Final Report and Code

Bryan Jianjun Chen

2023-08-21

Research Question 1: How does the size distribution of the smooth galaxies and the featured galaxies differ, and how can this be compared using linear regression with indicator variables?

In this study, we will construct a model in which we can compare the size distributions between the smooth and featured galaxies. We will be using a linear model with indicator variables, setting a baseline value and comparing it with an indicator value in order to compare the two values.

We must select a p-value threshold in our statistical tests before we construct our models at which to reject or accept the p-value. We will set the alpha value at 0.05, meaning that if our p-value is less than 0.05, we will reject the null hypothesis, and if our p-value is higher than 0.05, we will accept the null hypothesis.

```
library(tidyverse)
library(rpart)
library(partykit)
library(dplyr)
library(arrow)
```

```
gz2 <- read_parquet("gz2_catalog_with_modern_schema_no_paths.parquet")
nsa <- read_parquet("nsa_v1_0_1_key_cols.parquet")
```

```
nsa <- nsa %>% filter(!is.na(sersic_nmgy_r))
gz2 <- gz2 %>% mutate(
  smooth_or_not = ifelse(`smooth-or-featured-gz2_smooth` > `smooth-or-featured-gz2_featured-or-disk` +
                       yes = 'smooth', no = 'featured')
)
```

```
galaxy_data <- merge(gz2, nsa, by = "iauname")
```

```
galaxy_data %>% group_by(smooth_or_not) %>%
  summarise(mean = mean(petro_theta))
```

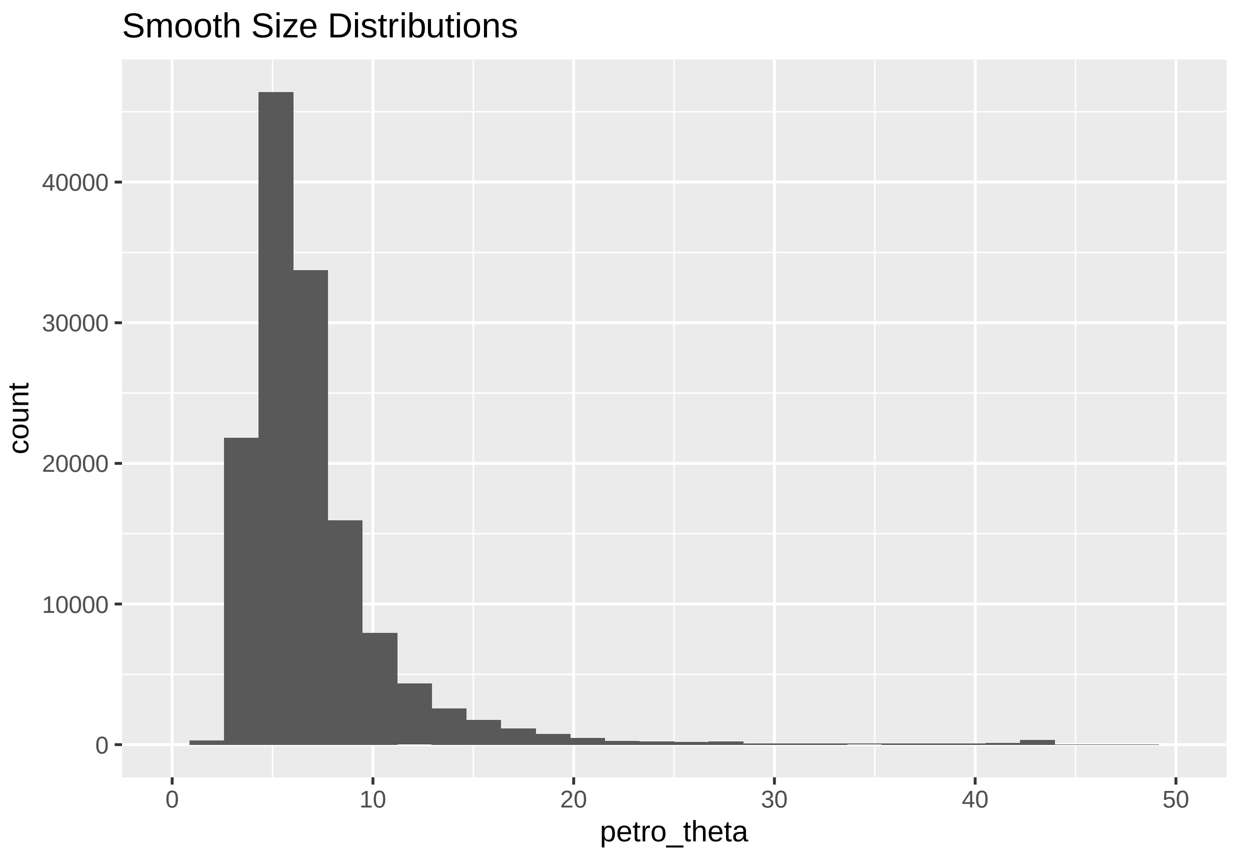
```
## # A tibble: 2 x 2
##   smooth_or_not  mean
##   <chr>         <dbl>
## 1 featured      11.2
## 2 smooth         7.50
```

```
galaxy_data %>% filter(smooth_or_not == "smooth") %>%
  ggplot() +
  aes(x = petro_theta) +
  geom_histogram() +
  labs(title = "Smooth Size Distributions") +
  xlim(0, 50)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 677 rows containing non-finite values (`stat_bin()`).
```

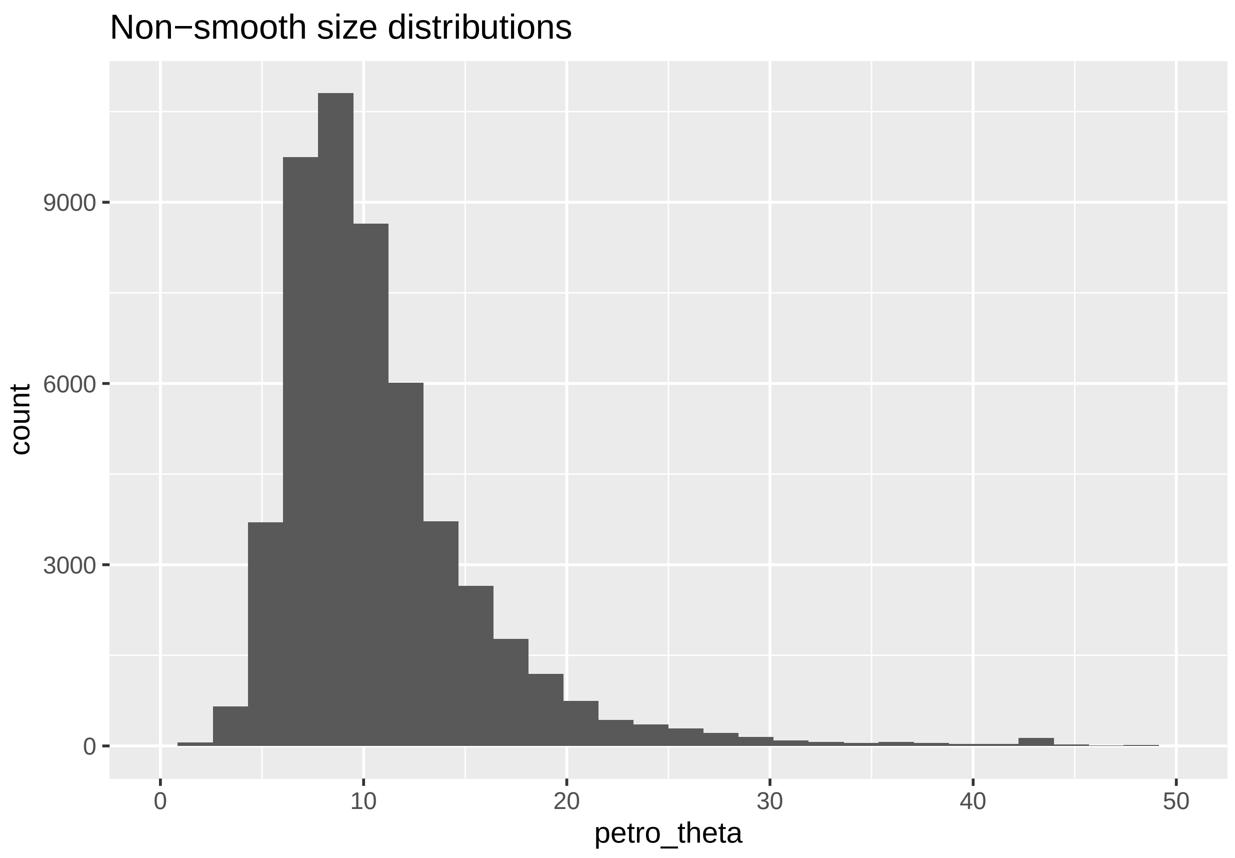
```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



```
galaxy_data %>% filter(smooth_or_not == "featured") %>%
  ggplot() +
  aes(x = petro_theta) +
  geom_histogram() +
  labs(title = "Non-smooth size distributions") +
  xlim(0, 50)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 237 rows containing non-finite values (`stat_bin()`).
## Removed 2 rows containing missing values (`geom_bar()`).
```



```r
size_mod <- lm(petro_theta ~ smooth_or_not, data = galaxy_data)
summary(size_mod)
```

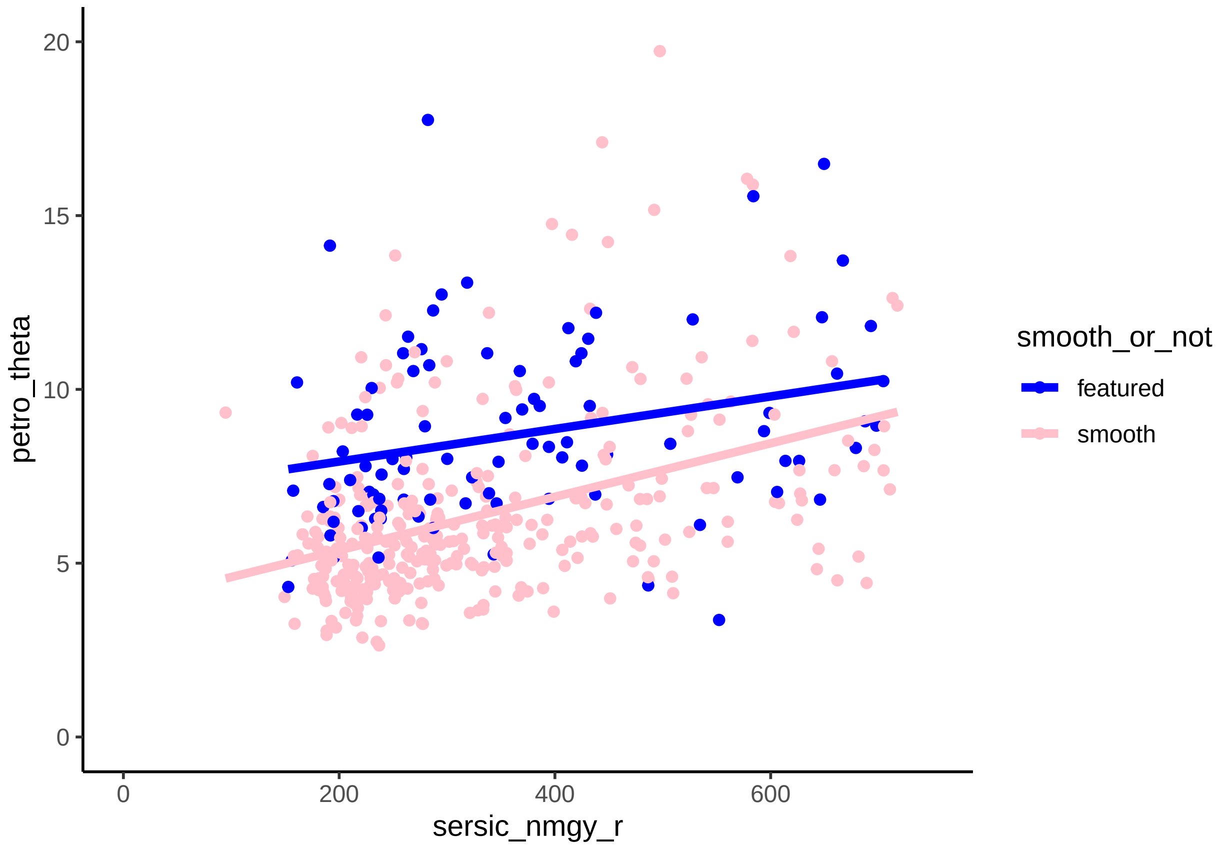
```
##
## Call:
## lm(formula = petro_theta ~ smooth_or_not, data = galaxy_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.217  -2.887  -1.454   0.669 243.533
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           11.23778    0.03232  347.67   <2e-16 ***
## smooth_or_notsmooth -3.73556    0.03784  -98.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.358 on 191766 degrees of freedom
## Multiple R-squared:  0.04837,    Adjusted R-squared:  0.04836
## F-statistic:  9747 on 1 and 191766 DF,  p-value: < 2.2e-16
```

#The results

In our linear regression model, we have used the indicator variable of smooth_or_not in order to find the relation between differing levels of smoothness and the size of petro_theta.

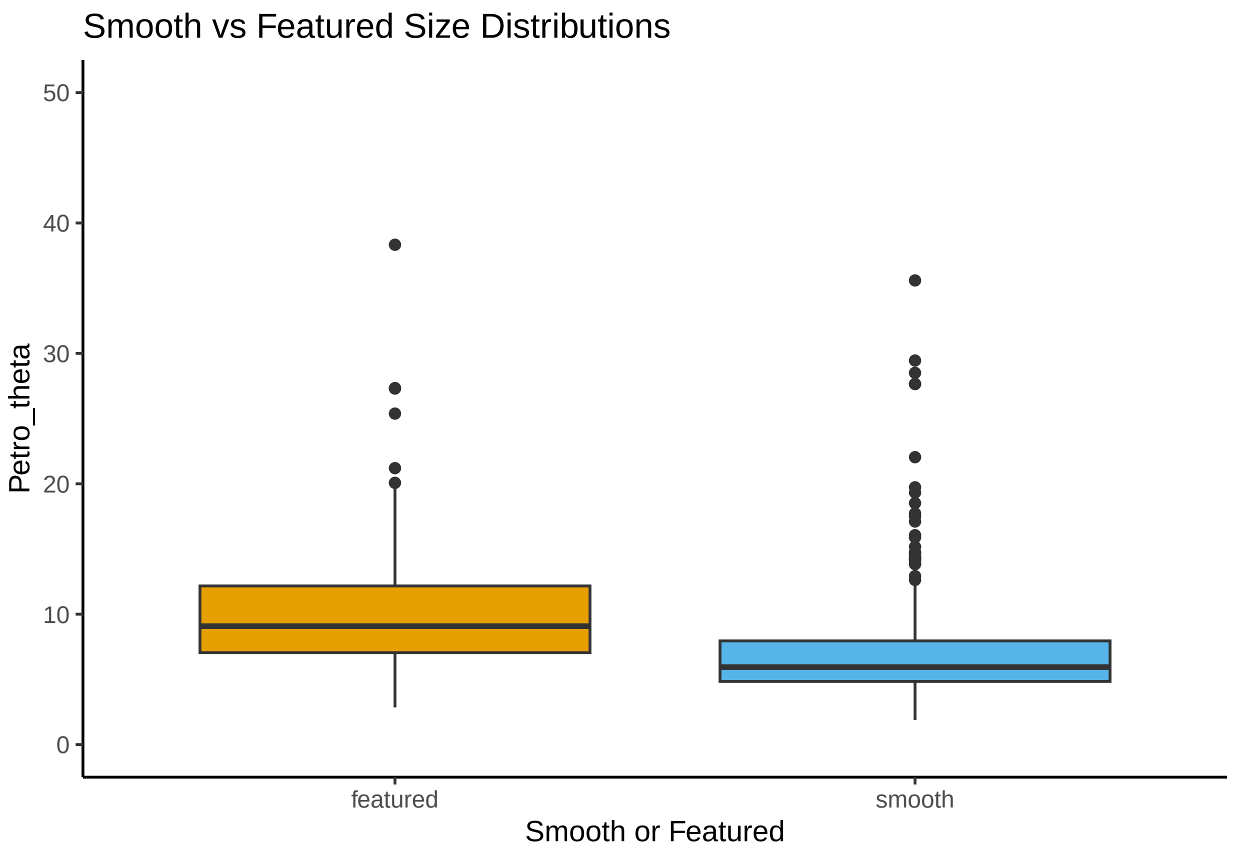
```r
galaxy_data_smaller <- galaxy_data[sample(nrow(galaxy_data), 500), ]
galaxy_data_smaller %>% ggplot(aes(x = sersic_nmgy_r, y = petro_theta, color = smooth_or_not)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, size = 1.5) +
  labs(x = "sersic_nmgy_r", y = "petro_theta") +
  scale_color_manual(values = c("blue", "pink")) +
  theme_classic() +
  xlim(0, 750) +
  ylim(0, 20)
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 85 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 85 rows containing missing values (`geom_point()`).
```



```r
galaxy_data_smaller %>% ggplot(aes(x = smooth_or_not, y = petro_theta, fill = smooth_or_not)) +
  geom_boxplot() +
  scale_fill_manual(values = c("#E69F00", "#56B4E9", "#009E73")) +
  labs(x = "Smooth or Featured", y = "Petro_theta", fill = "Smooth or Featured") +
  theme_classic() +
  theme(legend.position = "none") +
  ggtitle("Smooth vs Featured Size Distributions") +
  ylim(0, 50)
```

```
## Warning: Removed 5 rows containing non-finite values (`stat_boxplot()`).
```



Research Question 2: What factors finally decided smooth/feature in galaxies? (Classification tree)

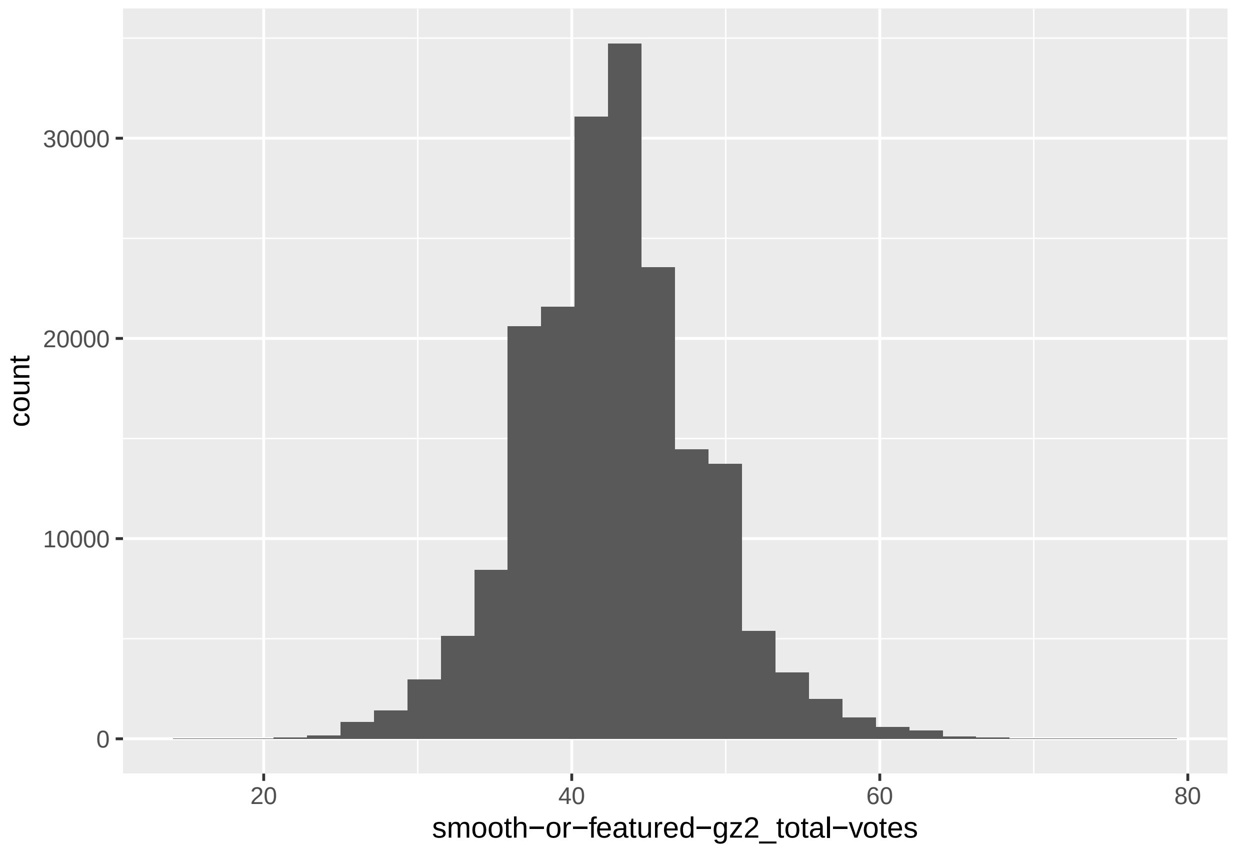
This question focuses on the Galaxy Zoo data set.

Step#1: Perform feature engineering to find important features

With the data, we use the histogram to look at two populations: smooth galaxies and featured galaxies. To find whether the galaxies are smooth or feature, we use the `smooth-or-featured-gz2_total-votes`column, which means we want to check the total number of votes on this question per galaxy.

```r
galaxy_data %>% ggplot() + aes(x = `smooth-or-featured-gz2_total-votes`) + geom_histogram()
```

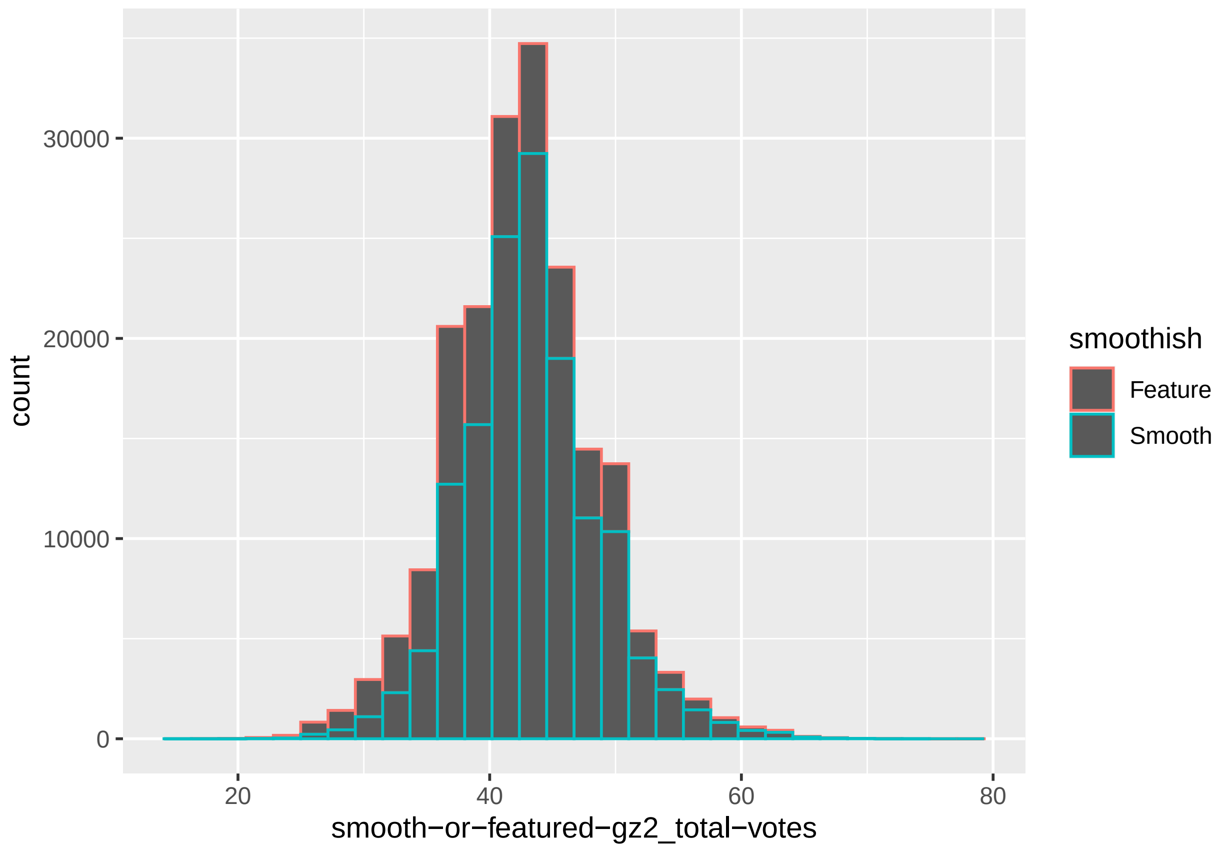
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We use "mutate()" and "case_when()" functions to create the new variable called smoothish that splits the `galazy_zoo` data set into two samples: a "smooth" sample and a "featured" sample. So, we define whether a galaxy is above or below 0.5 in the column `smooth-or-featured-gz2_smooth_fraction` and use the "filter()" function to find the row `smooth-or-featured-gz2_total-votes` greater than 10 and save these into a new data set called `gz_smoothosh`. Finally, visualize the relative number of galaxies in each group.

```
gz_smoothosh <- galaxy_data %>% filter(`smooth-or-featured-gz2_total-votes` > 10) %>% mutate(smoothish=
gz_smoothosh %>% group_by(smoothish) %>%
  ggplot() + aes(x=`smooth-or-featured-gz2_total-votes`, colour=smoothish) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



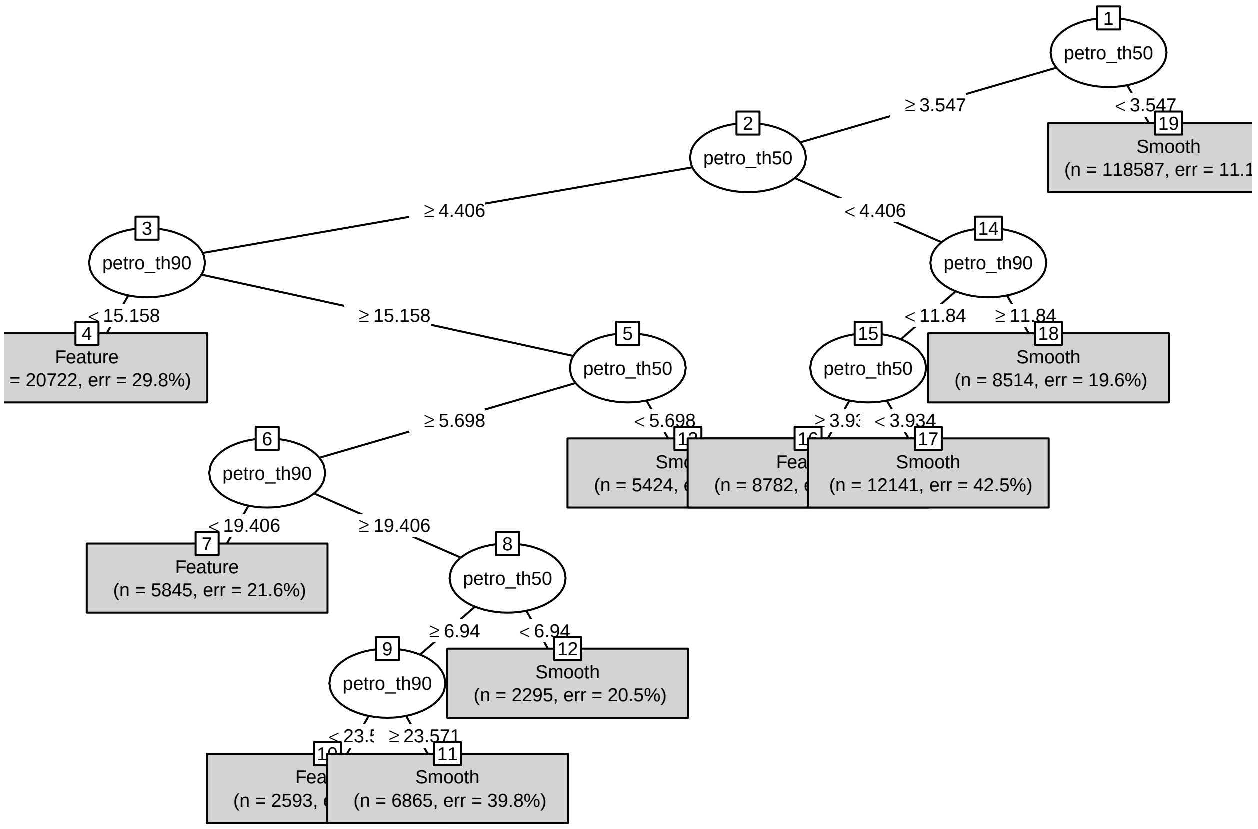
After splitting the galaxies into "Smooth" and "Feature," We select some relative objectivity reasons(variables) that may impact whether the galaxies are smooth or not. We create a data frame to find the importance of these variables. We choose the `smoothish` as the response variable.

```
tree <-rpart(smoothish ~ best_match + sky_separation + ra.y + dec.y + petro_theta + petro_th50 + petro_

v_importance <- as.data.frame(tree$variable.importance)
v_importance
```

```
##                    tree$variable.importance
## petro_th50                     1.632578e+04
## petro_theta                    1.406224e+04
## petro_th90                     1.162396e+04
## sersic_nmgy_r                  4.950914e+03
## mag_r                          4.949278e+03
## redshift                       3.199722e+03
## elpetro_absmag_r               3.170734e+03
## sky_separation                 4.552220e+01
## best_match                     5.222432e-01
```

From the data frame above, we find the `petro_th50`, `petro_theta`and `petro_th90` are the first three that have larger importance of variable to `smoothish`. So, we will create a classification tree to see the relationship between these three variables and the smoothish form of the `gz_smoothosh` data set.

```
tree <-rpart(smoothish ~ petro_theta + petro_th50 + petro_th90, data = gz_smoothosh)
plot(as.party(tree), gp=gpar(cex=0.6), type="simple")
```

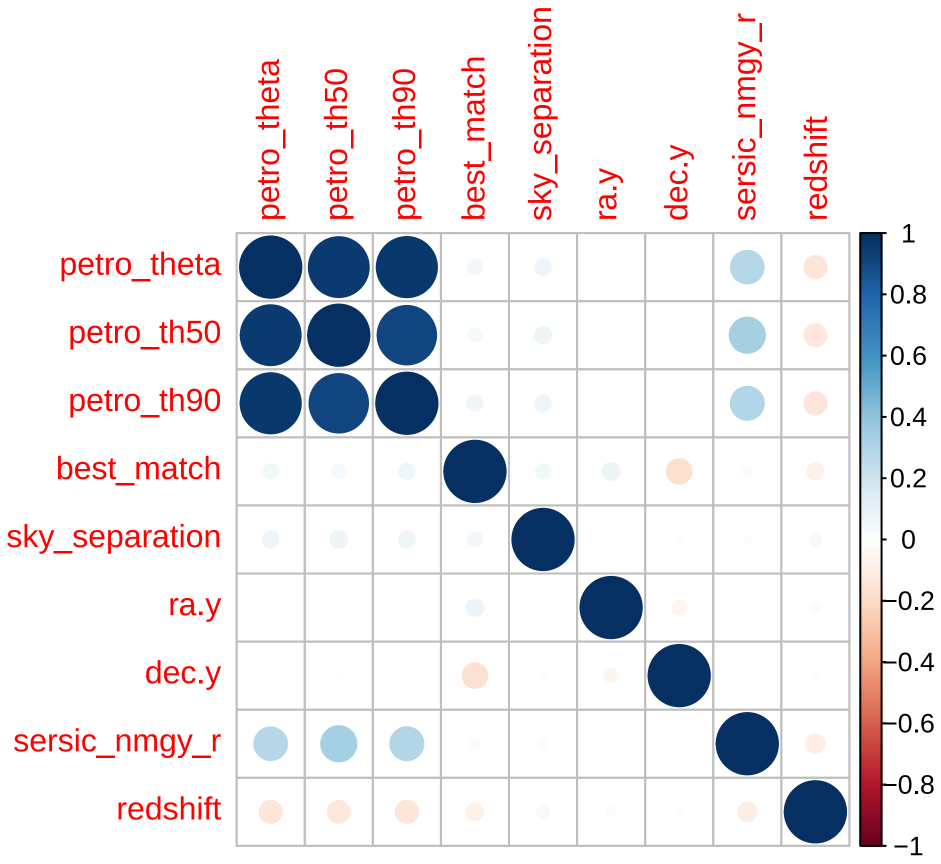


After creating a classification tree to predict how these three input variables (`petro_th50`, `petro_theta`and `petro_th90`) impact the smooth/feature of galaxies, we used `cor()` function to calculate the correlation coefficients between the variables we selected previously. Then, use a pairwise scatter plot to visualize the relationship between each variable. To do this, we need the `corrplot` library.

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
cor_result <- gz_smoothosh %>% select(c("petro_theta", "petro_th50", "petro_th90", "best_match", "sky_s
corrplot(cor_result)
```



From the pairwise scatter plot that shows the relationship between these variables, we find some dark blue circles focused on the top left. The dark blue circle represents a strong relationship between two variables(almost the same thing). It means the three variables `petro_th50`, `petro_theta` and `petro_th90` have a very strong relationship, so we should choose one to make the classification tree to predict the factor that impacts the smooth/feature in galaxies instead of including all three variables.

Based on the result from the pairwise scatter plot, we need to find the other two variables(without the three variable we chose previously) that has the most importance to impact the smooth/feature of galaxies using `variance.importance`.

```
tree <-rpart(smoothish ~ best_match + sky_separation + ra.y + dec.y + petro_theta + sersic_nmgy_r + reds

v_importance <- as.data.frame(tree$variable.importance)
v_importance
```

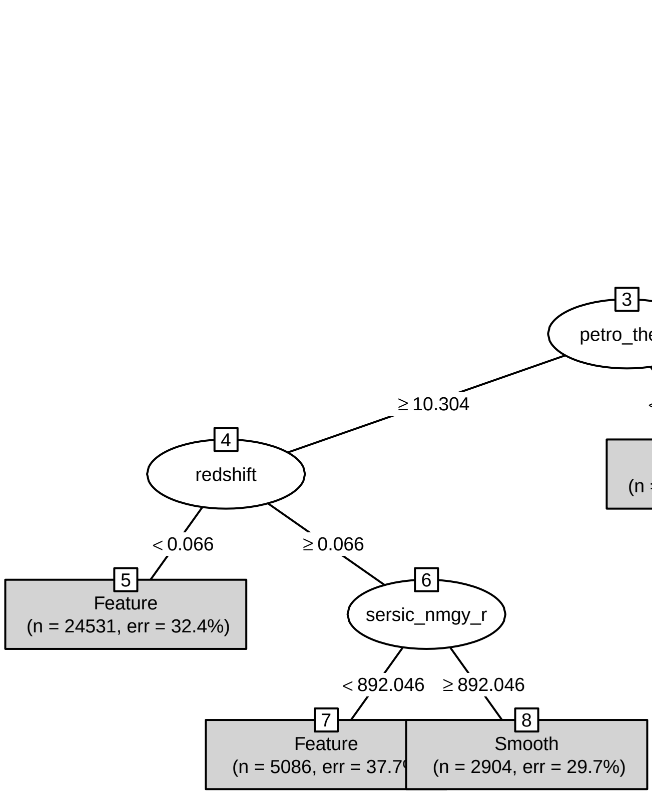
```
##                   tree$variable.importance
## petro_theta                   14691.107001
## sersic_nmgy_r                  5680.454079
## redshift                       3865.464053
## sky_separation                  738.813857
## best_match                      455.737747
## dec.y                             7.915967
## ra.y                              1.536650
```

From the data frame above, we find that the `petro_theta` and `sersic_nmgy_r`have the more significant variable importance, which means these two variables are more important for the model and can make a more accurate prediction.

Step#2: Use r-part to create a classification tree based on selected features

After we find out the response variable(smoothish) and a set of candidate predictors(petro_theta, sersic_nmgy_r and redshift) in the data set, we create a classification tree based on selected features and visualize this classification tree to make predicts an outcome.

```r
tree <-rpart(smoothish ~ petro_theta + sersic_nmgy_r + redshift, data = gz_smoothosh)
plot(as.party(tree), gp=gpar(cex=0.6), type="simple")
```



Step#3: Model evaluation with confusion matrix

We are creating the training and testing datasets from the "gz_smoothosh" data set and fitting a tree using the training data. Finally, we make predictions for new observations and create a confusion matrix to help us analyze the classification tree based on the value of accuracy, sensitivity and specificity, which are calculated from the confusion matrix.

```r
gz_smoothosh <- gz_smoothosh %>% rowid_to_column()
n <- nrow(gz_smoothosh)
set.seed(99)

training_indices <-sample(1:n, size = round(0.8 * n))
train <- gz_smoothosh %>% filter(rowid %in% training_indices)
test <- gz_smoothosh %>% filter(!(rowid %in% training_indices))

tree <-rpart(smoothish ~ petro_theta + sersic_nmgy_r + redshift, data = train)
tree_pred <- predict(tree, newdata = test)%>%
  as_tibble()%>%
  mutate(prediction = ifelse(Smooth >= 0.5, "Predict Smooth", "Predict Feature"))
table(tree_pred$prediction, test$smoothish)
```

```
##
##                   Feature Smooth
##   Predict Feature    4033   2155
##   Predict Smooth     5961  26205
```