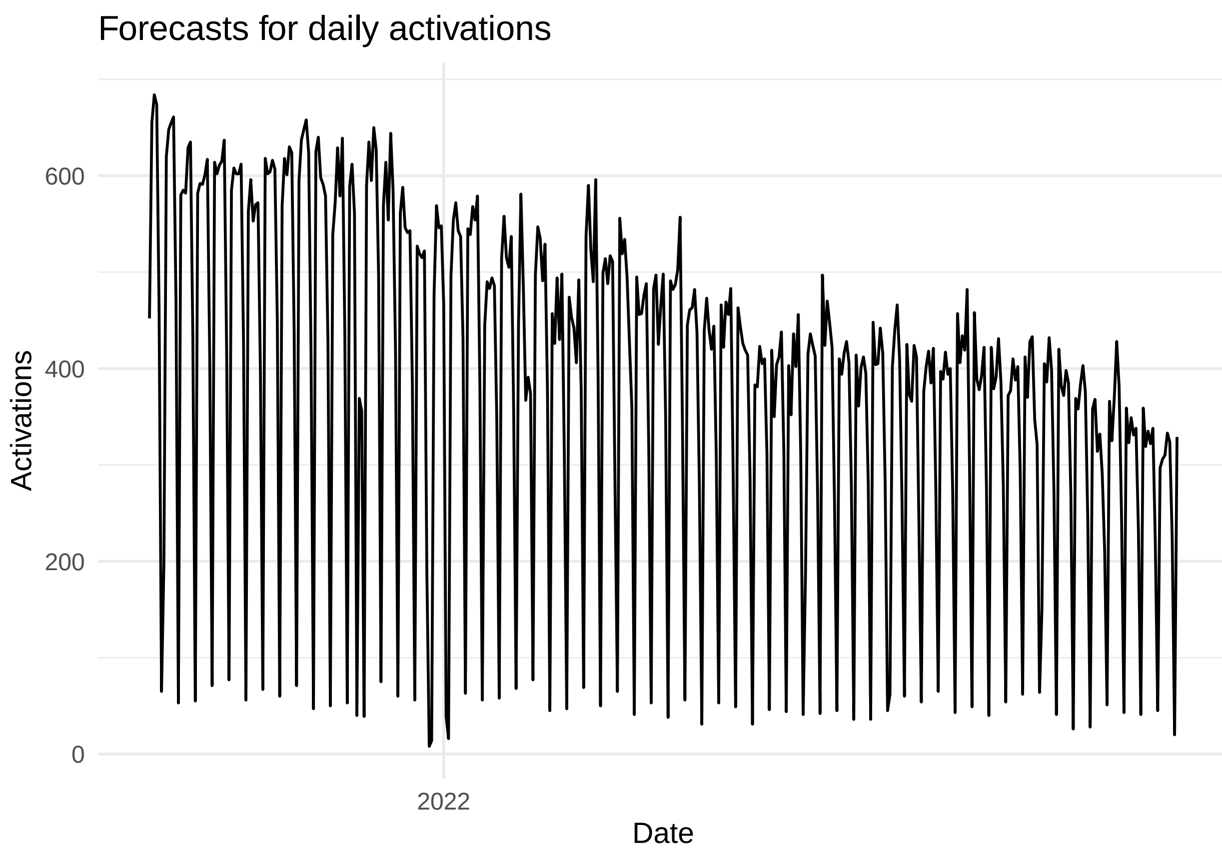
EDA

Zachariah Freitas

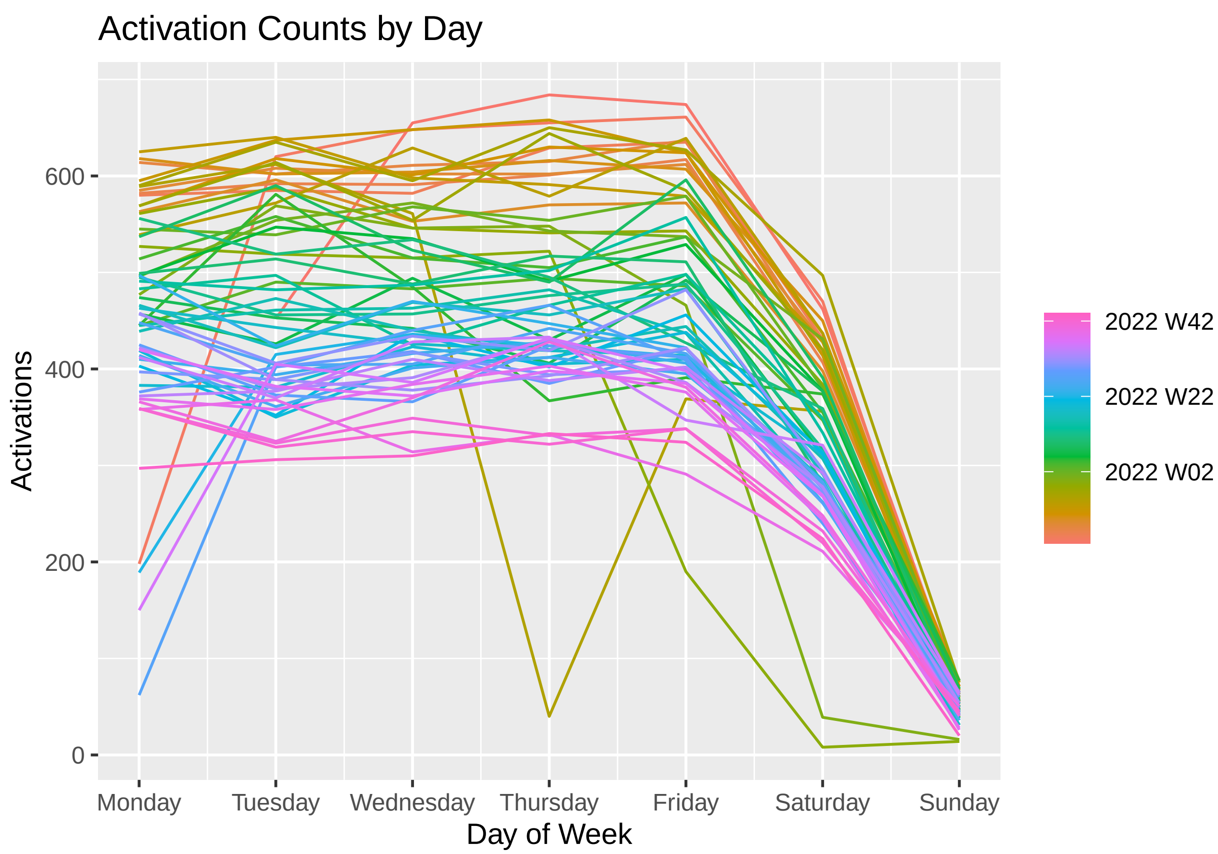
2022-12-05

EDA

```r
# Create Time Series Object
myts <- ts(df$activation_count,
           start = c(2021, as.numeric(format(df$date[1], "%j"))),
           frequency = 364)


autoplot(myts) +
  xlab("Date") +
  ylab("Activations") +
  ggtitle("Forecasts for daily activations") +
  theme_minimal() +
  guides(colour=guide_legend(title="Forecast"))
```



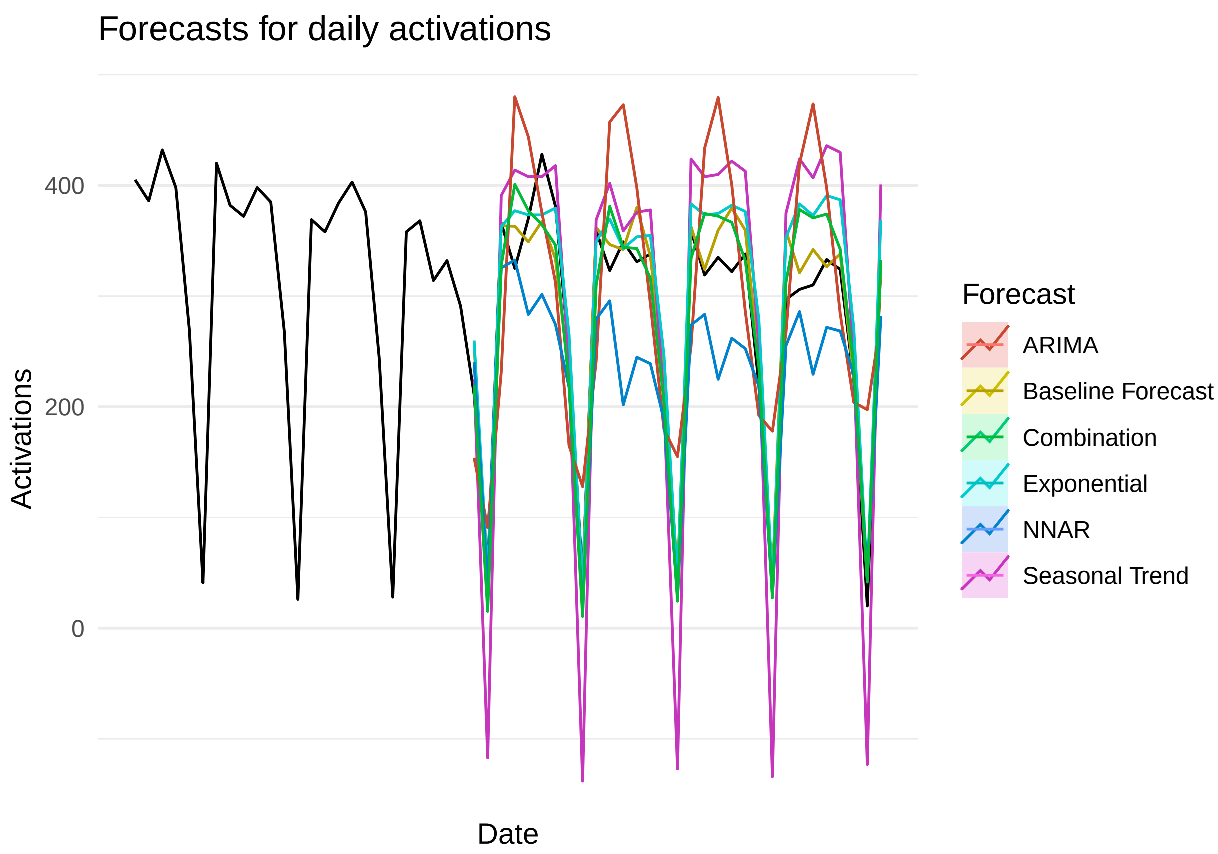

```
temp <- df %>%
  mutate(svc_agreement_activation_date = as.Date(svc_agreement_activation_date)) %>%
  as_tsibble(., index = svc_agreement_activation_date)

temp %>%
  gg_season(activation_count, period = "week") +
  theme(legend.position = "right") +
  labs(y="Activations", x="Day of Week", title="Activation Counts by Day")
```




```
library(forecast)

# Split Data
train <- window(myts, end = c(2022, 274)) #273 orig 300
valid <- window(myts, start = c(2022, 275)) # October of 2022

# Set Forecast periods
h <- length(myts) - length(train)

############
# Naïve Models



current_fit <- meanf(train,h=h) # Naïve: Mean
naive_fit <- naive(train,h=h,level = 95) # Naïve: Last Value
drift_fit <- rwf(train,h=h,drift=TRUE) # Naïve: Drift Method
snaive_fit <- snaive(train,h=h) # Naïve: Seasonal Naïve

#############
# Train Models
# Trend + Seasonal Model
ST <- forecast::forecast(forecast::tslm(train ~ trend + season),h=h)

# Exponential Model
EXP <- forecast::forecast(forecast::tslm(train ~ trend + season, lambda = 0),h=h)
```

```r
# NNAR - Autofit neural network
NNAR <- forecast::forecast(nnetar(train), h=h)#, lambda=0)


# Auto ARIMA
ARIMA <- forecast::forecast(auto.arima(train, lambda=0, biasadj=TRUE),h=h)


# Ensemble Approach
Combination <- (EXP[["mean"]] +
                  ARIMA[["mean"]] +
                  ST[["mean"]] +
                  NNAR[["mean"]])/4
```

```r
naive_df <- df %>%
  mutate(naive_forecast = (lag7 + lag14)/2) %>%
  select(svc_agreement_activation_date, naive_forecast, activation_count)


# Create Time Series Object
mynaivets <- ts(naive_df$naive_forecast,
           start = c(2021, as.numeric(format(naive_df$svc_agreement_activation_date[1], "%j"))),
           frequency = 364)

# Plot Results
autoplot(window(myts, start=c(2022, 250))) +
  autolayer(window(mynaivets, start=c(2022, 275)), series="Baseline Forecast") +
  autolayer(ST, series="Seasonal Trend", PI=FALSE) +
  autolayer(EXP, series="Exponential", PI=FALSE) +
  autolayer(ARIMA, series="ARIMA", PI=FALSE) +
  autolayer(NNAR, series="NNAR", PI=FALSE) +
  autolayer(Combination, series="Combination") +
  xlab("Date") +
  ylab("Activations") +
  ggtitle("Forecasts for daily activations") +
  theme_minimal() +
  guides(colour=guide_legend(title="Forecast"))
```



Including Plots

```r
# Baseline model

resids <- naive_df %>%
  filter(svc_agreement_activation_date >= as.Date('2022-10-01')) %>%
  mutate(residuals = naive_forecast - activation_count) %>%
  select(residuals)

# RMSE
print("Baseline Forecast")
```

```
## [1] "Baseline Forecast"
```

```r
print("RMSE - Test Set")
```

```
## [1] "RMSE - Test Set"
```

```r
round(sqrt(mean(resids$residuals^2)),2)
```

```
## [1] 28.16
```

```
print("")
```

```
## [1] ""
```

```
print(strrep("#", 80))
```

```
## [1] "################################################################################"
```

```
print("Seasonal Trend")
```

```
## [1] "Seasonal Trend"
```

```
round(forecast::accuracy(ST, valid),2)
```

```
##                  ME  RMSE   MAE   MPE  MAPE MASE ACF1 Theil's U
## Training set   0.00 17.74  5.55  0.85  4.89 0.03 0.29        NA
## Test set     -16.55 90.87 74.78 59.84 88.43 0.39 0.11      0.23
```

```
print("")
```

```
## [1] ""
```

```
print(strrep("#", 80))
```

```
## [1] "################################################################################"
```

```
print("Exponential")
```

```
## [1] "Exponential"
```

```
round(forecast::accuracy(EXP, valid),2)
```

```
##                  ME  RMSE   MAE    MPE  MAPE MASE ACF1 Theil's U
## Training set   0.13  9.87  2.77  -0.06  0.95 0.01 0.27        NA
## Test set     -26.82 40.26 32.92 -11.82 15.78 0.17 0.35      0.12
```

```
print("")
```

```
## [1] ""
```

```
print(strrep("#", 80))
```

```
## [1] "################################################################################"
```

```r
print("Neural Network")
```

```
## [1] "Neural Network"
```

```r
round(forecast::accuracy(NNAR, valid),2)
```

```
##                 ME  RMSE   MAE   MPE  MAPE MASE  ACF1 Theil's U
## Training set  0.00  0.06  0.05  0.00  0.03 0.00 -0.14        NA
## Test set     50.16 66.65 54.66 14.54 24.80 0.28  0.21      0.19
```

```r
print("")
```

```
## [1] ""
```

```r
print(strrep("#", 80))
```

```
## [1] "################################################################################"
```

```r
print("ARIMA")
```

```
## [1] "ARIMA"
```

```r
round(forecast::accuracy(ARIMA, valid),2)
```

```
##                  ME   RMSE    MAE    MPE  MAPE MASE  ACF1 Theil's U
## Training set -50.08 181.10 126.41 -51.30 69.38 0.65 -0.07        NA
## Test set     -27.29  97.78  87.68 -58.47 77.83 0.45 -0.17      0.27
```

```r
print("")
```

```
## [1] ""
```

```r
print(strrep("#", 80))
```

```
## [1] "################################################################################"
```

```r
print("Combination")
```

```
## [1] "Combination"
```

```r
round(forecast::accuracy(Combination, valid),2)
```

```
##             ME  RMSE   MAE  MPE MAPE ACF1 Theil's U
## Test set -5.13 37.11 31.04 1.02 19.2 0.28      0.08
```

```
print("")
```

```
## [1] ""
```

You can also embed plots, for example:

```
c(
  Baseline = round(sqrt(mean(resids$residuals^2)),2),
  ST = forecast::accuracy(ST, valid)["Test set","RMSE"],
  EXP = forecast::accuracy(EXP, valid)["Test set","RMSE"],
  NNAR = forecast::accuracy(NNAR, valid)["Test set","RMSE"],
  ARIMA = forecast::accuracy(ARIMA, valid)["Test set","RMSE"],
  Combination = forecast::accuracy(Combination, valid)["Test set","RMSE"])
```

```
##    Baseline          ST         EXP        NNAR       ARIMA Combination
##    28.16000    90.87486    40.25669    66.64676    97.78369    37.10603
```

Let's try adding the Baseline Forecast to the ensemble to see if we get an improvement in our prediction.

```
baseline.predict <- df %>%
  select(svc_agreement_activation_date, activation_count, lag7, lag14) %>%
  mutate(naive_forecast = (lag7 + lag14)/2) %>%
  filter(svc_agreement_activation_date >= as.Date('2022-10-01'))


NewCombination <- (EXP[["mean"]] +
                  ARIMA[["mean"]] +
                  ST[["mean"]] +
                  NNAR[["mean"]] +
                  baseline.predict[["naive_forecast"]])/5

round(forecast::accuracy(NewCombination, valid),2)
```

```
##             ME  RMSE  MAE   MPE  MAPE ACF1 Theil's U
## Test set -5.67 33.54 28.2 -0.22 17.41 0.29      0.07
```