PREDIKSI HARGA TIKET PESAWAT MASKAPAI PENERBANGAN DI INDONESIA MENGGUNAKAN *TREE-BASED MODELS*

Kanaya Tabhita Djie Lina Cahyadi Josephine

Abstrak

Setiap harinya, data yang dihasilkan pada industri penerbangan Indonesia sangat besar. Data yang tersedia seperti jam keberangkatan, jam kedatangan, jumlah transit, hari penerbangan, dan sebagainya dapat digunakan untuk memprediksi harga tiket pesawat. Penelitian ini menggunakan tiga metode *machine learning* yaitu *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest* untuk mendapatkan hasil prediksi harga tiket pesawat pada rute Jakarta-Surabaya dan Jakarta-Manado. Untuk setiap model prediksi akan dilakukan *parameter tuning* untuk mendapatkan model yang paling optimal. Setelah didapatkan tiga model yang optimal, dilakukan evaluasi model untuk menentukan model terbaik untuk setiap rute. Evaluasi model dilakukan dengan membandingkan nilai *Mean Absolute Error*, *Mean Absolute Percentage Error*, *Root Mean Squared Error*, dan Koefisien Determinasi dari tiap model. Hasil penelitian menunjukkan bahwa model yang memiliki akurasi paling baik untuk kedua rute tersebut adalah model *Random Forest*.

1 Latar Belakang

Dalam kehidupan sehari-hari, manusia tidak akan terlepas dari data. Bahkan sejak lahir, manusia sudah menghasilkan data seperti nama, tanggal lahir, jam lahir, berat badan, dan sebagainya. Sagiroglu & Sinanc [1] menyatakan bahwa jumlah data yang telah dihasilkan manusia pada tahun 2003 adalah sebesar 5 *exabit* (10^{18} bit) sedangkan pada tahun 2013, jumlah data ini dapat dihasilkan hanya dalam waktu dua hari. Fenomena data yang melimpah ini dikenal sebagai *big data*, yaitu sekumpulan data terstruktur maupun tidak terstruktur yang volumenya sangat besar. Pengolahan *big data* akan sangat sulit jika menggunakan metode yang tradisional, sehingga dapat digunakan aplikasi dari *artificial intelligence* yaitu *machine learning* (ML).

Wang & Alexander [2] menjelaskan bahwa ML dapat dimanfaatkan untuk memperoleh informasi dan wawasan yang akan digunakan untuk membuat kepu-

tusan tertentu. Algoritma ML dapat diklasifikasikan menjadi dua kategori, yaitu *supervised learning* dan *unsupervised learning*. Beberapa contoh metode yang menggunakan algoritma *supervised learning* adalah *Naïve Bayes* dan *Support Vector Machine* (SVM). Berbeda dengan algoritma *supervised learning* yang menggunakan data yang sudah memiliki label, *unsupervised learning* menggunakan data yang tidak memiliki label dan melakukan klasifikasi dengan membandingkan fitur dari data tersebut. Metode *Clustering* adalah salah satu contoh metode algoritma *unsupervised learning*.

Menurut Clark & Pregibon [3], *tree-based models* adalah metode *supervised learning* yang digunakan untuk menemukan struktur di data. Metode ini menggunakan beberapa pernyataan bersyarat untuk mempartisi data menjadi beberapa himpunan bagian. Beberapa contoh dari *tree-based models* adalah *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest*. Prasad *et al.* [4] menjelaskan perbedaan antara *Regression Tree* dengan teknik regresi klasik. Berbeda dengan teknik regresi klasik di mana hubungan antara variabel prediktor dan variabel responssnya sudah ditentukan seperti linear atau kuadratik, *Regression Tree* tidak mengasumsikan hubungan apa pun. *Output* dari metode tersebut berupa diagram pohon dengan cabang-cabang yang ditentukan dari aturan *splitting* dan sejumlah *terminal nodes* yang merupakan rata-rata dari variabel responss. ML dapat dimanfaatkan di berbagai industri dan salah satunya adalah di industri penerbangan. Industri penerbangan di Indonesia menghasilkan data dengan volume yang sangat besar setiap harinya.

Fatimah [5] menjelaskan bahwa Indonesia adalah negara kepulauan yang mencakup lima pulau besar, ratusan pulau sedang, dan ribuan pulau kecil. Luas wilayah Indonesia yang sangat besar juga mengakibatkan diperlukannya sarana transportasi yang dapat memadai kebutuhan mobilisasi banyak orang dari satu tempat menuju ke tempat yang lainnya, sehingga tidak jarang orang yang memanfaatkan sarana transportasi udara untuk melakukan kegiatan bepergian. Oleh karena itu, perusahaan maskapai penerbangan tentunya menginginkan pendapatan yang maksimum sebagai jasa penyedia transportasi udara tersebut. Untuk mencapai target pendapatan tersebut, perusahaan menggunakan *revenue management* untuk menentukan harga tiket pesawat yang paling optimum. Menurut Li & Peng [6], dalam menentukan harga tiket penerbangan, perusahaan mempertimbangkan berbagai banyak hal seperti tingkat permintaan dan *inventory level*. Data ini tidak terbuka bagi umum dan sulit bagi calon penumpang untuk memprediksi harga tiket pesawat karena ketidaktersediaannya data tersebut.

Untuk mengatasi masalah ini, dapat dimanfaatkan data penerbangan yang terbuka untuk umum sebagai faktor untuk memprediksi harga pesawat. Data yang dapat digunakan untuk membangun model prediksi yaitu jam kedatangan, jam keberangkatan, jumlah transit, hari penerbangan, dan sebagainya. Algoritma *tree-based models* yaitu *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest* adalah metode yang dipilih untuk memprediksi harga tiket pesawat. Ketiga metode tersebut juga akan dibandingkan akurasinya untuk mendapatkan model yang terbaik. Rute penerbangan domestik di Indonesia yaitu Jakarta-Surabaya dan Jakarta-Manado adalah salah satu rute dengan jumlah penerbangan yang cukup banyak yaitu 20-30

penerbangan setiap harinya. Kedua rute ini akan digunakan untuk membuat model prediksi.

2 Landasan Teori

2.1 *Tree-Based Models*

Tree-based models adalah salah satu metode dari algoritma *supervised learning*. Metode ini dapat digunakan untuk masalah klasifikasi maupun regresi. *Tree-based models* untuk masalah klasifikasi dikenal sebagai *Decision Tree* atau pohon keputusan, sedangkan *tree-based models* untuk masalah regresi dikenal sebagai *Regression Tree* atau pohon regresi. Kedua metode ini adalah metode nonparametrik karena metode ini dapat digunakan dengan mengabaikan asumsi-asumsi yang biasanya harus dipenuhi jika menggunakan metode parametrik. James *et al.* [7] menjelaskan bahwa metode ini melibatkan stratifikasi atau segmentasi ruang prediktor ke beberapa daerah yang sederhana. Untuk membuat prediksi terhadap suatu observasi, biasanya akan digunakan rata-rata atau modus dari data latihan yang sudah ditempatkan ke setiap daerah yang bersesuaian. *Splitting rules* adalah aturan yang digunakan untuk melakukan segmentasi data dan bisa digambarkan dalam bentuk diagram pohon.

Beberapa istilah yang biasa digunakan di *tree-based models* adalah sebagai berikut.

1. *Splitting*
   Splitting adalah proses membagi suatu *node* menjadi dua atau lebih *sub-node*.

2. *Branch*
   Branch atau *sub-tree* adalah sub-bagian dari suatu model pohon.

3. *Parent Node* dan *Child Node*
   Node yang berada di bawah *node* lain disebut sebagai *child-node* atau *sub-node*, sedangkan *node* yang mendahului *node* lain disebut sebagai *parent node*.

4. *Depth of Tree*
   Depth of tree atau kedalaman model pohon diukur dari banyaknya *edge* dari *root node* ke suatu *node*. *Edge* adalah jalur yang menghubungkan dua *node*.

Struktur model pohon dimulai dari sebuah *node* yang biasa disebut *root node* dan kemudian akan bercabang (*splitting*) hingga memiliki banyak *leaf node*. Terdapat tiga jenis *node*, yaitu *root node*, *internal node*, dan *leaf node*. *Root node* adalah *node* yang berada di pangkal model pohon. *Root node* merepresentasikan seluruh populasi data yang akan dianalisis. *Root node* tidak memiliki *parent node*. *Internal node* atau *decision node* adalah *node* yang memiliki satu atau lebih *child node*. *Leaf node* atau *terminal node* adalah *node* yang tidak melakukan *splitting* sehingga tidak memiliki *child node*.

2.2 *Regression Tree*

Pada penelitian ini, variabel dependen yang digunakan bersifat kontinu, sehingga akan digunakan metode *Regression Tree* untuk merancang model prediksi harga tiket pesawat. Pada *Regression Tree* diasumsikan sebuah model yaitu

$$f(X) = \sum_{m=1}^{M} c_m \cdot 1_{(X \in R_m)} \tag{2.1}$$

atau dapat ditulis juga dalam bentuk

$$f(X) = c_m,\ X \in R_m \tag{2.2}$$

dengan $R_1, ..., R_M$ merepresentasikan partisi dari ruang fitur dan c_m adalah rata-rata dari semua observasi yang ada pada daerah R_m. Indeks m bergerak dari satu hingga M yang artinya jika data X elemen dari suatu partisi R_m, maka hasil estimasi untuk data X adalah rata-rata semua observasi yang ada pada daerah R_m tersebut.

James *et al.* [7] menjelaskan bahwa proses membangun model *Regression Tree* memiliki kurang lebih dua langkah yaitu sebagai berikut.

1. Langkah pertama adalah dengan membagi ruang prediktor, yaitu himpunan nilai untuk $X_1, X_2, ..., X_p$ menjadi sebanyak J daerah yang berbeda dan saling terpisah, $R_1, R_2, ..., R_J$.

2. Untuk setiap observasi yang masuk ke dalam daerah R_j, hasil prediksinya adalah rata-rata variabel respons dari data observasi latihan di daerah R_j.

Sebagai contoh, misalkan pada langkah pertama didapatkan dua daerah R_1 dan R_2, dan rata-rata variabel respons dari data latihan di R_1 adalah sebesar a, sedangkan rata-rata variabel respons dari data latihan di R_2 adalah b. Maka untuk suatu observasi $X = x$, jika $x \in R_1$ akan diprediksi nilai sebesar a, dan jika $x \in R_2$ akan diprediksi nilai sebesar b.

Sekarang akan diuraikan langkah pertama yaitu cara membangun daerah $R_1, ..., R_J$. Secara teori, daerah-daerah tersebut bisa berbentuk apa saja. Namun, untuk mempermudah interpretasi hasil model prediksi, kita akan membagi ruangan prediktor menjadi segi empat berdimensi tinggi atau kotak. Tujuan yang ingin dicapai adalah menemukan kotak-kotak $R_1, ..., R_J$ yang meminimumkan *residual sum of squares* (SSR) yaitu

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2, \tag{2.3}$$

di mana $\hat{y}_{R_j}$ adalah rata-rata variabel respons dari data latihan di kotak ke-j dan y_i adalah nilai aktual dari data ke-i. Nilai eror yaitu selisih antara nilai aktual y_i dan nilai prediksi $\hat{y}_{R_j}$ akan dikuadratkan. Persamaan (2.3) artinya setiap nilai eror kuadrat data ke-i pada kotak R_j akan dijumlahkan. Kemudian indeks j akan bergerak dari satu ke J yang berarti total nilai eror kuadrat tiap kotak $R_1, ..., R_J$ akan dijumlahkan.

Namun, secara komputasi tidak mungkin untuk mempertimbangkan setiap kemungkinan partisi ruang prediktor ke dalam J kotak. Untuk alasan ini, akan digunakan sebuah pendekatan *top-down, greedy* yaitu dengan melakukan *recursive binary splitting*. Pendekatan *top-down* yang berarti dari atas ke bawah karena dilakukan dari puncak pohon, di mana seluruh data observasi masih ada di dalam satu daerah. Setelah itu akan dilakukan *splitting* terhadap daerah prediktor. Setiap *split* ditandai dengan dua *branch* atau cabang yang ada di bawah. Pendekatan ini disebut *greedy* atau serakah karena pada setiap langkah proses pembangunan model pohon, *split* terbaik dibuat di langkah itu sendiri, daripada melihat ke depan dan memilih *split* yang akan menghasilkan model pohon yang lebih baik di beberapa langkah selanjutnya.

Untuk melakukan *recursive binary splitting*, pertama akan dipilih variabel prediktor X_j dan titik potong s sehingga ruang prediktor akan dipecah menjadi daerah $\{X|X_j < s\}$ dan $\{X|X_j \geq s\}$ yang menghasilkan pengurangan terbesar pada nilai RSS. (Notasi $\{X|X_j < s\}$ artinya daerah dari ruang prediktor di mana nilai X_j lebih kecil dari s, sedangkan notasi $\{X|X_j \geq s\}$ artinya daerah dari ruang prediktor di mana nilai X_j lebih besar sama dengan s). Ini artinya, seluruh variabel prediktor $X_1, ..., X_p$ dan setiap titik potong s dari setiap prediktor akan dipertimbangkan dan dipilih yang akan menghasilkan model pohon dengan nilai RSS terendah. Secara lebih rinci, untuk suatu j dan s, definisikan pasangan *half-plane* atau paruh bidang yaitu

$$R_{1(j,\,s)} = \{X|X_j < s\} \text{ dan } R_{2(j,\,s)} = \{X|X_j \geq s\} \tag{2.4}$$

dan akan dicari nilai j dan s yang meminimumkan persamaan

$$\sum_{i:\, x_i \in R_{1(j,\,s)}} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\, x_i \in R_{2(j,\,s)}} (y_i - \hat{y}_{R_2})^2 \tag{2.5}$$

di mana $\hat{y}_{R_1}$ adalah rata-rata variabel respons untuk data latihan di $R_{1(j,\,s)}$ dan $\hat{y}_{R_2}$ adalah rata-rata variabel respons untuk data latihan di $R_{2(j,\,s)}$. Total nilai eror kuadrat tiap data ke-i yang ada pada tiap ruang $R_{1(j,\,s)}$ dan $R_{2(j,\,s)}$ akan dijumlahkan.

Selanjutnya akan diulangi proses tersebut yaitu mencari variabel prediktor dan titik potong yang terbaik untuk membagi data lebih jauh sehingga nilai RSS dapat diminimumkan untuk setiap daerah. Namun kali ini yang dibagi bukan seluruh ruang prediktor, melainkan salah satu dari dua daerah yang sudah dibagi sebelumnya. Maka sekarang diperoleh tiga daerah. Lalu akan dibagi lagi salah satu dari ketiga daerah tersebut yang meminimumkan RSS. Proses ini akan terus berlanjut hingga suatu kriteria *stopping* sudah dicapai. Misalkan, *splitting* akan dilakukan hingga tidak ada daerah yang memiliki lebih dari lima observasi. Jika daerah-daerah $R_1, ..., R_J$ sudah dibuat, maka prediksi terhadap data tes dapat dilakukan menggunakan rata-rata data latihan di daerah di mana data tes itu berada.

2.3 Meningkatkan Akurasi *Regression Tree*

Untuk meningkatkan akurasi pada model pohon regresi, dapat dilakukan ketiga metode yaiu *pruning*, *bagging*, dan *random forest*.

James *et al.* [7] menjelaskan proses melakukan *pruning* pada *Regression Tree*. Proses pembuatan model *Regression Tree* dapat menghasilkan prediksi yang baik pada data latihan, tetapi akan cenderung terjadi *overfitting* sehingga hasil prediksi pada data tes akan menjadi buruk. Ini dikarenakan model pohon yang dihasilkan terlalu kompleks. Model pohon dengan jumlah *split* yang lebih sedikit (yang artinya lebih sedikit daerah $R_1, ..., R_J$) dapat menghasilkan varians yang lebih rendah dengan mengorbankan sedikit bias.

Strategi yang dapat dilakukan adalah dengan membangun *tree* T_0 yang sangat besar, lalu lakukan *pruning* atau pemangkasan agar mendapatkan sebuah *sub-tree*. Tujuan yang ingin dicapai adalah memilih sebuah *sub-tree* yang menghasilkan tingkat eror terendah. Untuk menentukan *sub-tree* yang terbaik, akan sangat sulit jika harus mempertimbangkan seluruh kemungkinan *sub-tree* karena jumlahnya akan sangat besar. Maka dari itu, dapat digunakan *cost complexity pruning* yang dikenal juga sebagai *weakest link pruning*, sehingga hanya perlu mempertimbangkan urutan *tree* yang diindeks dengan *nonnegative tuning parameter* α. *Tuning parameter* akan mengatur *trade-off* antara kompleksitas *tree* dengan akurasinya. Algoritma *pruning Regression Tree* adalah sebagai berikut.

1. Lakukan *recursive binary splitting* untuk membangun sebuah *tree* yang besar menggunakan data latihan, dan berhenti ketika setiap *terminal node* sudah tidak memenuhi syarat jumlah minimum observasi.

2. Terapkan *cost complexity pruning* ke *tree* yang besar tersebut untuk mendapatkan sebuah urutan dari *sub-tree* yang terbaik, sebagai fungsi dari α.

3. Lakukan *K-fold cross-validation* untuk memilih α, yaitu dengan membagi data latihan menjadi sebanyak *K fold*. Untuk setiap *fold* dari $h = 1, ..., K$ akan dilakukan dua langkah berikut.

   a. Ulangi langkah pertama dan kedua untuk semua *fold* kecuali *fold* ke-*h*.

   b. Evaluasi *mean squared prediction error* pada data di fold ke-*h*, sebagai fungsi dari α.

   Kemudian rata-ratakan nilai dari setiap α, dan pilih α yang meminimumkan eror rata-rata.

4. Pilih *sub-tree* dari langkah kedua sesuai dengan nilai yang dipilih dari α.

Untuk setiap nilai dari α, ada *sub-tree* sehingga $T \subset T_0$ sehingga

$$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T| \tag{2.6}$$

menjadi sekecil mungkin. $|T|$ adalah jumlah *terminal node* dari *tree* T, R_m adalah *subset* dari ruang prediktor yaitu *terminal node* ke-m, dan $\hat{y}_{R_m}$ adalah prediksi respons dari R_m, yaitu rata-rata dari data observasi di R_m. Setiap nilai eror kuadrat dan $\alpha|T|$ dari data ke-i yang ada pada ruang R_m akan dijumlahkan. Indeks m bergerak dari satu hingga $|T|$ yang artinya total nilai eror kuadrat dan $\alpha|T|$ pada tiap ruang $R_1, ..., R_{|T|}$ akan dijumlahkan. Jika $\alpha = 0$, maka *sub-tree* T sama dengan T_0, karena Persamaan (2.6) menjadi hanya menghitung *training error*. Namun seiring nilai α meningkat, ada nilai yang harus dibayarkan kalau memilih *tree* dengan banyak *terminal nodes*, sehingga nilai dari Persamaan (2.6) cenderung akan mengecil untuk *sub-tree* yang lebih kecil.

James *et al.* [7] menjelaskan konsep metode *Bagging Regression Tree*. Model *Regression Tree* cenderung memiliki varians yang tinggi. Ini artinya jika data latihan dibagi menjadi dua dan dibuat dua model *Regression Tree* dari kedua data tersebut, hasil model yang didapatkan bisa sangat berbeda. *Bootstrap aggregation* atau *bagging* adalah prosedur yang dilakukan untuk mengurangi varians dari sebuah model pohon.

Untuk mengurangi varians dan meningkatkan akurasi prediksi dari model *Regression Tree*, dapat dilakukan banyak pengambilan data latihan dari sebuah populasi lalu buat model prediksi untuk masing-masing data latihan. Setelah itu, rata-ratakan hasil prediksi yang didapatkan. Dalam kata lain, nilai $\hat{f}^1(x), \hat{f}^2(x), ..., \hat{f}^B(x)$ dapat dihitung menggunakan B set data latihan yang berbeda, dan dirata-ratakan untuk mendapatkan sebuah model *learning* dengan varians rendah yaitu,

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x). \tag{2.7}$$

Tentunya ini bukan cara yang praktis karena pada kenyataannya, sulit untuk mendapatkan banyak data latihan. Oleh karena itu, dapat dilakukan *bootstraping* yaitu dengan mengambil ulang sampel dari sebuah data latihan. Melalui pendekatan ini, akan dihasilkan B set data latihan *bootstrap* yang berbeda. Lalu lakukan *training* metode pada set data *bootstrap* ke-b untuk mendapatkan $\hat{f}^{*b}(x)$ dan rata-ratakan semua hasil prediksi sehingga akan diperoleh

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x). \tag{2.8}$$

Model *Bagging Regression Tree* biasa dibuat tanpa adanya *pruning* sehingga model pohonnya biasa besar. Oleh karena itu, masing-masing *tree* mempunyai varians yang besar, namun biasnya rendah. Oleh karena itu, merata-ratakan setiap *tree* akan mengurangi varians.

Seperti yang kita ketahui, *bagging* biasanya akan meningkatkan akurasi prediksi dibandingkan jika hanya menggunakan satu *tree*. Namun akan sulit untuk menginterpretasikan model tersebut karena jumlahnya yang banyak, sehingga variabel-variabel yang penting menjadi kurang jelas. Oleh karena itu, *bagging* meningkatkan akurasi prediksi dengan mengorbankan kemudahan dalam interpretasi.

Walaupun kumpulan *bagged trees* lebih sulit untuk diinterpretasi dibandingkan dengan satu *tree*, kita masih dapat memperoleh rangkuman secara keseluruhan mengenai tingkat kepentingan setiap variabel prediktor melalui RSS seperti pada Persamaan (2.3). Hal ini dapat dilakukan dengan mengurutkan jumlah total pengurangan RSS yang disebabkan oleh *splitting* dari setiap prediktor, kemudian dirata-ratakan pada sebanyak B pohon. Nilai yang besar menandakan bahwa prediktor tersebut penting.

James *et al.* [7] menjelaskan bahwa metode *Random Forest* bisa memberikan sedikit perbaikan dari *bagged trees* dengan mengurangi korelasi antar *tree*. *Bagging Regression Tree* membangun sejumlah *tree* dari data latihan *bootstrap* dan mempertimbangkan seluruh prediktor saat menentukan *split*. Namun pada *Random Forest*, sebanyak m prediktor dari seluruh p predikor akan disampel secara acak sebagai kandidat *split*. *Split* akan ditentukan hanya dari m prediktor tersebut. Sampel m prediktor baru akan diambil lagi pada setiap *split*. Nilai m yang digunakan biasanya adalah $m \approx \sqrt{p}$. Dalam kata lain, jumlah prediktor yang dipertimbangkan pada setiap *split* sekitar akar dari jumlah total prediktor.

Ada alasan penting dibalik metode sampel acak prediktor tersebut. Misalkan ada satu prediktor yang sangat kuat dari sebuah set data dengan beberapa prediktor lainnya yang cukup kuat. Maka pada kumpulan *bagged trees*, sebagian besar atau semua *tree* akan menggunakan prediktor yang sangat kuat tersebut sebagai *split* yang paling atas. Akibatnya, semua *bagged trees* akan saling menyerupai dan hasil prediksinya akan saling berkorelasi. Merata-ratakan *tree* yang saling berkorelasi tidak akan mengurangi varians sebanyak *tree* yang saling tidak berkorelasi. Jika model *Random Forest* dibangun menggunakan $m = p$ prediktor, maka metode ini sama saja dengan *Bagging Regression Tree*.

Hyperparameter optimization juga dapat dilakukan untuk meningkatkan akurasi model berbasis pohon. Elshawi *et al.* [8] menjelaskan bahwa secara umum, membangun model ML yang efektif adalah proses yang kompleks dan memakan banyak waktu. Proses ini melibatkan penentuan algoritma yang sesuai dan mendapatkan arsitektur model yang optimal dengan melakukan *hyperparameter* (HP) *tuning*. Menurut Kuhn & Johnson [9], terdapat dua jenis parameter, yaitu parameter model yang dapat diinisialisasi dan diperbarui melalui proses pembelajaran data dan *hyperparameter* yang tidak dapat diestimasi langsung saat proses pembelajaran data dan harus ditentukan sebelum melakukan *training* model ML karena parameter ini mendefinisikan arsitektur model.

Hutter *et al.* [10] menjelaskan bahwa untuk membangun model ML yang optimal, ada banyak kemungkinan yang harus dieksplorasi. Proses merancang arsitektur model yang ideal dengan konfigurasi HP yang optimal disebut dengan *hyperparameter tuning*. Proses optimisasi ini dianggap sebagai kunci membangun model ML yang efektif, terutama untuk *tree-based models* dan *deep neural networks*, yang memiliki banyak HP. Sanders & Giraud-Carrier [11] menyatakan bahwa pada model berbasis pohon, proses optimisasi dapat dilakukan pada banyak HP seperti maksimum tingkat kedalaman model pohon, maksimum jumlah fitur, minimum jumlah data yang diperlukan untuk melakukan *splitting*, minimum jumlah data dalam se-

buah *terminal node*, maksimum jumlah *terminal node*, dan lain-lain.

3 Metode Penelitian

3.1 Pengumpulan Data

Data yang akan digunakan dalam penelitian ini diperoleh dari sebuah situs *web* penyedia jasa pemesanan tiket pesawat di Indonesia yaitu *tiket.com*. Data penerbangan yang akan diamati adalah data yang berada di antara 9 Juni 2022 hingga 31 Agustus 2022. Rute yang digunakan adalah dua domestik yaitu Jakarta-Surabaya (CGK-SUB) dan Jakarta-Manado (CGK-MDC). Data tersebut akan dikumpulkan setiap harinya menggunakan *browser extension* yaitu *Web Scraper*.

Data yang telah diperoleh akan dibersihkan sesuai dengan struktur yang sudah ditentukan. Proses *scraping* terkadang menghasilkan *missing values*. Data yang memiliki *missing values* tidak akan diikutsertakan dalam penelitian ini.

Variabel dependen pada penelitian ini adalah harga tiket pesawat yaitu variabel '*price*'. Variabel independen pada penelitian ini adalah variabel '*airline*', '*departure*', '*arrival*', '*transit*', '*day_of_week*', '*holiday*', '*days_left_until_departure*', '*duration*', '*scrape_day*', dan '*scrape_holiday*'.

3.2 Analisis Data

Pada data yang sudah dikumpulkan pada langkah sebelumnya, akan dilakukan beberapa analisis sederhana. Analisis data yang akan dilakukan adalah dengan memvisualisasikan harga tiket pesawat berdasarkan setiap variabel prediktor. Visualisasi data berupa grafik garis dan dibuat menggunakan tabel *pivot*.

3.3 Pembagian Data

Data yang telah dikumpulkan akan dibagi menjadi dua, yaitu data latihan dan data tes. Proporsi pembagian data adalah 80% untuk data latihan dan 20% untuk data tes. Pemilihan data akan dilakukan secara acak. Perancangan model akan menggunakan data latihan dan diuji menggunakan data tes.

3.4 Model Prediksi dengan *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest*

Data latihan akan digunakan untuk membangun tiga model prediksi menggunakan *tree-based models*, yaitu *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest*. Model *Regression Tree* akan dibangun menggunakan *package rpart*. Model *Bagging Regression Tree* dan *Random Forest* akan dibangun menggunakan *package caret*. Model prediksi tersebut akan menghasilkan *output* numerik yaitu

harga tiket pesawat. Model yang akan digunakan adalah model yang paling optimum. Nilai parameter yang digunakan untuk mendapatkan model yang paling optimum akan dicari menggunakan *k-fold cross-validation*.

3.5 Memprediksi Data

Jika model *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest* yang paling optimum sudah didapatkan, maka ketiga model tersebut sudah siap digunakan untuk melakukan prediksi terhadap data tes yaitu 20% dari keseluruhan data, sehingga data prediksi adalah data penerbangan selama tiga bulan dari 9 Juni 2022 hingga 31 Agustus 2022 dengan rute penerbangan Jakarta-Surabaya (CGK-SUB) dan Jakarta-Manado (CGK-MDC). Prediksi dilakukan menggunakan variabel-variabel seperti jam kedatangan, jam keberangkatan, dan sebagainya. Setiap model akan menghasilkan sebuah estimasi harga tiket pesawat untuk tiap data penerbangan. Harga estimasi tiket pesawat ini nantinya dapat dibandingkan dengan harga aslinya sebagai evaluasi model.

3.6 Evaluasi Model

Berdasarkan hasil estimasi yang didapatkan, setiap model akan dievaluasi performanya dalam memprediksi harga tiket pesawat. Evaluasi model akan dilakukan dengan menghitung besar eror antara data aktual dengan data prediksi menggunakan MAE, RMSE, dan MAPE. Performa model juga diamati menggunakan *goodness-of-fit* yaitu koefisien determinasi dari setiap model. Berdasarkan dari hasil evaluasi model, akan ditentukan model yang memiliki performa terbaik dalam memprediksi harga tiket pesawat, yaitu yang memiliki nilai MAE, RMSE, dan MAPE terkecil. Koefisien determinasi untuk model yang baik adalah yang nilainya besar.

3.7 *Web App* dengan *R Shiny*

Model prediksi harga tiket pesawat akan digunakan untuk membuat sebuah *web app* menggunakan *R Shiny*. Aplikasi ini akan menerima *input* seperti jenis maskapai, jam keberangkatan, jumlah transit, hari keberangkatan, dan sebagainya dan kemudian akan melakukan prediksi harga tiket pesawat menggunakan ketiga model yaitu *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest* berdasarkan data yang diterima.

4 Analisis dan Pembahasan

4.1 Data

Data yang telah dikumpulkan menggunakan *Web Scraper* akan dibagi menjadi dua sesuai dengan rutenya yaitu Jakarta-Surabaya dan Jakarta-Manado. Kedua

data tersebut akan diolah menggunakan *pivot table* untuk analisis data kemudian diolah menggunakan *R* untuk pembuatan modelnya.

Pada data penerbangan yang telah dikumpulkan, terdapat *missing values* pada tiap rute. Data dengan *missing values* tidak diikutsertakan dalam penelitian ini. Penjelasan jumlah data untuk tiap rute adalah sebagai berikut.

1. Rute Jakarta-Surabaya memiliki total 103.286 data termasuk 1.889 data dengan *missing values*, sehingga total data bersih adalah 101.397.

2. Rute Jakarta-Manado memiliki total 63.146 data termasuk 8.649 data dengan *missing values*, sehingga total data bersih adalah 54.497.

4.2 Analisis Data

Analisis data akan dilakukan menggunakan *pivot table* di *Microsoft Excel*. Harga tiket pesawat (variabel respons) akan divisualisasikan dengan grafik garis berdasarkan 10 faktor (variabel prediktor) yaitu jenis maskapai penerbangan, jam keberangkatan, jam kedatangan, jumlah transit, hari penerbangan, hari libur, jumlah hari tersisa sebelum keberangkatan, durasi penerbangan, hari data penerbangan didapatkan, dan hari libur data penerbangan didapatkan.

Hasil analisis harga tiket pesawat pada rute Jakarta-Surabaya terhadap setiap variabel prediktor adalah sebagai berikut.

Tabel 4.1: Hasil Analisis Harga Tiket Pesawat terhadap Variabel Prediktor pada Rute Jakarta-Surabaya

Variabel Prediktor	Hasil Analisis
Jenis Maskapai Penerbangan	Maskapai penerbangan dengan rata-rata harga tiket tertinggi adalah Nam Air, sedangkan maskapai penerbangan dengan rata-rata harga tiket terendah adalah Super Air Jet.
Jam Keberangkatan	Puncak harga tiket pesawat untuk setiap maskapai penerbangan ada di jam keberangkatan yang berbeda-beda sehingga tidak dapat disimpulkan jam keberangkatan dengan harga paling mahal untuk rute Jakarta-Surabaya.
Jam Kedatangan	Puncak harga tiket pesawat untuk setiap maskapai penerbangan ada di jam kedatangan yang berbeda-beda sehingga tidak dapat disimpulkan jam kedatangan dengan harga paling mahal untuk rute Jakarta-Surabaya.
Jumlah Transit	Harga tiket penerbangan dengan satu transit lebih mahal dari harga tiket penerbangan langsung.
Hari Penerbangan	Harga tiket pesawat cenderung lebih mahal di hari Jumat.
Hari Libur	Harga tiket pesawat cenderung lebih mahal di hari libur.
Jumlah Hari Tersisa Sebelum Keberangkatan	Harga tiket pesawat pada seluruh maskapai penerbangan di rute Jakarta-Surabaya akan semakin naik jika berdekatan dengan hari keberangkatan.
Durasi Penerbangan	Pada sebagian besar dari maskapai penerbangan pada rute ini, rata-rata harga tiket pesawat akan semakin naik jika durasinya semakin lama.
Hari Data Penerbangan Didapatkan	Hari data penerbangan didapatkan adalah hari di mana calon penumpang akan melakukan pembelian tiket pesawat. Harga tiket pesawat cenderung lebih murah dibeli di hari Kamis.
Hari Libur Data Penerbangan Didapatkan	Hari libur data penerbangan didapatkan ditentukan dari hari di mana calon penumpang akan melakukan pembelian tiket pesawat. Harga tiket pesawat cenderung lebih mahal jika dibeli di hari libur.

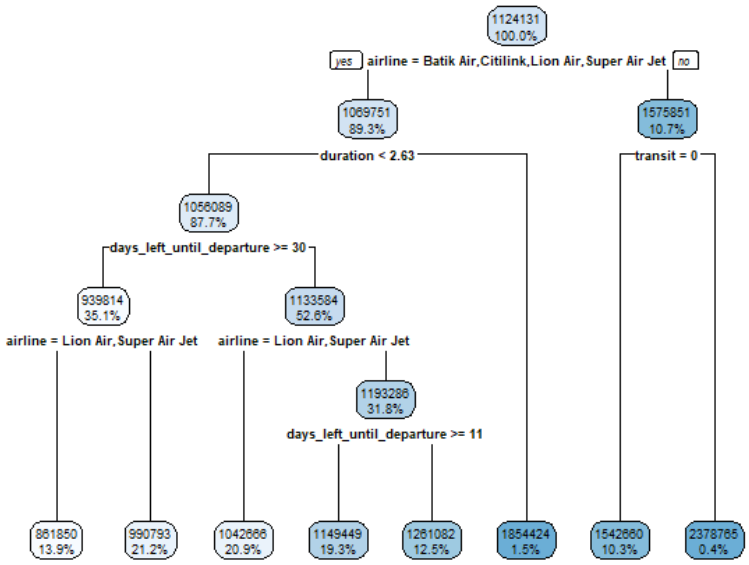
Hasil analisis harga tiket pesawat pada rute Jakarta-Manado terhadap setiap variabel prediktor adalah sebagai berikut.

Tabel 4.2: Hasil Analisis Harga Tiket Pesawat terhadap Variabel Prediktor pada Rute Jakarta-Manado

Variabel Prediktor	Hasil Analisis
Jenis Maskapai Penerbangan	Maskapai penerbangan dengan rata-rata harga tiket tertinggi adalah Batik Air, sedangkan askapai penerbangan dengan rata-rata harga tiket terendah adalah Citilink.
Jam Keberangkatan	Puncak harga tiket pesawat untuk setiap maskapai penerbangan ada di jam keberangkatan yang berbeda-beda sehingga tidak dapat disimpulkan jam keberangkatan dengan harga paling mahal untuk rute Jakarta-Manado.
Jam Kedatangan	Harga tiket pesawat cenderung mahal pada penerbangan dengan jam kedatangan 13.00-15.59.
Jumlah Transit	Semakin tinggi jumlah transit, harga tiket pesawat pada rute ini akan semakin tinggi juga.
Hari Penerbangan	Harga tiket pesawat cenderung lebih mahal di hari Sabtu.
Hari Libur	Harga tiket pesawat cenderung lebih mahal di hari bukan libur.
Jumlah Hari Tersisa Sebelum Keberangkatan	Harga tiket pesawat pada seluruh maskapai penerbangan di rute Jakarta-Surabaya akan semakin naik jika berdekatan dengan hari keberangkatan.
Durasi Penerbangan	Pada sebagian besar dari maskapai penerbangan pada rute ini, rata-rata harga tiket pesawat akan semakin naik jika durasinya semakin lama.
Hari Data Penerbangan Didapatkan	Harga tiket pesawat cenderung lebih murah dibeli di hari Kamis.
Hari Libur Data Penerbangan Didapatkan	Harga tiket pesawat cenderung lebih mahal jika dibeli di hari libur.

4.3 Model Prediksi untuk Rute Jakarta-Surabaya

Model *Regression Tree* paling optimal yaitu dengan parameter `minsplit=20` dan `maxdepth=14` untuk rute Jakarta-Surabaya ditunjukkan pada Gambar 4.1. Model ini hanya menggunakan empat dari sepuluh variabel prediktor yaitu `airline`, `duration`, `transit`, dan `days_left_until_departure`. Model ini dimulai dengan 81.117 observasi pada *root node*. Variabel yang menjadi titik *splitting* pertama adalah `airline`, sehingga variabel paling penting yang memberikan pengurangan terbanyak di SSE pada awalnya adalah `airline`. Pada *node* pertama, sebanyak 72.401 observasi dengan `airline=Batik Air, Citilink,Lion Air, Super Air Jet` dibagi ke *node* kedua. Rata-rata harga tiket pesawat pada *node* ini adalah Rp1.069.751,00. Sisanya sebanyak 8.716 observasi dengan `airline=Garuda Indonesia,NAM Air` dibagi ke *node* ketiga. Rata-rata harga tiket pesawat pada *node* ini adalah Rp1.575.851,00. Model pohon ini memiliki 6 *internal nodes* dan 8 *terminal nodes*, sehingga $|T| = 8$.



Gambar 4.1: *Plot* Model *Regression Tree* Awal untuk Rute Jakarta-Surabaya

Pada *Bagging Regression Tree*, model akan dibuat menggunakan beberapa sampel *bootstrap*, sehingga akan dibuat beberapa model pohon. Parameter yang mengatur jumlah model pohon adalah `nbagg`. Model *Bagging Regression Tree* paling optimal yaitu dengan parameter `nbagg=23` adalah sebagai berikut.

```
> bagged_ml
Bagged CART

81117 samples
   10 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 73006, 73006, 73005, 73004, 73005, 73005, ...
Resampling results:

  RMSE      Rsquared  MAE
  161858.6  0.657228  131455.8
```

Selain itu didapatkan juga tingkah *variabel importance* pada model *Bagging Regression Tree* ini. Variabel `duration` adalah prediktor yang memberikan dampak terbesar terhadap SSE sehingga menjadi variabel terpenting. Selain variabel `duration`, variabel `days_left_until_departure` dan `arrival` juga memberikan dampak yang cukup besar pada SSE.

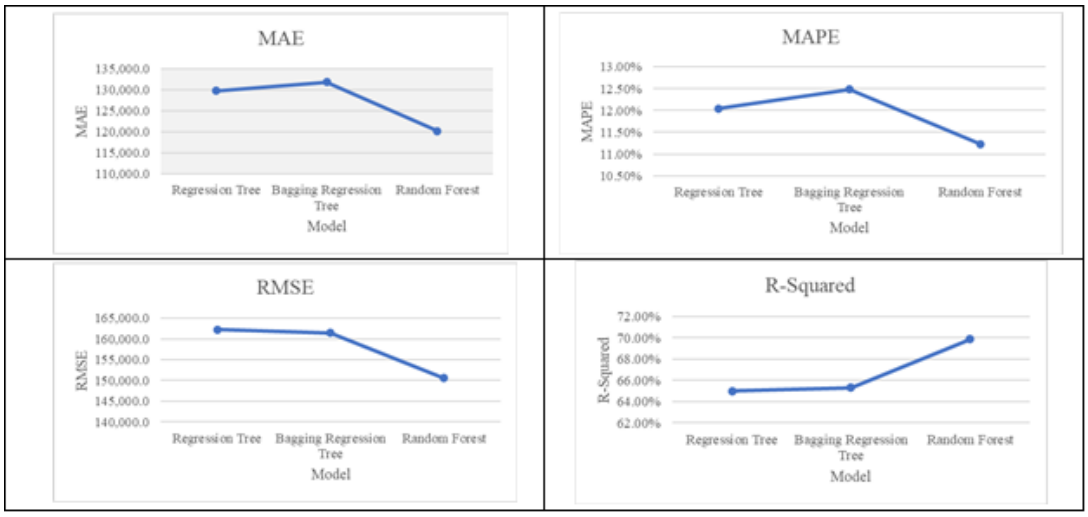
Model *Random Forest* paling optimal yaitu dengan parameter `mtry=18`, `ntree=1000`, dan `maxnodes=50` adalah sebagai berikut.

```
> rf_opt
Random Forest

81117 samples
   10 predictor

No pre-processing
Resampling: Cross-validated (5 fold)
Summary of sample sizes: 64894, 64892, 64894, 64894, 64892
Resampling results:

  RMSE      Rsquared    MAE
  150309    0.7043673   119775.9

Tuning parameter 'mtry' was held constant at a value of 18
```

Performa tiap model akan dibandingkan melalui kemampuan tiap model dalam melakukan prediksi. Nilai *testing error* setiap model ditunjukkan pada tabel 4.3 dan divisualisasikan pada Gambar 4.2.

Tabel 4.3: Performa Model untuk Rute Jakarta-Surabaya

Model	MAE	MAPE	RMSE	R^2
Regression Tree	129.764,2	12,04%	162.198,0	64,99%
Bagging Regression Tree	130.626,7	12,36%	160.892,8	65,56%
Random Forest	120.257,0	11,23%	150.536,2	69,87%



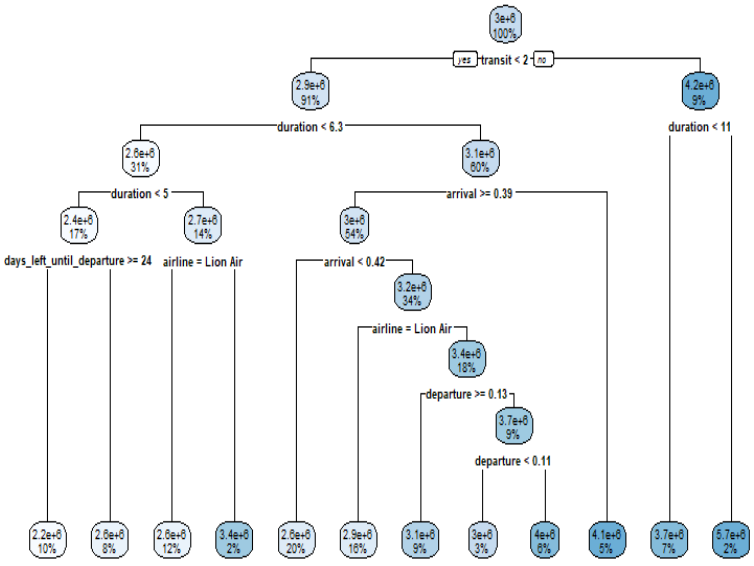
Gambar 4.2: Visualisasi Performa Model untuk Rute Jakarta-Surabaya

Model *Random Forest* memiliki nilai MAE, MAPE, dan RMSE terendah. Model *Random Forest* juga memiliki nilai R^2 tertinggi, sehingga model terbaik untuk rute Jakarta-Surabaya adalah model *Random Forest*. Model *Regression Tree* dan

Bagging Regression Tree tidak berbeda secara signifikan, namun model *Regression Tree* memiliki nilai MAE dan MAPE yang lebih baik, sedangkan model *Bagging Regression Tree* memiliki nilai RMSE dan R^2 yang lebih baik. RMSE memberikan penalti yang lebih besar pada nilai eror yang besar, sehingga model *Regression Tree* cenderung menghasilkan nilai eror yang besar dibandingkan dengan model *Bagging Regression Tree*.

4.4 Model Prediksi untuk Rute Jakarta-Manado

Model *Regression Tree* paling optimal yaitu dengan parameter `minsplit=17` dan `maxdepth=13` ditunjukkan pada Gambar 4.3. Model ini hanya menggunakan enam dari sepuluh variabel prediktor yaitu `transit`, `duration`, `arrival`, `days_left_until_departure`, `airline`, dan `departure`. Model ini dimulai dengan 43.597 observasi pada *root node*. Variabel yang menjadi titik *splitting* pertama adalah `transit`, sehingga variabel paling penting yang memberikan pengurangan terbanyak di SSE pada awalnya adalah `transit`. Pada *node* pertama, sebanyak 39.663 observasi dengan `transit<1.5` dibagi ke *node* kedua. Rata-rata harga tiket pesawat pada *node* ini adalah Rp2.893.626,00. Sisanya sebanyak 3934 observasi dengan `transit>=1.5` dibagi ke *node* ketiga. Rata-rata harga tiket pesawat pada *node* ini adalah Rp4.205.130,00. Model pohon ini memiliki 10 *internal nodes* dan 12 *terminal nodes*, sehingga $|T| = 12$.



Gambar 4.3: *Plot* Model *Regression Tree* Awal untuk Rute Jakarta-Manado

Pada *Bagging Regression Tree*, model akan dibuat menggunakan beberapa sampel *bootstrap*, sehingga akan dibuat beberapa model pohon. Parameter yang

mengatur jumlah model pohon adalah `nbagg`. Model *Bagging Regression Tree* paling optimal yaitu dengan parameter `nbagg=25` adalah sebagai berikut.

```
> bagged_ml
Bagged CART

43597 samples
   10 predictor

No pre-processing
Resampling: Cross-validated (10 fold)
Summary of sample sizes: 39239, 39238, 39237, 39237, 39237, 39237, ...
Resampling results:

  RMSE      Rsquared    MAE
  317942.2  0.8148509   244818.4
```

Selain itu didapatkan juga tingkah variabel importance pada model *Bagging Regression Tree* ini. Variabel `duration` adalah prediktor yang memberikan dampak terbesar terhadap SSE sehingga menjadi variabel terpenting. Selain variabel `duration`, variabel `departure` dan `arrival` juga memberikan dampak yang cukup besar pada SSE.

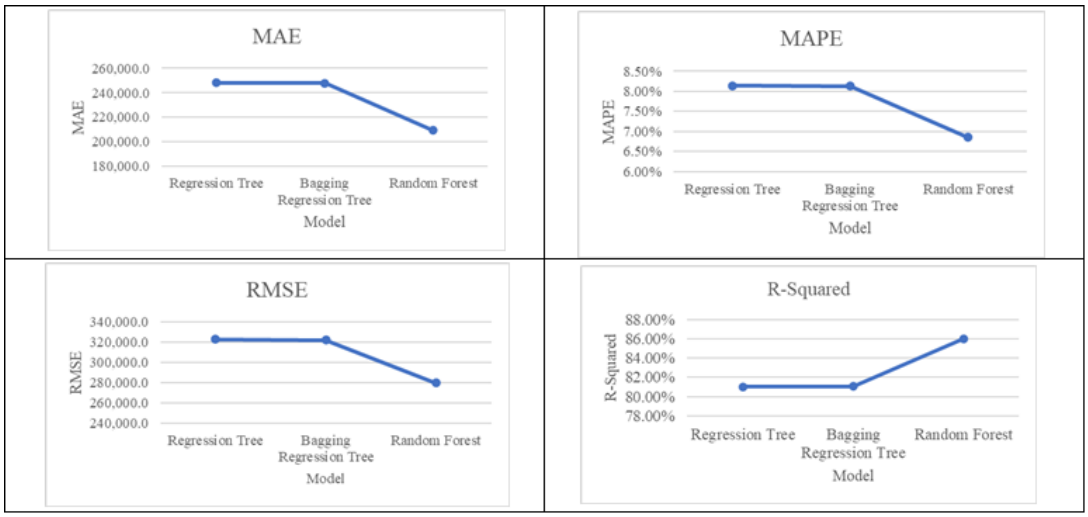
Model Random Forest paling optimal yaitu dengan parameter `mtry=11`, `ntree=1000`, dan `maxnodes=50` adalah sebagai berikut.

```
> rf_opt
Random Forest

43357 samples
   10 predictor

No pre-processing
Resampling: Cross-validated (5 fold)
Summary of sample sizes: 34880, 34876, 34878, 34877, 34877
Resampling results:

  RMSE      Rsquared    MAE
  278020.2  0.8625932   207933.6

Tuning parameter 'mtry' was held constant at a value of 11
```

Performa tiap model akan dibandingkan melalui kemampuan tiap model dalam melakukan prediksi. Nilai *testing error* setiap model ditunjukkan pada tabel 4.4 dan divisualisasikan pada Gambar 4.4.

Tabel 4.4: Performa Model untuk Rute Jakarta-Manado

Model	MAE	MAPE	RMSE	R^2
Regression Tree	248.163,6	8,14%	322.838,7	81,01%
Bagging Regression Tree	247.941,0	8,13%	322.254,0	81,08%
Random Forest	209.333,0	6,85%	279,537.7	86,04%



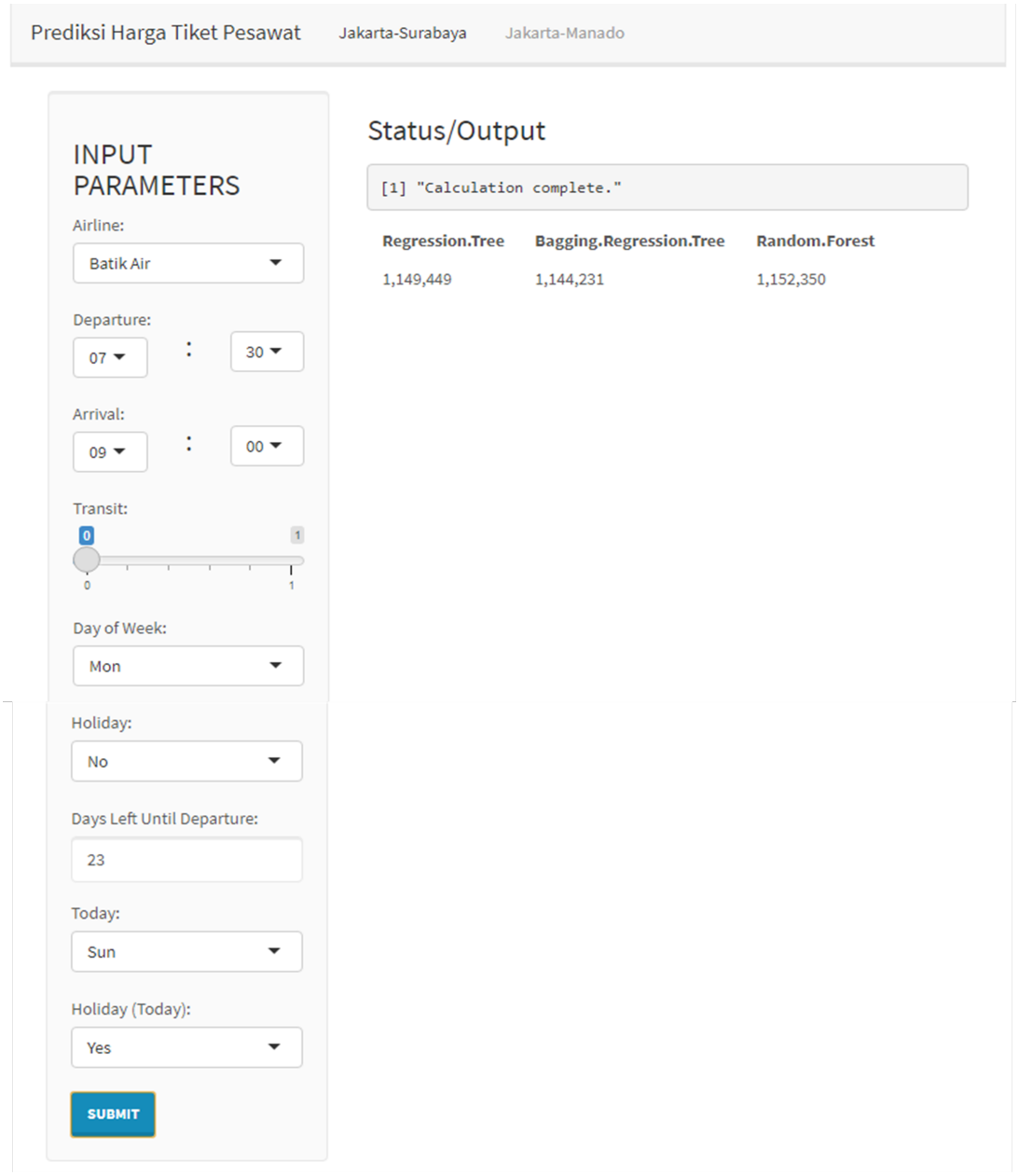
Gambar 4.4: Visualisasi Performa Model untuk Rute Jakarta-Manado

Model *Random Forest* memiliki nilai MAE, MAPE, dan RMSE terendah. Model *Random Forest* juga memiliki nilai R^2 tertinggi, sehingga model terbaik untuk rute Jakarta-Manado adalah model *Random Forest*. Model *Regression Tree* dan *Bagging Regression Tree* tidak berbeda secara signifikan, namun model *Regression Tree* memiliki nilai MAE, MAPE, RMSE, dan R^2 yang lebih baik.

4.5 *Web App* Prediksi Harga Tiket Pesawat Menggunakan *R Shiny*

Ketiga model dari setiap rute akan digunakan untuk membangun sebuah *web app* "Prediksi Harga Tiket Pesawat" menggunakan *R Shiny*. Aplikasi ini memiliki sebuah *navigation bar* yang dapat digunakan untuk berpindah halaman antara halaman "Jakarta-Surabaya" dan halaman "Jakarta-Manado".

Pada setiap halaman, terdapat *side bar panel* dan *main panel*. Pada *side bar panel* disediakan tempat untuk melakukan *input* parameter seperti jenis maskapai, jam keberangkatan, jam kedatangan, transit, dan sebagainya. Pada *main panel* ditampilkan status dari aplikasi, yaitu *"Server is ready for calculation."* jika model siap digunakan dan *"Calculation complete."* jika hasil prediksi sudah didapatkan. Pada *main panel* juga ditampilkan hasil prediksi harga tiket pesawat menggunakan ketiga model yaitu *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest*.



Gambar 4.5: *Web App* Prediksi Harga Tiket Pesawat

5 Kesimpulan

Prediksi harga tiket pesawat maskapai penerbangan di Indonesia pada rute Jakarta-Surabaya dan Jakarta-Manado berhasil dilakukan menggunakan ketiga model ML yang telah ditentukan yaitu *Regression Tree*, *Bagging Regression Tree*, dan *Random Forest*. Evaluasi terhadap model yang telah didapatkan tersebut dilakukan dengan membandingkan nilai MAE, MAPE, RMSE, dan R^2. Hasil prediksi harga tiket pesawat dari setiap model prediksi untuk kedua rute yang telah diperoleh

cukup akurat dengan nilai MAPE di bawah 20%. Model *Random Forest* memiliki tingkat kesalahan prediksi terkecil. Oleh karena itu, model prediksi terbaik untuk kedua rute tersebut adalah *Random Forest*.

Pustaka

[1] Seref Sagiroglu and Duygu Sinanc. Big data: A review. In *2013 international conference on collaboration technologies and systems (CTS)*, pages 42–47. IEEE, 2013.

[2] Lidong Wang and Cheryl Ann Alexander. Machine learning in big data. *International Journal of Mathematical, Engineering and Management Sciences*, 1(2):52–61, 2016.

[3] Linda A Clark and Daryl Pregibon. Tree-based models. In *Statistical models in S*, pages 377–419. Routledge, 2017.

[4] Anantha M Prasad, Louis R Iverson, and Andy Liaw. Newer classification and regression rree techniques: Bagging and random forests for ecological prediction. *Ecosystems*, 9(2):181–199, 2006.

[5] Siti Fatimah. *Pengantar Transportasi*. Myria Publisher, 2019.

[6] Luo Li and Ji-Hua Peng. Dynamic pricing model for airline revenue management under competition. *Systems Engineering-Theory & Practice*, 27(11):15–25, 2007.

[7] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*, volume 112. Springer, 2013.

[8] Radwa Elshawi, Mohamed Maher, and Sherif Sakr. Automated machine learning: State-of-the-art and open challenges. *arXiv preprint arXiv:1906.02287*, 2019.

[9] Max Kuhn, Kjell Johnson, et al. *Applied predictive modeling*, volume 26. Springer, 2013.

[10] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

[11] Samantha Sanders and Christophe Giraud-Carrier. Informing the use of hyperparameter optimization through metalearning. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 1051–1056. IEEE, 2017.