Comparison of Machine Learning Models with Deep Learning for Flight Price Prediction

Mentor :
Hanny Nurrohmah

Disusun Oleh :
Kelompok 2 (Dua)

1. Marshanda Gavrilla
2. Maulana Khisyam
3. Ridha Talasya Widyanti
4. Rifat Muhammad

DATA SCIENCE & ANALYST

PT MITRA TALENTA GROUP (CELERATES)

2024

Kata Pengantar

Segala puji dan syukur ke hadirat Tuhan Yang Maha Kuasa yang telah memberikan berkat, anugerah dan karunia yang melimpah, sehingga penulis dapat menyelesaikan Tugas Akhir ini. Tugas Akhir ini disusun guna melengkapi syarat penilaian akhir dalam program Kampus merdeka yang bekerja sama dengan Mitra Talenta Group (Celerates). Adapun judul Tugas Akhir ini adalah "Comparison of Machine Learning Models with Deep Learning for Flight Price Prediction".

Laporan ini merupakan hasil dari partisipasi kami dalam program Kampus Merdeka yang memberikan kesempatan kepada kami untuk mengembangkan pengetahuan dan keterampilan dalam bidang ilmu data, khususnya dalam penerapan pembelajaran *Machine Learning* dan *Deep Learning* untuk memprediksi harga tiket pesawat.

Kami menyadari bahwa penyusunan laporan ini tidak lepas dari bantuan berbagai pihak. Oleh karena itu, kami mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Pihak Mitra Talenta Group (Celerates), yang telah memberikan kesempatan dan dukungan selama program ini berlangsung.

2. Hanny Nurrohmah, S.Mat., selaku pembimbing yang telah memberikan bimbingan dan arahan kepada kami dalam penyusunan Tugas Akhir ini.

Kami menyadari bahwa laporan ini masih jauh dari sempurna. Oleh Karena itu, kami mengharapkan saran dan kritik yang membangun dari berbagai pihak guna penyempurnaan laporan ini di masa yang akan datang.

Akhir kata, kami berharap semoga laporan ini dapat bermanfaat bagi kita semua.

Jakarta, 20 Juni 2024

Penulis

Kelompok 2 (Dua)

1. Marshanda Gavrilla
2. Maulana Khisyam
3. Ridha Talasya Widyanti
4. Rifat Muhammad

Daftar Isi

Abstrak

Penelitian ini bertujuan untuk membandingkan performa model *Machine Learning* dengan *Deep Learning* dalam memprediksi harga tiket pesawat berdasarkan data pemasaran yang diperoleh dari situs *"Ease My Trip"*. . Fokus utama dari penelitian ini adalah mengestimasi nilai target yang bersifat kontinu menggunakan teknik regresi, serta mengidentifikasi pola dan faktor yang mempengaruhi harga tiket dan jumlah pemesanan. Melalui analisis yang komprehensif, penelitian ini akan mengeksplorasi distribusi data, menguji hipotesis statistik, dan menilai faktor-faktor signifikan yang mempengaruhi variabel target. Metode Penelitian melibatkan analisis untuk menemukan korelasi dan hubungan antara berbagai variabel yang mempengaruhi harga tiket. Model *Linear Regressor* digunakan sebagai *baseline* untuk memprediksi harga tiket, dan hasilnya akan dibandingkan dengan model *Deep Learning* untuk menilai keunggulan dan kelemahan masing-masing pendekatan. Hasil penelitian diharapkan dapat memberikan wawasan yang lebih mendalam tentang faktor-faktor yang mempengaruhi harga tiket dan jumlah pemesanan, serta memberikan rekomendasi praktis bagi calon penumpang dalam pengambilan keputusan yang lebih baik saat memesan tiket penerbangan. Selain itu, studi ini juga bertujuan untuk mengevaluasi efektivitas model *Machine Learning* dan *Deep Learning* dalam konteks prediksi harga tiket penerbangan, sehingga dapat memberikan kontribusi signifikan dalam pengembangan algoritma prediktif di industri perjalanan. Temuan dari penelitian ini akan memberikan manfaat praktis bagi pengguna dengan menyediakan informasi yang mendalam mengenai dinamika harga tiket, serta membantu *platform* tersebut dalam mengoptimalkan layanan mereka melalui pemahaman yang lebih baik tentang perilaku pemesanan pengguna. Dengan demikian, penelitian ini tidak hanya berkontribusi pada literatur akademik tentang prediksi harga tiket penerbangan tetapi juga menawarkan aplikasi praktis yang dapat meningkatkan pengalaman pengguna dan efisiensi operasional platform pemesanan tiket.

Kata kunci: *Flight Price Prediction, Machine Learning, Deep Learning, Ease My Trip*, Tugas Akhir

Bab 1 Pendahuluan

1.1 Latar Belakang

Dalam beberapa dekade terakhir, industri penerbangan telah mengalami pertumbuhan yang signifikan, baik dalam jumlah penerbangan maupun jumlah penumpang. Salah satu tantangan utama bagi para pelancong adalah fluktuasi harga tiket pesawat yang seringkali tidak dapat diprediksi. Harga tiket pesawat dapat berubah dalam hitungan jam atau bahkan menit, tergantung pada berbagai faktor seperti permintaan pasar, musim, hari dalam minggu, waktu pembelian, dan kebijakan maskapai.

Prediksi harga tiket pesawat yang akurat memiliki manfaat yang besar bagi konsumen dan penyedia layanan perjalanan. Konsumen dapat merencanakan perjalanan mereka dengan lebih baik dan menghemat biaya, sementara penyedia layanan dapat mengoptimalkan strategi penetapan harga mereka untuk meningkatkan keuntungan dan efisiensi operasional. Oleh karena itu, pengembangan model prediksi harga tiket pesawat yang akurat menjadi sangat penting.

Kemajuan dalam bidang pembelajaran mesin (*Machine Learning*) dan pembelajaran mendalam (*Deep Learning*) menawarkan peluang baru untuk meningkatkan akurasi prediksi harga tiket pesawat. Model pembelajaran mesin tradisional seperti *Linear Regression, Decision Tree*, dan *Random Forest* telah digunakan secara luas untuk berbagai aplikasi prediksi. Di sisi lain, model pembelajaran mendalam seperti jaringan saraf tiruan (*Artificial Neural Networks*) dan LSTM (*Long Short-Term Memory*) telah menunjukkan kinerja yang luar biasa dalam menangani data yang kompleks dan memiliki sifat temporal.

Namun, perbandingan komprehensif antara model pembelajaran mesin tradisional dan model pembelajaran mendalam dalam konteks prediksi harga tiket pesawat masih jarang dilakukan. Penelitian ini bertujuan untuk mengisi kekosongan tersebut dengan melakukan evaluasi dan perbandingan mendetail antara kedua pendekatan ini. Dengan demikian, penelitian ini diharapkan dapat memberikan wawasan yang lebih dalam mengenai keunggulan dan keterbatasan masing-masing pendekatan serta memberikan rekomendasi yang lebih baik bagi pengembang dan praktisi di bidang ini.

1.2 Tujuan Penelitian

- Mengevaluasi dan membandingkan tingkat akurasi prediksi harga tiket pesawat yang dihasilkan oleh model pembelajaran *Machine Learning* seperti *Linear Regression, Decision Tree*, dan *Random Forest* dengan model *Deep Learning* seperti ANN.

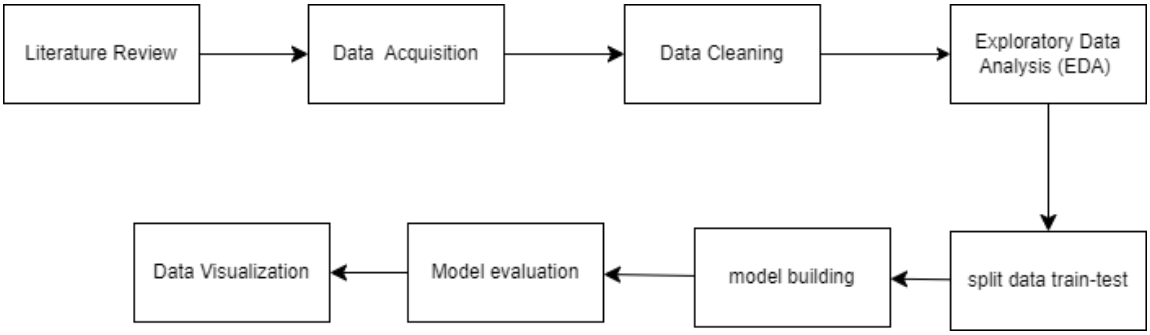
- Menganalisis kinerja model *Machine Learning* dan *Deep Learning* dalam hal metrik evaluasi seperti MSE (*Mean Squared Error*), MAE (*Mean Absolute Error*), dan *R-squared* (R^2).
- Menentukan variabel atau fitur apa saja yang paling berpengaruh terhadap prediksi harga tiket pesawat dalam kedua jenis model tersebut.

1.3 Manfaat Penelitian

Dengan adanya penelitian ini, diharapkan dapat dihasilkan model prediksi harga tiket pesawat yang lebih akurat serta mengetahui algoritma yang paling cocok dalam memprediksi harga tiket pesawat. Hasil penelitian ini diharapkan dapat membantu konsumen dalam merencanakan perjalanan dan menghemat biaya.

1.4 Metodologi Penelitian

Metodologi yang dilakukan pada penelitian dapat dilihat pada gambar dibawah ini.



1.4.1 Studi Literatur

Studi literatur ini bertujuan untuk memahami berbagai teknik dan algoritma yang telah digunakan dalam prediksi harga tiket pesawat serta mengevaluasi keunggulan dan keterbatasan masing-masing pendekatan. Berbagai penelitian telah mengeksplorasi penggunaan model pembelajaran mesin tradisional dan model pembelajaran mendalam dalam upaya untuk memprediksi harga tiket yang dinamis dan sering kali tidak terduga.

1.4.2 Data Acquisition

Penelitian ini menggunakan dataset yang berasal dari materi yang diberikan oleh Mitra Celerates yang berjudul "*Flight Price Prediction*". Dataset ini terdiri atas 11 kolom dan 300.183 baris data. Tiap kolom berisi data sebagai berikut:

1. *Airline*: Nama perusahaan maskapai penerbangan disimpan pada kolom maskapai penerbangan. Ini adalah fitur kategoris yang memiliki 6 maskapai berbeda.

2. *Flight*: Penerbangan menyimpan informasi mengenai kode penerbangan pesawat.
3. *Source City*: Kota asal penerbangan lepas landas. Ini adalah fitur kategoris yang memiliki 6 kota unik.
4. *Departure Time*: Ini adalah fitur kategoris turunan yang diperoleh dengan mengelompokkan periode waktu ke dalam bin. Ini menyimpan informasi tentang waktu keberangkatan dan memiliki 6 label waktu unik.
5. *Stops*: Fitur kategorik dengan 3 nilai berbeda yang menyimpan jumlah perhentian antara kota sumber dan kota tujuan.
6. *Arrival Time*: Ini adalah fitur kategoris turunan yang dibuat dengan mengelompokkan interval waktu ke dalam bin. Ini memiliki enam label waktu berbeda dan menyimpan informasi tentang waktu kedatangan.
7. *Destination City*: Kota dimana penerbangan akan mendarat. Ini adalah fitur kategoris yang memiliki 6 kota unik.
8. *Class*: Fitur kategorikal yang berisi informasi kelas kursi; ia memiliki dua nilai berbeda: Bisnis dan Ekonomi.
9. *Duration*: Fitur berkelanjutan yang menampilkan jumlah keseluruhan waktu yang diperlukan untuk melakukan perjalanan antar kota dalam hitungan jam.
10. *Days left*: Ini adalah karakteristik turunan yang dihitung dengan mengurangkan tanggal perjalanan dengan tanggal pemesanan.
11. *Price*: Variabel target menyimpan informasi harga tiket.

Berikut dataset dapat dilihat pada gambar dibawah ini.

airline	flight	source_cit	departure	stops	arrival_tin	destinatic	class	duration	days_left	price
SpiceJet	SG-8709	Delhi	Evening	zero	Night	Mumbai	Economy	2.17	1	5953
SpiceJet	SG-8157	Delhi	Early_Mor	zero	Morning	Mumbai	Economy	2.33	1	5953
AirAsia	I5-764	Delhi	Early_Mor	zero	Early_Mor	Mumbai	Economy	2.17	1	5956
Vistara	UK-995	Delhi	Morning	zero	Afternoor	Mumbai	Economy	2.25	1	5955
Vistara	UK-963	Delhi	Morning	zero	Morning	Mumbai	Economy	2.33	1	5955
Vistara	UK-945	Delhi	Morning	zero	Afternoor	Mumbai	Economy	2.33	1	5955
Vistara	UK-927	Delhi	Morning	zero	Morning	Mumbai	Economy	2.08	1	6060
Vistara	UK-951	Delhi	Afternoor	zero	Evening	Mumbai	Economy	2.17	1	6060
GO_FIRST	G8-334	Delhi	Early_Mor	zero	Morning	Mumbai	Economy	2.17	1	5954
GO_FIRST	G8-336	Delhi	Afternoor	zero	Evening	Mumbai	Economy	2.25	1	5954
GO_FIRST	G8-392	Delhi	Afternoor	zero	Evening	Mumbai	Economy	2.25	1	5954
GO_FIRST	G8-338	Delhi	Morning	zero	Afternoor	Mumbai	Economy	2.33	1	5954
Indigo	6E-5001	Delhi	Early_Mor	zero	Morning	Mumbai	Economy	2.17	1	5955
Indigo	6E-6202	Delhi	Morning	zero	Afternoor	Mumbai	Economy	2.17	1	5955
Indigo	6E-549	Delhi	Afternoor	zero	Evening	Mumbai	Economy	2.25	1	5955
Indigo	6E-6278	Delhi	Morning	zero	Morning	Mumbai	Economy	2.33	1	5955

1.4.3 Data Cleaning

Proses ini berguna untuk mendeteksi dan memperbaiki atau menghapus kesalahan dalam data untuk meningkatkan kualitasnya. Proses ini sangat penting dalam analisis data karena data yang tidak bersih dapat menghasilkan kesimpulan yang salah atau tidak akurat. Tahapan yang dilakukan pada proses ini adalah :

1. Identifikasi kesalahan data dengan mencari dan menangani nilai yang hilang dalam dataset, mengidentifikasi dan memutuskan apa yang harus dilakukan dengan data yang jelas tidak wajar, dan menangani inkonsistensi dalam penulisan data, seperti kesalahan ketik atau format yang salah.
2. Jika ada data yang hilang, maka gunakan opsi untuk mengisi nilai kosong atau menghapus baris atau kolom yang mengandung banyak nilai yang hilang.
3. Menyamakan skala data untuk memudahkan perbandingan dengan normalisasi

1.4.4 Exploratory Data Analysis

Exploratory Data Analysis (EDA) adalah proses eksplorasi dan analisis awal terhadap dataset untuk memahami karakteristik data yang dimiliki sebelum melakukan analisis lebih lanjut atau membangun model. Tujuan utama dari EDA adalah untuk menemukan pola-pola dalam data, mengidentifikasi anomali atau nilai yang hilang, serta memahami hubungan antar variabel.

1.4.5 Split Data Train-Test

Pada tahap ini, data asli dibagi menjadi dua subset: data pelatihan (*Training Data*) dan data pengujian (*Test Data*). Biasanya, data pelatihan digunakan untuk melatih model, sedangkan data pengujian digunakan untuk menguji kinerja model yang telah dilatih. *Train-test split* bertujuan untuk mengevaluasi kinerja model pada data yang independen atau data yang belum pernah dilihat sebelumnya. Hal ini penting untuk menghindari *Overfitting*, yaitu kondisi di mana model terlalu mempelajari detail dari data pelatihan dan tidak dapat menggeneralisasi dengan baik pada data baru.

1.4.6 Model Building

Model building adalah proses konstruksi dan pengembangan model *machine learning* atau statistik yang digunakan untuk memprediksi atau

menjelaskan perilaku atau fenomena dalam data. Algoritma yang digunakan adalah sebagai berikut:

1. Regresi linear
   Algoritma ini memodelkan hubungan linier antara variabel independen (fitur) dan variabel dependen (harga tiket). Algoritma ini bekerja dengan menggunakan metode kuadrat terkecil (*Ordinary Least Squares*) untuk meminimalkan jumlah kuadrat dari kesalahan prediksi.
2. *Decision Tree*
   Decision Tree adalah model prediktif yang menggunakan struktur pohon atau hirarki aturan keputusan untuk memprediksi nilai dari sebuah target berdasarkan fitur-fitur input. Algoritma ini membangun pohon keputusan dengan memisahkan data pada setiap node berdasarkan fitur yang mengurangi ketidakpastian secara maksimal.
3. *Random Forest*
   Random forest adalah ensemble learning yang terdiri dari banyak *Decision Tree* yang bekerja bersama-sama untuk meningkatkan akurasi prediksi dan mengurangi *Overfitting*. Algoritma ini menggunakan teknik *Bagging* untuk membangun setiap pohon pada subset data yang berbeda dan menggabungkan hasilnya dengan voting rata-rata.
4. *Artificial Neural Network* (ANN)
   ANN adalah model inspirasi biologis yang terdiri dari jaringan neuron atau unit pemrosesan yang saling terhubung. Mereka terdiri dari lapisan-lapisan neuron yang meneruskan sinyal dari input ke output.

1.4.7 Model Evaluation

Model *evaluation* adalah proses untuk mengukur kinerja model *machine learning* atau statistik dengan menggunakan metrik dan teknik evaluasi tertentu. Tujuan dari model *evaluation* adalah untuk menguji seberapa baik model dapat menggeneralisasi dari data pelatihan ke data yang belum pernah dilihat sebelumnya, serta untuk membandingkan berbagai model untuk menentukan model mana yang paling cocok untuk kasus penggunaan tertentu. Model *evaluation* dapat dilakukan dengan menggunakan matrik evaluasi. Metrik evaluasi yang digunakan pada kasus regresi adalah *mean squared error* (MSE), *mean absolute error* (MAE), koefisien determinasi (*R-squared*), dan visualisasi residual plot.

1.4.8 Data Visualization

Data *Visualization* adalah representasi grafis dari data menggunakan elemen visual seperti grafik, diagram, dan peta untuk menyampaikan informasi yang dapat dipahami dengan mudah dan cepat. Tujuan utama dari data visualization adalah untuk menggambarkan pola, tren, dan hubungan dalam data dengan cara yang intuitif dan efektif. Pada tahap ini, kami menggunakan aplikasi *Tableau* untuk membuat visualisasi data dalam bentuk *dashboard*.

Bab 2 Dasar Teori

2.1 Machine Learning

Machine Learning adalah cabang dari ilmu Kecerdasan Buatan yang berfokus pada bagaimana komputer dapat belajar dari data untuk meningkatkan kemampuan dan kecerdasannya. Menurut Budiharto (2016), Tipe dari kecerdasan buatan yang menyediakan komputer dengan kemampuan untuk belajar dari data, tanpa secara eksplisit harus mengikuti instruksi terprogram.

2.1.1 *Linear Regression*

Algoritma Linear Regression adalah salah satu jenis aturan dalam classification and regression pada data mining. Selain *Linear Regression*, yang termasuk dalam kategori ini adalah *Support Vector Machine, Logistic Regression*, dan lainnya. Analisis regresi linear adalah teknik data mining yang digunakan untuk menentukan adanya hubungan antara variabel yang ingin diprediksi dengan variabel lainnya.

2.1.2 Decision Tree Regressor

Decision Trees, atau Pohon Keputusan, adalah jenis algoritma untuk pembelajaran mesin yang digunakan dalam pemodelan prediktif. Model *Decision Trees* direpresentasikan sebagai pohon biner, di mana setiap node merepresentasikan variabel input (x) dan cabang-cabangnya merepresentasikan nilai dari variabel input tersebut. Sementara itu, simpul-simpul dalam pohon merepresentasikan variabel output (y) atau kelas. Node paling atas dari decision tree ini dikenal sebagai root. Algoritma ini dinamakan pohon keputusan karena aturan-aturan yang terbentuk menyerupai struktur pohon .

2.1.3 Random Forest Regressor

Random Forest adalah salah satu algoritma pembelajaran mesin yang paling populer dan kuat. Algoritma ini merupakan pengembangan dari pohon keputusan. Dinamakan *Random Forest* atau "hutan acak" karena terdiri dari sekumpulan pohon keputusan yang dibangun secara acak. Setiap node dalam pohon keputusan bekerja pada subset fitur yang dipilih secara acak (tidak menggunakan algoritma *greedy*) untuk menghitung outputnya. *Random Forest* kemudian menggabungkan output dari masing-masing pohon keputusan individu untuk menghasilkan output akhir.

2.2 Deep Learning

Proses pembelajaran mesin dilakukan pada komputer yang bertujuan untuk mengklasifikasikan data citra menjadi hasil klasifikasi berupa prediksi. Teknologi *Deep Learning* merepresentasikan konsep yang kompleks dengan memecahnya menjadi rangkaian konsep yang lebih sederhana. *Deep Learning* adalah algoritma jaringan saraf tiruan yang memanfaatkan data sebagai input dan memprosesnya melalui beberapa lapisan tersembunyi (*Hidden Layers*). Setelah itu, algoritma ini melakukan transformasi non-linear pada data masukan untuk menghitung nilai output.

2.2.1 *Artificial Neural Network* (ANN):

Jaringan saraf tiruan (*Artificial Neural Network* atau ANN) adalah model yang terdiri dari sekelompok unit pemrosesan, didesain untuk meniru jaringan saraf manusia. ANN adalah sistem adaptif yang dapat mengubah strukturnya untuk menyelesaikan masalah dengan menggunakan informasi dari luar maupun dari dalam sistem itu sendiri. Neuron-neuron dalam jaringan ini dikelompokkan dalam yang disebut sebagai layer atau lapisan. Secara umum, ANN terdiri dari tiga lapisan utama: input layer untuk menampung data masukan, *Hidden Layer* (lapisan proses) untuk mengenali pola atau objek, dan *Output Layer* untuk menghasilkan hasil dari pengenalan objek tersebut.

2.3 Evaluasi Model

Evaluasi model merujuk pada proses mengukur kinerja atau keefektifan suatu model prediksi atau klasifikasi berdasarkan data yang telah diuji. Tujuan utamanya adalah untuk menilai seberapa baik model dapat memprediksi nilai yang diharapkan atau mengklasifikasikan data dengan benar.

Evaluasi kinerja model prediksi dilakukan menggunakan berbagai metrik, termasuk:

2.3.1 *R-squared* (R^2)

R-squared (R^2) adalah salah satu metrik yang digunakan dalam analisis regresi untuk mengukur seberapa cocok model regresi dengan data yang diamati.Skor R-squared (R^2) memberikan ukuran seberapa dekat data yang diamati dengan garis regresi yang diprediksi oleh model.

2.3.1 *Mean Absolute Error* (MAE)

Mean Absolute Error (MAE) adalah metrik penting yang sering digunakan dalam evaluasi model. Meskipun telah lama digunakan untuk menilai kinerja model, belum ada kesepakatan umum mengenai metrik yang paling tepat untuk mengevaluasi kesalahan model.

2.3.3 *Mean Squared Error* (MSE)

MSE atau *Mean Squared Error* adalah salah satu metrik evaluasi yang umum digunakan dalam pembelajaran mesin, terutama dalam konteks regresi. MSE mengukur rata-rata dari kuadrat perbedaan antara nilai yang diprediksi oleh model dan nilai sebenarnya dari data.

2.3.4 *Root Mean Squared Error (*RMSE)

RMSE memberikan gambaran tentang tingkat kesalahan absolut dari model dalam memprediksi nilai, dengan memberikan bobot lebih besar pada kesalahan yang lebih besar. Dalam konteks yang lebih luas, RMSE sering digunakan dalam berbagai bidang penelitian seperti meteorologi, kualitas udara, dan penelitian iklim untuk mengevaluasi kinerja model.

Perbedaan utama antara RMSE dan MAE (Mean Absolute Error) adalah RMSE lebih sensitif terhadap perbedaan dalam nilai error yang besar, sementara MAE memberikan bobot yang sama pada semua kesalahan. Meskipun RMSE sering digunakan, terdapat juga argumen mengenai keambiguitasan dalam interpretasi, terutama dalam kasus distribusi error yang tidak normal atau ketika nilai ekstrim mempengaruhi hasil secara signifikan.

2.3.5 *Akaike Information Criterion* (AIC)

AIC dirancang untuk menyeimbangkan kesesuaian dan kompleksitas model, yang bertujuan untuk memilih model yang meminimalkan kesalahan kuadrat rata-rata dalam prediksi atau estimasi, terutama ketika model sebenarnya tidak termasuk dalam kandidat model yang dipertimbangkan. Hal ini efisien dalam hal ini tetapi kurang konsisten, yang berarti tidak selalu memilih model yang sebenarnya ketika dimasukkan dalam kumpulan kandidat.

2.3.6 *Bayesian Information Criterion* (BIC)

Berbanding terbalik dengan AIC, BIC konsisten secara asimtotik, artinya BIC akan memilih model yang sebenarnya jika kondisi tertentu terpenuhi, termasuk model yang sebenarnya ada di antara para kandidat. Namun, BIC tidak efisien jika model sebenarnya tidak ada dalam kumpulan kandidat.

Bab 3 Hasil dan Pembahasan

3.1 Pengolahan Data

3.1.1 Data Preparation

3.1.1.1 Check Missing Value

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

Unnamed: 0          0
airline             0
flight              0
source_city         0
departure_time      0
stops               0
arrival_time        0
destination_city    0
class               0
duration            0
days_left           0
price               0
dtype: int64
```

```
df.isin(['?']).sum()

Unnamed: 0          0
airline             0
flight              0
source_city         0
departure_time      0
stops               0
arrival_time        0
destination_city    0
class               0
duration            0
days_left           0
price               0
dtype: int64
```

3.1.1.2 Replace Value Data

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300153 entries, 0 to 300152
Data columns (total 12 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   Unnamed: 0        300153 non-null  int64
 1   airline           300153 non-null  object
 2   flight            300153 non-null  object
 3   source_city       300153 non-null  object
 4   departure_time    300153 non-null  object
 5   stops             300153 non-null  object
 6   arrival_time      300153 non-null  object
 7   destination_city  300153 non-null  object
 8   class             300153 non-null  object
 9   duration          300153 non-null  float64
 10  days_left         300153 non-null  int64
 11  price             300153 non-null  int64
dtypes: float64(1), int64(3), object(8)
memory usage: 27.5+ MB
```

```
df['airline'].unique()
```

```
array(['SpiceJet', 'AirAsia', 'Vistara', 'GO_FIRST', 'Indigo',
       'Air_India'], dtype=object)
```

```
# replace isi data variabel class
df['class'] = df['class'].replace('Business',1)
df['class'] = df['class'].replace('Economy',0)
```

```
# replace isi data variabel stop

df['stops'] = df['stops'].replace('zero',0)
df['stops'] = df['stops'].replace('one',1)
df['stops'] = df['stops'].replace('two_or_more',2)
```

Label Encoder

```
label_encoding_features = ['airline', 'flight','departure_time','arrival_time','source_city', 'destination_city']

label_encoder = LabelEncoder()
for col in label_encoding_features:
    df[col] = label_encoder.fit_transform(df[col])
```

```
df.dtypes
```

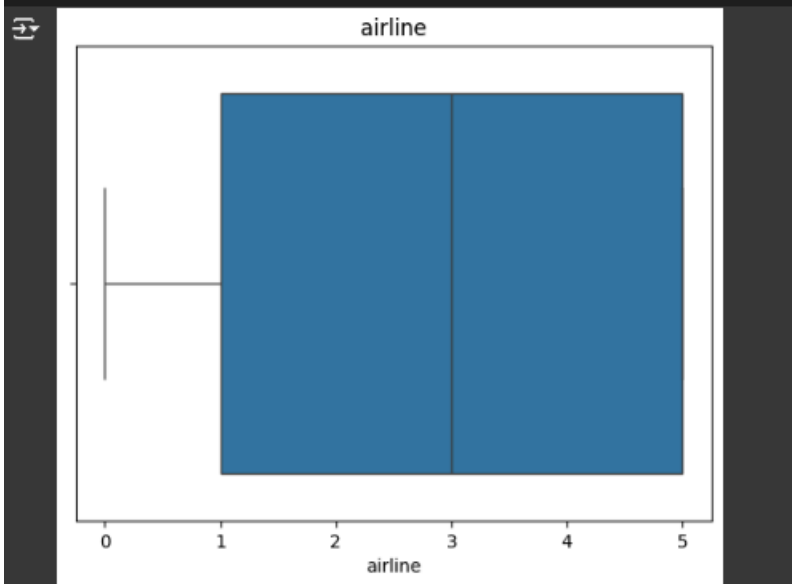
```
Unnamed: 0          int64
airline             int64
flight              int64
source_city         int64
departure_time      int64
stops               int64
arrival_time        int64
destination_city    int64
class               int64
duration          float64
days_left           int64
price               int64
dtype: object
```

Untuk mengubah fitur kategorik menjadi numerik, pertama tentukan fitur yang ingin di label, yaitu *airline, flight, departure time, arrival time, source city dan destination city*. Selanjutnya, lakukan iterasi terhadap setiap kolom dalam 'label_encoding_features' menggunakan *looping for*. Di setiap iterasi, 'fit_transform()' dari LabelEncoder digunakan untuk melakukan proses 'label encoding fit_transform()' dari *LabelEncoder* digunakan untuk melakukan proses *label encoding*.

3.1.2 EDA (Exploratory Data Analysis)

3.1.2.1 Check Outlier

```
num_cols = df.select_dtypes(include=['int64','float64']).columns.tolist()

# create boxplots for numerical columns
for col in num_cols:
    plt.figure()
    sns.boxplot(data=df, x=col)
    plt.title(col)
```



Untuk melihat *outlier*, buat *boxplot* untuk melihat persebaran data. Pertama, kolom tipe numerik dipilih dari dataframe ‘df’. Fungsi ‘select_dtypes’ digunakan untuk memilih kolom dengan tipe data int64 dan float64. Jika sudah, kolom akan dikonversi menjadi list.

Selanjutnya, iterasi akan dilakukan terhadap setiap kolom dalam ‘num_cols’ menggunakan *looping for*. Di setiap iterasi, dilakukan pembuatan *boxplot.* Setiap *boxplot* ditampilkan dalam sebuah *figure* yang baru dengan judul yang sesuai dengan nama kolom yang sedang diproses.

```
[19] Q1 = df['airline'].quantile(0.25)
     Q3 = df['airline'].quantile(0.75)
     IQR = Q3 - Q1
     lower_bound = Q1 - 1.5 * IQR
     upper_bound = Q3 + 1.5 * IQR
     outliers = df[(df['airline'] < lower_bound) | (df['airline'] > upper_bound)]
     print(f"Number of Outlier in airline Column : {len(outliers)}")

     Number of Outlier in airline Column : 0
```

Lalu untuk melihat outlier pada kolom airline, kuartil pertama (Q1) dan kuartil ketiga (Q3) dari data dalam kolom 'airline' dihitung menggunakan df['airline'].quantile(0.25) dan df['airline'].quantile(0.75), masing-masing.

Kemudian, rentang IQR *(Interquartile Range)* dihitung dengan mengurangi Q1 dari Q3 (IQR = Q3 - Q1).

Selanjutnya, batas bawah *(lower_bound)* dan batas atas *(upper_bound)* untuk *outlier* dihitung dengan rumus:

- *lower_bound* = Q1 - 1.5 * IQR
- *upper_bound* = Q3 + 1.5 * IQR

Outlier kemudian diidentifikasi dengan memfilter baris-baris dalam dataframe df di mana nilai dalam kolom 'airline' lebih kecil dari *lower_bound* atau lebih besar dari *upper_bound.*

Terakhir, jumlah *outlier* yang teridentifikasi dicetak menggunakan '*print()*'.

3.1.2.1 Nilai Statistik

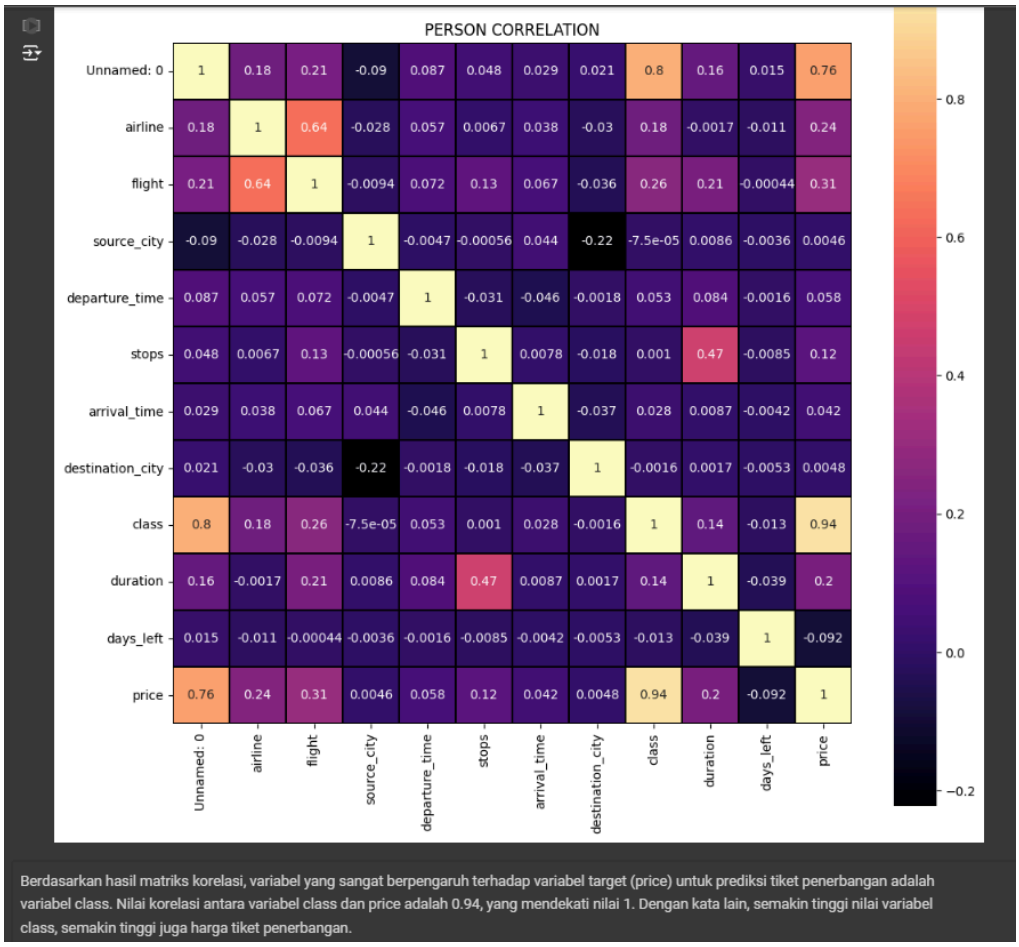
	airline	flight	source_city	departure_time	\
count	300153.000000	300153.000000	300153.000000	300153.000000	
mean	3.104873	1088.336618	2.577592	2.417337	
std	1.833265	426.694820	1.751762	1.754276	
min	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	783.000000	1.000000	1.000000	
50%	3.000000	1142.000000	2.000000	2.000000	
75%	5.000000	1486.000000	4.000000	4.000000	
max	5.000000	1560.000000	5.000000	5.000000	

	stops	arrival_time	destination_city	class	\
count	300153.000000	300153.000000	300153.000000	300153.000000	
mean	0.924312	3.074086	2.588303	0.311464	
std	0.398106	1.741666	1.744543	0.463093	
min	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	2.000000	1.000000	0.000000	
50%	1.000000	4.000000	3.000000	0.000000	
75%	1.000000	5.000000	4.000000	1.000000	
max	2.000000	5.000000	5.000000	1.000000	

	duration	days_left	price
count	300153.000000	300153.000000	300153.000000
mean	12.221021	26.004751	20889.660523
std	7.191997	13.561004	22697.767366
min	0.830000	1.000000	1105.000000
25%	6.830000	15.000000	4783.000000
50%	11.250000	26.000000	7425.000000
75%	16.170000	38.000000	42521.000000
max	49.830000	49.000000	123071.000000

3.1.2.3 *Pearson Correlation*

```
f,ax=plt.subplots(figsize=(12,12))
plt.title("PERSON CORRELATION")
sns.heatmap(
    df.astype(float).corr(),
    linewidth=0.25,
    vmax=1.0,
    square=True,
    cmap='magma',
    linecolor='black',
    annot=True
    )
```



Berdasarkan hasil matriks korelasi, variabel yang sangat berpengaruh terhadap variabel target (price) untuk prediksi tiket penerbangan adalah variabel class. Nilai korelasi antara variabel class dan price adalah 0.94, yang mendekati nilai 1. Dengan kata lain, semakin tinggi nilai variabel class, semakin tinggi juga harga tiket penerbangan.

Pada *code* ini, kita membuat sebuah *figure* adan *axis* dengan ukuran 12x12 *inci.* Judul pada *plo*t ini adalah "Pearson Correlation". Selanjutnya dengan menggunakan *library seaborn*, gunakan *plot* untuk membuat *heatmap*. Dengan menggunakan 'df.astype(float).corr()', DataFrame df diubah tipe datanya menjadi *float* dan kemudian dihitung matrik korelasinya. Matrik korelasi ini akan menunjukkan seberapa kuat dan arah korelasi antara setiap pasangan variabel dalam DataFrame.

Ketebalan garis antar sel *heatmap* adalah 0,25 lalu nilai maksimum pada skala warna *heatmap* adalah 0,1. Dengan menggunakan 'square = True',

buat *heatmap* menjadi kotak dan dengan menggunakan ‘cmap =’magma’’, *heatmap* akan menggunakan skala warna magma. Tambahkan ‘linecolor=’black’’ untuk mengubah garis yang memisahkan sel pada *heatmap* menjadi warna hitam dan terakhir, untuk menambahkan nilai korelasi di dalam setiap sel *heatmap*, gunakan ‘annot = True’.

Berdasarkan hasil matriks korelasi, variabel yang sangat berpengaruh terhadap variabel target (price) untuk prediksi tiket penerbangan adalah variabel class. Nilai korelasi antara variabel class dan price adalah 0.94, yang mendekati nilai 1. Dengan kata lain, semakin tinggi nilai variabel class, semakin tinggi juga harga tiket penerbangan.

3.2 Modeling Machine Learning/ Deep Learning

3.2.1 Train dan Test

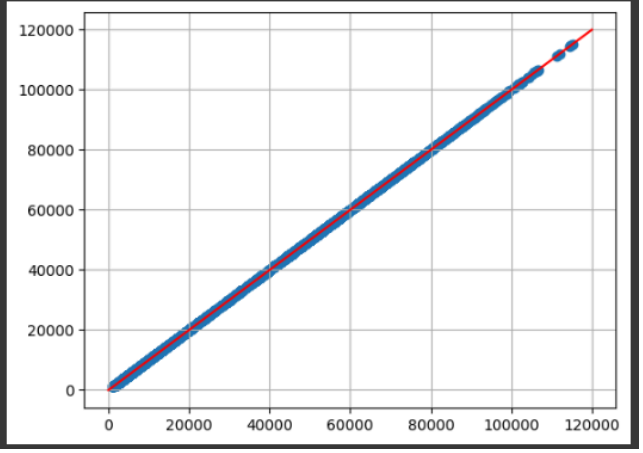
```
[24] # Membagi data menjadi training dan testing
     X_train, X_test, y_train, y_test = train_test_split(X,Y, test_size=0.2, random_state=42)
```

Pada pembagian data *train* dan *test* menggunakan parameter “test_size=0.2”. Dalam hal ini, 20% dari data total akan digunakan sebagai data penguji, dan sisanya 80% akan digunakan sebagai data pelatihan. Kemudian parameter selanjutnya “random_state=42” digunakan untuk memastikan bahwa pemisahan data konsisten setiap kali kode di jalankan. Angka 42 adalah seed untuk generator angka acak.

3.2.2 Algoritma Regresi Linear

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model_lr = model.fit(X_train, y_train)
y_pred_lr = model_lr.predict(X_test)

r2_s = r2_score(y_test, y_pred_lr)
print('R2 Score:', r2_s)

mae = mean_absolute_error(y_test, y_pred_lr)
print('MAE:', mae)

mse = mean_squared_error(y_test, y_pred_lr)
print('MSE:', mse)

rmse = np.sqrt(mse)
print('RMSE:', rmse)

# Menghitung AIC dan BIC
n = len(y_test)
p = X_train.shape[1] + 1

rss = np.sum((y_test - y_pred_lr) ** 2)
aic = n * np.log(rss/n) + 2 * p
bic = n * np.log(rss/n) + np.log(n) * p

print('AIC:', aic)
print('BIC:', bic)

print("="*71)

plt.scatter(y_test, y_pred_lr)
plt.plot(np.arange(0,120000), np.arange(0,120000), color = 'red')
plt.grid(True)
```

```
R2 Score: 1.0
MAE: 2.4606732085437902e-11
MSE: 8.485735196453297e-22
RMSE: 2.913028526543003e-11
AIC: -2912587.203310449
BIC: -2912470.1692975815
=======================================================================
```



Pertama, kita membuat objek model *Linear Regression*. Selanjutnya, model dilatih dengan menggunakan data latih (X_train sebagai fitur dan y_train sebagai target) menggunakan metode model.fit(X_train, y_train). Setelah model dilatih, kita melakukan prediksi terhadap data uji (X_test) dengan

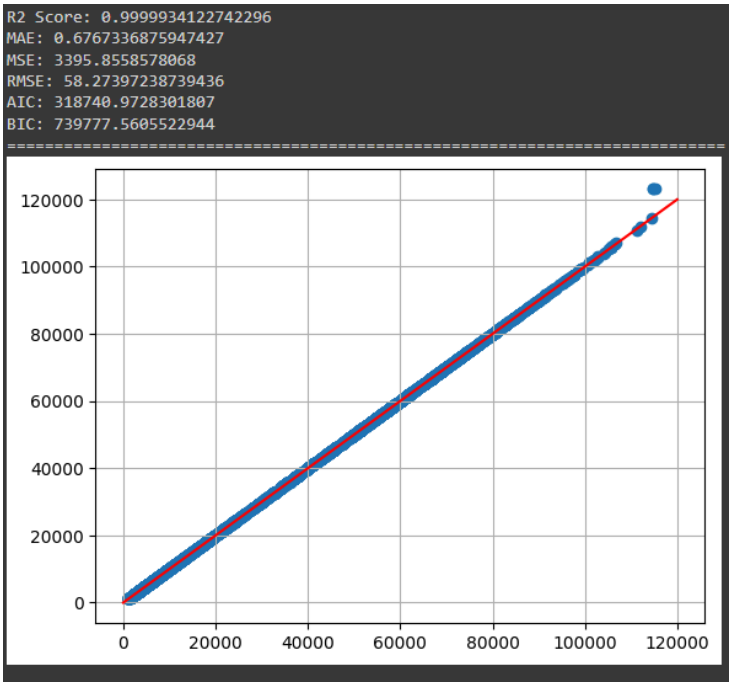
menggunakan model_lr.predict(X_test), yang menghasilkan prediksi yang disimpan dalam y_pred_lr.

Untuk mengevaluasi kinerja model, digunakan beberapa metrik evaluasi regresi seperti koefisien determinasi *(R-squared), mean absolute error* (MAE), *mean squared error* (MSE), dan *root mean squared error* (RMSE). Metrik ini dihitung menggunakan fungsi-fungsi dari *sklearn.metrics*. Selanjutnya, dilakukan perhitungan untuk *Akaike Information Criterion* (AIC) dan *Bayesian Information Criterion* (BIC) untuk memberikan informasi tambahan tentang kualitas model.

Terakhir, untuk memvisualisasikan hasil prediksi, kita menggunakan *matplotlib*. *Scatter plot* digunakan untuk membandingkan nilai aktual (y_test) dengan nilai prediksi (y_pred_lr), sedangkan garis merah menggambarkan garis diagonal y=x untuk menunjukkan seberapa baik prediksi mendekati nilai sebenarnya. *Plot* ini dilengkapi dengan *grid* untuk mempermudah interpretasi visualisasi.

3.2.3 Algoritma Decision Tree Regressor

```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor()
model_dtr = model.fit(X_train, y_train)
y_pred_dtr = model_dtr.predict(X_test)

r2_s = r2_score(y_test, y_pred_dtr)
print('R2 Score:', r2_s)

mae = mean_absolute_error(y_test, y_pred_dtr)
print('MAE:', mae)

mse = mean_squared_error(y_test, y_pred_dtr)
print('MSE:', mse)

rmse = np.sqrt(mse)
print('RMSE:', rmse)

# Menghitung AIC dan BIC
n = len(y_test)
p = model_dtr.tree_.node_count

rss = np.sum((y_test - y_pred_dtr) ** 2)
bic = n * np.log(rss/n) + np.log(n) * p

print('AIC:', aic)
print('BIC:', bic)

print("="*76)

plt.scatter(y_test, y_pred_dtr)
plt.plot(np.arange(0,120000), np.arange(0,120000), color = 'red')
plt.grid(True)
```



Pertama, definisikan model *Decision Tree Regressor* dengan membuat variabel model. Selanjutnya latih model dengan menggunakan model.fit(X_train, y_train). Jika sudah, lakukan prediksi terhadap data uji yaitu x_test menggunakan model_dtr.predict(X_test) yang hasilnya akan disimpan di 'y_pred_dtr'.

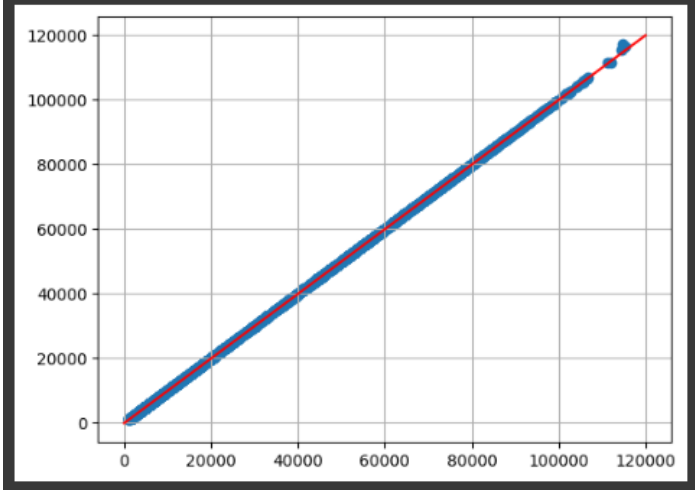
Untuk mengevaluasi kinerja model, digunakan beberapa metrik evaluasi regresi seperti koefisien determinasi *(R-squared), mean absolute error* (MAE), *mean squared error* (MSE), dan *root mean squared error* (RMSE). Metrik ini dihitung menggunakan fungsi-fungsi dari *sklearn.metrics*. Selanjutnya, dilakukan perhitungan untuk Akaike Information Criterion (AIC) dan *Bayesian Information Criterion* (BIC) untuk memberikan informasi tambahan tentang kualitas model.

Terakhir, untuk memvisualisasikan hasil prediksi, kita menggunakan *matplotlib. Scatter plot* digunakan untuk membandingkan nilai aktual (y_test) dengan nilai prediksi (y_pred_dtr), sedangkan garis merah menggambarkan garis diagonal y=x untuk menunjukkan seberapa baik prediksi mendekati nilai sebenarnya. *Plot* ini dilengkapi dengan *grid* untuk mempermudah interpretasi visualisasi.

3.2.4 Algoritma Random Forest Regressor

```
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor()
model_rf = model.fit(X_train, y_train)
y_pred_rf = model_rf.predict(X_test)

# Evaluasi model

r2_s = r2_score(y_test, y_pred_rf)
print('R2 Score:', r2_s)

mae = mean_absolute_error(y_test, y_pred_rf)
print('MAE:', mae)

mse = mean_squared_error(y_test, y_pred_rf)
print('MSE:', mse)

rmse = np.sqrt(mse)
print('RMSE:', rmse)

n = len(y_test)
p = len(model_rf.feature_importances_)

rss = np.sum((y_test - y_pred_rf) ** 2)
aic = n * np.log(rss/n) + 2 * p
bic = n * np.log(rss/n) + np.log(n) * p

print('AIC:', aic)
print('BIC:', bic)

print("="*76)

plt.scatter(y_test, y_pred_rf)
plt.plot(np.arange(0,120000), np.arange(0,120000), color = 'red')
plt.grid(True)
```

```
R2 Score: 0.9999996077659138
MAE: 0.34615282104247225
MSE: 202.18971850877062
RMSE: 14.21934311101503
AIC: 318740.9728301807
BIC: 318849.00422667403
============================================================================
```



Pertama, kita membuat objek model *Random Forest Regressor*. Selanjutnya, model dilatih dengan menggunakan data latih (X_train sebagai

fitur dan y_train sebagai target) menggunakan metode model.fit(X_train, y_train). Setelah model dilatih, kita melakukan prediksi terhadap data uji (X_test) dengan menggunakan model_rf.predict(X_test), yang menghasilkan prediksi yang disimpan dalam y_pred_rf.

Untuk mengevaluasi kinerja model, digunakan beberapa metrik evaluasi regresi seperti koefisien determinasi *(R-squared), mean absolute error* (MAE), *mean squared error* (MSE), dan *root mean squared error* (RMSE). Metrik ini dihitung menggunakan fungsi-fungsi dari *sklearn.metrics*. Selanjutnya, dilakukan perhitungan untuk *Akaike Information Criterion* (AIC) dan *Bayesian Information Criterion* (BIC) untuk memberikan informasi tambahan tentang kualitas model.

Terakhir, untuk memvisualisasikan hasil prediksi, kita menggunakan *matplotlib. Scatter plot* digunakan untuk membandingkan nilai aktual (y_test) dengan nilai prediksi (y_pred_rf), sedangkan garis merah menggambarkan garis diagonal y=x untuk menunjukkan seberapa baik prediksi mendekati nilai sebenarnya. *Plot* ini dilengkapi dengan *grid* untuk mempermudah interpretasi visualisasi.

3.2.5 Algoritma Artificial Neural Network (ANN)

```
network = models.Sequential()
network.add(layers.Dense(32, activation='relu', input_shape=(12,)))
network.add(layers.Dense(64, activation='relu'))
network.add(layers.Dense(1))

#Evaluate model
network.compile(loss='mse', optimizer='nadam', metrics=['mae'])

exc = network.fit(X_train, y_train, epochs=50, batch_size=32,  verbose=1, validation_split=0.3)
```

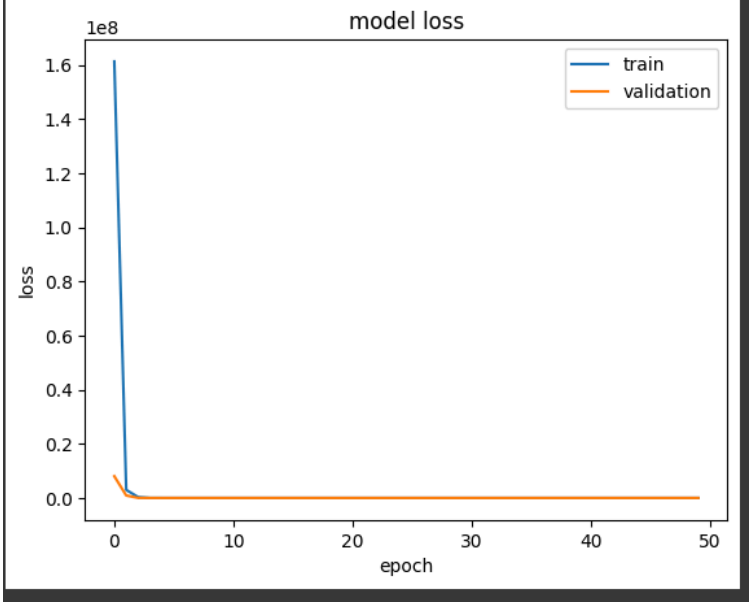
```
5253/5253 [==============================] - 15s 3ms/step - loss: 22.6202 - mae: 1.9006 - val_loss: 317.76
Epoch 23/50
5253/5253 [==============================] - 15s 3ms/step - loss: 25.4437 - mae: 1.9693 - val_loss: 31.119
Epoch 24/50
5253/5253 [==============================] - 15s 3ms/step - loss: 24.7900 - mae: 1.9349 - val_loss: 1.5412
Epoch 25/50
```

```
y_pred = network.predict(X_test)

r2_s = r2_score(y_test, y_pred)
print('R2 Score:', r2_s)

mae = mean_absolute_error(y_test, y_pred)
print('MAE:', mae)

mse = mean_squared_error(y_test, y_pred)
print('MSE:', mse)

rmse = np.sqrt(mse)
print('RMSE:', rmse)
```

```
1876/1876 [==============================] - 2s 1ms/step
R2 Score: 0.9999999987575747
MAE: 0.6150766826299902
MSE: 0.6404482087848052
RMSE: 0.8002800814619875
```

```
print(exc.history.keys())
plt.plot(exc.history['loss'])
plt.plot(exc.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper right')
plt.show()
```

```
dict_keys(['loss', 'mae', 'val_loss', 'val_mae'])
```



Pertama, lapisan *dense* dengan 32 unit dan fungsi aktivasi ReLU ditambahkan sebagai lapisan pertama dengan input 12 fitur. Lalu lapisan diikuti dengan lapisan *dense* kedua dengan 64 unit dan aktivasi ReLU. Lapisan terakhir adalah lapisan *dense* tanpa fungsi aktivasi tertentu yang digunakan untuk menghasilkan *output* regresi tunggal.

Setelah menambahkan semua lapisan, model di kompilasi menggunakan 'network_compile()' dengan fungsi kerugian mean *squared error ('loss ='mse"), optimizer nadam* dan *metrik mean absolute error.*

Selanjutnya, latih model menggunakan 'network_fit()' dengan data latih. Model dilatih selama 50 *epochs* dengan ukuran *batch* sebesar 32. Parameter *verbose* digunakan untuk menampilkan detail proses pelatihan. Disini, *validation split* adalah 0,3 yang menunjukkan bahwa 30% data latih digunakan untuk validasi.

Setelah latihan selesai, dilakukan prediksi menggunakan data uji (X_test), yang menghasilkan nilai prediksi yang disimpan pada 'y_pred'.

Untuk mengevaluasi kinerja model, digunakan beberapa metrik evaluasi regresi seperti koefisien determinasi *(R-squared)*, *mean absolute error* (MAE), *mean squared error* (MSE), dan *root mean squared error* (RMSE). Metrik ini dihitung menggunakan fungsi-fungsi dari *sklearn.metrics.*

Terakhir, lakukan visualisasi proses pelatihan model dengan menggunakan *matplotlib. Plot* menunjukkan grafik kerugian (*loss*) dari setiap *epoch* pada data latih (*train*) dan data validasi (*validation*). Ini membantu untuk mengevaluasi seberapa baik model berkinerja selama pelatihan dan apakah ada *overfitting* dan *underfitting* yang terjadi.

3.3 Visualisasi Data

Berikut adalah hasil visualisasi data menggunakan aplikasi Tableau :

3.3.1 Visualisasi Data Model

3.3.1.1 Summary Predict by Airline

Visualisasi menampilkan ringkasan nilai sebenarnya (*actual)* dengan nilai predik berdasarkan jenis pesawatnya.

Summary Predict by Airline

Airline	Actual	Predicted (Decission Tree)	Predicted (Random Forest)	Predicted (ANN)	Predicted (Linear Regression)
Air_India	577.173.245,00	577.173.112,00	577.173.003,73	577.172.640,12	577.173.245,00
AirAsia	46.605.910,00	46.605.868,00	46.605.869,88	46.606.377,60	46.605.910,00
GO_FIRST	63.147.269,00	63.147.315,00	63.147.286,46	63.147.853,81	63.147.269,00
Indigo	123.355.310,00	123.355.379,00	123.355.250,73	123.356.391,78	123.355.310,00
SpiceJet	46.254.945,00	46.254.938,00	46.254.884,66	46.255.302,49	46.254.945,00
Vistara	978.367.796,00	978.374.959,00	978.373.727,47	978.365.999,02	978.367.796,00

3.3.1.2 Actual & Predicted Difference

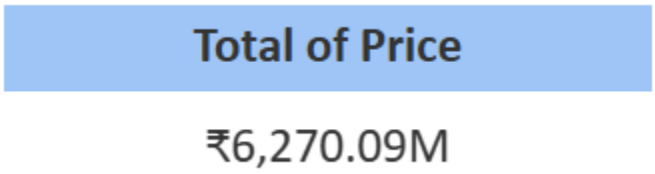
Visualisasi dibawah menampilkan selisih dari nilai sebenarnya *(actual)* dengan nilai prediksi. Dengan hasil model yang paling baik adalah Linear Regression dengan selisih 0 (nol).

Actual & Predicted Difference			
ANN	Decission Tree	Linerar Regression	Random Forest
1.462	-7.015	0	-5.471

3.3.2 Visualisasi Data Bisnis

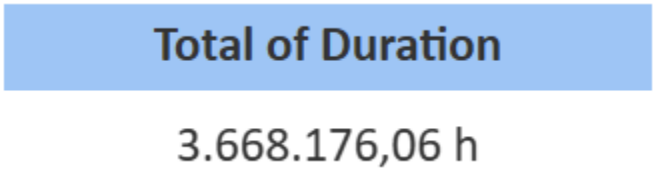
3.3.2.1 Total of Price

Total penjumlahan (SUM) dari semua *Price* sebesar ₹6.270.09*M*



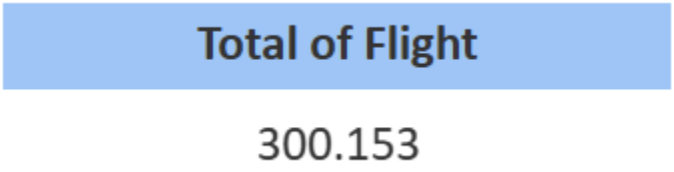
3.3.2.2 Total of Duration

Total durasi penerbangan sebesar 3.668.176,06 jam dihasilkan dari semua *flight*



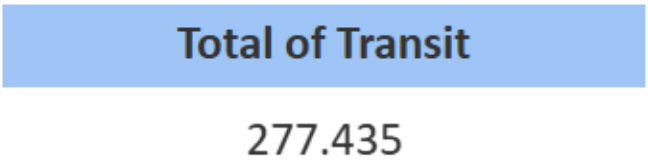
3.3.2.3 Total of Flight

Total penerbangan dari semua *Class* mencapai 300.153 *flight*



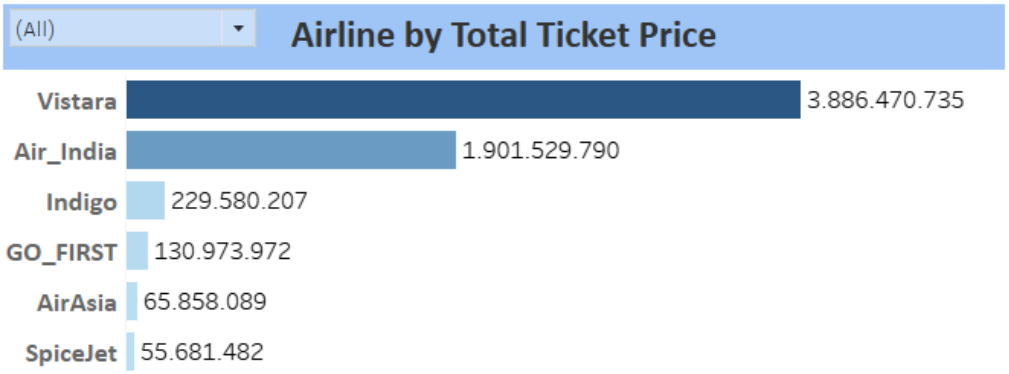
3.3.2.4 Total of Transit

Dari 300.153 *flight* memiliki jumlah transit yang berbeda , jumlah transit dari penerbangan ada yang *zero* ,*one* , atau *two* . Jika semua *flight* diakumulasikan berjumlah 277.435 kali transit .



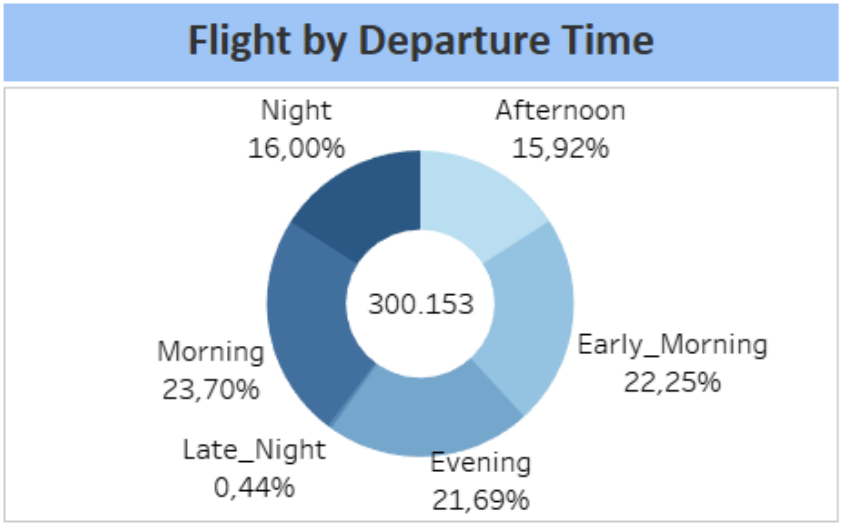
3.3.2.5 Airline by Total Ticket Price

Dari bagan batang *horizontal* ini telah diperoleh informasi bahwa *airline* Vistara memegang posisi dominan dalam hal total penjualan tiket, menunjukkan kemungkinan target pasar yang lebih besar atau harga tiket yang lebih tinggi, secara signifikan melebihi maskapai lain dengan total 3.896.470.735. Sedangkan total harga tiket terendah di antara maskapai yang terdaftar, yaitu SpiceJet 55.681.482 menunjukkan harga tiket yang lebih rendah, jumlah penerbangan yang lebih sedikit, atau cakupan pasar yang lebih kecil.



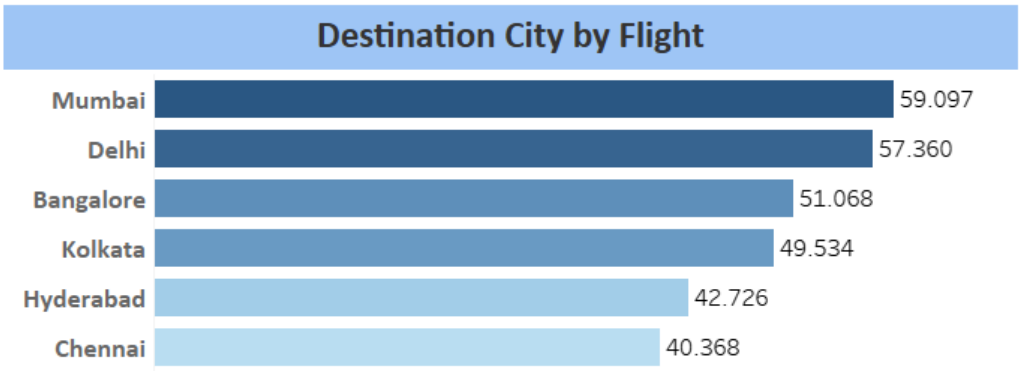
3.3.2.6 Flight by Departure Time

Dari *pie chart* dibawah ini, dapat disimpulkan dari 300.153 penerbangan yang sudah berjalan, mayoritas penerbangan berangkat pada *morning* dengan nilai 23,70% lalu disusul dengan *early morning* sebesar 22,25%. Pada peringkat ketiga, ada *evening* dengan persentase 21,69%. Sisanya, beberapa penerbangan berangkat pada *night* dengan persentase 16% dan *afternoon* dengan persentase 15,92%. Hanya ada sedikit penerbangan yang terbang pada *late_night,* hal ini dibuktikan dengan persentasenya yang sangat kecil, yaitu 0,44%.



3.3.2.7 Destination City by Flight

Pada bagan dibawah ini, dapat diperoleh bahwa Mumbai adalah kota tujuan dengan jumlah penerbangan terbanyak, sebanyak 59.097 penerbangan serta Chennai adalah kota tujuan dengan jumlah penerbangan paling sedikit,



yaitu 40.368 penerbangan. Dengan demikian, Mumbai merupakan kota dengan frekuensi penerbangan tertinggi, sedangkan Chennai memiliki frekuensi penerbangan terendah di antara kota-kota tujuan lainnya.

3.3.2.8 Average Price by Arrival and Departure Time

Pada tabel dibawah ini menunjukan rata-rata tiket berdasarkan waktu kedatangan dan keberangkatan . Untuk kombinasi harga tertinggi terjadi ketika tiba pada *evening* dan berangkat pada *night* yang mencapai harga ₹31.426 . Sebaliknya , kombinasi harga terendah terjadi ketika tiba pada *late_night* dan berangkat pada *late_night*, dengan harga mencapai ₹4.288 .

Dari tabel dibawah dapat disimpulkan ,umumnya keberangkatan pada *evening* dan *night* cenderung lebih mahal dibandingkan waktu lainnya . Sedangkan untuk kedatangan pada *late night* cenderung lebih murah dibanding waktu lainnya.

Average Price by Arrival and Departure Time

	Departure Time					
Arrival Time	Afternoon	Early_Mor..	Evening	Late_Night	Morning	Night
Afternoon	₹ 14,742	₹ 16,565	₹ 25,570	₹ 18,521	₹ 13,271	₹ 28,115
Early_Morning	₹ 21,160	₹ 9,540	₹ 18,176	₹ 4,549	₹ 22,626	₹ 12,076
Evening	₹ 15,549	₹ 24,804	₹ 24,613	₹ 21,217	₹ 21,637	₹ 31,426
Late_Night	₹ 10,812	₹ 29,638	₹ 8,547	₹ 4,288	₹ 23,625	₹ 6,591
Morning	₹ 26,057	₹ 12,300	₹ 28,326	₹ 7,104	₹ 17,549	₹ 25,574
Night	₹ 19,098	₹ 25,792	₹ 15,642	₹ 29,440	₹ 26,828	₹ 17,818

3.3.2.9 Most Flight by City

Pada peta dibawah ini, ada 6 kota yang terdapat penerbangan. Delhi memiliki jumlah penerbangan tertinggi dengan 61.343 penerbangan serta kota Chennai memiliki jumlah penerbangan paling sedikit di antara kota-kota yang dianalisis, yaitu 38.700 penerbangan.

3.4 Penjelasan Hasil

Dari evaluasi model *Linear Regressor, Decision Tree Regressor, Random Forest Regressor* dan *Artificial Neural Network,* model yang memiliki hasil kerja paling baik adalah model *Linear Regressor*. Hal ini dibuktikan dengan evaluasi model sebagai berikut :

- R2 Score : 1.0
- MAE: 2.4606732085437902e-11
- MSE: 8.485735196453297e-22
- RMSE: 2.913028526543003e-11
- AIC: -2912587.203310449
- BIC: -2912470.1692975815

Bab 4 Penutup

4.1 Kesimpulan

Penelitian ini bertujuan untuk membandingkan kinerja model *Machine Learning* dengan *Deep Learning* dalam memprediksi harga tiket pesawat. Hasilnya menunjukkan bahwa model *Linear Regression* memiliki kinerja terbaik dibandingkan dengan *Random Forest, Decision Tree Regressor*, dan *Artificial Neural Network*. *Linear Regression* menunjukkan nilai R2 Score sempurna (1.0) dan kesalahan prediksi yang sangat kecil(MAE: 2.46e-11, MSE: 8.49e-22, RMSE: 2.91e-11), menunjukkan akurasi tinggi dalam memprediksi harga tiket. Analisis korelasi mengidentifikasi bahwa kelas penerbangan adalah faktor utama yang mempengaruhi harga tiket. Temuan ini dapat membantu calon penumpang membuat keputusan lebih baik saat memesan tiket dan dapat digunakan oleh platform pemesanan untuk meningkatkan layanan.

4.2 Saran

Disarankan untuk mengeksplorasi penggunaan model *Hybrid* yang menggabungkan *Machine Learning* dan *Deep Learning* untuk meningkatkan akurasi prediksi lebih lanjut. Selain itu, menambahkan fitur-fitur baru yang relevan, seperti data musiman, tren perjalanan, dan preferensi pengguna, dapat membantu meningkatkan kinerja model. Penelitian ini telah memberikan kontribusi signifikan dalam bidang prediksi harga tiket pesawat dan dapat menjadi dasar bagi penelitian serta pengembangan teknologi prediksi harga yang lebih canggih di masa mendatang.

Daftar Pustaka

[1] J. Ali, R. Khan, N. Ahmad and I. Maq, "Random Forests and Decision Trees," IJCSI International Journal of Computer Science Issues, vol. 9, no. 5, 12 May 2012.

[2] Budiharto, W. (2017). Machine Learning dan Komputasional Intelligence. Penerbit Andi

[3] Zebua YA, Sitompul DR, Sinurat SH, Situmorang A, Ruben R, Ziegel DJ, Indra E. Prediksi Penetapan Tarif Penerbangan Menggunakan Auto-Ml Dengan Algoritma Random Forest. Jurnal Tekinkom (Teknik Informasi dan Komputer). 2022 Jun 30;5(1):115-22.

[4] Yan, X., & Su, X. G. (2009). *Linear regression analysis*. London: World Scientific Publishing Co. Pte. Ltd.

[5] R. Primartha, Belajar machine learning teori dan praktek. Bandung: Informatika, 2018

[6] Noviando, Edo S., et al. "Studi Penerapan ANN (Artificial Neural Network) untuk Menghilangkan Harmonisa pada Gedung Pusat Komputer." *Jurnal Online Mahasiswa Fakultas Teknik Universitas Riau*, vol. 3, no. 2, Oct. 2016, pp. 1-6.

[7] Hodson, T. O. (2022). Root mean square error (RMSE) or mean absolute error (MAE): When to use them or not. *Geoscientific Model Development Discussions*, *2022*, 1-10.

[8] Handelman, G. S., Kok, H. K., Chandra, R. V., Razavi, A. H., Huang, S., Brooks, M., ... & Asadi, H. (2019). Peering into the black box of artificial intelligence: evaluation metrics of machine learning methods. *American Journal of Roentgenology*, *212*(1), 38-43.

[9] Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. *Geoscientific model development*, *7*(3), 1247-1250.

Lampiran 1: Dataset Flight Price Prediction (file_flight_predict.csv)

Dataset ini berisi informasi tentang opsi pemesanan penerbangan perjalanan antara 6 kota metro teratas di India, terdapat 300.153 data dan 11 kolom.

Deskripsi Dataset: Dataset ini terdiri dari beberapa kolom yang digunakan untuk memprediksi harga tiket pesawat, meliputi *Airline, Flight, Source City, Departure Time, Stops, Arrival Time, Destination City, Class, Duration, Days Left, dan Price*.

File: file_flight_predict.csv (link untuk file csv)

Lampiran 2: Kode Program Python untuk Pengolahan Data

Lampiran ini berisi kode program Python yang digunakan untuk melakukan eksplorasi data, pemrosesan data, pembagian data, pemodelan data, dan analisis hasil.

Deskripsi Kode: *Library* yang digunakan dalam pengerjaan kode program meliputi

1. *Pandas*, untuk memanipulasi dan analisis data tabular.
2. *Numpy*, untuk operasi matematika dan array multidimensi.
3. *Matplotlib.pyplot*, untuk membuat grafik dan plot data.
4. *Seaborn*, untuk visualisasi data statistik dengan antarmuka grafik informatif.
5. *Sklearn*, untuk pra-pemrosesan data, membangun model *Machine Learning* seperti *linear regression, decision tree, dan random forest*.
6. *Keras*, *framework* untuk membangun dan melatih model *neural network*.
7. Google.colab, untuk mengelola file dalam Google Colab.

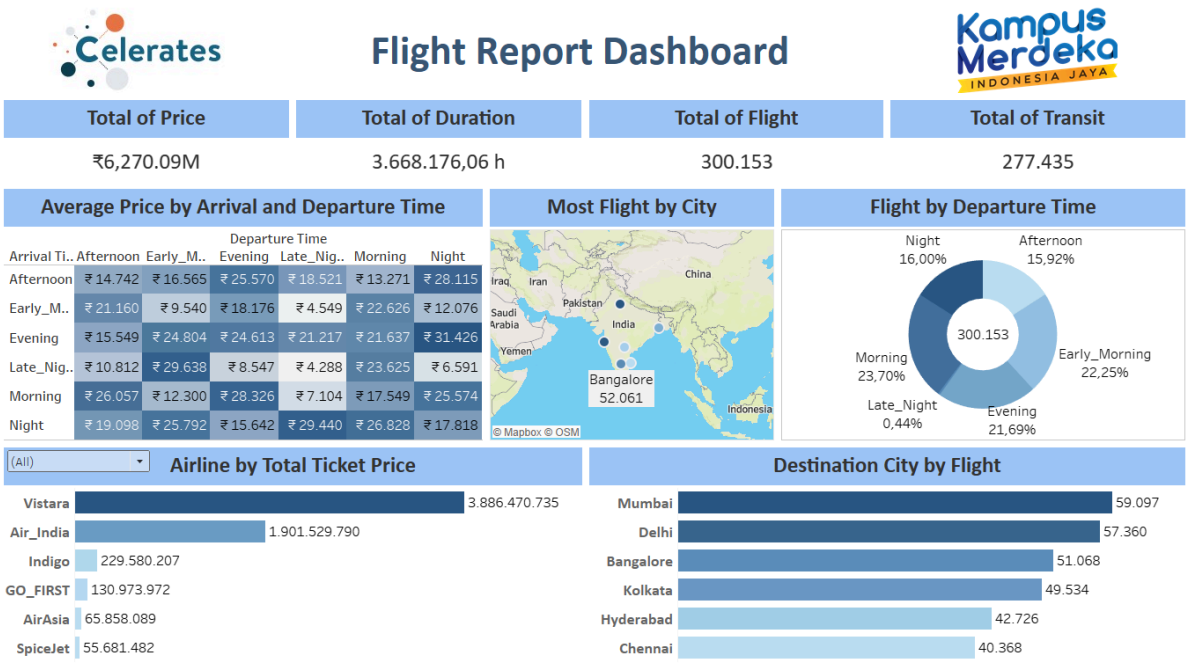
File: Final Project_Kelompok 2_Regression Dataset.ipynb (link untuk kode program)

Lampiran 3: Visualisasi Data menggunakan Tableau

Lampiran ini berisi visualisasi data yang dibuat menggunakan Tableau untuk mengeksplorasi tren dan pola dalam dataset prediksi penerbangan (file_flight_predict.csv)

Deskripsi Visualisasi: Visualisasi ini mencakup grafik batang horizontal, tabel persegi, bagan donat/pai, dan bagan simbol peta untuk memperlihatkan tren dan pola pada dataset.

File: Final Project_Kelompok 2.twbx (link untuk visualisasi data)



Lampiran 4: Timeline Pengerjaan Kelompok

Lampiran ini berisi timeline kami dalam menyelesaikan proyek ini.
File: GanChart_FinalProject.xlsx (link untuk timeline)