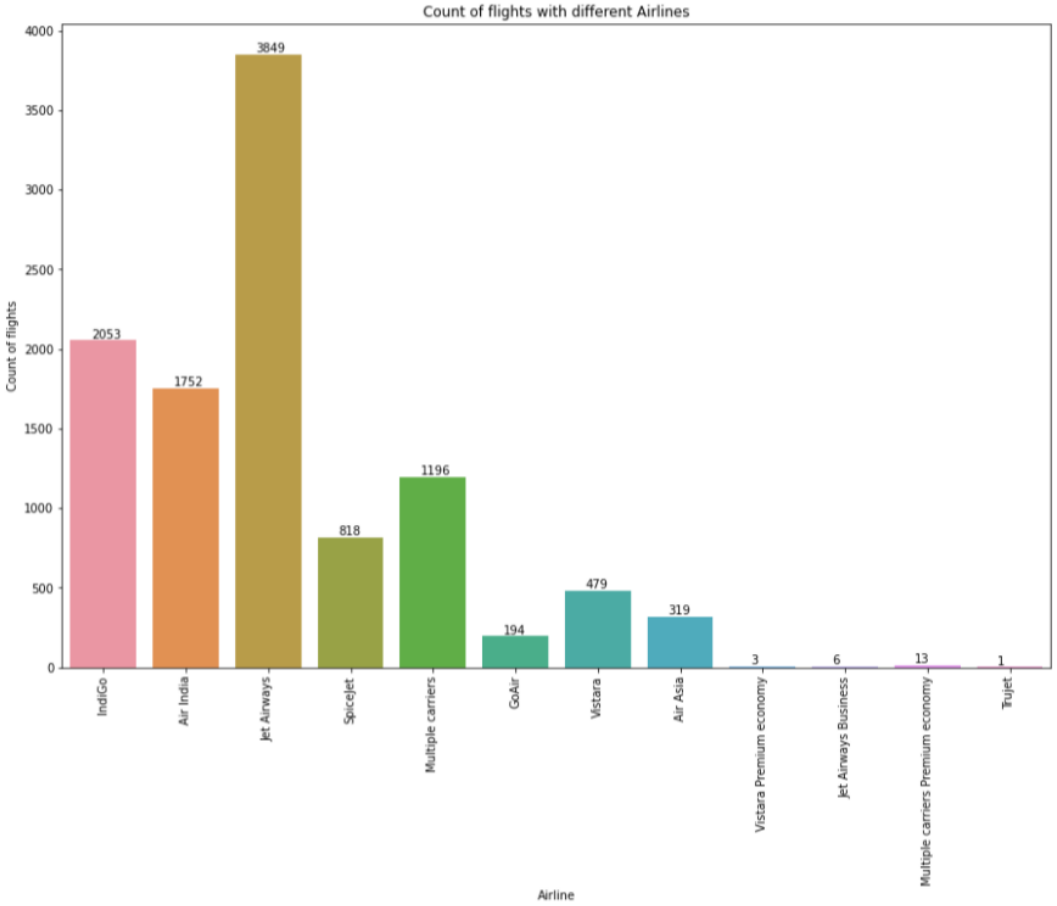
Price:

mean: 9087.064120565385 median: 8372.0 mode: 10262 Range: 77753 Variance: 4611.35916681709

For categorical variable Airline: Jet Airways take up the most.

```
Jet Airways                          3849
IndiGo                               2053
Air India                            1752
Multiple carriers                    1196
SpiceJet                              818
Vistara                               479
Air Asia                              319
GoAir                                 194
Multiple carriers Premium economy      13
Jet Airways Business                    6
Vistara Premium economy                 3
Trujet                                  1
Name: Airline, dtype: int64
```

Its distribution is as follow.



As for data quality issues, in this dataset, we can count duplicates in a data frame – 220. We can drop these using train_data.drop_duplicates(keep='first',inplace=True).This is also a part of pre-processing.

```
print(train_data.shape)
train_data.drop_duplicates(keep='first',inplace=True)
print(train_data.shape)
```

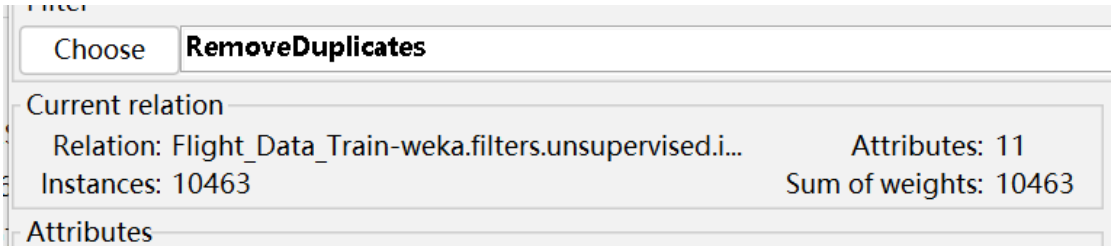
```
(10683, 11)
(10463, 11)
```

As for another data pre-processing skills required, splitting features of Date_of_Journey, Dep_Time, Arrival_Time into several columns to split month and day, hour and minute. Therefore, the datatype of these columns will be int. The additional info column has a great majority of two values.

```
No info                         8185
In-flight meal not included     1926
No check-in baggage included     318
1 Long layover                    19
Change airports                    7
Business class                     4
2 Long layover                     1
Red-eye flight                     1
1 Short layover                    1
Name: Additional_Info, dtype: int64
```

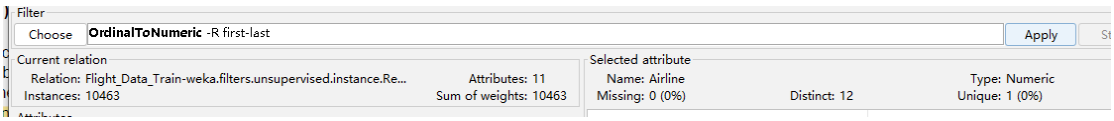
We can combine other values for this feature for better model fitting.

(1)Applying removing duplicates in weka,

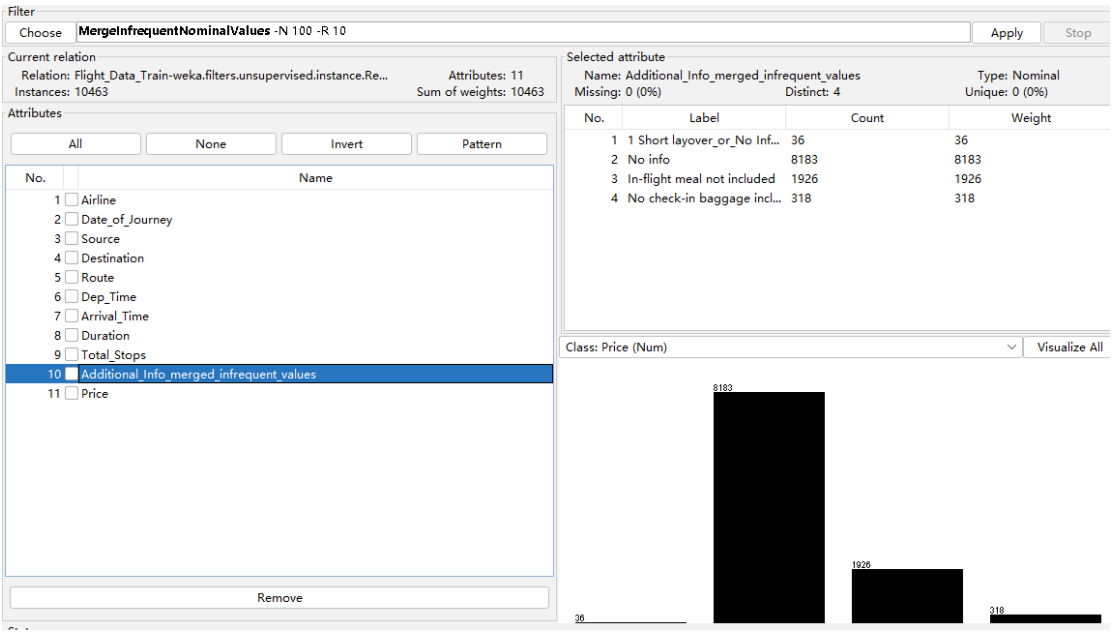


We get the same result as using Python.

(2)Applying ordinal to numeric for categorical variables to better prepare for machine learning regressor.



(3)Merging infrequent nominal values to improve regressor performance.



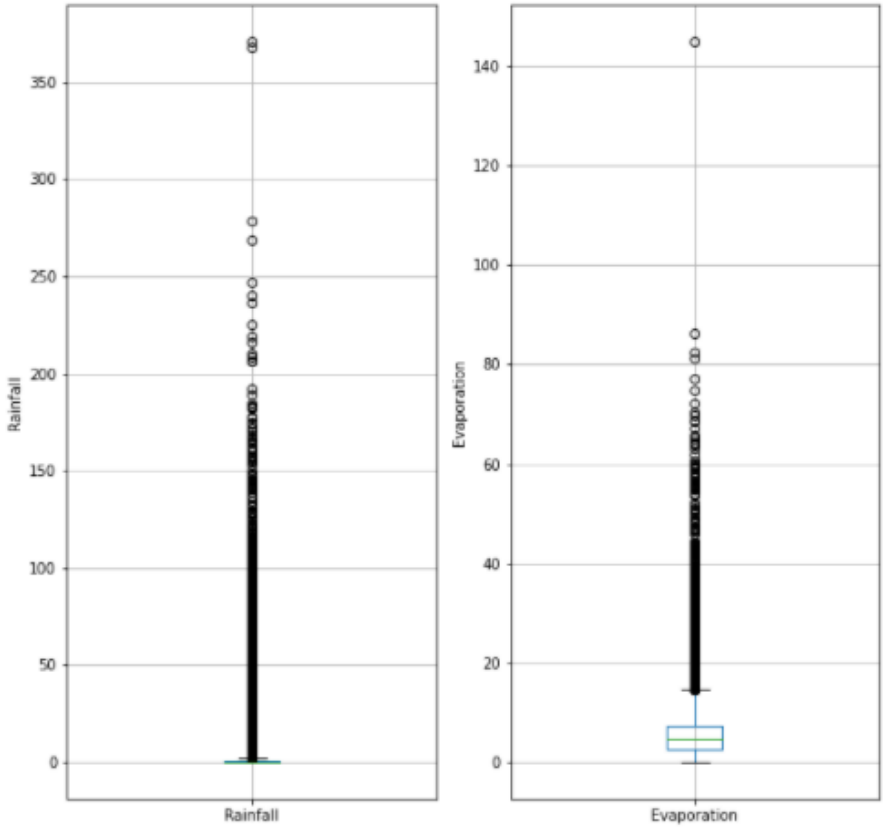
Dataset2:

Use df.describe() we can have a quick look at numeric values' distribution.

```
df.describe()
```

	MinTemp	MaxTemp	Rainfall	Evaporation	
count	143975.000000	144199.000000	42199.000000	82670.000000	7
mean	12.194034	23.221348	2.360918	5.468232	
std	6.398495	7.119049	8.478060	4.193704	
min	-8.500000	-4.800000	0.000000	0.000000	
25%	7.600000	17.900000	0.000000	2.600000	
50%	12.000000	22.600000	0.000000	4.800000	
75%	16.900000	28.200000	0.800000	7.400000	
max	33.900000	48.100000	371.000000	145.000000	

Here we select rainfall and evaporation attributes to see its statistics. The box plot are as follows.



Rainfall:

mean: 2.360918149917032 median: 0.0 mode: 0 Range: 371.0

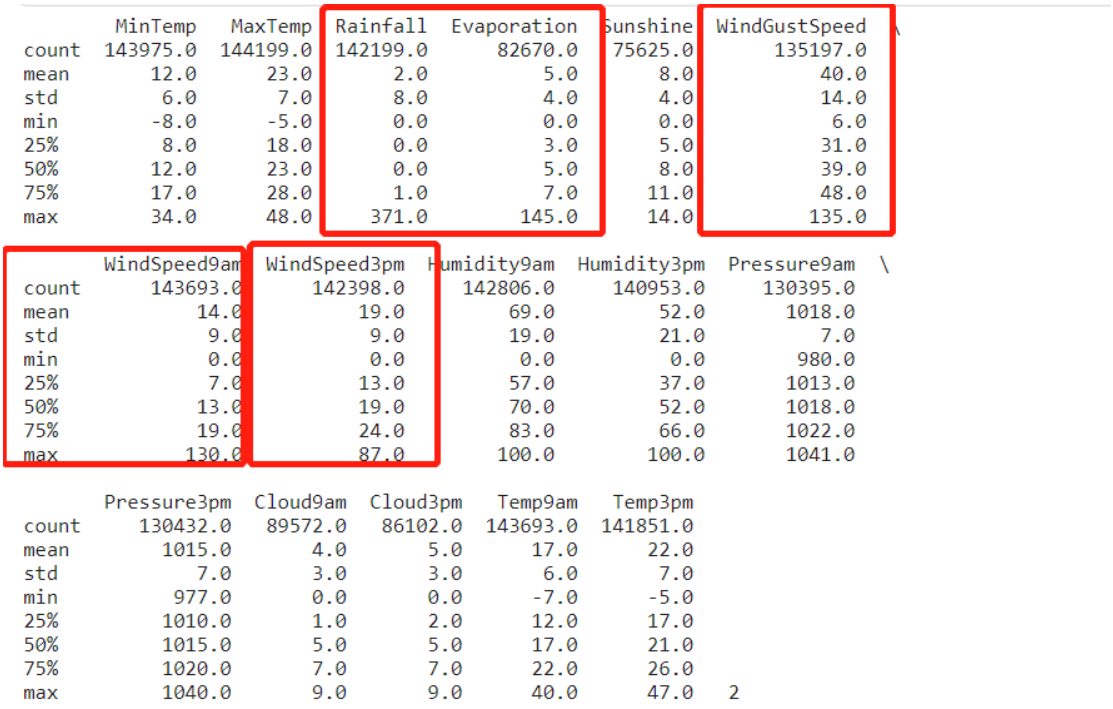
Variance: 8.478059737721674

Evaporation:

mean: 5.468231522922464 median: 4.8 mode: 4.0 Range: 145.0

Variance: 4.193704094151472

We can see for both attributes, there exist many outliers and need to be dropped. This is a data quality issue. These outliers can affect the performance of our model. We should solve this in pre-processing stage.



	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	\
count	143975.0	144199.0	142199.0	82670.0	75625.0	135197.0	
mean	12.0	23.0	2.0	5.0	8.0	40.0	
std	6.0	7.0	8.0	4.0	4.0	14.0	
min	-8.0	-5.0	0.0	0.0	0.0	6.0	
25%	8.0	18.0	0.0	3.0	5.0	31.0	
50%	12.0	23.0	0.0	5.0	8.0	39.0	
75%	17.0	28.0	1.0	7.0	11.0	48.0	
max	34.0	48.0	371.0	145.0	14.0	135.0	

	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am	\
count	143693.0	142398.0	142806.0	140953.0	130395.0	
mean	14.0	19.0	69.0	52.0	1018.0	
std	9.0	9.0	19.0	21.0	7.0	
min	0.0	0.0	0.0	0.0	980.0	
25%	7.0	13.0	57.0	37.0	1013.0	
50%	13.0	19.0	70.0	52.0	1018.0	
75%	19.0	24.0	83.0	66.0	1022.0	
max	130.0	87.0	100.0	100.0	1041.0	

	Pressure3pm	Cloud9am	Cloud3pm	Temp9am	Temp3pm	
count	130432.0	89572.0	86102.0	143693.0	141851.0	
mean	1015.0	4.0	5.0	17.0	22.0	
std	7.0	3.0	3.0	6.0	7.0	
min	977.0	0.0	0.0	-7.0	-5.0	
25%	1010.0	1.0	2.0	12.0	17.0	
50%	1015.0	5.0	5.0	17.0	21.0	
75%	1020.0	7.0	7.0	22.0	26.0	
max	1040.0	9.0	9.0	40.0	47.0	2

These columns may contain outliers. We can calculate lower bound and upper bound for these potential columns:

Rainfall -2.4000000000000004 3.2

Evaporation -11.800000000000002 21.800000000000004

WindGustSpeed -20.0 99.0

WindSpeed9am -29.0 55.0

WindSpeed3pm -20.0 57.0

In the pre-processing stage, we should replace outliers with bound numbers.

Missing data problem also exists in this dataset.

```
df[numeric_columns].isnull().sum()
```

```
MinTemp          1485
MaxTemp          1261
Rainfall         3261
Evaporation     62790
Sunshine        69835
WindGustSpeed   10263
WindSpeed9am     1767
WindSpeed3pm     3062
Humidity9am      2654
Humidity3pm      4507
Pressure9am     15065
Pressure3pm     15028
Cloud9am        55888
Cloud3pm        59358
Temp9am          1767
Temp3pm          3609
dtype: int64
```

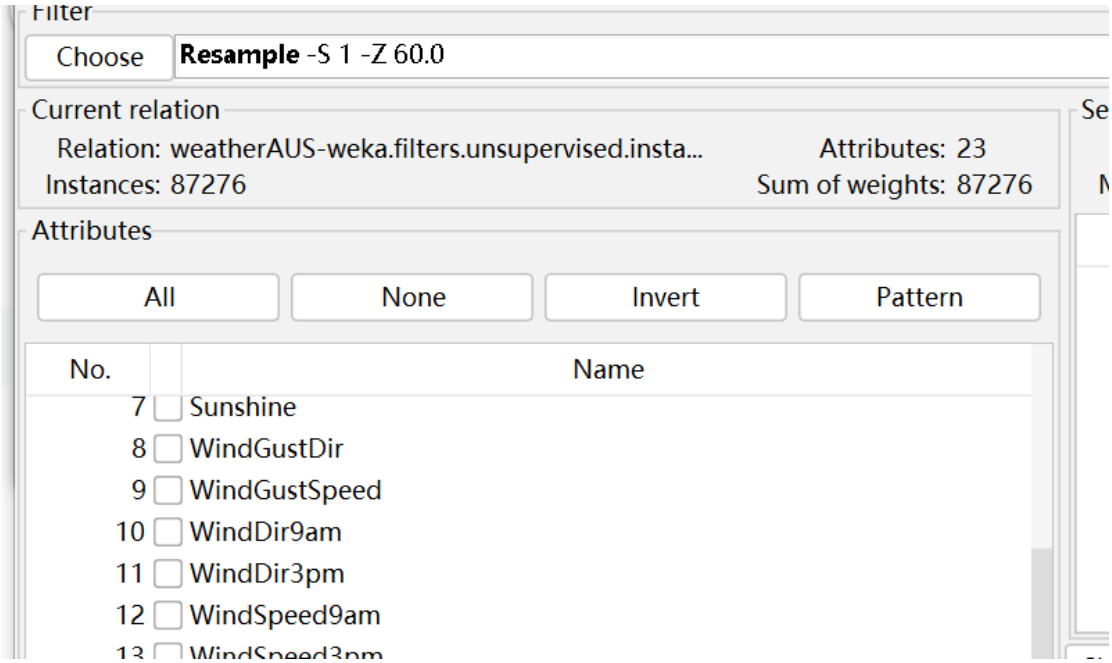
Missing values percentage here:

```
df[numeric_columns].isnull().sum()/len(df)
```

```
MinTemp          0.010209
MaxTemp          0.008669
Rainfall         0.022419
Evaporation      0.431665
Sunshine         0.480098
WindGustSpeed    0.070555
WindSpeed9am     0.012148
WindSpeed3pm     0.021050
Humidity9am      0.018246
Humidity3pm      0.030984
Pressure9am      0.103568
Pressure3pm      0.103314
Cloud9am         0.384216
Cloud3pm         0.408071
Temp9am          0.012148
Temp3pm          0.024811
dtype: float64
```

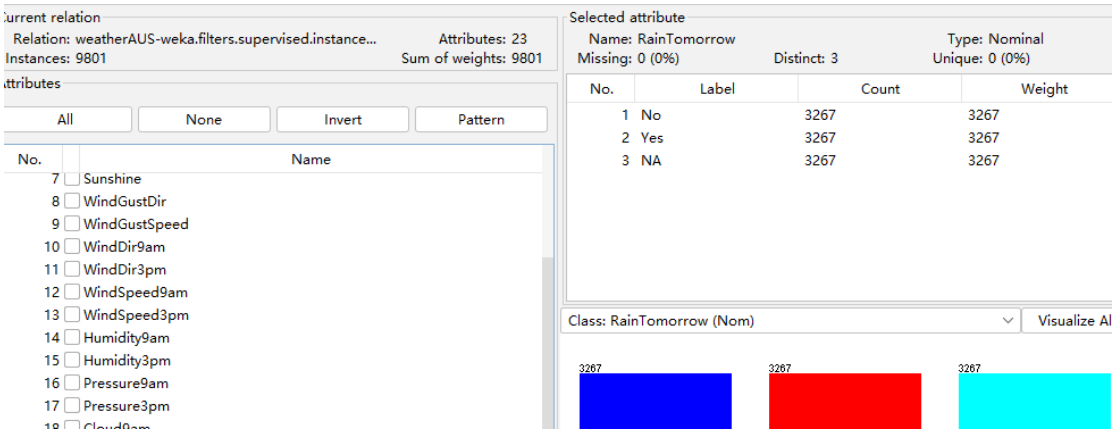
We can see that for Evaporation, Sunshine, Cloud9am, and Cloud3pm, there are around 40% are missing. In pre-processing stage, we can apply fillna with median of existing values in these columns.

(1) Applying resample to the dataset: since the original dataset is quite large, doing EDA can take a lot of time and memory may not enough. By resampling, we can use part of data to do analysis first.



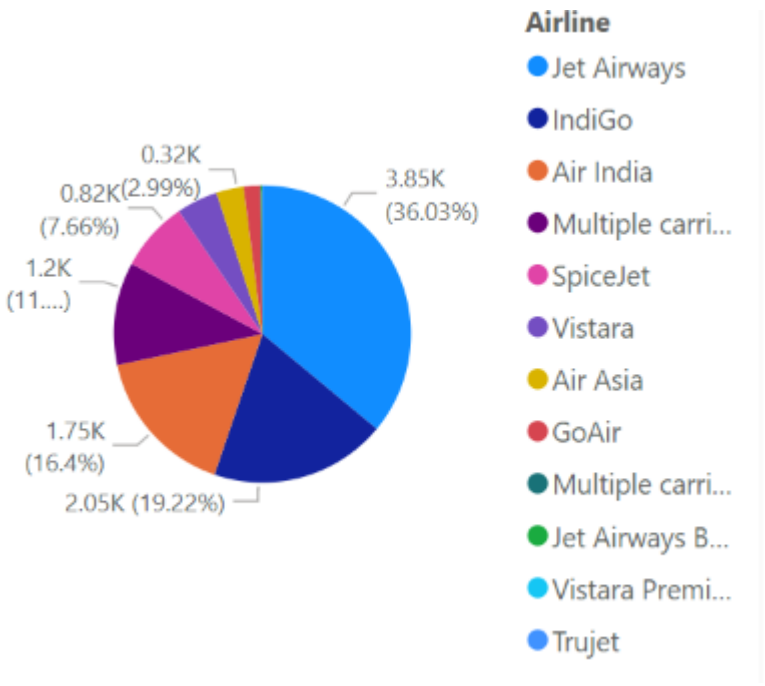
(2)When we apply classification models, we may need to map features into the same scale, in case that some variables have greater impact just because of large scale. Normalizing is an option. Use unsupervised.attribute.Normalize.

(3)In this classification model, we use SpreadSubsample to handle unbalanced dataset.

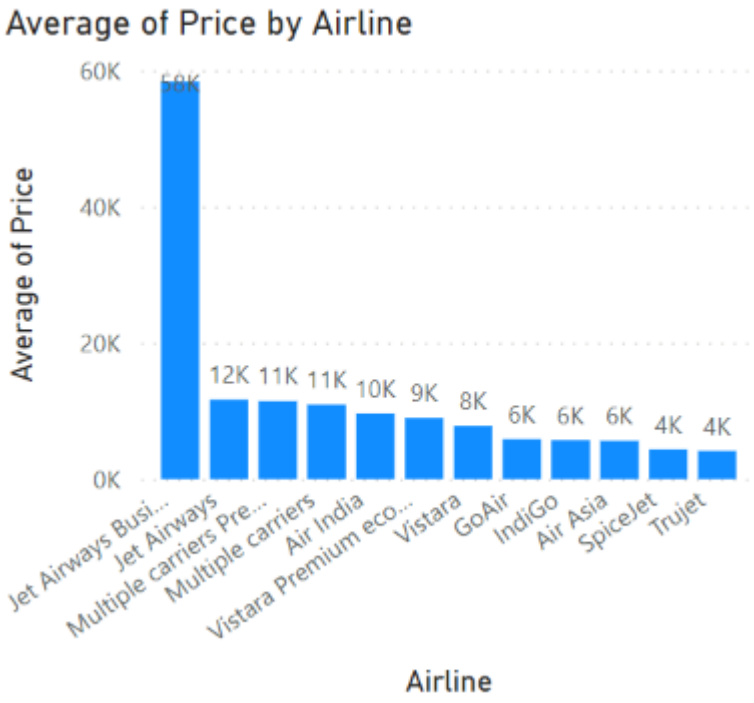


C.

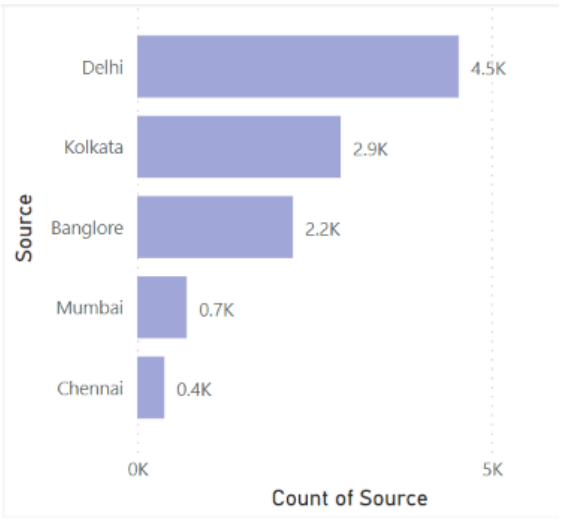
For flight_fare dataset, graph and corresponding interesting insights:

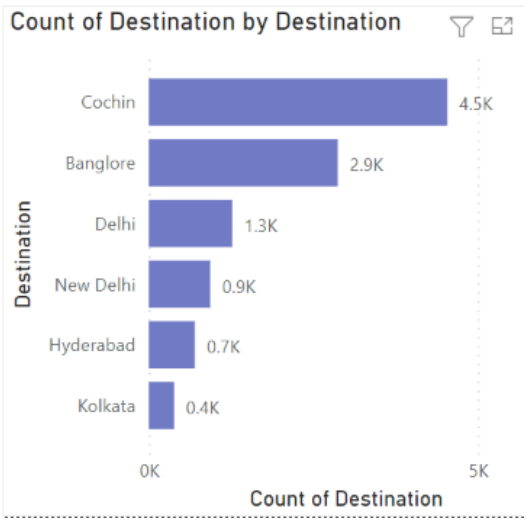


Jet Airways and IndiGo take up the most (over 50%) in the Indian flight market.

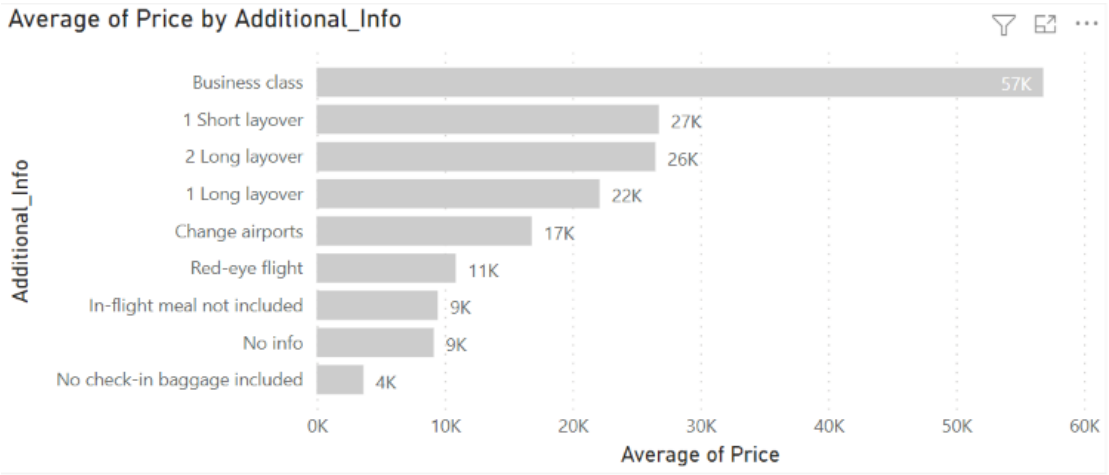


As for the average price of each airline, Jet Airways Business is the most expensive airline and Jet Airways goes next, which may imply Jet Airways' monopoly power in Indian flight market.

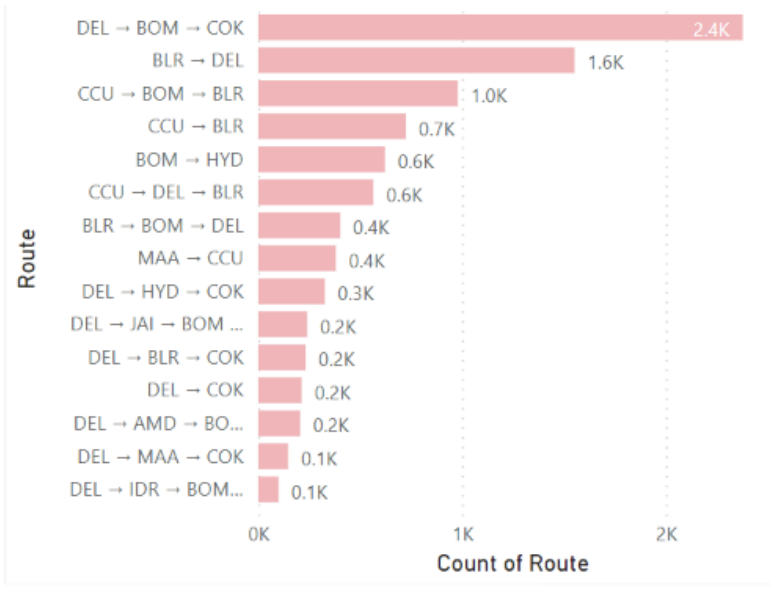




We can see from the above charts that most popular source spot and destination spot are Delhi and Cochin respectively, which imply high airport traffic in these two airports.



This chart shows that with additional info as Business Class, the ticket price is always higher, which makes sense.



This chart tells us which route is the most popular one, DEL-BOM-COK. The report dashboard of these chart is as follow.