Compartmentalized Baseline Modeling

Ryan Rogers

10/28/2022

Library imports are left as-is. They'll be necessary in almost every version. New imports added for helper libraries

Modeling with threshold 50 number of claims

Will leave data import alone for now.

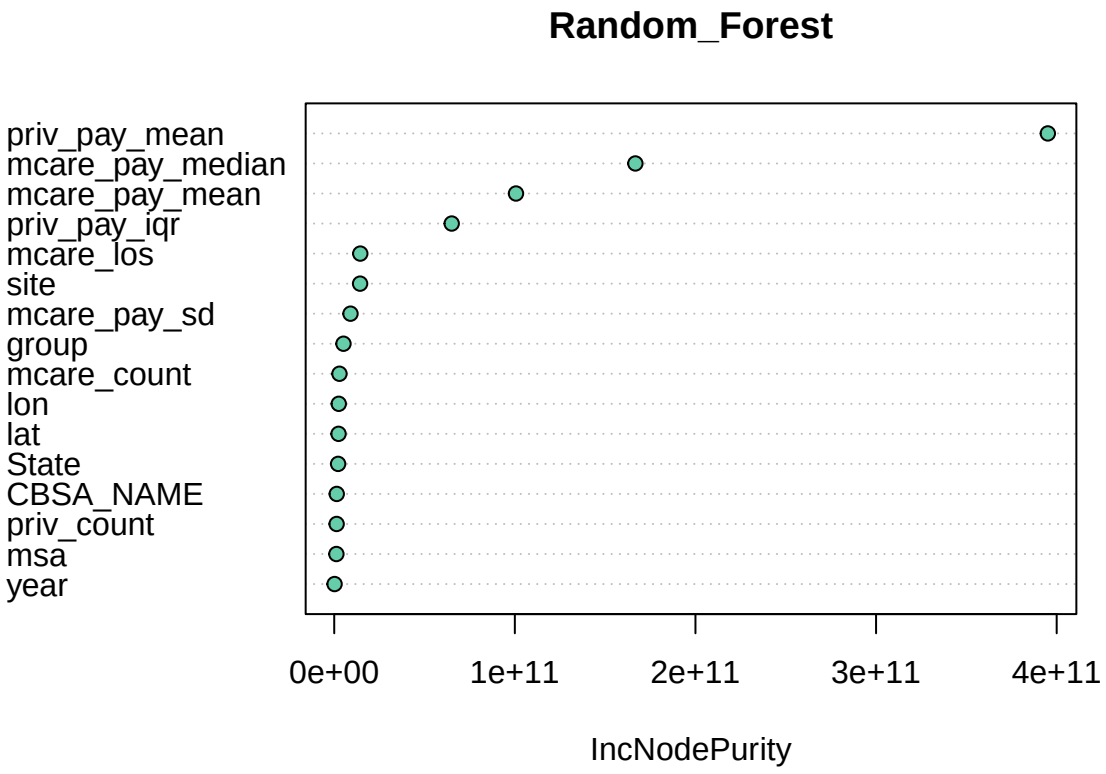
```
data <- read.csv("priv_mcare_f_pay_2022Oct18.csv")
hospital_data <- read.csv("Hospital_Master_Sheet.csv")
```

Hospital data aggregation, data split, and data filtering are now compartmentalized

```
hospitals_msa <- hospital_data %>% aggregate_hospital_features()

split_dataset <- data %>% data_split(count_thresh = 50)
working_set <- split_dataset[[1]]
predict_set <- split_dataset[[2]]

model_data <- left_join(working_set, hospitals_msa, by = "msa")  %>%
  select(-priv_pay_mean, -priv_pay_iqr, -mcare_pay_mean, -mcare_pay_sd, -Urban, -msa)

predict_data <- left_join(predict_set, hospitals_msa, by = "msa")  %>%
  select(-priv_pay_mean, -priv_pay_iqr, -mcare_pay_mean, -mcare_pay_sd, -Urban, -msa)

train_test_data <- model_data %>% train_test_split(proportion = 0.8)

train <- split_dataset[[1]]
test <- split_dataset[[2]]

# Random Forest model
set.seed(123) #Set seed for reproducibility
# Fit Random Forest Model on training data
Random_Forest <- randomForest(
  formula = priv_pay_median ~ .,
  data    = train,
  num.trees = 500,
  mtry = 7,
  nodesize = 20,
  na.action = na.omit
)

train_predict <- train %>%
  mutate(pred_priv_pay_median = predict(Random_Forest, train)) %>%
  filter(!is.na(pred_priv_pay_median))
```

```
trn_m = MAPE(train_predict$pred_priv_pay_median, train_predict$priv_pay_median)

train_mape_percent = mean(abs((train_predict$priv_pay_median - train_predict$pred_priv_pay_median)/trai
```

```
varImpPlot(Random_Forest, bg = "aquamarine3")
```




```
test_predict <- test %>%
  mutate(pred_priv_pay_median = predict(Random_Forest, test)) %>%
  filter(!is.na(pred_priv_pay_median))
tst_m = MAPE(test_predict$pred_priv_pay_median, test_predict$priv_pay_median)
test_mape_percent = mean(abs((test_predict$priv_pay_median - test_predict$pred_priv_pay_median)/test_pr
```

```
cat("With Threshold >50 claims for training set:\n")
```

```
## With Threshold >50 claims for training set:
```

```
cat("Train MAPE:" , round(train_mape_percent, 2), "%\n")
```

```
## Train MAPE: 8.05 %
```

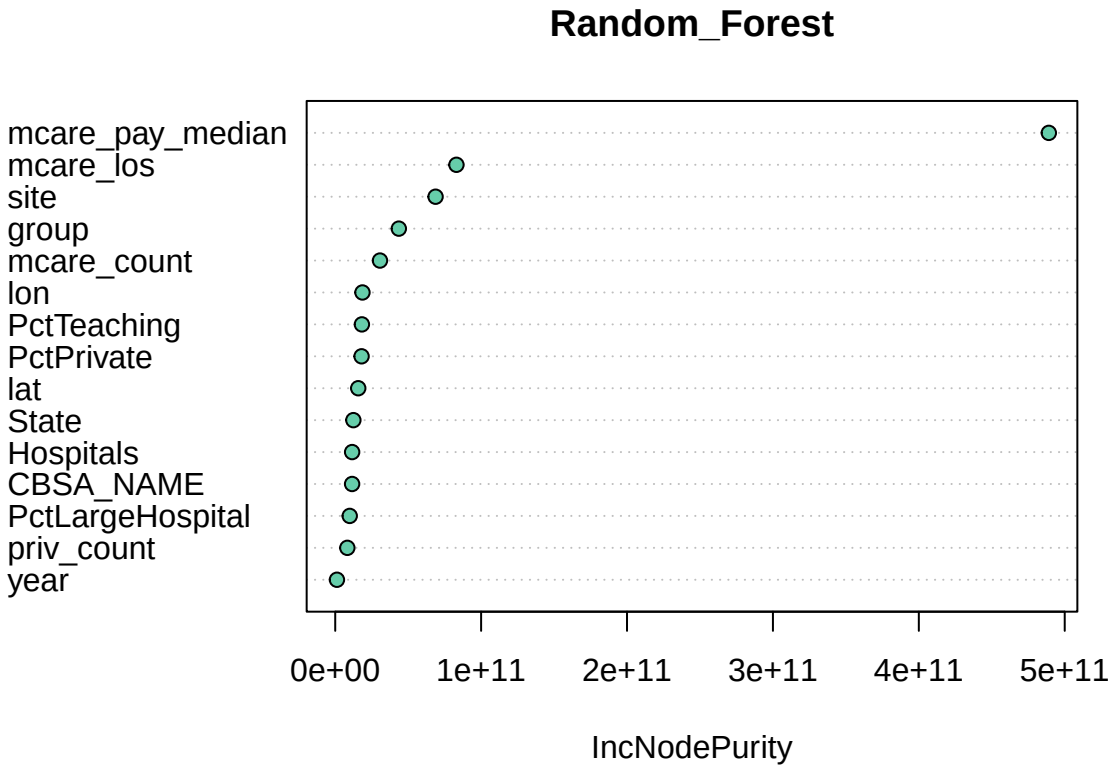
```
cat("Test MAPE:" , round(test_mape_percent, 2), "%\n")
```

```
## Test MAPE: Inf %
```

Modeling with threshold 35 number of claims

```r
data <- read.csv("priv_mcare_f_pay_2022Oct18.csv")
hospital_data <- read.csv("Hospital_Master_Sheet.csv")

hospitals_msa <- hospital_data %>%
  group_by(MSA_CD) %>%
  summarise(Hospitals = n(),
            PctTeaching = sum(teaching == "YES")/n(),
            PctLargeHospital = sum(beds_grp == "500+")/n(),
            Urban = ifelse(sum(urban_rural == "URBAN")/n() == 1, "Urban","Rural"),
            PctPrivate = sum(ownership == "PRIVATE (NOT FOR PROFIT)" | ownership == "PRIVATE (FOR PROFI
  rename(msa = MSA_CD)

new_data <- data %>%
  filter(priv_pay_median >= 0 | is.na(priv_pay_median)) %>%
  filter(priv_count != 0)

new_data_with_hospital <- left_join(new_data, hospitals_msa, by = "msa")


model_data <- new_data_with_hospital %>%
  filter(priv_count >= 35) %>%
  filter(!is.na(priv_pay_median)) %>%
  select(-priv_pay_mean, -priv_pay_iqr, -mcare_pay_mean, -mcare_pay_sd,-Urban, -msa)

predict_data <- new_data_with_hospital %>%
  filter(priv_count <= 10) %>%
  filter(priv_pay_median > 0)

set.seed(123) #Set seed for reproducible analysis
dt = sort(sample(nrow(model_data), nrow(model_data)*.8)) #Split data
train <-model_data[dt,] #80% training data
test <-model_data[-dt,] #20% test data



# Random Forest model
set.seed(123) #Set seed for reproducibility
# Fit Random Forest Model on training data
Random_Forest <- randomForest(
  formula = priv_pay_median ~ .,
  data    = train,
  num.trees = 500,
  mtry = 7,
  nodesize = 20,
  na.action = na.omit
)

train_predict <- train %>%
  mutate(pred_priv_pay_median = predict(Random_Forest, train)) %>%
  filter(!is.na(pred_priv_pay_median))
trn_m = MAPE(train_predict$pred_priv_pay_median, train_predict$priv_pay_median)
train_mape_percent = mean(abs((train_predict$priv_pay_median - train_predict$pred_priv_pay_median)/trai
```

```
varImpPlot(Random_Forest, bg = "aquamarine3")
```




```
test_predict <- test %>%
  mutate(pred_priv_pay_median = predict(Random_Forest, test)) %>%
  filter(!is.na(pred_priv_pay_median))
tst_m = MAPE(test_predict$pred_priv_pay_median, test_predict$priv_pay_median)
test_mape_percent = mean(abs((test_predict$priv_pay_median - test_predict$pred_priv_pay_median)/test_pr
```

```
cat("With Threshold >35 claims for training set:\n")
```

```
## With Threshold >35 claims for training set:
```

```
cat("Train MAPE:" , round(train_mape_percent, 2), "%\n")
```

```
## Train MAPE: 15.76 %
```

```
cat("Test MAPE:" , round(test_mape_percent, 2), "%\n")
```

```
## Test MAPE: 21.23 %
```