Санкт-Петербургский государственный университет
Прикладная математика и информатика

Отчет по научно-исследовательской работе

«Основные методы решения задачи классификации на примере данных Титаника»

Выполнил:
Дамбаев Биликто Намсараевич
группа 19.Б04-мм

Научный руководитель:
к. ф.-м. н., доцент
П. В. Шпилев
Кафедра Статистического Моделирования

Санкт-Петербург
2022

Оглавление

Введение

В данной работе рассматриваются основные методы решения задачи классификации, активно использующиеся в машинном обучении.

В первой главе ставится общая постановка задачи классификации, а также излагается теоретическое описание рассматриваемых методов.

Вторая глава содержит практическое применение рассматриваемых ранее методов на реальных данных соревнования Titanic: Machine Learning from Disaster, проходящего на платформе Kaggle [1]. В этой главе подробно описывается попытка построения моделей машинного обучения для решения задачи «Титаника». Приводится сравнение результатов каждой модели и их модификаций на тестовой выборке по разным критериям.

Таким образом, основной задачей, поставленной в данной научно-исследовательской работе, является решение задачи бинарной классификации, заключающаяся в прогнозировании выживаемости пассажиров Титаника. Цель данной работы — достичь максимально возможного результата при тестировании обученной модели на проверяющей выборке. Успешным решением поставленной задачи будем считать вхождение в лучшие 15% решений таблицы лидеров, находящейся на сайте Kaggle, в которой в качестве оценки качества решения используется метрика accuracy, т.е. доля верных прогнозов. Решаемая задача является одной из самых популярных активных соревнований по машинному обучению. В рамках исследования огромное внимание уделено анализу признаков, силе их влияния на целевую переменную, а также работе по созданию и выделению новых предикторов.

Для решения задачи используется язык программирования «R». Также для упрощения вида отчета используется язык разметки R Markdown.

Глава 1

Теоретическая часть

1.1. Общая постановка задачи классификации

Пусть X — множество описаний объектов, Y — множество номеров (или наименований) классов. Существует неизвестная целевая зависимость — отображение $y^*\colon X \to Y$, значения которой известны только на объектах конечной обучающей выборки $X^\ell = \{(x_1, y_1), \ldots, (x_\ell, y_\ell)\}$. Требуется построить алгоритм $a\colon X \to Y$, способный классифицировать произвольный объект $x \in X$.

1.2. Метод опорных векторов

Рассмотрим задачу бинарной классификации, в которой объектам из $X = \mathbb{R}^n$ соответствует один из двух классов $Y = \{-1, +1\}$.

Пусть задана обучающая выборка пар "объект-ответ": $X^\ell = (\mathbf{x}_i, y_i)_{i=1}^\ell \subset \mathbb{X}$. Необходимо построить алгоритм классификации $a(\mathbf{x}) : X \to Y$.

1.2.1. Разделяющая гиперплоскость

В пространстве $\mathbb{R}^n$ уравнение $\langle \mathbf{w}, \mathbf{x} \rangle - b = 0$ при заданных $\mathbf{w}$ и b определяет гиперплоскость — множество векторов $\mathbf{x} = (x_1, \ldots, x_n)$, принадлежащих пространству меньшей размерности $\mathbb{R}^{n-1}$. Например, для $\mathbb{R}^1$ гиперплоскостью является точка, для $\mathbb{R}^2$ — прямая, для $\mathbb{R}^3$ — плоскость и т.д. Параметр $\mathbf{w}$ определяет вектор нормали к гиперплоскости, а через $\frac{b}{\|\mathbf{w}\|}$ выражается расстояние от гиперплоскости до начала координат.

Гиперплоскость делит $\mathbb{R}^n$ на два полупространства: $\langle \mathbf{w}, \mathbf{x} \rangle - b > 0$ и $\langle \mathbf{w}, \mathbf{x} \rangle - b < 0$.

Говорят, что гиперплоскость разделяет два класса C_1 и C_2, если объекты этих классов лежат по разные стороны от гиперплоскости, то есть выполнено либо

$$\begin{cases} \langle \mathbf{w}, \mathbf{x} \rangle - b > 0, & \forall x \in C_1 \\ \langle \mathbf{w}, \mathbf{x} \rangle - b < 0, & \forall x \in C_2 \end{cases}$$

либо

$$\begin{cases} \langle \mathbf{w}, \mathbf{x} \rangle - b < 0, \quad \forall x \in C_1 \\ \langle \mathbf{w}, \mathbf{x} \rangle - b > 0, \quad \forall x \in C_2 \end{cases}$$

1.2.2. Линейно разделимая выборка

Пусть выборка линейно разделима, то есть существует некоторая гиперплоскость, разделяющая классы -1 и $+1$. Тогда в качестве алгоритма классификации можно использовать линейный пороговый классификатор:

$a(\mathbf{x}) = sign(\langle \mathbf{w}, \mathbf{x} \rangle - b) = sign\left(\sum_{i=1}^{\ell} w_i x_i - b\right)$

где $\mathbf{x} = (x_1, \ldots, x_n)$ — вектор значений признаков объекта, а $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{R}^n$ и $b \in \mathbb{R}$ — параметры гиперплоскости.

Но для двух линейно разделимых классов возможны различные варианты построения разделяющих гиперплоскостей. Метод опорных векторов выбирает ту гиперплоскость, которая максимизирует отступ между классами:

Если выборка линейно разделима, то существует такая гиперплоскость, отступ от которой до каждого объекта положителен:

$\exists \mathbf{w}, b: \ M_i(\mathbf{w}, b) = y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle - b) > 0, \ i = 1 \ldots \ell$

Мы хотим построить такую разделяющую гиперплоскость, чтобы объекты обучающей выборки находились на наибольшем расстоянии от неё.

Заметим, что при умножении $\mathbf{w}$ и b на константу $c \neq 0$ уравнение $\langle c\mathbf{w}, \mathbf{x} \rangle - cb = 0$ определяет ту же самую гиперплоскость, что и $\langle \mathbf{w}, \mathbf{x} \rangle - b = 0$. Для удобства проведём нормировку: выберем константу c таким образом, чтобы $\min M_i(\mathbf{w}, b) = 1$. При этом в каждом из двух классов найдётся хотя бы один "граничный"объект обучающей выборки, отступ которого равен этому минимуму: иначе можно было бы сместить гиперплоскость в сторону класса с большим отступом, тем самым увеличив минимальное расстояние от гиперплоскости до объектов обучающей выборки.

Обозначим любой "граничный"объект из класса $+1$ как $\mathbf{x}_+$, из класса -1 как $\mathbf{x}_-$. Их отступ равен единице, то есть

$$\begin{cases} M_+(\mathbf{w}, b) = (+1)(\langle \mathbf{w}, \mathbf{x}_+ \rangle - b) = 1 \\ M_-(\mathbf{w}, b) = (-1)(\langle \mathbf{w}, \mathbf{x}_- \rangle - b) = 1 \end{cases}$$

Нормировка позволяет ограничить разделяющую полосу между классами: $\{x : -1 < \langle \mathbf{w}, \mathbf{x}_i \rangle - b < 1\}$. Внутри неё не может лежать ни один объект обучающей выборки. Ширину разделяющей полосы можно выразить как проекцию вектора $\mathbf{x}_+ - \mathbf{x}_-$ на

нормаль к гиперплоскости $\mathbf{w}$. Чтобы разделяющая гиперплоскость находилась на наибольшем расстоянии от точек выборки, ширина полосы должна быть максимальной:

$$\frac{\langle \mathbf{x}_+ - \mathbf{x}_-, \mathbf{w}\rangle}{\|w\|} = \frac{\langle \mathbf{x}_+, \mathbf{w}\rangle - \langle \mathbf{x}_-, \mathbf{w}\rangle - b + b}{\|w\|} = \frac{(+1)(\langle \mathbf{x}_+, \mathbf{w}\rangle - b) + (-1)(\langle \mathbf{x}_-, \mathbf{w}\rangle - b)}{\|w\|} =$$
$$= \frac{M_+(\mathbf{w}, b) + M_-(\mathbf{w}, b)}{\|w\|} = \frac{2}{\|w\|} \to \max \;\Rightarrow\; \|w\| \to \min$$

Это приводит нас к постановке задачи оптимизации в терминах квадратичного программирования:

$$\begin{cases} \|\mathbf{w}\|^2 \to \min\limits_{w,b} \\ M_i(\mathbf{w}, b) \geq 1, \quad i = 1, \ldots, \ell \end{cases}$$

1.2.3. Линейно неразделимая выборка

В случае линейной неразделимости необходимо ослабить ограничения, позволив некоторым объектам попадать на "территорию"другого класса. Для каждого объекта отнимем от отступа некоторую положительную величину ξ_i, но потребуем чтобы эти введённые поправки были минимальны. Это приведёт к следующей постановке задачи:

$$\begin{cases} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum\limits_{i=1}^{\ell} \xi_i \to \min\limits_{w,b,\xi} \\ M_i(\mathbf{w}, b) \geq 1 - \xi_i, \quad i = 1, \ldots, \ell \\ \xi_i \geq 0, \quad i = 1, \ldots, \ell \end{cases}$$

1.3. Метод случайного леса

Метод случайного леса является одним из примеров объединения классификаторов в один ансамбль. Суть метода заключается в использовании большого ансамбля решающих деревьев, каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества результат получается хорошим.

Сначала рассмотрим решающее дерево, а затем сам метод случайного леса.

1.3.1. Решающее дерево

Дерево решений - алгоритм классификации, задающийся связным ациклическим графом, на каждом шаге которого необходимо выбирать признак и значения порога, по которому происходит оптимальное по заданному критерию разбиение. При каждом делении все объекты делятся на две более мелкие группы, т.е. рассматриваемая в каждом из узлов задача разбивается на две более мелкие подзадачи.

При построении дерева используются следующие критерии, по которым происходит разбиение:

- Энтропия Шеннона $S = -\sum_{i}^{N} p_i log_2 p_i$, определяется для каждого множества из разбиения, N — количество возможных классов, и p_i — вероятность объекта принадлежать i-ому классу.
- Неопределенность Джини $G = \sum_{i!=j} p_i p_j = \sum_i p_i * (1 - p_i)$. Максимизацию этого критерия можно интерпретировать как максимизацию числа пар объектов одного класса, оказавшихся в одном поддереве.

1.3.2. Алгоритм построения случайного леса

Пусть обучающая выборка состоит из N образцов.

Алгоритм построения случайного леса, состоящего из N деревьев:

```
for (n: 1,...,N):
  // сгенерировать выборку Xn с помощью бутстрэпа
  Xn = bootstrap(X)
  // построить решающее дерево tn по выборке Xn
  tn = ID3(Xn)
```

Классификация объектов проводится путём голосования: каждое дерево ансамбля относит классифицируемый объект к одному из классов, а побеждает класс, за который проголосовало наибольшее число деревьев.

1.4. Метод k-ближайших соседей

Пусть задана обучающая выборка пар "объект-ответ" $X^\ell = \{(x_1, y_1), \ldots, (x_\ell, y_\ell)\} \subset \mathbb{X}$. И пусть на множестве объектов задана функция расстояния $\rho : \mathbb{X} \times \mathbb{X} \to [0, \infty)$. Необходимо классифицировать новый объект $u \in \mathbb{X}$.

Для произвольного объекта u расположим объекты обучающей выборки x_i в порядке возрастания расстояний до u:

$\rho(u, x_u^{(1)}) \leq \rho(u, x_u^{(2)}) \leq \cdots \leq \rho(u, x_u^{(\ell)})$,

где через $x_u^{(i)}$ обозначается тот объект обучающей выборки, который является i-м соседом объекта u. Алгоритм k ближайших соседей относит объект u к тому классу, представителей которого окажется больше всего среди k его ближайших соседей:

$a(u) = \arg\max_{y \in Y} \sum_{i=1}^{k} [y_u^{(i)} = y]$

Параметр k обычно настраивается с помощью кросс-валидации.

Глава 2

Решение задачи классификации на примере данных Титаника

2.1. Постановка задачи

Дан набор данных, содержащий некоторую информацию о пассажирах корабля. Необходимо построить модель для предсказания выживших пассажиров Титаника.

2.2. Загрузка и обзор данных

Загружаем данные. А также объединим тренировочный и проверочный датасеты в общий для дальнейшей подготовки данных к лучшему виду.

```
train <- read.csv('datasets/train.csv', na.strings = c("NA", ""), stringsAsFactors = F)
test <- read.csv('datasets/test.csv', na.strings = c("NA", ""), stringsAsFactors = F)
test$Survived <- NA
full <- rbind(train, test)
```

Рассмотрим структуру загруженных данных.

```
str(full)
```

```
## 'data.frame':    1309 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence
Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques Heath (Lily May Peel)" ...
##  $ Sex        : chr  "male" "female" "female" "female" ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  NA "C85" NA "C123" ...
##  $ Embarked   : chr  "S" "C" "S" "S" ...
```

Всего 12 переменных 1309 объектов. Ниже представлено подробное описание каждой переменной.

Переменная	Описание
Survived	Выжил (1) или Погиб (0)
Pclass	Класс пассажира
Name	Имя пассажира
Sex	Пол пассажира
Age	Возраст пассажира
SibSp	Количество братьев и сестер/супругов на борту
Parch	Количество родителей/детей на борту
Ticket	Номер билета
Fare	Плата за проезд
Cabin	Номер каюты
Embarked	Порт отправления

Таблица 2.1. Описание переменных набора данных

2.3. Обработка отсутствующих значений

Прежде всего необходимо решить вопрос с отсутствующими значениями. Есть несколько вариантов:

- игнорировать признаки, имеющие недостающие значения;
- игнорировать наблюдения с отсутствующими значениями;
- заполнить недостающие значения (средним значением, медианой или модой);
- предсказать недостающие значения на основе других известных признаков;

Но сначала узнаем у каких признаков есть отсутствующие значения и сколько в каждом из них отсутствующих значений.

```
kable(miss_var_summary(full, order = F))
```

variable	n_miss	pct_miss
PassengerId	0	0.0000000
Survived	418	31.9327731
Pclass	0	0.0000000
Name	0	0.0000000
Sex	0	0.0000000
Age	263	20.0916730
SibSp	0	0.0000000
Parch	0	0.0000000
Ticket	0	0.0000000
Fare	1	0.0763942
Cabin	1014	77.4637128
Embarked	2	0.1527884

Таблица 2.2. Количество отсутствующих значений

Учитывая небольшой размер набора данных, нецелесообразно избавляться ни от целых признаков, ни от наблюдений, имеющих отсутствующие значения. Поэтому остается заполнить значения либо одним и тем же значением (средним, медианой или модой), либо попробовать предсказать их.

2.3.1. Возраст — Age

Ввиду достаточно большого количества пропущенных значений (263 пассажира, что составляет около 20% от общего числа пассажиров), следует спрогнозировать недостающие значения на основе других признаков. Если внимательно изучить строковую переменную Name, то из нее можно выделить такой признак, как титул, который тесно коррелирует с нашим искомым возрастом.

```
head(full$Name)
```

```
[1] "Braund, Mr. Owen Harris"
[2] "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
[3] "Heikkinen, Miss. Laina"
[4] "Futrelle, Mrs. Jacques Heath (Lily May Peel)"
[5] "Allen, Mr. William Henry"
[6] "Moran, Mr. James"
```

Выделим из Name новый признак Title. Далее найдем титулы, у которых пропущено хотя бы одно значение. И заменим недостающие значения на их средний возраст.

```
full$Title <- sapply(full$Name, function(x) {strsplit(x, split='[,.]')[[1]][2]})
full$Title <- trimws(full$Title, which = c("left"))
kable(full %>% group_by(Title) %>%
        summarise(Count = n(), MissingAge = sum(is.na(Age)), Mean = round(mean(Age, na.rm
        = T), 2)))
```

Title	Count	MissingAge	Mean
Capt	1	0	70.00
Col	4	0	54.00
Don	1	0	40.00
Dona	1	0	39.00
Dr	8	1	43.57
Jonkheer	1	0	38.00
Lady	1	0	48.00
Major	2	0	48.50
Master	61	8	5.48
Miss	260	50	21.77
Mlle	2	0	24.00
Mme	1	0	24.00
Mr	757	176	32.25
Mrs	197	27	36.99
Ms	2	1	28.00
Rev	8	0	41.25
Sir	1	0	49.00
the Countess	1	0	33.00

Таблица 2.3. Пропущенные значения возраста, сгруппированных по признаку Title

```
for(title in c("Dr","Master","Miss","Mr","Mrs","Ms")){
  full$Age[is.na(full$Age) & full$Title == title] <-
    mean(full$Age[full$Title == title], na.rm = T)
}
# Проверим количество отсутствующих значений
n_miss(full$Age)
```

```
[1] 0
```

Новосозданный признак Title также стоит сгруппировать и профакторизовать.

```
full$Title[full$Title %in% c("Ms")] <- "Mrs"
full$Title[full$Title %in% c("Mlle", "Mme")] <- "Miss"
full$Title[!(full$Title %in% c("Master", "Miss", "Mr", "Mrs"))] <- "Rare Title"
full$Title <- as.factor(full$Title)
kable(table(full$Sex, full$Title))
```

	Master	Miss	Mr	Mrs	Rare Title
female	0	263	0	199	4
male	61	0	757	0	25

Таблица 2.4. Количественные значения переменной Title, сгруппированные по полу

2.3.2. Плата за проезд — Fare

Fare имеет одно пропущенное значение. Заполним его медианой.

```
full$Fare[is.na(full$Fare)] = median(full$Fare, na.rm = T)
# Проверим количество отсутствующих значений
n_miss(full$Fare)
```

```
[1] 0
```

2.3.3. Номер каюты — Cabin

Cabin имеет 1014 недостающих значений. В процентном соотношении пропущена очень огромная часть — 77.46%. Поэтому разумно не использовать этот признак во избежание добавления шума.

2.3.4. Порт отправления — Embarked

В переменной Embark пропущено два значения. Заполним их модой.

```
#Так как в R нет функции вычисления моды, создадим ее сами
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

full$Embarked[is.na(full$Embarked)] = getmode(full$Embarked)
# Проверим количество отсутствующих значений
n_miss(full$Embarked)
```

[1] 0

Также факторизируем этот категориальный признак.

```
full$Embarked = factor(full$Embarked)
```

2.4. Корреляционная матрица

Для начала попробуем найти главные признаки, влияющие на искомую вероятность выживаемости. Для этого составим корреляционную матрицу и посмотрим на зависимости между имеющимися переменными.

```
full_dropped = full[, !(names(full) %in% c("Name","Cabin","Ticket", "Title"))]
model.matrix(~0+., data=full_dropped) %>%
  cor(use="complete.obs") %>%
  ggcorrplot(show.diag = F, type = "full", lab=TRUE, lab_size=2)
```

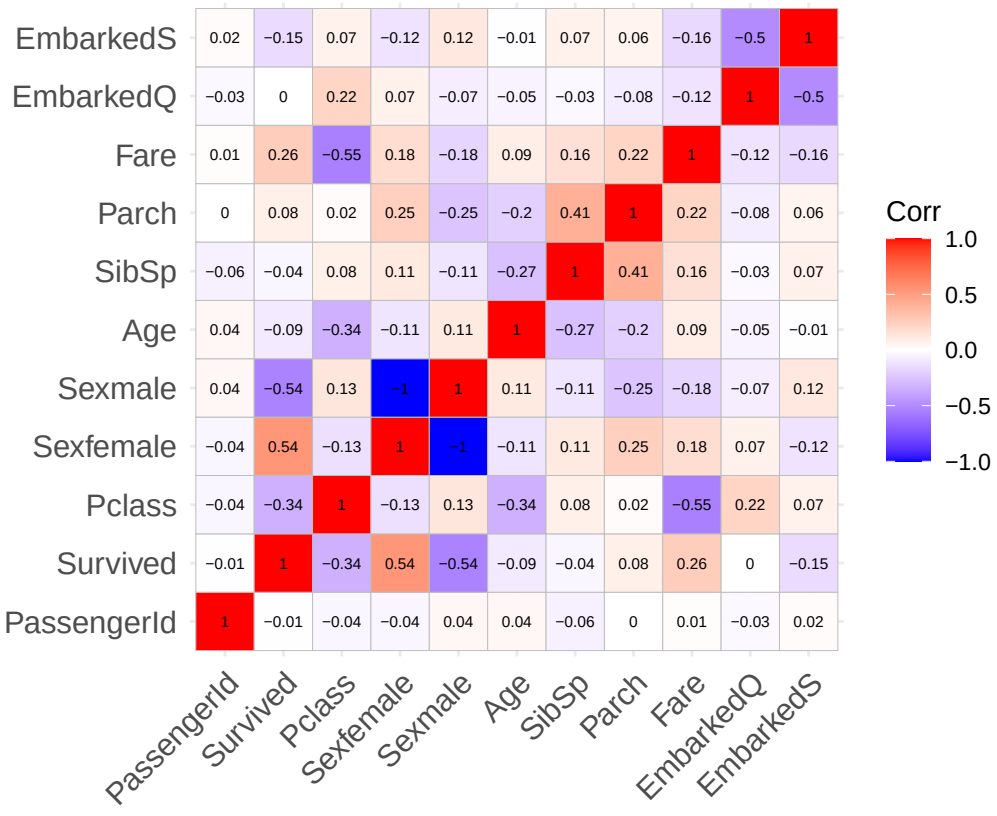

Рис. 2.1. Матрица корреляций

Из матрицы видно, что непосредственное влияние имеют такие признаки, как Pclass, Sex и немного Fare (хотя Fare сильно связан с классом пассажира).

2.5. Исследование наиболее важных признаков

Для нахождения зависимости или независимости между некоторыми признаками и показателем выживаемости построим соответствующие графики.

Но прежде всего, факторизируем категориальные полноценные признаки в тренировочном наборе для удобства последующей визуализации.

```
cat_vars = c("Survived", "Pclass", "Sex");
full[cat_vars] = lapply(full[cat_vars], function(x) as.factor(x));
```

2.5.1. Искомая переменная — Survived

```
ggplot(full[!is.na(full$Survived),], aes(Survived, fill = Survived)) +
  geom_bar(stat = "count") +
  geom_label(stat='count', aes(label=..count..), size = 6) +
  labs(title = "Количество погибших и выживших на Титанике", x = "Survived") +
  theme(plot.title = element_text(hjust = 0.5));
```

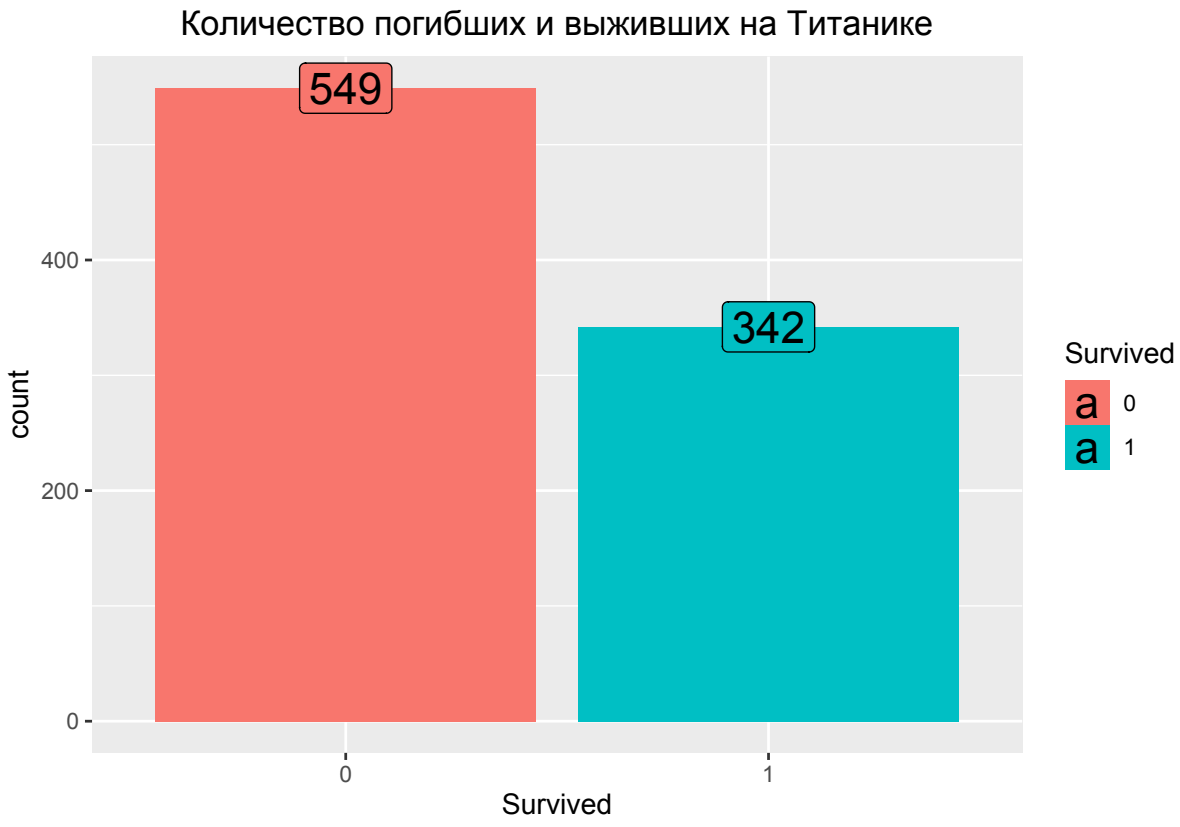

Рис. 2.2. Столбчатый график переменной Survived

Из 891 пассажира 342 выжило и 549 погибло, т.е. 38.4% и 61.6% соответственно. Исходя из этого соотношения, можно сказать в дальнейшем будет легче предсказать гибель чем его спасение.

2.5.2. Пол — Sex

```
plot1 = ggplot(full[!is.na(full$Survived),], aes(Sex, fill = Survived)) +
  geom_bar(stat = "count", position = position_dodge2()) +
  geom_label(stat='count', aes(label=..count..), size = 6, position = position_dodge(0.9),
  show.legend = F) +
  labs(x = "Sex") +
  theme(plot.title = element_text(hjust = 0.5));
plot2 = ggplot(full[!is.na(full$Survived),], aes(Sex, fill = Survived)) +
  geom_bar(stat = "count", position = "fill") +
  labs(x = "Sex", y = "percent") +
  theme(plot.title = element_text(hjust = 0.5));
grid.arrange(plot1, plot2, ncol = 2, top = textGrob("Количество погибших и выживших,
сгруппированных по полу",gp=gpar(fontsize=14,font=1)))
```

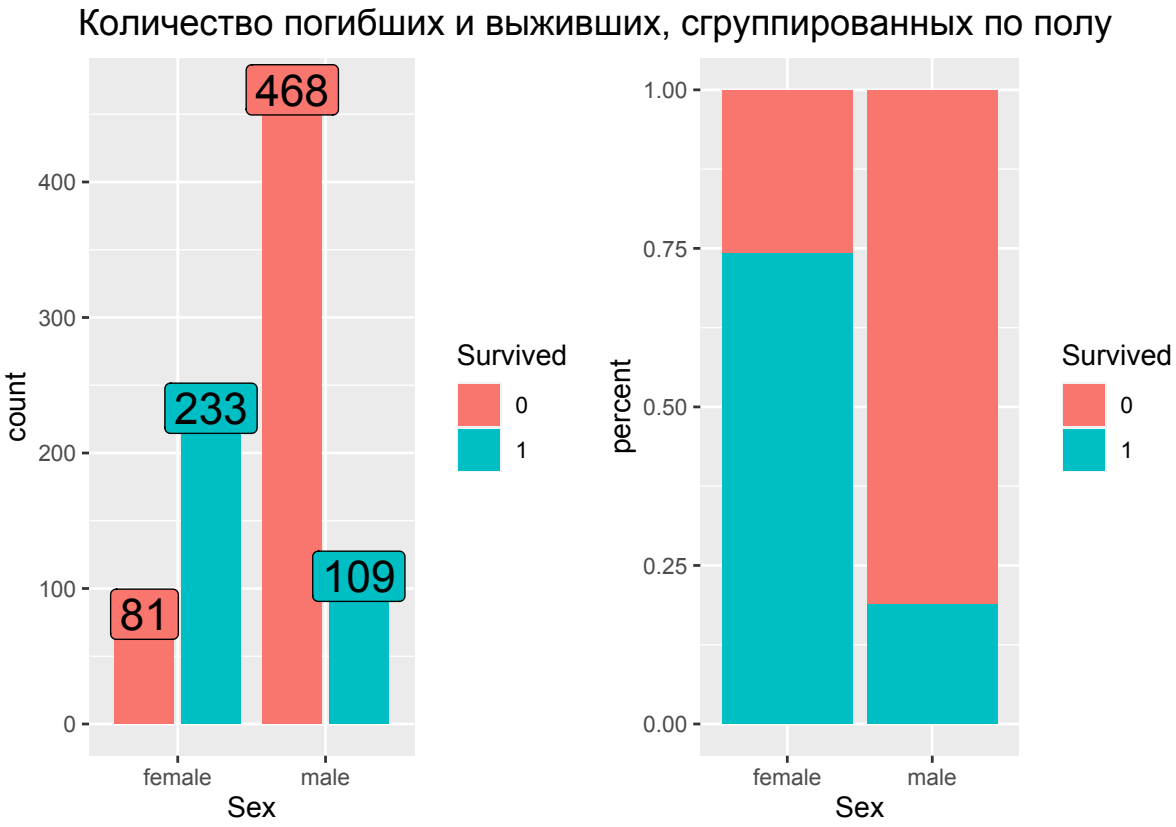

Рис. 2.3. Столбчатый график переменной Survived, сгруппированный по полу

В рамках данных обучения выжили 74.2% женщин и 18.9% мужчин. Благодаря

этой существенной разнице половой признак можно считать одним из важнейших факторов будущего предсказания.

2.5.3. Класс пассажира — Pclass

```
plot1 = ggplot(full[!is.na(full$Survived),], aes(Pclass, fill = Survived)) +
  geom_bar(stat = "count", position = position_dodge2()) +
  geom_label(stat='count', aes(label=..count..), size = 3.5, position = position_dodge(0.9),
  show.legend = F) +
  labs(title = "по классу", x = "Pclass") +
  theme(plot.title = element_text(hjust = 0.5));
plot2 = ggplot(full[!is.na(full$Survived),], aes(Pclass, fill = Survived)) +
  geom_bar(stat = "count", position = "fill") +
  labs(title = "по классу и полу", x = "Pclass", y = "percent") +
  facet_grid(.~Sex) +
  theme(plot.title = element_text(hjust = 0.5));
grid.arrange(plot1, plot2, ncol = 2,
             top = textGrob("Количество погибших и выживших,
             сгруппированных",gp=gpar(fontsize=14,font=1)))
```

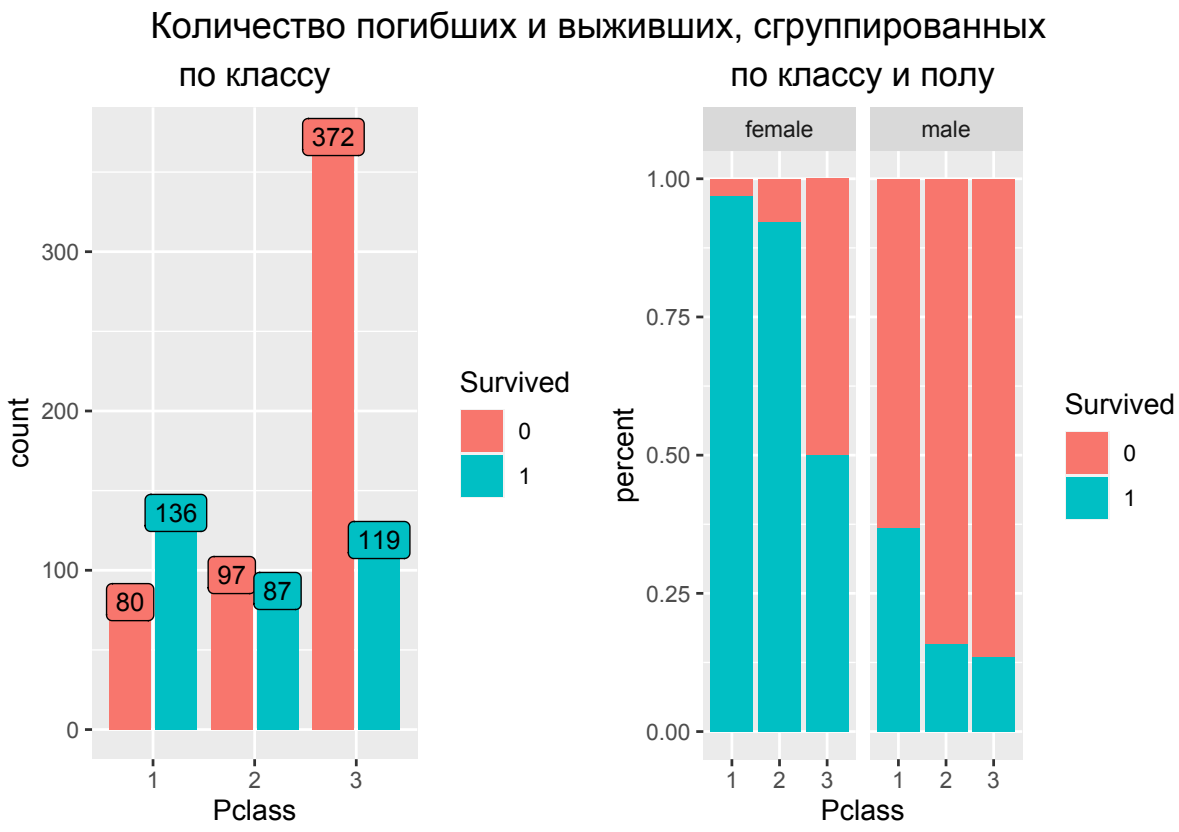

Рис. 2.4. Столбчатый график переменной Survived, сгруппированный по классу

По первому графику видно, что большинство людей было из 3-го класса. Более

того, как и ожидалось, большинство смертей приходится на 3-ий класс.

Также для полноты понимания добавим группирование по половому признаку.

Теперь можно заметить, что:

- женщины из 1-го и 2-го классов почти гарантированно выживают
- мужчины как из 2-го, так и из 3-го класса имеют очень плохие шансы на выживание
- у женщин из 3-го класса шансы на спасение и гибель равны
- доля выживших мужчин из 1-го класса также мала относительно женщин, но гораздо больше остальных мужчин

2.6. Исследование признаков и выделение новых

Попробуем извлечь полезную информацию из разных признаков.

2.6.1. Номер билета — Ticket

Переменная Ticket имеет огромный диапазон разнообразных значений, которые невозможно как-либо сгруппировать. Использование такой переменной несомненно не принесет какой-либо пользы.

```
length(unique(full$Ticket)) / length(full$Ticket)
```

```
[1] 0.7097021
```

2.6.2. Стоимость билета — Fare

Посмотрим на график зависимости Survived от переменной Fare, построив соответствующие плотности.

```
ggplot(full[!is.na(full$Survived),], aes(Fare)) +
  geom_histogram(aes(fill = Survived), binwidth = 5)
```

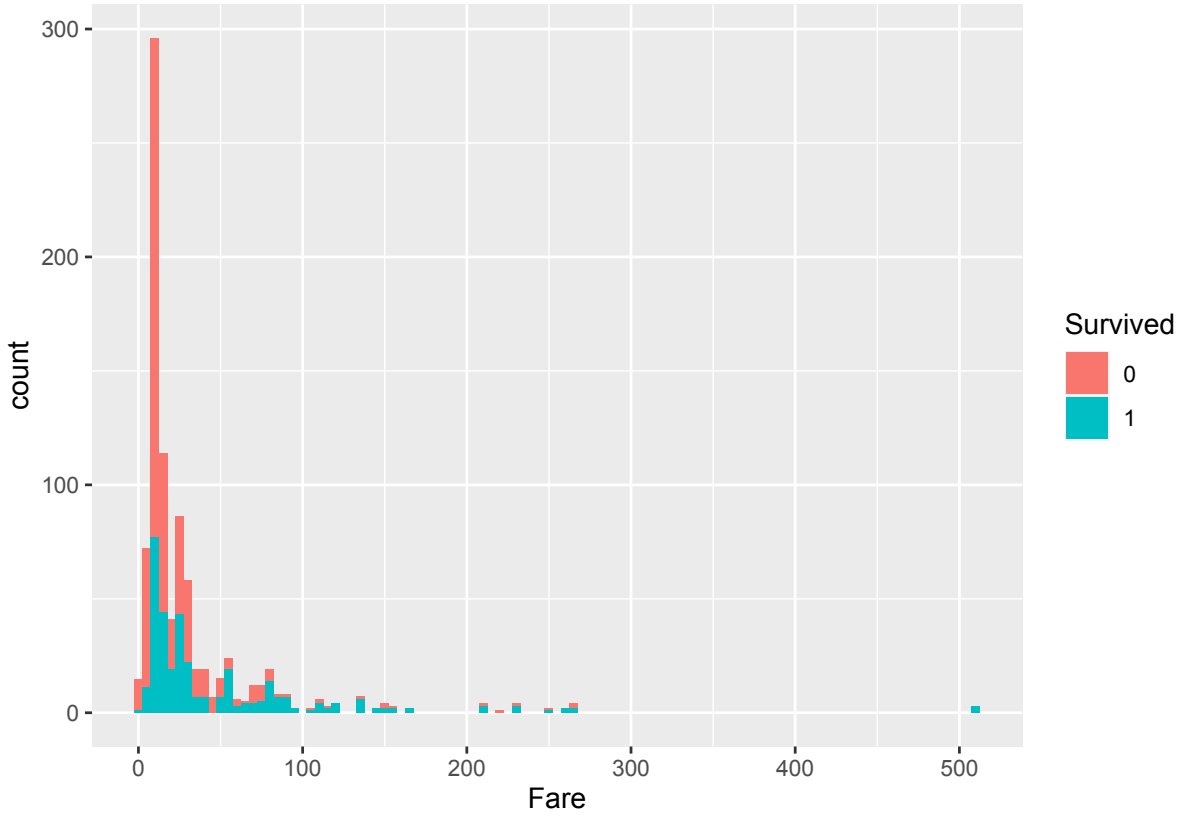

Рис. 2.5. Гистограмма признака Fare, разделенный по Survived

Разобъем переменную Fare на выделяющиеся подгруппы. Для этого найдем точки пересечения плотностей.

```
lower.limit <- min(full$Fare)
upper.limit <- max(full$Fare)
fare_survived_dens <- density(subset(full[!is.na(full$Survived),], Survived == "1")$Fare, from
= lower.limit, to = upper.limit, n = 2^10)
fare_dead_dens <- density(subset(full[!is.na(full$Survived),], Survived == "0")$Fare, from =
lower.limit, to = upper.limit, n = 2^10)

density.difference <- fare_survived_dens$y - fare_dead_dens$y
points <- fare_survived_dens$x[which(diff(density.difference > 0) != 0) + 1]
```

Снова посмотрим на график, но предварительно его сузим и добавим точки пересечения.

```
ggplot(full[!is.na(full$Survived),], aes(Fare)) +
  geom_density(alpha = 0.5, aes(fill = Survived)) +
  geom_vline(xintercept = points) +
  xlim(0, 200)
```

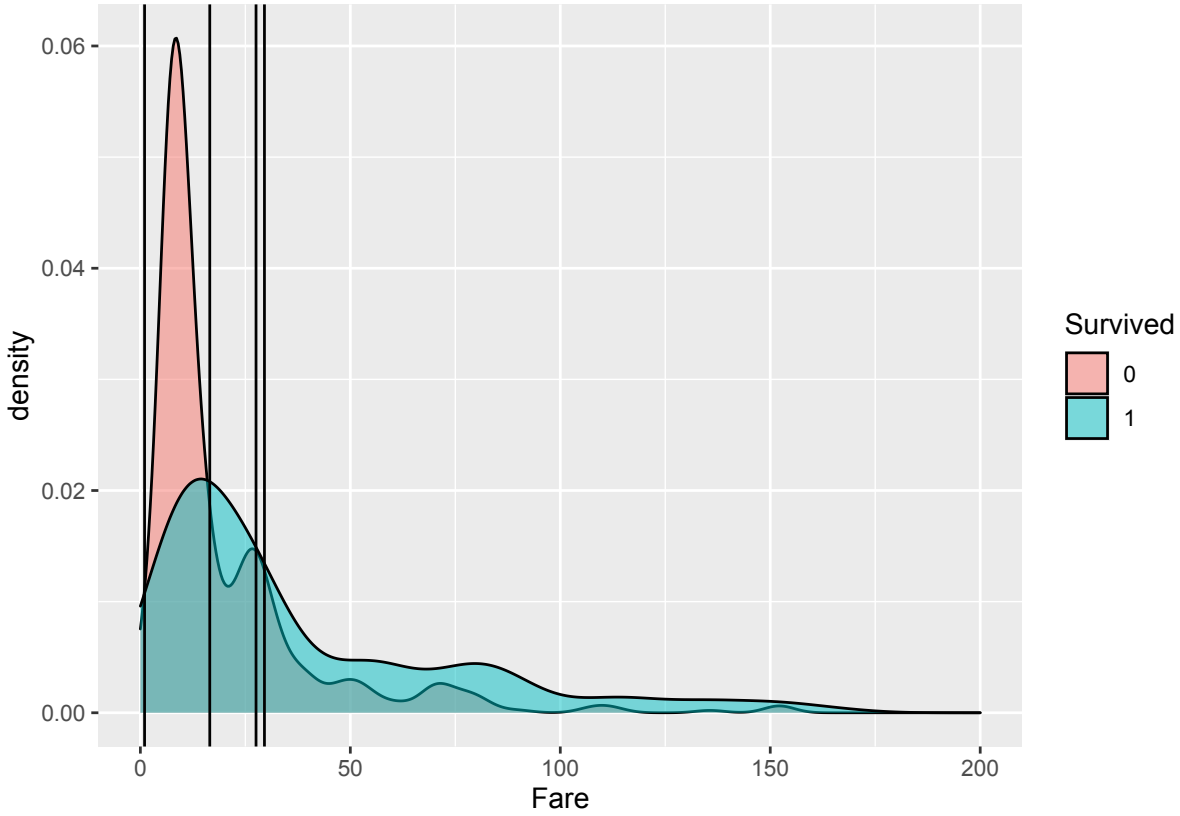

Рис. 2.6. График плотностей Fare, разделенный по Survived

Разобъем Fare на две подгруппы, создав новый переменную FareGroup. Выберем в качестве разделителя подгрупп — 2-ую точку слева.

```
full$FareGroup[full$Fare <= points[2]] <- "0"
full$FareGroup[full$Fare > points[2]] <- "1"
full$FareGroup = factor(full$FareGroup)
```

Далее построим график зависимость Survived по переменным FareGroup и Pclass.

```
ggplot(full[!is.na(full$Survived),], aes(FareGroup, fill = Survived)) +
  geom_bar(stat = "count", position = position_dodge2()) +
  facet_grid(.~ Pclass)
```

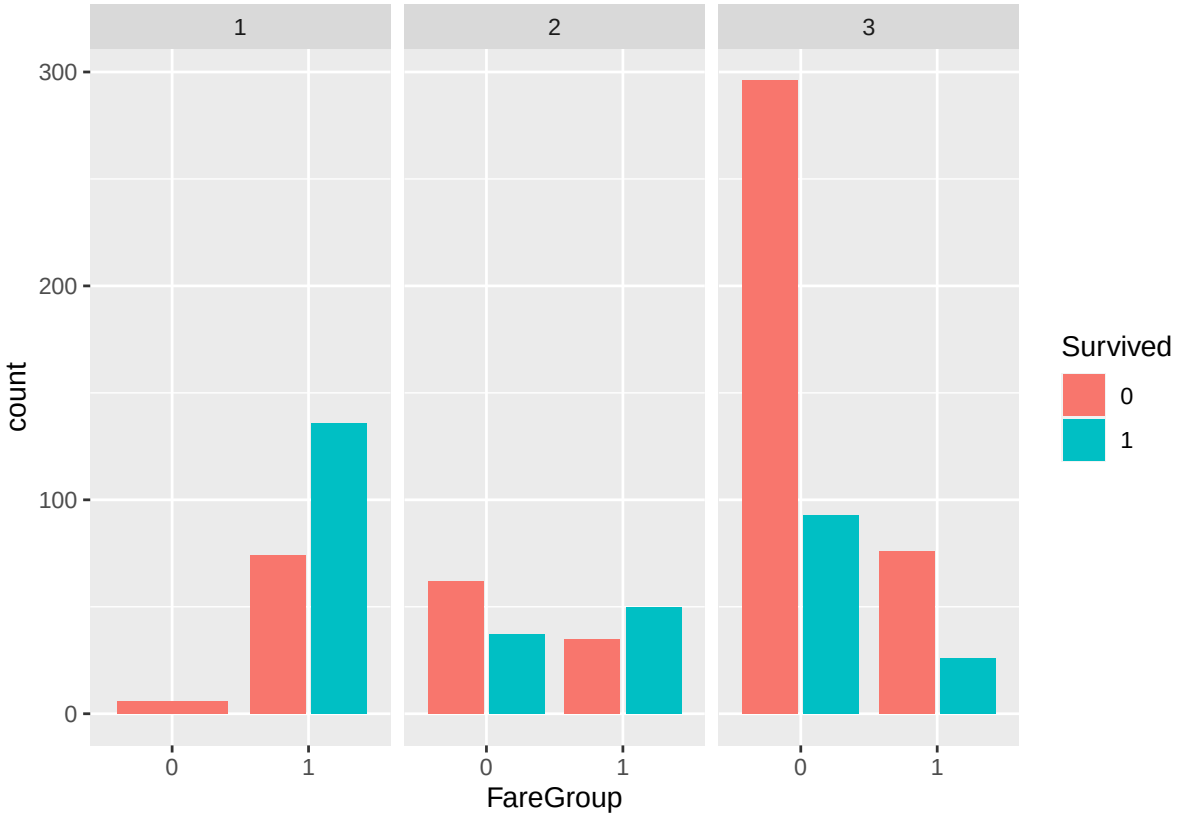

Рис. 2.7. Столбчатая диаграмма переменной Survived, сгруппированная по FareGroup и Pclass

Исходя из графика, можно сказать, что выделенный нами признак FareGroup вносит дополнительную информацию в разделении признака Pclass (1 и 2 классы), но при этом также имеет с ним некоторую зависимость (3 класс).

2.6.3. Размер семьи — FamSize

Создадим новый признак FamilySize, соединив два изначальных SibSp и Parch, обозначающих количество братьев и сестер/супругов на борту и количество родителей/детей на борту соотвественно.

```
full$FamSize <- full$SibSp + full$Parch + 1
```

Далее построим график влияния нашей новой переменной на показатель выживаемости.

```
ggplot(full[!is.na(full$Survived),], aes(FamSize, fill = Survived)) +
  geom_bar(stat = "count", position = position_dodge2()) +
  scale_x_continuous(breaks = c(1:max(full$FamSize))) +
```

```
  labs(x = "FamSize") +
  theme(plot.title = element_text(hjust = 0.5));
```

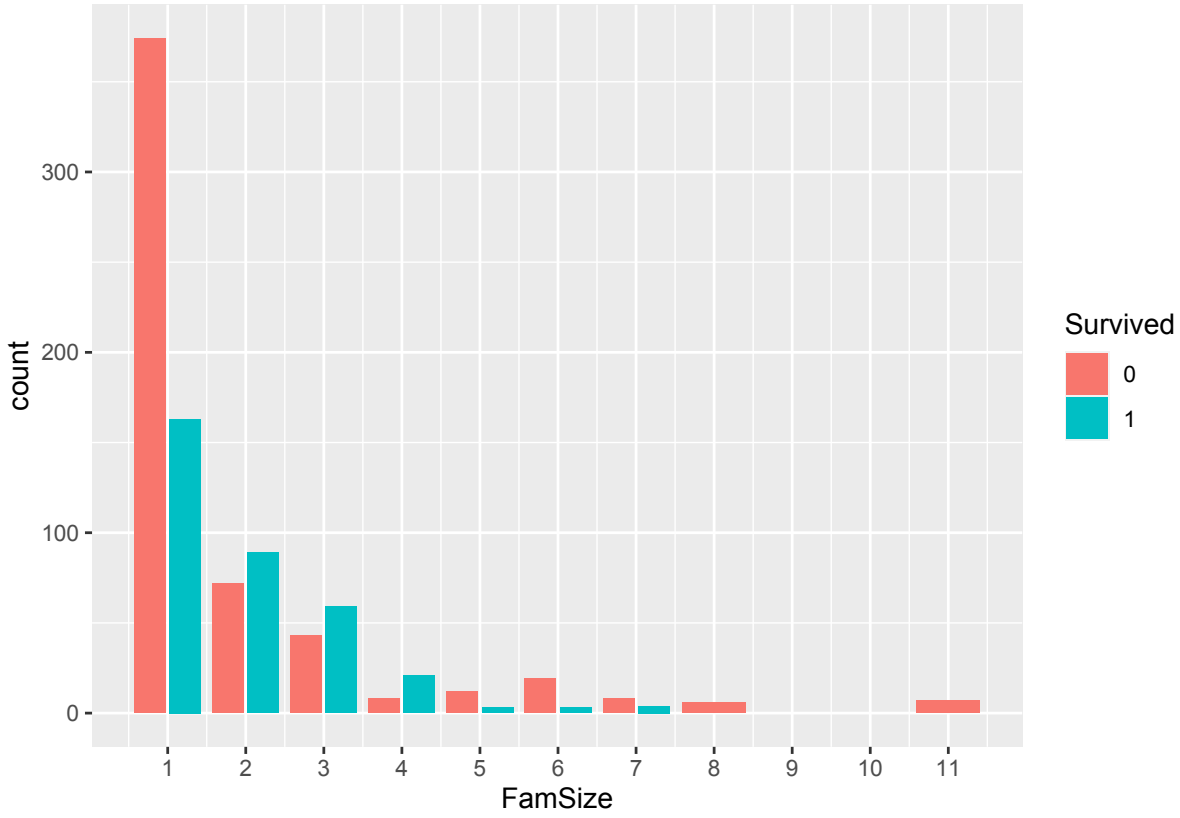

Рис. 2.8. Столбчатая диаграмма переменной Survived, сгруппированая по количеству родственников

По вышерасположенному графику можно заметить, что у людей, которые путешествовали в одиночку, шансы погибнуть были гораздо выше чем выжить. Также невезло и большим семьям. А люди, путешествовавшие небольшой семьей, имели относительно высокую вероятность спастись.

В силу относительно малого количества больших семей, будет логично разбить переменную на три разные группы.

```
full$GroupSize[full$FamSize == 1] <- "single"
full$GroupSize[full$FamSize > 1 & full$FamSize <= 4] <- "small"
full$GroupSize[full$FamSize > 4] <- "large"
full$GroupSize = factor(full$GroupSize, levels = c("single", "small", "large"))

ggplot(full[!is.na(full$Survived),], aes(GroupSize, fill = Survived)) +
  geom_bar(stat = "count", position = position_dodge2()) +
```

```
labs(x = "GroupSize") +
theme(plot.title = element_text(hjust = 0.5));
```

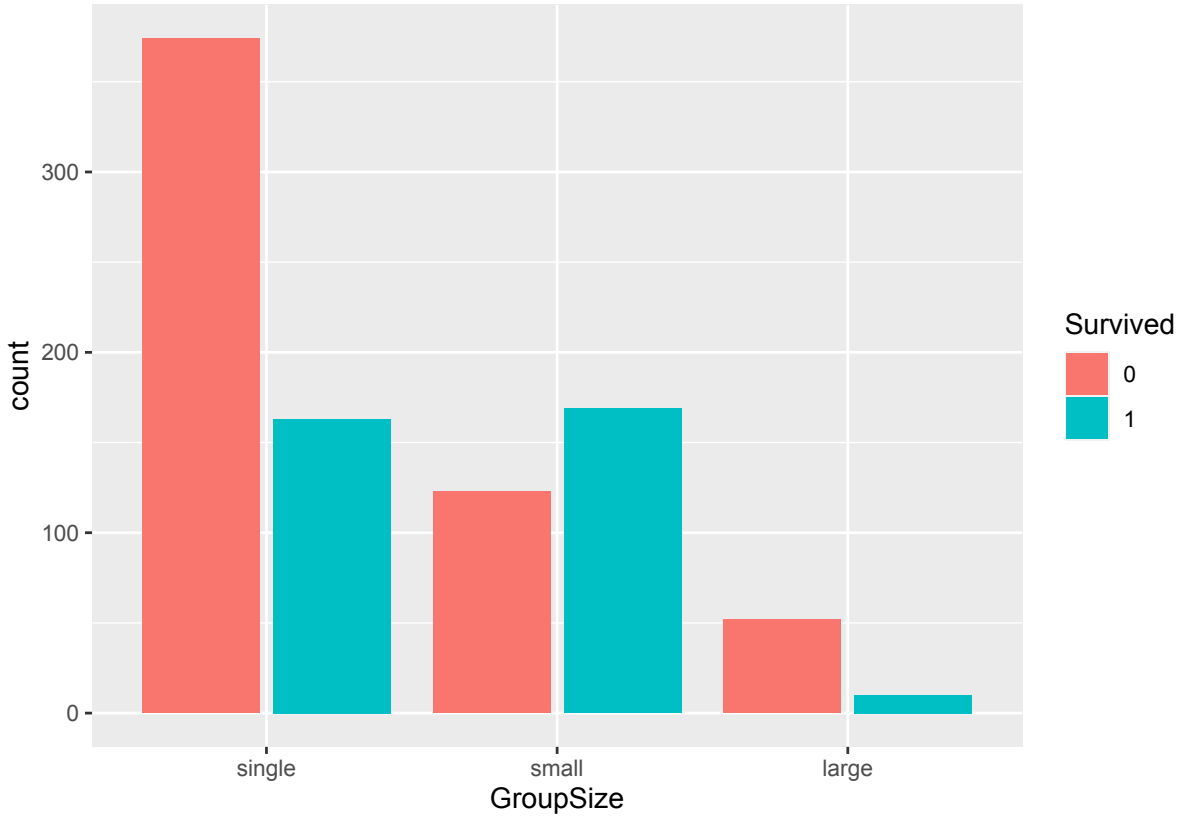

Рис. 2.9. Столбчатая диаграмма переменной Survived, сгруппированная по GroupSize

2.7. Корреляционная матрица (обновленная)

Снова посмотрим на корреляционную матрицу, но уже с новыми переменными.

```
full_dropped = full[, !(names(full) %in% c("Name","Cabin","Ticket", "SibSp", "Parch",
"Embarked"))]
model.matrix(~0+., data=full_dropped) %>%
  cor(use="complete.obs") %>%
  ggcorrplot(show.diag = F, type = "full", lab=TRUE, lab_size=2)
```

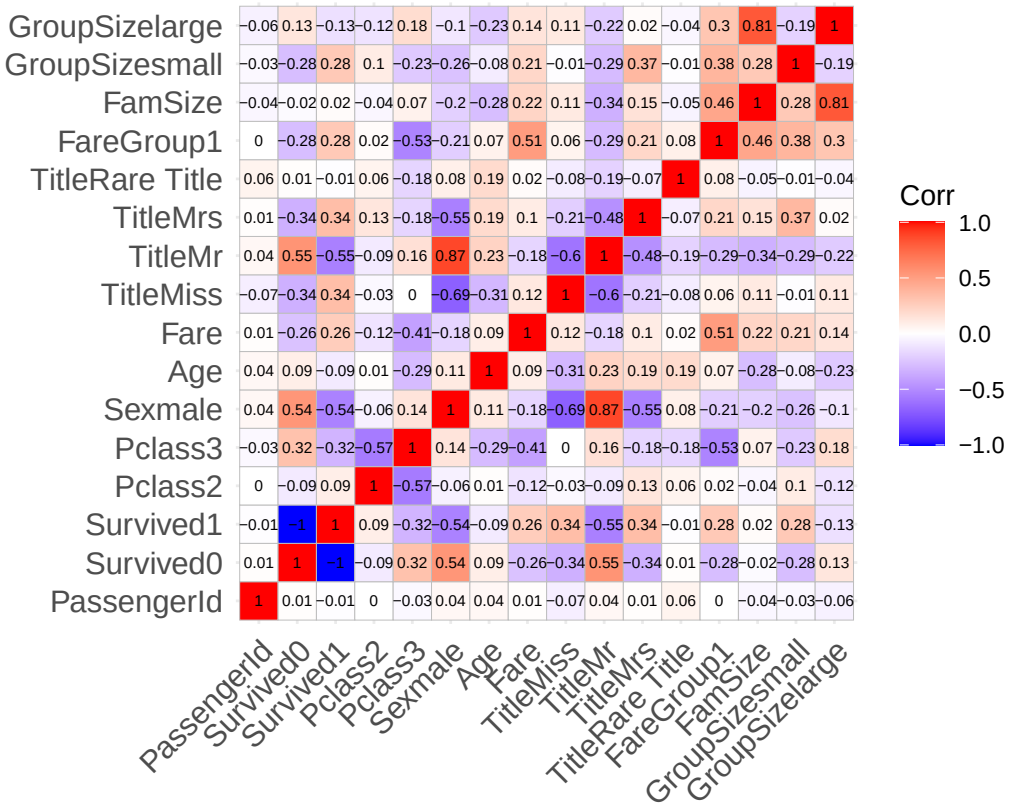

Рис. 2.10. Обновленная матрица корреляций

Из представленной выше матрицы получаем, что новый признак GroupSize оказывает примерно такое же влияние на выживаемость, что и признак Fare, но при этом первый в два раза меньше зависит от Pclass чем второй.

Также стоит отказаться от признака Title, так как он слишком сильно коррелирует с переменными Age и Sex, которые мы уже используем, что безусловно приведет к переобучению модели.

2.8. Прогнозирование

2.8.1. Разделение данных на обучающий и тестовый наборы

Для будущей оценки модели разобъем наши данные на обучающий и тестовый наборы в соотношении 8 к 2.

```
set.seed(2021)
train <- full[!is.na(full$Survived),]
split <- createDataPartition(train$Survived, p = 0.8, list = FALSE)
```

```
train_clean <- slice(train, split)
test_clean <- slice(train, -split)
```

2.8.2. Построение моделей

В качестве основных предикторов возьмем такие переменные, как Pclass, Sex, Age и GroupSize.

В дальнейшем для решения нашей задачи попробуем три разные модели — метод случайного леса, метод опорных векторов и метод k ближайших соседей.

Для получения более точных оценок используем пятикратную перекрестную проверку с разбиением выборки на 7 частей.

```
set.seed(2021)
trCtrl <- trainControl(method="repeatedcv", number=7, repeats = 5)
```

Модель Random Forest

```
set.seed(2021)
rf.grid <- data.frame(.mtry = c(2:4))

rf_model.1 <- train(Survived ~ Pclass+Sex+Age+GroupSize,
                    data=train_clean,
                    method='rf',
                    tuneGrid = rf.grid,
                    trControl=trCtrl)

kable(rf_model.1$results)
```

mtry	Accuracy	Kappa	AccuracySD	KappaSD
2	0.8307805	0.6309150	0.0405537	0.0912862
3	0.8288114	0.6268893	0.0403106	0.0907257
4	0.8265785	0.6249744	0.0370430	0.0823853

Таблица 2.5. Таблица результатов модели RF

```
set.seed(2021)
rf_model.2 <- train(Survived ~ Pclass+Sex+Age+GroupSize+Fare+Embarked,
                    data=train_clean,
                    method='rf',
                    tuneGrid = rf.grid,
                    trControl=trCtrl)

kable(rf_model.2$results)
```

mtry	Accuracy	Kappa	AccuracySD	KappaSD
2	0.8332849	0.6305789	0.0379093	0.0879545
3	0.8388737	0.6480765	0.0353362	0.0789968
4	0.8355259	0.6439369	0.0366335	0.0812133

Таблица 2.6. Таблица результатов модели RF+Fare+Embarked

Модель Support Vector Machine (SVM)

```
set.seed(2021)
svm_model.1 <- train(Survived ~ Pclass+Sex+Age+GroupSize,
                    data=train_clean, method='svmRadial',
                    tuneLength = 9,
                    preProcess = c("center", "scale"),
                    trControl=trCtrl)

kable(svm_model.1$results)
```

sigma	C	Accuracy	Kappa	AccuracySD	KappaSD
0.1471344	0.25	0.8243294	0.6192902	0.0353922	0.0782596
0.1471344	0.50	0.8307667	0.6303436	0.0416661	0.0929389
0.1471344	1.00	0.8321756	0.6319302	0.0393653	0.0883978
0.1471344	2.00	0.8355398	0.6389600	0.0405728	0.0917869
0.1471344	4.00	0.8338646	0.6348999	0.0424498	0.0956950
0.1471344	8.00	0.8324612	0.6315456	0.0417489	0.0941456
0.1471344	16.00	0.8307833	0.6280578	0.0425379	0.0959177
0.1471344	32.00	0.8296601	0.6255594	0.0429893	0.0970532
0.1471344	64.00	0.8282594	0.6225730	0.0414776	0.0937021

Таблица 2.7. Таблица результатов модели SVM

```
set.seed(2021)
svm_model.2 <- train(Survived ~ Pclass+Sex+Age+GroupSize+Fare+Embarked,
                     data=train_clean, method='svmRadial',
                     tuneLength = 9,
                     preProcess = c("center", "scale"),
                     trControl=trCtrl)

kable(svm_model.2$results)
```

sigma	C	Accuracy	Kappa	AccuracySD	KappaSD
0.1166408	0.25	0.8192653	0.6060606	0.0388261	0.0878932
0.1166408	0.50	0.8287949	0.6239966	0.0392420	0.0887987
0.1166408	1.00	0.8315850	0.6291348	0.0401193	0.0916481
0.1166408	2.00	0.8287811	0.6221141	0.0398880	0.0917447
0.1166408	4.00	0.8296241	0.6242148	0.0387254	0.0887044
0.1166408	8.00	0.8265290	0.6180455	0.0414248	0.0937391
0.1166408	16.00	0.8206684	0.6039713	0.0415517	0.0943872
0.1166408	32.00	0.8145112	0.5922043	0.0421320	0.0943251
0.1166408	64.00	0.8061295	0.5747233	0.0425681	0.0947615

Таблица 2.8. Таблица результатов модели SVM+Fare+Embarked

Модель k ближайших соседей (KNN)

```
set.seed(2021)
knn_model.1 <- train(Survived ~ Pclass+Sex+Age+GroupSize,
                     data=train_clean, method='knn',
                     trControl=trCtrl)

kable(knn_model.1$results)
```

k	Accuracy	Kappa	AccuracySD	KappaSD
5	0.7554117	0.4636945	0.0342209	0.0767826
7	0.7607173	0.4767982	0.0385017	0.0833237
9	0.7542910	0.4597192	0.0355642	0.0770317

Таблица 2.9. Таблица результатов модели KNN

```
set.seed(2021)
knn_model.2 <- train(Survived ~ Pclass+GroupSize+Title,
                     data=train_clean, method='knn',
                     trControl=trCtrl)

kable(knn_model.2$results)
```

k	Accuracy	Kappa	AccuracySD	KappaSD
5	0.8251726	0.6186149	0.0365926	0.0822373
7	0.8167608	0.5986232	0.0354602	0.0812834
9	0.8125590	0.5886667	0.0349097	0.0799286

Таблица 2.10. Таблица результатов модели KNN+Title-Sex-Age

2.8.3. Оценка моделей

Для всех моделей проведём оценку при помощи пересечения предсказанных на тестовой выборке и реальных значений целевого признака.

```
models <- list(rf_model.1, rf_model.2, svm_model.1, svm_model.2, knn_model.1,
knn_model.2)
models_name <- c("RandomForest", "RF+Fare+Embarked", "SVM",
"SVM+Fare+Embarked", "KNN", "KNN+Title-Sex-Age")

models_metrics <-
data.frame(No=numeric(),Yes=numeric(),obs=character(),Group=character())
accuracy <- numeric(length(models))

for (i in 1:length(models)) {
  prediction <- predict(models[i], test_clean, type="prob")
  model_prob <- data.frame(prediction, test_clean$Survived,
  rep(models_name[i],length(test_clean$Survived)))
  colnames(model_prob) <- c("No","Yes","obs","Group")
  models_metrics <- rbind(models_metrics,model_prob)
}

results <- evalm(models_metrics,showplots=F, fsize=11, plots="r", silent=T)
kable(res)
```

В качестве критериев оценок каждой модели будем рассматривать Accuracy и AUC-ROC. Ключевой же метрикой будет AUC-ROC — площадь графика под кривой ошибок.

Model	Accuracy	AUC-ROC
RandomForest	0.79	0.89
RF+Fare+Embarked	0.80	0.89
SVM	0.79	0.85
SVM+Fare+Embarked	0.82	0.84
KNN	0.74	0.80
KNN+Title-Sex-Age	0.79	0.88

Таблица 2.11. Оценки моделей

Лучший результат на тестовой выборке по основному критерию AUC-ROC показала модель RF с дополнительно учтенными признаками Fare и Embarked. Несмотря на небольшую уступку в доле правильных ответов, модель, основанная на случайном лесе, намного лучше классифицировала пассажиров, чем модель SVM.

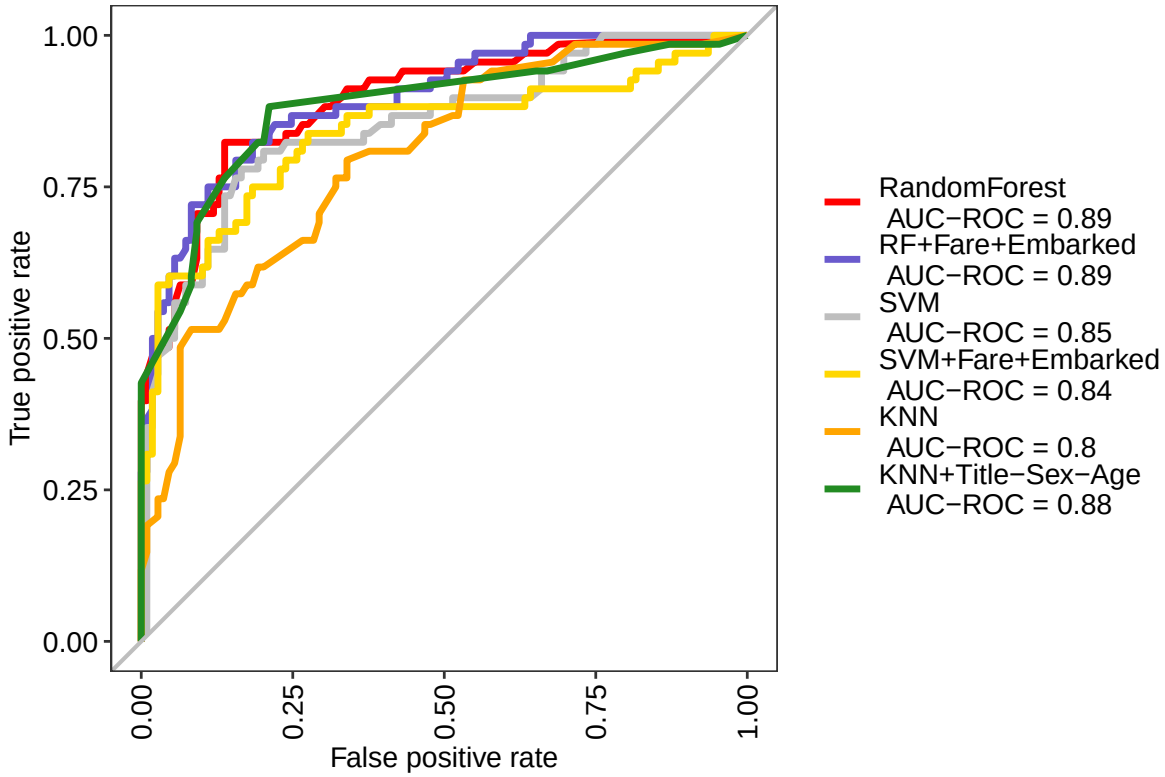

Рис. 2.11. Кривые ошибок моделей

2.8.4. Выгрузка результатов для Kaggle

По итогу оценивания лучшей моделью должна быть RF+Fare+Embarked.

После загрузки результатов применения моделей на Kaggle наилучший результат показала модель RF с дополнительными Fare и Embarked, получив оценку — 0.78229.

```
test <- full[is.na(full$Survived),]
y_predict <- predict(rf_model.2, test)
submission_df <- data.frame(PassengerId = test$PassengerId, Survived = y_predict)
write.csv(submission_df, file = "titanic_predictions.csv", row.names = F)
```

Заключение

Таким образом, в данной работе были получены следующие результаты:

- Исследованы три метода машинного обучения — метод случайного леса, метод опорных векторов и метод k ближайших соседей.
- Построено 6 моделей машинного обучения — 3 модели на основном наборе признаков и 3 модели с разными вариациями дополнительных признаков.
- Проведены оценка и сравнение результатов всех построенных моделей и выявлена лучшая из них. Модель случайного леса(RF), с дополнительно учтенными признаками Fare и Embarked, показала лучший результат на тестовой выборке по метрике AUC-ROC — 0,89.
- Достигнута публичная оценка на проверяющей выборке на платформе Kaggle — 78,22% доле правильных ответов, что входит в лучшие 15% результатов таблицы лидеров.

Несомненно, полученный результат можно улучшить многими способами. Путями улучшения результата могут быть:

- Большее углубление в данные, т.е. поиск и выделение новых признаков, которые потенциально могут принести новую полезную информацию и при этом не переобучить модель (например, исключенный признак Cabin, возможно, может быть полезен).
- Применение других моделей машинного обучения: логистические регрессии, деревья с градиентным усилением, XGBoost и т.д.
- Комбинирование разных моделей для разных подвыборок, т.е. разбиение обучающей выборки на подвыборки с последующим применением подходящей модели к соответствующей подвыборке.
- Подбор оптимальных гиперпараметров моделей.

Список литературы

1. Kaggle. Titanic — Machine Learning from Disaster. — https://www.kaggle.com/competitions/titanic.
2. Aurelien G. Hands-On Machine Learning with Scikit-Learn and TensorFlow. — O'Reilly Media, Inc., 2017. — 668 p. — ISBN: 9781491962299.
3. Morozov G. Titanic on Kaggle. — https://habr.com/ru/company/mlclass/blog/270973.
4. Takeshita O. Divide and Conquer. — https://www.kaggle.com/code/pliptor/divide-and-conquer-0-82296.
5. Risdal M. Exploring Survival on the Titanic. — https://www.kaggle.com/code/mrisdal/exploring-survival-on-the-titanic.