



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«МИРЭА - Российский технологический университет»

РТУ МИРЭА

Институт кибернетики

Кафедра высшей математики

**ОТЧЁТ ПО** Практике по получению первичных профессиональных умений  
и навыков  
(указать вид практики)

**Тема практики:** Построение классификатора вин по набору данных «Red  
Wine Quality» (kaggle.com)  
приказ университета о направлении на практику  
793 – С от 12.02.2019 г.

Отчет представлен к  
рассмотрению:

Студент группы  
КМБО-01-18

Алиев М.Х.  
(расшифровка подписи)  
«6» июня 2019 г.

Отчет утвержден.  
Допущен к защите:

Руководитель практики  
от кафедры

Петрусевич Д.А.  
(расшифровка подписи)  
«6» июня 2019 г.

Москва 2019



## МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«МИРЭА - Российский технологический университет»

**РТУ МИРЭА**

**ЗАДАНИЕ НА Практику по получению первичных профессиональных умений и навыков**

**Студенту 1 курса учебной группы КМБО-01-18 института кибернетики  
Алиеву Мишану Хаммад оглы**

---

(фамилия, имя и отчество)

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

**Время практики:** с «16» февраля 2019 по «31» мая 2019

**Должность на практике:** практикант

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

**2. СОДЕРЖАНИЕ ПРАКТИКИ:**

2.1 Изучить: литературу и практические примеры по темам: 1) построение линейной регрессии, 2) использование метода главных компонент, 3) поиск и устранение линейной зависимости в данных, 4) основы нормализации данных, 5) методы классификации и кластеризации («решающее дерево», «случайный лес», «k ближайших соседей»), 6) обучение с учителем («градиентный спуск»).

2.2 Практически выполнить: 1) снижение размерности исходных задач при помощи метода главных компонент при возможности; построение линейной регрессии для некоторого параметра, исключение регрессоров, не коррелирующих с объясняемой переменной; решение задачи классификации или кластеризации на основе открытого набора данных с ресурса kaggle.com

2.3 Ознакомиться: с применением метода главных компонент; методов классификации («решающего дерева», «случайного леса»); методов градиентного спуска («градиентным бустингом»); методов кластеризации («k ближайших соседей»).

**3. ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ:** построение классификатора вин по набору данных «Red Wine Quality» (kaggle.com).

**4. ОРГАНИЗАЦИОННО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ:** построить классификацию на основе нескольких методов и произвести сравнение результатов классификации; сделать выводы о применимости использованных методов; сформировать выводы по результатам задачи из предметной области: можно ли сопоставить полученные классы с известными типами вин; есть ли связь между физико-химическими

характеристиками вина и показателями качества, полученными из рассмотренного набора данных?

Заведующий кафедрой  
высшей математики



Ю.И.Худак

«1» марта 2019г.

СОГЛАСОВАНО

Руководитель практики от кафедры:

«16» февраля 2019 г.




(подпись)

(Петрусеvич Д.А.)

(фамилия и инициалы)

Задание получил:

«16» февраля 2019 г.



(подпись)

(Алиев М.Х.)

(фамилия и инициалы)

# ИНСТРУКТАЖ ПРОВЕДЕН:

Вид мероприятия	ФИО ответственного, подпись, дата	ФИО студента, подпись, дата
Охрана труда	Петрусеви́ч Д.А.  «16» февраля 2019 г.	Алиев М.Х.  «16» февраля 2019 г.
Техника безопасности	Петрусеви́ч Д.А.  «16» февраля 2019 г.	Алиев М.Х.  «16» февраля 2019 г.
Пожарная безопасность	Петрусеви́ч Д.А.  «16» февраля 2019 г.	Алиев М.Х.  «16» февраля 2019 г.
Правила внутреннего распорядка	Петрусеви́ч Д.А.  «16» февраля 2019 г.	Алиев М.Х.  «16» февраля 2019 г.



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«МИРЭА - Российский технологический университет»

РТУ МИРЭА

**РАБОЧИЙ ГРАФИК ПРОВЕДЕНИЯ** Практики по получению  
первичных профессиональных умений и навыков

студента Алиева М.Х. 1 курса группы КМБО-01-18 очной формы  
обучения, обучающегося по направлению подготовки 01.03.02  
«Прикладная математика и информатика»,  
профиль «Математическое моделирование и вычислительная математика»

Неделя	Сроки выполнения	Этап	Отметка о выполнении
1	16.02.2019	Выбор темы практики/НИР. Пройти инструктаж по технике безопасности.	✓
1	16.02.2019	Вводная установочная лекция.	✓
3	02.03.2019	Построение и оценка линейной регрессии с помощью языка R	✓
5	16.03.2019	Использование метода главных компонент, выделение линейной зависимости в данных	✓
7	30.03.2019	Методы классификации и кластеризации; построение решающего дерева;	✓
9	13.04.2019	Концепция бэггинга, «случайный лес»; концепция бустинга; градиентные методы обучения и кластеризации	✓

17	07.06.2019	Представление отчётных материалов по практике/НИР и их защита. Передача обобщённых материалов на кафедру для архивного хранения.	✓
		Зачётная аттестация.	

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


**Согласовано:**

Заведующий  
кафедрой



/ ФИО / Худак Ю.И.

Руководитель  
практики от кафедры



/ ФИО / Петрусеви́ч Д.А.

Обучающийся



/ ФИО / Алиев М.Х.



## Оглавление

Задание 1 .....	3
Условие задачи: .....	3
Краткий обзор: .....	3
Решение задачи: .....	3
Заключение .....	7
Задание 2 .....	7
Условие задачи: .....	7
Краткий обзор: .....	8
Решение задачи: .....	8
Заключение .....	11
Задание 3 и 4 .....	12
Условие задачи: .....	12
Краткий обзор: .....	12
Решение задачи: .....	13
Заключение .....	18
Общий Вывод .....	18
Задача 1. ....	18
Задача 2. ....	18
Задача 3-4 .....	18
Список литературы: .....	19
Приложение 1 .....	20
Приложение 2 .....	26
Приложение 3 .....	35

## Задание 1.

### Условие задачи:

- 1) Нормализовать данные из набора, вычтя из каждого столбца среднее значение  $\text{mean}(x)$  и поделив на среднеквадратическое отклонение  $\sigma \sim \sqrt{\text{var}(x)}$ , где  $x$  – столбец данных.
- 2) Проверить, что в наборе данных нет линейной зависимости (построить зависимости между переменными, указанными в варианте, и проверить, что  $R^2$  в каждой из них не высокий). В случае, если  $R^2$  большой, один из таких столбцов можно исключить из рассмотрения.
- 3) Построить линейную модель зависимой переменной от указанных в варианте регрессоров по методу наименьших квадратов (команда `lm` пакета `lmtest` в языке R). Оценить, насколько хороша модель, согласно: 1)  $R^2$ , 2) p-значениям каждого коэффициента.
- 4) Ввести в модель логарифмы регрессоров. Сравнить модели и выбрать наилучшую.
- 5) Ввести в модель всевозможные произведения из пар регрессоров, в том числе квадраты регрессоров. Найдите одну или несколько наилучших моделей по доле объяснённого разброса в данных  $R^2$ .

Мой вариант (1 вариант): набор данных Swiss, объясняемая переменная - Fertility, регрессоры - Agriculture, Education, Catholic.

### Краткий обзор:

Мне предстоит поработать с языком программирования R и с встроенным в него датасетом `swiss`, в котором содержатся стандартизированные показатели рождаемости и социально-экономические показатели для каждой из 47 франкоязычных провинций Швейцарии по состоянию на 1888 год. Моя основная задача - нормализация данных, построение линейных моделей зависимости от некоторых переменных и сравнение моделей. Используемые библиотеки: “psych”, “dplyr”, “ggplot2”, “GGally”, “car”. Код к программе в Приложении 1.

### Решение задачи:

Разберемся со значениями переменных в задаче: Fertility - стандартная мера рождаемость в данной провинции; Agriculture - процент мужчин, занимающихся сельским хозяйством в качестве основной деятельности в данной провинции; Education - процент призывников, получивших какое-либо образование после начальной школы в данной провинции; Catholic - доля католического населения (в противовес протестантскому населению) в данной провинции;

Выделим столбцы нужных переменных в отдельную таблицу и будем работать уже с ней. Теперь нам предстоит нормализовать данные, вычтя из каждого столбца его среднее значение и разделив на среднеквадратическое отклонение. Далее для проверки отсутствия линейной зависимости среди регрессоров, построим первую линейную



модель `ml` с помощью метода наименьших квадратов – метод нахождения оптимальных параметров линейной регрессии, таких, что сумма квадратов ошибок минимальна(1). Посмотрим её характеристики для оценки параметра  $R^2$ . равен 64.23 %, что говорит о линейной независимости между регрессорами (утверждение верно, когда  $R^2 > 50\%$ ). Вообще  $R^2$  измеряет долю общей дисперсии зависимой переменной (меру разброса значений данной величины относительно её математического ожидания), объяснённую моделью (не зависит от числа параметров). Кроме  $R^2$  существует коэффициент `adj R^2` - скорректированный на число параметров  $R^2$ , т.е. данная характеристика зависит от "числа входных переменных"(чем больше количество, тем больше штраф => лучше точность => эта характеристика более точна). В нашем случае `adjusted R-squared` = 0.6173.(2) Линейную независимость также можно было бы проверить с помощью функции `vif(variance inflation factor)` - коэффициента мультиколлинеарности, от модели `ml`, посмотрев на возвращаемое значение: 0-5 - регрессоры независимы(их убирать не надо), 5-10 - существует неявная зависимость(стоит подумать на целесообразность сохранения регрессора), 10 и больше - существует однозначная зависимость(регрессор лучше убрать)(3).

Далее обратим внимание на *p*-значения переменных, которые описывают их качество (от 0 звездочек и точки до 3 звездочек). В нашем случае везде от 2 до 3 звёзд, кроме свободного члена (0 звезд), но от него никак не избавиться, поэтому будем работать непосредственно с самими регрессорами(4). В целом, для первой модели результат неплох!!

```
Call:
lm(formula = Fertility ~ Agriculture + Education + Catholic,
    data = d)

Residuals:
    Min       1Q   Median       3Q      Max
-1.2151 -0.5242  0.1104  0.4661  1.1880

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.701e-16   9.024e-02   0.000  1.00000
Agriculture -3.691e-01   1.294e-01  -2.854  0.00662 **
Education   -8.253e-01   1.199e-01  -6.881 1.91e-08 ***
Catholic     4.848e-01   1.006e-01   4.817 1.84e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6186 on 43 degrees of freedom
Multiple R-squared:  0.6423,    Adjusted R-squared:  0.6173
F-statistic: 25.73 on 3 and 43 DF,  p-value: 1.089e-09
```

Введем в нашу модель логарифмы всех регрессоров и посмотрим результат, но сразу мы это сделать не можем, т.к. у нас после нормализации появились отрицательные значения. Тогда сделаем так: посмотрим с помощью функции `describe` минимальные значения в каждом столбце и прибавим ко всем данным этой переменной округленный в большую сторону модуль этого числа (чтобы не было 0, а, следовательно, и минус бесконечности в логарифме), и это несильно повлияет на нашу модель, т.к. она будет отличаться от стандартной на константу. После данной операции строим `model2` и видим, что это никак не повлияло на качество модели. Теперь спокойно можно вводить логарифмы и смотреть результат:  $R^2$  немного увеличился и

стал равен 0.6492, adj R<sup>2</sup>, наоборот, уменьшился и стал 0.5966; p-значение всех коэффициентов кроме свободного и при Education ужасны; все vif больше 5. Такая модель нам не подходит.

```
Call:
lm(formula = Fertility ~ Agriculture + Education + Catholic +
    log(Agriculture) + log(Education) + log(Catholic), data = d)

Residuals:
    Min       1Q   Median       3Q      Max
-1.24781 -0.49876  0.06172  0.43715  1.10392

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    4.05822    0.66836   6.072 3.74e-07 ***
Agriculture   -0.39873    0.23347  -1.708  0.09542 .
Education     -0.86388    0.25352  -3.408  0.00151 **
Catholic       0.66599    0.40098   1.661  0.10456
log(Agriculture) 0.06216    0.28363   0.219  0.82764
log(Education)  0.12650    0.24839   0.509  0.61336
log(Catholic)  -0.13003    0.28078  -0.463  0.64581
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6351 on 40 degrees of freedom
Multiple R-squared:  0.6492,    Adjusted R-squared:  0.5966
F-statistic: 12.34 on 6 and 40 DF,  p-value: 8.151e-08
```

В сумме логарифмы ничего дельного не показали. Попробуем добавить их по очереди(model4, model5, model6), но и это не дает результат, тогда заменим сами регрессоры на их логарифмы(spec\_model), но и здесь ничего интересного. Таким образом, введенные логарифмы ничего нам не дали, все модели с ними плохи и сравнивать их нет смысла, на данный момент лучшая модель - линейная. Теперь попробуем другой способ улучшения характеристик построенной регрессии, а именно прибавление всевозможных квадратов и попарных произведений. Произведения в языке R вводятся с помощью оператора I(), в переменные которого записываются множители. Первым введём произведение регрессоров Agriculture и Education - это model7. Её характеристики: R<sup>2</sup> = 0.6426, adj R<sup>2</sup> = 0.6085; p-значения регрессора I(Agriculture \* Education) и Agriculture 0 и 1 звезда, остальные по 3 звезды, все vif < 2.6. Это неплохая модель, но мне всё-таки кажется, что линейная лучше.

```

Call:
lm(formula = Fertility ~ Agriculture + Education + Catholic +
    I(Agriculture * Education), data = d)

Residuals:
    Min       1Q   Median       3Q      Max
-1.2336 -0.5081  0.1007  0.4649  1.1720

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    4.24384    0.37832   11.218 3.26e-14 ***
Agriculture   -0.38183    0.14667   -2.603  0.0127 *
Education     -0.84004    0.14373   -5.845 6.64e-07 ***
Catholic       0.48676    0.10232    4.757 2.33e-05 ***
I(Agriculture * Education) 0.01866    0.09746    0.191  0.8491
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6257 on 42 degrees of freedom
Multiple R-squared:  0.6426,    Adjusted R-squared:  0.6085
F-statistic: 18.88 on 4 and 42 DF,  p-value: 6.002e-09

```

Модели model8, model10, model11, model12 показали линейную зависимость между регрессорами и произведениями регрессоров, так что они однозначно не подходят (в model10, model11, model12 использовались квадраты регрессоров). Осталась последняя модель, которую я не упомянул, - model9. Она получилась достаточно противоречивой, поэтому я оставил её напоследок. Характеристики:  $R^2 = 0.6592$ ,  $\text{adj } R^2 = 0.6267$ ; p-значения I(Education \* Catholic) - 0 звезд, остальные - по 2-3 звезды; vif получился таким: Education = 5.414695, I(Education \* Catholic) = 5.832477. В принципе, в данной модели смущает лишь коэффициент мультиколлинеарности, но он в окрестности 5, так что я бы за него не беспокоился.

```

Call:
lm(formula = Fertility ~ Agriculture + Education + Catholic +
    I(Agriculture^2), data = d)

Residuals:
    Min       1Q   Median       3Q      Max
-1.3021 -0.4502  0.1161  0.3993  1.1106

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.84401    0.53045    7.247 6.46e-09 ***
Agriculture     0.03788    0.41508    0.091  0.928
Education     -0.79155    0.12422   -6.372 1.16e-07 ***
Catholic       0.50654    0.10276    4.929 1.34e-05 ***
I(Agriculture^2) -0.09679    0.09380   -1.032  0.308
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6182 on 42 degrees of freedom
Multiple R-squared:  0.6511,    Adjusted R-squared:  0.6179
F-statistic: 19.59 on 4 and 42 DF,  p-value: 3.658e-09

```

Итого, лучшие модели с внесенными туда произведениями - model9, model7(их доля общей дисперсии зависимой переменной максимальны, а также хорош vif). Также, я бы отметил, что model7 - немного худшая версия линейной model1.

## Заключение

Таким образом, лучшие модели в нашем исследовании - model9 и model1.

## Задание 2.

### Условие задачи:

Прочитайте данные, выберите столбцы, которые Вам кажутся необходимыми, чтобы описать социально-экономическое положение граждан Российской Федерации. Минимальный набор параметров: зарплата, пол, семейное положение, наличие высшего образования, возраст, тип населенного пункта, длительность рабочей недели. Из параметра, отвечающего семейному положению, сделать дамми-переменные: 1) переменная *wed1* имеет значение 1 в случае, если респондент женат, 0 – в противном случае; 2) *wed2*=1, если респондент разведён или вдовец; 3) *wed3* = 1, если респондент никогда не состоял в браке; 4) если считаете необходимым, введите другие параметры. Следите за мультиколлинеарностью (убедитесь в её отсутствии, оценив вспомогательную регрессию любого параметра (например, зарплату или одного из параметров *wed*) на эти переменные и используя команду *VIF* для неё). Из параметра *пол* сделайте переменную *sex*, имеющую значение 1 для мужчин и равную 0 для женщин. Из параметра, отвечающего типу населённого пункта, создайте одну дамми-переменную *city\_status* со значением 1 для города или областного центра, 0 – в противоположном случае. Введите один параметр *higher\_educ*, характеризующий наличие полного высшего образования. Факторные переменные, «имеющие много значений», такие как: зарплата, длительность рабочей недели и возраст, - необходимо преобразовать в вещественные переменные и нормализовать их: вычесть среднее значение по этой переменной, разделить её значения на стандартное отклонение.

- 1) Постройте линейную регрессию зарплаты на все параметры, которые Вы выделили из данных мониторинга. Не забудьте оценить коэффициент вздутия дисперсии *vif*.
- 2) Поэкспериментируйте с функциями вещественных параметров: используйте логарифм и степени (хотя бы от 0.1 до 2 с шагом 0.1).
- 3) Выделите наилучшие модели из построенных: по значимости параметров, включённых в зависимости, и по объяснённой с помощью построенных зависимостей разбросу  $\text{adj } R^2$ .
- 4) Сделайте вывод о том, какие индивиды получают наибольшую зарплату.
- 5) Оцените регрессии для подмножества индивидов, указанных в варианте.

Подмножество индивидов для моего варианта:

- городские жители, состоящие в браке;
- разведенные без высшего образования.

## Краткий обзор:

В данном задании я продолжаю работать на языке программирования R, теперь мне предстоит исследовать данные волны мониторинга экономического положения и здоровья населения РФ (данные обследования РМЭЗ НИУ ВШЭ). Основная цель данной работы - сформировать представление об индивиде, являющимся гражданином Российской Федерации, который получает наибольшую зарплату путем построения линейной регрессии, а затем оценить модель для определенных категорий индивидов. Использованные библиотеки: “lmtest”, “rmls”, “dplyr”, “Gally”, “car”, “sandwich”, “Hmisc”. Код к программе в Приложении 2.

## Решение задачи:

Прочитаем наш файл, запишем его в data, выделим нужные нам регрессоры в отдельную таблицу d и попытаемся с ними разобраться: поменяем названия их столбцов и частично изменим значения для более удобного их восприятия. Я считаю, что параметров, представленных мне в условии задачи, вполне достаточно для построения хорошей модели. Список этих параметров и их наименование в файле:

- “tj13.2” - среднемесячная зарплата индивида за последние 12 месяцев на предприятии после вычета налогов (изменим название столбца на “salary”);
- “t\_age” - количество полных лет (изменим название столбца на “age”);
- “th5” - поле респондента (изменим название столбца на “sex”);
- “t\_educ” – подробное образование по категориям (изменим название столбца на “higher\_educ”);
- “status” - тип населённого пункта по категориям (изменим название столбца на “city\_status”);
- “tj6.2” - среднее количество часов рабочей недели (изменим название столбца на “workweek”);
- “t\_marst” - семейное положение по категориям (сделаем из данного столбца дамми-переменную, т.е. разобьем на несколько столбцов его значения).

Первым делом удалим строки с отсутствующими значениями, далее:

- 1) Создадим столбец “wed1” по размеру “t\_marst”, заполним нулями, поставим единицы там, где стоят значения два и шесть в столбце “t\_marst”. Этот регрессор будет говорить, если значение равно одному, что респондент состоит в зарегистрированном браке или официально зарегистрирован, но вместе с супругом не проживает, нулю в случаях wed2, wed3;
- 2) Создадим столбец “wed2” по размеру “t\_marst”, заполним нулями, поставим единицы там, где стоят значения четыре и пять в столбце “t\_marst” именно в местах, где говорится, что респондент не женат или вдовец. Этот регрессор будет говорить, если значение равно одному, что респондент разведен и в браке не состоит или вдовец/вдова, нулю в случаях wed1, wed3;
- 3) Создадим столбец “wed3” по размеру “t\_marst”, заполним нулями, поставим единицы там, где стоит значение один в столбце “t\_marst”; Этот регрессор будет говорить, если значение равно одному, что респондент никогда в браке не состоял, нулю в случаях wed2, wed1;

- 4) Создадим столбец “sex” по размеру “th5”, заполним нулями, поставим единицы там, где стоит значение один в столбце “th5”. Этот регрессор будет говорить, если значение равно одному, что респондент мужского пола, если значение равно нулю, то - женского пола;
- 5) Создадим столбец “city\_status” по размеру “status”, заполним нулями, поставим единицы там, где стоит значения один и два в столбце “status”. Этот регрессор будет говорить, если значение равно одному, что респондент живет в городе или областном центре, если значение равно 0, то живет в селе или ПГТ.
- 6) Создадим столбец “higher\_educ” по размеру “t\_educ”, заполним нулями, поставим единицы там, где стоит значения двадцать один, двадцать два, двадцать три в столбце “t\_educ”. Этот регрессор будет говорить, если значение равно одному, что респондент имеет высшее образование, если значение равно нулю - отсутствие высшего образования;
- 7) Создадим столбец “salary” и скопируем в него значения столбца “tj13.2”, потом преобразуем значения в недавно созданном столбце в вещественный тип и нормализуем их, вычтем среднее значение по этой переменной и разделив на стандартное отклонение;
- 8) Создадим столбец “age” и скопируем в него значения столбца “t\_age”, потом преобразуем значения в недавно созданном столбце в вещественный тип и нормализуем их, вычтем среднее значение по этой переменной и разделив на стандартное отклонение;
- 9) Создадим столбец “workweek” и скопируем в него значения столбца “tj6.2”, потом преобразуем значения в недавно созданном столбце в вещественный тип и нормализуем их, вычтем среднее значение по этой переменной и разделив на стандартное отклонение.

Теперь уже выделим созданные столбцы из d в таблицу d2 и наконец приступим к анализу данных и построению различных моделей. Первым делом построим линейную регрессию зарплаты на все параметры, которые были выделены из мониторинга, model1. Получим: 1) vif для всех регрессоров  $< 2.4$ , что говорит и линейной независимости объясняющих параметров; 2)  $R^2 = 0.175$ ,  $\text{adj } R^2 = 0.1735$ , что в пределах нормы для этого набора данных, так как мы обобщили многие параметры в нем; 3) p-значения всех аргументов по 2 – 3 звезды, кроме wed3 (0 звезд) и wed2(1 звезда). В целом, неплохо для первой модели. Теперь построим model2, содержащую те же самые параметры, кроме wed3. Получим: 1) линейную независимость всех регрессоров; 2)  $R^2 = 0.1745$ ,  $\text{adj } R^2 = 0.1732$  (что хоть и на немного, но меньше, чем в предыдущей модели); 3) p-значения всех регрессоров по 3 звезды.

```

Residuals:
    Min       1Q   Median       3Q      Max
-1.8840 -0.5223 -0.1519  0.3118 12.0786

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.75604    0.04764  -15.868 < 2e-16 ***
age          -0.07836    0.01529   -5.125 3.10e-07 ***
sex           0.50270    0.02927  17.177 < 2e-16 ***
higher_educ   0.55478    0.02984  18.591 < 2e-16 ***
city_status   0.36011    0.03120  11.542 < 2e-16 ***
workweek      0.11224    0.01415   7.935 2.66e-15 ***
wed1          0.12073    0.04271   2.827 0.00473 **
wed2          0.14035    0.05456   2.572 0.01014 *
wed3         -0.08672    0.05426  -1.598 0.11009
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9091 on 4373 degrees of freedom
Multiple R-squared:  0.175,    Adjusted R-squared:  0.1735
F-statistic: 116 on 8 and 4373 DF,  p-value: < 2.2e-16

```

Далее “поиграемся” с логарифмами, но сразу мы это сделать не можем, т.к. у нас присутствуют отрицательные значения. От них мы избавимся как в задаче №1, теперь же можно смело использовать эту математическую функцию для некатегориальных объясняющих переменных (для категориальных смысла никакого, так как будем получать либо 0, либо минус бесконечность), в том числе и для объясняемой переменной. Рассмотрим model3 (введены всевозможные логарифмы) и model4 (введены логарифмы для всех регрессоров), они нам не подходят из-за линейной зависимости между регрессорами. Из model4 стало ясно, что следует убрать регрессоры workweek и log(age), так как у них самые маленькие p-значения по сравнению с остальными объясняющими переменными. У полученной model5 следующие характеристики: 1) vif всех параметров < 1.7, что просто отлично; 2)  $R^2 = 0.1768$ ,  $\text{adj } R^2 = 0.1754$ ; 3) p-значения всех аргументов по 3 звезды. Мы получили отличную модель.

```

Residuals:
    Min       1Q   Median       3Q      Max
-1.8335 -0.5251 -0.1571  0.3148 12.0610

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.34513    0.07413   4.656 3.32e-06 ***
age          -0.06964    0.01506  -4.625 3.86e-06 ***
sex           0.49854    0.02921  17.066 < 2e-16 ***
higher_educ   0.55048    0.02972  18.524 < 2e-16 ***
city_status   0.35815    0.03116  11.493 < 2e-16 ***
wed1          0.15647    0.03380   4.629 3.78e-06 ***
wed2          0.17464    0.04877   3.581 0.000346 ***
log(workweek) 0.41614    0.04783   8.701 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9081 on 4374 degrees of freedom
Multiple R-squared:  0.1768,    Adjusted R-squared:  0.1754
F-statistic: 134.2 on 7 and 4374 DF,  p-value: < 2.2e-16

```



Теперь попробуем улучшить нашу модель прибавлением различных произведений, но и это нам особо не помогает (model6, model7, model8). Таким образом, наши лучшие модели - model5, model2

Во время работы с логарифмами я заметил одну интересную особенность, что если от объясняемой модели брать логарифм, то понижается р-значение некоторых регрессоров (несильно), но возрастает  $R^2$  и adj  $R^2$ , попробуем проверить на практике. Построим специальную модель spec\_model и посмотрим её характеристики: 1)  $R^2 = 0.1974$ , adj  $R^2 = 0.1961$  (уже лучше, чем наши предыдущие модели); 2) модель линейно независима; 3) р-значения всех аргументов по 3 звезды, кроме wed1 и wed2, у этих по 2. Однозначно, эта наша лучшая модель!

```
Residuals:
    Min       1Q   Median       3Q      Max
-2.7070 -0.3096  0.0115  0.3385  2.3111

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.478954   0.041427  -11.562 < 2e-16 ***
age          -0.050834   0.009215   -5.516 3.66e-08 ***
sex           0.346430   0.017928  19.323 < 2e-16 ***
higher_educ   0.323261   0.018264  17.700 < 2e-16 ***
city_status   0.250875   0.019115  13.124 < 2e-16 ***
workweek      0.088260   0.008665  10.186 < 2e-16 ***
wed1          0.060366   0.020723   2.913 0.00360 **
wed2          0.085338   0.029913   2.853 0.00435 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5571 on 4374 degrees of freedom
Multiple R-squared:  0.1974,    Adjusted R-squared:  0.1961
F-statistic: 153.7 on 7 and 4374 DF,  p-value: < 2.2e-16
```

Теперь можно и обратиться и к главному вопросу этого задания: какой индивид, судя по всем данным, получает наибольшую зарплату. Для этого рассмотрим уравнение нашей лучшей модели:

$$\log(\text{salary}) = -0.478954 - 0.050834 \cdot \text{age} + 0.346430 \cdot \text{sex} + 0.323261 \cdot \text{higher\_educ} + 0.250875 \cdot \text{city\_status} + 0.088260 \cdot \text{workweek} + 0.060366 \cdot \text{wed1} + 0.085338 \cdot \text{wed2}.$$

Но есть один нюанс с  $\log(\text{salary})$ . Так как для того, чтобы взять логарифм от данной категории после нормализации мы прибавили к столбцу значений 1.5, то есть реально мы описывали переменную  $\log(\text{salary}+1.5)$ , где salary - значение после нормализации, но так как преобразование линейное, то мы просто должны будем вычесть из свободного члена константу, но это не повлияет на общий вид картины описания индивидов, поэтому имеет смысл не обращать внимание на эту константу.

## Заключение

Из построенной модели видно, что на данный момент больше всего заработает мужчина(sex = 1) с высшим образованием(higher\_educ = 1), проживающий в городе или областном центре;

$$\log(\text{salary}) = 0.441612 - 0.050834 \cdot \text{age} + 0.088260 \cdot \text{workweek} + 0.060366 \cdot \text{wed1} + 0.085338 \cdot \text{wed2}.$$

Поймем, что больше зарабатывать будет мужчина с высшим образованием, разведенный, или не состоящий в браке, или вдовец ( $\text{wed2} = 1 \Rightarrow \text{wed1} = 0$ ,  $\text{wed2}$  на немного, но дает больший вклад):

$$\text{salary} = 0.52695 - 0.050834 \cdot \text{age} + 0.088260 \cdot \text{workweek}$$

Данный мужчина также должен перерабатывать и должен быть относительно времени получения высшего образования быть молодым. Именно тогда он будет зарабатывать больше всех!!!

Теперь оценим регрессию для подмножества индивидов, которые даны мне в варианте:

- городские жители, состоящие в браке

$$\log(\text{salary}) = -0.167713 - 0.050834 \cdot \text{age} + 0.346430 \cdot \text{sex} + 0.323261 \cdot \text{higher\_educ} + 0.088260 \cdot \text{workweek}$$

Такая будет зависимость для первой категории индивидов. Индивид будет получать больше, т.к. он проживает в городе или областном центре, но и немного потеряет в зарплате, т.к. он находится в браке ( $\text{wed1} = 1 \Rightarrow \text{wed2} = 0$ , а это потеря, т. к. коэф. при  $\text{wed2} >$  коэф. при  $\text{wed1}$ ).

- разведенные, без высшего образования

$$\log(\text{salary}) = -0.393616 - 0.050834 \cdot \text{age} + 0.346430 \cdot \text{sex} + 0.250875 \cdot \text{city\_status} + 0.088260 \cdot \text{workweek}$$

Такая будет зависимость для второй категории индивидов. Индивид будет меньше получать, чем мог бы, т. к. он очень сильно теряет часть баллов к свободному члену из-за отсутствия высшего образования ( $\text{higher\_educ} = 0$ ), но немного отыгрывает (хотя очень не сильно) из-за того, что он разведен ( $\text{wed2} = 1 \Rightarrow \text{wed1} = 0$ ).

## Задание 3 и 4.

### Условие задачи:

Построение классификатора вин по набору данных “Red Wine Quality”.

### Краткий обзор:

В этом задании я должен поменять язык программирования с R на Python, провести анализ и обработку данных, а также построить классификатор несколькими способами и сравнить полученные результаты. Как дополнительное задание, я должен проделать все то же самое для ненормированных данных и сравнить результаты.

Использованные библиотеки языка программирования и их цели при работе:

- numpy – для работы со значениями в столбцах датасета;
- pandas – для считывания данных с csv-файла;
- matplotlib.pyplot – для визуализации зависимостей данных;
- warnings - для блокировки отображений предупредительных сообщений;
- scikit-learn и его различные классы – для использования большинства алгоритмов машинного обучения.

Код к программе в Приложении 3.

## Решение задачи:

Для начала прочтем матрицу с данными из файла и посмотрим её размер. С помощью команды `data.shape` получаем кортеж (1599, 12), говорящий о том, что в датасете содержится 1599 строк (объектов) и 12 столбцов (признаков), в этом также можно убедиться, посмотрев несколько первых и последних строк нашего файла:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

Теперь получим информацию о признаках датасета (типов данных в столбцах и значение признака):

- fixed acidity - вещественный тип, фиксированная кислотность;
- volatile acidity - вещественный тип, летучая кислотность;
- citric acid - вещественный тип, лимонная кислота;
- residual sugar - вещественный тип, остаточный сахар;
- chlorides - вещественный тип, хлориды;
- free sulfur dioxide - вещественный тип, свободный диоксид серы;
- total sulfur dioxide - вещественный тип, общий диоксид серы;
- density - вещественный тип, плотность;
- pH - вещественный тип, водородный показатель;
- sulphates - вещественный тип, сульфаты;
- alcohol - вещественный тип, алкогольность;
- quality - целочисленный тип, качество (оценка от 0 до 10);

Отсюда сразу можно сделать вывод, что категориальный признак в датасете один – quality, соответственно он и является столбцом с максимальным количеством уникальных значений категориального признака. Значения здесь варьируются от 3 до 8,

то есть ни одно из вин не получило наилучшую и наихудшую оценки. Бинарных признаков в датасете также не наблюдаю.

Далее посмотрим информацию именно по значениям в столбцах данных. Исходя из проделанных операций, пропусков и различных аномальных значений в данных не наблюдается.

Теперь, после первичного анализа, можно приступить к непосредственной работе с данными:

- Единственный категориальный признак сделаем бинарным. Почему я решил, что разделение значений на две части будет лучше? Ответ очень прост: у нас маленький разброс оценок качества вина, и, разделив именно на две категории, мы лучше классифицируем наши данные (это было сделано к 4 задаче). Получаем, что вин нехорошего качества (со значением 0 в столбце) больше 75%.
- Нормируем все столбцы данных, кроме столбца 'quality' (так как он категориальный), с помощью стандартного отклонения. По полученным данным столбец с максимальным средним значением после нормировки признаков – столбец 'pH' и, как нетрудно догадаться, целевой признак – качество вина или столбец 'quality', а остальные столбцы это данные, по которым будем предсказывать целевой признак.

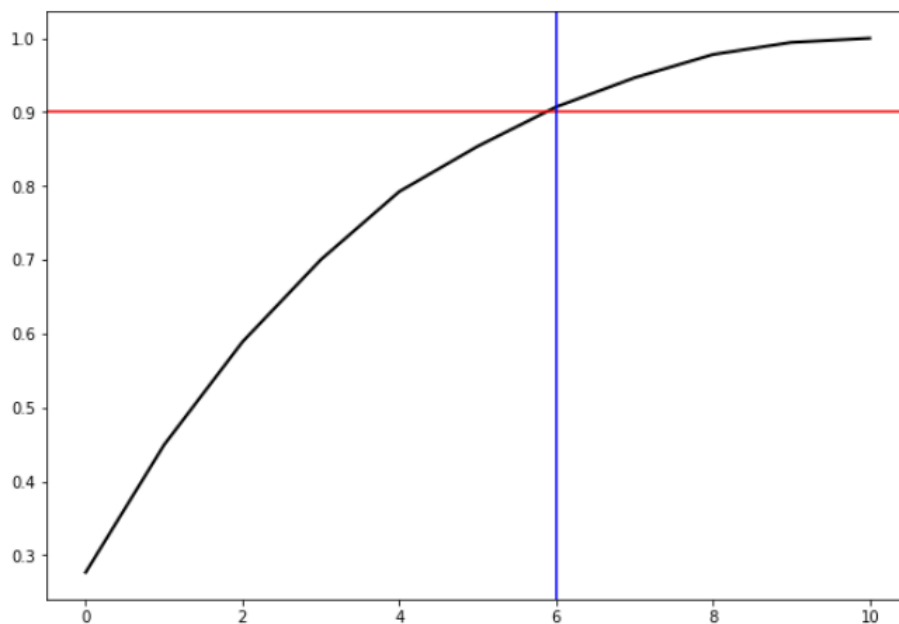
	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates
count	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03
mean	3.435512e-16	1.699704e-16	4.335355e-16	-1.905223e-16	4.838739e-16	1.432042e-16	1.289532e-16	-3.482795e-14	3.002879e-15	7.639596e-16
std	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00
min	-2.137045e+00	-2.278280e+00	-1.391472e+00	-1.162696e+00	-1.603945e+00	-1.422500e+00	-1.230584e+00	-3.538731e+00	-3.700401e+00	-1.936507e+00
25%	-7.007187e-01	-7.699311e-01	-9.293181e-01	-4.532184e-01	-3.712290e-01	-8.487156e-01	-7.440403e-01	-6.077557e-01	-6.551405e-01	-6.382196e-01
50%	-2.410944e-01	-4.368911e-02	-5.636026e-02	-2.403750e-01	-1.799455e-01	-1.793002e-01	-2.574968e-01	1.760083e-03	-7.212705e-03	-2.251281e-01
75%	5.057952e-01	6.266881e-01	7.652471e-01	4.341614e-02	5.384542e-02	4.901152e-01	4.723184e-01	5.768249e-01	5.759223e-01	4.240158e-01
max	4.355149e+00	5.877976e+00	3.743574e+00	9.195681e+00	1.112703e+01	5.367284e+00	7.375154e+00	3.680055e+00	4.528282e+00	7.918677e+00

	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
399000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1599.000000
699704e-16	4.335355e-16	-1.905223e-16	4.838739e-16	1.432042e-16	1.289532e-16	-3.482795e-14	3.002879e-15	7.639596e-16	9.437243e-16	0.135710	0.135710
100313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	1.000313e+00	0.342587	0.342587
278280e+00	-1.391472e+00	-1.162696e+00	-1.603945e+00	-1.422500e+00	-1.230584e+00	-3.538731e+00	-3.700401e+00	-1.936507e+00	-1.898919e+00	0.000000	0.000000
699311e-01	-9.293181e-01	-4.532184e-01	-3.712290e-01	-8.487156e-01	-7.440403e-01	-6.077557e-01	-6.551405e-01	-6.382196e-01	-8.663789e-01	0.000000	0.000000
368911e-02	-5.636026e-02	-2.403750e-01	-1.799455e-01	-1.793002e-01	-2.574968e-01	1.760083e-03	-7.212705e-03	-2.251281e-01	-2.093081e-01	0.000000	0.000000
266881e-01	7.652471e-01	4.341614e-02	5.384542e-02	4.901152e-01	4.723184e-01	5.768249e-01	5.759223e-01	4.240158e-01	6.354971e-01	0.000000	0.000000
377976e+00	3.743574e+00	9.195681e+00	1.112703e+01	5.367284e+00	7.375154e+00	3.680055e+00	4.528282e+00	7.918677e+00	4.202453e+00	1.000000	1.000000

После этих двух действий можно сказать, что чистку и нужную обработку данных мы уже сделали, так что теперь можно выделить тренировочную (обучающую), на которой будет строить модель, и тестовую, на которой будем проверять качество построенной модели, выборки. При этом при использовании `train_test_split` с параметрами `test_size = 0.3`, `random_state = 42` в тренировочную выборку попадают 1119 объектов, а в тестовую 480. Теперь воспользуемся готовой функцией PCA, т.е. методом главных компонент, которая поможет выделить набор тех параметров, которые лучше всего описывают наш набор данных. После применения его стало вполне ясно, что для объяснения 90 % дисперсии достаточно 7 признаков, а наибольший вклад в первую компоненту PCA вносят столбцы данных 'fixed acidity' и 'citric acid' (у них множители по 0.478).

1 component: 27.68% of initial variance

$0.478 \times \text{fixed acidity} + -0.255 \times \text{volatile acidity} + 0.478 \times \text{citric acid} + 0.168 \times \text{residual sugar} + 0.214 \times \text{chlorides} + -0.028 \times \text{free sulfur dioxide} + 0.030 \times \text{total sulfur dioxide} + 0.376 \times \text{density} + -0.437 \times \text{pH} + 0.250 \times \text{sulphates} + -0.088 \times \text{alcohol}$



После определённого количества стандартных операций над датасетом, мы наконец пришли к творческому заданию. Для построения классификации будем пользоваться несколькими различными классификаторами и смотреть их стандартную ошибку на тестовой и обучающей выборке. Стоит отметить, результат ошибки на тестовой выборке находится в приоритете, т. к. построенная модель сталкивается с новыми данными, которые при обучении недоступны, и её качества проявляются на новых данных, поэтому в основном будем смотреть её.

Методы классификации:

1. метод k-ближайших соседей(5, 9).

- Суть

Для каждого нового объекта алгоритм ищет в обучающей выборке k наиболее близких объекта и относит новый объект к тому классу, которому принадлежит большинство из них, количество соседей k соответствует параметру `n_neighbors`. По умолчанию, `n_neighbors = 5`.

- Наши результаты

Ошибки на обучающей модели = 9%, на тестовой = 12.9%. Основным параметр метода - k, попробуем с ним поработать. Найдем наилучшее значение k от 1 до 11. Это 4, при этом ошибка кросс-валидации (7)(кросс-валидация, которую ещё называют перекрестной проверкой - это техника оценки модели для проверки того, насколько успешно применяемый в модели статистический анализ способен работать на независимом наборе данных) составляет 12.4 процентов, что немного ниже ошибки на тестовой модели. После работы алгоритма с наилучшим k получаем ошибку тестовой модели в 11.85 %, что является пока что лучшим результатом.

## 2. метод Random Forest(5)

- Суть

Данный алгоритм строит ансамбль (- набор предсказателей, вместе дающих ответ) независимых случайных деревьев, каждое из которых обучается на выборке, полученной из исходной с помощью процедуры изъятия с возвращением, а затем голосованием по большинству выбирается лучшие ответы.

- Наши результаты

Получили ошибку на тестовой выборке в 11.25 процентов, что чуть лучше прежних результатов. На тренировочной выборке мы получили ошибку в 0%, тогда может возникнуть проблема переобучения, но т. к. наша ошибка тестовой выборки очень хороша, проблема переобучения отпадает.

## 3. метод GBT – градиентный бустинг деревьев решений(5, 6)

- Суть

Данная техника машинного обучения строит модель предсказания в форме ансамбля слабых предсказывающих моделей, обычно деревьев решений, с целью определить функцию потерь (функция, характеризующая потери при неправильном принятии решений на основе наблюдаемых данных) и минимизировать её.

- Наши результаты

Ошибка на тестовой выборке в 13.9 %, что хуже, чем при других методах.

## 4. метод SVC - support vector classifier(5, 8)

- Суть

Перевод исходных векторов в пространство более высокой размерности и поиск разделяющей гиперплоскости с максимальным зазором в этом пространстве.

- Наши результаты

- радиальное ядро

Для стандартного набора параметров метода мы получили ошибку тестовой выборки в 11.6 %, что является хорошим результатом, но не лучшим. Попробуем подобрать наилучшие параметры для радиального ядра: best C = 10.0, best gamma = 1.0. При этих параметрах процент ошибки равен 10.2 %, что является лучшим результатом.

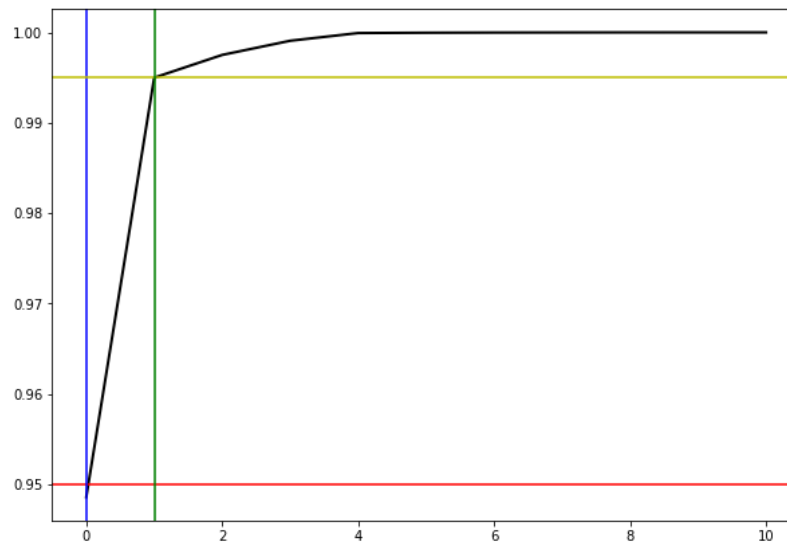
- линейное ядро

Для стандартного набора параметров метода мы получили ошибку тестовой выборки в 13.9 %. Наилучшие параметры мы подобрать не смогли, видимо, эта задача не решается.

- полиномиальное ядро

Для стандартного набора параметров метода мы получили ошибку тестовой выборки в 12.08 %. Наилучшие параметры мы также подобрать не смогли, эта задача не решается.

Теперь к дополнительному заданию. Используя метод главных компонент для данных без нормирования, получаем, что для описания 95% данных нам достаточно 1 признака, а именно 'total sulfur dioxide', а для описания 99.5% признаков нам достаточно 2 признаков: 'total sulfur dioxide' и 'free sulfur dioxide'.



Далее используем те же методы и посмотрим ошибку на тестовой выборке:

#### 1. метод k-ближайших соседей

Первичная ошибка на тестовой выборке 14.16 %, после подбора параметров (лучший  $k = 2$ ) получаем результат ещё хуже: 14.3 %.

#### 2. метод Random Forest

Ошибка тестовой выборки составила 11.04 %, что вполне себе отличный результат (пока второй в общем зачёте).

#### 3. метод GBT

Здесь получился очень интересный результат, ведь ошибка оказалась такой же, как и в методе случайного леса, т. е. 11.04 %.

#### 4. метод SVC

В данном случае, нам опять предстоит рассматривать несколько ядер:

- радиальное

Первичная ошибка = 13.75%, а после подбора параметров, где  $\text{best } C = 1.0$ ,  $\text{best } \gamma = 10.0$ , мы получили ошибку в 11.04 %, как и в других двух методах, что очень интересно!

- линейное

Ошибка составила 13.95 %, подбор параметров прошёл безуспешно.

- полиномиальное

Здесь процент ошибки не смог посчитаться, эта часть задачи не решается для данного ядра.



## **Заключение**

Проведя, классификацию вин с нормализованными данными мы получили лучшую ошибку тестовой выборки в 10.2 %, что свидетельствует тому, что мы классифицировали 89.8 % данных, что вполне неплохо! А в случае то же самой работы, но без нормализации, мы несколькими методами описали 88.96 % данных.

## **Общий Вывод**

### **Задача 1.**

В первой задаче практике нужно было построить линейную модель с помощью метода наименьших квадратов. Я считаю, что с задачей я справился, выделив две хорошие модели с  $R^2$  примерно в 65%, с  $\text{adj } R^2$  примерно 61 %.

### **Задача 2.**

С этой задачей всё более интересно, так как мы получали вполне себе реальный ответ настоящего исследования. Так мы примерно определили стиль жизни человека, который получает наибольшую заработную плату. Им оказался неженатый мужчина или вдовец с высшим образованием, проживающий в городе или областном центре, молодого возраста относительно получения диплома и перерабатывающий на работе – вполне себе реальный ответ.

### **Задача 3-4.**

На мой взгляд, это была самая интересная задача, так как в ней была творческая составляющая. Сделав относительно стандартную работу в 3 задаче, мы перешли к креативной части всей практики. Мы использовали несколько методов классификации и получили внушительные 89.8 % процентов описанных данных, что вполне неплохо, для первой работы в этой области программирования.

В целом, практика мне очень понравилась, я получил бесценный опыт. Мой преподаватель однажды сказал: «Попробуйте, пока молоды, все сферы программирования, определите свою нишу и будете получать удовольствие от своей работы!!!», так что, познакомившись с анализом данных, я узнал еще одну большую область своей будущей профессии и «попробовал её на вкус»!

## Список литературы:

1. //Метод наименьших квадратов - 2012 год.

<http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D0%BD%D0%B0%D0%B8%D0%BC%D0%B5%D0%BD%D1%8C%D1%88%D0%B8%D1%85%D0%BA%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82%D0%BE%D0%B2>

2. // Линейная регрессия: насколько хорошо построенная модель описывает данные? - 2014 год.

[https://r-analytics.blogspot.com/2014/05/blog-post\\_18.html#.XPoxoxYzblU](https://r-analytics.blogspot.com/2014/05/blog-post_18.html#.XPoxoxYzblU)

3.// Протокол разведочного анализа данных: выявление коллинеарности - 2012 год

<https://r-analytics.blogspot.com/2012/07/blog-post.html#.XPox3hYzblU>

4. // р-значение - 2016 год

<http://datascientist.one/p-value/>

5.// Руководство для начинающих - 2017 год

<https://mlbootcamp.ru/article/tutorial/>

6. // Градиентный бустинг - просто о сложном - 2018 год

<https://neurohive.io/ru/osnovy-data-science/gradientyi-busting/>

7. // Скользящий контроль - 2010 год

<http://www.machinelearning.ru/wiki/index.php?title=%D0%A1%D0%BA%D0%BE%D0%BB%D1%8C%D0%B7%D1%8F%D1%89%D0%B8%D0%B9%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C>

8.// Машина опорных векторов - 2018 год

<http://www.machinelearning.ru/wiki/index.php?title=SVM>

9. // Метод ближайших соседей - 2015 год

<http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D0%B1%D0%BB%D0%B8%D0%B6%D0%B0%D0%B9%D1%88%D0%B5%D0%B3%D0%BE%D1%81%D0%BE%D1%81%D0%B5%D0%B4%D0%B0>

## Приложение 1.

#Чтобы программа заработала, надо пробежаться с начала с помощью ctrl + Enter

```
library("psych")
```

```
library("dplyr")
```

```
library("ggplot2")
```

```
library("GGally")
```

```
library("car")
```

#посмотрим на набор данных

```
data = swiss
```

data #выводит сам датасет в консоль сп.1

```
help(data) #выводит информацию по датасету
```

```
glimpse(data) #выводит сам датасет в консоль сп.2
```

#выберем лишь те переменные, с которыми будем работать

```
d = select(data, Fertility, Agriculture, Education, Catholic)
```

```
d
```

#№1

#Нормализуем данные, вычтя из каждого столбца среднее значение mean(x) и

#поделив на среднеквадратическое отклонение  $\sigma \sim \sqrt{\text{var}(x)}$ , где x – столбец

#данных

```
d[1] = (d[1] - mean(d$Fertility)) / sqrt(var(d$Fertility))
```

```
d[2] = (d[2] - mean(d$Agriculture)) / sqrt(var(d$Agriculture))
```

```
d[3] = (d[3] - mean(d$Education)) / sqrt(var(d$Education))
```

```
d[4] = (d[4] - mean(d$Catholic)) / sqrt(var(d$Catholic))
```

```
d
```

```
describe(d)
```

```
summary(d)
```

#### #№2

#Проверить, что в наборе данных нет линейной зависимости, если же она есть,  
#то исключить соответствующее значение(смотреть по  $R^2$ )

#Построим модели зависимости между регрессорами и оценим их  $R^2$

```
m1 = lm(Fertility~Agriculture+Education+Catholic, d)
```

```
summary(m1)
```

#  $R^2$  примерно равен 64 %, что говорит о лин. независимости регрессоров

# посмотрим vif регрессоров

```
vif(m1)
```

#  $vif < 2.1$ , что и подтверждает нашу догадку

#### #№3

#Построить линейную модель зависимой переменной от указанных в варианте

#регрессоров по методу наименьших квадратов. Оценить удачность модели согласно:

#1)  $R^2$ , 2) p-значениям каждого коэффициента

```
model1 = lm(Fertility~Agriculture+Education+Catholic, d)
```

```
model1
```

```
summary(model1)
```

# Из данных получаем, что multiple R-squared = 0.6423, adjusted R-squared = 0.6173;

# p-значения каждого коэффициента хороши(2-3 звездочки), за исключением  
свободного члена,

# объединяющий все остальные регрессоры, так что убрать мы его не можем.

# В целом, для первой модели результат неплох!

#### #№4

# Ввести в модель логарифмы регрессоров. Сравнить модели и выбрать наилучшую.

#Так как при начальной нормализации данных выяснилось, что там есть отрицательные значения,

#мы не можем логарифмировать регрессоры. Но можно к столбцам с отрицательными значениями

#прибавить минимальное число для приведения их к положительным.

describe(d)#отсюда узнаем мин отриц. зн-я и добавляем к столбцам их округленное

$d[2] = d[2] + 2.2$

$d[3] = d[3] + 1.1$

$d[4] = d[4] + 1$

$d[1] = d[1] + 3$

d

model2 = lm(Fertility~Agriculture+Education+Catholic, d)

summary(model2)

vif(model2)

# по сравнению с model1 ничего не изменилосью

model3 =

lm(Fertility~Agriculture+Education+Catholic+log(Agriculture)+log(Education)+log(Catholic), d)

summary(model3)

vif(model3)

#Multiple R-squared = 0.6492(немного увеличился), Adjusted R-squared = 0.5966(уменьшился).

#P-значение всех коэффициентов кроме свободного и Education ужасны.

#Vif всех коэффициентов больше 5, что говорит лин. зависимости между ними.

#Попробуем еще модели(возьмем те, где vif меньше)

model4 = lm(Fertility~Agriculture+Education+Catholic+log(Agriculture), d)

summary(model4)

```
vif(model4)
```

```
# Данная модель чуть лучше предыдущей, т. к. p-значения коэффициентов все хороши(по три звезды) кроме
```

```
# Agriculture(1 звезда) и log(Agriculture)(0 звезд), что говорит о ненужности логарифма. Vif сильно уменьшились(все < 5),
```

```
# но vif(Agriculture) и vif(log(Agriculture)) около 5.
```

```
# Multiple R-squared: 0.6448, Adjusted R-squared: 0.611.
```

```
model5 = lm(Fertility~Agriculture+Education+Catholic+log(Education), d)
```

```
summary(model5)
```

```
vif(model5)
```

```
# Эта модель практически ничем не отличается от предыдущей.
```

```
model6 = lm(Fertility~Agriculture+Education+Catholic+log(Catholic), d)
```

```
summary(model6)
```

```
vif(model6)
```

```
# Catholic и log(Catholic) сильно лин зависимы.
```

```
spec_model = lm(Fertility~log(Agriculture)+Education+Catholic, d)
```

```
summary(spec_model)
```

```
vif(spec_model)
```

```
# введение логарифмов вместо самих регрессоров также ничего не дало
```

```
# Введение логарифма сильно полезного ничего не дало, линейная модель лучше.
```

```
#5
```

```
# Ввести в модель всевозможные произведения из пар регрессоров, в том числе
```

```
# квадраты регрессоров. Найдите одну или несколько наилучших моделей по доле
```

```
# объяснённого разброса в данных  $R^2$ 
```

```
summary(model2)$r.squared # = 0.6422541
```

```
model7 = lm(Fertility~Agriculture+Education+Catholic+I(Agriculture*Education), d)
summary(model7)
vif(model7)
# Multiple R-squared: 0.6426,      Adjusted R-squared: 0.6085
# Р-значения регрессора I(Agriculture * Education) и Agriculture 0 и 1 звезда
# соответственно, остальные по три звезды. Все vif < 2.6.
# В целом введение множителя I(Agriculture*Education) ничего особо ценного не дало.
```

```
model8 = lm(Fertility~Agriculture+Education+Catholic+I(Agriculture*Catholic), d)
summary(model8)
vif(model8)
# Vif(I(Agriculture * Catholic)) = 16.170904, vif(Catholic) = 12.076665, модель не
подходит.
# Multiple R-squared: 0.6426,      Adjusted R-squared: 0.6086
```

```
model9 = lm(Fertility~Agriculture+Education+Catholic+I(Education*Catholic), d)
summary(model9)
vif(model9)
# Модель получилась достаточно противоречивой.
# С одной стороны, Multiple R-squared: 0.6592, Adjusted R-squared: 0.6267, с другой
# р-значения I(Education * Catholic) - 0 звезд, vif(Education) = 5.414695,
# I(Education * Catholic) = 5.832477.
```

```
model10 = lm(Fertility~Agriculture+Education+Catholic+I(Agriculture^2), d)
summary(model10)
vif(model10)
# vif(Agriculture) = 20.740242, vif(I(Agriculture^2)) = 18.703768, модель не подходит.
# Multiple R-squared: 0.6511,      Adjusted R-squared: 0.6179.
```

```
model11 = lm(Fertility~Agriculture+Education+Catholic+I(Education^2), d)
```



```
summary(model11)
vif(model11)
# vif(Education) = 10.392245, vif(I(Education^2)) = 8.540909, модель не подходит.
# Multiple R-squared: 0.6437,      Adjusted R-squared: 0.6098.

model12 = lm(Fertility~Agriculture+Education+Catholic+I(Catholic^2), d)
summary(model12)
vif(model12)
# vif(Catholic) = 76.136499, vif(I(Catholic^2)) = 81.211267, модель не подходит.
# Multiple R-squared: 0.6527,      Adjusted R-squared: 0.6196

# Итого, лучшая модели по доле разброса в данных  $R^2$  - model7, model9.
# Model7 - немного худшая версия model1.
# Model9 - противоречивая модель, но прирост  $R^2$  происходит(на примерно 1%).
```

## Приложение 2.

```
library("lmtest")

library("rlms")

library("dplyr")

library("GGally")

library("car")

library("sandwich")

library("Hmisc")


data =
=rlms_read("C:\\Users\\Mishan\\Desktop\\Программирование\\Практика\\r24i_os26c.sav")

glimpse(data)

d = select(data, tj13.2, t_age, th5, t_educ, status, tj6.2, t_marst)


#исключаем строки с отсутствующими значениями NA
d = na.omit(d)


# Изменим названия столбцов и преобразуем их значения

# семейное положение
d["wed1"]=d$t_marst
d["wed1"]=0

# поставим 1, если респондент женат
d$wed1[which(d$t_marst == '2')] = 1 # состоит в зарегистрированном браке
d$wed1[which(d$t_marst == '6')] = 1 # официально зарегистрированы, но вместе не
проживают
d$wed1 = as.numeric(d$wed1)


d["wed2"]=d$t_marst
d["wed2"]=0
d$wed2[which(d$t_marst == '4')] = 1 # разведен и в браке не состоит
d$wed2[which(d$t_marst == '5')] = 1 # вдовец(вдова)
```

```
d$wed2 = as.numeric(d$wed2)
```

```
d["wed3"]=d$t_marst
```

```
d["wed3"]=0
```

```
d$wed3[which(d$t_marst == '1')] = 1 # никогда в браке не состоял
```

```
d$wed3 = as.numeric(d$wed3)
```

```
#пол
```

```
d["sex"]=d$th5
```

```
d$sex[which(d$sex == '2')] = 0 # женщина
```

```
d$sex[which(d$sex == '1')] = 1 # мужчина
```

```
d$sex = as.numeric(d$sex)
```

```
# населенный пункт
```

```
d["city_status"]=d$status
```

```
d["city_status"]=0
```

```
d$city_status[which(d$status == '1')] = 1 # областной центр
```

```
d$city_status[which(d$status == '2')] = 1 # город
```

```
d$city_status = as.numeric(d$city_status)
```

```
# высшее образование
```

```
d["higher_educ"]=d$t_educ
```

```
d["higher_educ"]=0
```

```
d$higher_educ[which(d$t_educ == '21')] = 1 # есть диплом о высшем образовании
```

```
d$higher_educ[which(d$t_educ == '22')] = 1 # аспирантура и т.п. без диплома
```

```
# случай 22 характеризуется тем, что есть диплом о высшем образовании, но нет диплома аспирантуры
```

```
d$higher_educ[which(d$t_educ == '23')] = 1 # аспирантура и т.п. с дипломом
```

```
d$higher_educ = as.numeric(d$higher_educ)
```

```
# Нормализуем данные некоторых столбцов
```

```

# зарплата
d["salary"]=d$tj13.2
d$salary = as.numeric(d$salary)
d["salary"] = (d["salary"] - mean(d$salary)) / sqrt(var(d$salary))

# возраст
d["age"]=d$t_age
d$age = as.numeric(d$age)
d["age"] = (d["age"] - mean(d$age)) / sqrt(var(d$age))

# продолжительность рабочей недели в часах
d["workweek"]=d$tj6.2
d$workweek = as.numeric(d$workweek)
d["workweek"] = (d["workweek"] - mean(d$workweek)) / sqrt(var(d$workweek))

d2 = select(d, salary, age, sex, higher_educ, city_status, workweek, wed1, wed2, wed3)
d2

#1

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

model1 = lm(salary~age+sex+higher_educ+city_status+workweek+wed1+wed2+wed3, d2)
vif(model1)

# Vif для всех регрессоров < 2.4, что говорит о лин независимости лин. регрессоров.

summary(model1)

# Из данных можно сказать, что все регрессоры, кроме wed2(1 звезда) и wed3(0 звёзд)
неплохо описывают данные(2-3 звезды),

```

# разброс они описывают маленький: multiple R-squared = 0.175, adjusted R-squared = 0.1735.

#Попробуем удалить регрессор wed3:

```
model2 = lm(salary~age+sex+higher_educ+city_status+workweek+wed1+wed2, d2)
```

```
vif(model2)
```

# все регрессоры лин независимы

```
summary(model2)
```

# Multiple R-squared: 0.1745, Adjusted R-squared: 0.1732,

# p-значение всех регрессоров увеличились до трех звезд.

#2

#Поэкспериментируйте с функциями вещественных параметров: используйте

# логарифм и степени

# в некоторых наших столбцах есть отрицательные значения,

# добавим им мин значения, которые доведут их до положительных

```
describe(d2)
```

# salary lowest = -1.396326

```
d2["salary"] = d2["salary"] + 1.5
```

# age lowest = -1.969413

```
d2["age"] = d2["age"] + 2
```

# workweek lowest = -3.362522

```
d2["workweek"] = d2["workweek"] + 3.5
```

# Наверное, имеет смысл работать с некатегориальными регрессорами.

# Это регрессоры salary, age и workweek

```

model3 =
lm(log(salary)~age+sex+higher_educ+city_status+workweek+wed1+wed2+log(age)+log(workweek), d2)

summary(model3)

vif(model3)

# Multiple R-squared:  0.2201,    Adjusted R-squared:  0.2185

# p-значения wed1, wed2 плохи(0 звезд и точка), у workweek - одна звезда, остальные
регрессоры по 3.

# Vif большей части регрессоров нормальны.

# Но vif log(age) и age около 10, log(workweek) и workweek около 6. Это скорее
говорит о лин зависимости.

# попробуем убрать логарифм с salary

model4 =
lm(salary~age+sex+higher_educ+city_status+workweek+wed1+wed2+log(age)+log(workweek), d2)

summary(model4)

vif(model4)

# Vif не изменился => модель не подходит.

# Multiple R-squared:  0.1874,    Adjusted R-squared:  0.1858

# P-значения всех регрессоров хороши(3 звезды), кроме workweek(0 звезд), но мы
видим, что потеряли

# часть значений в разбросе данных

# уберем регрессор workweek(0 звезд) и регрессор log(age)(p-значение меньше, чем у
age)

model5 = lm(salary~age+sex+higher_educ+city_status+wed1+wed2+log(workweek), d2)

summary(model5)

vif(model5)

# Multiple R-squared:  0.1768,    Adjusted R-squared:  0.1754.

# P-значения всех регрессоров по 3 звезды.

# vif < 1.7, что просто отлично.

# С логарифмами мы получили отличную модель, попробуем улучшить её

```

# Multiple R-squared и Adjusted R-squared с помощью прибавления регрессора в какой-то степени.

```
model6 = lm(salary~age+sex+higher_educ+city_status+wed1+wed2+log(workweek) +  
I(age*age), d2)
```

```
summary(model6)
```

```
vif(model6)
```

# age и I(age\*age) лин зависимы, модель нам не подходит.

```
model7 = lm(salary~age+sex+higher_educ+city_status+wed1+wed2+log(workweek) +  
I(age^1.3), d2)
```

```
vif(model7)
```

```
summary(model7)
```

# При подстановке других степеней age и I(age\*age) лин зависимы, модель нам не подходит.

# Попробуем другой параметр.

```
model8 = lm(salary~age+sex+higher_educ+city_status+wed1+wed2+log(workweek) +  
I((log(workweek))^2), d2)
```

```
vif(model8)
```

```
summary(model8)
```

# При работе со степенями и workweek, и log(workweek) ничего дельного не вышло. То в модели были линейно

# зависимые регрессоры, то Multiple R-squared и Adjusted R-squared не менялись, а p-значения

# вводимого аргумента не менялись и оставались 0 звёзд.

#3.

#Выделите наилучшие модели из построенных: по значимости параметров,

#включённых в зависимости, и по объяснённой с помощью построенных зависимостей разбросу adjusted R<sup>2</sup>.

# Лучшие модели - model5(используется log(workweek)), model2(просто линейная модель)



# Учитывая противоречивость model5, лучшей становится обычная линейная модель model2.

# Протестируя несколько различных моделей стало ясно, что добавление к параметру salary логарифма

# даёт прирост  $R^2$  и adj  $R^2$ , хотя мы всегда теряем в p-значениях. Попробуем это проделать с линейной

# моделью, т.е. добавим к ней логарифм объясняемой переменной. Построим специальную модель:

```
spec_model = lm(log(salary)~age+sex+higher_educ+city_status+workweek+wed1+wed2,  
d2)
```

```
summary(spec_model)
```

```
vif(spec_model)
```

# Multiple R-squared: 0.1974, Adjusted R-squared: 0.1961

# p-значения всех переменных 2-3 звезды(2 звезды у wed1 и wed2, что тоже неплохо)

# однозначно, это наша лучшая модель

#4

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

# Вывод мы сможем сделать написав лин регрессию параметра "salary"

```
spec_model = lm(log(salary)~age+sex+higher_educ+city_status+workweek+wed1+wed2,  
d2)
```

```
summary(spec_model)
```

```
# log(salary) = -0.478954 - 0.050834*age + 0.346430*sex + 0.323261*higher_educ +  
0.250875*city_status +
```

```
# + 0.088260*workweek + 0.060366*wed1 + 0.085338*wed2
```

# Теперь нам стоит еще немного поработать с log(salary). Так как для того, чтобы взять логарифм от данной

# категории после нормализации мы прибавили к столбцу значений 1.5, то есть реально мы описывали переменную

#  $\log(\text{salary}+1.5)$ , но так преобразование линейное, то мы просто должны будем вычесть из свободного члена

# константу, но это не повлияет на общий вид картины описания индивидов,

# так что я продолжу описывать  $\log(\text{salary}+1.5)$ !

# из линейной регрессии видно, что на данный момент больше всего заработает мужчина( $\text{sex} = 1$ ) с высшим

# образованием( $\text{higher\_educ} = 1$ ), проживающий в городе или областном центре;

#  $\log(\text{salary}) = 0.441612 - 0.050834*\text{age} + 0.088260*\text{workweek} + 0.060366*\text{wed1} + 0.085338*\text{wed2}$

# Поймем, что больше зарабатывать будет мужчина с высшим образованием, разведенный,

# или не состоящий в браке, или вдовец( $\text{wed2} = 1 \Rightarrow \text{wed1} = 0$ ,  $\text{wed2}$  на немного, но дает больший вклад):

#  $\text{salary} = 0.52695 - 0.050834*\text{age} + 0.088260*\text{workweek}$

# Данный мужчина также должен перерабатывать и должен быть относительно

# времени получения высшего образования быть молодым. Именно тогда он будет зарабатывать больше всех.

#5.

# Оцените регрессии для подмножества индивидов, указанных в варианте:

# 1. городские жители, состоящие в браке;

# 2. разведенные, без высшего образования.

#  $\log(\text{salary}) = -0.478954 - 0.050834*\text{age} + 0.346430*\text{sex} + 0.323261*\text{higher\_educ} + 0.250875*\text{city\_status} +$

#  $+ 0.088260*\text{workweek} + 0.060366*\text{wed1} + 0.085338*\text{wed2}$

# 1.  $\log(\text{salary}) = -0.167713 - 0.050834*\text{age} + 0.346430*\text{sex} + 0.323261*\text{higher\_educ} + 0.088260*\text{workweek}$

# Такая будет зависимость для первой категории индивидов. Индивид будет получать больше, т.к.

# он проживает в городе или областном центре, но и немного потеряет в зарплате, т.к. он

# находится в браке( $wed1 = 1 \Rightarrow wed2 = 0$ , а это потеря, т. к. коэф. при  $wed2 >$  коэф. при  $wed1$ ).

#  $2. \log(\text{salary}) = -0.393616 - 0.050834 * \text{age} + 0.346430 * \text{sex} + 0.250875 * \text{city\_status} + 0.088260 * \text{workweek}$

# Такая будет зависимость для второй категории индивидов. Индивид будет меньше получать, чем мог бы,

# т. к. он очень сильно теряет часть баллов к свободному члену из-за отсутствия высшего образования

# ( $\text{higher\_educ} = 0$ ), но немного отыгрывает(хотя очень даже не сильно) из-за того, что он разведен

# ( $wed2 = 1 \Rightarrow wed1 = 0$ ).

## Приложение 3.

```
# произвели загрузку библиотек
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# прочтем данные
data = pd.read_csv('C:/Users/Admin/Downloads/winequality-red.csv')

# Для предупреждений.
import warnings as warnings
warnings.filterwarnings('ignore')

# теперь посмотрим несколько первых и последних строк датасета
data.head()

data.tail()

# посмотрим информацию по столбцам
data.info()

data.head()
data.describe()

data.isna().sum() #получим количество пропусков построчно

data.describe()

# сделаем признак quality категориальным; если quality > 6.5(из предисловия к
датасету), то вино хорошее(будем обозначать 1),
# в противном случае, вино плохое(будем обозначать за 0)
```

```

data.loc[data['quality'] < 6.5, 'quality'] = 0
data.loc[data['quality'] > 6.5, 'quality'] = 1
data.describe()

# нормируем данные с помощью стандартного отклонения
# я не стал нормировать признак
columns = data.columns.tolist()
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
average_values = []
for i in columns:
    if i == 'quality': #не берем данный признак, т к он категориальный
        break
    data[i] = scale.fit_transform(data[[i]])
    average_values.append(data[i].mean())
data.describe()

target = data.quality
del data['quality']
train = data

train.head()

from sklearn.decomposition import PCA
%matplotlib inline

pca = PCA()
pca.fit(X_train)
X_pca = pca.transform(X_train)

```

```

for i, component in enumerate(pca.components_):
    print("{} component: {}% of initial variance".format(i + 1,
        round(100 * pca.explained_variance_ratio_[i], 2)))
    print(" + ".join("%.3f x %s" % (value, name)
        for value, name in zip(component, train.columns)))

plt.figure(figsize=(10,7))
plt.plot(np.cumsum(pca.explained_variance_ratio_), color='k', lw=2)
plt.axhline(0.9, c='r')
plt.axvline(6, c='b')

# Для нового объекта алгоритм ищет в обучающей выборке k наиболее близких
# объекта и относит новый объект к тому классу,
# которому принадлежит большинство из них.
# Количество соседей k соответствует параметру n_neighbors. По умолчанию,
n_neighbors = 5.
# не работает при нормированном столбце 'quality'
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()
knn.fit(X_train, y_train)

# Предскажем значение целевого признака по входным признакам для новых объектов.
# Нас интересует качество построенной модели, поэтому будем предсказывать
# значение выходного признака на тех данных,
# для которых оно известно: на обучающей и (что более важно) тестовой выборках:
y_train_predict = knn.predict(X_train)
y_test_predict = knn.predict(X_test)

err_train = np.mean(y_train != y_train_predict)
err_test = np.mean(y_test != y_test_predict)

```

```
print(err_train, err_test)
```

```
from sklearn.model_selection import GridSearchCV
```

```
n_neighbors_array = [i for i in range(1,12)]
```

```
knn = KNeighborsClassifier(n_neighbors=5)
```

```
grid = GridSearchCV(knn, param_grid={'n_neighbors': n_neighbors_array})
```

```
grid.fit(X_train, y_train)
```

```
best_cv_err = 1 - grid.best_score_
```

```
best_n_neighbors = grid.best_estimator_.n_neighbors
```

```
print(best_cv_err, best_n_neighbors)
```

```
# Кросс-валидация, которую ещё называют перекрестной проверкой, это техника  
# валидации модели для проверки того,
```

```
# насколько успешно применяемый в модели статистический анализ способен работать  
# на независимом наборе данных.
```

```
# Как видно в качестве оптимального метод выбрал значение k равное 4, при которой  
# ошибка перекрестной проверки составляет
```

```
# примерно 12.4 процентов, что немного ниже ошибки на тестовой модели. Теперь  
# попробуем метод с данным параметром k:
```

```
knn = KNeighborsClassifier(n_neighbors=best_n_neighbors)
```

```
knn.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != knn.predict(X_train))
```

```
err_test = np.mean(y_test != knn.predict(X_test))
```

```
print(err_train, err_test)
```

```
from sklearn import ensemble
```

```
rf = ensemble.RandomForestClassifier(n_estimators=100, random_state=11)
```

```
rf.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != rf.predict(X_train))
```

```
err_test = np.mean(y_test != rf.predict(X_test))  
print(err_train, err_test)
```

```
from sklearn import ensemble  
gbt = ensemble.GradientBoostingClassifier(n_estimators=100, random_state=11)  
gbt.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != gbt.predict(X_train))  
err_test = np.mean(y_test != gbt.predict(X_test))  
print(err_train, err_test)
```

```
from sklearn.svm import SVC  
svc = SVC()  
svc.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != svc.predict(X_train))  
err_test = np.mean(y_test != svc.predict(X_test))  
print(err_train, err_test)
```

```
# мы получили результат - ошибка в 11.6 %. Попробуем поменять ядро  
svc = SVC(kernel='linear')  
svc.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != svc.predict(X_train))  
err_test = np.mean(y_test != svc.predict(X_test))  
print(err_train, err_test)
```

```
# Результат хуже  
svc = SVC(kernel='poly')  
svc.fit(X_train, y_train)
```



```

err_train = np.mean(y_train != svc.predict(X_train))
err_test = np.mean(y_test != svc.predict(X_test))
print(err_train, err_test)

# попробуем подобрать наилучшие параметры для радиального ядра
from sklearn.model_selection import GridSearchCV
C_array = np.logspace(-3, 3, num=7)
gamma_array = np.logspace(-5, 2, num=8)
svc = SVC(kernel='rbf')
grid = GridSearchCV(svc, param_grid={'C': C_array, 'gamma': gamma_array})
grid.fit(X_train, y_train)
print('CV error = ', 1 - grid.best_score_)
print('best C = ', grid.best_estimator_.C)
print('best gamma = ', grid.best_estimator_.gamma)

svc = SVC(kernel='rbf', C=grid.best_estimator_.C, gamma=grid.best_estimator_.gamma)
svc.fit(X_train, y_train)

err_train = np.mean(y_train != svc.predict(X_train))
err_test = np.mean(y_test != svc.predict(X_test))
print(err_train, err_test)

# Теперь моя дополнительная задача: проделать все то же самое, но уже без
нормирования данных
data = pd.read_csv('C:/Users/Admin/Downloads/winequality-red.csv')

data.loc[data['quality'] < 6.5, 'quality'] = 0
data.loc[data['quality'] > 6.5, 'quality'] = 1

target = data.quality

```

```

del data['quality']

train = data

# Выделим тренировочную и тестовую выборку
# y - целевая переменная (target)
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(train, target, test_size = 0.3, random_state =
42)

from sklearn.decomposition import PCA

%matplotlib inline

pca = PCA()
pca.fit(X_train)
X_pca = pca.transform(X_train)

for i, component in enumerate(pca.components_):
    print("{} component: {} % of initial variance".format(i + 1,
        round(100 * pca.explained_variance_ratio_[i], 2)))
    print(" + ".join("%.3f x %s" % (value, name)
        for value, name in zip(component, train.columns)))

plt.figure(figsize=(10,7))
plt.plot(np.cumsum(pca.explained_variance_ratio_), color='k', lw=2)
plt.axhline(0.95, c='r')
plt.axvline(0, c='b')
plt.axhline(0.995, c='y')
plt.axvline(1, c='g')

#1) Метод k-ближайших соседей.
from sklearn.neighbors import KNeighborsClassifier

```

```

knn = KNeighborsClassifier()
knn.fit(X_train, y_train)

y_train_predict = knn.predict(X_train)
y_test_predict = knn.predict(X_test)

err_train = np.mean(y_train != y_train_predict)
err_test = np.mean(y_test != y_test_predict)
print(err_train, err_test)

# Результат на тестовой модели без нормализовки оказался хуже, чем аналогичный с
нормализовкой.
from sklearn.model_selection import GridSearchCV
n_neighbors_array = [i for i in range(1,12)]
knn = KNeighborsClassifier(n_neighbors=5)
grid = GridSearchCV(knn, param_grid={'n_neighbors': n_neighbors_array})
grid.fit(X_train, y_train)

best_cv_err = 1 - grid.best_score_
best_n_neighbors = grid.best_estimator_.n_neighbors
print(best_cv_err, best_n_neighbors)

knn = KNeighborsClassifier(n_neighbors=best_n_neighbors)
knn.fit(X_train, y_train)

err_train = np.mean(y_train != knn.predict(X_train))
err_test = np.mean(y_test != knn.predict(X_test))
print(err_train, err_test)

```

# Оптимальное  $k = 2$ , и при нем же ошибка на тестовой выборке увеличилась 0.20%. Попробуем другой метод.

# 2) Метод - Random Forest;

```
from sklearn import ensemble
```

```
rf = ensemble.RandomForestClassifier(n_estimators=100, random_state=11)
```

```
rf.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != rf.predict(X_train))
```

```
err_test = np.mean(y_test != rf.predict(X_test))
```

```
print(err_train, err_test)
```

# Метод случайного леса показал себя много лучше метода  $k$  ближайших соседей. Здесь ошибка на тестовой выборке составила примерно

# 11%.

# 3) метод GBT

```
from sklearn import ensemble
```

```
rf = ensemble.RandomForestClassifier(n_estimators=100, random_state=11)
```

```
rf.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != rf.predict(X_train))
```

```
err_test = np.mean(y_test != rf.predict(X_test))
```

```
print(err_train, err_test)
```

# Результат совпал с результатом метода случайного леса. Попробуем последний метод - SVC.

# 4) метод SVC

#а) Радиальное ядро

```
from sklearn.svm import SVC
```

```
svc = SVC()
```

```
svc.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != svc.predict(X_train))
err_test = np.mean(y_test != svc.predict(X_test))
print(err_train, err_test)
```

#b) Линейное ядро

```
svc = SVC(kernel='linear')
svc.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != svc.predict(X_train))
err_test = np.mean(y_test != svc.predict(X_test))
print(err_train, err_test)
```

#c) Полиномиальное ядро

# Для него задача, видимо, также не решается.

# попробуем улучшить параметры для работающих 2 ядер.

```
from sklearn.model_selection import GridSearchCV
```

```
C_array = np.logspace(-3, 3, num=7)
```

```
gamma_array = np.logspace(-5, 2, num=8)
```

```
svc = SVC(kernel='rbf')
```

```
grid = GridSearchCV(svc, param_grid={'C': C_array, 'gamma': gamma_array})
```

```
grid.fit(X_train, y_train)
```

```
print('CV error = ', 1 - grid.best_score_)
```

```
print('best C = ', grid.best_estimator_.C)
```

```
print('best gamma = ', grid.best_estimator_.gamma)
```

```
svc = SVC(kernel='rbf', C=grid.best_estimator_.C, gamma=grid.best_estimator_.gamma)
```

```
svc.fit(X_train, y_train)
```

```
err_train = np.mean(y_train != svc.predict(X_train))
```

```
err_test = np.mean(y_test != svc.predict(X_test))  
print(err_train, err_test)
```