CBAM Data Analysis

Anusha Kuppahally

2023-03-27

Contents

Loading Data

Statistical Capability Data

```
SPI_index = read_delim("~/Downloads/SPI_index.csv")
```

```
## Rows: 3488 Columns: 79
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): country, iso3c, income, region
## dbl (75): date, SPI.INDEX.PIL1, SPI.INDEX.PIL2, SPI.INDEX.PIL3, SPI.INDEX.PI...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
SPI_index_2019 = SPI_index[grep('2019', SPI_index$date),]
stats_capab_data = SPI_index_2019[c(1,2,9,76,77)]
colnames(stats_capab_data) = c("Country_Name",
↪  "Country_Code","Statistical_Capability", "Income", "Region")
stats_capab_data$Income <- factor(stats_capab_data$Income , levels=c("Low
↪  income", "Lower middle income", "Upper middle income", "High income"))
```

```
dim(stats_capab_data)
```

```
## [1] 218   5
```

LDC Column

```
ldcs = c("Angola", "Benin", "Burkina Faso", "Burundi", "Central African
↪  Republic", "Chad", "Comoros", "Congo, Dem. Rep.", "Djibouti", "Eritrea",
↪  "Ethiopia", "Gambia, The", "Guinea", "Guinea-Bissau", "Lesotho", "Liberia",
↪  "Madagascar", "Malawi", "Mali", "Mauritania", "Mozambique", "Niger",
↪  "Rwanda", "Sao Tome and Principe", "Senegal", "Sierra Leone", "Somalia",
↪  "South Sudan", "Sudan", "Togo", "Uganda", "Tanzania", "Zambia",
↪  "Afghanistan", "Bangladesh", "Bhutan", "Cambodia", "Lao PDR", "Myanmar",
↪  "Nepal", "Timor-Leste", "Yemen, Rep.", "Haiti", "Kiribati", "Solomon
↪  Islands", "Tuvalu")

stats_capab_data$LDC = with(stats_capab_data,
↪  ifelse(stats_capab_data$Country_Name %in% ldcs, 1, 0))
table(stats_capab_data$LDC)
```

```
##
##   0   1
## 172  46
```

Export Data

```
comtrade_aluminum = read.csv('~/Downloads/comtrade (4).csv')
aluminum_76 = comtrade_aluminum[c(13,14,32)]
colnames(aluminum_76) = c("Country_Name","Country_Code","Aluminum_Exports")
dim(aluminum_76)
```

```
## [1] 161   3
```

Energy Data

```
iea_data = read_excel('~/Downloads/energy statistics data.xlsx')
dim(iea_data)
```

```
## [1] 146   9
```

Data Cleaning

Standardizing country names.

```
aluminum_76 <- mutate(aluminum_76, Country_Name=recode(Country_Name, "Bolivia
↪  (Plurinational State of)" = "Bolivia",
     "Bosnia Herzegovina" = "Bosnia and Herzegovina",
     "Congo" = "Congo, Rep.",
     "Czechia" = "Czech Republic",
     "Dominican Rep." = "Dominican Republic",
     "China, Hong Kong SAR" = "Hong Kong SAR, China",
     "Côte d'Ivoire" = "Cote d'Ivoire",
     "Rep. of Korea" = "Korea, Rep.",
     "Kyrgyzstan" = "Kyrgyz Republic",
     "Lao People's Dem. Rep."= "Lao PDR",
     "China, Macao SAR" = "Macao SAR, China",
     "Slovakia" = "Slovak Republic",
     "Viet Nam" = "Vietnam",
     "United Rep. of Tanzania" = "Tanzania",
     "USA" = "United States",
     "Yemen" = "Yemen, Rep.",
```

```
    "Bahamas" = "Bahamas, The",
    "Central African Rep." = "Central African Republic",
    "Dem. Rep of Congo" = "Congo, Dem. Rep.",
    "Gambia" = "Gambia, The",
    "Rep. of Moldova" = "Moldova",
    "Saint Lucia" = "St. Lucia",
    "Egypt" = "Egypt, Arab Rep.",
    "Venezuela" = "Venezuela, RB",
    "Syria" = "Syrian Arab Republic",
    "Solomon Isds" = "Solomon Islands",
    "Br. Virgin Isds" = "British Virgin Islands",
    "Cayman Isds" = "Cayman Islands",
    "Dem. Rep. of the Congo" = "Congo, Dem. Rep.",
    "Faeroe Isds" = "Faroe Islands",
    "State of Palestine" = "West Bank and Gaza",
    "Iran" = "Iran, Islamic Rep.",
    "Curaçao" = "Curacao",
    "Saint Maarten" = "Sint Maarten (Dutch part)",
    "Marshall Isds" = "Marshall Islands",
    "Turks and Caicos Isds" = "Turks and Caicos Islands",
    "Saint Vincent and the Grenadines" = "St. Vincent and the Grenadines"))
```

```
df_stats_exports_combined <- inner_join(aluminum_76, stats_capab_data,
↪  by="Country_Code")

df_stats_exports_combined = df_stats_exports_combined[-c(4)]
```

```
iea_data <- mutate(iea_data, Country_Name_New =
↪  stringr::str_replace_all(Country_Name,"[:space:]"," "))

df_stats_exports_combined <- mutate(df_stats_exports_combined, Country_Name_New =
↪  stringr::str_replace_all(Country_Name.x,"[:space:]"," "))
```

Creating Final Data Set

```
df_all <- inner_join(df_stats_exports_combined, iea_data, by="Country_Name_New")
```

```
df_filtered=df_all[-c(9, 12, 13, 14, 15)]
colnames(df_filtered) = c("Country_Name", "Country_Code", "Trade_Value",
↪  "Statistical_Capability", "Income", "Region", "LDC", "Country_Name_New",
↪  "Population", "GDP", "Electricity_Consumption", "CO2_Emissions")
```

```
dim(df_filtered)
```

```
## [1] 108  12
```

```
head(df_filtered)
```

```
##   Country_Name Country_Code Trade_Value Statistical_Capability
## 1      Albania          ALB    37139568               75.38292
## 2      Algeria          DZA     1834154               55.14917
## 3       Angola          AGO      243849               54.94583
## 4   Azerbaijan          AZE    34976688               68.14125
## 5    Argentina          ARG    22199243               64.59583
## 6    Australia          AUS    63475543               88.24167
##                 Income                      Region LDC Country_Name_New
## 1 Upper middle income      Europe & Central Asia   0          Albania
## 2 Lower middle income Middle East & North Africa   0          Algeria
## 3 Lower middle income         Sub-Saharan Africa   1           Angola
## 4 Upper middle income      Europe & Central Asia   0       Azerbaijan
## 5 Upper middle income  Latin America & Caribbean   0        Argentina
## 6         High income        East Asia & Pacific   0        Australia
##   Population    GDP Electricity_Consumption CO2_Emissions
## 1        2.9   13.0                    6.6           4.0
## 2       43.1  177.4                   71.5         142.4
## 3       31.8  109.8                   13.7          18.8
## 4       10.0   53.5                   22.5          34.1
## 5       44.9  570.5                  129.4         162.2
## 6       25.4 1338.6                  251.1         380.7
```

```
table(df_filtered$LDC,df_filtered$Income)
```

```
##
##     Low income Lower middle income Upper middle income High income
##   0          2                  24                  38          26
##   1          8                  10                   0           0
```

Calculated Columns

- Aluminum exports originally were in kg of aluminum, and needed to be divided by 1000 to get metric tonne
- Energy intensity is in kwh/tonne (15474 is the default value for Europe from the International ALuminium Institute for 2019)
- The conversion factor is in CO_2/kwh (using the average across all countries to simulate default values)

- GDP is in billion 2015 USD
- Formulas:
  - Tonne Aluminum * (kwh/tonne) *(tCO_2/kwh) = tonne CO_2 total
  - Percent Cost = Default Cost/(GDP/10^7)

```
df_filtered$Aluminum_Exports = df_filtered$Trade_Value/1.794

df_filtered$Conversion_Factor = (df_filtered$CO2_Emissions *
↪  10^6)/(df_filtered$Electricity_Consumption * 10^9)

df_filtered$Default_CO2_Output = (df_filtered$Aluminum_Exports /1000) * 15474 *
↪  mean(df_filtered$Conversion_Factor)

df_filtered$Default_Cost = df_filtered$Default_CO2_Output * 86.284

df_filtered$Default_Percent_Cost =
↪  (df_filtered$Default_Cost/(df_filtered$GDP*10^7))
```

Visualizations

Statistical Capability Data

```
table(stats_capab_data$Region,stats_capab_data$Income)
```

```
##
##                               Low income Lower middle income Upper middle income
##   East Asia & Pacific                   1                  12                  10
##   Europe & Central Asia                 1                   4                  15
##   Latin America & Caribbean             1                   4                  20
##   Middle East & North Africa            2                   6                   5
##   North America                         0                   0                   0
##   South Asia                            1                   6                   1
##   Sub-Saharan Africa                   23                  18                   5
##
##                               High income
##   East Asia & Pacific                  15
##   Europe & Central Asia                38
##   Latin America & Caribbean            17
##   Middle East & North Africa            8
##   North America                         3
##   South Asia                            0
##   Sub-Saharan Africa                    2
```

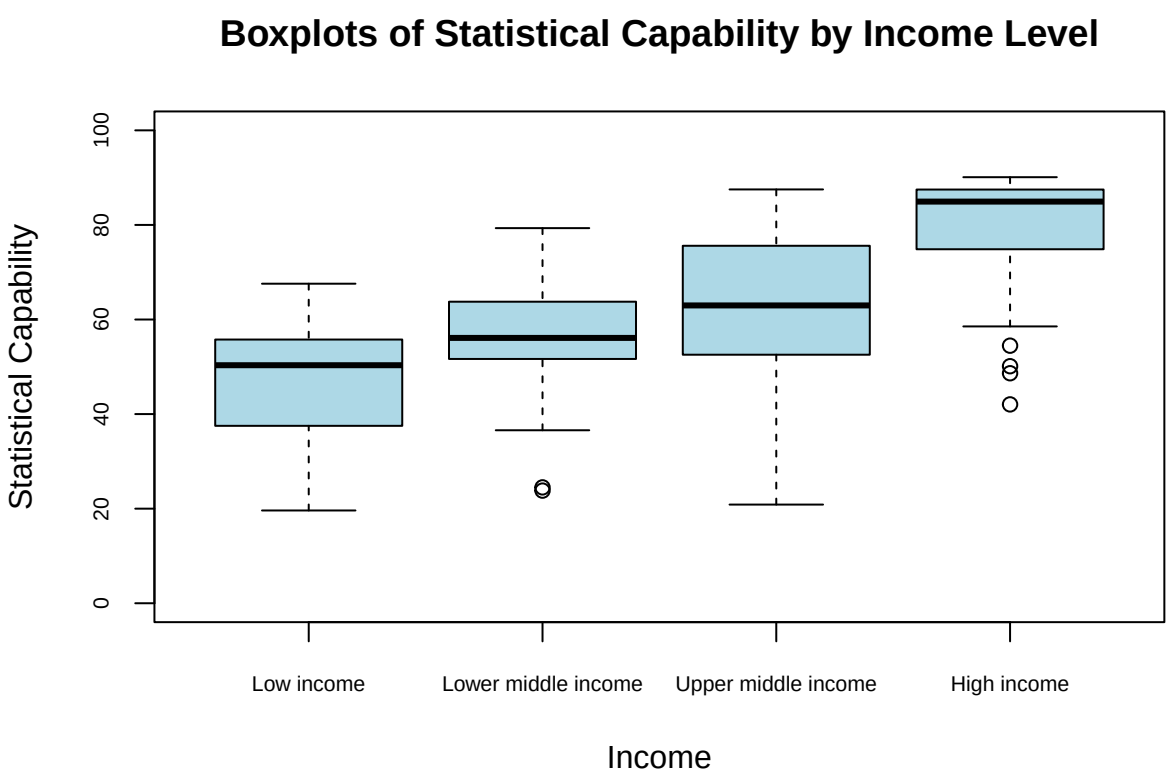
```r
table(stats_capab_data$LDC,stats_capab_data$Income)
```

```
##
##     Low income Lower middle income Upper middle income High income
##   0          3                   31                   55          83
##   1         26                   19                    1           0
```

```r
by(stats_capab_data$Statistical_Capability, stats_capab_data$Income, summary)
```

```
## stats_capab_data$Income: Low income
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   19.62   37.77   50.33   46.96   55.23   67.57       3
## ------------------------------------------------------------
## stats_capab_data$Income: Lower middle income
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   23.82   51.66   56.11   56.68   63.76   79.32       1
## ------------------------------------------------------------
## stats_capab_data$Income: Upper middle income
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   20.86   52.73   62.98   61.76   75.49   87.51       8
## ------------------------------------------------------------
## stats_capab_data$Income: High income
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   42.05   74.86   84.93   78.53   87.48   90.09      32
```

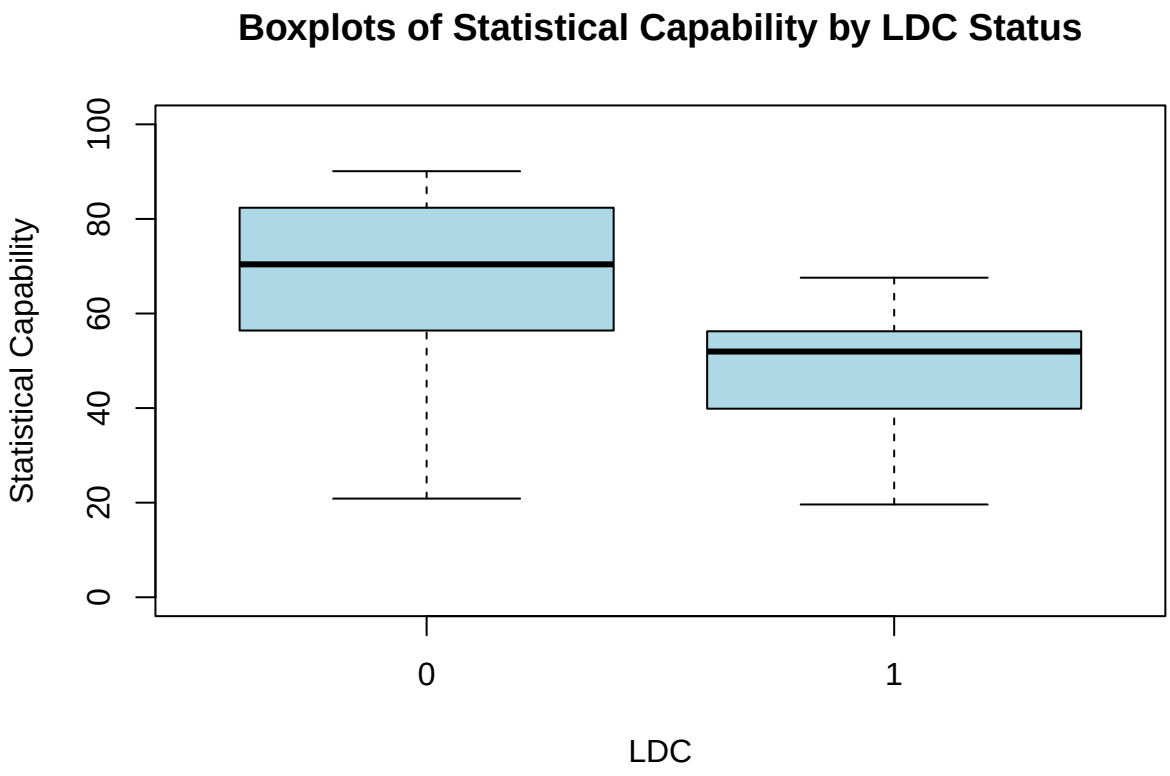
```r
boxplot(stats_capab_data$Statistical_Capability ~ stats_capab_data$Income, main =
↪  "Boxplots of Statistical Capability by Income Level", ylab="Statistical
↪  Capability", xlab="Income", cex.axis = 0.7, col = "light blue",
↪  ylim=c(0,100))
```




```
by(stats_capab_data$Statistical_Capability, stats_capab_data$LDC, summary)
```

```
## stats_capab_data$LDC: 0
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   20.86   56.54   70.40   67.51   82.35   90.09      40
## ------------------------------------------------------------
## stats_capab_data$LDC: 1
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   19.62   40.32   51.95   48.98   56.23   67.57       4
```

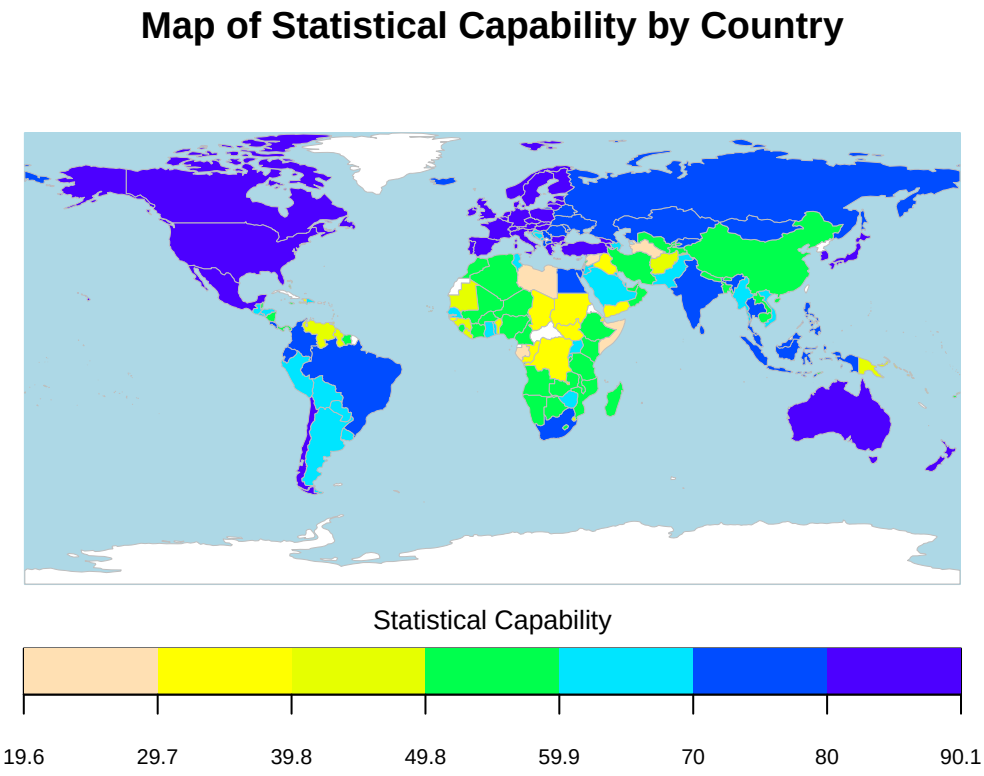
```
boxplot(stats_capab_data$Statistical_Capability ~ stats_capab_data$LDC,
↪  ylab="Statistical Capability", xlab="LDC",col = "light blue", ylim=c(0,100),
↪  main = "Boxplots of Statistical Capability by LDC Status")
```




```
legendTitle = "Statistical Capability"

sdat <- joinCountryData2Map(stats_capab_data, joinCode="ISO3",
↪  nameJoinColumn="Country_Code", verbose = TRUE)
```

```
## 215 codes from your data successfully matched countries in the map
## 3 codes from your data failed to match with a country code in the map
##      failedCodes
## [1,] "CHI"
## [2,] "GIB"
## [3,] "XKX"
## 28 codes from the map weren't represented in your data
```

```
mapParams = mapCountryData(sdat, nameColumnToPlot="Statistical_Capability",
↪  catMethod="fixedWidth", addLegend = FALSE, missingCountryCol = "white",
↪  mapTitle="Map of Statistical Capability by Country", oceanCol = "lightblue",
↪  colourPalette = "topo")

do.call(addMapLegend, c(mapParams, labelFontSize = 0.7, horizontal = TRUE,
↪  legendShrink = 0.928, legendLabels = "all",legendArgs=c(mtext(paste(""),
↪  side=3, adj=0.5, padj=0.4, cex=0.8),
                                                mtext(paste(legendTitle, sep=""),
                                                ↪  side=1, adj=0.5, padj=-0.8,
                                                ↪  cex=0.8)), digits=3))
```



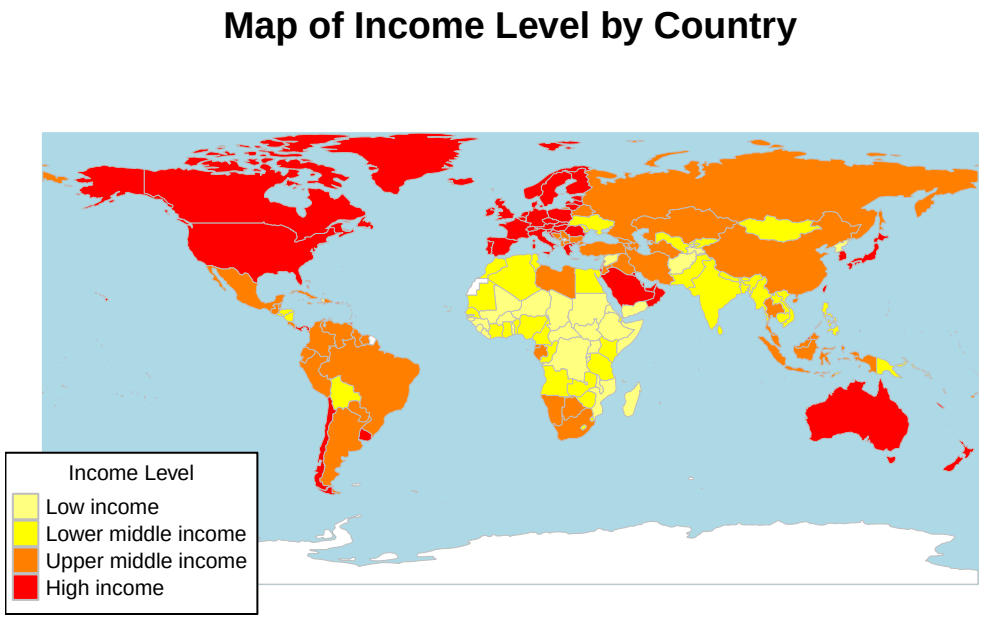

```
sdat <- joinCountryData2Map(stats_capab_data, joinCode="ISO3",
 ↪  nameJoinColumn="Country_Code", verbose = TRUE)
```

```
## 215 codes from your data successfully matched countries in the map
## 3 codes from your data failed to match with a country code in the map
##      failedCodes
## [1,] "CHI"
## [2,] "GIB"
## [3,] "XKX"
## 28 codes from the map weren't represented in your data
```

```
mapParams = mapCountryData(sdat, nameColumnToPlot="Income",
 ↪  catMethod="Categorical", addLegend = FALSE, missingCountryCol = "white",
 ↪  mapTitle="Map of Income Level by Country", oceanCol = "lightblue",
 ↪  colourPalette = "heat")
```

```
## using catMethod='categorical' for non numeric data in mapCountryData
```

```
do.call( addMapLegendBoxes, c( mapParams, title = "Income Level", cex = 0.7))
```



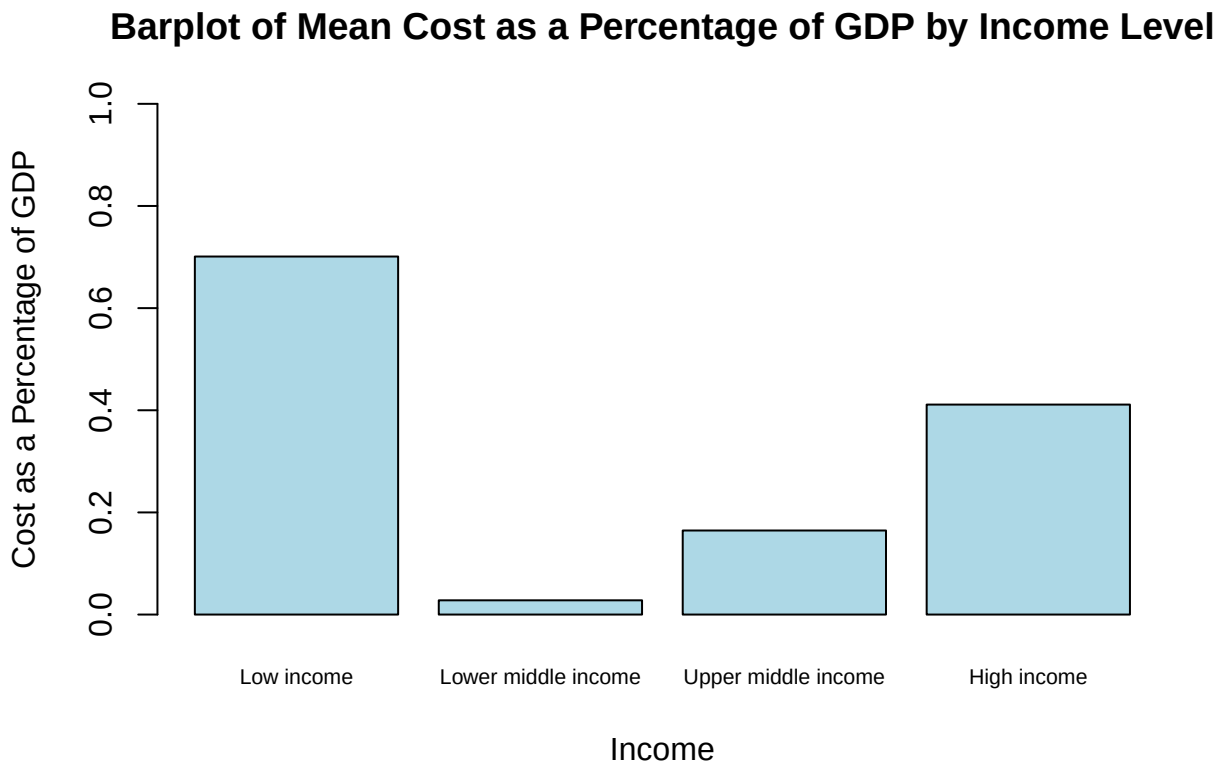
Cost Data

```
by(df_filtered$Default_Percent_Cost, df_filtered$Income, summary)
```

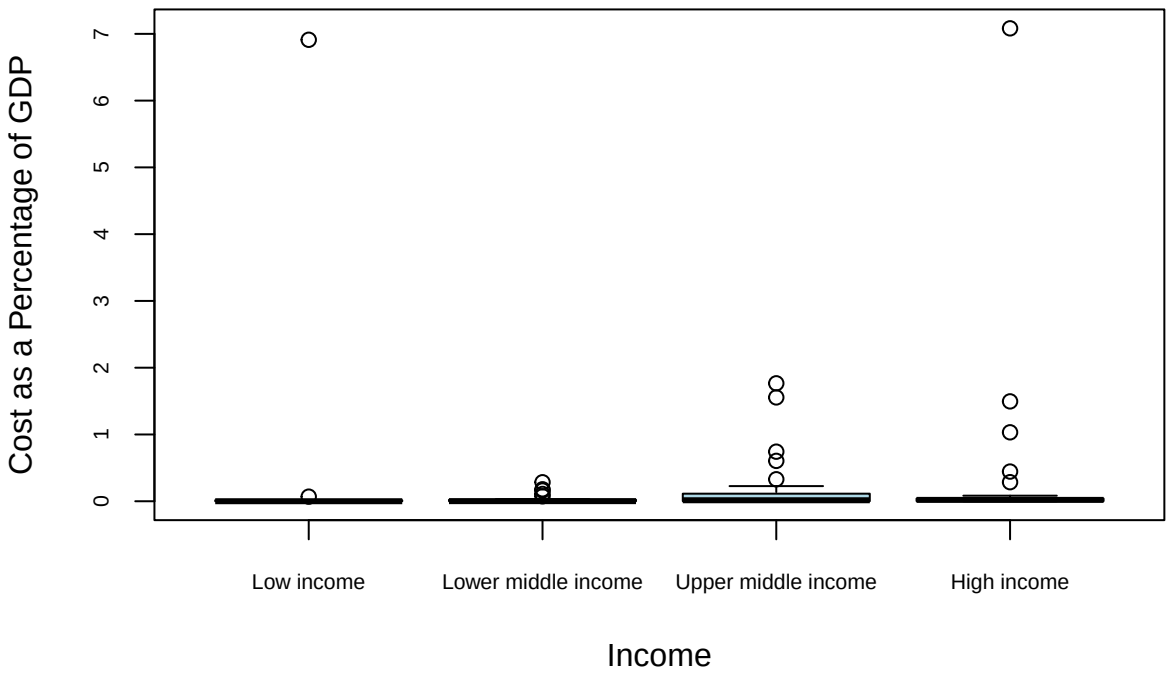
```
## df_filtered$Income: Low income
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000032 0.000140 0.000437 0.701000 0.014752 6.911363
## ------------------------------------------------------------
## df_filtered$Income: Lower middle income
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 2.250e-06 1.762e-04 1.422e-03 2.787e-02 1.805e-02 2.841e-01
## ------------------------------------------------------------
## df_filtered$Income: Upper middle income
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000056 0.001193 0.014673 0.164603 0.105081 1.765953
## ------------------------------------------------------------
## df_filtered$Income: High income
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.000075 0.003079 0.020292 0.411137 0.038213 7.082377
```

```
barplot(by(df_filtered$Default_Percent_Cost, df_filtered$Income, mean),cex.names
↪  = 0.7, ylim = c(0,1), main = "Barplot of Mean Cost as a Percentage of GDP by
↪  Income Level", ylab = "Cost as a Percentage of GDP", xlab = "Income",col =
↪  "lightblue")
```



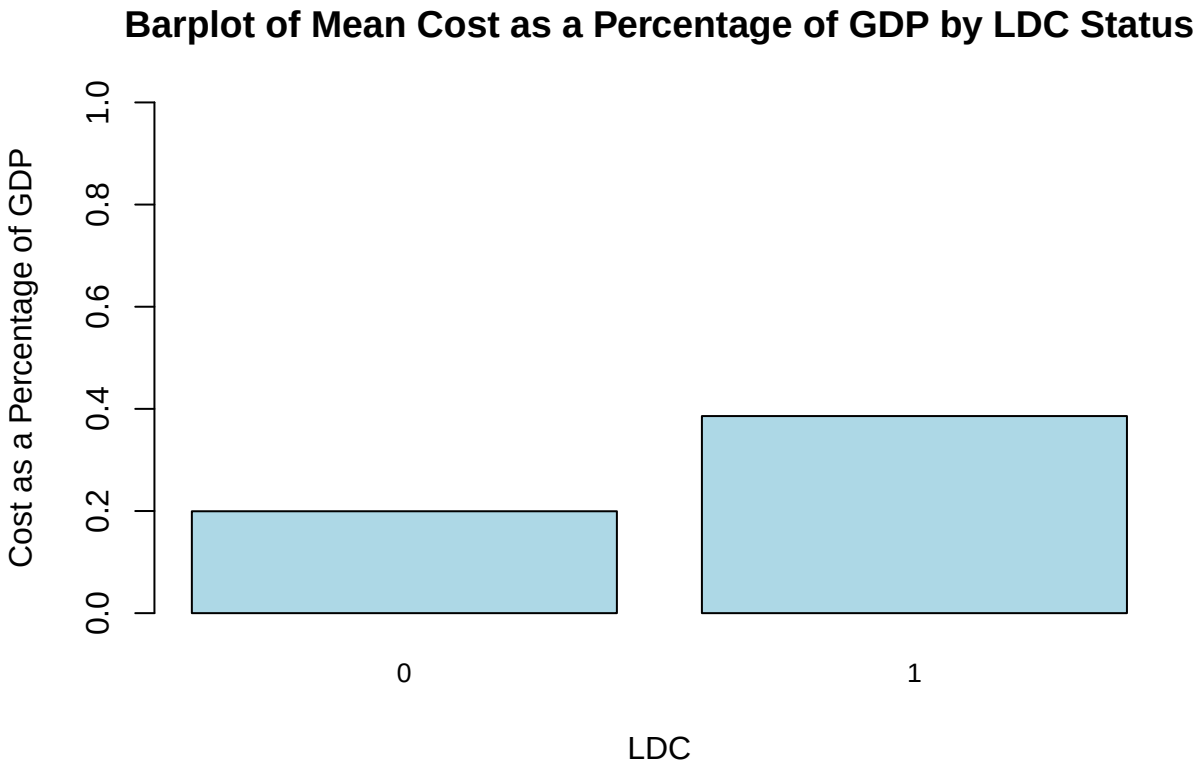

```
boxplot(df_filtered$Default_Percent_Cost ~ df_filtered$Income, cex.axis = 0.7,
↪  ylab = "Cost as a Percentage of GDP", xlab = "Income", main = "Boxplots of
↪  Cost as a Percentage of GDP by Income Level", col = "lightblue")
```



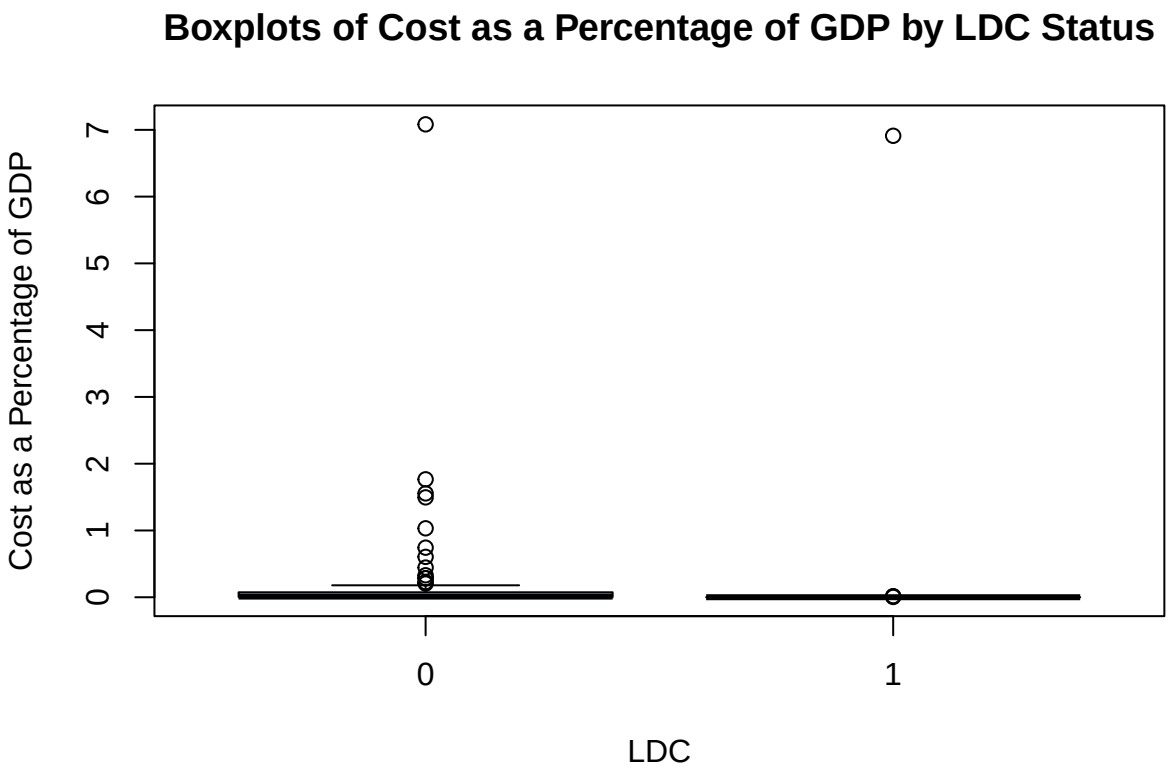

```
by(df_filtered$Default_Percent_Cost, df_filtered$LDC, summary)
```

```
## df_filtered$LDC: 0
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000006 0.001334 0.009919 0.199508 0.072774 7.082377
## ------------------------------------------------------------
## df_filtered$LDC: 1
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000002 0.000028 0.000125 0.385901 0.000491 6.911363
```

```
barplot(by(df_filtered$Default_Percent_Cost, df_filtered$LDC, mean),cex.names =
↪  0.83, ylim = c(0,1), main = "Barplot of Mean Cost as a Percentage of GDP by
↪  LDC Status", xlab = "LDC",ylab = "Cost as a Percentage of GDP", col =
↪  "lightblue")
```




```
boxplot(df_filtered$Default_Percent_Cost ~ df_filtered$LDC,  ylab = "Cost as a
↪  Percentage of GDP", xlab = "LDC", main = "Boxplots of Cost as a Percentage of
↪  GDP by LDC Status", col = "lightblue")
```

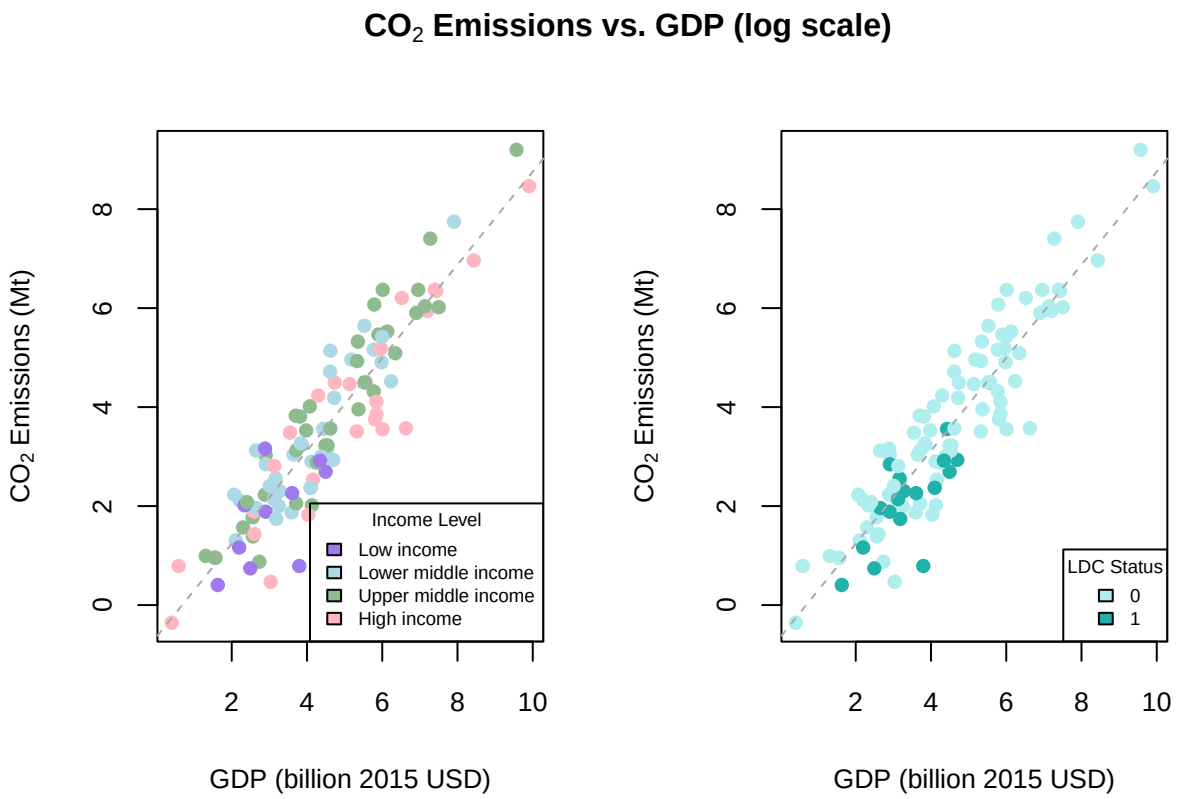


Scatterplots

Emissions vs. GDP

```
par(mfrow = c(1, 2))

cols = c("mediumpurple2","lightblue","darkseagreen","lightpink")

plot(log(CO2_Emissions) ~ log(GDP),  data=df_filtered,xlab = "GDP (billion 2015
↪  USD)", ylab = expression("CO"[2] * " Emissions (Mt)"), col =
↪  cols[df_filtered$Income], pch=16, cex.lab = 0.8, cex.axis = 0.8, cex.main =
↪  1)

legend("bottomright", legend=levels(df_filtered$Income), fill=cols, title="Income
↪  Level", cex = 0.6)

abline(lm(log(df_filtered$CO2_Emissions)~log(df_filtered$GDP)), lty="dashed",
↪  col="darkgray")
```

```r
cols = c("paleturquoise2","lightseagreen")

plot(log(CO2_Emissions) ~ log(GDP),  data=df_filtered,xlab = "GDP (billion 2015
↪  USD)", ylab = expression("CO"[2] * " Emissions (Mt)"), col =
↪  cols[as.factor(df_filtered$LDC)], pch=16, cex.lab = 0.8, cex.axis = 0.8,
↪  cex.main = 1)

legend("bottomright", legend=levels(as.factor(df_filtered$LDC)), fill=cols,
↪  title="LDC Status", cex = 0.6)
abline(lm(log(df_filtered$CO2_Emissions) ~ log(df_filtered$GDP)), lty="dashed",
↪  col="darkgray")


mtext(expression(~bold("CO"[2] * " Emissions vs. GDP (log scale)")),
↪
      side = 3,
      line = - 2,
      outer = TRUE,
      font=2)
```



Cost vs. Statistical Capability

```
summary(lm(log(CO2_Emissions) ~ log(GDP), data=df_filtered))
```

```
##
## Call:
## lm(formula = log(CO2_Emissions) ~ log(GDP), data = df_filtered)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1362 -0.4683  0.0093  0.4828  1.4358
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.64252    0.18627  -3.449 0.000807 ***
## log(GDP)     0.93984    0.03892  24.149  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7393 on 106 degrees of freedom
## Multiple R-squared:  0.8462, Adjusted R-squared:  0.8447
## F-statistic: 583.2 on 1 and 106 DF,  p-value: < 2.2e-16
```

```
cor(log(df_filtered$CO2_Emissions), log(df_filtered$GDP))
```

```
## [1] 0.9198871
```

```
par(mfrow = c(1, 2))


cols = c("mediumpurple2","lightblue","darkseagreen","lightpink")

plot(log(Default_Percent_Cost)~log(Statistical_Capability), data=df_filtered, col
↪  = cols[as.factor(df_filtered$Income)], pch=16, cex.lab = 0.8, cex.axis = 0.8,
↪  cex.main = 0.8, xlab = "Statistical Capability", ylab = "Cost as a Percentage
↪  of GDP")

legend("topleft", legend=levels(df_filtered$Income), fill=cols, title="Income
↪  Level", cex = 0.6)



cols = c("paleturquoise2","lightseagreen")
```

```
plot(log(Default_Percent_Cost)~log(Statistical_Capability), data=df_filtered, col
↪  = cols[as.factor(df_filtered$LDC)], pch=16, cex.lab = 0.8, cex.axis = 0.8,
↪  cex.main = 1, xlab = "Statistical Capability", ylab = "Cost as a Percentage
↪  of GDP")

legend("topleft", legend=levels(as.factor(df_filtered$LDC)), fill=cols,
↪  title="LDC Status", cex = 0.6)



mtext("Cost as a Percentage of GDP vs. Statistical Capability (log scale)",
↪
      side = 3,
      line = - 2,
      outer = TRUE,
      font=2)
```