Prediksi Harga Tiket Penerbangan

Introduction

Harga sebuah tiket pesawat sangat dipengaruhi oleh banyak faktor. Selain waktu keberangkatan, harga juga dapat dipengaruhi oleh kondisi ekonomi negara asal maupun negara tujuan (Bagaimana Harga Tiket Pesawat Ditetapkan Maskapai? Ini 8 Rahasianya - KabarPenumpang - Jalur Informasi Penumpang Tiga Moda).

Objektif dari proyek ini adalah untuk mengidentifikasi faktor-faktor yang memengaruhi harga tiket pesawat serta mengeksplorasi penerapan ilmu data (*data science*) dalam membangun model prediksi harga tiket penerbangan. Kami menggunakan algoritma *Random Forest* dan *XGBoost* sebagai model prediksi, dan mengevaluasi performanya menggunakan metrik *Root Mean Squared Error* (RMSE) dan koefisien determinasi (R^2).

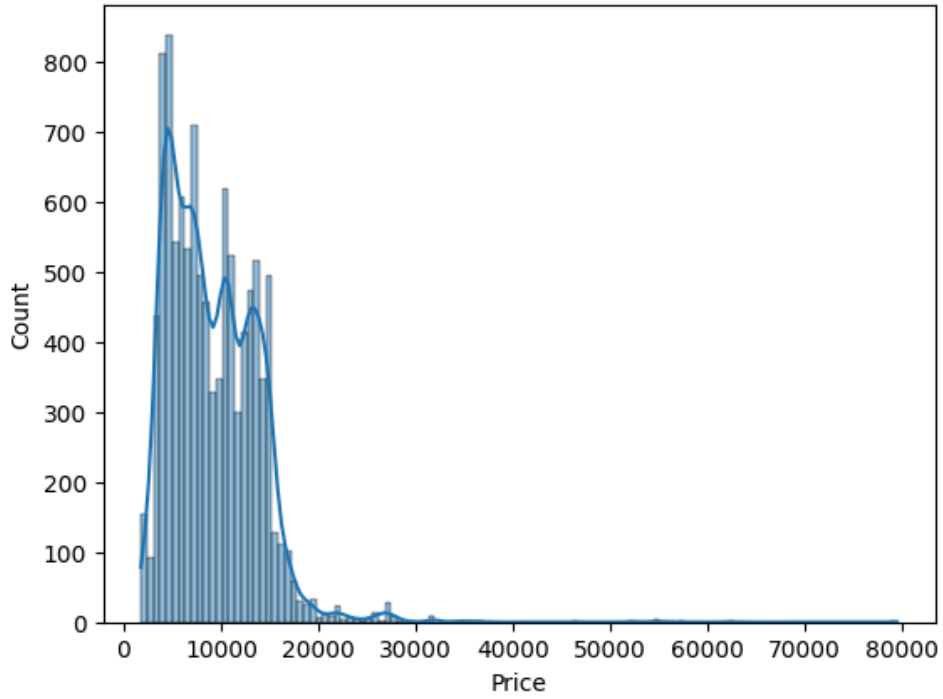
Dataset

Dataset yang kami gunakan berasal dari Kaggle (Flight price dataset). Dataset ini memiliki kekurangan lokasi geografis sebab hanya difokuskan ke negara India.

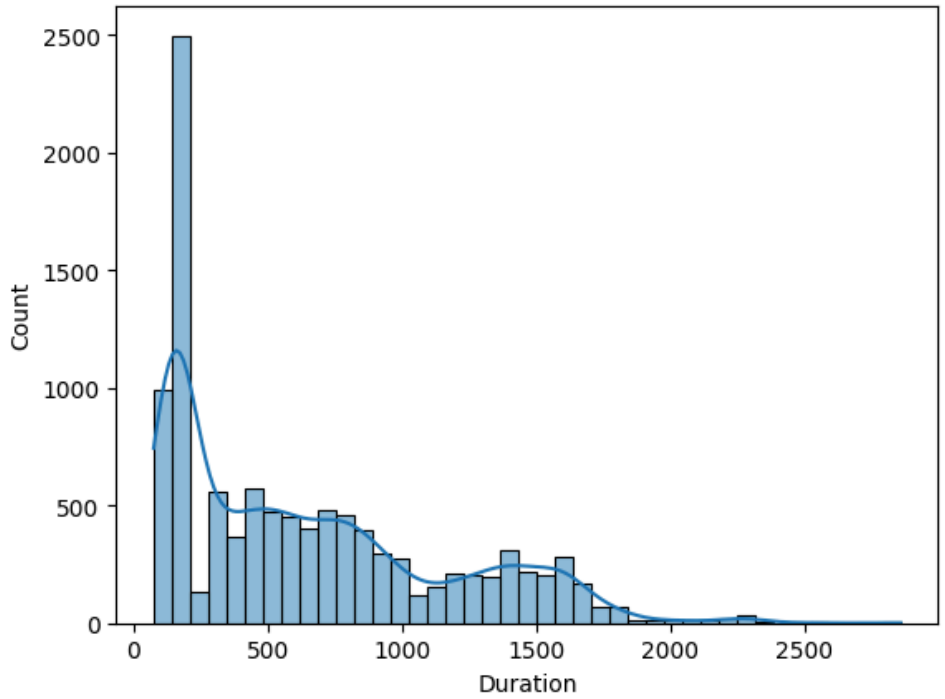
Dataset ini memiliki 11 atribut, di model kami nanti hanya menggunakan 7 atribut. Hal ini dikarenakan kami merasa bahwa atribut lain kurang cocok untuk dimasukkan ke dalam model.

	Airline	Date_of_Journey	Source	Destination	Duration	Total_Stops	Price
0	IndiGo	24/03/2019	Banglore	New Delhi	2h 50m	non-stop	3897
1	Air India	1/05/2019	Kolkata	Banglore	7h 25m	2 stops	7662
2	Jet Airways	9/06/2019	Delhi	Cochin	19h	2 stops	13882
3	IndiGo	12/05/2019	Kolkata	Banglore	5h 25m	1 stop	6218
4	IndiGo	01/03/2019	Banglore	New Delhi	4h 45m	1 stop	13302

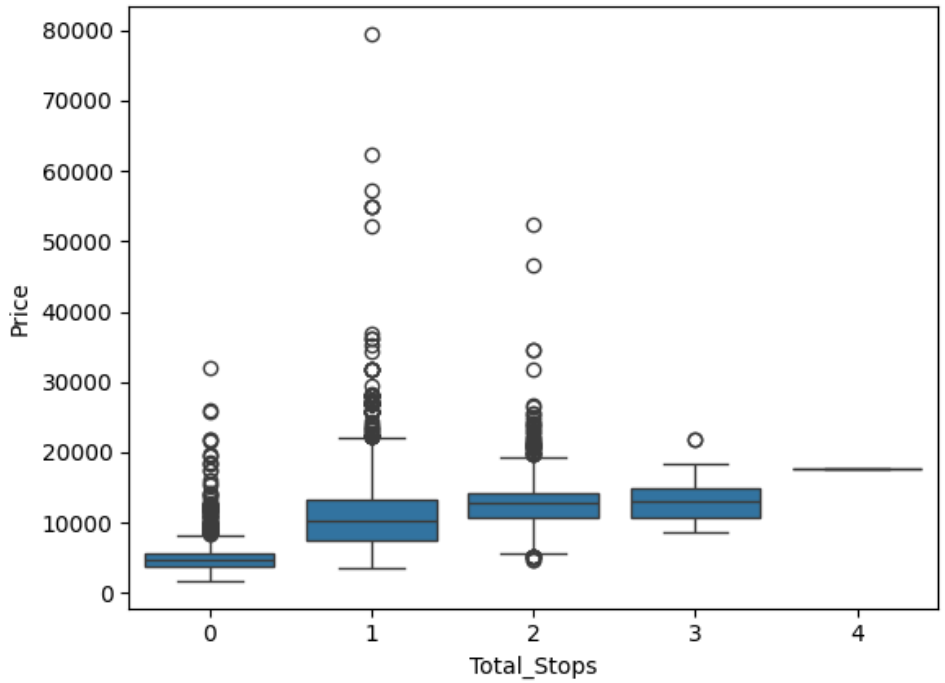
EDA dan Preprocessing



Berikut merupakan hasil histogram perbandingan jumlah kolom dengan harga tiket penerbangan. Jumlah harga yang terdapat di dataset sangat variatif karena banyak varian asal, tujuan, maskapai, dan juga tanggal penerbangan yang berbeda.



Berikut merupakan hasil histogram perbandingan antara jumlah kolom dengan durasi penerbangan.



Berikut merupakan hasil *boxplot* yang menunjukan perbandingan antara jumlah perhentian dan harga tiket penerbangan. Dari sini kami melihat bahwa banyak data yang tergolong sebagai *outlier*. *Outlier* tersebut disebabkan oleh banyaknya faktor yang dapat mempengaruhi harga tiket penerbangan.

Dalam tahap *preprocessing*, kami melakukan beberapa langkah seperti *encoding* dan perubahan tipe data. *Encoding* yang digunakan adalah *one hot encoding* dan juga *label encoding*. Kedua tipe *encoding* digunakan dalam konteks berbeda, dimana *label encoding* digunakan untuk merubah jumlah perhentian penerbangan menjadi angka numerik sesuai dengan jumlah perhentian yang tertera. Lalu, *one hot encoding* digunakan untuk menjadikan kolom sebagai label klasifikasi. *One hot encoding* diterapkan dalam kolom *Airline, Source, Destination,* dan juga *Journey*.

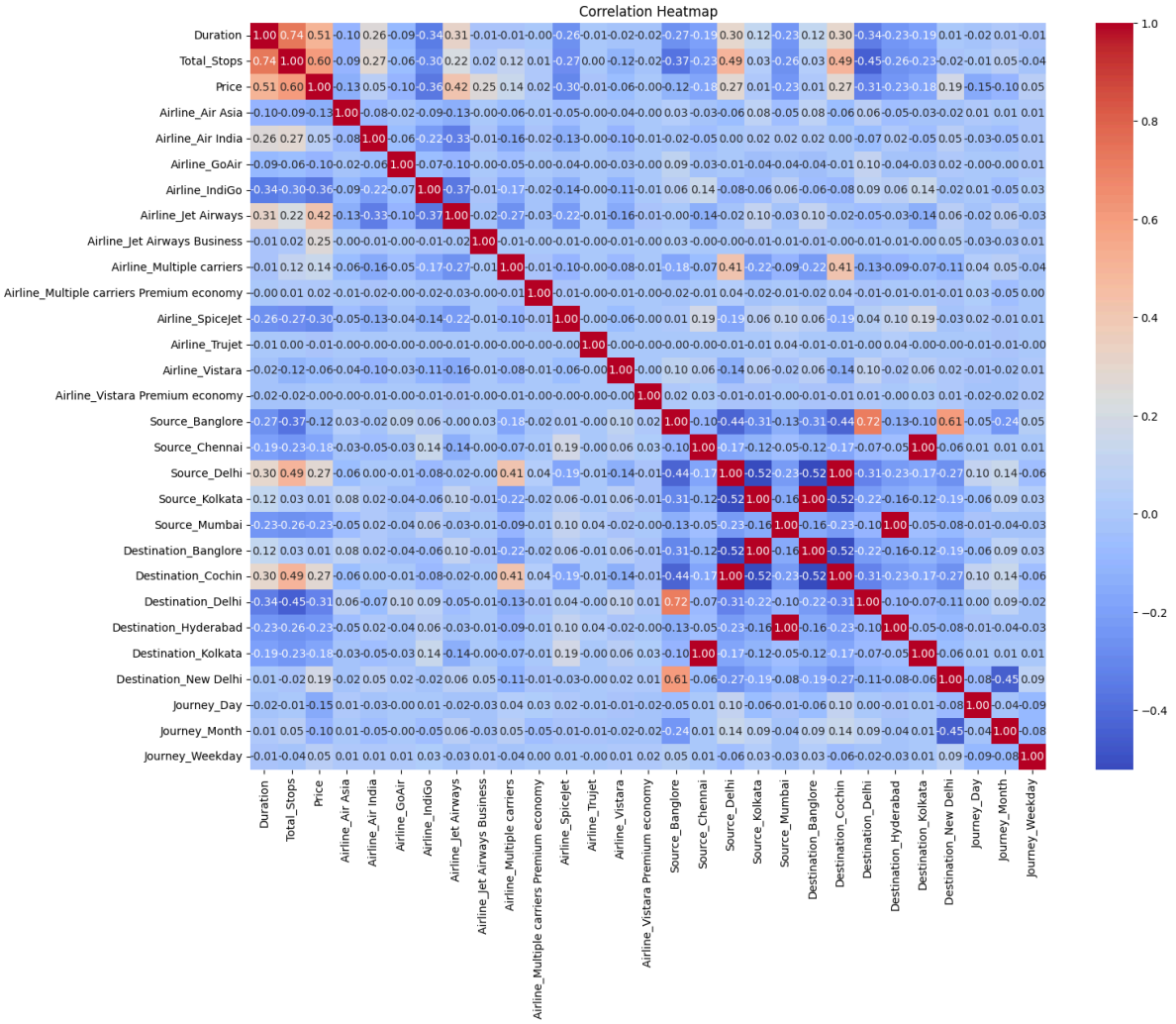
```python
# Label Encoding
stop_mapping = {
    '4 stops': 4,
    '3 stops': 3,
    '2 stops': 2,
    '1 stop': 1,
    'non-stop': 0
}
dataset['Total_Stops'] = dataset['Total_Stops'].map(stop_mapping)
dataset['Total_Stops'] = dataset['Total_Stops'].fillna(0).astype(int)

# One Hot Encoding
# Duration
def convert_to_minutes(time_str):
    if pd.isna(time_str):  # Handle potential NaN values
        return None
    parts = time_str.replace(' ', '').replace('h', ' ').replace('m', '').split(' ')
    hours = int(parts[0]) if parts[0] else 0
    minutes = int(parts[1]) if len(parts) > 1 and parts[1] else 0
    return hours * 60 + minutes

# Apply the function to the 'Duration_Time' column
dataset['Duration'] = dataset['Duration'].apply(convert_to_minutes)

# AirLine
categorical_columns = dataset[['Airline', 'Source', 'Destination']].columns.tolist()
encoder = OneHotEncoder(sparse_output=False)
one_hot_encoded = encoder.fit_transform(dataset[categorical_columns])
one_hot_df = pd.DataFrame(one_hot_encoded, columns=encoder.get_feature_names_out(categorical_columns))
dataset = pd.concat([dataset, one_hot_df], axis=1)
dataset = dataset.drop(categorical_columns, axis=1)
```

```python
# Changing Data Type
dataset['Date_of_Journey'] = pd.to_datetime(dataset['Date_of_Journey'], dayfirst=True)
dataset['Date_of_Journey'] = dataset['Date_of_Journey'].apply(lambda x: x.strftime('%Y-%m-%d'))
dataset['Date_of_Journey'] = pd.to_datetime(dataset['Date_of_Journey'])

# Separate Date Format
dataset['Date_of_Journey'] = pd.to_datetime(dataset['Date_of_Journey'], dayfirst=True)

dataset['Journey_Day'] = dataset['Date_of_Journey'].dt.day
dataset['Journey_Month'] = dataset['Date_of_Journey'].dt.month
dataset['Journey_Weekday'] = dataset['Date_of_Journey'].dt.dayofweek

dataset = dataset.drop('Date_of_Journey', axis=1)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 29 columns):
 #   Column                                     Non-Null Count  Dtype
---  ------                                     --------------  -----
 0   Duration                                   10683 non-null  int64
 1   Total_Stops                                10683 non-null  int64
 2   Price                                      10683 non-null  int64
 3   Airline_Air Asia                           10683 non-null  float64
 4   Airline_Air India                          10683 non-null  float64
 5   Airline_GoAir                              10683 non-null  float64
 6   Airline_IndiGo                             10683 non-null  float64
 7   Airline_Jet Airways                        10683 non-null  float64
 8   Airline_Jet Airways Business               10683 non-null  float64
 9   Airline_Multiple carriers                  10683 non-null  float64
 10  Airline_Multiple carriers Premium economy  10683 non-null  float64
 11  Airline_SpiceJet                           10683 non-null  float64
 12  Airline_Trujet                             10683 non-null  float64
 13  Airline_Vistara                            10683 non-null  float64
 14  Airline_Vistara Premium economy            10683 non-null  float64
 15  Source_Banglore                            10683 non-null  float64
 16  Source_Chennai                             10683 non-null  float64
 17  Source_Delhi                               10683 non-null  float64
 18  Source_Kolkata                             10683 non-null  float64
 19  Source_Mumbai                              10683 non-null  float64
 20  Destination_Banglore                       10683 non-null  float64
 21  Destination_Cochin                         10683 non-null  float64
 22  Destination_Delhi                          10683 non-null  float64
 23  Destination_Hyderabad                      10683 non-null  float64
 24  Destination_Kolkata                        10683 non-null  float64
 25  Destination_New Delhi                      10683 non-null  float64
 26  Journey_Day                                10683 non-null  int32
 27  Journey_Month                              10683 non-null  int32
 28  Journey_Weekday                            10683 non-null  int32
dtypes: float64(23), int32(3), int64(3)
memory usage: 2.2 MB
```

Encoding ini dilakukan agar model yang dibuat lebih mudah mengerti ataupun mengklasifikan kolom-kolom yang terdapat dalam dataset ini. Berikut merupakan hasil *encoding* yang dilakukan ketika sudah diterapkan kepada isi dari kolom dataset yang digunakan.

Destination_Kolkata	Destination_New Delhi	Journey_Day	Journey_Month	Journey_Weekday
0.0	1.0	24	3	6
0.0	0.0	1	5	2
0.0	0.0	9	6	6
0.0	0.0	12	5	6
0.0	1.0	1	3	4
...	...	...	...	...
0.0	0.0	9	4	1
0.0	0.0	27	4	5
0.0	0.0	27	4	5
0.0	1.0	1	3	4
0.0	0.0	9	5	3



Berdasarkan matriks korelasi, faktor yang paling berkorelasi positif dengan harga tiket adalah jumlah transit (0.60) dan durasi perjalanan (0.51). Artinya, semakin banyak transit dan semakin lama durasi, semakin mahal harga tiket. Penggunaan maskapai Jet Airways juga menunjukkan korelasi positif (0.42). Sebaliknya, penggunaan maskapai IndiGo memiliki korelasi negatif yang cukup kuat dengan harga tiket (-0.36), yang menunjukkan bahwa tiket IndiGo cenderung lebih murah.

Modelling

Algoritma yang kami gunakan berasal dari konsep *ensemble learning*, dimana beberapa performa model yang sama maupun berbeda akan diagregasikan menjadi satu. Teknik ini biasa digunakan untuk meningkatkan performa dengan cara *bias-variance trade-off* (Why Use Ensemble Learning? - MachineLearningMastery.com). Selain itu, juga perlu diketahui bahwa teknik ini memiliki kekurangan dimana akan memakan tingkat komputasi yang lebih dibanding hanya menggunakan 1 algoritma sendiri (When You

Shouldn't Use Ensemble Learning | Deepchecks). Sebab projek ini menggunakan *Google Collab* dan dataset kami masih tergolong kecil, menurut kami kekurangan ini dapat diabaikan.

Algoritma *Random Forest Regressor* digunakan dengan alasan mampu untuk melakukan prediksi terhadap nilai yang berkelanjutan dengan melihat rata-rata dari hasil beberapa *decision trees*. Dibandingkan hanya menggunakan decision trees, menggunakan konsep ensemble learning dapat meningkatkan akurasi yang lebih baik dan menghindari *overfitting* dengan menggunakan *bagging* dan *random feature selection*. (Random Forest Regression)

Algoritma XGBoost digunakan dengan alasan mampu untuk menangani berbagai jenis data yang kompleks. Model ini juga menggunakan *ensemble method* dari *decision trees*, tetapi dengan memperbaiki kesalahan pada *decision tree* sebelumnya hingga tingkat kesalahan sudah tidak dapat diperbaiki. Dengan menggunakan XGBoost juga dapat melakukan *pruning* dan *early stopping* untuk menghindari *overfitting* dan membuat struktur menjadi lebih sederhana. (XGBoost)

Dataset train kami, dipecah menjadi training set dan testing set sebesar 80:20.

Evaluasi

Untuk mengevaluasi model regression, 2 metrik akan digunakan yaitu *Root Mean Squared Error* (RMSE) dan koefisien determinasi (R^2).

Berbeda dengan model klasifikasi yang memprediksi kategori, model regresi bertujuan memprediksi nilai kontinu. Oleh karena itu, evaluasinya fokus pada seberapa dekat prediksi model terhadap nilai sebenarnya.

RMSE mengukur rata-rata selisih kuadrat antara nilai prediksi dan nilai aktual, yang kemudian diakarkan. Semakin kecil nilai RMSE, semakin baik performa model. Metrik ini dipilih karena satuannya sama dengan target yang diprediksi, sehingga lebih mudah diinterpretasikan.

Koefisien determinasi (R^2) mengukur seberapa besar proporsi variansi dari variabel target yang bisa dijelaskan oleh fitur-fitur input. Nilai R^2 mendekati 1 menunjukkan bahwa model dapat menjelaskan variansi dengan baik, sedangkan nilai yang rendah menandakan sebaliknya.

Source: [Evaluating Metrics: Classification vs. Regression - Knapsackpy]

Metrik	**Random Forest**	**XGBoost**
RMSE	2198.8019033676424	1993.060272545715
R^2	0.7715584216335669	0.8123088479042053

Dari hasil di atas, dapat dilihat bahwa model XGBoost memberikan performa yang lebih baik dibandingkan Random Forest, baik dari segi akurasi prediksi (RMSE lebih rendah) maupun kemampuan menjelaskan variansi data (R^2 lebih tinggi).

Conclusion

Melalui projek ini, kami semakin paham bagaimana cara menganalisa data melalui *exploratory data analysis* menggunakan beberapa *library* di *python* dan sebagai tambahan kami juga lebih paham mengenai model dan metrik evaluasi *regression.*

Kedua *modelling* yang digunakan menunjukkan hasil yang tidak jauh berbeda, tetapi *modelling* menggunakan XGBoost memberikan hasil yang lebih baik berdasarkan dari hasil RMSE dan R^2 yang didapatkan. Hal ini tidak terlepas dari pentingnya *trial and error* terhadap parameter yang ada untuk mendapatkan angka evaluasi metrik yang baik.