Polynomial , KNN and Regression Models

Prabin Kharel

Supervised Learning with Polynomial, KNN and ANN regression model

```
library(readxl)
covid_data <- read_xlsx("/home/prabin/Downloads/covid_tbl_final.xlsx")
head(covid_data)
```

Loading covid_tbl_final.xslx

```
## # A tibble: 6 x 14
##      SN Date                Confirmed_cases_t~ Confirmed_cases~ `Confirmed _ca~`
##   <dbl> <dttm>                           <dbl>            <dbl>            <dbl>
## 1     1 2020-01-23 00:00:00                  1                1                1
## 2     2 2020-01-24 00:00:00                  1                0                1
## 3     3 2020-01-25 00:00:00                  1                0                1
## 4     4 2020-01-26 00:00:00                  1                0                1
## 5     5 2020-01-27 00:00:00                  1                0                1
## 6     6 2020-01-28 00:00:00                  1                0                1
## # ... with 9 more variables: Recoveries_total <dbl>, Recoveries_daily <dbl>,
## #   Deaths_total <dbl>, Deaths_daily <dbl>, `RT-PCR_tests_total` <dbl>,
## #   `RT-PCR_tests_daily` <dbl>, Test_positivity_rate <dbl>,
## #   Recovery_rate <dbl>, Case_fatality_rate <dbl>
```

```
str(covid_data)
```

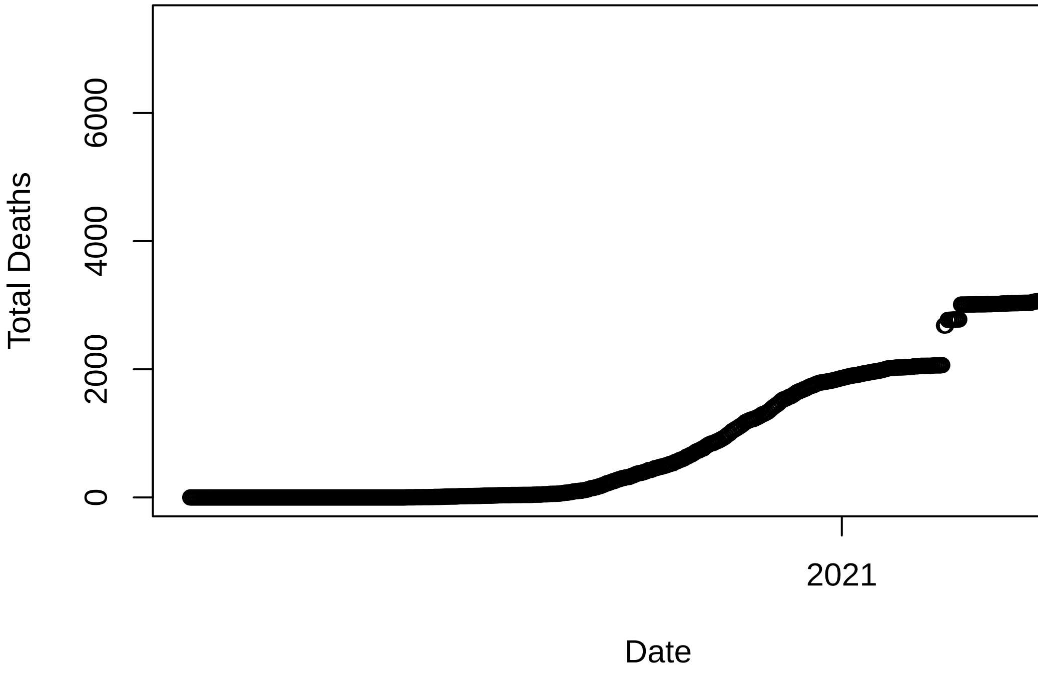
Cleaning covid_data

```
## tibble [495 x 14] (S3: tbl_df/tbl/data.frame)
##  $ SN                    : num [1:495] 1 2 3 4 5 6 7 8 9 10 ...
##  $ Date                  : POSIXct[1:495], format: "2020-01-23" "2020-01-24" ...
##  $ Confirmed_cases_total : num [1:495] 1 1 1 1 1 1 1 1 1 1 ...
##  $ Confirmed_cases_new   : num [1:495] 1 0 0 0 0 0 0 0 0 0 ...
##  $ Confirmed _cases_active: num [1:495] 1 1 1 1 1 1 0 0 0 0 ...
##  $ Recoveries_total      : num [1:495] 0 0 0 0 0 0 1 1 1 1 ...
##  $ Recoveries_daily      : num [1:495] 0 0 0 0 0 0 1 0 0 0 ...
##  $ Deaths_total          : num [1:495] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Deaths_daily          : num [1:495] 0 0 0 0 0 0 0 0 0 0 ...
##  $ RT-PCR_tests_total    : num [1:495] NA NA NA NA NA 3 4 5 5 NA ...
##  $ RT-PCR_tests_daily    : num [1:495] NA NA NA NA NA NA 1 1 0 NA ...
##  $ Test_positivity_rate  : num [1:495] NA NA NA NA NA ...
##  $ Recovery_rate         : num [1:495] 0 0 0 0 0 0 100 100 100 100 ...
##  $ Case_fatality_rate    : num [1:495] 0 0 0 0 0 0 0 0 0 0 ...
```

```
covid_data$Date<-as.Date(as.POSIXct(covid_data$Date))
str(covid_data)
```

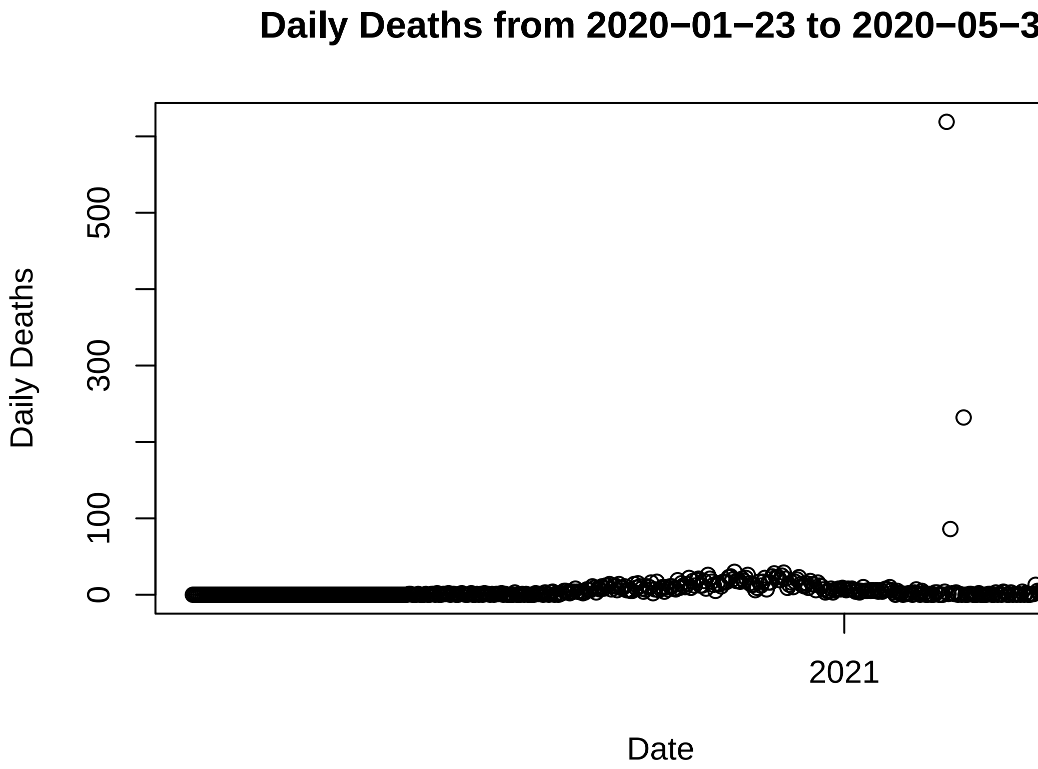
```
## tibble [495 x 14] (S3: tbl_df/tbl/data.frame)
##  $ SN                    : num [1:495] 1 2 3 4 5 6 7 8 9 10 ...
##  $ Date                  : Date[1:495], format: "2020-01-23" "2020-01-24" ...
##  $ Confirmed_cases_total  : num [1:495] 1 1 1 1 1 1 1 1 1 1 ...
##  $ Confirmed_cases_new    : num [1:495] 1 0 0 0 0 0 0 0 0 0 ...
##  $ Confirmed _cases_active: num [1:495] 1 1 1 1 1 1 0 0 0 0 ...
##  $ Recoveries_total       : num [1:495] 0 0 0 0 0 0 1 1 1 1 ...
##  $ Recoveries_daily       : num [1:495] 0 0 0 0 0 0 1 0 0 0 ...
##  $ Deaths_total           : num [1:495] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Deaths_daily           : num [1:495] 0 0 0 0 0 0 0 0 0 0 ...
##  $ RT-PCR_tests_total     : num [1:495] NA NA NA NA NA 3 4 5 5 NA ...
##  $ RT-PCR_tests_daily     : num [1:495] NA NA NA NA NA NA 1 1 0 NA ...
##  $ Test_positivity_rate   : num [1:495] NA NA NA NA NA ...
##  $ Recovery_rate          : num [1:495] 0 0 0 0 0 0 100 100 100 100 ...
##  $ Case_fatality_rate     : num [1:495] 0 0 0 0 0 0 0 0 0 0 ...
```

```
plot(covid_data$Date,covid_data$Deaths_total, xlab="Date",ylab="Total Deaths")
```



Plotting total deaths vs date

```
plot(covid_data$Date,covid_data$Deaths_daily,main="Daily Deaths from 2020-01-23 to 2020-05-31", xlab="Da
```



Plotting daily deaths vs date

```
summary(covid_data$Deaths_daily)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    0.00    2.00   14.92   11.00  619.00
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
filter(covid_data, Deaths_daily >=50 & Date <=as.Date("2021-03-05"))
```

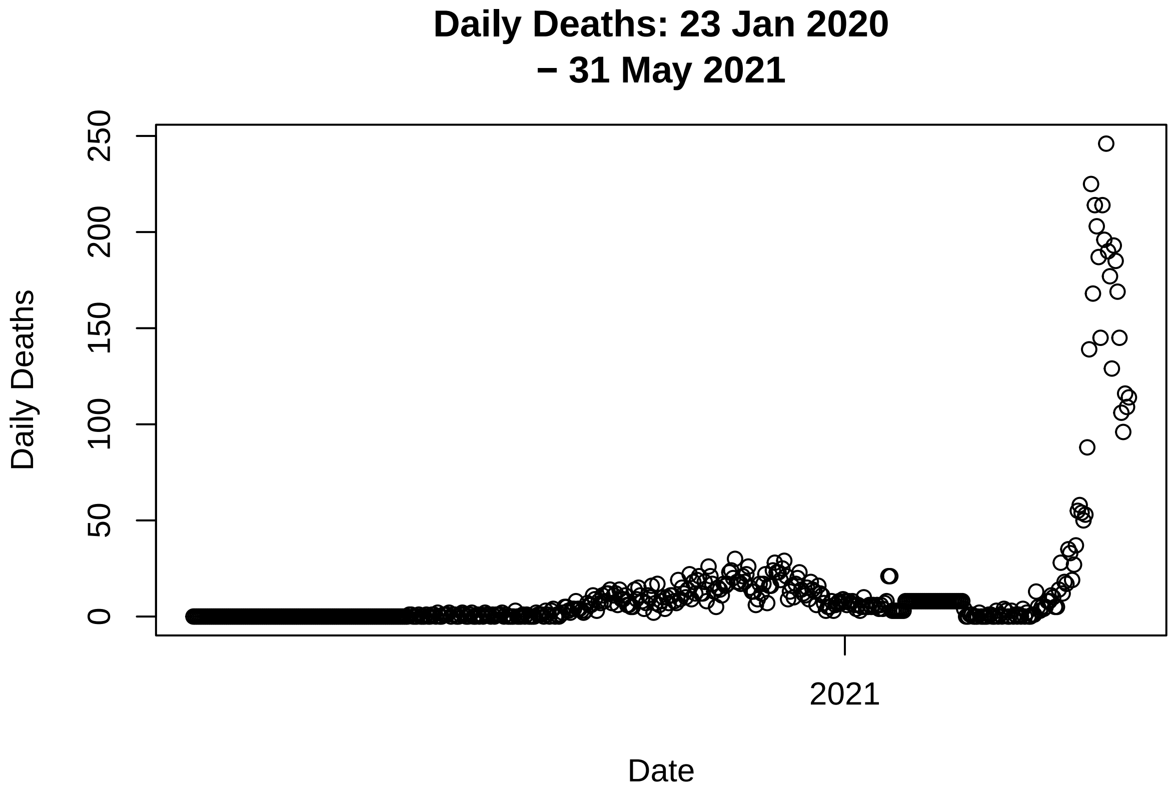
```
## # A tibble: 3 x 14
##      SN Date       Confirmed_cases_total Confirmed_cases_new `Confirmed _cases~`
##   <dbl> <date>                     <dbl>               <dbl>               <dbl>
## 1   399 2021-02-24                273760                  94                 937
## 2   401 2021-02-26                273984                 112                 936
## 3   408 2021-03-05                274608                 120                 832
## # ... with 9 more variables: Recoveries_total <dbl>, Recoveries_daily <dbl>,
## #   Deaths_total <dbl>, Deaths_daily <dbl>, `RT-PCR_tests_total` <dbl>,
## #   `RT-PCR_tests_daily` <dbl>, Test_positivity_rate <dbl>,
## #   Recovery_rate <dbl>, Case_fatality_rate <dbl>
```

```
wsn<-c(399,401,408)
for(i in 1:length(wsn)){

temp_sn = wsn[i]
# Get the Value to be adjusted
curr_val<-covid_data[covid_data$SN==temp_sn,"Deaths_daily"]
# Calculate the average daily deaths for last 30 days
avg_daily_deaths<-ceiling(mean(covid_data[covid_data$SN %in% c((temp_sn-1):(temp_sn-1-30)),]$Deaths_dai

# Change the Value for given SN
covid_data[covid_data$SN==temp_sn,"Deaths_daily"]=avg_daily_deaths
# Change values for last 30 days
covid_data[covid_data$SN %in% c((temp_sn-1):(temp_sn-1-30)),]$Deaths_daily=as.integer( round(curr_val/3
}
```

```
plot(covid_data$Date,
covid_data$Deaths_daily,
main = "Daily Deaths: 23 Jan 2020
- 31 May 2021",
xlab = "Date",
ylab = "Daily Deaths")
```



Since the data is now clean, we divide the data into training and testing set for our regression mode

```
set.seed(1234)
ind <- sample(2, nrow(covid_data), replace=T, prob=c(0.7,0.3))
train_data <- covid_data[ind==1,]
test_data <- covid_data[ind==2,]
```

Splitting the data into training and testing set

```
library(caret)
```

Linear regression model

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
lm1 <- train(Deaths_daily~SN, data = train_data, method="lm")
summary(lm1)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -35.658 -11.892  -2.591   4.622 205.538
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.50449    3.28569  -4.110 4.95e-05 ***
## SN            0.11173    0.01169   9.561  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.51 on 345 degrees of freedom
## Multiple R-squared:  0.2094, Adjusted R-squared:  0.2072
## F-statistic:  91.4 on 1 and 345 DF,  p-value: < 2.2e-16
```

```
predict1 <- predict(lm1, newdata = test_data)
```

######Evaluating metrics

```
R2 <- R2(predict1,test_data$Deaths_daily)
RMSE <- RMSE(predict1,test_data$Deaths_daily)
MAE <- MAE(predict1,test_data$Deaths_daily)
R2
```

```
## [1] 0.1887896
```

```
RMSE
```

```
## [1] 32.1613
```

```
MAE
```

```
## [1] 17.61361
```

```
lm2 <- train(Deaths_daily~poly(SN,2),data=train_data, method="lm")
summary(lm2)
```

Quadratic linear regression model

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.544 -10.617   1.553   6.616 181.775
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        13.427      1.558   8.618 2.52e-16 ***
## `poly(SN, 2)1`    301.229     29.022  10.379  < 2e-16 ***
## `poly(SN, 2)2`    229.647     29.022   7.913 3.48e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29.02 on 344 degrees of freedom
## Multiple R-squared:  0.3312, Adjusted R-squared:  0.3273
## F-statistic: 85.17 on 2 and 344 DF,  p-value: < 2.2e-16
```

```
predict2 <- predict(lm2,newdata = test_data)
```

```
head(predict2)
```

```
##         1         2         3         4         5         6
## 11.310983  9.504666  9.117869  7.263563  6.908637  6.733166
```

```
R2 <- R2(predict2,test_data$Deaths_daily)
RMSE <- RMSE(predict2,test_data$Deaths_daily)
MAE <- MAE(predict2,test_data$Deaths_daily)
R2
```

```
## [1] 0.3143297
```

```
RMSE
```

```
## [1] 29.52953
```

```
MAE
```

```
## [1] 18.11123
```

```
lm3 <- train(Deaths_daily~poly(SN,3), data = train_data, method="lm")
summary(lm3)
```

Cubic linear regression model

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max 
## -56.401  -9.822  -2.567  10.088 157.909 
## 
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)    
## (Intercept)       13.427      1.404   9.566   <2e-16 ***
## `poly(SN, 3)1`   301.229     26.145  11.522   <2e-16 ***
## `poly(SN, 3)2`   229.647     26.145   8.784   <2e-16 ***
## `poly(SN, 3)3`   235.151     26.145   8.994   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 26.14 on 343 degrees of freedom
## Multiple R-squared:  0.4588, Adjusted R-squared:  0.4541 
## F-statistic: 96.93 on 3 and 343 DF,  p-value: < 2.2e-16
```

```
predict3 <- predict(lm3,newdata = test_data)
head(predict3)
```

```
##          1          2          3          4          5          6 
## -16.281088 -11.650850 -10.691909  -6.265272  -5.451722  -5.053706
```

```
R2 <- R2(predict3,test_data$Deaths_daily)
RMSE <- RMSE(predict3,test_data$Deaths_daily)
MAE <- MAE(predict3,test_data$Deaths_daily)
R2
```

```
## [1] 0.4823308
```

```
RMSE
```

```
## [1] 25.6787
```

```
MAE
```

```
## [1] 16.66555
```

```

lm4 <- train(Deaths_daily~poly(SN,4), data=train_data, method="lm")
summary(lm4)
```

Double quadratic linear model

```
## 
## Call:
## lm(formula = .outcome ~ ., data = dat)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -53.511  -9.839   1.374   8.894 133.202 
## 
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)    
## (Intercept)       13.427      1.166   11.52   <2e-16 ***
## `poly(SN, 4)1`   301.229     21.720   13.87   <2e-16 ***
## `poly(SN, 4)2`   229.647     21.720   10.57   <2e-16 ***
## `poly(SN, 4)3`   235.151     21.720   10.83   <2e-16 ***
## `poly(SN, 4)4`   270.390     21.720   12.45   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.72 on 342 degrees of freedom
## Multiple R-squared:  0.6276, Adjusted R-squared:  0.6232
## F-statistic: 144.1 on 4 and 342 DF,  p-value: < 2.2e-16
```

```
predict4 <- predict(lm4, newdata = test_data)
head(predict4)
```

```
##          1          2          3          4          5          6
## 15.9429213  7.9565969  6.4173420 -0.0976663 -1.1777845 -1.6915056
```

```
R2 <- R2(predict4,test_data$Deaths_daily)
RMSE <- RMSE(predict4,test_data$Deaths_daily)
MAE <- MAE(predict4,test_data$Deaths_daily)
R2
```

```
## [1] 0.6857402
```

```
RMSE
```

```
## [1] 19.98498
```

```
MAE
```

```
## [1] 14.03474
```

```

lm5 <- train(Deaths_daily ~SN, data= train_data, methos="lm")
summary(lm5)
```

Fifth order polynomial regression model

```
##                 Length Class      Mode
## call              5    -none-     call
## type              1    -none-     character
## predicted       347    -none-     numeric
## mse             500    -none-     numeric
## rsq             500    -none-     numeric
## oob.times       347    -none-     numeric
## importance        1    -none-     numeric
## importanceSD      0    -none-     NULL
## localImportance   0    -none-     NULL
## proximity         0    -none-     NULL
## ntree             1    -none-     numeric
## mtry              1    -none-     numeric
## forest           11    -none-     list
## coefs             0    -none-     NULL
## y               347    -none-     numeric
## test              0    -none-     NULL
## inbag             0    -none-     NULL
## xNames            1    -none-     character
## problemType       1    -none-     character
## tuneValue         1    data.frame list
## obsLevels         1    -none-     logical
## param             1    -none-     list
```

```
predict5 <- predict(lm5, newdata = test_data)
head(predict5)
```

```
##             1             2             3             4             5 
## -6.220802e-15 -7.407408e-15 -7.545964e-15 -8.260059e-15 -8.245848e-15 
##             6 
## -8.277823e-15
```

```
R2 <- R2(predict5,test_data$Deaths_daily)
RMSE <- RMSE(predict5,test_data$Deaths_daily)
MAE <- MAE(predict5,test_data$Deaths_daily)
R2
```

```
## [1] 0.9520691
```

```
RMSE
```

```
## [1] 7.985125
```

```
MAE
```

```
## [1] 3.207969
```

```

knnmodel <- train(Deaths_daily ~SN, data=train_data, method="knn")
summary(knnmodel)
```

KNN regression model

```
##             Length Class      Mode     
## learn       2      -none-     list     
## k           1      -none-     numeric  
## theDots     0      -none-     list     
## xNames      1      -none-     character
## problemType 1      -none-     character
## tuneValue   1      data.frame list     
## obsLevels   1      -none-     logical  
## param       0      -none-     list
```

```
predict6 <- predict(knnmodel, newdata = test_data)
head(predict6)
```

```
## [1] 0 0 0 0 0 0
```

```
R2 <- R2(predict6,test_data$Deaths_daily)
RMSE <- RMSE(predict6,test_data$Deaths_daily)
MAE <- MAE(predict6,test_data$Deaths_daily)
R2
```

```
## [1] 0.9777022
```

```
RMSE
```

```
## [1] 5.806763
```

```
MAE
```

```
## [1] 2.827703
```

```
#install.packages("neuralnet")
library(neuralnet)
neural1 <- neuralnet(Deaths_daily ~ SN, data= train_data, hidden=c(3,2),linear.output = F)
plot(neural1, main="Neural network with 2 hidden layers with 3 and 2 neurons")
summary(neural1)
```

ANN-MLP regression model with 2 hidden layers with 3 and 2 neurons

```
##                     Length Class      Mode    
## call                  5    -none-     call    
## response            347    -none-     numeric 
## covariate           347    -none-     numeric 
## model.list            2    -none-     list    
## err.fct               1    -none-     function
## act.fct               1    -none-     function
## linear.output         1    -none-     logical 
## data                 14    data.frame list    
## exclude               0    -none-     NULL    
## net.result            1    -none-     list    
## weights               1    -none-     list    
## generalized.weights   1    -none-     list    
## startweights          1    -none-     list    
## result.matrix        20    -none-     numeric 
```

```
predict7 <- predict(neural1, newdata = test_data)
head(predict7)
```

```
##           [,1]
## [1,] 0.9999980
## [2,] 0.9999980
## [3,] 0.9999980
## [4,] 0.9999981
## [5,] 0.9999981
## [6,] 0.9999981
```

```
R2 <- R2(predict7,test_data$Deaths_daily)
RMSE <- RMSE(predict7,test_data$Deaths_daily)
MAE <- MAE(predict7,test_data$Deaths_daily)
R2
```

```
##            [,1]
## [1,] 0.02739734
```

```
RMSE
```

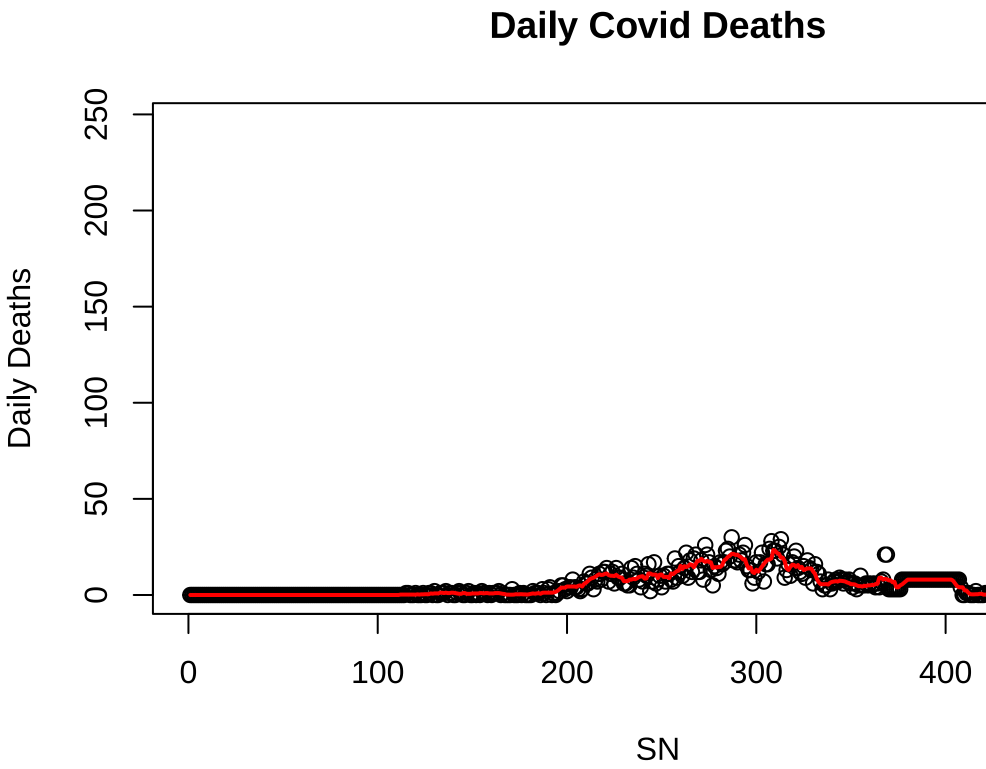
```
## [1] 37.80605
```

```
MAE
```

```
## [1] 13.32432
```

Selection of best model based on RMSE on test data Comparing the RMSE of all the models above , we see that RMSE of the KNN model is the least. So, KNN model is our best model.

```
plot(covid_data$SN, covid_data$Deaths_daily,
main = "Daily Covid Deaths",
xlab = "SN",
ylab = "Daily Deaths")
lines(predict(knnmodel,newdata = covid_data), col = "red", lwd=2)
```



Summary and Recommendation

From the prediction from our best model and the data in hand it is safe to say that there will be a rise in the cases for a certain period. What could be advised on this is that the ministry of health provide vaccination and maintain awareness among the people. Since more deaths would mean more infected people and more carriers(although they might not show symptoms or be medically diagnosed) the government should find a way to impose lockdown and minimal crowd gatherings keeping in mind a way to fulfill the basic requirements of the people for the time being.