cho_research

19. June 2017

The research is focused on the pronunciation of the word “что” in Russian dialects.

The data was taken from the Russian National Dialect Corpus.

In total, there were 99 observations.

Columns data: variant, region, gender, age, birth year, position, yat before hard consonants, yat before soft consonants, type of vocalism, realisation of /г/, /ц/ and /ч/.

We hypothesize that pronunciation of “что” depends on all the above-mentioned data.

```
cho <- read.csv("C:/Users/1/Desktop/cho_dataset.csv", sep = ";")
```

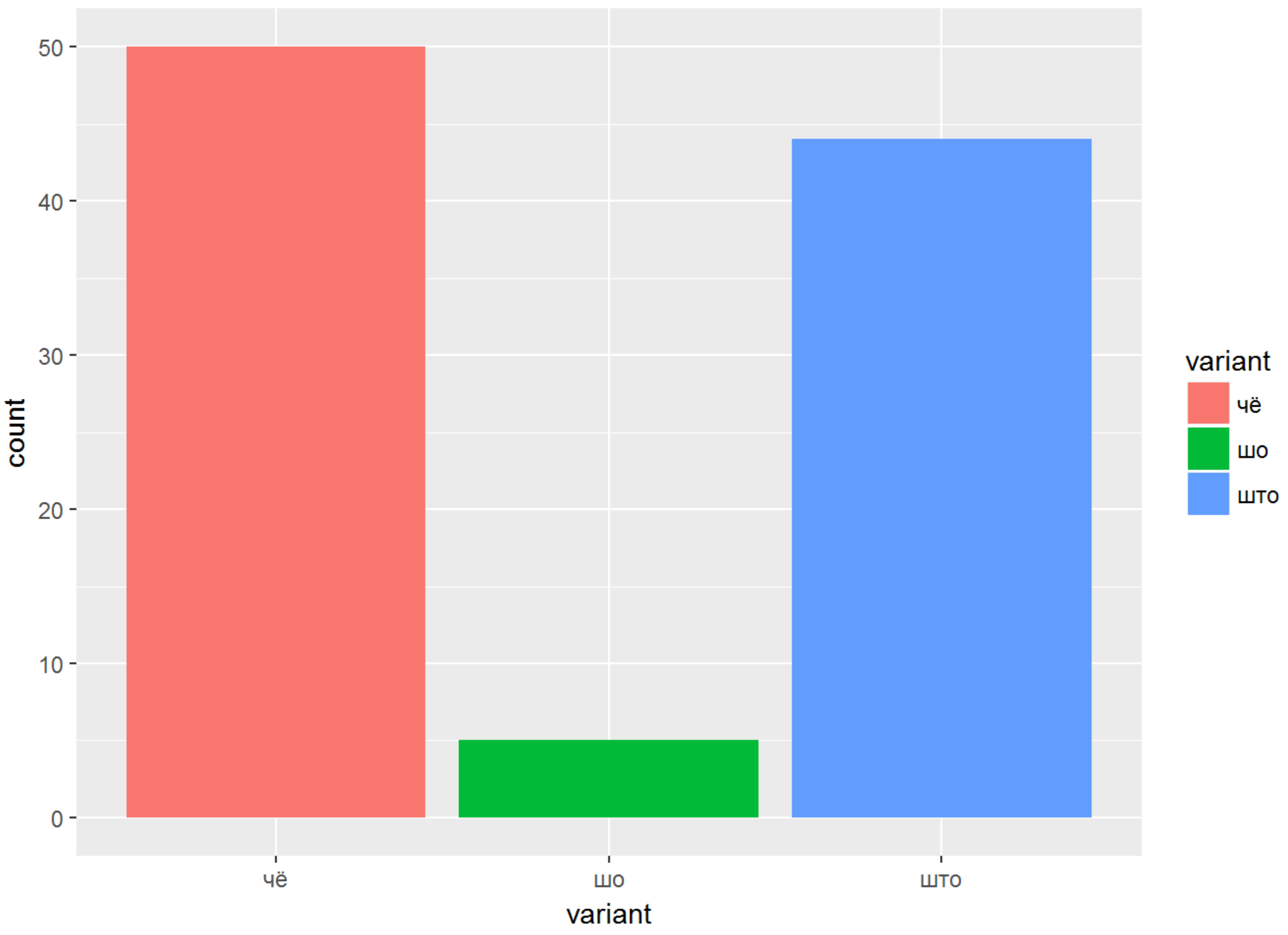
```
library(tidyverse)
```

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ----------------------------------------------
```

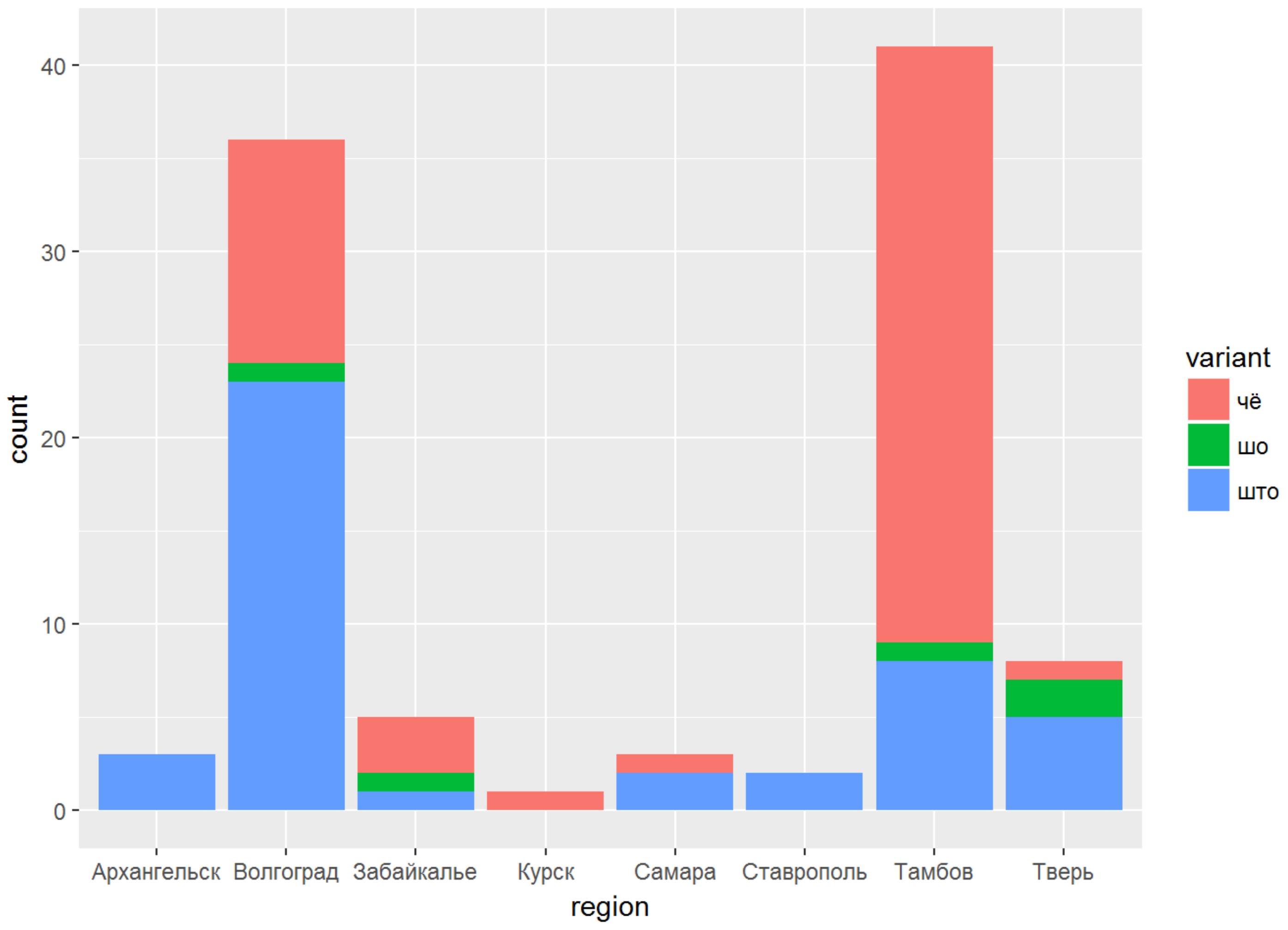
```
## filter(): dplyr, stats
## lag():    dplyr, stats
```

```
cho %>%
  ggplot(aes(variant, fill = variant))+
  geom_bar()
```



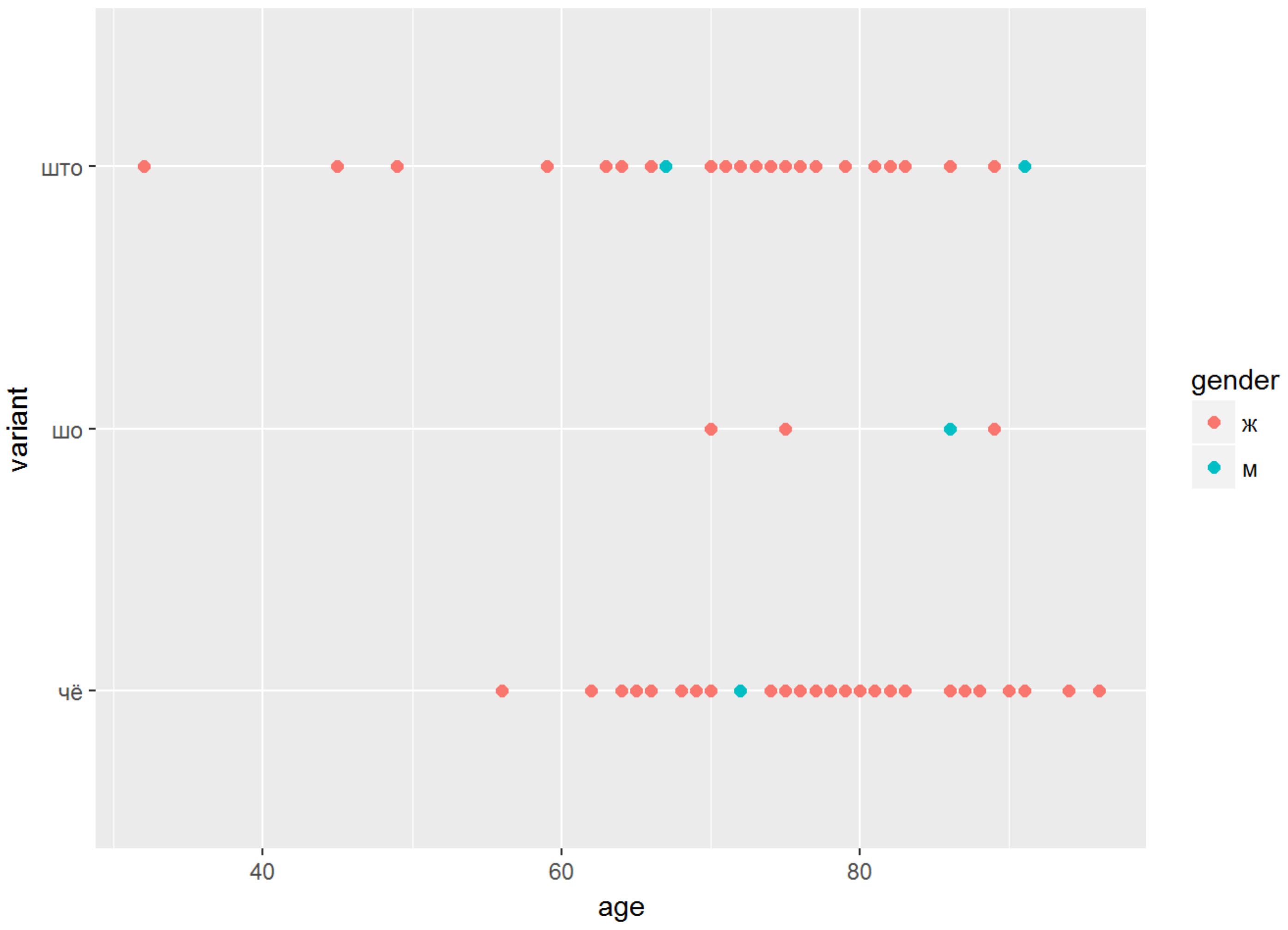
On this graph, we can see distribution of three possible variants of “что” pronunciation in the dialect corpus: “чё”, “шо” and “што”. 50 cases of “чё”, 44 cases of “што” and 5 cases of “шо”.

```
cho %>%
  ggplot(aes(region, fill = variant)) +
  geom_bar()
```



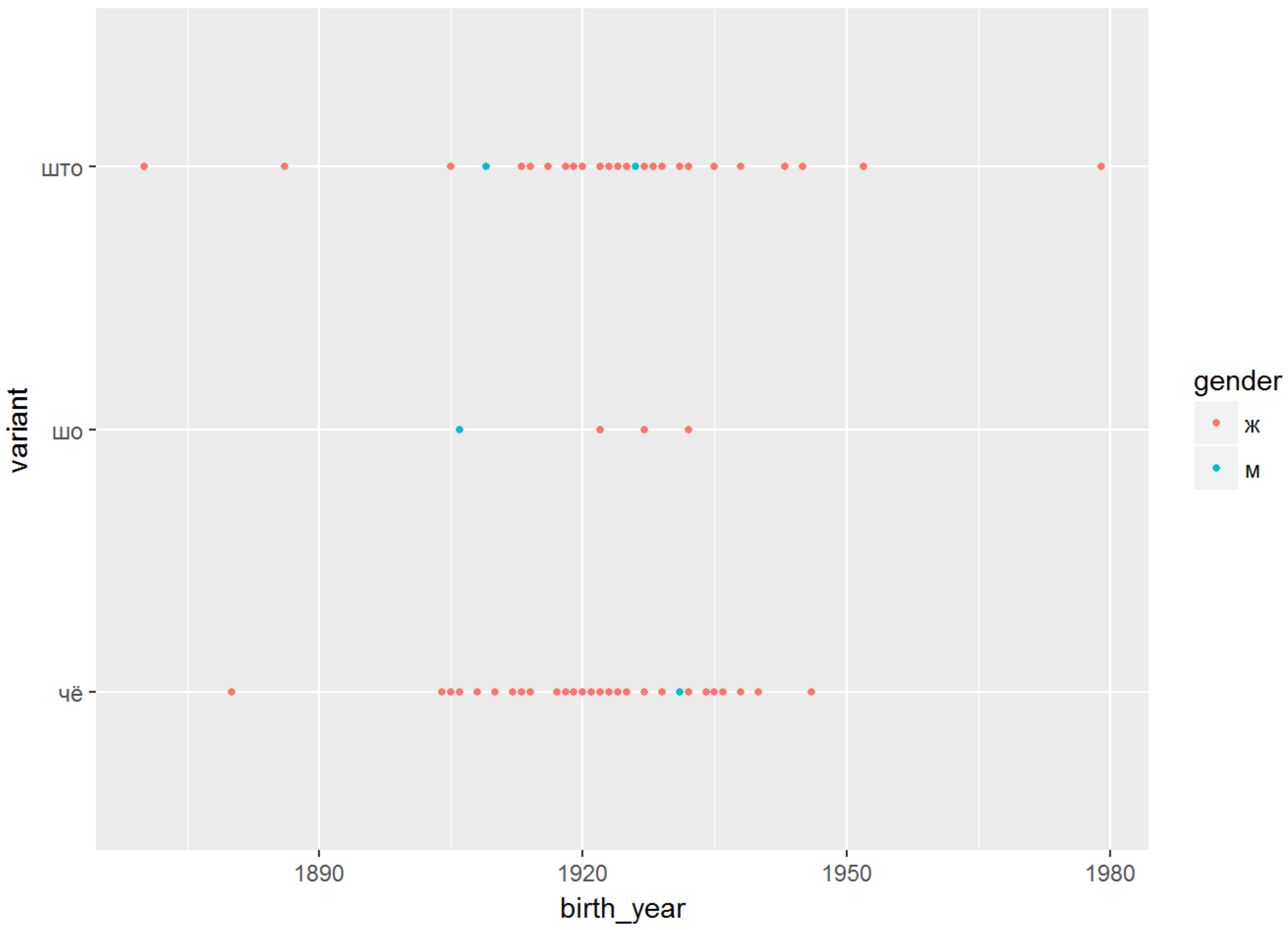
This graph illustrates which regions are included in the dialect corpus and which kinds of “что” pronunciation exist in these regions. Although the dialect variant “чё” is leading in the whole corpus (50 cases), in 5 regions out of 8 the non-dialect variant “што” is predominant.

```
cho %>%
  ggplot(aes(age, variant, colour = gender)) +
  geom_point(size = 2)
```



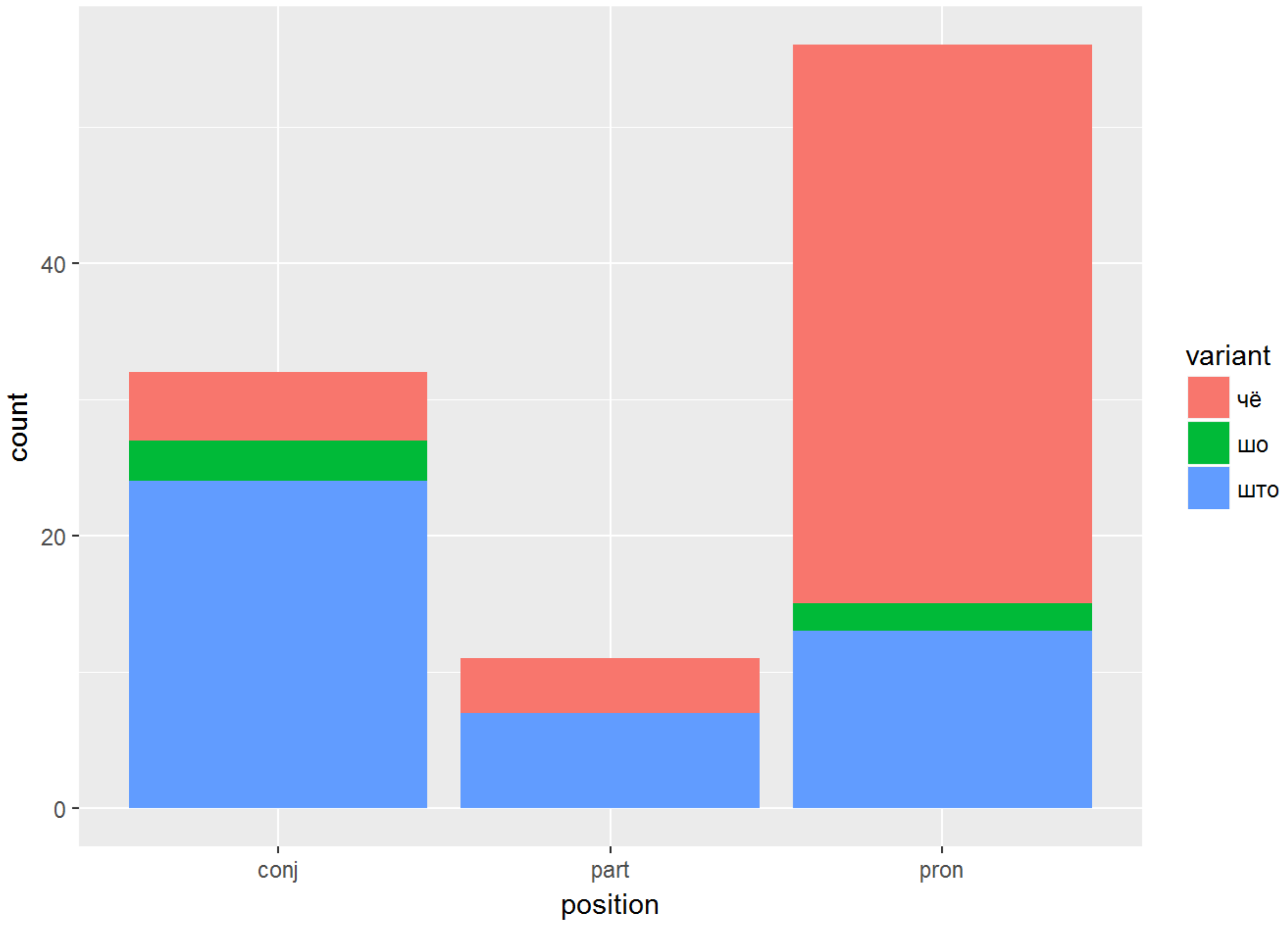
This graph shows the gender and the age of the informants. 96% of the informants are female. The median age is 76. The maximum age is 96. The minimum age is 32.

```
cho %>%
  ggplot(aes(birth_year, variant, colour = gender)) +
  geom_point(size = 1)
```



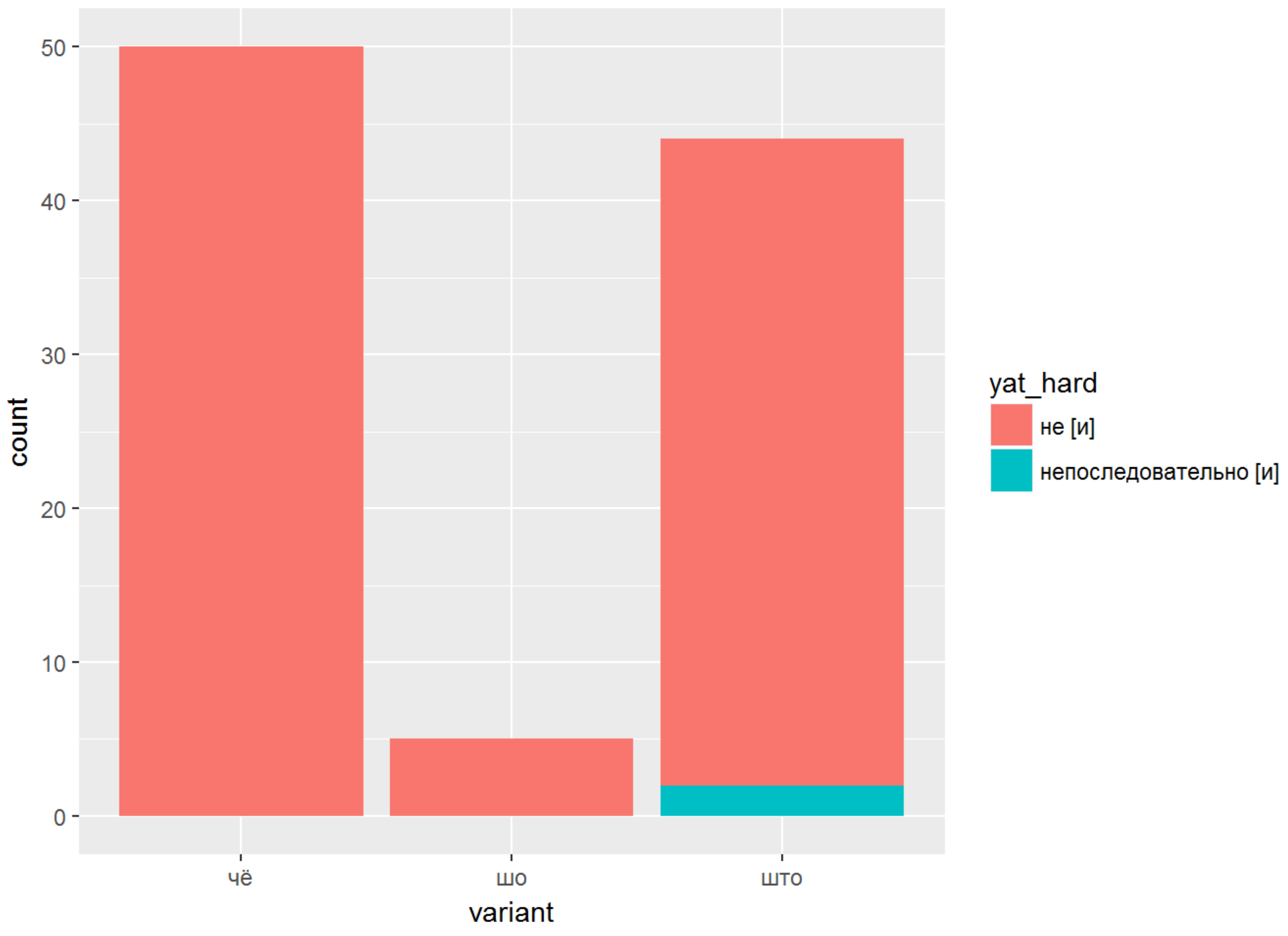
Here is a distribution of our informants according to their age of birth. The median year of birth is 1924. The earliest year of birth is 1870. The latest year of birth is 1979.

```
cho %>%
  ggplot(aes(position, fill = variant)) +
  geom_bar()
```

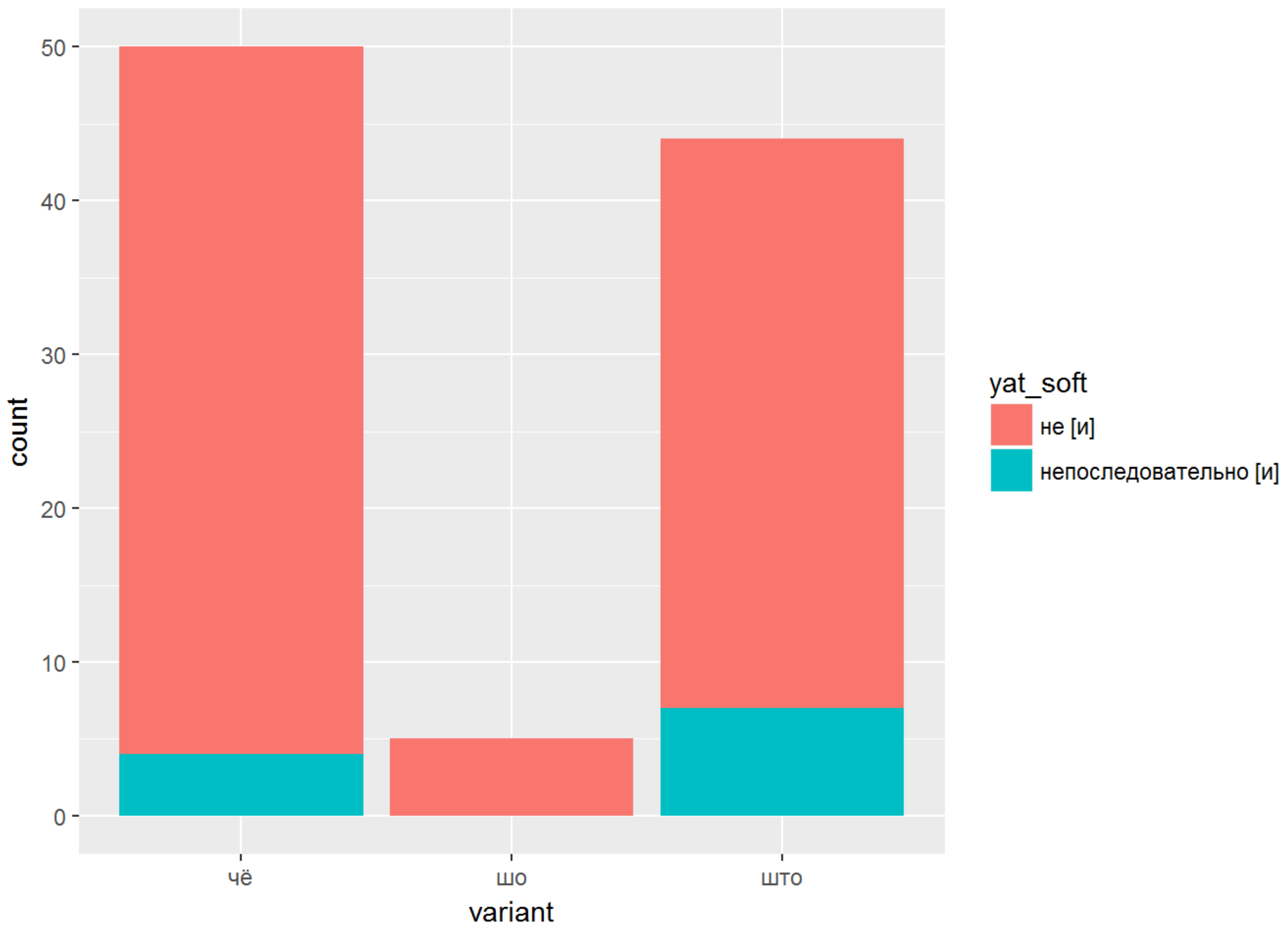


We also explored which parts of speech “что”-words are in each case. In the corpus, there are 56 pronouns, 32 conjunctions and 11 particles. According to the graph, pronouns are mostly pronounced as “чё”, while conjunctions and particles as “што”.

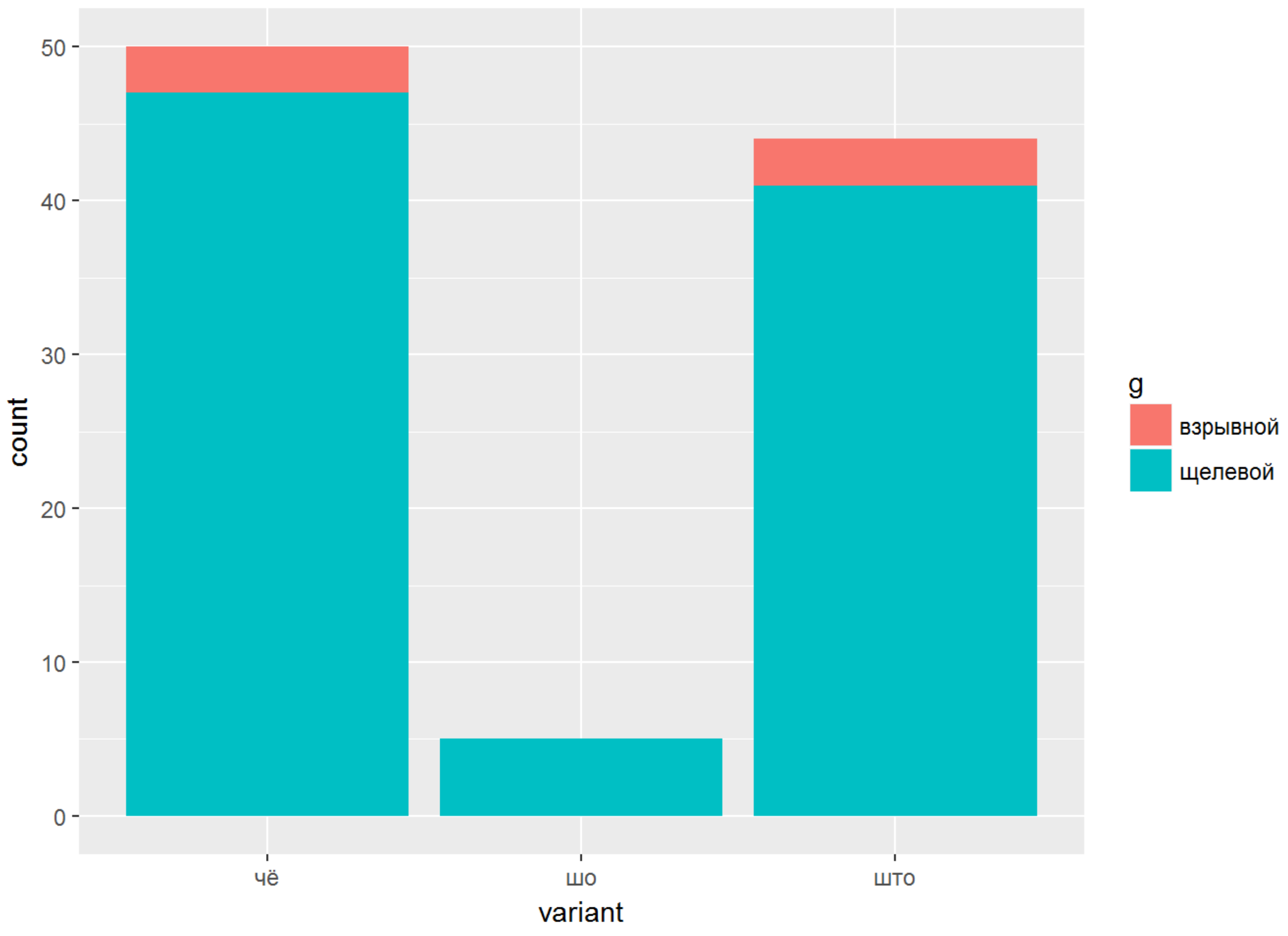
```
cho %>%
  ggplot(aes(variant, fill = yat_hard)) +
  geom_bar()
```



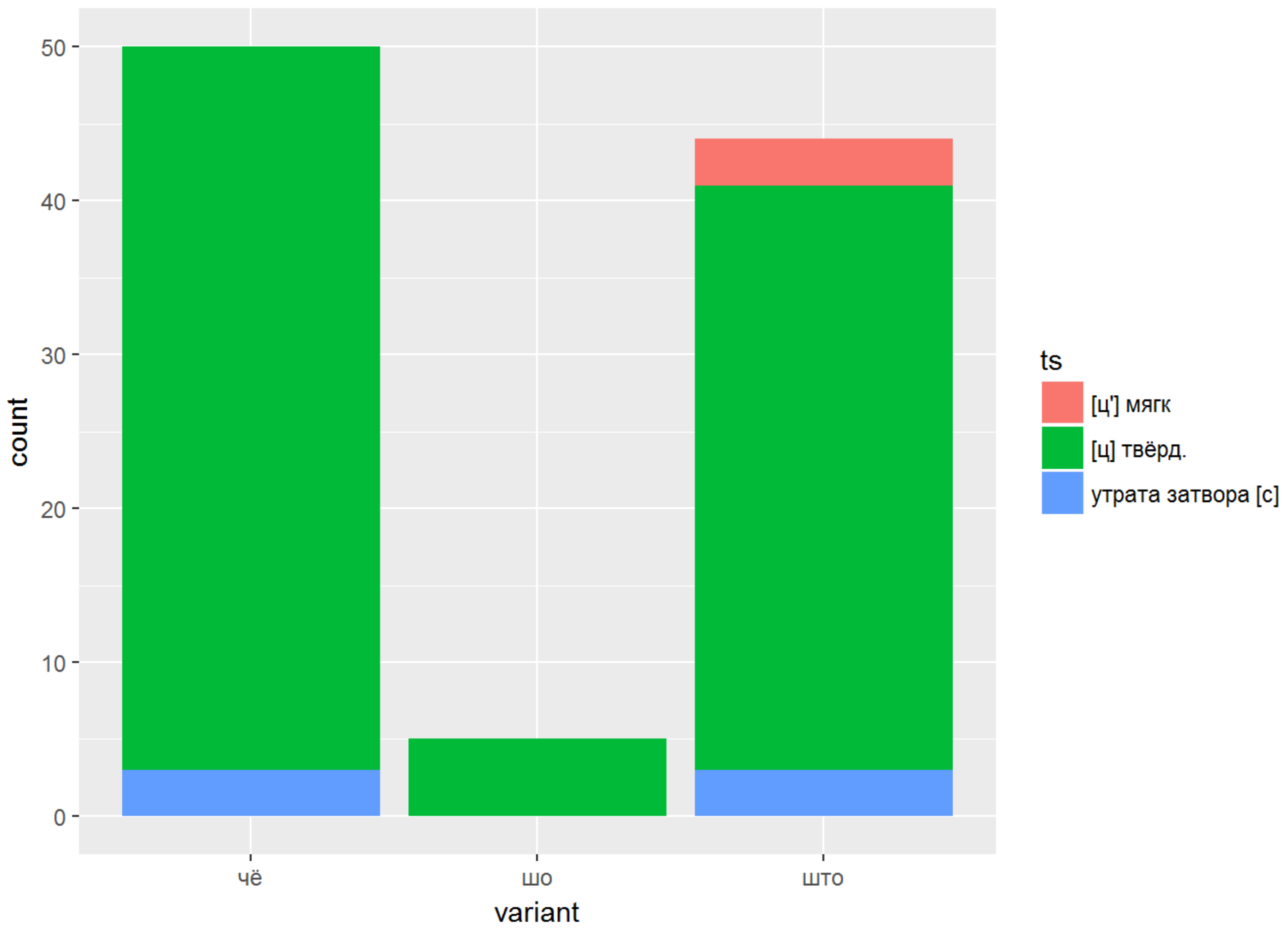
```
cho %>%
  ggplot(aes(variant, fill = yat_soft)) +
  geom_bar()
```



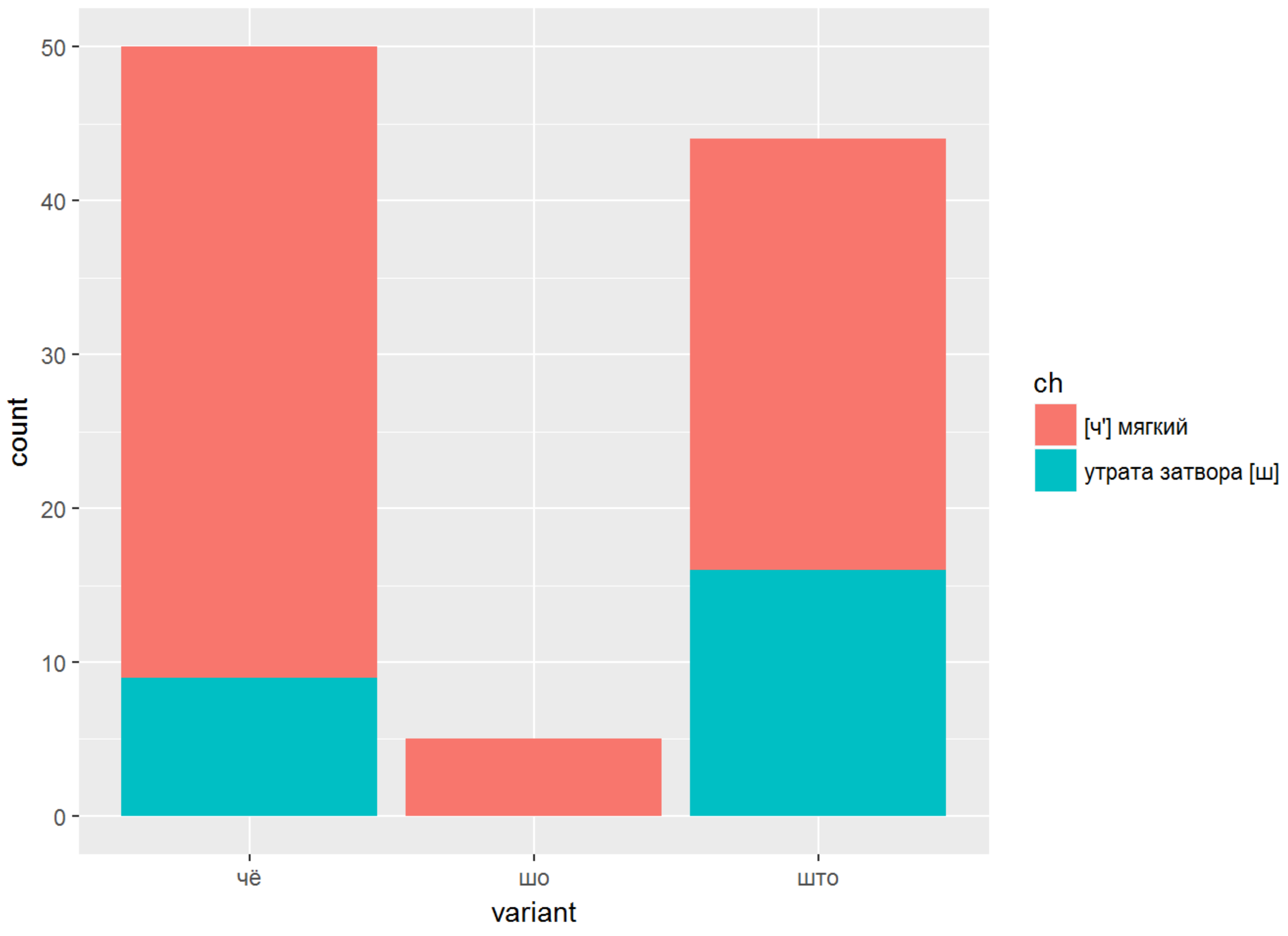
```
cho %>%
  ggplot(aes(variant, fill = g)) +
  geom_bar()
```



```
cho %>%
  ggplot(aes(variant, fill = ts)) +
  geom_bar()
```



```
cho %>%
  ggplot(aes(variant, fill = ch)) +
  geom_bar()
```



Having explored the dataset, we found out that there is not much difference concerning other dialect features, as we expected at the beginning. We decided not to comment each of these graphs. We can only point out that that the biggest difference was found with the last ch feature.

In order to predict one of three possible variants of pronunciation, we decided to apply decision trees. Firstly, it was necessary to define the importance of our variables.

```
library(party)
```

```
## Warning: package 'party' was built under R version 3.3.3
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
## Loading required package: strucchange
```

```
## Warning: package 'strucchange' was built under R version 3.3.3
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 3.3.3
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
## Warning: package 'sandwich' was built under R version 3.3.3
```

```
cho$region_num <- as.factor(cho$region_num)
set.seed(123)
train <- 1:68
train.cho <- cho[train,]
test.cho <- cho[-train,]
tr <- cforest(variant~region_num + gender + age + birth_year + position + yat_hard
+ yat_soft + vocalism + g + ts + ch, data = train.cho,
           controls = cforest_unbiased(ntree=1000, mtry=3))
varimp(tr)
```

```
## region_num     gender        age birth_year   position   yat_hard
##    0.06236    0.00000    0.01136   -0.00080    0.07268    0.00000
##   yat_soft   vocalism          g         ts         ch
##    0.00092    0.00000    0.00000    0.00012    0.00180
```

According to the results, we choosed the following features for our model: region, age and position.

```
test_answers <- test.cho$variant
tr <- cforest(variant~region_num + age + position, data = train.cho,
           controls = cforest_unbiased(ntree=1000, mtry=3))
predictions <- predict(tr, newdata=test.cho)
results <- test_answers == predictions
results
```

```
##  [1]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE
## [12]  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE  TRUE
## [23]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
```

```
results_true <- results[results == TRUE]
accuracy  <- length(results_true) / length(test_answers)
accuracy
```

```
## [1] 0.6129032
```

The accuracy of the model was 0.61.

Then we decided to delete all the “шо” variants from the corpus, in order to make the choice of our model simpler.

```
cho <- read.csv("C:/Users/1/Desktop/cho_dataset.csv", sep = ";")
cho <- cho[-c(34, 35, 36, 86, 87),]
train <- 1:65
train.cho <- cho[train,]
test.cho <- cho[-train,]
test_answers <- test.cho$variant
tr <- cforest(variant~region_num + age + position, data = train.cho,
           controls = cforest_unbiased(ntree=1000, mtry=3))
predictions <- predict(tr, newdata=test.cho)
results <- test_answers == predictions
results
```

```
##  [1]  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [12]  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
## [23]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
```

```
results_true <- results[results == TRUE]
accuracy  <- length(results_true) / length(test_answers)
accuracy
```

```
## [1] 0.7586207
```

The result was better. The accuracy of the model became 0.76.

Finally, we made a decision to also delete several outliers (age).

```
cho <- read.csv("C:/Users/1/Desktop/cho_dataset.csv", sep = ";")
cho <- cho[-c(34, 35, 36, 49, 50, 63, 86, 87),]
train <- 1:62
train.cho <- cho[train,]
test.cho <- cho[-train,]
test_answers <- test.cho$variant
tr <- cforest(variant~region_num + age + position, data = train.cho,
           controls = cforest_unbiased(ntree=1000, mtry=3))
predictions <- predict(tr, newdata=test.cho)
results <- test_answers == predictions
results
```

```
##  [1]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [12]  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
## [23]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE
```

```
results_true <- results[results == TRUE]
accuracy  <- length(results_true) / length(test_answers)
accuracy
```

```
## [1] 0.7586207
```

The accuracy of the model did not change. So the best accuracy with decision trees was after deleting “шо”, but without deleting outliers.

Apart from CART, we used KNN for classification. Before applying the method, we normalized our data:

```
library(class)
cho <- read.csv("C:/Users/1/Desktop/cho_dataset.csv", sep = ";")
myvars <- c("variant_num", "region_num", "gender_num", "age", "birth_year", "posit
ion", "yat_hard_num", "yat_soft_num", "vocalism_num", "g_num", "ts", "ch_num")
cho.subset <- cho[myvars]
cho_new <- cho.subset
cho$variant_num <- as.factor(cho$variant_num)
cho_new$gender_num <- scale(cho_new$gender_num)
cho_new$age <- scale(cho_new$age)
cho_new$birth_year <- scale(cho_new$birth_year)
cho_new$yat_hard_num <- scale(cho_new$yat_hard_num)
cho_new$yat_soft_num <- scale(cho_new$yat_soft_num)
cho_new$vocalism_num <- scale(cho_new$vocalism_num)
cho_new$g_num <- scale(cho_new$g_num)
cho_new$ch_num <- scale(cho_new$ch_num)
cho_new$region_num <- as.factor(cho_new$region_num)
region <- model.matrix(~region_num + 0, data=cho_new)
region <- scale(region)
cho_new$position <- as.factor(cho_new$position)
position <- model.matrix(~position + 0, data=cho_new)
position <- scale(position)
cho_new$ts <- as.factor(cho_new$ts)
ts <- model.matrix(~ts + 0, data=cho_new)
ts <- scale(ts)
cho_new$region_num <- NULL
cho_new$position <- NULL
cho_new$ts <- NULL
cho_final <- cbind(cho_new, region)
cho_final <- cbind(cho_final, position)
cho_final <- cbind(cho_final, ts)
set.seed(123)
```

We wanted to know not only the accuracy, but also which features are the most important for our model with KNN.

```
train <- 1:68
max <- 0
for (i in 1:511){
  copy_i <- i
  copy_cho <- cho_final
  comb <- vector()
  for (j in 9:2){
    k <- copy_i %% 2
    if (k == 0){
      copy_cho[,j] <- NULL
    } else{
      comb <- c(comb, j)
    }
    copy_i <- copy_i %/% 2
  }
  train.cho <- copy_cho[train,]
  test.cho <- copy_cho[-train,]
  train.var <- copy_cho$variant_num[train]
  knn.2 <-  knn(train.cho, test.cho, train.var, k=2)
  test_answers <- test.cho$variant_num
  results <- test_answers == knn.2
  results_true <- results[results == TRUE]
  accuracy  <- length(results_true) / length(test_answers)
  if (accuracy > max){
    max <- accuracy
    best_comb <- comb
  }
}
print (max)
```

```
## [1] 0.8709677
```

```
print (best_comb)
```

```
## [1] 8 6 3
```

The best accuracy was 0.87. The features were region, position, ts (these three remained in any case), age, yat soft and g. We were not sure that ts is an important feature, so we deleted it and checked the accuracy.

```
cho_final[, 21:23] <- NULL
train <- 1:68
max <- 0
for (i in 1:511){
  copy_i <- i
  copy_cho <- cho_final
  comb <- vector()
  for (j in 9:2){
    k <- copy_i %% 2
    if (k == 0){
      copy_cho[,j] <- NULL
    } else{
      comb <- c(comb, j)
    }
    copy_i <- copy_i %/% 2
  }
  train.cho <- copy_cho[train,]
  test.cho <- copy_cho[-train,]
  train.var <- copy_cho$variant_num[train]
  knn.2 <-  knn(train.cho, test.cho, train.var, k=2)
  test_answers <- test.cho$variant_num
  results <- test_answers == knn.2
  results_true <- results[results == TRUE]
  accuracy  <- length(results_true) / length(test_answers)
  if (accuracy > max){
    max <- accuracy
    best_comb <- comb
  }
}
print (max)
```

```
## [1] 0.8709677
```

```
print (best_comb)
```

```
## [1] 3
```

The accuracy did not change. The features were region, position, age and yat_soft.

Then we tried one neighbour with these features.

```
notvars <- c("gender_num", "birth_year", "yat_hard_num", "vocalism_num", "g_num",
"ts", "ch_num")
cho_final[notvars] <- NULL
train <- 1:68
train.cho <- cho_final[train,]
test.cho <- cho_final[-train,]
train.var <- cho_final$variant_num[train]
knn.1 <-  knn(train.cho, test.cho, train.var, k=1)
results <- test_answers == knn.1
results
```

```
##  [1]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [12]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE  TRUE  TRUE  TRUE
## [23]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```
results_true <- results[results == TRUE]
accuracy  <- length(results_true) / length(test_answers)
accuracy
```

```
## [1] 0.8709677
```

The accuracy was 0.87.

Finally, we deleted “шо”-variants and get our best accuracy with KNN, which was 0.93:

```
cho_final <- cho_final[-c(34, 35, 36, 86, 87),]
train <- 1:65
train.cho <- cho_final[train,]
test.cho <- cho_final[-train,]
train.var <- cho_final$variant_num[train]
test.var <- cho_final$variant_num[-train]
knn.1 <-  knn(train.cho, test.cho, train.var, k=1)
test_answers <- test.cho$variant_num
results <- test_answers == knn.1
results
```

```
##  [1]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [12]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [23]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```
results_true <- results[results == TRUE]
accuracy  <- length(results_true) / length(test_answers)
accuracy
```

```
## [1] 0.9310345
```

We chose region, position and age to built linear regression model. We also included realisation of /ч/ as the most important feature among other dialect features according to the graphs and the results of varimp() with decision trees to check whether it will be important for our model. We deleted lines with “шо”.

```
cho_1 <- read.csv('C:/Users/1/Desktop/cho_dataset.csv', header = TRUE, sep=";", st
ringsAsFactors = FALSE)
cho <- cho_1[-c(34, 35, 36, 63, 86, 87),]

cho_dataset <- function(f) {
  cho <- cho

  cho$variant <- as.factor(cho$variant)
  cho$ch <- as.factor(cho$ch)
  cho$position <- as.factor(cho$position)

  cho <- cho[,c("variant","ch","position","age","region_num")]
  cho <- cho[complete.cases(cho),]

  return(cho)
}

cho <- cho_dataset(cho)


str(cho)
```

```
## 'data.frame':    93 obs. of  5 variables:
##  $ variant   : Factor w/ 2 levels "чё","што": 1 1 1 1 1 1 1 1 1 1 ...
##  $ ch        : Factor w/ 2 levels "[ч'] мягкий",..: 1 1 1 1 1 1 1 1 1 2 ...
##  $ position  : Factor w/ 3 levels "conj","part",..: 3 2 3 1 3 3 3 3 3 3 ...
##  $ age       : int  91 75 75 75 81 80 96 70 83 68 ...
##  $ region_num: int  2 3 3 3 7 7 7 7 7 7 ...
```

The model was trained on N-10 cases, while remaining 10 were used for testing.

```
log_reg <- function(df, size=10) {
  N <- nrow(df)
  size=10

  df <- df[sample(N),]

  num <- floor(N/size)
  rest <- N - num * size
  ncv <- cumsum(c(rep(size,num), rest))

  predictions <- data.frame(variant = df$variant, pred = NA)

  for(n in ncv) {
    v <- rep(TRUE, N)
    v[(n-size+1):n] <- FALSE

    lr <- glm(variant ~ ., data = df[v,], family = binomial(logit))
    predictions[!v,"pred"] <- predict(lr, newdata=df[!v,], type="response")
  }

  return(predictions)
}

predictions <- log_reg(cho, size=10)
str(predictions)
```

```
## 'data.frame':    93 obs. of  2 variables:
##  $ variant: Factor w/ 2 levels "чё","што": 1 1 2 2 2 2 2 2 1 2 ...
##  $ pred   : num  0.337 0.161 0.768 0.438 0.477 ...
```

```
lr <- glm(variant ~ ., data = cho, family = binomial(logit))

summary(lr)
```

```
##
## Call:
## glm(formula = variant ~ ., family = binomial(logit), data = cho)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2924  -0.6497  -0.4276   0.6348   2.1173
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)            4.74328    2.44187   1.942   0.0521 .
## chутрата затвора [ш]  0.43269    0.64377   0.672   0.5015
## positionpart          -0.43055    0.85578  -0.503   0.6149
## positionpron          -2.60660    0.62623  -4.162 3.15e-05 ***
## age                   -0.03102    0.02999  -1.034   0.3011
## region_num            -0.24162    0.11763  -2.054   0.0400 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 128.398  on 92  degrees of freedom
## Residual deviance:  89.493  on 87  degrees of freedom
## AIC: 101.49
##
## Number of Fisher Scoring iterations: 4
```

```
exp(lr$coefficients)
```

```
##          (Intercept) chутрата затвора [ш]         positionpart
##         114.81048157           1.54139155           0.65015261
##         positionpron                  age           region_num
##           0.07378482           0.96945890           0.78535579
```

The results say that ch type is not important for logistic regression model (>1).

```
lr_reduced <- glm(variant ~ position + age + region_num , data = cho, family = bin
omial(logit))

summary(lr_reduced)
```

```
##
## Call:
## glm(formula = variant ~ position + age + region_num, family = binomial(logit),
##     data = cho)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1987  -0.6562  -0.4385   0.6565   2.0951
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   5.00386    2.41900   2.069   0.0386 *
## positionpart -0.46362    0.85743  -0.541   0.5887
## positionpron -2.61098    0.62269  -4.193 2.75e-05 ***
## age          -0.03099    0.03003  -1.032   0.3021
## region_num   -0.27111    0.10971  -2.471   0.0135 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 128.398  on 92  degrees of freedom
## Residual deviance:  89.944  on 88  degrees of freedom
## AIC: 99.944
##
## Number of Fisher Scoring iterations: 4
```

```
exp(lr_reduced$coefficients)
```

```
##  (Intercept) positionpart positionpron          age   region_num
## 148.98719728   0.62900353   0.07346225   0.96948950   0.76253136
```

Then we compared two models with and without ch variable.

```
anova(lr, lr_reduced, test = 'Chisq')
```

```
## Analysis of Deviance Table
##
## Model 1: variant ~ ch + position + age + region_num
## Model 2: variant ~ position + age + region_num
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1        87     89.493
## 2        88     89.944 -1 -0.45157   0.5016
```

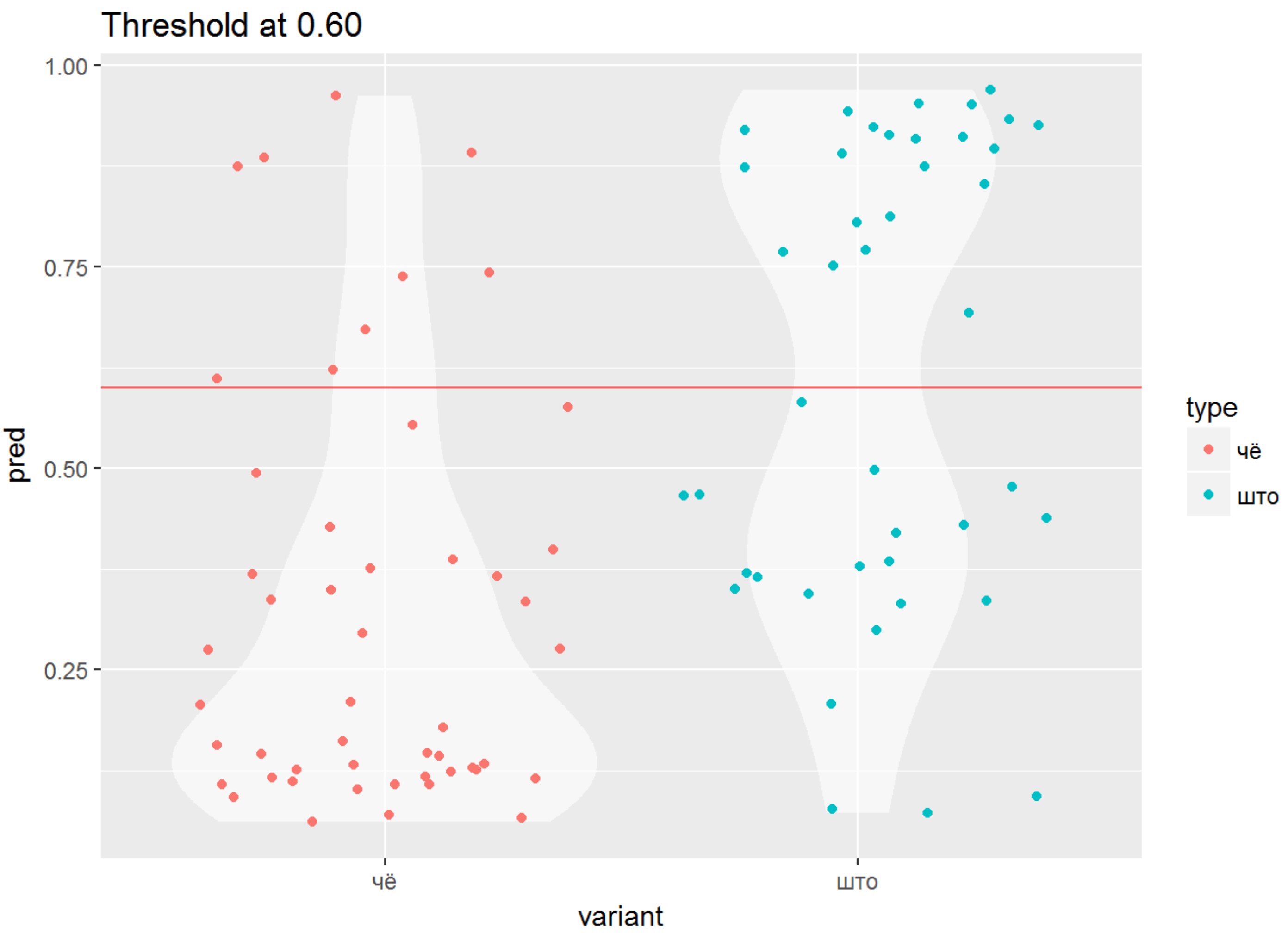
If we compare two models with and without ch-type factor, we see, that there is no significant difference between them (Pr(>Chi) = 0.5016). So we can use a reduced model. The results mean that ch-type is actualy not important.

```
library('tidyverse')
plot_pred_type_distribution <- function(df, threshold) {
  v <- rep(NA, nrow(df))
  v <- ifelse(df$pred >= threshold & df$variant == 1, "TP", v)
  v <- ifelse(df$pred >= threshold & df$variant == 0, "FP", v)
  v <- ifelse(df$pred < threshold & df$variant == 1, "FN", v)
  v <- ifelse(df$pred < threshold & df$variant == 0, "TN", v)

  df$pred_type <- v

  ggplot(data=df, aes(x=variant, y=pred)) +
    geom_violin(fill=rgb(1,1,1,alpha=0.6), color=NA) +
    geom_jitter(aes(color = variant)) +
    geom_hline(yintercept=threshold, color="red", alpha=0.6) +
    scale_color_discrete(name = "type") +
    labs(title=sprintf("Threshold at %.2f", threshold))
}

plot_pred_type_distribution(predictions, 0.6)
```



Let’s consider that points closer to 1 - “што” pronunciation. And those closer to 0 - “чё”. We desided to put a threshold at 0.6. If we move up a threshold, the number of FP lowers and number of FN increases. This plot illlustrates distribution of predictions made by the model.

Conclusion: We tried different models to predict type of pronunciation of “что” in Russian dialects. The results show that not all of the variables in our dataset are significant for predictions. For instance, g, ts and vocalism have a very low influence on the что pronunciation. However, age, region and pos type are important. That means that our initial hypothesis wasn’t confirmed. According to the results, we can say that “что” pronunciation is independent from other pronunciation features described in the dataset.