Содержание

Введение

В последнее время информация, растущая в колоссальных объёмах, рождает потребность в обработке больших объёмов данных. В этом направлении большое место отведено интеллектуальному анализу данных. Это направление включает в себя методы, отличные от классического анализа, основанные на моделировании, вероятностных, и решающие задачи обобщения, ассоциирования и отыскания закономерностей. В большой степени развитию этой дисциплины способствовало проникновение в сферу анализа данных идей, возникших в теории искусственного интеллекта.

В данной работе я хотел бы рассмотреть частную задачу подобного анализа, а именно задачу классификации

Классификация — один из разделов машинного обучения, посвященный решению следующей задачи. Имеется множество объектов (ситуаций), разделённых некоторым образом на классы. Задано конечное множество объектов, для которых известно, к каким классам они относятся. Это множество называется обучающей выборкой. Классовая принадлежность остальных объектов не известна. Требуется построить алгоритм, способный классифицировать произвольный объект из исходного множества.

Задача 1

Необходимо загрузить данные из указанного набора и произвести следующие действия.

1. Нормализовать данные, вычтя из каждого столбца среднее значение mean(x) и поделив на среднеквадратическое отклонение σ ~ sqrt(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 R2 в каждой из них не высокий). В случае, если R2 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.

Набор данных – Swiss.

Объясняемая переменная – Fertility ().

Регрессоры (объясняющие переменные): Agriculture (%Мужчин в сельхозе), Catholic (%Католиков) и Infant.Mortality(%Смертность среди детей до 1 года)

Решение Задача 1

В переменную data2 загружены данные из набора данных Swiss. На *рисунке 1* показана часть таблицы из набора данных.

	Fertility	Agriculture	Examination	Education	Catholic	Infant.Mortality
Courtelary	80.2	17.0	15	12	9.96	22.2
Delemont	83.1	45.1	6	9	84.84	22.2
Franches-Mnt	92.5	39.7	5	5	93.40	20.2
Moutier	85.8	36.5	12	7	33.77	20.3
Neuveville	76.9	43.5	17	15	5.16	20.6
Porrentruy	76.1	35.3	9	7	90.57	26.6

Рис. 1. Набор данных в таблице data.

Из данного набора данных в новую переменную data при помощи функции select выбираю столбцы из data2: Fertility, Agriculture, Catholic, Infant.Mortality. В дальнейшем буду работать с данной таблицей. Далее мне необходимо нормализовать регрессоры (, Agriculture, Catholic, Infant.Mortality), но перед этим логарифмирую их. Возьму логарифмы данных из столбцов, Agriculture, Catholic, Infant.Mortality. Полученные данные помещаю в созданные столбцы Log_Agr, Log_Cat и Log_IM

```
data["Log_Agriculture»] =log(data$Agriculture)

data["Log_Catholic»] =log(data$Catholic)

data["Log_Infant.Mortality»] =log(data$Infant.Mortality)
```

“Вычтя из каждого столбца среднее значение mean(x) и поделив на среднеквадратическое отклонение σ ~ sqrt(var(x)), где x – столбец данных. “

Нормализация данных(пример Agriculture)

```
Agr1<-as.character(data$Agriculture)

Agr2<-lapply(Agr1,as.integer)

Agr3<-as.numeric(unlist(Agr2))

data["Agr"]=(Agr3-mean(Agr3))/sqrt(var(Agr3))
```

Также логарифмированы и нормализированы Cat и IM и нормализованы все регрессоры и их логарифмы

Создадим первую модель взяв в нее все самое лучшее, то есть Agr, IM и Cat. При помощи функции summary определяю, как регрессоры влияют на Fertility. Показатель R^2 равен 0.3449. При это коэффициент p – () у Agr, (*) у Cat и (**) у IM

```
model1<-lm(Fertility~Agr+Cat+IM,data)
```

Уберем из модели Agr так как у него наибольший коэффициент P

model2<-lm(Fertility~Cat+IM,data)

model2

summary(model2) #R^2=0.302

#p-(**)(**)

$\mathbf{R^2}$ уменьшился первая модель лучше

Далее в разделе «Код» (как у Петцольда только просто код)

Представлена часть моделей, которые я опробовал (какую-то часть я не записал так как сложно печатать много моделей, когда можно корректировать и улучшать одну не перепечатывая ее. Лучший результат, полученный мной за 20-25 моделей представлен в модели под номером 12.

model12<-lm (Fertility~Ln_IM+I(Cat^2)+I(Agr^2),data)#Best

model12

summary(model12) #R^2=0.564

Output:

```
Call:
lm(formula = Fertility ~ Ln_IM + I(Cat^2) + I(Agr^2), data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-12.4617  -5.9444   0.1059   3.4609  21.6632

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   59.500      2.568  23.173  < 2e-16 ***
Ln_IM          4.860      1.251   3.886 0.000347 ***
I(Cat^2)      12.668      1.962   6.458 7.88e-08 ***
I(Agr^2)      -1.794      1.155  -1.554 0.127616
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 8.248 on 43 degrees of freedom
Multiple R-squared:  0.5924,            Adjusted R-squared:  0.564
F-statistic: 20.83 on 3 and 43 DF,  p-value: 1.732e-08
```

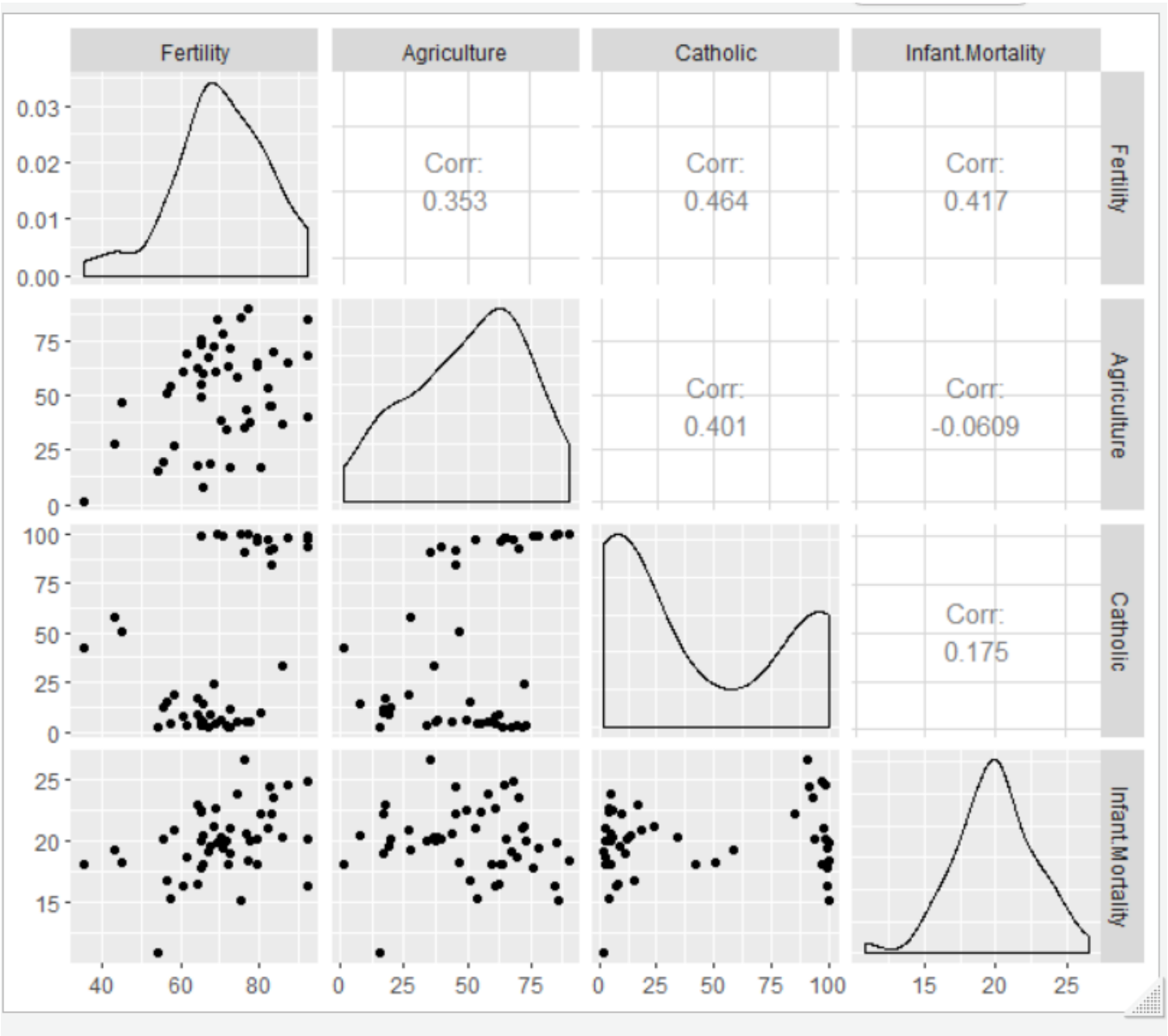

Рис. 2 результат работы функции ggpairs

Так же при помощи функции “ggpairs(data)” можно получить графики зависимостей из базы данных, это тоже может помочь при анализе данных зачёт большей наглядности

Выполняя эту задачу по анализу данных, я пришел в следующим выводам:

1. Чем выше рождаемость среди детей до года, тем выше смертность
2. Заводят детей либо только католики, либо только не католики (Сильное влияние)
3. Рождаемость не очень сильно зависит от агрокультуры (наиболее высокая рождаемость при уровне развития агрокультуры чуть выше среднего)

Задача 2

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

В случае, если не удаётся прочесть соответствующий файл, обратитесь к материалу, выложенному вместе с заданием.

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 (с помощью one-hot-encoding): 1) переменная wed1 имеет значение 1 в случае, если респондент женат, 0 – в противном случае; 2) wed2=1, если респондент разведён или вдовец; 3) wed3 = 1, если респондент никогда не состоял в браке; 4) если считаете необходимым, введите другие параметры. Следите за мультиколлинеарностью (убедитесь в её отсутствии, оценив вспомогательную регрессию любого параметра (например, зарплаты или одного из параметров wed) на эти переменные и использовав команду VIF для неё).

Из параметра пол сделайте переменную sex, имеющую значение 1 для мужчин и равную 0 для женщин.

Из параметра, отвечающего типу населённого пункта, создайте одну дамми-переменную city_status со значением 1 для города или областного центра, 0 – в противоположном случае.

Введите один параметр higher_educ, характеризующий наличие полного высшего образования.

Факторные переменные, «имеющие много значений», такие как: зарплата, длительность рабочей недели и возраст, - необходимо преобразовать в вещественные переменные и нормализовать их: вычесть среднее значение по этой переменной, разделить её значения на стандартное отклонение.

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

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

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

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

5. Оцените регрессии для подмножества индивидов, указанных в варианте

Решение Задача 2

1 Обработка данных

В переменную data загружены данные волны № 19 мониторинга экономического положения и здоровья населения РФ.

Из данного набора данных при помощи функции select выбираю столбцы:

1. idind (индивидуальный номер),
2. oj13.2 (средняя зарплата за последние 12 месяцев),
3. oh5 (пол респондента),
4. o_marst (семейное положение),
5. o_educ (образование),
6. o_age (количество полных лет),
7. status (тип населенного пункта)
8. oj6.2 (продолжительность рабочей недели).

Удаляю строки содержащие пустые значения (Чертов Unicode).

Далее веду работу с полученной «Датой»

If (был женат или замужем)

Wed2 == 1

If (Состоит в браке)

Wed1 == 1

If ((Никогда не был в браке)

Wed3 == 1

```
data2["wed1"]=data2$o_marst
data2["wed1"]=0
data2$wed1[which(data2$o_marst=='2')] <- 1
data2$wed1[which(data2$o_marst=='6')] <- 1

data2["wed2"]=data2$o_marst
data2["wed2"]=0
data2$wed2[which(data2$o_marst=='4')] <- 1
data2$wed2[which(data2$o_marst=='5')] <- 1
#data2$wed2 = as.numeric(data2$wed2)
data2["wed3"]=data2$o_marst
data2["wed3"]=0
```

data2$wed3[which(data2$o_marst=='1')] <- 1

#data2$wed3 = as.numeric(data2$wed3)

Переменная sex: пол человека 1 – мужчина, 0-женщина

data2["sex"]=data2$oh5

data2$sex[which(data2$sex!='1')] <- 0

data2$sex[which(data2$sex=='1')] <- 1

data2$sex = as.numeric(data2$sex)

Аналогично city_status: 1-город, 0-нет

Далее Высшее образование да либо нет

Также возраст, логарифм от возраста, часов в неделю, и логарифм от часов в неделюю

2 Построение моделей

model1 = lm(salary~wed1+wed2+wed3+sex+city_status+higher_educ+age+dur, data2)

model1

summary(model1) #R^2 = 0.137

vif(model1)

output:

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.54373    0.04021 -13.521  < 2e-16 ***
wed1         0.01081    0.03674   0.294   0.7686
wed2        -0.01093    0.04710  -0.232   0.8164
wed3        -0.11441    0.04585  -2.495   0.0126 *
sex          0.40666    0.02466  16.492  < 2e-16 ***
city_status  0.30445    0.02608  11.674  < 2e-16 ***
higher_educ  0.50632    0.02576  19.658  < 2e-16 ***
age         -0.05955    0.01310  -4.545 5.59e-06 ***
dur          0.13043    0.01196  10.904  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.929 on 6415 degrees of freedom
Multiple R-squared:  0.1381,             Adjusted R-squared:  0.137
F-statistic: 128.5 on 8 and 6415 DF,  p-value: < 2.2e-16
```

```
    wed1      wed2      wed3       sex city_status higher_educ       age       dur
2.434904  2.024549  1.935660  1.118280  1.022135  1.049440  1.277603  1.064891
```

model2 = lm(salary~wed2+wed3+sex+city_status+higher_educ+age+dur, data2)

model2 #

summary(model2) #R^2 = 0.1371

vif(model2)

Лучшая модель

model35 = lm(salary~wed2+wed3+sex+city_status+higher_educ+I(age^2)+dur, data2)

model35

summary(model35) #R^2 = 0.148

vif(model35)

Output:

```
Call:
lm(formula = salary ~ wed2 + wed3 + sex + city_status + higher_educ +
    I(age^2) + dur, data = data2)

Residuals:
    Min      1Q  Median      3Q     Max
-2.0180 -0.5156 -0.1828  0.2699 15.6246

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.459394   0.027609 -16.640   <2e-16 ***
wed2        -0.011660   0.034662  -0.336    0.737
wed3         0.007128   0.033949   0.210    0.834
sex          0.422936   0.024554  17.225   <2e-16 ***
city_status  0.320224   0.025919  12.355   <2e-16 ***
higher_educ  0.501966   0.025509  19.678   <2e-16 ***
I(age^2)    -0.112735   0.011140 -10.120   <2e-16 ***
dur          0.124105   0.011897  10.432   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.9231 on 6416 degrees of freedom
Multiple R-squared:  0.1489,             Adjusted R-squared:  0.148
F-statistic: 160.3 on 7 and 6416 DF,  p-value: < 2.2e-16
```

```
> vif(model35)
      wed2       wed3        sex city_status higher_educ    I(age^2)        dur
  1.110464   1.074906   1.123126   1.022625   1.042538   1.061074   1.066929
```

Вывод: Пол, Место жительства влияют сильно. Высшее образование сильнее всего.

Задача 3

Необходимо провести анализ вашего датасета и сделать обработку данных по предложенному выше алгоритму. Код подготовить в виде питоновских файлов *.py и сделать отчет в виде ноутбука.

- Сколько в датасете объектов и признаков? Дать описание каждому признаку, если оно есть.
- Сколько категориальных признаков, какие?
- Столбец с максимальным количеством уникальных значений категориального признака?
- Есть ли бинарные признаки?
- Есть ли пропуски?
- Сколько объектов с пропусками?
- Столбец с максимальным количеством пропусков?
- Есть ли на ваш взгляд выбросы, аномальные значения?
- Столбец с максимальным средним значением после нормировки признаков через стандартное отклонение?
- Столбец с целевым признаком?
- Сколько объектов попадает в тренировочную выборку при использовании train_test_split с
- параметрами test_size = 0.3, random_state = 42?
- Сколько признаков достаточно для объяснения 90% дисперсии после применения метода PCA?
- Какой признак вносит наибольший вклад в первую компоненту?

1.Загрузка и предварительный просмотр данных

Библиотеки:

import numpy as np

import pandas as pd

Загружаем данные из файла:

data = pd.read_csv("winequality-red.csv") #Чтение данных

Рисунок.10

Первичный анализ :

Объем датасета

data.shape

output:

(1599,12)

Наша таблица содержит 1599 столбцов и 9 строк

Смотрим на верхушку

```
data.head()
```

	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

Рис. 3

Выводим информацию о наших данных:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
fixed acidity           1599 non-null float64
volatile acidity        1599 non-null float64
citric acid             1599 non-null float64
residual sugar          1599 non-null float64
chlorides               1599 non-null float64
free sulfur dioxide     1599 non-null float64
total sulfur dioxide    1599 non-null float64
density                 1599 non-null float64
pH                      1599 non-null float64
sulphates               1599 non-null float64
alcohol                 1599 non-null float64
quality                 1599 non-null int64
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

Рис. 4

Делая вывод из проделанных операций, можно сказать, что датасет содержит 1599 объект и у каждого объекта 12 признаков. Описание признаков:

1) fixed acidity - дробный признак, фиксированная кислотность вина

2) volatile acidity - дробный признак,летучая кислотность вина

3) citric acid - дробный признак, лимонная кислота

4) residual sugar - дробный признак,остаточный сахар

5) chlorides, - дробный признак,хлориды

6) free sulfur dioxide, - дробный признак,св диоксид серы

7) total sulfur dioxide - дробный признак,весь диоксид серы

8) density - дробный признак,плотность

9) pH- дробный признак,кислотность и щелочность

10) sulphates - дробный признак, сульфаты

11) alcohol - дробный признак, алкоголь

12) quality - целочисленный признак , качество вина

Смотрим на наши данные и видим, что категориальным признаком является *quality.*

Столбцом с максимальным количеством уникальных значений категориального признака так же является столбец *quality*

(он принимает значения от 0 до 10)

По нашей таблице мы так же можем заметить то что у нас нет бинарных признаков.

Далее проверяем нашу таблицу на пропуски:

```
np.sum(pd.isnull(data))
```

```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

Рис. 6

Здесь видно, что количество пропусков в каждом признаке равно 0

Далее смотрим на наши данные снова на наличие аномалий функцией data.describe():

```
data.describe()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	0.658149	10.422983
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	0.169507	1.065668
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	0.550000	9.500000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	0.620000	10.200000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	0.730000	11.100000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000	2.000000	14.900000

Рис.7

Просматривая данные таблицы, можно сделать вывод о том, что возможно есть аномалия в total sulfur dioxide очень высокий в max.

Далее копируем нашу таблицу для дальнейшего использования без потери данных и применим функцию corr() для удаления NaN\Null значений и поиска зависимостей.

```
data_cleaned = data.copy()
data.corr()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
fixed acidity	1.000000	-0.256131	0.671703	0.114777	0.093705	-0.153794	-0.113181	0.668047	-0.682978	0.183006	-0.061668	0.124052
volatile acidity	-0.256131	1.000000	-0.552496	0.001918	0.061298	-0.010504	0.076470	0.022026	0.234937	-0.260987	-0.202288	-0.390558
citric acid	0.671703	-0.552496	1.000000	0.143577	0.203823	-0.060978	0.035533	0.364947	-0.541904	0.312770	0.109903	0.226373
residual sugar	0.114777	0.001918	0.143577	1.000000	0.055610	0.187049	0.203028	0.355283	-0.085652	0.005527	0.042075	0.013732
chlorides	0.093705	0.061298	0.203823	0.055610	1.000000	0.005562	0.047400	0.200632	-0.265026	0.371260	-0.221141	-0.128907
free sulfur dioxide	-0.153794	-0.010504	-0.060978	0.187049	0.005562	1.000000	0.667666	-0.021946	0.070377	0.051658	-0.069408	-0.050656
total sulfur dioxide	-0.113181	0.076470	0.035533	0.203028	0.047400	0.667666	1.000000	0.071269	-0.066495	0.042947	-0.205654	-0.185100
density	0.668047	0.022026	0.364947	0.355283	0.200632	-0.021946	0.071269	1.000000	-0.341699	0.148506	-0.496180	-0.174919
pH	-0.682978	0.234937	-0.541904	-0.085652	-0.265026	0.070377	-0.066495	-0.341699	1.000000	-0.196648	0.205633	-0.057731
sulphates	0.183006	-0.260987	0.312770	0.005527	0.371260	0.051658	0.042947	0.148506	-0.196648	1.000000	0.093595	0.251397
alcohol	-0.061668	-0.202288	0.109903	0.042075	-0.221141	-0.069408	-0.205654	-0.496180	0.205633	0.093595	1.000000	0.476166
quality	0.124052	-0.390558	0.226373	0.013732	-0.128907	-0.050656	-0.185100	-0.174919	-0.057731	0.251397	0.476166	1.000000

Рис.8

После оценки данных видно, что можно убрать слишком маленькие значения quality

```
In [354]:
del data_cleaned['residual sugar']
del data_cleaned['free sulfur dioxide']
del data_cleaned['pH']
cleaned_data
```

25	6.3	0.390	0.16	0.080	23.0	0.99550	0.56	9.3	5
26	7.6	0.410	0.24	0.080	11.0	0.99620	0.59	9.5	5
27	7.9	0.430	0.21	0.106	37.0	0.99660	0.91	9.5	5
28	7.1	0.710	0.00	0.080	35.0	0.99720	0.55	9.4	5
29	7.8	0.645	0.00	0.082	16.0	0.99640	0.59	9.8	6
...	...	...	...	...	...	...	...	...	...
1569	6.2	0.510	0.14	0.056	34.0	0.99396	0.57	11.5	6
1570	6.4	0.360	0.53	0.230	35.0	0.99340	0.93	12.4	6
1571	6.4	0.380	0.14	0.038	25.0	0.99514	0.65	11.1	6
1572	7.3	0.690	0.32	0.069	104.0	0.99632	0.51	9.5	5
1573	6.0	0.580	0.20	0.075	50.0	0.99467	0.67	12.5	6
1574	5.6	0.310	0.78	0.074	92.0	0.99677	0.48	10.5	6
1575	7.5	0.520	0.40	0.060	20.0	0.99474	0.64	11.8	6

Рис.9

Теперь нормализируем признаки:

```
n [355]:
#нормализация
data_cleaned["fixed acidity"] = (data_cleaned["fixed acidity"] - data_cleaned["fixed acidity"].mean()) / data_cleaned["fixed
data_cleaned["volatile acidity"] = (data_cleaned["volatile acidity"] - data_cleaned["volatile acidity"].mean()) / data_cleane
data_cleaned["citric acid"] = (data_cleaned["citric acid"] - data_cleaned["citric acid"].mean()) / data_cleaned["citric acid"
data_cleaned["chlorides"] = (data_cleaned["chlorides"] - data_cleaned["chlorides"].mean()) / data_cleaned["chlorides"].std()
data_cleaned["citric acid"] = (data_cleaned["citric acid"] - data_cleaned["citric acid"].mean()) / data_cleaned["citric acid"
data_cleaned["density"] = (data_cleaned["density"] - data_cleaned["density"].mean()) / data_cleaned["density"].std()
data_cleaned["sulphates"] = (data_cleaned["sulphates"] - data_cleaned["sulphates"].mean()) / data_cleaned["sulphates"].std()
data_cleaned["alcohol"] = (data_cleaned["alcohol"] - data_cleaned["alcohol"].mean()) / data_cleaned["alcohol"].std()
data_cleaned["quality"] = (data_cleaned["quality"] - data_cleaned["quality"].mean()) / data_cleaned["quality"].std()
```

Рис.10

Далее загрузив наши данные видим ,что максимальное среднее значение нормированных признаков в total sulfur dioxide:

```
In [356]:
data_cleaned.head()
```

Out[356]:

	fixed acidity	volatile acidity	citric acid	chlorides	total sulfur dioxide	density	sulphates	alcohol	quality
0	-0.528194	0.961576	-1.391037	-0.243630	34.0	0.558100	-0.579025	-0.959946	-0.787576
1	-0.298454	1.966827	-1.391037	0.223805	67.0	0.028252	0.128910	-0.584594	-0.787576
2	-0.298454	1.296660	-1.185699	0.096323	54.0	0.134222	-0.048074	-0.584594	-0.787576
3	1.654339	-1.384011	1.483689	-0.264878	60.0	0.664069	-0.461036	-0.584594	0.450707
4	-0.528194	0.961576	-1.391037	-0.243630	34.0	0.558100	-0.579025	-0.959946	-0.787576

```
In [357]:
data_cleaned.describe()
```

Out[357]:

	fixed acidity	volatile acidity	citric acid	chlorides	total sulfur dioxide	density	sulphates	alcohol	quality
count	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1599.000000	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03
mean	-1.570643e-14	-1.590973e-15	-7.915286e-17	5.098590e-15	46.467792	4.946064e-13	-2.175036e-15	2.580411e-14	1.081756e-15
std	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	32.895324	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00
min	-2.136377e+00	-2.277567e+00	-1.391037e+00	-1.603443e+00	6.000000	-3.537625e+00	-1.935902e+00	-1.898325e+00	-3.264143e+00
25%	-7.004996e-01	-7.696903e-01	-9.290275e-01	-3.711129e-01	22.000000	-6.075656e-01	-6.380200e-01	-8.661079e-01	-7.875763e-01
50%	-2.410190e-01	-4.367545e-02	-5.634264e-02	-1.798892e-01	38.000000	1.759533e-03	-2.250577e-01	-2.092427e-01	4.507074e-01
75%	5.056370e-01	6.264921e-01	7.650078e-01	5.382858e-02	62.000000	5.766445e-01	4.238832e-01	6.352984e-01	4.507074e-01
max	4.353787e+00	5.876138e+00	3.742403e+00	1.112355e+01	289.000000	3.678904e+00	7.916200e+00	4.201138e+00	2.927275e+00

Рис.11

Целевой переменной делаем quality

10) Выберите целевую переменную:

```
In [71]:
target = data.quality
data = data.drop(['residual sugar','free sulfur dioxide','pH'],axis = 1)

In [72]:
from sklearn.decomposition import PCA
%matplotlib inline
import matplotlib.pyplot as plt

In [73]:
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data, target , test_size = 0.30, random_state = 42)

In [74]:
X_train
```

Рис.12

В тренировочную выборку попадает 1119 объектов.

Метод главных компонент

```
In [358]:
target = data.quality
data = data.drop(['residual sugar','free sulfur dioxide','pH'],axis = 1)

In [359]:
from sklearn.decomposition import PCA
%matplotlib inline
import matplotlib.pyplot as plt

In [360]:
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data, target , test_size = 0.30, random_state = 42)

In [361]:
X_train
```

Out[361]:

	fixed acidity	volatile acidity	citric acid	chlorides	total sulfur dioxide	density	sulphates	alcohol	quality
925	8.6	0.220	0.36	0.064	77.0	0.99604	0.87	11.00	7
363	12.5	0.460	0.63	0.071	15.0	0.99880	0.87	10.20	5
906	7.2	0.540	0.27	0.084	78.0	0.99640	0.71	11.00	5
426	6.4	0.670	0.08	0.045	48.0	0.99490	0.49	11.40	6
1251	7.5	0.580	0.14	0.077	60.0	0.99630	0.59	9.80	5
1274	7.8	0.580	0.13	0.102	36.0	0.99440	0.53	11.20	6
346	6.6	0.815	0.02	0.072	34.0	0.99550	0.89	12.30	7
1105	6.3	0.570	0.28	0.048	49.0	0.99374	0.60	12.80	5
1564	7.2	0.695	0.13	0.076	20.0	0.99546	0.54	10.10	5
643	9.5	0.590	0.44	0.071	68.0	0.99920	0.63	9.50	5
844	9.9	0.250	0.46	0.062	42.0	0.99590	0.83	10.60	6
1050	7.1	0.430	0.17	0.082	51.0	0.99634	0.64	10.40	5

```
В тренировочную выборку попадает 1119 обьектов
```

Рис.13

```
[363]:
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,data.columns)))
```

```
1 component: 99.57% of initial variance
-0.005 x fixed acidity + 0.000 x volatile acidity + 0.000 x citric acid + 0.000 x chlorides + 1.000 x total sulfur dioxide +
0.000 x density + 0.000 x sulphates + -0.007 x alcohol + -0.005 x quality
2 component: 0.27% of initial variance
0.994 x fixed acidity + -0.026 x volatile acidity + 0.077 x citric acid + 0.003 x chlorides + 0.005 x total sulfur dioxide +
0.001 x density + 0.017 x sulphates + -0.056 x alcohol + 0.040 x quality
3 component: 0.12% of initial variance
0.023 x fixed acidity + -0.050 x volatile acidity + 0.036 x citric acid + -0.008 x chlorides + 0.009 x total sulfur dioxide
+ -0.001 x density + 0.031 x sulphates + 0.857 x alcohol + 0.510 x quality
4 component: 0.04% of initial variance
0.066 x fixed acidity + 0.063 x volatile acidity + -0.005 x citric acid + -0.001 x chlorides + -0.000 x total sulfur dioxide
+ -0.000 x density + -0.043 x sulphates + 0.511 x alcohol + -0.854 x quality
5 component: 0.0% of initial variance
-0.065 x fixed acidity + -0.677 x volatile acidity + 0.534 x citric acid + 0.060 x chlorides + -0.001 x total sulfur dioxide
+ -0.000 x density + 0.489 x sulphates + -0.021 x alcohol + -0.095 x quality
6 component: 0.0% of initial variance
0.014 x fixed acidity + 0.460 x volatile acidity + -0.211 x citric acid + 0.112 x chlorides + -0.000 x total sulfur dioxide
+ 0.001 x density + 0.855 x sulphates + 0.007 x alcohol + -0.003 x quality
7 component: 0.0% of initial variance
-0.048 x fixed acidity + 0.562 x volatile acidity + 0.806 x citric acid + 0.134 x chlorides + -0.001 x total sulfur dioxide
+ 0.001 x density + -0.120 x sulphates + -0.013 x alcohol + 0.031 x quality
8 component: 0.0% of initial variance
0.006 x fixed acidity + -0.088 x volatile acidity + -0.118 x citric acid + 0.983 x chlorides + 0.000 x total sulfur dioxide
+ -0.001 x density + -0.110 x sulphates + 0.010 x alcohol + 0.005 x quality
9 component: 0.0% of initial variance
-0.001 x fixed acidity + -0.001 x volatile acidity + -0.000 x citric acid + 0.001 x chlorides + -0.000 x total sulfur dioxid
e + 1.000 x density + -0.001 x sulphates + 0.001 x alcohol + 0.000 x quality
```

Для обьяснения 99 % дисперсии после применения метода PCA хватает одной компоненты.

Рис.14

Для обьяснения 99% дисперсии после применения метода PCA хватает одной компоненты.

```
In [364]: 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(3, c='b')

Out[364]: <matplotlib.lines.Line2D at 0x1eccdb9d710>
```

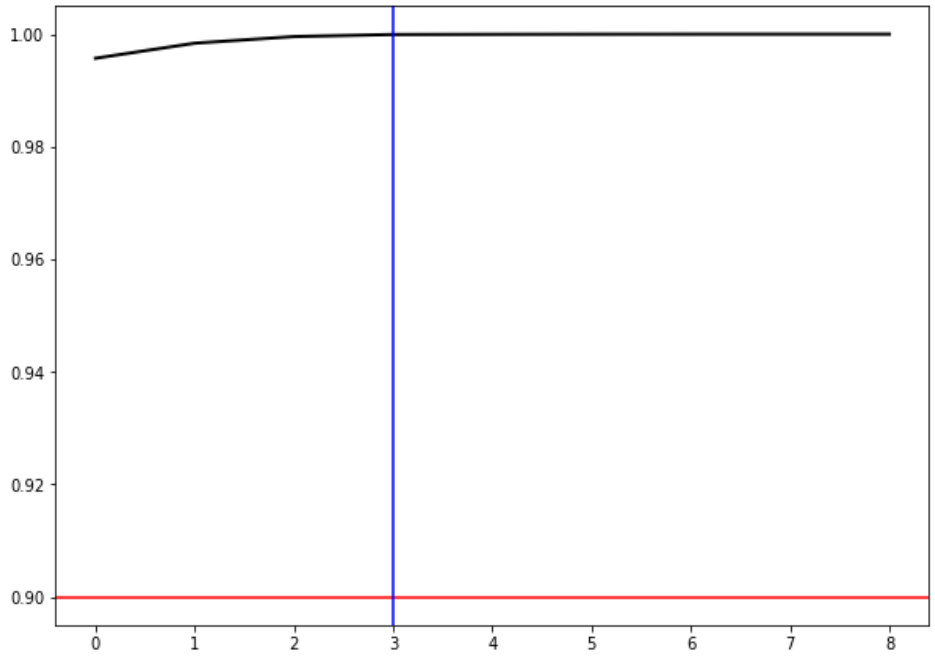

Рис.15

Вывод:

В анализе данных, как и в любом другом анализе, порой бывает не лишним создать упрощенную модель, максимально точно описывающую реальное положение дел. Часто бывает так, что признаки довольно сильно зависят друг от друга и их одновременное наличие избыточно и для упрощения нашего вывода существует метод главных компонент PCA.

Метод главных компонент (англ. principal component analysis, PCA) — один из основных способов уменьшить размерность данных, потеряв наименьшее количество информации.В данном случае мы обработали наши данные и получили что одной переменной можем описать 99% наших данных.

4. Заключение

В ходе летней практики мы научились пользоваться такими аналитическим языком, как R, а так же научились пользоваться языком Python в аналитических целях.
Было выполнено 3 задания:

В задаче 1 мы анализировали датасет с швейцарскими кантонами, проверяя от чего зависит рождаемость , путем метода наименьших квадратов.

В задаче 2 мы пытались описать, как различные параметры влияют на заработную плату, основываясь на Данных "Российского мониторинга экономического положения и здоровья населения НИУ-ВШЭ". Было отобрано не так много переменных , поэтому точность построенных моделей не очень высока, но и по ним можно сделать выводу, что большую заработную плату получают люди, имеющее высшее образование, проживающие в крупных городах, работающие больше, чем полагается по законодательной норме.

В 3тьей задаче мы анализировали данные по качеству вина,пользуясь методом главных компонент PCA,пытаясь уменьшить размерность данных, потеряв наименьшее количество информации.

5. Список литературы

1. Основы анализа данных ВШЭ // youtube.com URL: https://www.youtube.com/playlist?list=PLu5flfwrnSD5d02G9YJcDv30Fp5_70-sI (дата обращения: 24.04.2019).

2. Кабаков Р. И. R в действии. Анализ и визуализация данных на языке R. - 1-е изд. - М.: ДМК Пресс, 2014.

3.Тоби Сегаран.” Программируем коллективный разум. // Пер. с англ.//СПб: Символ-Плюс, 2008. – 368 с “

4. С. Попов. “Финансовая отчетность в эпоху экономики знаний”. // Библиотека Креативной экономики. – 2005. [Электронный ресурс]. URL: https://habr.com/ru/company/ods/blog/325654/

Приложение

Задача 1

```
library("IMTest")
library("rlms")
library("dplyr")
library("GGally")
library("car")
library("sandwich")
data2=swiss
data2
help(swiss)

#glimpse(data)

data=dplyr::select(data2,Fertility,Agriculture, Catholic, Infant.Mortality)
data
ggpairs(data)



data["Log_Agriculture"]=log(data$Agriculture)
data["Log_Catholic"]=log(data$Catholic)
data["Log_Infant.Mortality"]=log(data$Infant.Mortality)


Agr1<-as.character(data$Agriculture)
Agr2<-lapply(Agr1,as.integer)
Agr3<-as.numeric(unlist(Agr2))
data["Agr"]=(Agr3-mean(Agr3))/sqrt(var(Agr3))

Cat1<-as.character(data$Catholic)
Cat2<-lapply(Cat1,as.integer)
Cat3<-as.numeric(unlist(Cat2))
data["Cat"]=(Cat3-mean(Cat3))/sqrt(var(Cat3))

IM1<-as.character(data$Infant.Mortality)
IM2<-lapply(IM1,as.integer)
IM3<-as.numeric(unlist(IM2))
data["IM"]=(IM3-mean(IM3))/sqrt(var(IM3))



Ln_Agr1<-as.character(data$Log_Agriculture)
Ln_Agr2<-lapply(Ln_Agr1,as.integer)
Ln_Agr3<-as.numeric(unlist(Ln_Agr2))
data["Ln_Agr"]=(Ln_Agr3-mean(Ln_Agr3))/sqrt(var(Ln_Agr3))

Ln_Cat1<-as.character(data$Log_Catholic)
Ln_Cat2<-lapply(Ln_Cat1,as.integer)
```

```
Ln_Cat3<-as.numeric(unlist(Ln_Cat2))
data["Ln_Cat"]=(Ln_Cat3-mean(Ln_Cat3))/sqrt(var(Ln_Cat3))

LN_IM1<-as.character(data$Log_Infant.Mortality)
LN_IM2<-lapply(LN_IM1,as.integer)
LN_IM3<-as.numeric(unlist(LN_IM2))
data["Ln_IM"]=(LN_IM3-mean(LN_IM3))/sqrt(var(LN_IM3))


model1<-lm(Fertility~Agr+Cat+IM,data)
model1
summary(model1)#R^2=0.3449
#p -()(*)(**)

model2<-lm(Fertility~Cat+IM,data)
model2
summary(model2)#R^2=0.302
#p-(**)(**)

model3<-lm(Fertility~IM,data)
model3
summary(model3)#R^2=0.1562
#p-(**)

model4<-lm(Fertility~Cat,data)
model4
summary(model4)#R^2=0.1964
#p-(**)

model5<-lm(Fertility~Agr,data)
model5
summary(model5)#R^2=0.1057
#p-(*)

model6<-lm(Fertility~Ln_Agr+Ln_Cat+Ln_IM,data)
model6
summary(model6)#R^2=0.3766
#p-(**)(*)(**)

model7<-lm(Fertility~Ln_Agr+Ln_IM,data)
model7
summary(model7)#R^2=0.3146
#p-(**)(***)

model8<-lm(Fertility~Ln_IM+Cat,data)
model8
summary(model8)#R^2=0.3279
#p-(**)(***)

model9<-lm(Fertility~Ln_IM+Cat+Ln_Agr,data)
model9
summary(model9)#R^2=0.42317
```

```
#p-(***)(**)(**)

model10<-lm(Fertility~I(Ln_IM^2)+Cat+Ln_Agr,data)
model10
summary(model10)#R^2=0.4237
#p-(***)(**)(**)

model12<-lm(Fertility~Ln_IM+I(Cat^2)+I(Agr^2),data)#Best
model12
summary(model12)#R^2=0.564

model13<-lm(Fertility~I(Ln_IM*Cat)+Cat+I(Ln_Agr^2),data)
model13
summary(model13)#R^2=0.3536


model14<-lm(Fertility~I(Ln_IM*Cat)+I(Ln_Cat*Agr)+I(Ln_Agr*IM),data)
model14
summary(model14)#R^2=0.1801


model15<-lm(Fertility~I(IM*Cat)+I(Cat*Ln_Agr)+I(Agr^2),data)
model15
summary(model15)#R=0.1341


model16<-lm(Fertility~I(IM*Cat)+I(Cat*Ln_Agr)+I(Agr*Ln_IM),data)
model16
summary(model16)#R=0.1341
```

Задача 2

```
library("rlms")
library("dplyr")
library("GGally")
library("car")
library("sandwich")
library("memisc")


data <- rlms_read("C:\\r19i_os26c.sav")
glimpse(data)
data2 = dplyr::select(data, idind, oj13.2, oh5, o_marst, o_educ, o_age, status, oj6.2)
#ggpairs(data2)
data2 = na.omit(data2)



data2["log_age"]=log(data2$o_age)
data2["log_dur"]=log(data2$oj6.2)
```

```
data2["wed1"]=data2$o_marst
data2["wed1"]=0
data2$wed1[which(data2$o_marst=='2')] <- 1
data2$wed1[which(data2$o_marst=='6')] <- 1
#data2$wed1 = as.numeric(data2$wed1)


data2["wed2"]=data2$o_marst
data2["wed2"]=0
data2$wed2[which(data2$o_marst=='4')] <- 1
data2$wed2[which(data2$o_marst=='5')] <- 1
#data2$wed2 = as.numeric(data2$wed2)


data2["wed3"]=data2$o_marst
data2["wed3"]=0
data2$wed3[which(data2$o_marst=='1')] <- 1
#data2$wed3 = as.numeric(data2$wed3)


data2["sex"]=data2$oh5
data2$sex[which(data2$sex!='1')] <- 0
data2$sex[which(data2$sex=='1')] <- 1
data2$sex = as.numeric(data2$sex)


data2["city_status"]=data2$status
data2["city_status"]=0
data2$city_status[which(data2$status=='1')] <- 1
data2$city_status[which(data2$status=='2')] <- 1
#data2$city_status = as.numeric(data2$city_status)


data2["higher_educ"]=data2$o_educ
data2["higher_educ"]=0
data2$higher_educ[which(data2$o_educ=='21')] <- 1
data2$higher_educ[which(data2$o_educ=='22')] <- 1
data2$higher_educ[which(data2$o_educ=='23')] <- 1


sal = as.numeric(data2$oj13.2)
sal1 = as.character(data2$oj13.2)
sal2 = lapply(sal1, as.integer)
sal = as.numeric(unlist(sal2))
data2["salary"] = (sal - mean(sal)) / sqrt(var(sal))

age1 = as.character(data2$o_age)
age2 = lapply(age1, as.integer)
age3 = as.numeric(unlist(age2))
data2["age"]= (age3 - mean(age3)) / sqrt(var(age3))
```

```
ln_age1 = as.character(data2$log_age)
ln_age2 = lapply(ln_age1, as.integer)
ln_age3 = as.numeric(unlist(ln_age2))
data2["ln_age"]= (ln_age3 - mean(ln_age3)) / sqrt(var(ln_age3))


dur1 = as.character(data2$oj6.2)
dur2 = lapply(dur1, as.integer)
dur3 = as.numeric(unlist(dur2))
data2["dur"] = (dur3 - mean(dur3)) / sqrt(var(dur3))

ln_dur1 = as.character(data2$log_dur)
ln_dur2 = lapply(ln_dur1, as.integer)
ln_dur3 = as.numeric(unlist(ln_dur2))
data2["ln_dur"] = (ln_dur3 - mean(ln_dur3)) / sqrt(var(ln_dur3))



model1 = lm(salary~wed1+wed2+wed3+sex+city_status+higher_educ+age+dur, data2)
model1
summary(model1) #R^2 = 0.137
vif(model1)

model2 = lm(salary~sex+city_status+higher_educ+age+dur, data2)
model2
summary(model2) #R^2 = 0.135
vif(model2)

model397 = lm(salary~wed2+wed3+sex+city_status+higher_educ+age+dur, data2)
model397 #
summary(model397) #R^2 = 0.1371
vif(model397)


model3 = lm(salary~wed3+sex+city_status+higher_educ+age+dur, data2)
model3 #
summary(model3) #R^2 = 0.1372
vif(model3)


model4 = lm(salary~wed2+wed3+sex+city_status+higher_educ+dur, data2)
model4 #
summary(model4) #R^2 = 0.1345
vif(model4)

model5 = lm(salary~wed3+sex+city_status+higher_educ+dur, data2)
model5 #
summary(model5) #R^2 = 0.1031
vif(model5)

model21 = lm(salary~wed2+wed3+sex+city_status+higher_educ+age+ln_dur, data2)
```

```
model21 #
summary(model21) #R^2 = 0.1385
vif(model21)

model33 = lm(salary~wed2+wed3+sex+city_status+higher_educ+ln_age+ln_dur, data2)
model33 #
summary(model33) #R^2 = 0.141
vif(model33)

model34 = lm(salary~wed2+wed3+sex+city_status+higher_educ+ln_age+ln_dur, data2)
model34 #
summary(model34) #R^2 = 0.141
vif(model34)



model35 = lm(salary~wed2+wed3+sex+city_status+higher_educ+I(age^2)+dur, data2)
model35
summary(model35) #R^2 = 0.148
vif(model35)

model36 = lm(salary~wed2+I(wed3^2)+sex+city_status+higher_educ+age+dur, data2)
model36
summary(model36) #R^2 = 0.1371
vif(model36)

model37 = lm(salary~wed1+higher_educ, data2)
model37
summary(model37) #R^2=0.04646
vif(model37)

model38 = lm(salary~wed3+city_status, data2)
model38 #
summary(model38) #R^2=0.0268
vif(model38)

model51=lm(salary~wed2+city_status, data2)
model51
summary(model51)
vif(model51)
#R^2=0.04
model52=lm(salary~wed2+higher_educ, data2)
model52
summary(model52)
vif(model52)
#R^2=0.03
compare5 = mtable(model34,model33,model21,model52,model51)
compare5
```