Оглавление

[Практическая работа №1 3](#_Toc136001397)

[Условие 3](#_Toc136001398)

[Решение 4](#_Toc136001399)

[Вывод 5](#_Toc136001400)

[Практическая работа №2.1 7](#_Toc136001401)

[Условие 7](#_Toc136001402)

[Решение 8](#_Toc136001403)

[Вывод 9](#_Toc136001404)

[Практическая работа №2.2 10](#_Toc136001405)

[Условие 10](#_Toc136001406)

[Решение 11](#_Toc136001407)

[Вывод 12](#_Toc136001408)

[Практическая работа №3 13](#_Toc136001409)

[Условие 13](#_Toc136001410)

[Решение 14](#_Toc136001411)

[Вывод 19](#_Toc136001412)

[Практическая работа №4 20](#_Toc136001413)

[Условие 20](#_Toc136001414)

[Решение 21](#_Toc136001415)

[Вывод 22](#_Toc136001416)

[Практическая работа *№5* 24](#_Toc136001417)

[Условие 24](#_Toc136001418)

[Решение 25](#_Toc136001419)

[Заключение 37](#_Toc136001420)

[Список литературы 38](#_Toc136001421)

[Приложения 39](#_Toc136001422)

# Практическая работа №1

Необходимо загрузить данные из указанного набора и произвести следующие действия.

Набор данных: *Swiss*.

Объясняемая переменная: *Education*.

Регрессоры: *Fertility, Examination.*

## Условие

1. Оцените среднее значение, дисперсию и СКО переменных, указанных во втором и

третьем столбце.

2. Постройте зависимости вида у = a + bx, где у - объясняемая переменная, х -

регрессор (для каждого варианта по две зависимости).

3. Оцените, насколько «хороша» модель по коэффициенту детерминации R2?

4. Оцените, есть ли взаимосвязь между объясняемой переменной и объясняющей

переменной (по значению р-статистики, «количеству звездочек» у регрессора в

модели).

## Решение

Задание №1

Оценка среднего значения, дисперсии и СКО переменных использовалась функция mean(), которая выводит среднее значение по передаваемому ей списку элементов, а так же найдено СКО с помощью формулы: sqrt(sum((A - mean(A))^2/(length(A)-1))). Кроме того получена дисперсия благодаря функции var(). После выполнения анализа получим данные значения в таблице 1:

Таблица 1. Характеристики параметров *Examination, Agriculture, Fertility в наборе Swiss*

|  |  |  |  |
| --- | --- | --- | --- |
| Переменные | Среднее значение | Дисперсия | СКО |
| *Examination* | 16.48936 | 7.977883 | 63.64662 |
| *Agriculture* | 50.65957 | 22.71122 | 515.7994 |
| *Fertility* | 70.14255 | 12.4917 | 156.0425 |

Задание №2

Построение линейной зависимости используем команду lm():

lm(*Examination~Fertility+Agriculture, data*), в результате получаем общую характеристику для модели в таблице 1 ,а так же побочные для зависимостей *Examination ~ Fertility* в таблице 2 и *Examination ~ Agriculture* в таблице 3:

Таблица 2. Общая характеристика модели зависимости *Examination* от регрессоров *Agriculture* и *Fertility*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Параметр\ Характеристики** | **Значение** | **Std. Error** | **t value** | **Pr(>|t|)** | **Уровень значимости** |
| *(Intercept)* | 46.45927 | 4.01858 | 11.561 | 6.33e-15 | \*\*\* |
| *Fertility* | -0.29438 | 0.06024 | -4.887 | 1.40e-05 | \*\*\* |
| *Agriculture* | -0.18400 | 0.03313 | -5.553 | 1.52e-06 | \*\*\* |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Таблица 3. Характеристика зависимости *Examination* от регрессора *Fertility.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Параметр\ Характеристики** | **Значение** | **Std. Error** | **t value** | **Pr(>|t|)** | **Уровень значимости** |
| *(Intercept)* | 45.42289 | 5.17670 | 8.774 | 2.65e-11 | \*\*\* |
| *Fertility* | -0.41250 | 0.07268 | -5.675 | 9.45e-07 | \*\*\* |

Имеем зависимость cat(*data$Examination\_Fertility*<-45.42 + (-0.412)\**data$Fertility*) из таблицы 3.

Таблица 4 Характеристика для зависимости *Examination* от регрессора *Agriculture.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Параметр\ Характеристики** | **Значение** | **Std. Error** | **t value** | **Pr(>|t|)** | **Уровень значимости** |
| *(Intercept)* | 28.70668 | 2.11001 | 13.605 | < 2e-16 | \*\*\* |
| *Agriculture* | -0.24117 | 0.03807 | -6.334 | 9.95e-08 | \*\*\* |
|  |  |  |  |  |  |

Имеем зависимость cat(*data$Examination\_Agriculter*<-28.70 + (-0.241)\**data$Agriculture*) из таблицы 4.

Из взаимосвязи вида y= a + bx можно сделать вывод, с ростом коэффициента *Agriculture* и *Fertility* уменьшается % призывников, получивших наивысшую оценку на армейском экзамене, следовательно взаимосвязи являются отрицательными.

Задание №3

Рассмотрение того на сколько «хороша» модель по R2 нужно использовать команду *summary(lm(Examination~Fertility+Agriculture, data))*. Из этого следует, что 64% наших данных описывается нашей моделью. Можем сделать вывод (*R-squared* = 64%), что модель относительно хороша: для такой зависимости (только две объясняемых переменных) коэффициент не мал, но для полного описания процесса нужно добавлять другие параметры. Также при рассмотрении зависимостей *Examination~Fertility*, R2 равен 40% и *Examination~Agriculture,* R2 равен 45%, что говорит о неплохом качестве моделей.

Задание №4

Оценка взаимосвязи между объясняемой переменной и объясняющей переменной, воспользуемся функцией summary.

При рассмотрении таблицы 1 значений *p* для переменной-предиктора *Fertility* равно 1.40e-05(\*\*\*). Поскольку это значение меньше 0,05, оно имеет статистически значимую связь с переменной отклика в модели.

Значение *p* для переменной-предиктора *Agriculture* равно 1.52e-06(\*\*\*). Поскольку это значение меньше 0,05, оно имеет статистически значимую связь с переменной отклика в модели.

## Вывод

Анализ проведен на данных пакета *swiss*, который содержит информацию об уровне образования, занятости в сельском хозяйстве, рождаемости и других факторах в 47 регионах Швейцарии. В результате анализа были получены следующие выводы:

* Были построены графики зависимости показателя *Examination от Agriculture* и Fertility с помощью графической библиотеки ggplot2.
* Были построены модели зависимости *Examination от Agriculture и Fertility*. При этом коэффициент детерминации для модели с двумя объясняющими переменными составил 64%, что говорит о хорошей адаптации модели.
* Были проведены статистические тесты, которые показали, что обе переменные, *Agriculture и Fertility*, имеют статистически значимую связь с переменной отклика *Examination*.
* Из взаимосвязи вида y = a + bx был сделан вывод, что при увеличении значений *Agriculture и Fertility* показатель *Examination* уменьшается, что говорит о наличии отрицательной корреляции между этими переменными.

Таким образом, можно сделать вывод, что в данных *swiss* существует статистически значимая зависимость между показателем *Examination* и переменными *Agriculture* и *Fertility*. При этом увеличение значений *Agriculture и Fertility* сопровождается снижением показателя *Examination*. Однако для более точного предсказания значения *Examination* необходимо включать в модель другие переменные.

Код решения задачи и сведения о проверенных моделях приведены в Приложении 1

# Практическая работа №2.1

Набор данных: *Seatbelts*

Объясняемая переменная: *front*

Регрессоры: *law, kms, PetrolPrice*

## Условие

1. Проверьте, что в наборе данных нет линейной зависимости (построить зависимости между переменными, указанными в варианте, и проверить, что *R2* в каждой из них невысокий). В случае, если *R2* большой, один из таких столбцов можно исключить из рассмотрения.

2. Постройте 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) *R2*, 2) p-значениям каждого коэффициента.

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

4. Введите в модель всевозможные произведения пар регрессоров, в том числе квадраты регрессоров. Найдите одну или несколько наилучших моделей по доле объяснённого разброса в данных *R2*.

## Решение

Задание №1

Проверка наличия в наборе данных линейной зависимости. В случае, если R2 большой, один из таких столбцов можно исключить из рассмотрения.

В результате построения зависимостей имеем, что :

* *model1*=(lm(*law*~*kms* + *PetrolPrice*, *data*))R^2 = 28% относительно независимые регрессоры
* *model1*=(lm(*kms*~*law* + *PetrolPrice*, *data*))R^2 = 27% относительно независимые регрессоры
* *model1*=(lm(*PetrolPrice*~*law* + *kms*, *data*))R^2 = 19% относительно независимые регрессоры

Из списка делаем вывод, что все регрессоры относительно независимы и их необязательно исключать из дальнейшего рассмотрения.

Задание№2

Построение модели, используя команду lm(). Рассмотрим *R2* и p-статистику, с помощью summary().

* *model2 = lm(front~law + kms + PetrolPrice, data)*
* summary(*model2*)

Модель неплохая, но есть невзаимосвязанная переменная kms (с отсутствием звездочек). *Adjusted R-squared* = 42%. По результатам зависимости *front*~law и *front*~*PetrolPrice* имеет три звезды.

Получим коэф. R^2 для каждого отдельно:

* summary (lm (*front~law, data*))$r.squared, *R^2* равен 31%.
* Summary (lm (front~kms, data))$r.squared, *R^2* равен 12%.
* Summary (lm (front~PetrolPrice, data))$r.squared, *R^2* равен 29%.

В результате пробуем исключить *kms* как наименее значимый параметр:

* Summary (lm (*front~law + PetrolPrice, data*)) *Adjuster R-squared* = 43%.

Коэф. увеличился на 1% => исключение параметра не стало критичным. Следовательно, далее пользуемся моделью *front~law + PetrolPrice*, так как она имеет неплохой *R2*, и хорошие показатели p-статистики.

Задание№3

Введение в модель логарифмов регрессоров, не забывая про проверку vif(). Сравним модели и выберем наилучшую. Хочу заметить, что модель нельзя ввести log регрессоров без замены значений равных 0 в таблице *law* на 0.001(*data$law*[*data$law* == 0] <- 0.001).

Лучшей моделью в результате сравнений среди логарифмов оказалась:

* *model3\_1 = (lm(log(front)~ I(log(law)) + I(log(PetrolPrice)), data))*
* *model 3\_1* имеет коэффициент *R2 =* 0.4835, а так же неплохие значения vif.

Задание №4

Введение в модель всевозможных произведений пар регрессоров, в том числе квадратов регрессоров, не забывая о проверке значений vif() и найдем лучшую модель основываясь на доле объяснённого разброса в данных *R2*.

Лучшей моделью среди произведений пар регрессоров оказалась:

*model4\_1 = lm(front ~ law + I(PetrolPrice^2), data)*

*model4* имеет коэффициент R2 = 0.4306, а так же хорошие значения vif.

## Вывод

Из проведенного анализа данных следует, что зависимость между параметрами *"front", "law", "kms" и "PetrolPrice"* является линейной. Попытки использования всех регрессоров в одной модели показали, что они относительно независимы, что было подтверждено значениями VIF, но коэффициент детерминации (*R-squared*) был низким для каждого регрессора по отдельности.

Наилучшая модель получилась, когда были использованы только регрессоры *"law"* и *"PetrolPrice".* Коэффициент детерминации (*Adjusted R-squared*) улучшился, и статистические характеристики также улучшились.

Попытка ввести логарифмическое преобразование параметра *"front"* показала некоторое улучшение модели, но в целом не дало заметного прироста качества.

Попытки ввести все возможные произведения пар регрессоров не привели к улучшению модели, а лишь ухудшили ее показатели, поэтому было принято решение использовать только регрессоры *"law" и "PetrolPrice".* Исходя из данного исследования:

Положительные зависимости:

* model2, которая включает переменные law, kms и PetrolPrice, имеет Adjusted R-squared равный 42%. Здесь переменные law и PetrolPrice показывают положительную связь с front, что подразумевает, что увеличение law и PetrolPrice приводит к увеличению front.
* model3, которая использует логарифмическое преобразование переменных, имеет Adjusted R-squared равный 48%. В этой модели log(front) положительно связан с log(law) и log(PetrolPrice). Увеличение log(law) и log(PetrolPrice) приводит к увеличению log(front), что можно интерпретировать как положительную зависимость между этими переменными.
* model4, которая использует всевозможные произведения пар регрессоров переменных, имеет Adjusted R-squared равный 43%. В этой модели front положительно связан с law и (PetrolPrice^2). Увеличение law и (PetrolPrice^2) приводит к увеличению front, что является положительной зависимостью между этими переменными.

Таким образом, в данном коде были выявлены положительные зависимости между различными регрессорами и зависимой переменной.

Код решения задачи и сведения о проверенных моделях приведены в Приложении 2.1.

# Практическая работа №2.2

Набор данных: *Seatbelts*

Объясняемая переменная: *front*

Регрессоры: *law, kms, PetrolPrice*

## Условие

1. Доверительные интервалы для всех коэффициентов в модели, p = 95%.

2. Сделайте вывод о отвержении или невозможности отвергнуть статистическую гипотезу о том, что коэффициент равен 0.

3. Доверительный интервал для одного прогноза (p = 95%, набор значений регрессоров выбираете сами).

## Решение

Задание№1

Доверительные интервалы для всех коэффициентов в модели*(model*), p = 95%.

*model = lm(front~law + kms + PetrolPrice, data)*

confint(*model*, level = 0.95)

И получим интервалы в таблице 5:

Таблица 5. Доверительные интервалы для общей модели

|  |  |  |
| --- | --- | --- |
|  | 2.5% | 97.5% |
| *(Intercept)* | 1.246773e+03 | 1.618217e+03 |
| *law* | -2.894007e+02 | 1.517127e+02 |
| *kms* | -8.353530e-03 | 6.856602e-03 |
| *PetrolPrice* | -7.118528e+03 | -3.644096e+03 |

*Для model3\_1:*

Доверительный интервал для прогноза лучшей модели(*model3\_1*):

Для подсчета вручную требуется найти число степеней свободы в модели: 189 - 3 = 186. После найдем для такого числа степеней свободы и p = 95% значение t-критерия Стьюдента: t = 1.97, с помощью cat(t\_critical = qt(0.975, df = 186)).

Тогда для каждого из коэффициентов будут справедливы уравнения:

Таблица 6. Доверительные интервалы для *model3\_1*

|  |  |  |
| --- | --- | --- |
|  | 2.5% | 97.5% |
| *law\_con* | -0.05899181 | -0.03742819 |
| *PetrolPRice\_con* | -0.821111 | -0.415815 |

Задание№2

Выводы о отвержении или невозможности отвергнуть статистическую гипотезу о том, что коэффициент равен 0 для параметров из таблицы 5:

*Для model:*

* Для *Intercept*, *B* не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.
* Для *law*, *B* не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.
* Для kms, *B* может равняться 0. Следовательно подтверждаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.
* Для *PetrolPrice*, B не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.

Выводы о отвержении или невозможности отвергнуть статистическую гипотезу о том, что коэффициент равен 0 для параметров из таблицы 6:

*Для model3\_1:*

* Для *law\_con*, B не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%
* Для *PetrolPrice\_con*, B не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.

Задание№3

Анализ оценки доверительного интервала для *model3\_1* прогноза

(p = 95%, с набором значений: *law* = 20, *PetrolPrice* = 10), используя функцию predict:

Таблица 7 для *model3\_1:*

|  |  |  |
| --- | --- | --- |
| **fit** | **lwr** | **upr** |
| 3.439129 | 2.543131 | 4.335128 |

Получим прогноз для *model3\_1* в котором:

* прогноз модели = 3.439129
* нижняя граница доверительного интервала = 2.543131
* верхняя граница доверительного интервала = 4.335128

## Вывод

В результате данной работы была построена общая модель регрессии, описывающая зависимость переменной *"front" от "law", "kms" и "PetrolPrice"*. Были рассчитаны доверительные интервалы для всех коэффициентов в модели с уровнем доверия 95%. Также была проверена гипотеза о том, может ли коэффициент равняться 0, на уровне значимости 5%. В результате проверки гипотезы оказалось, что коэффициенты *"Intercept",* *"law" и "PetrolPrice"* не могут равняться 0, а коэффициент *"kms"* может равняться 0.

Далее была построена лучшая модель с использованием переменных *"law" и "PetrolPrice"* и были рассчитаны доверительные интервалы для коэффициентов этой модели. На основании результатов расчетов было выявлено, что коэффициенты *"law" и "PetrolPrice"* не могут равняться 0 на уровне значимости 5%.

Был также рассчитан доверительный интервал для первого прогноза на основе лучшей модели. Предсказанный прогноз составил 3.439129, а нижняя и верхняя границы доверительного интервала равны соответственно 2.543131 и 4.335128.

Таким образом, на основании данной работы можно сделать вывод о том, что переменные *"law" и "PetrolPrice"* значимо взаимосвязаны с переменной *"front"*, а переменная *"kms"* имеет небольшую статистически значимую связь с *"front".* Кроме того, можно использовать лучшую модель для прогнозирования значения переменной *"front"* на основе значений *"law"* и *"PetrolPrice"*, при этом уровень доверия составляет 95%. В итоге на основе прогнозов можно утверждать о том, что имеются зависимости, выявленные в модели, включают положительную зависимость между "front" и "kms", а также отрицательные зависимости между "front" и "law", а также между "front" и "PetrolPrice".

Код решения задачи и сведения о проверенных моделях приведены в Приложении 2.2.

# Практическая работа №3

Набор данных: r12i\_os26c.scv

Объясняемая переменная: *salary*

Регрессоры:

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) переменная *sex* – пол

5) переменная *age* – возраст

6) переменная *higher\_educ -* наличие высшего образования

7) переменная *city\_status – место проживания*

*8)* переменная *working\_hours –* среднее число рабочих часов в неделю

## Условие

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 логарифмы, степени (хотя бы от 0.1 до 2 с шагом 0.1), произведения вещественных регрессоров

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

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

5. Оцените лучшие модели для подмножества индивидов, указанных в варианте (Не состоявшие в браке мужчины, с высшим образованием; городские жители, состоящие в браке, женщины). Сделайте вывод о том, какие индивиды получают наибольшую зарплату.

## Решение

Необходимо считать данные исследования из файла и убрать значения NA, преобразовав их в удобном виде:

1. Переменная *sex* принимает значение 1 для мужского пола, 0 – для женского
2. *age* ­– переменная с нормализованным возрастом (формула нормализации значения: data2["*age*"] = (*age* - mean(*age*)) / sqrt(var(*age*))
3. Семейное положение:
   1. *wed1* = 1, если человек состоит в зарегистрированном браке, иначе 0
   2. *wed2* = 1, если человек разведён или вдовец, иначе 0
   3. *wed3* = 1, если человек никогда не был в браке, иначе 0
   4. Проверим, что между *wed1, wed2, wed3* нет линейной зависимости
4. *higher\_edu*c = 1, если у человека есть высшее образование, иначе 0
5. *city\_status* = 1, если человек живёт в городе, иначе 0
6. *working\_hours* – переменная с нормализованным числом рабочих часов в неделю (формула для нормализации значения: (*working\_hours* - mean(*working\_hours*)) / sqrt(var(*working\_hours*)))
7. *salary* – переменная с нормализованной зарплатой (формула для нормализации значения: ((*salary* - mean(*salary*)) / sqrt(var(*salary*)))

Задание №1

Линейная регрессия зарплаты на все параметры и оценим vif на рисунке №1:

*model1 = lm(data = data2, salary ~ sex + age + wed1 + wed2 + wed3 + higher\_educ + city\_status + working\_hours)*

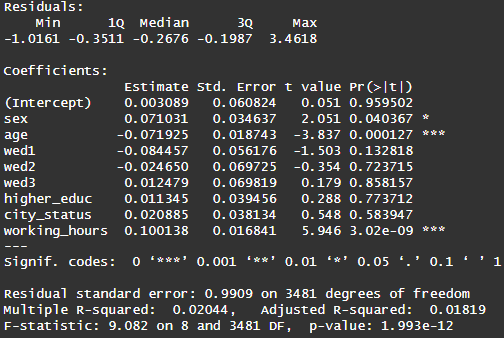


Рисунок №1. Характеристики model1(вида: linear regression).

Оценим vif для model1 из Рисунка №1:



Рисунок №2. vif(model1)

Из рисунка №1 и №2 видим, что переменные *wed1* имеет плохую p-статистику и vif. Уберём ее и посмотрим, как изменится *R2* на рисунке №3:

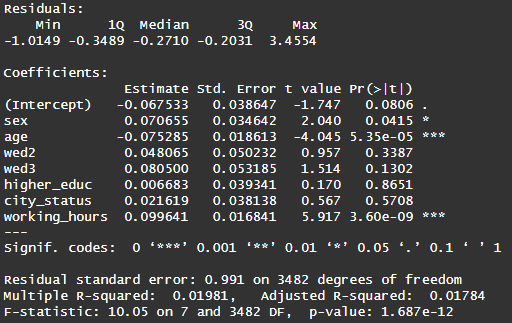


Рисунок №3. Результат работы *model1* с исключенным регрессором *wed1.*

 Оценим vif для model1 из Рисунка №3:

Рисунок №4. Результат vif(*model1*) без регрессора *wed1*.

Из рисунка №3 и №4 видим, что переменная *higher\_educ* имеет плохую p-статистику. Уберём ее и посмотрим, как изменится *R2* на рисунке №5:

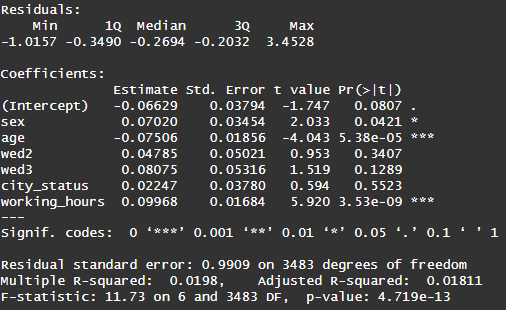


Рисунок №5. Результат работы *model1* с исключенным регрессором *wed1* и *higher\_educ.*

 Оценим vif для model1 из Рисунка №5:

Рисунок №6. Результат vif(*model1*) без регрессора *wed1 и higher\_educ.*

Из рисунка №5 и №6 видим, что p-статистика чуть улучшилась и параметры vif стабилизировались.

Задание №2

Эксперимент с функциями вещественных параметров, используя логарифмы (логарифмы и степени имеет смысл вводить только для параметров *age* и *working\_hours*, так как остальные принимают только значения 0 или 1.) и степени (хотя бы от 0.1 до 2 с шагом 0.1).

Модель с log на рисунке №7:

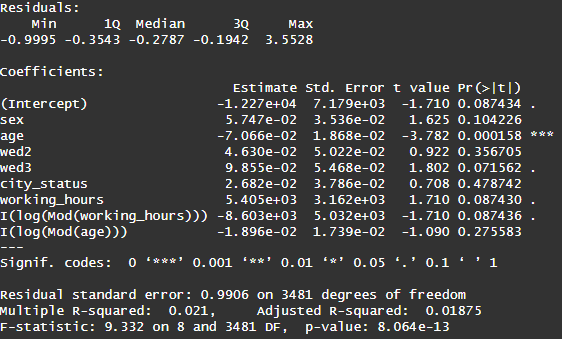
*model1 = lm( salary ~ sex + age + wed2 + wed3 + city\_status + working\_hours + I(log(Mod(working\_hours))) + I(log(Mod(age))), data=data2)*

Рисунок №7. Результат работы *model1* c I*(log(Mod(working\_hours))) + I(log(Mod(age))).*

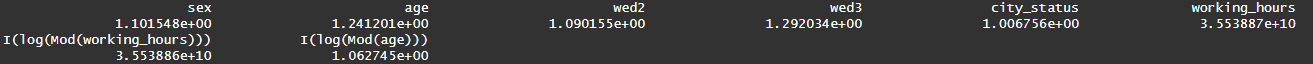
 Оценим vif для model1 из Рисунка №7:

Рисунок №8. Результат работы vif(*model1)* c I*(log(Mod(working\_hours))) + I(log(Mod(age))).*

Из рисунка №7 и №8 видим, что p-статистика плохая и параметры vif ухудшились для *working\_hours* и *I(log(Mod(working\_hours))),* исключаю *working\_hours* на рисунке №9*:*

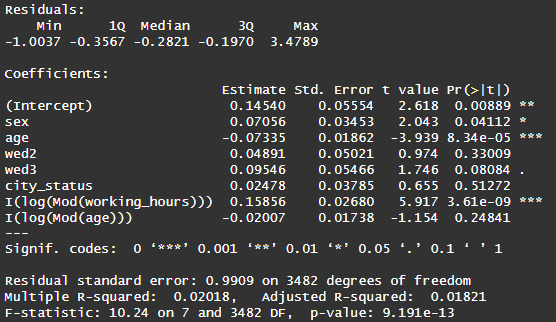


Рисунок №9. Результат работы *model1* с исключенным параметром *working\_hours.*

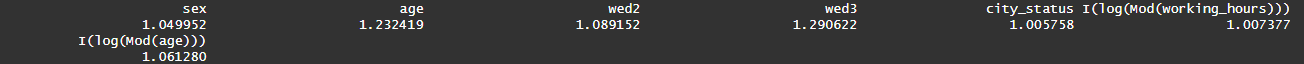
 Оценим vif для model1 из Рисунка №9:

Рисунок №10. Результат работы vif(*model1)* без регрессора *working\_hours.*

Из рисунка №9 и №10 видим, что p-статистика чуть улучшилась и параметры vif стабилизировались. Дальнейшее исключение регрессоров не приведет к значительным результатам, следовательно дальше можно работать с набором регрессоров:

(*sex + age + wed2 + wed3 + city\_status + I(log(Mod(working\_hours)))+ I(log(Mod(age))))* ).

Добавим модели со степенями для набора регрессоров с шагом 0.1 от 0.1 до 2.0:

current\_pow=0.1

*model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))*

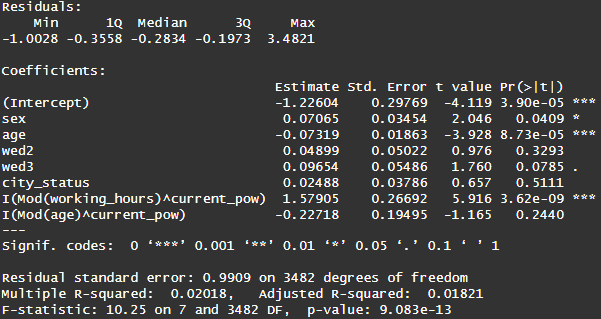


Рисунок №11. Результат работы *model1* в степени 0.1.

Оценим vif для model1 из Рисунка №11:



Рисунок 12. Результат работы vif(*model1*) в степени 0.1.

Из рисунка №11 и №12 видим, что p-статистика плохая, параметры vif хорошие, дальнейшую работу проводим с имеющейся моделью с шагом степени 0.1, так как исключение параметров не даст значимых изменений*.* Подмечу что дальнейшее увеличение степени приведут к ухудшению p-статистики и постепенному уменьшению *R2*, а так же мало колебанию значений vif, что не особо скажется на точности моделей.

Задание №3

Наилучшие модели из построенных: по значимости параметров, включённых в зависимости, и по объяснённому с помощью построенных зависимостей разбросу *adjusted R2 – R2\_adj.*

Наилучшими по значению R2 из всех моделей являются модели для степеней 0.1, 0.2, 0.3. Разброс R2 - R2\_adj одинаковый, аналогично модели являются лучшими по p-статистике у регрессоров. Из этих трёх моделей лучшей является с *current\_pow*=0.1 показателем R2 ~ 0.01821:

*model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))*

Задание №4

Анализ лучшей модели показал, что больше всего зарабатывают молодые мужчины с высшим образованием, которые разведены или не когда не были женаты, проживающие в городах, работающие большое число часов в неделю.

Задание №5

Оценка регрессии для подмножества индивидов:

а) Не состоявшие в браке мужчины, с высшим образованием на рисунке №13:

data3 = subset(data2, wed3 == 1)

data3 = subset(data3, higher\_educ == 1)

data3 = subset(data3, sex == 1)

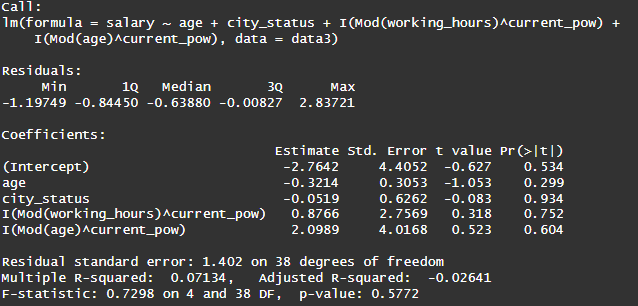


Рисунок №13. Результат вычисления модели *model1 = lm(data = data3, salary ~ age + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))* для не состоявших в браке мужчин, с высшим образованием.

Вывод: R^2 ~ 0.07134. Наибольшая зарплата у мужчин с высшим образованием старшего возраста, живущих в городе, не состоявшие в браке, работающих много.

б) Городские жители, состоящие в браке, женщины на рисунке №14:

data3 = subset(data2, sex == 0)

data3 = subset(data3, city\_status == 1)

data3 = subset(data3, wed1 == 1)

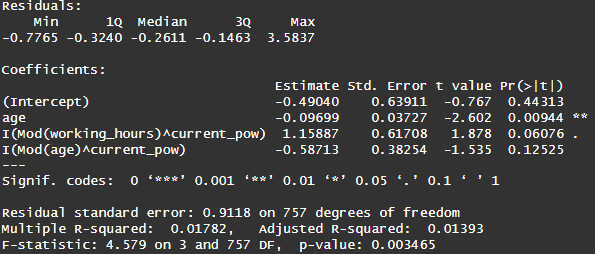


Рисунок №14. Результат вычисления модели *model1 = lm(data = data3, salary ~ age + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))* для городских жителей, состоящих в браке, женщин.

Вывод: *R^2* ~ 0.01782. Наибольшая зарплата у работающих много, молодого возраста.

## Вывод

Из данной работы была получена база данных с выбранными параметрами для анализа, которая была нормализована и проанализирована на наличие линейной зависимости между семейными положениями. После этого была построена линейная регрессия зарплаты на все параметры, которые были выделены из данных мониторинга. Поскольку полученные результаты были неудовлетворительными, были использованы функции вещественных параметров, в том числе логарифм и степени (хотя бы от 0.1 до 2 с шагом 0.1). Исходя из полученных результатов, можно сделать вывод, что ни одна из построенных моделей не дал удовлетворительных результатов, так как значения коэффициента детерминации R^2 оказались низкими. Следует отметить, что при использовании логарифма и степеней вещественных параметров для улучшения результатов, были достигнуты небольшие улучшения. Однако, в целом, даже после применения таких методов, результаты остались неудовлетворительными.

Так же стоит отметить, что из всей статистики больше всего зарабатывают мужчины с высшим образованием, старшего возраста, живущие в городах, работающие большое число часов в неделю.

Среди не состоявших в браке мужчин, с высшим образованием, наибольшая зарплата у мужчин с высшим образованием старшего возраста, живущих в городе, не состоявшие в браке, работающих много.

Среди городских жителей, состоящих в браке, женщин, наибольшая зарплата у работающих много, молодого возраста.

Код решения задачи и сведения о проверенных моделях приведены в Приложении 3.

# Практическая работа №4

Набор данных: Students performance in exams

<https://www.kaggle.com/datasets/spscientist/students-performance-in-exams>

Классификация по столбцу: *race/ethnicity (group C –* класс 0, остальные уровни – класс 1*)*

Тип классификатора: *LogisticRegression* (логистическая регрессия)

Регрессоры:

1. *"gender"-* female/male
2. *"parental* level of education"- some college, associate's degree and other
3. *"lunch" –* standard, free/reduced
4. *"test preparatio*n course"- none, completed
5. *"**math score"-* math score (0-100)
6. *"reading score"*- reading score (0-100)
7. *"writing score"*- writing score (0-100)

## Условие

1. Обработайте набор данных набор данных, указанный во втором столбце таблицы 4.1, подготовив его к решению задачи классификации. Выделите целевой признак, указанный в последнем столбце таблицы, и удалите его из данных, на основе которых будет обучаться классификатор. Разделите набор данных на тестовую и обучающую выборку. Постройте классификатор типа, указанного в третьем столбце, для задачи классификации по параметру, указанному в последнем столбце. Оцените точность построенного классификатора с помощью метрик *precision, recall и F1* на тестовой выборке.

2. Постройте классификатор типа Случайный Лес *(Random Forest)* для решения той же задачи классификации. Оцените его качество с помощью метрик *precision, recall и F1* на тестовой выборке. С помощью *GridSearch* переберите различные комбинации гиперпараметров: на первой итерации задайте большие шаги (50 или 100) по числу деревьев *n\_estimators*. На следующих итерациях определите лучшее количество деревьев *n\_estimators* с точностью до 10. Какой из классификаторов оказывается лучше?

## Решение

Задание №1

Необходимо обработать набор данных, указанный во втором столбце таблицы 4.1, подготовив его к решению задачи классификации. Для этого считаем данные из файла с помощью команды *pd$read\_csv* и выделяем целевой признак, указанный в последнем столбце таблицы, и исключаем его из данных, на основе которых будет обучаться классификатор с помощью преобразования столбца *"race/ethnicity"* в факторный тип данных, используя функцию *factor*(), содержащего факторную переменную *"race/ethnicity"*.

После выделения столбца необходимо разделить датасет на тестовую и обучающую выборку с помощью команды *model\_selection$train\_test\_split.*

Далее необходимо преобразовать некоторые переменные в факторные с помощью команды *factor()* и заменить категориальные переменные на фиктивные переменные, используя обучающие данные содержащие только категориальные признаки, для которых требуется создание фиктивных переменных.

Далее с помощью команды *linear\_model$LogisticRegression()* создаем экземпляр класса *LogisticRegression* из модуля *linear\_model* и проводим прогноз на основе данных тестового набора с помощью линейной регрессии.Последним шагом будет оценика точности построенного классификатора с помощью метрик *precision, recall и F1* на тестовой выборке. В результате получаем оценки:

* *Precision* = 0.5148515
* *Recall* = 0.3421053
* *F1* = 0.4110672

Задание №2

Для построения классификатора типа случайный лес необходимо с помощью команды *ensemble$RandomForestClassifier* создать экземпляр класса *RandomForestClassifier* из модуля *ensemble* и провести прогноз на основе данных тестового набора с помощью случайного леса.

На первой итерации с помощью *GridSearch* переберем различные комбинации гиперпараметров: с шагом 50 по числу деревьев *n\_estimators.* В результате обучения модели получаем оценки:

* *Precision* = 0.4356436
* *recall* = 0.3697479
* *F1* = 0.4

На второй итерации с помощью *GridSearch* переберем различные комбинации гиперпараметров: с шагом 10 по числу деревьев *n\_estimators.* В результате обучения модели получаем оценки:

* *Precision* = 0.4455446
* *Recall* = 0.3571429
* *F1* = 0.3964758

Низкая точность классификации обуславливается:

* Недостаточном количеством данных для обучения модели, особенно если в данных присутствует много шума и несбалансированных классов.
* Неправильным выбором гиперпараметров модели, таких как количество деревьев и их глубина.

Можно попытаться увеличить точность модели с помощью добавления других гиперпараметров и анализа важности признаков, таких как:

* *max\_depth* = c(3L, 5L, 7L, 9L, 11L),
* *min\_samples\_split* = c(2L, 5L, 10L, 15L, 20L),
* *min\_samples\_leaf* = c(1L, 2L, 4L, 8L, 16L),
* *max\_features* = c(5L, 10L, "sqrt", "log2", NULL))

В результате, при добавлении новых гиперпараметров для модели с шагом 50 имеем оценку:

* *precision* = 0.9306931
* *recall* = 0.3574144
* *F1* = 0.5164835

А также для модели с шагом 10 значения оценки равны:

* *Precision* = 0.8613861
* *Recall* = 0.3411765
* *F1* = 0.488764

В итоге добавление новых параметров привело к улучшениям моделей и увеличило время их обучения.

Лучшим из все представленных классификаторов оказывается *RandomForest* с шагом 50, так как имеет большую точность чем *LogisticRegression.*

## Вывод

Данный код выполняет задачу классификации студентов на основе их характеристик и оценок с помощью двух методов: логистической регрессии и случайного леса. В начале кода данные извлекаются из основного набора данных и выбираются только необходимые переменные. Пропущенные значения удаляются. Затем категориальные переменные преобразуются в фиктивные переменные.

Далее данные разбиваются на обучающий и тестовый наборы, на которых обучается модель.

Затем производится прогнозирование на тестовых данных, используя два метода: логистическую регрессию и случайный лес.

Кроме того, проводится поиск наилучшей комбинации гиперпараметров для случайного леса и производится прогноз с использованием этой комбинации параметров.

В итоге имеем:

* Результаты логистической регрессии показывают хорошую точность (0.5) и плохим отзыв (0.3), а также оценкой F1 (0.4).
* После поиска наилучшей комбинации гиперпараметров для случайного леса с шагом 50 в параметре *n\_estimators* были получены результаты с точностью (0.9) и отзывом (0.35), оценкой F1 (0.5).
* Поиск наилучшей комбинации гиперпараметров для случайного леса с шагом 10 в параметре *n\_estimators* привел к результатам с точностью (0.86), а также отзывом (0.34), оценкой F1 (0.48).
* Увеличение количества параметров по которым обучается модель привело к их улучшению.

Таким образом, наилучшим методом является случайный лес с шагом 50 в параметре *n\_estimators*. Точность была низкая, т.к. данные не разбиваются по этому параметру на группы, которые можно объяснить с помощью других параметров. Требовалось учитывать важность признаков для классификации, что в итоге привело к улучшению оценки.

Код решения задачи и сведения о проверенных моделях приведены в Приложении 4.

# Практическая работа *№5*

Набор данных: Hotel booking demand

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

Этот набор данных содержит информацию о бронировании городского отеля и курортного отеля и включает в себя, среди прочего, такую информацию, как время совершения бронирования, продолжительность пребывания, количество взрослых, детей и/или грудничков, а также количество доступных парковочных мест.

Вся идентифицирующая личность информация была удалена из данных

## Условие

1. Сколько в наборе данных объектов и признаков? Дать описание каждому признаку,

если оно есть.

2. Сколько категориальных признаков, какие?

3. Столбец с максимальным количеством уникальных значений категориального

признака?

4. Есть ли бинарные признаки?

5. Какие числовые признаки?

6. Есть ли пропуски?

7. Сколько объектов с пропусками?

8. Столбец с максимальным количеством пропусков?

9. Есть ли на ваш взгляд выбросы, аномальные значения?

10. Столбец с максимальным средним значением после нормировки признаков через

стандартное отклонение?

11. Столбец с целевым признаком?

12. Сколько объектов попадает в тренировочную выборку при использовании

*train\_test\_split* с параметрами *test\_size = 0.3, random\_state = 42*?

13. Между какими признаками наблюдается линейная зависимость (корреляция)?

14. Сколько признаков достаточно для объяснения 90% дисперсии после применения

метода *PCA*?

15. Какой признак вносит наибольший вклад в первую компоненту?

16. Построить двухмерное представление данных с помощью алгоритма *t-SNE*. На

сколько кластеров визуально, на ваш взгляд, разделяется выборка? Объяснить

смысл кластеров.

## Решение

Задание №1

Необходимо выяснить сколько в наборе данных объектов и признаков. Дать описание каждому признаку, если оно есть. С помощью *info()* выводим основную информацию по датасету, которая изображена на рисунке 15:

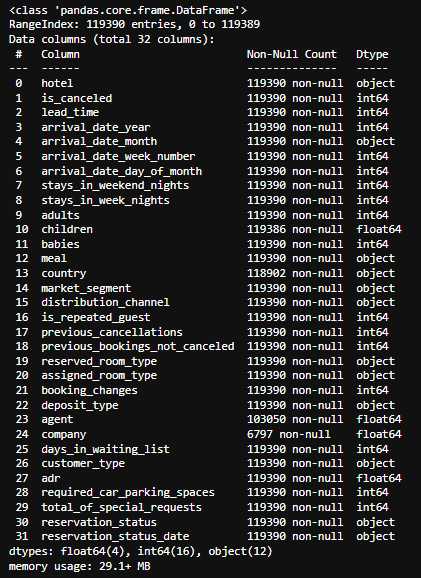


Рисунок №15. Результат работы *info().* Основная информация по датасету.

На основе Рисунка №15 можно сказать, что в выбранном наборе 119390 объектов и 32 признака, кроме-того проведем анализ параметров в датасете и опишем каждый параметр:

1. *hotel* (H1 = Курортный отель или H2 = Городской отель) object
2. *is\_canceled* (Значение, указывающее, было ли отменено бронирование (1) или нет (0)) int64
3. *lead\_time* (Количество дней, прошедших между датой внесения бронирования в PMS и датой прибытия) int64
4. *arrival\_date\_year* (Год прибытия) int64
5. *arrival\_date\_month* (Месяц прибытия) object
6. *arrival\_date\_week\_number* (Номер недели в году для даты прибытия) int64
7. *arrival\_date\_day\_of\_month* (День прибытия) int64
8. *stays\_in\_weekend\_nights* (Количество ночей на выходных, в которые гость останавливался или бронировал номер отеля) int64
9. *stays\_in\_week\_nights* (Количество ночей в неделю, в которые гость останавливался или бронировал номер в отеле) int64
10. *adults* (Количество взрослых) int64
11. *children* (Количество детей) float64
12. *babies* (Количество малышей) int64
13. *meal* (Тип забронированного питания. Категории представлены в стандартных пакетах гостиничного питания: Undefined/SC – без пакета питания; BB – Кровать и завтрак; HB – полупансион (завтрак и еще один прием пищи – обычно ужин); FB – Полный пансион (завтрак, обед и ужин)) object
14. *country* (Страна происхождения. Категории представлены в формате ISO 3155-3:2013) object
15. *market\_segment* (Обозначение сегмента рынка. В категориях термин “TA” означает “Туристические агенты”, а “TO” означает “Туроператоры”.) object
16. *distribution\_channel* (Канал распространения бронирования. Термин “TA” означает “Туристические агенты”, а “TO” означает “Туроператоры”.) object
17. *is\_repeated\_guest* (Значение, указывающее, было ли имя при бронировании от повторного гостя (1) или нет (0)) int64
18. *previous\_cancellations* (Кол. предыдущих бронирований, которые были отменены клиентом до текущего бронирования) int64
19. *previous\_bookings\_not\_canceled* (Кол. предыдущих бронирований, не отмененных клиентом до текущего бронирования) int64
20. *reserved\_room\_type* (Код забронированного типа номера. Код представлен вместо обозначения в целях анонимности.) object
21. *assigned\_room\_type* (Код для типа номера, указанного при бронировании. Иногда назначенный тип номера отличается от забронированного по причинам работы отеля (например, овербукинг) или по запросу клиента. Код представлен вместо обозначения по соображениям анонимности.) object
22. *booking\_changes* (Кол. изменений, внесенных в бронирование с момента ввода в PMS до регистрации заезда или отмены) int64
23. *deposit\_type* (Указание на то, внес ли клиент депозит для гарантии бронирования. Эта переменная может принимать три категории: Без депозита – депозит не был внесен; Невозвратный – депозит был внесен в размере общей стоимости проживания; Подлежащий возврату – депозит был внесен на сумму, меньшую общей стоимости проживания.) object
24. *agent* (Идентификационный номер туристического агентства, оформившего бронирование) float64
25. *company* (Идентификационный номер компании/юридического лица, оформившего бронирование или ответственного за оплату бронирования. Идентификатор указывается вместо обозначения по соображениям анонимности) float64
26. *days\_in\_waiting\_list* (Кол. дней, в который бронирование было в списке ожидания, прежде чем оно было подтверждено) int64
27. *customer\_type* (Тип бронирования, предполагающий одну из четырех категорий: Контрактный - когда с бронированием связано выделение или другой тип контракта; Групповой – когда бронирование связано с группой; Временный – когда бронирование не является частью группы или контракта и не связано с другим временным бронированием; Временная сторона - когда бронирование является временным, но связано, по крайней мере, с другим временным бронированием) object
28. *adr* (Средняя суточная ставка, полученная путем деления суммы всех сделок по размещению на общее количество ночей проживания) float64
29. *required\_car\_parking\_spaces* (Количество парковочных мест, требуемое заказчиком) int64
30. *total\_of\_special\_requests* (Кол. особых запросов заказчика (например, двуспальная кровать или высокий этаж)) int64
31. *reservation\_status* (Статус последнего бронирования, предполагающий одну из трех категорий: Отменено – бронирование было отменено клиентом; Выезд - клиент зарегистрировался, но уже отбыл; Незаезд – клиент не зарегистрировался и не сообщил отелю о причине) object
32. *reservation\_status\_date* (Дата, на которую был установлен последний статус. Эту переменную можно использовать в сочетании со статусом бронирования, чтобы понять, когда было отменено бронирование или когда клиент выписался из отеля) object

Задание №2

Необходимо определить сколько в датасете категориальных признаков, какие именно?

# отбор категориальных признаков

cat\_columns = df.select\_dtypes(['object']).columns

num\_cat\_columns = len(cat\_columns)

#Вывод категориальных признаков

print("Количество категориальных признаков в датасете: ", num\_cat\_columns)

print("Список категориальных признаков: ", list(cat\_columns))

Листинг №1. Отбор категориальных признаков.

Из 32 признаков, выявленных с помощью кода в листинге 1, категориальными являются:

*hotel* - тип отеля (City Hotel или Resort Hotel)

*arrival\_date\_month* - месяц прибытия гостя

*meal* - тип питания, предоставляемый в отеле

*country* - страна происхождения гостя

*market\_segment* - рыночный сегмент, в котором было сделано бронирование

*distribution\_channel* - канал распространения, через который было сделано бронирование

*is\_repeated\_guest* - бинарный признак, указывающий, является ли гость повторным (1) или нет (0)

*reserved\_room\_type* - тип забронированного номера

*assigned\_room\_type* - тип номера, который был назначен при заселении гостя (может отличаться от забронированного)

*deposit\_type* - тип депозита, который был внесен для гарантирования бронирования

customer\_type - тип клиента (индивидуальный или групповой)

reservation\_status - текущий статус бронирования (забронирован, изменен или отменен)

Итого 12 категориальных признаков

Задание №3

Найти столбец с макс. количеством уникальных значений категориального признака.

# Создаем пустой словарь

unique\_counts = {}

**for** col **in** df.columns:

# Если тип столбца является object (категориальный), то подсчитать количество уникальных значений

**if** df[col].dtype == 'object':

unique\_values = df[col].nunique()

# Количество уникальных значений в словарь

unique\_counts[col] = unique\_values

# Столбец с максимальным количеством уникальных значений категориального признака

max\_column = max(unique\_counts, key=unique\_counts.get)

print("Столбец с максимальным количеством уникальных значений категориального признака: ", max\_column)

print("Количество уникальных значений: ", unique\_values)

Листинг 2. Отбор категориальных признаков.

Из данных приведённых в пункте 1 и результатов работы функции *nunique*() в листинге 2 видно, максимальное количество уникальных значений = 926 для стобца "*reservation\_status\_date*"

Задание №4

Проанализировать есть ли бинарные признаки.

i = 1

print("Переменные имеющие бинарные признаки: ")

# Перебор признаков у которых максимум 2 значения

**for** col **in** df.columns:

**if** df[col].nunique() == 2:

print(i, col)

i += 1

print("Количество переменных имеющих бинарные признаки: ", i-1)

Листинг 3. Отбор бинарных признаков.

Из данных приведённых в пункте 1 и результатов работы функции в листинге 3, выполненной в блоке выше, видно, что бинарными являются 3 признака: *hotel, is\_canceled, is\_repeated\_guest.*

Задание №5

Проанализировать какие числовые признаки.

# выбор только числовых признаков

numeric\_cols = df.select\_dtypes(include='number').columns.tolist()

# вывод списка числовых признаков

print(numeric\_cols)

# вывод количества числовых признаков

print("Количество числовых признаков: ",len(numeric\_cols))

Листинг 4. Отбор числовых признаков.

Из данных приведённых в пункте 1 и результатов работы функции, выполненной в листинге 4, видно, что чисто числовыми являются 20 признаков: [*'is\_canceled', 'lead\_time', 'arrival\_date\_year','arrival\_date\_week\_number','arrival\_date\_day\_of\_month','stays\_in\_weekend\_nights','stays\_in\_week\_nights','adults','children','babies','is\_repeated\_guest', 'previous\_cancellations','previous\_bookings\_not\_canceled','booking\_changes','agent','company','days\_in\_waiting\_list', 'adr', 'required\_car\_parking\_spaces', 'total\_of\_special\_requests'*].

Задание №6

Узнать есть ли пропуски.

# Создание объекта а с вычислением суммы уникальных объектов

a = df.isnull().sum()

print(a)

Листинг 5. Создание объекта с вычислением суммы объектов с пропусками.

Из результата работы функции *isnull().sum()* в листинге 5, в датасете имеются пропуски некоторых признаках.

Задание №7

Проанализировать количество объектов с пропусками.

# Вычисление количества объектов у которых имеется больше 0 пропусков

num\_cols\_with\_missing\_values = df.isnull().any().sum()

print("Количество стобцов с пропущенными переменными: ",num\_cols\_with\_missing\_values)

Листинг 6. Вычисление количества объектов у которых имеется больше 0 пропусков.

Из результата работы функции в листинге 6, в датасете имеются пропуски в 4 признаках: *children, country, agent, company*.

Задание №8

Найти столбец с максимальным количеством пропусков.

# Создание объекта а с вычислением суммы уникальных объектов

a = df.isnull().sum()

# Нахождение признака с максимальным количеством пропусков

max\_col\_name = a.idxmax()

print(a)

print("Маскимальное число пропусков в стобце: ", max\_col\_name, max(a))

Листинг 7. Вычисление количества объектов с макс. числом пропусков.

Из результата работы блока кода в листинге 7, можно сделать вывод, что столбцом с максимальным числом пропусков это *company* с количество пропусков равным 112593.

Задание №9

Определить есть ли выбросы, аномальные значения.

df.describe()

numeric\_cols = ['is\_canceled', 'lead\_time', 'arrival\_date\_year', 'arrival\_date\_week\_number', 'arrival\_date\_day\_of\_month',

'stays\_in\_weekend\_nights', 'stays\_in\_week\_nights', 'adults', 'children', 'babies', 'is\_repeated\_guest',

'previous\_cancellations', 'previous\_bookings\_not\_canceled', 'booking\_changes', 'agent', 'company',

'days\_in\_waiting\_list', 'adr', 'required\_car\_parking\_spaces', 'total\_of\_special\_requests']

# создание матрицы признаков

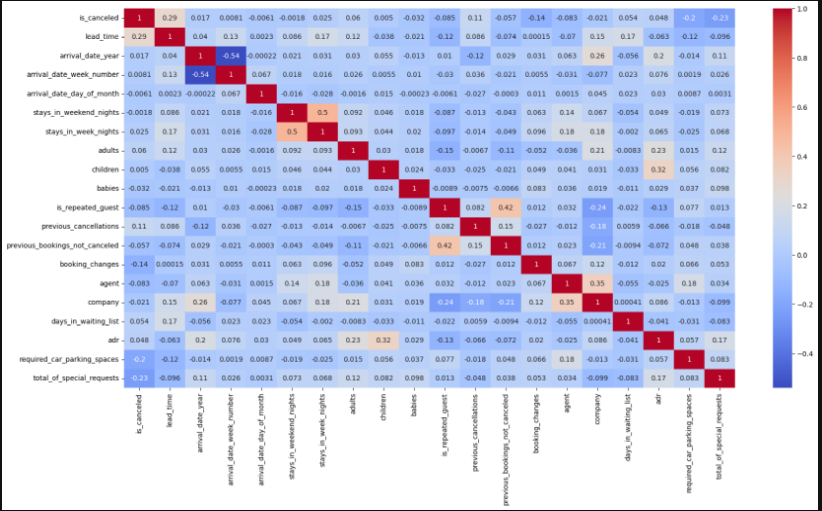
X = df[numeric\_cols]

plt.figure(figsize=(20,10))

sns.heatmap(X.corr(), annot=True, cmap='coolwarm')

plt.show()

Листинг 8. Вывод графиков для определения выбросов и аномальных значений.

Рисунок №16. Температурный график для определения аномальных значений.

Из таблицы выведенной с помощью листинга 8 графика из рисунка №16 можно увидеть, что некоторые переменные имеют выбросы, например, переменные: *lead\_time, adr, previous\_cancellations, days\_in\_waiting\_list, stay\_in\_weekend\_nights, stays\_in\_week\_nights.*

Задание №10

Найти столбец с максимальным средним значением после нормировки признаков через стандартное отклонение.

numeric\_cols = ['is\_canceled', 'lead\_time', 'arrival\_date\_year', 'arrival\_date\_week\_number', 'arrival\_date\_day\_of\_month',

'stays\_in\_weekend\_nights', 'stays\_in\_week\_nights', 'adults', 'children', 'babies', 'is\_repeated\_guest',

'previous\_cancellations', 'previous\_bookings\_not\_canceled', 'booking\_changes', 'agent', 'company',

'days\_in\_waiting\_list', 'adr', 'required\_car\_parking\_spaces', 'total\_of\_special\_requests']

# создание матрицы признаков

X = df[numeric\_cols]

# создание объекта StandardScaler

scaler = StandardScaler()

# стандартизация признаков

X\_scaled = scaler.fit\_transform(X)

# создание датафрейма с нормированными признаками

df\_scaled = pd.DataFrame(X\_scaled, columns=numeric\_cols)

# выбор столбца с максимальным средним значением

max\_mean\_col = df\_scaled.mean().idxmax()

# вывод столбца с максимальным средним значением

print("Cтолбец с максимальным средним значением: ",max\_mean\_col)

# вывод таблицы после нормировки признаков через стандартное отклонение

df\_scaled.describe()

Листинг 9. Нормировка признаков через стандартное отклонение и выявление столбцы с максимальным отклонением.

Из данных работы функции, выполненной в листинге 9, видим, что столбец с максимальным средним значением после нормировки признаков - *adr*.

Задание №11

Выбрать столбец с целевым признаком.

Поскольку мы собираемся делать прогноз отмены брони. В качестве целевого признака будет выбран - *booking\_changes.*

Задание №12

Выявить сколько объектов попадает в тренировочную выборку при использовании *train\_test\_split* с параметрами *test\_size* = 0.3, *random\_state* = 42.

# Исключим столбец с целевым признаком из датасета для тренировки

training = df.drop('booking\_changes', axis=1)

test = df['booking\_changes']

# Разделим данные на тренировочные и проверочные

x\_train, x\_test, y\_train, y\_test = train\_test\_split(training, test, test\_size=0.3, random\_state=42)

print("Тренировчная выборка: ", x\_train.shape, "Тестовая выборка: ", x\_test.shape)

Листинг 10. Выделение количества объектов из тестовой выборки при параметрах *test\_size* = 0.3, *random\_state* = 42.

Судя по размерам таблицы, в тренировочную выборку пойдет 83573 объектов, а в тестовую 35817.

Задание №13

Проанализировать между какими признаками наблюдается линейная зависимость (корреляция).

# Для выявления мультиколлениарности признаков, будем использовать VIF

numeric\_cols = ['is\_canceled', 'lead\_time', 'arrival\_date\_year', 'arrival\_date\_week\_number', 'arrival\_date\_day\_of\_month',

'stays\_in\_weekend\_nights', 'stays\_in\_week\_nights', 'adults', 'children', 'babies', 'is\_repeated\_guest', 'previous\_cancellations', 'previous\_bookings\_not\_canceled', 'booking\_changes', 'agent', 'company', 'days\_in\_waiting\_list', 'adr', 'required\_car\_parking\_spaces', 'total\_of\_special\_requests']

df = df.dropna()

# создание матрицы признаков

X = df[numeric\_cols]

# добавление константы к матрице признаков

X = sm.add\_constant(X)

# определение VIF для каждого признака

vif = pd.Series([variance\_inflation\_factor(X.values, i) **for** i **in** range(X.shape[1])], index=X.columns)

# вывод результатов

print(vif)

Листинг 11. Определение корреляции для каждого признака.

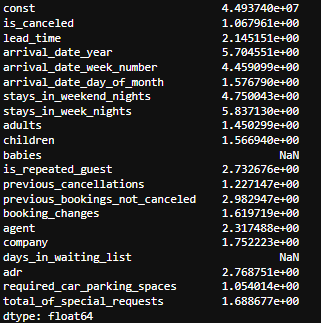


Рисунок №17. Корреляция для каждого признака.

На основе данных полученных из листинга 11 и рисунка №17 можно сделать вывод: однозначная корреляция наблюдается между признаками у которых vif > 4, то есть у *признаков arrival\_date\_year, arrival\_date\_week\_number, stays\_in\_weekend\_nights, stays\_in\_week\_nights.*

Задание №14

Определить сколько признаков достаточно для объяснения 90% дисперсии после применения метода PCA.

# отбор только числовых признаков

numeric\_cols = x\_train.select\_dtypes(include='number').columns.tolist()

# стандартизация числовых признаков

stand\_X = pd.DataFrame(preprocessing.scale(x\_train[numeric\_cols]), columns=numeric\_cols)

stand\_X = stand\_X.dropna()

# PCA

pca = PCA(0.90) # 90% дисперсии

pca.fit(stand\_X)

print("Количество компонентов достаточных для объяснения 90% дисперсии: ", pca.n\_components\_)

print("Суммарная доля объясненной дисперсии для модели PCA: " ,sum(pca.explained\_variance\_ratio\_))

Листинг 12. Применение метода PCA для числовых признаков.

На основе кода из листинга 12. Можно сделать вывод:

* количество компонентов достаточных для объяснения 90% дисперсии: 8
* суммарная доля объясненной дисперсии для модели PCA: 0.9006748788508198

Задание №15

Выявить какой признак вносит наибольший вклад в первую компоненту.

pca = PCA(n\_components=8)

x = pca.fit(stand\_X)

first\_component = pca.components\_[0]

greatest\_contribution\_i = 0

greatest\_contribution = first\_component[greatest\_contribution\_i]

**for** i **in** range(0, len(first\_component)):

**if** abs(first\_component[i]) > abs(greatest\_contribution):

greatest\_contribution = first\_component[i]

greatest\_contribution\_i = i

print("Наибольший вклад вносит: " ,greatest\_contribution\_i, greatest\_contribution)

stand\_X.iloc[:, 6]

Листинг 13. Выявление признака внесшего больший вклад в компоненту

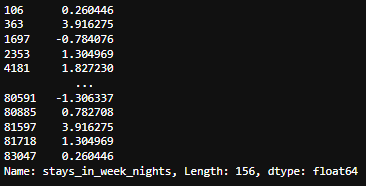
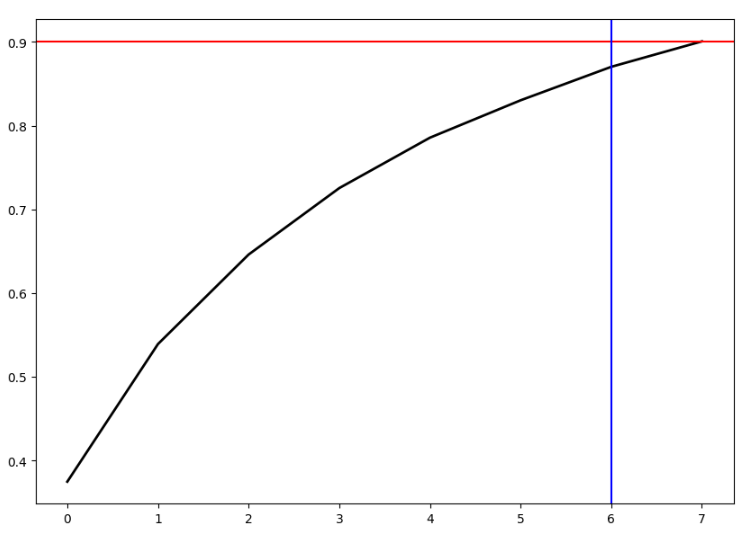


Рисунок №18. Вывод конкретного параметра отвечающего за больший вклад

Рисунок №19. Вывод графика зависимости кумулятивной суммы объясненной дисперсии по главным компонентам, полученным с использованием метода главных компонент (PCA).

На основе данных из листинга 13 и рисунка №18, №19 можно сказать, что наибольший вклад в первую компоненту вносит 6 признак(stays\_in\_week\_nights).

Задание №16

Построить двухмерное представление данных с помощью алгоритма *t-SNE*. На сколько кластеров визуально, на ваш взгляд, разделяется выборка? Объяснить смысл кластеров.

# выбор признаков для анализа

numeric\_cols = ['is\_canceled', 'lead\_time', 'arrival\_date\_year', 'arrival\_date\_week\_number', 'arrival\_date\_day\_of\_month',

'stays\_in\_weekend\_nights', 'stays\_in\_week\_nights', 'adults', 'children', 'babies', 'is\_repeated\_guest',

'previous\_cancellations', 'previous\_bookings\_not\_canceled', 'booking\_changes', 'agent', 'company',

'days\_in\_waiting\_list', 'adr', 'required\_car\_parking\_spaces', 'total\_of\_special\_requests']

# применение алгоритма t-SNE

tsne = TSNE(n\_components=2, random\_state=42)

df\_tsne = tsne.fit\_transform(df[numeric\_cols])

# создание графика

plt.scatter(df\_tsne[:, 0], df\_tsne[:, 1], alpha=0.5)

plt.xlabel('t-SNE 1')

plt.ylabel('t-SNE 2')

plt.title('t-SNE visualization')

plt.show()

Листинг 14. Применение алгоритма *t-SNE* и создание графика для определения кластеров.

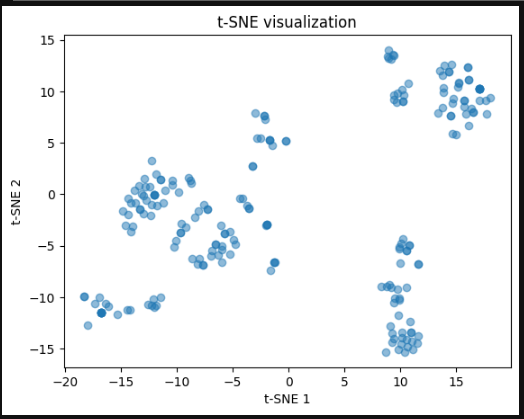


Рисунок №20. График, отображающий визуализацию данных с использованием алгоритма t-SNE для выявления кластеров.

На основе рисунка №20 можно сказать, что визуально выборка разделяется от 4 до 8 кластеров, однако чёткого разделения не видно.

Каждый кластер точек, т.е. группа точек, близких друг к другу на графике, указывает на определенный тип бронирования. Таким образом, можно использовать кластерный анализ для выделения схожих групп объектов и анализа их характеристик.

Код решения задачи и сведения о проверенных моделях приведены в Приложении 5.

# Заключение

В ходе Практической работы №1 были созданы линейные регрессии, которые позволили нам выяснить, что с ростом коэффициента *Agriculture* (процент населения Уэльса занято сельским хозяйством в качестве профессии) и *Fertility* (​общий стандартизированный показатель рождаемости) уменьшается процент призывников, получивших наивысшую оценку на армейском экзамене, следовательно взаимосвязи являются отрицательными.

Во время Практической работы №2 мы построили доверительные интервалы для регрессоров и линейной модели, что позволило нам оценить точность каждого из них, а также общую точность линейной регрессии.

В Практической работе №3 мы проанализировали данные из исследования РМЭЗ НИУ ВШЭ. Оказалось, среди не состоявших в браке мужчин, с высшим образованием, наибольшая зарплата у мужчин с высшим образованием старшего возраста, живущих в городе, не состоявшие в браке, работающих много. А также, среди городских жителей, состоящих в браке, женщин, наибольшая зарплата у работающих много, молодого возраста.

Практическая работа №4 включала классификацию респондентов по статусу оценкок, полученных студентами с использованием классификатора "*Linear Regression*" и метода "*Random Forest*".

В Практической работе №5 мы представили и описали данные, основанные на наборe данных, который содержит информацию о бронировании городского отеля и курортного отеля и включает в себя, среди прочего, такую информацию, как время совершения бронирования, продолжительность пребывания, количество взрослых, детей и/или грудничков, а также количество доступных парковочных мест.

# Список литературы

1. Wickham, H. ggplot2 Elegant Graphics for Data Analysis / H. Wickham. – Houston, Texas, USA: Springer, 2016. – 260 с.
2. Hyndman, R.J. Forecasting: Principles and Practice / R.J. Hyndman, G. Athanasopoulos. – Australia: OTexts, 2018. – 382 с.
3. Faraway, J.J. Practical Regression and Anova using R / J.J. Faraway. – Monash University, Australia: University of Bath, 2002. – 210 с.
4. Fox, J. An R Companion to Applied Regression / J. Fox, S. Weisberg. – University of California, Davis: SAGE Publications, 2018. – 608 с.
5. Артамонов, Н.В. Эконометрика с R / Н.В. Артамонов. – Россия, 119002, Москва: МЦНМО, 2019. – 256 с.
6. Вербик, М. Путеводитель по современной эконометрике / М. Вербик. – Москва: Научная книга, 2008. – 616 с.

Приложения

Приложение к практической работе №1

|  |
| --- |
| library("lmtest")  library("GGally")  library('tidyverse')  data=swiss  summary(data)  ##Среднее значение элементов - " Education, Examination, Fertility пакета swiss.  #data$Examination = 16.48936  #data$Agriculture = 50.65957  #data$Fertility = 70.14255  cat(mean(data$Examination))  cat(mean(data$Agriculture))  cat(mean(data$Fertility))  ##Среднее квадратичное отклонение - "Examination, Agriculture, Fertility пакета swiss.  #standard\_deviation\_Examination: 7.977883  #standard\_deviation\_Agriculture: 22.71122  #standard\_deviation\_Fertility: 12.4917  standard\_deviation\_Examination<-sqrt(sum((data$Examination - mean(data$Examination))^2/(length(data$Examination)-1)))  standard\_deviation\_Agriculture<-sqrt(sum((data$Agriculture - mean(data$Agriculture))^2/(length(data$Agriculture)-1)))  standard\_deviation\_Fertility<-sqrt(sum((data$Fertility - mean(data$Fertility))^2/(length(data$Fertility)-1)))  cat("standard\_deviation\_Examination: " , standard\_deviation\_Examination)  cat("standard\_deviation\_Agriculture: " ,standard\_deviation\_Agriculture)  cat("standard\_deviation\_Fertility: " , standard\_deviation\_Fertility)  ##Дисперсия - Education, Examination, Fertility пакета swiss.  #data$Examination = 63.64662  #data$Agriculture = 515.7994  #data$Fertility = 156.0425  var(data$Examination)  var(data$Agriculture)  var(data$Fertility)  ##Зависимости вида y= a + bx.  summary(lm(Examination~Fertility, data))  summary(lm(Examination~Agriculture, data))  cat(data$Examination\_Fertility<-45.42 + (-0.412)\*data$Fertility)  cat(data$Examination\_Agriculter<-28.70 + (-0.241)\*data$Agriculture)  #График зависимости data$Agriculture  ggplot(data=swiss, aes(x = data$Examination, y = data$Examination\_Agriculter)) +    geom\_point(color= "red" , shape=25, fill= "blue" , size = 2.2, alpha = 0.5) +    geom\_smooth(method = 'loess',size=1.3, level=0.80)  #График зависимости data$Fertility  ggplot(data=swiss, aes(x = data$Examination, y = data$Examination\_Fertility)) +    geom\_point(color= "red" , shape=25, fill="blue" , size =2.2, alpha= 0.5) +    geom\_smooth(method = 'loess' ,size=1.3, level=0.80)  ##Коэффициент детерминации равен 64%, из этого следует, что 64% наших данных описывается нашей моделью.  #Можем сделать вывод (R-squared = 64%), что модель относительно хороша: для такой  #зависимости (только две объясняющая переменная) коэффициент не мал, но для  #полного описания процесса нужно добавлять другие параметры.  ##P-начение p для переменной-предиктора Fertility равно 1.40e-05.(\*\*\*)  #Поскольку это значение меньше 0,05, оно имеет статистически значимую связь с переменной отклика в модели.  #P-начение p для переменной-предиктора Agriculture равно 1.52e-06.(\*\*\*)  #Поскольку это значение меньше 0,05, оно имеет статистически значимую связь с переменной отклика в модели.  model = lm(Examination~Fertility+Agriculture, data )  model  summary(model)  ## Из взаимосвязи вида y= a + bx можно сделать вывод, с ростом коэффициента Agriculture и Fertility уменьшается  # % призывников, получивших наивысшую оценку на армейском экзамене, следовательно взаимосвязи являются отрицательными.  cat(data$Examination\_Fertility<-45.42 + (-0.412)\*data$Fertility)  cat(data$Examination\_Agriculter<-28.70 + (-0.241)\*data$Agriculture)  #Вывод: обе взаимосвязи являются отрицательными. |

Приложение к практической работе №2.1

|  |
| --- |
| library("lmtest")  library("GGally")  library("car")  library("dplyr")  data(Seatbelts)  data = na.omit(Seatbelts)  data <- as.data.frame(data)  ##проверим возможность использования всех зявленных регрессоров в одной модели  model1=(lm(law~kms + PetrolPrice, data))  vif(model1)  summary(model1) #R^2 = 28%. относительно независимые регрессоры  model1=(lm(kms~law + PetrolPrice, data))  vif(model1)  summary(model1) #R^2 = 27%. относительно независимые регрессоры  model1=(lm(PetrolPrice~law + kms, data))  vif(model1)  summary(model1) #R^2 = 19%. относительно независимые регрессоры  ##построим общую модель регрессии  model2 = lm(front~law + kms + PetrolPrice, data)  vif(model2)  summary(model2)  #модель неплохая, но есть невзимосвязанная переменная kms (с отсутствием звездочек)  #Adjusted R-squared = 42%. По результатам зависимости front~law и front~PetrolPrice имеет три звездды.  #р-значение у kms большое.  ##Получим коэф. R^2 для каждого отдельно  summary(lm(front~law, data))$r.squared #R^2 равен 31%.  summary(lm(front~kms, data))$r.squared #R^2 равен 12% Попробуем исключить.  summary(lm(front~PetrolPrice, data))$r.squared #R^2 равен 29%.  summary(lm(front~law + PetrolPrice, data)) #Adjuster R-squared = 43%. Коэф. увеличился на 1% и улучшилась p-статистика, следовательно далее пльзуемся этой моделью.  ##Введем логарифмы  #В модель нельзя ввести log регрессоров без замены значений равных 0 в таблице law на 0.001  data$law[data$law == 0] <- 0.001  #log регрессоров  model3=(lm(log(front)~ I(log(law)) + I(log(PetrolPrice)), data))  vif(model3)  summary(model3) #Лучшая модель log(front)~I(log(law), Adjusted R-squared: 0.4835  summary(lm(log(front)~I(log(law)), data))$r.squared #R^2 равен 39%.  model3 = lm(front ~ I(log(law)) + I(log(PetrolPrice)), data)  vif(model)  summary(model) #Лучшая модель log(front)~I(log(law), Adjusted R-squared: 0.4299  summary(lm(log(front)~I(log(law)), data))$r.squared #R^2 равен 39%.  model3 = lm(front ~ I(log(law)) + PetrolPrice, data)  vif(model)  summary(model) #Лучшая модель log(front)~I(log(law), Adjusted R-squared: 0.4309  summary(lm(log(front)~I(log(law)), data))$r.squared #R^2 равен 39%.  model3 = lm(front ~ law + I(log(PetrolPrice)), data)  vif(model)  summary(model) #Лучшая модель log(front)~I(log(law), Adjusted R-squared: 0.4299  summary(lm(log(front)~I(log(law)), data))$r.squared #R^2 равен 39%.  ##Введем в модель всевозможные произведения пар регрессоров  model4 = lm(front ~ law + PetrolPrice + I(law^2) + I(PetrolPrice^2) + I(law\*PetrolPrice), data)  vif(model4) # есть линейная зависимость, уберем регрессоры с максимальным VIF  summary(model4)#Лучшая модель front~law, Adjusted R-squared: 0.428  model4 = lm(front ~ law + PetrolPrice + I(PetrolPrice^2) + I(law\*PetrolPrice), data)  vif(model4) # есть линейная зависимость, уберем регрессоры с максимальным VIF  summary(model4)#Лучшая модель front~law, Adjusted R-squared: 0.428  model4 = lm(front ~ law + PetrolPrice + I(PetrolPrice^2), data)  vif(model4) # есть линейная зависимость, уберем регрессоры с максимальным VIF  summary(model4)#Лучшая модель front~law, Adjusted R-squared: 0.4278  model4 = lm(front ~ law + I(PetrolPrice^2), data)  vif(model4) # линейной зависимости нет  summary(model4)#Лучшая модель front~law, Adjusted R-squared: 0.4306  # Наилучшей из них будет следующая модель:  model3\_1 = (lm(log(front)~ I(log(law)) + I(log(PetrolPrice)), data)) # R^2 = 0.4835  ##вывод: исходя из данного исследования самой лучшей моделью будет model3\_1 со значением Adjusted R-squared: 0.4835,  #самая сильная связь у log(front)~I(log(law) со значением Adjusted R-squared: 0.3911, но также существует хорошая зависимость front~law. |

Приложение к практической работе №2.2

|  |
| --- |
| library("lmtest")  library("GGally")  library("car")  library("dplyr")  data(Seatbelts)  data = na.omit(Seatbelts)  data <- as.data.frame(data)  ##Построим общую модель регрессии  model = lm(front~law + kms + PetrolPrice, data)  summary(model)  ##Доверительные интервалы для всех коэффициентов в модели, p = 95%.  confint(model, level = 0.95)  ##проверим гипотезу о том может ли B=0.  #(Intercept) [1.246773e+03 , 1.618217e+03], B не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.  #law [-2.894007e+02 ,-1.517127e+02], B не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.  #kms [-8.353530e-03 , 6.856602e-03], B может равняться 0. Следовательно подтверждаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.  #PetrolPrice [-7.118528e+03 ,-3.644096e+03], B не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.  ##Для подсчета доверительного интервала для лучшей модели(model3\_1):  data$law[data$law == 0] <- 0.001  model3\_1 = (lm(log(front)~ I(log(law)) + I(log(PetrolPrice)), data))  summary(model3\_1) #Лучшая модель log(front)~I(log(law), Adjusted R-squared: 0.4835  #найдем Число степеней свободы в модели: 189 - 3 = 186.  #После найдем для такого числа степеней свободы и p = 95% значение t-критерия Стьюдента: t = 1.97  cat(t\_critical = qt(0.975, df = 186))  #Тогда для каждого из коэффициентов будут справедливы уравнения:  cat(law\_con <- cbind(((-0.048210) - 1.97\*0.005473), ((-0.048210) + 1.97\*0.005473)))  #law\_con [-0.05899181 -0.03742819], B не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.  cat(PetrolPRice\_con <- cbind(((-0.618463) - 1.97\* 0.102867), ((-0.618463) + 1.97\* 0.102867)))  #PetrolPRice\_con [-0.821111 -0.415815], B не может равняться 0. Следовательно отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5%.  ##Доверительный интервал для первого прогноза.  new.data = data.frame(law = 20, PetrolPrice = 10)  predict(model3\_1, new.data, interval = "confidence")  #прогноз модели = 3.439129  #нижняя граница доверительного интервала = 2.543131  #верхняя граница доверительного интервала = 4.335128 |

Приложение к практической работе №3

|  |
| --- |
| library("lmtest")  library("GGally")  library("car")  library("foreign")  library("dplyr")  library("sandwich")  library("rlms")  library("rstatix")  library("haven")  library("devtools")  data <- read.csv("C:/Users/Иван/Desktop/уник/R/r12i\_os26b.csv",sep=",", dec = ".", header=TRUE)  glimpse(data)  data = select(data, hh5, h\_age, h\_marst, h\_diplom, status, hj13.2, hj6.2)  data = na.omit(data)  glimpse(data)  ##Обновленная база данных для нормализованных значений  data2 = select(data)  glimpse(data)  #Пол  data2["sex"] = 0  data2$sex[which(data$hh5 == 1)] <- 1  #Возраст  age = data$h\_age  data2["age"] = (age - mean(age)) / sqrt(var(age))  ##Семейное положение:  #Состоит ли в зарегестрированном браке?  data2$wed1 = 0  data2$wed1[which(data$h\_marst==2)] <- 1  data2$wed1[which(data$h\_marst==6)] <- 1  #Разведен или вдовец?  data2$wed2 = 0  data2$wed2[which(data$h\_marst==4)] <- 1  data2$wed2[which(data$h\_marst==5)] <- 1  #Никогда не состоял в браке?  data2$wed3 = 0  data2$wed3[which(data$h\_marst==1)] <- 1  #Проверим, что отсутствует линейная зависимость между семейными положениями  vif(lm(data$hj13.2 ~ data2$wed1 + data2$wed2 + data2$wed3))  #Наличие высшего образования  data2$higher\_educ = 0  data2$higher\_educ[which(data$h\_diplom==6)] <- 1  #Живет ли в городе  data2$city\_status = 0  data2$city\_status[which(data$status==1)] <- 1  data2$city\_status[which(data$status==2)] <- 1  #Нормализованное среднее число рабочих часов в неделю  working\_hours = data$hj6.2  data2$working\_hours = (working\_hours - mean(working\_hours)) / sqrt(var(working\_hours))  #Нормализованная средняя зарплата  salary = data$hj13.2  data2$salary = (salary - mean(salary)) / sqrt(var(salary))  #####Постройте линейную регрессию зарплаты на все параметры, которые вы выделили из данных мониторинга. Не забудьте оценить коэффициент вздутия дисперсии VIF.  model1 = lm(data = data2, salary ~ sex + age + wed1 + wed2 + wed3 + higher\_educ + city\_status + working\_hours)  vif(model1)  summary(model1)  #R^2 ~ 0.01819, плохая p-статистика и vif, исключаю wed1  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + higher\_educ + city\_status + working\_hours)  vif(model1)  summary(model1)  #R^2 ~ 0.01784, плохая p-статистика, исключаю higher\_educ, увеличевается R^2.  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + working\_hours)  vif(model1)  summary(model1)  #R^2 ~ 0.01811, плохая p-статистика.  #####Поэкспериментируйте с функциями вещественных параметров: используйте логарифм и степени (хотя бы от 0.1 до 2 с шагом 0.1).  #Логарифмы и степени имеет смысл вводить только для параметров age и working\_hours, так как остальные принимают только значения 0 или 1.  #с логарифмами:  model1 = lm( salary ~ sex + age + wed2 + wed3 + city\_status + working\_hours + I(log(Mod(working\_hours))) + I(log(Mod(age))), data=data2)  vif(model1)  summary(model1)  #R^2 ~ 0.01875  #плохая p-статистика и vif для working\_hours и I(log(Mod(working\_hours))), исключаю working\_hours  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(log(Mod(working\_hours)))+ I(log(Mod(age))))  vif(model1)  summary(model1)  #R^2 ~ 0.01821  #чуть улучшилась p-статистика и хороший vif.  #Со степенями:  current\_pow=0.1  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01821  #Хорошие значения vif  current\_pow = 0.2  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01821  #Хорошие значения vif  current\_pow = 0.3  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.0182  #Хорошие значения vif  current\_pow = 0.4  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01818  #Хорошие значения vif  current\_pow = 0.5  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01816  #Хорошие значения vif  current\_pow = 0.6  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01813  #Хорошие значения vif  current\_pow = 0.7  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.0181  #Хорошие значения vif  current\_pow = 0.8  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01807  #Хорошие значения vif  current\_pow = 0.9  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01803  #Хорошие значения vif  #Для степени 1 результат мы уже имеем  current\_pow = 1.1  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status +I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01797  #Хорошие значения vif  current\_pow = 1.2  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01794  #Хорошие значения vif  current\_pow = 1.3  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01791  #Хорошие значения vif  current\_pow = 1.4  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01789  #Хорошие значения vif  current\_pow = 1.5  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #R^2 ~ 0.01787  #Далее в период с current\_pow = 1.5 до current\_pow = 2.0: чуть уменьшгается R^2 везде хорошие значения VIF и везде плохая p-статистика.  #####Выделите наилучшие модели из построенных: по значимости параметров, включенных в зависимости , и по объясненному с помощью построенных зависимостей разбросу adjusted R2 - R2adj.  #Сравним лучшие модели:  current\_pow=0.1  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #Multiple R-squared: 0.02018, Adjusted R-squared: 0.01821  current\_pow = 0.2  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status +I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #Multiple R-squared: 0.02018, Adjusted R-squared: 0.01821  current\_pow = 0.3  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status +I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #Multiple R-squared: 0.02017, Adjusted R-squared: 0.0182  #Разброс R2 - R2\_adj везде одинаковый, а R^2 больше для степени 0.2  #Итого, среди моделей без линейной зависимости параметров с хорошими показателями p-статистики у регрессоров лучшей по R^2 оказалась модель для степени 0.1:  current\_pow=0.1  model1 = lm(data = data2, salary ~ sex + age + wed2 + wed3 + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  vif(model1)  summary(model1)  #####Сделайте вывод о том, какие индивиды получают наибольшую зарплату.  #Учитывая лучшую модель больше всего зарабатывают молодые мужчины с высшим образованием, которые разведены или не когда не были женаты, проживающие в городах,  #работающие большое число часов в неделю.  #####Оцените регрессии для подмножества индивидов: а) Не состоявшие в браке мужчины, с высшим образованием; б )городские жители, состоящие в браке, женщины.  current\_pow = 0.1  #Не состоявшие в браке мужчины, с высшим образованием  data3 = subset(data2, wed3 == 1)  data3 = subset(data3, higher\_educ == 1)  data3 = subset(data3, sex == 1)  model1 = lm(data = data3, salary ~ age + city\_status + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  #для добавления проверки vif требуется убрать значения NA по этому я удалил wed2 из списка регрессоров модели.  vif(model1) #хорошие значения vif  summary(model1)  # R^2 ~ 0.07134  #Наибольшая зарплата у мужчин с высшим образованием старшего возраста, живущих в городе , не состоявшие в браке, работающих много.  #городские жители, состоящие в браке, женщины  data3 = subset(data2, sex == 0)  data3 = subset(data3, city\_status == 1)  data3 = subset(data3, wed1 == 1)  model1 = lm(data = data3, salary ~ age + I(Mod(working\_hours)^current\_pow) + I(Mod(age)^current\_pow))  #для добавления проверки vif требуется убрать значения NA по этому я удалил wed2 и wed3 из списка регрессоров модели.  vif(model1) #хорошие значения vif  summary(model1)  # R^2 ~ 0.01782  #Наибольшая зарплата у работающих много, молодого возраста. |

Приложение к практической работе №4

|  |
| --- |
| library("reticulate")  library("caret")  library("dplyr")  #Необходимые модули Python  pd <- import("pandas")  np <- import("numpy")  model\_selection <- import("sklearn.model\_selection")  linear\_model <- import("sklearn.linear\_model")  ensemble <- import("sklearn.ensemble")  #Считывание данных из CSV-файла  data = pd$read\_csv('C:/Users/Иван/Desktop/уник/R/StudentsPerformance.csv')  #Соответствующие столбцы и строки с отсутствующими значениями  data\_sel <- data[ ,c("gender", "race/ethnicity", "parental level of education", "lunch", "test preparation course", "math score", "reading score", "writing score")]  data\_sel <- na.omit(data\_sel)  data\_sel$race\_ethnicity <- ifelse(data\_sel$`race/ethnicity` == "group C", 0, 1)  #Новый столбец и факторную переменную на основе столбца "раса/этническая принадлежность"  Race\_ethnicity = factor(data\_sel[ ,c("race/ethnicity")])  X\_data = data[ ,c("gender", "parental level of education", "lunch", "test preparation course", "math score", "reading score", "writing score")]  #Данные на обучающие и тестовые наборы  train\_test\_split <- model\_selection$train\_test\_split(X\_data, Race\_ethnicity, train\_size = 0.7, random\_state = 123L)  X\_train <- train\_test\_split[[1]]  X\_test <- train\_test\_split[[2]]  y\_train <- train\_test\_split[[3]]  y\_test <- train\_test\_split[[4]]  #Определенние переменных в коэффициенты  X\_train$gender <- factor(X\_train$gender)  X\_test$gender <- factor(X\_test$gender)  y\_train <- factor(y\_train)  y\_test <- factor(y\_test)  #Замена категориальных переменных на Фиктивные переменных, используя обучающие данные  dummyvars <- dummyVars(" ~ .", data = X\_train[, c("gender", "parental level of education", "lunch", "test preparation course")])  X\_train\_dummy <- predict(dummyvars, newdata = X\_train)  X\_test\_dummy <- predict(dummyvars, newdata = X\_test)  #Прогноз на основе данных тестирования с помощью линейной регрессии  logreg <- linear\_model$LogisticRegression()  logreg$fit(X\_train\_dummy, y\_train)  y\_pred <- logreg$predict(X\_test\_dummy)  y\_pred <- factor(y\_pred, levels = levels(y\_test))  #Точность, отзыв и оценку F1, используя метод линейной регрессии  print(confusion\_matrix <- confusionMatrix(y\_pred, y\_test)$table)  print(precision <- confusion\_matrix[3, 3] / sum(confusion\_matrix[, 3]))# 0.5148515  print(recall <- confusion\_matrix[3, 3] / sum(confusion\_matrix[3, ]))# 0.3421053  print(F1 <- 2 \* precision \* recall / (precision + recall))# 0.4110672  #Пример использования Random Forest  rf <- ensemble$RandomForestClassifier(n\_estimators = 100L, random\_state = 123L)  rf$fit(X\_train\_dummy, y\_train)  y\_pred <- rf$predict(X\_test\_dummy)  y\_pred <- factor(y\_pred, levels = levels(y\_test))  #Точность, отзыв и оценку F1, используя метод случайный лес  print(confusion\_matrix <- confusionMatrix(y\_pred, y\_test)$table)  print(precision <- confusion\_matrix[3, 3] / sum(confusion\_matrix[, 3]))# 0.4356436  print(recall <- confusion\_matrix[3, 3] / sum(confusion\_matrix[3, ]))# 0.3697479  print(F1 <- 2 \* precision \* recall / (precision + recall))# 0.4  #Различные комбинации гиперпараметров для случайного дерева с шагом 50 в параметре n\_estimators  param\_grid <- list(n\_estimators = c(50L, 100L, 150L, 200L, 250L))  rf <- ensemble$RandomForestClassifier(random\_state = 123L)  rf\_grid <- model\_selection$GridSearchCV(rf, param\_grid = param\_grid, cv = 5L, scoring = "f1\_macro")  rf\_grid$fit(X\_train\_dummy, y\_train)  #Вывод лучших значений для шага 50 в параметре n\_estimators  print(rf\_grid$best\_params\_)  #Прогноз на основе данных тестирования с помощью случайного дерева с шагом 50 в параметре n\_estimators  rf\_best <- ensemble$RandomForestClassifier(n\_estimators = rf\_grid$best\_params\_$n\_estimators, random\_state = 123L)  rf\_best$fit(X\_train\_dummy, y\_train)  y\_pred <- rf\_best$predict(X\_test\_dummy)  y\_pred <- factor(y\_pred, levels = levels(y\_test))  #Точность, отзыв и оценку F1, используя метод случайный лес с шагом 50  print(confusion\_matrix <- confusionMatrix(y\_pred, y\_test)$table)  print(precision <- confusion\_matrix[3, 3] / sum(confusion\_matrix[, 3]))# 0.4356436  print(recall <- confusion\_matrix[3, 3] / sum(confusion\_matrix[3, ]))# 0.3697479  print(F1 <- 2 \* precision \* recall / (precision + recall))# 0.4  #Различные комбинации гиперпараметров для случайного дерева с шагом 10 в параметре n\_estimators  param\_grid <- list(n\_estimators = c(10L, 20L, 30L, 40L, 50L, 60L, 70L, 80L, 90L))  rf <- ensemble$RandomForestClassifier(random\_state = 123L)  rf\_grid <- model\_selection$GridSearchCV(rf, param\_grid = param\_grid, cv = 5L, scoring = "f1\_macro")  rf\_grid$fit(X\_train\_dummy, y\_train)  #Вывод лучших значений для шага 10 в параметре n\_estimators  print(rf\_grid$best\_params\_)  #Прогноз на основе данных тестирования с помощью случайного дерева с шагом 10 в параметре n\_estimators  rf\_best <- ensemble$RandomForestClassifier(n\_estimators = rf\_grid$best\_params\_$n\_estimators, random\_state = 123L)  rf\_best$fit(X\_train\_dummy, y\_train)  y\_pred <- rf\_best$predict(X\_test\_dummy)  y\_pred <- factor(y\_pred, levels = levels(y\_test))  #Точность, отзыв и оценку F1, используя метод случайный лес с шагом 10  print(confusion\_matrix <- confusionMatrix(y\_pred, y\_test)$table)  print(precision <- confusion\_matrix[3, 3] / sum(confusion\_matrix[, 3]))# 0.4455446  print(recall <- confusion\_matrix[3, 3] / sum(confusion\_matrix[3, ]))# 0.3571429  print(F1 <- 2 \* precision \* recall / (precision + recall))# 0.3964758  #Недостаточном количеством данных для обучения модели, особенно если в данных присутствует много шума и несбалансированных классов.  #Неправильным выбором гиперпараметров модели, таких как количество деревьев и их глубина.  ##### Можно попытаться увеличить точность модели с помощью добавление других гиперпараметров и анализа важности признаков:  param\_grid <- list(n\_estimators = c(50L, 100L, 150L, 200L, 250L),  max\_depth = c(3L, 5L, 7L, 9L, 11L),  min\_samples\_split = c(2L, 5L, 10L, 15L, 20L),  min\_samples\_leaf = c(1L, 2L, 4L, 8L, 16L),  max\_features = c(5L, 10L, "sqrt", "log2", NULL))  rf <- ensemble$RandomForestClassifier(random\_state = 123L)  rf\_grid <- model\_selection$GridSearchCV(rf, param\_grid = param\_grid, cv = 5L, scoring = "f1\_macro")  rf\_grid$fit(X\_train\_dummy, y\_train)  #Вывод лучших значений для шага 50 в параметре n\_estimators  print(rf\_grid$best\_params\_)  #Прогноз на основе данных тестирования с помощью случайного дерева с шагом 50 в параметре n\_estimators  rf\_best <- ensemble$RandomForestClassifier(n\_estimators=150L,  max\_depth=9L,  min\_samples\_split=20L,  min\_samples\_leaf=2L,  max\_features="sqrt",  random\_state=123L)  rf\_best$fit(X\_train\_dummy, y\_train)  importance <- rf\_best$feature\_importances\_  sorted\_indices <- order(importance, decreasing = TRUE)  top\_features <- sorted\_indices[1:4]  X\_train\_top <- X\_train\_dummy[, top\_features]  X\_test\_top <- X\_test\_dummy[, top\_features]  rf\_top <- ensemble$RandomForestClassifier(n\_estimators = 150L, max\_depth = 9L, min\_samples\_split = 20L, min\_samples\_leaf = 2L, max\_features = "sqrt", random\_state = 123L)  rf\_top$fit(X\_train\_top, y\_train)  y\_pred <- rf\_top$predict(X\_test\_top)  y\_pred <- factor(y\_pred, levels = levels(y\_test))  #Точность, отзыв и оценку F1, используя метод случайный лес с шагом 50  print(confusion\_matrix <- confusionMatrix(y\_pred, y\_test)$table)  print(precision <- confusion\_matrix[3, 3] / sum(confusion\_matrix[, 3]))# 0.9306931  print(recall <- confusion\_matrix[3, 3] / sum(confusion\_matrix[3, ]))# 0.3574144  print(F1 <- 2 \* precision \* recall / (precision + recall))# 0.5164835  #Различные комбинации гиперпараметров для случайного дерева с шагом 10 в параметре n\_estimators  param\_grid <- list(n\_estimators = c(10L, 20L, 30L, 40L, 50L, 60L, 70L, 80L),  max\_depth = c(3L, 5L, 7L, 9L, 11L),  min\_samples\_split = c(2L, 5L, 10L, 15L, 20L),  min\_samples\_leaf = c(1L, 2L, 4L, 8L, 16L),  max\_features = c(5L, 10L, "sqrt", "log2", NULL))  rf <- ensemble$RandomForestClassifier(random\_state = 123L)  rf\_grid <- model\_selection$GridSearchCV(rf, param\_grid = param\_grid, cv = 5L, scoring = "f1\_macro")  rf\_grid$fit(X\_train\_dummy, y\_train)  #Вывод лучших значений для шага 10 в параметре n\_estimators  print(rf\_grid$best\_params\_)  #Прогноз на основе данных тестирования с помощью случайного дерева с шагом 10 в параметре n\_estimators  rf\_best <- ensemble$RandomForestClassifier(n\_estimators=80L,  max\_depth=9L,  min\_samples\_split=2L,  min\_samples\_leaf=1L,  max\_features="sqrt",  random\_state=123L)  rf\_best$fit(X\_train\_dummy, y\_train)  importance <- rf\_best$feature\_importances\_  sorted\_indices <- order(importance, decreasing = TRUE)  top\_features <- sorted\_indices[1:4]  X\_train\_top <- X\_train\_dummy[, top\_features]  X\_test\_top <- X\_test\_dummy[, top\_features]  rf\_top <- ensemble$RandomForestClassifier(n\_estimators = 80L, max\_depth = 9L, min\_samples\_split = 2L, min\_samples\_leaf = 1L, max\_features = "sqrt", random\_state = 123L)  rf\_top$fit(X\_train\_top, y\_train)  y\_pred <- rf\_top$predict(X\_test\_top)  y\_pred <- factor(y\_pred, levels = levels(y\_test))  #Точность, отзыв и оценку F1, используя метод случайный лес с шагом 10  print(confusion\_matrix <- confusionMatrix(y\_pred, y\_test)$table)  print(precision <- confusion\_matrix[3, 3] / sum(confusion\_matrix[, 3]))# 0.8613861  print(recall <- confusion\_matrix[3, 3] / sum(confusion\_matrix[3, ]))# 0.3411765  print(F1 <- 2 \* precision \* recall / (precision + recall))# 0.488764  ## Добавление новых параметров не привело к улучшениям моделей  ###Вывод: Основываясь на данных исследования Students performance in exams можно сделать вывод о том,  #что лучшим типом классификатора является Random Forest так как имеет большую точность чем LogisticRegression |

Приложение к практической работе №5

|  |
| --- |
| import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  import seaborn as sns  from sklearn.preprocessing import StandardScaler  from sklearn.preprocessing import OneHotEncoder  from sklearn.model\_selection import train\_test\_split  import statsmodels.api as sm  from statsmodels.stats.outliers\_influence import variance\_inflation\_factor  from sklearn import preprocessing  from sklearn.decomposition import PCA  from sklearn.manifold import TSNE |

|  |
| --- |
| df = pd.read\_csv('C:\\Users\\Иван\\Desktop\\уник\\R\\hotel\_bookings.csv') |

|  |
| --- |
| # получение информации о типах данных и количестве непустых значений для каждого признака  df.info() |

|  |
| --- |
| # отбор категориальных признаков  cat\_columns = df.select\_dtypes(['object']).columns  num\_cat\_columns = len(cat\_columns)  #Вывод категориальных признаков  print("Количество категориальных признаков в датасете: ", num\_cat\_columns)  print("Список категориальных признаков: ", list(cat\_columns)) |

|  |
| --- |
| # Сосздаем пустой словарь  unique\_counts = {}  **for** col **in** df.columns:  # Если тип столбца является object (категориальный), то подсчитать количество уникальных значений  **if** df[col].dtype == 'object':  unique\_values = df[col].nunique()  # Количество уникальных значений в словарь  unique\_counts[col] = unique\_values  # Столбец с максимальным количеством уникальных значений категориального признака  max\_column = max(unique\_counts, key=unique\_counts.get)  print("Столбец с максимальным количеством уникальных значений категориального признака: ", max\_column)  print("Количество уникальных значений: ", unique\_values) |

|  |
| --- |
| i = 1  print("Переменные имеющие бинарные признаки: ")  # Перебор признаков у которых максимум 2 значения  **for** col **in** df.columns:  **if** df[col].nunique() == 2:  print(i, col)  i += 1    print("Количество переменных имеющих бинарные признаки: ", i-1) |

|  |
| --- |
| # выбор только числовых признаков  numeric\_cols = df.select\_dtypes(include='number').columns.tolist()  # вывод списка числовых признаков  print(numeric\_cols)  # вывод количества числовых признаков  print("Количество числовых признаков: ",len(numeric\_cols)) |

|  |
| --- |
| # Создание объекта а с вычислением суммы уникальных объектов  a = df.isnull().sum()  print(a) |

|  |
| --- |
| # Вычисление количества объектов у которых имеется больше 0 пропусков  num\_cols\_with\_missing\_values = df.isnull().any().sum()  print("Количество стобцов с пропущенными переменными: ",num\_cols\_with\_missing\_values) |

|  |
| --- |
| # Создание объекта а с вычислением суммы уникальных объектов  a = df.isnull().sum()  # Нахождение признака с максимальным количеством пропусков  max\_col\_name = a.idxmax()  print(a)  print("Маскимальное число пропусков в стобце: ", max\_col\_name, max(a)) |

|  |
| --- |
| df.describe()  numeric\_cols = ['is\_canceled', 'lead\_time', 'arrival\_date\_year', 'arrival\_date\_week\_number', 'arrival\_date\_day\_of\_month',  'stays\_in\_weekend\_nights', 'stays\_in\_week\_nights', 'adults', 'children', 'babies', 'is\_repeated\_guest',  'previous\_cancellations', 'previous\_bookings\_not\_canceled', 'booking\_changes', 'agent', 'company',  'days\_in\_waiting\_list', 'adr', 'required\_car\_parking\_spaces', 'total\_of\_special\_requests']  # создание матрицы признаков  X = df[numeric\_cols]  plt.figure(figsize=(20,10))  sns.heatmap(X.corr(), annot=True, cmap='coolwarm')  plt.show() |

|  |
| --- |
| # Исключим столбец с целевым признаком из датасета для тренировки  training = df.drop('booking\_changes', axis=1)  test = df['booking\_changes']  # Разделим данные на тренировочные и проверочные  x\_train, x\_test, y\_train, y\_test = train\_test\_split(training, test, test\_size=0.3, random\_state=42)  print("Тренировчная выборка: ", x\_train.shape, "Тестовая выборка: ", x\_test.shape) |

|  |
| --- |
| # Для выявления мультиколлениарности признаков, будем использовать VIF  numeric\_cols = ['is\_canceled', 'lead\_time', 'arrival\_date\_year', 'arrival\_date\_week\_number', 'arrival\_date\_day\_of\_month',  'stays\_in\_weekend\_nights', 'stays\_in\_week\_nights', 'adults', 'children', 'babies', 'is\_repeated\_guest',  'previous\_cancellations', 'previous\_bookings\_not\_canceled', 'booking\_changes', 'agent', 'company',  'days\_in\_waiting\_list', 'adr', 'required\_car\_parking\_spaces', 'total\_of\_special\_requests']  df = df.dropna()  # создание матрицы признаков  X = df[numeric\_cols]  # добавление константы к матрице признаков  X = sm.add\_constant(X)  # определение VIF для каждого признака  vif = pd.Series([variance\_inflation\_factor(X.values, i) **for** i **in** range(X.shape[1])], index=X.columns)  # вывод результатов  print(vif)  # отбор только числовых признаков  numeric\_cols = x\_train.select\_dtypes(include='number').columns.tolist()  # стандартизация числовых признаков  stand\_X = pd.DataFrame(preprocessing.scale(x\_train[numeric\_cols]), columns=numeric\_cols)  stand\_X = stand\_X.dropna()  # PCA  pca = PCA(0.90) # 90% дисперсии  pca.fit(stand\_X)  print("Количество компонентов достаточных для объяснения 90% дисперсии: ", pca.n\_components\_)  print("Суммарная доля объясненной дисперсии для модели PCA: " ,sum(pca.explained\_variance\_ratio\_)) |

|  |
| --- |
| pca = PCA(n\_components=8)  x = pca.fit(stand\_X)  first\_component = pca.components\_[0]  greatest\_contribution\_i = 0  greatest\_contribution = first\_component[greatest\_contribution\_i]  **for** i **in** range(0, len(first\_component)):  **if** abs(first\_component[i]) > abs(greatest\_contribution):  greatest\_contribution = first\_component[i]  greatest\_contribution\_i = i  print("Наибольший вклад вносит: " ,greatest\_contribution\_i, greatest\_contribution)  stand\_X.iloc[:, 6] |

|  |
| --- |
| # выбор признаков для анализа  numeric\_cols = ['is\_canceled', 'lead\_time', 'arrival\_date\_year', 'arrival\_date\_week\_number', 'arrival\_date\_day\_of\_month',  'stays\_in\_weekend\_nights', 'stays\_in\_week\_nights', 'adults', 'children', 'babies', 'is\_repeated\_guest',  'previous\_cancellations', 'previous\_bookings\_not\_canceled', 'booking\_changes', 'agent', 'company',  'days\_in\_waiting\_list', 'adr', 'required\_car\_parking\_spaces', 'total\_of\_special\_requests']  # применение алгоритма t-SNE  tsne = TSNE(n\_components=2, random\_state=42)  df\_tsne = tsne.fit\_transform(df[numeric\_cols])  # создание графика  plt.scatter(df\_tsne[:, 0], df\_tsne[:, 1], alpha=0.5)  plt.xlabel('t-SNE 1')  plt.ylabel('t-SNE 2')  plt.title('t-SNE visualization')  plt.show() |