Star Classification

Background

For thousands of years, civilizations have seemed to be fascinated with the night sky, naming constellations, even using stars to guide their paths at night. 2,000 years ago, the Greek astronomer Hipparchus made a catalog of stars according to their brightness. With advances in technology, however, modern day astronomers can be even more precise in their measurement. They generally classify stars into one of six types. In this project, I attempt to create a multinomial logit regression model using Bayesian methods to classify stars into one of their six types based on two variables about each star.

Exploratory Data Analysis

One way to classify stars is using the Hertzsprung-Russell diagram. This is shown below. Using this diagram, stars fall into one of six classes: Brown Dwarfs, Red Dwarfs, White Dwarfs, Main Sequence, Supergiants, and Hypergiants.

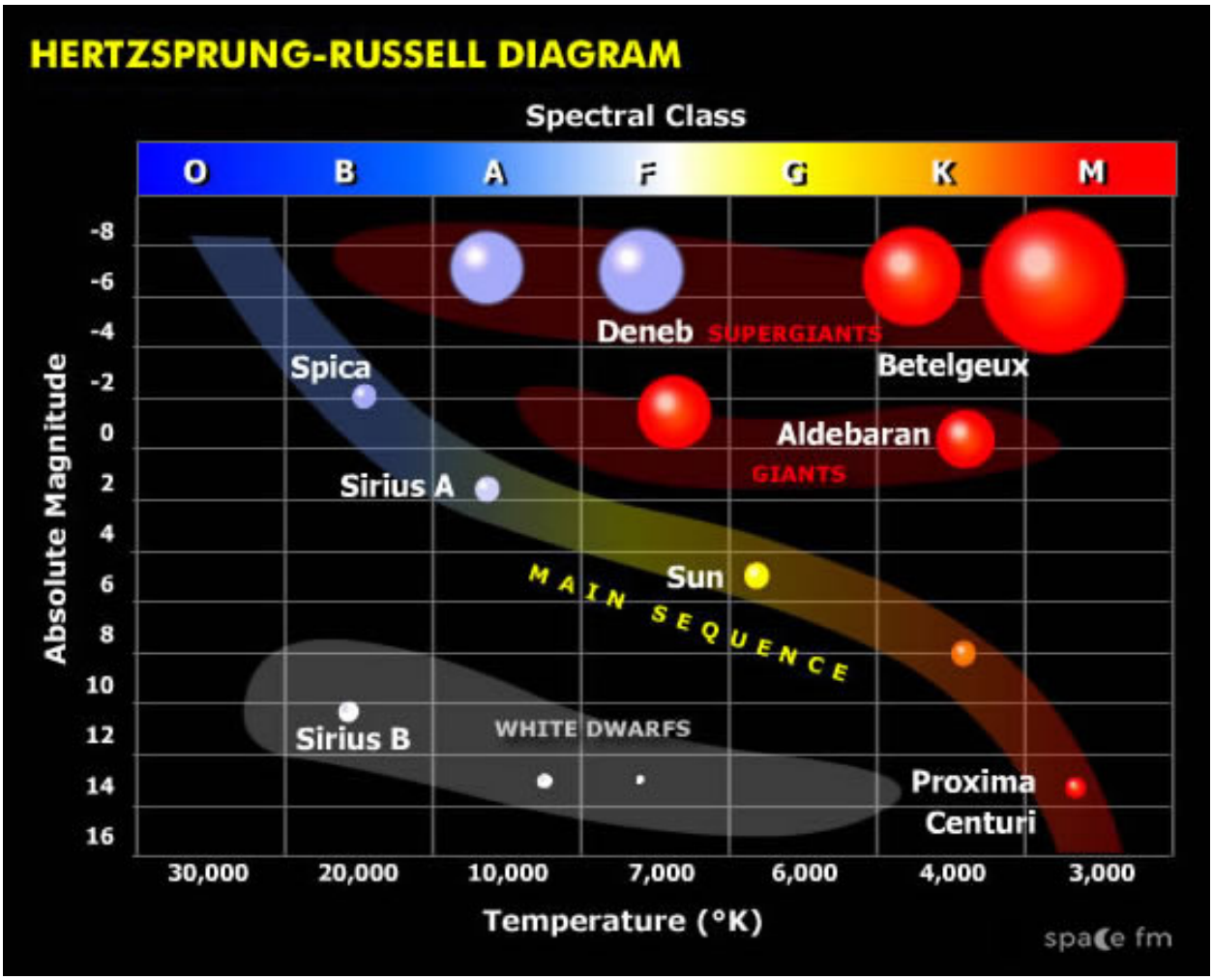

Fig. 1: Hertzsprung-Russell Diagram

I am using a dataset of 240 stars found on Kaggle (link in Appendix). Using this data, I was able to plot the magnitude and log of temperature for each star, similar to the Hertzsprung-Russell diagram shown above.

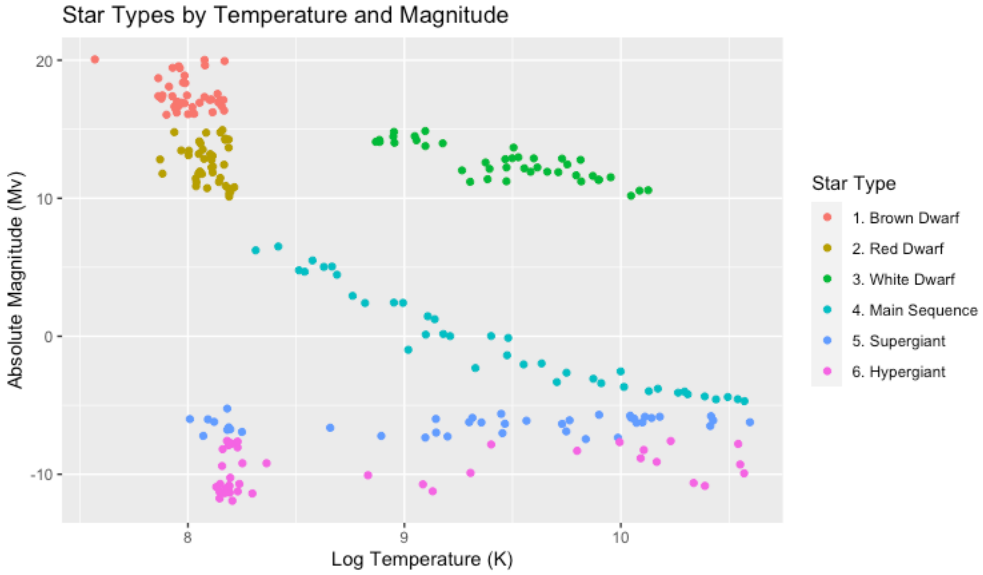

Fig. 2: Plot of Star Types

Model

Using the data above, I defined a train/test split in the data of 80/20%. Using the 80% training data, I fit a multinomial logistic regression using the magnitude and log of temperature to predict star type. I used Stan in R to approach this problem through Bayesian methods. See the appendix (or attached code) for reference on how this was done in Stan.

Model: Diagnostics

First, to check convergence of the estimates for the betas (beta 1 represents log temperature; beta 2 represents magnitude), I checked the traceplots to ensure the two chains mixed and seemed to converge. The estimates appear to have converged as shown in the traceplots below.

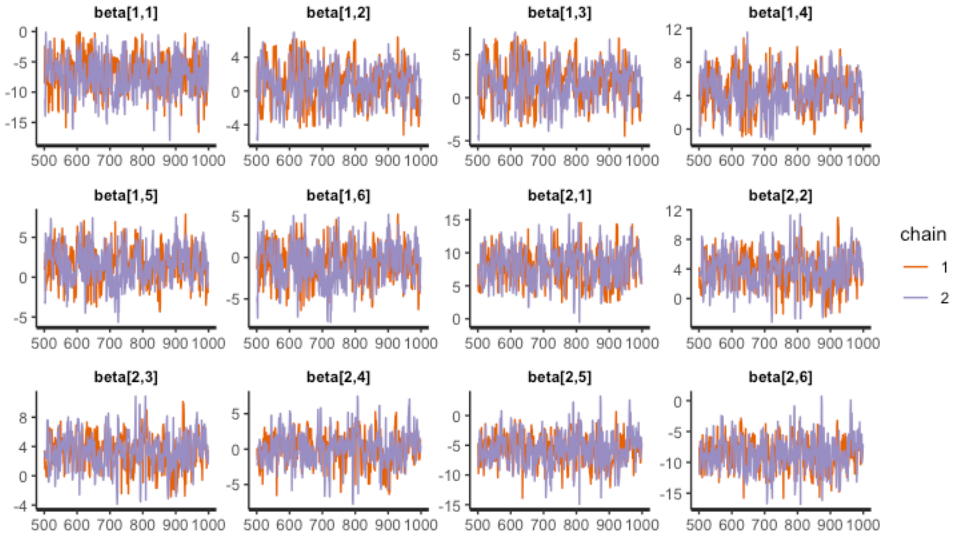

Fig. 3: Traceplots of Estimated Coefficients

Also, I checked the values of the estimated coefficients for the betas. These are shown below. While not all beta 1 and beta 2 coefficients are statistically significantly different from zero for all 6 star types, at least one star type for each beta coefficient is statistically significantly different from zero.

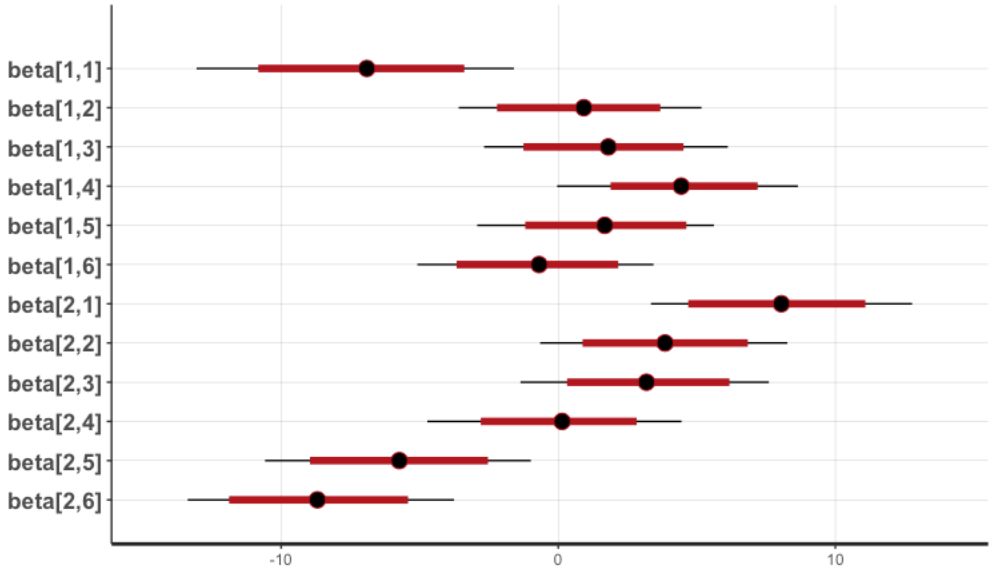

Fig. 4: Estimated Coefficients

As an alternative view of the plot above, I show a histogram of the beta coefficients below.

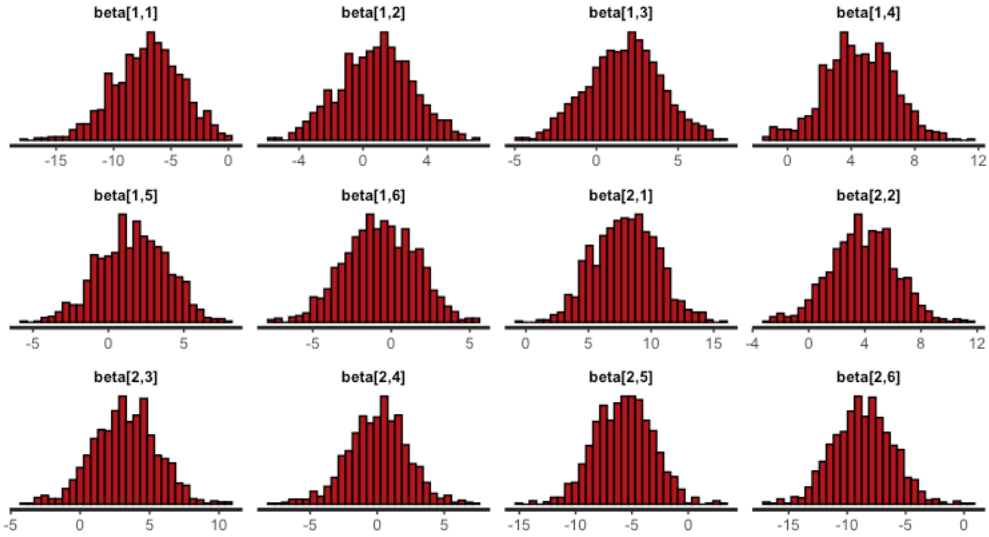

Fig. 5: Histogram of Estimated Coefficients

Model: Predictions

Using the fit model above, I then predicted star type for the 20% test data. Since this is a Bayesian method, we are given probabilities for each test star being of each star type. For plotting purposes below, I classified test stars to a single type based on whichever star type had the highest probability. The plots below show each star type (horizontal facets) and test stars (vertical facets) by probability of being in that star type (x-axis) and samples for that particular probability (y-axis). The color represents whether the classification was accurate or not. To be able to see all the data, I had to show this in four plots, shown below. As can be seen in the plots below, the majority of the test stars were correctly classified (as seen by plots where there is a high proportion of samples with high probability of being "True"). In fact, 43 out of 48 stars were correctly classified using this model (an accuracy of 89.5%).

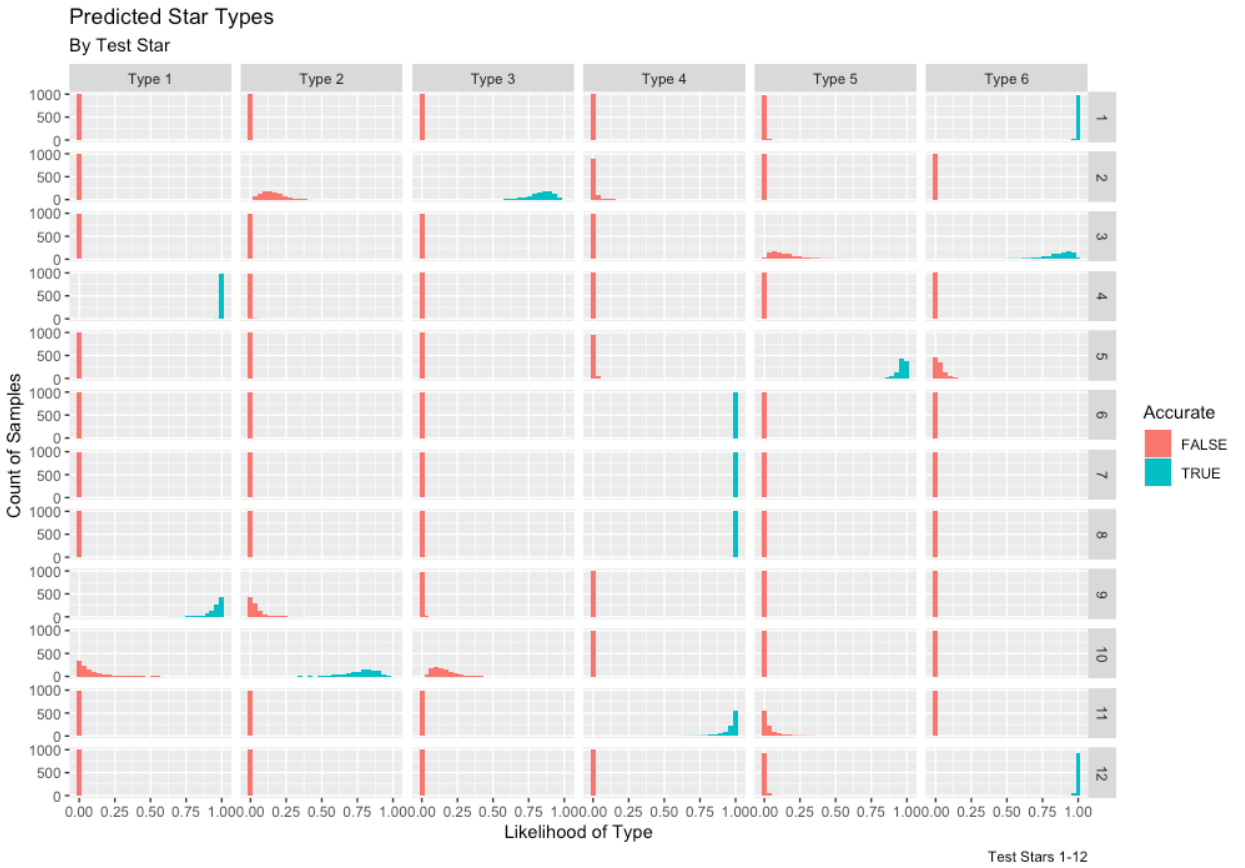

Fig. 6: Classification of Test Stars 1-12

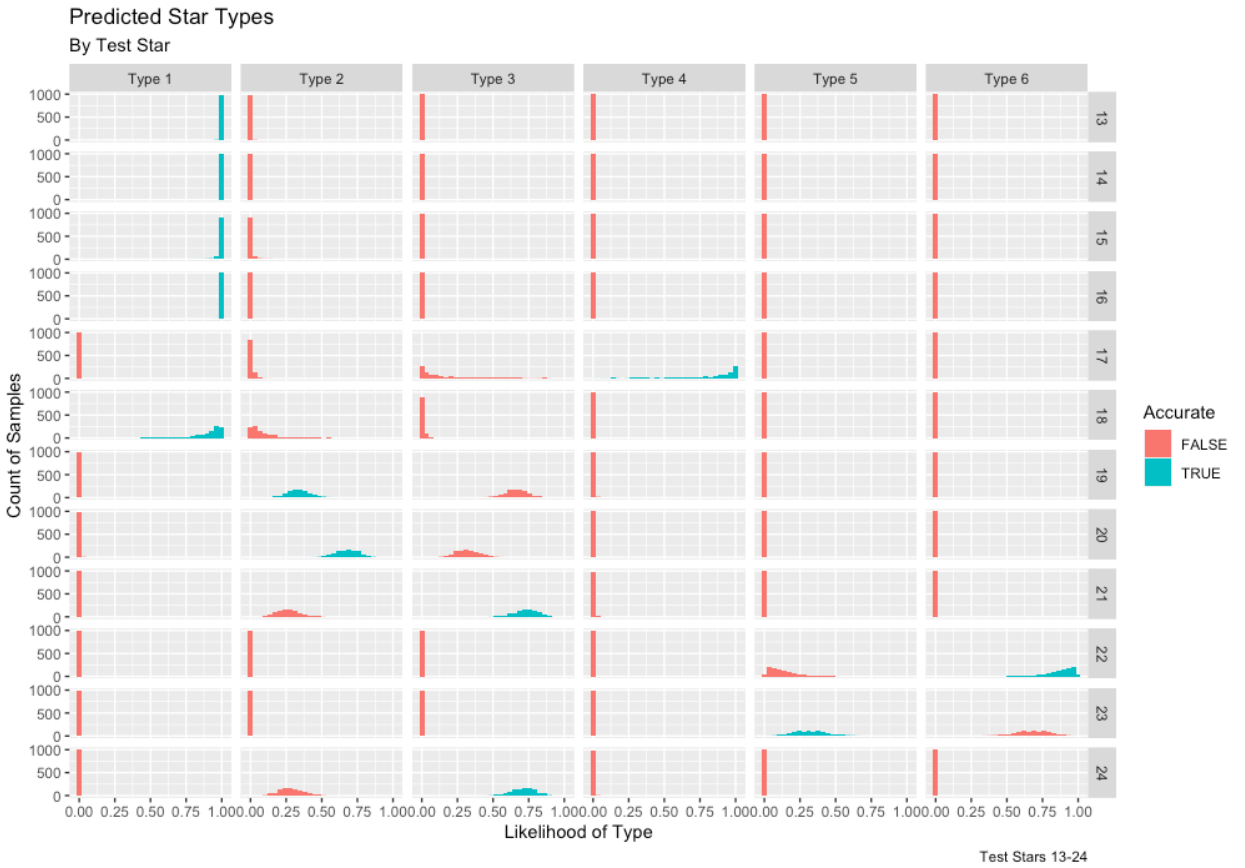

Fig. 7: Classification of Test Stars 13-24

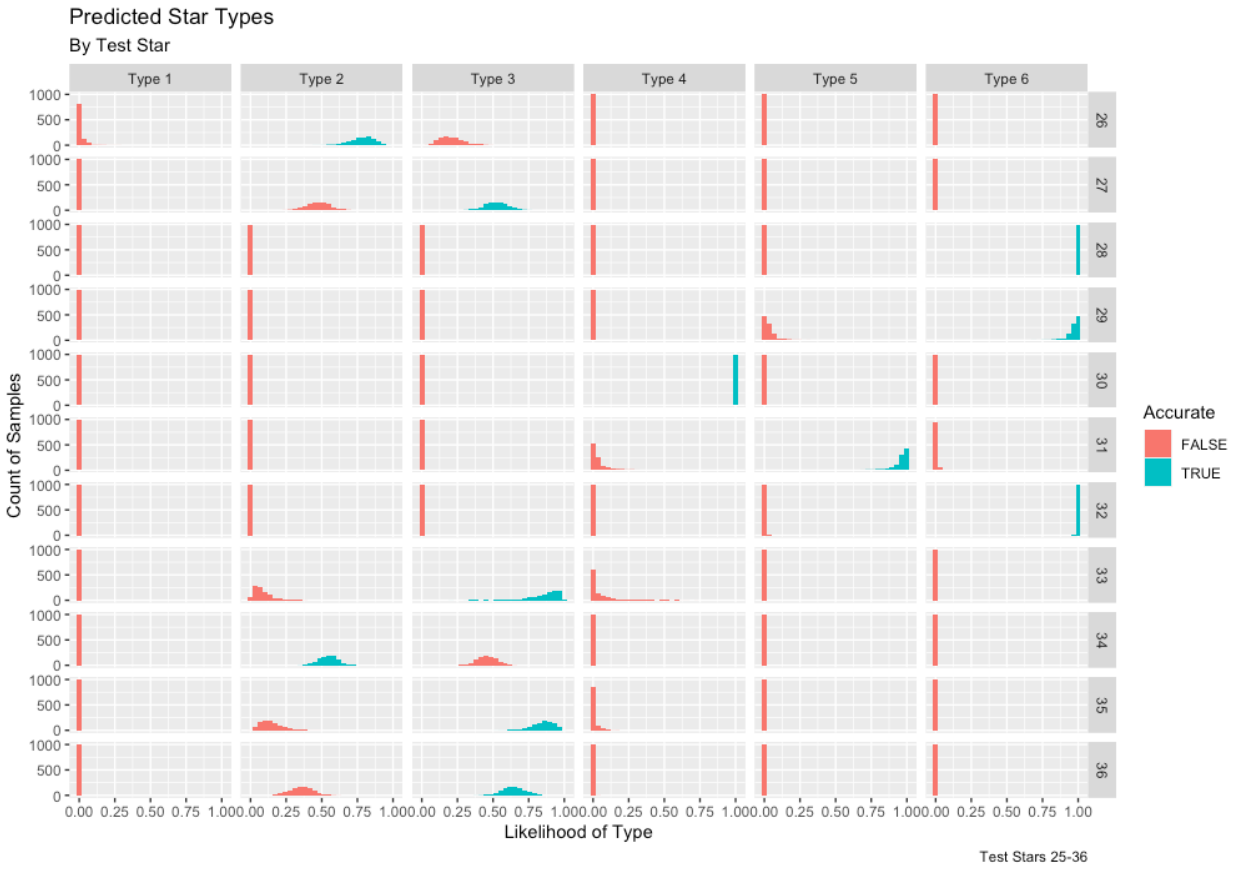

Fig. 8: Classification of Test Stars 25-36

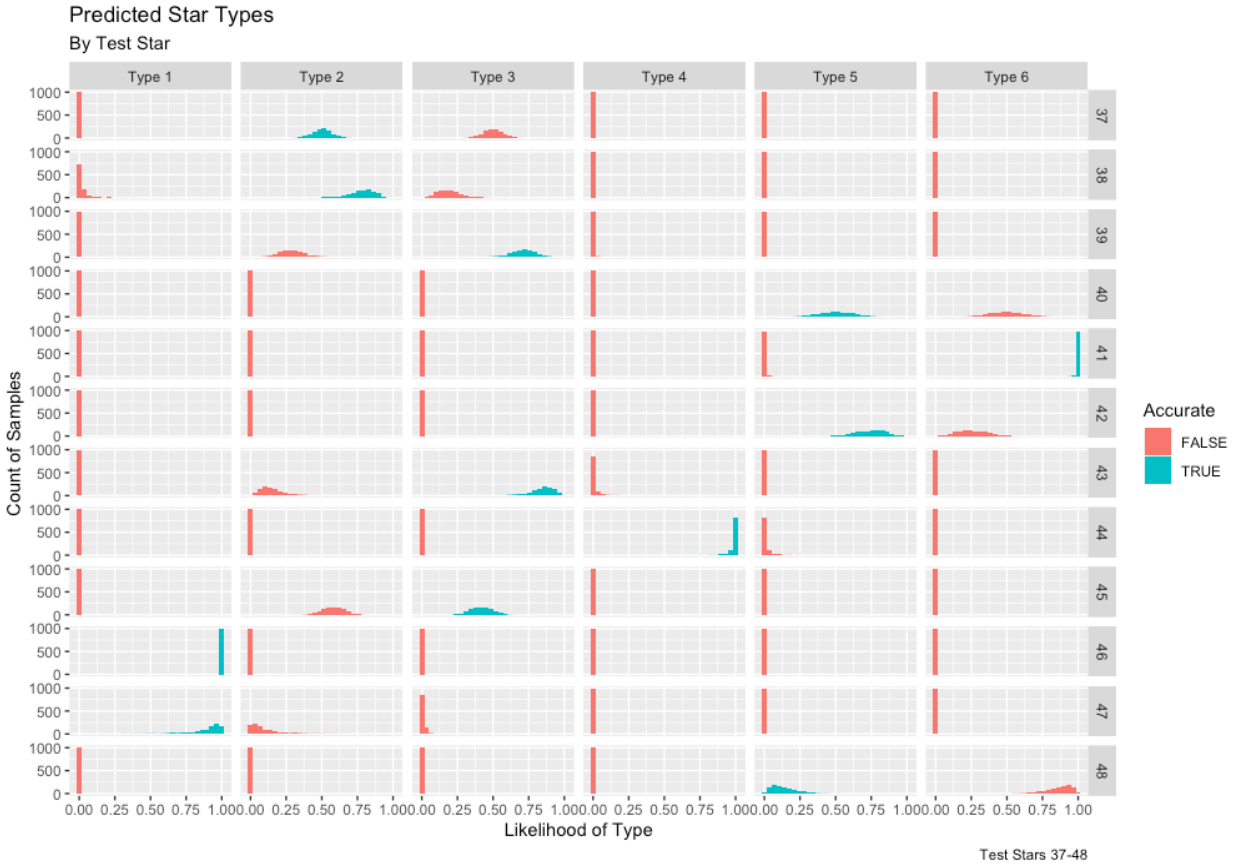

Fig. 9: Classification of Test Stars 37-48

Lastly, I plotted these predictions using the original axes of magnitude and log temperature. The color represents whether the model accurately classified each test star. While the two upper right and two bottom left incorrect points were near boundaries (making the inaccurate predictions understandable), the upper right incorrect point isn't near a boundary.

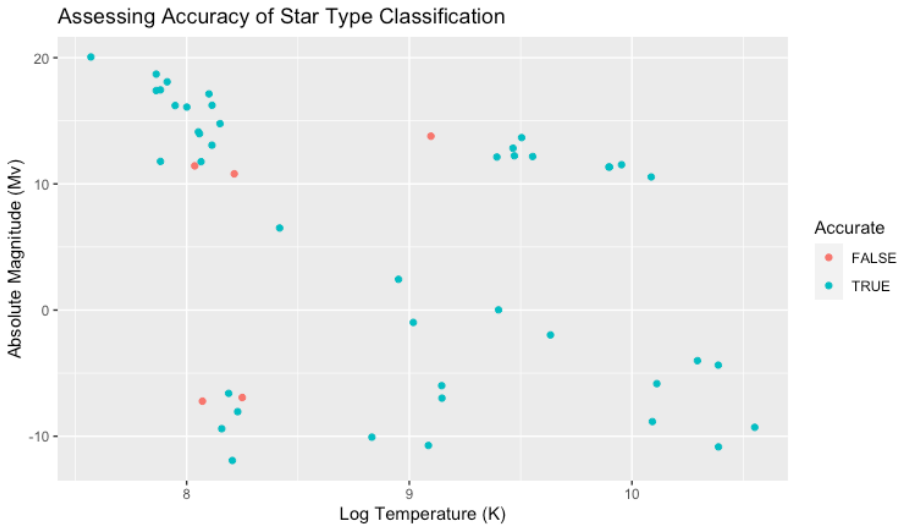

Fig. 10: Accuracy of Test Star Predictions

I also plotted this accuracy by the true star type. As seen below, types 1, 4, and 6 had all test points correctly classified, while type 3 had 90% correctly classified, type 2 had 75% correctly classified, and type 5 only had 67% correctly classified.

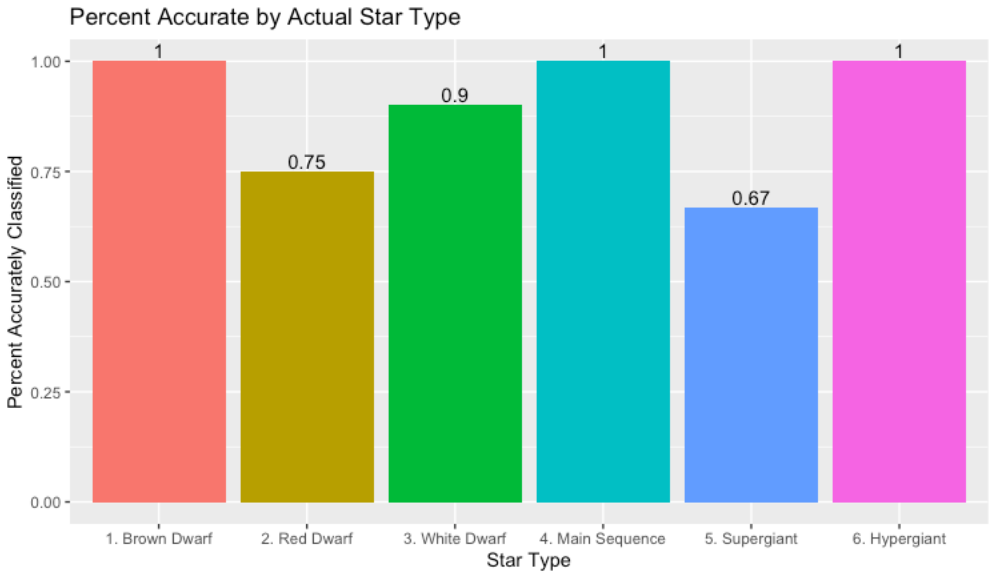

Fig. 11: Accuracy of Test Star Predictions by Type

Summary

In this project, I created a multinomial logistic regression model using Bayesian methods to classify star types based on their temperature and magnitude. In the test predictions, accuracy was relatively high, although certain star types tended to be misclassified, such as type 5 supergiants. If I were to pursue this further, I might would compare this Bayesian method to other methods for classifying data into multiple categories, such as support vector machines, nearest neighbors, or decision trees.

A Data Reference

https://www.kaggle.com/deepu1109/star-dataset

B Code

Below is the R code used for this project. The Stan model is included as a text block within the code.

R Code for Project

```
# load libraries and set options
library(tidyverse)
library(GGally)
library(rstan)
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)

# load data
df <- read_csv('Stars.csv')

# check for na's
any(is.na(df))

# rename fields, remove zero indexing on star type, and create log temp
df <- df %>%
  rename(temp = `Temperature (K)`, lum = `Luminosity(L/Lo)`,
         rad = `Radius(R/Ro)`, mag = `Absolute magnitude(Mv)`,
         type = `Star type`, color = `Star color`,
         class = `Spectral Class`) %>%
  mutate(type = type+1,
         ln_temp = log(temp),
         star_type_title = as.factor(case_when(
           type == 1 ~ '1. Brown Dwarf',
           type == 2 ~ '2. Red Dwarf',
           type == 3 ~ '3. White Dwarf',
           type == 4 ~ '4. Main Sequence',
           type == 5 ~ '5. Supergiant',
           type == 6 ~ '6. Hypergiant'
         )))

# correlations between variables
ggcorr(df)

# investigate relationship between log temp and magnitude
df %>%
  ggplot(aes(x = log(temp), y = mag, color = star_type_title)) +
  geom_point() +
  labs(title = 'Star Types by Temperature and Magnitude',
       x = 'Log Temperature (K)',
       y = 'Absolute Magnitude (Mv)',
       color = 'Star Type')
```

```
# shuffle data frame
rows <- sample(nrow(df))
df <- df[rows,]

# split to train and test data frames
dfTrain <- df[1:floor(nrow(df)*0.8),]
dfTest <- df[(floor(nrow(df)*0.8)+1):nrow(df),]

# set up data for stan
# beta1 = log temperature
# beta2 = magnitude
stars_dat <- list(N = nrow(dfTrain),
                  K = 6,
                  D = 2,
                  x = dfTrain %>% select(ln_temp, mag) %>% as.matrix(ncol =
2),
                  y = dfTrain$type,
                  N_new = nrow(dfTest),
                  x_new = dfTest %>% select(ln_temp, mag) %>% as.matrix(ncol
= 2))

# set up stan model
m1 <-
  "data {
int<lower=2> K; // number of categories
int<lower=1> N; // number of observations
int<lower=1> D; // number of predictors
int<lower=1,upper=K> y[N];
matrix[N,D] x;

int<lower=0> N_new;
matrix[N_new,D] x_new;
}

parameters {
matrix[D,K] beta;  // variable effects

}

model {
matrix[N,K] x_beta = x * beta;

to_vector(beta) ~ normal(0,5);

for (n in 1:N)
  y[n] ~ categorical_logit(x_beta[n]');
}

generated quantities {
matrix[N_new,K] y_new;

matrix[N_new,K] x_beta_new_exp = exp(x_new * beta);

vector[N_new] sum_eta;
```

```
for (n in 1:N_new)
  sum_eta[n] = sum(x_beta_new_exp[n]);

for (n in 1:N_new)
  for (k in 1:K)
    y_new[n,k] = x_beta_new_exp[n,k]/sum_eta[n];

}

"

# fit model
fit1 <- stan(model_code = m1, data = stars_dat, iter = 1000, chains = 2,
             seed = 42, control = list(max_treedepth = 15))

# print model results
print(fit1)

# create traceplots of betas to check convergence
traceplot(fit1, pars = c('beta'))

# check coefficient values
rstan::stan_plot(fit1, pars = c('beta'))
stan_hist(fit1, pars = c('beta'))

# extract samples
dfSamples <- as.data.frame(fit1)

# manipulate data to get to tidy samples
dfPredictions <- dfSamples[,13:((nrow(dfTest)*6)+12)] %>%
  mutate(fake_pivot = 1) %>%
  pivot_longer(cols = -fake_pivot) %>%
  mutate(rowid = as.numeric(str_sub(name, 7, (str_length(name)-3))),
         sample_type = str_sub(name, (str_length(name)-1), (str_length(name)-
1)),
         sample_type_title = str_c('Type', sample_type, sep = ' ')) %>%
  select(rowid, sample_type, sample_type_title, value)

# join in actual type
dfAccuracy <- dfTest %>%
  rowid_to_column() %>%
  select(rowid, type, ln_temp, mag) %>%
  inner_join(dfPredictions, by = 'rowid') %>%
  mutate(accurate = case_when(
    type == sample_type ~ TRUE,
    TRUE ~ FALSE
  ))

# graph distributions
dfAccuracy %>%
  filter(rowid <= 12) %>%
  ggplot(aes(x = value, fill = accurate)) +
  geom_histogram() +
```

```
  scale_y_continuous(breaks = c(0,500,1000)) +
  facet_grid(rowid~sample_type_title) +
  labs(title = 'Predicted Star Types',
       subtitle = 'By Test Star',
       x = 'Likelihood of Type',
       y = 'Count of Samples',
       fill = 'Accurate',
       caption = 'Test Stars 1-12')

dfAccuracy %>%
  filter(rowid > 12 & rowid <= 24) %>%
  ggplot(aes(x = value, fill = accurate)) +
  geom_histogram() +
  scale_y_continuous(breaks = c(0,500,1000)) +
  facet_grid(rowid~sample_type_title) +
  labs(title = 'Predicted Star Types',
       subtitle = 'By Test Star',
       x = 'Likelihood of Type',
       y = 'Count of Samples',
       fill = 'Accurate',
       caption = 'Test Stars 13-24')

dfAccuracy %>%
  filter(rowid > 25 & rowid <= 36) %>%
  ggplot(aes(x = value, fill = accurate)) +
  geom_histogram() +
  scale_y_continuous(breaks = c(0,500,1000)) +
  facet_grid(rowid~sample_type_title) +
  labs(title = 'Predicted Star Types',
       subtitle = 'By Test Star',
       x = 'Likelihood of Type',
       y = 'Count of Samples',
       fill = 'Accurate',
       caption = 'Test Stars 25-36')

dfAccuracy %>%
  filter(rowid > 36) %>%
  ggplot(aes(x = value, fill = accurate)) +
  geom_histogram() +
  scale_y_continuous(breaks = c(0,500,1000)) +
  facet_grid(rowid~sample_type_title) +
  labs(title = 'Predicted Star Types',
       subtitle = 'By Test Star',
       x = 'Likelihood of Type',
       y = 'Count of Samples',
       fill = 'Accurate',
       caption = 'Test Stars 37-48')

# create summary accuracy
dfAccuracySummary <- dfAccuracy %>%
  group_by(rowid, type, ln_temp, mag, sample_type, accurate) %>%
  summarize(mean_value = mean(value)) %>%
  ungroup() %>%
  arrange(rowid, desc(mean_value)) %>%
```

```
  group_by(rowid) %>%
  slice(1)

# calculate percent accurate
dfAccuracySummary %>%
  group_by(accurate) %>%
  summarize(count = n())

# calculate percent accurate by true type
dfAccuracySummary %>%
  group_by(type, accurate) %>%
  summarize(count = n()) %>%
  ungroup() %>%
  group_by(type) %>%
  mutate(total = sum(count),
         pct = count/total,
         star_type_title = as.factor(case_when(
           type == 1 ~ '1. Brown Dwarf',
           type == 2 ~ '2. Red Dwarf',
           type == 3 ~ '3. White Dwarf',
           type == 4 ~ '4. Main Sequence',
           type == 5 ~ '5. Supergiant',
           type == 6 ~ '6. Hypergiant'
         ))) %>%
  filter(accurate == TRUE) %>%
  ggplot(aes(x = star_type_title, y = pct, fill = star_type_title)) +
  geom_col() +
  geom_text(aes(label = round(pct, 2)), vjust = -0.25) +
  labs(title = 'Percent Accurate by Actual Star Type',
       x = 'Star Type',
       y = 'Percent Accurately Classified') +
  theme(legend.position = 'none')

# plot on original scale
dfAccuracySummary %>%
  ggplot(aes(x = ln_temp, y = mag, color = accurate)) +
  geom_point() +
  labs(title = 'Assessing Accuracy of Star Type Classification',
       x = 'Log Temperature (K)',
       y = 'Absolute Magnitude (Mv)',
       color = 'Accurate')
```