Анализ количества выпадаемых осадков по данным поляриметрических радаров с апреля по август 2014

Умрихин Александр

Факультет экономических наук

Группа БЭК-142

Научный руководитель: Демешев Борис Борисович

Департамент прикладной экономики

Москва
2016

Оглавление

Введение

Изучение осадков – актуальное и востребованное занятие. Например, в сельском хозяйстве важно знать точное количество выпавших осадков, чтобы оценивать требуемую величину внесения различных удобрений, улучшать методы обработки, предсказывать объемы урожая и так далее. Для более точной и повсеместной оценки используются поляриметрические радары. Они уже все чаще вводятся в эксплуатацию в США, но в нашей стране еще не получили широкого распространения. Эти радары не измеряют само количество осадков, поэтому оценить его можно только с помощью эконометрического анализа данных, который сопоставляет измерения радаров данным с обычного дождевого датчика и по прецедентам обучает модель, которая позволит прогнозировать осадки точнее, чем это делают сами датчики.

Цель работы – визуализация данных и обучение по прецедентам модели, которая спрогнозирует количество осадков по данным, снятых с радаров.

Задачи построены последовательно.

1. Изучение данных: просмотр содержимого таблицы, описание признаков и прогнозируемой переменной, вывод основных характеристик.

2. Предобработка: приведение данных в тот вид, с которым можно будет работать. Исключение пробелов, изучение выбросов.

3. Визуализация: построение различных графиков для понимания распределения признаков и их зависимостей друг от друга.

4. Построение моделей. Завершающий этап, который использует информацию, собранную в предыдущих пунктах.

Методология

Работа будет проведена в среде программирования RStudio, которая интерпретирует язык R. Он использует широкий спектр статистических, численных и других методов и обладает легкой расширяемостью с помощью дополнительных

пакетов, которые могут разрабатываться самими пользователями. В моей работе используются следующие пакеты:

```
#Пакеты для графиков
library(corrplot)
library(ggplot2)
library(grid)
library(Rmisc)
library(hexbin)
library(GGally)
library(rattle)
library(rpart.plot)
library(RColorBrewer)
#Пакет для более быстрой работы с данными
library(data.table)
#Пакет для построения регрессий
library(h2o)

#Пакет для построения деревьев

library(rpart)
```

Работа состоит из двух частей: визуализация данных и сам анализ. На сегодняшний день существует множество инструментов для анализа данных и визуализации результатов, некоторые из них позволяют применять довольно широкий спектр статистических методов, не имея никакого опыта программирования (например, SPSS). Также весьма распространен для анализа данных язык программирования Python. Преимущество R – его простота в освоении и возможность понимать на интуитивном уровне нужные действия. Также визуализация в R, благодаря огромному спектру дополняющих язык пакетов, является даже более успешной, чем у специализированных программ с графическим интерфейсом. Основным пакетом для визуализации выбран **ggplot2**, но для некоторых специфических графиков используются другие пакеты из списка, приведенного выше.

Будут построены различные графики, отображающие распределения величин и их попарные зависимости, корреляции. Визуализация позволяет увидеть, как именно ведут себя признаки, что помогает в самом построении модели. Еще одна часть визуализации – изображение непосредственно результатов анализа, то есть весов признаков для разных моделей, ошибок и решающего дерева.

Изучение данных

Данные взяты с сайта kaggle.com. Этот сайт является платформой для людей, интересующихся анализом данных разного уровня. На сайте хранится большое количество весьма актуальных наборов данных в открытом доступе, а также проводятся соревнования по построению наилучшей прогностической модели. Мои данные взяты из такого соревнования с названием «How Much Did It Rain? II». Они в формате CSV (Comma-Separated Values — значения, разделённые запятыми) — текстовый формат, предназначенный для представления табличных данных. Набор данных состоит из 23х признаков и более 13 миллионов объектов. Объекты – измерения радара (делаются в определенный момент времени).

Описание признаков:

1. Id: Уникальный номер набора наблюдений за час при одном датчике.
2. minutes_past: Количество минут, прошедших с начала часа до снятия наблюдений с радара.
3. radardist_km: расстояние от датчика до радара.
4. Ref: отражательная способность чуть выше датчика в dBZ (decibel relative to Z)
5. Ref_5x5_10th: 10й перцентиль значений отражательной способности в 25 пикселях, находящихся в квадрате 5 на 5, окружающем датчик.
6. Ref_5x5_50th: 50й перцентиль.
7. Ref_5x5_90th: 90й перцентиль.
8. RefComposite: Максимальное значение отражательной способности в вертикальной колонке над датчиком.
9. RefComposite_5x5_10th
10. RefComposite_5x5_50th
11. RefComposite_5x5_90th
12. RhoHV: коэффициент корреляции между вертикальными и горизонтальными измерениями радара. Т.е. реагирует на форму встречающихся препятствий.
13. RhoHV_5x5_10th
14. RhoHV_5x5_50th
15. RhoHV_5x5_90th
16. Zdr: разница в возвращенной энергии между горизонтальными и вертикальными волнами. Чем больше 0, тем сильнее препятствия вытянуты горизонтально.
17. Zdr_5x5_10th

18. Zdr_5x5_50th
19. Zdr_5x5_90th
20. Kdp: схожий с предыдущим показатель, но зависящий от концентрации частиц. Т.е. чем больше горизонтально вытянутых частиц, тем больше данный показатель.
21. Kdp_5x5_10th
22. Kdp_5x5_50th
23. Kdp_5x5_90th
24. Expected: Реальные измерения датчика, по количеству осадков за час.

Загрузим данные и посмотрим саммари.

```
dt <- data.table::fread("train.csv", sep=',', header=TRUE)
options(digits = 10)
summary(dt)
```

```
##        Id           minutes_past       radardist_km           Ref
##  Min.   :      1   Min.   : 0.00000   Min.   : 0.00000   Min.   :-31.000
##  1st Qu.: 296897   1st Qu.:15.00000   1st Qu.: 9.00000   1st Qu.: 16.000
##  Median : 592199   Median :30.00000   Median :11.00000   Median : 22.500
##  Mean   : 592337   Mean   :29.52373   Mean   :11.06794   Mean   : 22.927
##  3rd Qu.: 889582   3rd Qu.:44.00000   3rd Qu.:14.00000   3rd Qu.: 29.500
##  Max.   :1180945   Max.   :59.00000   Max.   :21.00000   Max.   : 71.000
##                                                          NA's   :7415826
##   Ref_5x5_10th      Ref_5x5_50th      Ref_5x5_90th      RefComposite
##  Min.   :-32.000   Min.   :-32.00    Min.   :-28.500   Min.   :-32.000
##  1st Qu.: 14.000   1st Qu.: 16.00    1st Qu.: 18.000   1st Qu.: 17.500
##  Median : 20.000   Median : 22.50    Median : 25.500   Median : 24.000
##  Mean   : 19.952   Mean   : 22.61    Mean   : 25.898   Mean   : 24.711
##  3rd Qu.: 26.000   3rd Qu.: 29.00    3rd Qu.: 33.500   3rd Qu.: 31.500
##  Max.   : 62.500   Max.   : 69.00    Max.   : 72.500   Max.   : 92.500
##  NA's   :8481213   NA's   :7408719   NA's   :6213920   NA's   :7048858
##  RefComposite_5x5_10th RefComposite_5x5_50th RefComposite_5x5_90th
##  Min.   :-31.000       Min.   :-27.500       Min.   :-25.000
##  1st Qu.: 16.000       1st Qu.: 17.500       1st Qu.: 19.500
##  Median : 22.000       Median : 24.000       Median : 27.000
##  Mean   : 22.158       Mean   : 24.421       Mean   : 27.369
##  3rd Qu.: 28.500       3rd Qu.: 31.500       3rd Qu.: 35.000
##  Max.   : 66.000       Max.   : 71.000       Max.   : 93.500
##  NA's   :8009528       NA's   :7053538       NA's   :5935998
##      RhoHV         RhoHV_5x5_10th    RhoHV_5x5_50th    RhoHV_5x5_90th
##  Min.   :0.208     Min.   :0.208     Min.   :0.208     Min.   :0.208
##  1st Qu.:0.972     1st Qu.:0.915     1st Qu.:0.975     1st Qu.:0.998
##  Median :0.992     Median :0.958     Median :0.992     Median :1.012
##  Mean   :0.973     Mean   :0.919     Mean   :0.974     Mean   :1.015
##  3rd Qu.:1.002     3rd Qu.:0.982     3rd Qu.:0.998     3rd Qu.:1.052
##  Max.   :1.052     Max.   :1.052     Max.   :1.052     Max.   :1.052
##  NA's   :8830285   NA's   :9632047   NA's   :8828633   NA's   :7859617
##       Zdr           Zdr_5x5_10th      Zdr_5x5_50th      Zdr_5x5_90th
##  Min.   :-7.875    Min.   :-7.875    Min.   :-7.875    Min.   :-7.875
##  1st Qu.:-0.188    1st Qu.:-1.125    1st Qu.:-0.062    1st Qu.: 1.062
##  Median : 0.375    Median :-0.625    Median : 0.250    Median : 1.688
```

```
##  Mean   : 0.537    Mean   :-0.719    Mean   : 0.338    Mean   : 2.073
##  3rd Qu.: 1.062    3rd Qu.:-0.188    3rd Qu.: 0.688    3rd Qu.: 2.625
##  Max.   : 7.938    Max.   : 7.938    Max.   : 7.938    Max.   : 7.938
##  NA's   :8830285   NA's   :9632047   NA's   :8828633   NA's   :7859617
##       Kdp           Kdp_5x5_10th       Kdp_5x5_50th      Kdp_5x5_90th
##  Min.   :-96.040   Min.   :-80.790    Min.   :-78.770   Min.   :-100.20
##  1st Qu.: -1.410   1st Qu.: -4.580    1st Qu.: -0.710   1st Qu.:   2.07
##  Median :  0.000   Median : -2.820    Median :  0.000   Median :   3.52
##  Mean   :  0.035   Mean   : -3.482    Mean   : -0.474   Mean   :   4.08
##  3rd Qu.:  1.750   3rd Qu.: -1.760    3rd Qu.:  0.350   3rd Qu.:   5.64
##  Max.   :179.750   Max.   :  3.520    Max.   : 12.800   Max.   : 144.60
##  NA's   :9582566   NA's   :10336419   NA's   :9577920   NA's   :8712425
##     Expected
##  Min.   :    0.0100
##  1st Qu.:    0.2540
##  Median :    1.0160
##  Mean   :  108.6263
##  3rd Qu.:    3.8100
##  Max.   :33017.7300
##
```

```
dt_na <- na.omit(dt)
fixed_dt <- dt_na
```

Предобработка

Можно заметить, что у многих переменных очень сильные выбросы, в том числе невозможные. Например, отражательная способность (Ref и другие) не может быть меньше 0, корреляция(RhoHV) больше 1. Так же есть большие выбросы по количеству осадков, измеренного осадкомерами. Это может быть связано с их засорением или другими причинами. Чтобы устранить статистическую погрешность нереальные данные будут удалены.

Еще один примечательный факт – большое количество пропущенных значений (для измерений радара варьируется примерно от 7 до 10 миллионов при 13 миллионах строк всего). Поэтому принято решение устранить все строки, где есть пропущенные значения, чтобы не искажать модель.

Также я решил добавить еще одну переменную:

$$\text{logExpected} = \ln(1 + \text{Expected})$$

```
fixed_dt$Ref_5x5_50th[which(fixed_dt$Ref_5x5_50th < 0)] <- NA
fixed_dt$Ref_5x5_90th[which(fixed_dt$Ref_5x5_90th < 0)] <- NA
fixed_dt$RefComposite[which(fixed_dt$RefComposite < 0)] <- NA
fixed_dt$RefComposite_5x5_50th[which(fixed_dt$RefComposite_5x5_50th_5x5_50th
< 0)] <- NA
fixed_dt$RefComposite_5x5_90th[which(fixed_dt$RefComposite_5x5_50th_5x5_90th
< 0)] <- NA
fixed_dt$Ref[which(fixed_dt$Ref < 0)] <- NA
```

```
fixed_dt$Ref[which(fixed_dt$RhoHV > 1)] <- NA
fixed_dt$Expected[which(fixed_dt$Expected >= 70)] <- NA
fixed_dt$logExpected <- log1p(fixed_dt$Expected)
```

Визуализация

Корреляции

Первым является вывод графика корреляций. Для него я пользуюсь специализированным пакетом «corrplot»

```
mcor <- cor(fixed_dt, use="complete")
corrplot(mcor, type="upper", tl.col="black", tl.srt=45)
```

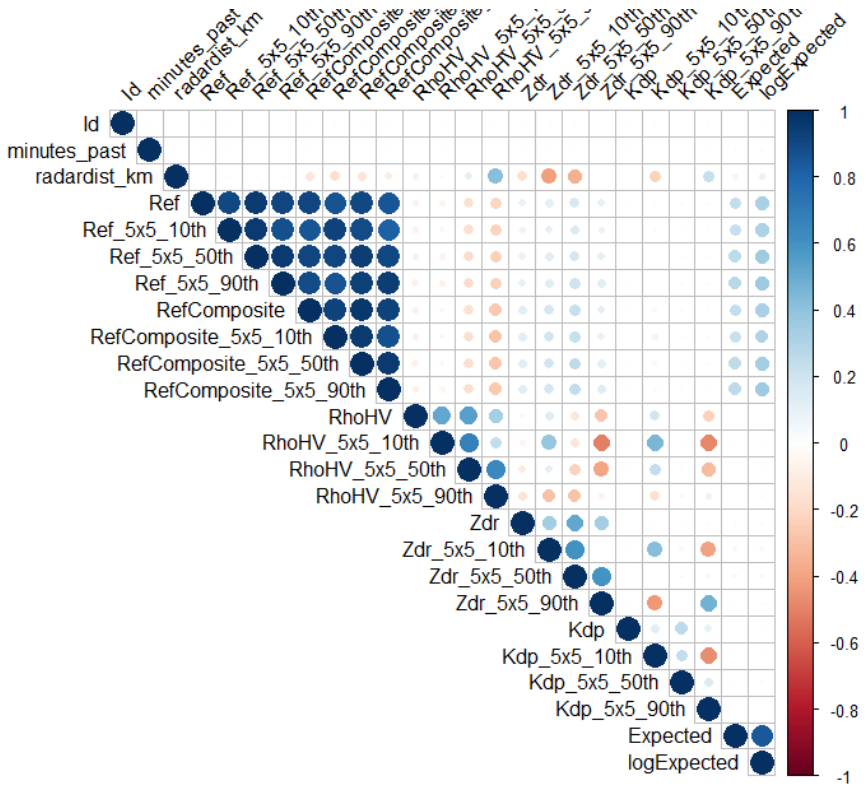


Рисунок 1. Таблица корреляций.

*чем больше круг, тем выше корреляция

синяя – положительная, красная – отрицательная

Заметно, что все признаки, связанные с отражательной способностью, сильно коррелируют друг с другом. Эти же признаки являются единственными, имеющими заметную корреляцию с прогнозируемой переменной.

Распределения признаков

Следующий этап – отдельная визуализация некоторых признаков. И сравнение их с распространенными распределениями.

Есть несколько признаков с распределением, похожим на нормальное.

1. Kdp

```
g_kdp <- ggplot(fixed_dt, aes(x = Kdp, colour = "Kdp"))+
  geom_density(size = 1, adjust = 3)
g_kdp + stat_function(fun = dnorm, aes(colour = "stand norm"), size = 1, args
 = list(mean = mean(fixed_dt$Kdp, na.rm = TRUE), sd = sd(fixed_dt$Kdp, na.rm
= TRUE)))+
  scale_colour_manual("Line", values = c("red", "blue"), breaks=c("stand norm
", "Kdp"))+
  xlim(-40, 40)+
  ggtitle("Kdp and normal dist")+
  theme(text = element_text(size=20))+
  geom_hline(yintercept=0, colour="white", size=0.8)
```

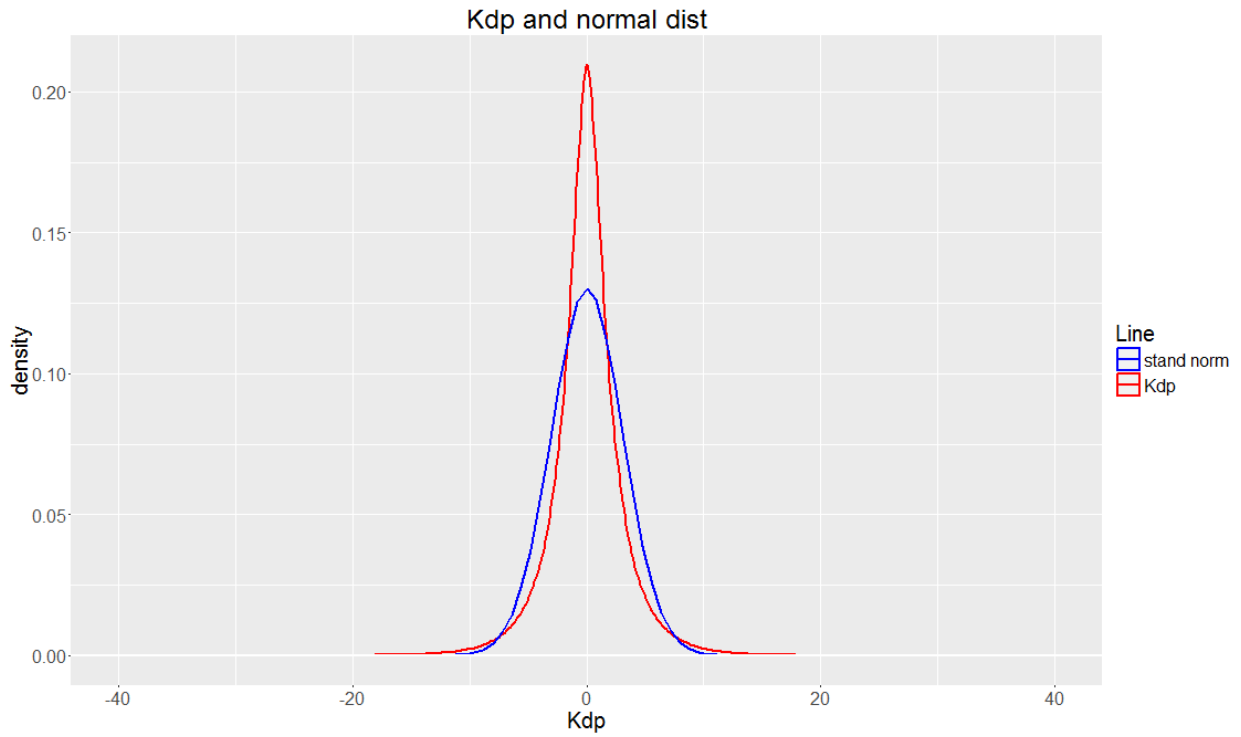


Рисунок 2. Сравнение распределения признака Kdp и нормального.

2. Ref

```
g_ref <- ggplot(fixed_dt, aes(x = Ref, colour = "Ref"))+
  geom_density(size = 1, adjust = 2)+
  geom_hline(yintercept=0, colour="white", size=1)
g_ref + stat_function(fun = dnorm, aes(colour = "stand norm"), size = 1, args
 = list(mean = mean(fixed_dt$Ref, na.rm = TRUE), sd = sd(fixed_dt$Ref, na.rm
= TRUE)))+
```

```
  scale_colour_manual("Line", values = c("red", "blue"), breaks=c("stand norm
", "Ref"))+
  ggtitle("Ref and normal dist")+
  theme(text = element_text(size=20))
```

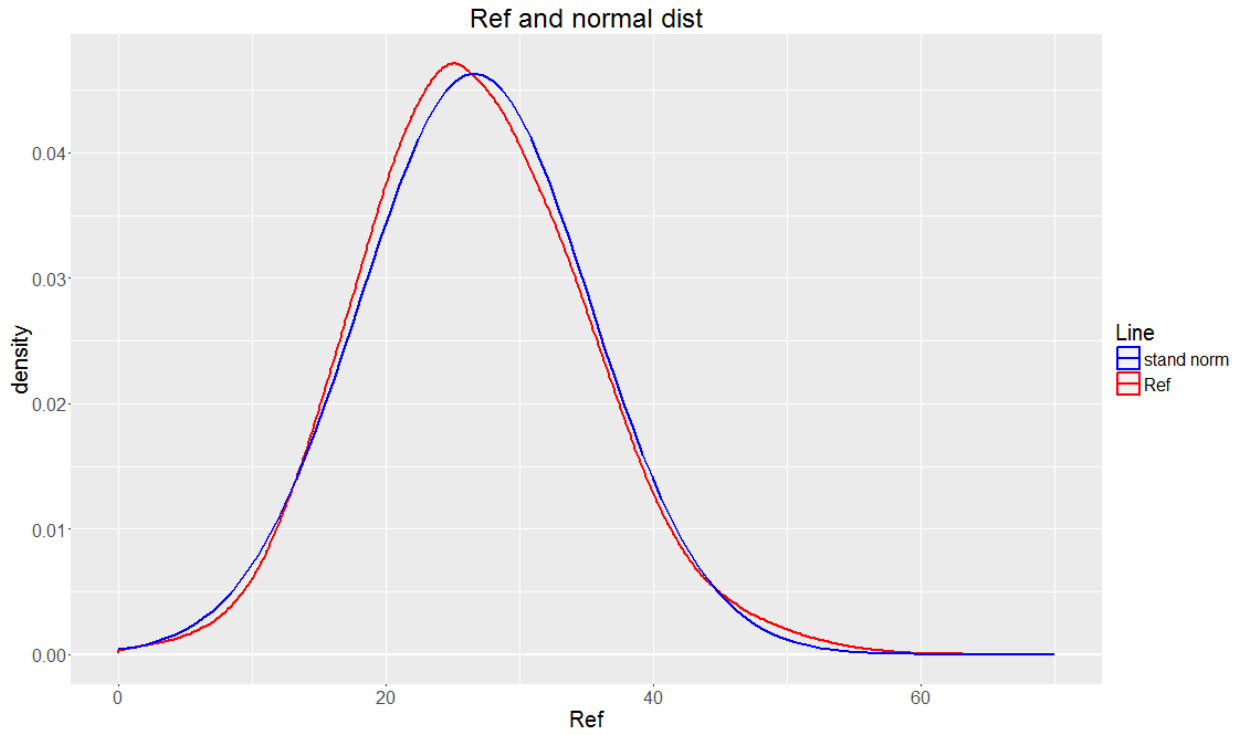


Рисунок 3. Сравнение распределения признака Ref и нормального.

3. Zdr

```
g_zdr <- ggplot(fixed_dt, aes(x = Zdr, colour = "Zdr"))+
  geom_density(size = 1, adjust = 3)
g_zdr + stat_function(fun = dnorm, aes(colour = "stand norm"), size =
1, args = list(mean = mean(fixed_dt$Zdr, na.rm = TRUE), sd = sd(fixed_d
t$Zdr, na.rm = TRUE)))+
  scale_colour_manual("Line", values = c("red", "blue"), breaks=c("stan
d norm", "Zdr"))+
  xlim(-20,20)+
  ggtitle("Zdr and normal dist")+
  theme(text = element_text(size=20))+
  geom_hline(yintercept=0, colour="white", size=0.8)
```

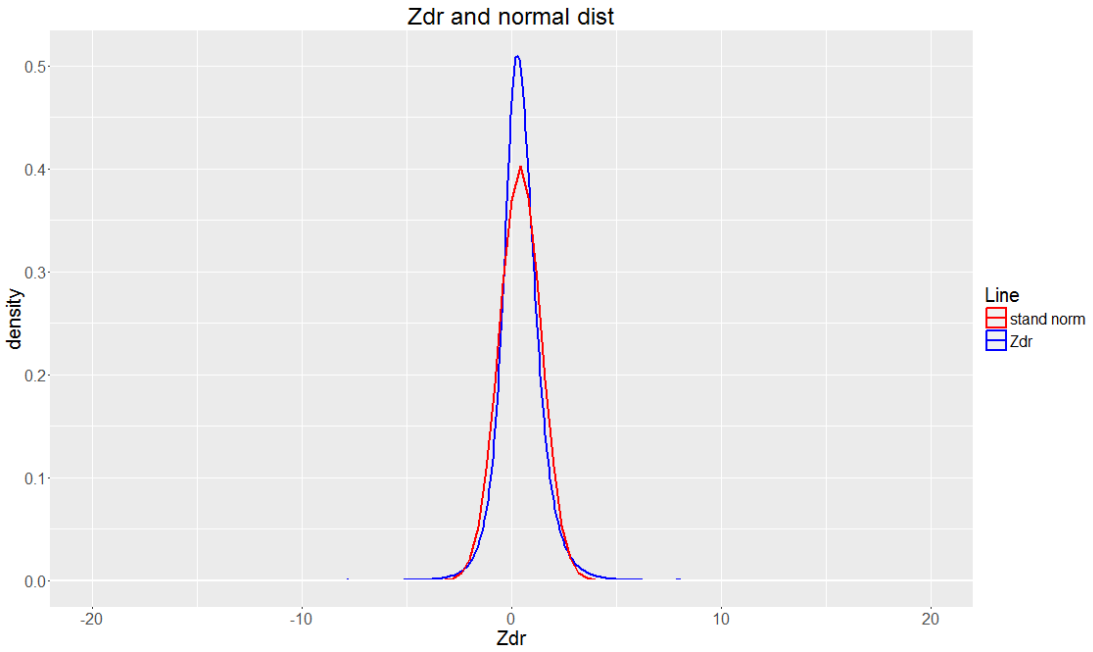


Рисунок 4. Сравнение распределения признака Zdr и нормального.

4. RefComposite

```
g_rc <- ggplot(fixed_dt, aes(x = RefComposite, colour = "RefComp"))+
  geom_density(size = 1, adjust = 3)
g_rc + stat_function(fun = dnorm, aes(colour = "stand norm"), size = 1,
 args = list(mean = mean(fixed_dt$RefComposite, na.rm = TRUE), sd = sd
(fixed_dt$RefComposite, na.rm = TRUE)))+
  scale_colour_manual("Line", values = c("red", "blue"), breaks=c("stan
d norm", "RefComp"))+
  ggtitle("RefComposite and normal dist")+
  theme(text = element_text(size=20))+
  geom_hline(yintercept=0, colour="white", size=0.8)
```

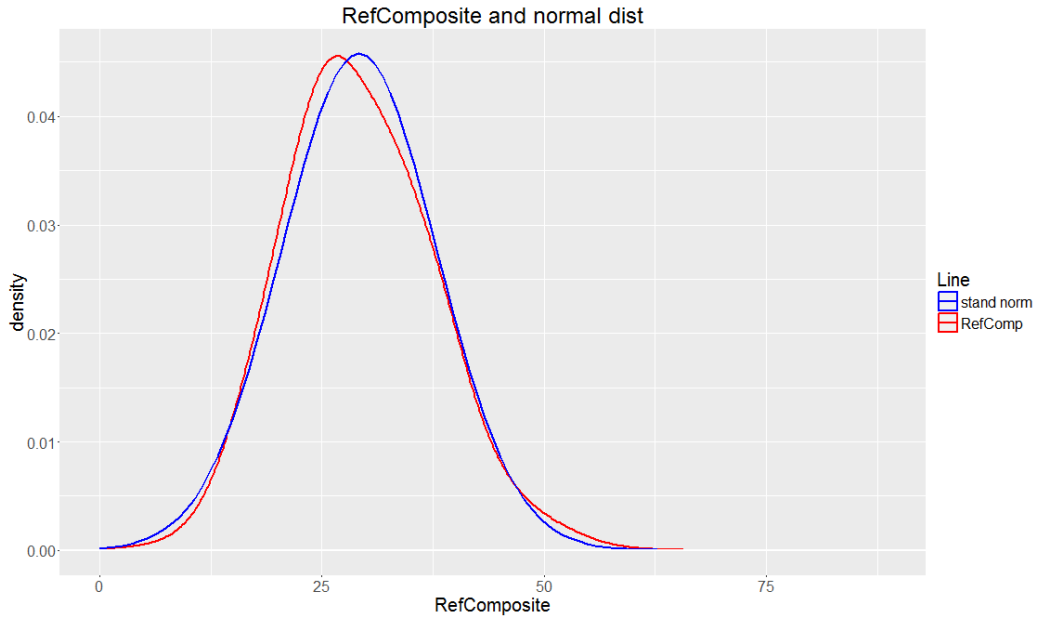


Рисунок 5. Сравнение распределения признака RefComposite и нормального.

Видно, что Ref и RefComposite распределены почти нормально и смещены почти одинаково. Kdp и Zdr имеют более вытянутую форму.

Далее выведем распределения нескольких признаков.

```
ggplot(fixed_dt, aes(x = RhoHV, colour = "RhoHV"))+
  geom_density(size = 1, adjust = 5)+
  geom_vline(xintercept=1, colour="black", size=1)+
  xlim(0.5,1.1)+
  ggtitle("RhoHV")+
  theme(text = element_text(size=20))
```

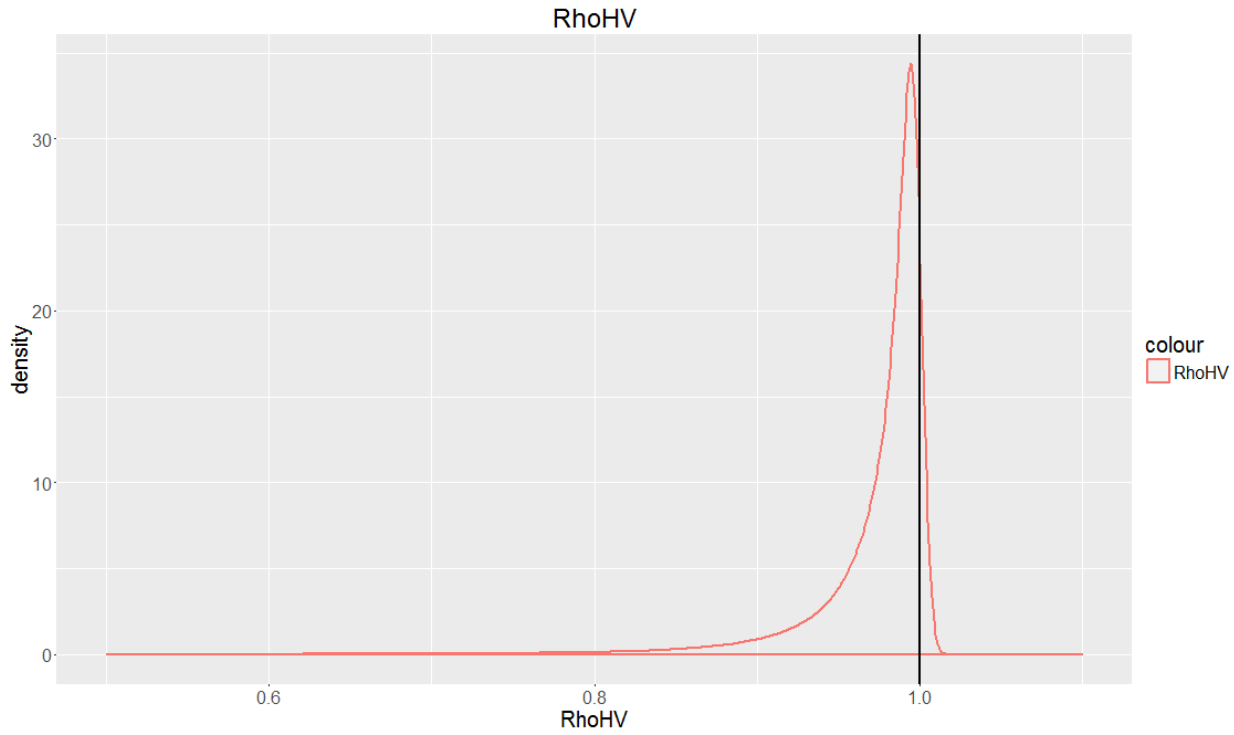


Рисунок 6. Распределение признака RhoHV

Почти все значения скопились у 1.

Дальше посмотрим на распределение признаков, связанных с отражаемостью, выведем их рядом и сравним.

```
g1 <-  ggplot(fixed_dt, aes(x = Ref)) + geom_density()
g2 <-  ggplot(fixed_dt, aes(x = Ref_5x5_10th)) + geom_density()
g3 <-  ggplot(fixed_dt, aes(x = Ref_5x5_50th)) + geom_density()
g4 <-  ggplot(fixed_dt, aes(x = Ref_5x5_90th)) + geom_density()
g5 <-  ggplot(fixed_dt, aes(x = RefComposite)) + geom_density()
g6 <-  ggplot(fixed_dt, aes(x = RefComposite_5x5_10th)) + geom_density()
g7 <-  ggplot(fixed_dt, aes(x = RefComposite_5x5_50th)) + geom_density()
g8 <-  ggplot(fixed_dt, aes(x = RefComposite_5x5_90th)) + geom_density()
multiplot(g1, g2, g3, g4, g5, g6, g7, g8, cols=4)
```

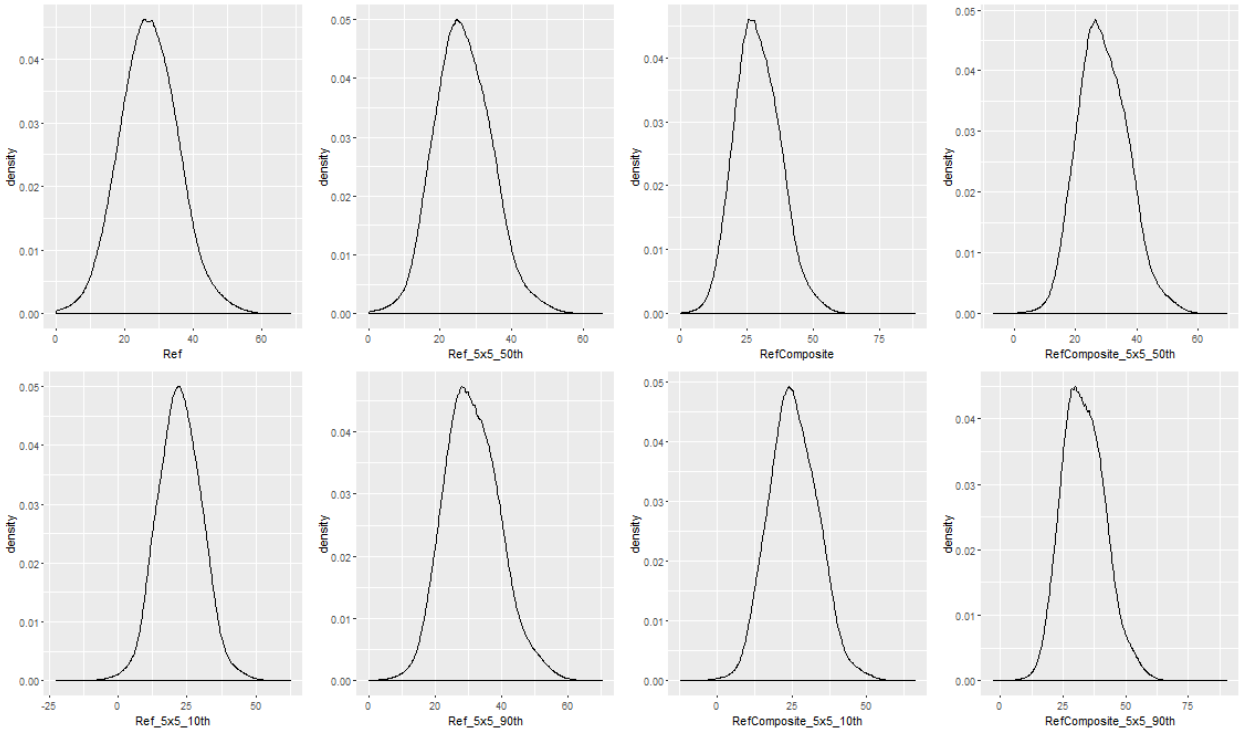


Рисунок 7. распределение признаков, связанных с отражаемостью.

Заметно, что у этих признаков близки средние значения, и у некоторых схожая форма, со смещенной влево «головой».

Теперь выведем распределения прогнозируемой переменной и логарифма от нее.

```
ggplot(fixed_dt, aes(x = Expected, colour = "Expected"))+
  geom_density(size = 1, adjust = 3)+
  ggtitle("Expected dist")+
  theme(text = element_text(size=20))
```

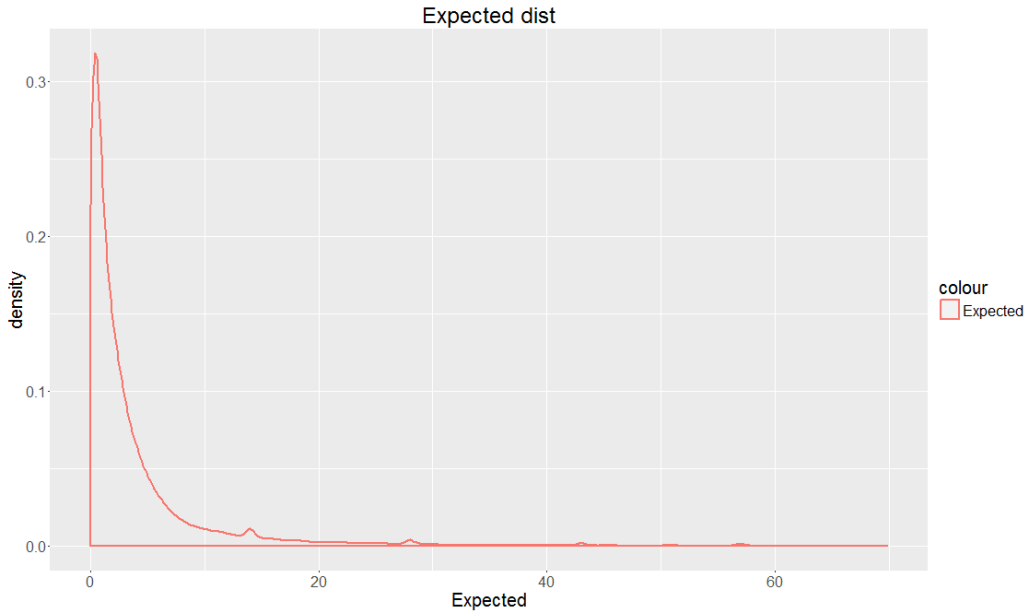


Рисунок 8 Распределение Expected.

Примечательно, что в некоторых значениях находятся «бугры», то есть по какой-то причине датчик выдает определенные значения чаще.

```
ggplot(fixed_dt, aes(x = logExpected, colour = "logExpected"))+
  geom_density(size = 1, adjust = 3)+
  ggtitle("logExpected dist")+
  theme(text = element_text(size=20))
```

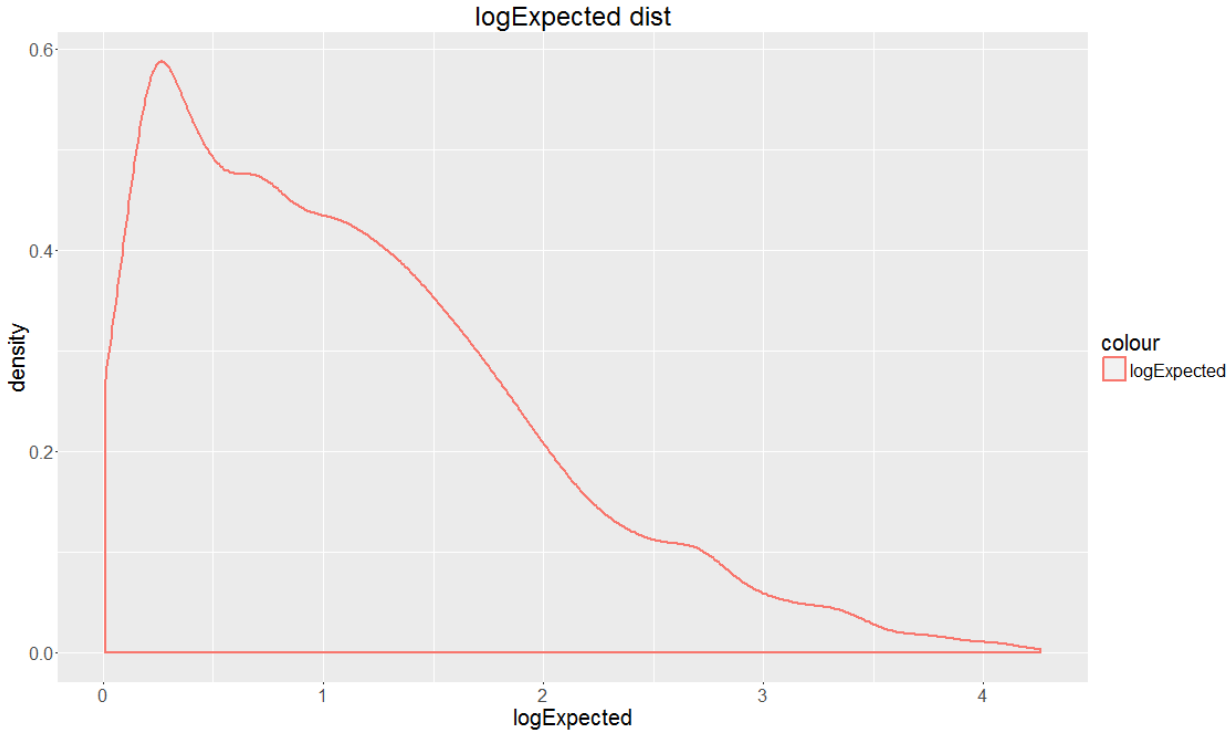


Рисунок 9 Распределение logExpected

Попарные зависимости

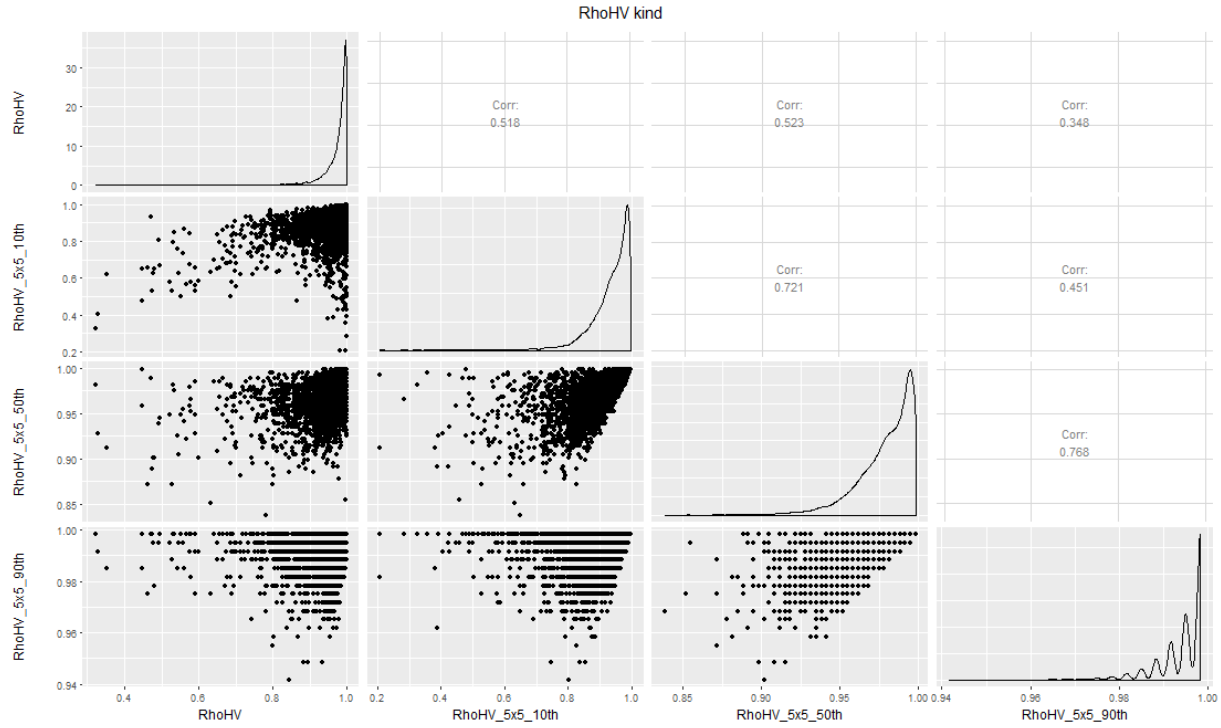
Для следующих графиков создадим случайную подвыборку из 10000 объектов, чтобы не перегружать их.

```
sample <- fixed_dt[sample(nrow(fixed_dt), 10000), ]
```

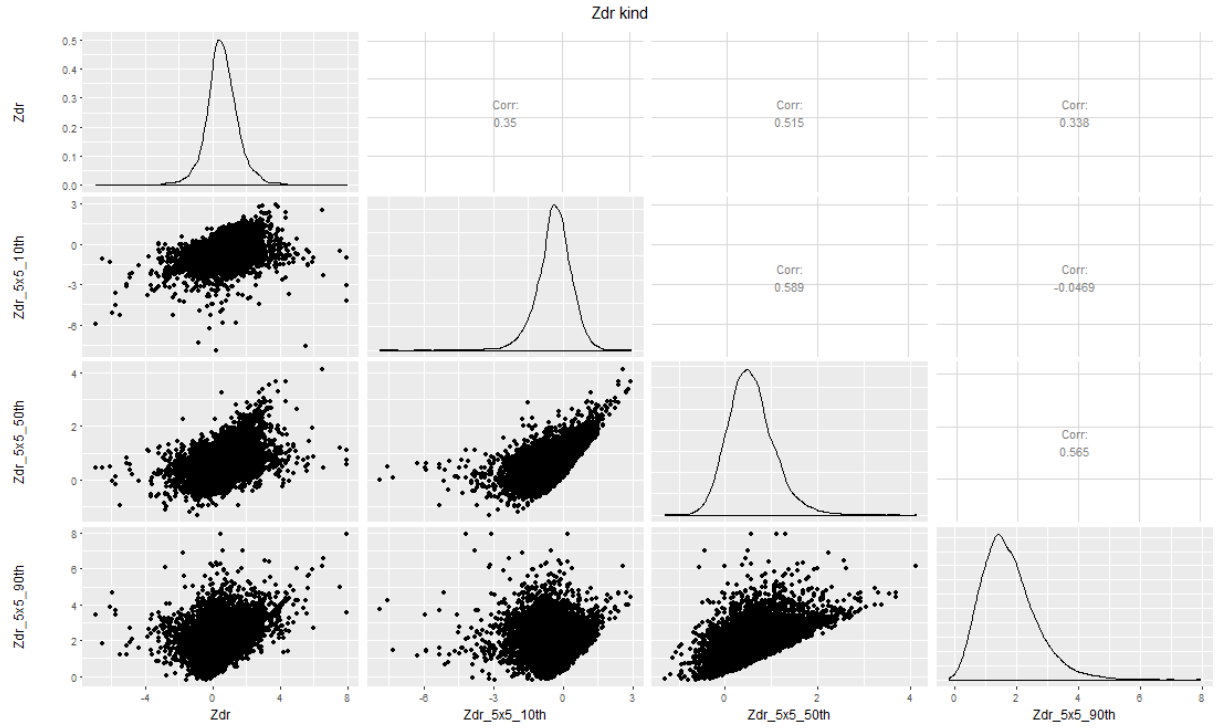
Кроме Ref и RefComposite есть еще 3 «семейства» признаков. Т.е. 3 группы, внутри каждой признаки относятся к схожим показателям. Это RhoHV, Zdr и Kdp.

Изобразим попарную зависимость внутри этих групп.

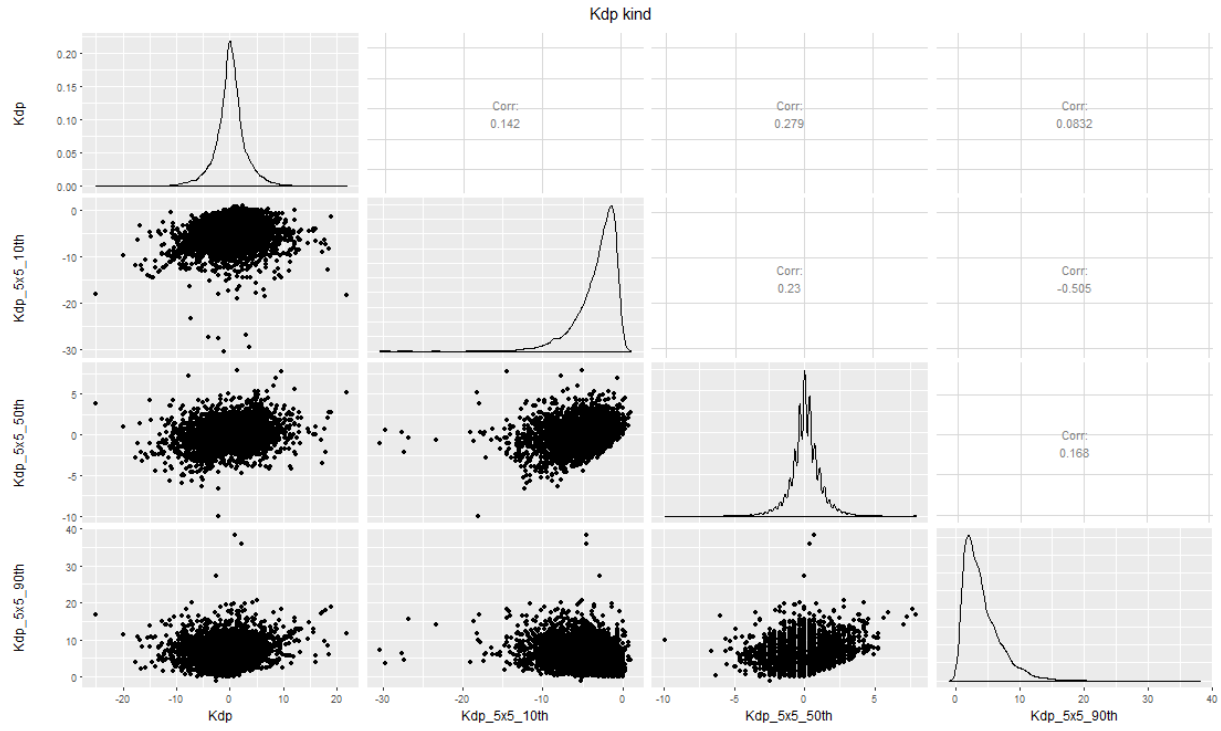
```
ggpairs(sample, columns = c(12:15), title = 'RhoHV kind')

ggpairs(sample, columns = c(16:19), title = 'Zdr kind')

ggpairs(sample, columns = c(20:23), title = 'Kdp kind')
```



В группе RhoHV признаки между собой коррелируют со значением примерно 0.5, на графиках видно, что много точек скапливается около правого угла графика.



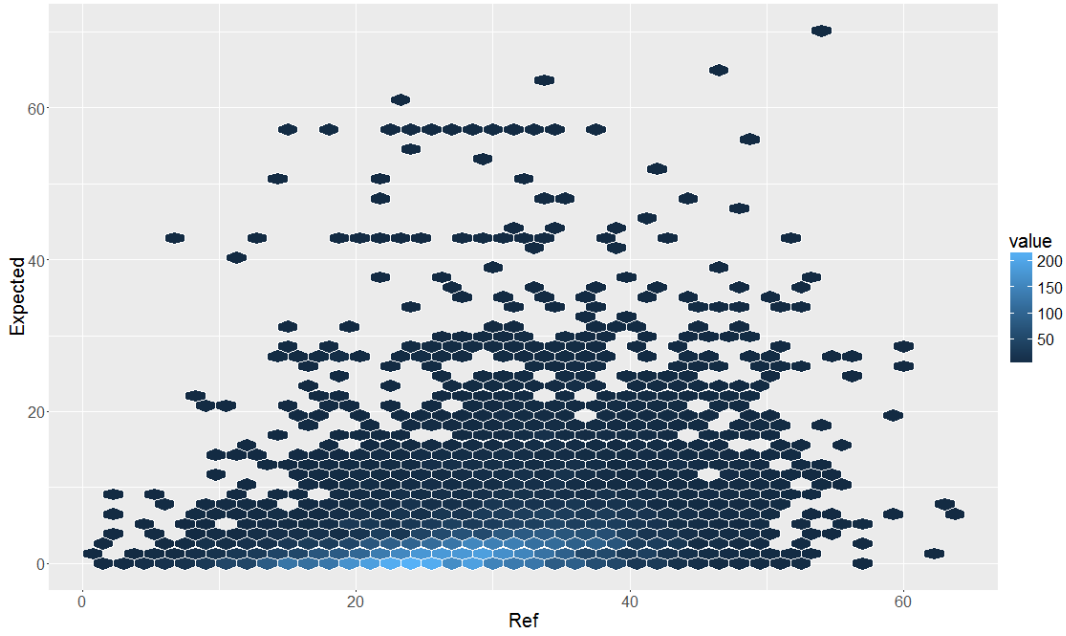
В группе Zdr наблюдаются небольшие корреляции. Примечательно, что корреляция между 10м перцентилем и 90м немного меньше 0. Это связано с тем, что сами значения Zdr могут быть как меньше, так и больше 0.

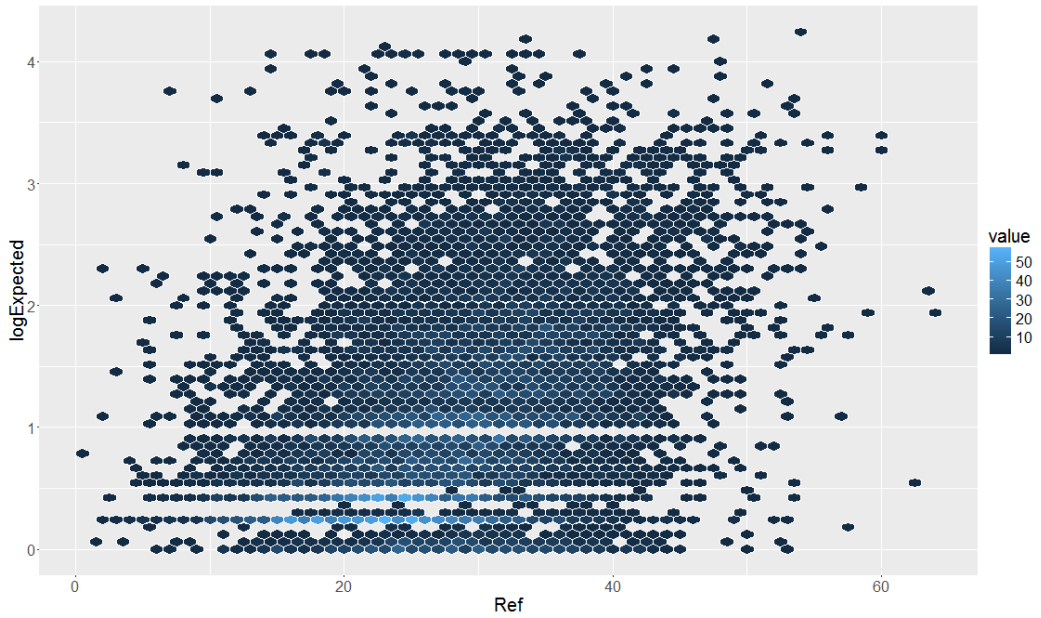


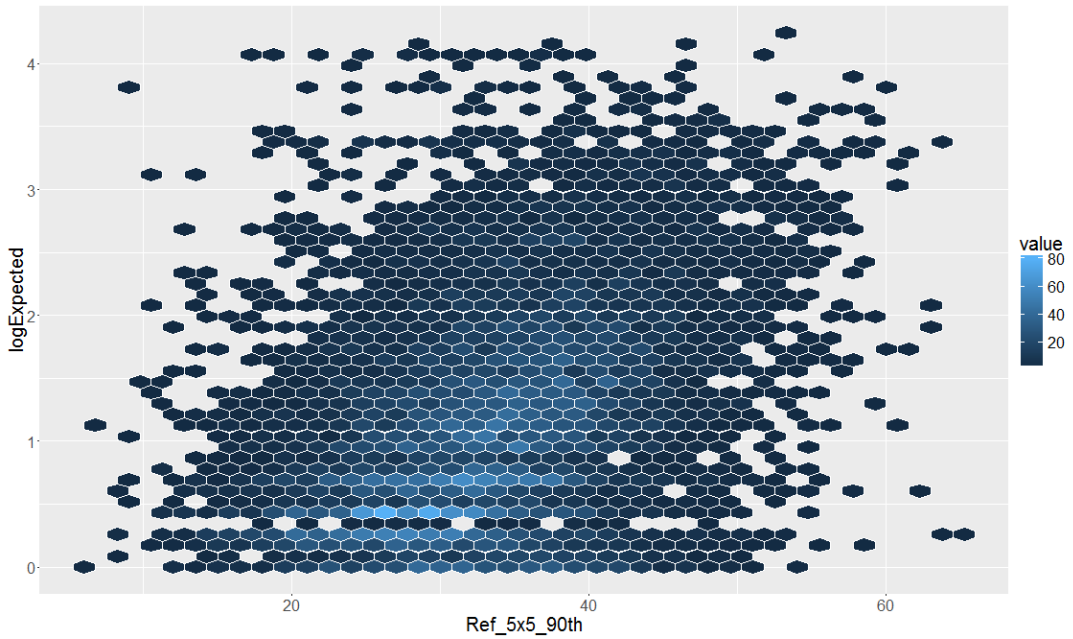
Признаки в семействе Kdp показали наименьшие корреляции, а 10й и 90й перцентили опять имеют отрицательную зависимость, причем довольно сильную

Следующие графики – попарные зависимости прогнозируемых переменных с признаками, показавшими хоть какую-то корреляцию с ними. Эти графики будут построены в стиле binhex: на графике отмечаются шестиугольники, цвет которых зависит от количества попавших в них точек.

```
bh1 <- ggplot(sample, aes(Ref, Expected))
bh1 + stat_binhex(binwidth = c(1.5, 1.5), colour="white")+
  theme(text = element_text(size=20))

bh2 <- ggplot(sample, aes(Ref, logExpected))
bh2 + stat_binhex(binwidth = c(1, 0.07), colour="white")+
  theme(text = element_text(size=20))

bh3 <- ggplot(sample, aes(Ref_5x5_90th, logExpected))
bh3 + stat_binhex(binwidth = c(1.5, 0.1), colour="white")+
  theme(text = element_text(size=20))
```







В явном виде зависимость пронаблюдать не удалось: нет такого, чтобы точки лежали почти на одной прямой. Но заметно, что светлое пятно вытянуто в положительную сторону, что говорит о небольшой зависимости.

Построение моделей

Линейная регрессия

Для начала я решил построить несколько линейных регрессий. То есть исследовать линейную зависимость. Для них, как и в дальнейшем для построения Random Forest будет использован пакет **h2o**, так как он показывает высокую производительность по времени и низкую загрузку памяти, что очень полезно и важно для большого набора данных.

Запустим его и переведем датасет в нужный формат.

```
conn <- h2o.init()
h2o_dt <- as.h2o(fixed_dt)
```

Далее разобьем выборку на обучающую и тестовую в отношении 80/20.

```
dt_split <- h2o.splitFrame(h2o_dt, ratios = 0.8, seed = -1)
train <- dt_split[[1]]
test <- dt_split[[2]]
```

Отмечу, что метрикой для оценки качества модели я буду использовать R^2, так как в моделях будут применяться разные прогнозируемые переменные, а он является относительным показателем:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - y_{pred_i}\right)^2}{\sum_{i=1}^{n}(y_i - y_{mean})^2}$$

Построим 1ю регрессию: Используем все признаки, а прогнозируемой переменной будет Expected.

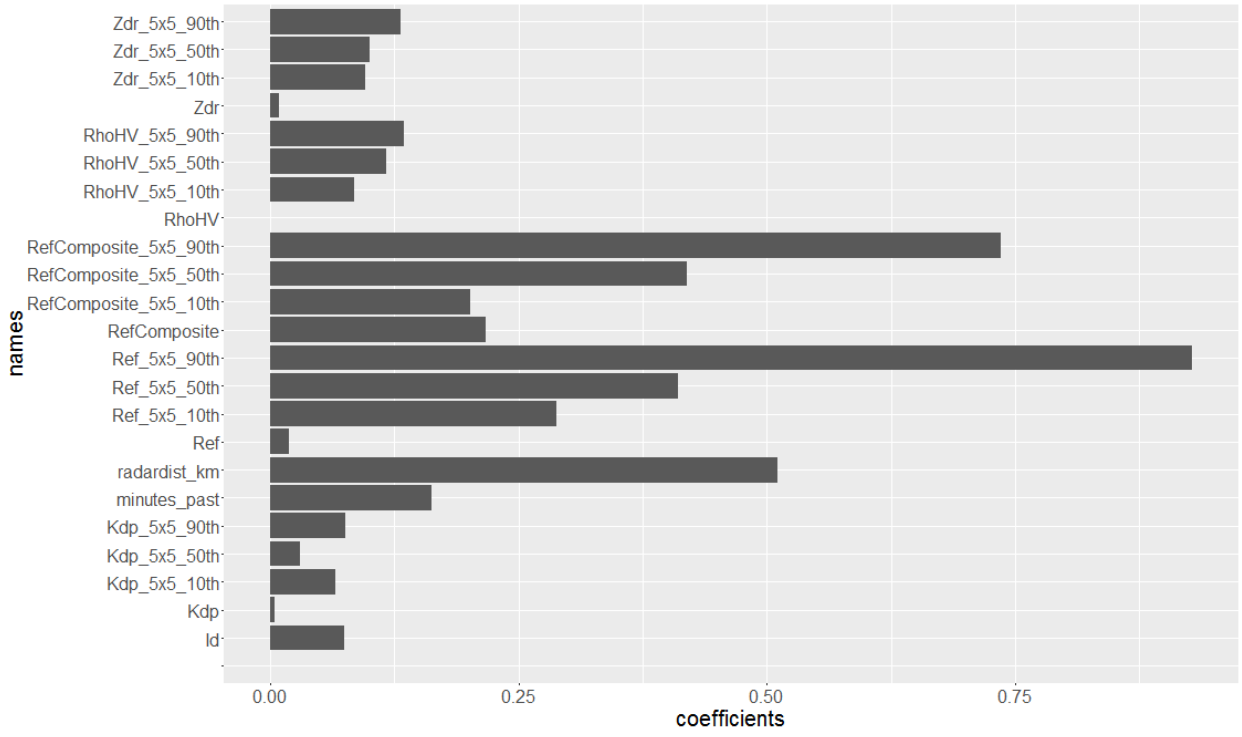
```
glm1 = h2o.glm(y = "Expected", x = c(1:23),
                        training_frame = train, family = "gaussian")
perf <- h2o.performance(glm1, test)
r2_glm1 <- h2o.r2(perf)
```

$R^2 = 0.08554762$

Значение невысокое, но хотя бы какой-то результат есть.

Дальше построим график весов признаков.

```
varimp <- h2o.varimp(glm1)
p1 <- ggplot(data = varimp, aes(x = names, y = coefficients))
p1 <- p1 + geom_bar(stat = "identity")
p1 <- p1 + coord_flip()
p1
```

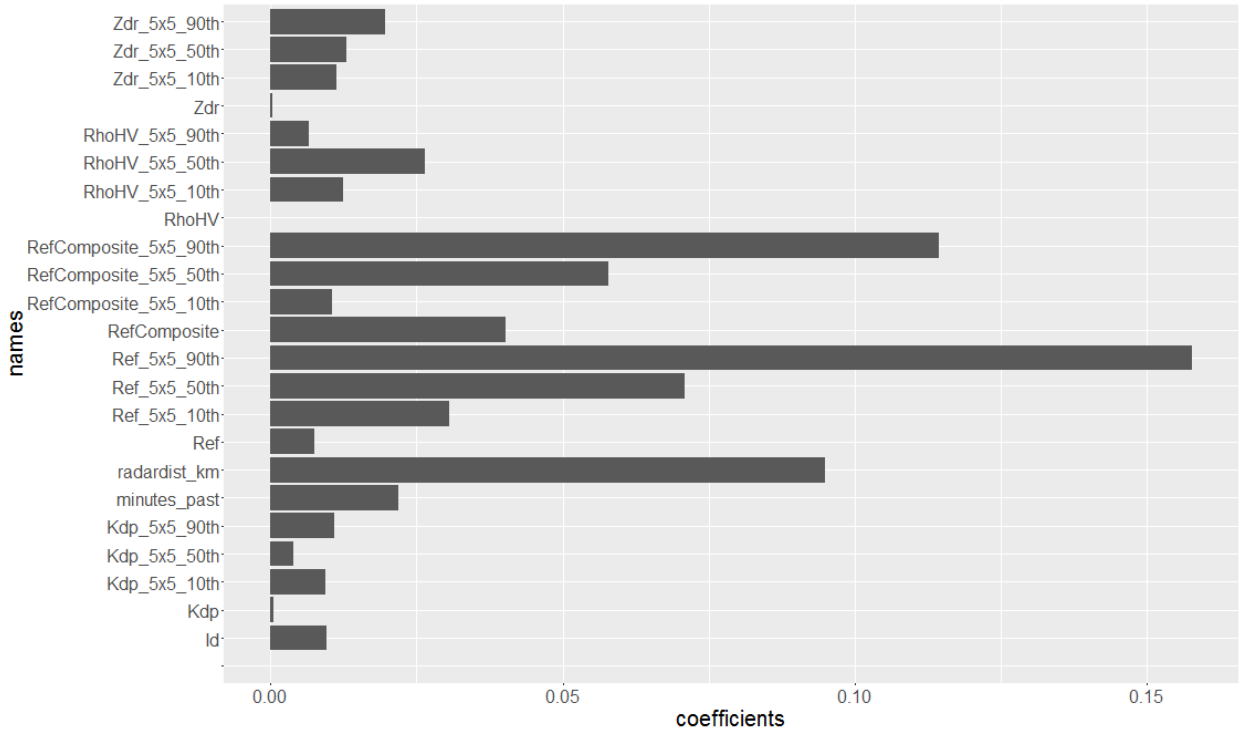


Как и ожидалось, наибольшие веса показали переменные, коррелирующие с Expected: отражаемость света и дистанция от радара до датчика.

2я регрессия будет тоже со всеми признаками, но прогнозировать будем logExpected.

```
glm2 = h2o.glm(y = "logExpected", x = c(1:23),
                          training_frame = train, family = "gaussian")
perf <- h2o.performance(glm2, test)
r2_glm2 <- h2o.r2(perf)
h2o.varimp(glm2)
varimp <- h2o.varimp(glm2)
p2 <- ggplot(data = varimp, aes(x = names, y = coefficients))
p2 <- p2 + geom_bar(stat = "identity")
p2 <- p2 + coord_flip()
p2
```

$R^2 = 0.147$



Разницы в весах между 1й и 2й моделью почти нет.

Теперь попробуем взять всего один самый весомый признак: Ref_5x5_90th и посмотреть, что будет.

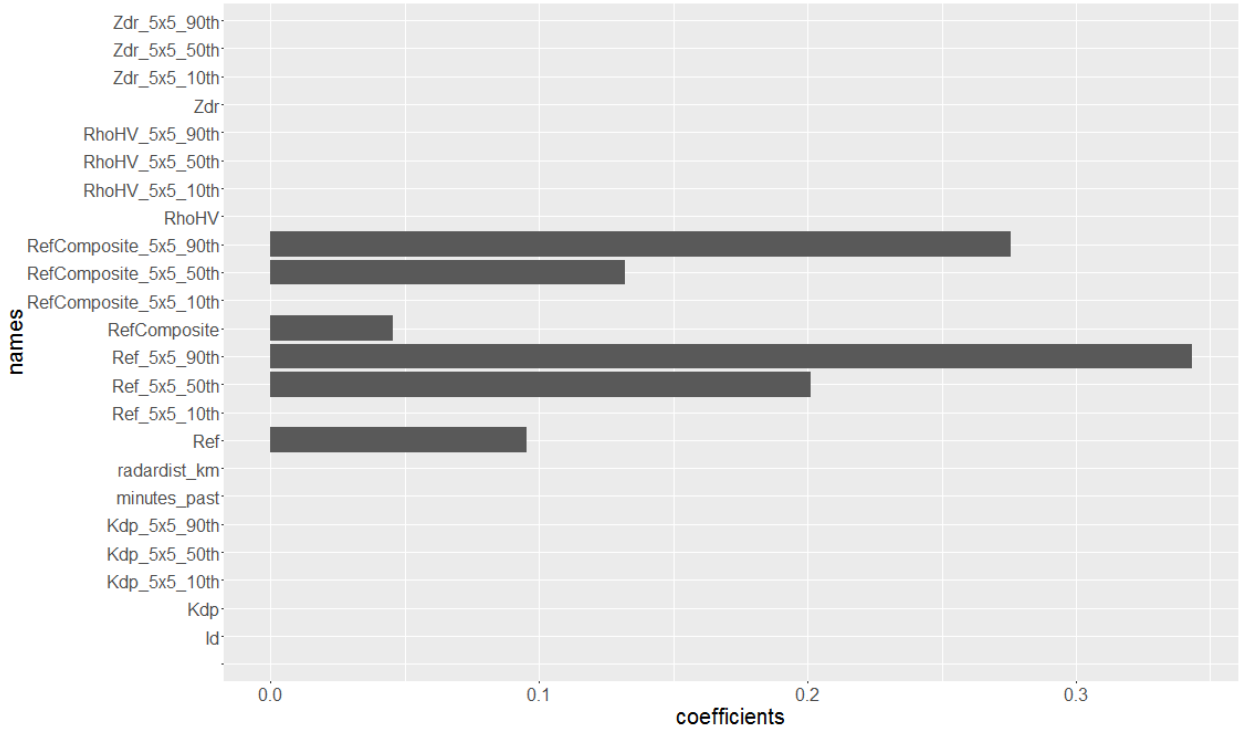
```
glm3 = h2o.glm(y = "logExpected", x = c('Ref_5x5_90th'),
                 training_frame = train, family = "gaussian")
print(glm3)
perf <- h2o.performance(glm3, test)
r2_glm3 <- h2o.r2(perf)
h2o.varimp(glm3)
varimp <- h2o.varimp(glm3)
p3 <- ggplot(data = varimp, aes(x = names, y = coefficients))
p3 <- p3 + geom_bar(stat = "identity")
p3 <- p3 + coord_flip()
p3 + theme(text = element_text(size=20))
```

$$R^2 = 0.132228$$

В четвертой модели я решил применить L1-регуляризацию, чтобы посмотреть,

какие признаки она отсеет.

```
glm4 = h2o.glm(y = "Expected", x = c(1:23),
               training_frame = train, family = "gaussian", lambda = 1)
perf <- h2o.performance(glm4, test)
r2_glm4 <- h2o.r2(perf)
varimp <- h2o.varimp(glm4)
p4 <- ggplot(data = varimp, aes(x = names, y = coefficients))
p4 <- p4 + geom_bar(stat = "identity")
p4 <- p4 + coord_flip()
p4 + theme(text = element_text(size=20))
```



Как видно, остались только признаки связанные с отражаемостью.

$$R^2 = 0.063815$$

Random Forest

Следующий алгоритм – Random Forest. Он универсален и имеет высокую функциональность в моделях с большим количеством признаков. Описание: на каждой итерации делается случайная выборка переменных, после чего, на этой новой выборке запускают построение дерева принятия решений. Операцию проделывают сотни или тысячи раз. Результирующая модель будет результатом “голосования” набора полученных при моделировании деревьев.

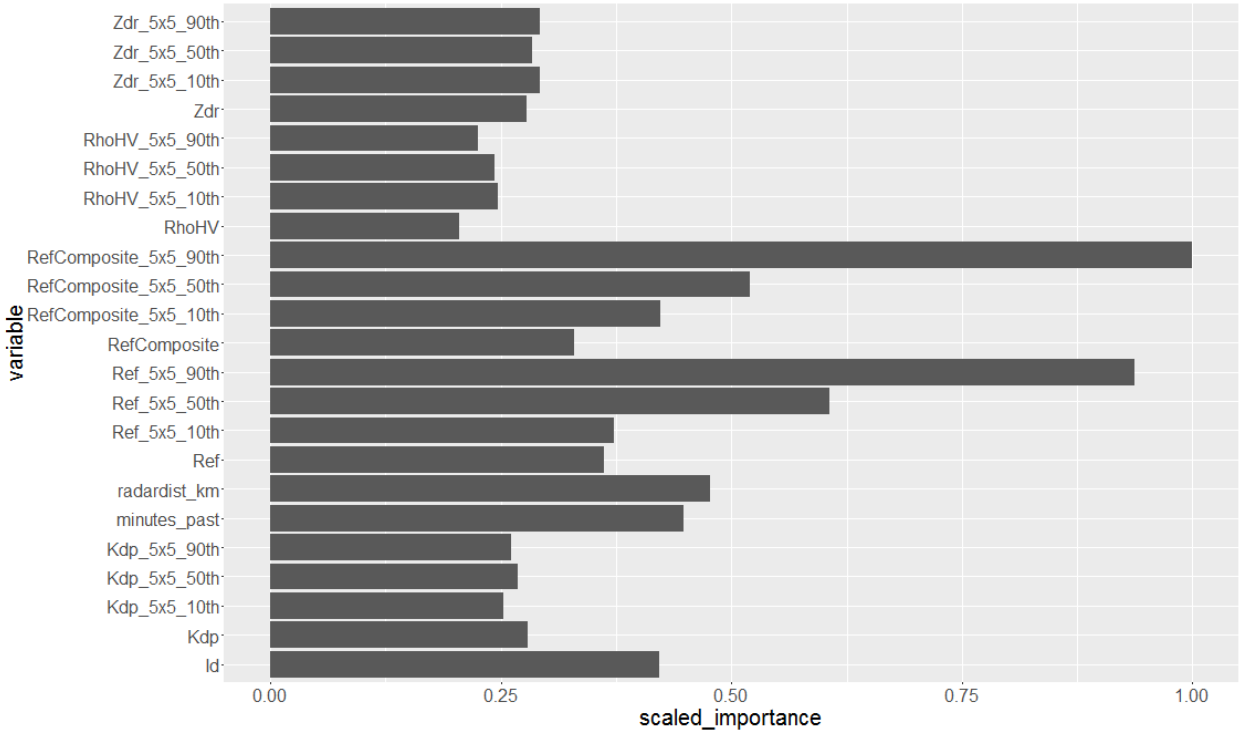
1я модель: все признаки прогнозируют Expected.

Так как выборка очень большая и модель строится долго (примерно 40 минут) я решил не выполнять подбор гиперпараметров (глубины дерева и количества деревьев), а оставить значения по умолчанию.

```
rf1 = h2o.randomForest(y = "Expected", x = c(1:23),
                       training_frame = train)

perf <- h2o.performance(rf1, test)
r2_rf1 <- h2o.r2(perf)
varimp <- h2o.varimp(rf1)
p5 <- ggplot(data = varimp, aes(x = variable, y = scaled_importance))
p5 <- p5 + geom_bar(stat = "identity")
p5 <- p5 + coord_flip()
p5 + theme(text = element_text(size=20))
```

$$R^2 = 0.29516$$

R^2 значительно вырос, что говорит об улучшении модели, но значение все равно не столь велико.



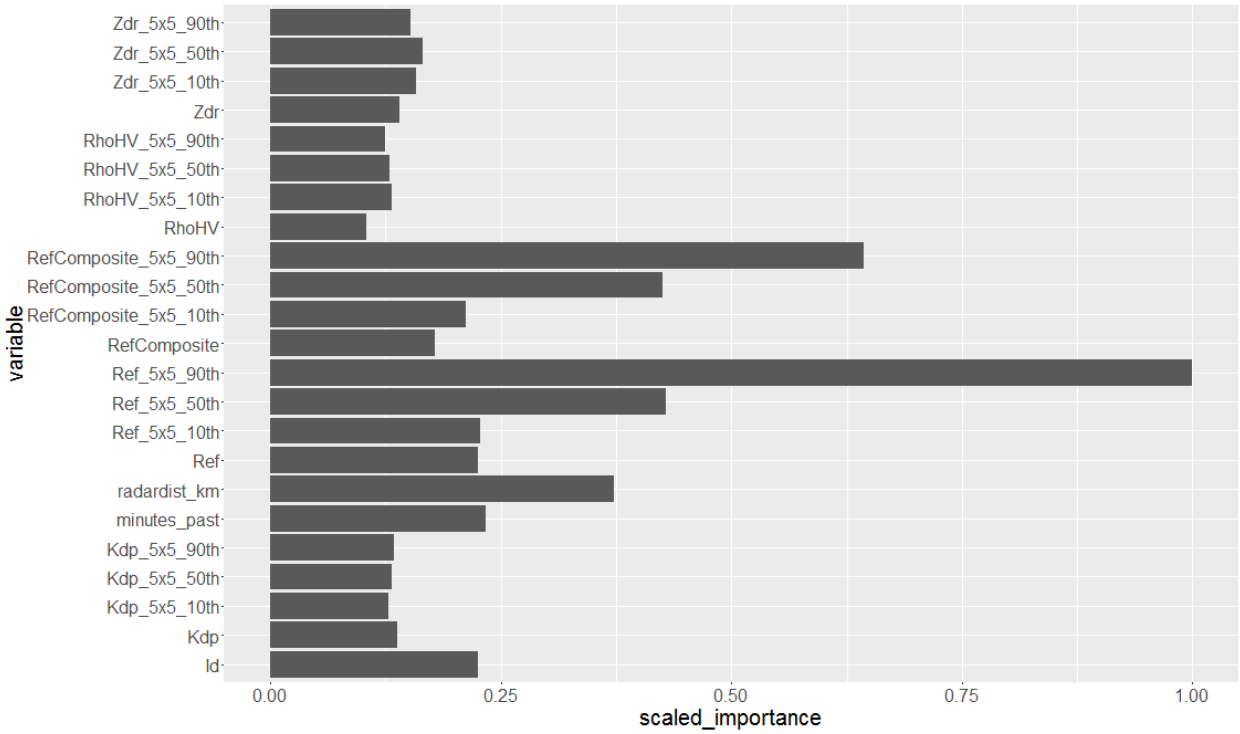
Относительные веса других переменных гораздо больше.

2я модель такая же, только прогнозирует logExpected

```
rf2 = h2o.randomForest(y = "logExpected", x = c(1:23), training_frame = trai
n)
perf <- h2o.performance(rf2, test)
r2_rf2 <- h2o.r2(perf)
varimp <- h2o.varimp(rf2)
p6 <- ggplot(data = varimp, aes(x = variable, y = scaled_importance))
p6 <- p6 + geom_bar(stat = "identity")
p6 <- p6 + coord_flip()
p6 + theme(text = element_text(size=20))
```

$$R^2 = 0.347714$$

R^2 снова вырос, но все равно остался не очень высок.



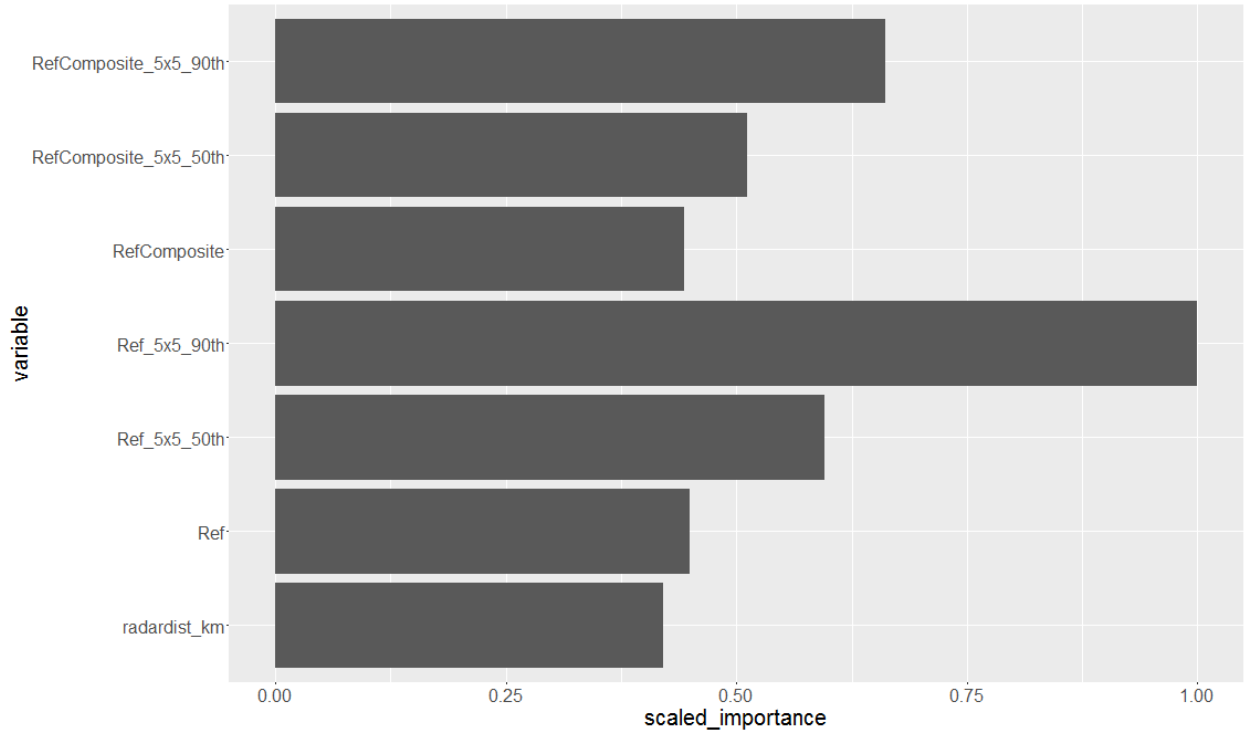
Веса похожи на линейную модель с теми же признаками и прогнозируемой переменной.

В 3й модели я взял наиболее коррелирующие с прогнозируемой переменной признаки.

```
rf3 = h2o.randomForest(y = "logExpected", x = c(3, 4, 6, 7, 8, 10, 11),
                       training_frame = train)

perf <- h2o.performance(rf3, test)
r2_rf3 <- h2o.r2(perf)

varimp <- h2o.varimp(rf3)
p7 <- ggplot(data = varimp, aes(x = variable, y = scaled_importance))
p7 <- p7 + geom_bar(stat = "identity")
p7 <- p7 + coord_flip()
p7 + theme(text = element_text(size=20))
```

$$R^2 = 0.2766969$$

R^2 упал, это показывает, что в моделях случайного леса большинство несут пользу и повышают точность прогноза.



Веса в модели ничего неожиданного не преподнесли.

4ю модель я решил сделать с одним признаком, аналогично линейной.

```
rf4 = h2o.randomForest(y = "logExpected", x = c(7),
                       training_frame = train)
perf <- h2o.performance(rf4, test)
r2_rf4 <- h2o.r2(perf)
varimp <- h2o.varimp(rf4)
p8 <- ggplot(data = varimp, aes(x = variable, y = scaled_importance))
p8 <- p8 + geom_bar(stat = "identity")
p8 <- p8 + coord_flip()
p8 + theme(text = element_text(size=20))
```

$$R^2 = 0.1417992$$

R^2 сильно упал, это доказывает, что в отличие от линейной регрессии в случайном лесу сильно коррелирующие признаки все равно дают немало информации.

Решающее дерево

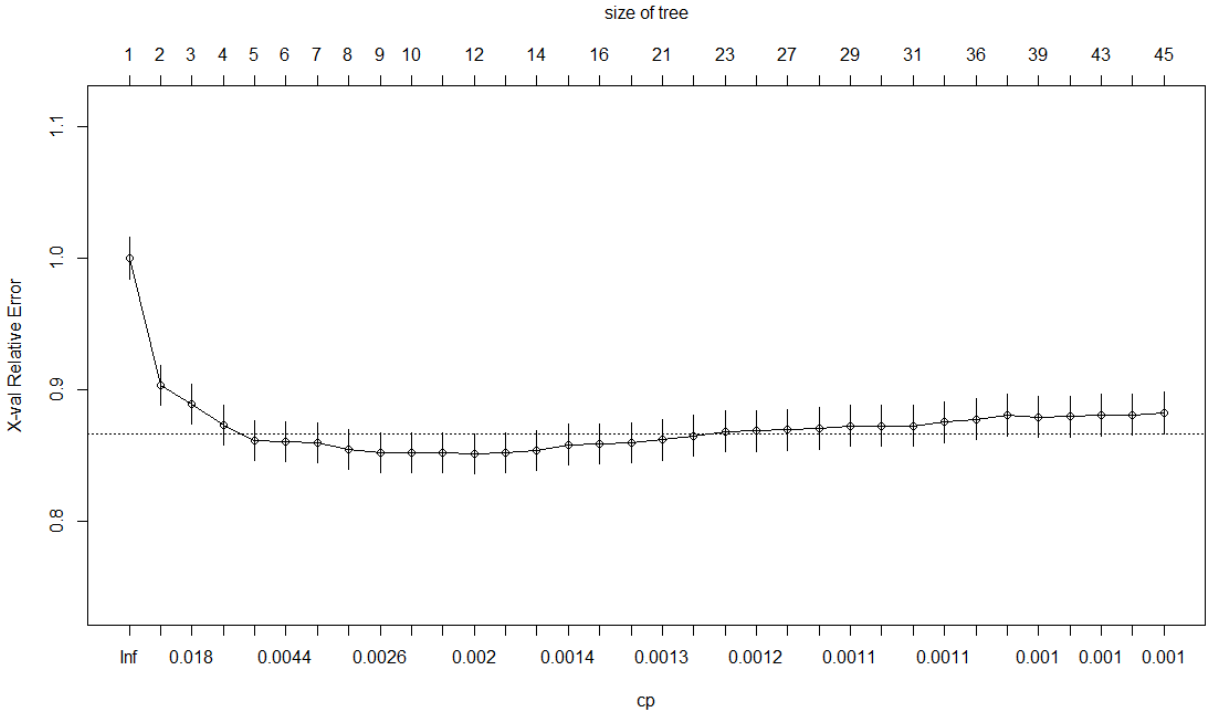
Последним алгоритмом, которым я решил воспользоваться стали решающие деревья. Описание: дерево состоит из веток и листьев. Каждой ветке соответствует признак, а каждому пересечению веток какой-то критерий, с помощью которого объекты разбиваются. Это происходит, пока путь не дойдет до листа, где уже составляется прогноз. Для классификации берется класс с наибольшим количеством объектов, а для регрессии (наш случай) обычно берется среднее значение. Для построения деревьев используется пакет **rpart**. Что удобно, он считает

ошибку кросс-валидации, поэтому необязательно разбивать выборку на тестовую и обучающую. Для визуализации дерева используются пакеты **rattle, rpart.plot, RColorBrewer.**

Для 1го дерева будут взяты все признаки. Поставим минимальное ограничение на количество наблюдений в узле (если меньше, то дальше дерево не растет), а также значение минимального комплексного параметра, чтобы пытаться обрезать малоэффективные разрастания.

```
dtr1 <- rpart(logExpected ~ minutes_past + Ref + Ref_5x5_10th + Ref_5x5_50th
+ Ref_5x5_90th + radardist_km + RefComposite + RefComposite_5x5_10th + RefCom
posite_5x5_50th + RefComposite_5x5_90th + RhoHV + RhoHV_5x5_10th + RhoHV_5x5_
50th + RhoHV_5x5_90th + Zdr + Zdr_5x5_10th + Zdr_5x5_50th + Zdr_5x5_90th + Kd
p + Kdp_5x5_10th + Kdp_5x5_50th + Kdp_5x5_90th, data=sample, method= 'anova',
 control = rpart.control(minsplit=20, cp=0.001))
plotcp(dtr1)
rsq.rpart(dtr1)
```

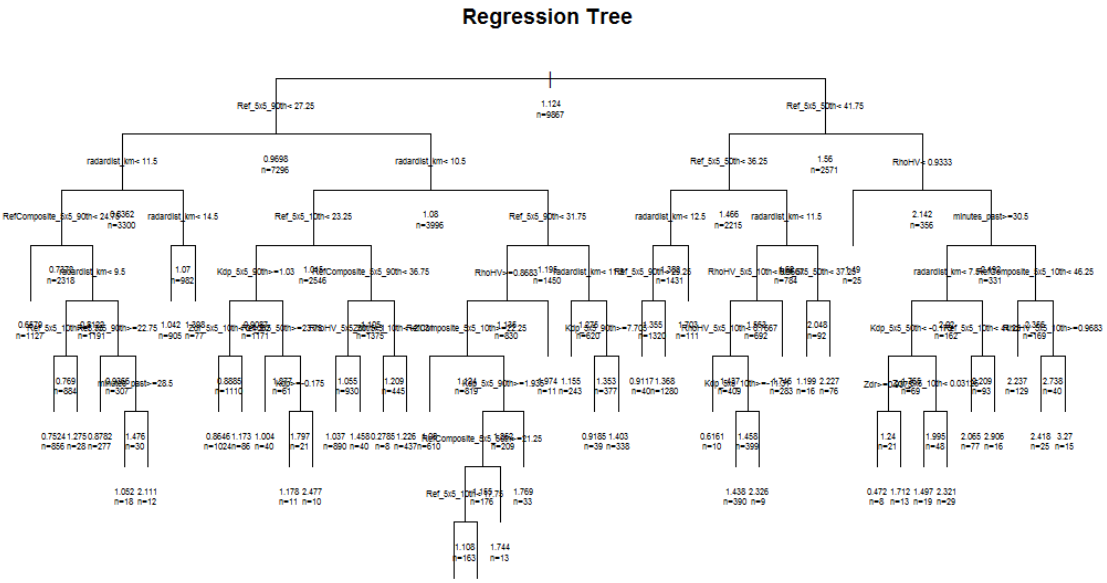
Построим график зависимости ошибки кросс-валидации от размера дерева.



Нетрудно заметить, что в какой-то момент ошибка начинает расти, значит углубления только ухудшают ситуацию.

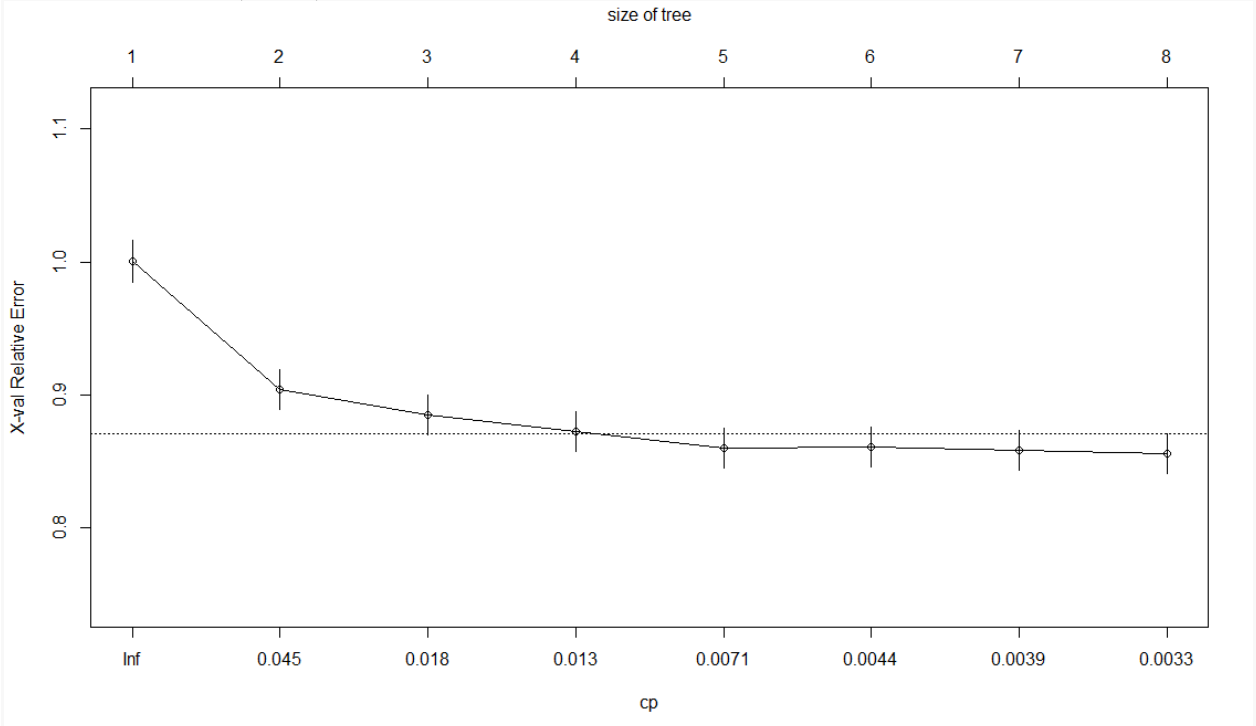
Нарисуем дерево:

```
plot(dtr1, uniform=TRUE, main="Regression Tree")
text(dtr1, use.n=TRUE, all=TRUE, cex=.5)
```



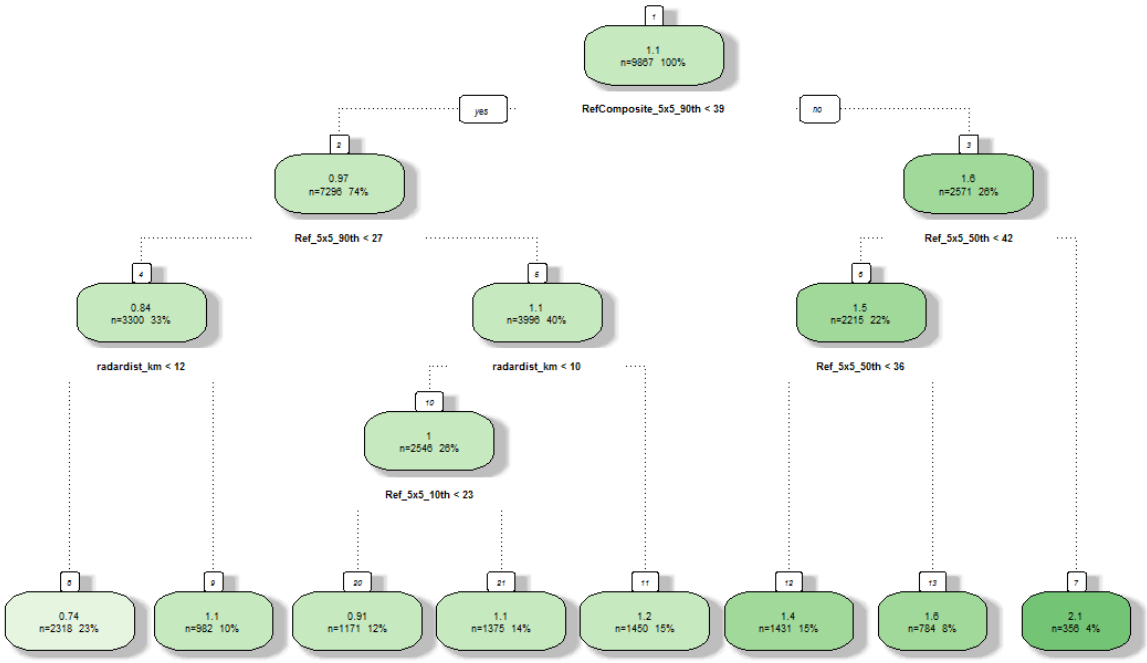
Оно получилось большим, поэтому мало что можно разглядеть. Попробуем повысить комплексный параметр, чтобы избежать излишнего разрастания.

```
dtr2 <- rpart(logExpected ~ minutes_past + Ref + Ref_5x5_10th + Ref_5x5_50th
+ Ref_5x5_90th + radardist_km + RefComposite + RefComposite_5x5_10th + RefCom
posite_5x5_50th + RefComposite_5x5_90th + RhoHV + RhoHV_5x5_10th + RhoHV_5x5_
50th + RhoHV_5x5_90th + Zdr + Zdr_5x5_10th + Zdr_5x5_50th + Zdr_5x5_90th + Kd
p + Kdp_5x5_10th + Kdp_5x5_50th + Kdp_5x5_90th, data=sample, method= 'anova',
 control = rpart.control(minsplit=20, cp=0.003))
plotcp(dtr2)
rsq.rpart(dtr2)
fancyRpartPlot(dtr2)
```



При таких параметрах число листьев достигает 8 и ошибка кросс-валидации все еще убывает.

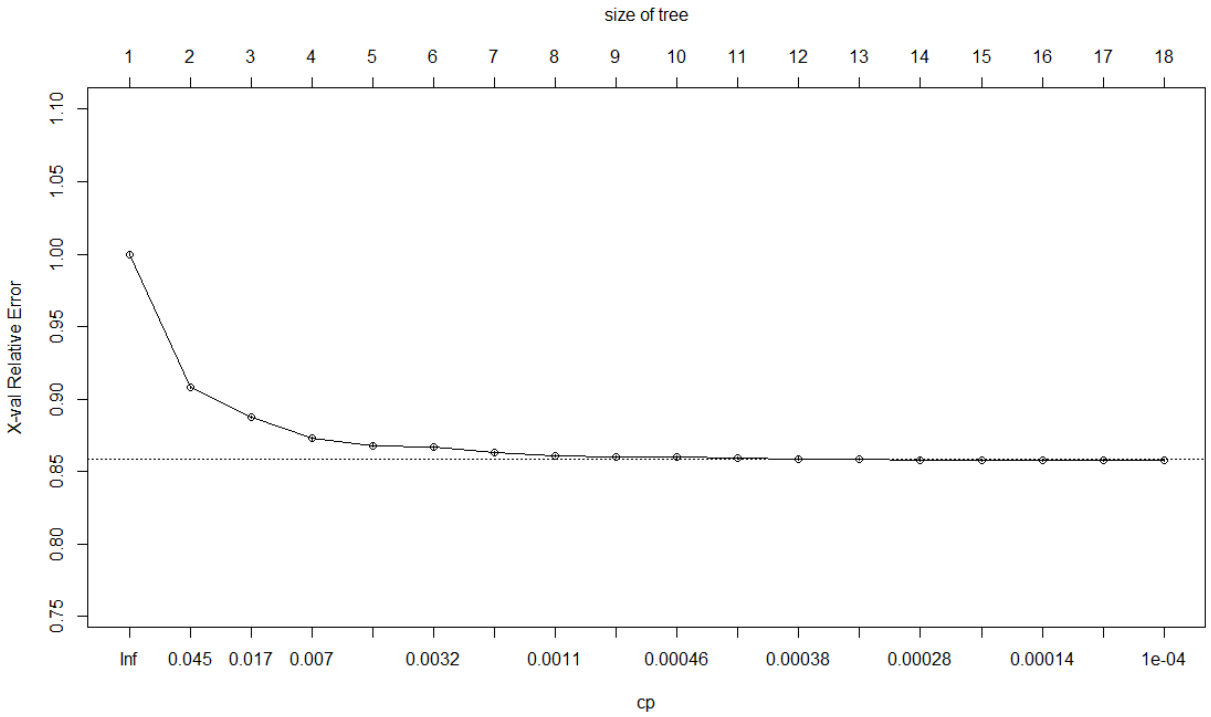
Само дерево выглядит так:



Для построения 3го дерева я решил выбрать 2 признака: Ref_5x5_50th и Ref_5x5_90th, прогнозируем опять logExpexted.

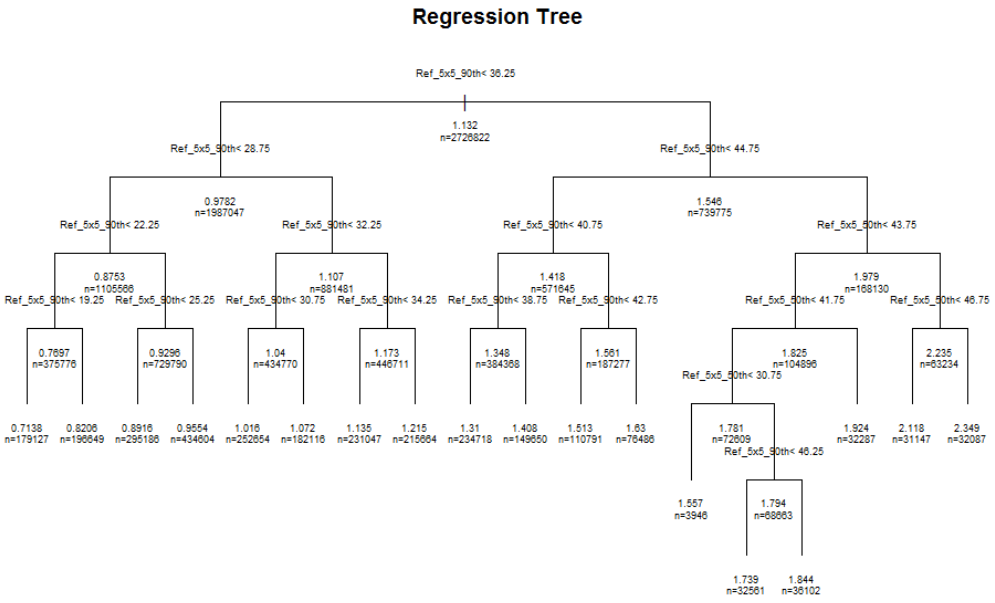
Минимальный комплексный параметр я поставил на уровне 0.0001.

```
dtr3 <- rpart(logExpected ~ Ref_5x5_50th + Ref_5x5_90th, data=fixed_dt, metho
d= 'anova', control = rpart.control(minsplit=20, cp=0.0001))
plotcp(dtr3)
rsq.rpart(dtr3)
plot(dtr3, uniform=TRUE, main="Regression Tree", margin=0.05)
text(dtr3, use.n=TRUE, all=TRUE, cex=.55)
```



Примечательно, что значение относительной ошибки кросс-валидации не падает при росте этого дерева, хотя минимальный комплексный параметр довольно низкий.

Само дерево выглядит так:



Стоит отметить, что относительная ошибка кросс-валидации последнего дерева почти такая же, как и у предыдущего, несмотря на использование всего 2х признаков. Эта ошибка сопоставима с ошибкой лучшей линейной регрессии.

Выводы

Мною была выполнена визуализация и построены модели анализа данных по измерениям поляриметрических радаров, сопоставленных с количеством осадков, оцененным осадкомером. Были изображены различные признаки, отображено их сходство со стандартными распределениями, показаны парные зависимости, а также другие графики, используемые для визуализации параметров самих моделей.

Среди всех моделей наибольшую точность показал Random Forest, приме ненный к переменной logExpected, как прогнозируемой. Наибольший $R^2 = 0.347714$, что не является большим показателем и подтверждает очень слож ную зависимость между прогнозируемой переменной и признаками.

Список источников:

1. Платформа для прогностического моделирования и анализа,

https://www.kaggle.com/

2. Специализированный ресурс о поляриметрических радарах,

http://radarscope.tv/hrf_faq/specific-differential-phase-kdp/