1. Intro
2. Cross validation

Metrics

1. Pre-processing
   1. Base recipe
   2. Spline recipe
   3. Prophet boost recipe 4 Modelling
   4. GLM
   5. MARS
   6. RF
   7. XGB
   8. Prophet boost

5. Evaluation Conclusion

# Intro

The aim of this series of blog is to predict monthly admissions to Singapore public acute adult hospitals.

library(tidyverse) library(tidymodels) library(timetk) library(modeltime) library(modeltime.ensemble)

# dataset and dataset for future forecast dats and pre-processing recipes were done in the past post. The output was uploaded onto my github.

url\_datasets<-url("https://github.com/notast/hierarchical-forecasting/blob/ main/4Dataset\_ML.rds?raw=true")

load(url\_datasets) close(url\_datasets)

head(to\_train,10)

## # A tibble: 10 x 47

## row\_id Level Name Date Admission Admission\_lag10 Admission\_lag10\_r~

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | <int> | <chr> | <chr> | <date> | <dbl> | <dbl> |
| <dbl> |  |  |  |  |  |  |
| ## 1 | 1 | Cluster~ | NHG | 2016-01-01 | 8035 | 8184. |

7840.

## 2 2 Cluster~ NHG 2016-02-01 7526 7801.

8185.

## 3 3 Cluster~ NHG 2016-03-01 8419 8283.

8072.

## 4 4 Cluster~ NHG 2016-04-01 7934 7926.

8188.

## 5 5 Cluster~ NHG 2016-05-01 8048 8168.

8169.

## 6 6 Cluster~ NHG 2016-06-01 8199 8229.

8097.

## 7 7 Cluster~ NHG 2016-07-01 8230 7608.

7982.

## 8 8 Cluster~ NHG 2016-08-01 8496 7969.

7844.

## 9 9 Cluster~ NHG 2016-09-01 7991 7792.

8009.

## 10 10 Cluster~ NHG 2016-10-01 8284 8041.

8035

## # ... with 40 more variables: Admission\_lag10\_roll\_6 <dbl>,

## # Admission\_lag10\_roll\_12 <dbl>, Covid <chr>, F\_trend\_strength

<dbl>,

## # F\_seasonal\_strength\_year <dbl>, F\_seasonal\_peak\_year <dbl>, ## # F\_seasonal\_trough\_year <dbl>, F\_spikiness <dbl>, F\_linearity

<dbl>,

## # F\_curvature <dbl>, F\_stl\_e\_acf1 <dbl>, F\_stl\_e\_acf10 <dbl>, F\_acf1 <dbl>,

## # F\_acf10 <dbl>, F\_diff1\_acf1 <dbl>, F\_diff1\_acf10 <dbl>, F\_diff2\_acf1 <dbl>,

## # F\_diff2\_acf10 <dbl>, F\_season\_acf1 <dbl>, F\_kpss\_stat <dbl>, ## # F\_kpss\_pvalue <dbl>, F\_pp\_stat <dbl>, F\_pp\_pvalue <dbl>, F\_ndiffs <int>,

## # F\_bp\_stat <dbl>, F\_bp\_pvalue <dbl>, F\_lb\_stat <dbl>, F\_lb\_pvalue

<dbl>,

## # F\_var\_tiled\_var <dbl>, F\_var\_tiled\_mean <dbl>, F\_shift\_level\_max

<dbl>,

## # F\_shift\_level\_index <dbl>, F\_shift\_var\_max <dbl>, F\_shift\_var\_index <dbl>,

## # F\_shift\_kl\_max <dbl>, F\_shift\_kl\_index <dbl>, F\_spectral\_entropy

<dbl>,

## # F\_n\_crossing\_points <int>, F\_longest\_flat\_spot <int>, F\_stat\_arch\_lm <dbl>

The admissions were treated as a hierarchical time series as every country has a hierarchical order to its public hospitals including Singapore. The levels are

National level

|– Cluster level (Clusters are a network of hospitals based on geographical regions. There are 3 health clusters in Singapore.)

|– Hospital level (There are 8 public acute adult hospitals.)

# Cross validation

The dataset starts from Jan 2016 and ends in Feb 2021 (to\_train). This dataset was split into training and testing set. The training set was from Jan 16 to Apr 20 (3 years, 4months) and the test set was from May 20 to Feb 21 (10 months). Cross validation was applied to the training set

as the forecasting was a machine learning problem.

# train/test

splits<- to\_train %>%

time\_series\_split(Date, assess= "10 months", cumulative = T)

## Data is not ordered by the 'date\_var'. Resamples will be arranged by

`Date`.

## Overlapping Timestamps Detected. Processing overlapping time series together using sliding windows.

# training set

splits\_train<- training(splits)

# vfold from training set set.seed(69)

folds<-vfold\_cv(splits\_train, strata = Admission, v=5)

## Metrics

metrics\_custom= metric\_set(rmse, mae

# Pre-processing

The base recipe, rec\_PC, was crafted in the previous post where different combination of predictors and features engineered for machine learning were screened The predictors are:

* 1. Lags (Admission\_lag10)
  2. Rolling lags (Admission\_lag\_roll\_???)
  3. Covid peak period (Covid)
  4. The PCA of time series features and statistics (F\_???)
  5. The hierarchical level (Level) and its members (Name)

rec\_PC %>% summary() %>% filter(role=="predictor") %>% pull(variable)

|  |  |  |  |
| --- | --- | --- | --- |
| ## | [1] | "Level" | "Name" |
| ## | [3] | "Covid" | "Admission\_lag10" |
| ## | [5] | "Admission\_lag10\_roll\_3" | "Admission\_lag10\_roll\_6" |
| ## | [7] | "Admission\_lag10\_roll\_12" | "F\_trend\_strength" |
| ## | [9] | "F\_seasonal\_strength\_year" | "F\_seasonal\_peak\_year" |
| ## | [11] | "F\_seasonal\_trough\_year" | "F\_spikiness" |
| ## | [13] | "F\_linearity" | "F\_curvature" |
| ## | [15] | "F\_stl\_e\_acf1" | "F\_stl\_e\_acf10" |
| ## | [17] | "F\_acf1" | "F\_acf10" |
| ## | [19] | "F\_diff1\_acf1" | "F\_diff1\_acf10" |
| ## | [21] | "F\_diff2\_acf1" | "F\_diff2\_acf10" |
| ## | [23] | "F\_season\_acf1" | "F\_kpss\_stat" |
| ## | [25] | "F\_kpss\_pvalue" | "F\_pp\_stat" |
| ## | [27] | "F\_pp\_pvalue" | "F\_ndiffs" |
| ## | [29] | "F\_bp\_stat" | "F\_bp\_pvalue" |
| ## | [31] | "F\_lb\_stat" | "F\_lb\_pvalue" |
| ## | [33] | "F\_var\_tiled\_var" | "F\_var\_tiled\_mean" |
| ## | [35] | "F\_shift\_level\_max" | "F\_shift\_level\_index" |
| ## | [37] | "F\_shift\_var\_max" | "F\_shift\_var\_index" |
| ## | [39] | "F\_shift\_kl\_max" | "F\_shift\_kl\_index" |

## [41] "F\_spectral\_entropy" "F\_n\_crossing\_points" ## [43] "F\_longest\_flat\_spot" "F\_stat\_arch\_lm"

The pre-processing steps are:

tidy(rec\_PC)

## # A tibble: 8 x 6

## number operation type trained skip id

## <int> <chr> <chr> <lgl> <lgl> <chr> ## 1 1 step timeseries\_signature FALSE FALSE timeseries\_signature\_ExurK

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## | 2 | 2 | step | rm | FALSE | FALSE | rm\_OHKgv |
| ## | 3 | 3 | step | rm | FALSE | FALSE | rm\_GnkOP |
| ## | 4 | 4 | step | rm | FALSE | FALSE | rm\_CeaGH |
| ## | 5 | 5 | step | rm | FALSE | FALSE | rm\_BZcN8 |
| ## | 6 | 6 | step | dummy | FALSE | FALSE | dummy\_xLyW5 |
| ## | 7 | 7 | step | normalize | FALSE | FALSE |  |

normalize\_pAZbw

## 8 8 step pca FALSE FALSE pca\_aRkvg

The following machine learning models were trial thus the base recipes had to be revised to compliment some of the models.

1. Elastic net regression GLM (Linear model)
2. Multivariate adaptive regression spline MARS (Non-linear model)
3. Random forest RF (Tree model)
4. Extreme gradient boost XGB (Tree-model)
5. Boosted PROPHET PB (Classical approach + Tree-model ) ~~6. LightGBM (Tree-model,~~ ~~Light GBM has seen success with hierarchical time series in the M5 competition but fata~~l ~~errors were encountered when running it in R)~~

## Base recipe

The base recipe was extended to remove non-zero variance.

rec\_PC<- rec\_PC %>% step\_nzv(all\_numeric\_predictors())

rec\_PC %>% tidy()

## # A tibble: 9 x 6

## number operation type trained skip id

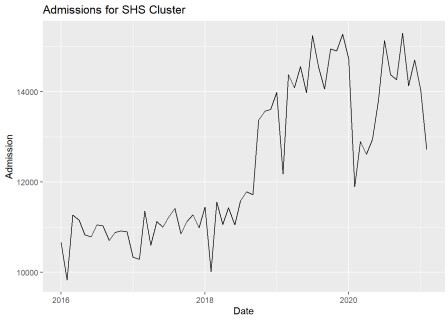
## <int> <chr> <chr> <lgl> <lgl> <chr> ## 1 1 step timeseries\_signature FALSE FALSE timeseries\_signature\_ExurK

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## | 2 | 2 | step | rm | FALSE | FALSE | rm\_OHKgv |
| ## | 3 | 3 | step | rm | FALSE | FALSE | rm\_GnkOP |
| ## | 4 | 4 | step | rm | FALSE | FALSE | rm\_CeaGH |
| ## | 5 | 5 | step | rm | FALSE | FALSE | rm\_BZcN8 |
| ## | 6 | 6 | step | dummy | FALSE | FALSE | dummy\_xLyW5 |
| ## | 7 | 7 | step | normalize | FALSE | FALSE |  |
| normalize\_pAZbw | | | | | | | |
| ## 8 | | 8 step | | pca | FALSE | FALSE | pca\_aRkvg |
| ## 9 | | 9 step | | nzv | FALSE | FALSE | nzv\_bU0rc |

## Spline recipe

To assist the linear model GLM to capture the wiggles of the time series, splines were included.

to\_train %>% filter(Name=="SHS") %>% ggplot(aes(Date, Admission)) + geom\_line() + labs(title = "Admissions for SHS Cluster")



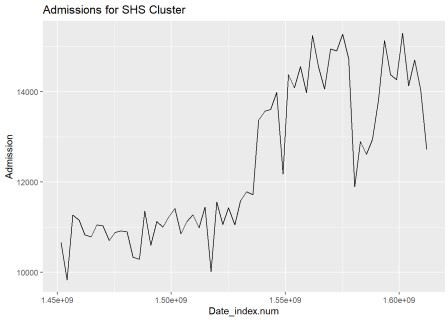
However glmnet does not tolerate non-numeric variables thus Date cannot be used as the variable for creating the splines. step\_timeseries\_signature derives various features from Date including a numeric equivalent, Date\_index.num.

to\_train %>% filter(Name=="SHS") %>%

recipe(~.) %>% step\_timeseries\_signature(Date) %>% prep() %>% juice()

%>%

ggplot(aes(Date\_index.num, Admission)) + geom\_line() + labs(title = "Admissions for SHS Cluster")



B-splines were created as there are more flexible than natural splines; hopefully, capturing the

wiggles of the time series. The degrees of freedom was tuned.

All numeric predictors with zero variance were removed and then all numeric predictors were normalized to prepare it for elastic net regression.

rec\_spline<- rec\_PC %>%

step\_bs(Date\_index.num, deg\_free = tune("bs\_df")) %>% step\_zv(all\_numeric\_predictors())%>% step\_normalize(all\_numeric\_predictors())

rec\_spline %>% tidy() %>% pull(type)

## [1] "timeseries\_signature" "rm" "rm"

## [4] "rm" "rm" "dummy"

## [7] "normalize" "pca" "nzv"

## [10] "bs" "zv" "normalize"

## Prophet boost recipe

When using modeltime for forecasting, the date column is treated differently for classical and machine learning approach. Previously, the date column was not treated as a predictor as a machine learning approach was adopted. However, Prophet Boost requires the date to be a predictor for the Prophet part of the model as prophet is a classical approach.

rec\_Date<- rec\_PC %>% update\_role(Date, new\_role = "predictor")

# 4 Modelling

Modelling steps are easy with tidymodels.

1. Set up the model
2. Add the recipe and model into a workflow
3. Tune the workflow which in turn tunes parameters of the recipe and/or the model inside the workflow
4. Finalize the workflow with the best tuned parameters
5. Fit the finalized workflow with its best tuned recipe and best tuned model onto the whole training data.

## GLM

While it is possible to increase the speed of glmnet modelling with blueprint = hardhat::default\_recipe\_blueprint(composition = "dgCMatrix"), this sparse matrix is not tolerated further downstream when modeltime functions are used for forecasting.

glm\_m<- linear\_reg(penalty = tune(), mixture = tune()) %>% set\_engine("glmnet") %>% set\_mode("regression")

glm\_wf<- workflow() %>% add\_recipe(rec\_spline) %>% add\_model(glm\_m)

The current degree of freedom is:

parameters(glm\_wf)$object[[3]]

## Piecewise Polynomial Degree (quantitative)

## Range: [1, 15]

Splines have been recommended to use a different range of degrees of freedom.

param\_spline<-glm\_wf %>% parameters() %>% update(bs\_df=spline\_degree())

param\_spline$object[[3]]

## Piecewise Polynomial Degree (quantitative) ## Range: [1, 10]

set.seed(69) glm\_t<- tune\_grid(

object= glm\_wf, resamples= folds, param\_info= param\_spline, grid= 20,

metrics= metrics\_custom)

fun\_fwf<- function(wf, t\_wf){ finalize\_workflow(

x= wf,

parameters= select\_best(t\_wf, "rmse")) %>% fit(splits\_train)

}

glm\_f<-fun\_fwf(glm\_wf, glm\_t)

## MARS

mars\_m<- mars(num\_terms = tune(),prod\_degree = tune()) %>% set\_engine("earth") %>% set\_mode("regression")

mars\_wf<- workflow() %>% add\_recipe(rec\_PC) %>% add\_model(mars\_m set.seed(69)

mars\_t<- tune\_grid( object= mars\_wf,

resamples= folds, para\_info\_NULL, grid=10, metrics=metrics\_custom)

mars\_f<- fun\_fwf(mars\_wf, mars\_t

## RF

rf\_m<- rand\_forest(min\_n = tune(), trees = tune()) %>% set\_engine("ranger") %>% set\_mode("regression")

rf\_wf<- workflow() %>% add\_recipe(rec\_PC) %>% add\_model(rf\_m) set.seed(69)

rf\_t<- tune\_grid( object=rf\_wf, resamples = folds, grid=10,

param\_info = NULL, metrics = metrics\_custom)

rf\_f<-fun\_fwf(rf\_wf, rf\_t)

## XGB

xgb\_m<- boost\_tree(sample\_size = tune(),

min\_n = tune(), tree\_depth = tune(), loss\_reduction = tune(), trees=tune()) %>%

set\_engine("xgboost") %>% set\_mode("regression")

xgb\_wf<- workflow() %>% add\_recipe(rec\_PC ) %>% add\_model(xgb\_m)

all\_cores <- parallel::detectCores(logical = FALSE) library(doParallel)

cl <- makePSOCKcluster(all\_cores) registerDoParallel(cl)

set.seed(69) xgb\_t<-tune\_bayes(

object= xgb\_wf, resamples = folds, iter = 20,

param\_info = NULL, initial = 9, metrics = metrics\_custom,

control = control\_bayes(no\_improve = 20))

xgb\_f<-fun\_fwf(xgb\_wf, xgb\_t)

## Prophet boost

pb\_m<- prophet\_boost(

seasonality\_daily = FALSE, seasonality\_weekly = FALSE, seasonality\_yearly = FALSE,

min\_n = tune(), tree\_depth = tune(), loss\_reduction = tune(), trees=tune()) %>%

set\_engine("prophet\_xgboost")

pb\_wf<- workflow() %>% add\_recipe(rec\_Date) %>% add\_model(pb\_m) set.seed(69)

pb\_t<- tune\_bayes( object= pb\_wf, resamples = folds, iter = 20,

param\_info = NULL, initial = 9, # Generate five at semi-random to start

metrics = metrics\_custom,

control = control\_bayes(no\_improve = 20))

pb\_f<- fun\_fwf(pb\_wf, pb\_t)

# 5. Evaluation

After machine learning was completed with tidymodels, the evaluation and forecasting was completed with modeltime. The finalized and fitted workflows were housed in a modeltime\_table.

(e\_table<-modeltime\_table(glm\_f, mars\_f, rf\_f, xgb\_f, pb\_f))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ##  ## ## | # # | Modeltime A tibble:  .model\_id | Table 5 x 3  .model | .model\_desc |
| ## |  | <int> | <list> | <chr> |
| ## | 1 | 1 | <workflow> | GLMNET |
| ## | 2 | 2 | <workflow> | EARTH |
| ## | 3 | 3 | <workflow> | RANGER |
| ## | 4 | 4 | <workflow> | XGBOOST |
| ## | 5 | 5 | <workflow> | PROPHET W/ XGBOOST ERRORS |

The workflows in the modeltime\_table were calibrated on the testing set.

(e\_cal <- e\_table %>% modeltime\_calibrate(testing(splits)))

## Warning in bs(x = c(1588291200, 1590969600, 1593561600, 1596240000,

## 1598918400, : some 'x' values beyond boundary knots may cause ill- conditioned

## bases

## Warning in bs(x = c(1588291200, 1590969600, 1593561600, 1596240000,

## 1598918400, : some 'x' values beyond boundary knots may cause ill- conditioned

## bases

## Warning in bs(x = c(1588291200, 1590969600, 1593561600, 1596240000,

## 1598918400, : some 'x' values beyond boundary knots may cause ill- conditioned

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ##  ## | bases  # Modeltime | | Table |  | | | | |
| ##  ## | # A tibble:  .model\_id | | 5 x 5  .model | .model\_desc | | .type | | |
| .calibration\_data  ## <int> <list> | | | | <chr> |  | <chr> | <list> |  |
| ## 1  x 4]>  ## 2 | | 1  2 | <workflow>  <workflow> | GLMNET  EARTH |  | Test  Test | <tibble  <tibble | [120  [120 |
| x 4]>  ## 3 | | 3 | <workflow> | RANGER |  | Test | <tibble | [120 |
| x 4]>  ## 4 | | 4 | <workflow> | XGBOOST |  | Test | <tibble | [120 |
| x 4]>  ## 5 | | 5 | <workflow> | PROPHET | W/ XGBOOST | ERRORS Test | <tibble | [120 |
| x 4]> | |  |  |  |  |  |  |  |

The tree models performed well on the testing set. XGB performer poorer than prophet boost likely due to XGB’s reduced ability to capture trends which is better captured by the Prophet part of Prophet Boost.

e\_cal %>% modeltime\_accuracy(metric\_set = metrics\_custom) %>% arrange(rmse, sort=T)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ##  ## | # | A tibble:  .model\_id | 5 x 5  .model\_desc |  |  | .type | rmse | mae |
| ## |  | <int> | <chr> |  |  | <chr> | <dbl> | <dbl> |
| ## | 1 | 3 | RANGER |  |  | Test | 549. | 410. |
| ## | 2 | 5 | PROPHET W/ | XGBOOST | ERRORS | Test | 1137. | 799. |
| ## | 3 | 4 | XGBOOST |  |  | Test | 1231. | 888. |
| ## | 4 | 2 | EARTH |  |  | Test | 3796. | 3312. |
| ## | 5 | 1 | GLMNET |  |  | Test | 9847. | 8281. |

# Conclusion

The top 2 models, Random forest and Prophet Boost, were selected for retuning to improve performance. Retuning would be covered in the next post.

#The folllowing were exported to save code and computational time for future posts

#1. Datasets and the cross validation splits AND folds save(to\_train, to\_predictfuture, splits, splits\_train, folds, file="5Data\_CV.RData")

#2. tuned grids and workflows (for retunining)

save(rf\_t, rf\_wf, pb\_t, pb\_wf, file="5Retunning\_objects.RData") #3. finalized and fitted wf (to add to modeltime table and compare retunned and OG wf)

save(glm\_f, mars\_f, rf\_f, xgb\_f, pb\_f, file="5FFWf.RData")