Eduvest – Journal of Universal Studies
Volume 2 Number 7, July, 2022
p- ISSN 2775-3735- e-ISSN 2775-3727

PREDICTION OF THE DEVELOPMENT OF COVID-19 CASE IN INDONESIA BASED ON GOOGLE TREND ANALYSIS

Sulastri[1], Eri Zuliarso[2], Arief Jananto[3]
Universitas Stikubank (Unisbank) Semarang, Indonesia
Email: sulastri@edu.unisbank.ac.id, eri299@edu.unisbank.ac.id, ajananto09@edu.unisbank.ac.id

ABSTRACT

The global outbreak of the coronavirus disease (COVID-19) has recently hit many countries around the world. Indonesia is one of the 10 most affected countries. Search engines such as Google provide data on search activity in a population, and this data may be useful for analyzing epidemics. Leveraging data mining methods on electronic resource data can provide better insights into the COVID-19 outbreak to manage health crises in every country and around the world. This study aims to predict the incidence of COVID-19 by utilizing data from the Covid 19 Task Force and the Google Trends website. Linear regression and long-term memory (LSTM) models were used to estimate the number of positive COVID-19 cases.

KEYWORDS Covid-19, Long ShortTerm Memori, Google Trend

INTRODUCTION

Long holidays often encourage people to travel, even though movement and crowds can have an impact on increasing Covid-19 cases (Wen, Kozak, Yang, & Liu, 2020). According to data from the Covid-19 Handling Task Force, there is always an upward trend of positive cases occurring every holiday period (Sharpe Jr, Kuszyk, & Mossa-Basha, 2021). Google Trends is a website owned by Google.Inc that contains trends in the use of keywords on the Google search engine website and trending news (Jun, Yoo, & Choi, 2018). One of the benefits of Google Trends is for research. RNN has been used for sequential time series applications with temporal dependencies.

RNN which has the ability to process the current data by using the previous data. Meanwhile, the RNN is problematic to train long-term dependency data, which is solved

How to cite:	Sulastri, Eri Zuliarso, Arief Jananto (2022). Prediction of the Development of Covid-19 Case In Indonesia Based on Google Trend Analysis. Journal Eduvest. *Vol 2*(7): 1.407-1.425
E-ISSN:	2775-3727
Published by:	https://greenpublisher.id/

by one of the RNN variants. The LSTM was anticipated by Hochreiter and Schmidhuber, has been used as an advanced version of the RNN network and has overcome the limitations of RNN by using a hidden layer unit known as a memory cell. The memory cells are self-connected which store the temporal state of the network and are controlled through three named gates: input gate, output gate and forget gate (Gers & Schmidhuber, 2001).

The work of input and output gates is used to control the flow of input and output of memory cells throughout the network (Sak, Senior, & Beaufays, 2014). In addition, a forget gate has been added to the memory cell, which passes high-weighted output information from the previous neuron to the next neuron. Information residing in memory depends on high activation yield; if the input unit has high activation, the information is stored in the memory cell. In addition, if the output unit has a high activation, the information will be passed on to the next neuron (Shahid, Zameer, & Muneeb, 2020). Otherwise, the high-weighted input information resides in the memory cell.

This study analyzes the development of Covid-19 cases associated with several keywords on Google Trends. In this study, several algorithms were tested to analyze the development of Covid 19 cases associated with keywords in Google Trends (Pan, Nguyen, Abu-Gellban, & Zhang, 2020).

RESEARCH METHOD

In the first stage of this research, we will explore the data in Google Trends. The keywords used are 'covid 19', 'ppkm', 'lockdown', 'ptm', 'wfh', 'vaccination', 'cluster', 'coronavirus', 'psbb', 'delta variant'. With a period starting from 2020-01-01 to 2021-11-10. The study began by downloading data on the development of daily spread on the COVID-19 website. Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (https://github.com/CSSEGISandData/COVID-19) and from https://data.humdata.org/dataset/indonesia-covid-19-cases-recoveries-and-deaths-per-province

The data in Google Trends is a random sample of Google search data. This data is anonymized (identity not disclosed), classified (search query topics defined), and aggregated (grouped together). Google Trends data can be filtered in two ways: real time and non-real time. Real time refers to a random sample of searches from the previous seven days, while non-real time refers to a random sample of the entire Google dataset, which can range from 2004 to 36 hours ago (Pretorius, Kruger, & Bezuidenhout, 2022). Google Trends are two separate random samples, so the graph will show one or the other, but not both at the same time.

RESULT AND DISCUSSION

1. Analysis of Community Activities on Daily Cases of Confirmed Covid 19 and their Visualization.

This step begins with preparing the data to be used, namely daily case variables with global mobility, visualizing daily cases with each global mobility variable and analyzing their correlation.

```
[18]:  dataakhir.head

[18]:  <bound method NDFrame.head of           tanggal  Kasusharian     Totalkasus  totalmeninggal  \
       0     2020-03-02     0.000035  0.000000e+00        0.000000
       1     2020-03-03     0.000000  0.000000e+00        0.000000
       2     2020-03-04     0.000000  0.000000e+00        0.000000
       3     2020-03-05     0.000000  0.000000e+00        0.000000
       4     2020-03-06     0.000035  4.706634e-07        0.000000
       ..           ...          ...           ...             ...
       613   2021-11-06     0.007065  9.996230e-01        0.999596
       614   2021-11-07     0.007823  9.997275e-01        0.999673
       615   2021-11-08     0.004299  9.997849e-01        0.999756
       616   2021-11-09     0.007647  9.998870e-01        0.999903
       617   2021-11-10     0.008457  1.000000e+00        1.000000

            retail_and_recreation  grocery_and_pharmacy     parks  \
       0                 0.920635              0.516854  0.630435
       1                 0.888889              0.494382  0.608696
       2                 0.841270              0.426966  0.576087
       3                 0.904762              0.471910  0.619565
       4                 0.873016              0.426966  0.597826
       ..                     ...                   ...       ...
       613               0.936508              0.674157  0.608696
       614               0.968254              0.719101  0.652174
       615               0.888889              0.629213  0.663043
       616               0.968254              0.696629  0.673913
```

```
[19]:
for column in dataakhir.columns:
    print(column)
```

```
tanggal
Kasusharian
Totalkasus
totalmeninggal
retail_and_recreation
grocery_and_pharmacy
parks
transit_stations
workplaces
residential
```

```
[20]:
dataakhir3=dataakhir.loc[:,['tanggal','retail_and_recreation','Kasusharian']]
dataakhir4=dataakhir.loc[:,['tanggal','grocery_and_pharmacy ','Kasusharian']]
dataakhir5=dataakhir.loc[:,['tanggal','parks ','Kasusharian']]
dataakhir6=dataakhir.loc[:,['tanggal','transit_stations ','Kasusharian']]
dataakhir8=dataakhir.loc[:,['tanggal','workplaces ','Kasusharian']]
dataakhir9=dataakhir.loc[:,['tanggal','residential ','Kasusharian']]
```

Figure 1.11 Preparing data for analysis of community activities and daily confirmed cases of covid 19

```
[21]:
dataakhir3.plot.line(x="tanggal", title="retail_and_recreation Mobility and Confirm Positif Covid 19");

plot.show(block=True);
dataakhir4.plot.line(x="tanggal", title="grocery_and_pharmacy Mobility and Confirm Positif Covid 19");

plot.show(block=True);
dataakhir5.plot.line(x="tanggal", title="parks  Mobility and Confirm Positif Covid 19");
plot.show(block=True);

dataakhir6.plot.line(x="tanggal", title="transit_stations Mobility and Confirm Positif Covid 19");
plot.show(block=True);

dataakhir8.plot.line(x="tanggal", title="workplaces  Mobility and Confirm Positif Covid 19");
plot.show(block=True);
dataakhir9.plot.line(x="tanggal", title="residential Mobility and Confirm Positif Covid 19");
plot.show(block=True);
```

Figure 1. 12 Coding for visualization of daily cases and community activities

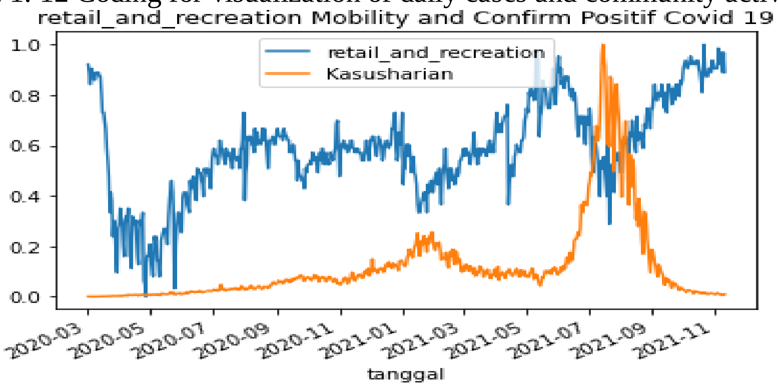

Figure 1.13 Visualization between Retail_and_recreation and daily cases

Figure 1.13 shows that if daily cases decrease then retail and recreation increases, otherwise if daily cases increase then retail and recreation decreases.

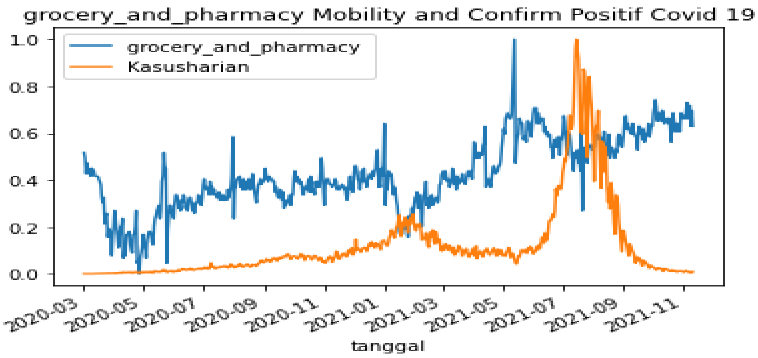

Figure 1.14 Visualization between grocery and pharmacy and daily cases

Figure 1.14 shows that grocery and pharmacy during the pandemic is consistently high, this shows that the public's need for medicines is quite high during the pandemic.

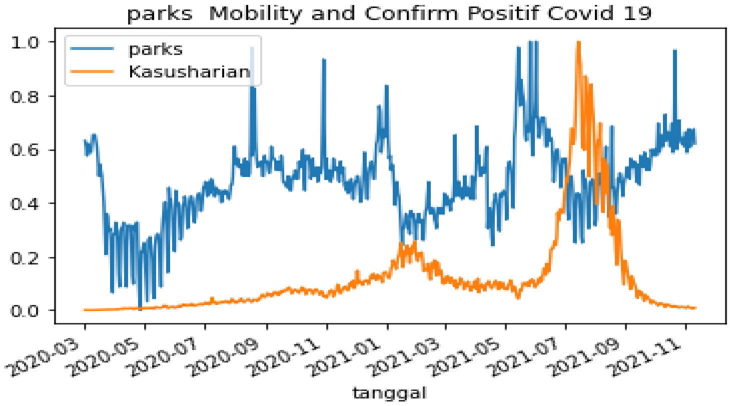

Figure 1.15 Visualization between parks and daily cases

Figure 1.15 shows that if daily cases decrease then activities in parks increase, on the contrary if daily cases increase then community activities in parks decrease.

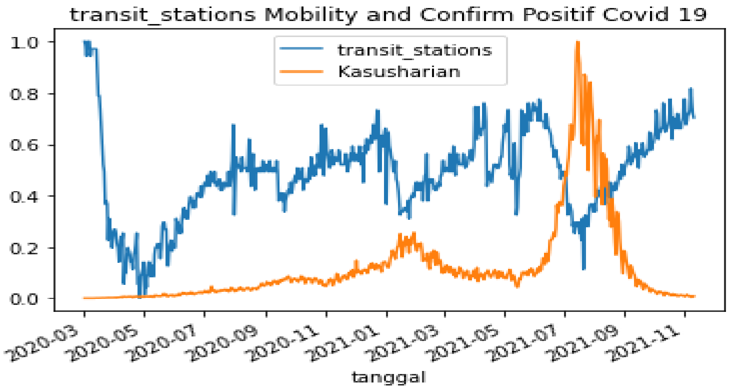

Figure 1.16 Visualization between transit_station and daily cases

Figure 1.16 shows that if daily cases decrease then activity at the station (transit_station) increases, otherwise if daily cases increase then activity at the station (transit_station) decreases.

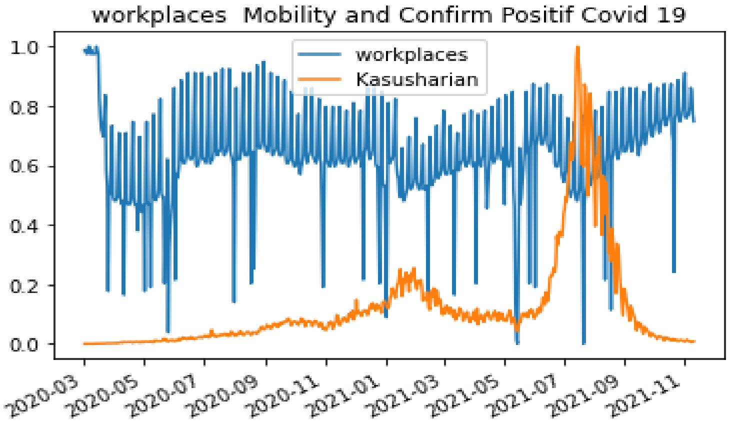

Figure 1.17 Visualization between workplace and daily cases

Figure 1.17 shows that in the early days of the pandemic work activities were quite high, but during high daily cases it can be seen that work activities fell drastically due to the lockdown.

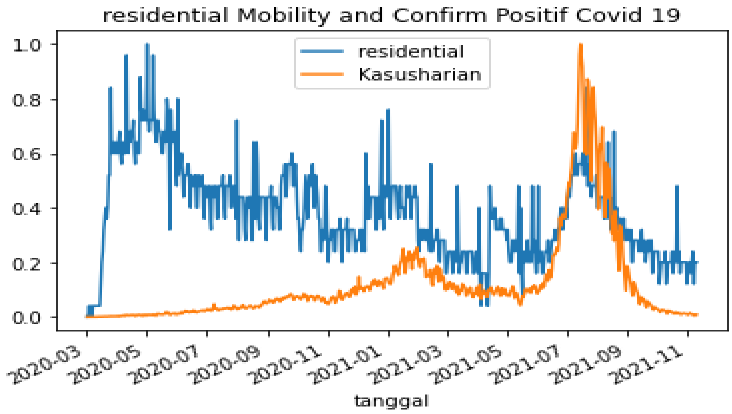

Figure 1.18 Visualization between residential and daily cases

Figure 1.18 shows that activity in housing is high at the beginning of the pandemic, daily cases are low, but at high daily cases activity in housing decreases but does not decrease at all.

```
[22]:
import pandas as pd
from scipy.stats import pearsonr

#dataakhir3=dataakhir.loc[:,['date','retail_and_recreation','positif']]

# Convert dataframe into series
list1 = dataakhir3['retail_and_recreation']
list2 = dataakhir3['Kasusharian']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir4['grocery_and_pharmacy ']
list2 = dataakhir4['Kasusharian']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir5['parks ']
list2 = dataakhir5['Kasusharian']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir6['transit_stations ']
list2 = dataakhir6['Kasusharian']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir8['workplaces ']
list2 = dataakhir8['Kasusharian']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")

Pearsons correlation: -0.005

Pearsons correlation: 0.252

Pearsons correlation: -0.047

Pearsons correlation: -0.172

Pearsons correlation: -0.160

Pearsons correlation: 0.102
```

Figure 1.19 Pearson Correlation Analysis of daily cases and community activities

Figure 1.19 shows the results of correlation analysis on daily cases and community activities using Pearson Correlation. From the picture, it can be seen that the negative correlations are retail and recreation, parks, transit station, workplaces retail and pharmacy meaning that if daily cases increase, these four activities will decrease and vice versa. While the correlation between daily cases and grocery and pharmacy, residential is positive even though the value is small.

```
[23]:
x_simple = dataakhir3
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
x_simple = dataakhir4
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
x_simple = dataakhir5
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
x_simple = dataakhir6
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
x_simple = dataakhir8
my_r = x_simple.corr(method="spearman")
print(my_r)
```

```
                      retail_and_recreation  Kasusharian
retail_and_recreation              1.000000     0.100709
Kasusharian                        0.100709     1.000000

                      grocery_and_pharmacy  Kasusharian
grocery_and_pharmacy              1.000000     0.312371
Kasusharian                       0.312371     1.000000

                parks  Kasusharian
parks        1.000000     0.042117
Kasusharian  0.042117     1.000000

                  transit_stations  Kasusharian
transit_stations          1.000000     0.040701
Kasusharian               0.040701     1.000000

             workplaces  Kasusharian
workplaces     1.000000    -0.175022
Kasusharian   -0.175022     1.000000

             residential  Kasusharian
residential     1.000000    -0.109121
Kasusharian    -0.109121     1.000000
```

Figure 1.20 Spearman Correlation Analysis of daily cases and community activities

Figure 1.20 shows a correlation analysis using Spearman correlation, from the figure it shows that the positive correlations are retail_and_recreation, grocery_and_pharmacy, parks, transit_station. While the value is worksplaces and residential.

2. Analysis of Community Activities on the Total Confirmed Cases of Covid 19 and its Visualization.

This step begins with preparing the data to be used, namely the Total Case variable with global mobility, visualizing daily cases with each global mobility variable and analyzing the correlation.

```
[24]:
dataakhir13=dataakhir.loc[:,['tanggal','retail_and_recreation','Totalkasus']]
dataakhir14=dataakhir.loc[:,['tanggal','grocery_and_pharmacy ','Totalkasus']]
dataakhir15=dataakhir.loc[:,['tanggal','parks ','Totalkasus']]
dataakhir16=dataakhir.loc[:,['tanggal','transit_stations ','Totalkasus']]
dataakhir18=dataakhir.loc[:,['tanggal','workplaces ','Totalkasus']]
dataakhir19=dataakhir.loc[:,['tanggal','residential ','Totalkasus']]
```

```
[25]:
dataakhir13.plot.line(x="tanggal", title="retail_and_recreation Mobility and Total Cumalatif Confirm Pos

plot.show(block=True);
dataakhir14.plot.line(x="tanggal", title="grocery_and_pharmacy Mobility and Total Cumalatif Confirm Posi

plot.show(block=True);
dataakhir15.plot.line(x="tanggal", title="parks  Mobility and Total Cumalatif Confirm Positif Covid 19")
plot.show(block=True);

dataakhir16.plot.line(x="tanggal", title="transit_stations Mobility and Total Cumalatif Confirm Positif (
plot.show(block=True);
```

Figure 1.21 Preparation of data for analysis of total cases and community activities

Figure 1.21 shows the coding of data preparation for visualization between the total cases and community activities.

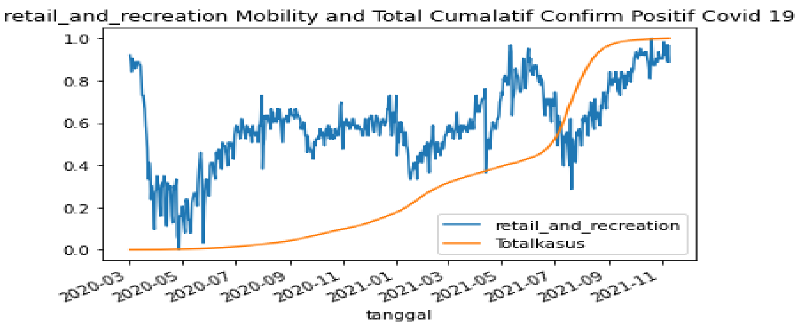

Figure 1.22 Visualization between Retail and recreation and total cases

Figure 1.22 shows a visualization between retail and recreation and total cases, where when the total cases are high, retail and recreation decreases.

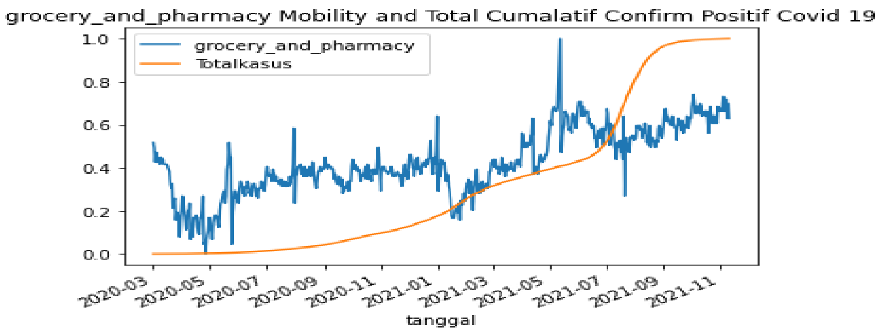

Figure 1.23 Visualization between grocery_and_pharmacy and total cases

Figure 1.23 shows that grocery_and_pharmacy activities are quite stable, meaning that drug buying activities are quite stable, the increase occurs when the total number of cases is high.

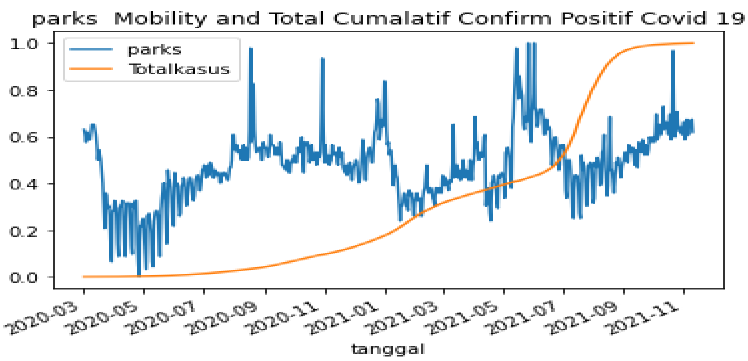

Figure 1.24 Visualization between parks and total cases

Figure 1.24 shows that community activities in parks at the beginning of the pandemic were quite high, but when total cases were high, parks activities decreased.

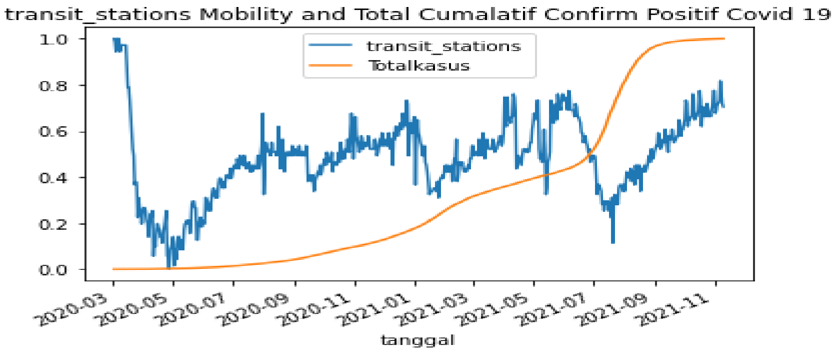

Figure 1.25 Visualization between transit_station and daily cases

Figure 1.25 shows that transit_station activity is high if daily cases are low. When the daily case goes up, transit_station goes down.

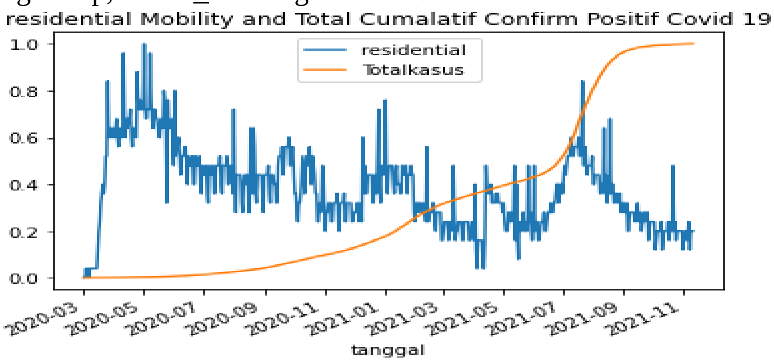

Figure 1.26 Visualization between residential and daily cases

Figure 1.26 shows that daily cases are low, so residential is high. However, when daily cases are high, it can be seen that residential drops drastically.

```
[26]:
# Convert dataframe into series
list1 = dataakhir13['retail_and_recreation']
list2 = dataakhir13['Totalkasus']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir14['grocery_and_pharmacy ']
list2 = dataakhir14['Totalkasus']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir15['parks ']
list2 = dataakhir15['Totalkasus']

# Apply the pearsonr()

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir16['transit_stations ']
list2 = dataakhir16['Totalkasus']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir18['workplaces ']
list2 = dataakhir18['Totalkasus']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
```

```
# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
```

```
Pearsons correlation: 0.593

Pearsons correlation: 0.748

Pearsons correlation: 0.325

Pearsons correlation: 0.304

Pearsons correlation: 0.144

Pearsons correlation: -0.401
```

Figure 1.27 Pearson correlation analysis between daily cases and community activities

Figure 1.27 shows the correlation value between daily cases and community mobility using the Pearson correlation formula. It can be seen that the highest correlation is 0.748, namely the correlation between daily cases and grocery_and_pharmacy, meaning that the correlation is quite high and positive.

```
[27]:
x_simple = dataakhir13
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
x_simple = dataakhir14
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
x_simple = dataakhir15
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
x_simple = dataakhir16
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
x_simple = dataakhir18
my_r = x_simple.corr(method="spearman")
print(my_r)
```

```
retail_and_recreation                1.000000     0.595947
Totalkasus                           0.595947     1.000000

                       grocery_and_pharmacy   Totalkasus
grocery_and_pharmacy               1.000000     0.784269
Totalkasus                         0.784269     1.000000

                 parks   Totalkasus
parks         1.000000     0.402209
Totalkasus    0.402209     1.000000

                   transit_stations   Totalkasus
transit_stations            1.00000      0.42261
Totalkasus                  0.42261      1.00000

              workplaces   Totalkasus
workplaces      1.000000     0.158051
Totalkasus      0.158051     1.000000

               residential   Totalkasus
residential       1.000000    -0.506859
Totalkasus       -0.506859     1.000000
```

Figure 1. 28 Spearman Correlation Analysis between total cases and community activities

Figure 1.28 shows the correlation analysis using Spearman, from the results obtained the highest correlation value is 0.784, namely the correlation between grocery_and_pharmacy and total cases. This means that the correlation is high and positive.

3. **Analysis of Community Activities on the Total Confirmed Deaths of Covid 19 and its Visualization.**

This step begins with preparing the data to be used, namely the Total Died variable with global mobility, visualizing daily cases with each global mobility variable and analyzing the correlation.

```
[28]:
dataakhir23=dataakhir.loc[:,['tanggal','retail_and_recreation','totalmeninggal']]
dataakhir24=dataakhir.loc[:,['tanggal','grocery_and_pharmacy ','totalmeninggal']]
dataakhir25=dataakhir.loc[:,['tanggal','parks ','totalmeninggal']]
dataakhir26=dataakhir.loc[:,['tanggal','transit_stations ','totalmeninggal']]
dataakhir28=dataakhir.loc[:,['tanggal','workplaces ','totalmeninggal']]
dataakhir29=dataakhir.loc[:,['tanggal','residential ','totalmeninggal']]
```

```
[29]:
dataakhir23.plot.line(x="tanggal", title="retail_and_recreation Mobility and total meninggal Confirm  Covid 19");

plot.show(block=True);
dataakhir24.plot.line(x="tanggal", title="grocery_and_pharmacy Mobility and total meninggal Confirm  Covid 19");

plot.show(block=True);
dataakhir25.plot.line(x="tanggal", title="parks  Mobility and total meninggal Confirm  Covid 19");
plot.show(block=True);

dataakhir26.plot.line(x="tanggal", title="transit_stations Mobility and total meninggal Confirm  Covid 19");
plot.show(block=True);

dataakhir28.plot.line(x="tanggal", title="workplaces  Mobility and total meningga lConfirm  Covid 19");
```

Figure 1.29 Data preparation for analysis of total deaths by community activities

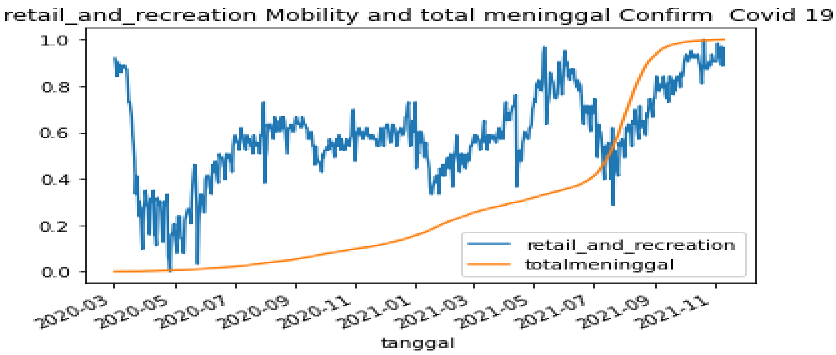

Figure 1.30 Visualization between Retail and recreation and Total death

Figure 1.30 shows a visualization between retail and recreation and total deaths, where when total deaths are low, retai and recreation is high, on the other hand, total deaths are high, retail and recreation is low.

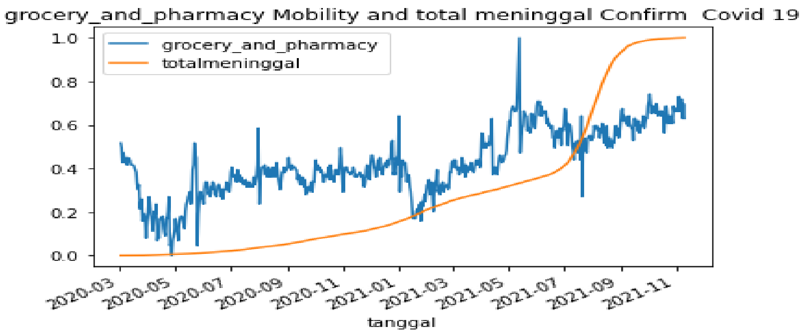

Figure 1.31 Visualization between grocery_and_pharmacy and Total died

Figure 1.31 shows a visualization between grocery_and pharmacy and total deaths, where when the total deaths are low, grocery_and_pharmacy is high, on the other hand, the total deaths are high, so grocery and pharmacy is low.

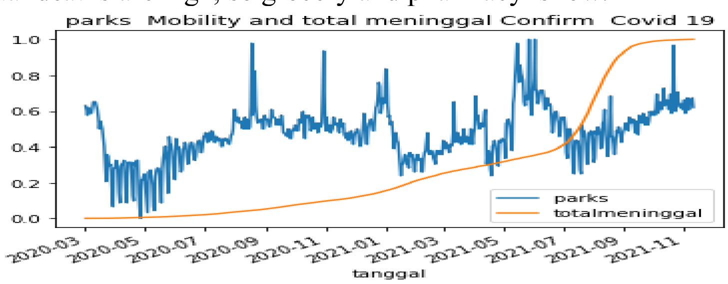

Figure 5.32 Visualization between Parks and Total died

Figure 1.32 shows a visualization between Parks and the total death toll, where when the total death toll is low, Parks is high, on the other hand, when the total death toll is high, Parks is low.

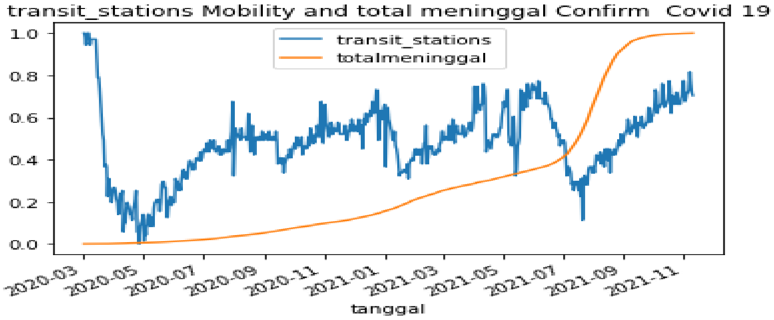

Figure 1.33 Visualization between transit_station and Total died

Figure 1.33 shows a visualization between transit_station and total deaths, where when the total death toll is low, the transit_station is high, otherwise the total death is high, the transit_station is low.

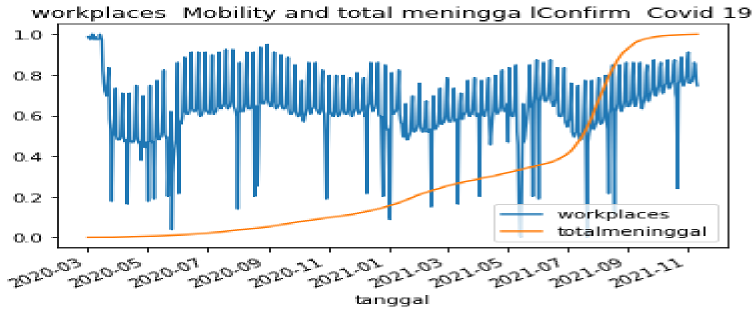

Figure 1.34 Visualization between workplaces and Total dies

Figure 1.34 shows a visualization between the workplaces and the total number of deaths, where when the total number of deaths is low, the workplaces are high, otherwise the total number of deaths is high, the workplaces are low.

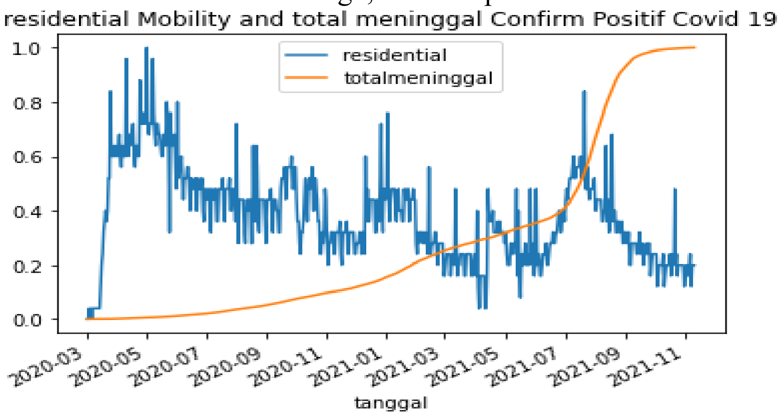

Figure 1.35 Visualization between Residential and Total Dies

Figure 1.35 shows a visualization between residential and total deaths, where when the total death toll is low, residential is high, on the other hand, the total death toll is high, the residential is low.

```
# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir26['transit_stations ']
list2 = dataakhir26['totalmeninggal']
# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
# Convert dataframe into series
list1 = dataakhir28['workplaces ']
list2 = dataakhir28['totalmeninggal']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")

# Convert dataframe into series
list1 = dataakhir29['residential ']
```

```
# Convert dataframe into series
list1 = dataakhir29['residential ']
list2 = dataakhir29['totalmeninggal']

# Apply the pearsonr()
corr, _ = pearsonr(list1, list2)
print('Pearsons correlation: %.3f' % corr)
print("\n")
```

```
Pearsons correlation: 0.607

Pearsons correlation: 0.733

Pearsons correlation: 0.338

Pearsons correlation: 0.318

Pearsons correlation: 0.174

Pearsons correlation: -0.402
```

Figure 1.36 Pearson Correlation Analysis between total deaths and community activities

Figure 5.36 shows the Pearson correlation value between total deaths and community mobility using the Pearson correlation formula. It can be seen that the highest correlation is 0.733, which is the correlation between total deaths and grocery and pharmacy, meaning that the correlation is quite high and positive (Prawoto, Priyo Purnomo, & Az Zahra, 2020).

```
my_r = x_simple.corr(method="spearman")
print(my_r)
print('\n')
```

```
                       retail_and_recreation  totalmeninggal
retail_and_recreation               1.000000        0.595956
totalmeninggal                      0.595956        1.000000

                      grocery_and_pharmacy  totalmeninggal
grocery_and_pharmacy              1.000000        0.784289
totalmeninggal                    0.784289        1.000000

                   parks  totalmeninggal
parks           1.000000        0.402212
totalmeninggal  0.402212        1.000000

                  transit_stations  totalmeninggal
transit_stations          1.000000        0.422612
totalmeninggal            0.422612        1.000000

                workplaces  totalmeninggal
workplaces        1.000000        0.158052
totalmeninggal    0.158052        1.000000

                residential  totalmeninggal
residential        1.000000       -0.506863
totalmeninggal    -0.506863        1.000000
```

Figure 1.37 Spearman Correlation Analysis between total deaths and community activities

Figure 1.37 shows the correlation value between total deaths and community mobility using the Spearman correlation formula. It can be seen that the highest correlation is 0.784, which is the correlation between total deaths and grocery and pharmacy, meaning that the correlation is quite high and positive.

4. Prediction of Number of New Cases Per Day using Long Short Term Memory (LSTM)

At the initial stage, determine the dataset_train, namely the number of new_cases_per_day and as the dataset_test is the number of new_cases_per_days, the code is as follows:

```
dataset_train= pd.DataFrame (training_set, columns = ['Jumlah_Kasus_Baru_per_Hari'])
dataset_test= pd.DataFrame (test_set, columns =['Jumlah_Kasus_Baru_per_Hari'])
```

Figure 1.38 Setting training data and testing data for prediction

Then import the packages needed for prediction.

```
# Mengimpor library yang diperlukan
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from keras.layers import Dense,Dropout,SimpleRNN,LSTM

# Proses feature scaling
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(training_set)
rentang=7
# Membuat prediksi dengan 60 time-window (3 bulan)
X_train = []
y_train = []
for i in range(rentang, training_set.shape[0]):
    X_train.append(training_set_scaled[i-rentang:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)

# Reshaping
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
# Mulai membuat RNN
Mesin_saham = Sequential()

# Menambah layer LSTM yang pertama dan Dropout regularisation
Mesin_saham.add(SimpleRNN(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
Mesin_saham.add(Dropout(0.2))

# Menambah layer LSTM yang kedua dan Dropout regularisation
Mesin_saham.add(SimpleRNN(units = 50, return_sequences = True))
Mesin_saham.add(Dropout(0.2))

# Menambah layer LSTM yang ketiga dan Dropout regularisation
Mesin_saham.add(SimpleRNN(units = 50, return_sequences = True))
Mesin_saham.add(Dropout(0.2))
Mesin_saham.add(Dropout(0.2))

# Menambahkan output layer
Mesin_saham.add(Dense(units = 1))

# Melihat rancangan network LSTM kita
Mesin_saham.summary()

# Compile RNN
Mesin_saham.compile(optimizer = 'adam', loss = 'mean_squared_error', metrics=["acc"])

# Menjalankan RNN ke Training set
hist =Mesin_saham.fit(X_train, y_train,  validation_split=0.3,epochs = 100, batch_size = 32,verbose=2

# Mengimpor data saham sesungguhnya untuk Test set

X_test = []
for i in range(rentang, saham_real.shape[0] + X_train.shape[1]):
    X_test.append(inputs[i-rentang:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))

predicted_stock_price = Mesin_saham.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)

# Visualisasi perbandingan hasil prediksi dan data sesunguhnya
plt.plot(saham_real, color = 'red', label = 'Jumlah_Kasus_Baru_per_Hari sesungguhnya')
plt.plot(predicted_stock_price, color = 'blue', label = 'Jumlah_Kasus_Baru_per_Hari prediksi')
plt.title('Prediksi Jumlah_Kasus_Baru_per_Hari')
plt.xlabel('Waktu')
plt.ylabel('Prediksi Jumlah Kasus Baru Covid 1 per Hari')
plt.legend()
```

Figure 1.39 Importing packages needed for prediction

The next process is to build the model with epoch.

```
Total params: 17,801
Trainable params: 17,801
Non-trainable params: 0
_________________________________________________________________
Epoch 1/100
2021-12-22 00:35:06.724945: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR Optimization
Passes are enabled (registered 2)
13/13 - 5s - loss: 0.1170 - acc: 0.0025 - val_loss: 0.0461 - val_acc: 0.0057
Epoch 2/100
13/13 - 0s - loss: 0.0312 - acc: 0.0025 - val_loss: 0.0369 - val_acc: 0.0000e+00
Epoch 3/100
13/13 - 0s - loss: 0.0207 - acc: 0.0025 - val_loss: 0.0291 - val_acc: 0.0000e+00
Epoch 4/100
13/13 - 0s - loss: 0.0133 - acc: 0.0025 - val_loss: 0.0153 - val_acc: 0.0000e+00
Epoch 5/100
13/13 - 0s - loss: 0.0101 - acc: 0.0025 - val_loss: 0.0140 - val_acc: 0.0057
Epoch 6/100
13/13 - 0s - loss: 0.0089 - acc: 0.0025 - val_loss: 0.0234 - val_acc: 0.0000e+00
Epoch 7/100
13/13 - 0s - loss: 0.0060 - acc: 0.0025 - val_loss: 0.0159 - val_acc: 0.0057
```

Figure 1.40 The process of building a predictive model

Prediction results are visualized as follows:

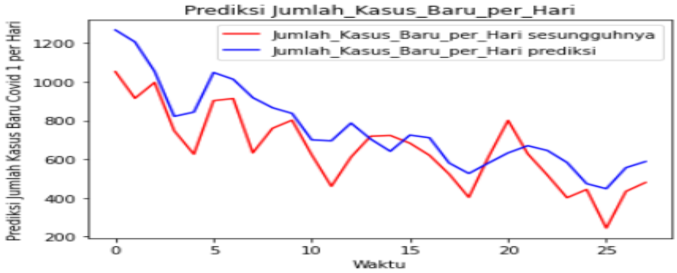

Figure 1.41 Predicted number of cases per day

After making predictions, then testing the models that have been obtained and making visualizations.

```
import matplotlib.pyplot as plt

fig, ax = plt.subplots(figsize=(14,6))
ax.plot(hist.history['loss'], 'b' ,label = 'train loss', linewidth=2)
ax.plot(hist.history['val_loss'], 'r', label ='Validation loss', linewidth=2)
ax.set_title('model loss')
ax.set_ylabel('mse')
ax.set_xlabel('epoch')
ax.legend()
plt.show()
```

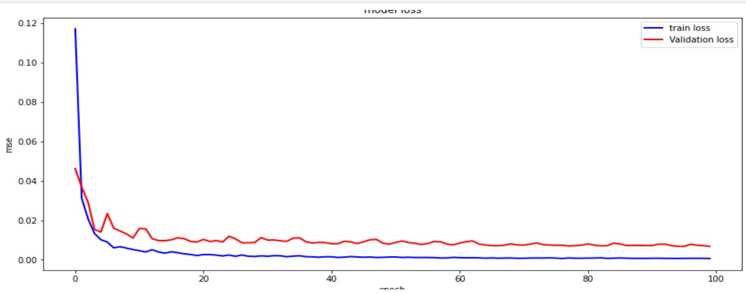

Figure 1. 42 Prediction Graph

From Figure 5.42 it can be seen that between the validation loss values (red) and train loss (blue), the graphs are close together. This shows that the predictions made are quite accurate and are also shown in Figure 5.43 with a value of RMSE = 145,135 which is quite small.

```
from sklearn.metrics import mean_squared_error
from math import sqrt
rmse = sqrt(mean_squared_error(saham_real, predicted_stock_price))
print('Test RMSE: %.3f' % rmse)

Test RMSE: 145.135
```

Figure 1.43 Evaluation Results using RMSE

5. Predicting the Number of Cumulative Cases using Long Short Term Memory (LSTM)

At the initial stage, determine the dataset_train, namely the number of new cases per day and as the dataset test is the Cumulative Number of Cases, the coding is as follows:

```
training_seta = df.iloc[:, 2:3].values
```

```
training_set=training_seta[0:590]
```

```
test_set=training_seta[590:618]
```

```
dataset_train= pd.DataFrame (training_set, columns = ['Jumlah_Kasus_Kumulatif'])
dataset_test= pd.DataFrame (test_set, columns =['Jumlah_Kasus_Kumulatif'])
```

Figure 1.43 Setting training data and testing data for prediction

Then import the packages needed for prediction.

```
# Mengimpor library yang diperlukan
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from keras.layers import Dense,Dropout,SimpleRNN,LSTM

# Proses feature scaling
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(training_set)
rentang=7
# Membuat prediksi dengan 60 time-window (3 bulan)
X_train = []
y_train = []
for i in range(rentang, training_set.shape[0]):
    X_train.append(training_set_scaled[i-rentang:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)
# Reshaping
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))

# Mengimpor library Keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM,SimpleRNN
from tensorflow.keras.layers import Dropout

# Mulai membuat RNN
Mesin_saham = Sequential()

# Menambah layer LSTM yang pertama dan Dropout regularisation
Mesin_saham.add(SimpleRNN(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
Mesin_saham.add(Dropout(0.2))

# Menambah layer LSTM yang kedua dan Dropout regularisation
Mesin_saham.add(SimpleRNN(units = 50, return_sequences = True))
Mesin_saham.add(Dropout(0.2))
```

```
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(rentang, saham_real.shape[0] + X_train.shape[1]):
    X_test.append(inputs[i-rentang:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))

predicted_stock_price = Mesin_saham.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)

# Visualisasi perbandingan hasil prediksi dan data sesunguhnya
plt.plot(saham_real, color = 'red', label = 'Jumlah_Kasus_Kumulatif sesungguhnya')
plt.plot(predicted_stock_price, color = 'blue', label = 'Jumlah_Kasus_Kumulatif prediksi')
plt.title('Prediksi Jumlah_Kasus_Kumulatif')
plt.xlabel('Waktu')
plt.ylabel('Prediksi Jumlah Kumulatif Covid')
plt.legend()
plt.show()
```

```
Model: "sequential_1"
```

Figure 1. 44 Setting up the necessary packages for modeling

The next step is to build a model using epochs and then visualize it.

```
Epoch 1/100
13/13 - 4s - loss: 0.1989 - acc: 0.0000e+00 - val_loss: 0.0651 - val_acc: 0.0057
Epoch 2/100
13/13 - 0s - loss: 0.0861 - acc: 0.0000e+00 - val_loss: 0.1286 - val_acc: 0.0000e+00
Epoch 3/100
13/13 - 0s - loss: 0.0470 - acc: 0.0000e+00 - val_loss: 0.2368 - val_acc: 0.0000e+00
Epoch 4/100
13/13 - 0s - loss: 0.0486 - acc: 0.0000e+00 - val_loss: 0.0290 - val_acc: 0.0057
Epoch 5/100
13/13 - 0s - loss: 0.0346 - acc: 0.0000e+00 - val_loss: 0.0480 - val_acc: 0.0057
Epoch 6/100
13/13 - 0s - loss: 0.0301 - acc: 0.0000e+00 - val_loss: 0.0934 - val_acc: 0.0057
Epoch 7/100
13/13 - 0s - loss: 0.0223 - acc: 0.0000e+00 - val_loss: 0.0207 - val_acc: 0.0057
Epoch 8/100
13/13 - 0s - loss: 0.0196 - acc: 0.0000e+00 - val_loss: 0.0493 - val_acc: 0.0057
Epoch 9/100
13/13 - 0s - loss: 0.0166 - acc: 0.0000e+00 - val_loss: 0.0344 - val_acc: 0.0057
Epoch 10/100
13/13 - 0s - loss: 0.0139 - acc: 0.0000e+00 - val_loss: 0.0542 - val_acc: 0.0057
Epoch 11/100
13/13 - 0s - loss: 0.0126 - acc: 0.0000e+00 - val_loss: 0.0552 - val_acc: 0.0057
Epoch 12/100
13/13 - 0s - loss: 0.0147 - acc: 0.0000e+00 - val_loss: 0.0096 - val_acc: 0.0057
Epoch 13/100
13/13 - 0s - loss: 0.0134 - acc: 0.0000e+00 - val_loss: 0.0244 - val_acc: 0.0057
Epoch 14/100
13/13 - 0s - loss: 0.0108 - acc: 0.0000e+00 - val_loss: 0.0078 - val_acc: 0.0057
Epoch 15/100
13/13 - 0s - loss: 0.0099 - acc: 0.0000e+00 - val_loss: 0.0194 - val_acc: 0.0057
```

Figure 1.45 Model building process

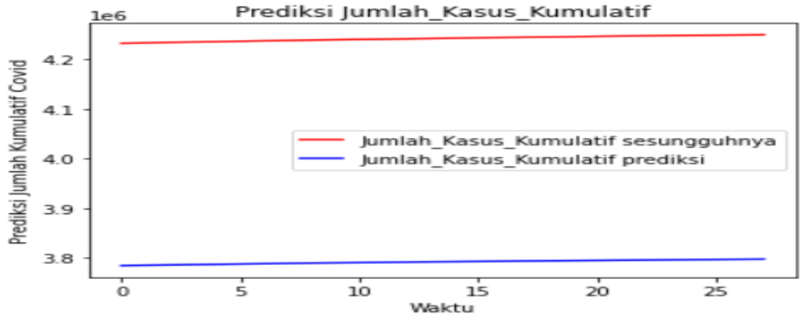

Figure 1. 46 Visualization of the built prediction model

```
fig, ax = plt.subplots(figsize=(14,6))
ax.plot(hist.history['loss'], 'b' ,label = 'train loss', linewidth=2)
ax.plot(hist.history['val_loss'], 'r', label ='Validation loss', linewidth=2)
ax.set_title('model loss')
ax.set_ylabel('mse')
ax.set_xlabel('epoch')
ax.legend()
plt.show()
```

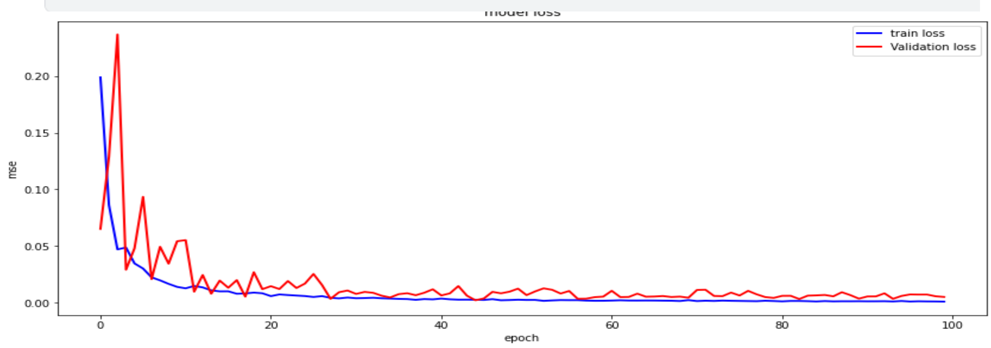

Figure 1.47 Prediction graph

From Figure 1.47 it can be seen that between the validation loss values (red) and the train loss (blue), the graphs are close together. This shows that the predictions made are quite accurate and are also shown in Figure 1.48 with a sufficient value of RMSE = 449516,694.

```
rmse = sqrt(mean_squared_error(saham_real, predicted_stock_price))
print('Test RMSE: %.3f' % rmse)
```

```
Test RMSE: 449516.694
```

Figure 5.48 Evaluation Results using RMSE

6. Prediction of the Number of Cumulative Death Cases using Long Short Term Memory (LSTM)

At the initial stage, the dataset_train is determined, namely the cumulative number of death cases and as the dataset_test is the Cumulative Number of Cases Death Cumulative, the coding is as follows:

```
training_seta = df.iloc[:, 3:4].values
training_set=training_seta[0:590]
test_set=training_seta[590:618]
```

```
dataset_train= pd.DataFrame (training_set, columns = ['Jumlah_Kasus_Kematian_Kumulatif'])
dataset_test= pd.DataFrame (test_set, columns =['Jumlah_Kasus_Kematian_Kumulatif'])
```

Figure 1.49 Setting training data and testing data for prediction

```
# Mengimpor library yang diperlukan
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from keras.layers import Dense,Dropout,SimpleRNN,LSTM

# Proses feature scaling
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(training_set)
rentang=7
# Membuat prediksi dengan 60 time-window (3 bulan)
X_train = []
y_train = []
for i in range(rentang, training_set.shape[0]):
    X_train.append(training_set_scaled[i-rentang:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)
```

```
# Memprediksi harga saham
dataset_total = pd.concat((dataset_train['Jumlah_Kasus_Kematian_Kumulatif'], dataset_test['Jumlah_Kasu
inputs = dataset_total[len(dataset_total) - len(dataset_test) - rentang:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(rentang, saham_real.shape[0] + X_train.shape[1]):
    X_test.append(inputs[i-rentang:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))

predicted_stock_price = Mesin_saham.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)

# Visualisasi perbandingan hasil prediksi dan data sesunguhnya
plt.plot(saham_real, color = 'red', label = 'Jumlah_Kasus_Kematian Kumulatif sesungguhnya')
plt.plot(predicted_stock_price, color = 'blue', label = 'Jumlah_Kasus_Kematian Kumulatif prediksi')
plt.title('Prediksi Jumlah_Kasus_Baru_per_Hari')
plt.xlabel('Waktu')
plt.ylabel('Prediksi Jumlah Kasus Baru Covid 1 per Hari')
plt.legend()
plt.show()
```

Figure 1.50 Importing packages needed for prediction

The next step is to build a model using epochs and then visualize it

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
simple_rnn_8 (SimpleRNN)     (None, 7, 50)             2600
_________________________________________________________________
dropout_8 (Dropout)          (None, 7, 50)             0
_________________________________________________________________
simple_rnn_9 (SimpleRNN)     (None, 7, 50)             5050
_________________________________________________________________
dropout_9 (Dropout)          (None, 7, 50)             0
_________________________________________________________________
simple_rnn_10 (SimpleRNN)    (None, 7, 50)             5050
_________________________________________________________________
dropout_10 (Dropout)         (None, 7, 50)             0
_________________________________________________________________
simple_rnn_11 (SimpleRNN)    (None, 50)                5050
_________________________________________________________________
dropout_11 (Dropout)         (None, 50)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 51
=================================================================
Total params: 17,801
Trainable params: 17,801
Non-trainable params: 0
```

Figure 1.51 The process of building a predictive model

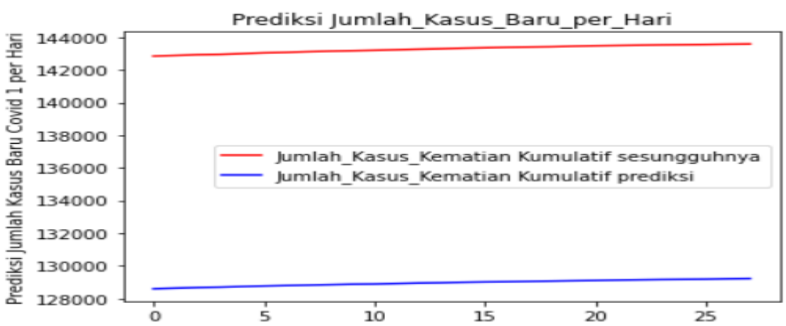

Figure 1. 52 Visualization of the built prediction model

```python
fig, ax = plt.subplots(figsize=(14,6))
ax.plot(hist.history['loss'], 'b' ,label = 'train loss', linewidth=2)
ax.plot(hist.history['val_loss'], 'r', label ='Validation loss', linewidth=2)
ax.set_title('model loss')
ax.set_ylabel('mse')
ax.set_xlabel('epoch')
ax.legend()
plt.show()
```

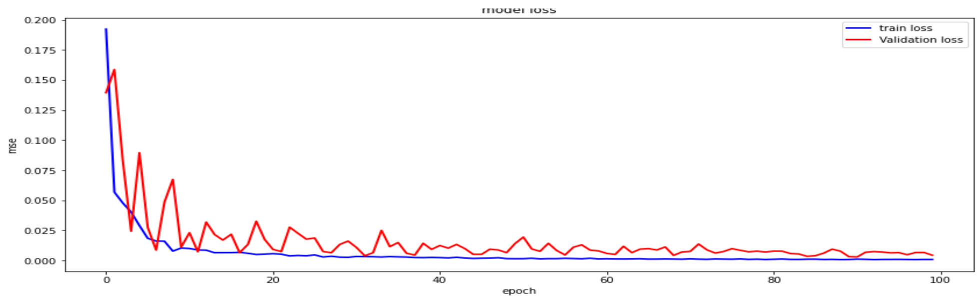

Figure 1.53 Prediction graph

From Figure 1.53, it can be seen that between the validation loss values (red) and the train loss (blue), the graphs are close together. This shows that the predictions made are quite accurate and are also shown in Figure 5.54 with a value of RMSE = 14331,656 which is quite small.

```python
rmse = sqrt(mean_squared_error(saham_real, predicted_stock_price))
print('Test RMSE: %.3f' % rmse)
```

```
Test RMSE: 14331.656
```

Figure 1.54 Evaluation Results using RMSE

CONCLUSION

Based on the results of research that has been carried out on positive confirmed COVID-19 data downloaded from Google Trend from January 1, 2020 to November 10, 2021 with 617 records including daily case variables, total cases, total deaths with global mobility variables (community activities) including retail and recreation, grocery and pharmacy, parks, transit stations, workplaces, a model has been obtained to predict the number of cases per day, predict the number of cumulative cases, and the number of cumulative deaths.

The best prediction result is the prediction of the number of cases per day with an RMSE = 145,135. Meanwhile, the highest correlation analysis is 0.784 between the total death variable and grocery and pharmacy.

REFERENCES

Ahmetolan, Semra, Bilge, Ayse Humeyra, Demirci, Ali, Peker-Dobie, Ayse, & Ergonul, Onder. (2020). What can we estimate from fatality and infectious case data using the susceptible-infected-removed (SIR) model? A case study of Covid-19 pandemic. *Frontiers in Medicine*, *7*, 556366.

Devaraj, Jayanthi, Elavarasan, Rajvikram Madurai, Pugazhendhi, Rishi, Shafiullah, G. M., Ganesan, Sumathi, Jeysree, Ajay Kaarthic, Khan, Irfan Ahmad, & Hossain, Eklas. (2021). Forecasting of COVID-19 cases using deep learning models: Is it reliable and practically significant? *Results in Physics*, *21*, 103817.

Gers, Felix A., & Schmidhuber, E. (2001). LSTM recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, *12*(6), 1333–1340.

Jun, Seung Pyo, Yoo, Hyoung Sun, & Choi, San. (2018). Ten years of research change using Google Trends: From the perspective of big data utilizations and applications. *Technological Forecasting and Social Change*, *130*, 69–87.

Kondo, Kenjiro, Ishikawa, Akihiko, & Kimura, Masashi. (2019). Sequence to sequence with attention for influenza prevalence prediction using google trends. *Proceedings of the 2019 3rd International Conference on Computational Biology and Bioinformatics*, 1–7.

Maaliw, Renato R., Mabunga, Zoren P., & Villa, Frederick T. (2021). Time-Series Forecasting of COVID-19 Cases Using Stacked Long Short-Term Memory Networks. *2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, 435–441. IEEE.

Pan, Zhenhe, Nguyen, Hoang Long, Abu-Gellban, Hashim, & Zhang, Yuanlin. (2020). Google trends analysis of covid-19 pandemic. *2020 IEEE International Conference on Big Data (Big Data)*, 3438–3446. IEEE.

Prawoto, Nano, Priyo Purnomo, Eko, & Az Zahra, Abitassha. (2020). *The impacts of Covid-19 pandemic on socio-economic mobility in Indonesia*.

Pretorius, A., Kruger, E., & Bezuidenhout, S. (2022). Google trends and water conservation awareness: the internet's contribution in South Africa. *South African Geographical Journal*, *104*(1), 53–69.

Sak, Hasim, Senior, Andrew W., & Beaufays, Françoise. (2014). *Long short-term memory recurrent neural network architectures for large scale acoustic modeling*.

Shahid, Farah, Zameer, Aneela, & Muneeb, Muhammad. (2020). Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM. *Chaos, Solitons & Fractals*, *140*, 110212.

Sharpe Jr, Richard E., Kuszyk, Brian S., & Mossa-Basha, Mahmud. (2021). Special report of the RSNA COVID-19 Task Force: the short-and long-term financial impact of the COVID-19 pandemic on private radiology practices. *Radiology*.

Wen, Jun, Kozak, Metin, Yang, Shaohua, & Liu, Fang. (2020). COVID-19: potential effects on Chinese citizens' lifestyle and travel. *Tourism Review*, *76*(1), 74–87.

Zhang, Kefei, Thé, Jesse, Xie, Guangyuan, & Yu, Hesheng. (2020). Multi-step ahead forecasting of regional air quality using spatial-temporal deep neural networks: a case study of Huaihai Economic Zone. *Journal of Cleaner Production*, *277*, 123231.