# Perbandingan Algoritma *Long Short Term Memory* (LSTM) dan *Multilayer Perceptron* (MLP) dalam Memprediksi Nilai Indeks Harga Saham Gabungan (IHSG) di Indonesia

Untuk Memenuhi Tugas Mata Kuliah Jaringan Syaraf Tiruan

Dosen Pengampu : Dr. Winita Sulandari, S.Si., M.Si.


Disusun oleh :

| | |
|---|---|
| Fadia Mulyarti | (M0718018) |
| Fida Mardliyah ` | (M0718023) |
| Fitri Azizah | (M0718024) |
| Rida Afifatama Hidayat | (M0718046) |

**PROGRAM STUDI STATISTIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS SEBELAS MARET**

**SURAKARTA**

**2021**

## DAFTAR ISI

# BAB I
# PENDAHULUAN

## 1.1. Latar Belakang

Indonesia termasuk salah satu negara yang masih tergolong menjadi negara berkembang. Ciri – ciri yang menunjukkan negara berkembang adalah saat perekonomian suatu negara masih bergantung pada perekonomian luar. Hal ini mengakibatkan jika terjadi guncangan dalam perekonomian global maka Indonesia akan ikut merasakan dampaknya, baik dampak positif maupun dampak negatif. Dampak negatif yang paling parah adalah dapat membuat negara Indonesia mengalami krisis ekonomi yang dapat menghancurkan sebuah negara. Salah satu indicator penting dalam perekonomian Indonesia adalah Indeks Harga Saham Gabungan (IHSG). IHSG atau dalam bahasa inggrinya *Indonesia Composite Index* (ICI) menurut Tandelilin (2010) adalah indeks yang secara umum menunjukkan pergerakan harga saham yang menjadi acuan mengenai perkembangan kegiatan di pasar modal yang tercatat di bursa efek. Index saham ini bisa digunakan untuk menilai situasi pasar secara umum atau mengukur apakah harga saham mengalami kenaikan atau penurunan. ISHG juga melibatkan seluruh harga saham yang tercatat di bursa.

Ada banyak metode yang dapat digunakan untuk memprediksi IHSG, salah satunya adalah *Artificial Neural Network* atau Jaringan Syaraf Tiruan, Terdapat banyak algoritma pembelajaran dalam *neural network*, salah satunya adalah *backpropagation*. Algoritma ini mampu untuk memperbaiki bobot pada lapisan tersembunyi (*hidden layer*) (Purnomo dan Kurniawan, 2006). Dari penelitian sebelumnya, Hanifah dan Fahturohman (2017) melakukan prediksi kebangkrutan Bank Islami di Indonesia menggunakan Neural Network Backpropagation. Hasil dari pelatihan algoritma pada data tersebut menghasilkan akurasi pada data latihnya adalah 100% yang artinya model tersebut dapat memprediksi kebangkrutan dengan baik. Pada penelitian ini, akan dilakukan peramalan nilai IHSG (Indeks Harga Saham Gabungan) di Indonesia

menggunakan dua algoritma *Neural Network* yaitu Algoritma *Long Short Term Memory* (LSTM) dan *Multilayer Perceptron* (MLP).

## 1.2. Rumusan Masalah

Berdasarkan uraian pada latar belakang diatas, permasalahan yang dibahas pada penelitian ini adalah sebagai berikut:

a. Bagaimana hasil peramalan data latih dari Indeks Harga Saham Gabungan (IHSG) Indonesia menggunakan Algoritma *Long Short Term Memory* ?
b. Bagaimana hasil peramalan data uji dari Indeks Harga Saham Gabungan (IHSG) Indonesia menggunakan Algoritma *Multilayer Perceptron* ?
c. Bagaimana perbandingan hasil klasifikasi menggunakan Algoritma *Long Short Term Memory* dan *Multilayer Perceptron* ?

## 1.3. Tujuan Penelitian

Berdasarkan rumusan masalah diatas, didapatkan tujuan penelitian adalah sebagai berikut:

a. Mengetahui hasil peramalan data latih dari Indeks Harga Saham Gabungan (IHSG) Indonesia menggunakan Algoritma *Long Short Term Memory* ?
b. Mengetahui hasil peramalan data uji dari Indeks Harga Saham Gabungan (IHSG) Indonesia menggunakan Algoritma *Multilayer Perceptron* ?
c. Mengetahui perbandingan hasil klasifikasi menggunakan Algoritma *Long Short Term Memory* dan *Multilayer Perceptron* ?

# BAB II
# LANDASAN TEORI

## 2.1. Indeks Harga Saham Gabungan (IHSG)

Indeks Harga Saham (IHS) adalah simpulan dari pengaruh simultan dan kompleks dari macam-macam variabel yang berpengaruh, terutama menengai kegiatan ekonomi. IHS sekarang tidak hanya menampung kejadian ekonomi, namun juga menampung kejadian mengenai politik, sosial, dan keamanan. Dengan begitu, IHS dapat digunakan sebagai dasar melakukan analisis statistik berdasarkan kondisi pasar terakhir.

IHSG merupakan rangkaian informasi sejarah atau historis tentang suatu pergerakan harga saham secara gabungan, sampai tanggal tertentu serta mencerminkan suatu nilai yang bisa berfungsi sebagai pengukur kinerja suatu saham gabungan di bursa efek (Muis, 2008).

Secara umum, IHSG menunjukkan pergerakan harga saham yang tercatat di bursa efek yang dapat menjadi dasar mengenai perkembangan kegiatan ekonomi di pasar modal. Penggunaan nilai IHSG untuk menilai kondisi pasar adalah untuk mengukur harga saham ketika mengalami kenaikan atau penurunan. IHSG juga melibatkan seluruh harga saham yang tercatat di bursa.

## 2.2. *Long Short Term Memory*

Arsitektur *Recurrent Neural Network* (RNN) mengalami evolusi yang dinamakan *Long Short Term Memory* (LSTM), di mana pertama kali ditemukan oleh Hochreiter & Schmidhuber (1997). Para peneliti berusaha mengembangkan arsitektur LSTM di berbagai bidang ,diantaranya dalam bidang *forecasting* dan *speech recognition* hingga saat penelitian ini dilakukan . RNN mempunyai memori jangka pendek, sehingga tidak mampu membawa informasi yang sebelumnya diperoleh ke proses selanjutnya. Solusi dari hal tersebut adalah menggunakan algoritma LSTM, LSTM adalah unit khusus RNN yang dirancang untuk mengatasi masalah ketergantungan jangka panjang karena LSTM

mempunyai *cell states* yakni mekanisme internal dan *gates* yang dapat mengatur memori dalam setiap masukannya. Ada empat *gates* yaitu *input gate, forget gate, cell gate* dan *output gate* (Roondiwala, 2017) .

### 2.3. *Multilayer Perceptron*

Multilayer perceptron (MLP) merupakan pemodelan dalam teknologi JST yang mempunyai ciri khusus yaitu mempunyai nilai bobot (*weight*) yang lebih baik dari pada algoritma model yang lain, sehingga dapat melakukan pengelompokkan yang lebih akurat. Seperti namanya, jenis jaringan *multilayer perceptron* ini merupakan hasil generalisasi dari arsitektur perceptron dengan satu layer, sehingga mempunyai beberapa lapisan atau *hidden layer*, yang letaknya di antara input layer dan output layer. Jaringan ini umunya terdiri dari beberapa unit neuron sebagai lapisan input-an, satu atau lebih lapisan simpul-simpul neuron komputasi *hidden layer*, dan satu buah lapisan simpul-simpul neuron komputasi lapisan *output*. Untuk menghasilkan hasil output, pada MLP unit-unit diatur dalam lapisan topologi *feed-forward* yang disebut *Feed Forward Neural Network*. Fungsi standar sigmoid digunakan ketika jumlah pembobotan dari banyaknya *input* dan bias di-*input*-kan ke *activation level* melalui fungsi transfer (Venkatesan & Anitha, 2006).

### 2.4. Fungsi Aktivasi

Fungsi aktivasi terdapat pada setiap layer jaringan syaraf tiruan. untuk membawa input menuju output yang diinginkan, fungsi ini adalah fungsi umum yang akan digunakan. Fungsi aktivasi ini yang akan menentukan besarnya bobot. Penggunaan fungsi ini tergantung pada kebutuhan dan desired output (Fausett, 1994).

Terdapat beberapa fungsi aktivasi yang biasa digunakan dalam jaringan syaraf tiruan, yang digunakan dalam penelitian ini antara lain :

a. Rectified Linear Unit (ReLU)

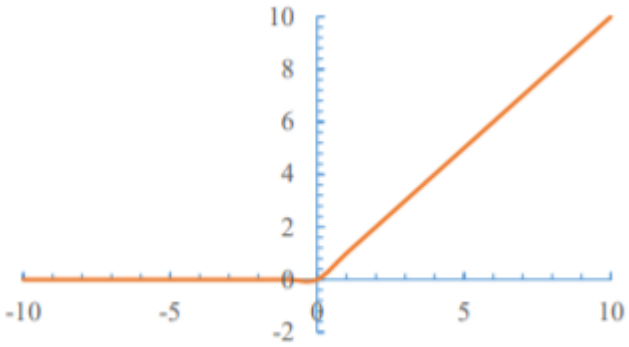
Pada tahun 2010 G.Hinton dan V.Nair memperkenalkan Fungsi ReLU. Fungsi ini dapat digunakan dalam kontex convolutional neural networks. Fungsi ReLU dinyatakan dengan persamaan

$$y(u) = \max(0, u)$$

atau

$$y(u) = \begin{cases} u, & \text{if } u \geq 0 \\ 0, & \text{if } u < 0 \end{cases}$$

output yang dihasilkan dari fungsi ReLU akan membentuk grafik yang ditunjukan pada Gambar 2.2.



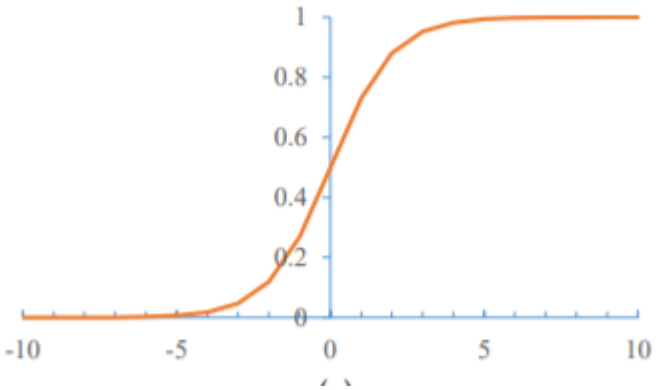

Gambar 2.2 Grafik Fungsi Aktivasi ReLU

b. Fungsi Sigmoid

Fungsi sigmoid adalah fungsi yang digunakan ketika output yang ingin diperoleh bersifat nonlinear. Rumus fungsi sigmoid adalah sebagai berikut,

$$y(u) = \frac{1}{1 + e^{(-u)}}$$

Keluaran yang dihasilkan dari fungsi sigmoid akan membentuk grafik yang ditunjukan pada Gambar 2.3.




Gambar 2.3 Grafik Fungsi Aktivasi Sigmoid

## 2.5. *Adaptive Moment Estimation* Optimization (Adam)

Adam atau optimasi *Adaptive Moment Estimation* merupakan perluasan dari *Stochastic Gradient Descent* (SGD) yang telah digunakan sebagai pembelajaran yang mendalam dalam *computer vision* dan *natural language processing*. Algoritma optimasi Adam pertama kali diperkenalkan oleh Kingma and Ba (2014). Adam dikembangkan dengan memanfaatkan kelebihan dari algoritma *Adaptive Gradient* (AdaGrad) dan *Root Mean Square Propagation* (RMSProp). Selain mengadaptasi tingkat pembelajaran parameter menggunakan rata-rata pertama (*mean*) seperti dalam RMSProp, Adam juga menggunakan rata-rata kedua dari gradien (*varians uncentered*). Algoritma menghitung rata-rata pergerakan *exponential* dari gradien dan gradien kuadratnya, serta parameter beta 1 dan beta 2 yang mengontrol tingkat peluruhan rata-rata pergerakan.

## 2.6. *Root Mean Square Error* (RMSE)

Perhitungan *Root Mean Square Error* (RMSE) digunakan untuk melihat performansi suatu sistem. RMSE yaitu jumlah dari kesalahan kuadrat atau selisih antara nilai aktual dengan nilai peramalan yang telah didapatkan. Model terbaik ditentukan dengan melihat nilai RMSE terkecil (Suyanto, 2018). Rumus RMSE dapat ditulis seperti dibawah ini:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_{i\square} - y_{i\square})^2}$$

Keterangan :

N = banyaknya data

$f_{i\square}$ = nilai *output* dari model

$y_i$ = nilai data aktual

# BAB III
# METODOLOGI PENELITIAN

## 3.1. Sumber Data

Penelitian ini menggunakan satu variabel yaitu Indeks Harga Saham Gabungan (IHSG) Indonesia. Data IHSG yang digunakan adalah data mulai bulan Januari 1990 sampai Februari 2021. Data ini diperoleh dari website *International Financial Statistics* (data.imf.org).

## 3.2. Tahapan Analisis

Dalam menjalankan algoritma analisis, digunakan *software* pendukung yaitu *Google Colaboratory* dan *Jupyter Notebook* dengan bahasa pemrograman Python. Tahapan analisis pada penelitian terdiri dari penentuan dan pengidentifikasi masalah yang diteliti, pengumpulan data penelitian, *preprocessing* data, pembuatan model *Long Short Term Memory* dan *Multilayer Perceptron*, pelatihan serta pengujian model. Langkah analisis yang dilakukan pada penelitian ini adalah sebagai berikut :

1. Mengentukan variabel dan mengumpulkan data penelitian yang digunakan.
2. Melakukan analisis eksplorasi data.
3. Melakukan *preprocessing* data.
    a. Melakukan pengecekan *missing value*
    b. Menentukan variabel input dan target
    c. Melakukan normalisasi data menggunakan *MinMax Scaling*
    d. Melakukan *partisi* data menjadi data latih data uji dengan perbandingan 8:2.
4. Perancangan model dengan algoritma *Long Short Term Memory* (LSTM) .
5. Pengujian model LSTM yang didapatkan pada data uji
6. Perancangan model dengan algoritma *Multilayer Perceptron* (MLP).
7. Pengujian model MLP yang didapatkan pada data uji.

8. Menentukan model terbaik.

# BAB IV
# HASIL DAN PEMBAHASAN

## 4.1. Eksplorasi Data IHSG

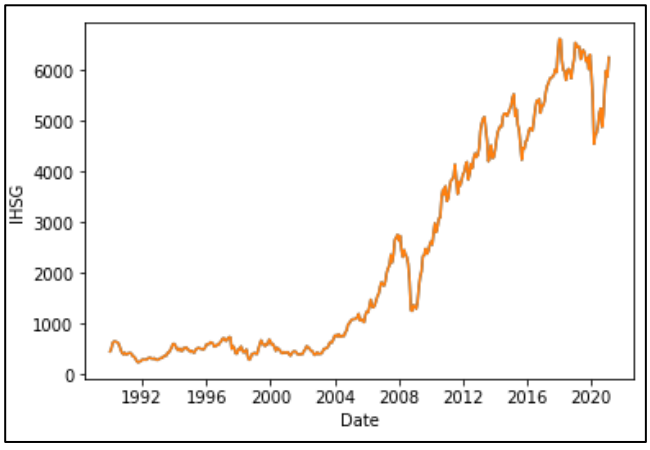
Data Indeks Harga Saham Gabungan (IHSG) Indonesia yang digunakan pada penelitian ini adalah IHSG mulai bulan Januari 1990 hingga bulan Februari 2021 dengan total sebanyak 374 data. Deksripsi data IHSG ditampilkan pada Tabel 4.1

Tabel 4.1 Statistika Deskriptif

| Statistika Deskriptif | Nilai |
|---|---|
| Minimum | 226.68 |
| Rata-Rata | 2278.636 |
| Median | 1080.17 |
| Maksimum | 6605.63 |
| Standar Deviasi | 2118.064 |

Berdasarkan Tabel 4.1 Didapatkan nilai IHSG terendah adalah 226.68 pada bulan Oktober 1991 dan tertinggi sebesar 6605.63 pada Januari 2018.

Selanjutnya adalah berupa gambaran plot data runtun waktu untuk mengetahui karakteristik data pada waktu-waktu tertentu seperti terlihat pada Gambar 4.1.




Gambar 4.1 Plot *Time Series* IHSG

Plot runtun waktu data IHSG pada Gambar 4.1 diketahui bahwa nilai IHSG relative berfluktuasi dari tahun 1990, dan mulai mengalami kenaikan mulai tahun 2004 dengan pernah turun tajam pada Oktober 2008 dari 1832,51 menjadi 1256,70 dan pada Maret 2020 sebesar 4538,93 dari 5452,7.

## 4.2. Preprocessing Data

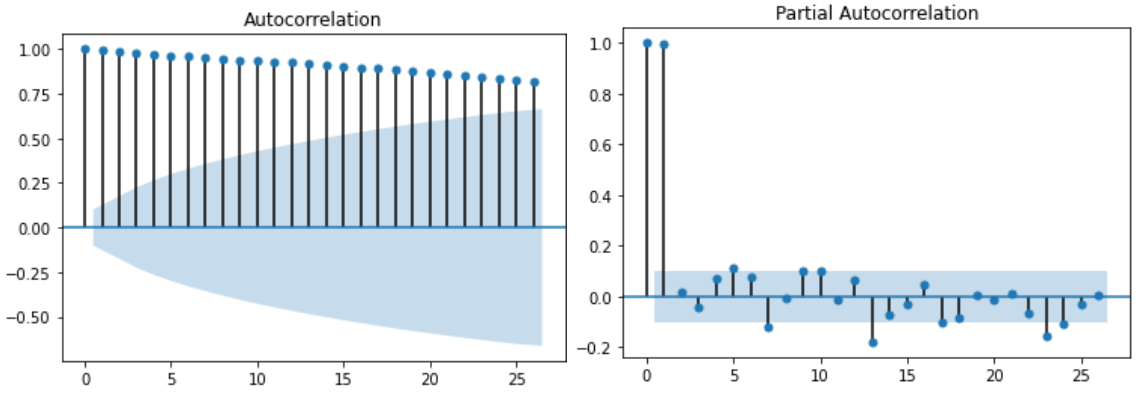
Sebelum melakukan analisis lebih lanjut, diperlukan proses *preprocessing data* agar data yang digunakan sesuai dengan format untuk analisis. Proses *preprocessing* yang dilakukan pada penelitian ini yakni pengecekan *missing value*, standarisasi data, dan partisi data

1. Melakukan pengecekan *missing value*

   Tidak terdapat *missing value* yang ditemukan pada data penelitian, sehingga tidak dilakukan penanganan *missing value* lebih lanjut.

2. Penentuan Input dan Target

   Pada penelitian ini target (Y) adalah nilai IHSG pada waktu ke-t, sedangkan penentuan input berdasarkan jumlah lag keluar dari batas signifikan pada plot ACF dan PACF. Berikut merupakan plot ACF dan PACF dari data IHSG.




Gambar 4.2 Plot ACF dan PACF Data IHSG

Terlihat pada plot PACF setelah lag kedua, lag sudah masuk pada batas signifikansi, Karena lag yang keluar dari batas signifikansi di awal berjumlah 1, maka look back atau input yang digunakan adalah 1 input. Input ($X_1$) dengan Xi untuk i=1,2,…,n berisi data ke t-1

3. Melakukan normalisasi data

   Data IHSG diketahui bahwa data memiliki *range* yang berbeda-beda, hal ini akan menyebabkan model menjadi sulit menemukan pola yang tepat. Normalisasi data menggunakan *MinMax scaling* untuk membuat *range* data menjadi seragam dengan rentang 0-1.

*4.* Melakukan *partisi data*

   Selanjutnya dilakukan pembagian data dengan perbandingan 8:2 menjadi data latih dan data uji. Januari 1990 sampai November 2014 menjadi data latih dan data IHSG daru Desember 2014 sampai Februari 2021 menjadi data uji. Data Pada analisis, model dibangun dan dilatih menggunakan data latih sedangkan evalusi model menggunakan data uji.

### 4.3. Model *Long Short Term Memory*

Pembangunan arsitektur LSTM yang optimal dilakukan dengan menggunakan sistem *trial* dan *error* dalam penentuan parameter yang terbaik. Tahapan awal model dibangun menggunakan fungsi *sequential*. Model dibentuk dengan menggunakan parameter-parameter model LSTM yang ditunjukkan pada Tabel 4.2 sebagai berikut.

Tabel 4.2 Parameter Model LSTM

| Parameter | Jumlah |
|---|---|
| Jumlah Layer | 4 |
| *Hidden Layer* | 1 layer, 32 neuron |
| Jumlah input *(LSTM Cell*) | 64 neuron |
| *Output Layer* | 1 neuron |
| Jumlah epoch | 100 |

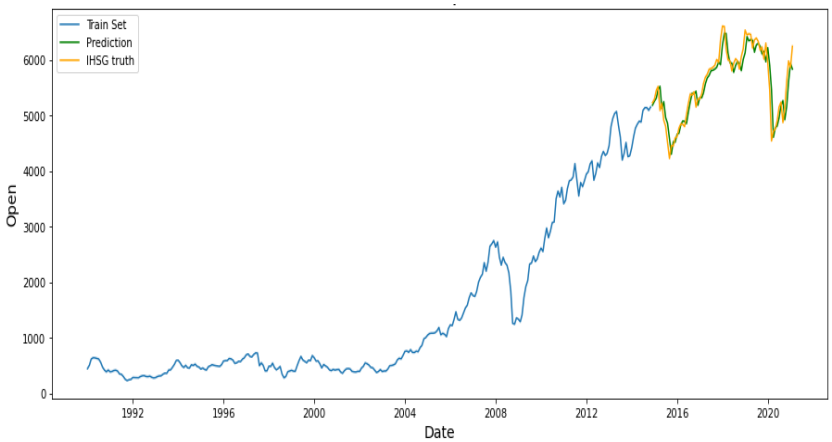
*Loss function* yang digunakan adalah RMSE dengan *optimizer* Adam. Pada penelitian ini akan dilakukan analisa dampak tiap parameter yang ditentukan terhadap hasil akurasi model. Parameter yang akan diuji adalah banyaknya neuron pada *hidden layer*. Jumlah neuron *hidden layer* yang akan diuji coba adalah 32, 64, dan 128 neuron.

Tabel 4.3 Hasil Uji Coba Model LSTM

| Uji Coba | Jumlah Neuron *Hidden Layer* | RMSE Data Latih | RMSE Data Uji |
|---|---|---|---|
| 1. | 32 | 107,859 | 226,864 |
| 2. | 64 | 211.836 | 352.056 |
| 3. | 128 | 111.243 | 228.964 |

Berdasarkan Tabel 4.3 menunjukkan bahwa jumlah neuron 128 pada *hidden layer* pada model LSTM menghasilkan RMSE lebih kecil daripada dengan neuron 64 dan 128. Oleh karena itu, dipilih *hidden layer* dengan neuron 32 dalam pemodelan LSTM untuk meramlakan IHSG.

Perbandingan hasil prediksi nilai IHSG menggunakan metode LSTM dengan data aktual dapat dilihat pada gambar 4.3




Gambar 4.3 Perbandingan Data Aktual dengan Hasil Prediksi Model LSTM

### 4.4. Model *Multilayer Perceptron* (MLP)

Tahap awal model MLP dibangun dengan menggunakan fungsi *sequential*. Pada model MLP ini akan menggunakan 1 *input layer*, 3 *hidden layer*, dam 1 *output layer*. Model dibentuk menggunakan library *tensorflow* dengan parameter MLP sebagai berikut :

Tabel 4.4 Parameter Model MLP

| Parameter | Jumlah |
|---|---|
| Jumlah Layer | 5 |
| *Input Layer* | 1 layer |
| *Hidden Layer* 1 | 32 neuron |
| *Hidden Layer* 2 | 16 neuron |
| *Hidden Layer* 3 | 8 neuron |
| *Output Layer* | 1 neuron |
| Jumlah epoch | 100 |

Pada *hidden layer* menggunakan fungsi aktivasi reLU, sedangkan *output layer* menggunakan fungsi aktivasi linear. *Loss function* yang digunakan adalah RMSE dengan *optimizer* Adam. Pada pemodelan MLP ini akan dilakukan analisa dampak tiap parameter yang ditentukan terhadap hasil akurasi model. Parameter yang akan diuji adalah jumlah neuron pada *hidden layer*.

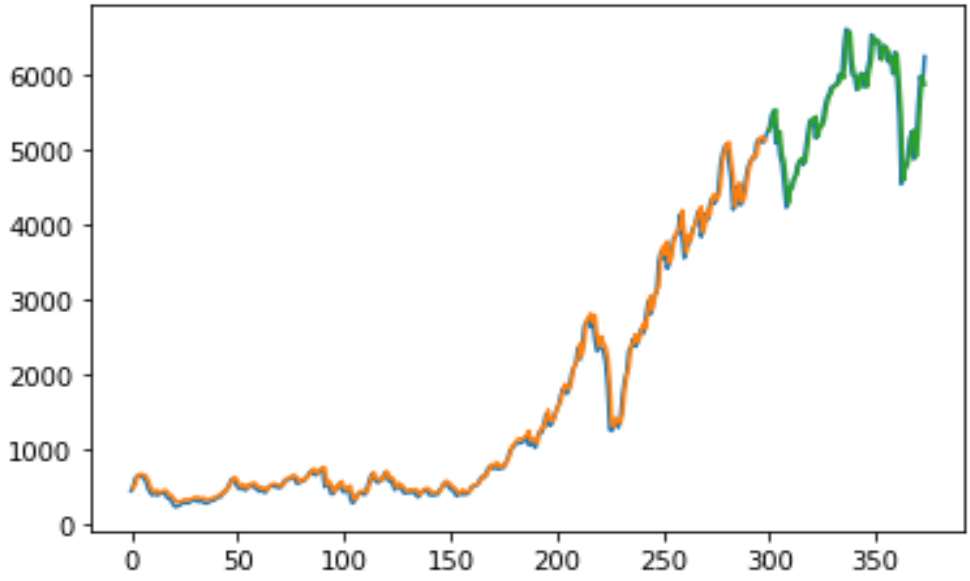
Tabel 4.5 Hasil Uji Coba Model MLP

| Uji Coba | Jumlah Neuron *Hidden Layer* | RMSE Data Latih | RMSE Data Uji |
|---|---|---|---|
| 1. | (32,16,8) | 105,834 | 244,056 |
| 2. | (54,32,16) | 115,994 | 228,172 |
| 3. | (128,54,36) | 108,931 | 227,061 |

Berdasarkan Tabel 4.5 menunjukkan bahwa jumlah neuron pada *hidden layer* yakni (128,54,36) menghasilkan RMSE data latih yang lebih kecil dari

pada (54,32,16) dan tidak selisih sedikit dengan neuron (32,16,8). Sedangkan pada data uji hasil RMSE lebih kecil dibandingkan dengan 2 lainnya. Oleh karena itu, dipilih *hidden layer* dengan neuron 128 pada hidden layer satu, 32 pada hidden layer 2, dan 16 neuron pada *hidden layer* 3 untuk pembuatan model MLP dalam meramalkan IHSG kedepan.

Perbandingan hasil prediksi nilai IHSG menggunakan metode MLP dengan data aktual dapat dilihat pada Gambar 4.4




Gambar 4.4 Perbandingan Data Asli dengan Hasil Prediksi Model MLP

## 4.5. Pemilihan Model Terbaik

Dari dua model yang dipilih dari masing-masing metode yakni MLP dan LSTM kemudian dibandingkan untuk menentukan model terbaik yang akan digunakan utnuk memprediksi nilai IHSG Indonesia.

Tabel 4.6 Perbandingan Model LSTM dan MLP

| Model | RMSE Data Uji |
|---|---|
| LSTM | 226,864 |
| MLP | 228,172 |

Tabel 4.6 terlihat bahwa nilai RMSE pada model LSTM lebih kecil daripada model MLP. Dapat disimpulkan bahwa model LSTM lebih baik dalam melakukan pemodelan untuk meramalkan IHSG

# BAB V
# KESIMPULAN

## 5.1. Kesimpulan

Peramalan nilai IHSG (Indeks Harga Saham Gabungan) di Indonesia dirancang dengan melakukan perbandingan dua model yakni *Long Short Term Memory* dan *Multilayer Perceptron*. Kesimpulan yang dapat diambil berdasarkan analisis runtun waktu nilai IHSG adalah :

1. Hasil analisis prediksi nilai IHSG Indonesia dengan algoritma LSTM didapatkan model terbaik yaitu LSTM dengan jumlah neuron 32 pada *input layer*, 64 *neuron hidden layer*, dan 1 neuron *output layer*. Model ini diuji menggunakan data uji diperoleh RMSE sebesar 226,864.
2. Analisis prediksi nilai IHSG Indonesia dengan algoritma MLP didapatkan model terbaik menggunakan 1 *input layer*, 3 *hidden layer* dengan neuron (128,54,36), 1 neuron *output layer*. Model ini diuji menggunakan data uji diperoleh RMSE sebesar 228,172.
3. Berdasarkan hasil evaluasi model menggunakan nilai RMSE, didapatkan bahwa model LSTM memiliki RMSE terkecil sehingga lebih baik dalam melakukan prediksi nilai IHSG Indonesia dibandingkan dengan model MLP.

## 5.2. Saran

Penelitian ini belum melakukan *tuning* pada semua *hyperparameter* dan hanya menggunakan satu optimasi Adam. Diharapkan pada penelitian selanjutnya dapat mencari kombinasi *hyperparameter* optimal dan menggunakan perbandingan dengan menggunakan optimasi lainnya.

## DAFTAR PUSTAKA


Fausett, L., 1994. Fundamentals of Neural Networks: Architectures, Algorithms, and Application. 1 penyunt. Prentice Hall, New Jersey.

Hanifah, S. dan Faturohman, T., 2017. Using Artificial Neural Network (ANN) Backpropagation to Predict the Bankcrupty of Islamic Bank in Indonesia. International Journal of Management and Applied Science , pp. 2394-7926.

Hendri, H. 2014. Character Recognition Dengan Menggunakan Jaringan Syaraf Tiruan.Jurnal TIMES, 3(2), 1-5.

Kingma D, Ba J. 2014. Adam: A method for stochastic optimization. ArXiv Prepr ArXiv14126980.

Muis, Saludin. 2008. Meramal Pergerakan Harga Saham. Yogyakarta : Graha Ilmu.

Roondiwala, Murtaza & Patel, Harshal & Varma, Shraddha. 2017. Predicting Stock Prices Using LSTM. International Journal of Science and Research (IJSR). 6. 10.21275/ART20172755.

Purnomo, M. H. dan Kurniawan , A., 2006. Supervised Neural Network. Garaha Ilmu, Surabaya.

Suyanto, 2018. Machine Learning Tingkat Dasar dan Lanjut. INFORMATIKA. Bandung.

Tandelilin, E., 2010. Portofolio dan Sehat Investasi Teori Aplikasi. Pertama penyunt. Kanisuis, Yogyakarta.

Venkatesan, & S. Anitha. (2006, November 10). Application of a Radial Basis Function Neural Network for Diagnosis of Diabetes Mellitus. Current Science, Vol. 91, No. 9. pp. 1195-1199.

## LAMPIRAN

Lampiran 1. Syntax Python

```
# -*- coding: utf-8 -*-
"""JST-IHSG Indonesia- Kel I.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1hwO9VS8nzEdrJuBs02FYMeK_24Fp4LQ7

#LSTM

##Input Data
"""

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import seaborn as sns # Visualization
import matplotlib.pyplot as plt # Visualization

from sklearn.metrics import mean_absolute_error, mean_squared_error
import math

import warnings # Supress warnings
warnings.filterwarnings('ignore')

np.random.seed(7)

!pip install colorama
from colorama import Fore

from google.colab import drive
```

```
drive.mount('/content/drive')

from google.colab import drive
drive.mount('/content/drive')

#Load Data From Local File
data=pd.read_csv("/content/drive/MyDrive/IHSG.csv",sep=',')

data.head()

data.describe()

data['Open']= data['IHSG (stock prices)']

data.shape

data.info()

data1=data.copy()

"""## Data Understanding"""

# Proses membangkitkan data waktu (Boleh diskip)
dates = pd.date_range('19900101',periods=374,freq='MS')
dates

data['Date']=pd.DataFrame(dates)
data.head()

from datetime import datetime, date

data['Date'] = pd.to_datetime(data['Date'], format = '%Y-%m-%d')
data.head().style.set_properties(subset=['Date'], **{'background-
color': 'dodgerblue'})

# Setting indeks menjadi indeks runtun waktu (Boleh diskip)
```

```
ts_df=data.set_index('Date')
ts_df.head()

"""##Plot Time Series"""

#Plot Time Series ini jika tidak dilakukan prosen pengindeksan
runtun waktu, maka variabel Y hanya berupa urutan observasi

fig=plt.figure()
plt.plot(ts_df)
fig.suptitle('Time Series Plot of IHSG')
plt.xlabel('Date')
plt.ylabel('IHSG')

# Plot ACF dan PACF
import statsmodels.api as sm

sm.graphics.tsa.plot_acf(data[['IHSG (stock prices)']])
sm.graphics.tsa.plot_pacf(data[['IHSG (stock prices)']])
plt.show()

# A month has 4 weeks
rolling_window = 1
sns.lineplot(x=data['Date'], y=data['Open'],color='dodgerblue')
sns.lineplot(x=data['Date'],
y=data['Open'].rolling(rolling_window).mean(), color='black',
label='rolling mean')
sns.lineplot(x=data['Date'],
y=data['Open'].rolling(rolling_window).std(),  color='orange',
label='rolling std')
plt.title('IHSG: Non-stationary \nnon-constant mean & non-constant
variance', fontsize=14)
plt.ylabel(ylabel='Open', fontsize=14)
plt.xlim([date(1990, 1, 1), date(2021, 2,1)])
```

```
plt.tight_layout()
plt.show()

"""##ADF Test"""

#
https://www.statsmodels.org/stable/generated/statsmodels.tsa.stattoo
ls.adfuller.html
from statsmodels.tsa.stattools import adfuller
series = data['Open']
X = series.values
result = adfuller(X)
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))

"""##Transformasi"""

# Log Transform of absolute values
# (Log transoform of negative values will return NaN)
data1['Open_log'] = np.log(abs(data1['Open']))

f, ax = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))
visualize_adfuller_results(data1['Open_log'], 'Transformed \n Open',
ax[0])

sns.distplot(data1['Open_log'], ax=ax[1])

"""##Differencing"""

# First Order Differencing
ts_diff = np.diff(data1['Open'])
data1['Open_diff_1'] = np.append([0], ts_diff)
```

```
f, ax = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))
visualize_adfuller_results(data1['Open_diff_1'], 'Differencing Lag 1
\n Open', ax[0])

sns.distplot(data1['Open_diff_1'], ax=ax[1])

data1.head()

"""##EDA 2"""

from pandas.plotting import autocorrelation_plot

autocorrelation_plot(data['IHSG (stock prices)'])
plt.show()

"""##Split Data"""

train_size = int(0.8 * len(data1))
test_size = len(data1) - train_size

univariate_df = data1[['Date', 'Open']].copy()
univariate_df.columns = ['ds', 'y']

train = univariate_df.iloc[:train_size, :]

x_train, y_train = pd.DataFrame(univariate_df.iloc[:train_size, 0]),
pd.DataFrame(univariate_df.iloc[:train_size, 1])
x_valid, y_valid = pd.DataFrame(univariate_df.iloc[train_size:, 0]),
pd.DataFrame(univariate_df.iloc[train_size:, 1])

print(len(train), len(x_valid))

"""## Min Max Scaling"""

from sklearn.preprocessing import MinMaxScaler
```

```
datafiks = univariate_df.filter(['y'])
#Convert the dataframe to a numpy array
dataset = datafiks.values

scaler = MinMaxScaler(feature_range=(-1, 0))
scaled_data = scaler.fit_transform(dataset)

scaled_data[:10]

# Defines the rolling window
look_back = 1
# Split into train and test sets
train, test = scaled_data[:train_size-look_back,:],
scaled_data[train_size-look_back:,:]

def create_dataset(dataset, look_back=1):
    X, Y = [], []
    for i in range(look_back, len(dataset)):
        a = dataset[i-look_back:i, 0]
        X.append(a)
        Y.append(dataset[i, 0])
    return np.array(X), np.array(Y)

x_train, y_train = create_dataset(train, look_back)
x_test, y_test = create_dataset(test, look_back)

# reshape input to be [samples, time steps, features]
x_train = np.reshape(x_train, (x_train.shape[0], 1,
x_train.shape[1]))
x_test = np.reshape(x_test, (x_test.shape[0], 1, x_test.shape[1]))

print(len(x_train), len(x_test))

"""##Modeling"""

from keras.models import Sequential
```

```
from keras.layers import Dense, LSTM

#Build the LSTM model
model = Sequential()
model.add(LSTM(64, return_sequences=True,
input_shape=(x_train.shape[1], x_train.shape[2])))
model.add(LSTM(32, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

#Train the model
model.fit(x_train, y_train, batch_size=1, epochs=100,
validation_data=(x_test, y_test))

model.summary()

# Lets predict with the model
train_predict = model.predict(x_train)
test_predict = model.predict(x_test)

# invert predictions
train_predict = scaler.inverse_transform(train_predict)
y_train = scaler.inverse_transform([y_train])

test_predict = scaler.inverse_transform(test_predict)
y_test = scaler.inverse_transform([y_test])

# Get the root mean squared error (RMSE) and MAE
score_rmse = np.sqrt(mean_squared_error(y_test[0],
test_predict[:,0]))
score_mae = mean_absolute_error(y_test[0], test_predict[:,0])
train_score_rmse = np.sqrt(mean_squared_error(y_train[0],
train_predict[:,0]))
```

```
train_score_mae = mean_absolute_error(y_train[0],
train_predict[:,0])
print(Fore.GREEN + 'RMSE Test: {}'.format(score_rmse))
print(Fore.GREEN + 'RMSE Train: {}'.format(train_score_rmse))

"""##Uji Residu Autokerasi"""

res = abs(y_test.transpose()-test_predict)
res

RES = pd.DataFrame(res)

sm.graphics.tsa.plot_acf(res)
sm.graphics.tsa.plot_pacf(res)
plt.show()

"""##Plot Perbandingan"""

x_train_ticks = univariate_df.head(train_size)['ds']
y_train = univariate_df.head(train_size)['y']
x_test_ticks = univariate_df.tail(test_size)['ds']

# Plot the forecast
f, ax = plt.subplots(1)
f.set_figheight(6)
f.set_figwidth(15)

sns.lineplot(x=x_train_ticks, y=y_train, ax=ax, label='Train Set')
#navajowhite
sns.lineplot(x=x_test_ticks, y=test_predict[:,0], ax=ax,
color='green', label='Prediction') #navajowhite
sns.lineplot(x=x_test_ticks, y=y_test[0], ax=ax, color='orange',
label='IHSG truth') #navajowhite

ax.set_title(f'Prediction \n RMSE Data Uji: {score_rmse:.2f}',
fontsize=14)
```

```
ax.set_xlabel(xlabel='Date', fontsize=14)
ax.set_ylabel(ylabel='Open', fontsize=14)

plt.show()

"""#MLP"""

df= data.copy()

# Scalling data
from sklearn import preprocessing

min_max_scaler = preprocessing.MinMaxScaler(feature_range=(0, 1))
df_scale = min_max_scaler.fit_transform(df['Open'].values.reshape(-
1, 1))

#Split data menjadi training dan testing
train_size = int(len(df_scale) * 0.8)
test_size = len(df_scale) - train_size
train, test = df_scale[0:train_size,:],
df_scale[train_size:len(df_scale),:]
print(len(train), len(test))

# Fungsi untuk membangun variabel input dan target
def input_target(dataset, look_back):
    dataX, dataY = [], []
    for i in range(len(dataset)-look_back):
        a = dataset[i:(i+look_back), 0]
        dataX.append(a)
        dataY.append(dataset[i + look_back, 0])
    return np.array(dataX), np.array(dataY)

# Membangun data input dan target pada data yang kita miliki
x_train, y_train = input_target(train, 1)
x_test, y_test = input_target(test, 1)
print(x_train.shape)
```

```
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)

"""##Model MLP"""

#Import library yang dibutuhkan
import tensorflow as tf
from keras.layers import Dense, Input, Dropout
from tensorflow.keras.optimizers import Adam
from keras.models import Model
from keras.models import load_model
from keras.callbacks import ModelCheckpoint

#Mendefinisikan layer input sebagai (look_back, none)
input_layer = Input(shape=(1,))

#Dense layer di sini seperti hidden layer
dense1 = Dense(128, activation='relu')(input_layer)
dense2 = Dense(64, activation='relu')(dense1)
dense3 = Dense(32, activation='relu')(dense2)

#Mendefinisikan output layer
output_layer = Dense(1, activation='linear')(dense3)

#Membentuk model
ts_model = Model(inputs=input_layer, outputs=output_layer)
ts_model.compile(loss='mean_squared_error', optimizer='adam')
ts_model.summary()

import os
#Running model pada data training
save_path = "."
save_weights_at=os.path.join('keras_model','Bobot_Open.hdf5')
save_best=ModelCheckpoint(save_weights_at,
monitor='val_loss',verbose=0,
```

```
save_best_only=True,save_weights_only=False)
ts_model.fit(x=x_train, y=y_train, batch_size=16, epochs=100,
             verbose=1, callbacks=[save_best],
validation_data=(x_train, y_train),
             shuffle=True)
ts_model.save(os.path.join(save_path,"network.h5"))

#Prediksi data training dan testing berdasarkan bobot terbaik yang
didapat
best_model = load_model(os.path.join(save_path,"network.h5"))
trainPredict = best_model.predict(x_train)
testPredict = best_model.predict(x_test)

#Re-shape data Y train dan testing agar dapat dikonversi menjadi
data asli kembali
y_tr=y_train.reshape(-1,1)
y_ts=y_test.reshape(-1,1)

#Konfersi data train dan test
trainPredict=min_max_scaler.inverse_transform(trainPredict)
y_train=min_max_scaler.inverse_transform(y_tr)
testPredict=min_max_scaler.inverse_transform(testPredict)
y_test=min_max_scaler.inverse_transform(y_ts)

#Melihat R-square dari data training dan testing
#R-Square, RMSE, MSE, MAPE, dsb sering digunakan untuk kasus
regresi, karena tidak dapat menggunakan akurasi, presisi, recall,
dsb.
from sklearn.metrics import r2_score

r2tr = r2_score(y_train, trainPredict)
print('R-squared data training:', round(r2tr,4)*100,"%")
r2ts = r2_score(y_test, testPredict)
print('R-squared data testing:', round(r2ts,4)*100,"%")
```

```
# Menghitung RMSE
from sklearn import metrics

score = np.sqrt(metrics.mean_squared_error(trainPredict,y_train))
print(f"RMSE data training: {score}")
score = np.sqrt(metrics.mean_squared_error(testPredict,y_test))
print(f"RMSE data testing: {score}")

#Memvisualisasi hasil prediksi vs. aktual
look_back=1
# shift train predictions for plotting
trainPredictPlot = np.empty_like(df_scale)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(trainPredict)+look_back, :] =
trainPredict
# shift test predictions for plotting
testPredictPlot = np.empty_like(df_scale)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(trainPredict)+(look_back*2):len(df_scale), :] =
testPredict
# plot baseline and predictions
plt.plot(df[['Open']])
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()

"""## Uji Residu Autokorelasi MLP"""

res2= abs(testPredict-y_test)

RES2= pd.DataFrame(res2)

sm.graphics.tsa.plot_acf(RES2)
sm.graphics.tsa.plot_pacf(RES2)
plt.show()
```

Lampiran 2. Data Penelitian

| Tanggal | IHSG |
|---|---|
| Jan-90 | 441.81 |
| Feb-90 | 504.27 |
| Mar-90 | 612.2 |
| Apr-90 | 638.79 |
| May-90 | 636.4 |
| Jun-90 | 624.33 |
| Jul-90 | 614.41 |
| Aug-90 | 556.34 |
| Sep-90 | 468.51 |
| Oct-90 | 416.49 |
| Nov-90 | 383.2 |
| Dec-90 | 417.79 |
| Jan-91 | 383.02 |
| Feb-91 | 391.33 |
| Mar-91 | 408.11 |
| Apr-91 | 413.71 |
| May-91 | 397.6 |
| Jun-91 | 346.27 |
| Jul-91 | 339.96 |
| Aug-91 | 300.98 |
| Sep-91 | 249.19 |
| Oct-91 | 226.68 |
| Nov-91 | 241.32 |
| Dec-91 | 247.39 |
| Jan-92 | 282.2 |
| Feb-92 | 280.9 |
| Mar-92 | 278.6 |
| Apr-92 | 277.2 |

| Tanggal | IHSG |
|---|---|
| Feb-93 | 300.38 |
| Mar-93 | 310.76 |
| Apr-93 | 314.1 |
| May-93 | 341.85 |
| Jun-93 | 360.35 |
| Jul-93 | 356.72 |
| Aug-93 | 417.3 |
| Sep-93 | 419.96 |
| Oct-93 | 466.15 |
| Nov-93 | 518.78 |
| Dec-93 | 588.77 |
| Jan-94 | 592.02 |
| Feb-94 | 546.23 |
| Mar-94 | 492.37 |
| Apr-94 | 462.4 |
| May-94 | 501.79 |
| Jun-94 | 457.3 |
| Jul-94 | 451.08 |
| Aug-94 | 510.26 |
| Sep-94 | 497.97 |
| Oct-94 | 523.49 |
| Nov-94 | 482.63 |
| Dec-94 | 469.64 |
| Jan-95 | 433.83 |
| Feb-95 | 453.58 |
| Mar-95 | 428.64 |
| Apr-95 | 416.45 |
| May-95 | 475.28 |

| Tanggal | IHSG |
|---|---|
| Mar-96 | 585.7 |
| Apr-96 | 623.9 |
| May-96 | 617.46 |
| Jun-96 | 594.25 |
| Jul-96 | 536.02 |
| Aug-96 | 547.61 |
| Sep-96 | 573.3 |
| Oct-96 | 568.02 |
| Nov-96 | 613.01 |
| Dec-96 | 637.43 |
| Jan-97 | 691.11 |
| Feb-97 | 705.37 |
| Mar-97 | 662.23 |
| Apr-97 | 652.05 |
| May-97 | 696.02 |
| Jun-97 | 724.55 |
| Jul-97 | 721.27 |
| Aug-97 | 493.96 |
| Sep-97 | 546.68 |
| Oct-97 | 500.41 |
| Nov-97 | 401.7 |
| Dec-97 | 401.71 |
| Jan-98 | 485.93 |
| Feb-98 | 482.37 |
| Mar-98 | 541.42 |
| Apr-98 | 460.13 |
| May-98 | 420.46 |
| Jun-98 | 445.92 |

| | | | | | |
|---|---|---|---|---|---|
| May-92 | 299.2 | Jun-95 | 492.27 | Jul-98 | 481.71 |
| Jun-92 | 313.6 | Jul-95 | 512.06 | Aug-98 | 342.43 |
| Jul-92 | 317.2 | Aug-95 | 500.74 | Sep-98 | 276.15 |
| Aug-92 | 301.27 | Sep-95 | 493.24 | Oct-98 | 300.77 |
| Sep-92 | 298.39 | Oct-95 | 488.44 | Nov-98 | 386.27 |
| Oct-92 | 307.58 | Nov-95 | 481.73 | Dec-98 | 398.03 |
| Nov-92 | 285.6 | Dec-95 | 513.84 | Jan-99 | 411.93 |
| Dec-92 | 274.34 | Jan-96 | 578.55 | Feb-99 | 396 |
| Jan-93 | 280.15 | Feb-96 | 585.2 | Mar-99 | 393.62 |

| Tanggal | IHSG | Tanggal | IHSG | Tanggal | IHSG |
|---|---|---|---|---|---|
| Apr-99 | 495.22 | Apr-02 | 544.85 | Apr-05 | 1080.17 |
| May-99 | 585.24 | May-02 | 530.79 | May-05 | 1088.17 |
| Jun-99 | 662.02 | Jun-02 | 505.01 | Jun-05 | 1122.37 |
| Jul-99 | 597.87 | Jul-02 | 463.67 | Jul-05 | 1182.3 |
| Aug-99 | 572.66 | Aug-02 | 456.4 | Aug-05 | 1050.09 |
| Sep-99 | 547.94 | Sep-02 | 412.43 | Sep-05 | 1079.27 |
| Oct-99 | 593.86 | Oct-02 | 371.14 | Oct-05 | 1058.26 |
| Nov-99 | 583.8 | Nov-02 | 390.42 | Nov-05 | 1017.73 |
| Dec-99 | 676.92 | Dec-02 | 424.94 | Dec-05 | 1162.63 |
| Jan-00 | 636.37 | Jan-03 | 388.44 | Jan-06 | 1229.7 |
| Feb-00 | 576.54 | Feb-03 | 399.22 | Feb-06 | 1216.14 |
| Mar-00 | 583.27 | Mar-03 | 398 | Mar-06 | 1322.97 |
| Apr-00 | 526.73 | Apr-03 | 435.04 | Apr-06 | 1464.4 |
| May-00 | 454.32 | May-03 | 494.78 | May-06 | 1330 |
| Jun-00 | 515.11 | Jun-03 | 497.81 | Jun-06 | 1310.26 |
| Jul-00 | 492.19 | Jul-03 | 508.7 | Jul-06 | 1351.65 |
| Aug-00 | 466.38 | Aug-03 | 530.86 | Aug-06 | 1444.49 |
| Sep-00 | 421.33 | Sep-03 | 599.84 | Sep-06 | 1534 |

| | | | | | |
|---|---|---|---|---|---|
| Oct-00 | 405.34 | Oct-03 | 629.05 | Oct-06 | 1582.62 |
| Nov-00 | 429.21 | Nov-03 | 617.08 | Nov-06 | 1718.96 |
| Dec-00 | 416.32 | Dec-03 | 679.3 | Dec-06 | 1805.52 |
| Jan-01 | 425.61 | Jan-04 | 752.93 | Jan-07 | 1757.26 |
| Feb-01 | 428.3 | Feb-04 | 761.08 | Feb-07 | 1740.97 |
| Mar-01 | 381.05 | Mar-04 | 735.67 | Mar-07 | 1830.92 |
| Apr-01 | 358.23 | Apr-04 | 783.41 | Apr-07 | 1999.17 |
| May-01 | 405.86 | May-04 | 733.99 | May-07 | 2084.32 |
| Jun-01 | 437.62 | Jun-04 | 732.4 | Jun-07 | 2139.28 |
| Jul-01 | 444.08 | Jul-04 | 756.98 | Jul-07 | 2348.67 |
| Aug-01 | 435.55 | Aug-04 | 746.76 | Aug-07 | 2194.34 |
| Sep-01 | 392.47 | Sep-04 | 819.82 | Sep-07 | 2359.21 |
| Oct-01 | 383.74 | Oct-04 | 860.35 | Oct-07 | 2643.49 |
| Nov-01 | 380.31 | Nov-04 | 977.77 | Nov-07 | 2688.33 |
| Dec-01 | 392.03 | Dec-04 | 1000.23 | Dec-07 | 2745.83 |
| Jan-02 | 392.03 | Jan-05 | 1045.44 | Jan-08 | 2627.25 |
| Feb-02 | 453.25 | Feb-05 | 1073.83 | Feb-08 | 2721.94 |
| Mar-02 | 481.86 | Mar-05 | 1080.17 | Mar-08 | 2447.3 |

| Tanggal | IHSG | Tanggal | IHSG | Tanggal | IHSG |
|---|---|---|---|---|---|
| Apr-08 | 2304.52 | Apr-11 | 3819.62 | Apr-14 | 4840.15 |
| May-08 | 2444.35 | May-11 | 3836.97 | May-14 | 4893.91 |
| Jun-08 | 2349.11 | Jun-11 | 3888.57 | Jun-14 | 4878.58 |
| Jul-08 | 2304.51 | Jul-11 | 4130.8 | Jul-14 | 5088.8 |
| Aug-08 | 2165.94 | Aug-11 | 3841.73 | Aug-14 | 5136.86 |
| Sep-08 | 1832.51 | Sep-11 | 3549.03 | Sep-14 | 5137.58 |
| Oct-08 | 1256.7 | Oct-11 | 3790.85 | Oct-14 | 5089.55 |
| Nov-08 | 1241.54 | Nov-11 | 3715.08 | Nov-14 | 5149.89 |
| Dec-08 | 1355.41 | Dec-11 | 3821.99 | Dec-14 | 5226.95 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Jan-09 | 1332.67 | | Jan-12 | 3941.69 | | Jan-15 | 5289.4 |
| Feb-09 | 1285.48 | | Feb-12 | 3985.21 | | Feb-15 | 5450.29 |
| Mar-09 | 1406.65 | | Mar-12 | 4121.55 | | Mar-15 | 5518.68 |
| Apr-09 | 1722.77 | | Apr-12 | 4180.73 | | Apr-15 | 5086.43 |
| May-09 | 1916.83 | | May-12 | 3832.82 | | May-15 | 5216.38 |
| Jun-09 | 2026.78 | | Jun-12 | 3955.58 | | Jun-15 | 4910.66 |
| Jul-09 | 2323.24 | | Jul-12 | 4142.34 | | Jul-15 | 4802.53 |
| Aug-09 | 2341.54 | | Aug-12 | 4060.33 | | Aug-15 | 4509.61 |
| Sep-09 | 2467.59 | | Sep-12 | 4262.56 | | Sep-15 | 4223.91 |
| Oct-09 | 2367.7 | | Oct-12 | 4350.29 | | Oct-15 | 4455.18 |
| Nov-09 | 2415.84 | | Nov-12 | 4276.14 | | Nov-15 | 4446.46 |
| Dec-09 | 2534.36 | | Dec-12 | 4316.69 | | Dec-15 | 4593.01 |
| Jan-10 | 2610.8 | | Jan-13 | 4453.7 | | Jan-16 | 4615.16 |
| Feb-10 | 2549.03 | | Feb-13 | 4795.79 | | Feb-16 | 4770.96 |
| Mar-10 | 2777.3 | | Mar-13 | 4940.99 | | Mar-16 | 4845.37 |
| Apr-10 | 2971.25 | | Apr-13 | 5034.07 | | Apr-16 | 4838.58 |
| May-10 | 2796.96 | | May-13 | 5068.63 | | May-16 | 4796.87 |
| Jun-10 | 2913.68 | | Jun-13 | 4818.9 | | Jun-16 | 5016.65 |
| Jul-10 | 3069.28 | | Jul-13 | 4610.38 | | Jul-16 | 5215.99 |
| Aug-10 | 3081.88 | | Aug-13 | 4195.09 | | Aug-16 | 5386.08 |
| Sep-10 | 3501.3 | | Sep-13 | 4316.18 | | Sep-16 | 5364.8 |
| Oct-10 | 3635.32 | | Oct-13 | 4510.63 | | Oct-16 | 5422.54 |
| Nov-10 | 3531.21 | | Nov-13 | 4256.44 | | Nov-16 | 5148.91 |
| Dec-10 | 3703.51 | | Dec-13 | 4274.18 | | Dec-16 | 5296.71 |
| Jan-11 | 3409.17 | | Jan-14 | 4418.76 | | Jan-17 | 5294.1 |
| Feb-11 | 3470.35 | | Feb-14 | 4620.22 | | Feb-17 | 5386.69 |
| Mar-11 | 3678.67 | | Mar-14 | 4768.28 | | Mar-17 | 5568.11 |

| Tanggal | IHSG |
|---|---|
| Apr-17 | 5685.3 |
| May-17 | 5738.16 |
| Jun-17 | 5829.71 |
| Jul-17 | 5840.94 |
| Aug-17 | 5864.06 |
| Sep-17 | 5900.85 |
| Oct-17 | 6005.78 |
| Nov-17 | 5952.14 |
| Dec-17 | 6355.65 |
| Jan-18 | 6605.63 |
| Feb-18 | 6597.22 |
| Mar-18 | 6188.99 |
| Apr-18 | 5994.6 |
| May-18 | 5983.59 |
| Jun-18 | 5799.24 |
| Jul-18 | 5936.44 |
| Aug-18 | 6018.46 |
| Sep-18 | 5976.55 |
| Oct-18 | 5831.65 |
| Nov-18 | 6056.12 |
| Dec-18 | 6194.5 |
| Jan-19 | 6532.97 |
| Feb-19 | 6443.35 |
| Mar-19 | 6468.76 |
| Apr-19 | 6455.35 |
| May-19 | 6209.12 |
| Jun-19 | 6358.63 |
| Jul-19 | 6390.5 |
| Aug-19 | 6328.47 |

| Tanggal | IHSG |
|---|---|
| May-20 | 4753.61 |
| Jun-20 | 4905.39 |
| Jul-20 | 5149.63 |
| Aug-20 | 5238.49 |
| Sep-20 | 4870.04 |
| Oct-20 | 5128.22 |
| Nov-20 | 5612.42 |
| Dec-20 | 5979.07 |
| Jan-21 | 5862.35 |
| Feb-21 | 6241.8 |

| | |
|---|---|
| Sep-19 | 6169.1 |
| Oct-19 | 6228.32 |
| Nov-19 | 6011.83 |
| Dec-19 | 6299.54 |
| Jan-20 | 5940.05 |
| Feb-20 | 5452.7 |
| Mar-20 | 4538.93 |