R project: Prepositions

Code

Соня Никифорова, Яна Курмачева

19/06/2017

Предмет исследования: распределение предлогов “про” и “о/об” в речи.
Материал исследования: примеры из Национального Корпуса Русского Языка.

Факторы, которые рассматриваются как возможно влияющие на выбор предлога (независимые переменнные):

- регистр речи (письменный, устный);
- жанр текста (художественный, нехудожественный)*;
- часть речи слова, от которого зависит предлог (глагол, существительное);
- время создания текста (до 1950 года, после 1950 года);
- тип корня.

*Жанр для письменных текстов в НКРЯ уже задан как дихотомия “художественных текстов” и “нехудожественных текстов”; в устном корпусе под “художественными” мы понимали жанры “театральная речь” и “речь кино”, под “нехудожественными” — жанры “устная публичная речь” и “устная непубличная речь”. Жанры “авторское чтение” и “художественное чтение” было решено не учитывать, так как тексты этих жанров — это фактически озвученная письменная, а не устная речь.

У каждой из первых четырех переменных два значения (уровня), и следовательно, при работе с каждым из рассмотренных корней было задано 16 комбинаций параметров поиска. Из выдачи были взяты все примеры, если их было 50 или меньше; если примеров в выдаче было больше 50, забирались первые 50 подходящих примеров.

Рассмотренные лексемы:

- разговаривать (говорить, поговорить) / разговор
- знать (узнать) / знание
- спрашивать (спросить) / вопрос (в значении question, не issue).
- помнить (вспомнить, вспоминать) / воспоминание (память)
- шутить (пошутить) / шутка
- рассказывать (рассказать) / рассказ
- писать (написать) / письмо
- рассуждать / рассуждение
- слышать (услышать) / слух
- петь (спеть) / песня

Нулевой гипотезой является предположение о том, что единственными факторами, влияющими на выбор предлога, являются регистр и жанр текста.

В результате получен 5151 пример. Для каждого примера указано, какой предлог был использован. Данные представлены в файле prepositions.csv и проиллюстрированы в графиках.

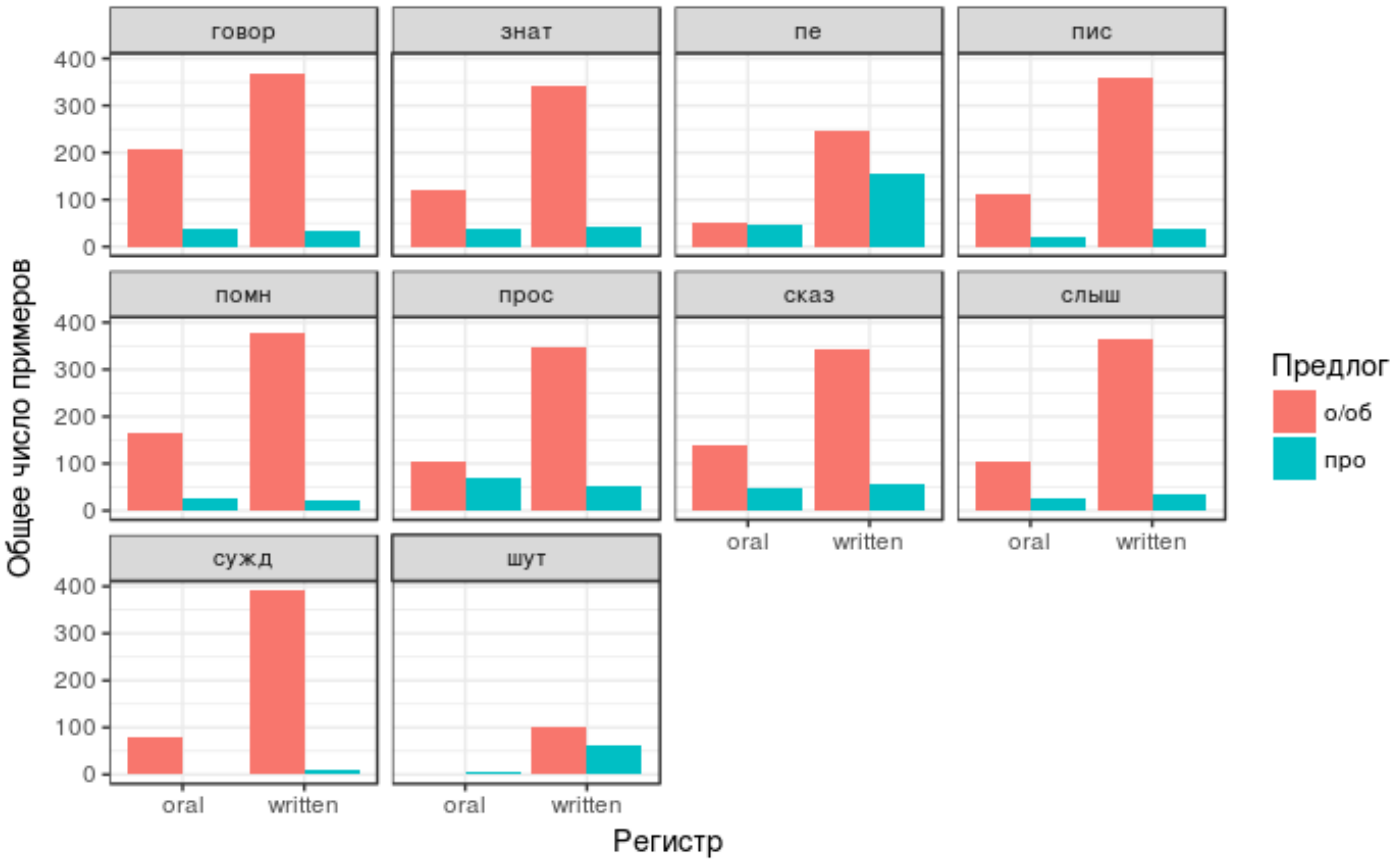
Hide

```
library(randomForest)
library(tidyverse)
library(caret)

data <- read.csv("prepositions.csv", head = TRUE, sep = ";", encoding = "UTF-8")
data <- data[, -c(7,8)]
```

Hide

```
data %>%
  ggplot(aes(register, fill = preposition)) +
  geom_bar(position = "dodge") +
  facet_wrap(~root) +
  labs(x = "Регистр", y = "Общее число примеров") +
  theme_bw() +
  guides(fill = guide_legend(title = "Предлог"))
```



Hide

```
# data %>%
#   ggplot(aes(genre, fill = preposition)) +
#   geom_bar(position = "dodge") +
#   facet_wrap(~root) +
#   labs(x = "Жанр текста", y = "Общее число примеров") +
#   theme_bw() +
#   guides(fill = guide_legend(title = "Предлог"))
```

Hide

```
# data %>%
#   ggplot(aes(date_of_creation, fill = preposition)) +
#   geom_bar(position = "dodge") +
#   facet_wrap(~root) +
#   labs(x = "Время создания текста", y = "Общее число примеров") +
#   theme_bw() +
#   guides(fill = guide_legend(title = "Предлог"))
```

Hide

```
# data %>%
#   ggplot(aes(head_pos, fill = preposition)) +
#   geom_bar(position = "dodge") +
#   facet_wrap(~root) +
#   labs(x = "Часть речи вершины", y = "Общее число примеров") +
#   theme_bw() +
#   guides(fill = guide_legend(title = "Предлог"))
```

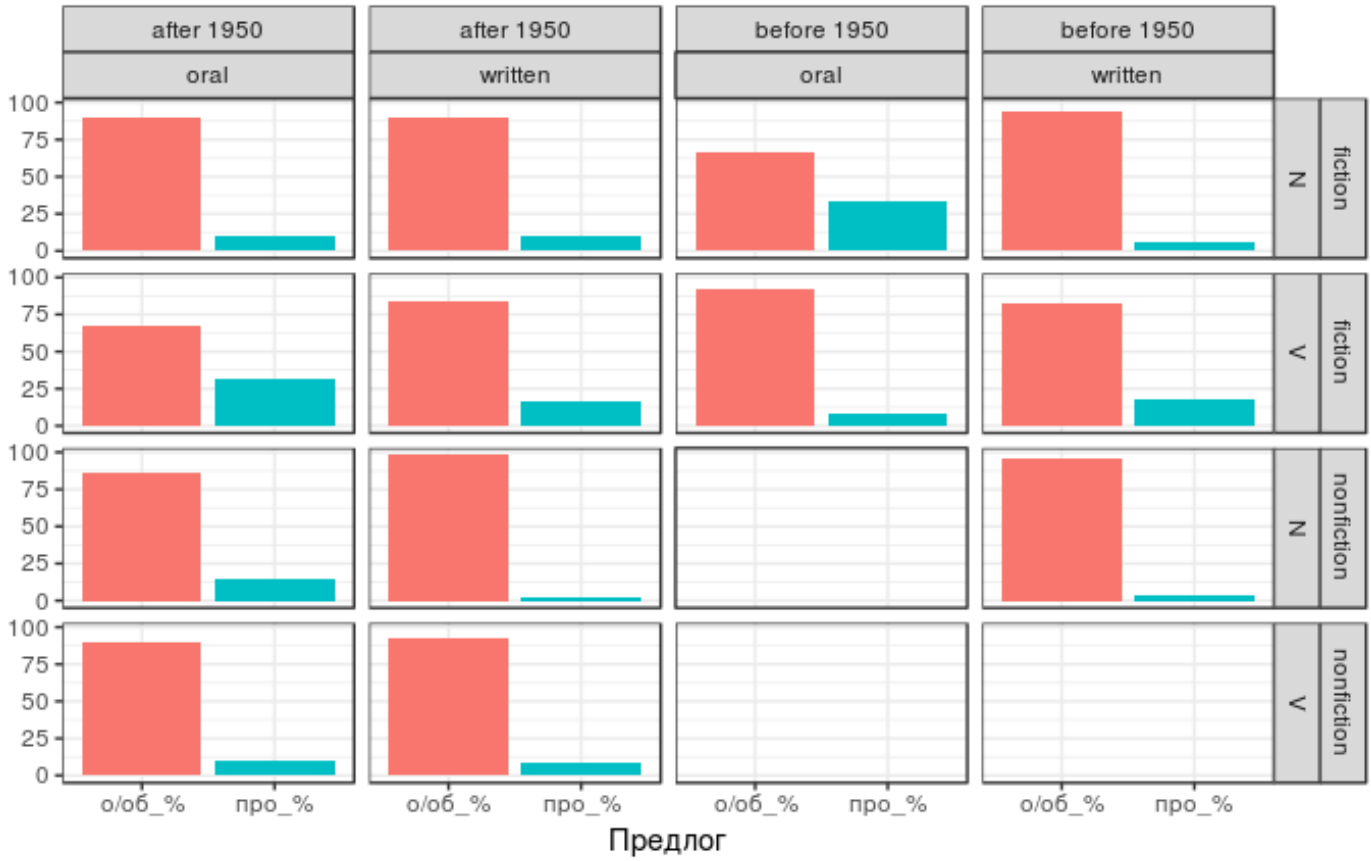
Нагляднее представить изменение соотношения предлогов в зависимости от значений параметров можно по графикам, показывающим не абсолютное количество использованных “про” и “о/об”, а проценты от общего числа примеров:

Hide

```
data %>%
  count(root, genre, head_pos, date_of_creation, register, preposition) %>%
  spread(key = preposition, value = n) %>%
  mutate(sum = `о/об`+`про`,
         `о/об_%` = `о/об`/sum*100,
         `про_%` = `про`/sum*100) %>%
  select(-c(`о/об`, `про`, sum)) %>%
  gather(key = preposition, value = percent, `о/об_%`:`про_%`) %>%
  na.omit() -> data_pct
```

Hide

```
data_pct %>%
  filter(root == "говор") %>%
  ggplot(aes(preposition, percent, fill = preposition)) +
  geom_bar(stat = "identity") +
  facet_grid(genre + head_pos ~ date_of_creation + register) +
  labs(x = "Предлог", y = " ") +
  theme_bw() +
  guides(fill = FALSE)
```



Hide

```
# data_pct %>%
#   filter(root == "знат") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Hide

```
# data_pct %>%
#   filter(root == "прос") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Hide

```
# data_pct %>%
#   filter(root == "помн") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Hide

```
# data_pct %>%
#   filter(root == "шут") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Hide

```
# data_pct %>%
#   filter(root == "сказ") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Hide

```
# data_pct %>%
#   filter(root == "пис") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Hide

```
# data_pct %>%
#   filter(root == "сужд") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Hide

```
# data_pct %>%
#   filter(root == "слыш") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Hide

```
# data_pct %>%
#   filter(root == "пе") %>%
#   ggplot(aes(preposition, percent, fill = preposition)) +
#   geom_bar(stat = "identity") +
#   facet_grid(genre + head_pos ~ date_of_creation + register) +
#   labs(x = "Предлог", y = " ") +
#   theme_bw() +
#   guides(fill = FALSE)
```

Препроцессинг данных

Данные были разделены на два сабсета - тренировочный (75%) и тестовый (25%) - с помощью метода стратифицированной выборки. Тестовый сабсет использовался при проверке обобщающей способности моделей.

Hide

```
#table(data$preposition)/sum(table(data$preposition)) #процентное соотношение пред
логов 'о/об' и 'про'
set.seed(123)
dataR <- data[order(runif(nrow(data))),]
data_split <- createDataPartition(y = dataR$preposition, p = 0.75, list = FALSE)
#head(data_split)
trainSet <- dataR[data_split, ]
testSet <- dataR[-data_split, ]
head(trainSet)
```

	genre <fctr>	register <fctr>	date_of_creation <fctr>	head_pos <fctr>	root <fctr>	preposition <fctr>
478	fiction	oral	after 1950	N	говор	о/об
74	fiction	written	after 1950	N	говор	о/об
1254	fiction	written	after 1950	V	знат	про
740	fiction	written	after 1950	N	прос	о/об
4712	fiction	written	after 1950	N	пе	о/об
4942	fiction	written	before 1950	N	пе	о/об

6 rows

Логистическая регрессия

Мы обучили модель логистической регрессии с использованием всех возможных предикторов:

Hide

```
fit <- glm(preposition ~ ., data = trainSet, family = "binomial")
summary(fit)
```

```
Call:
glm(formula = preposition ~ ., family = "binomial", data = trainSet)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7684  -0.5782  -0.3793  -0.1874   3.0390

Coefficients:
                             Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -1.942501   0.178827 -10.862  < 2e-16 ***
genrenonfiction             -0.551943   0.098399  -5.609 2.03e-08 ***
registerwritten             -0.521263   0.108399  -4.809 1.52e-06 ***
date_of_creationbefore 1950 -0.891741   0.117472  -7.591 3.17e-14 ***
head_posV                    1.126598   0.108261  10.406  < 2e-16 ***
rootзнат                     0.227708   0.207427   1.098  0.27230
rootпе                       2.144540   0.190094  11.281  < 2e-16 ***
rootпис                      0.002653   0.220135   0.012  0.99039
rootпомн                    -0.402096   0.232081  -1.733  0.08317 .
rootпрос                     0.759767   0.192715   3.942 8.07e-05 ***
rootказ                      0.568335   0.193774   2.933  0.00336 **
rootслыш                    -0.175711   0.226043  -0.777  0.43696
rootсужд                    -2.144164   0.475893  -4.506 6.62e-06 ***
rootшут                      2.158246   0.250032   8.632  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3397.0  on 3863  degrees of freedom
Residual deviance: 2780.6  on 3850  degrees of freedom
AIC: 2808.6

Number of Fisher Scoring iterations: 7
```

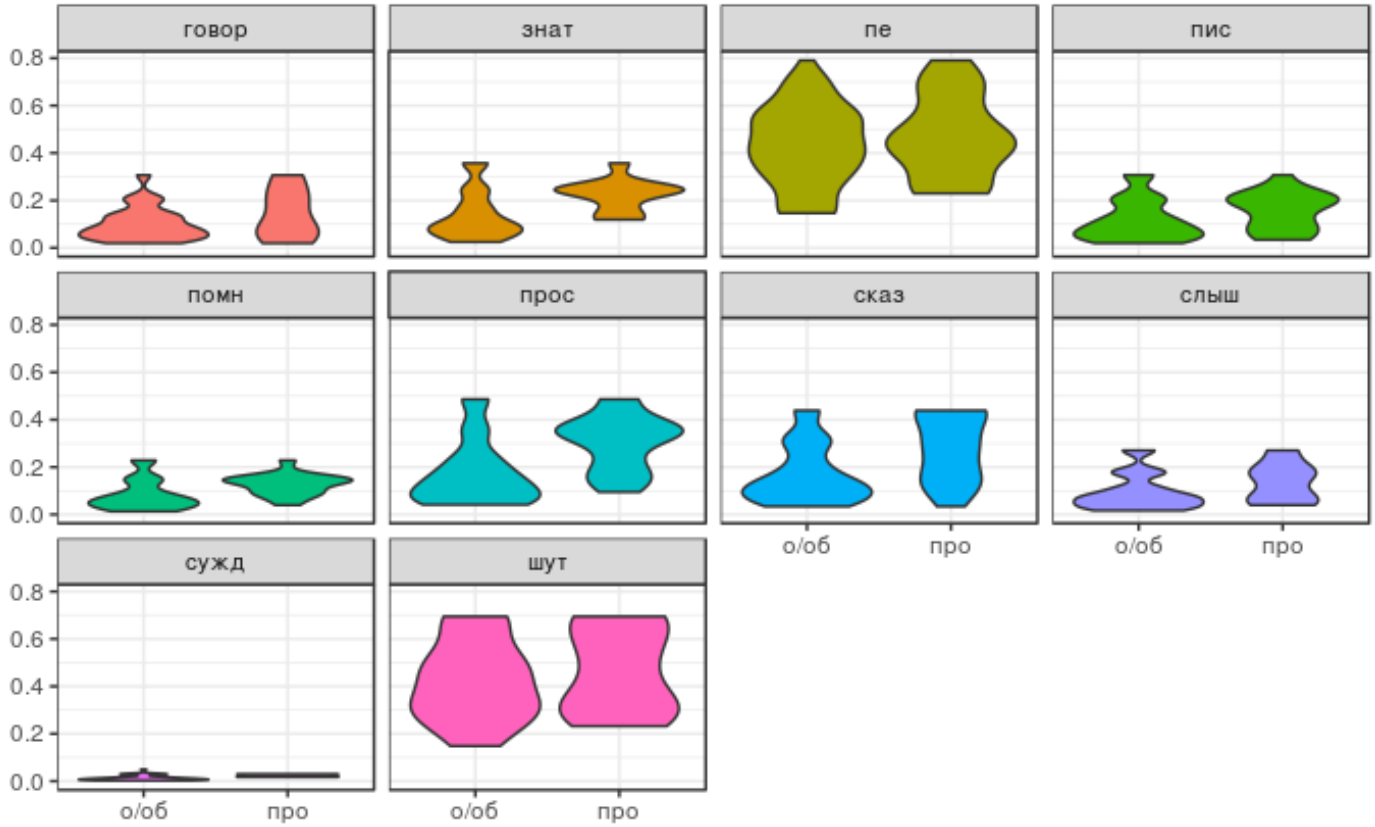
Как видно, бОльшую часть признаков алгоритм считает значимыми для предсказания предлога. Проиллюстрируем его предсказательные возможности с помощью violin plots:

Hide

```
rd <- data.frame(p = predict(fit, newdata = testSet, type = "response"), prep = te
stSet$preposition, rt = testSet$root, hp = testSet$head_pos, gnr = testSet$genre,
rgstr = testSet$register, dt = testSet$date_of_creation)
```

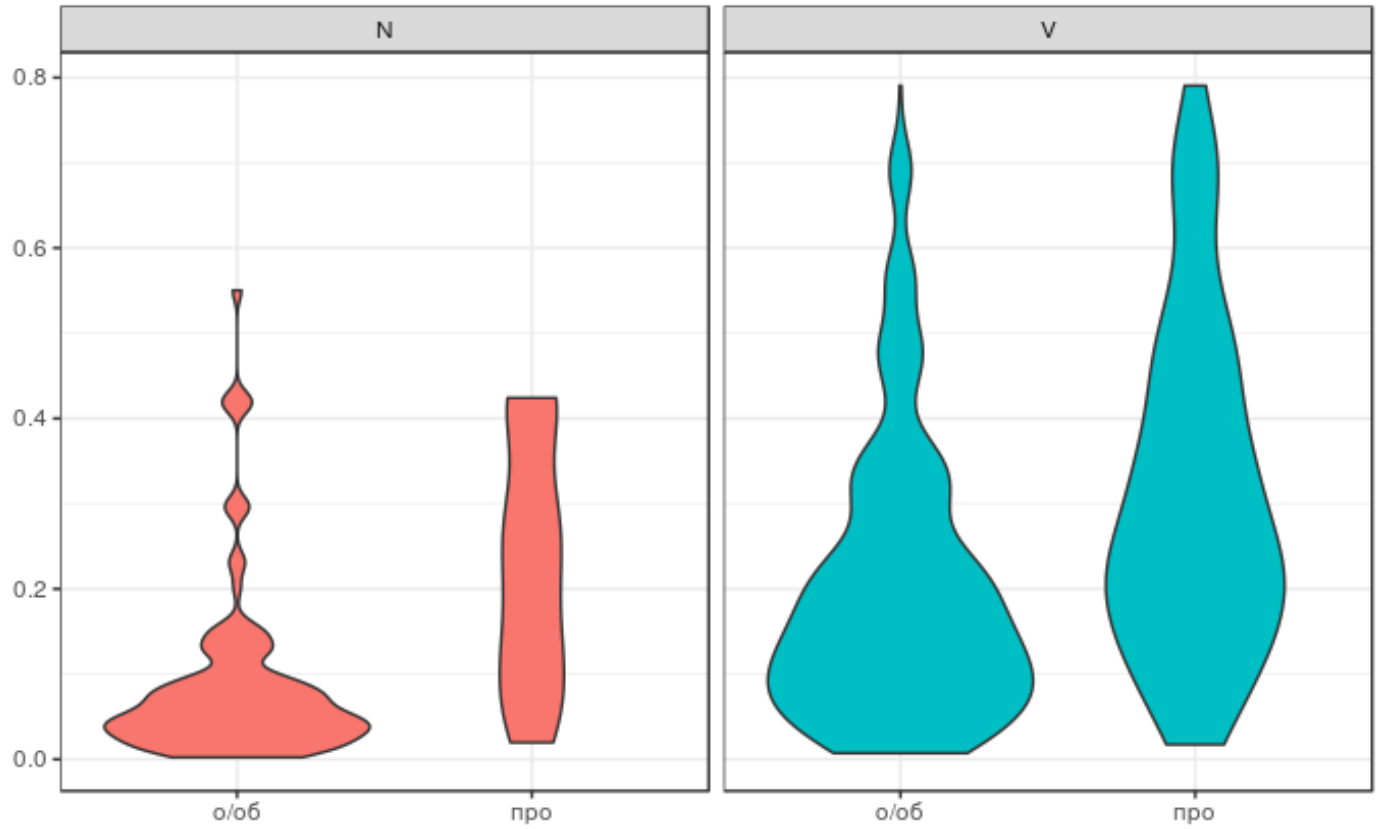
Hide

```
rd %>%
  arrange(p) %>%
  ggplot(aes(prep, p, fill = rt)) +
  geom_violin() +
  facet_wrap(~rt) +
  labs(x = "", y = "") +
  theme_bw() +
  guides(fill = FALSE)
```



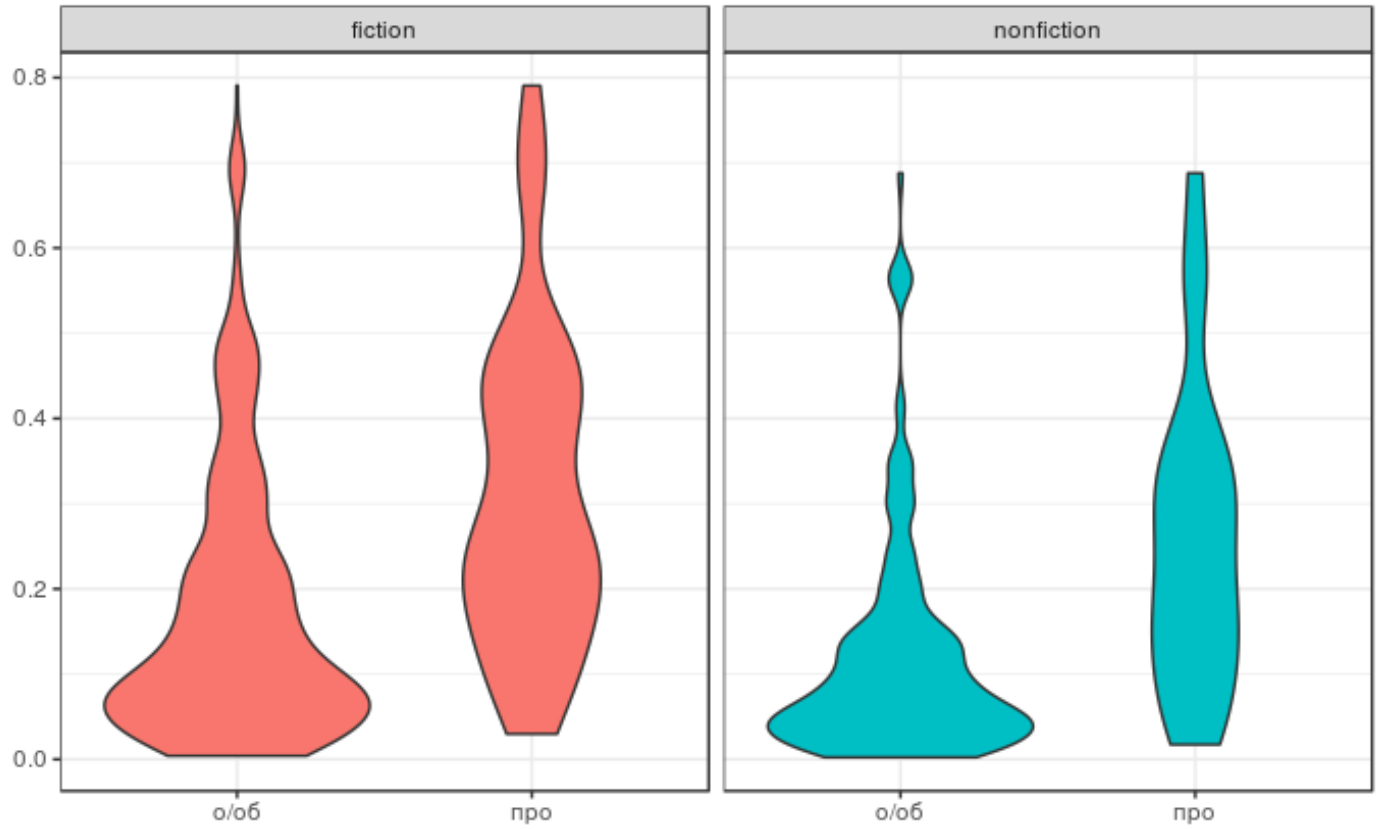
Hide

```
rd %>%
  arrange(p) %>%
  ggplot(aes(prep, p, fill = hp)) +
  geom_violin() +
  facet_wrap(~hp) +
  labs(x = "", y = "") +
  theme_bw() +
  guides(fill = FALSE)
```



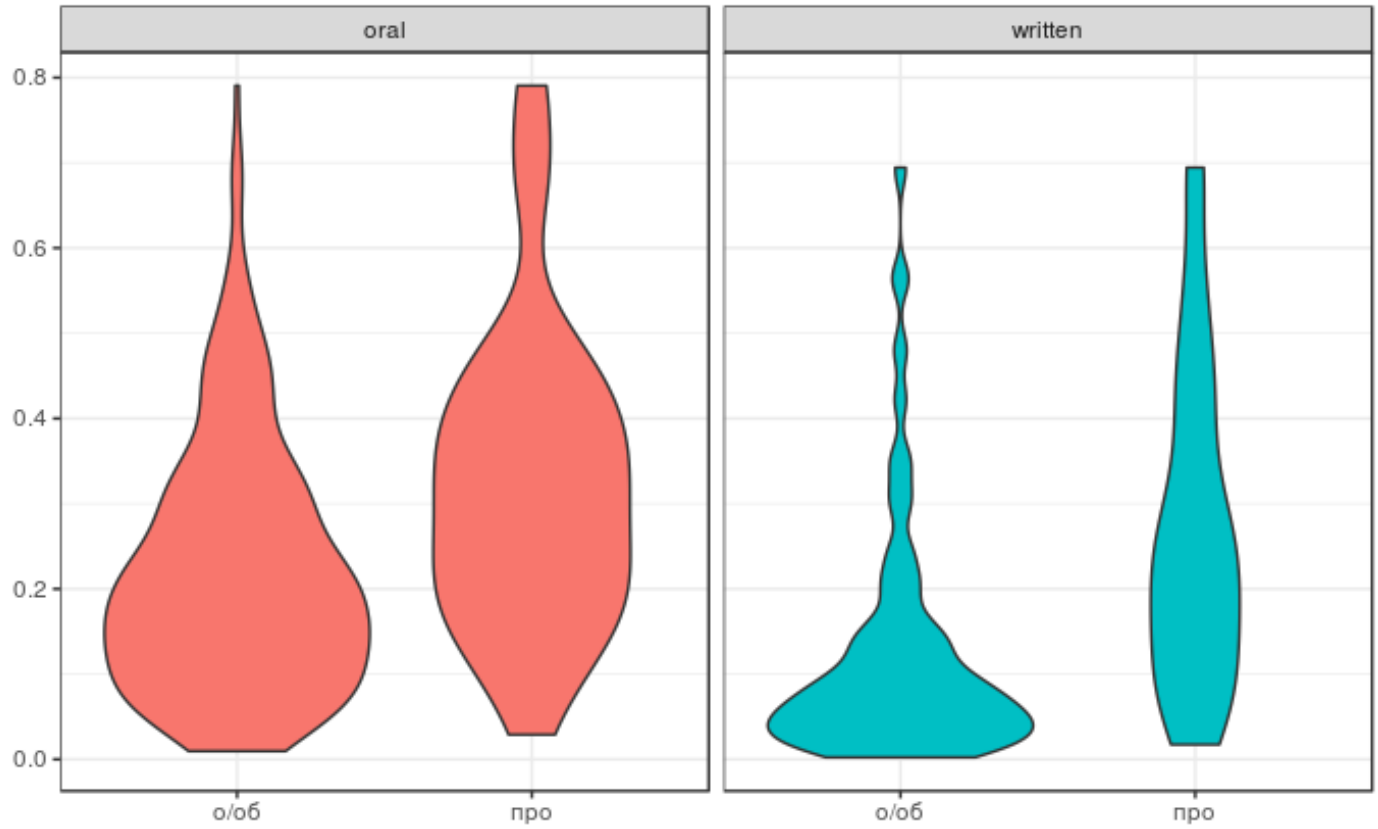
Hide

```
rd %>%
  arrange(p) %>%
  ggplot(aes(prep, p, fill = gnr)) +
  geom_violin() +
  facet_wrap(~gnr) +
  labs(x = "", y = "") +
  theme_bw() +
  guides(fill = FALSE)
```



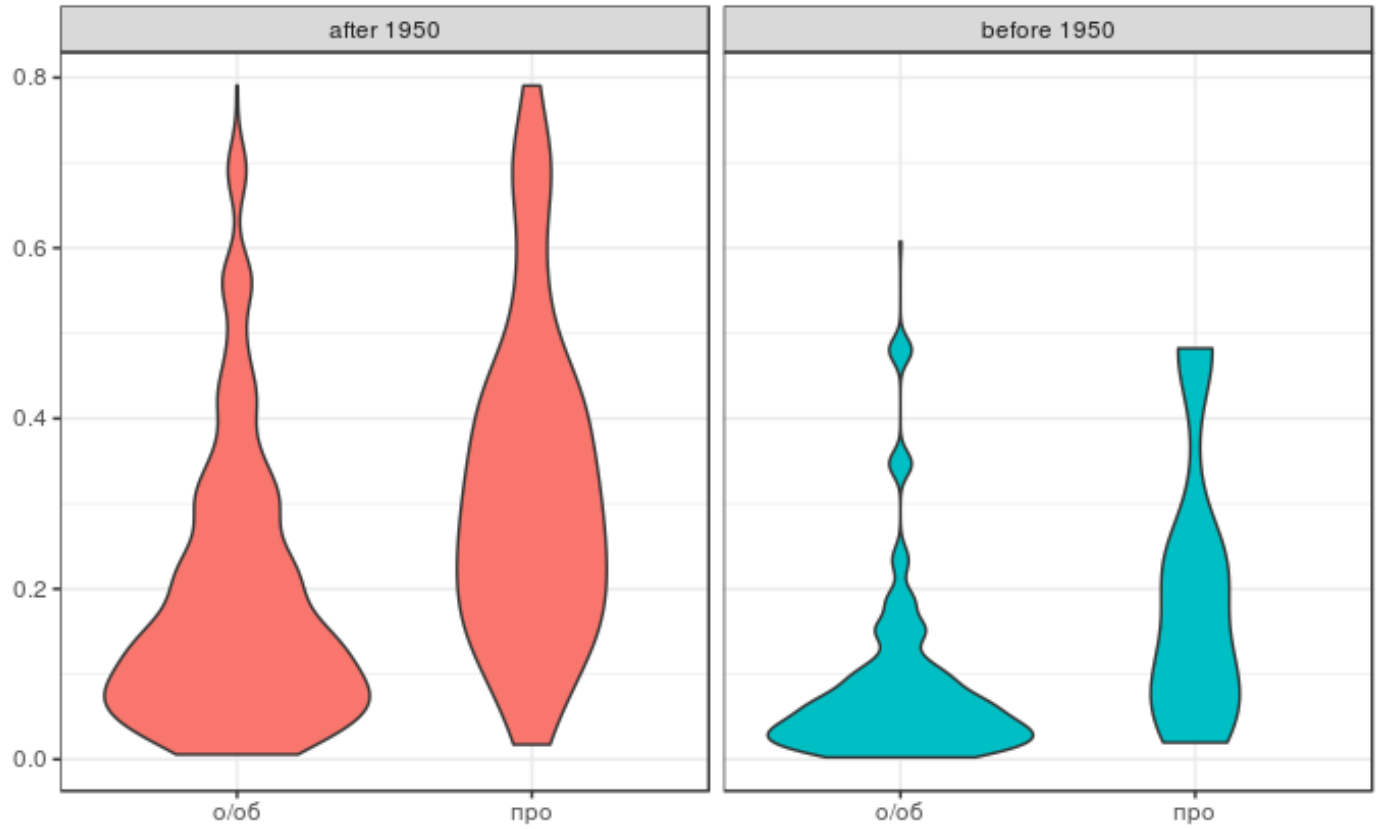
Hide

```
rd %>%
  arrange(p) %>%
  ggplot(aes(prep, p, fill = rgstr)) +
  geom_violin() +
  facet_wrap(~rgstr) +
  labs(x = "", y = "") +
  theme_bw() +
  guides(fill = FALSE)
```



Hide

```
rd %>%
  arrange(p) %>%
  ggplot(aes(prep, p, fill = dt)) +
  geom_violin() +
  facet_wrap(~dt) +
  labs(x = "", y = "") +
  theme_bw() +
  guides(fill = FALSE)
```



Как можно видеть, все параметры, кроме корня вершины, не имеют значений, которые бы хорошо разделяли предлоги “о/об” и “про” - вероятность получить “о/об” выше во всех случаях. Три значения признака root (корень вершины) резко выделяются на общем фоне: вероятность увидеть предлог “про” еще более низка, чем обычно, если корень вершины - “сужд” (рассуждать/рассуждение), и неожиданно (принимая во внимание общие тенденции) высока, если корень вершины - “пе” (петь/песня) или “шут” (шутить/шутка).

Случайные леса (RandomForests)

Случайный лес - ансамбль решающих деревьев (строим много деревьев на случайных подвыборках и усредняем их результаты). Зд. алгоритм применяется для задач классификации (зависимая переменная - категориальная).

Hide

```
set.seed(123)
rf <- randomForest(preposition ~ ., data = trainSet, ntree = 1000, mtry = 3, impor
tance = TRUE, do.trace = 500/10) #ntree - количество построенных деревьев, mtry -
случайно выбираемые при каждом делении предикторы
```

```
ntree      OOB      1      2
   50:  15.73%  3.45% 80.26%
  100:  15.53%  3.67% 77.83%
  150:  15.48%  3.51% 78.32%
  200:  15.37%  3.54% 77.51%
  250:  15.45%  3.57% 77.83%
  300:  15.30%  3.57% 76.86%
  350:  15.42%  3.57% 77.67%
  400:  15.45%  3.57% 77.83%
  450:  15.35%  3.57% 77.18%
  500:  15.35%  3.57% 77.18%
  550:  15.30%  3.57% 76.86%
  600:  15.11%  3.57% 75.73%
  650:  15.09%  3.57% 75.57%
  700:  15.11%  3.57% 75.73%
  750:  15.11%  3.57% 75.73%
  800:  15.17%  3.57% 76.05%
  850:  15.17%  3.57% 76.05%
  900:  15.14%  3.57% 75.89%
  950:  15.14%  3.57% 75.89%
 1000:  15.17%  3.57% 76.05%
```

Hide

```
print(rf)
```

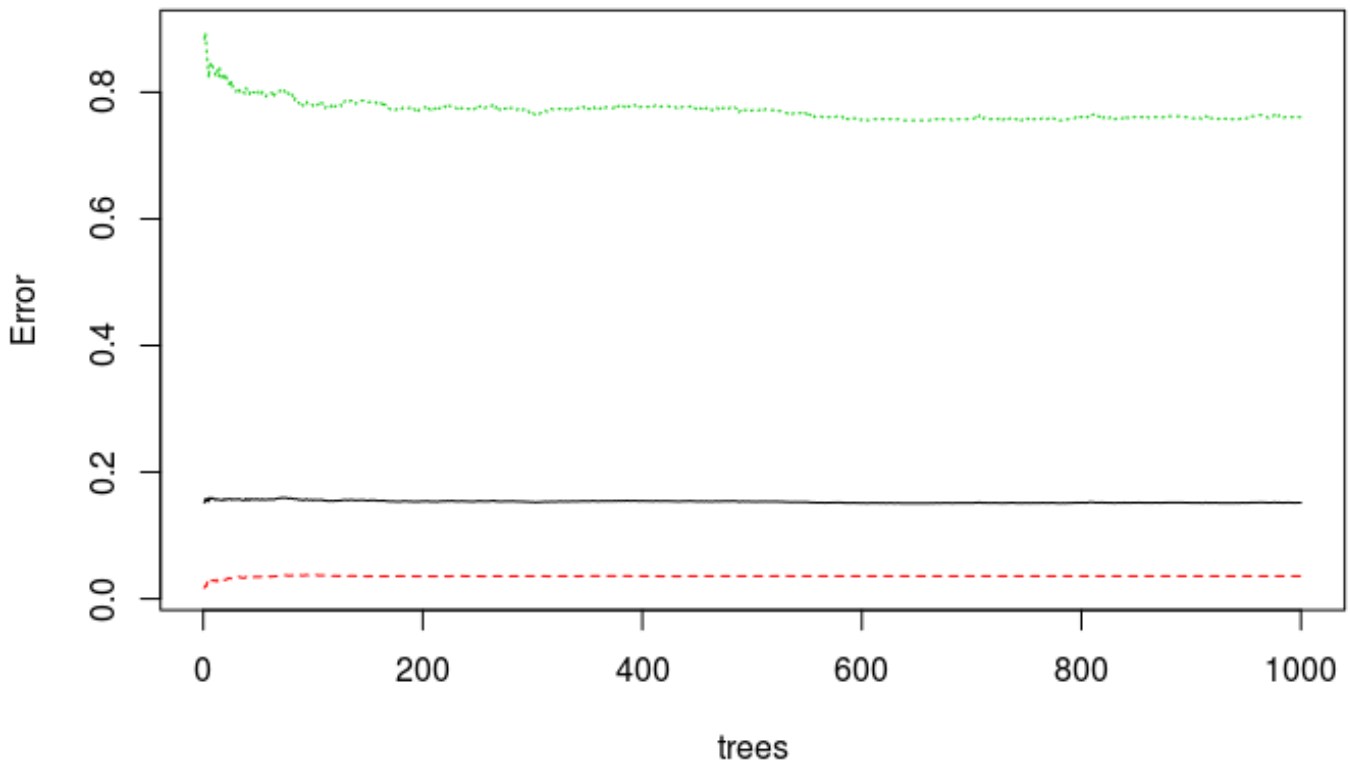
```
Call:
 randomForest(formula = preposition ~ ., data = trainSet, ntree = 1000,      mtry
= 3, importance = TRUE, do.trace = 500/10)
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 3

        OOB estimate of  error rate: 15.17%
Confusion matrix:
     о/об про class.error
о/об 3130 116  0.03573629
про   470 148  0.76051780
```

OOB (out-of-bag error rate) - оценка качества модели на неиспользуемой части выборки (out-of-bag samples).

Hide

```
plot(rf, main = "") #черная кривая на графике - out-of-bag ошибка
```



“Тонкая настройка”

Судя по выдаче функции и графику, при изменении количества деревьев в ансамбле out-of-bag ошибка меняется незначительно; при этом минимальное значение она принимает при ntree = 650. Скорректируем количество деревьев и определим оптимальное число выбираемых при каждом делении предикторов с помощью функции tuneRF():

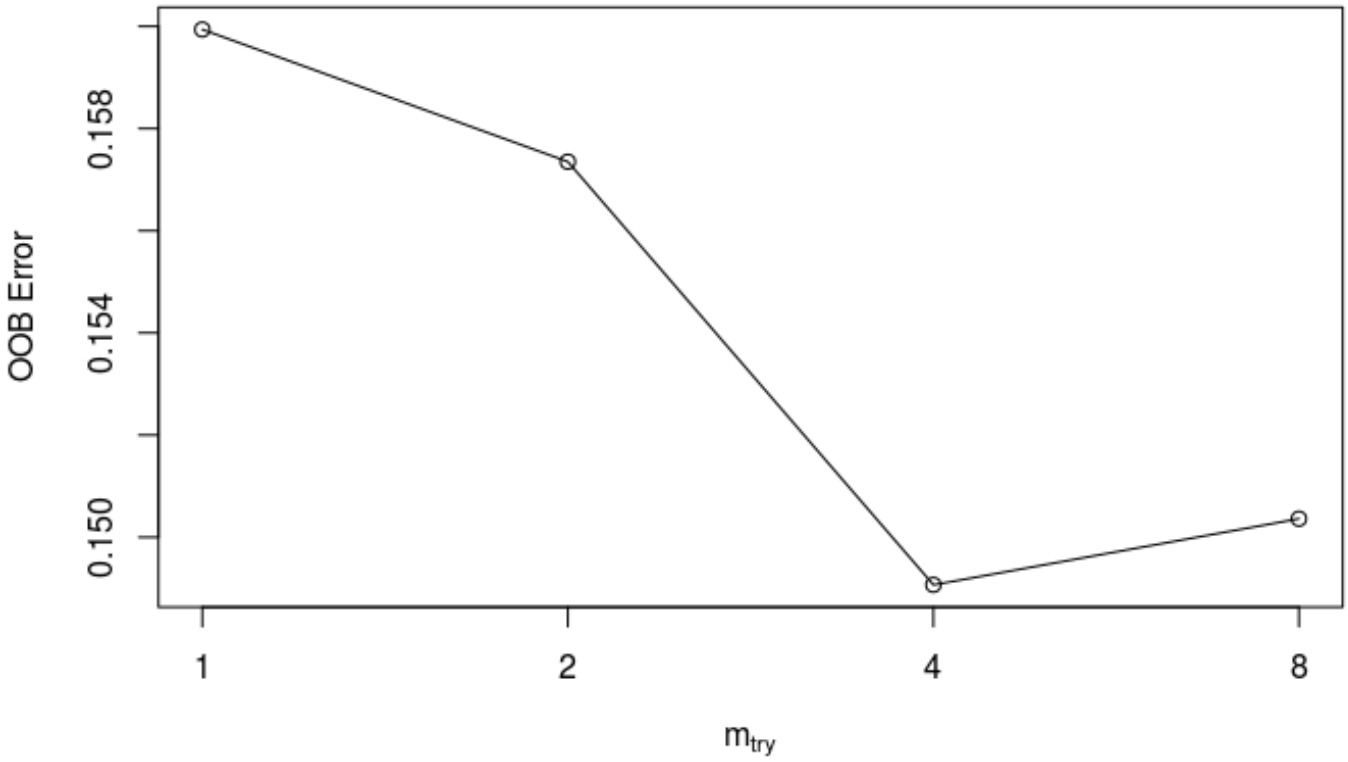
Hide

```
tuneRF(trainSet[,-6], trainSet[,6], stepFactor = 0.5, plot = TRUE, ntreeTry = 650,
trace = TRUE, improve = 0.05)
```

```
mtry = 2  OOB error = 15.73%
Searching left ...
mtry = 4    OOB error = 14.91%
0.05263158 0.05
```

```
invalid mtry: reset to within valid range
```

```
mtry = 8    OOB error = 15.04%
-0.008680556 0.05
Searching right ...
mtry = 1    OOB error = 15.99%
-0.07291667 0.05
      mtry  OOBError
1.OOB    1 0.1599379
2.OOB    2 0.1573499
4.OOB    4 0.1490683
8.OOB    8 0.1503623
```



Наименьшую ошибку прогноза получаем при делении на основе четырех случайно выбранных переменных (mtry = 4).

Hide

```
set.seed(123)
tuned_rf <- randomForest(preposition ~ ., data = trainSet, ntree = 650, mtry = 4,
importance = TRUE, do.trace = 650/10) #обучаем модель с новыми параметрами
```

```
ntree      OOB      1      2
   65:  15.14%  3.45% 76.54%
  130:  15.14%  3.60% 75.73%
  195:  15.14%  3.67% 75.40%
  260:  15.14%  3.67% 75.40%
  325:  15.14%  3.67% 75.40%
  390:  15.19%  3.70% 75.57%
  455:  15.22%  3.70% 75.73%
  520:  15.19%  3.67% 75.73%
  585:  15.14%  3.67% 75.40%
  650:  14.96%  3.57% 74.76%
```

Hide

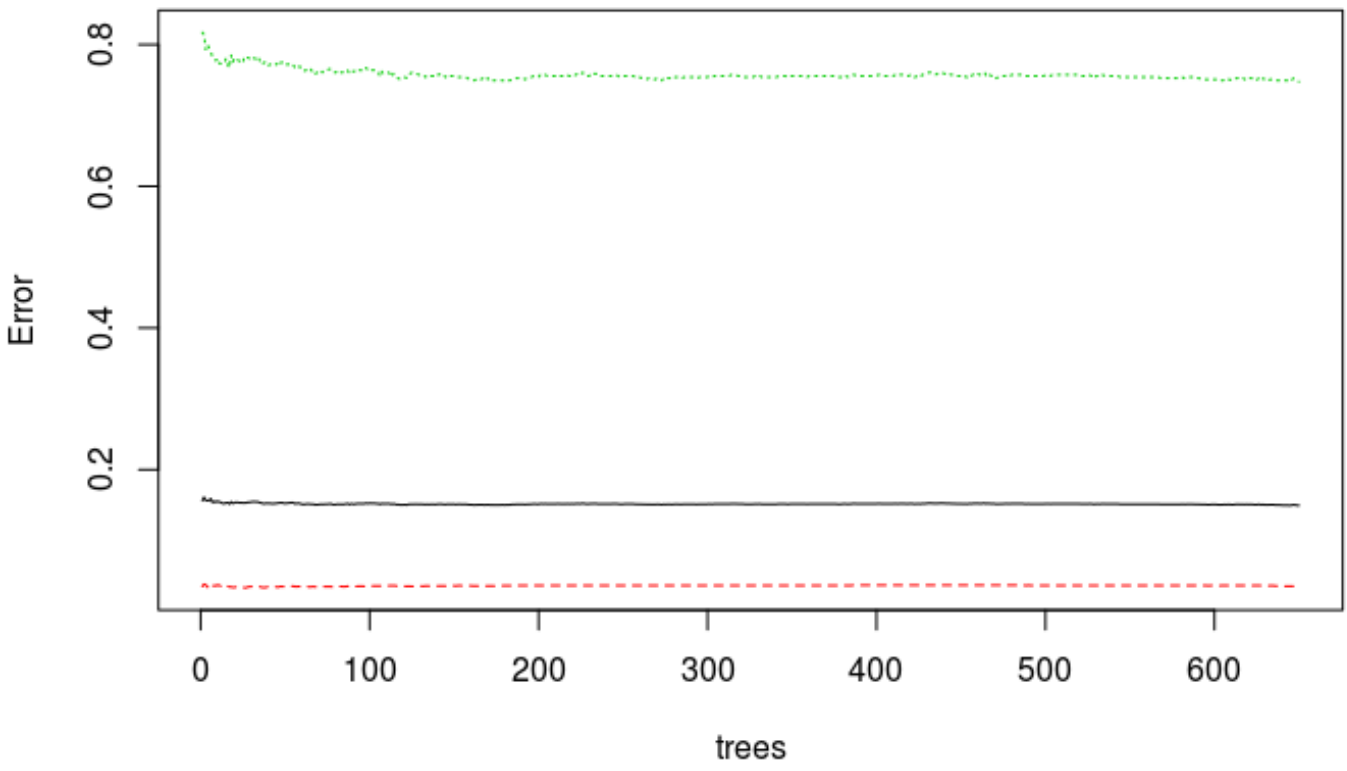
```
print(tuned_rf)
```

```
Call:
 randomForest(formula = preposition ~ ., data = trainSet, ntree = 650,      mtry =
4, importance = TRUE, do.trace = 650/10)
               Type of random forest: classification
                     Number of trees: 650
No. of variables tried at each split: 4

        OOB estimate of  error rate: 14.96%
Confusion matrix:
     о/об про class.error
о/об 3130 116  0.03573629
про   462 156  0.74757282
```

Hide

```
plot(tuned_rf, main = "")
```



Hide

```
tree <- getTree(tuned_rf, 1, labelVar = TRUE) #достаем первое дерево
head(tree)
```

	left daughter <dbl>	right daughter <dbl>	split var <fctr>	split point <dbl>	status <dbl>	prediction <chr>
1	2	3	head_pos	1	1	*NA*
2	4	5	root	507	1	*NA*
3	6	7	root	473	1	*NA*
4	8	9	root	105	1	*NA*
5	10	11	register	1	1	*NA*
6	12	13	genre	1	1	*NA*

6 rows

Hide

```
tail(tree)
```

	left daughter <dbl>	right daughter <dbl>	split var <fctr>	split point <dbl>	status <dbl>	prediction <chr>

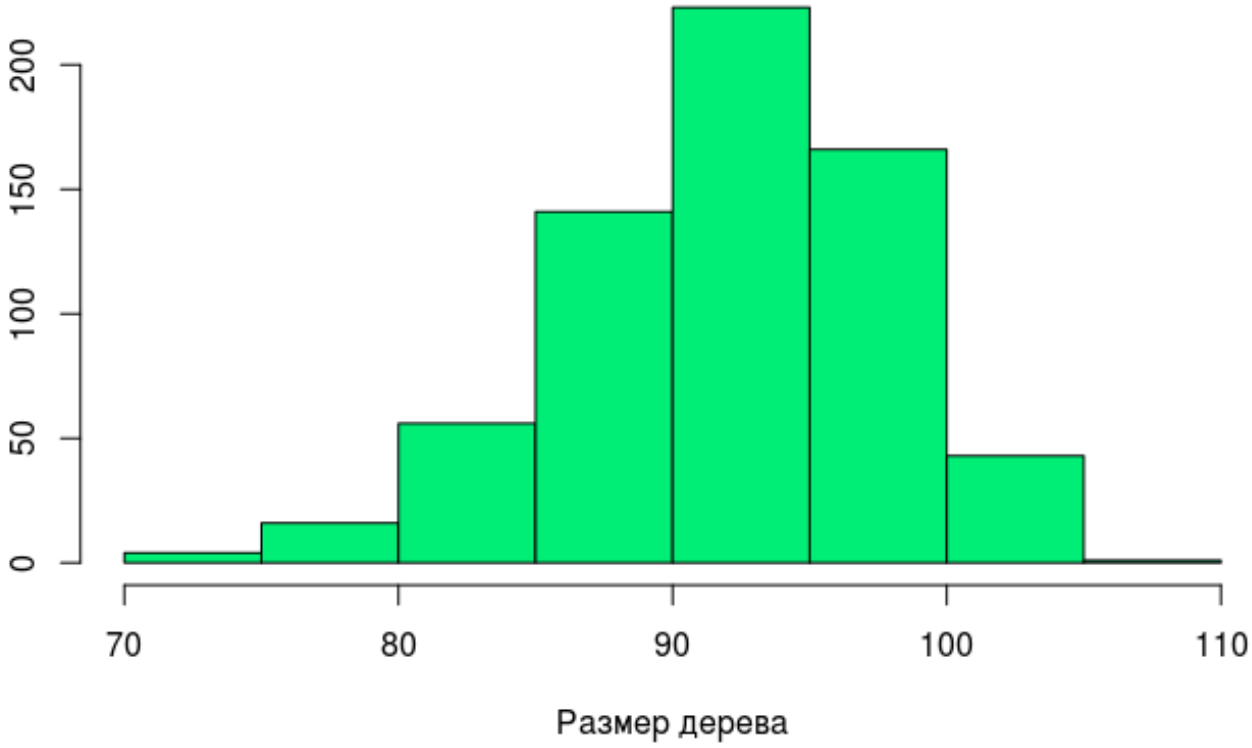
164	0	0	*NA*	0	-1 о/об
165	0	0	*NA*	0	-1 о/об
166	0	0	*NA*	0	-1 о/об
167	0	0	*NA*	0	-1 о/об
168	0	0	*NA*	0	-1 о/об
169	0	0	*NA*	0	-1 о/об

6 rows

split var - разделяющая переменная, split point - значение, с которым она сравнивается, status=1 - нетерминальный узел, left daughter, right daughter - следующие за ним левый и правый узлы, status=-1 - терминальный узел, prediction - прогноз для терминального узла.

Hide

```
hist(treesize(tuned_rf), main = "", xlab = "Размер дерева", ylab = "", col = "spri
nggreen2") #график, показывающий количество узлов у построенных алгоритмом деревье
в; больше всего деревьев (около 225) содержат от 90 до 95 узлов
```



Алгоритм позволяет добиться высокого качества классификации, но совершенно не объясняет, как устроены данные. Понять, что происходит “внутри”, достаточно сложно даже по одному дереву (а всего их несколько сотен). Для описания полученных результатов мы можем оценить, насколько важен тот или иной предиктор:

Hide

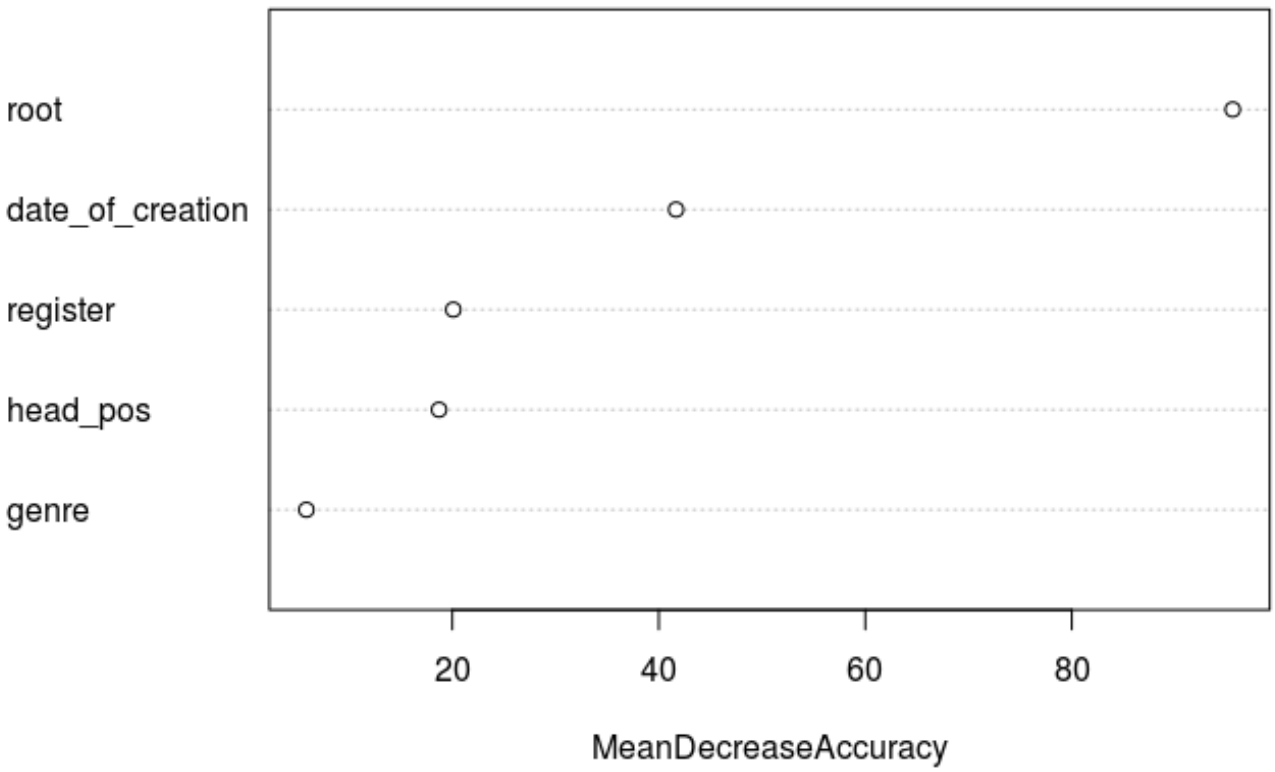
```
importance(tuned_rf, type = 1)
```

```
                 MeanDecreaseAccuracy
genre                         5.87068
register                     20.09619
date_of_creation             41.67389
head_pos                     18.71616
root                         95.58541
```

Самым значимым предиктором оказывается root, “полезность” всех остальных переменных для классификации данных гораздо ниже.

Hide

```
varImpPlot(tuned_rf, type = 1, main = "")
```



Hide

```
rfPredicted <- predict(tuned_rf, testSet)
confusionMatrix(rfPredicted, testSet$preposition) #строим матрицу ошибок на тестов
ых данных
```

```
Confusion Matrix and Statistics

          Reference
Prediction о/об  про
      о/об 1024  160
      про    58   45

               Accuracy : 0.8306
                 95% CI : (0.809, 0.8507)
    No Information Rate : 0.8407
    P-Value [Acc > NIR] : 0.848

                  Kappa : 0.2078
 Mcnemar's Test P-Value : 7.887e-12

            Sensitivity : 0.9464
            Specificity : 0.2195
         Pos Pred Value : 0.8649
         Neg Pred Value : 0.4369
             Prevalence : 0.8407
         Detection Rate : 0.7956
   Detection Prevalence : 0.9200
      Balanced Accuracy : 0.5830

       'Positive' Class : о/об
```

Hide

```
# metrics <- c(Accuracy = (1024 + 45)/nrow(testSet), Precision = 1024/(1024 + 160)
, Recall = 1024/(1024 + 58), Specificity = 45/(45 + 160))
# metrics
```

Модель склонна предсказывать предлог “о/об” в большинстве случаев - поэтому самой частотной ошибкой является предсказание “о/об” при истинном значении “про”. Это происходит потому, что случаев употребления предлога “о/об” в наших данных намного больше - 4328 наблюдений с “о/об” против 823 наблюдений с “про”. Метрики качества при этом оказываются неплохими, но причина этого - не удачный подбор признаков и успешное обучение алгоритма, а высокая частотность одного из классов, к которому модель в результате “склоняется”.

Вывод

Итак, построенные нами модели опровергают нашу нулевую гипотезу сразу по двум, причем противоречащим друг другу причинам. С одной стороны, модель логистической регрессии признает значимыми, помимо регистра и жанра текста, и другие параметры - и корень слова, и его часть речи, и дату возникновения текста. Деревья в случайном лесе также производят деления по всем этим параметрам, самым значимым при этом полагая корень вершины, а наименее значимым - жанр. С другой стороны, абсолютное преобладание предлога “о/об” над предлогом “про” в

текстах приводит к тому, что ни один из выбранных нами признаков не может хорошо разделить случаи употребления “о/об” и “про” - в любых комбинациях “о/об” оказывается частотнее. Единственные значения параметров, при которых вероятность “про” действительно сильно повышается (что “ловится” обоими алгоритмами) - это корни “пе” (петь/песня) и “шут” (шутить/шутка). Поиск других признаков, которые бы хорошо разделили наши предлоги, остается задачей будущих исследований.

Appendix A

Trellis graphs (графики, визуализирующие определенный сабсет данных):

Hide

```
# data %>%
#   count(root, genre, head_pos, date_of_creation, register, preposition) %>%
#   spread(key = preposition, value = n) -> data_summed
```

Hide

```
# barchart(`о/об`+`про` ~ genre | register + root + head_pos + date_of_creation, d
ata_summed, layout = c(5, 2), auto.key = TRUE, xlab = "Жанр текста", ylab = "Число
примеров")
```

Hide

```
# barchart(`о/об`+`про` ~ head_pos | register + genre + root + date_of_creation, d
ata_summed, layout = c(5, 2), auto.key = TRUE, xlab = "Часть речи вершины", ylab =
"Число примеров")
```

Hide

```
# barchart(`о/об`+`про` ~ register | root + genre + head_pos + date_of_creation, d
ata_summed, layout = c(5, 1), aspect = 1.5, auto.key = TRUE, xlab = "Регистр речи"
, ylab = "Число примеров")
```

Hide

```
# barchart(`о/об`+`про` ~ root | register + genre + head_pos + date_of_creation, d
ata_summed, layout = c(1, 2), auto.key = TRUE, xlab = "Корень", ylab = "Число прим
еров")
```

Hide

```
# barchart(`о/об`+`про` ~ date_of_creation | register + root + head_pos + genre, d
ata_summed, layout = c(4, 2), auto.key = TRUE, xlab = "Дата создания текста", ylab
= "Число примеров")
```