Untitled

Kerem Turgutlu

November 26, 2017

```
library(tidyverse)
library(forecast)
library(lawstat)
library(tseries)
```

In this part of the project we will explore arima, sarima and sarimax models in order to predict bankruptcy rate. Arima or in general sarima models are univariate models which depend on previous time series data and try to predict future values. Through this part we will check these models with their assumptions. Sarimax on the other hand is a multivariate time series model which has the sarima component but also regresses on the given multivariate data at time t.

```
train <- read.csv('../../../train.csv')[1:288,]
test <- read.csv('../../../test.csv')
```

```
train %>% glimpse()
```

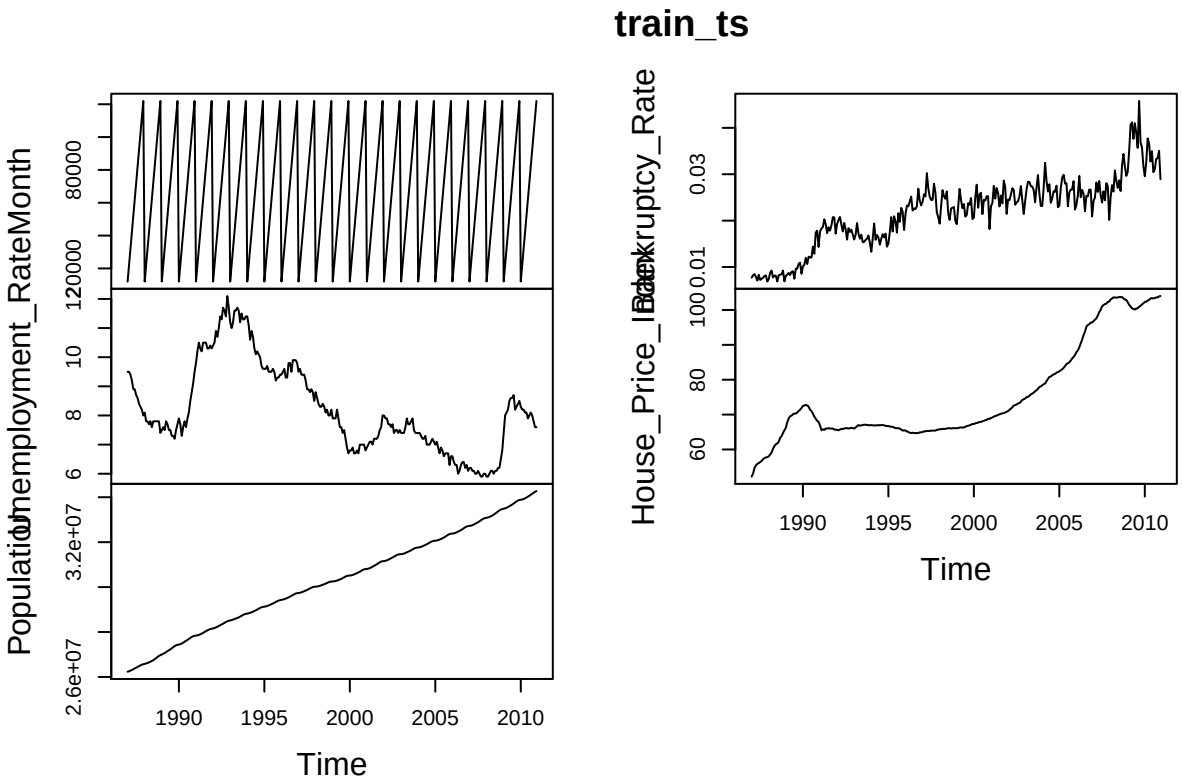
```
## Observations: 288
## Variables: 5
## $ Month             <int> 11987, 21987, 31987, 41987, 51987, 61987, 71...
## $ Unemployment_Rate <dbl> 9.5, 9.5, 9.4, 9.2, 8.9, 8.9, 8.7, 8.6, 8.4,...
## $ Population        <int> 26232423, 26254410, 26281420, 26313260, 2634...
## $ Bankruptcy_Rate   <dbl> 0.0077004, 0.0082196, 0.0084851, 0.0078326, ...
## $ House_Price_Index <dbl> 52.2, 53.1, 54.7, 55.4, 55.9, 56.1, 56.4, 56...
```

We separated train and test data as; time series until 2011 and time series after 2011.

```
train_ts <- ts(train,start = 1987, frequency = 12)
test_ts <- ts(test, start = 2011, frequency = 12)
```

Here we can see time series that are provided in training data. House index and population seems to have a positive correlation with bankruptcy where as unemployment has a negative one.

```
plot(train_ts)
```



There is great correlation between House_Price_Index and Bankruptcy_Rate, probably even a higher one with lagged values of House_Price_Index. So, here we plot the change in correlation between bankruptcy and housing index by different lagged values of housing index. Basically what we do here is to shift housing index to the right.

Correlation matrix of multivariate data provided.

```
cor(train)
```

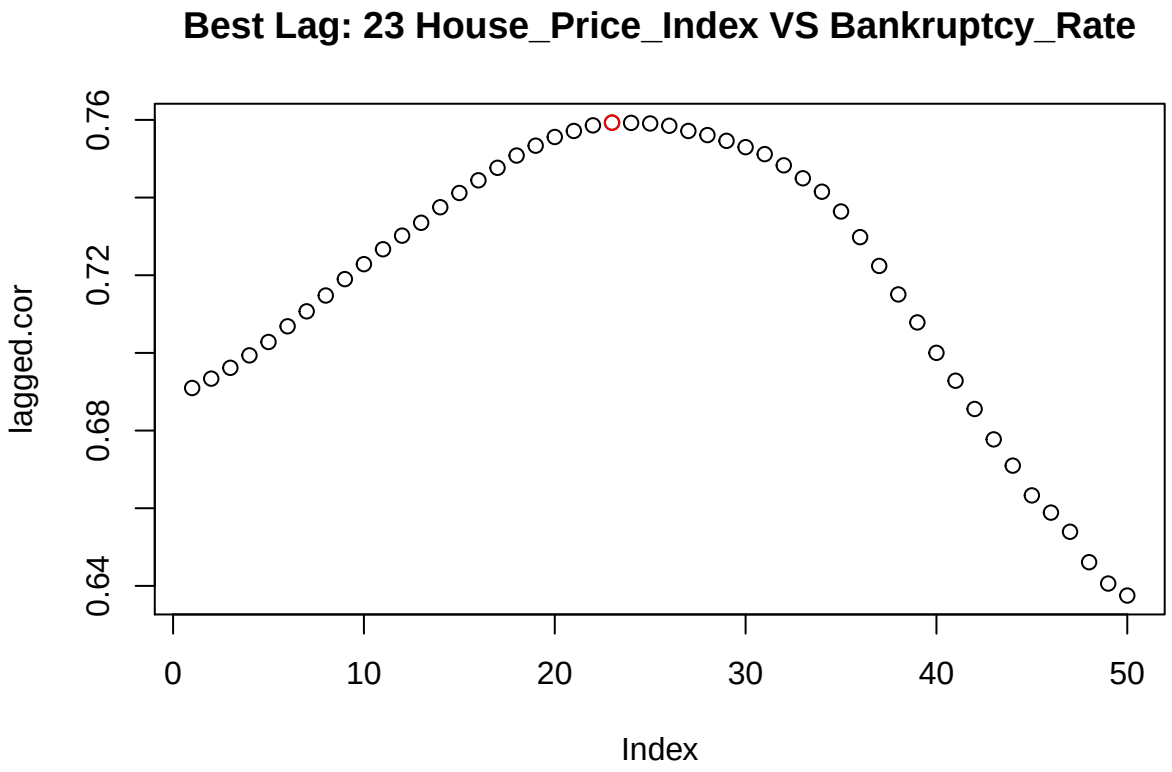
```
##                       Month Unemployment_Rate Population Bankruptcy_Rate
## Month              1.00000000       -0.02322856  0.0501926     -0.00459977
## Unemployment_Rate -0.02322856        1.00000000 -0.5431182     -0.31690705
## Population         0.05019260       -0.54311821  1.0000000      0.89840496
## Bankruptcy_Rate   -0.00459977       -0.31690705  0.8984050      1.00000000
## House_Price_Index  0.04548785       -0.54305931  0.8601513      0.68970802
##                   House_Price_Index
## Month                    0.04548785
## Unemployment_Rate       -0.54305931
## Population               0.86015125
## Bankruptcy_Rate          0.68970802
## House_Price_Index        1.00000000
```

```
lagged.cor <- c()

h = 50
for (i in (seq(h))){
  lagged_house <- lag(train$House_Price_Index, n = i)
  cor.i <- cor(lagged_house, train$Bankruptcy_Rate, use = 'complete.obs')
  lagged.cor <- c(lagged.cor, cor.i)
}
```

Lagged Correlation Plot h vs Correlation

We observe that highest correlation between housing index and bankruptcy happens to be with 23 lagged version of housing index. Meaning that there might be a pattern that bankruptcy follows from housing index after 23 months of occurence. Of course this is just a hypothetical assumptions which needs to be tried out. Hence, we can try out different lagegd versions of housing index and use it in our sarimax model. Another assumptions we are making with sarimax is that any regressed variable during modeling is an exogenous variable, meaning that they have a uni-directional effect on dependent variable; bankruptcy rate but not the other way around.

```
best.idx <- which.max(lagged.cor)
plot(lagged.cor)
points(best.idx, lagged.cor[best.idx], col='red')
title(paste('Best Lag:' , best.idx,'House_Price_Index VS Bankruptcy_Rate'))
```



SARIMA MODEL (Univariate Bankruptcy)

Let's start our sarima model.

```
bankruptcy_ts <- ts(train$Bankruptcy_Rate, frequency = 12)
```

We have a total 24 years fo data in our training sample.

```
length(bankruptcy_ts) /12
```

```
## [1] 24
```

Split train - valid (Last 2 Years as Valid)

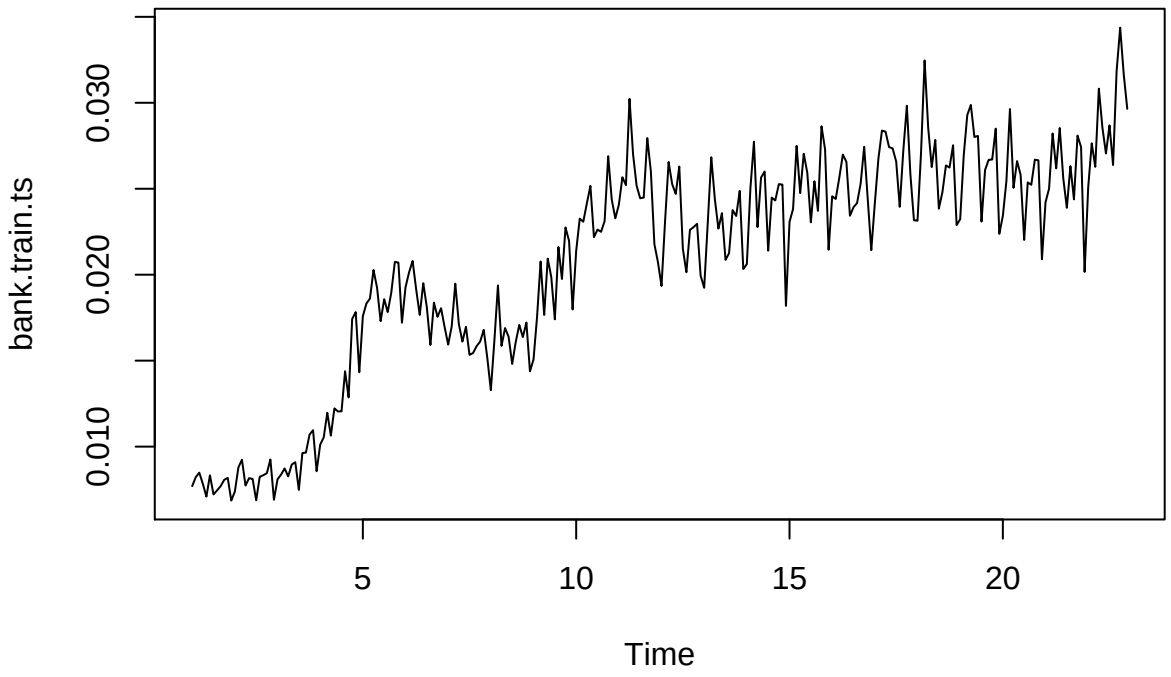
Every predictive modeling task has a evaluation metric in order to assess the performance different type of models and in order to pick the best available model that we hope to generalize to our hypothesis. Also during these predictive modeling tasks we create a hold-out set which is also out validation set. We will use last 2 years of our data in order to assess our models with evaluation metric as RMSE. Another important point which shouldn't be forgotten is that one should be careful about not overfitting to the validation set. So here we will also care about less complex models which will helps us avoid overfitting and as well as models that give good performance in validation set meaning that they generalize good enough to reflect the pattern on unseen data.

```
bank.train.ts <- ts(bankruptcy_ts[1:264], frequency = 12)
bank.valid.ts <- ts(bankruptcy_ts[265:288], frequency = 12)
```

Plot Training

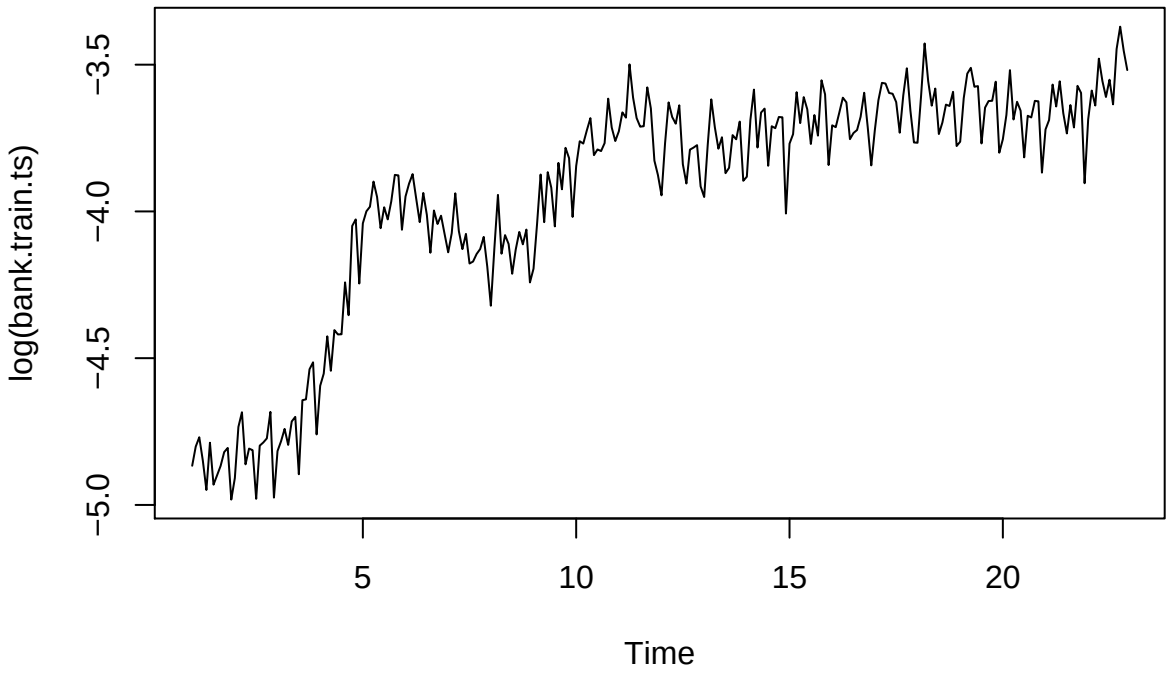
Here we observe a change in scale as we move forward in our time series, and this might be a problem during checking the constant variance of residuals, which is an important assumption when we fit our model with MLE. We generally make a transformation and look at the plot again to see if this change doesn't occur anymore; some common transformations are log, square root or in general boxcox transforms.

```
plot(bank.train.ts)
```



So we apply a log transform to our data to see if it becomes better in terms of constant variance over time. The change in variance is not as bad as before, so we will proceed our analysis with transformed version of our time series model.

```
bank.train.ts.log <- log(bank.train.ts)
bank.valid.ts.log <- log(bank.valid.ts)
plot(log(bank.train.ts))
```



During time series modeling another important matter that one should to pay attention is stationarity of data. This is important since ARMA models account for only stationary data, so we will try to decompose our time series first, the decide our parameters p, P, q, Q in order feed our data into a SARIMA model. There other types of models such as exponential smoothing models which takes care seasonality and trend with the given parameters alpha, beta and gamma. But these models are subsets of general sarima models. So it's always better to pay good attention to sarima models since they can be more powerful in terms of capability of capturing many different combinations of patterns.

We are using ndiffs on our log transformed data in order to decide the number of trend differencing we need. One can also apply and ADF test in order to check stationarity of their data before and after applying differences. But readily available functions has these properties in themselves so they come in handy. As seen below solely trusting in these function can also be a naive mistake, for example below nsdiffs suggests we don't require a seasonal differencing but in fact we might need to do it.

```
ndiffs(bank.train.ts.log)
```

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

```
bank.train.ts.log.D10 <- diff(bank.train.ts.log)
ndiffs(bank.train.ts.log.D10)
```

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

```
nsdiffs(bank.train.ts.log.D10)
```

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

We are going check ADF test after 1 trend differencing. It's good practice to take lag as "m*2 - m*4" when conducting these tests for stationary since one can misguided with small lagged tests. ADF test suggest that our time series is not stationary so we will need to apply more differencing and check the test again. In fact after applying another trend differencing we are now confident with 99% confidence level that are time series indeed stationary.

```
adf.test(bank.train.ts.log.D10, k = 48)
```

```
##
```

```
##  Augmented Dickey-Fuller Test
##
## data:  bank.train.ts.log.D10
## Dickey-Fuller = -3.2651, Lag order = 48, p-value = 0.07745
## alternative hypothesis: stationary
```

```
adf.test(diff(bank.train.ts.log.D10), k = 48)
```

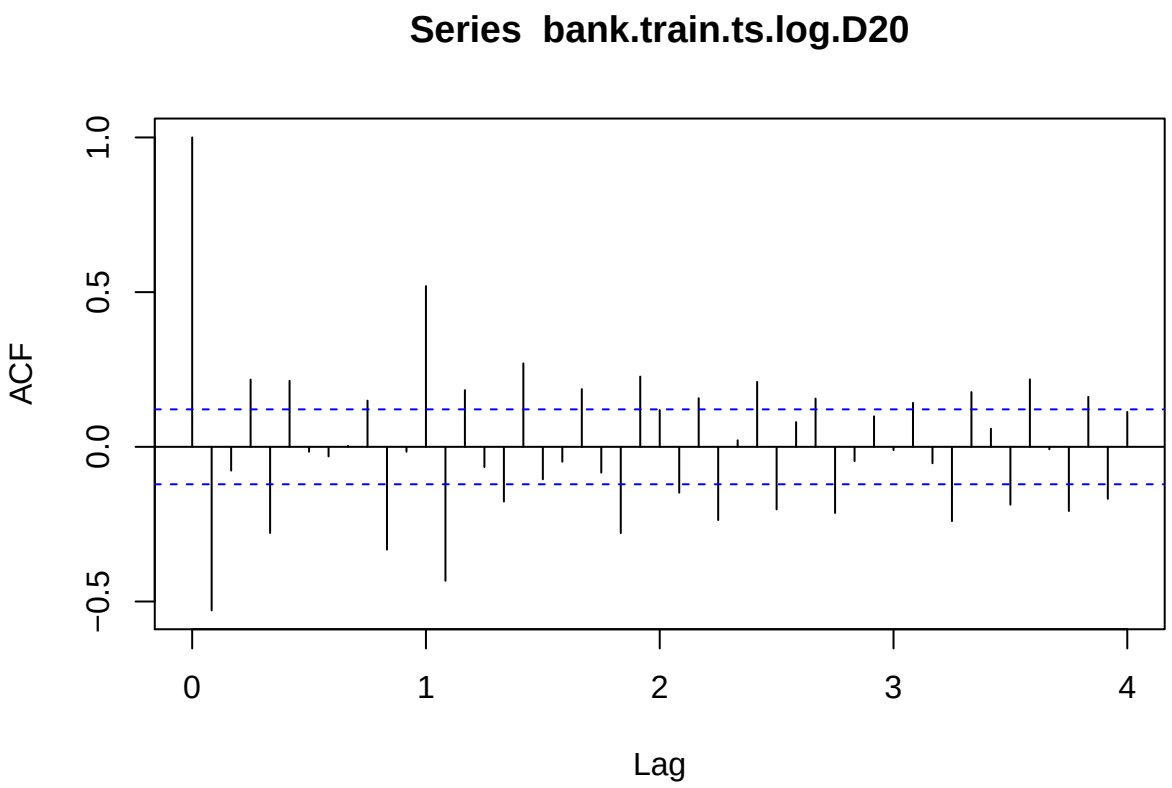
```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(bank.train.ts.log.D10)
## Dickey-Fuller = -4.5199, Lag order = 48, p-value = 0.01
## alternative hypothesis: stationary
```

```
bank.train.ts.log.D20 <-  diff(bank.train.ts.log.D10)
```

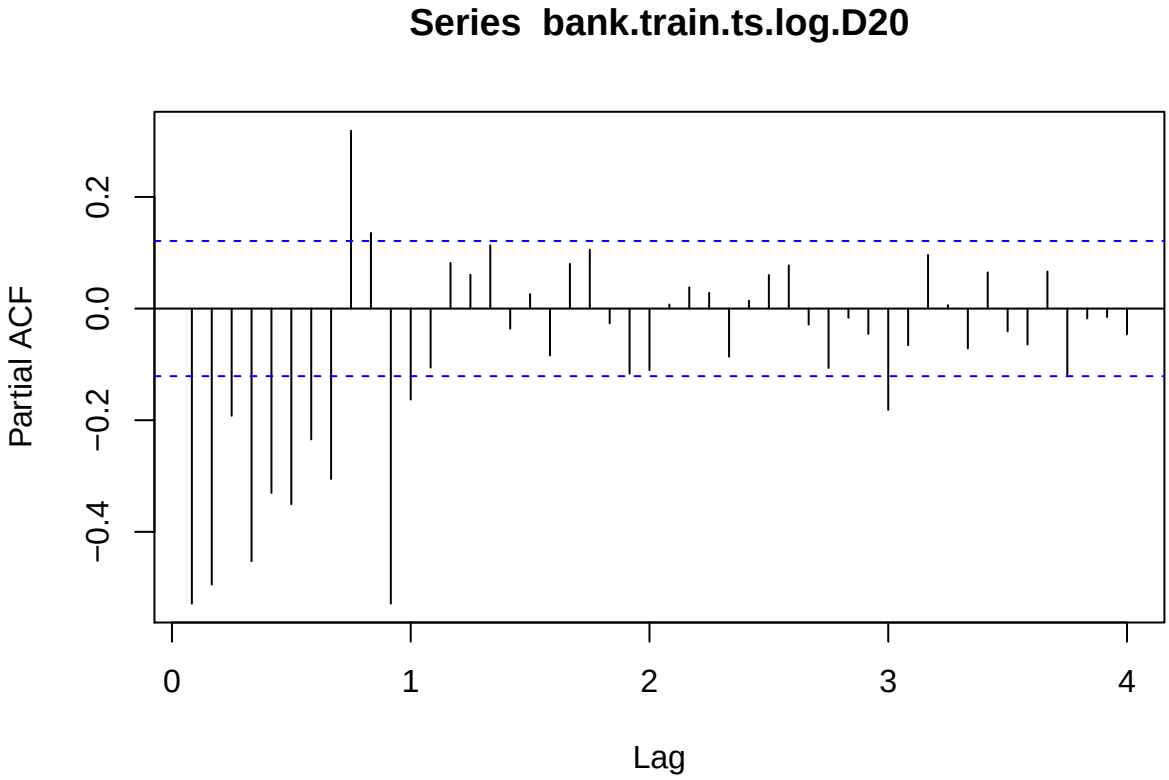
Futhermore, now we need to identify our candidates for p, P, q and Q. We will check acf and pacf plots after 2 trend differencing in order find these candidates. Lookng at acf plot we will decide the value for p and 5 seems to be a reasonable candidate. By looking at pacf we will decide our q value which seems to be 2. But for variety we will try all subset of combinations of these p and q values. Since we didn't require any seasonal differencing we will not search for P and Q.

Pick p, q, p <= 5, q <= 2

```
acf(bank.train.ts.log.D20, lag.max = 48)
```




```
pacf(bank.train.ts.log.D20, 48)
```



Forecast package comes with a nice to use function called auto.arima, we will run it for once to see what it suggest as a canididate model and chceck whether it is very different than what we have decided. As it is seen below auto arima suggests no seasonal differencing as we observed before but one trend differencing. For completeness we will also run this suggested model and calcualte rmse on the validation set.

```
auto.arima(bank.train.ts.log, d=1)
```

```
## Series: bank.train.ts.log
## ARIMA(3,1,2)(1,0,0)[12]
##
## Coefficients:
##           ar1      ar2      ar3     ma1     ma2    sar1
##       -1.8164  -1.3225  -0.4436  1.2309  0.2573  0.7521
## s.e.   0.2091   0.2833   0.1179  0.2269  0.2095  0.0418
##
## sigma^2 estimated as 0.005405:  log likelihood=311.3
## AIC=-608.6   AICc=-608.16   BIC=-583.6
```

```
arima.model.312.100 <- arima(bank.train.ts.log, order = c(3, 1, 2), seasonal = c(1, 0, 0))
```

With suggested auto.arima model let's define rmse and make predictions, to see how it perfoms in validation set. So it gives a number around ~0.0077, with AIC ~ -608, BIC ~ -583 and log likelihood ~311. We will compare these results with own model and we will eventually see that even though auto.arima is a handy and fast tool one should use their own insights and check acf, pacf and tests in order to define candidate models.

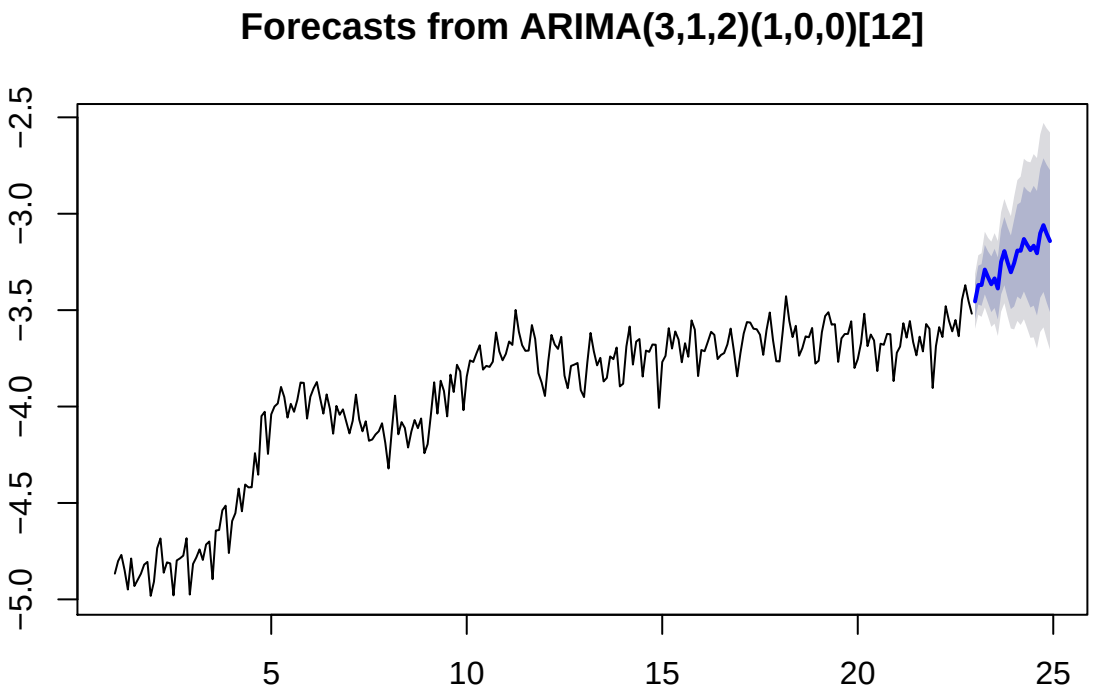
```
#rmse
rmse <- function(true, preds){return(sqrt(mean((true - preds)**2)))}

#preds
valid.preds <- forecast(arima.model.312.100, length(bank.valid.ts))
valid.rmse <- rmse(as.numeric(exp(valid.preds$mean)), bank.valid.ts)
```

```
paste(valid.rmse)
```

```
## [1] "0.00766115438404967"
```

This what predicitions on validation set looks like when modeled by suggested auto.arima model.

```
#plot predictions
plot(valid.preds)
```



Search for optimal p, q based on rmse on validation

Here we will do a grid search with our identified p <= 5 and q <=5, later check rmse on validation set. To see how each of these models perform compared to any other. We can also see (p, q) , rmse, aic and loglikelihood of each model from the output. So our own models give best model as ARIMA parameters: p = 3, q = 1, d = 2, rmse ~ 0.00520. It has a better rmse on validation set but it's AIC and loglikelihood is worse than what we got from auto.arima. Since our goal is to predict future but not to find the best fit for our data we will be in favor of our own model. Which is infact less complex in terms of p and q selection compared to auto.arima, which is prefered in terms of overfitting. So to wrap it up, we here favor a model which is performing better on unseen data and is less complex.

```
valid_rmse <- function(model, valid_ts){
  valid.preds <- forecast(model, length(valid_ts))
  valid.rmse <- rmse(as.numeric(exp(valid.preds$mean)), exp(valid_ts))
  return(valid.rmse)
}


p <- seq(5)
q <- seq(2)
comb <- expand.grid(p, q)
names(comb) <- c('p', 'q')
for (i in 1:nrow(comb)){
```

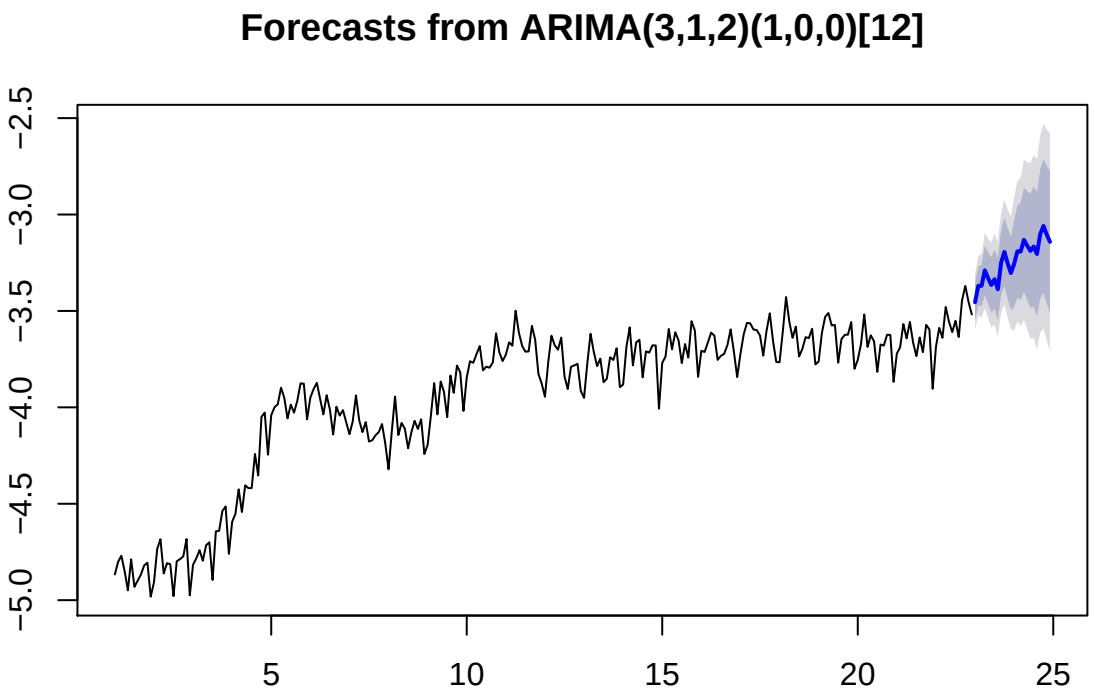
```
  p <- comb[i, 'p']
  q <- comb[i, 'q']
  print(paste(p, q))
  model <- arima(bank.train.ts.log, order = c(p, 2, q), seasonal = c(0, 0, 0))
  val_rmse <- valid_rmse(model, bank.valid.ts.log)
  print(val_rmse)
  print(model$aic)
  print(model$loglik)
  cat('\n')
}
```

```
## [1] "1 1"
## [1] 0.005742243
## [1] -397.3856
## [1] 201.6928
##
## [1] "2 1"
## [1] 0.005295898
## [1] -423.9008
## [1] 215.9504
##
## [1] "3 1"
## [1] 0.005203687
## [1] -426.6911
## [1] 218.3455
##
## [1] "4 1"
## [1] 0.005422778
## [1] -467.3553
## [1] 239.6777
##
## [1] "5 1"
## [1] 0.005748783
## [1] -467.1168
## [1] 240.5584
##
## [1] "1 2"
## [1] 0.01146711
## [1] -448.6996
## [1] 228.3498
##
## [1] "2 2"
## [1] 0.005634718
## [1] -419.9192
## [1] 214.9596
##
## [1] "3 2"
## [1] 0.01116891
## [1] -450.5636
## [1] 231.2818
##
## [1] "4 2"
## [1] 0.006402931
## [1] -468.5395
```

```
## [1] 241.2698
##
## [1] "5 2"
## [1] 0.005424213
## [1] -463.6069
## [1] 239.8035
```

Here we can see our validation predictions.

```
best.model <- arima(bank.train.ts.log, order = c(3, 2, 1), seasonal = c(0, 0, 0))
arima.preds <- forecast(best.model, h = length(bank.valid.ts.log))
plot(valid.preds)
```

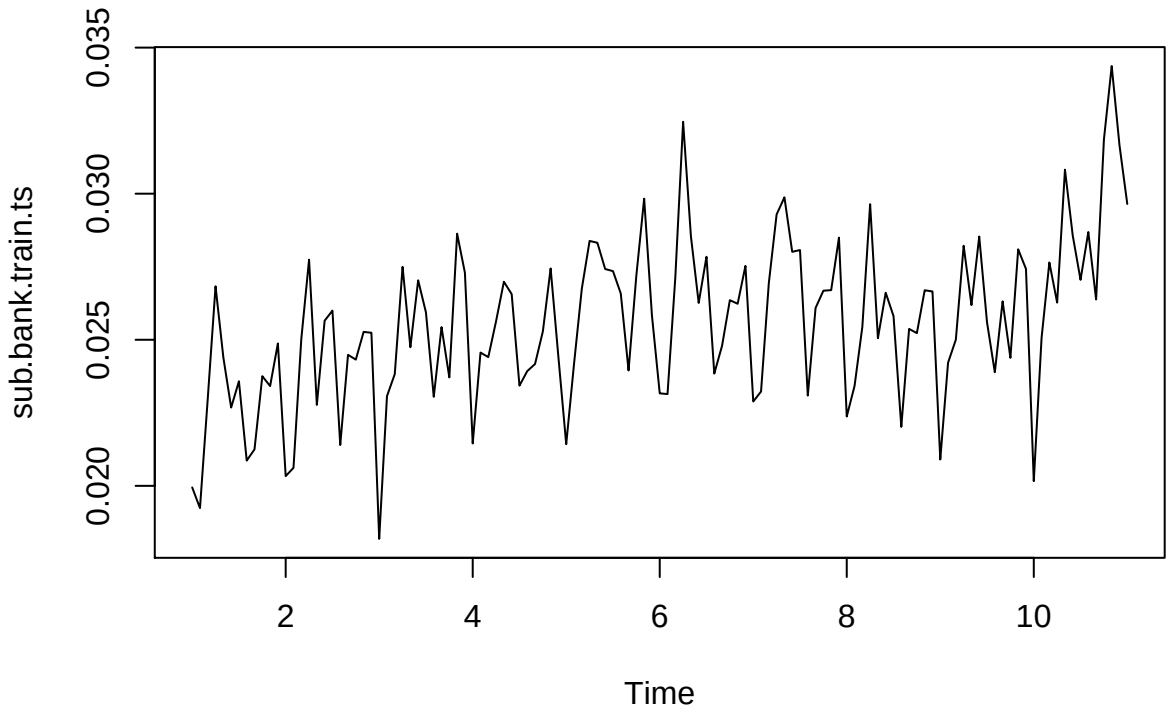


Subset time series for SARIMA model

In this part we will try out a hypothesis that our data infact is not stable, in other words it has a clear and instant pattern change around at year 10. To overcome this instant change we will take the time series that is after year 10 and use this subset of data in order to come up with a better predictive model. Anyway, our main goal here is to predict the future as good as possible. And our hypothesis is that having this subset will provide us a more generalizable model with better performance on unseen data. One drawback is that we can only compare this method in terms of rmse with our previous models that used full data. Because AIC, BIC or loglikelihood should be compared when time series are modeled with same data.

We will take data after year 12, this was determined after several experiments. We didn't apply any transforms since data seems to have a constant variance over time.

```
# number of years to discard from 24 years
# we can search for optimal years to discard by search
out_years = 12
sub.bank.train.ts <- ts(bankruptcy_ts[(out_years*12):264], frequency = 12)
bank.valid.ts <- ts(bankruptcy_ts[265:288], frequency = 12)

plot(sub.bank.train.ts)
```



ADF test is conducted again to see if our time series is indeed stationary.

```
adf.test(sub.bank.train.ts, k = 24)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  sub.bank.train.ts
## Dickey-Fuller = -1.843, Lag order = 24, p-value = 0.6418
## alternative hypothesis: stationary
```

One seasonal and trend differencing seems to be good enough.

```
adf.test(diff(diff(sub.bank.train.ts, lag = 12)), k = 12)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(diff(sub.bank.train.ts, lag = 12))
## Dickey-Fuller = -3.4474, Lag order = 12, p-value = 0.05027
## alternative hypothesis: stationary
```

We will do a grid search over p, P, q, Q...

```
valid_rmse <- function(model, valid_ts){
  valid.preds <- forecast(model, length(valid_ts))
  valid.rmse <- rmse(as.numeric(valid.preds$mean), valid_ts)
  return(valid.rmse)
}
```

```
P <- seq(3)
Q <- seq(5)
p <- seq(2)
q <- seq(3)
```

```
comb <- expand.grid('p' = p, 'q' = q, 'P' = P, 'Q' = Q)


best.rmse <- Inf
best.comb <- NA
for (i in 1:nrow(comb)){
  p <- comb[i, 'p']
  q <- comb[i, 'q']
  P <- comb[i, 'P']
  Q <- comb[i, 'Q']

  model <- arima(sub.bank.train.ts, order = c(p, 1, q), seasonal = list(order = c(P, 1, Q), period = 12)
  val_rmse <- valid_rmse(model, bank.valid.ts)
  if (val_rmse < best.rmse){
    best.rmse <- val_rmse
    best.comb <- c(p, q, P, Q)
  }
}
```

We observe that best parameters are (1, 1, 3) (3, 1, 2) with rmse of ~ 0.0029 which is lower than what we see during sarima and auto.sarima models. Another important note is that since are main goal is to come up with the best predictive model we choose our optimization method as LSE rather than MLE.

```
model <- arima(sub.bank.train.ts, order = c(1, 1, 3), seasonal = list(order = c(3, 1, 2), period = 12),
val_rmse <- valid_rmse(model, bank.valid.ts)
sarima.preds <- forecast(model, h = length(bank.valid.ts))
paste('best rmse', best.rmse)
```

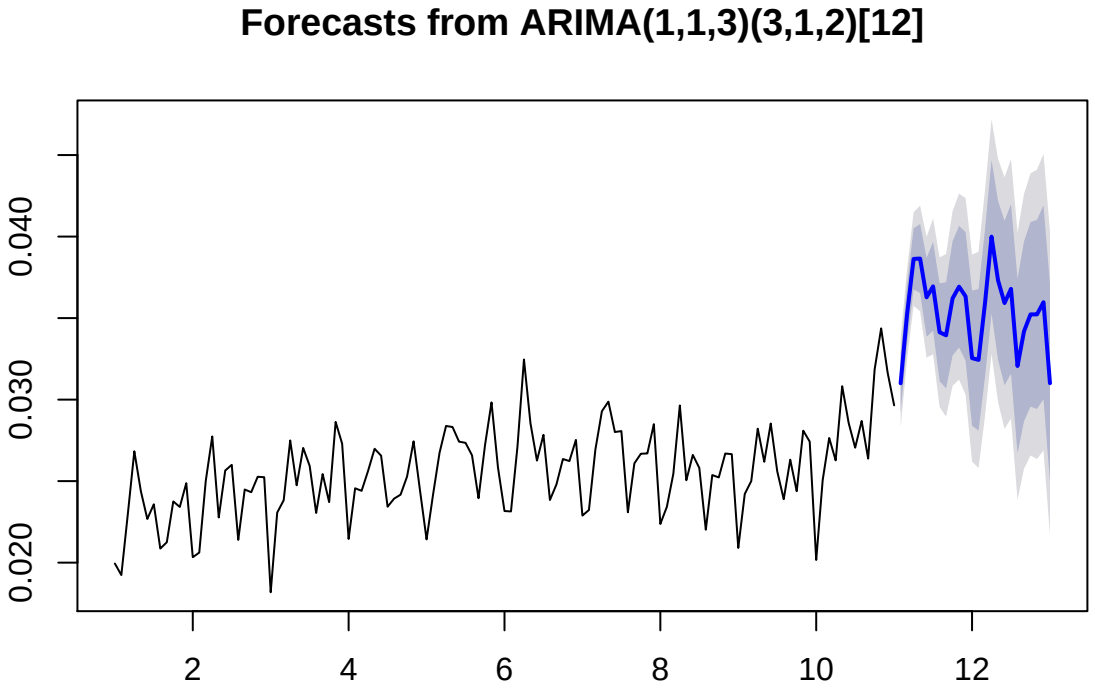
```
## [1] "best rmse 0.00296372580827692"
```

```
paste(c('p:', 'q:', 'P:', 'Q:'), best.comb)
```

```
## [1] "p: 1" "q: 3" "P: 3" "Q: 2"
```

Here we can see our sarima predictions on validation set.

```
plot(sarima.preds)
```



SARIMAX MODEL

In this part we will use exgenous data, assuming that there is a uni-directional relationship, meaning only independent variables effect bankruptcy not the other way around. We tried lagged 23 value of housing index, since it holds the highest correlation with bankcruptcy but it doesn't seem to perform better.

```
out_years <- 0 # years to exclude
train <- read.csv('../../../train.csv')[1:288,]
test <- read.csv('../../../test.csv')
h <- 0 # lag to use
train$House_Price_Index <- lag(train$House_Price_Index, n = h)
exo_endo_train <- train[(h + 1) + (out_years*12):263, ]
exo_endo_valid <- train[265:288, ]
```

```
names(exo_endo_train)
```

```
## [1] "Month"             "Unemployment_Rate" "Population"
## [4] "Bankruptcy_Rate"   "House_Price_Index"
```

```
valid_rmse <- function(model, valid_ts){
  valid.preds <- forecast(model, h = length(valid_ts),xreg = exo_endo_valid[c("Population", "House_Price
  valid.rmse <- rmse(as.numeric(valid.preds$mean), valid_ts)
  return(valid.rmse)
}
```

Again do a grid search over combinations of candidate sarima parameters. Our final best model in sarimax has rmse of ~0.0033 by using exogenous variables; population and house_price_index.

```
P <- c(0, seq(3))
Q <- c(0, seq(5))
p <- c(0, seq(2))
q <- c(0, seq(3))
```

```r
comb <- expand.grid('p' = p, 'q' = q, 'P' = P, 'Q' = Q)


best.rmse <- Inf
best.comb <- NA
for (i in 1:nrow(comb)){
  p <- comb[i, 'p']
  q <- comb[i, 'q']
  P <- comb[i, 'P']
  Q <- comb[i, 'Q']

  model <- arima(ts(exo_endo_train$Bankruptcy_Rate, frequency = 12), order = c(p, 1, q), seasonal = lis
                 method = 'CSS', xreg = exo_endo_train[c("Population", "House_Price_Index")])

  val_rmse <- valid_rmse(model, exo_endo_valid$Bankruptcy_Rate)
  if (val_rmse < best.rmse){
    best.rmse <- val_rmse
    best.comb <- c(p, q, P, Q)
  }
}
```

```r
# best with all data sarimax
best.rmse
```

```
## [1] 0.003342575
```

```r
best.comb
```

```
## [1] 1 2 2 5
```