Assignment 2

Shaun Harrington

Setup

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

library(tidyverse)
library(fpp3)
library(fredr)
library(scales)

theme_set(theme_bw())

# if(!str_detect(basename(getwd()), "Time Series") & str_detect(dirname(getwd()), "Time Se
#   repeat{
#     setwd("../")
#     if(str_detect(basename(getwd()), "Time Series")){
#       break
#     }
#   }
# }
#
# if(basename(getwd()) != "Assignment 2") setwd(file.path(getwd(), "Assignments", "Assignm
```

Get Data

Gasoline station sales will be retrieved from the US Census.

```
url <- "https://www.census.gov/retail/marts/www/adv44700.txt"

# gasoline sales
data.gas <- read_table(url, skip = 1, n_max = 31) %>%
  rename_with(~c("year", as.character(1:12))) %>%
  pivot_longer(-year, names_to = "month", values_to = "sales.gas") %>%
  mutate(date = ymd(paste(year, month, "1", sep = "-"))) %>%
  select(date, sales.gas)
```

Gasoline and diesel prices will be retrieved via the Federal Reserve Economics Database.

```
fred.data <- c(
  "GASREGW", #"CUUR0000SETB01", "TRFVOLUSM227NFWA",
  "GASDESM"
) %>%
  lapply(\(x){
    fredr(x, frequency = "m")
  }) %>%
  reduce(., bind_rows) %>%
  select(date, series_id, value) %>%
  pivot_wider(names_from = series_id, values_from = value) %>%
  rename(
    gas.price = "GASREGW",
    # cpi.gas = "CUUR0000SETB01",
    # miles.driven = "TRFVOLUSM227NFWA",
    diesel.price = "GASDESM"
  )
```

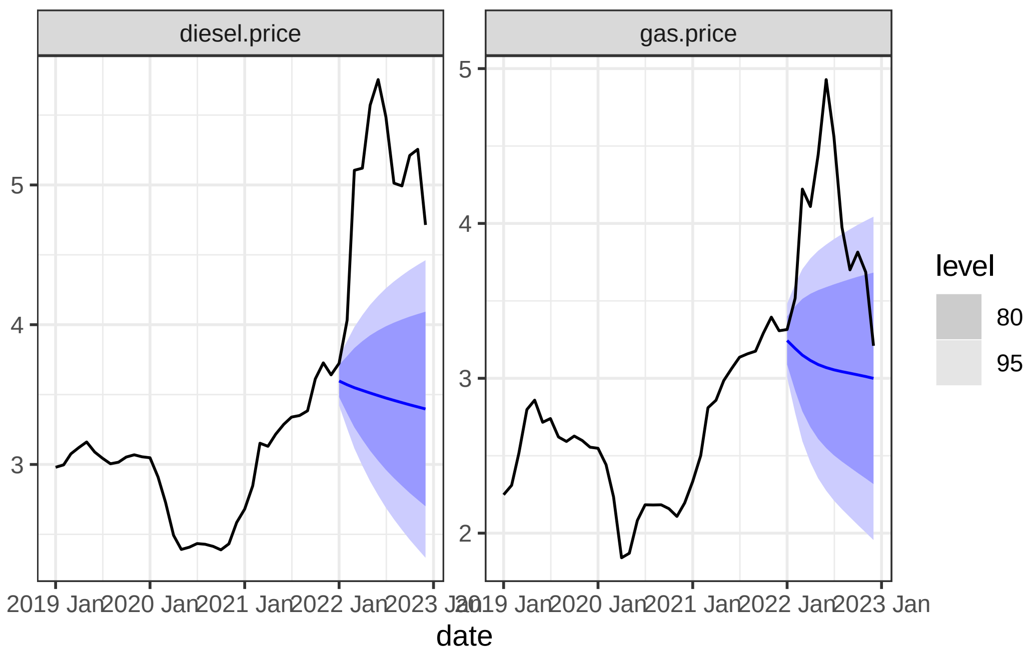
The data is split into two datasets, a training and testing dataset. The testing set are the most recent 12 months, while the training set are the 48 months preceding that.

```
data <- left_join(
  x = data.gas,
  y = fred.data,
  by = "date"
)

test <- data %>%
  slice_max(order_by = date, n = 12) %>%
  mutate(date = yearmonth(date)) %>%
  tsibble()
```

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

However, a true forecasts will not have gas and diesel prices available so we must adjust the test set for that. These prices will be estimated using a vector autoregression (VAR).

```
test.fx <- train %>%
  model(VAR(vars(gas.price, diesel.price))) %>%
  forecast(h = 12)

test.fx %>%
  autoplot(data %>% filter(year(date) > 2018) %>% tsibble())
```



Since gas and diesel prices are very difficult to forecast, we will consider a scenario forecasting approach with a low price (20th percentile), medium price (point estimate), and high price (80th percentile) scenarios. Despite actual prices being well above even the 95th percentile, the 80th percentile will be used and my knowledge of gas prices over the last 12 months shouldn't be a cause of bias.

```r
test_all.scenarios <- test.fx %>%
  hilo(level = c(80)) %>%
  mutate(
    gas.price_med = .mean[,"gas.price"],
    gas.price_low = `80%`$gas.price$lower,
    gas.price_high = `80%`$gas.price$upper,

    diesel.price_med = .mean[,"diesel.price"],
    diesel.price_low = `80%`$diesel.price$lower,
    diesel.price_high = `80%`$diesel.price$upper
  ) %>%
  select(
    date, gas.price_med, gas.price_high, gas.price_low,
    diesel.price_med, diesel.price_high, diesel.price_low
  ) %>%
  left_join(x = test, by = "date")

test.low <- test_all.scenarios %>%
  select(date, sales.gas, contains("low")) %>%
  rename_with(~str_replace_all(.x, "_low", ""))

test.med <- test_all.scenarios %>%
  select(date, sales.gas, contains("med")) %>%
  rename_with(~str_replace_all(.x, "_med", ""))

test.high <- test_all.scenarios %>%
  select(date, sales.gas, contains("high")) %>%
  rename_with(~str_replace_all(.x, "_high", ""))
```

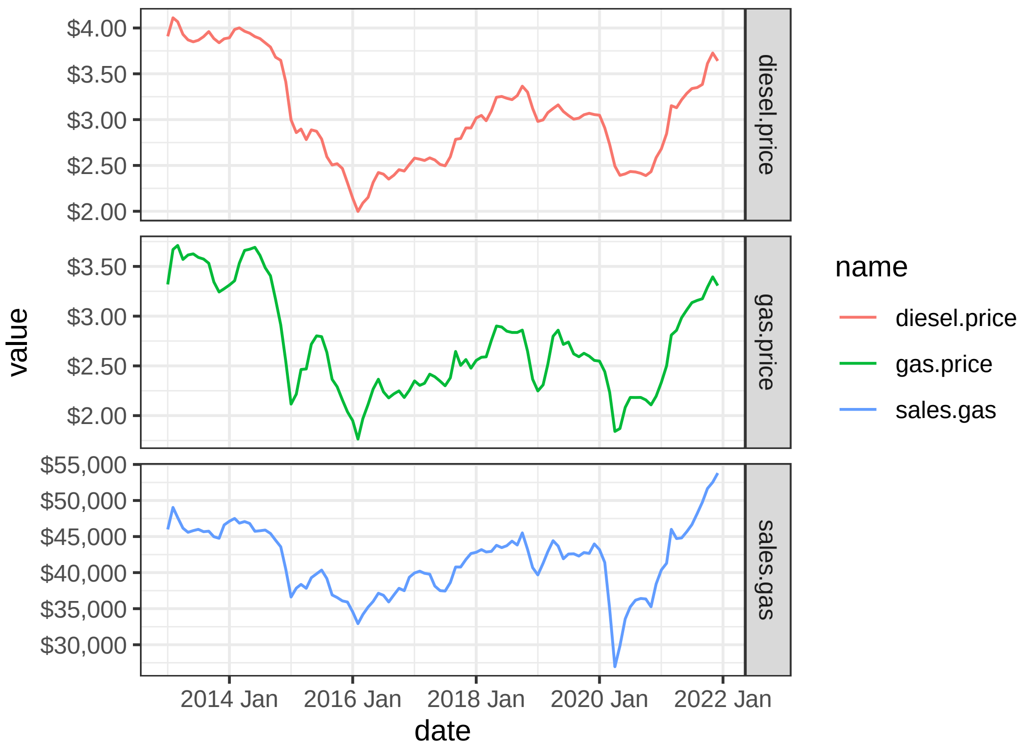
Preliminary Analysis

Data Exploration

Gasoline Station Sales Plots

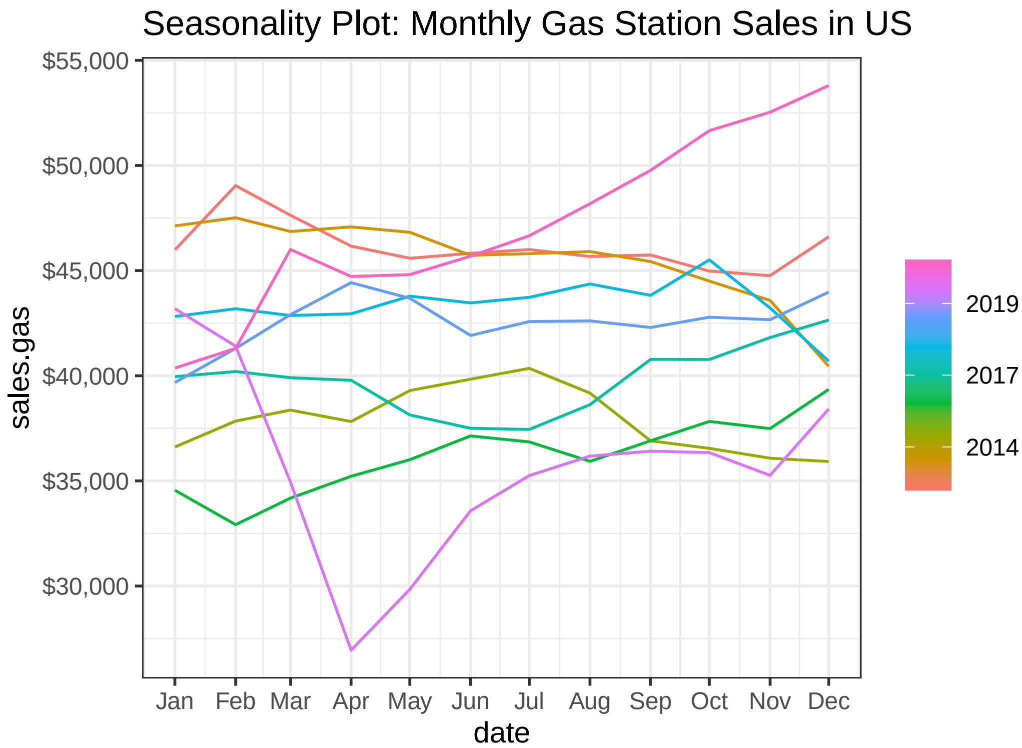
Gasoline station sales have been very volatile during the training set period, especially so since the COVID-19 pandemic started. The sales have been highly correlated with gas and diesel prices.

```
train %>%
  # select(date, sales.gas, gas.price, miles.driven) %>%
  pivot_longer(-date) %>%
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  facet_grid(name ~ ., scales = "free") +
  scale_y_continuous(labels = label_dollar())
```



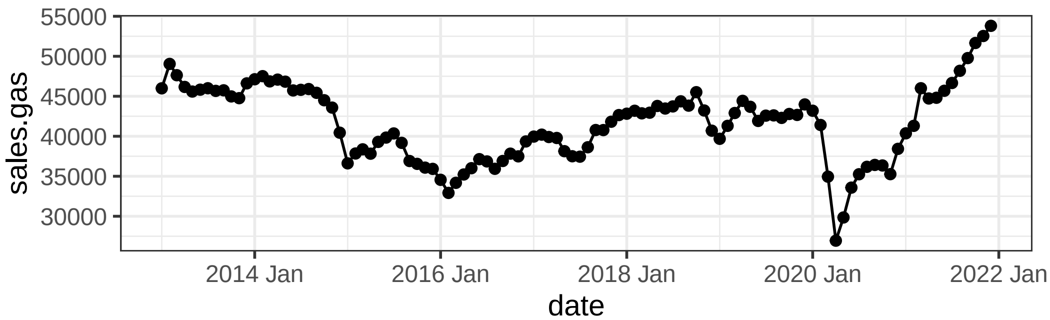
Surprisingly, little if any seasonality exist in the series.

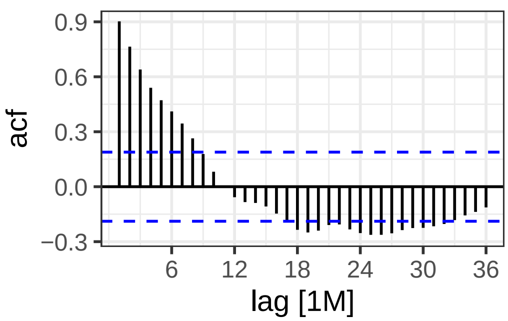
```
train %>%
  gg_season(sales.gas) +
  ggtitle("Seasonality Plot: Monthly Gas Station Sales in US") +
  scale_y_continuous(labels = label_dollar())
```

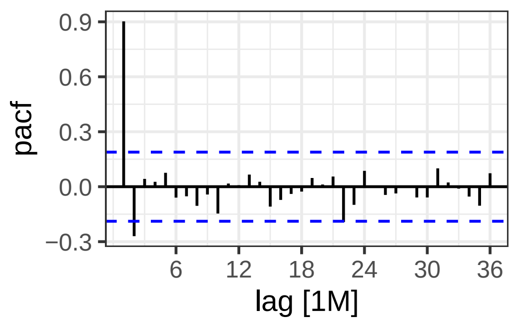


The ACF and PACF also back this up with a low correlation against the 12-month lag.

```
train %>%
  gg_tsdisplay(sales.gas, plot_type = "partial", lag_max = 36)
```



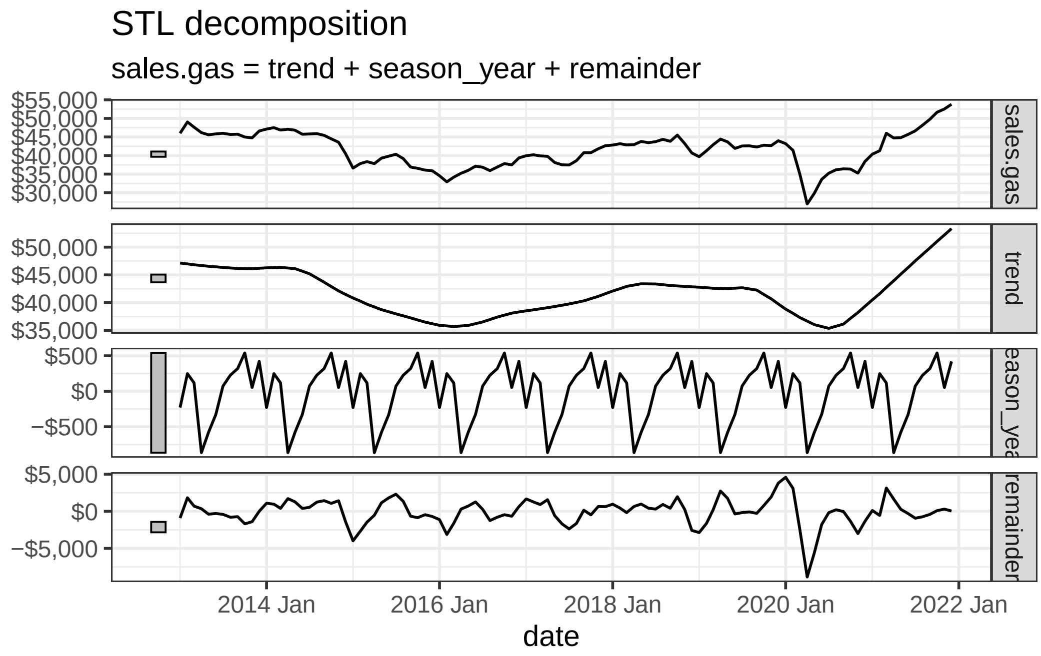




Decomposition

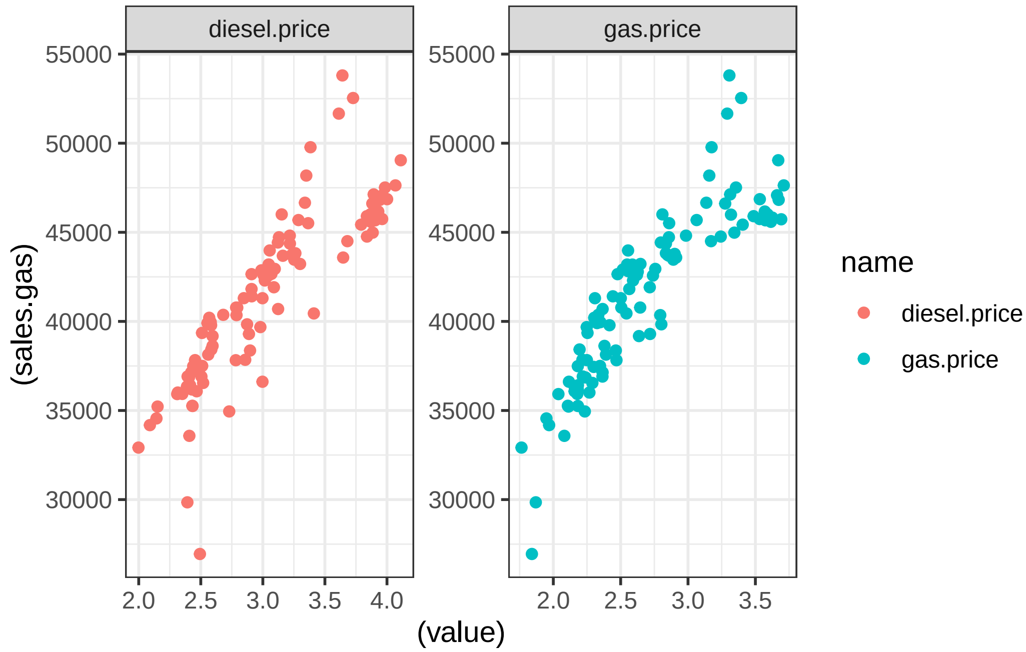
The STL Decomposition does appear to find some seasonal component, though it may not be statistically significant. This is especially so since it appears the large dip in March 2020, a likely result of COVID-19 shutdowns, has affected the seasonal component. The seasonal window was forced to be periodic, resulting in an unchanging seasonal pattern since only 4 years of data exist in the training set.

```
train %>%
  model(STL(sales.gas ~ season(window = 'periodic'))) %>%
  components() %>%
  autoplot() +
  scale_y_continuous(labels = label_dollar())
```



Correlations with Gas Station Sales

```
train %>%
  pivot_longer(-c(date, sales.gas)) %>%
  ggplot(aes(y = (sales.gas), x = (value), color = name)) +
  geom_point() +
  facet_wrap(name ~ ., scales = "free")
```



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(sales.gas),
    "arima" = ARIMA((sales.gas)),
    "dynamic" = ARIMA(sales.gas ~ gas.price + diesel.price)
  ) %>%
   mutate(ensemble = (ets + arima + dynamic) / 3))
```

```
# A mable: 1 x 4
          ets                     arima                                  dynamic
      <model>                   <model>                                  <model>
1 <ETS(A,N,N)> <ARIMA(1,0,1) w/ mean> <LM w/ ARIMA(0,1,2)(2,0,0)[12] errors>
# ... with 1 more variable: ensemble <model>
```

```r
fit %>%
  select(-ensemble) %>%
  glance()
```

```
# A tibble: 3 x 11
  .model  sigma2 log_lik   AIC  AICc   BIC     MSE    AMSE   MAE ar_ro~1 ma_ro~2
  <chr>    <dbl>   <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl> <dbl> <list>  <list>
1 ets     2.87e6  -1055. 2116. 2116. 2124.  2.82e6  7.44e6 1142. <NULL>  <NULL>
2 arima   2.37e6   -946. 1900. 1900. 1910. NA      NA        NA  <cpl>   <cpl>
3 dynamic 7.54e5   -878. 1771. 1772. 1790. NA      NA        NA  <cpl>   <cpl>
# ... with abbreviated variable names 1: ar_roots, 2: ma_roots
```

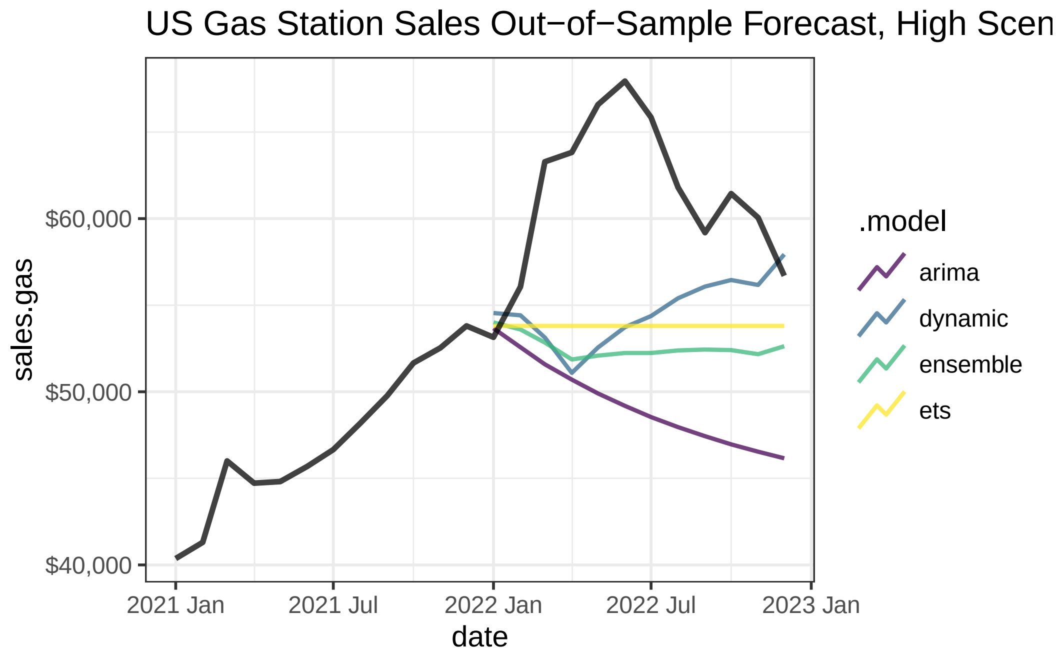
Forecast

The models were trained on data prior to 2022-01-01. The forecast period is the interval 2022-01-01 UTC–2022-12-01 UTC. Three forecasts are produced from each model, one from each of the three scenarios (low, medium, and high).

High

Given the historically high gas and diesel prices during the test set, the high scenario will most likely produce the best forecast. However, as shown below the forecast does underestimate total sales by a considerable amount.

```r
fx.high <- fit %>%
        forecast(test.high)


  fx.high %>%
    autoplot(
        # data %>% filter(year(date)>2020),
        level = NULL, size = .75, alpha = .75
      ) +
    autolayer(
      data %>% tsibble() %>% filter(year(date)>2020), sales.gas,
      size = 1, alpha = .75#, linetype = "dashed"
    ) +
    ggtitle("US Gas Station Sales Out-of-Sample Forecast, High Scenario") +
    scale_color_viridis_d() +
    scale_y_continuous(labels = label_dollar())
```



The dynamic model does produce a slightly better forecast with an RMSE of $8,624.

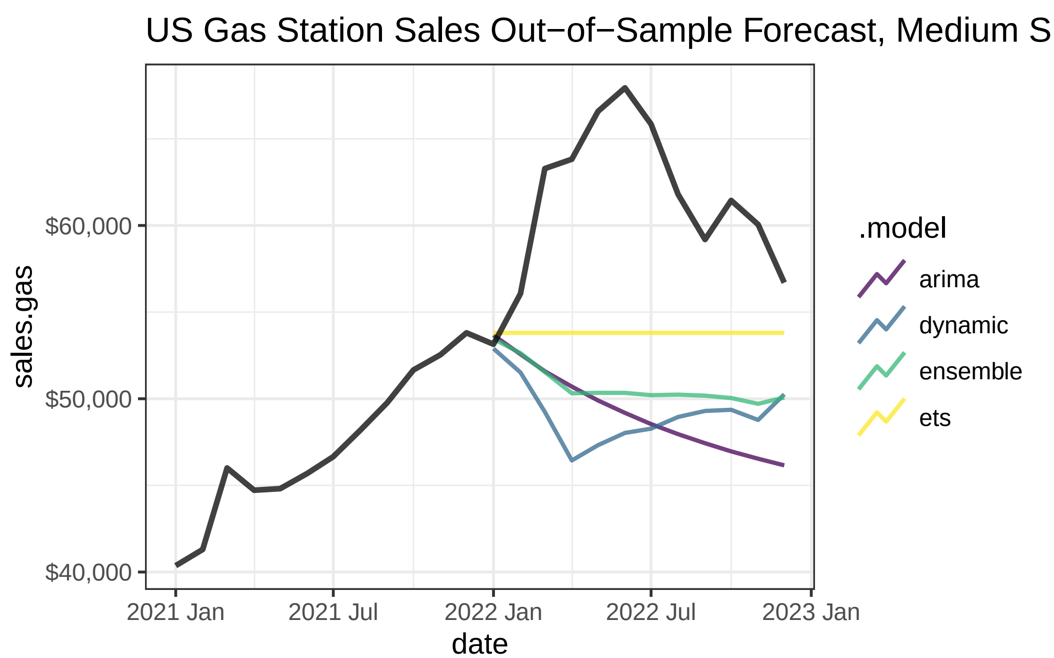
```
fx.high %>%
  accuracy(test.high, measures = list(point_accuracy_measures, distribution_accuracy_measu
  arrange(RMSE)
```

```
# A tibble: 4 x 12
  .model  .type      ME   RMSE    MAE   MPE  MAPE  MASE RMSSE  ACF1 perce~1  CRPS
  <chr>   <chr>   <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl>
1 dynamic Test    6667.  8624.  7106.  10.3  11.1   NaN   NaN 0.593   6226. 6196.
2 ets     Test    7520.  8674.  7629.  11.8  12.0   NaN   NaN 0.544   5826. 5772.
3 ensemb~ Test    8748.  9981.  8891.  13.8  14.0   NaN   NaN 0.550   8891. 8891.
4 arima   Test   12057. 13180. 12143.  19.2  19.3   NaN   NaN 0.561  10023. 9949.
# ... with abbreviated variable name 1: percentile
```

Medium

The medium scenario performs very poorly because of the large deviation of actual versus medium gas/diesel prices.

```
fx.med <- fit %>%
        forecast(test.med)


fx.med %>%
  autoplot(
      # data %>% filter(year(date)>2020),
      level = NULL, size = .75, alpha = .75
    ) +
  autolayer(
    data %>% tsibble() %>% filter(year(date)>2020), sales.gas,
    size = 1, alpha = .75#, linetype = "dashed"
  ) +
  ggtitle("US Gas Station Sales Out-of-Sample Forecast, Medium Scenario") +
  scale_color_viridis_d() +
  scale_y_continuous(labels = label_dollar())
```



The ETS model performs best by a large margin with an RMSE of $8,674.

```
fx.med %>%
  accuracy(test.med, measures = list(point_accuracy_measures, distribution_accuracy_measur
```

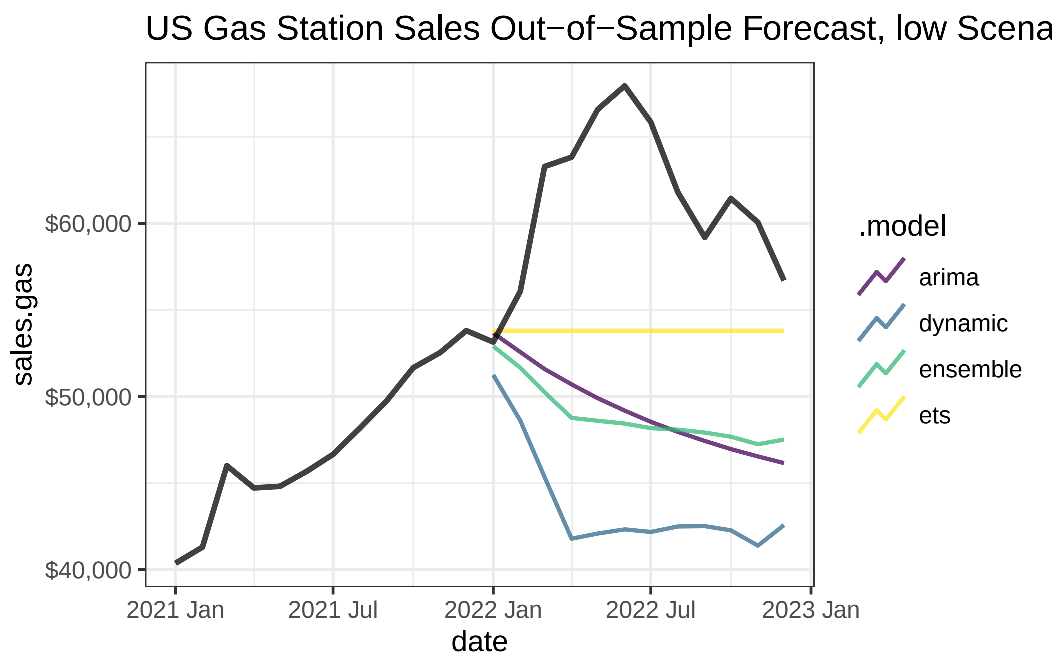
```r
  arrange(RMSE)
```

```
# A tibble: 4 x 12
  .model .type     ME   RMSE    MAE   MPE  MAPE  MASE RMSSE  ACF1 perce~1   CRPS
  <chr>  <chr>  <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>  <dbl>
1 ets    Test   7520.  8674.  7629.  11.8  12.0   NaN   NaN 0.544   5826.  5772.
2 ensem~ Test  10568. 11714. 10620.  16.7  16.8   NaN   NaN 0.548  10620. 10620.
3 arima  Test  12057. 13180. 12143.  19.2  19.3   NaN   NaN 0.561  10023.  9949.
4 dynam~ Test  12127. 13461. 12127.  19.2  19.2   NaN   NaN 0.558  11188. 11146.
# ... with abbreviated variable name 1: percentile
```

Low

The low scenario is a very poor forecast. The ETS model once again does best since it does not use any exogenous regressors.

```r
fx.low <- fit %>%
        forecast(test.low)


  fx.low %>%
    autoplot(
        # data %>% filter(year(date)>2020),
        level = NULL, size = .75, alpha = .75
      ) +
    autolayer(
      data %>% tsibble() %>% filter(year(date)>2020), sales.gas,
      size = 1, alpha = .75#, linetype = "dashed"
    ) +
    ggtitle("US Gas Station Sales Out-of-Sample Forecast, low Scenario") +
    scale_color_viridis_d() +
    scale_y_continuous(labels = label_dollar())
```




```
fx.low %>%
  accuracy(test.low, measures = list(point_accuracy_measures, distribution_accuracy_measur
  arrange(RMSE)
```

```
# A tibble: 4 x 12
  .model .type     ME   RMSE    MAE   MPE  MAPE  MASE RMSSE  ACF1 perce~1   CRPS
  <chr>  <chr>  <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>  <dbl>
1 ets    Test   7520.  8674.  7629.  11.8  12.0   NaN   NaN 0.544   5826.  5772.
2 arima  Test  12057. 13180. 12143.  19.2  19.3   NaN   NaN 0.561  10023.  9949.
3 ensem~ Test  12388. 13496. 12388.  19.7  19.7   NaN   NaN 0.550  12388. 12388.
4 dynam~ Test  17588. 18818. 17588.  28.1  28.1   NaN   NaN 0.554  16610. 16568.
# ... with abbreviated variable name 1: percentile
```

While averaging forecasts usually creates a better estimate, it performs poorly due to the poor models used in the evaluation.

Bagged Forecasts

Due to the uncertainty in the regressors, bagging the forecast could aid tremendously. We'll generate 100 new training sets by simulating an STL model. We'll create a new test set that combines them into one and updates the tsibble key to include these scenarios.

Simulating New Data

```
sales_stl <- train %>%
  model(STL(sales.gas))

sim <- sales_stl %>%
  generate(new_data = train, times = 100, bootstrap_block_size = 12) %>%
  select(-.model, -sales.gas)

test.sim <- list(
    test.high %>% rename(.sim = sales.gas) %>% mutate(.scenario = "high") %>% as_tibble(),
    test.med %>% rename(.sim = sales.gas) %>% mutate(.scenario = "medium") %>% as_tibble()
    test.low %>% rename(.sim = sales.gas) %>% mutate(.scenario = "low") %>% as_tibble()
  ) %>%
  do.call(bind_rows, .) %>%
  cross_join(y = tibble(.rep = as.character(1:100))) %>%
  tsibble(key = c(.rep, .scenario))
```

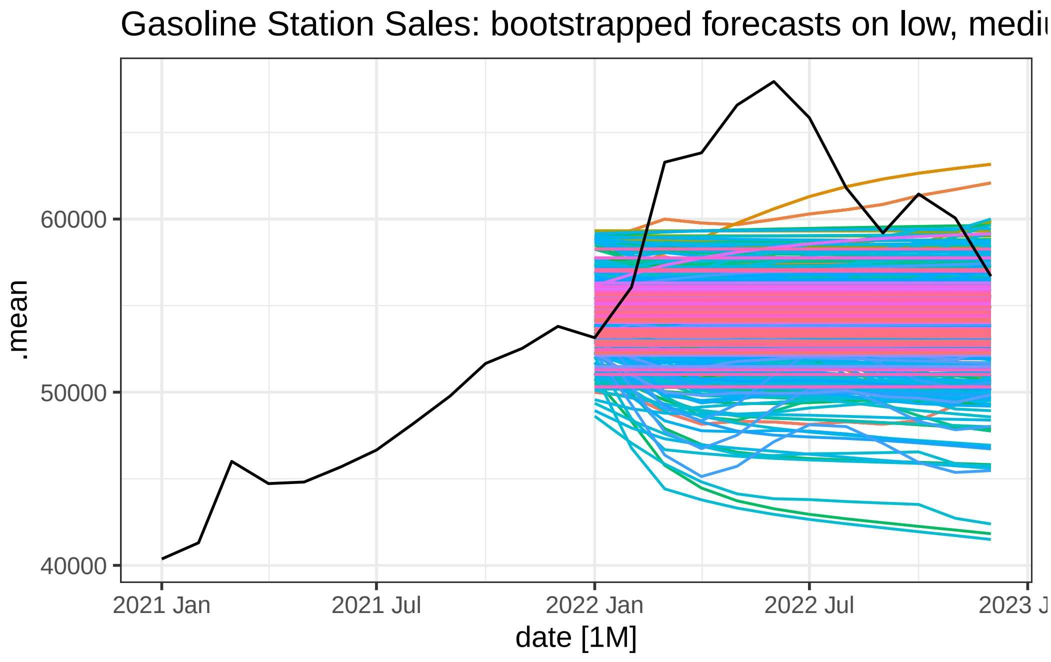
Using this simulated training set, we build new models of the same type just as before. Each model type will be fit to each of the 20 training sets in turn giving $100*3 = 300$ different models. These models will then forecast the testing data for each scenario, resulting in $300 * 3 = 900$ different forecasts.

Model & Forecast

The below code estimated these models and saved the output for this report compilation to read.

```
sim_models <- sim %>%
  model(
    ets = ETS(.sim),
    arima = ARIMA(.sim ~ pdq(d = 1) + PDQ(D=0)),
    dynamic = ARIMA(.sim ~ gas.price + pdq(d = 1) + PDQ(D=0)),
    # dynamic2 = ARIMA(.sim ~ gas.price + diesel.price + pdq(d = 1) + PDQ(D=0))
  )


saveRDS(sim_models, "sim_models.RDS")
```

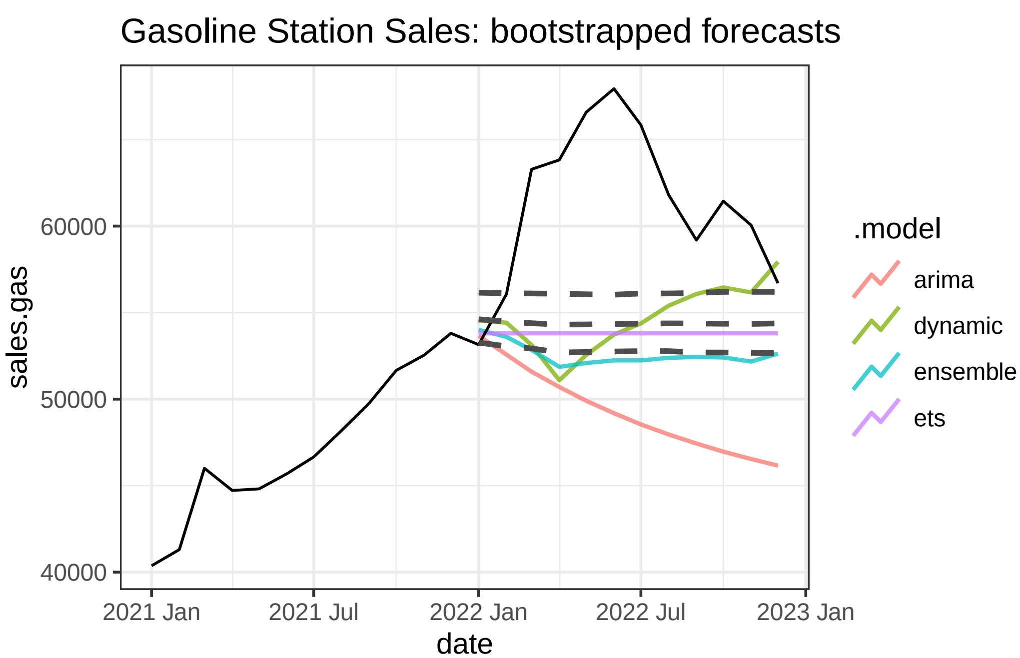
```
sim_models <- readRDS("sim_models.RDS") %>%
  select(-dynamic2)


sim_forecasts <- lapply(test.sim %>% split(~.scenario), \(x){
  sim_models %>%
    forecast(update_tsibble(x, key = .rep))
}) %>%
  lapply(as_tibble) %>%
  do.call(bind_rows, .) %>%
  tsibble(key = c(.scenario, .rep, .model))


sim_forecasts %>%
  autoplot(.mean) +
  autolayer(
    data %>% tsibble() %>% filter(year(date)>2020), sales.gas
  ) +
  guides(colour = "none") +
  labs(title = "Gasoline Station Sales: bootstrapped forecasts on low, medium, and high sc
```



These bootstrapped forecasts are then averaged together to create a new ensemble forecast.

The below plot compares this bagged forecast against the best scenario, the high scenario. The dashed lines are the bagged forecast at the 80th percentile, mean, and 20th percentile, respectively.

```
bagged <- sim_forecasts %>%
  summarise(
    bagged_mean = mean(.mean),
    bagged_median = median(.mean),
    bagged_p80 = quantile(.mean, .8),
    bagged_p20 = quantile(.mean, .2)
  )

fx.high %>%
  autoplot(
      data %>% tsibble() %>% filter(year(date)>2020),
      level = NULL,
      size = .75, alpha = .75#, linetype = "dashed"
  ) +
  autolayer(bagged, bagged_mean, col = "gray30", size = 1, linetype = "dashed") +
  autolayer(bagged, bagged_p80, col = "gray30", size = 1, linetype = "dashed") +
  autolayer(bagged, bagged_p20, col = "gray30", size = 1, linetype = "dashed") +
  labs(title = "Gasoline Station Sales: bootstrapped forecasts")
```



Model Comparison

The following table summarizes each of these models and scenarios:

```
bind_rows(
    fx.low %>% mutate(.scenario = "low") %>% as_tibble() %>% select(.scenario, .model, dat
    fx.med %>% mutate(.scenario = "med") %>% as_tibble() %>% select(.scenario, .model, dat
    fx.high %>% mutate(.scenario = "high") %>% as_tibble() %>% select(.scenario, .model, d
    bagged %>%
      as_tibble() %>%
      select(date, bagged_mean) %>%
      pivot_longer(-date, names_to = ".model", values_to = ".mean") %>%
      mutate(.scenario = "bagged")
  ) %>%
  left_join(test, by = "date") %>%
  as_tibble() %>%
  group_by(.scenario, .model) %>%
  summarize(
    RMSE = RMSE(.mean - sales.gas),
    MAPE = MAPE(.mean - sales.gas, .actual = sales.gas),
    MAE = MAE(.mean - sales.gas)
  ) %>%
  arrange(RMSE)
```

```
# A tibble: 13 x 5
# Groups:   .scenario [4]
   .scenario .model        RMSE  MAPE    MAE
   <chr>     <chr>        <dbl> <dbl>  <dbl>
 1 bagged    bagged_mean  8207.  11.3  7182.
 2 high      dynamic      8624.  11.1  7106.
 3 high      ets          8674.  12.0  7629.
 4 low       ets          8674.  12.0  7629.
 5 med       ets          8674.  12.0  7629.
 6 high      ensemble     9981.  14.0  8891.
 7 med       ensemble    11714.  16.8 10620.
 8 high      arima       13180.  19.3 12143.
 9 low       arima       13180.  19.3 12143.
10 med       arima       13180.  19.3 12143.
11 med       dynamic     13461.  19.2 12127.
12 low       ensemble    13496.  19.7 12388.
13 low       dynamic     18818.  28.1 17588.
```

The bagged forecast outperforms all the other forecasts, even the dynamic model in the high scenario. This is despite the high scenario dynamic model only containing the *better* data in the test set and the bagged forecast containing all low, medium, and high scenarios. A better forecast could possibly also be achieved by simulating the test set gas and diesel prices rather than using scenario forecasts. This approach would also enable us to quantify the uncertainty present in the exogenous regressors better. The first approach produces near meaningless confidence intervals, since those do not consider the uncertainty in the regressors. Simulating these variables and calculating the quantiles enables us to embed this uncertainty within the forecast.