

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА - Российский технологический университет»**

# РТУ МИРЭА

**Институт кибернетики**

**Кафедра высшей математики**

**ОТЧЁТ ПО** Научно-Исследовательской Работе

(указать вид практики)

**Тема практики:** Построить модель прогнозирования отмены брони в отелях «Hotel booking demand» (kaggle.com)

приказ университета о направлении на практику

758 – С от 10.02.2020 г.

|  |  |  |
| --- | --- | --- |
| Отчет представлен к рассмотрению: |  |  |
| Студентка группы КМБО-04-19 | \_\_\_\_\_\_\_\_\_\_\_\_\_ | Аполлонов И.А.  (расшифровка подписи) «\_\_» \_\_\_\_\_\_ 2020г. |
|  |  |  |
| Отчет утвержден. |  |  |
| Допущена к защите: |  |  |
|  |  |  |
| Руководитель практики от кафедры | \_\_\_\_\_\_\_\_\_\_\_\_\_ | Петрусевич Д.А.  (расшифровка подписи) «\_\_» \_\_\_\_\_\_ 2020г. |

Москва 2020



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА - Российский технологический университет»**

# РТУ МИРЭА

**ЗАДАНИЕ НА** Научно-Исследовательскую Работу

**Студенту 1 курса учебной группы КМБО-04-19 института кибернетики Аполлонову Ивану Анатольевичу**

(фамилия, имя и отчество)

**Место и время практики:** Институт кибернетики, кафедра высшей математики

**Время практики:** с «10» февраля 2020 по «30» мая 2020

**Должность на практике: практикант**

1. **ЦЕЛЕВАЯ УСТАНОВКА: изучение основ анализа данных и машинного обучения**

1. **СОДЕРЖАНИЕ ПРАКТИКИ:** 
   1. Изучить: литературу и практические примеры по темам: 1) построение линейной регрессии, 2) использование метода главных компонент, 3) поиск и устранение линейной зависимости в данных, 4) основы нормализации данных, 5) методы классификации и кластеризации («решающее дерево»,

«случайный лес», «k ближайших соседей»).

* 1. Практически выполнить: 1) снижение размерности исходных задач при помощи метода главных компонент при возможности; построение линейной регрессии для некоторого параметра, исключение регрессоров, не коррелирующих с объясняемой переменной; решение задачи классификации или кластеризации на основе открытого набора данных с ресурса kaggle.com 2.3 Ознакомиться: с применением метода главных компонент; методов классификации («решающего дерева», «случайного леса»); методов кластеризации («k ближайших соседей»).

**3.ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ:** построить модель прогнозирования отмены брони в отелях «Hotel booking demand» (kaggle.com).

**4. ОГРАНИЗАЦИОННО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ:** применить алгоритмы кластеризации. Что общего между объектами в каждом кластере? Выделить аномальные объекты, которые далеко отстоят от кластеров. Построить модель прогнозирования отмены брони.

Заведующий кафедрой

высшей математики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ю.И.Худак

«10» февраля 2020г.

СОГЛАСОВАНО

Руководитель практики от кафедры:

«10» февраля 2020 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Петрусевич Д.А.)

(подпись) (фамилия и инициалы)

Задание получил:

«10»февраля 2020 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Аполлонов И.А.)

(подпись) (фамилия и инициалы)

**ИНСТРУКТАЖ ПРОВЕДЕН:**

|  |  |  |
| --- | --- | --- |
| **Вид мероприятия** | **ФИО ответственного, подпись, дата** | **ФИО студентки, подпись, дата** |
| Охрана труда | Петрусевич Д.А.    \_\_\_\_\_\_\_\_\_\_  «10» февраля 2020 г. | Аполлонов И.А.    \_\_\_\_\_\_\_\_\_\_  «10» февраля 2020 г. |
| Техника безопасности | Петрусевич Д.А.    \_\_\_\_\_\_\_\_\_\_  «10» февраля 2020 г. | Аполлонов И.А.    \_\_\_\_\_\_\_\_\_\_  «10» февраля 2020 г. |
| Пожарная безопасность | Петрусевич Д.А.    \_\_\_\_\_\_\_\_\_\_  «10» февраля 2020 г. | Аполлонов И.А.    \_\_\_\_\_\_\_\_\_\_  «10» февраля 2020 г. |
| Правила внутреннего распорядка | Петрусевич Д.А.    \_\_\_\_\_\_\_\_\_\_  «10» февраля 2020 г. | Аполлонов И.А.    \_\_\_\_\_\_\_\_\_\_  «10» февраля 2020 г. |



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА - Российский технологический университет»**

# РТУ МИРЭА

**РАБОЧИЙ ГРАФИК ПРОВЕДЕНИЯ** Научно-Исследовательской

Работы

студента Аполлонова И.А. 1 курса группы КМБО-04-19 очной формы обучения, обучающегося по направлению подготовки 01.03.02

«Прикладная математика и информатика», профиль «Математическое моделирование и вычислительная математика»

|  |  |  |  |
| --- | --- | --- | --- |
| Неделя | Сроки выполнения | Этап | Отметка о выполнении |
| 1 | 10.02.2020 | Выбор темы НИР. Пройти инструктаж по технике безопасности. |  |
| 1 | 10.02.2020 | Вводная установочная лекция. |  |
| 3 | 29.02.2020 | Построение и оценка линейной регрессии с помощью языка R |  |
| 5 | 14.03.2020 | Использование метода главных компонент,  выделение линейной зависимости в данных |  |
| 7 | 14.03.2020 | Методы классификации и кластеризации;  построение решающего  дерева |  |
| 9 | 28.03.2020 | Концепция бэггинга,  «случайный лес»; концепция бустинга; градиентные методы обучения и кластеризации |  |
| 17 | 05.06.2020 | Представление отчётных |  |
|  |  | материалов по НИР и их защита. Передача  обобщённых материалов  на кафедру для архивного хранения. |  |
|  |  | Зачётная аттестация. |  |

Содержание практики и планируемые результаты согласованы с руководителем практики от профильной организации.

**Согласовано:**

Заведующий \_\_\_\_\_\_\_\_\_\_\_\_\_\_ / ФИО / Худак Ю.И. кафедрой

Руководитель практики от кафедры \_\_\_\_\_\_\_\_\_\_\_\_\_\_ / ФИО / Петрусевич Д.А.

Обучающаяся \_\_\_\_\_\_\_\_\_\_\_\_\_\_ / ФИО / Аполлонов И.А.

**Оглавление**

Задача 1………………………………………………………………………..3

Задача 2………………………………………………………………………..7

Задача 3………………………………………………………………………..9

Индивидуальное задание (задача 4)...............................................................15

Заключение……………………………………………………………….......17

Список литературы…………………………………………………………..18

Приложения…………………………………………………………………..19

**Задача 1**

*Условие*

Необходимо загрузить данные из указанного набора и произвести следующие действия.

1. Проверить, что в наборе данных нет линейной зависимости (построить

зависимости между переменными, указанными в варианте, и проверить, что в каждой из них не высокий). В случае, если большой, один из таких столбцов можно исключить из рассмотрения.

2. Построить линейную модель зависимой переменной от указанных в варианте регрессоров по методу наименьших квадратов (команда lm пакета lmtest в языке R). Оценить, насколько хороша модель, согласно: 1) ,

2) p-значениям каждого коэффициента.

3. Ввести в модель логарифмы регрессоров (если возможно). Сравнить модели и выбрать наилучшую.

*Решение задачи*

Имеется набор данных mtcars.

Регрессоры: disp, hp, drat

С помощью данных переменных мы описываем переменную mpg.

*Ход решения:*

1. Для начала читаем данные из mtcars и нормализуем их. Для этого вычитаем из каждого рассматриваемого столбца его среднее значение (mean(x)) и делим на его среднеквадратичное отклонение (sqrt(var(x))).

2. Проверяем, являются ли регрессоры линейно зависимыми друг от друга

с помощью команды lm() строим между ними попарно зависимости. Построив модели model1, model2, model3 и посмотрев на их скорректированный коэффициент детерминации (далее Adjusted R-Squared), мы можем заметить, что Adjusted R-Squared во всех трёх моделях достаточно низок:

Таблица 1.1 Параметры моделей, связывающих регрессоры с объясняемой переменной

|  |  |  |  |
| --- | --- | --- | --- |
| Номер модели | Multiple R-squared | Adjusted R-squared | p-value |
| model1 | 0.7836 | 0.7687 | 2.291e-10 |
| model2 | 0.5385 | 0.5067 | 1.351e-05 |
| model3 | 0.6514 | 0.6273 | 2.315e-07 |

Коэффициент детерминации во всех построенных моделях низкий, из чего следует, что линейной зависимости между регрессорами не существует.

3. Строим модели для выявления лучшей комбинации регрессоров. Оцениваем модель по параметру Adjusted R-Squared, чем больше значение - тем лучше модель описывает нужную нам переменную mpg.

Таблица 2.1 Параметры комбинаций регрессоров

|  |  |  |  |
| --- | --- | --- | --- |
| Номер модели | Multiple R-squared | Adjusted R-squared | p-value |
| model4 | 0.775 | 0.7509 | 3.28e-09 |
| model5 | 0.7808 | 0.7573 | 2.291e-09 |
| model6 | 0.7495 | 0.7227 | 1.452e-08 |
| model7 | 0.7472 | 0.7201 | 1.649e-08 |
| model8 | 0.8053 | 0.7764 | 3.03e-09 |
| model9 | 0.8047 | 0.7757 | 3.161e-09 |
| model10 | 0.7857 | 0.754 | 1.078e-08 |
| model11 | 0.82 | 0.7934 | 1.063e-09 |
| model12 | 0.8198 | 0.8005 | 1.499e-10 |
| model13 | 0.7914 | 0.7605 | 7.566e-09 |
| model14 | 0.4678 | 0.3226 | 0.06506 |
| model15 | 0.3649 | 0.1917 | 0.1575 |
| model16 | 0.4617 | 0.3148 | 0.06894 |
| model17 | 0.7245 | 0.6654 | 0.0003301 |
| model18 | 0.2961 | 0.1041 | 0.2586 |
| model19 | 0.645 | 0.569 | 0.001852 |

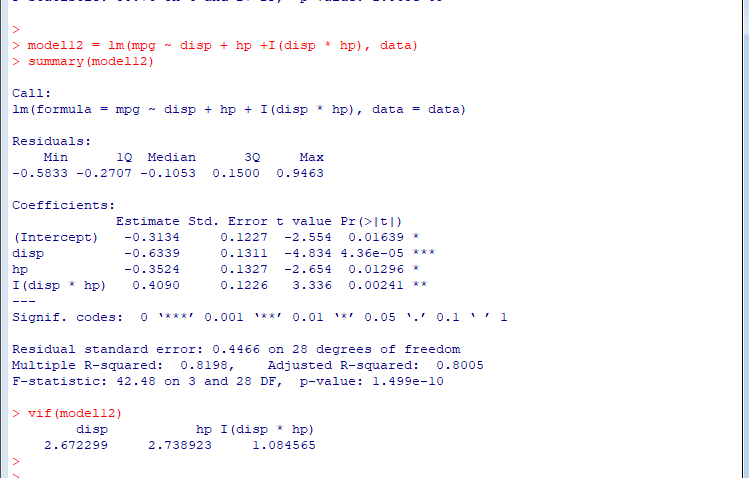
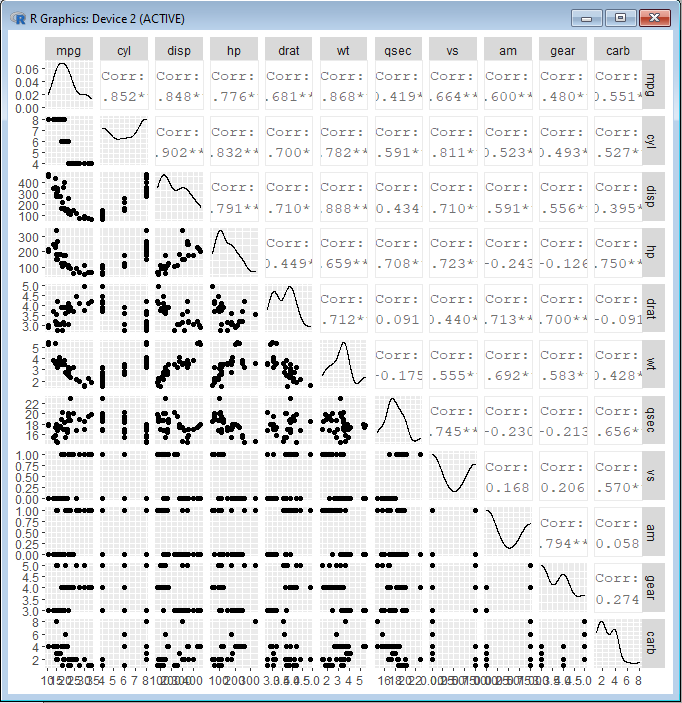


Рис. 1.1 Скриншот результата model12

*Вывод:*

Лучше всего mpg описывает model12.

**Задача 2**

*Условие.*

В этой задаче необходимо проанализировать данные волны мониторинга экономического положения и здоровья населения РФ (данные обследования РМЭЗ НИУ ВШЭ).

Минимальный набор параметров для анализа: зарплата, пол, семейное положение, наличие высшего образования, возраст, тип населённого пункта, длительность рабочей недели. Из параметра, отвечающего семейному положению, сделать дамми-переменные (с помощью one-hot-encoding):

1) переменная wed1 имеет значение 1 в случае, если респондент женат, 0 – в противном случае;

2) wed2=1, если респондент разведён или вдовец;

3) wed3=1, если респондент никогда не состоял в браке;

4) если считаете необходимым, введите другие параметры. Следите за мультиколлинеарностью (убедитесь в её отсутствии, оценив вспомогательную регрессию любого параметра (например, зарплаты или одного из параметров wed) на эти переменные и использовав команду vif для неё). Из параметра пол сделайте переменную sex, имеющую значение 1 для мужчин и равную 0 для женщин. Из параметра, отвечающего типу населённого пункта, создайте одну дамми-переменную city\_status со значением 1 для города или областного центра, 0 – в противоположном случае. Введите один параметр higher\_educ, характеризующий наличие полного высшего образования. Факторные переменные, «имеющие много значений», такие как: зарплата, длительность рабочей недели и возраст, необходимо преобразовать в вещественные переменные и нормализовать их: вычесть среднее значение по этой переменной, разделить её значения на стандартное отклонение.

1. Постройте линейную регрессию зарплаты на все параметры, которые Вы выделили из данных мониторинга. Не забудьте оценить коэффициент вздутия дисперсии VIF.

2.Поэкспериментируйте с функциями вещественных параметров: используйте логарифм и степени (хотя бы от 0.1 до 2 с шагом 0.1).

3. Выделите наилучшие модели из построенных: по значимости параметров, включённых в зависимости, и по объяснённому с помощью построенных зависимостей разбросу adjusted R2 - R 2 adj.

4. Сделайте вывод о том, какие индивиды получают наибольшую зарплату.

5. Оцените регрессии для подмножества индивидов, указанных в варианте.

*Решение*

Были использованы данные r22i\_os26c.sav

Для анализа были взяты следующие переменные:

rj10 – зарплата;

rh5 – пол;

r\_marst – семейное положение;

r\_diplom – наличие высшего образования

r\_age – возраст;

status – тип населённого пункта;

rj6.2 – длинна рабочей недели

Строим простую регрессию и смотрим, есть ли линейная зависимость между регрессорами: R-squared: 0.09347, Adjusted R-squared: 0.09242, линейной зависимости нет.

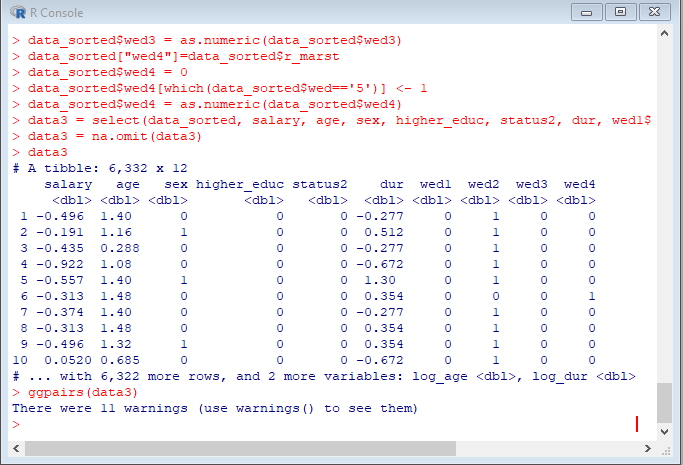
Нормализуем переменные :

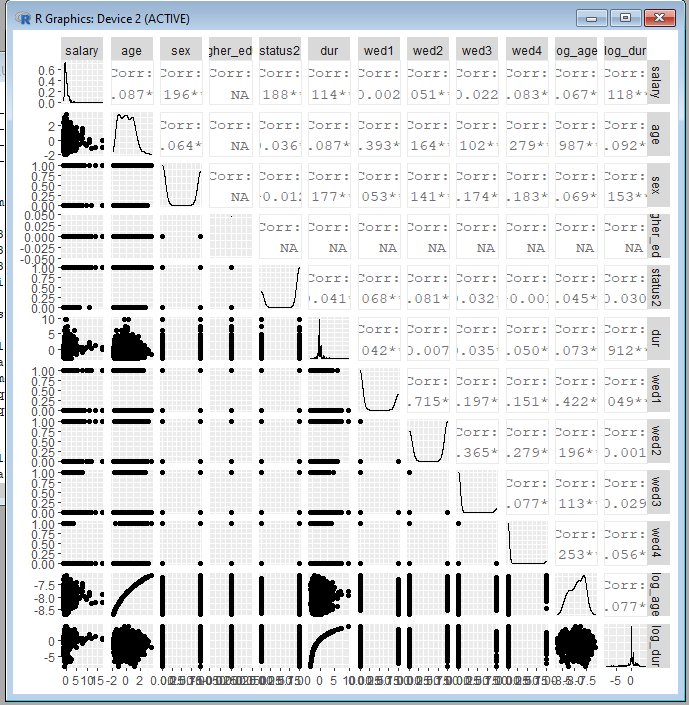
Таблица 3.1 Соответствие старых переменных новым

|  |  |  |
| --- | --- | --- |
| Старый параметр | Новый параметр | Логарифм |
| rj10 | salary | - |
| rh5 | sex | - |
| r\_marst | marital status | - |
| r\_diplom | higher education | - |
| r\_age | age | log\_age |
| status | type of settlement | - |
| rj6.2 | working week duration | log\_dur |

Применим метод one-hot-encoding, который заключается в представлении параметра в виде набора столбцов значений параметра в виде единиц и нулей в столбцах, соответствующих категории объекта и нет соответственно. Так же применим метод label-encoding, суть которого заключается в представлении параметра в виде высказывания, имеющего значения 1 и 0, если высказывание верно или ложно соответственно. Создаем столбец sex присваиваем ему значения, соответствующие полу респондента: для мужчин – 0, для женщин – 1. Добавляем новые столбцы wed1, wed2, wed3. Переменная wed1 имеет значение 1 в случае, если респондент женат, 0 – в противном случае; wed2=1, если респондент разведен/а или вдовец/ва; wed3 = 1, если респондент никогда не состоял в браке. Добавим столбец h\_educ, в который будет записываться 1, если у респондента есть законченное высшее образование и 0 в ином случае.

После этих операций создаём набор data3, включающий только выбранные нами параметры.





Нормализовав данные, мы строим разные модели, экспериментируя со степенями вещественных переменных, с их логарифмами и произведениям, оценивая p-value параметров. У всех моделей очень низкая степень соответствия. Находим модель с наилучшим результатом, это model5.

*Вывод:*

Посмотрев на столбец Estimate в выводе функции summary, можно указать, что model5 задается формулой:

salary ~ -1.16901\*age + 0.41447\*sex + 0.49226\*status2 + 0.59972\*log\_dur -0.30353\*dur + 3.43939\*log\_age

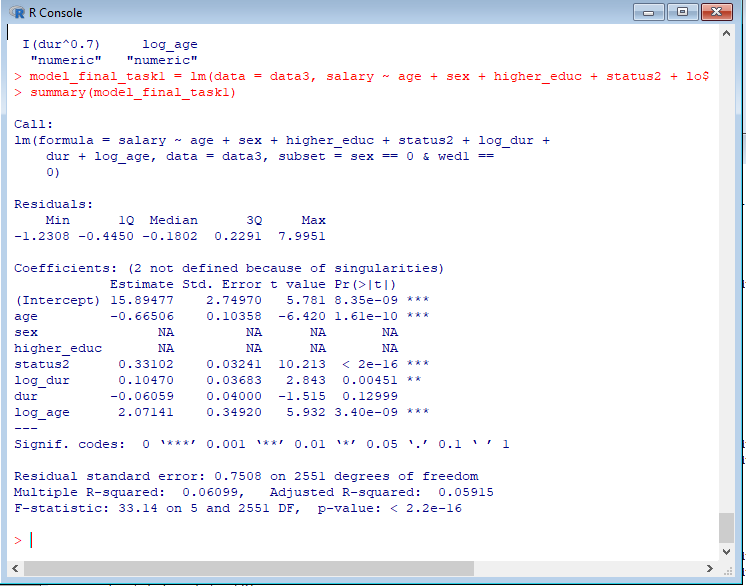
Таким образом, самую высокую зарплату имеет мужчина средних лет, с высшим образованием, живущий в крупном городе и работающий дольше остальных.

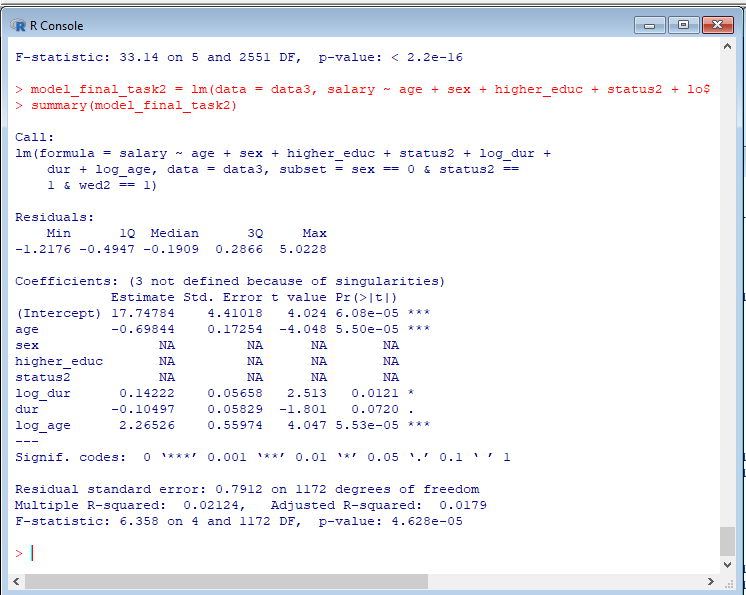
Теперь оценим модель для двух подмножеств:

1. Женщины, не замужем (task1);
2. Женщины, живущие в городе, разведенные (task2)

Таблица 4.1 Оценка регрессии для подмножеств

|  |  |
| --- | --- |
| task | Вывод |
| 1 | Наибольшую зарплату получают женщины средних лет, живущие в городе, которые перерабатывают |
| 2 | Наибольшую зарплату получают женщины средних лет, которые перерабатывают |





**Задача 3**

*Условие*

В данной задаче нужно было проанализировать датасет Hotel booking demand по предложенному алгоритму.

*Ход решения:*

Загружаем данные, проводим первичный анализ и выясняем, что таблица данных содержит 119390 строк (объекта) и 32 столбца (признака).

Описание признаков:

* hotel – название отеля
* is\_canceled – отмена брони bool
* lead\_time – время выполнения заказа
* arrival\_date\_year – год заселения
* arrival\_date\_month – месяц заселения
* arrival\_date\_week\_number – номер недели заселения
* arrival\_date\_day\_of\_month – день заселения
* stays\_in\_weekend\_nights – ночь в выходные
* stays\_in\_week\_nights – на ночь
* adults – взрослые
* children – дети
* babies – груднички
* meal – еда
* country – страна
* market\_segment – сегмент рынка
* distribution\_channel – канал перехода
* is\_repeated\_guest – повторное заселение bool
* previous\_cancellations – ранее были отмены bool
* previous\_bookings\_not\_canceled – ранее брони не отменялись bool
* reserved\_room\_type – тип номера
* assigned\_room\_type – присвоен номер
* booking\_changes – изменения в брони
* deposit\_type – тип депозита
* agent – агент
* company – компания
* days\_in\_waiting\_list – дни в списке ожидания
* customer\_type – тип клиента
* adr –
* required\_car\_parking\_spaces – парковочное место
* total\_of\_special\_requests – сумма спец. запросов
* reservation\_status – статус брони
* reservation\_status\_date – дата обновления статуса брони

Проверяем количество NaN значений в каждом столбце, их нет.

Далее проверяем коэффициенты корреляции между переменными.

Подготовим данные: следующим шагом через Label Encoding кодируем категориальный столбец is\_canceled в бинарный.

Выполняем нормализацию признаков по методу MinMax, заключительным шагом на этапе подготовки данных будет выделение целевой переменной, затем тренировочной и тестовой выборки.

Применяем к тренировочной выборке метод главных компонент (PCA) из которого следовало, что для описания 90% данных требуется 9 компонент.

*Вывод:*

Первично обработали данные, подготовили их к применению алгоритмов классификации и регрессии. С помощью алгоритма PCA мы поняли, что для описания нашей целевой переменной достаточно 9 переменных.

**Индивидуальное задание**

**Задача 4**

Условие

Классификация сердечно-сосудистых заболеваний на основе набора данных “ Hotel booking demand”

Решение

Выполняем загрузку данных и их первичную обработку, проверяем данные на количество NaN значений.

Проанализируем свойства данных, попробуем выявить общие закономерности или аномалии.

Заболевания оказались более распространены среди пациентов старше 55 лет.

Можно заметить, что у женщин чаще возникают проблемы с сердцем, чем у мужчин.

После нормализации данных можно приступить к построению моделей с целью улучшения прогнозов и проведения классификации объектов.

Построим модели для анализа наших данных, используя метод k-ближайших соседей и решающее дерево, из которых потом выберем более эффективную.

Для этого нам понадобится контрольная выборка, которая представляет собой часть данных, которая была исключена при построении модели и будет использоваться для её последующей оценки.

1. Метод k-ближайших соседей (K Nearest Neighbors)

Метод k-ближайших соседей представляет собой один из простых методов классификации элементов данных путём их соотнесения с ближайшими элементами.

k - число таких ближайших элементов для расчёта, которое определяется с помощью кросс-валидации.

Задаваемые параметры в модели:

X\_train, Y\_train, X\_test, Y\_test

k=7

1. Решающее дерево (Decision Tree)

Решающее дерево создаёт прогноз на основании серии бинарных вопросов. Набор данных последовательно разбивается на более однородные группы в ходе рекурсивного деления.

Задаваемые параметры в модели:

X\_train, Y\_train, X\_test, Y\_test

Используя метод перекрестной проверки, мы можем получить оценку того, насколько точной может быть каждая модель, чтобы выбрать лучшую из них.

Вывод: Достичь максимальной точности можно с помощью Decision Tree.

**Заключение.**

В данной работе был представлен анализ набора данных об отмене бронирования отелей Hotel booking demand с использованием методик, изученных в курсе по подготовке к НИР. Исследовательский анализ позволил нам лучше интерпретировать имеющиеся данные. Процедуры кластеризации, такие как K Nearest Neighbors были использованы для поиска групп между людьми с аналогичными характеристики. Метод классификации "решающее дерево" был использован как модель с точностью более 70%.

Каждый подход, использованный во время этой работы, помог нам расширить наши знания и дал возможность освоиться с языками программирования R и Python.

**Список литературы**

1. Набор данных “Hotel booking demand” (Kaggle.com)

URL: <https://www.kaggle.com/jessemostipak/hotel-booking-demand>

1. *Грас Дж*. Data Science. Наука о данных с нуля: Пер. с англ. — СПб.:БХВ-Петербург, 2019. — 336 с.: ил.

ISBN 978-5-9775-3758-2

1. *Силен Дэви, Мейсман Арно, Али Мохамед.* Основы Data Science и Big Data. Python и наука о данных. — СПб.: Питер, 2017. — 336с.: ил. — (Серия "Библиотека программиста").

ISBN 978-5-496-02517-1

**Приложения**

Код к задаче 1

#connect libraries

library ("IMTest")

library ("car")

library ("GGally")

library ("psych")

**data** = mtcars

**data**

summary(**data**)

ggpairs(**data**)

#normalize the data:

**data**$mpg = data$mpg - mean(data$mpg)

**data**$ mpg = data$ mpg /sqrt(var(data$mpg))

**data**$disp = data$disp - mean(data$disp)

**data**$disp = data$disp /sqrt(var(data$disp))

**data**$hp = data$hp - mean(data$hp)

**data**$hp = data$hp /sqrt(var(data$hp))

**data**$drat= data$drat - mean(data$drat)

**data**$drat = data$drat /sqrt(var(data$drat))

#construct a linear regression in pairs between the parameters to determine the linear independence

model1 = lm(disp ~ drat + hp, data)

model1

summary(model1)

model2 = lm(drat ~ disp + hp, data)

model2

summary(model2)

model3 = lm(hp ~ disp + drat, data)

model3

summary(model3)

#build a simple linear model of the dependence of mpg on all describing variables

model4 = lm(mpg ~ disp + hp + drat, data)

model4

summary(model4)

model5 = lm(mpg ~ disp + drat +I(disp \* drat), data)

summary(model5)

model6 = lm(mpg ~ drat + hp +I(hp\* disp),data)

summary(model6)

model7 = lm(mpg ~ disp + drat +I(drat ^2), data)

summary(model7)

model8 = lm(mpg ~ disp + drat + hp +I(disp ^2), data)

summary(model8)

model9 = lm(mpg ~ disp + drat + hp +I(hp^2), data)

summary(model9)

model10=lm(mpg~ disp + drat + hp +I(hp\* drat), data)

summary(model10)

model11 = lm(mpg ~ disp + drat + hp +I(hp\* disp), data)

summary(model11)

model12 = lm(mpg ~ disp + hp +I(disp \* hp), data)

summary(model12)

vif(model12)

model13 = lm(mpg ~ disp + drat + hp +I(disp \* drat), data)

summary(model13)

model14 = lm(mpg ~ disp + hp +log(disp), data)

summary(model14)

model15 = lm(mpg ~ disp + drat +log(disp), data)

summary(model15)

model16 = lm(mpg ~ disp + hp +log(hp), data)

summary(model16)

model17 = lm(mpg ~ disp + drat +log(drat), data)

summary(model17)

model18 = lm(mpg ~ hp + drat +log(hp), data)

summary(model18)

model19 = lm(mpg ~ hp + drat +log(drat), data)

summary(model19)

# the model12 best describes the mpg variable

**Код к задаче 2**

---

title: "R Notebook"

output: html\_notebook

---

Sys.setlocale(locale = "English")

library("lmtest")

library("rlms")

library("dplyr")

library("GGally")

library(car)

library(sandwich)

#choose regressors:

#oj10 - salary

#oh5 - sex

#o\_marst - marital status

#o\_diplom - higher education

#o\_age - age

#status - type of settlement

#oj6.2 - working week duration

data <- rlms\_read("C:/Users/annby/Downloads/r22i\_os26c.sav")

data\_sorted = select(data, rj10, rh5, status, r\_age, r\_marst, rj6.2, r\_diplom)

data\_sorted = na.omit(data\_sorted)

glimpse(data\_sorted)

#normalize salary

salary1 = as.character(data\_sorted$rj10)

salary2 = lapply(salary1, as.integer)

salary = as.numeric(unlist(salary2))

data\_sorted["salary"] = (salary - mean(salary))/sqrt(var(salary))

#normalize age

age1 = as.character(data\_sorted$r\_age)

age2 = lapply(age1, as.integer)

age3 = as.numeric(unlist(age2))

log\_age1 = log(age3)

data\_sorted["age"]= (age3 - mean(age3)) / sqrt(var(age3))

data\_sorted["log\_age"] = (log\_age1 - mean(log\_age1)/sqrt(var(log\_age1)))

data\_sorted["age"]

data\_sorted["log\_age"]

#normalize sex

data\_sorted["sex"]=data\_sorted$rh5

data\_sorted["sex"] = lapply(data\_sorted["sex"], as.character)

data\_sorted$sex[which(data\_sorted$sex!='1')] <- 0

data\_sorted$sex[which(data\_sorted$sex=='1')] <- 1

data\_sorted$sex = as.numeric(data\_sorted$sex)

#normalize higher education

#1 - with higher education (h\_educ=21,22,23)

#0 - all other

data\_sorted["h\_educ"] = data\_sorted$r\_diplom

data\_sorted["h\_educ"] = lapply(data\_sorted["h\_educ"], as.character)

data\_sorted["higher\_educ"] = data\_sorted$r\_diplom

data\_sorted["higher\_educ"] = 0

data\_sorted$higher\_educ[which(data\_sorted$h\_educ=='21')] <- 1

data\_sorted$higher\_educ[which(data\_sorted$h\_educ=='22')] <- 1

data\_sorted$higher\_educ[which(data\_sorted$h\_educ=='23')] <- 1

#normalize type of settlement

#1 - large (status=1; status=2)

#0 - all other

data\_sorted["status1"]=data\_sorted$status

data\_sorted["status1"] = lapply(data\_sorted["status1"], as.character)

data\_sorted["status2"] = 0

data\_sorted$status2[which(data\_sorted$status1=='1')] <- 1

data\_sorted$status2[which(data\_sorted$status1=='2')] <- 1

data\_sorted$status2 = as.numeric(data\_sorted$status2)

#normalize working week duration

dur1 = as.character(data\_sorted$rj6.2)

dur2 = lapply(dur1, as.integer)

dur3 = as.numeric(unlist(dur2))

data\_sorted["log\_dur"] = (log(dur3) - mean(log(dur3)))/sqrt(var(log(dur3)))

data\_sorted["dur"] = (dur3 - mean(dur3)) / sqrt(var(dur3))

#normalize marital status

#wed1 - never married

#wed2 - married

#wed3 - widower/widow

data\_sorted["wed"]= data\_sorted$r\_marst

data\_sorted["wed"] = lapply(data\_sorted["wed"], as.character)

data\_sorted$wed1 = 0

data\_sorted$wed1[which(data\_sorted$wed=='1')] <- 1

data\_sorted$wed1[which(data\_sorted$wed=='3')] <- 1

data\_sorted$wed1 = as.numeric(data\_sorted$wed1)

data\_sorted["wed2"] = lapply(data\_sorted["wed"], as.character)

data\_sorted$wed2 = 0

data\_sorted$wed2[which(data\_sorted$wed=='2')] <- 1

data\_sorted$wed2 = as.numeric(data\_sorted$wed2)

data\_sorted["wed3"]=data\_sorted$r\_marst

data\_sorted$wed3 = 0

data\_sorted$wed3[which(data\_sorted$wed=='4')] <- 1

data\_sorted$wed3 = as.numeric(data\_sorted$wed3)

data\_sorted["wed4"]=data\_sorted$r\_marst

data\_sorted$wed4 = 0

data\_sorted$wed4[which(data\_sorted$wed=='5')] <- 1

data\_sorted$wed4 = as.numeric(data\_sorted$wed4)

#form the final dataset, remove from it all respondents with incomplete data

data3 = select(data\_sorted, salary, age, sex, higher\_educ, status2, dur, wed1,wed2,wed3,wed4, log\_age, log\_dur)

data3 = na.omit(data3)

data3

ggpairs(data3)

#construct a simple regression and see if there are linear relationships between the regressors

model1 = lm(data = data3, salary~age + sex + higher\_educ + status2 + dur + wed1 + wed2 + wed3 + wed4)

summary(model1)

vif(model1)

#Multiple R-squared: 0.09028, Adjusted R-squared: 0.08913

model2 = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + dur + log\_dur)

summary(model2)

#Multiple R-squared: 0.08732, Adjusted R-squared: 0.0866

model3 = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + dur + log\_age)

summary(model3)

#Multiple R-squared: 0.1039, Adjusted R-squared: 0.1032

model4 = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + dur + I(dur^0.1) + log\_age)

summary(model4)

#Multiple R-squared: 0.1055, Adjusted R-squared: 0.103

model5 = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + dur + I(dur^0.7) + log\_age)

summary(model5)

vif(model5)

#Multiple R-squared: 0.1069, Adjusted R-squared: 0.1045

#the best model

model6 = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + dur + I(dur^1.2))

summary(model6)

#Multiple R-squared: 0.08589, Adjusted R-squared: 0.0838

model7 = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + dur + I(age^2) + wed1 + wed2 + wed3)

summary(model7)

#Multiple R-squared: 0.1017, Adjusted R-squared: 0.1006

model\_final = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + log\_dur + dur + log\_age)

summary(model\_final)

model5$terms

#model is inaccurate

#individual tasks

#task1

model\_final\_task1 = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + log\_dur + dur + log\_age, sex == 0 & wed1 == 0)

summary(model\_final\_task1)

#task2

model\_final\_task2 = lm(data = data3, salary ~ age + sex + higher\_educ + status2 + log\_dur + dur + log\_age, sex == 0 & status2 == 1 & wed2 == 1)

summary(model\_final\_task2)

**Код к задаче 3**

**#!/usr/bin/env python**

**# coding: utf-8**

**import numpy as np**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**import os**

**os.getcwd()**

**pd.read\_csv('hotel\_bookings.csv')**

**data = pd.read\_csv("hotel\_bookings.csv")**

**data = data [["hotel", "is\_canceled","lead\_time",**

**"arrival\_date\_year", "arrival\_date\_month",**

**"arrival\_date\_week\_number", "arrival\_date\_day\_of\_month",**

**"stays\_in\_weekend\_nights", "stays\_in\_week\_nights",**

**"adults", "children", "babies", "meal", "country",**

**"market\_segment", "distribution\_channel", "is\_repeated\_guest",**

**"previous\_cancellations", "previous\_bookings\_not\_canceled",**

**"reserved\_room\_type", "assigned\_room\_type", "booking\_changes",**

**"deposit\_type", "agent", "company", "days\_in\_waiting\_list",**

**"customer\_type", "adr", "required\_car\_parking\_spaces",**

**"total\_of\_special\_requests", "reservation\_status", "reservation\_status\_date"]]**

**data.shape()**

**data.head()**

**data.tail()**

**#hotel – название отеля**

**#is\_canceled – отмена брони bool**

**#lead\_time – время выполнения заказа**

**#arrival\_date\_year – год заселения**

**#arrival\_date\_month – месяц заселения**

**#arrival\_date\_week\_number – номер недели заселения**

**#arrival\_date\_day\_of\_month – день заселения**

**#stays\_in\_weekend\_nights – ночь в выходные**

**#stays\_in\_week\_nights – на ночь**

**#adults – взрослые**

**#children – дети**

**#babies – груднички**

**#meal – еда**

**#country – страна**

**#market\_segment – сегмент рынка**

**#distribution\_channel – канал перехода**

**#is\_repeated\_guest – повторное заселение bool**

**#previous\_cancellations – ранее были отмены bool**

**#previous\_bookings\_not\_canceled – ранее брони не отменялись bool**

**#reserved\_room\_type – тип номера**

**#assigned\_room\_type – присвоен номер**

**#booking\_changes – изменения в брони**

**#deposit\_type – тип депозита**

**#agent – агент**

**#company – компания**

**#days\_in\_waiting\_list – дни в списке ожидания**

**#customer\_type – тип клиента**

**#adr –**

**#required\_car\_parking\_spaces – парковочное место**

**#total\_of\_special\_requests – сумма спец. запросов**

**#reservation\_status – статус брони**

**#reservation\_status\_date – дата обновления статуса брони**

**data.info()**

**#Нет пропусков в данных**

**data.describe()**

**data.corr()**

**# ## Подготовка данных**

**# ### Обработка пустых значений**

**# \*\*Удаление строк с пропущенными значениями\*\***

**# Удалить строки с пропусками можно с помощью команды data.dropna(inplace=True). Inplace - позволяет изменить текущую таблицу, иначе новые данные будут возвращены этой функцией.**

**# Так как пропущенных значений мало мы просто удалим все строки с NaN значениями**

**data.dropna(inplace=True)**

**data.info()**

**# ### Обработка категориальных признаков**

**# \*\*Кодируем столбец из категориального признака (Label Encoding)\*\***

**from sklearn.preprocessing import LabelEncoder**

**label = LabelEncoder()**

**label.fit(data.is\_canceled)**

**data.is\_canceled = label.transform(data.is\_canceled)**

**data.is\_canceled.head()**

**# \*\*Нормализация признаков. Стандартное отклонение\*\***

**from sklearn.preprocessing import StandardScaler**

**scale\_features\_std = StandardScaler()**

**features = data**

**features[:] = scale\_features\_std.fit\_transform(data)**

**features**

**features.mean()**

**# \*\*Нормализация признаков. MinMax\*\***

**from sklearn.preprocessing import MinMaxScaler**

**scale\_features\_mm = MinMaxScaler()**

**data[:] = scale\_features\_mm.fit\_transform(data)**

**data**

**# \*\*'Выделяем целевую переменную\*\***

**target = data**

**train = data.drop(['is\_repeated\_guest', 'previous\_cancellations',], axis=1)**

**train.head()**

**# \*\*'Выделяем тренировочную и тестовую выборки\*\***

**# y - целевая переменнная (target)**

**from sklearn.model\_selection import train\_test\_split**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(train, target, test\_size = 0.3, random\_state = 42)**

**N\_train, \_ = X\_train.shape**

**N\_test, \_ = X\_test.shape**

**print (N\_train, N\_test)**

**# # Метод главных компонент (PCA)**

**from sklearn.decomposition import PCA**

**get\_ipython().run\_line\_magic('matplotlib', 'inline')**

**import matplotlib.pyplot as plt**

**pca = PCA()**

**pca.fit(X\_train)**

**X\_pca = pca.transform(X\_train)**

**for i, component in enumerate(pca.components\_):**

**print("{} component: {}% of initial variance".format(i + 1, round(100 \* pca.explained\_variance\_ratio\_[i], 2)))**

**print(" + ".join("%.3f x %s" % (value, name)**

**for value, name in zip(component,train.columns)))**

**plt.figure(figsize=(10,7))**

**plt.plot(np.cumsum(pca.explained\_variance\_ratio\_), color='k', lw=2)**

**plt.axhline(0.9, c='r')**

**plt.axvline(9, c='b')**

**# "Для описания 90% дисперсии данных достаточно 9-и компонент.**

**data.to\_csv(path\_or\_buf="h\_bookings.csv", index=False)**

**Код к задаче 4**

#!/usr/bin/env python

# coding: utf-8

import pandas as pd

import numpy as np

from fancyimpute import KNN

from scipy.stats import chi2\_contingency

import matplotlib.pyplot as plt

import seaborn as sns

import matplotlib.pyplot as plt

from random import randrange,uniform

from sklearn.model\_selection import train\_test\_split

from sklearn import tree

from sklearn.tree import export\_graphviz

from sklearn.metrics import accuracy\_score

from sklearn.metrics import confusion\_matrix

from sklearn.ensemble import RandomForestClassifier

import statsmodels.api as sn

from sklearn.neighbors import KNeighborsClassifier

from sklearn import model\_selection

fromsklearn.metrics import classification\_report,roc\_auc\_score,roc\_curve

from sklearn.metrics import classification\_report

import pickle

import statsmodels.api as sm

from statsmodels.stats.outliers\_influence import variance\_inflation\_factor

from statsmodels.tools.tools import add\_constant

np.random.seed(123)

pd.options.mode.chained\_assignment = None

data = pd.read\_csv('hotel\_bookings.csv')

data = data [["hotel", "is\_canceled","lead\_time",

"arrival\_date\_year", "arrival\_date\_month",

"arrival\_date\_week\_number", "arrival\_date\_day\_of\_month",

"stays\_in\_weekend\_nights", "stays\_in\_week\_nights",

"adults", "children", "babies", "meal", "country",

"market\_segment", "distribution\_channel", "is\_repeated\_guest",

"previous\_cancellations", "previous\_bookings\_not\_canceled",

"reserved\_room\_type", "assigned\_room\_type", "booking\_changes",

"deposit\_type", "agent", "company", "days\_in\_waiting\_list",

"customer\_type", "adr", "required\_car\_parking\_spaces",

"total\_of\_special\_requests", "reservation\_status", "reservation\_status\_date"]]

#hotel – название отеля

#is\_canceled – отмена брони bool

#lead\_time – время выполнения заказа

#arrival\_date\_year – год заселения

#arrival\_date\_month – месяц заселения

#arrival\_date\_week\_number – номер недели заселения

#arrival\_date\_day\_of\_month – день заселения

#stays\_in\_weekend\_nights – ночь в выходные

#stays\_in\_week\_nights – на ночь

#adults – взрослые

#children – дети

#babies – груднички

#meal – еда

#country – страна

#market\_segment – сегмент рынка

#distribution\_channel – канал перехода

#is\_repeated\_guest – повторное заселение bool

#previous\_cancellations – ранее были отмены bool

#previous\_bookings\_not\_canceled – ранее брони не отменялись bool

#reserved\_room\_type – тип номера

#assigned\_room\_type – присвоен номер

#booking\_changes – изменения в брони

#deposit\_type – тип депозита

#agent – агент

#company – компания

#days\_in\_waiting\_list – дни в списке ожидания

#customer\_type – тип клиента

#adr –

#required\_car\_parking\_spaces – парковочное место

#total\_of\_special\_requests – сумма спец. запросов

#reservation\_status – статус брони

#reservation\_status\_date – дата обновления статуса брони

data.head()

data.describe()

data.info()

type(data)

data.shape

y = data["is\_canceled"]

sns.countplot(y)

is\_canceled\_often = data.is\_canceled.value\_counts()

print(is\_canceled\_often)

sns.barplot(data["children"],data["is\_canceled"])

#При бронировании с детьми, отмены происходят реже

data.columns = ["hotel", "is\_canceled","lead\_time",

"arrival\_date\_year", "arrival\_date\_month",

"arrival\_date\_week\_number", "arrival\_date\_day\_of\_month",

"stays\_in\_weekend\_nights", "stays\_in\_week\_nights",

"adults", "children", "babies", "meal", "country",

"market\_segment", "distribution\_channel", "is\_repeated\_guest",

"previous\_cancellations", "previous\_bookings\_not\_canceled",

"reserved\_room\_type", "assigned\_room\_type", "booking\_changes",

"deposit\_type", "agent", "company", "days\_in\_waiting\_list",

"customer\_type", "adr", "required\_car\_parking\_spaces",

"total\_of\_special\_requests", "reservation\_status", "reservation\_status\_date"]

data['children'][data['children'] == 0] = 'without\_ch'

data['children'][data['children'] > 0] = 'with\_ch'

data['babies'][data['babies'] == 0] = 'without\_bab'

data['babies'][data['babies'] > 0] = 'with\_bab'

data['is\_repeated\_guest'][data['is\_repeated\_guest'] == 0] = 'non\_repeated\_guest'

data['is\_repeated\_guest'][data['is\_repeated\_guest'] == 1] = 'repeated\_guest'

data['previous\_cancellations'][data['previous\_cancellations'] == 0] = 'without\_cancellations'

data['previous\_cancellations'][data['previous\_cancellations'] == 1] = 'cancellations'

data['exercise\_induced\_angina'][data['exercise\_induced\_angina'] == 0] = 'no'

data['exercise\_induced\_angina'][data['exercise\_induced\_angina'] == 1] = 'yes'

data['customer\_type'][data['customer\_type'] == 'Contract'] = 'Contract'

data['customer\_type'][data['customer\_type'] == 'Group'] = 'Group'

data['customer\_type'][data['customer\_type'] == 'Transient'] = 'Transient'

data['customer\_type'][data['customer\_type'] == 'Transient-Party'] = 'Transient-Party'

data['meal'][data['meal'] == 'BB'] = 'meal\_BB'

data['meal'][data['meal'] == 'FB'] = 'meal\_FB'

data['meal'][data['meal'] == 'HB'] = 'meal\_HB'

data['meal'][data['meal'] == 'SC'] = 'meal\_SC'

data['meal'][data['meal'] == 'Undefined'] = 'meal\_Undefined'

data['booking\_changes'][data['booking\_changes'] == 0] = 'no\_booking\_changes'

data['booking\_changes'][10 >= data['booking\_changes'] >= 1] = 'booking\_changes\_before\_10'

data['booking\_changes'][data['booking\_changes'] > 11] = 'booking\_changes\_more\_10'

data['days\_in\_waiting\_list'][100 >= data['days\_in\_waiting\_list'] >= 0] = 'days\_in\_waiting\_list\_small'

data['days\_in\_waiting\_list'][data['days\_in\_waiting\_list'] > 100] = 'days\_in\_waiting\_list\_long'

#Процент с отменами и без отмен

list = []

for i in range(0,data.shape[1]):

if(data.iloc[:,i].dtypes == 'object'):

data.iloc[:,i] = pd.Categorical(data.iloc[:,i])

data.iloc[:,i] = data.iloc[:,i].cat.codes

data.iloc[:,i] = data.iloc[:,i].astype('object')

list.append(data.columns[i])

sns.countplot(x='is\_canceled',data=data,palette="bwr")

plt.show()

countWithout\_ch = len(data[data.children == 0])

countWith\_ch = len(data[data.children > 0])

print("Процент посетителей без детей:{:.2f}%".format((countWithout\_ch)/(len(data.children))\*100))

print("Процент посетителей с детьми:{:.2f}%".format((countWith\_ch)/(len(data.children))\*100))

countNoCanceled = len(data[data.is\_canceled == 0])

countHaveCanceled = len(data[data.is\_canceled == 1])

print("Процент без отмены брони: {:.2f}%%".format((countNoCanceled / (len(data.is\_canceled))\*100)))

print("Процент отмененных броней: {:.2f}%%".format((countHaveCanceled / (len(data.is\_canceled))\*100)))

data.groupby('is\_canceled').mean()

#Частота отмен брони при изменениях в брони

pd.crosstab(data.booking\_changes,data.is\_canceled).plot(kind="bar",figsize=(20,6))

plt.title('Частота отмен брони при изменениях в брони')

plt.xlabel('booking changes')

plt.ylabel('Frequency')

plt.show()

#Частота отмен с детьми и без

pd.crosstab(data.children,data.is\_canceled).plot(kind="bar",figsize=(15,6),color=['blue','#AA1111' ])

plt.title('Частота отмен с детьми и без')

plt.xlabel('children (0 = without, 1 = with)')

plt.xticks(rotation=0)

plt.legend(["Haven't Canceled", "Have Canceled"])

plt.ylabel('Frequency')

plt.show()

#Частота отмен с младенцами и без

pd.crosstab(data.babies,data.is\_canceled).plot(kind="bar",figsize=(15,6),color=['blue','#AA1111' ])

plt.title('Частота отмен с младенцами и без')

plt.xlabel('babies (0 = without, 1 = with)')

plt.xticks(rotation=0)

plt.legend(["Haven't Canceled", "Have Canceled"])

plt.ylabel('Frequency')

plt.show()

#Сегмент рынка и максимальная частота отмен брони

plt.scatter(x=data.market\_segment[data.is\_canceled==1], y=data.thalassemia[(data.is\_canceled==1)], c="green")

plt.scatter(x=data.market\_segment[data.is\_canceled==0], y=data.thalassemia[(data.is\_canceled==0)])

plt.legend(["Canceled", "Not Canceled"])

plt.xlabel("Marketing Segment")

plt.ylabel("Maximum Canceled")

plt.show()

#Канал перехода и максимальная частота отмен брони

plt.scatter(x=data.distribution\_channel[data.is\_canceled==1], y=data.thalassemia[(data.is\_canceled==1)], c="green")

plt.scatter(x=data.distribution\_channel[data.is\_canceled==0], y=data.thalassemia[(data.is\_canceled==0)])

plt.legend(["Canceled", "Not Canceled"])

plt.xlabel("Distribution Channel")

plt.ylabel("Maximum Canceled")

plt.show()

#Test Split

predictors = data.drop("is\_canceled",axis=1)

is\_canceled = data["is\_canceled"]

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(predictors,is\_canceled,test\_size=0.20,random\_state=0)

X\_train.shape

X\_test.shape

Y\_train.shape

Y\_test.shape

#Определение лучшей модели

#Метод k-ближайших соседей

#k=7

knn = KNeighborsClassifier(n\_neighbors=7)

knn.fit(X\_train,Y\_train)

Y\_pred\_knn=knn.predict(X\_test)

Y\_pred\_knn.shape

score\_knn\_7 = round(accuracy\_score(Y\_pred\_knn,Y\_test)\*100,2)

print("вычислить оценку точности: "+str(score\_knn)+" %")

#k=4

knn\_model=KNeighborsClassifier(n\_neighbors=4).fit(X\_train,Y\_train)

knn\_predictions=knn\_model.predict(X\_test)

CM=pd.crosstab(y\_test,knn\_predictions)

CM

k\_range = range(1, 26)

scores = []

from sklearn import metrics

for k in k\_range:

knn = KNeighborsClassifier(n\_neighbors=k)

knn.fit(X\_train, y\_train)

y\_pred = knn.predict(X\_test)

scores.append(metrics.accuracy\_score(y\_test, y\_pred))

print(scores)

TN=CM.iloc[0,0]

FP=CM.iloc[0,1]

FN=CM.iloc[1,0]

TP=CM.iloc[1,1]

#точность модели

score\_knn\_4=((TP+TN)\*100)/(TP+TN+FP+FN)

score\_knn\_4

#ложноотрицательный показатель модели

fnr=FN\*100/(FN+TP)

fnr

#Решающее дерево

#Строим модель на тренировочных данных и протестируем её на тестовых данных

C50\_model = tree.DecisionTreeClassifier(criterion='entropy').fit(X\_train, Y\_train)

C50\_Predictions = C50\_model.predict(X\_test)

CM = pd.crosstab(Y\_test, C50\_Predictions)

CM

TN=CM.iloc[0,0]

FP=CM.iloc[0,1]

FN=CM.iloc[1,0]

TP=CM.iloc[1,1]

#проверим точность модели

score\_dt=((TP+TN)\*100)/(TP+TN+FP+FN)

score\_dt

fnr=FN\*100/(FN+TP)

fnr

#Сравнение моделей

scores = [score\_lr, score\_knn]

algorithms = ["Decision Tree", "K-Nearest Neighbors"]

for i in range(len(algorithms)):

print("вычислить оценку точности:"+algorithms[i]+" is: "+str(scores[i])+" %")

sns.set(rc={'figure.figsize':(15,8)})

plt.xlabel("Алгоритм")

plt.ylabel("Точность")

sns.barplot(algorithms,scores)

#Достичь максимальной точности можно с помощью решающего дерева