Initial models for time series

#Read Data and Basic Plots

```
#Load packages
library(readr)
library(fpp3) #this imports a few things including dplyr, tidyr,ggplot2, and more
```

```
## -- Attaching packages -------------------------------------------- fpp3 0.4.0 --

## v tibble      3.1.2      v tsibble     1.0.1
## v dplyr       1.0.7      v tsibbledata 0.3.0
## v tidyr       1.1.3      v feasts      0.2.2
## v lubridate   1.7.10     v fable       0.3.1
## v ggplot2     3.3.4

## -- Conflicts -------------------------------------------------- fpp3_conflicts --
## x lubridate::date()    masks base::date()
## x dplyr::filter()      masks stats::filter()
## x tsibble::intersect() masks base::intersect()
## x tsibble::interval()  masks lubridate::interval()
## x dplyr::lag()         masks stats::lag()
## x tsibble::setdiff()   masks base::setdiff()
## x tsibble::union()     masks base::union()
```

```
library(tsibble)
library(forecast)
```

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

```
#read in cvs as dataframe and convert to time series
#key = c(Mode,ORegionDAT, DRegionDAT) since these 4 categories each have one observation at each time
df <- readr::read_csv(file = "data_shipping_and_weather_joined_cleaned.csv") %>% mutate(yw = yearweek(y
```

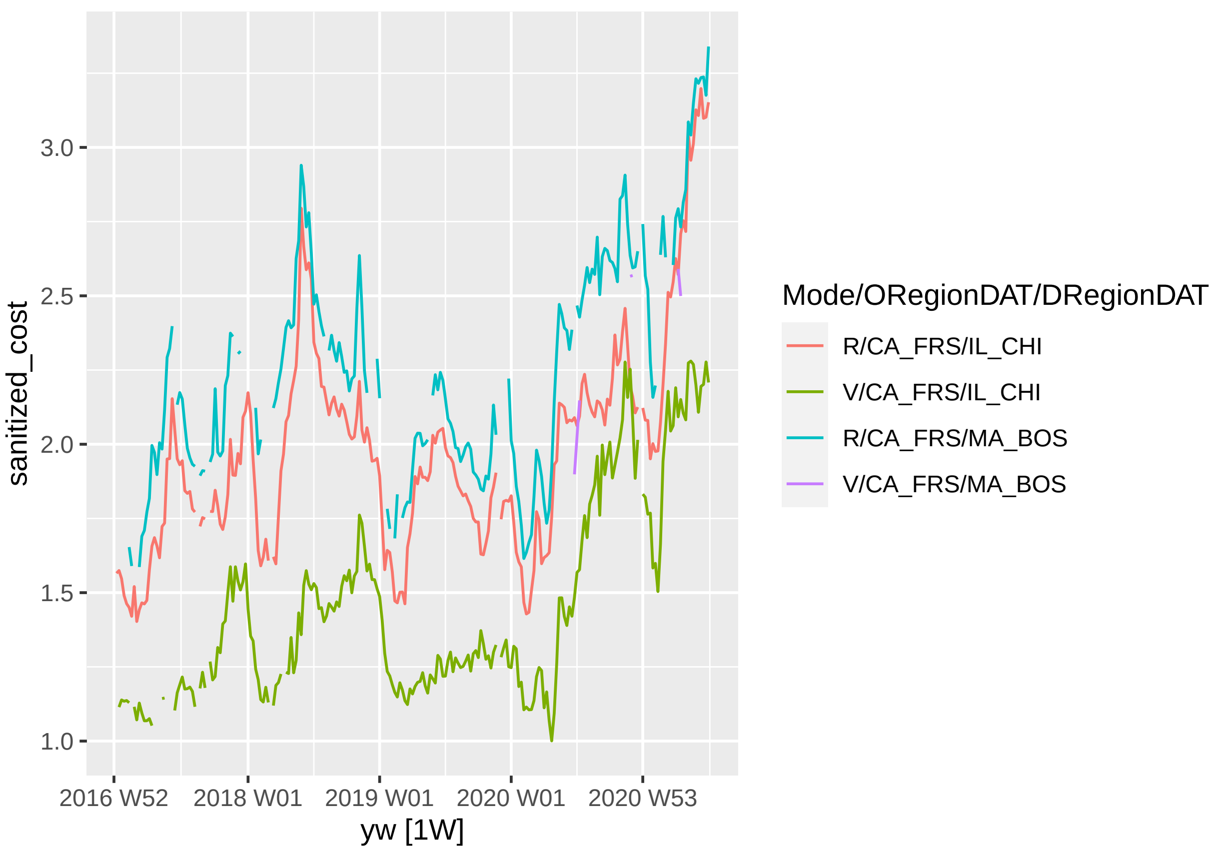
```
## Warning: Missing column names filled in: 'X1' [1]

##
## -- Column specification --------------------------------------------------------
## cols(
##   X1 = col_double(),
##   Mode = col_character(),
##   ORegionDAT = col_character(),
##   DRegionDAT = col_character(),
##   yw = col_character(),
##   sanitized_cost = col_double(),
##   prcp = col_double(),
##   tavg = col_double(),
##   tmax = col_double(),
##   tmin = col_double()
## )
```

```r
head(df)
```

```
## # A tsibble: 6 x 10 [1W]
## # Key:       Mode, ORegionDAT, DRegionDAT [1]
##      X1 Mode  ORegionDAT DRegionDAT       yw sanitized_cost   prcp  tavg  tmax
##   <dbl> <chr> <chr>      <chr>          <week>          <dbl>  <dbl> <dbl> <dbl>
## 1     1 R     CA_FRS     IL_CHI     2017 W01            1.57 0.309   49.4  56.9
## 2     2 R     CA_FRS     IL_CHI     2017 W02            1.57 0.357   49.9  53.1
## 3     3 R     CA_FRS     IL_CHI     2017 W03            1.55 0.238   47.7  54.9
## 4     4 R     CA_FRS     IL_CHI     2017 W04            1.49 0.0781  44.7  55.9
## 5     5 R     CA_FRS     IL_CHI     2017 W05            1.46 0.115   53.4  63.9
## 6     6 R     CA_FRS     IL_CHI     2017 W06            1.45 0.176   57    61.9
## # ... with 1 more variable: tmin <dbl>
```

```r
autoplot(df, sanitized_cost)
```

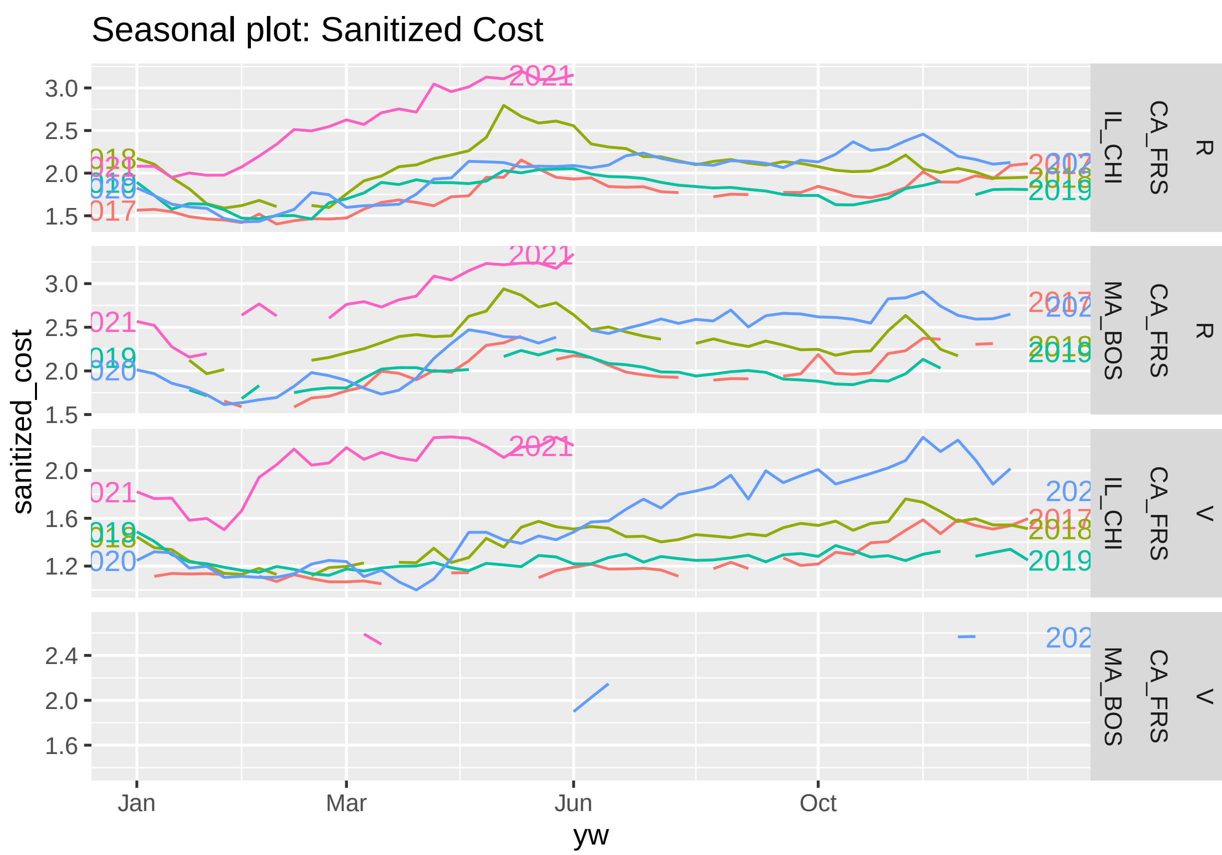
```
## Warning: Removed 41 row(s) containing missing values (geom_path).
```



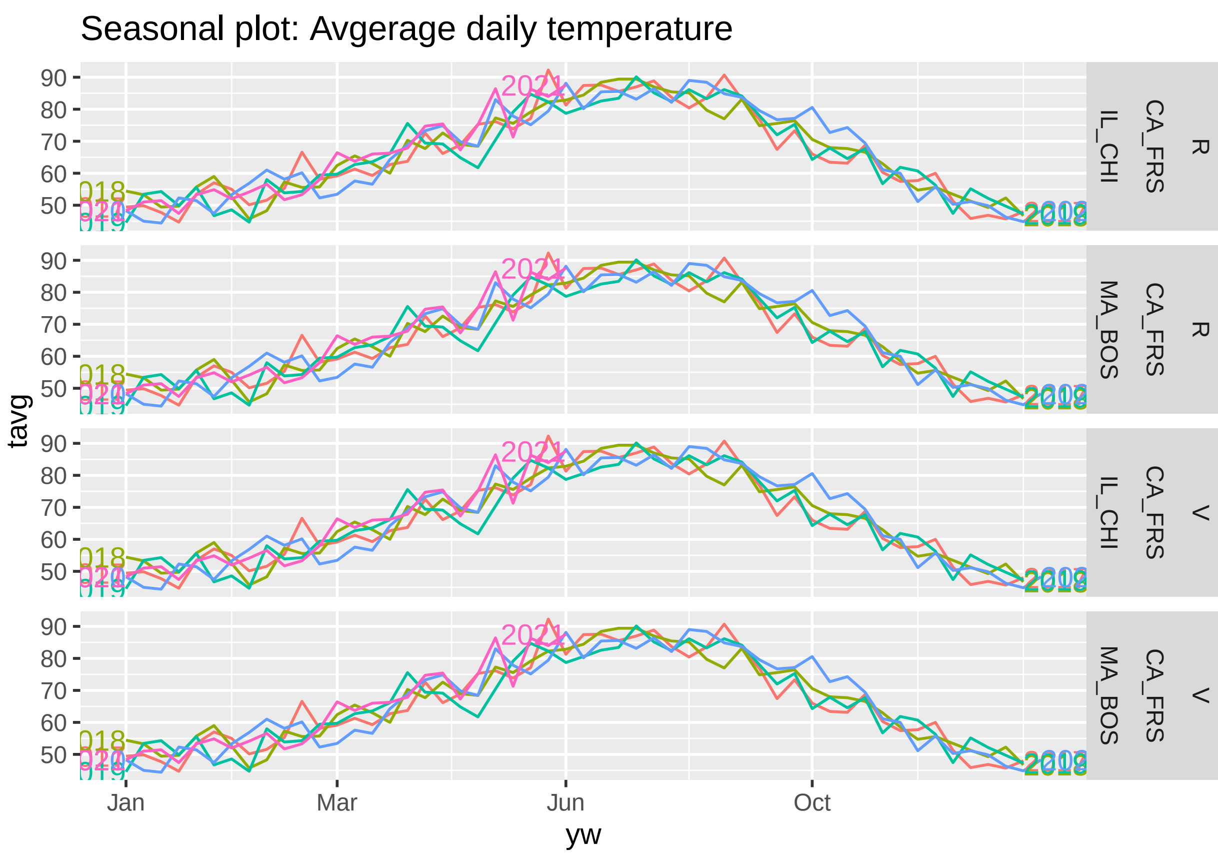

```r
#seasonal plot for cost
gg_season(df, y = sanitized_cost, labels = "both") +
  labs(title = "Seasonal plot: Sanitized Cost")
```

```
## Warning: Removed 39 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 12 rows containing missing values (geom_text).
```




```r
# seasonal plot for temperature
gg_season(df, y = tavg, labels = "both") +
  labs(title = "Seasonal plot: Avgerage daily temperature")
```



#Looking at Basic Models

Let's follow this article with our data set

```
is.ts(df)
```

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

##Avg, Naive, SNaive (seasonal naive(!))

mean = the forecasts of all future values are equal to the average (or "mean") of the historical data

naive = the forecasts for every horizon correspond to the last observed value

Seasonal Naive = we set each forecast to be equal to the last observed value from the same season of the year

```
snaive(
  df,
  h = 2 * frequency(x),
  level = c(80, 95),
  fan = FALSE,
  lambda = NULL,
  biasadj = FALSE,
  ...,
  x = y
)
```

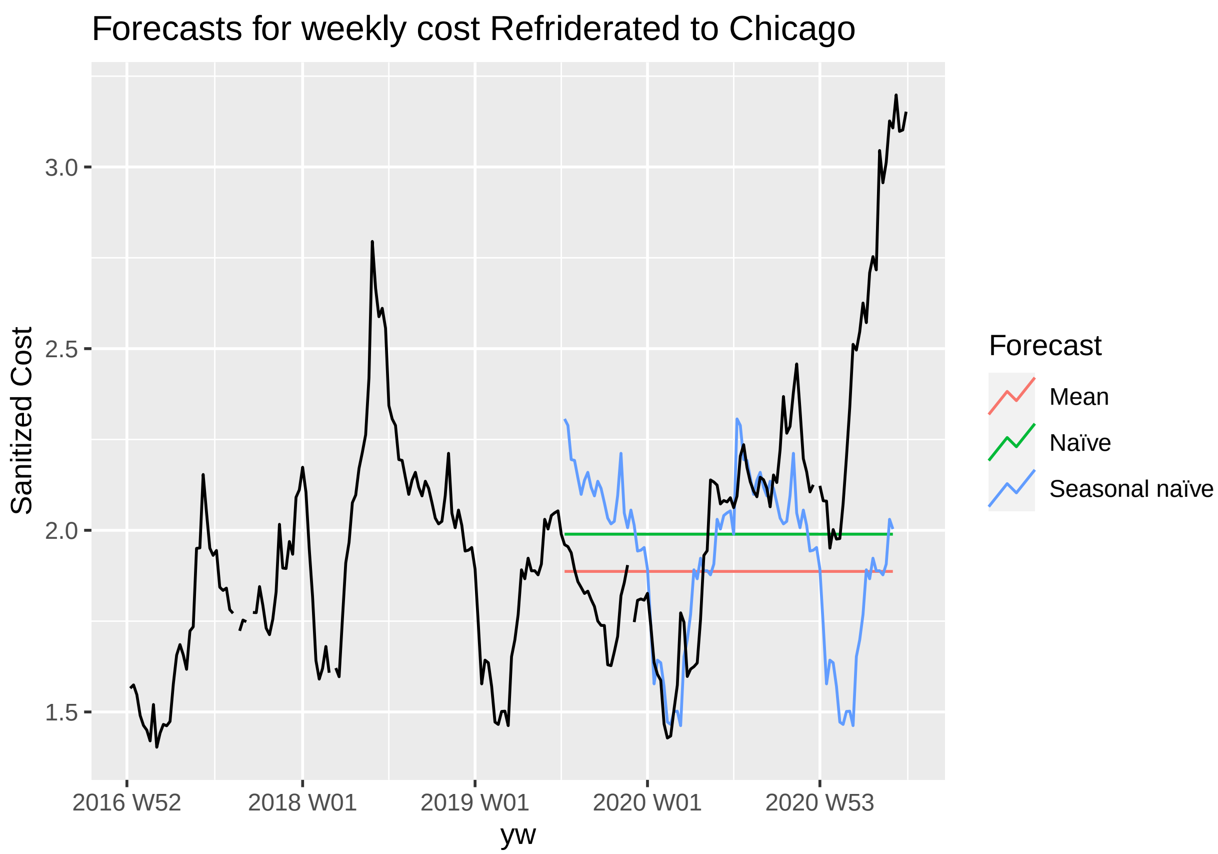
```r
#Look at just refridgerated trucks with dest IL_CHI
df_R_IL <- df %>%
  filter(df$Mode == "R", df$DRegionDAT == "IL_CHI")
```

```r
# Set training data from 2017 W01 to 2019 W27
train_R_IL <- df_R_IL %>%
  filter_index("2017 W01" ~ "2019 W27")
```

```r
# Fit the models
cost_fit <- train_R_IL %>%
  model(
    Mean = MEAN(sanitized_cost),
    `Naïve` = NAIVE(sanitized_cost),
    `Seasonal naïve` = SNAIVE(sanitized_cost)
  )
```

```r
# Generate forecasts for 54 weeks
cost_fc <- cost_fit %>% forecast(h = 100)
```

```r
# Plot forecasts against actual values
cost_fc %>%
  autoplot(df_R_IL, level = NULL) +
  labs(
    y = "Sanitized Cost",
    title = "Forecasts for weekly cost Refriderated to Chicago"
  ) +
  guides(colour = guide_legend(title = "Forecast"))
```




```
#mean =  the forecasts of all future values are equal to the average (or "mean") of the historical data

#naive = the forecasts for every horizon correspond to the last observed value

#Seasonal Naive = we set each forecast to be equal to the last observed value from the same season of t
```

```
# Set training data from 2017 W01 to 2019 W27
train <- df %>%
  filter_index("2017 W01" ~ "2019 W27")
```

```
# Fit the models
cost_fit <- train %>%
  model(
    Mean = MEAN(sanitized_cost),
    `Naïve` = NAIVE(sanitized_cost),
    `Seasonal naïve` = SNAIVE(sanitized_cost)
  )
```

```
# Generate forecasts for 54 weeks
cost_fc <- cost_fit %>% forecast(h = 100)
```

```
# Plot forecasts against actual values
cost_fc %>%
  autoplot(df,level = NULL) +
  labs(
```

```
    y = "Sanitized Cost",
    title = "Forecasts for weekly cost"
  )+
  guides(colour = guide_legend(title = "Forecast"))
```

```
## Warning: Removed 126 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 41 row(s) containing missing values (geom_path).
```