BAB II
TINJAUAN PUSTAKA

Penelitian pada skripsi ini diperkaya dengan adanya tinjauan pustaka sebagai pedoman peneliti terhadap permasalahan yang diangkat. Tinjauan pustaka dilakukan dengan membaca jurnal baik nasional maupun internasional yang bereputasi baik, buku, dan artikel pada halaman web. Pustaka yang digunakan yaitu penelitian terdahulu dan teori-teori mengenai kasus pada penelitian ini.

2.1. Pencapaian Terdahulu

Referensi penelitian terdahulu digunakan sebagai studi literatur, perbandingan, dan acuan pembaharuan kasus pada penelitian ini. Penelitian tentang sistem prediksi harga batubara menggunakan *machine learning* pada kasus pemodelan pergerakan harga batubara Newcastle berjangka belum pernah dilakukan sebelumnya. Dengan demikian, referensi yang digunakan berupa topik tentang penelitian prediksi harga batubara maupun penelitian penggunaan algoritma *machine learning* yang relevan.

Referensi penelitian yang pertama berjudul "Prediksi Harga Batubara Menggunakan *Support Vector Regression* (SVR)" dengan menggunakan data Harga Batubara Acuan (HBA) pada Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer (Bonita et al., 2018, pp. 6603–6609). Pada penelitian tersebut digunakan dua kernel pada algoritma *Support Vector Regression* dan nilai prediksi terbaik dihasilkan saat menggunakan kernel ANOVA dengan parameter optimal sebesar 9.64% sedangkan kernel Gaussian RBF hanya menghasilkan 8.38%.

Referensi lainnya berjudul "*Analysis of Decision Tree and K-Nearest Neighbors Algorithm in Classification of Breast Cancer*" yang membahas tentang dua tingkat penyakit kanker payudara pada *Asian Pacific Journal of Cancer Prevention* (Rajaguru & Sannasi Chakravarthy, 2019, pp. 3777–3781). Penelitian tersebut menghasilkan data bahwa algoritma *K-Nearest Neighbors* mengeluarkan nilai akurasi sebesar 95.61% dan *Decision Tree* sebesar 91.23%. Dengan demikian pada penelitian tersebut dapat ditarik kesimpulan bahwa klasifikasi antara dua tingkat penyakit kanker payudara diprediksi dengan baik saat menggunakan algoritma *K-Nearest Neighbors.*

Penelitian lain yang berjudul "*Heart Disease Prediction Using Machine Learning Techniques: a Survey*" pada *International Journal of Engineering & Technology* bertujuan untuk mengetahui algoritma *machine learning* yang memiliki tingkat akurasi terbaik dalam memprediksi penyakit jantung (Ramalingam et al., 2018, pp. 684–687). Hasil penelitian tersebut yaitu algoritma *K-Nearest Neighbors* memiliki tingkat akurasi sebesar 87.5% sedangkan *Decision Tree* sebesar 78.46%. Sehingga dari kedua algoritma tersebut didapatkan kesimpulan bahwa algoritma *K-Nearest Neighbors* memperoleh hasil yang lebih baik pada topik penelitian tersebut.

Pada jurnal lain yang berjudul "*Topology-Regularized Universal Vector Autoregression for Traffic Forecasting in Large Urban Areas"* pada jurnal *Expert System with Applications* membahas tentang prediksi lalu lintas perkotaan menggunakan algoritma *Vector Autoregression* (Schimbinschi et al., 2017, pp. 301–316). Penelitian tersebut menghasilkan suatu data prediksi lalu lintas di daerah perkotaan pada masa yang akan datang dimana volume kendaraan dapat bertambah banyak. Algoritma *Vector Autoregression* memberikan performa *error* yang rendah dan menghasilkan tingkat akurasi yang baik dalam melakukan prediksi pada data historis yang digunakan.

Selain itu terdapat berbagai referensi lainnya sebagai pendukung penelitian ini. Dengan mengacu pada berbagai penelitian yang berbeda pada algoritma yang sama, dihasilkan prediksi yang beragam, sehingga penelitian ini belum tentu mendapatkan hasil yang sama dengan penelitian sebelumnya dalam membuat pemodelan pergerakan harga batubara Newcastle berjangka dan menguji tingkat hasil akurasi antara algoritma *K-Nearest Neighbors* dan *Vector Autoregression.*

2.2. Tinjauan Teoritis

Tinjauan teori digunakan sebagai pendukung pada penelitian ini. Teori-teori yang dibahas adalah teori yang memiliki keterkaitan dengan topik skripsi ini. Selain itu teori tersebut juga digunakan peneliti sebagai pedoman untuk melakukannya penelitian.

2.2.1. Investasi

Investasi merupakan penanaman modal baik barang bergerak maupun tidak pada masa tertentu untuk memperoleh keuntungan di masa mendatang. Selain itu, investasi dapat menjadi tabungan masa depan baik jangka pendek maupun panjang. Ketika melakukan investasi, investor perlu informasi bisnis yang lengkap, relevan, dan akurat (Hindrayani et al., 2020, p. 71). Informasi tersebut dapat berupa data historis pergerakan harga produk. Informasi yang lengkap dan jelas dapat mendukung keputusan investor untuk berinvestasi. Investasi penting untuk perkembangan ekonomi dimana hal ini dapat menopang proses produksi (Apriani et al., 2021, p. 1). Seorang investor dapat belajar secara langsung dengan melakukannya investasi karena akan lebih terlihat seperti apa risiko yang dialami dalam melakukan investasi yang baik dan benar.

2.2.2. Batubara

Batubara termasuk sebagai sumber daya alam yang tidak dapat diperbaharui sebagai penghasil bahan bakar fosil. Batubara juga termasuk ke dalam salah satu komoditas pertambangan yang digunakan sebagai sumber penghasil energi, bahan baku industri kimia, dan petrokimia (Andayani, 2019, p. 1). Selain itu, produk aluminium, baja, dan semen juga dibuat menggunakan batubara. Batubara saat ini masih menjadi sumber energi utama yang tidak hanya menopang kebutuhan secara nasional tetapi juga memenuhi kebutuhan energi di negara lain yang menggunakan pembangkit listrik bertenaga batubara. Komoditas pertambangan tersebut menarik banyak investor mancanegara. Oleh karena itu, batubara tidak hanya sebagai sumber penghasil kebutuhan industri tetapi juga wadah untuk berinvestasi yang dapat menguntungkan.

2.2.3. Bahasa Pemrograman Python

Bahasa pemrograman Python memiliki keunggulan berupa *source code* yang simpel, mudah digunakan, multifungsi, dan memiliki komunitas yang memadai (Retnoningsih & Pramudita, 2020, p. 57). Python menggunakan *library* atau pustaka untuk mempermudah dan mempersingkat pemrograman. Berbagai macam pekerjaan dapat diselesaikan oleh Python seperti membuat halaman web,

aplikasi, dan penerapan *machine learning*. Terdapat banyak perangkat lunak *code editor* yang dapat menjalankan Python seperti Pycharm, Sublime Text 3, dan Visual Studio Code. Namun bahasa pemrograman tersebut dapat juga dijalankan tanpa harus mengunduh perangkat lunak. Hal tersebut dapat dilakukan dengan menggunakan aplikasi berbasis web seperti Google Colab dan Programiz. Sistem operasi *Windows, Mac OS,* dan *Linux* juga mendukung bahasa pemrograman Python karena bersifat *open source*. Dengan demikian, Python memiliki berbagai manfaat dan keuntungan yang dapat digunakan oleh peneliti untuk menyelesaikan penelitiannya di bidang *machine learning.*

2.2.4. *Library* dan *Tools* Pada Python

Pada pembuatan program pemodelan harga batubara Newcastle berjangka menggunakan algoritma *machine learning* dengan bahasa pemrograman Python, digunakan berbagai *library* dan *tools* yang dapat memudahkan peneliti. Adapun *library* dan *tools* yang digunakan pada penelitian ini dibahas secara singkat sebagai berikut.

(1) NumPy

NumPy merupakan *library* yang dapat digunakan sebagai alat komputasi data numerik untuk berbagai ekosistem penelitian (Harris et al., 2020, p. 357). NumPy dapat mendukung pengolahan data dengan logika *machine learning* yang digunakan sesuai kebutuhan. Sesuai dengan namanya yang berarti suatu nilai atau bilangan, oleh karena itu NumPy dibuat untuk memahami kebutuhan pengguna. Pada penelitian ini, *library* tersebut digunakan sebagai komputasi data numerik dalam membuat pemodelan harga batubara Newcastle berjangka. Pembuatan *array* dan operasi matematika didukung oleh NumPy.

(2) Pandas

Pandas merupakan salah satu *library* yang populer digunakan pada *machine learning*. Pandas dapat mempermudah pengembang dalam membuat suatu sistem dengan penawaran berbagai keunggulan salah satunya yaitu bersifat *open source*. *Library* yang ditawarkan dapat membaca dan memproses suatu tabel yang diprogram melalui bahasa pemrograman Python (Lemenkova, 2020, p. 21). *Library* tersebut digunakan pada penelitian ini untuk membaca file pada program yang telah

disimpan di Google Drive jika menggunakan Google Colab, mengelola isi dari historis harga batubara Newcastle berjangka, serta menentukan batasan prediksi tanggal secara hari kerja. Selain itu, Pandas dapat membuat visualisasi grafik dengan fitur yang cukup lengkap. Pandas mempunyai peran penting pada penelitian ini karena seluruh data tabel dikelola menggunakan *library* tersebut.

(3) Matplotlib

Matplotlib merupakan *library* yang digunakan sebagai luaran berupa visualisasi data. Visualisasi tersebut tersaji dalam grafik yang dapat mempermudah dalam membaca data secara jangka panjang. Sesuai dengan Namanya yaitu *plot*, maka grafik yang dihasilkan didasarkan pada *plot* yang diinginkan. *Plot* tersebut dituliskan pada suatu tabel yang berisikan banyak data. Grafik yang dihasilkan dapat disatukan bersama dengan data lain sehingga akan terlihat perbedaan nilai pada setiap kolom yang di inisialisasikan untuk dituangkan ke dalam suatu grafik. Selain itu, grafik yang dihasilkan dapat menentukan nama dari sumbu *x* dan *y* baik pada grafik garis maupun batang. Versi Matplotlib yang digunakan pada penelitian ini adalah 3.4.2. Versi tersebut dapat mendukung visualisasi penulisan angka di dalam diagram batang.

(4) Sklearn

Sklearn merupakan *library* pada Python yang memiliki nama lain scikit-learn. Pada penelitian ini, *library* tersebut digunakan sebagai penghitung akurasi pada algoritma *K-Nearest Neighbors*. Nilai akurasi tersebut berupa skor *Mean Squared Error* (MSE) yang diakhir akan digunakan sebagai rumus dari *Root Mean Squared Error* (RMSE) serta nilai *Mean Absolute Percentage Error* (MAPE). Selain itu, Sklearn digunakan sebagai alat untuk membagi data menjadi data latih dan data uji. Tidak hanya untuk hal tersebut, Sklearn memiliki peran penting dalam melakukan prediksi data pada algoritma *K-Nearest Neighbors*, karena digunakan untuk melakukan pengolahan prediksi data harga serta normalisasi data. Secara umum, Sklearn memiliki peran pada *machine learning* untuk klasifikasi, regresi, pengelompokan, pemodelan, dan lain-lain.

(5) Statsmodels

Statsmodels merupakan modul yang digunakan pada bahasa pemrograman Python untuk fungsi statistik (Jenkins, 2020, p. 9). Statsmodels difungsikan sebagai

alat untuk memprediksi harga (*High* dan *Low*) batubara Newcastle berjangka menggunakan algoritma *Vector Autoregression*. Secara umum, Statsmodels dapat digunakan untuk pemodelan data statistik, melakukan hipotesis data, dan eksplorasi data. Pengembang sistem dapat menggunakan alat tersebut dengan mudah tanpa memerlukan waktu panjang untuk membuat programnya. Pada penelitian ini, Statsmodels digunakan sebagai alat untuk menghitung stasioneritas dari suatu kolom dan *lag order* AIC untuk pemodelan algoritma *Vector Autoregression*.

(6) Warnings

Warnings merupakan salah satu *library* yang dikeluarkan oleh Python. Pada sebagian kondisi, peringatan perlu dihapus untuk menghilangkan ambigu pada luaran data. Namun, peringatan penting perlu diketahui oleh pengembang sistem untuk dapat segera diperbaiki. Pada penelitian ini, Warnings digunakan untuk tidak mencetak peringatan yang dianggap hanya sebagai informasi tambahan dan tidak diperlukan sebagai luaran pada sistem yang dibangun. Hal tersebut sesuai dengan *filter* yang dipilih yaitu *ignore*. Terdapat berbagai *filter* yang dapat digunakan sesuai dengan kebutuhan sistem.

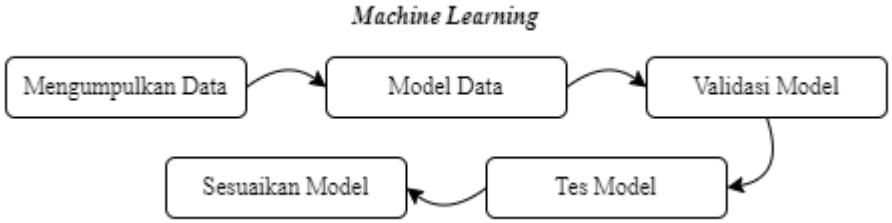
2.2.5. Algoritma *Machine Learning*

Algoritma merupakan kumpulan perintah dalam suatu permasalahan yang diselesaikan secara efisien untuk menghasilkan informasi nyata dan tepat (Maulana, 2017, p. 9). Permasalahan tersebut dapat diselesaikan secara berurutan dengan adanya algoritma yang baik. *Machine learning* merupakan turunan dari kecerdasan buatan yang dapat membangun sistem dari data yang dimasukan untuk menirukan perilaku manusia dalam menyelesaikan suatu permasalahan yang konkret (Ahmad, 2017, p. 3). Tanpa data yang jelas, *machine learning* tidak dapat melakukan pemrosesan data. *Machine learning* melatih data dengan algoritma yang dirancang untuk mengidentifikasi kumpulan prediktor dalam memprediksi hasil seakurat mungkin diantara data baru (Kleinberg et al., 2018, p. 117).

Machine learning mempunyai empat konsep algoritma pembelajaran. Konsep pertama dinamakan *supervised learning* yang merupakan pengelompokan data berdasarkan pelabelan yang dimana akan menghasilkan suatu fungsi untuk memetakan masukan ke luaran yang diinginkan. Pada konsep kedua yang bernama

unsupervised learning merupakan pengelompokan dengan mencari kesamaan pada suatu data (Osisanwo et al., 2017, p. 128). Selain itu pada konsep algoritma lainnya bernama *semi-supervised learning* yang merupakan konsep pembelajaran dimana hanya sebagian data yang dilabeli serta memiliki masukan data dalam jumlah besar. Konsep algoritma *machine learning* terakhir yaitu *reinforcement learning* dimana konsep tersebut terjadi saat algoritma yang dilakukan kekurangan label namun dapat diberikan umpan balik yang positif atau negatif sesuai dengan kondisi data (Satria, 2018).

Machine learning dapat digunakan sebagai sistem prediksi data dengan melakukan pelatihan pada data yang dimiliki. Tahap kerja dari *machine learning* menurut (Zailani et al., 2020, p. 9) terdapat pada gambar berikut.



Gambar 2.1 Tahap Kerja *Machine Learning*

Berdasarkan gambar 2.1 di atas, dapat diuraikan berdasarkan poin-poin tahap kerja *machine learning* sebagai berikut.

(1) Mengumpulkan data, yaitu dilakukan dengan membagi data menjadi dua bagian berupa data latih (*training data*) dan data yang digunakan untuk dilakukannya percobaan (*test data*).

(2) Model data, yaitu membangun model menggunakan data latih dengan menentukan fitur yang sesuai pada tujuan dilakukannya penelitian.

(3) Validasi Model, yaitu pengujian model yang telah dibuat dengan memasukkan sejumlah data untuk menghasilkan luaran berupa prediksi data.

(4) Tes model, yaitu melihat kinerja antara data uji dengan data latih, sehingga prediksi baru dapat dihasilkan dengan diaplikasikannya model yang telah dilatih.

(5) Sesuaikan model, yaitu memperbaiki kinerja algoritma dengan memberikan lebih banyak data dan dapat menambahkan fitur baru sebagai penyempurna.

2.2.6. Pengolahan Data

Data yang diproses menggunakan algoritma *K-Nearest Neighbors* dan *Vector Autoregression* terlebih dahulu dilakukan pembersihan dan di normalisasi. Pembersihan data bertujuan untuk menghilangkan data yang tidak lengkap seperti kosongnya salah satu kolom pada suatu baris data sedangkan normalisasi dilakukan untuk menghindari anomali data dan tidak konsistennya data (Drajana, 2017, p. 119). Salah satu persamaan yang dapat melakukan normalisasi adalah dengan menggunakan metode *MinMax scaler*. Metode tersebut dapat berjalan baik apabila tidak ada data yang kosong atau data kosong diisi dengan kata *NaN*. Adapun rumus normalisasi data metode *MinMax scaler* sebagai berikut.

$$x_{normalisasi} = \frac{(x - min)}{(max - min\ )} \quad (1)$$

Berdasarkan rumus ke-1 di atas, *x* merupakan nilai data, *min* sebagai nilai data minimum, dan *max* sebagai nilai data maksimum dari suatu kolom. Selanjutnya, data dapat dimodelkan menggunakan algoritma *K-Nearest Neighbors* dan *Vector Autoregression*. Setelah tahapan proses algoritma tersebut telah dilakukan secara runutan, maka selanjutnya data di denormalisasi untuk mengembalikan data yang digunakan sebagai luaran. Jika data tidak didenormalisasi maka hasil dari algoritma yang dikerjakan tidak dapat digunakan sebagaimana mestinya dikarenakan angka yang dihasilkan bukan merupakan nilai asli. Adapun rumus denormalisasi data adalah sebagai berikut.

$$x_{denormalisasi} = Y(max - min) + min \quad (2)$$

Berdasarkan rumus ke-2 di atas, dimana *Y* merupakan data hasil perhitungan algoritma, *min* sebagai nilai data minimum, dan *max* sebagai nilai data maksimum. Setelah data di denormalisasi, maka data sudah dapat digunakan sebagai suatu luaran hasil prediksi.

Pembagian rasio data latih dan data uji diperlukan dalam proses pengolahan data. Setiap rasio dapat memberikan hasil pemodelan dan evaluasi hasil yang berbeda. Pada penelitian berjudul "*Categorization of Gelam, Acacia, and Tualang Honey Odor-Profile Using K-Nearest Neighbors*" membandingkan hasil prediksi dengan menggunakan berbagai rasio dari data latih dan data uji (Zahed et al., 2018, pp. 15–28). Penelitian tersebut melakukan percobaan rasio mulai dari 10:90 sampai

dengan 90:10 terhadap data latih dan data uji. Hasil yang diperoleh dengan menggunakan K=1 didapatkan evaluasi hasil *error* tertinggi saat menggunakan rasio 10:90 dan diantara 70:30 sampai 90:10 memberikan hasil yang rendah. Namun, hasil akhir pada proses klasifikasi menggunakan algoritma *K-Nearest Neighbors* berdasarkan kasus tersebut menunjukkan tingkat akurasi, sensitivitas, dan spesifisitas 100% saat menggunakan rasio 90:10 dimana jarak Euclidean yang diharapkan dapat tercapai dan memperoleh MSE sebesar 0.

Pada penelitian lain yang berjudul "Komparasi Metode *Data Mining K-Nearest Neighbor* dengan *Naïve Bayes* untuk Klasifikasi Kualitas Air Bersih (Studi Kasus PDAM Tirta Kencana Kabupaten Jombang)" menghasilkan berbagai rasio pengujian untuk mendapatkan hasil komparasi yang terbaik pada klasifikasi kualitas air bersih (Rahman et al., 2018, pp. 6346–6353). Penelitian tersebut menggunakan rasio 90:10 sampai 60:40 terhadap data latih dan data uji. Berdasarkan rasio pada penelitian tersebut, hasil akurasi tertinggi diperoleh saat menggunakan rasio 90:10 dimana algoritma *K-Nearest Neighbors* sebesar 84.71% dan *Naïve Bayes* sebesar 73.53%.

Pada penelitian selanjutnya yang berjudul "*Evolving Hybrid Cascade Neural Network Genetic Algorithm Space–Time Forecasting*" menggunakan *Vector Autoregression* sebagai algoritma simulasi data (Caraka et al., 2021, pp. 1–20). Penelitian tersebut menggunakan rasio mulai dari 90:10 sampai 50:50 terhadap data latih dan data uji. Hasil dari analisis algoritma *Casade Neural Network Genetic* menggunakan *Vector Autoregression* sebagai algoritma simulasi data memberikan nilai RMSE, MAE, dan sMAPE terendah saat menggunakan rasio 90:10. Dengan demikian, berdasarkan berbagai jurnal yang dipaparkan sebelumnya dapat diketahui bahwa rasio data latih dan data uji dapat ditentukan dengan melakukan percobaan berbagai rasio. Namun, berdasarkan berbagai jurnal tersebut didapatkan bahwa rasio 90:10 terhadap data latih dan data uji dengan menggunakan algoritma dan kasus yang berbeda memberikan evaluasi hasil akhir yang lebih baik daripada menggunakan tingkat rasio lainnya.

2.2.7. *K-Nearest Neighbors*

K-Nearest Neighbors merupakan salah satu dari algoritma *machine learning* yang bertujuan untuk mengklasifikasi data uji berdasarkan pola uji dan dapat belajar mandiri dalam menangani suatu permasalahan (Cai et al., 2020, p. 22687). Algoritma tersebut digunakan untuk memperoleh hasil dengan mengambil data terdekat yang dicari berdasarkan pembelajaran data yang disimpan pada sistem. *K-Nearest Neighbors* termasuk ke dalam konsep *supervised learning* pada *machine learning* karena algoritma tersebut bekerja berdasarkan data yang telah diketahui statusnya (dilabeli) dan menghasilkan luaran berupa prediksi yang ditentukan sesuai dengan tujuan dilakukannya penelitian.

K-Nearest Neighbors mempunyai keunggulan mudah dipahami karena untuk melakukan klasifikasi jarak suatu data hanya dilakukan dengan mendefinisikan fungsi (Cahya, 2018). Selain itu, algoritma tersebut dapat digunakan pada berbagai kasus yang memiliki data numerik untuk diproses. Namun jika data yang dimiliki berupa suatu kata (*string*) dapat dikonversi menjadi numerik terlebih dahulu sebelum dilakukannya pemrosesan pada algoritma *K-Nearest Neighbors.* Adapun langkah – langkah pemrosesan algoritma *K-Nearest Neighbors* menurut (Shabani et al., 2020, p. 5) sebagai berikut.

(1) Membagi data menjadi data latih dan uji. Pembagian data menggunakan rasio 90:10. Rasio tersebut digunakan berdasarkan studi literatur di berbagai penelitian yang memberikan hasil terbaik saat menggunakan 90:10 terhadap data latih dan data uji.

(2) Menghitung jarak antara data latih dan data uji menggunakan rumus *Euclidean Distance* pada metode Elbow seperti berikut ini.

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (3)$$

Berdasarkan rumus ke-3 di atas, *x* adalah data latih, *y* adalah data uji, *n* merupakan banyaknya data, dan *d* adalah jarak.

(3) Menentukan nilai jumlah tetangga (*K*) terbaik menggunakan MSE. *Mean Square Error* (MSE) digunakan sebagai alat ukur akurasi dimana hasil terkecilnya adalah nilai *K* terbaik. Besaran nilai *K* dapat mempengaruhi

tingkat akurasi dan kinerja pada algoritma *K-Nearest Neighbors.* Semakin kecil nilai *K* yang dipilih maka kinerja yang dihasilkan akan menampilkan skor terbaik.

(4) Data jarak diurutkan dari terkecil hingga terbesar (*ascending*) sesuai dengan nilai *K* yang ditentukan.

(5) Hitung rata-rata dari data yang telah diurutkan untuk mendapatkan prediksi dari algoritma *K-Nearest Neighbors*. Setelah data prediksi didapatkan, maka terlebih dahulu dilakukan proses denormalisasi data berdasarkan rumus yang diuraikan pada sub sub bab pengolahan data. Denormalisasi akan menghasilkan data akhir prediksi yang memiliki nilai asli.

2.2.8. *Vector Autoregression*

Algoritma *Vector Autoregression* (VAR) merupakan pilihan yang tepat untuk menghadapi berbagai masalah terutama pada pembuatan suatu prediksi menggunakan deret data harian atau *time series* (Schimbinschi et al., 2017, p. 314). *Time series* merupakan pergerakan data secara berkala yang diperbaharui berupa bilangan numerik maupun diagram dalam waktu tertentu (Ismail Fawaz et al., 2019, p. 917). *Vector Autoregression* adalah pengembangan dari algoritma *Univariate Autoregressive* yang dapat memprediksi data *time series* kontemporer. Kontemporer yang dimaksudkan adalah data yang tersaji secara *real-time* atau sesuai dengan kondisi masa kini. Algoritma *Vector Autoregression* memberikan hasil yang optimal dengan berbagai tahapan yang memiliki kompleksitas dalam mengolah data *time series*.

Model dari *Vector Autoregression* ini memiliki variabel sebagai kombinasi linear dari data historis dimana data tersebut saling mempengaruhi variabel lain dalam sistem. Misalnya dalam suatu prediksi akan dimiliki dua variabel (*time series*) berupa Y_1 dan Y_2. Kedua nilai variabel tersebut perlu diprediksi pada suatu waktu (t). Nilai variabel pada $Y_{1(t)}$ dan $Y_{2(t)}$ dapat ditentukan dengan menggunakan data historis yang digunakan pada Y_1 dan Y_2. Adapun formula atau rumus tersebut dapat dituliskan sebagai berikut.

$$Y_{1,t} = \alpha_1 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \epsilon_{1,t}$$
$$Y_{2,t} = \alpha_2 + \beta_{21,1} Y_{1,t-1} + \beta_{22,1} Y_{2,t-1} + \epsilon_{2,t} \quad (4)$$

Persamaan rumus ke-4 di atas merupakan model dari $VAR_{(1)}$ dimana Y_1 dan Y_2 memuat satu *lag* karena model tersebut berorde 1. Model tersebut dapat dikembangkan sesuai dengan tujuan dari dilakukannya penelitian berdasarkan data historis yang digunakan. Semakin banyaknya data yang digunakan maka persamaan juga menjadi lebih besar (Prabhakaran, 2019). Pada pemrosesan data menjadi suatu prediksi menggunakan algoritma *Vector Autoregression* dihasilkan melalui berbagai tahapan. Tahapan tersebut digunakan supaya peneliti dapat mengetahui bagaimana cara kerja dari algoritma *Vector Autoregression.* Adapun tahap – tahap pada algoritma tersebut menurut (Dissanayake, 2020) adalah sebagai berikut.

(1) Membagi Data.

Pada tahap pertama, data dibagi menjadi dua yaitu data latih dan data uji. Pembagian data ini berdasarkan pada total hari yang akan diprediksi pada data uji dengan algoritma *Vector Autoregression*.

(2) Pengujian Stasioneritas dengan AD-*Fuller*.

Stasioneritas adalah statistik yang tidak berubah waktu ke waktu dari deret waktu pada suatu data. Pada deret waktu yang bernilai nol maka dianggap tidak stasioner sehingga diperlukannya pengujian untuk membuat deret tersebut menjadi stasioner (menolak hipotesis yang bernilai nol). Hipotesis bernilai nol merupakan data yang diperoleh memiliki hasil lebih dari nilai signifikan. Nilai signifikan pada pengujian *Augmented Dickey-Fuller* (AD-*Fuller*) adalah 0.05 (*default value*). Nilai yang dihasilkan pada proses AD-*Fuller* harus kurang dari ketetapan pada nilai signifikan supaya kolom yang akan diproses memiliki data stasioner. Data yang stasioner diperlukan pada pemrograman menggunakan algoritma *Vector Autoregression* karena data historis berupa *time series* perlu diketahui bahwa data seperti rata-rata dan varian tidak berubah seiring waktu. Dengan demikian, pengujian stasioneritas diperlukan pada pemrograman pemodelan menggunakan algoritma *Vector Autoregression*.

Apabila data yang dihasilkan belum stasioner, maka diperlukan proses lanjutan supaya data tersebut dapat stasioner. Proses lanjutan tersebut dengan digunakannya fungsi *diff* untuk melakukan transformasi data. Fungsi *diff* memiliki logika perhitungan pengurangan data sekarang dengan lampau sebagai contoh yaitu nilai pada baris kedua kolom pertama dikurangi dengan baris pertama kolom

pertama, maka hasil pengurangan tersebut diletakkan pada baris pertama kolom pertama. Proses tersebut dijalankan sampai dengan baris dan kolom terakhir. Setelah transformasi data dilakukan, maka program pengujian AD-*Fuller* dijalankan kembali. Jumlah banyaknya proses transformasi perlu diketahui karena nilai setiap kolom dan baris perlu dikembalikan menjadi nilai yang asli.

(3) Memilih Urutan Model.

Vector Autoregression memberikan berbagai pilihan model seperti AIC, BIC, FPE, dan HQIC. Pada umumnya *Akaike Information Criterion* (AIC) dipilih karena memiliki model yang lebih baik dan efektif untuk memproses algoritma deret waktu (*time series*) daripada model lainnya serta sebagai penyelarasan penggunaan urutan model yang sama pada proses *Augmented Dickey-Fuller* dalam menguji stasioneritas data (Goloboff & Arias, 2019, p. 714). Namun model lainnya dapat diuji terlebih dahulu jika dibutuhkan untuk memastikan nilai urutan model yang akan digunakan.

Cara menentukan urutan model terbaik adalah dengan melihat nilai yang dikeluarkan oleh AIC dimana nilai terendah dalam suatu deret dengan urutan model lebih dari 1 maka itu yang akan dipilih, namun pertimbangan lain seperti perbedaan AIC yang tidak begitu signifikan dapat diabaikan (Portet, 2020, p. 123).

(4) Melatih Model.

Pada tahap ini dilakukannya pelatihan model yang telah dipilih dengan menggunakan set pelatihan yang tersedia di *library* Python. Model dilatih berdasarkan nilai AIC yang terbaik dan data terakhir yang digunakan.

(5) Prediksi Data.

Data yang telah dinormalisasi sebelumnya dilakukan pemrosesan algoritma menggunakan Statsmodels untuk menghasilkan data prediksi. Ditentukan juga indeks tabel yang akan digunakan yaitu berupa tanggal prediksi. Pada penentuan tanggal, digunakan frekuensi setiap hari Senin sampai Jumat, maka digunakan *freq="B"* untuk mewujudkan hal tersebut.

(6) Mengembalikan Data Transformasi.

Data pemodelan prediksi yang telah dihasilkan akan dikembalikan ke dalam format nilai asli. Jika sebelumnya dilakukan transformasi data sebanyak satu kali, maka proses pengembalian data dilakukan sebanyak satu kali juga. Begitu juga jika

dilakukan lebih dari satu kali, maka proses pengembalian data dilakukan sebanyak jumlah dilakukannya transformasi data.

(7) Denormalisasi Hasil.

Prediksi data yang telah dihasilkan dan telah dikembalikan ke nilai sebenarnya, maka selanjutnya dilakukan proses denormalisasi. Denormalisasi akan mengembalikan nilai di setiap kolom dan baris seperti kondisi nyata.

2.2.9. Evaluasi Hasil

Evaluasi hasil bertujuan untuk mengukur tingkat hasil akurasi pada algoritma *machine learning* yang digunakan. *Root Mean Squared Error* (RMSE) merupakan formula atau rumus yang dapat digunakan untuk mengukur nilai akurasi dari kinerja algoritma *machine learning* (Wang & Lu, 2018, p. 1). Algoritma terakurat memiliki nilai RMSE terkecil karena tingkat *error* yang diperoleh sangat rendah. Adapun rumus RMSE sebagai berikut.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{n}} \quad (5)$$

Berdasarkan rumus ke-5 di atas, *y* merupakan hasil data yang sebenarnya, $\hat{y}$ adalah data hasil prediksi, dan *n* sebagai jumlah data percobaan. Setiap algoritma yang sama dengan data yang berbeda belum tentu menghasilkan RMSE yang sama. Oleh karena itu, tidak ada yang salah dalam menghitung *Root Mean Squared Error* (RMSE) pada algoritma *machine learning*. Selain itu, digunakan *Mean Absolute Percentage Error* (MAPE) dalam mengukur tingkat akurasi dari algoritma *machine learning* yang digunakan. MAPE bekerja dengan cara mempertimbangkan rasio antara *error* dengan nilai yang sebenarnya (Tian et al., 2018, p. 7). Adapun rumus MAPE sebagai berikut.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{A_i - F_i}{A_i}\right| x\ 100\% \quad (6)$$

Berdasarkan rumus ke-6 di atas, *n* merupakan jumlah sampel, A_i adalah nilai data sebenarnya, dan F_i sebagai nilai data prediksi. Setiap nilai MAPE yang dihasilkan akan dalam satuan persen. Adapun setiap hasil MAPE memiliki arti menurut (Aindhae, 2019) sebagai berikut.

Tabel 2.1 Arti Hasil Nilai MAPE

Nilai MAPE (%)	Keterangan
≤ 10	Prediksi sangat akurat
10 – 20	Prediksi baik
20 – 50	Prediksi cukup baik
> 50	Prediksi tidak akurat

Berdasarkan tabel 2.1 di atas, dapat diketahui bahwa semakin rendah nilai MAPE yang dihasilkan, maka semakin akurat model tersebut digunakan dalam melakukan suatu pemodelan prediksi data. Misalkan nilai MAPE yang dihasilkan adalah 3%, maka rata-rata nilai prediksi yang diperoleh mempunyai selisih dengan nilai sebenarnya sebesar 3%.