Introducing {workboots}

Generate prediction intervals from tidymodel workflows

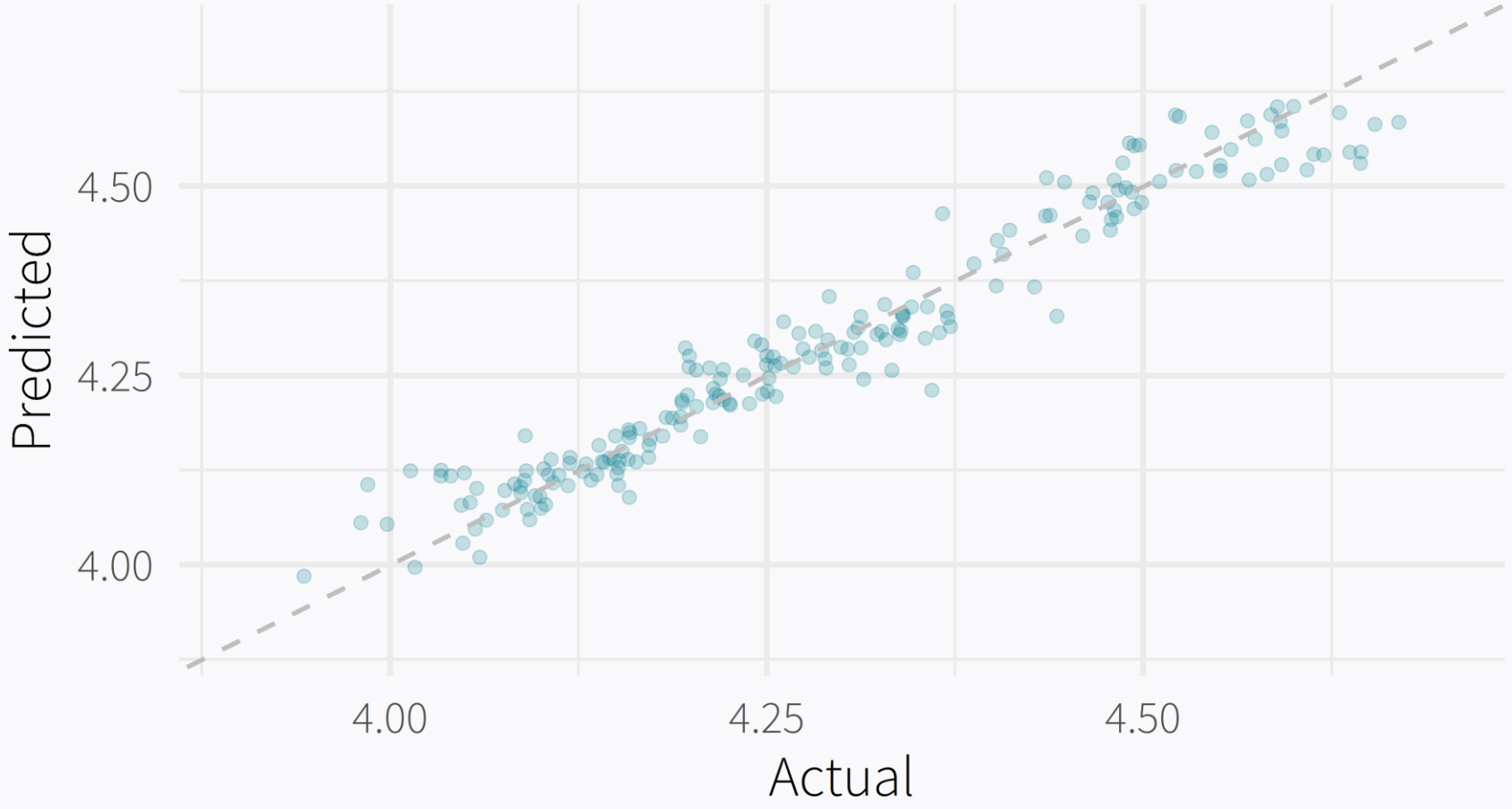
Mark Rieke
@markjrieke
2022-07-27


BURNDY
PATRIOT® PAT750XT-18V
HIGH PERFORMANCE


AUTOMAXX

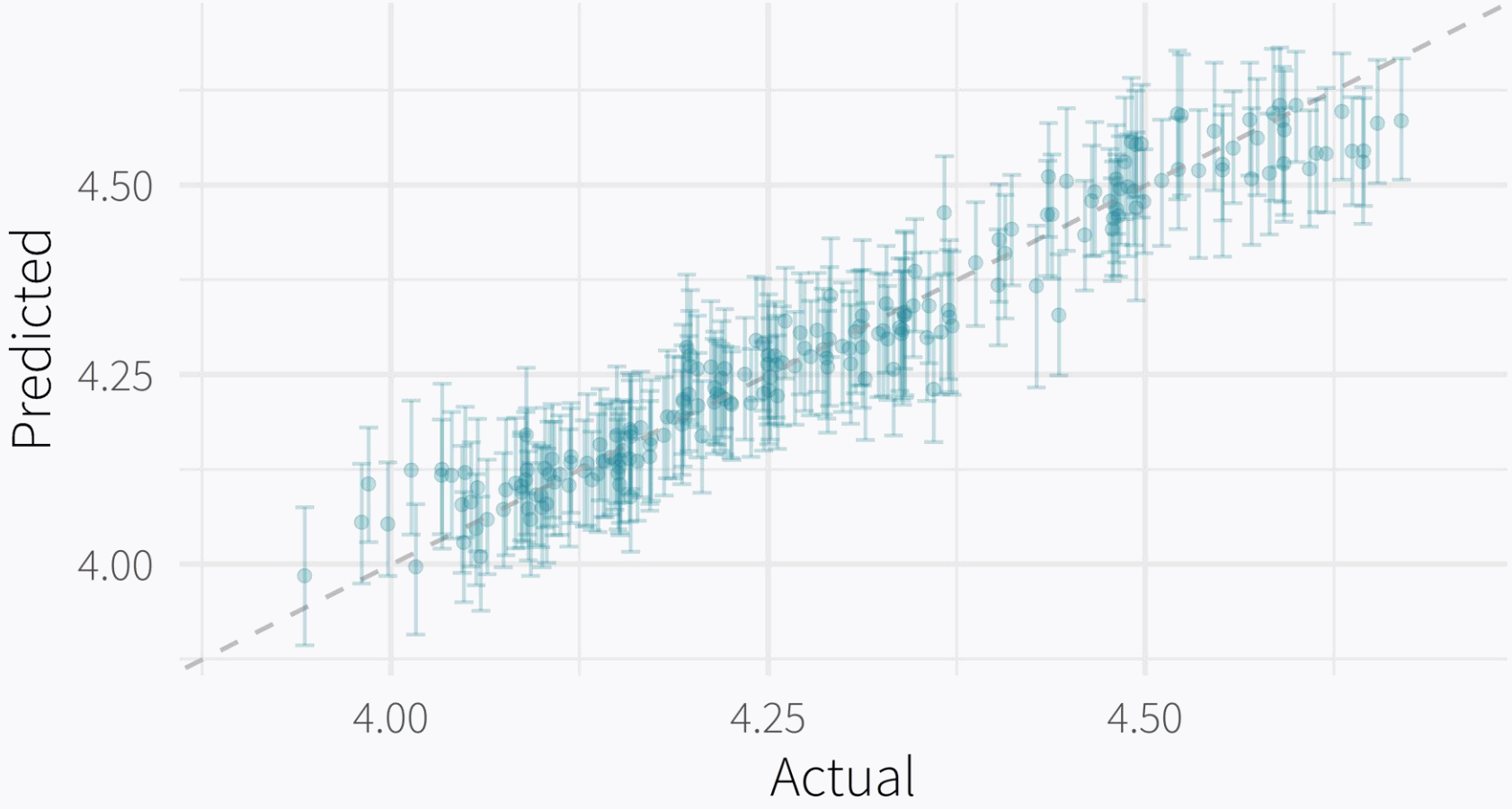
A model without {workboots}

On its own, XGBoost can only generate point predictions



A model with {workboots}

With workboots, we can generate prediction intervals!



Motivation/Background

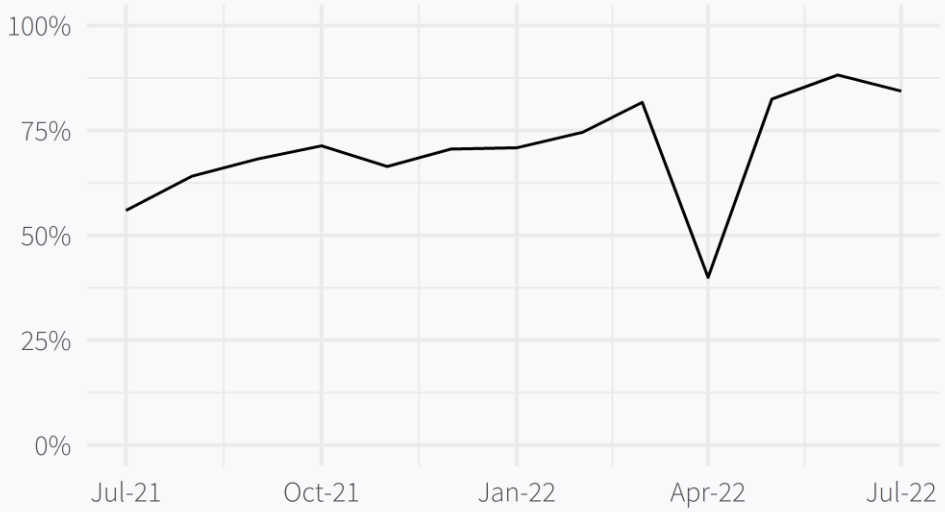
about me

MEMORIAL HERMANN

- Sr. CX Analyst, MHHS
- Patient satisfaction survey data
- Administer, understand, improve
- Answering questions

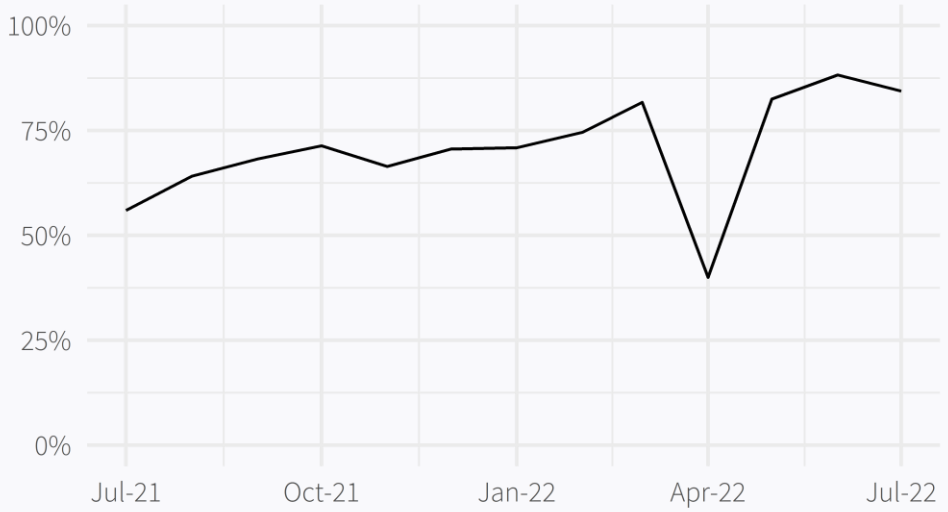
Monthly Patient Satisfaction Scores

Why the large drop in April?



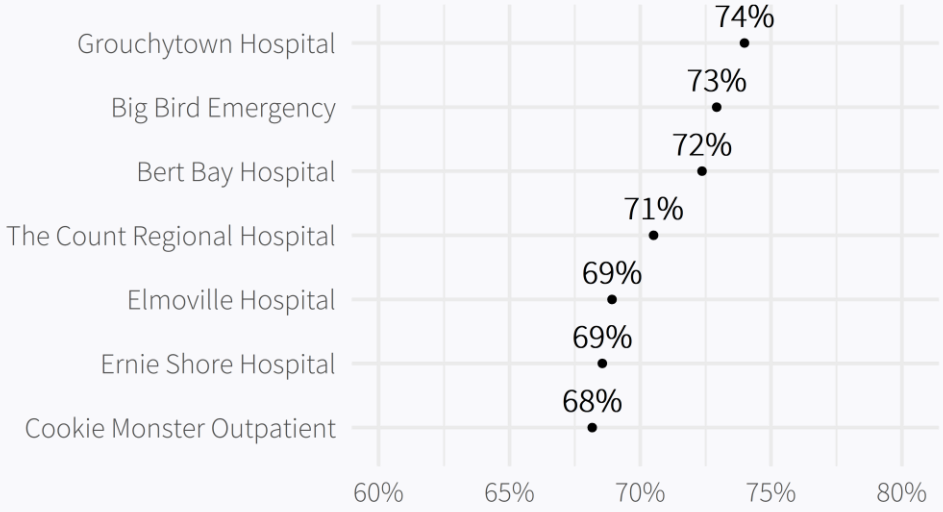
Monthly Patient Satisfaction Scores

Why the large drop in April?



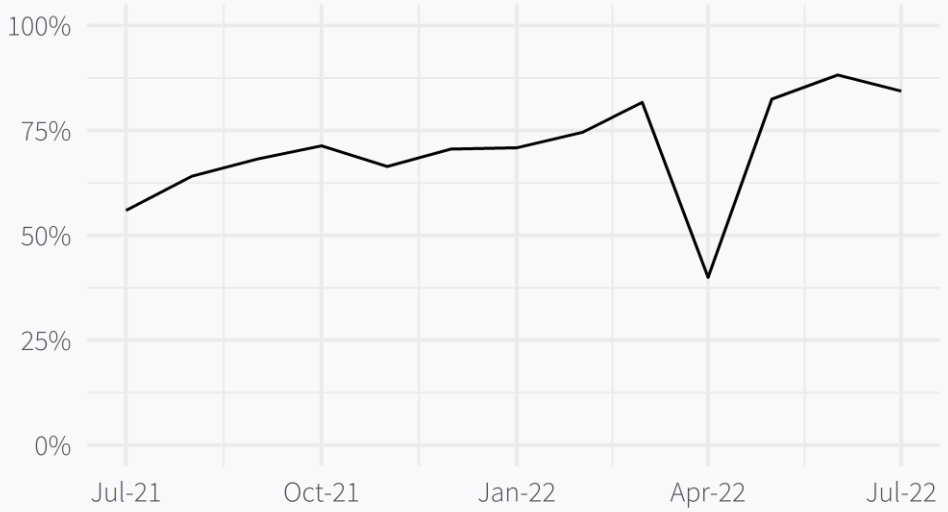
Patient Satisfaction Scores by Hosptial

Are these real differences across the hospitals?



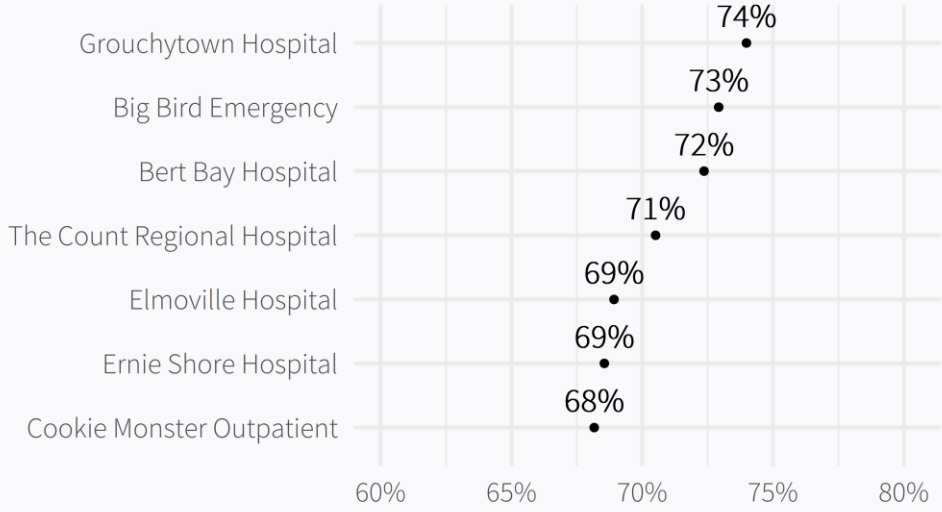
Monthly Patient Satisfaction Scores

Why the large drop in April?



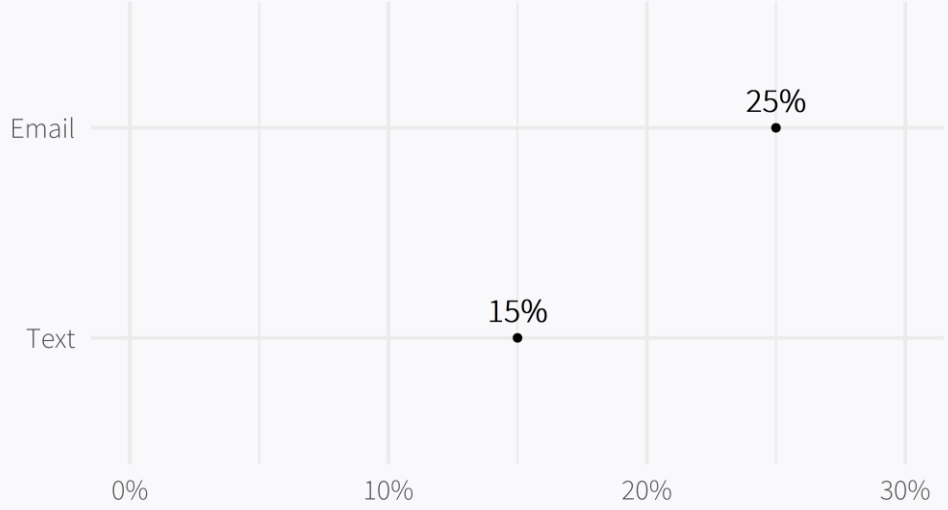
Patient Satisfaction Scores by Hosptial

Are these real differences across the hospitals?



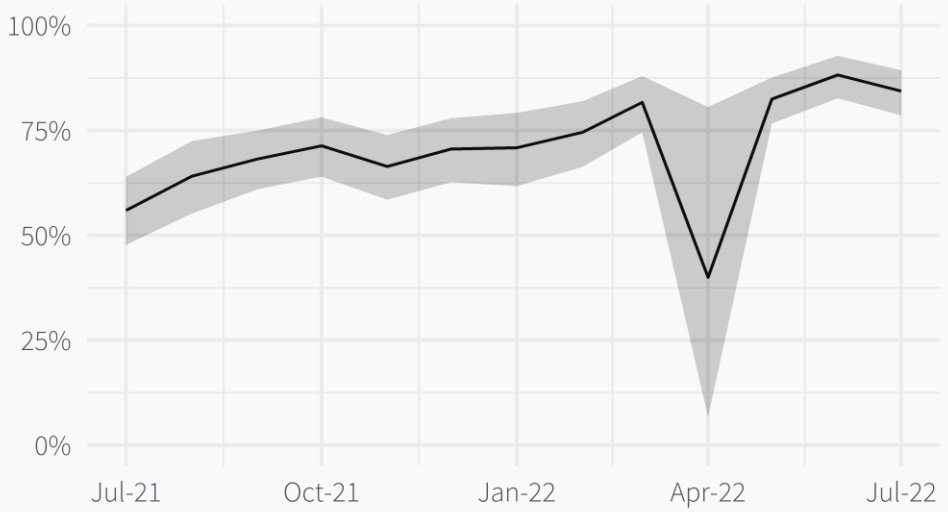
Response Rates by Distribution Method

Which method gets more responses?



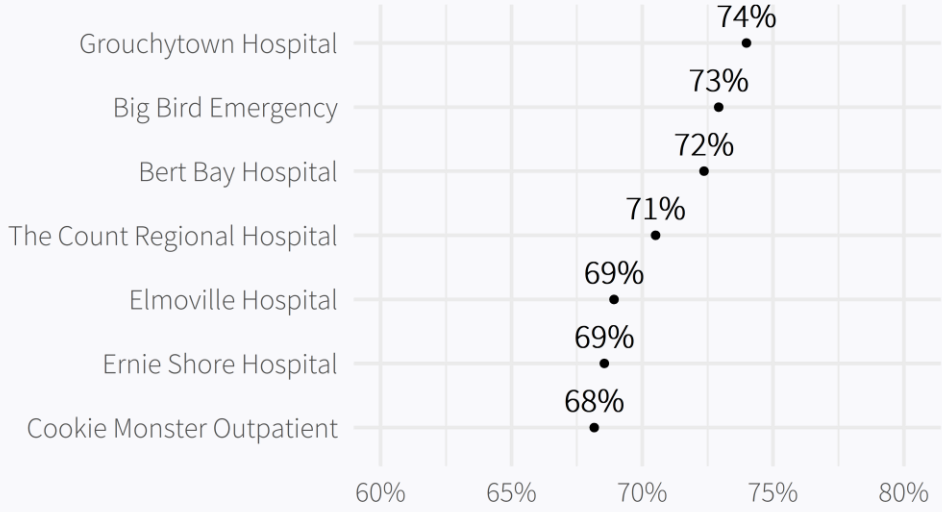
Monthly Patient Satisfaction Scores

Why the large drop in April?



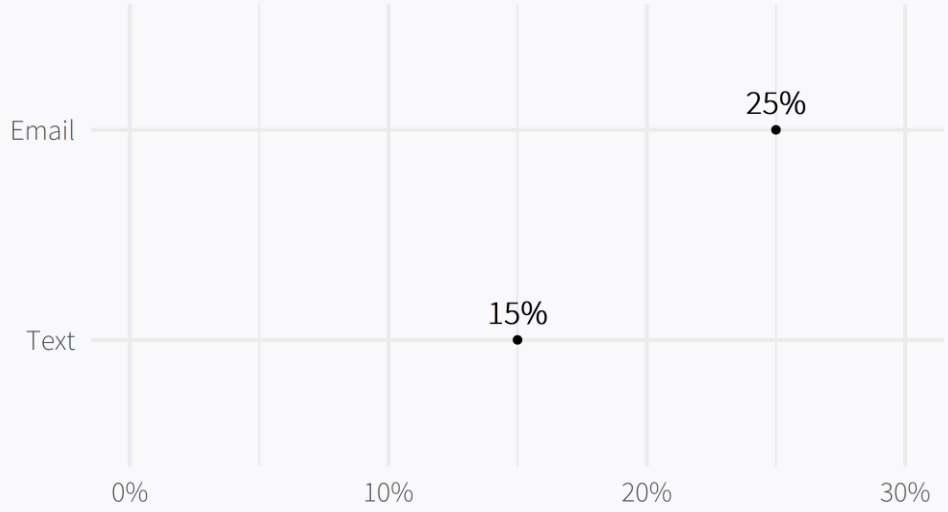
Patient Satisfaction Scores by Hosptial

Are these real differences across the hospitals?



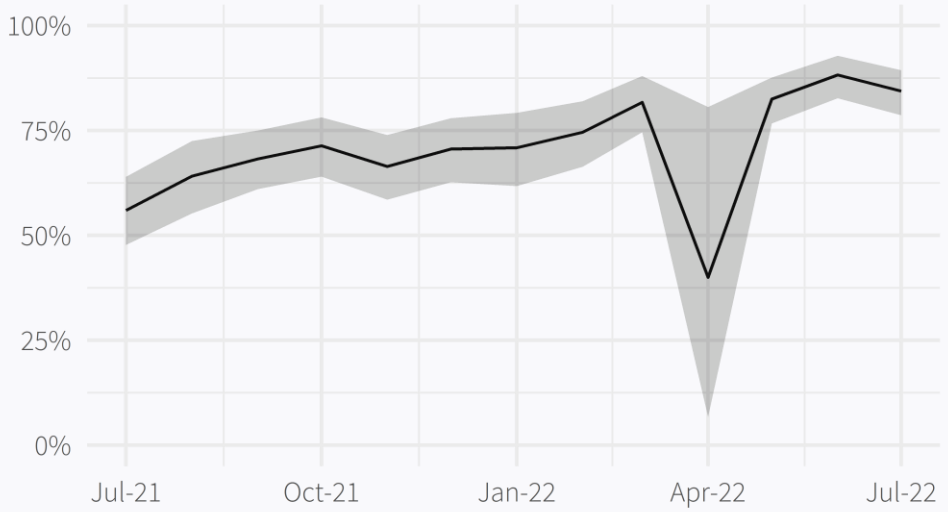
Response Rates by Distribution Method

Which method gets more responses?



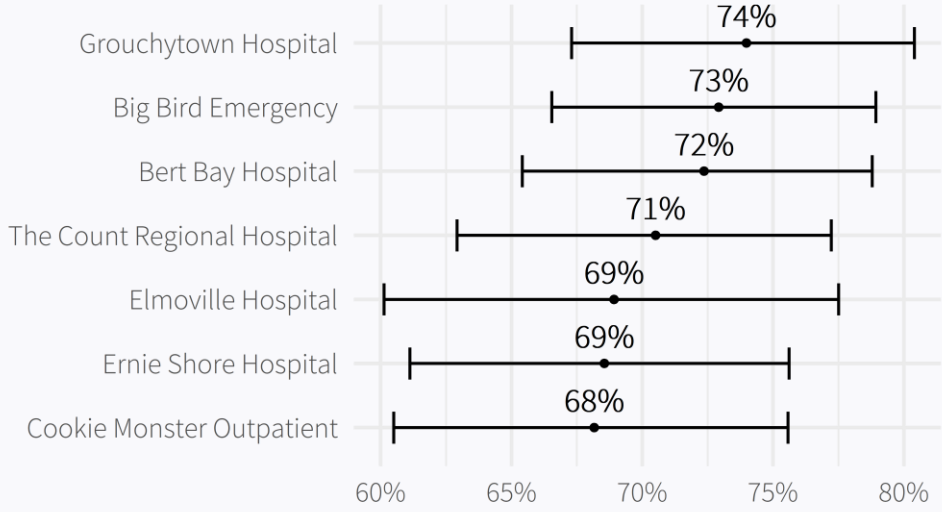
Monthly Patient Satisfaction Scores

Why the large drop in April?



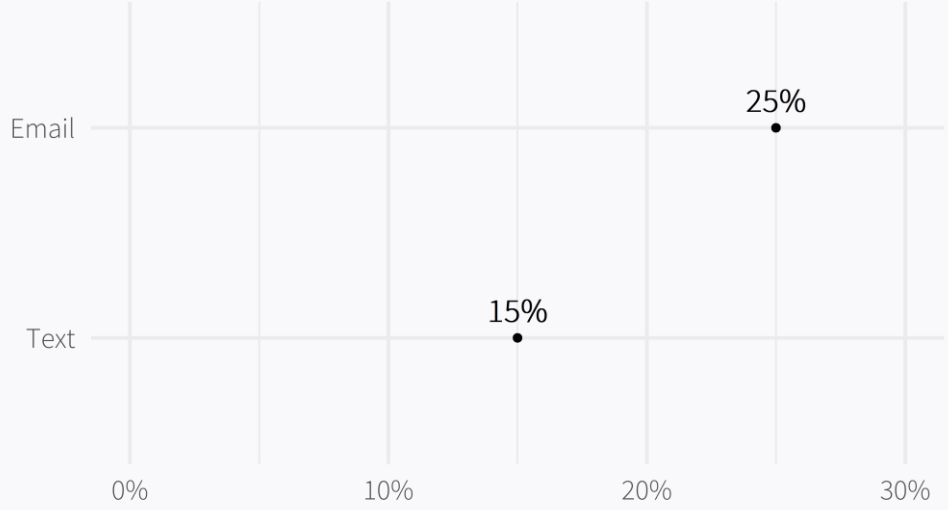
Patient Satisfaction Scores by Hosptial

Are these real differences across the hospitals?



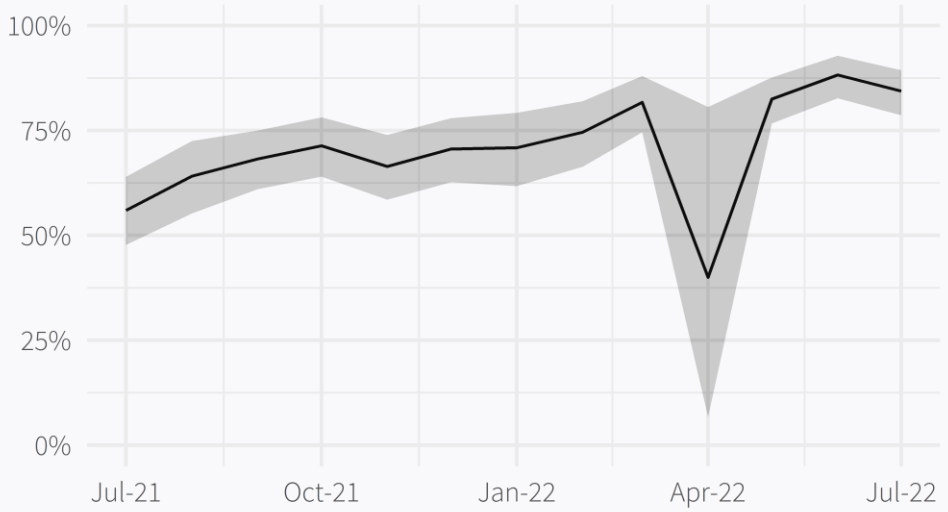
Response Rates by Distribution Method

Which method gets more responses?



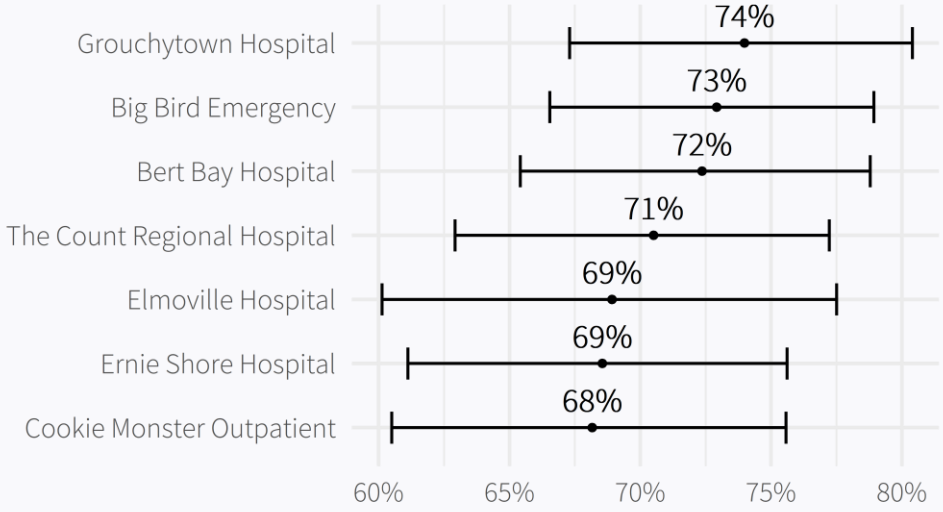
Monthly Patient Satisfaction Scores

Why the large drop in April?



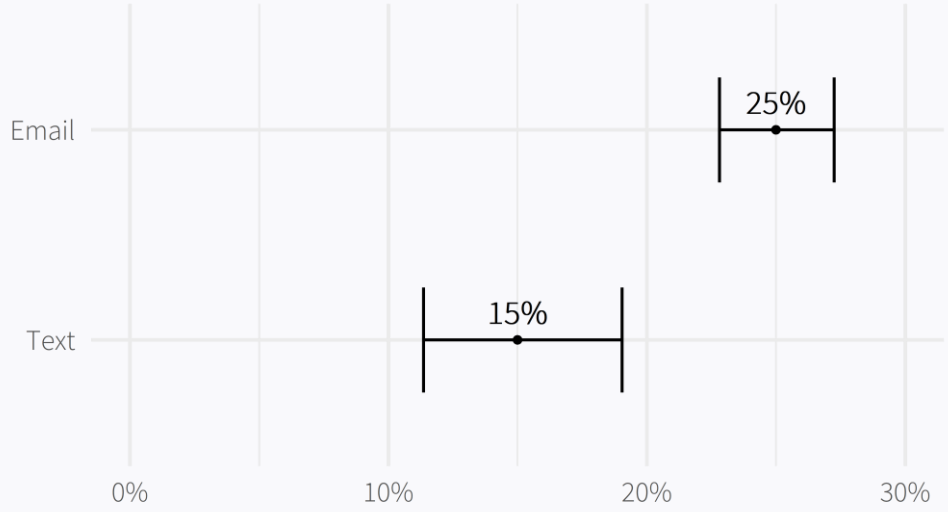
Patient Satisfaction Scores by Hosptial

Are these real differences across the hospitals?



Response Rates by Distribution Method

Which method gets more responses?



5.0 (19)

★★★★★

4.6 (2,280)

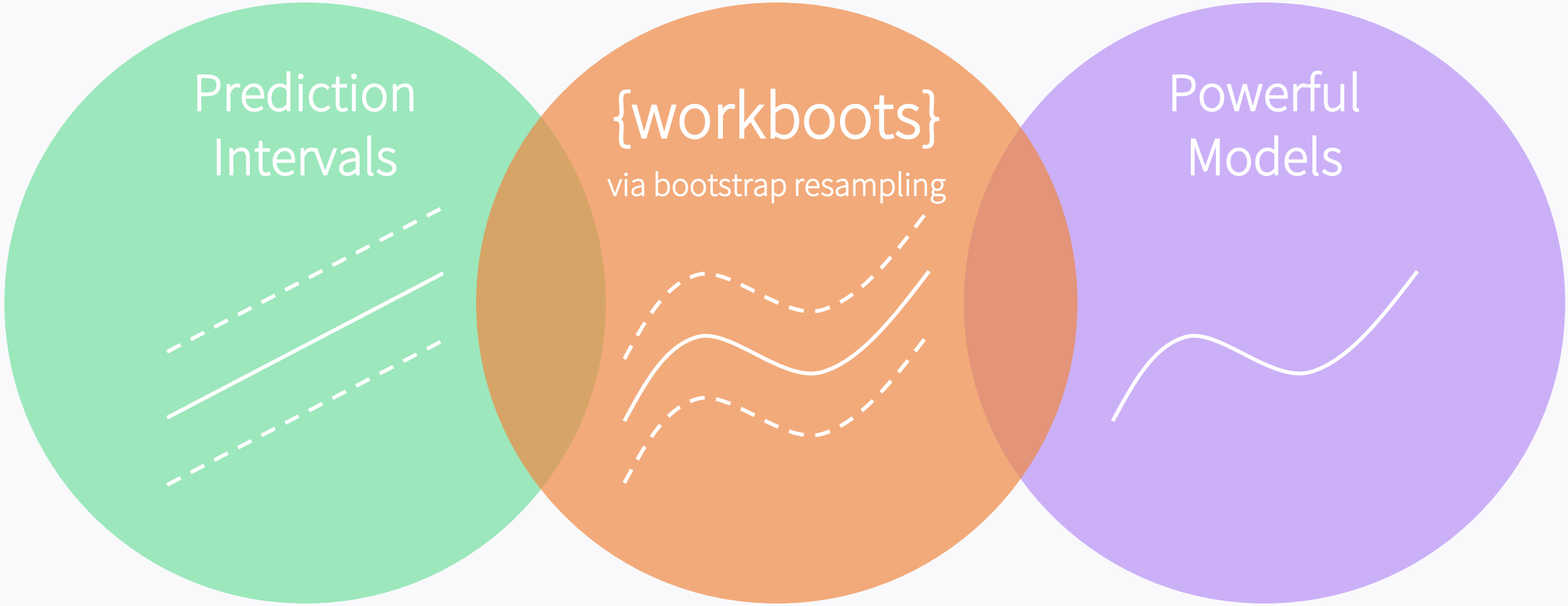
★★★★½

What are the drivers of patient satisfaction?

What will our score be?

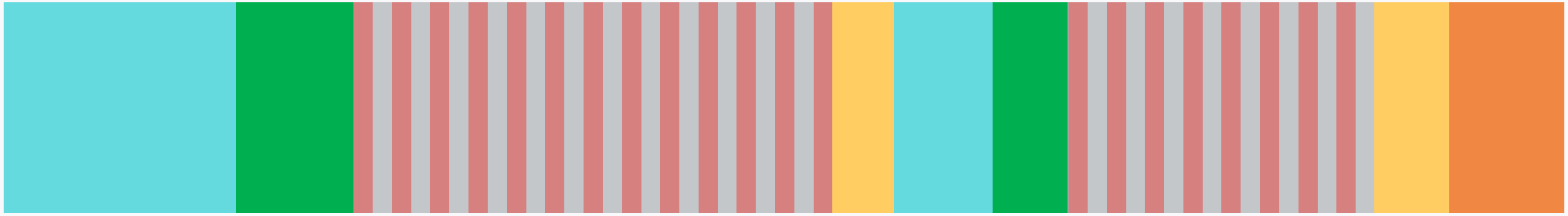
How many responses are we expecting?


XGBoost
me
lm()


Prediction Intervals
{workboots}
via bootstrap resampling
Powerful Models

Package Overview

model building timeline



Exploratory Data Analysis

Model Fit

Model Evaluation

Feature Engineering

Model Tuning

Model Finalization

model building timeline

workboots

Exploratory Data Analysis

Model Fit

Model Evaluation

Feature Engineering

Model Tuning

Model Finalization

how this works

1. Generate resamples
2. Fit many models to resamples
3. Predict on new data
4. Summarize prediction range

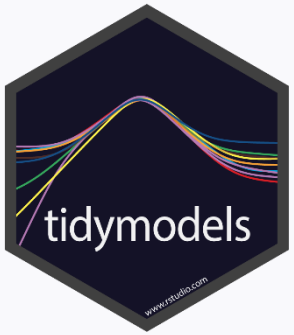
workboots checklist

- ✓ Normally distributed residuals
- ✓ Time to run
- ✓ Actually need powerful model

using workboots

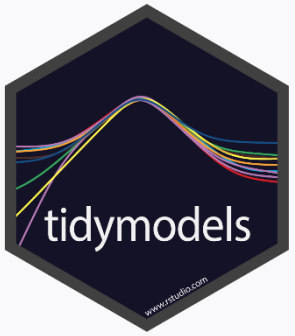
1. Set up a workflow
2. Generate predictions
3. Summarise results

set up a workflow



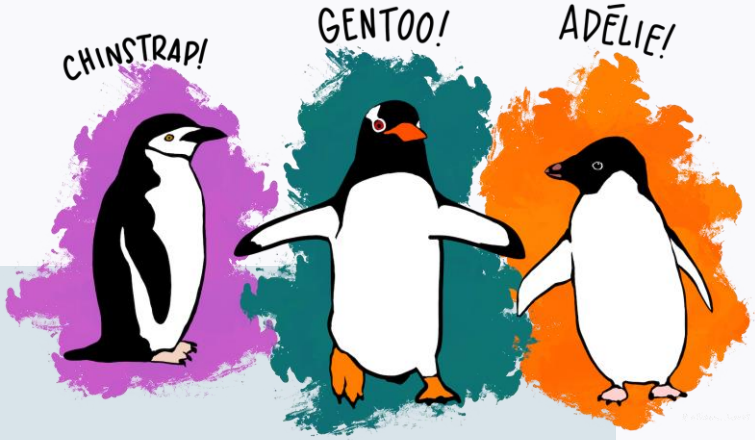

```
library(tidymodels)
```

set up a workflow



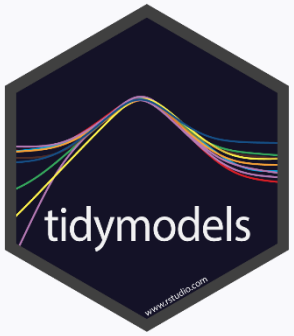

```
library(tidymodels)

# setup our data
data("penguins")
penguins <- penguins %>% drop_na()
```

set up a workflow



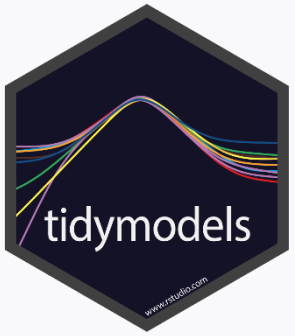

```
penguins %>% glimpse()

#> Rows: 333
#> Columns: 7
#> $ species           <fct> Adelie, Adelie, Adelie, ...
#> $ island            <fct> Torgersen, Torgersen, To...
#> $ bill_length_mm    <dbl> 39.1, 39.5, 40.3, 36.7, ...
#> $ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, 19.3, ...
#> $ flipper_length_mm <int> 181, 186, 195, 193, 190,...
#> $ body_mass_g       <int> 3750, 3800, 3250, 3625, ...
#> $ sex               <fct> male, female, female, fe...
```

set up a workflow



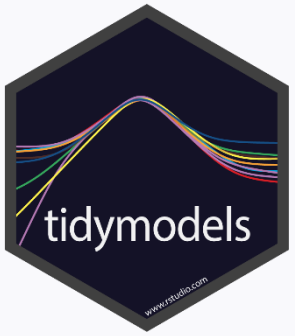

```
library(tidymodels)

# setup our data
data("penguins")
penguins <- penguins %>% drop_na()

set.seed(123)
penguins_split <- initial_split(penguins)
penguins_test <- testing(penguins_split)
penguins_train <- training(penguins_split)
```

set up a workflow



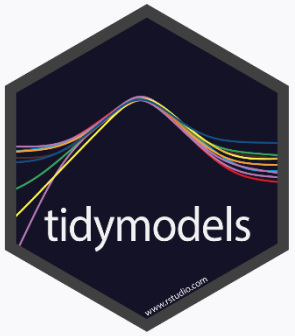

```
# setup a basic preprocessing recipe
penguins_rec <-
  recipe(body_mass_g ~ ., data = penguins_train) %>%
  step_dummy(all_nominal())
```

set up a workflow



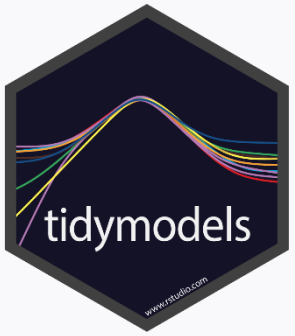

```
# setup a basic preprocessing recipe
penguins_rec <-
  recipe(body_mass_g ~ ., data = penguins_train) %>%
  step_dummy(all_nominal())
```

set up a workflow




```
# setup a basic preprocessing recipe
penguins_rec <-
  recipe(body_mass_g ~ ., data = penguins_train) %>%
  step_dummy(all_nominal())

# put together a workflow
penguins_wf <-
  workflow() %>%
  add_recipe(penguins_rec) %>%
  add_model(boost_tree("regression"))
```

set up a workflow




```
# setup a basic preprocessing recipe
penguins_rec <-
  recipe(body_mass_g ~ ., data = penguins_train) %>%
  step_dummy(all_nominal())

# put together a workflow
penguins_wf <-
  workflow() %>%
  add_recipe(penguins_rec) %>%
  add_model(boost_tree("regression"))
```

time for workboots!

generate predictions


```
library(workboots)
```

generate predictions


```
library(workboots)

# generate a bootstrap prediction interval
set.seed(345)
penguins_preds <-
  penguins_wf %>%
  predict_boots(
    n = 2000,
    training_data = penguins_train,
    new_data = penguins_test,
    interval = "prediction"
  )
```

generate predictions


```
library(workboots)

# generate a bootstrap prediction interval
set.seed(345)
penguins_preds <-
  penguins_wf %>%
  predict_boots(
    n = 2000,
    training_data = penguins_train,
    new_data = penguins_test,
    interval = "prediction"
  )
```

generate predictions


```
library(workboots)

# generate a bootstrap prediction interval
set.seed(345)
penguins_preds <-
  penguins_wf %>%
  predict_boots(
    n = 2000,
    training_data = penguins_train,
    new_data = penguins_test,
    interval = "prediction"
  )
```

generate predictions


```
library(workboots)

# generate a bootstrap prediction interval
set.seed(345)
penguins_preds <-
  penguins_wf %>%
  predict_boots(
    n = 2000,
    training_data = penguins_train,
    new_data = penguins_test,
    interval = "prediction"
  )
```

generate predictions


```
penguins_preds
```

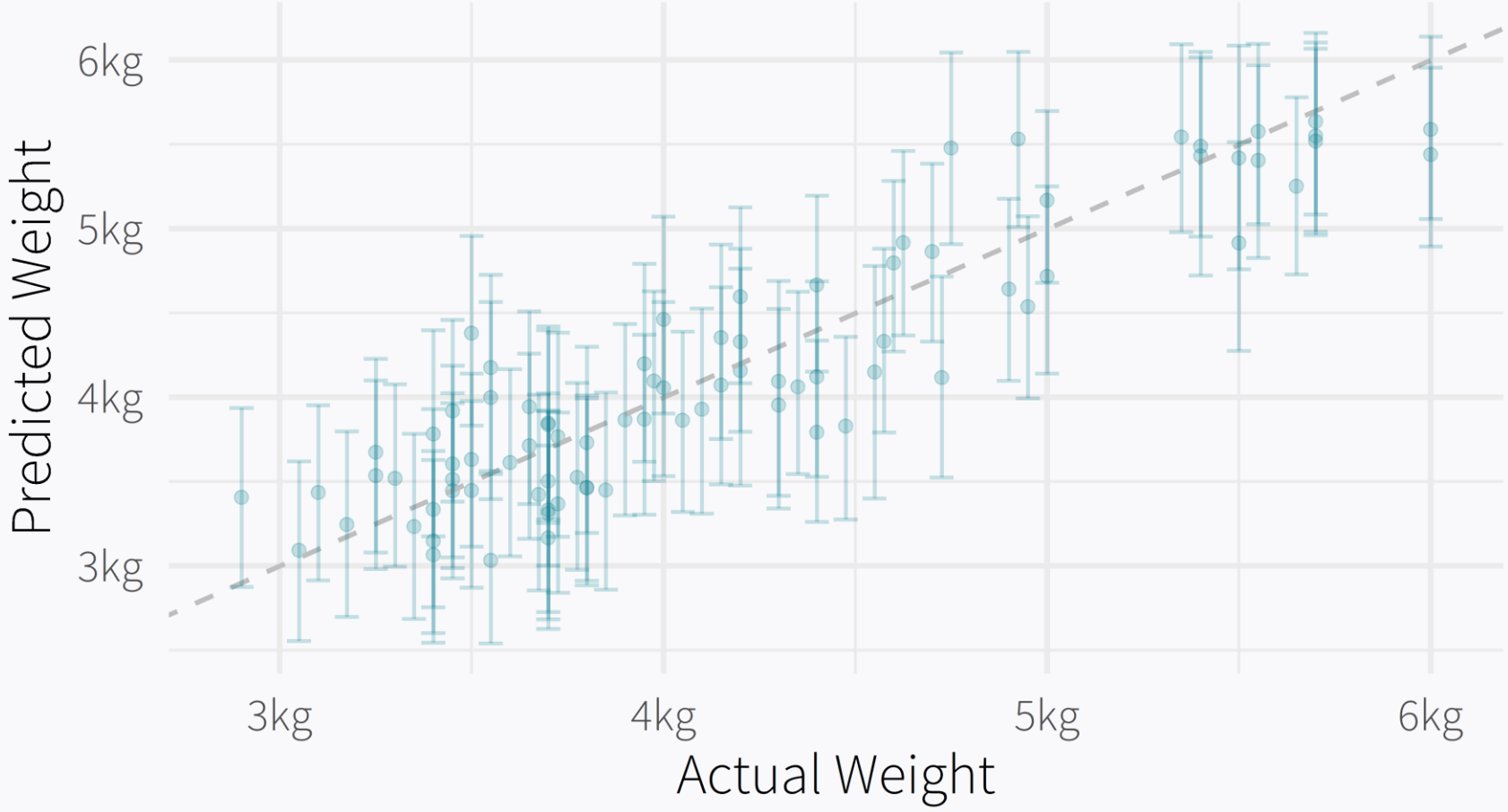
```
#> # A tibble: 84 x 5
#>     rowid .preds
#>     <int> <list>
#> 1       1 <tibble [2,000 x 2]>
#> 2       2 <tibble [2,000 x 2]>
#> 3       3 <tibble [2,000 x 2]>
#> 4       4 <tibble [2,000 x 2]>
#> 5       5 <tibble [2,000 x 2]>
#> # ... with 79 more rows
```

summarise results


```
penguins_preds %>%
  summarise_predictions() %>% select(-.preds)

#> # A tibble: 84 x 5
#>     rowid .pred .pred_lower .pred_upper
#>     <int> <dbl>       <dbl>       <dbl>
#> 1       1 3465.       2913.       3994.
#> 2       2 3535.       2982.       4100.
#> 3       3 3604.       3050.       4187.
#> 4       4 4157.       3477.       4764.
#> 5       5 3868.       3305.       4372.
#> # ... with 79 more rows
```

Predicting Palmer Penguin Pounds

workboots allows us to generate prediction intervals



generate predictions


```
library(workboots)

# generate a bootstrap prediction interval
set.seed(345)
penguins_preds <-
  penguins_wf %>%
  predict_boots(
    n = 2000,
    training_data = penguins_train,
    new_data = penguins_test,
    interval = "prediction"
  )
```

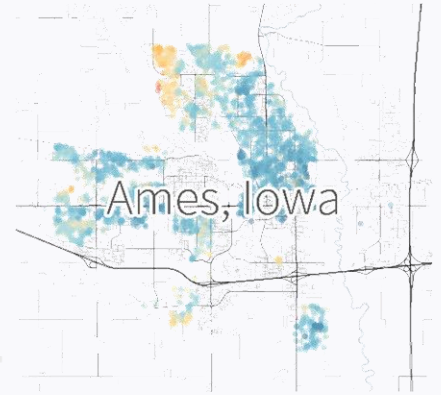
generate predictions

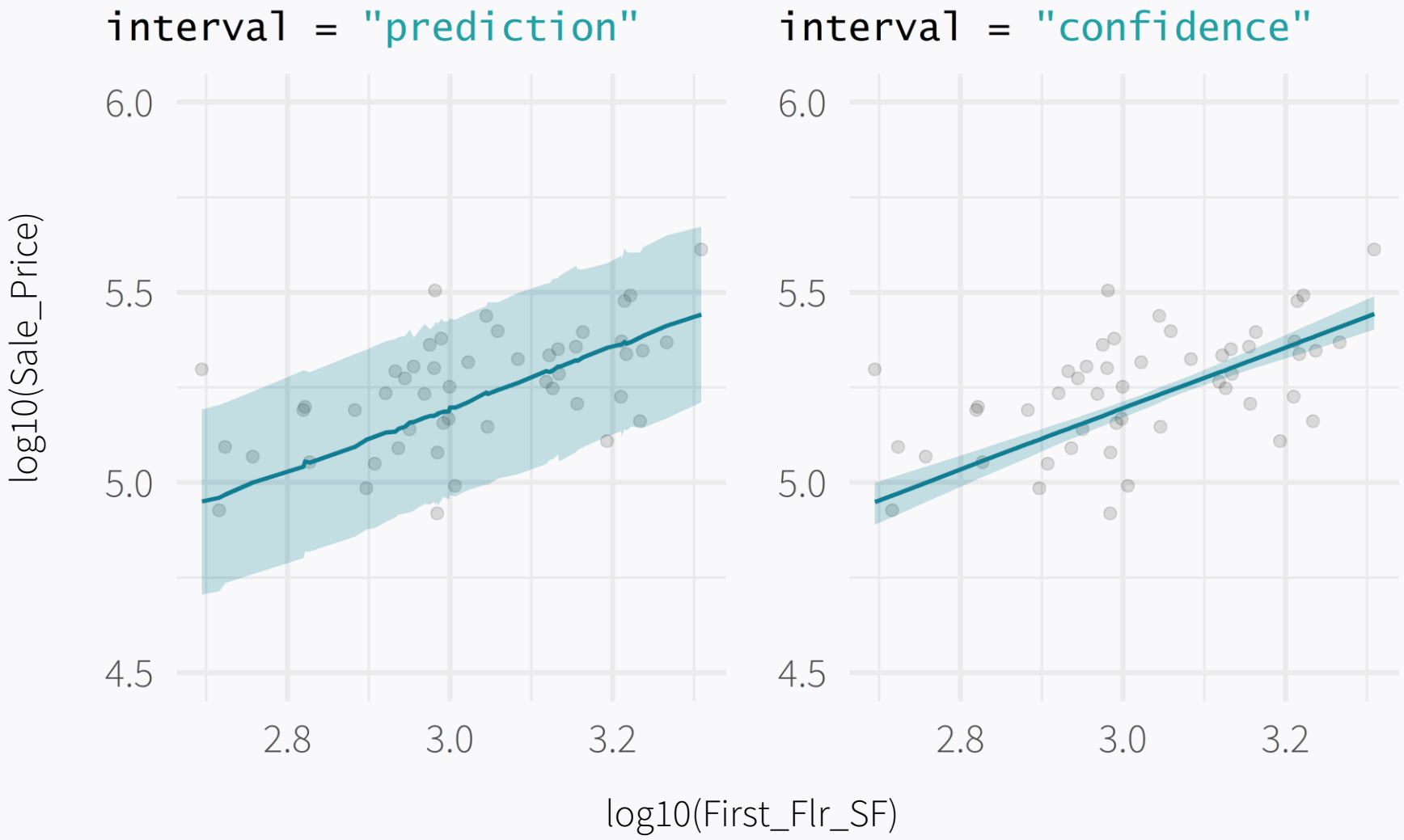

```
library(workboots)

# generate a bootstrap prediction interval
set.seed(345)
penguins_preds <-
  penguins_wf %>%
  predict_boots(
    n = 2000,
    training_data = penguins_train,
    new_data = penguins_test,
    interval = "prediction"
  )
```

generate predictions


```
library(workboots)

# generate a bootstrap prediction interval
set.seed(345)
penguins_preds <-
  penguins_wf %>%
  predict_boots(
    n = 2000,
    training_data = penguins_train,
    new_data = penguins_test,
    interval = "confidence"
  )
```

{workboots} can generate multiple interval types





generate importances


```
# generate a bootstrap variable importances
set.seed(345)
penguins_vi <-
  penguins_wf %>%
  vi_boots(
    n = 2000,
    training_data = penguins_train
  )
```

generate importances

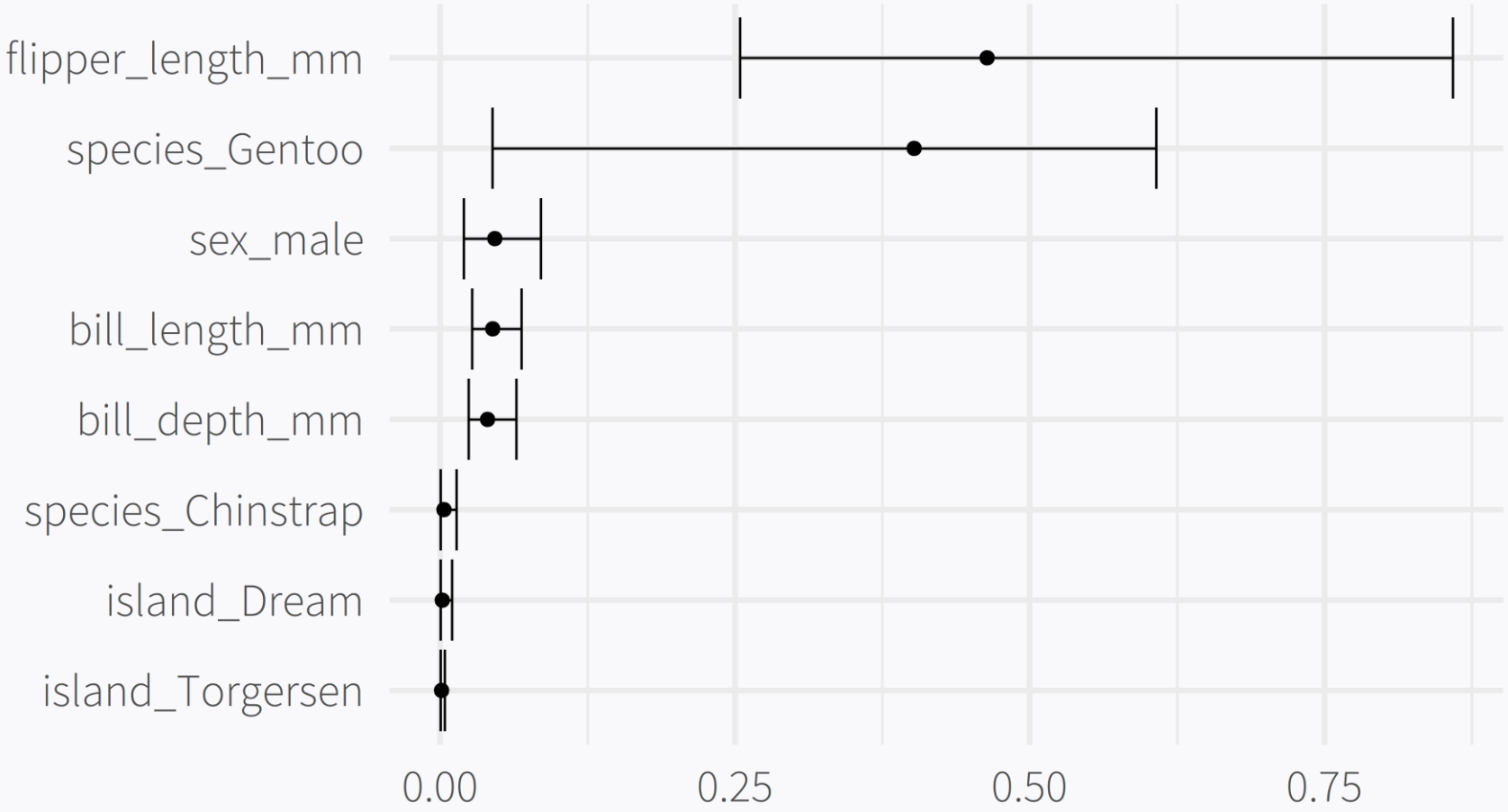

```
penguins_vi

#> # A tibble: 8 x 2
#>   variable           .importances
#>   <int>              <list>
#> 1 flipper_length_mm <vi [2,000 x 2]>
#> 2 species_Gentoo    <vi [1,928 x 2]>
#> 3 bill_length_mm    <vi [2,000 x 2]>
#> 4 bill_depth_mm     <vi [2,000 x 2]>
#> 5 sex_male          <vi [2,000 x 2]>
#> # ... With 3 more rows
```

summarise importances


```
penguins_vi %>%
  summarise_importance() %>% select(-.importances)

#> # A tibble: 8 x 2
#>   variable           .importance .importance_lower
#>   <int>                    <dbl>               <dbl>
#> 1 flipper_length_mm        0.464               0.254
#> 2 species_Gentoo           0.402               0.044
#> 3 bill_length_mm           0.045               0.027
#> 4 bill_depth_mm            0.040               0.024
#> 5 sex_male                 0.046               0.020
#> # ... with 3 more rows, and 1 more variable:
#> #     .importance_upper <dbl>
```

Bootstrap estimations of variable importance

Uses vip::vi() under the hood



give it a shot!

- Specific tool, specific job
- Generate prediction intervals
- Use any model type

thank you!

connect with me!

Social

- twitter: @markjrieke
- github: github.com/markjrieke
- Linkedin: linkedin.com/in/mark-j-rieke-ab4b0ab4

additional resources

Packages

- ngboostForecast : Probabilistic forecasting with Python's ngboost [CRAN/github]
- ggdist : Visualizations of distributions and uncertainty [CRAN/github]
- spin : Functions for simulating prediction intervals [github]
- ungeviz : Tools for visualizing uncertainty with ggplot2 [github]

Reading

- Bootstrap Methods and their Application, (Davison and Hinkley)
- Improvements on Cross Validation – the .632+ Bootstrap Method, (Efron and Tibshirani)
- Tidy Modeling with R (Kuhn and Silge)
- Applied Predictive Modeling (Kuhn and Johnson)

Backup