PENCARIAN ALGORITMA TERBAIK DALAM PREDIKSI HARGA OPEN SAHAM BERDASARKAN TINGKAT KASUS COVID-19

DIUSULKAN OLEH: TIM MIRACLE

Wisanti ; 18102072

Muhamad Aldi Hafidzudin ; 18102059

Rolanita Scenic Faravati ; 18102069

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

BANYUMAS

2020

PENDAHULUAN

1. Latar Belakang

Covid-19 merupakan penyakit yang menyerang pada saluran pernapasan manusia. Virus ini dideteksi kemunculannya pertama kali di Wuhan, Tiongkok. Virus ini mulai menyebar di penjuru dunia hingga menjadikan sebuah wabah dan dunia mengalami pandemi besar-besaran. Berdasarkan data WHO per 29 November 2020 tercatat kasus positif sebanyak 534.266 dengan jumlah 445.793 di nyatakan sembuh dan 16.815 meninggal di Indonesia (https://covid19.go.id/). Dalam mengantisipasi melonjaknya kasus, Indonesia melakukan sistem pembatasan sosial berskala besar sejak tanggal 10 April 2020 (Purnaningrum, 2020). Hal ini cukup mempengaruhi perkembangan ekonomi di Indonesia.

Pada pandemi Covid-19 ini, masyarakat banyak mengalami kesulitan finansial. Banyak diantara mereka yang terkena pemutusan hubungan kerja secara mendadak. Hal ini dapat mengakibatkan menurunnya kondisi perekonomian pada masyarakat Indonesia. Menurut Bodie, et al (2007) dalam Kartikaningsih (2020) mengatakan bahwa faktor makro ekonomi yang dapat mempengaruhi harga saham adalah suku bunga, nilai tukar dan inflasi. Menurut Zhang, Hu, & Ji (2020) dalam Purnaningrum (2020) Ketidakpastian penyebaran virus dan kerugian perekonomian menyebabkan pasar menjadi sangat fluktuatif dan tidak dapat diprediksi. Krisis tersebut mempengaruhi penjualan saham sehingga mengakibatkan penurunan harga saham (Ahmar & del Val, 2020). Pada laman finance.yahoo.com ditunjukkan bahwa harga saham mengalami dead cat bounce pada waktu pertama kali virus Covid-19 mewabah pada bulan Maret 2020. Harga saham mengalami pemantulan kenaikan pada awal bulan April 2020.

Data sampel pada kasus Covid-19 yang sedikit dan sulit untuk diprediksi menjadikan peneliti membandingkan beberapa algoritma data mining *Classifier* untuk mencari algoritma terbaik dalam memprediksi harga saham. Algoritma yang dipakai adalah K-Nearest Neighbor (KNN), *AdaBoost*, *Bagging Classifier*, Random Forest Classifier, *Extra Trees*

Classifier, *Gradient Boosting Classifier*, *Stacking Classifier*, *Dummy Classifier* dan *SGDClassifier*.

Algoritma K-Nearest Neighbor (KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data contoh yang mempunyai jarak paling dekat dengan objek tersebut. Algoritma ini hanya melakukan penyimpanan dan klasifikasi data dan memiliki konsistensi yang kuat. Algoritma *AdaBoost* dari Freund dan Schapire (1995) merupakan algoritma penguat praktis pertama, dan tetap menjadi salah satu yang paling banyak digunakan dan dipelajari, dengan aplikasi di berbagai bidang. Algoritma ini dapat dikombinasikan dengan algoritma lain untuk meningkatkan performa klasifikasinya. Teknik bagging merupakan salah satu teknik ensemble dan teknik ini digunakan pada klasifikasi untuk memisahkan data training ke dalam beberapa data training baru dengan random sampling dan membangun model berbasis data training baru (Wahono & Suryana, 2013). *Random Forest* merupakan salah satu metode yang digunakan untuk klasifikasi dan regresi. Sedangkan *Extra Trees Classifier* merupakan pemilihan fitur yang berbasis *decision tree*. Boosting merupakan meta-algoritma dalam *machine learning* untuk melakukan *supervised learning*. *Stacking* atau *Stacked generalization* adalah teknik lain untuk menggabungkan *multi classifier*. Tidak seperti teknik bagging dan *boosting*, *stacking* digunakan untuk menggabungkan classifier yang berbeda. *Dummy variable* merupakan pengkodean ulang dari *categorical variables* yang mempunyai lebih dari dua kategori yang diubah menjadi beberapa *binary variable*.

Pada penelitian kali ini, peneliti melakukan perbandingan dari sepuluh algoritma tersebut untuk mencari algoritma terbaik yang mampu memprediksi tingkat harga *open* saham per hari dengan keakuratan tertinggi. Kemudian peneliti mencari ketergantungan dari tingkat harga *open* saham per hari dengan kasus Covid-19 di Indonesia. Peneliti menggunakan Bahasa Python dalam melakukan kajian serta menggunakan populasi data harga *open* saham, harga *high* saham, harga *low* saham dan harga *close* saham pada perusahaan Telkomsel, Indosat dan Smartfren pada

kurun waktu 8 tahun. Data tersebut dijadikan format .csv dan dimasukkan ke dalam Google Collaboraty. Hal ini dilakukan untuk menghemat waktu pengerjaan dan dapat membuktikan algoritma terbaik yang dapat digunakan dengan cepat.

2. Rumusan Masalah

1. Mencari algoritma terbaik dalam memperoleh tingkat akurasi tertinggi harga open saham.
2. Membandingkan tingkat harga *open* saham dari populasi dataset.
3. Keterkaitan antara tingkat harga *open* saham dengan kasus Covid-19 di Indonesia.

3. Batasan Masalah

1. Dataset berupa data harga open saham Telkomsel, Indosat dan Smartfren dalam kurung waktu 8 tahun terakhir.
2. Sampel algoritma yang digunakan sebanyak 10 algoritma klasifikasi.
3. Sampel data kasus Covid-19 per hari.
4. Sistem yang digunakan menggunakan Bahasa Python pada Google Collaboratory.

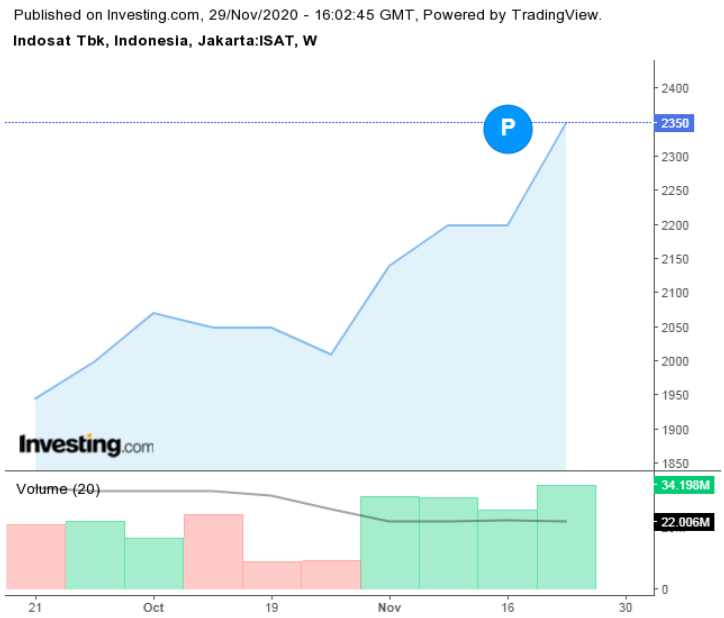
TUJUAN DAN MANFAAT

1. Tujuan

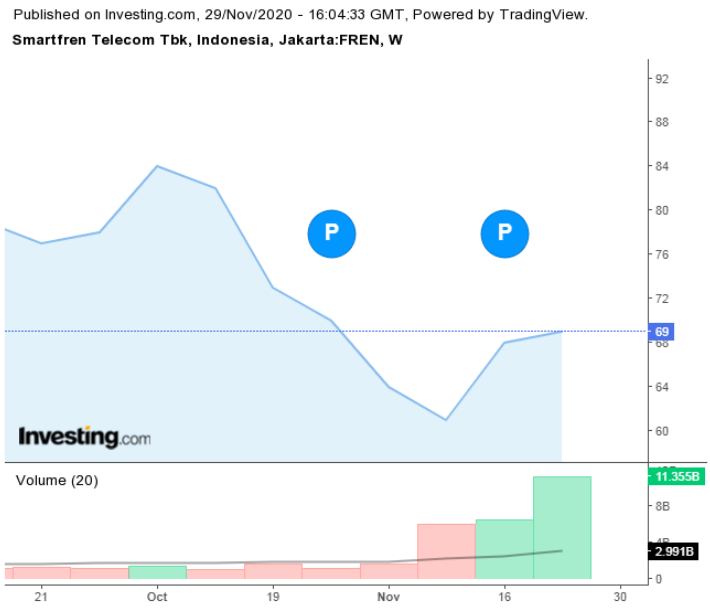
1. Memprediksi harga saham dimasa depan dari ketiga dataset yang telah dikumpulkan.

Contoh perbandingan harga 3 saham dari rentang waktu 1 minggu terakhir, hasil output dari metode yang kami gunakan adalah tingkat akurasi prediksi kenaikan harga saham dari ke 3 dataset.

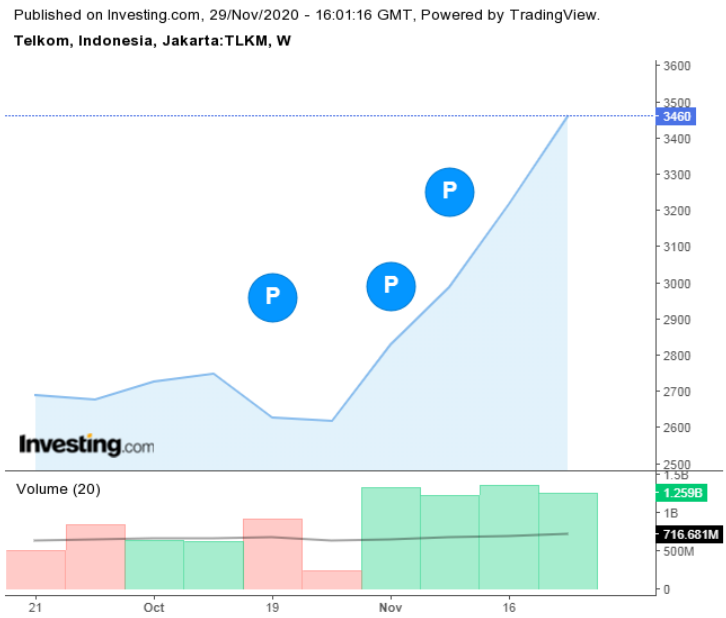
Indosat Tbk, Indonesia, Jakarta:ISAT



Smartfren Telecom Tbk, Indonesia, Jakarta:FREN



Telkom, Indonesia, Jakarta:TLKM



2. Mengidentifikasi keterhubungan antara data kasus persebaran covid-19 di indonesia terhadap harga saham

Informasi grafik persebaran covid-19 dalam rentang waktu satu minggu

2. Manfaat

Manfaat dari pencarian algoritma terbaik dalam peramalan harga open saham berdasarkan tingkat kasus covid-19 adalah untuk memprediksi harga open saham saat pandemic berlangsung, hasil output berupa tingkat akurasi dari 3 kategori data set saham, dengan melihat tingkat akurasi kita bisa memprediksi harga saham dimasa depan hingga mempermudah keputusan dalam menjual atau membeli saham guna mendapat keuntungan.

METODE PENGOLAHAN DATA

1. Metode KNN

Algoritma K-Nearest Neighbor (KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data contoh yang mempunyai jarak paling dekat dengan objek tersebut.

Algoritma ini hanya melakukan penyimpanan dan klasifikasi data contoh. Pada fase klasifikasi, fitur-fitur yang sama dihitung untuk testing data (klasifikasinya belum diketahui). Jarak dari data contoh yang baru ini terhadap seluruh data contoh dihitung, dan sejumlah K yang paling dekat diambil. Titik yang baru klasifikasinya diprediksikan termasuk pada klasifikasi terbanyak dari titik-titik tersebut.

Ketepatan algoritma KNN ditentukan oleh ada atau tidak adanya data yang tidak relevan, atau jika bobot fitur tersebut setara dengan relevansinya terhadap klasifikasi. Algoritma K Nearest Neighbor memiliki kelebihan yaitu dapat menghasilkan data yang kuat atau jelas dan efektif jika digunakan pada data dengan jumlah yang cukup besar. Namun selain beberapa kelebihan tersebut, K-Nearest Neighbor juga memiliki kekurangan yaitu membutuhkan nilai K sebagai parameter, jarak dari data percobaan tidak dapat jelas dengan tipe jarak yang digunakan dan dengan atribut yang digunakan untuk memperoleh hasil yang terbaik, maka harus menggunakan semua atribut atau hanya satu atribut yang telah pasti.

Algoritma KNN ini memiliki konsistensi yang kuat. Ketika jumlah data mendekati tak terhingga, algoritma ini menjamin error rate yang tidak lebih dari dua kali Bayes error rate (error rate minimum untuk distribusi data tertentu).

2. Metode AdaboostClassifier

Adaptive boosting (adaboost) merupakan salah satu dari beberapa varian pada algoritma boosting (Liu, 2015).

Adaboost merupakan ensemble learning yang sering digunakan pada algoritma boosting. Algoritma *AdaBoost* dari Freund dan Schapire (1995)

merupakan algoritma penguat praktis pertama, dan tetap menjadi salah satu yang paling banyak digunakan dan dipelajari, dengan aplikasi di berbagai bidang. Boosting bisa dikombinasikan dengan *classifier* algoritma yang lain untuk meningkatkan performa klasifikasi. Tentunya secara intuitif, penggabungan beberapa model akan membantu jika model tersebut berbeda satu sama lain.

Adaboost dan variannya telah sukses diterapkan pada beberapa bidang (domain) karena dasar teorinya yang kuat, prediksi yang akurat, dan kesederhanaan yang besar. Langkah-langkah pada algoritma adaboost adalah sebagai berikut.

a. *Input* : Suatu kumpulan sample penelitian dengan label $\{(x_i, y_i), \dots, (x_N, y_N)\}$, suatu component learn algoritma, jumlah perputaran T.
b. *Inititalize* : Bobot suatu sampel pelatihan w_i^1 = 1/N, untuk semua i=1,…, N
c. Do fot t = 1,…,T
    1. Gunakan component learn algoritma untuk melatih suatu komponen klasifikasi, , pada sample bobot pelatihan.
    2. Hitung kesalahan pelatihannya pada h_t: $\mathcal{E}_i = \sum_{i=1}^{N} w_i^t , y_i \neq h_t(x_i)$
    3. Tetapkan bobot untuk component classifier $h_t = \alpha_i = \frac{1}{2} \ln (\frac{1-\mathcal{E}_t}{\mathcal{E}_t})$
    4. Update bobot sample pelatihan $w_i^{t+1} = \frac{w_i^1 \exp\{-\alpha_t y_i h_t(x_i)\}}{C_i}$, i = 1,…,N

    C_t
d. Output: f(x) = sign($\sum_{t=1}^{T} a_t, h_t(x)$).

3. Metode BaggingClassifier

Teknik bagging merupakan salah satu teknik ensemble dan teknik ini digunakan pada klasifikasi untuk memisahkan data training ke dalam beberapa data training baru dengan random sampling dan membangun model berbasis data training baru (Wahono & Suryana, 2013). Di bawah ini adalah algoritma Bagging untuk model klasifikasi (Liu & Zhou, 2013).

Input : Data set D= $\{(x_i, y_i)\}_{i=1}^{n}$

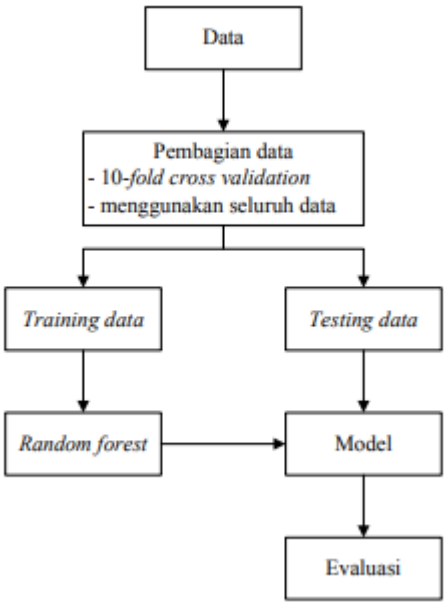
Base learning algorithm Ƹ The number of iterations T

1. for t=1 to T do
2. h_t = ℜ (D, D_{bs})
3. End for
4. Output: H(x) = $max_y \sum_{t=1}^{n} I \ (h_t(x) = y)$.

4. Metode RandomForestClassifier

Random forest merupakan salah satu metode yang digunakan untuk klasifikasi dan regresi. Metode ini merupakan sebuah ensemble (kumpulan) metode pembelajaran menggunakan pohon keputusan sebagai base classifier yang dibangun dan dikombinasikan [8]. Ada tiga aspek penting dalam metode random forest, yaitu: (1) melakukan bootstrap sampling untuk membangun pohon prediksi; (2) masing-masing pohon keputusan memprediksi dengan prediktor acak; (3) lalu random forest melakukan prediksi dengan mengkombinasikan hasil dari setiap pohon keputusan dengan cara majority vote untuk klasifikasi atau rata-rata untuk regresi.

Pada Gambar dibawah ini diberikan informasi terkait dengan langkah implementasi algoritma random forest untuk prediksi curah hujan. Langkah awal adalah melakukan input data hasil dari transformasi data dimana terdiri dari atribut penjelas dan atribut target. Setelah itu data dibagi menjadi dua jenis (training data dan testing data) dengan menggunakan metode K-Fold Cross Validation dimana nilai K dipilih 10. Selain itu, penentuan training data dan testing data juga dilakukan dengan menggunakan seluruh data. Nantinya akan dilakukan perbandingan hasil antara kedua jenis metode penentuan training data dan testing data tersebut. Algoritma random forest pada penelitian ini menggunakan sepuluh pohon keputusan yang dibangkitkan secara acak. Training data digunakan sebagai data masukan untuk algoritma random forest sedangkan testing data digunakan untuk menguji atau mengevaluasi output atau model yang dihasilkan dari algoritma random forest.



Implementasi Random Forest

5. **Metode Extra Trees Classifier**

Extra trees classifier merupakan pemilihan fitur yang berbasis decision tree. Algoritma ETC berbeda dari metode decision tree lainnya, yaitu menggunakan parameter, membagi node dengan memilih titik potong seluruhnya secara acak dan membangun setiap tree dengan menggunakan sampel data pelatihan asli. Parameter yang digunakan dalam algoritma ETC yaitu:

- K merupakan parameter yang menentukan jumlah fitur yang diambil secara acak pada setiap node.
- n_{min} merupakan parameter yang menentukan banyaknya sampel minumum yang digunakan untuk melakukan pemisahan node.
- M merupakan parameter yang menentukan jumlah pohon di ensemble. Metode ETC mengeliminasi fitur – fitur yang berada di bawah bobot yang ditentuka sesuai dengan algoritma Pseudo – code dari algoritma Extra Trees (Geurts, 2006)

6. **Metode GaussianProcessClassifier**

Boosting merupakan meta-algoritma dalam machine learning untuk melakukan supervised learning. Secara umum, boosting terjadi dalam iterasi, secara incremental menambahkan weak learner ke dalam satu strong learner. Pada setiap iterasi, satu weak learner belajar dari suatu data latihan. Kemudian, weak learner itu ditambahkan ke dalam strong learner. Setelah weak learner ditambahkan, data-data kemudian diubah masing-masing beratnya. Data-data yang mengalami kesalahan klasifikasi akan mengalami penambahan bobot, dan data-data yang terklasifikasi dengan benar akan mengalami pengurangan bobot. Oleh karena itu, weak learner pada iterasi selanjutnya akan lebih terfokus pada data-data yang mengalami kesalahan klasifikasi oleh weak learner yang sebelumnya.

Gradient boosting adalah salah satu dari algoritma boosting, dimana menghasilkan model prediksi dari weak learner berbentuk decision tree. Gradient boosting melatih decision tree untuk meminimalkan loss function, Gradient boosting menangan fitur kategoris, perluasan dari multiclass classification setting, tidak memerlukan fitur scaling, dan dapat menangkap fitur nonlinearities dan interaksi.

7. **Metode DummyClassifier**

Dummy variable merupakan pengkodean ulang dari categorical variables yang mempunyai lebih dari dua kategori yang diubah menjadi beberapa binary variable. Contoh : Status Pernikahan, jika data asli dilabeli dengan 1 = Menikah, 2 = Belum Menikah, 3 = Cerai/Janda/ Duda/Berpisah, dapat diubah menjadi dua variable sebagai berikut : var_1 : 1 = Belum Menikah, 0 = Lain, var_2 : 1 = Cerai/Janda/ Duda/Berpisah.

Untuk kasus diatas jika ada seorang yang sudah menikah, maka kedua var_1 dan var_2 akan memiliki nilai 0. Umumnya, categorical variable dengan kategori (k) akan dikodekan mennjadi (k -1) untuk dummy variable.

Contoh penggunaan dummy variable

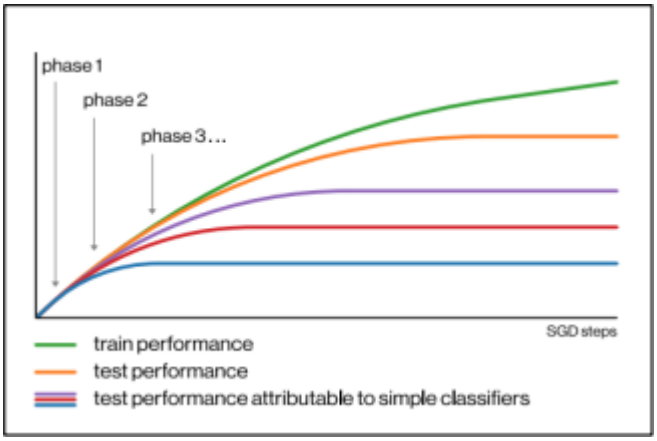
8. **Metode GradientBoostingClassifier**

Boosting merupakan meta-algoritma dalam machine learning untuk melakukan supervised learning. Secara umum, boosting terjadi dalam iterasi, secara incremental menambahkan weak learner ke dalam satu strong learner. Pada setiap iterasi, satu weak learner belajar dari suatu data latihan. Kemudian, weak learner itu ditambahkan ke dalam strong learner. Setelah weak learner ditambahkan, data-data kemudian diubah masing-masing bobotnya. Data-data yang mengalami kesalahan klasifikasi akan mengalami penambahan bobot, dan data-data yang terklasifikasi dengan benar akan mengalami pengurangan bobot. Oleh karena itu, weak learner pada iterasi selanjutnya akan lebih terfokus pada data-data yang mengalami kesalahan klasifikasi oleh weak learner yang sebelumnya.

Gradient boosting adalah salah satu dari algoritma boosting, dimana mengasilkan model prediksi dari weak learner berbentuk decision tree. Gradient boosting melatih decision tree untuk meminimalkan loss function, Gradient boosting menangan fitur kategoris, perluasan dari multiclass classification setting, tidak memerlukan fitur scaling, dan dapat menangkap fitur nonlinearities dan interaksi.

9. **Metode SGDClassifier**

Stochastic Gradient Descent ialah pendekatan sederhana dan efisien dalam melakukan klasifikasi linier dengan pembelajaran diskriminatif . Proses klasifikasi menggunakan SGD dapat dilihat pada Gambar dibawah



SGD Classification

Metode SGD merupakan algoritma optimasi iteratif untuk mencari titik fungsi minimum yang dapat diturunkan. Algoritmanya dimulai dengan cara melakukan penebakan di awal proses. Kesalahan penebakan-penebakan kemudian diperbaiki seiring adanya perulangan tebakan yang menggunakan aturan gradient (turunan) dari fungsi yang hendak diminimalkan. Penurunan fungsi minimum digunakan secara khusus dengan rumus.

$$\omega_i + 1 = \omega_i - \eta \nabla_{\omega_i} L(\omega_i)$$

Keterangan :

$\omega_i + 1$ = Model parameter prediksi
ω_i = Parameter model pada iterasi sebelumnya
η = Tingkat pembelajaran
L = *loss cost function*

Pelaksanaanya berdasarkan ukuran dataset latih. Metode SGD memiliki kemampuan belajar yang lebih cepat. Fungsi hinge loss yang digunakan sebagai pelatihan classifier dapat dijelaskan dengan rumus.

$$L(x_j, y_j) = \max(0,1 - y_j . (\omega x_j + b))$$

Keterangan:

$\omega\ dan\ b$ = Model parameter untuk prediksi
x_j = Contoh input
y_j = Target kelas

Persamaan diatas merupakan fungsi metrik klasifikasi sebagai alat ukur kemampuan model linier yang telah diprediksi menggunakan metode SGD pada setiap perulangan fase pembelajaran.

Loss function juga bekerja dengan regularisasi yang bertujuan untuk membantu model yang diprediksi serta menggeneralisasi data yang tidak memiliki label. Regularisasi bertugas sebagai protokol untuk menghukum model kompleks yang lebih dominan terjadi overfitting, yang ditandai dengan nilai yang lebih besar untuk parameter ω_i. Regularisasi dapat dilihat pada rumus.

$$L1 = \sum_{i=1}^{m} |\omega_i|$$

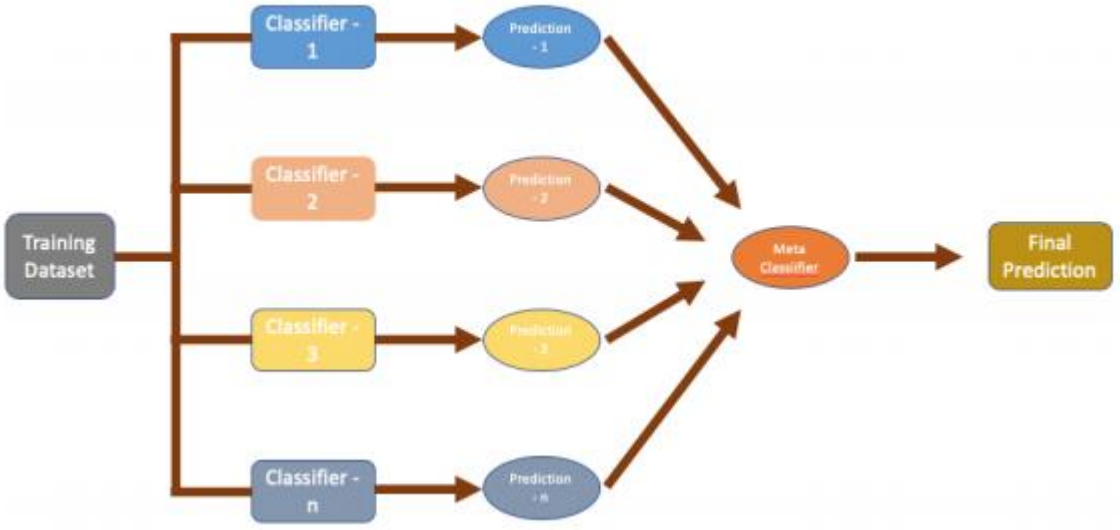
$$L2 = \sum_{i=1}^{m} \omega_i^2$$

Keterangan:
m = Prediktor variabel
ω = Model parameter untuk prediksi

10. StackingClassifier

Stacking atau Stacked generalization adalah teknik lain untuk menggabungkan multi classifier. Tidak seperti teknik bagging dan boosting, stacking digunakan untuk menggabungkan classifier yang berbeda, misal : decision tree, neural network, rule induction, naïve bayes, logistic regression dan lain-lain.

Stacking terdiri dari dua level yaitu base learner sebagai level-0 dan stacking model learner sebagai level-1 atau meta learner. Base learner (level-0) menggunakan model yang berbeda untuk belajar dari suatu dataset. Output dari masingmasing model dikumpulkan untuk membuat dataset baru. Dalam dataset baru, setiap instance berhubungan dengan nilai sesungguhnya yang seharusnya diprediksi. Kemudian dataset tersebut digunakan oleh stacking model learner (level-1) untuk memberikan hasil akhir.



Tahapan Metode *Stacking*

Langkah-langkah *Stacking* :

1. Memisahkan data training menjadi dua bagian
2. Melakukan training pada beberapa *base learner* menggunakan bagian data 1
3. Membuat prediksi dengan *base learner* menggunakan bagian data 2
4. Menggunakan hasil prediksi dari langkah 3 sebagai *input* (data train) *learner* kedua

HASIL DAN PEMBAHASAN

1. Desain Penelitian

a. Pendekatan Penelitian

Pendekatan penelitian yang digunakan adalah pendekatan kuantitatif. Pendekatan kuantitatif merupakan pendekatan yang digunakan pada proposal penelitian, proses, hipotesis, analisis data, kesimpulan data sampai kepada penulisannya menggunakan aspek pengukuran, perhitungan, rumus dan kepastian pada data numerik (Musianto, 2002). Pendekatan penelitian kuantitatif ini dapat digunakan dalam meneliti populasi maupun sampel tertentu, pengumpulan data dengan instrumen penelitian, analisis data bersifat numerik atau statistik dengan tujuan menguji hipotesis yang telah ditetapkan sebelumnya. Sehingga dapat ditarik sebuah kesimpulan, pendekatan kuantitatif merupakan suatu pendekatan yang digunakan pada penelitian untuk menguji hipotesis dengan menggunakan pengujian data numerik atau statistik yang akurat.

b. Metode Penelitian

Metode penelitian yang digunakan adalah metode analisis deskriptif. Menurut Nasir (2002 : 61) Dalam Rukajat (2018 : 1) Metode deskriptif adalah suatu metode penelitian status suatu sekelompok manusia, objek, set kondisi, sistem pemikiran maupun kelas peristiwa yang terjadi pada masa sekarang. Tujuan menggunakan metode deskriptif adalah membuat deskriptif maupun gambaran dilengkapi dengan fakta dari hubungan antar peristiwa yang diselidiki secara sistematis dan tepat. Metode deskriptif tidak hanya menerangkan tentang situasi atau kejadian saja, akan tetapi juga menjelaskan mengenai hubungan, menguji, hipotesis serta membuat prediksi dan mendapatkan sebuah kesimpulan dan implikasi dari suatu masalah yang akan diteliti.

c. Teknik Pengambilan Sampel

Teknik pengambilan sampel adalah total sampling dengan menggunakan dataset harga saham dari perusahaan Telkomsel, Indosat dan Smartfren sebanyak 175 data serta dataset kasus Covid-19 per hari di Indonesia sebanyak 256 dataset dengan rentang waktu dimulai dari tanggal 16 Maret 2020 sampai 27 November 2020 (https://finance.yahoo.com/).

Sampel yang diambil adalah 3 data set TLKM.JK, ISAT.JK, FREN.JK dari wesite https://finance.yahoo.com/ yang selanjutnya, setelah ditemukan atur tanggal menjadi 8 tahun sebelumnya. Lalu kemudian download file dalam bentuk csv agar memudahkan dalam proses klasifikasi.

2. Implementasi Penambangan Data

Prediksi harga open saham berdasarkan tingkat kasus COVID-19 ini dikembangkan dengan bahasa Python dengan menggunakan google collab. Prediksi harga open saham ini ditujukan untuk para investor dalam wait and see dalam menentukan keputusan sukses atau tidaknya trading saham, sehingga para investor bisa mengatur strategi harus menjual atau membeli saham.

Implementasi antarmuka(interface) ditampilkan dari screenshot dari google collab yang telah dilakukan.

1. Import library

Import Library

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Metode
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.datasets import load_iris
from sklearn.gaussian_process.kernels import RBF
from sklearn.datasets import make_classification
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

# Klasifikasi
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import StackingClassifier
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.dummy import DummyClassifier
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.linear_model import SGDClassifier
```

Source code ini berisi import metode dan klasifikasi yang akan digunakan untuk mencari metode terbaik dalam memprediksi harga open saham berdasarkan tingkat kasus COVID-19

2. Fungsi

Fungsi

```
def printHasil(data_y, predic_y, judul):
  print(judul, '\n')
  print("Matrik\n")
  print(confusion_matrix(data_y, predic_y))

  print("\nHasil\n")
  print(classification_report(data_y, predic_y))

  print("\nAkurasi\n")
  accuracy = accuracy_score(data_y, predic_y)
  print(accuracy)

  print("\nGrafik\n")
  plt.plot(data_y)
  plt.plot(predic_y)
```

```
def klasifikasiData(data = 'telkom', split_jadi = 50, column_ke = 1,):
  global x_train, x_test, y_train, y_test, data_x, data_y, dset

  data_x = []
  data_y = []

  if data == 'telkom':
    # Telkomsel
    dset = TELKM[:,column_ke]

  elif data == 'indosat':
    # Indosat
    dset = ISAT[:,column_ke]

  else:
    # Freen
    dset = FREEN[:,column_ke]


  # Ambil sejumlah count
  for i in range(0,len(dset)-split):
    data_x.append(dset[i:i+split])
    data_y.append((dset[i+split]>dset[i+split-1])*1)

  x_train, x_test, y_train, y_test = train_test_split(data_x, data_y, random_state = 100, test_size=0.1)

  scaler = StandardScaler()
  scaler.fit(x_train)
  x_train = scaler.transform(x_train)
  x_test = scaler.transform(x_test)
```

Program ini adalah sebuah inisialisasi fungsi awal penyajian data.

3. Klasifikasi KNN

```
[ ] # Telkomsel
    klasifikasiData(data='telkomsel', split_jadi=50, column_ke=1)
    klasifikasiKNeighbor = KNeighborsClassifier(n_neighbors=5)
    klasifikasiKNeighbor.fit(x_train, y_train)
    predic_y = klasifikasiKNeighbor.predict(x_test)
    klasifikasiKNeighbor.predict_proba(x_test)

    printHasil(y_test, predic_y, "Telkomsel")
```

```
    # Indosat
    klasifikasiData(data='indosat', split_jadi=50, column_ke=1)
    klasifikasiKNeighbor = KNeighborsClassifier(n_neighbors=5)
    klasifikasiKNeighbor.fit(x_train, y_train)
    predic_y = klasifikasiKNeighbor.predict(x_test)
    klasifikasiKNeighbor.predict_proba(x_test)

    printHasil(y_test, predic_y, "Indosat")
```

```
[ ] # Freen
    klasifikasiData(data='freen', split_jadi=200, column_ke=1)
    klasifikasiKNeighbor = KNeighborsClassifier(n_neighbors=100)
    klasifikasiKNeighbor.fit(x_train, y_train)
    predic_y = klasifikasiKNeighbor.predict(x_test)
    klasifikasiKNeighbor.predict_proba(x_test)

    printHasil(y_test, predic_y, "Freen")
```

4. Klasifikasi Adaboost

```
[ ] # Telkomsel
    klasifikasiData(data='telkomsel', split_jadi=100, column_ke=1)
    klasifikasiAdaBoost = AdaBoostClassifier(n_estimators=100)
    klasifikasiAdaBoost.fit(x_train, y_train)
    predic_y = klasifikasiAdaBoost.predict(x_test)
    klasifikasiAdaBoost.predict_proba(x_test)

    printHasil(y_test, predic_y, "Telkomsel")
```

```
    # Indosat
    klasifikasiData(data='indosat', split_jadi=100, column_ke=1)
    klasifikasiAdaBoost = AdaBoostClassifier(n_estimators=100)
    klasifikasiAdaBoost.fit(x_train, y_train)
    predic_y = klasifikasiAdaBoost.predict(x_test)
    klasifikasiAdaBoost.predict_proba(x_test)

    printHasil(y_test, predic_y, "Indosat")
```

```
[ ] # Freen
    klasifikasiData(data='freen', split_jadi=100, column_ke=1)
    klasifikasiAdaBoost = AdaBoostClassifier(n_estimators=100)
    klasifikasiAdaBoost.fit(x_train, y_train)
    predic_y = klasifikasiAdaBoost.predict(x_test)
    klasifikasiAdaBoost.predict_proba(x_test)

    printHasil(y_test, predic_y, "Freen")
```

5. Klasifikasi Bagging

```
[ ] # Telkomsel
    klasifikasiData(data='telkomsel', split_jadi=200, column_ke=1)
    klasifikasiBagging = BaggingClassifier(n_estimators=100)
    klasifikasiBagging.fit(x_train, y_train)
    predic_y = klasifikasiBagging.predict(x_test)
    klasifikasiBagging.predict_proba(x_test)

    printHasil(y_test, predic_y, "Telkomsel")
```

```
    # Indosat
    klasifikasiData(data='isat', split_jadi=200, column_ke=1)
    klasifikasiBagging = BaggingClassifier(n_estimators=100)
    klasifikasiBagging.fit(x_train, y_train)
    predic_y = klasifikasiBagging.predict(x_test)
    klasifikasiBagging.predict_proba(x_test)

    printHasil(y_test, predic_y, "Indosat")
```

```
[ ] # Smartfren
    klasifikasiData(data='fren', split_jadi=200, column_ke=1)
    klasifikasiBagging = BaggingClassifier(n_estimators=100)
    klasifikasiBagging.fit(x_train, y_train)
    predic_y = klasifikasiBagging.predict(x_test)
    klasifikasiBagging.predict_proba(x_test)

    printHasil(y_test, predic_y, "fren")
```

6. Klasifikasi Random Forest

```
# Telkomsel
klasifikasiData(data='telkomsel', split_jadi=200, column_ke=1)
klasifikasiRandomForestClassifier = RandomForestClassifier(n_estimators=100)
klasifikasiRandomForestClassifier.fit(x_train, y_train)
predic_y = klasifikasiRandomForestClassifier.predict(x_test)
klasifikasiRandomForestClassifier.predict_proba(x_test)

printHasil(y_test, predic_y, "TLKM")
```

+ Kode + Teks

```
[ ] # Indosat
klasifikasiData(data='isat', split_jadi=200, column_ke=1)
klasifikasiRandomForestClassifier = RandomForestClassifier(n_estimators=100)
klasifikasiRandomForestClassifier.fit(x_train, y_train)
predic_y = klasifikasiRandomForestClassifier.predict(x_test)
klasifikasiRandomForestClassifier.predict_proba(x_test)

printHasil(y_test, predic_y, "ISAT")
```

```
[ ] # Smartfren
klasifikasiData(data='fren', split_jadi=200, column_ke=1)
klasifikasiRandomForestClassifier = RandomForestClassifier(n_estimators=100)
klasifikasiRandomForestClassifier.fit(x_train, y_train)
predic_y = klasifikasiRandomForestClassifier.predict(x_test)
klasifikasiRandomForestClassifier.predict_proba(x_test)

printHasil(y_test, predic_y, "FREN")
```

7. Klasifikasi Extra Trees

```
# Telkomsel
klasifikasiData(data='TLKM', split_jadi=200, column_ke=1)
klasifikasiExtraTrees = ExtraTreesClassifier(n_estimators=100)
klasifikasiExtraTrees.fit(x_train, y_train)
predic_y = klasifikasiExtraTrees.predict(x_test)
klasifikasiExtraTrees.predict_proba(x_test)

printHasil(y_test, predic_y, "TLKM")
```

```
[ ] # Indosat
klasifikasiData(data='isat', split_jadi=200, column_ke=1)
klasifikasiExtraTrees = ExtraTreesClassifier(n_estimators=100)
klasifikasiExtraTrees.fit(x_train, y_train)
predic_y = klasifikasiExtraTrees.predict(x_test)
klasifikasiExtraTrees.predict_proba(x_test)

printHasil(y_test, predic_y, "ISAT")
```

```
[ ] # Smartfren
klasifikasiData(data='fren', split_jadi=200, column_ke=1)
klasifikasiExtraTrees = ExtraTreesClassifier(n_estimators=100)
klasifikasiExtraTrees.fit(x_train, y_train)
predic_y = klasifikasiExtraTrees.predict(x_test)
klasifikasiExtraTrees.predict_proba(x_test)

printHasil(y_test, predic_y, "FREN")
```

8. Klasifikasi Gradient Boosting

```
[ ] # Telkomsel
klasifikasiData(data='tlkm', split_jadi=200, column_ke=1)
klasifikasiGradientBoosting = GradientBoostingClassifier(n_estimators=100)
klasifikasiGradientBoosting.fit(x_train, y_train)
predic_y = klasifikasiGradientBoosting.predict(x_test)
klasifikasiGradientBoosting.predict_proba(x_test)

printHasil(y_test, predic_y, "TLKM")
```

```
[ ] # Indosat
klasifikasiData(data='isat', split_jadi=200, column_ke=1)
klasifikasiGradientBoosting = GradientBoostingClassifier(n_estimators=100)
klasifikasiGradientBoosting.fit(x_train, y_train)
predic_y = klasifikasiGradientBoosting.predict(x_test)
klasifikasiGradientBoosting.predict_proba(x_test)

printHasil(y_test, predic_y, "ISAT")
```

```
[ ] # Smartfren
klasifikasiData(data='fren', split_jadi=200, column_ke=1)
klasifikasiGradientBoosting = GradientBoostingClassifier(n_estimators=100)
klasifikasiGradientBoosting.fit(x_train, y_train)
predic_y = klasifikasiGradientBoosting.predict(x_test)
klasifikasiGradientBoosting.predict_proba(x_test)

printHasil(y_test, predic_y, "FREN")
```

9. Klasifikasi Stacking

```
[ ] # Telkomsel
    klasifikasiData(data='tlkm', split_jadi=200, column_ke=1)
    # Ngestack randomforesclassifier
    estimators = [('rf', RandomForestClassifier(n_estimators=100, random_state=42))]

    klasifikasiStackingClassifier = StackingClassifier(estimators=estimators)
    klasifikasiStackingClassifier.fit(x_train, y_train)
    predic_y = klasifikasiStackingClassifier.predict(x_test)
    klasifikasiStackingClassifier.predict_proba(x_test)

    printHasil(y_test, predic_y, "TLKM")
```

```
    # Indosat
    klasifikasiData(data='isat', split_jadi=200, column_ke=1)
    # Ngestack randomforesclassifier
    estimators = [('rf', RandomForestClassifier(n_estimators=100, random_state=42))]

    klasifikasiStackingClassifier = StackingClassifier(estimators=estimators)
    klasifikasiStackingClassifier.fit(x_train, y_train)
    predic_y = klasifikasiStackingClassifier.predict(x_test)
    klasifikasiStackingClassifier.predict_proba(x_test)

    printHasil(y_test, predic_y, "ISAT")
```

10. Klasifikasi Gaussian Process

```
[ ] # Telkomsel
    klasifikasiData(data='tlkm', split_jadi=200, column_ke=1)
    klasifikasiGaussianProcessClassifier = GradientBoostingClassifier(n_estimators=100)
    klasifikasiGaussianProcessClassifier.fit(x_train, y_train)
    predic_y = klasifikasiGaussianProcessClassifier.predict(x_test)
    klasifikasiGaussianProcessClassifier.predict_proba(x_test)

    printHasil(y_test, predic_y, "TLKM")
```

```
[ ] # Smartfren
    klasifikasiData(data='fren', split_jadi=200, column_ke=1)
    klasifikasiGaussianProcessClassifier = GradientBoostingClassifier(n_estimators=100)
    klasifikasiGaussianProcessClassifier.fit(x_train, y_train)
    predic_y = klasifikasiGaussianProcessClassifier.predict(x_test)
    klasifikasiGaussianProcessClassifier.predict_proba(x_test)

    printHasil(y_test, predic_y, "FREN")
```

```
[ ] # Indosat
    klasifikasiData(data='isat', split_jadi=200, column_ke=1)
    klasifikasiGaussianProcessClassifier = GradientBoostingClassifier(n_estimators=100)
    klasifikasiGaussianProcessClassifier.fit(x_train, y_train)
    predic_y = klasifikasiGaussianProcessClassifier.predict(x_test)
    klasifikasiGaussianProcessClassifier.predict_proba(x_test)

    printHasil(y_test, predic_y, "ISAT")
```

11. Klasifikasi Dummy

```
# Telkomsel
klasifikasiData(data='telkomsel', split_jadi=200, column_ke=1)
klasifikasiDummy = DummyClassifier(strategy="most_frequent")
klasifikasiDummy.fit(x_train, y_train)
DummyClassifier(strategy='most_frequent')
predic_y = klasifikasiDummy.predict(x_test)
klasifikasiDummy.predict_proba(x_test)

printHasil(y_test, predic_y, "Telkomsel")
```

```
[ ] # Indosat
klasifikasiData(data='isat', split_jadi=200, column_ke=1)
klasifikasiDummy = DummyClassifier(strategy="most_frequent")
klasifikasiDummy.fit(x_train, y_train)
DummyClassifier(strategy='most_frequent')
predic_y = klasifikasiDummy.predict(x_test)
klasifikasiDummy.predict_proba(x_test)

printHasil(y_test, predic_y, "ISAT")
```

```
[ ] # Smartfren
klasifikasiData(data='fren', split_jadi=200, column_ke=1)
klasifikasiDummy = DummyClassifier(strategy="most_frequent")
klasifikasiDummy.fit(x_train, y_train)
DummyClassifier(strategy='most_frequent')
predic_y = klasifikasiDummy.predict(x_test)
klasifikasiDummy.predict_proba(x_test)

printHasil(y_test, predic_y, "FREN")
```

12. Klasifikasi SGD

```
# Telkomsel
klasifikasiData(data='tlkm', split_jadi=200, column_ke=1)
klasifikasiSGDClassifier = make_pipeline(StandardScaler(),SGDClassifier(max_iter=100, tol=1e-3))
klasifikasiSGDClassifier.fit(x_train, y_train)
predic_y = klasifikasiSGDClassifier.predict(x_test)

printHasil(y_test, predic_y, "TLKM")
```

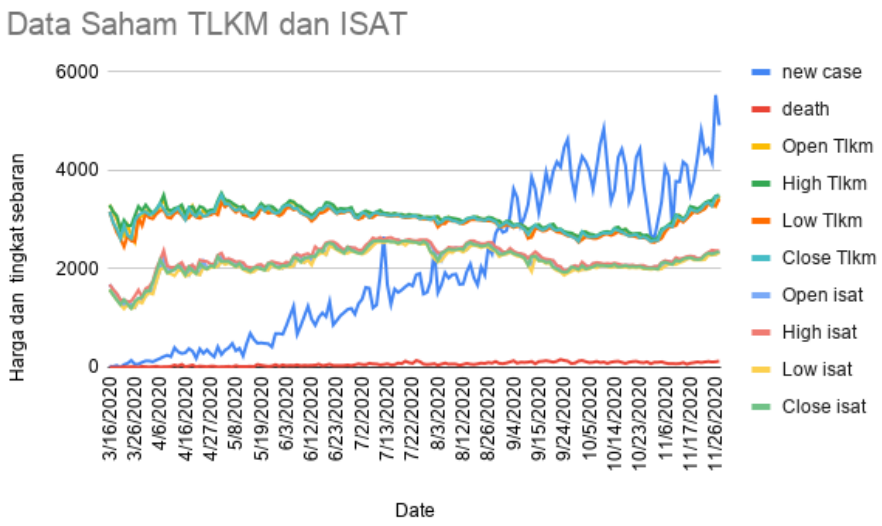
```
[ ] # Indosat
klasifikasiData(data='isat', split_jadi=200, column_ke=1)
klasifikasiSGDClassifier = make_pipeline(StandardScaler(),SGDClassifier(max_iter=100, tol=1e-3))
klasifikasiSGDClassifier.fit(x_train, y_train)
predic_y = klasifikasiSGDClassifier.predict(x_test)

printHasil(y_test, predic_y, "ISAT")
```

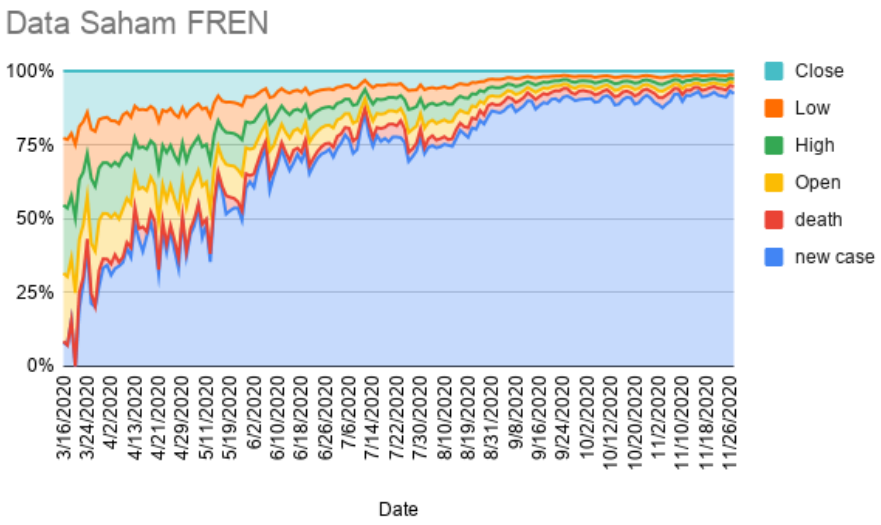
```
[ ] # Smartfren
klasifikasiData(data='fren', split_jadi=200, column_ke=1)
klasifikasiSGDClassifier = make_pipeline(StandardScaler(),SGDClassifier(max_iter=100, tol=1e-3))
klasifikasiSGDClassifier.fit(x_train, y_train)
predic_y = klasifikasiSGDClassifier.predict(x_test)

printHasil(y_test, predic_y, "FREN")
```

3. Analisis Penambangan Data



Gambar 1 TLKM.JK dan ISAT.JK



Gambar 2 FREN.JK

Persebaran Covid-19 di Indonesia semakin hari semakin meningkat, hal ini berimbas pada harga open saham. Pada tanggal 16 Maret 2020 harga saham mengalami *Dead Cat Bounce* karena para pemilik saham panik akibat kedatangan Covid-19 yang tidak terduga sehingga mengakibatkan lumpuhnya berbagai sektor di Indonesia. Selang 2 minggu setelah Covid-19 menyebar di Indonesia harga open saham sempat mengalami kenaikan dikarenakan masyarakat sudah membiasakan diri dari Covid-19 agar seluruh sektor kembali stabil.

Walaupun harga open saham dilihat dari grafik diatas setelah tanggal 2 April mengalami fluktuasi yang stabil, akan tetapi disarankan untuk pemilik saham melakukan strategi wait and see terlebih dahulu sebelum menjual ataupun membeli saham. Strategi wait and see ini bertujuan untuk melihat perkembangan harga saham kedepannya.

Pertanggal 26 November 2020, saham FREN ditutup turun -4,11% di harga Rp. 70, kemudian untuk saham ISAT ditutup naik 1,30% di harga Rp. 2290, dan untuk saham TLKM ditutup naik 5,79% di harga Rp. 3470. Dua dari ketiga dataset mengalami kenaikan. melihat dari tingkat kenaikan dan penurunan 3 dataset pemilik saham bisa menjual jika sudah memenuhi target jual harga saham dan profit yang didapatkannya atau tetap meng hold sahamnya ataupun membeli saham karena saat ini saham FREN banyak diincar oleh para Investor.

KESIMPULAN

Masuknya Covid-19 ke Indonesia sempat membuat para investor panic dan menjual sahamnya dengan harga murah sehingga mengakibatkan harga saham turun, walau sempat mengalami kenaikan namun setelah dianalisis dan diprediksi menggunakan 10 metode klasifikasi penambangan data, metode terbaik penambangan data dengan akurasi tertinggi adalah DummyClassifier dengan akurasi 71% untuk dataset Telkom, 71% untuk dataset Indosat, dan 76% untuk dataset Smartfreen, dan dapat disimpulkan bahwa untuk prediksi harga open saham untuk ke 3 dataset adalah cenderung turun.

Untuk dimasa mendatang ketika ada wabah sejenis Covid-19 melanda di Indonesia, berdasarkan analisis dan klasifikasi yang telah kita teliti kita mendapatkan pola fluktuasi saham saat Covid-19 melanda, disarankan untuk para investor jangan dulu menjual saham saat bulan pertama wabah melanda gunakan strategi wait and see pada saat dead cat bounce saham berlangsung, karena ketika saham menembus Support pasti harga saham akan meningkat kembali, dan ketika saham menembus Resistance pasti harga saham akan menurun kembali.

LAMPIRAN

1. Link akurasi tiap metode

   https://drive.google.com/drive/folders/1QSuyvXfywQ3ROcGpZF2_deepaaALooRO

2. Hasil kesimpulan Tiap Metode

Metode	Kesimpulan			Akurasi			Akurasi(%)		
	Telkom	Indosat	Smartfren	Telkom	Indosat	Smartfren	Telkom	Indosat	Smartfren
KNN	Turun	Turun	Turun	0.47	0.47	0.76	47%	47%	76%
Adaboost	Turun	Turun	Turun	0.47	0.41	0.71	47%	41%	71%
Bagging	Turun	Turun	Turun	0.35	0.65	0.71	35%	65%	71%
Random Forest	Turun	Turun	Turun	0.41	0.59	0.65	41%	59%	65%
Extra Trees	Turun	Turun	Turun	0.47	0.59	0.65	47%	59%	65%
Gradient Boosting	Turun	Turun	Turun	0.35	0.53	0.65	35%	53%	65%
Stacking	Turun	Turun	Turun	0.41	0.59	0.76	41%	59%	76%
Gaussian Process	Turun	Turun	Turun	0.35	0.53	0.65	35%	53%	65%
Dummy	Turun	Turun	Turun	0.71	0.71	0.76	71%	71%	76%
SGD	Turun	Turun	Turun	0.35	0.65	0.35	35%	65%	35%

DAFTAR PUSTAKA

[1] M. Mulyadi, "Penelitian Kuantitatif Dan Kualitatif Serta Pemikiran Dasar Menggabungkannya," *J. Stud. Komun. dan Media*, vol. 15, no. 1, p. 128, 2013, doi: 10.31445/jskm.2011.150106.

[2] D. Kartikaningsih, Nugraha, and Sugiyanto, "Pengaruh Nilai Tukar Terhadap Harga Saham Sektor Infrastruktur Pada Masa Pandemi Covid-19," *J. Akunt. dan Keuang.*, vol. 3, no. 1, pp. 53–60, 2020.

[3] E. Purnaningrum and V. Ariyanti, "Pemanfaatan Google Trends Untuk Mengetahui Intervensi Pandemi Covid-19 Terhadap Pasar Saham Di Indonesia," *Jurnal.Unipasby.Ac.Id*, vol. 25, no. 1411, pp. 93–101, 2020, [Online]. Available: https://finance.yahoo.com/.

[4] L. S. Musianto, "Perbedaan Pendekatan Kuantitatif Dengan Pendekatan Kualitatif Dalam Metode Penelitian," *J. Manaj. dan Kewirausahaan*, vol. 4, no. 2, pp. 123–136, 2002, doi: 10.9744/jmk.4.2.pp.123-136.

[5] A. Saleh and E. Boj, "Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID- 19 . The COVID-19 resource centre is hosted on Elsevier Connect , the company ' s public news and information website . Elsevier hereby grants permission to make all its COVID-19-related research that is available on the COVID-19 resource centre - including this research content - immediately available in PubMed Central and other publicly funded repositories , such as the WHO COVID database with rights for unrestricted research re-use and analyses in any form or by any means with acknowledgement of the original source . These permissions are granted for free by Elsevier for as long as the COVID-19 resource centre remains active . SutteARIMA : Short-term forecasting method , a case : Covid-19 and stock market in Spain," no. January, 2020.