Stock Market Prediction Project - Advanced Dataset

```
library(ISLR)
library(caTools)
library(MASS)
library(class)
library(forecast)
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.fracdiff    fracdiff
##   residuals.fracdiff fracdiff
```

```
library(ggplot2)
library(TTR)
library(fpp2)
```

```
## Loading required package: fma
```

```
##
## Attaching package: 'fma'
```

```
## The following objects are masked from 'package:MASS':
##
##     cement, housing, petrol
```

```
## Loading required package: expsmooth
```

```
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
smp=read.csv("/Users/bonnie/Desktop/Smarketadvanced.csv", header=T, sep=',',
         na.strings="NA")
attach(smp)
dim(smp)
```

```
## [1] 801  11
```

Our goal is to create insights and draw predictions for the percentage returns from the advanced stock market dataset over 801 months (66 years and 9 months) in the past, from January 1952 to September 2018.

For each month, we have recorded:

1. Direction: whether the market was Up or Down of this month
2. MktReturn: market percentage return of the month
3. X12m_cpi_forecast: CPI Forecast on a semi-annual basis
4. X12m_real_gdp_forecast_calculated: calculated real GDP forecast on a semi-annual basis
5. X12m_unemployment_rate: unemployment rate on a semi-annual basis
6. wti_prev_yoy_chg: WTI previous year-over-year change
7. PE_ratio_shiller: Shiller PE ratio for the S&P 500
8. corp_bond_yield_prev_month: corporate bond yield in previous month
9. MktReturn_prev_month: market return of the previous month from the date
10. MktReturn_prev_yr: market return of the previous year from the date

Slide Notes on Financial Terminology:

- CPI (Customer Price Index): a measure that examines the weighted average of prices of a basket of consumer goods and services, such as transportation, food, and medical care)
- Shiller PE ratio: a market valuation indicator, usually for S&P 500
- WTI: Crude Oil Price

```r
summary(smp)
```

```
##       Date        Direction    MktReturn        X12m_cpi_forecast
##  Min.   :195201   Down:296   Min.   :-22.6400   Min.   :-1.631   
##  1st Qu.:196809   Up  :505   1st Qu.: -1.5800   1st Qu.: 2.040   
##  Median :198505              Median :  1.2500   Median : 2.874   
##  Mean   :198494              Mean   :  0.9599   Mean   : 3.491   
##  3rd Qu.:200201              3rd Qu.:  3.6700   3rd Qu.: 4.627   
##  Max.   :201809              Max.   : 16.6100   Max.   :12.013   
##  X12m_real_gdp_forecast_calculated X12m_unemployment_rate
##  Min.   :-2.581                    Min.   :2.50          
##  1st Qu.: 2.719                    1st Qu.:4.90          
##  Median : 3.415                    Median :5.50          
##  Mean   : 3.685                    Mean   :5.82          
##  3rd Qu.: 4.581                    3rd Qu.:6.70          
##  Max.   :10.045                    Max.   :9.90          
##  wti_prev_yoy_chg  PE_ratio_shiller corp_bond_yield_prev_month
##  Min.   :-58.930   Min.   : 6.64    Min.   : 2.850            
##  1st Qu.: -2.733   1st Qu.:13.98    1st Qu.: 4.380            
##  Median :  0.000   Median :19.23    Median : 6.400            
##  Mean   :  9.240   Mean   :19.60    Mean   : 6.683            
```

```
##  3rd Qu.: 13.180   3rd Qu.:23.71    3rd Qu.: 8.360
##  Max.   :183.989   Max.   :44.20    Max.   :15.490
##  MktReturn_prev_month MktReturn_prev_yr
##  Min.   :-21.7630     Min.   :-44.756
##  1st Qu.: -1.7528     1st Qu.: -1.306
##  Median :  0.9171     Median : 10.016
##  Mean   :  0.6917     Mean   :  8.684
##  3rd Qu.:  3.4444     3rd Qu.: 18.732
##  Max.   : 16.3047     Max.   : 52.942
```

Next, using a naive strategy, we can produce a matrix that contains all of the correlations between market return and all other predictors in the data set.

```
cor.mat=cor(smp[, -2])
cor.mat[,1:2]
```

```
##                                          Date    MktReturn
## Date                              1.000000000 -0.008478807
## MktReturn                        -0.008478807  1.000000000
## X12m_cpi_forecast                 0.093775580 -0.005337032
## X12m_real_gdp_forecast_calculated -0.066421123 -0.078912319
## X12m_unemployment_rate            0.267709874  0.096385376
## wti_prev_yoy_chg                  0.034646205 -0.068277457
## PE_ratio_shiller                  0.511903766 -0.068933919
## corp_bond_yield_prev_month        0.058436379 -0.001101784
## MktReturn_prev_month              0.002855145  0.057877608
## MktReturn_prev_yr                 0.001809251  0.014349075
```

As we can see, the correlations between all the predictor variables and market montly returns are close to zero. Only the variables X12m_real_gdp_forecast_calculated, X12m_unemployment_rate, wti_prev_yoy_chg, PE_ratio_shiller, MktReturn_prev_month have correlations that is relatively larger and close to 0.1.

Logistic Regression

Next, we want to predict Direction by fitting a logistic regression model using the predictor variables that have a relatively high correlation with the market return.

```
glm_fit=glm(Direction~X12m_real_gdp_forecast_calculated+X12m_unemployment_rate+wti_prev_yoy_chg+PE_ratic
summary(glm_fit)
```

```
##
## Call:
## glm(formula = Direction ~ X12m_real_gdp_forecast_calculated +
##     X12m_unemployment_rate + wti_prev_yoy_chg + PE_ratio_shiller +
##     MktReturn_prev_month, family = binomial, data = smp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8018  -1.3306   0.8631   0.9690   1.4299
##
## Coefficients:
##                                    Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)                        0.181669   0.476497   0.381  0.70301
## X12m_real_gdp_forecast_calculated -0.097776   0.037677  -2.595  0.00946 **
## X12m_unemployment_rate             0.097833   0.056455   1.733  0.08311 .
## wti_prev_yoy_chg                  -0.003349   0.002153  -1.556  0.11979
## PE_ratio_shiller                   0.008417   0.010767   0.782  0.43437
## MktReturn_prev_month               0.029032   0.018159   1.599  0.10986
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1055.3  on 800  degrees of freedom
## Residual deviance: 1039.7  on 795  degrees of freedom
## AIC: 1051.7
##
## Number of Fisher Scoring iterations: 4
```

Assume the threshold for small p-value is 0.05. Then, only the p-value of X12m_real_gdp_forecast_calculated suggests that there is a strong evidence showing there are some relationship between Direction and X12m_real_gdp_forecast_calculated.

Then, we can improve our model by reducing insignificant variables.

```r
glm_fit=glm(Direction~ X12m_real_gdp_forecast_calculated, data=smp, family=binomial)
summary(glm_fit)
```

```
##
## Call:
## glm(formula = Direction ~ X12m_real_gdp_forecast_calculated,
##     family = binomial, data = smp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6339  -1.3844   0.9189   0.9618   1.2077
##
## Coefficients:
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                        0.89212    0.16004   5.574 2.48e-08 ***
## X12m_real_gdp_forecast_calculated -0.09589    0.03763  -2.548   0.0108 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1055.3  on 800  degrees of freedom
## Residual deviance: 1048.7  on 799  degrees of freedom
## AIC: 1052.7
##
## Number of Fisher Scoring iterations: 4
```

```r
glm.probs=predict(glm_fit,type='response')
contrasts(Direction)
```

```
##      Up
```

```
## Down   0
## Up     1
```

```
glm.probs[1:10]
```

```
##         1         2         3         4         5         6         7
## 0.6158221 0.6158221 0.6158221 0.6158221 0.6158221 0.6564860 0.6564860
##         8         9        10
## 0.6564860 0.6564860 0.6564860
```

```
max(glm.probs)
```

```
## [1] 0.7576058
```

```
min(glm.probs)
```

```
## [1] 0.4822445
```

We then can get the predicted probability of the market going up from the logistic regression model. The range is between 48.22% and 75.76%.

Next, we want to convert the above predicted probability to class label in order to make a prediction for the market direction going up or down on a particular day. So, we create a vector of class predictions based on whether the predicted probability of a market increase is greater than or less than 0.5.

```
glm.pred=rep("Down",801)
glm.pred[glm.probs >.5]="Up"
table(glm.pred,Direction) #confusion matrix
```

```
##         Direction
## glm.pred Down  Up
##     Down    3   3
##     Up    293 502
```

```
mean(glm.pred==Direction )
```

```
## [1] 0.6304619
```

This logistic regression model correctly predicted the movement of the market 63.05 % of the time. However, the training error rate is 36.95%, which might be overly optimistic.

To better access the accuracy in this model, we split the data into a training set and a test set holding 30% of data for testing.

```
set.seed(101)
sample = sample.split(1:nrow(smp), SplitRatio=0.3)
test = subset(smp, sample==TRUE)
train = subset(smp, sample==FALSE)
direction.test=test$Direction
glm_fit=glm(Direction~X12m_real_gdp_forecast_calculated, data=train, family=binomial)
summary(glm_fit)
```

```
##
## Call:
## glm(formula = Direction ~ X12m_real_gdp_forecast_calculated,
##     family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6549  -1.3620   0.9313   0.9850   1.2919
##
## Coefficients:
##                                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)                         0.90874    0.19077   4.763  1.9e-06 ***
## X12m_real_gdp_forecast_calculated -0.11687    0.04522  -2.585  0.00974 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 746.84  on 560  degrees of freedom
## Residual deviance: 740.00  on 559  degrees of freedom
## AIC: 744
##
## Number of Fisher Scoring iterations: 4
```

```r
glm.probs=predict(glm_fit,test, type='response')
glm.pred=rep("Down",240)
glm.pred[glm.probs >.5]="Up"
table(glm.pred,direction.test)
```

```
##         direction.test
## glm.pred Down  Up
##     Down    1   7
##     Up     80 152
```

```r
mean(glm.pred==direction.test)
```

```
## [1] 0.6375
```

```r
mean(glm.pred!=direction.test)
```

```
## [1] 0.3625
```

```r
152/(152+79)
```

```
## [1] 0.6580087
```

In this improved model, we got a (slightly) better prediction rate of 63.75%, and we lowered the test set error rate to 36.25%.

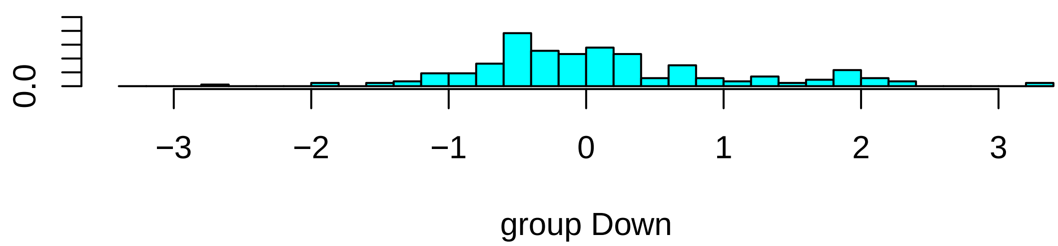
The confusion matrix also shows that on days when logistic regression predicts an increase in the market, it hasa 65.8% accuracy rate. This suggests a possible trading strategy of buying on days when the model predict an up market, and avoid trades on days when a down is predicted.

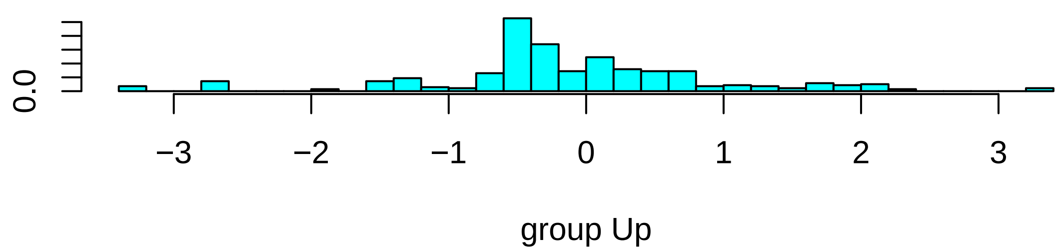
LDA

```
lda.fit=lda(Direction~X12m_real_gdp_forecast_calculated, data=train)
lda.fit
```

```
## Call:
## lda(Direction ~ X12m_real_gdp_forecast_calculated, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.3832442 0.6167558
##
## Group means:
##      X12m_real_gdp_forecast_calculated
## Down                          3.928224
## Up                            3.484046
##
## Coefficients of linear discriminants:
##                                         LD1
## X12m_real_gdp_forecast_calculated 0.5118909
```

```
plot(lda.fit) #Plot the linear discriminants for each training observation
```





The LDA output indicates that 38.3% of the training observations correspond to the months during which the market went down.

LDA decision rule: $\delta_k(x) = x^T \sum^{-1} \mu_k - \frac{1}{2}\mu_k^T \sum^{-1} \mu_k + log(\pi_k)$

If 0.512 * X12m_real_gdp_forecast_calculated is large, then the LDA classifier will predict a market increase; and if it's small, then the LDA classifer will predict a market decline.

```
lda.pred=predict(lda.fit, test)
lda.class=lda.pred$class
table(lda.class, direction.test)
```

```
##          direction.test
## lda.class Down  Up
##      Down    1   7
##      Up     80 152
```

```
mean(lda.class==direction.test)
```

```
## [1] 0.6375
```

The LDA and logistic regression predictions are almost the same.

Then applying a 50% threshold to the posterior probabilities to recreate the predictions contained in lda.class.

```
lda.pos=lda.pred$posterior
sum(lda.pos[,1]>=.5)
```

```
## [1] 8
```

```
sum(lda.pos[,1]<.5)
```

```
## [1] 232
```

```
lda.pos[1:20,1] #posterior probability of a decreasing market
```

```
##         2         5        18        20        23        27        33
## 0.4020273 0.4020273 0.2550221 0.2550221 0.2550221 0.2301368 0.3046113
##        34        37        39        46        47        50        51
## 0.3046113 0.3656372 0.3656372 0.3421474 0.3421474 0.3394139 0.3394139
##        67        71        76        77        78        85
## 0.3270697 0.3270697 0.2894187 0.2894187 0.3553210 0.4779433
```

```
lda.class[1:20]
```

```
##  [1] Up Up Up Up Up Up Up Up Up Up Up Up Up Up Up Up Up Up Up Up
## Levels: Down Up
```

```
sum(lda.pos[,1]>=.5651)
```

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

The greatest posterior probability of decrease in the test data was 56.51%. ???

QDA

```
qda.fit=qda(Direction~X12m_real_gdp_forecast_calculated, data=train)
qda.fit
```

```
## Call:
## qda(Direction ~ X12m_real_gdp_forecast_calculated, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.3832442 0.6167558
##
## Group means:
##      X12m_real_gdp_forecast_calculated
## Down                          3.928224
## Up                            3.484046
```

```
qda.class=predict(qda.fit,test)$class
table(qda.class, direction.test)
```

```
##          direction.test
## qda.class Down  Up
##      Down    0   0
##      Up     81 159
```

```
mean(qda.class==direction.test)
```

```
## [1] 0.6625
```

The QDA predictions are accurate 66.25% of the time, which has been improved a lot from all the other models. This suggests that the quadratic form assumed by QDA may be more accurate than the linear form assumed by LDA and logistic regression.

KNN

```
train.gdp=cbind(train$X12m_real_gdp_forecast_calculated)
test.gdp=cbind(test$X12m_real_gdp_forecast_calculated)
train.dir=train$Direction
```

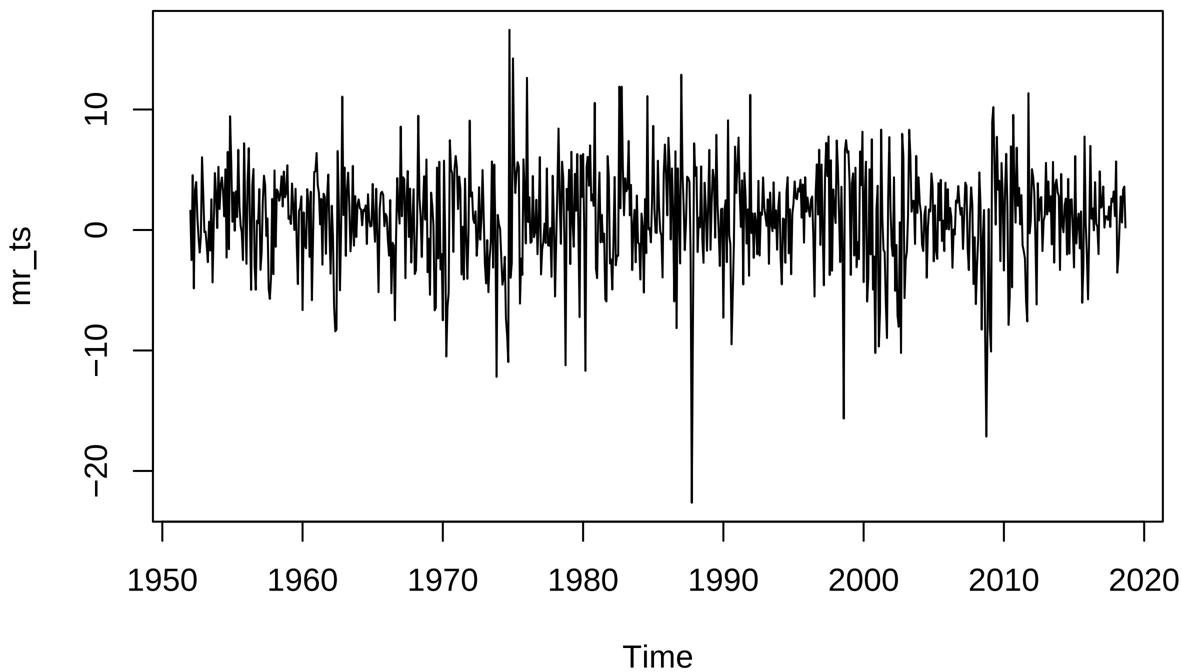
```
test.error.rate <- c()
k_value=c(1,5,10,15,20,30,50,100,150,200)
for (i in k_value){
knn_model=knn(train=train.gdp, test=test.gdp, cl= train.dir, k=i)
test.error.rate=c(test.error.rate, mean(knn_model != direction.test))
}
test.error.rate
```

```
##  [1] 0.4333333 0.4375000 0.3666667 0.3458333 0.3666667 0.3583333 0.3375000
##  [8] 0.3375000 0.3375000 0.3375000
```

KNN performs best around K=5, which gives a 42.08% accuracy rate.

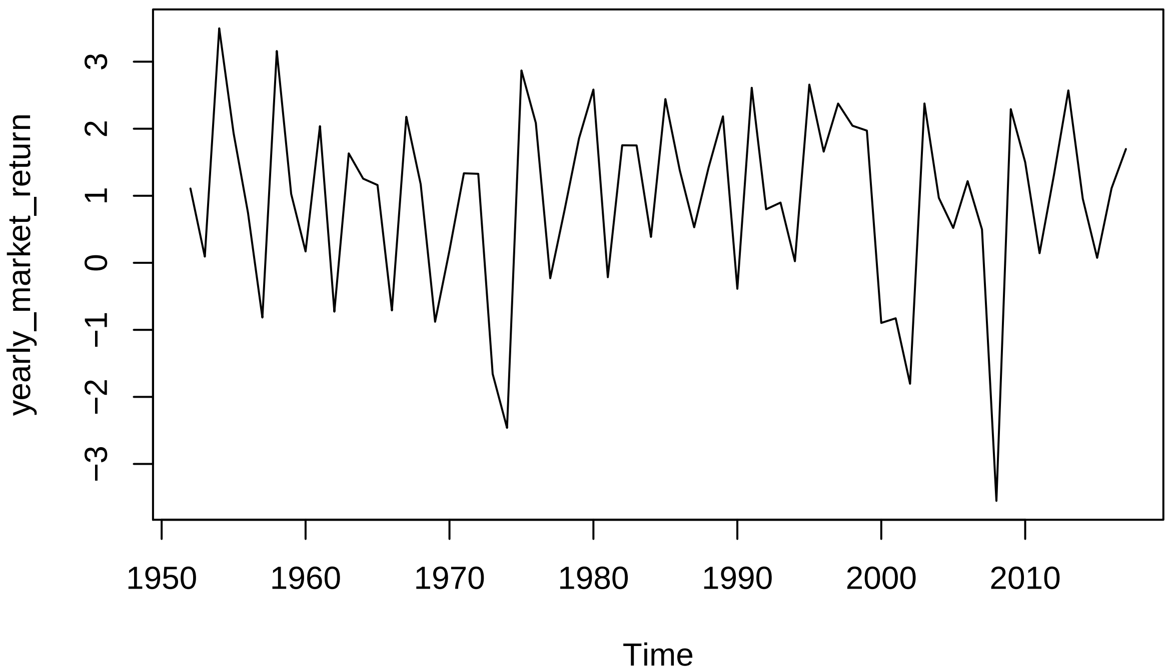
Time Series

```
market_return=smp[,3]
mr_ts=ts(market_return,frequency=12, start=c(1952,1))
plot.ts(mr_ts)
```



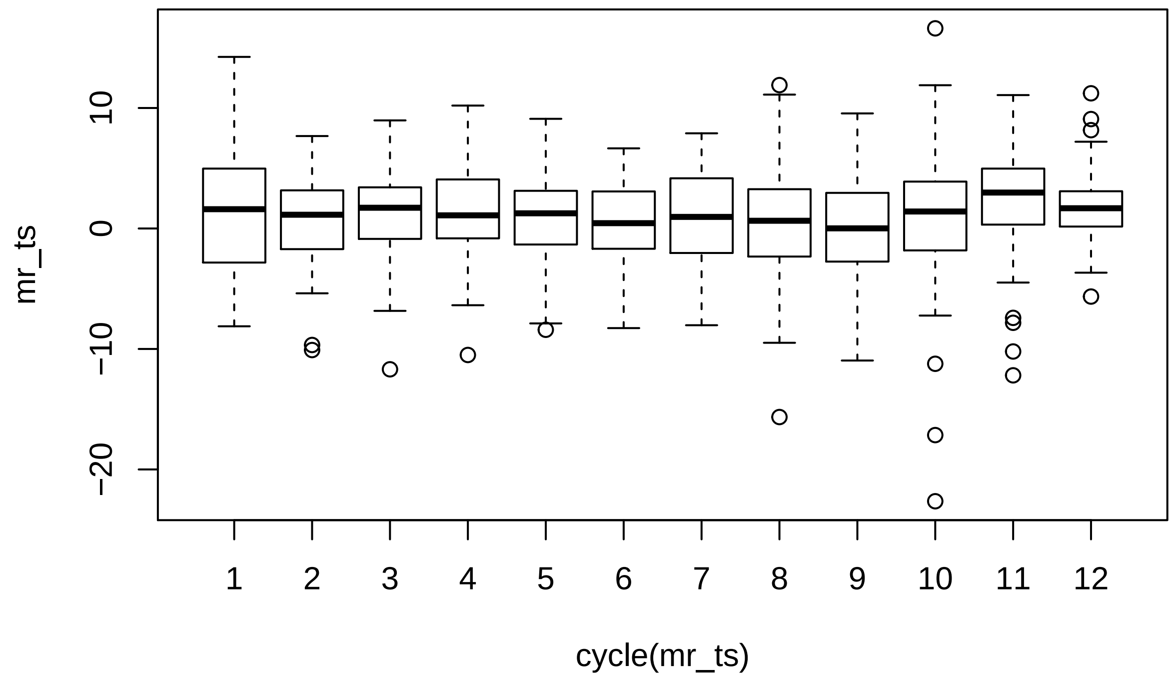
Varing spread - the time series does not have constant variance. Fail stationary criterion.

```
plot(aggregate(mr_ts,FUN=mean), ylab="yearly_market_return")
```




```
# a year on year trend
```

```
bp=boxplot(mr_ts~cycle(mr_ts)) #seasonal effect
```



Inferences: 1. The year on year trend shows that there is a periodical change in the stock market index. 2. The mean value in November and December is higher than the rest of the months. The variance in October is much higher than the rest. 3. The mean value in each month is quite similar and their variance is small. Hence, we have a seasonal effect with a cycle of 12 months or less.

```
ts_lag1=lag(mr_ts, lag=1)
head(cbind(mr_ts, ts_lag1))
```

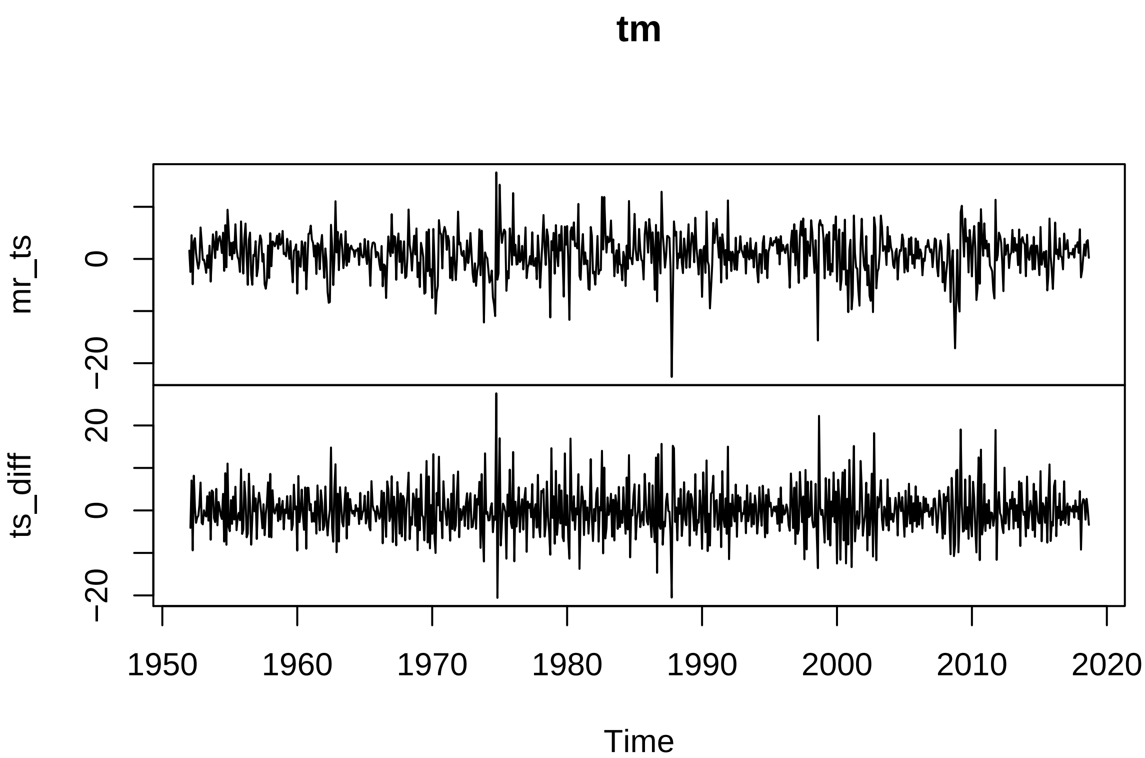
```
##          mr_ts ts_lag1
## Dec 1951    NA    1.60
## Jan 1952  1.60   -2.50
## Feb 1952 -2.50    4.55
## Mar 1952  4.55   -4.85
## Apr 1952 -4.85    3.33
## May 1952  3.33    3.98
```

```
ts_diff=diff(mr_ts, lag=1)
tm=cbind(mr_ts, ts_diff)
head(tm)
```

```
##          mr_ts ts_diff
## Jan 1952  1.60      NA
## Feb 1952 -2.50   -4.10
## Mar 1952  4.55    7.05
## Apr 1952 -4.85   -9.40
## May 1952  3.33    8.18
```

```
## Jun 1952  3.98    0.65
```

```
plot.ts(tm)
```




```
library(tseries)
adf.test(ts_diff)
```

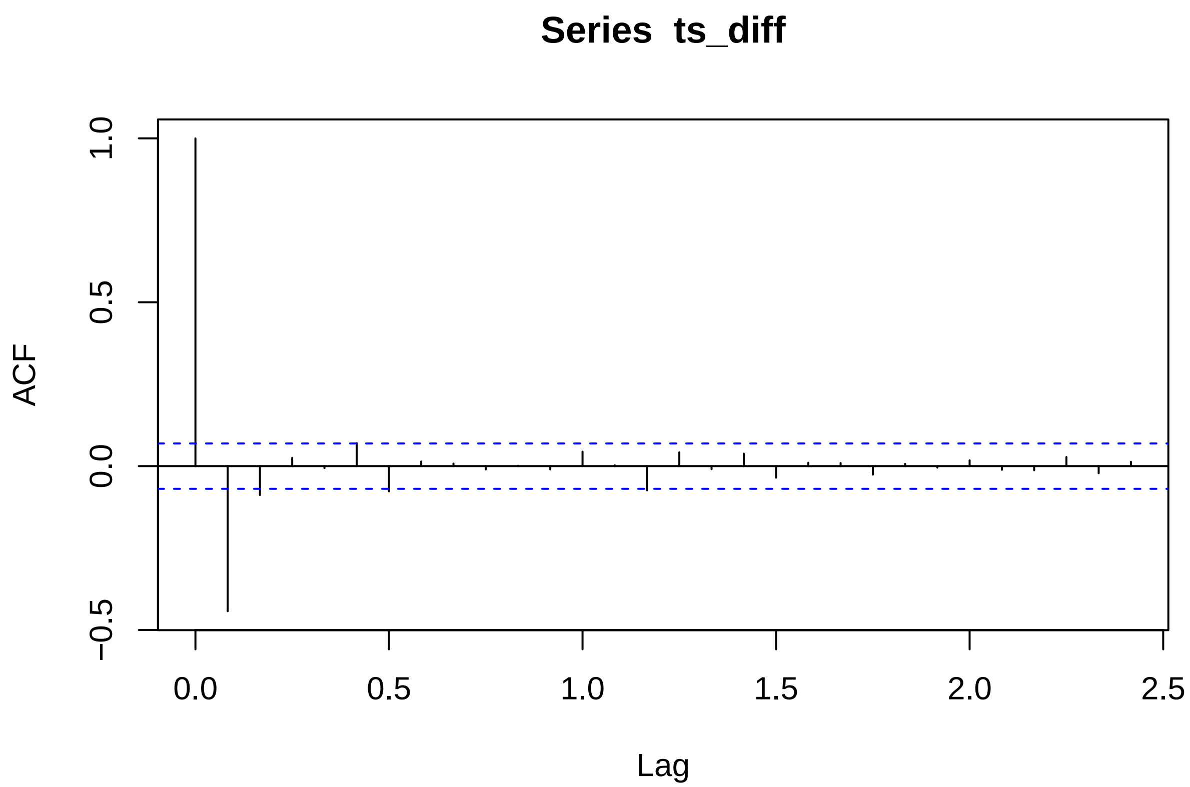
```
## Warning in adf.test(ts_diff): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_diff
## Dickey-Fuller = -15.192, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

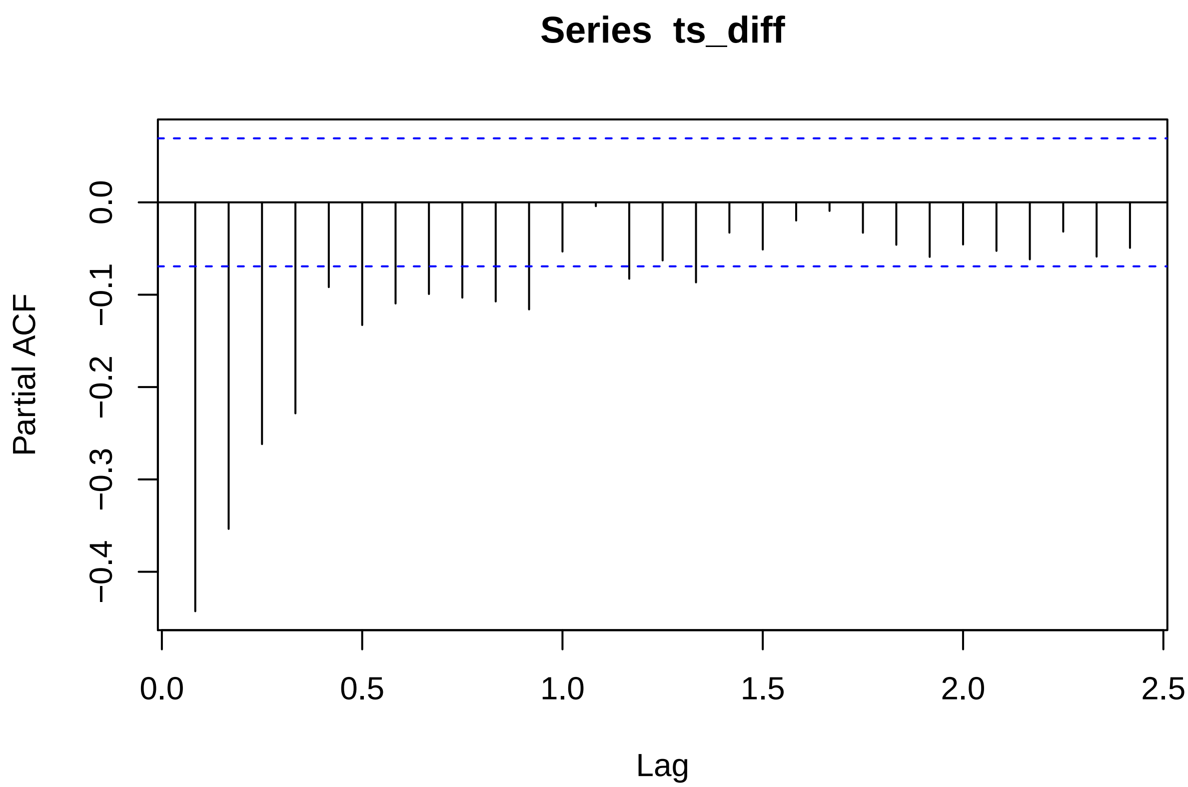
By taking the difference, we make the non-stationary series stationary.

Plot ACF/PACF charts

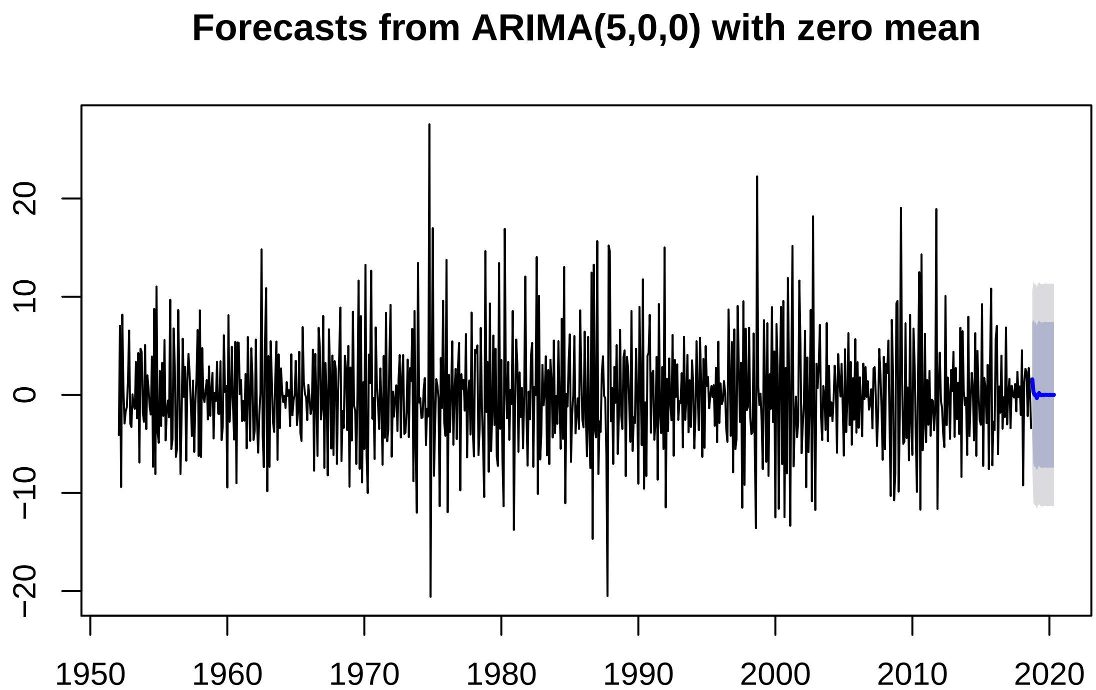
```
acf(ts_diff)
```



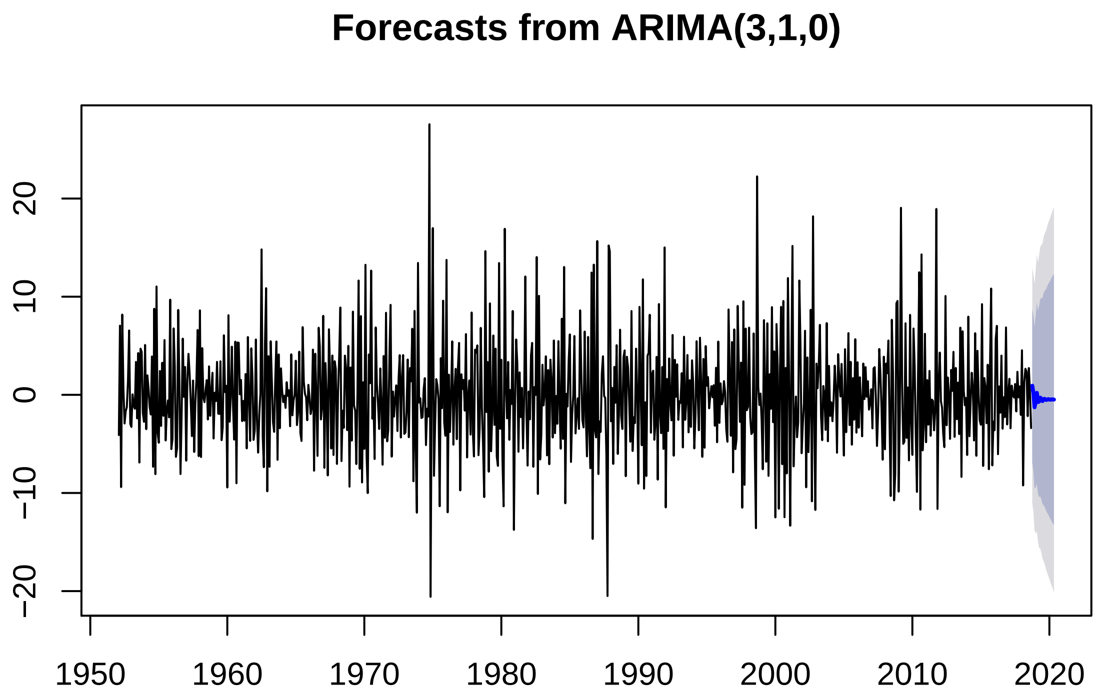

```r
pacf(ts_diff)
```



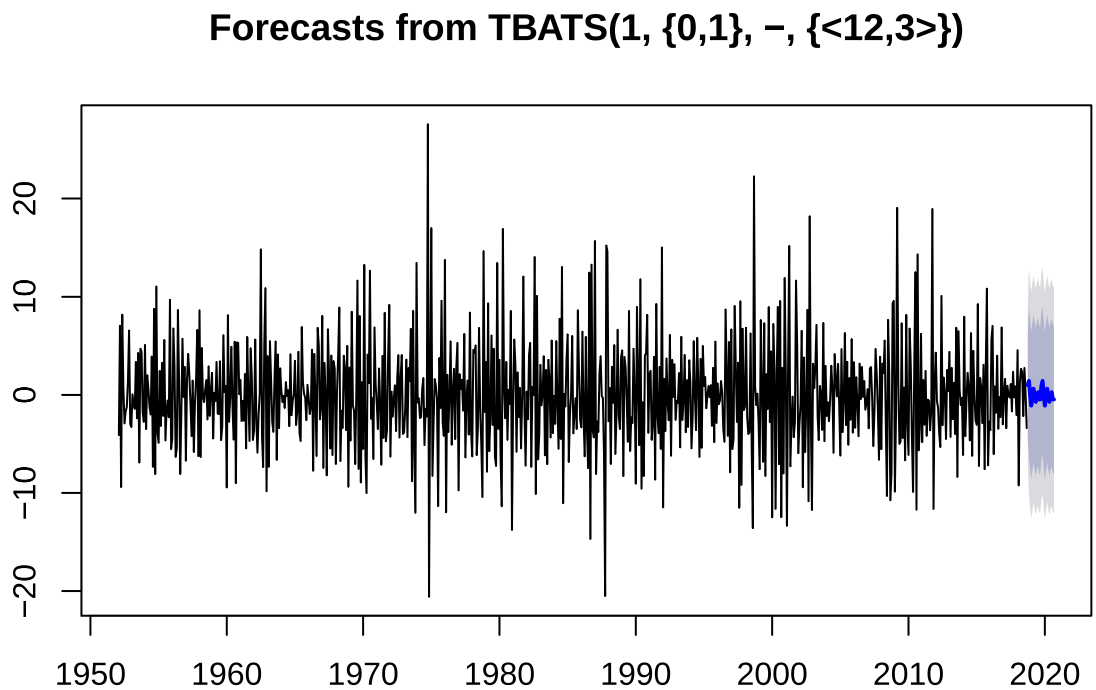

```
autoArimaFit=auto.arima(ts_diff)
plot(forecast(autoArimaFit, h=20))
```



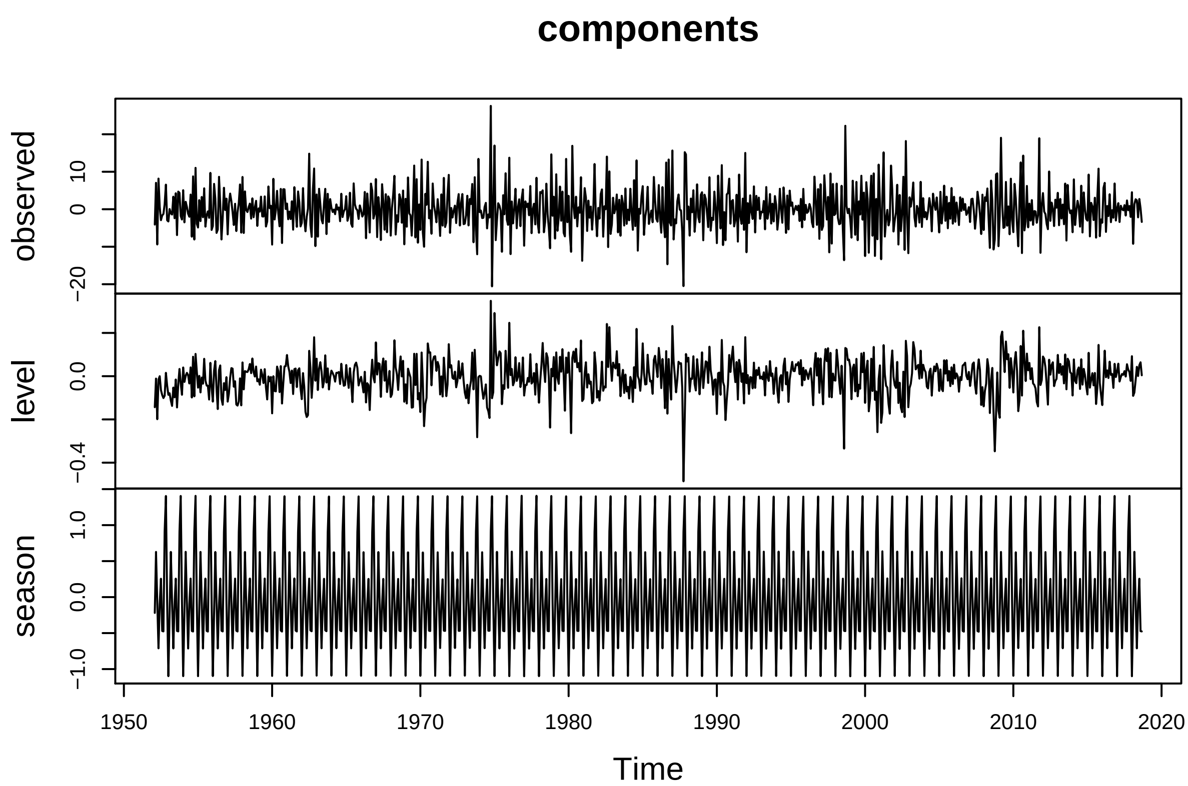

```
arimaFit=Arima(ts_diff,order=c(3,1,0))
plot(forecast(arimaFit,h=20))
```




```
tbatsFit=tbats(ts_diff, use.parallel=T, num.cores=2) # fit tbats model
plot(forecast(tbatsFit)) # plot
```




```
components <- tbats.components(tbatsFit)
plot(components)
```



Confidence Interval for my Forecasts

```
model=HoltWinters(ts_diff)
predict(model, 50, prediction.interval=T, level= 0.99)
```

```
##                 fit      upr       lwr
## Oct 2018  1.42185950 16.70120 -13.85749
## Nov 2018  0.63849609 15.91788 -14.64089
## Dec 2018 -0.25370993 15.02575 -15.53317
## Jan 2019 -1.19533545 14.08426 -16.47493
## Feb 2019  0.03652239 15.31632 -15.24327
## Mar 2019  0.61755164 15.89763 -14.66253
## Apr 2019 -0.05322429 15.22724 -15.33368
## May 2019 -0.55686035 14.72410 -15.83782
## Jun 2019 -0.95425579 14.32732 -16.23583
## Jul 2019  1.04377063 16.32611 -14.23857
## Aug 2019 -1.01717717 14.26609 -16.30044
## Sep 2019 -0.33278855 14.95157 -15.61714
## Oct 2019  1.42219950 16.73742 -13.89302
## Nov 2019  0.63883609 15.95553 -14.67786
## Dec 2019 -0.25336994 15.06501 -15.57175
## Jan 2020 -1.19499545 14.12530 -16.51529
## Feb 2020  0.03686239 15.35931 -15.28559
## Mar 2020  0.61789164 15.94276 -14.70697
## Apr 2020 -0.05288430 15.27466 -15.38043
## May 2020 -0.55652036 14.77400 -15.88704
```

```
## Jun 2020 -0.95391580 14.37987 -16.28770
## Jul 2020  1.04411062 16.38148 -14.29326
## Aug 2020 -1.01683718 14.32444 -16.35811
## Sep 2020 -0.33244856 15.01308 -15.67798
## Oct 2020  1.42253949 16.81114 -13.96606
## Nov 2020  0.63917609 16.03274 -14.75439
## Dec 2020 -0.25302994 15.14589 -15.65195
## Jan 2021 -1.19465546 14.21001 -16.59932
## Feb 2021  0.03720238 15.44803 -15.37362
## Mar 2021  0.61823163 16.03564 -14.79918
## Apr 2021 -0.05254430 15.37189 -15.47698
## May 2021 -0.55618036 14.87573 -15.98809
## Jun 2021 -0.95357580 14.48628 -16.39343
## Jul 2021  1.04445062 16.49273 -14.40383
## Aug 2021 -1.01649718 14.44070 -16.47369
## Sep 2021 -0.33210856 15.13451 -15.79873
## Oct 2021  1.42287949 16.94651 -14.10075
## Nov 2021  0.63951608 16.17359 -14.89456
## Dec 2021 -0.25268995 15.29238 -15.79776
## Jan 2022 -1.19431546 14.36231 -16.75094
## Feb 2022  0.03754238 15.60629 -15.53120
## Mar 2022  0.61857163 16.20003 -14.96288
## Apr 2022 -0.05220431 15.54256 -15.64696
## May 2022 -0.55584037 15.05283 -16.16452
## Jun 2022 -0.95323581 14.66998 -16.57645
## Jul 2022  1.04479062 16.68318 -14.59360
## Aug 2022 -1.01615718 14.63805 -16.67036
## Sep 2022 -0.33176857 15.33892 -16.00246
## Oct 2022  1.42321948 17.16628 -14.31984
## Nov 2022  0.63985608 16.40069 -15.12098
```