Assignment 3: Unemployment and GDP

Joshua Goldberg

April, 25 2019

Load data

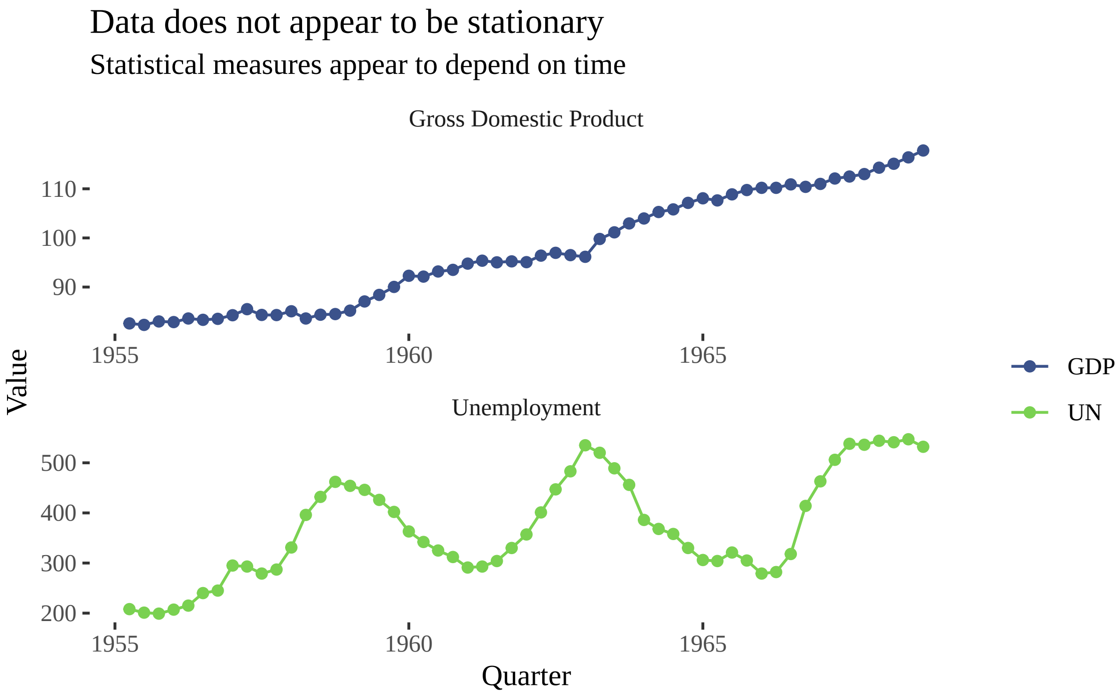
```
raw_data <- readxl::read_excel("Unemployment_GDP_UK.xlsx") %>%
  tidyr::fill(Year, .direction = "down") %>%
  gather(key = metric, value = value, -c(1, 2)) %>%
  janitor::clean_names() %>%
  mutate(month = case_when(
    quarter == 1 ~ 03,
    quarter == 2 ~ 06,
    quarter == 3 ~ 09,
    quarter == 4 ~ 12
  ),
  day = case_when(
    quarter == 1 ~ 31,
    quarter == 2 ~ 30,
    quarter == 3 ~ 30,
    quarter == 4 ~ 31
  ),
  date = ymd(glue::glue("{year}-{month}-{day}"))) %>%
  mutate(qtr = yearquarter(date)) %>%
  as_tsibble(key = "metric", index = "qtr") %>%
  select(qtr, metric, value)

train_data <- raw_data %>%
  filter_index("1955 Q1" ~ "1968 Q4")

test_data <- raw_data %>%
  filter_index("1968 Q4" ~ "1969 Q4")
```

Explore data

GDP shows a steady upward trend, while Unemployment shows some signs of seasonality. Considering the trend for GDP may assist in identifying stationarity.

```
metric_labels <- list(
  "GDP" = "Gross Domestic Product",
  "UN" = "Unemployment"
)

metric_labeller <- function(variable, value) {
  metric_labels[value]
}

train_data %>%
```

```r
  ggplot(aes(qtr, value, color = metric)) +
  geom_line() +
  geom_point() +
  labs(title = "Data does not appear to be stationary",
       subtitle = "Statistical measures appear to depend on time",
       x = "Quarter",
       y = "Value") +
  facet_wrap( ~ metric, nrow = 2, scales = "free", labeller = metric_labeller) +
  scale_color_viridis_d(name = NULL, begin = .25, end = .80)
```



`adf.test` confirms non-stationarity. `kpss.test` agrees with this as well. We see that GDP is stationary when considering the trend.

```r
train_data %>%
  split(.$metric) %>%
  map( ~ tseries::adf.test(.x$value))
```

```
## $GDP
##
##  Augmented Dickey-Fuller Test
##
## data:  .x$value
## Dickey-Fuller = -2.8458, Lag order = 3, p-value = 0.2337
## alternative hypothesis: stationary
##
##
## $UN
##
##  Augmented Dickey-Fuller Test
##
## data:  .x$value
## Dickey-Fuller = -3.3053, Lag order = 3, p-value = 0.07994
```

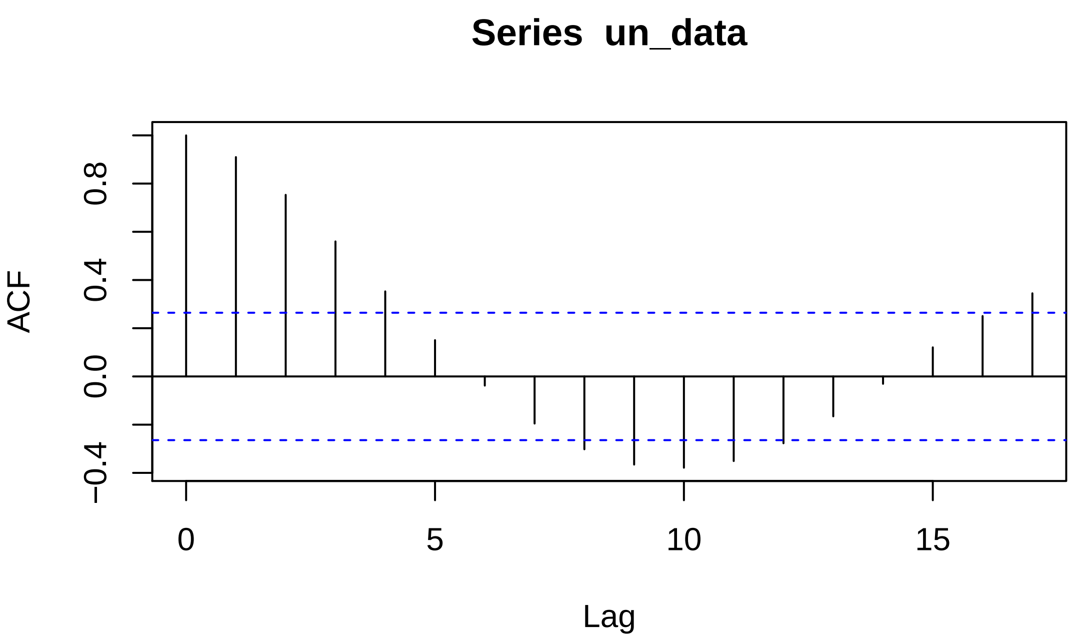
```
## alternative hypothesis: stationary
```

```r
train_data %>%
  split(.$metric) %>%
  map( ~ tseries::kpss.test(.x$value, null = "Trend"))
```

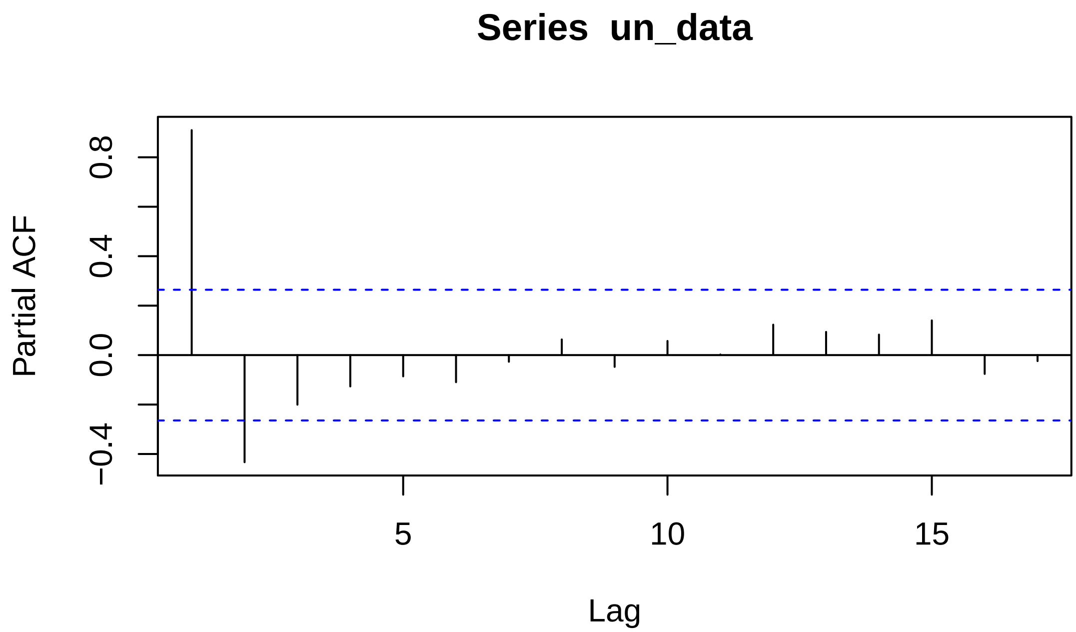
```
## $GDP
##
##  KPSS Test for Trend Stationarity
##
## data:  .x$value
## KPSS Trend = 0.17867, Truncation lag parameter = 3, p-value =
## 0.024
##
##
## $UN
##
##  KPSS Test for Trend Stationarity
##
## data:  .x$value
## KPSS Trend = 0.092915, Truncation lag parameter = 3, p-value = 0.1
```

The ACF plot confirms the non-stationarity of the data for unemployment due to the oscillation of the ACF plot.

```r
un_data <- train_data %>%
  filter(metric == "UN") %>%
  pull(value)

acf(un_data)
```

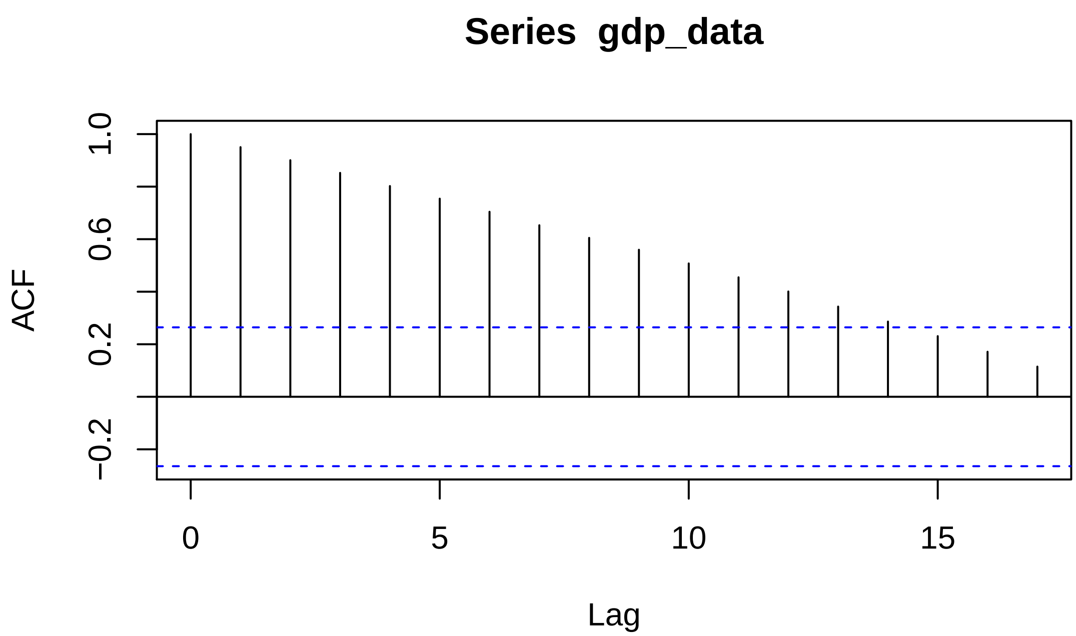



```r
pacf(un_data)
```

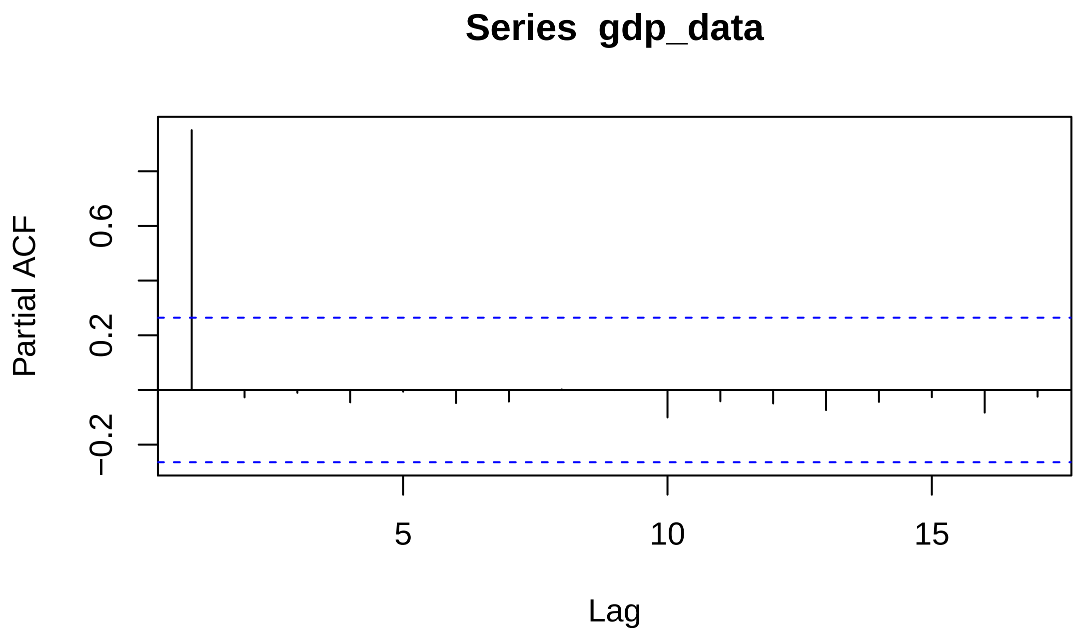


GDP shows a declining ACF with no non-zero lags that are not significant, which agrees with our generally observation that the process is stationary.

```
gdp_data <- train_data %>%
  filter(metric == "GDP") %>%
  pull(value)

acf(gdp_data)
```




```
pacf(gdp_data)
```



ARIMA modeling

1. Use datasets from 1955 to 1968 to build an ARMA or ARIMA models for UN and GDP. Use `auto.arima()` from package forecast.

```
un_boxcox <- BoxCox.lambda(un_data)

un_arima_model <- auto.arima(
  un_data,
  lambda = un_boxcox,
  max.p = 20,
  max.P = 20,
  max.D = 20,
  max.Q = 20,
  max.q = 20,
  max.order = 20
)

summary(un_arima_model)
```

```
## Series: un_data
## ARIMA(2,1,1)
## Box Cox transformation: lambda= 1.999924
##
## Coefficients:
##          ar1      ar2      ma1
##       1.6189  -0.7247  -0.9212
## s.e.  0.1005   0.0942   0.0723
##
## sigma^2 estimated as 75226830:  log likelihood=-565.42
## AIC=1138.83   AICc=1139.65   BIC=1146.79
##
## Training set error measures:
```

```
##                     ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 4.926746 21.76095 16.41774 1.308164 4.561678 0.6809199
##                     ACF1
## Training set -0.08438812
```

```r
gdp_boxcox <- BoxCox.lambda(gdp_data)
gdp_arima_model <- auto.arima(gdp_data, lambda = gdp_boxcox)
summary(gdp_arima_model)
```

```
## Series: gdp_data
## ARIMA(0,1,0) with drift
## Box Cox transformation: lambda= 1.612917
##
## Coefficients:
##         drift
##       10.9643
## s.e.   1.9210
##
## sigma^2 estimated as 203:  log likelihood=-219.58
## AIC=443.16   AICc=443.39   BIC=447.13
##
## Training set error measures:
##                       ME      RMSE       MAE         MPE      MAPE
## Training set -0.01398637 0.8595533 0.6429265 -0.04465616 0.6759638
##                   MASE       ACF1
## Training set 0.7424729 0.08570203
```

```r
gdp_arima_model <- auto.arima(gdp_data)
summary(gdp_arima_model)
```

```
## Series: gdp_data
## ARIMA(0,1,0) with drift
##
## Coefficients:
##        drift
##       0.6519
## s.e.  0.1168
##
## sigma^2 estimated as 0.7508:  log likelihood=-68.38
## AIC=140.75   AICc=140.99   BIC=144.73
##
## Training set error measures:
##                       ME     RMSE       MAE         MPE      MAPE
## Training set 0.001489966 0.850614 0.6402644 -0.02313787 0.6734653
##                   MASE       ACF1
## Training set 0.7393986 0.06539675
```

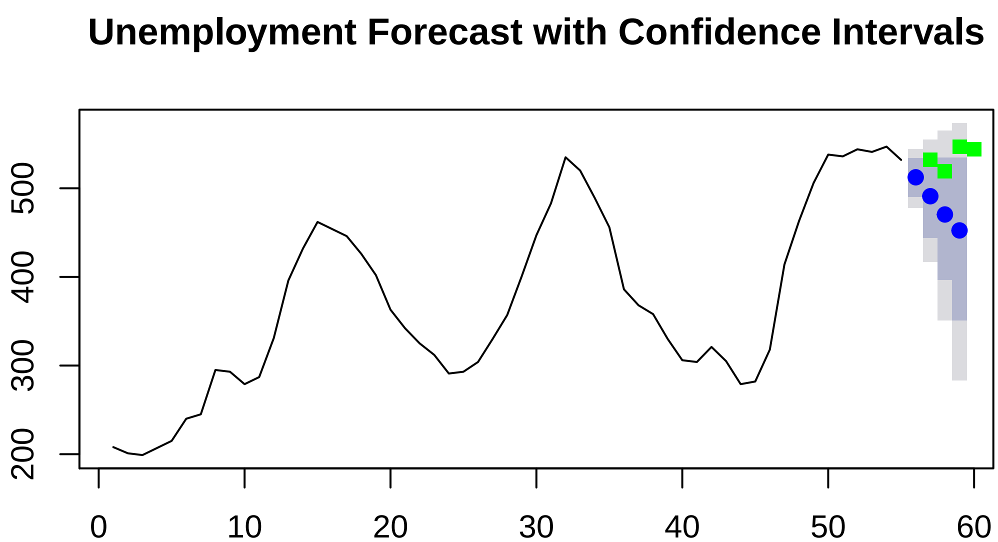
GDP model has a much better AIC without `BoxCox.lambda`, so this model is preferred.

2. Justify why you chose (ARMA or ARIMA) one over the other. Note there will be 2 models, one for UN and another for GDP.

`auto.arima` fixes differencing at 1, so ARIMA is preferred in this case due to the transformation required to achieve stationarity.

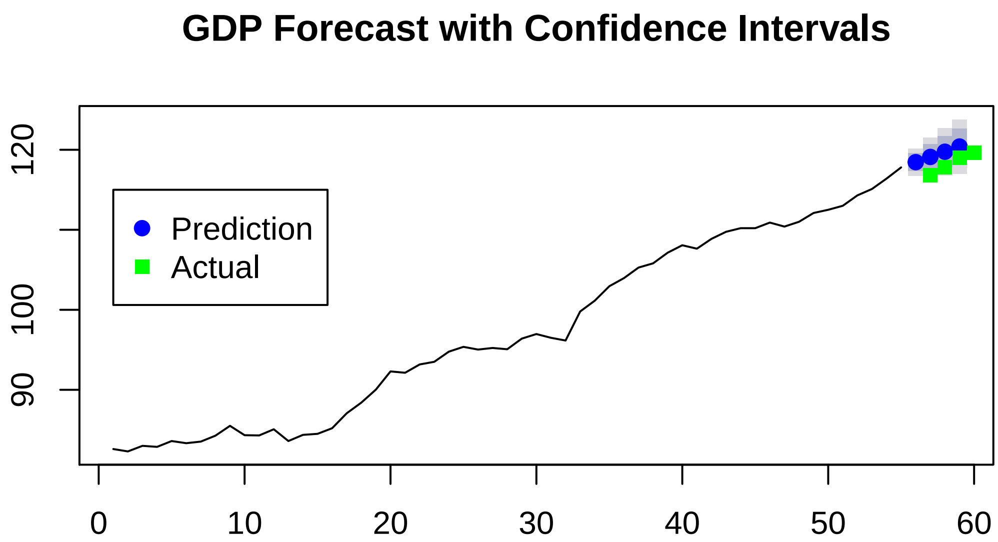
3. Compare your forecasts with the actual values using error = actual - estimate and plot the errors.

```r
un_test <- test_data %>%
  filter(metric == "UN") %>%
  pull(value)

gdp_test <- test_data %>%
  filter(metric == "GDP") %>%
  pull(value)

un_pred <- forecast(un_arima_model, h = 4)
gdp_pred <- forecast(gdp_arima_model, h = 4)

plot(un_pred, main = "Unemployment Forecast with Confidence Intervals")
legend(
  1,
  2500,
  legend = c("Prediction", "Actual"),
  col = c("blue", "green"),
  pch = c(15, 15)
)
points(
  x = c(57:60),
  y = un_test,
  col = "green",
  pch = 15
)
```




```r
plot(gdp_pred, main = "GDP Forecast with Confidence Intervals")
legend(
  1,
  115,
  legend = c("Prediction", "Actual"),
  col = c("blue", "green"),
  pch = c(19, 15)
)
points(
  x = c(57:60),
  y = gdp_test,
```

```r
  col = "green",
  pch = 15
)
```




```r
preds <- data.frame(pred = c(gdp_pred$mean %>% as.vector(), un_pred$mean %>% as.vector()))

errors <- test_data %>%
  bind_cols(preds) %>%
  mutate(error = value - pred)

errors
```

```
## # A tsibble: 8 x 5 [1Q]
## # Key:       metric [2]
##       qtr metric value  pred   error
##     <qtr> <chr>  <dbl> <dbl>   <dbl>
## 1 1969 Q1 GDP     117.  118.  -1.65
## 2 1969 Q2 GDP     118.  119.  -1.30
## 3 1969 Q3 GDP     119   120.  -0.756
## 4 1969 Q4 GDP     120.  120.  -0.807
## 5 1969 Q1 UN      532   512.  19.6
## 6 1969 Q2 UN      519   491.  27.9
## 7 1969 Q3 UN      547   470.  76.6
## 8 1969 Q4 UN      544   452.  91.6
```

5. Calculate the sum of squared error for each UN and GDP models.

```r
errors %>%
  as_tibble() %>%
  group_by(metric) %>%
  summarise(sum_squared_errors = sqrt(sum(error^2)))
```

```
## # A tibble: 2 x 2
##   metric sum_squared_errors
##   <chr>               <dbl>
## 1 GDP                  2.38
## 2 UN                 124.
```

Regression

1. Unemployment as the independent variable and GDP as the dependent variable - use data from 1955 to 1968 to build the model. Forecast for 1969 and plot the errors as a percentage of the mean. Also calculate the sum of squared(error) as a percentage of the mean.

```
lm_train_data <- train_data %>%
  spread(metric, value)

lm_test_data <- test_data %>%
  spread(metric, value)

gdp_un_lm <- lm(GDP ~ UN, lm_train_data)
summary(gdp_un_lm)
```

```
##
## Call:
## lm(formula = GDP ~ UN, data = lm_train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.990  -6.906  -1.535   8.126  18.108
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 76.23803    5.00889  15.221  < 2e-16 ***
## UN           0.05682    0.01299   4.374 5.74e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.88 on 53 degrees of freedom
## Multiple R-squared:  0.2653, Adjusted R-squared:  0.2514
## F-statistic: 19.13 on 1 and 53 DF,  p-value: 5.737e-05
```

```
(errors <- lm_test_data %>%
   as_tibble() %>%
   mutate(error = GDP - predict(gdp_un_lm, lm_test_data),
         pct_mean = (error / mean(UN)) * 100))
```

```
## # A tibble: 4 x 5
##       qtr   GDP    UN error pct_mean
##     <qtr> <dbl> <dbl> <dbl>    <dbl>
## 1 1969 Q1  117.   532  10.3     1.93
## 2 1969 Q2  118.   519  12.1     2.25
## 3 1969 Q3  119    547  11.7     2.18
## 4 1969 Q4  120.   544  12.5     2.33
```

```
errors %>%
  as_tibble() %>%
  summarise(total_mean_pct_error = (sum(pct_mean^2) / mean(GDP)) * 100)
```

```
## # A tibble: 1 x 1
##   total_mean_pct_error
##                  <dbl>
## 1                 16.0
```

GDP as the independent variable and UN as the dependent variable - use data from 1955 to 1968 to build the model. Forecast for 1969 and plot the errors as a percentage of the mean. Also calculate the sum of squared(error) as a percentage of the mean of the actual values.

```
lm_train_data <- train_data %>%
  spread(metric, value)

lm_test_data <- test_data %>%
  spread(metric, value)

un_gdp_lm <- lm(UN ~ GDP, lm_train_data)
summary(un_gdp_lm)
```

```
##
## Call:
## lm(formula = UN ~ GDP, data = lm_train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -152.65  -69.02  -16.52   86.44  168.89
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -82.794    104.602  -0.792    0.432
## GDP            4.668      1.067   4.374 5.74e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 89.55 on 53 degrees of freedom
## Multiple R-squared:  0.2653, Adjusted R-squared:  0.2514
## F-statistic: 19.13 on 1 and 53 DF,  p-value: 5.737e-05
```

```
(errors <- lm_test_data %>%
   as_tibble() %>%
   mutate(error = UN - predict(un_gdp_lm, lm_test_data),
          pct_mean = (error / mean(UN)) * 100))
```

```
## # A tibble: 4 x 5
##       qtr   GDP    UN error pct_mean
##     <qtr> <dbl> <dbl> <dbl>    <dbl>
## 1 1969 Q1  117.   532  69.5    13.0
## 2 1969 Q2  118.   519  51.9     9.69
## 3 1969 Q3  119    547  74.3    13.9
## 4 1969 Q4  120.   544  68.5    12.8
```

```
errors %>%
  as_tibble() %>%
  summarise(total_mean_pct_error = (sum(pct_mean^2) / mean(UN)) * 100)
```

```
## # A tibble: 1 x 1
##   total_mean_pct_error
##                  <dbl>
## 1                 115.
```

3. Compare the 2 models using the sum of squared error as a percentage of the mean of the actual values - any reason to believe which should be the independent and the dependent variable?

If only on the basis of squared error as a percentage, the model with GDP as the response and UN as the independent performs better.