

# BDA - Project

Jan Nyberg, Carl-Victor Schauman

4/12/2022

## Contents

Introduction	2
Description of the data	2
Description of models	2
Priors used	2
Rstan code	2
Running of stan model	4
Convergence diagnostics	7
Posterior predictive checks	7
Predictive performance assessment	7
Sensitivity analysis	11
Model comparison	11
Discussion of issues and potential improvements	12
Conclusion what was learned from the data analysis	12
Self-reflection	12

## Introduction

For this project, we are trying to predict Lewis Hamilton's average score in a season based on previous years scores. This is mostly just due to our curiosity if we are able to use this to somehow predict the score. We are modeling his scores from five years, and trying to build a model using it. We want to see what kind of distribution the answer will be and how well it is able to estimate the following year. There are many factors we don't take into consideration, but we hope to see relatively good results and predictions.

## Description of the data

The data we use is from Kaggle and can be found [here](#). We took Lewis Hamilton out of the data and chose the years 2015-2019. We selected his scores from all the races from those years. One thing to note with the data is that every year doesn't have an equal amount of races. To account for this we chose to fill in the missing races with the median for the year. This makes the data a bit inaccurate, however, it shouldn't have too big an effect on the data.

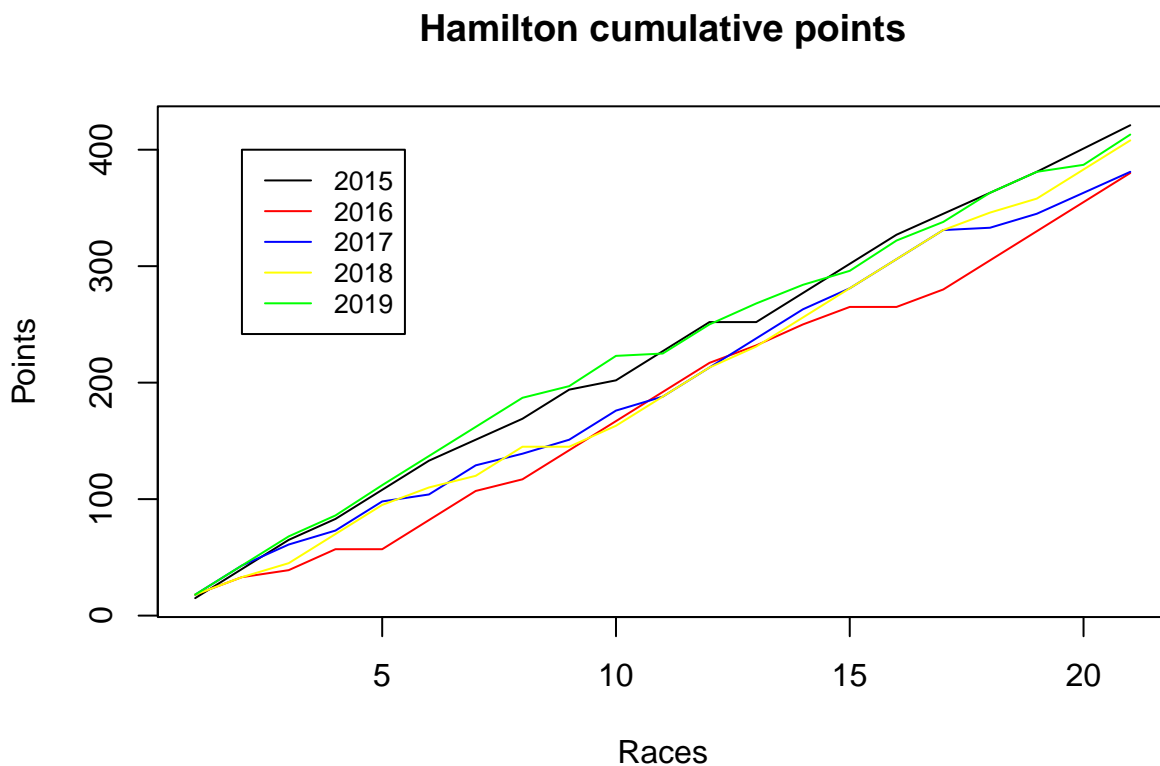


Figure 1: The cumulative points Hamilton got during the seasons 2015-2019.

## Description of models

### Priors used

### Rstan code

### Hierarchical stan model

Below is the code for the hierarchical model for Hamilton's points.

```

data {
  int<lower=0> N;
  int<lower=0> J;
  vector[J] y[N];
  real<lower=0> mu_s;
  real<lower=0> sigma_prior;
}

parameters {
  real<lower=0> mu;
  real<lower=0> sigma;
  real<lower=0> tau;
  vector[J] mus;
}

model {
  mu ~ normal(0, mu_s);
  tau ~ inv_chi_square(sigma_prior);
  sigma ~ gamma(1,1);
  mus ~ normal(mu, tau);
  for (j in 1:J)
    y[,j] ~ normal(mus[j], sigma);
}

generated quantities {
  vector[J] log_lik[N];
  real ypred;
  real ypred_6;
  ypred = normal_rng(mus[5], sigma);
  ypred_6 = normal_rng(mu, sigma);
  for (j in 1:J){
    for (n in 1:N){
      log_lik[n,j] = normal_lpdf(y[n,j] | mus[j], sigma);
    }
  }
}

```

## Non-hierarchical stan model

```

data {
  int<lower=0> N;
  int<lower=0> J;
  vector[N*J] y;
  real mean_mu;
  real<lower=0> mean_sigma;
}

parameters {
  real mu;
  real<lower=0> sigma;
}

model {
  // prior

```

```

mu ~ normal(mean_mu, mean_sigma);
sigma ~ inv_chi_square(mean_sigma);
// likelihood
y ~ normal(mu, sigma);
}
generated quantities {
  real ypred;

  // Distribution based on all seasons
  ypred = normal_rng(mu, sigma);
}

```

## Running of stan model

### Hierarchical model

Below is the hierarchical model run with the corresponding histogram with the data.

```

hier_data = list(
  y = ham_data,
  N = nrow(ham_data),
  J = ncol(ham_data),
  U = 26,
  mu_s = 20,
  sigma_prior = 7
)

hier_fit = sampling(
  hier_ham,
  data = hier_data,
  chains = 4,
  iter = 2000,
  warmup = 1000,
  refresh = 0
)

```

### Nonhierarcial model (Pooled model)

```

pool_data = list(
  y = unlist(ham_data),
  N = nrow(ham_data),
  J = ncol(ham_data),
  mean_mu = 18,
  mean_sigma = 6
)

pool_fit = sampling(
  pool_ham,
  data = pool_data,
  chains = 4,
  iter = 2000,
  warmup = 1000,
  refresh = 0
)

```

## Predictive distribution of the mean hamilton the next season

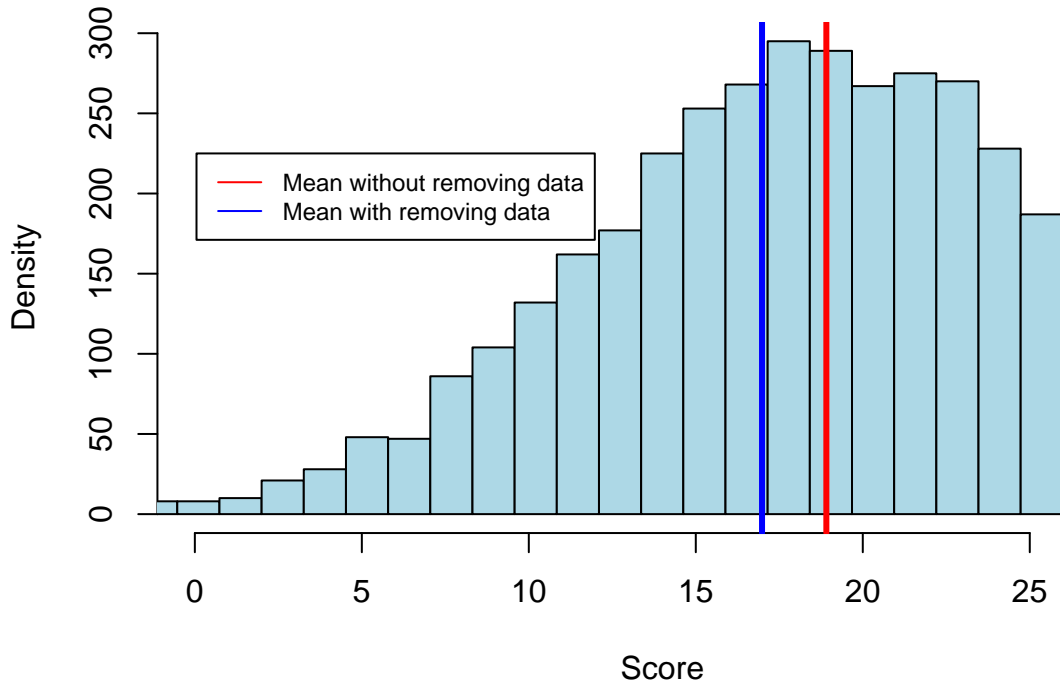


Figure 2: Histograms of mean score with mean lines for limited and unlimited sample.

## Density plot of the mean hamilton the next season

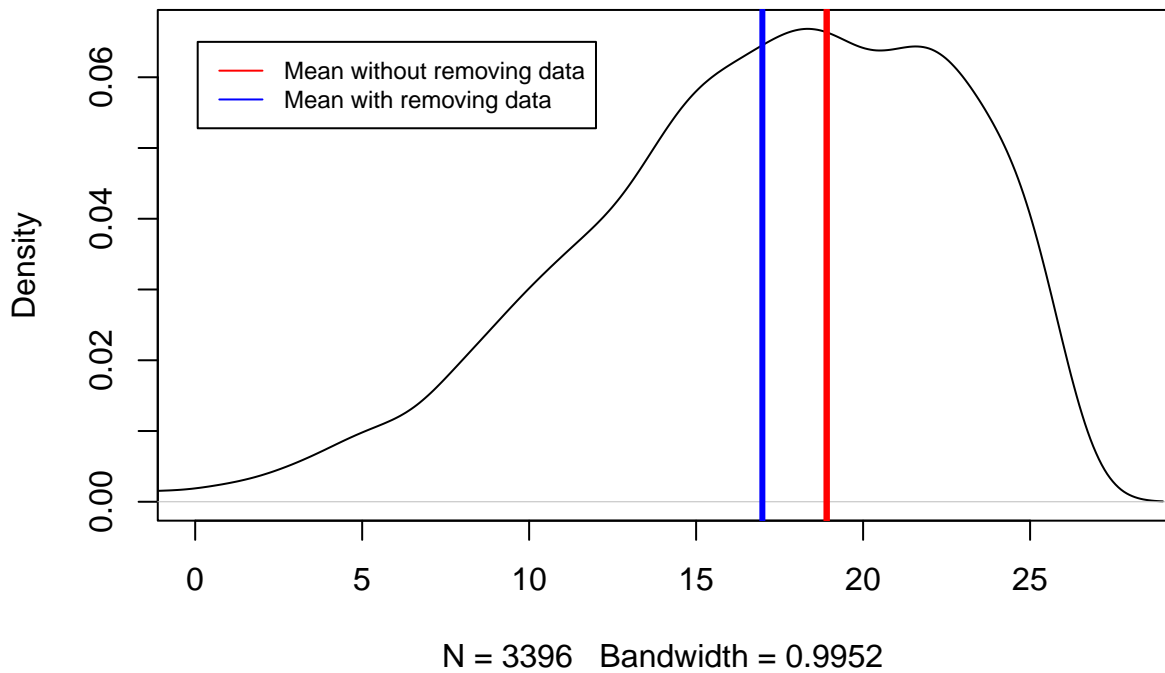


Figure 3: Density plot of mean score with mean lines for limited and unlimited sample.

)

## Predictive distribution of the mean hamilton the next season

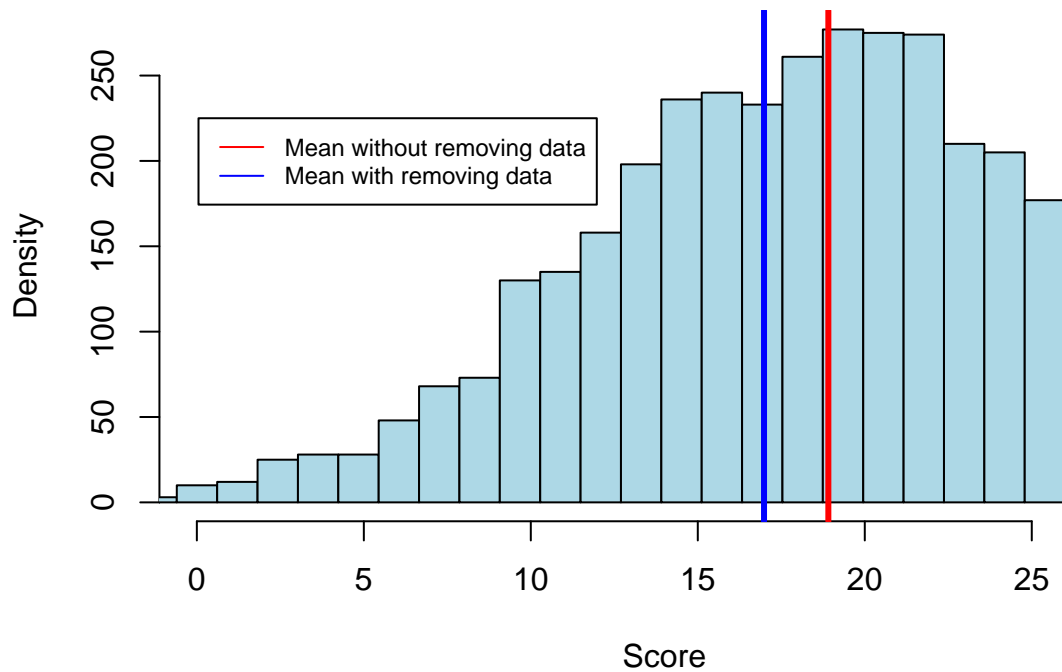


Figure 4: Histograms of mean score with mean lines for limited and unlimited sample.

Since the values of these normal distributions, go beyond the max points, i.e. 26, we have limited them a bit. We still plot the mean of both the limited and unlimited data. As can be seen, there isn't a lot of difference, however, over several races, this difference can be quite large. Below is also the histogram as a density plot.

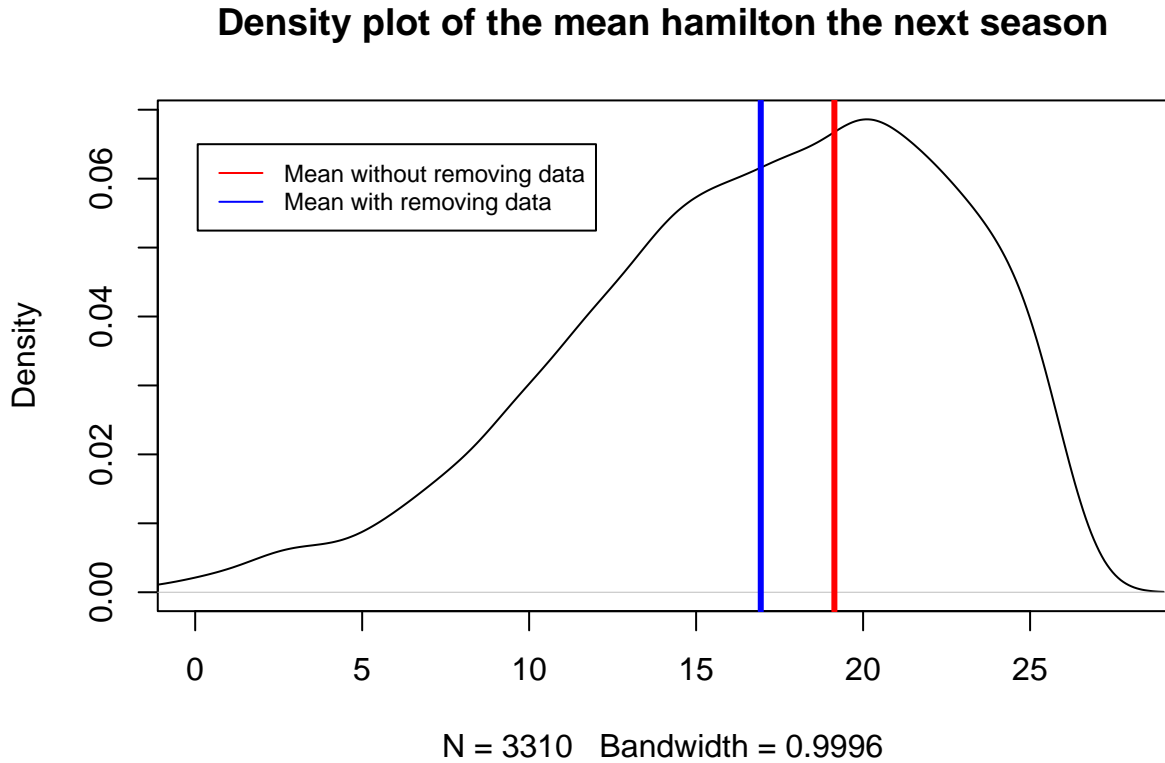


Figure 5: Density plot of mean score with mean lines for limited and unlimited sample.

## Convergence diagnostics

The  $\hat{R}$  for our fits are as follows:

- Hierarchical model: 1.01
- Pooled model: 1

Since these  $\hat{R}$  values are under 1.05, the chains have most likely mixed well.

Another convergence diagnostic we can look at is the ESS value we get out of the fits.

- Bulk ESS of the hierarchical model: 521.86
- Tail ESS of the hierarchical model: 638.22
- Bulk ESS of the pooled model: 3727.67
- Tail ESS of the pooled model: 3035.67

These ESS values measure the crude effective sample size for the bulk and tail quantities. A value over 100 is good and all of our values are over it.

Below are the traceplots

## Posterior predictive checks

### Predictive performance assessment

To assess the performance of the models we simulate 1000 seasons with the help of our models and compare the outcomes with the real world data.

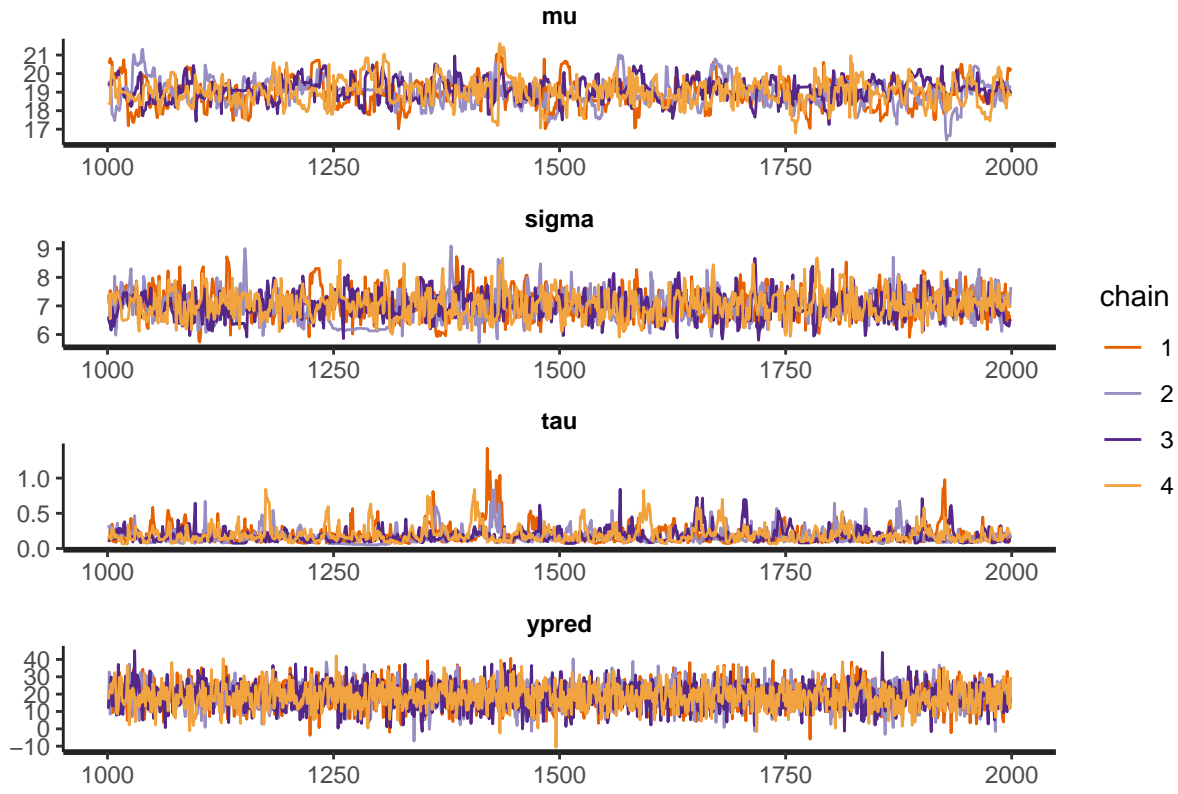


Figure 6: Trace plot of the hierarchical model.

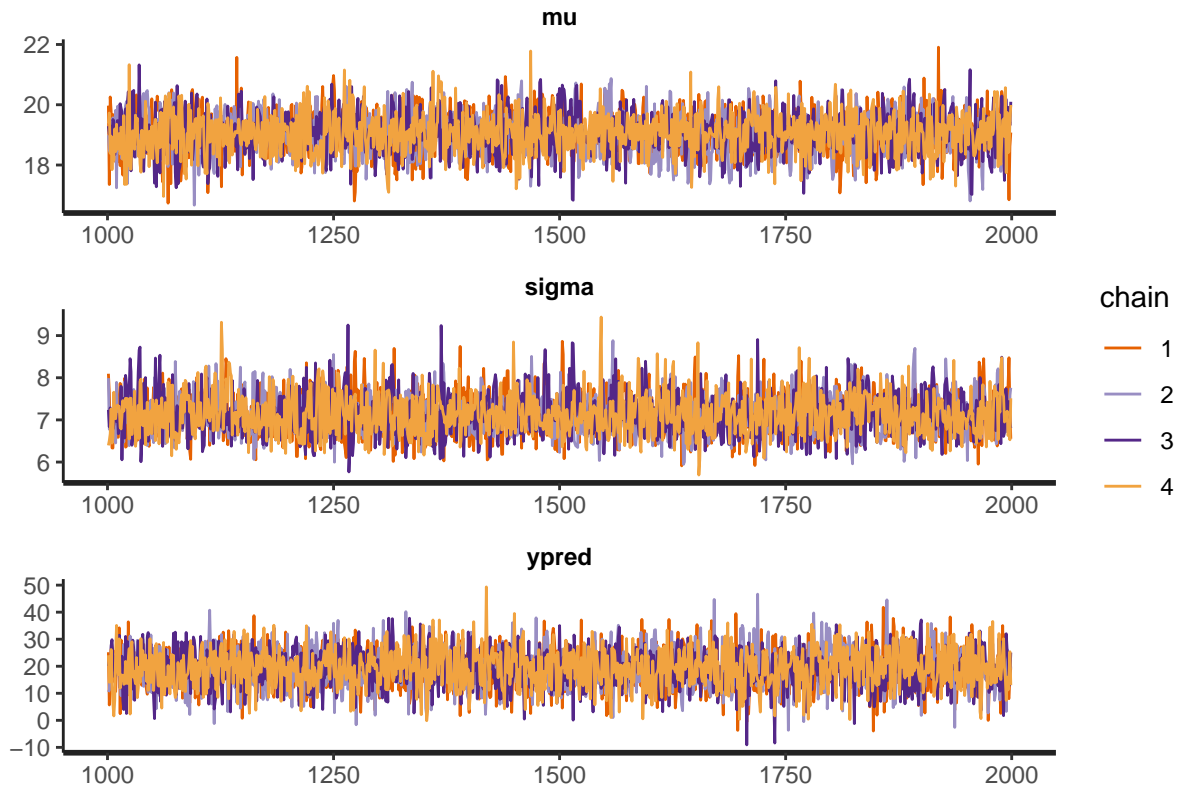


Figure 7: Trace plot of the nonhierarchical model.



Year	Driver	Points
2015	Hamilton	413
2016	Hamilton	408
2017	Hamilton	363
2018	Hamilton	380
2019	Hamilton	381

*# Code for simulating 1000 seasons*

```

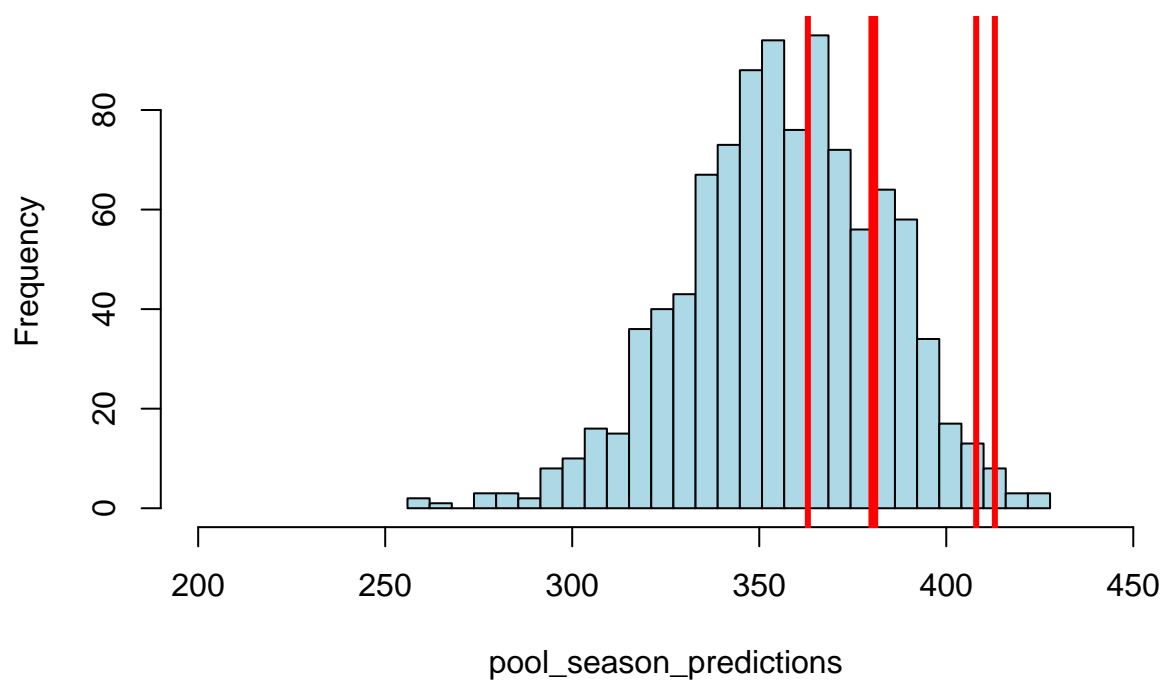
pool_season_predictions = c()
hier_season_predictions = c()

for(i in 1:1000) {
  dens = ham_pool_extracted_density
  N <- 21
  newx <- sample(x = dens$x, N,
    prob = dens$y, replace=TRUE)
  + rnorm(N, 0, dens$bw)
  pool_season_predictions <- append(pool_season_predictions, sum(newx))
}

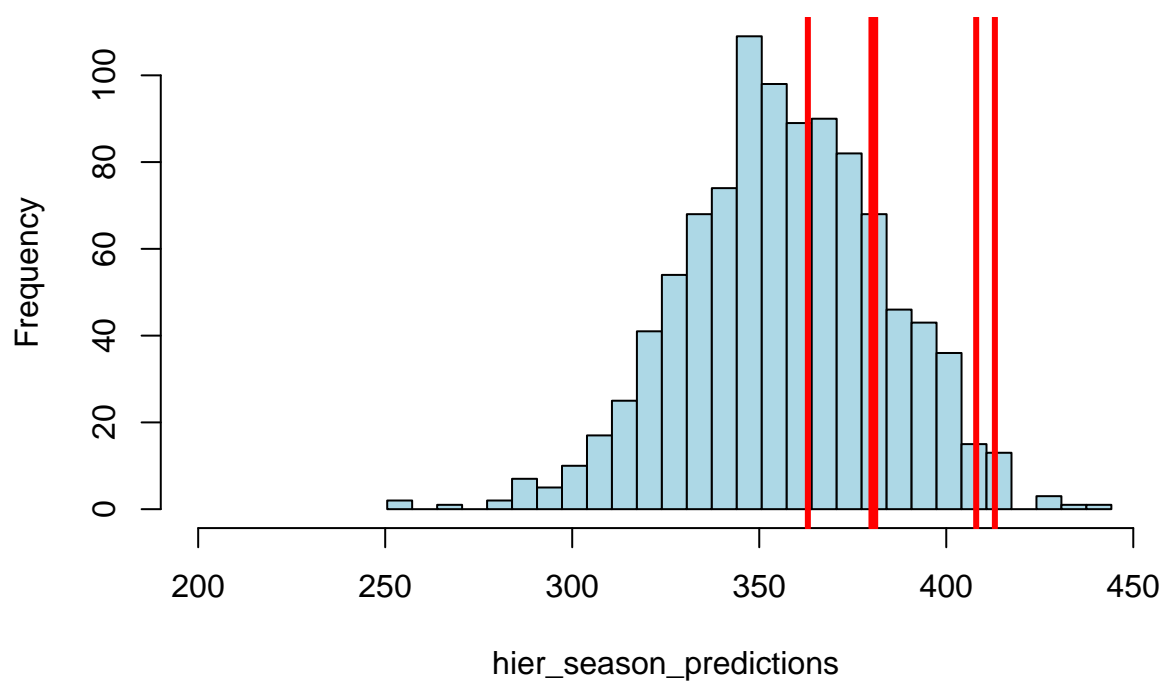
for(i in 1:1000) {
  dens = ham_extracted_density
  N <- 21
  newx <- sample(x = dens$x, N,
    prob = dens$y, replace=TRUE)
  + rnorm(N, 0, dens$bw)
  hier_season_predictions <- append(hier_season_predictions, sum(newx))
}

```

**Histogram of pool\_season\_predictions**

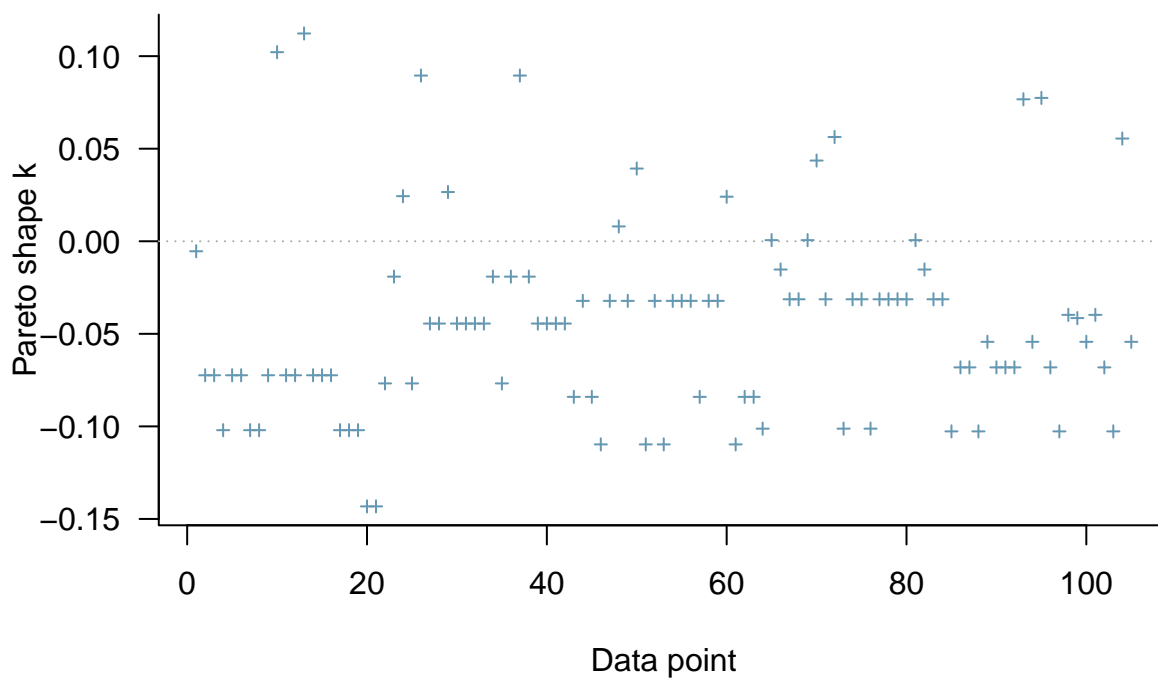
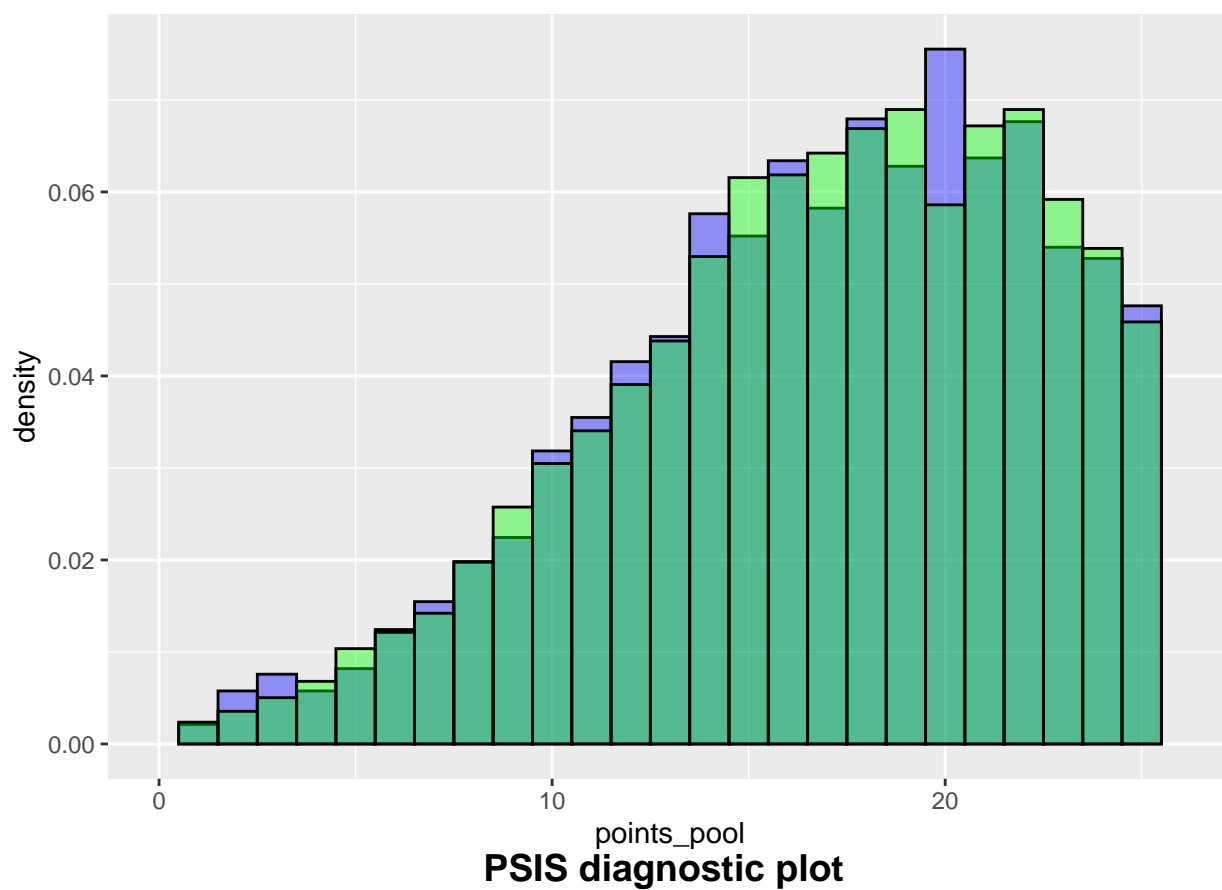


**Histogram of hier\_season\_predictions**



## Sensitivity analysis

### Model comparison



Discussion of issues and potential improvements

Conclusion what was learned from the data analysis

Self-reflection