Assignment 3

Shaun Harrington

Introduction

I'll be using data from the EIA to predict wind generation in Texas using prices. Theory would indicate that higher prices will incentivize more electricity production. Because wind turbines are relatively cheap to build and Texas' unique electricity market enables a wide number of suppliers to join the market, it would make sense for prices to be a predictor of wind turbine generation.

One source of volatility that could diminish this relation would be the variation of wind. I was unable to find wind data for Texas, but an alternative data series that could fit is capacity factor. This is defined as $\frac{AverageMWh}{TotalCapacity}$. When prices are positive (and above marginal cost $\approx$ 0), wind turbines will generate as long as there is wind. Thus while a drop in capacity factor could be due to negative pricing, it is much more likely to be from a decrease in wind.

```
knitr::opts_chunk$set(
    echo = TRUE,
    message = FALSE,
    warning = FALSE
)

library(tidyverse)
library(fpp3)
library(fredr)
library(scales)
library(jsonlite)
library(zoo)

theme_set(theme_bw())

if(!str_detect(basename(getwd()), "Time Series") & str_detect(dirname(getwd()), "Time Seri
  repeat{
    setwd("../")
```

```
    if(str_detect(basename(getwd()), "Time Series")){
      break
    }
  }
}

if(basename(getwd()) != "Assignment 3") setwd(file.path(getwd(), "Assignments", "Assignmen
```

Get Data

```
eia.key <- Sys.getenv("EIA_API_KEY")


 fn_query_eia <- function( api_url = NULL,
   the_series_id, the_source = "steo", the_frequency = "monthly", the_facet = "seriesId",
   the_offset = 0, the_length = 5000, the_eia_key = eia.key){

   if(is.null(api_url)){
     the_url = "https://api.eia.gov/v2/"

     # Query must be no more than 5,000
     if(the_length > 5000) break

     get_call <- paste0(the_url, the_source, "/data/?", paste(
       paste0("frequency=", the_frequency),
       "data[0]=value",
       paste0("facets[", the_facet, "][]=", the_series_id),
       "sort[0][column]=period",
       "sort[0][direction]=desc",
       paste0("offset=", the_offset),
       paste0("length=", the_length),
       sep = "&"
     ))

     eia_list <- fromJSON(str_c(get_call, "&api_key=", the_eia_key))

     eia_data <- eia_list$response$data

     eia_data %>%
       as_tibble() %>%
```

```
        return()
      }

    else{

      eia_list <- fromJSON(str_c(api_url, "&api_key=", the_eia_key))

      eia_data <- eia_list$response$data

      eia_data %>%
        as_tibble() %>%
        return()

    }
  }
```

```
url.wind <- "https://api.eia.gov/v2/electricity/electric-power-operational-data/data/?freq

data.wind.gen <- fn_query_eia(api_url = url.wind)

data.price <- fn_query_eia(the_series_id = "ELWHU_TX", the_source = "steo", the_facet = "s

data.wind.cf <- fn_query_eia(api_url = "https://api.eia.gov/v2/total-energy/data/?frequenc
```

The data is split into two datasets, a training and testing dataset. The testing set are the most recent 12 months, while the training set are the 48 months preceding that.

```
data <- left_join(
  x = data.price %>%
    select(period, value) %>%
    rename(date = period, price = value) %>%
    mutate(date = ym(date) %>% yearmonth()),
  y = data.wind.gen %>%
    select(period, generation) %>%
    rename(date = period) %>%
    mutate(date = ym(date) %>% yearmonth()),
  by = "date"
) %>%
  left_join(
    y = data.wind.cf %>%
    select(period, value) %>%
```

```
    rename(date = period, capacity.factor = value) %>%
    mutate(date = ym(date) %>% yearmonth()),
  by = "date"
  ) %>%
  arrange(date) %>%
  mutate(
    price_lag12 = lag(price, n = 12),
    price_lag18 = lag(price, n = 18),
    price_lag24 = lag(price, n = 24),
    price_lag30 = lag(price, n = 30),
    price_lag36 = lag(price, n = 36),
    price_lag48 = lag(price, n = 48),
    price_lag60 = lag(price, n = 60),
    price_lag72 = lag(price, n = 72),
    price_lag12_ma = rollmean(price, k = 12, fill = NA, align = "right") %>% lag(n = 12),
    price_lag24_ma = rollmean(price, k = 24, fill = NA, align = "right") %>% lag(n = 24),
    price_lag36_ma = rollmean(price, k = 36, fill = NA, align = "right") %>% lag(n = 36),
    price_lag48_ma = rollmean(price, k = 48, fill = NA, align = "right") %>% lag(n = 48)
  ) %>%
  drop_na()

avg.capacity <- data %>%
  mutate(month = month(date)) %>%
  group_by(month) %>%
  summarize(capacity.factor = mean(capacity.factor)) %>%
  ungroup()

test <- data %>%
  select(-capacity.factor) %>%
  slice_max(order_by = date, n = 12) %>%
  mutate(date = yearmonth(date), month = month(date)) %>%
  left_join(avg.capacity, by = "month") %>%
  select(-month) %>%
  tsibble()

train <- data %>%
  slice_max(order_by = date, n = 12*10) %>%
  mutate(date = yearmonth(date)) %>%
  anti_join(y = test, by = "date") %>%
  tsibble()
```

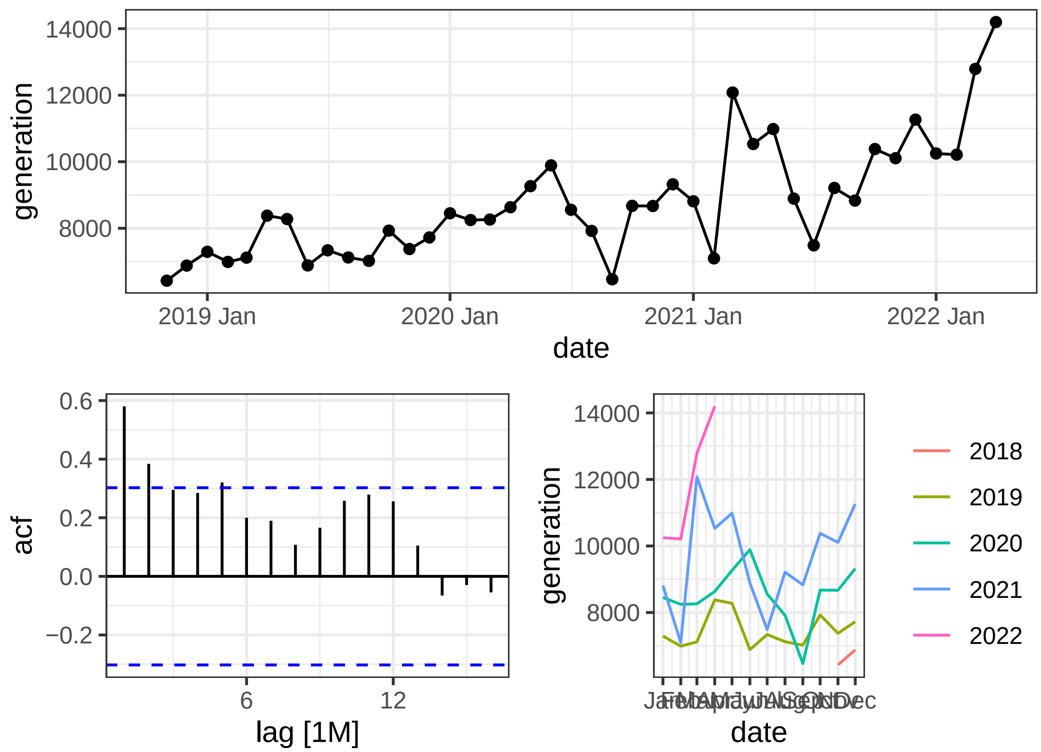
```r
data <- data %>%
  tsibble(index = date)
```

Preliminary Analysis

Data Exploration

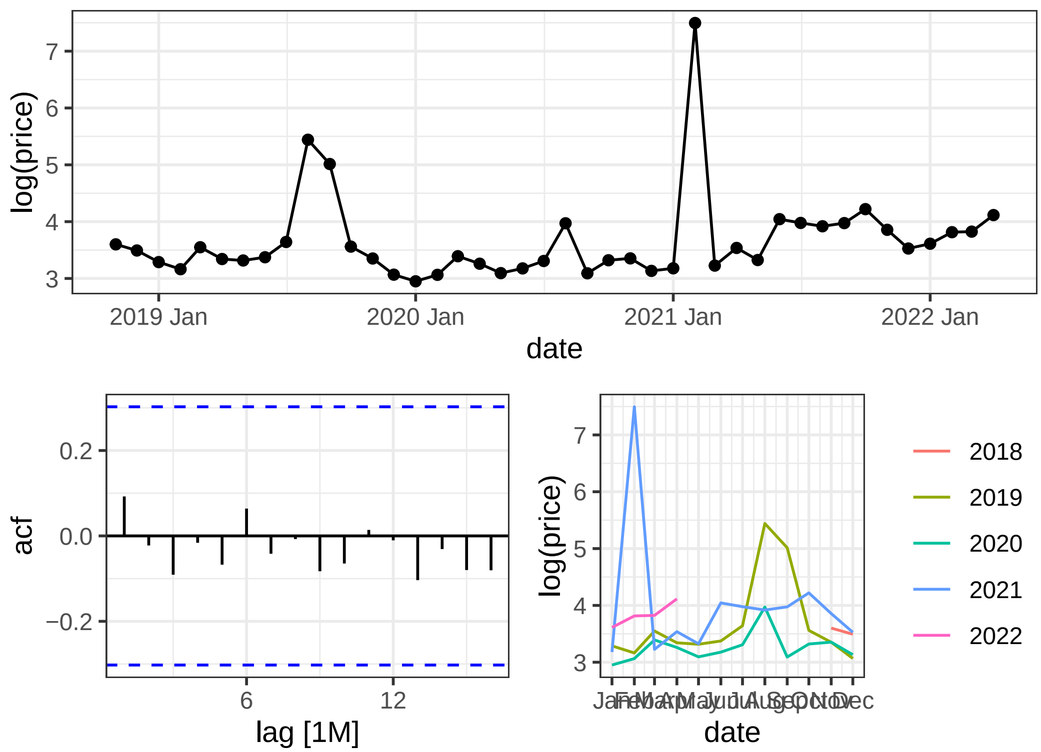
Monthly Wind Generation in Texas

```r
train %>%
  gg_tsdisplay(generation)
```



Monthly Average Wholesale Electricity Price in Texas (log scale)

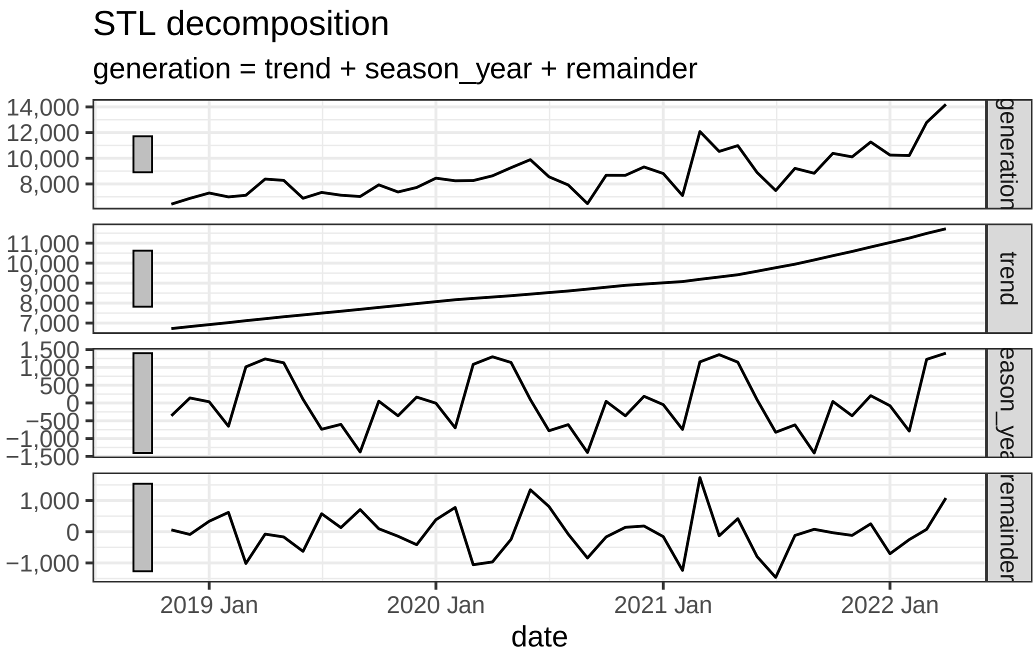
```r
train %>%
  gg_tsdisplay(log(price))
```



Decomposition

The STL Decomposition breaks out the data into the trend, season, and remainder components. The seasonality will largely be driven by wind though I would suspect repairs in the shoulder months playing a role. The trend will show the general increase in supply over time. The remainder will contain wind deviations from normal, unexpected turbine outages, and other unforeseeable factors.

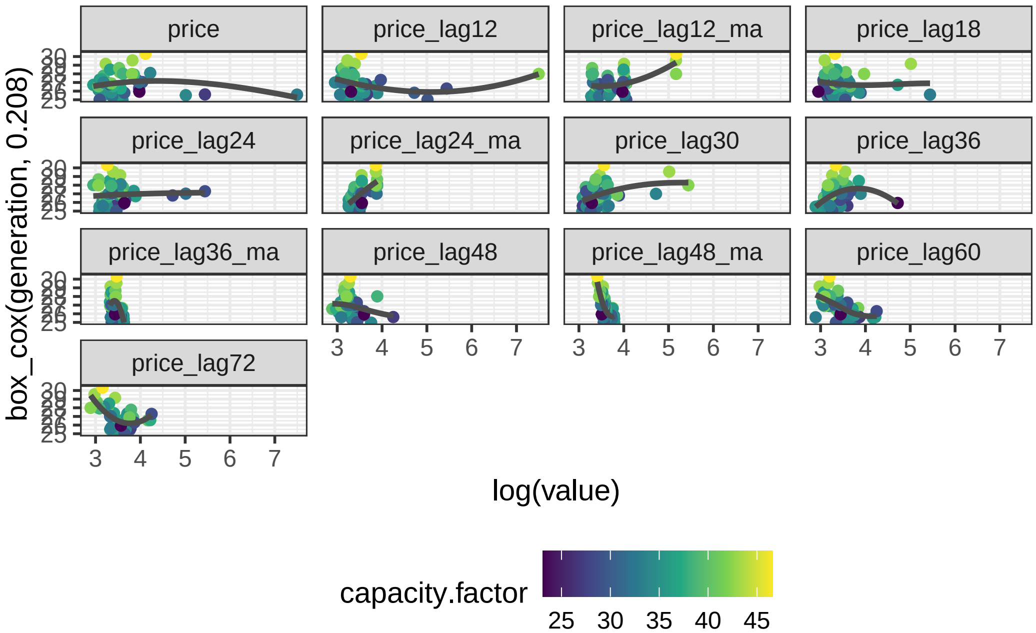
```
train %>%
  model(STL(generation)) %>%
  components() %>%
  autoplot() +
  scale_y_continuous(labels = label_comma())
```



Correlations

The 30-month lag appears to be the most correlated.

```
train %>%
  pivot_longer(-c(date, generation, capacity.factor), names_to = "lag", values_to = "value
  ggplot(aes(
    x = log(value),
    # x = value,
    y = box_cox(generation, .208),
    # y = generation,
    color = capacity.factor
  )) +
  geom_point() +
  geom_smooth(se = F, scales = "free", span = 2, color = "gray30") +
  facet_wrap(lag ~ .) +
  scale_color_viridis_c() +
  theme(
    legend.position = "bottom"
  )
```



Modeling

Estimation

Four models will be estimated: an ETS, an auto-ARIMA, and an ARIMAX using gas and diesel prices as regressors. The fourth model will be a simple average of the others.

```
(fit <- train %>%
  model(
    "ets" = ETS(box_cox(generation, .208)),
    "arima" = ARIMA(box_cox(generation, .208) ~ log(price_lag12_ma) + log(price_lag24_ma)
    "nn2" = NNETAR(
      box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
        # price_lag18 + price_lag24 +
        # AR(P = 1),
      n_nodes = 2, scale_inputs = TRUE
    ),
    "nn5" = NNETAR(
      box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
        price_lag30, # + price_lag24, #+
        # AR(P = 1),
      n_nodes = 5, scale_inputs = TRUE
```

```r
    )
  ) %>%
   mutate(ensemble = (ets + arima + nn2) / 3))
```

```
# A mable: 1 x 5
           ets                                      arima                nn2
       <model>                                    <model>            <model>
1 <ETS(M,A,N)> <LM w/ ARIMA(0,0,0)(0,1,0)[12] errors> <NNAR(1,1,2)[12]>
# i 2 more variables: nn5 <model>, ensemble <model>
```

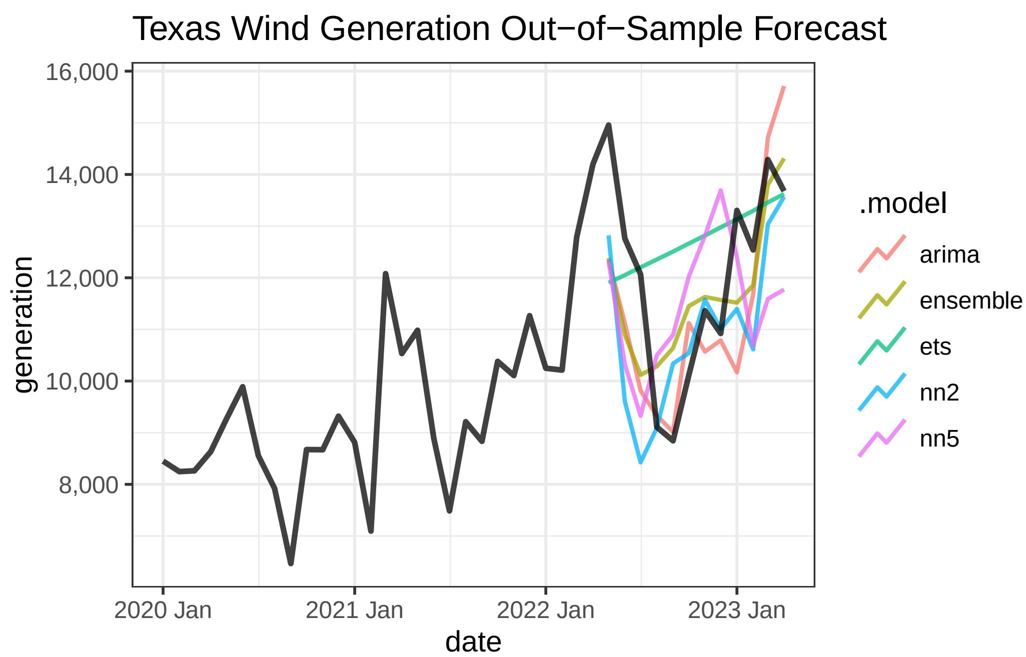
Forecast

The models were trained on data prior to 2022-05-01. The forecast period is the interval 2022-05-01 UTC–2023-04-01 UTC.

For the forecast period, the average capacity factor by month will be used along with lagged transformations of price. Two neural net models are included, one with 2 nodes and the other with 5.

```r
fx <- fit %>%
    forecast(test, times = 50)


fx %>%
  autoplot(
      level = NULL, size = .75, alpha = .75
    ) +
  autolayer(
    data %>% tsibble() %>% filter(year(date)>=2020), generation,
    size = 1, alpha = .75#, linetype = "dashed"
  ) +
  ggtitle("Texas Wind Generation Out-of-Sample Forecast") +
  scale_y_continuous(labels = label_comma()); fx %>%
  accuracy(test, measures = point_accuracy_measures) %>%
  arrange(RMSE)
```




```
# A tibble: 5 x 10
  .model   .type    ME  RMSE   MAE    MPE  MAPE  MASE RMSSE  ACF1
  <chr>    <chr> <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
1 ensemble Test   291. 1446. 1267.  0.910 10.8    NaN   NaN 0.446
2 arima    Test   630. 1612. 1274.  4.61   9.98   NaN   NaN 0.220
3 nn2      Test   993. 1807. 1364.  7.07  11.0    NaN   NaN 0.529
4 ets      Test  -754. 1999. 1556. -8.97  14.5    NaN   NaN 0.478
5 nn5      Test   467. 2144. 2062.  1.53  17.4    NaN   NaN 0.599
```

The ensemble model outperforms the rest, likely a result of the forecasts not being biased one way or the other in comparison to the actual generation. The ARIMA is next and the remaining neural nets and ETS clustered at the bottom.

```
fx %>%
  accuracy(test, measures = list(point_accuracy_measures, distribution_accuracy_measures))
  arrange(RMSE)
```

```
# A tibble: 5 x 12
  .model .type    ME  RMSE   MAE    MPE  MAPE  MASE RMSSE  ACF1 percentile  CRPS
  <chr>  <chr> <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>      <dbl> <dbl>
1 ensem~ Test   291. 1446. 1267.  0.910 10.8    NaN   NaN 0.446      1267. 1267.
```

```
2 arima  Test   630. 1612. 1274.  4.61   9.98   NaN    NaN 0.220      1019. 1011.
3 nn2    Test   993. 1807. 1364.  7.07  11.0    NaN    NaN 0.529      1214. 1210.
4 ets    Test  -754. 1999. 1556. -8.97  14.5    NaN    NaN 0.478      1172. 1162.
5 nn5    Test   467. 2144. 2062.  1.53  17.4    NaN    NaN 0.599      2035. 2034.
```

Cross Validation

One issue with neural networks is selecting the appropriate number of nodes. One way this can be handled is through cross validation. By splitting the dataset into various components and training models on each of these, we can compare how each does over multiple out-of-sample forecasts. We'll split the training set into 9 different parts and train neural nets with 2, 3, 4, 5, 6, 7, and 8 nodes to determine which would perform best on average.

```
train.cv <- train %>%
  stretch_tsibble(.init = 18, .step = 3)

train.cv$.id %>% max()
```

```
[1] 9
```

```
(fit.cv <- train.cv %>%
  model(
    "nn2" = NNETAR(
      box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
        # price_lag18 + price_lag24 +
        # AR(P = 1),
      n_nodes = 2, scale_inputs = TRUE
    ),
     "nn3" = NNETAR(
      box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
        # price_lag18 + price_lag24 +
        # AR(P = 1),
      n_nodes = 3, scale_inputs = TRUE
    ),
     "nn4" = NNETAR(
      box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
        # price_lag18 + price_lag24 +
        # AR(P = 1),
      n_nodes = 4, scale_inputs = TRUE
    ),
     "nn5" = NNETAR(
```

```
        box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
          # price_lag18 + price_lag24 +
          # AR(P = 1),
        n_nodes = 5, scale_inputs = TRUE
      ),
       "nn6" = NNETAR(
        box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
          # price_lag18 + price_lag24 +
          # AR(P = 1),
        n_nodes = 6, scale_inputs = TRUE
      ),
       "nn7" = NNETAR(
        box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
          # price_lag18 + price_lag24 +
          # AR(P = 1),
        n_nodes = 7, scale_inputs = TRUE
      ),
       "nn8" = NNETAR(
        box_cox(generation, .208) ~ log(price_lag24_ma) + log(price_lag36_ma) + (capacity.fa
          # price_lag18 + price_lag24 +
          # AR(P = 1),
        n_nodes = 8, scale_inputs = TRUE
      )
      )
    )
```

```
# A mable: 9 x 8
# Key:     .id [9]
    .id               nn2               nn3               nn4               nn5
  <int>           <model>           <model>           <model>           <model>
1     1 <NNAR(1,1,2)[12]> <NNAR(1,1,3)[12]> <NNAR(1,1,4)[12]> <NNAR(1,1,5)[12]>
2     2 <NNAR(1,1,2)[12]> <NNAR(1,1,3)[12]> <NNAR(1,1,4)[12]> <NNAR(1,1,5)[12]>
3     3 <NNAR(1,1,2)[12]> <NNAR(1,1,3)[12]> <NNAR(1,1,4)[12]> <NNAR(1,1,5)[12]>
4     4 <NNAR(1,1,2)[12]> <NNAR(1,1,3)[12]> <NNAR(1,1,4)[12]> <NNAR(1,1,5)[12]>
5     5 <NNAR(1,1,2)[12]> <NNAR(1,1,3)[12]> <NNAR(1,1,4)[12]> <NNAR(1,1,5)[12]>
6     6 <NNAR(1,1,2)[12]> <NNAR(1,1,3)[12]> <NNAR(1,1,4)[12]> <NNAR(1,1,5)[12]>
7     7 <NNAR(5,1,2)[12]> <NNAR(5,1,3)[12]> <NNAR(5,1,4)[12]> <NNAR(5,1,5)[12]>
8     8 <NNAR(1,1,2)[12]> <NNAR(1,1,3)[12]> <NNAR(1,1,4)[12]> <NNAR(1,1,5)[12]>
9     9 <NNAR(1,1,2)[12]> <NNAR(1,1,3)[12]> <NNAR(1,1,4)[12]> <NNAR(1,1,5)[12]>
# i 3 more variables: nn6 <model>, nn7 <model>, nn8 <model>
```

Cross validation suggests the simple 2-node network has a lower median RMSE than the others.

```
fx.cv <- fit.cv %>%
  forecast(train.cv, times = 10)

fx.cv %>%
  mutate(.model = factor(.model, levels = paste("nn", c(2:8), sep = ""))) %>%
  accuracy(train.cv) %>%
  ggplot(aes(x = RMSE, y = .model)) +
  geom_boxplot()
```