



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА - Российский технологический университет»

РТУ МИРЭА

Институт искусственного интеллекта

Кафедра высшей математики

ОТЧЁТ ПО НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ
(получение первичных навыков научно-исследовательской работы)

Тема НИР: Предсказание спроса на медицинскую страховку (kaggle.com)
приказ университета о направлении на НИР
от «9» февраля 2023 г. № 735 - С

Отчет представлен к
рассмотрению:
Студент группы КМБО-04-
22

Кемаев И.О.
(расшифровка подписи)
«9 мар 2023 г.

Отчет утвержден.
Допущен к защите:

Руководитель НИР от
кафедры

Петрусевич Д.А.
(расшифровка подписи)
«9 мар 2023 г.

Москва 2023



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА - Российский технологический университет»

РТУ МИРЭА

ЗАДАНИЕ

на НАУЧНО-ИССЛЕДОВАТЕЛЬСКУЮ РАБОТУ

(получение первичных навыков научно-исследовательской работы)

Студенту 1 курса учебной группы КМБО-04-22 института искусственного
интеллекта Кемаеву Ивану Олеговичу

(фамилия, имя и отчество)

Место и время НИР: Институт искусственного интеллекта, кафедра высшей математики
Время НИР: с «09» февраля 2023 по «31» мая 2023

Должность на НИР: практикант

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

2. СОДЕРЖАНИЕ НИР:

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»).

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 кластеризации («k ближайших соседей»); построением модели линейной регрессии

3. ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ: обработка данных о клиентах банка (kaggle.com).

4. ОГРАНИЦИОННО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ:

Построить модель предсказания: будет ли клиент брать страховку, включающую покрытие рисков, связанных с коронавирусом? Оценить вклад каждой компоненты.

Есть ли выбросы среди данных, какие? Применить алгоритмы кластеризации.

Охарактеризовать кластеры

Заведующий кафедрой
высшей математики

«09» февраля 2023 г.

Ю.И.Худак

СОГЛАСОВАНО

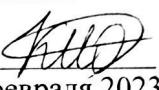
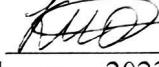
Руководитель НИР от кафедры:
«09» февраля 2023 г.

(Петрусеевич Д.А.)
(фамилия и инициалы)

Задание получило:
«09» февраля 2023 г.

(Кемаев И.О.)
(фамилия и инициалы)

ИНСТРУКТАЖ ПРОВЕДЕН:

Вид мероприятия	ФИО ответственного, подпись, дата	ФИО студента, подпись, дата
Охрана труда	Петрусевич Д.А.  «09» февраля 2023 г.	Кемаев И.О.  «09» февраля 2023 г.
Техника безопасности	Петрусевич Д.А.  «09» февраля 2023 г.	Кемаев И.О.  «09» февраля 2023 г.
Пожарная безопасность	Петрусевич Д.А.  «09» февраля 2023 г.	Кемаев И.О.  «09» февраля 2023 г.
Правила внутреннего распорядка	Петрусевич Д.А.  «09» февраля 2023 г.	Кемаев И.О.  «09» февраля 2023 г.



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

**«МИРЭА - Российский технологический университет»
РТУ МИРЭА**

РАБОЧИЙ ГРАФИК ПРОВЕДЕНИЯ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЫ (получение первичных навыков научно-исследовательской работы)

студента Кемаева И.О. 1 курса группы КМБО-04-22 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	09.02.2023	Выбор темы НИР. Пройти инструктаж по технике безопасности	✓
1	09.02.2023	Вводная установочная лекция	✓
2	18.02.2023	Построение и оценка парной регрессии с помощью языка R	✓
3	25.02.2023	Построение и оценка множественной регрессии с помощью языка R	✓
4	04.03.2023	Построение доверительных интервалов. Обработка факторных переменных. Мультиколлинеарность	✓
5	11.03.2023	Гетероскедастичность	✓
6	18.03.2023	Классификация	✓
7	25.03.2023	Кластеризация. Предобработка данных	✓
8	01.04.2023	Метод главных компонент	✓
9	08.04.2023	Ансамбли классификаторов.	✓

		Беггинг. Бустинг	
16	27.05.2023	Представление отчётных материалов по НИР и их защиты. Передача обобщённых материалов на кафедру для архивного хранения	✓
		Зачётная аттестация	✓

Согласовано:

Заведующий кафедрой

/ ФИО / Худак Ю.И.

Руководитель НИР от
кафедры

/ ФИО / Петрусевич Д.А.

Обучающийся

/ ФИО / Кемаев И.О.

Оглавление

Задача №1: Решение и выводы	3
Задача №2: Решение и выводы	5
Задача №3: Решение и выводы	12
Задача №4: Решение и выводы	15
Задача №5: Решение и выводы	20
Задача №6: Решение и выводы	24
Заключение: Краткая характеристика выводов по каждой задаче	33
Список использованной литературы (по ГОСТу)	36
Приложения:	37
Код для решения задачи №1	37
Код для решения задачи №2	39
Код для решения задачи №3	47
Код для решения задачи №4	59
Код для решения задачи №5	67
Код для решения задачи №6	88

Отчет по задаче №1

Задача 1

Необходимо загрузить данные из указанного набора и произвести следующие действия.

Набор данных: Swiss.

Объясняемая переменная: Examination

Регрессоры: Fertility, Education

- Оцените среднее значение, дисперсию и СКО переменных Examination, Fertility, Education

Examination: Среднее арифметическое = 16.48936, Дисперсия = 63.64662, СКО = 7.977883.

Fertility: Среднее арифметическое = 70.14255, Дисперсия = 156.0425, СКО = 12.4917.

Education: Среднее арифметическое = 10.97872, Дисперсия = 92.45606, СКО = 9.615407.

Результат посчитанных функций в Приложении 1.

- Постройте зависимости вида $y = a + bx$, где y – объясняемая переменная, x – регрессор (для каждого варианта по две зависимости). Оцените, насколько «хороша» модель по коэффициенту детерминации. Оцените, есть ли взаимосвязь между объясняемой переменной и объясняющей переменной (по значению р-статастики, «количество звездочек» у регрессора в модели).

Зависимость *Examination~Fertility*. $R^2 = 40\%$. Значение коэффициента детерминации меньше 80%, поэтому заключаем, что линейной зависимости нет. $p-value = 9.45*(10^{-7})$, *** у Fertility означают сильную взаимосвязь Fertility и Examination. Fertility и Examination связаны отрицательно (-0.41)

Таблица 1.1. Характеристики модели зависимости параметра *Examination* от параметра *Fertility* в наборе данных Swiss.

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

Зависимость *Examination~Education*. $R^2 = 47\%$. Значение коэффициента детерминации меньше 80%, поэтому заключаем, что линейной зависимости нет. $p-$

$\text{value} = 4.811*(10^{**-8})$, *** у *Education* означают сильную взаимосвязь *Education* и *Examination*. *Education* и *Examination* связаны положительно (0.57)

Таблица 1.2. Характеристики модели зависимости параметра *Examination* от параметра *Education* в наборе данных Swiss.

Параметр Характеристики \	Значение	Std. Error	t value	Pr(> t)	Уровень значимости
(Intercept)	10.12748	1.28589	7.876	5.23e-10	***
Education	0.57947	0.08852	6.546	4.81e-08	***

Код решения задачи в Приложении 1.

Во время практики вы освоили расчет среднего арифметического, дисперсии и среднеквадратического отклонения с использованием встроенных функций. Эти статистические метрики позволяют получить информацию о распределении данных и их разбросе.

Относительно зависимости между переменными, вы сделали вывод, что показатель "Examination" не имеет линейной зависимости от переменных "Education" и "Fertility".

Education и *Examination* взаимосвязаны положительно (0.57)

Отчет по задаче №2

Задача 2

Необходимо загрузить данные из указанного набора и произвести следующие действия.

Набор данных: Mtcars.

Объясняемая переменная: *mpg*.

Регрессоры: *cyl, disp, hp, drat*.

1. 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.

Проверим линейную регрессию $disp \sim cyl, hp, drat$. R^2 в этой модели около 82%, поэтому мы заключаем, что параметр *disp* зависит от других регрессоров линейно и может быть удален из рассмотрения при построении математических моделей.

Значения коэффициентов в этой модели и их стандартные ошибки представлены в таблице 1.

Зависимость $cyl \sim disp, hp, drat$. $R^2 = 85\%$. Значение коэффициента детерминации больше 82% предыдущей зависимости, поэтому заключаем, что линейная зависимость есть. Нежелательно использовать *Cyl* в линейных моделях. Значения коэффициентов в этой модели и их стандартные ошибки представлены в таблице 2.

В регрессии $hp \sim cyl, disp, drat$ значение R^2 близко к 72%. Параметр *hp* можно использовать в линейной регрессии вместе с *cyl, disp, drat*. Значения коэффициентов в этой модели и их стандартные ошибки представлены в таблице 3.

Проверим линейную регрессию $drat \sim disp, hp, cyl$. R^2 в этой модели около 55%, поэтому мы заключаем, что параметр *drat* независит от других регрессоров линейно и может быть использован при построении математических моделей. Значения коэффициентов в этой модели и их стандартные ошибки представлены в таблице 4.

Таким образом, заключаем, что самая большая линейная зависимость у регрессора *cyl* (порядка 85%), поэтому удаляем его из дальнейшего рассмотрения, а остальные указанные в задании регрессоры использовать при построении моделей линейной регрессии можно.

Таблица 2.1. Характеристики модели зависимости параметра *disp* от параметров *cyl, drat, hp* в наборе данных Mtcars

Параметр \ Характеристики	Значение	Std. Error	t value	Pr(> t)	Уровень значимости
(Intercept)	105.8694	137.1689	0.772	0.44669	
cyl	39.3327	дек-13	3.119	0.00418	**
hp	0.4039	0.2625	1.539	0.13514	
drat	-49.4274	26.1188	-1.892	0.06882	.

Таблица 2.2. Характеристики модели зависимости параметра *cyl* от параметров *disp*, *drat*, *hp* в наборе данных Mtcars

Параметр \ Характеристики	Значение	Std. Error	t value	Pr(> t)	Уровень значимости
(Intercept)	5.701261	1.428829	3.990	0.000431	***
disp	0.006555	0.002102	3.119	0.004178	**
hp	0.009900	0.002993	3.308	0.002589	**
drat	-0.689084	0.333579	-2.066	0.048220	*

Таблица 2.3. Характеристики модели зависимости параметра *hp* от параметров *cyl*, *drat*, *disp* в наборе данных Mtcars

Параметр \ Характеристики	Значение	Std. Error	t value	Pr(> t)	Уровень значимости
(Intercept)	-219.3802	86.3820	-2.540	0.01693	*
cyl	28.3807	8.5802	3.308	0.00259	**
disp	0.1930	0.1254	1.539	0.13514	
drat	40.5768	17.5732	2.309	0.02854	*

Таблица 2.4. Характеристики модели зависимости параметра $drat$ от параметров $cyl, disp, hp$ в наборе данных Mtcars

Параметр Характеристики \	Значение	Std. Error	t value	Pr(> t)	Уровень значимости
(Intercept)	4.735125	0.301898	15.684	2.13e-15	***
disp	-0.002294	0.001212	-1.892	0.0688	.
hp	0.003942	0.001707	2.309	0.0285	*
cyl	-0.191917	0.092905	-2.066	0.0482	*

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

Проверим линейную регрессию $mpg \sim drat, disp, hp$. Характеристики модели зависимости mpg от регрессоров $disp, drat, hp$ приведены в таблице 5.

Таблица 2.5. Характеристики модели зависимости параметра mpg от параметров $drat, disp, hp$ в наборе данных Mtcars

Параметр Характеристики \	Значение	Std. Error	t value	Pr(> t)	Уровень значимости
(Intercept)	19.344293	6.370882	3.0360	0.00513	**
disp	-0.019232	0.009371	-2.052	0.04960	*
hp	-0.031229	0.013345	-2.340	0.02663	*
drat	2.714975	1.487366	1.825	0.07863	.

У данной модели $R^2=75\%$, можно сделать вывод что она очень хороша. $Disp, hp$ близки к 0.05 по р-статистике (mpg в основном описывается через них) и также они зависят от mpg отрицательно, у $drat$ одна *, значит он менее сильно влияет на mpg и также он зависит положительно от mpg .

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

При решении этой задачи были проверены модели, в которых были добавлены параметры: $mpg \sim log(disp) + (hp) + (drat)$, $mpg \sim log(disp) + log(hp) + (drat)$, $mpg \sim log(disp) + log(hp) + log(drat)$. Выбираем последнюю модель с логарифмами от всех регрессоров так как ее $R^2=82\%$ (> чем у всех предыдущих моделей)

Таблица 2.6. Характеристики модели зависимости параметра *mpg* от логарифмов регрессоров *drat*, *disp*, *hp* в наборе данных Mtcars

Параметр Характеристики	\	Значение	Std. Error	t value	Pr(> t)	Уровень значимости
(Intercept)		69.541	11.443	2,83e+04	1.49e-06	***
I(log(disp))		-6.514	1,83e+04	-3.177	0.0036	**
I(log(hp))		-3.523	1.986	-1.774	0.0869	.
I(log(drat))		1.714	4.917	0.349	0.7300	

Так как $\log(drat)$ в новой модели не влияет на *mpg* (нет звездочек и даже точки), не берем его логарифм (здесь можно его вообще убрать).

mpg~log(disp)+log(hp) - наилучшая модель с $R^2=82,9$.

Код решения задачи и сведения о проверенных моделях приведены в Приложении 2.

4. Вводим в модель всевозможные попарные произведения регрессоров и их квадраты.

При решении этой задачи были проверены модели, в которых были добавлены попарные произведения регрессоров и их квадраты :

*mpg~(disp*hp)+drat* → $R^2=71\%$, *mpg~(disp*drat)+hp* → $R^2=68\%$, *mpg~(hp*drat)+disp* → $R^2=72\%$, *mpg~(disp^2)+(hp^2)+(drat^2),data* → $R^2=69\%$, *mpg~(disp^2*hp^2)+(drat)* → $R^2=66\%$, *mpg~(disp^2*hp^2)+(drat^2)* → $R^2=67\%$, *mpg~(drat^2*hp^2)+(disp^2)* → $R^2=60\%$, *mpg~(drat^2*hp^2)+(disp)* → $R^2=70\%$, *mpg~(drat^2*disp^2)+(hp^2)* → $R^2=56\%$, *mpg~(drat^2*disp^2)+(hp)* → $R^2=62\%$,

*mpg~(log(disp)^2)+(log(hp)^2)+(log(drat)^2)+(log(hp)*log(drat))* → $R^2=81\%$,

*mpg~(log(disp))+(log(hp))+(log(drat))+(log(hp)*log(drat))* → $R^2=82\%$,

*mpg~(log(disp))+(log(hp))+(log(drat))+(log(disp)*log(hp))* → $R^2=83,8\%$

Таким образом из 4ого пункта лучше всех по доле объясненного разброса в данных $R^2 = 83,8\% : mpg~(log(disp))+(log(hp))+(log(drat))+(log(disp)*log(hp))$

После выполнения первой части практической работы №2 можно сделать вывод что для набора данных *mtcars* и четырех переменных *mpg*, *disp*, *hp*, *drat* применение логарифмической функции к регрессорам может привести к линеаризации отношения

между переменными и улучшению качества модели, устраниТЬ гетероскедастичность или облегчить интерпретацию коэффициентов.

Disp, hp близки к 0.05 по р-статистке (*mpg* в основном описывается через них) и также они зависят от *mpg* отрицательно, у *drat* одна *, значит этот менее сильно влияет на *mpg* и также он зависит положительно от *mpg*.

Использование логарифмов регрессоров в модели может иметь несколько причин, которые могут приводить к улучшению показателя R^2:

1.Линейность отношения: В некоторых случаях зависимость между регрессорами и зависимой переменной может быть нелинейной. Применение логарифмической функции к регрессорам может привести к линеаризации отношения между переменными и улучшению качества модели.

2.Устранение гетероскедастичности: Гетероскедастичность означает, что дисперсия ошибок модели изменяется в зависимости от значений регрессоров. Если гетероскедастичность присутствует в данных, логарифмирование регрессоров может помочь устраниТЬ или уменьшить этот эффект, что может привести к более точной модели и высокому значению R^2.

3. Интерпретация коэффициентов: Использование логарифмической шкалы для регрессоров может упростить интерпретацию коэффициентов модели. Логарифмическое преобразование может привести к изменению масштаба и интерпретации коэффициентов как процентных изменений в зависимой переменной при изменении регрессора на одну единицу.

Вторая часть практической работы №2

Для зависимости, построенной при решении практического задания №2, оцените:

1,2) Доверительные интервалы для каждого коэффициента с p=95% и проверка статистической гипотезы коэф = 0 :

Лучшая модель из первой части работы:
*mpg~(log(disp))+(log(hp))+(log(drat))+(log(disp)*log(hp))*

Число степеней свободы модели: 37-4=33, где 37 - число параметров набора данных *mtcars*, 4 - количество линейно независимых переменных.

t-критериq Стьюдента = 2.034515.

Интервал для свободного коэффициента : [285.876751224019 , 58.2592487759815]

> Отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале).

Интервал для log(disp): [-4.43748771814166 , -46.9385122818583]

> Отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале).

Интервал для log(hp) : [-1.4508971835809 , -44.4931028164191]

> Отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале).

Интервал для log(drat) : [8.41633865466867 , -12.3723386546687]

> Этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале).

Интервал для log((disp) * log(hp)) : [7.89682444674001, 0.334824446740012]

> Этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале).

3. Доверительный интервал для одного прогноза:

Таблица 1.7. Результаты предсказания лучшей модели.

Предсказание	fit	lwr	upr
1	71.81007	43.79245	99.8277

Доверительный интервал для одного прогноза [43.79245, 99.8277]

После выполнения второй части работы №2 можно понять, что доверительные интервалы модели в линейной регрессии являются статистическими инструментами, которые позволяют оценить неопределенность и уверенность в оценках коэффициентов регрессии. Они предоставляют информацию о диапазоне значений, в котором, с определенной вероятностью, находится истинное значение параметра модели.

На основе этих данных мы можем сделать следующие выводы:

Свободный коэффициент, log(disp) и log(hp) значимо отличаются от нуля. Это

означает, что эти переменные оказывают влияние на зависимую переменную (mpg) в модели.

Коэффициент $\log(\text{drat})$ не значимо отличается от нуля, что может говорить о его незначительном влиянии на зависимую переменную.

Коэффициент $\log((\text{disp}) * \log(\text{hp}))$ также не значимо отличается от нуля, что может указывать на его ограниченное влияние на модель.

На основе этого доверительного интервала мы можем сделать следующие выводы:

Предсказанное значение составляет 71.81007.

С вероятностью 95%, фактическое значение прогнозируемой переменной будет находиться в интервале [43.79245, 99.8277].

Таким образом, мы можем быть уверены с 95% вероятностью, что фактическое значение прогнозируемой переменной будет в указанном интервале. Это позволяет нам оценить диапазон значений и измерить уровень неопределенности при прогнозировании.

Отчет по задаче №3

Задача 3

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

Построение вспомогательной регрессии $\text{salary} \sim \text{age} + \text{sex} + \text{higher_educ} + \text{status2} + \text{dur} + \text{wed1} + \text{wed2} + \text{wed3}$. Переменные все хорошие , vif < 5

2. Поэкспериментируйте с функциями вещественных параметров: используйте логарифмы, степени (хотя бы от 0.1 до 2 с шагом 0.1), произведения вещественных регрессоров

Модели с логарифмами , степенями и произведениями

$\text{salary} \sim (\log(\text{age}+3)) + \text{sex} + \text{higher_educ} + \text{status2} + \text{dur} + \text{wed1} + \text{wed2} + \text{wed3}$
 $\text{adj } R^2 = 0.025$, параметры $I(\log(\text{age}+3)) + \text{higher_educ} + \text{dur} + \text{wed1} + \text{wed2}$ значимые

$\text{salary} \sim (\log(\text{age}+3)) + \text{sex} + \text{higher_educ} + \text{status2} + \text{dur} + \text{wed1} + \text{wed2} + \text{wed3}$
 $\text{adj } R^2 = 0.0256$,параметры $I(\log(\text{age}+3)) + \text{higher_educ} + \text{dur} + \text{wed1} + \text{wed2}$ значимые

$\text{salary} \sim (\log(\text{age}+3)) + \text{sex} + \text{higher_educ} + \text{status2} + \text{dur} + (\text{age} * \text{dur}) + \text{wed1} + \text{wed2} + \text{wed3}$, $\text{adj } R^2 = 0.02602$, параметры параметры $I(\log(\text{age}+3)) + \text{higher_educ} + \text{dur} + \text{wed1} + \text{wed2} + I(\text{age} * \text{dur})$ значимые

$\text{salary} \sim (\log(\text{age}+3)) + \text{sex} + \text{higher_educ} + \text{status2} + I(\text{dur}^2) + I(\text{age} * \text{dur}) + \text{wed1} + \text{wed2} + \text{wed3}$, $\text{adj } R^2 = 0.02602$, параметры $\text{age} + \text{higher_educ} + I(\text{dur}^2) + I(\text{age} * \text{dur}) + \text{wed1} + \text{wed2}$ значимые

$\text{salary} \sim I((\text{age}+3)^{1.5}) + \text{sex} + \text{higher_educ} + \text{status2} + (\text{dur}^2) + I(\text{age} * \text{dur}) + \text{wed1} + \text{wed2} + \text{wed3}$, $\text{adj } R^2 = 0.0258$, параметры $I(\log(\text{age}+3)) + \text{higher_educ} + \text{wed1} + \text{wed2} + (\text{age} * \text{dur}) + (\text{dur}^2)$ - значимые; менее значимые $(\text{age} * \text{dur}) + \text{wed3}$

$\text{salary} \sim (\log(\text{age}+3)^{(0.1)}) + \text{sex} + \text{higher_educ} + \text{status2} + (\text{age} * \text{dur}) + (\text{dur}) + \text{wed1} + \text{wed2} + \text{wed3}$, $\text{adj } R^2 = 0.02601$, параметры $(\log(\text{age}+3)) + \text{higher_educ} + \text{wed1} + \text{wed2} + (\text{dur})$ - значимые; менее значимые: $(\text{age} * \text{dur})$

$\text{salary} \sim I((\text{age}+3)^{(0.1)}) + \text{sex} + \text{higher_educ} + \text{status2} + I((\text{dur}+1)^{(0.9)}) + I(\text{age} * \text{dur}) + \text{wed1} + \text{wed2} + \text{wed3}$; $\text{adj } R^2 = 0.02602$, параметры $I(\log(\text{age}+3)^{(0.1)}) + \text{higher_educ}$

$+wed1 + wed2 + I((dur+1)^(0.9))$ - значимые; менее значимые: $I(age*dur)$

$salary + 3)^2 \sim (age + 3^{(1.9)}) + sex + higher_educ + status2 + ((dur+1)^(0.1)) + (age*dur) + wed1 + wed2 + wed3$, adj $R^2 = 0.025$, параметры $(log(age+3)^{(1.9)}) + higher_educ + wed1 + wed2 + ((dur+1)^(0.1))$ - значимые; менее значимые: $I(age*dur)$

$(salary + 3)^{0.1} \sim ((age + 3)^{0.1}) + sex + higher_educ + status2 + ((dur+1)^(0.1)) + (age*dur) + wed1 + wed2 + wed3$, adj $R^2 = 0.02603 = 2,603\%$, параметры $I(log(age+3)^{(0.1)}) + higher_educ + I((dur+1)^(0.1)) + wed1 + wed2$ - значимые; менее значимые: $I(age*dur)$

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

Лучшая модель - $(salary + 3)^{0.1} \sim ((age + 3)^{0.1}) + sex + higher_educ + status2 + ((dur+1)^(0.1)) + (age*dur) + wed1 + wed2 + wed3$, adj $R^2 = 0.02603 = 2,603\%$, параметры $I(log(age+3)^{(0.1)}) + higher_educ + I((dur+1)^(0.1)) + wed1 + wed2$ значимые; менее значимые: $I(age*dur)$

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

Из последней (лучшей) модели можно сделать вывод что лучше всех зарабатывают более взрослые респонденты с высшим образованием, более долгой рабочей неделей, (женатые или разведенные).

5. Оцените лучшие модели для подмножества индивидов, с высшим образованием, не из города; женщины, с высшим образованием. Сделайте вывод о том, какие индивиды получают наибольшую зарплату.

Подмножество : С высшим образованием, не из города

$(salary + 3)^{0.1} \sim I((age + 3)^{0.1}) + sex + I + status2 + I((dur+1)^(0.1)) + I(age*dur) + wed1 + wed2 + wed3$

Большой VIF у $wed1$, потому здесь его можно убрать и на результат он не повлияет.

adj $R^2 = 0.06632 = 6,632\%$, параметры $I((age + 3)^{0.1}) + I((dur+1)^(0.1)) + I(age*dur)$ значимые

Вывод : В подмножестве(С высшим образованием, не из города) хорошо зарабатывают более взрослые респонденты с более длинной рабочей неделей.

Подмножество : женщины с высшим образованием

$$(salary+3)^{0.1} \sim I((age+3)^{0.1}) + 0 + I + status2 + I((dur+1)^{0.1}) + I(age*dur) + wed1 + wed2 + wed3)$$

Большой VIF у wed1 , потому здесь его можно убрать и на результат он не повлияет

$adj\ R^2 = 0.07919 = 7,919\%$, параметры $I((age+3)^{0.1}) + I((dur+1)^{0.1}) + I(age*dur)$ значимые

Вывод: В подмножестве (женщины с высшим образованием) хорошо зарабатывают более взрослые респонденты с более длинной рабочей неделей.

На основании лучшей модели и анализа всей выборки можно сделать вывод, что уровень заработной платы наиболее высок у респондентов с более старшим возрастом, имеющих высшее образование, работающих больше часов в неделю и состоящих в браке или состоявших в разводе.

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

Отчет по задаче №4

Задача 4

1. Обработайте набор данных [набор данных, Video game sales](https://www.kaggle.com/gregorut/videogamesales) <https://www.kaggle.com/gregorut/videogamesales>, подготовив его к решению задачи классификации.

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
1	Wii Sports	26	2006.0	0	359	41.49	29.02	3.77	8.46	82.74
2	Super Mario Bros.	11	1985.0	1	359	29.08	3.58	6.81	0.77	40.24
3	Mario Kart Wii	26	2008.0	1	359	15.85	12.88	3.79	3.31	35.82
4	Wii Sports Resort	26	2009.0	0	359	15.75	11.01	3.28	2.96	33.00
5	Pokemon Red/Pokemon Blue	5	1996.0	1	359	11.27	8.89	10.22	1.00	31.37
...
16596	Woody Woodpecker in Crazy Castle 5	6	2002.0	1	269	0.01	0.00	0.00	0.00	0.01
16597	Men in Black II: Alien Escape	7	2003.0	1	241	0.01	0.00	0.00	0.00	0.01
16598	SCORE International Baja 1000: The Official Game	16	2008.0	1	21	0.00	0.00	0.00	0.00	0.01
16599	Know How 2	4	2010.0	1	8	0.00	0.01	0.00	0.00	0.01
16600	Spirits & Spells	6	2003.0	1	544	0.01	0.00	0.00	0.00	0.01

Рисунок 4.1. Необработанный набор данных videogamesales.

Выделите целевой признак, указанный в последнем столбце таблицы, и удалите его из данных, на основе которых будет обучаться классификатор:

Целевой признак - `data['Genre']`. Удаляем его из рассмотрения

Разделите набор данных на тестовую и обучающую выборку:

```
Y = data.loc[:, data.columns.isin(['Genre'])]
X = data.loc[:, data.columns.isin(['Year', 'Publisher', 'NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales'])]
X_train = X.sample(frac= 0.8 ,random_state= 0)
X_test= X.drop(X_train. index )
y_train = Y.sample(frac= 0.8 ,random_state= 0)
y_test= Y.drop(y_train. index )
importances = tree1.feature_importances_
```

```
array([0.20350082, 0.3018054 , 0.19686459, 0.13312733, 0.09516995,
       0.0695319 ])
```

Вставка с кодом №1.

Важность признаков (делаем вывод что самым важным является : Publisher)

Постройте классификатор типа, DecisionTreeClassifier (решающее дерево), для задачи классификации по параметру, указанному в последнем столбце. Оцените точность построенного классификатора с помощью метрик precision, recall и F1 на тестовой выборке.

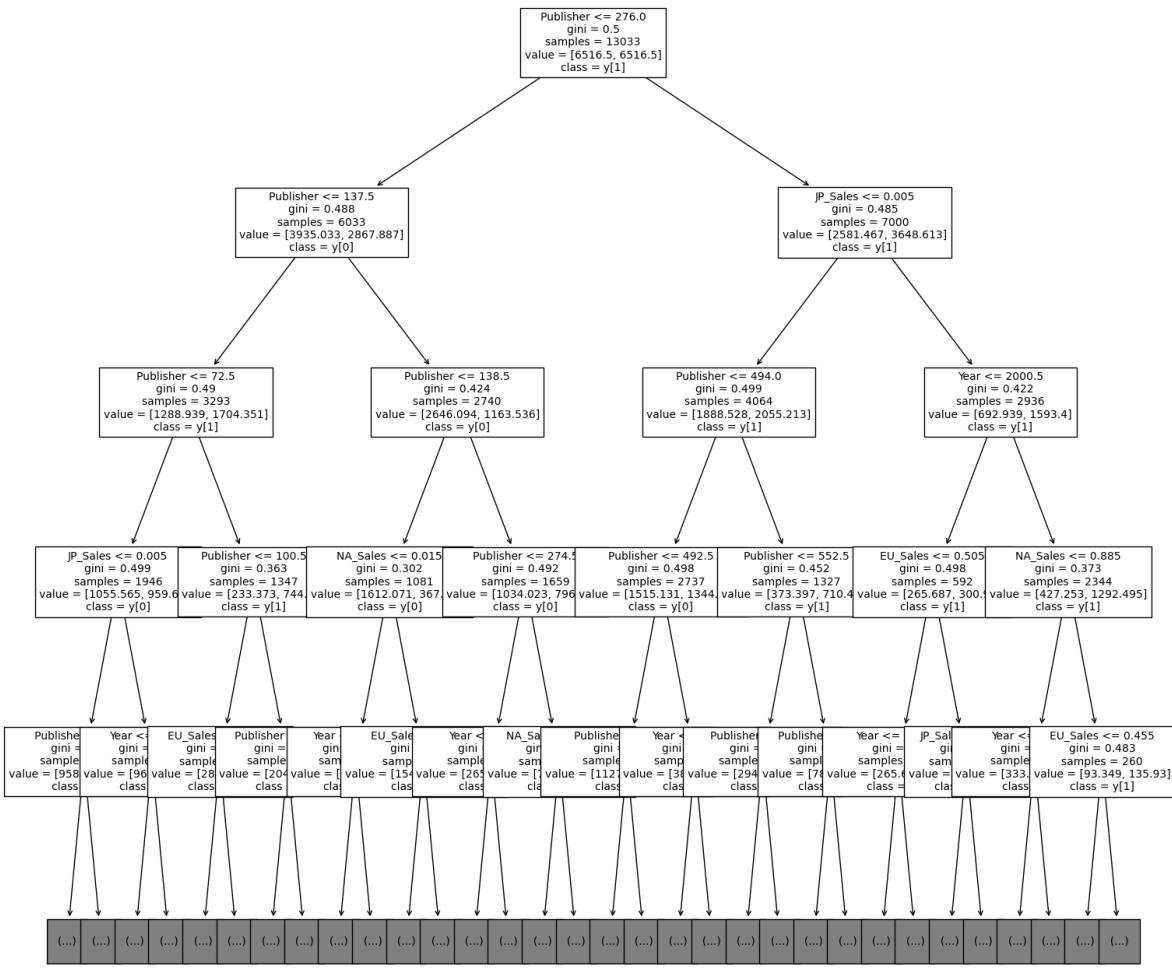


Рисунок 4.2. *DecisionTreeClassifier* построенный на данных videogamesales.

Опишем характеристики некоторых объектов класса 1 : Publisher ≥ 276 , JP_Sales ≥ 0.005 , Year ≥ 2000.5

Опишем характеристики некоторых объектов класса 0: 137 \leq Publisher ≤ 276 , JP_Sales ≤ 0.005 , EU_Sales ≤ 0.505 .

DecisionTreeClassifier:

Accuracy : 0.8136893799877225

Precision : 0.6381022210221527

Recall : 0.642025980010945

F1-score : 0.6400029489960215

Так как данные не сбалансираны , делаем вывод что процент правильности решающего дерева по метрике F1 порядка 64%.

2. Постройте классификатор типа Случайный Лес (Random Forest) для решения той же задачи классификации. Оцените его качество с помощью метрик precision, recall и F1 на тестовой выборке.

Random Forest:

Accuracy : 0.875023247163846

Precision : 0.7641319847186843

Recall : 0.6210986571366688

F1-score : 0.6535865952980463

Точность предсказания выросла и стала порядка 65%. Наши данные не сбалансираны ,поэтому у алгоритма Random Forest, построенного на несбалансируемых данных, точность может быть ниже 70% по некоторым причинам:

1. Предсказания деревьев: Random Forest состоит из множества деревьев, каждое из которых строится на подвыборках данных. Если данные несбалансираны, то некоторые деревья могут быть сильно смещены к предсказанию доминирующего класса, игнорируя редкий класс. Это может привести к снижению точности для редкого класса.
2. Случайные подвыборки: Random Forest случайным образом выбирает подмножество данных для построения каждого дерева. Если редкий класс имеет меньшее количество примеров, то существует вероятность, что случайная подвыборка не будет содержать достаточно количество примеров этого класса. Это может привести к недостаточной информации для корректного предсказания редкого класса.
3. Взвешивание классов: Random Forest позволяет задать веса классов, чтобы учесть несбалансированность данных. Однако, если веса классов не были правильно настроены или не соответствуют действительности, то это может повлиять на точность предсказания редкого класса.
4. Ошибки деревьев: Деревья в Random Forest могут допускать ошибки классификации, особенно для редкого класса. Если большинство деревьев в ансамбле делает неправильное предсказание для редкого класса, то это может снизить точность алгоритма в целом.

Чтобы улучшить точность Random Forest на несбалансированных данных, можно применить следующие подходы:

1. Балансировка классов: Применить стратегии балансировки классов, такие как upsampling или downsampling, чтобы сделать распределение классов более равномерным. Это поможет увеличить представительность редкого класса и улучшить точность его предсказания.
2. Настройка весов классов: Использовать веса классов для учета несбалансированности данных. Корректное настройка весов классов может помочь модели более справедливо учесть редкий класс и повысить точность его предсказания.
3. Использование подходов обработки несбалансированных данных: Применить специальные методы, такие как SMOTE (Synthetic Minority Over-sampling

С помощью GridSearch переберите различные комбинации гиперпараметров: на первой итерации задайте большие шаги (50 или 100) по числу деревьев n_estimators. На следующих итерациях определите лучшее количество деревьев n_estimators с точностью до 10.

Лучшие гиперпараметры : {'max_depth': 12, 'n_estimators': 110}
F1-score : 0.596405386446071 ~ 60%

Из проделанной практики делаем вывод что Случайный Лес + GridSearch дают худшую точность = 60% > 64% = точность Решающего Дерева по метрике F1 , так как данные не сбалансированы. Лучшую точность дает Random Forest без перебора гиперпараметров (точность порядка 65%)

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

Отчет по задаче №5

Задача 5

Задание 5. Предобработка данных и РСА Необходимо провести анализ датасета _____ (<https://www.kaggle.com/datasets/tejashvi14/travel-insurance-prediction-data>) и сделать обработку данных по предложенному алгоритму. Код подготовить в виде файлов *.ipynb и сделать отчет в виде ноутбука с описанием процесса анализа (*.pdf).

1. Всего в наборе 1987 объектов и 9 признака.

Age- Возраст учащегося.

Employment Type - Тип занятости.

GraduateOrNot - Выпускник Или Нет.

AnnualIncome - Годовой доход.

FamilyMembers - Члены семьи.

ChronicDiseases - Хронические Заболевания.

FrequentFlyer - Часто летающий пассажир.

EverTravelledAbroad - Когда-либо путешествовал за границу.

TravelInsurance - Страхование путешествий.

2. Сколько категориальных признаков, какие?

В данном наборе данных 4 категориальных признака. К категориальным признакам относятся:

Employment Type - Тип занятости.

GraduateOrNot - Выпускник Или Нет.

FrequentFlyer - Часто летающий пассажир.

EverTravelledAbroad - Когда-либо путешествовал за границу.

3. Столбец с максимальным количеством уникальных значений категориального признака?

Все эти 4 столбца имеют максимальное количество уникальных значений

4. Есть ли бинарные признаки?

Да, к таким признакам относятся:

Employment Type- Тип занятости.

GraduateOrNot - Выпускник Или Нет.

FrequentFlyer - Часто летающий пассажир.

EverTravelledAbroad - Когда-либо путешествовал за границу.

ChronicDiseases - Хронические Заболевания.

TravelInsurance - Страхование путешествий.

5. Какие числовые признаки?

К числовым признакам относятся:

Age - Возраст учащегося.

AnnualIncome - Годовой доход.

FamilyMembers - Члены семьи.

6. Есть ли пропуски?

Пропусков нет.

7. Сколько объектов с пропусками?

0

8. Столбец с максимальным количеством пропусков?

Такого столбца нет, так как пропусков в наборе данных нет.

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

Здесь нет сильных различий между максимальными значениями и средними, потому можно сделать вывод что выбросов/аномалий в данных нет.

10. Столбец с максимальным средним значением после нормировки признаков через стандартное отклонение?

В ходе нормализации будем нормализовывать числовые признаки, но которые принимают реальные вещественные значения, а не являются описанием какого-либо признака.

К таким признакам относятся:

Age

AnnualIncome

11. Столбец с целевым признаком?

В наборе данных можно выделить один целевой признак: TravelInsurance - Страхование путешествий.

12. Сколько объектов попадает в тренировочную выборку при использовании train test split с параметрами test_size = 0.3, random_state = 42?

При заданных параметрах в тренировочную выборку попадает 597 объектов.

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

Из таблицы можно сделать вывод, что в этом датасете нет линейно-зависимых параметров.

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

Для объяснения 90% дисперсии после применения метода PCA достаточно 10 компонент.

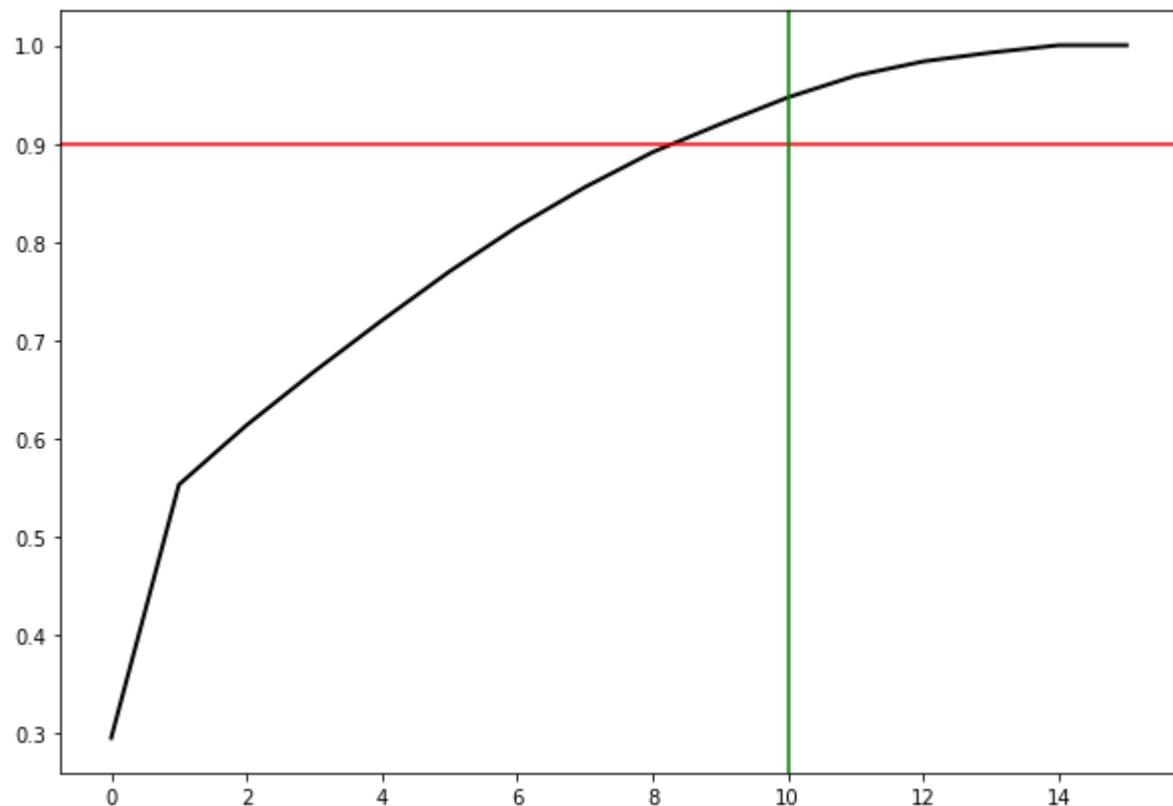


Рисунок 5.1. График важности компонент PCA и увеличения процента объясненной дисперсии.

15. Какой признак вносит наибольший вклад в первую компоненту?

Признак `$AnnualIncome_std$` вносит наибольший вклад в первую компоненту.

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

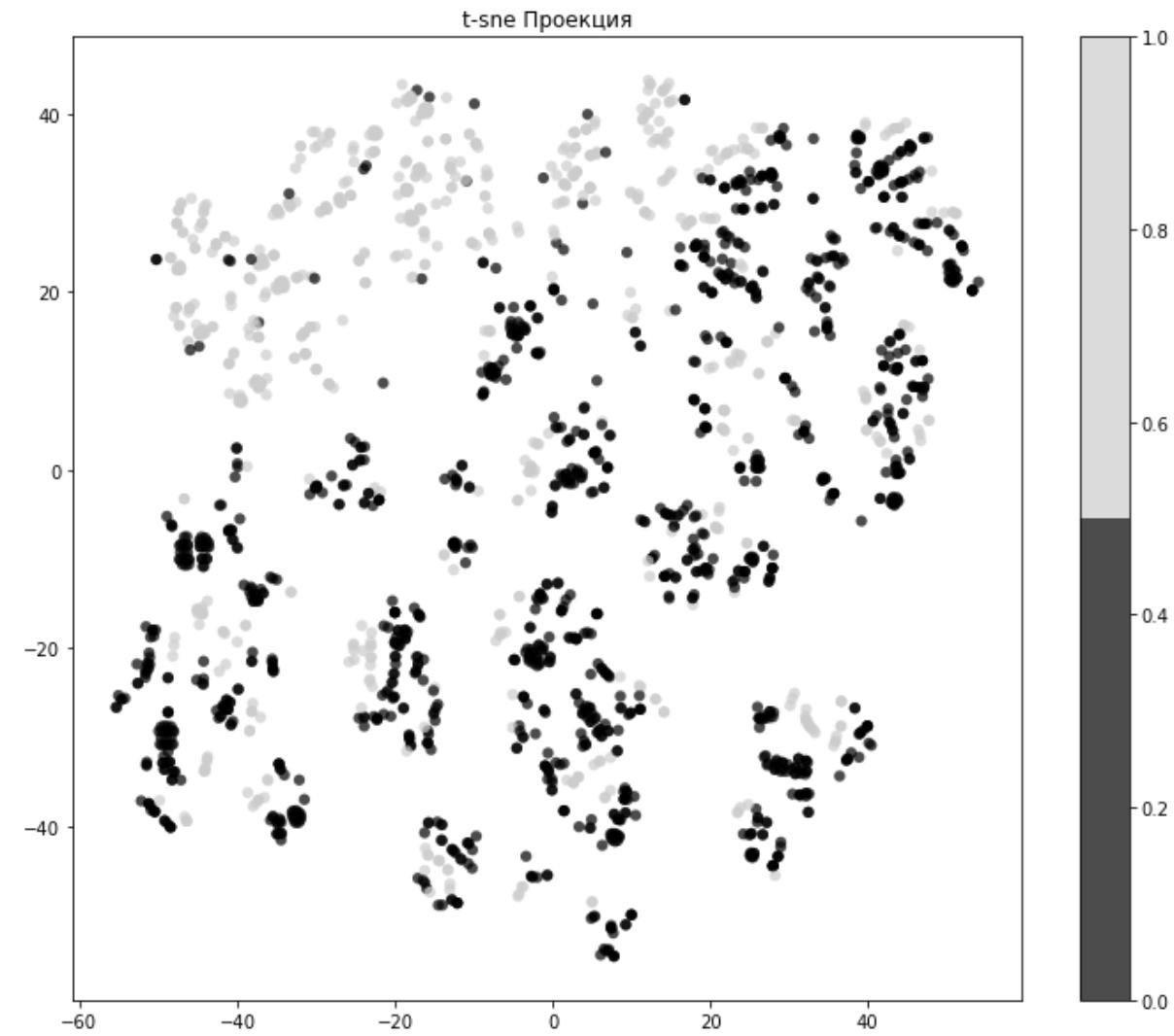


Рисунок 5.2. График визуального представления t-SNE.

Выборка разделяется на 2 класса, верхний кластер показывает что объекты покупают страховку, а объекты нижнего кластера - не покупают.

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

Отчет по задаче №6

Задача 6

Для набора данных (<https://www.kaggle.com/datasets/tejashvi14/travel-insurance-prediction-data>) построить модель предсказания: будет ли клиент брать страховку, включающую покрытие рисков, связанных с коронавирусом? Оценить вклад каждой компоненты. Есть ли выбросы среди данных, какие? Применить алгоритмы кластеризации. Охарактеризовать кластеры.

Для описания 90% дисперсии достаточно 10 компонент РСА.

AgglomerativeClustering (Иерархической кластеризация)

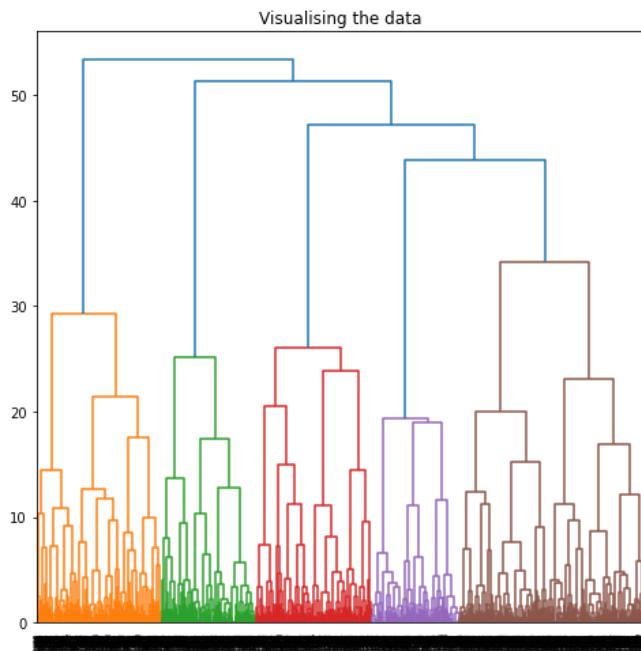


Рисунок 6.1. Дендрограмма иерархического алгоритма кластеризации.

Дендрограмма иерархического алгоритма кластеризации дает нам следующее:

иерархическую структуру кластеров, информацию о расстояниях между кластерами помочь в определении оптимального числа кластеров.

По этому графику можно сделать выводы, что данные оптимально разбиваются на 5 кластеров на высоте 40.

Визуальное представление кластеров:

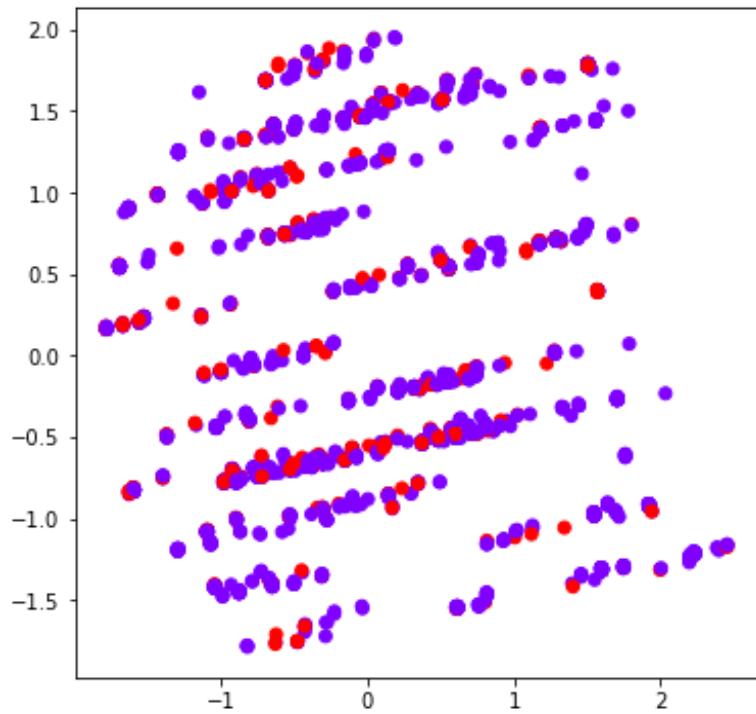


Рисунок 6.2. График разбиения данных на 2 кластера.

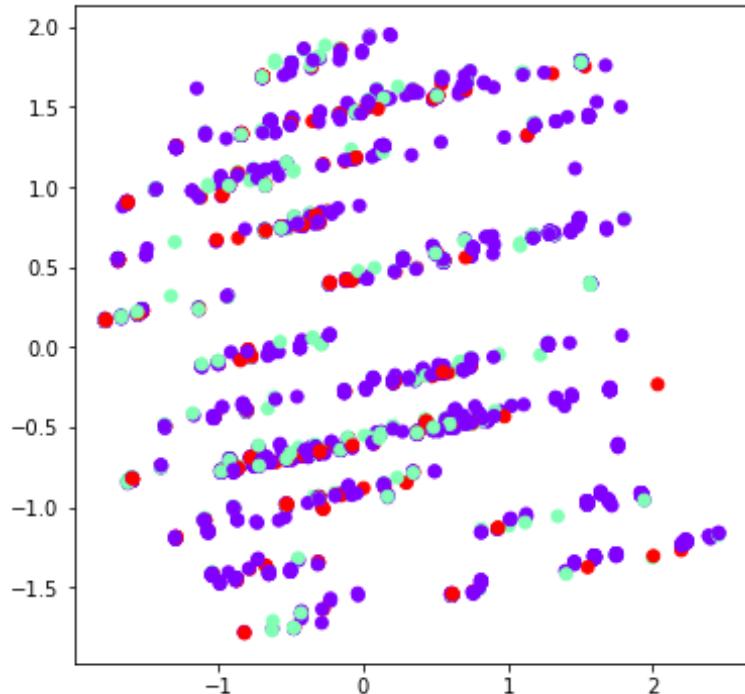


Рисунок 6.3. График разбиения данных на 3 кластера.

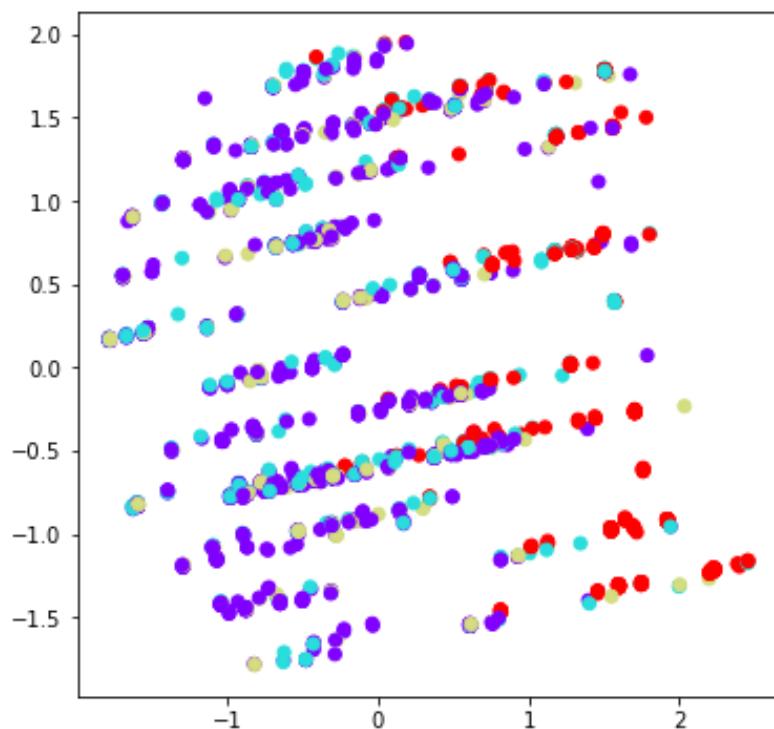


Рисунок 6.4. График разбиения данных на 4 кластера.

Можно сделать вывод , что чем больше кластеров давать алгоритму, тем менее разделимы они получаются на наших данных.

DBSCAN

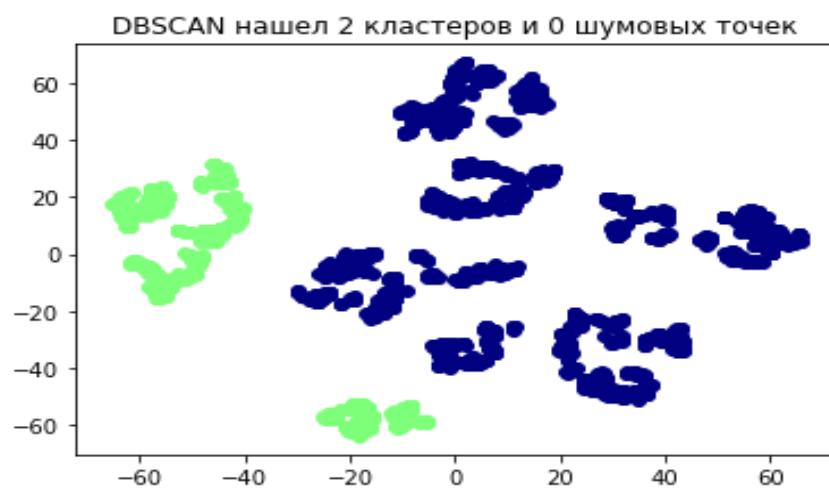


Рисунок 6.5. График результатов работы алгоритма DBSCAN.

Индивиды разделились на 2 класса, зеленая группа покупает страховки, а синяя группа - нет.

K-means

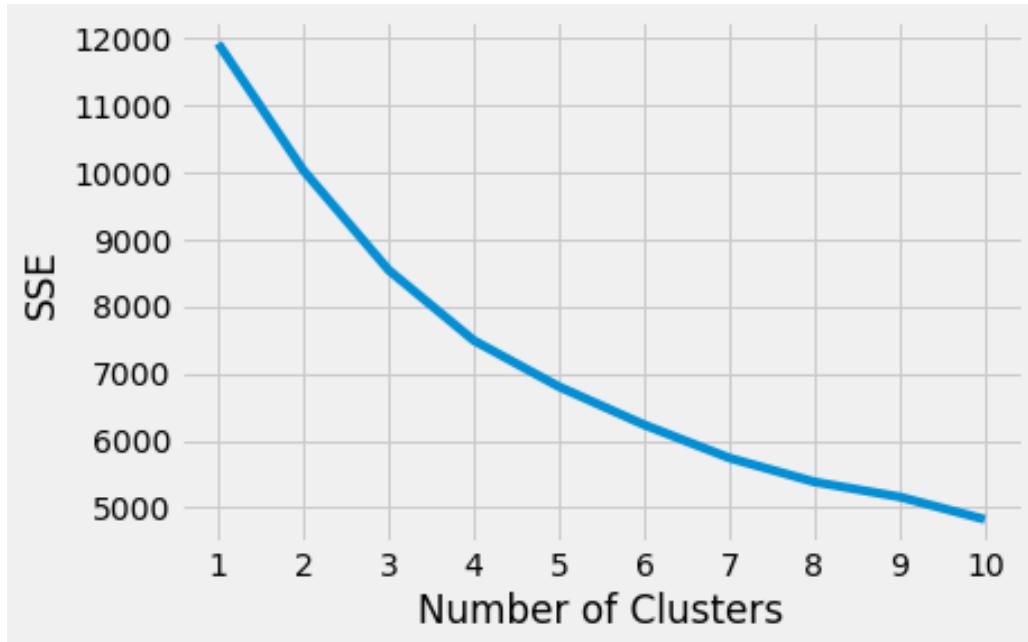


Рисунок 6.6. График зависимости суммы квадратов ошибок от количества кластеров.

Из рисунка 6 видно, что оптимальное количество кластеров - 4.

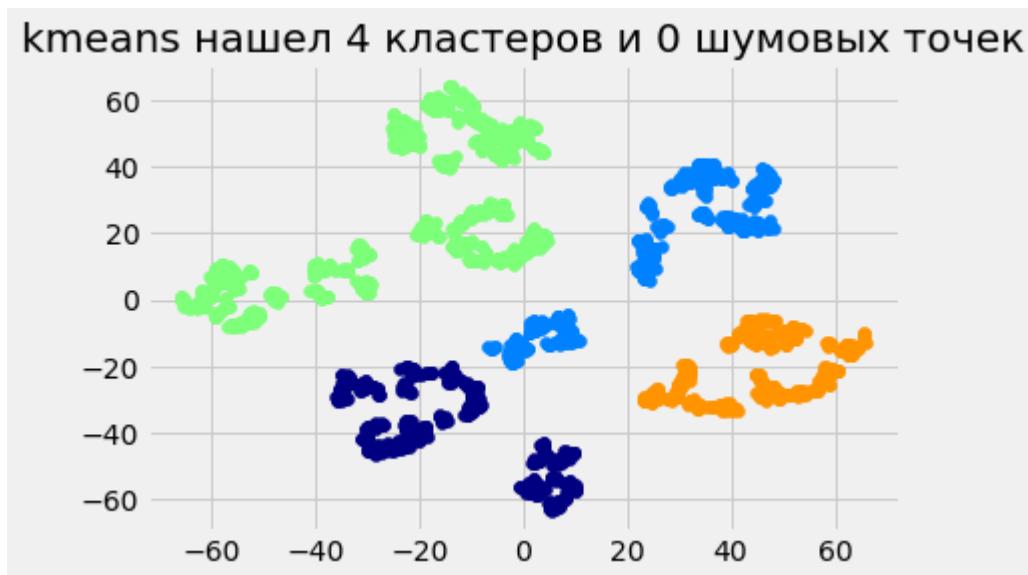


Рисунок 6.7. График результатов работы алгоритма k-means.

Вычисленные центры классов:

[[-0.02338718 0.01653908 0.12747969 -0.38446642 1.40057273 0.80884714]

[0.03223941 -0.00598203 -0.97650456 1.29426444 0.19268617 -0.70149935]

[-0.00717411 0.00362513 -0.33309664 -0.52959815 -0.73477452 0.18495898]

[0.00469648 -0.01965695 1.65835384 0.04514494 -0.32784704 -0.50751104]]

Объясним смысл каждого из 4рех кластеров:

Таблица 6.1. Таблица объясняющая усредненные характеристики каждого класса

Age	Employment	Type	GraduateOr Not	AnnualIncome	ChronicDiseases
0	0.007864	0.629113	0.912942	-0.001067	0.972436
1	-0.011303	0.685189	0.889088	0.055563	0.271174
2	0.006592	0.756882	0.796469	-0.043795	0.030614
3	-0.009633	0.752306	0.850673	0.027383	-0.008222
FrequentFlyer	EverTravelledAbroad	TravelInsurance	FamMem_2	FamMem_3	
0.071783	0.210861	0.403489	0.052937	0.236327	
0.309669	0.158524	0.233949	0.050433	-0.110043	
0.239356	0.219074	0.437767	0.047907	0.458790	
0.193985	0.149017	0.280502	0.033298	-0.072420	
FamMem_4	FamMem_5	FamMem_6	FamMem_7	FamMem_8	FamMem_9
0.316191	0.017050	0.197313	0.117420	0.021395	0.041367
-0.077586	0.891660	0.115490	0.078440	0.027740	0.023867
0.064806	0.078950	0.185096	0.099687	0.037007	0.027757

0.952398	-0.057424	0.052043	0.049508	0.026487	0.016110
----------	-----------	----------	----------	----------	----------

1. Люди среднего возраста, которые работают в частном секторе, учились в колледже, имеют чуть ниже среднего заработок, количество детей - среднее. У них есть хроническое заболевание. Покупают авиаилеты и путешествуют за рубеж редко, кто-то покупал страховой пакет, а кто-то нет
2. Люди среднего возраста, которые работают в частном секторе, учились в колледже, имеют средний заработок , количество детей - среднее. У них нет болезней. Покупают авиаилеты и путешествуют за рубеж редко, не покупают страховой пакет.
3. Люди немного старше среднего, которые работают в государственном секторе, учились в колледже, имеют заработок ниже среднего, количество детей - среднее. У них нет болезней. Покупают авиаилеты и путешествуют за рубеж редко, большинство не купило страховой пакет.
4. Люди среднего возраста, которые работают в частном секторе, учились в колледже, имеют заработок выше среднего, количество детей - среднее. У них нет болезней. Покупают авиаилеты и путешествуют за рубеж чаще обычного (больше половины да чем нет), почти все купили страховой пакет.

Классификация на основе кластеров лучшего алгоритма (K-means)

Строим дерево решений на наших кластерах:

Перебираем высоту дерева:

2: 0.7674418604651163

3: 0.7751937984496124

4: 0.7906976744186046

5: 0.7984496124031008

6: 0.7984496124031008

7: 0.8062015503875969

8: 0.7829457364341085

9: 0.7441860465116279

Лучший результат дает дерево с глубиной 4.

Строим дерево решений глубиной 4. Получили модель с правильностью порядка 76% по метрике F1.

Важность признаков:

[0.1382899	0.02543763	0.00905349	0.54925611	0.	0.01466946
0.06357554	0.	0.	0.	0.08450275	
0.11521512	0.	0.]		

Самым важным признаком оказался - AnnualIncome.

Опишем характеристики некоторых объектов класса 1. AnnualIncome ≥ 1.041 , GraduateOrNot ≥ 0.5 , Age в окрестности -0.052, EverTravelledAdroab ≥ 0.5 .

Опишем характеристики некоторых объектов класса 0. AnnualIncome ≤ 1.041 , Age ≤ 0.978 or Age ≥ 0.978 , EverTravelledAdroab ≤ 0.5 , FamMem_6 ≤ 0.5 .

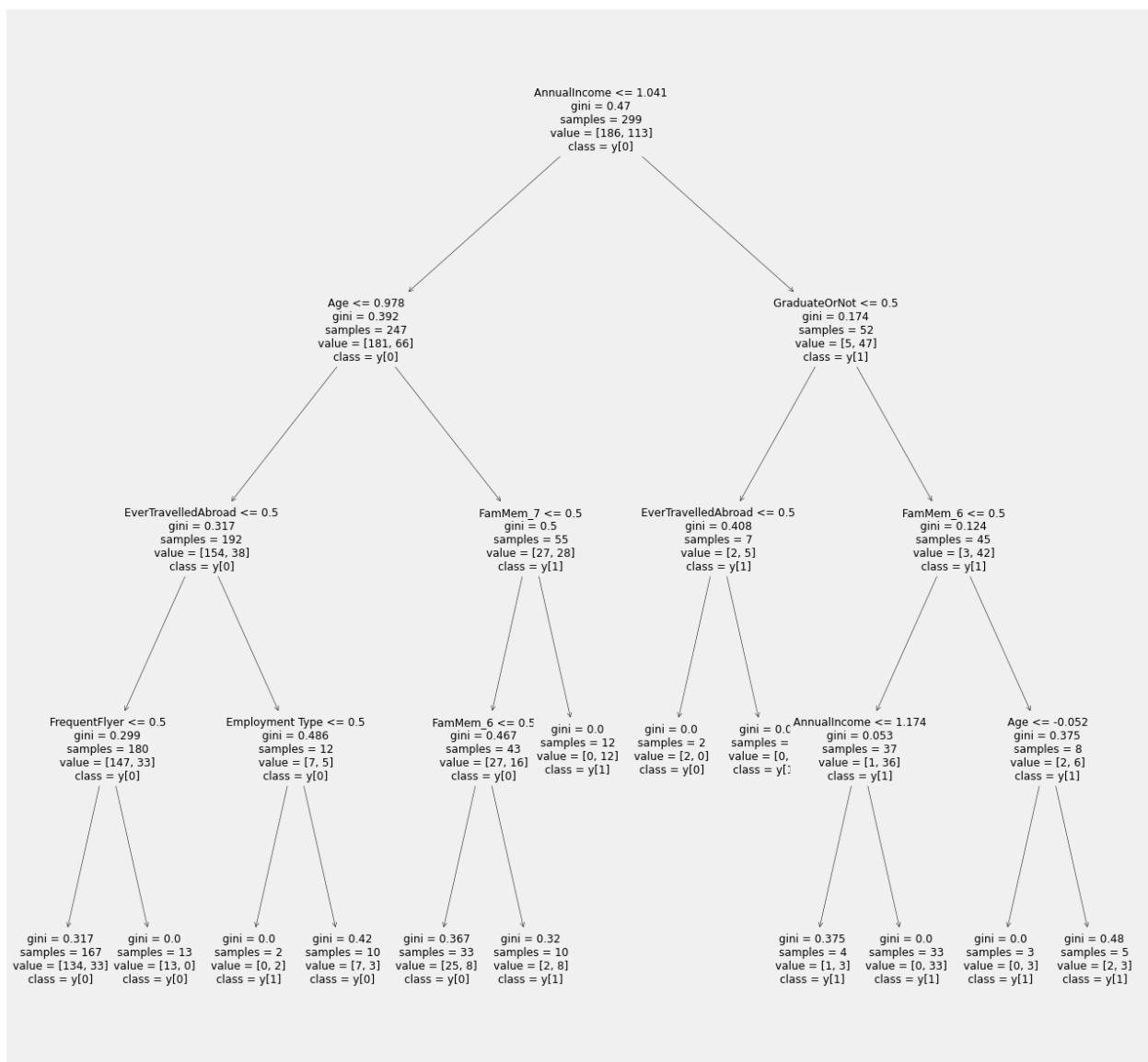


Рисунок 6.8. *DecisionTreeClassifier* построенный на кластерах алгоритма k-means.

Попробуем RandomForestClassifier + GridSearch

Лучшие гиперпараметры: {'max_depth': 9, 'n_estimators': 250}

Best score F1 is: 0.7005622871624296

Лучшие гиперпараметры: {'max_depth': 9, 'n_estimators': 230}

Таблица 6.2. Оценочные метрики RandomForestClassifier + GridSearch.

		precision	recall	f1-score	support
	0	0.77	0.94	0.84	78
	1	0.85	0.57	0.68	51
accuracy				0.79	129
macro avg		0.81	0.75	0.76	129
weighted avg		0.80	0.79	0.78	129

F1-score : 0.7631417885073104

Лучший F1 = 76%. Это также меньше у дерева решений.

Попробуем GradientBoostingClassifier:

Таблица 6.3. Оценочные метрики GradientBoostingClassifier.

		<u>precision</u>	<u>recall</u>	<u>f1-score</u>	<u>support</u>
	<u>0</u>	<u>0.76</u>	<u>0.92</u>	<u>0.83</u>	<u>78</u>
	<u>1</u>	<u>0.82</u>	<u>0.5</u>	<u>0.66</u>	<u>51</u>
<u>accuracy</u>				<u>0.78</u>	<u>129</u>
<u>macro avg</u>		<u>0.79</u>	<u>0.74</u>	<u>0.75</u>	<u>129</u>
<u>weighted avg</u>		<u>0.78</u>	<u>0.78</u>	<u>0.76</u>	<u>129</u>

F1-score : 0.7455967358041483

Получили F1 = 74%, все еще меньше чем у дерева решений (F1=76%), оставляем дерево решений в качестве лучшей модели.

Таким образом, можно сделать вывод, что страховку с большей вероятностью купят люди среднего возраста, которые работают в частном секторе, учились в колледже, имеют заработок выше среднего, количество детей - среднее. У них нет болезней. Они покупают авиабилеты и путешествуют за рубеж чаще обычного (больше половины да чем нет).

Заключение

1. Практика №1:

Коэффициент детерминации (Coefficient of Determination, R²): R² измеряет пропорцию дисперсии зависимой переменной, объясненную моделью. Значение R² находится в диапазоне от 0 до 1, где 1 означает, что модель полностью объясняет вариацию данных, а 0 - что модель не объясняет никакую вариацию. Чем ближе R² к 1, тем лучше модель.

Зависимость Examination~Fertility. R² = 40%. Значение коэффициента детерминации меньше 80%, поэтому заключаем, что линейной зависимости нет. p-value = 9.45*(10**-7) , *** у Fertility означают сильную взаимосвязь Fertility и Examination. Fertility и Examination связаны отрицательно (-0.41)

Зависимость Examination~Education. R² = 47%. Значение коэффициента детерминации меньше 80%, поэтому заключаем, что линейной зависимости нет. p-value = 4.811*(10**-8) , *** у Education означают сильную взаимосвязь Education и Examination. Education и Examination связаны положительно (0.57)

Также из проделанной практики можно сделать вывод о том, что Examination не зависит линейно от Education и Fertility.

2. Практика №2:

Целевая переменная mpg описывается через log(disp) , log(hp) на 82,9%

После выполнения первой части практической работы №2 можно сделать вывод что для набора данных mtcars и четырех переменных mpg, disp, hp, drat применение логарифмической функции к регрессорам может привести к линеаризации отношения между переменными и улучшению качества модели, устраниить гетероскедастичность или облегчить интерпретацию коэффициентов.

После выполнения второй части практической работы №2 можно понять что доверительные интервалы модели предоставляют информацию о статистической значимости, уровне неопределенности и помогают сделать более информированные выводы о модели и ее прогностической силе.

3. Практическая работа №3:

Лучшая модель - (salary+3)^{0.1}~((age+3)^{0.1}) + sex + higher_educ + status2 +((dur+1)^{0.1}) +(age*dur) + wed1+wed2+wed3) , adj R² = 0.02603=2,603% , параметры I(log(age+3)^{0.1})+higher_educ +I((dur+1)^{0.1})+wed1+wed2 значимые; менее значимые: I(age*dur).

Из последней (лучшей) модели можно сделать вывод что лучше всех зарабатывают более взрослые респонденты с высшим образованием, более долгой рабочей неделей, (женатые или разведенные).

В подмножестве (С высшим образованием, не из города) хорошо зарабатывают более взрослые респонденты с более длинной рабочей неделей.

В подмножестве (женщины с высшим образованием) хорошо зарабатывают более взрослые респонденты с более длинной рабочей неделей.

4. Практическая работа №4:

Из проделанной практики делаем вывод что по метрике F1: Случайный Лес + GridSearch дают худшую точность = $60\% > 64\%$ = точность Решающего Дерева по метрике F1 , так как данные не сбалансированы. Лучшую точность дает Random Forest без перебора гиперпараметров (точность порядка 65%)

Улучшить алгоритмы классификации можно сбалансировав данные (убрать данные большего класса или сгенерировать приближенные данные другого класса.

5. Практическая работа №5:

1. В наборе данных содержится информация о 1987 объектах и 9 признаках.
2. Из них 4 признака являются категориальными: Employment Type, GraduateOrNot, FrequentFlyer и EverTravelledAbroad.
3. Все категориальные признаки имеют максимальное количество уникальных значений.
4. Все признаки, включая категориальные, могут быть рассмотрены как бинарные, так как они принимают только два возможных значения.
5. Числовыми признаками являются: Age, AnnualIncome и FamilyMembers.
6. В наборе данных отсутствуют пропуски.
7. Количество объектов с пропусками составляет 0.
8. В данных нет выбросов или аномальных значений, так как максимальные значения не сильно отличаются от средних значений.
9. После нормировки признаков через стандартное отклонение, столбец с максимальным средним значением будет зависеть от конкретных значений в данных и не может быть определен без их рассмотрения.
10. Целевым признаком является TravelInsurance - страхование путешествий.
11. При использовании train_test_split с параметрами test_size = 0.3 и random_state = 42, в тренировочную выборку попадает 597 объектов.

12. В данном наборе данных нет линейной зависимости (корреляции) между признаками.
 13. Для объяснения 90% дисперсии после применения метода РСА достаточно 10 компонент.
 14. Признак AnnualIncome_std вносит наибольший вклад в первую компоненту при применении РСА.
 15. По визуальному представлению данных с помощью алгоритма t-SNE выборка разделяется на 2 кластера, где верхний кластер соответствует объектам, которые покупают страховку, а нижний кластер - объектам, которые не покупают страховку.
6. Практическая работа №6:

Таким образом, можно сделать вывод, что люди среднего возраста, работающие в частном секторе, закончившие колледж, выше среднего уровня дохода, среднее количество детей и без каких-либо заболеваний, с большей вероятностью приобретут страховку. Кроме того, эта группа людей чаще, чем обычно (более половины), покупает авиабилеты и осуществляет поездки за рубеж.

Дерево решений, основанное на кластеризации с использованием алгоритма k-means, проявляет точность около 76% по метрике F1. Это означает, что модель достигает высокого уровня правильных предсказаний, учитывая общую точность и полноту модели. Метрика F1 является сбалансированным показателем, учитывающим как точность, так и полноту, что делает ее полезной для оценки производительности классификационных моделей. Общая точность дерева решений на основе k-means-кластеризации находится на уровне 76%, что указывает на способность модели классифицировать данные с неплохой точностью и показывать удовлетворительные результаты.

Список литературы

- 1.** Introduction to Econometrics with R/C. Hanck, M. Arnold, A. Gerber, M. Schmelzer.
- Essen, Germany: University of Duisburg-Essen, 2021.
- 2.** Айвазян, С.А. Основы эконометрики/С.А. Айвазян, В.С. Мхитарян – Москва:
Изд. объединение «ЮНИТИ», 1998. – 1005 с.
- 3.** Вербик, М. Путеводитель по современной эконометрике/М. Вербик – Москва:
«Научная книга», 2008. – 616 с.
- 4.** Доугерти, К. Введение в эконометрику/К. Доугерти – Москва: ИНФРА-М, 2009.
– 465 с.
- 5.** Магнус, Я.Р. Эконометрика. Начальный курс/Я.Р. Магнус, П.К. Катышев, А.А.
Пересецкий – Москва: Изд-во «ДЕЛО», 2004. – 576 с.

Приложения

Приложение 1.

```
library("lmtest")
data = swiss
#1)
#среднее арифметическое,Дисперсия и СКО Examination
mean(data$Examination)
var(data$Examination)
sd(data$Examination)
#среднее арифметическое,Дисперсия и СКО Fertility
mean(data$Fertility)
var(data$Fertility)
sd(data$Fertility)
#среднее арифметическое,Дисперсия и СКО Education
mean(data$Education)
var(data$Education)
sd(data$Education)
#2,3,4)
#Регрессия y =ax^b для Examination~Fertility
model1=lm(Examination~Fertility,data)
model1
summary(model1)
# Оценка по R^2 : модель среднего качества, выдает правильные ответы в 40%
случаев
# p-value = 9.45*(10**-7) , *** y Fertility означают сильную взаимосвязь Fertility и
Examination
```

```

#Fertility и Examination связаны отрицательно (-0.41)

#Регрессия y =ax*b для Examination~Education

model2=lm(Examination~Education,data)

model2

summary(model2)

# Оценка по R^2 : модель среднего качества, выдает правильные ответы в 47% случаев

# p-value = 4.811*(10**-8) , *** у Education означают сильную взаимосвязь Education и Examination

# Education и Examination связаны положительно (0.57)

```

```

--> #среднее арифметическое,дисперсия и СКО Examination
> mean(data$Examination)
[1] 16.48936
> var(data$Examination)
[1] 63.64662
> sd(data$Examination)
[1] 7.977883
>
>
>
> #среднее арифметическое,дисперсия и СКО Fertility
> mean(data$Fertility)
[1] 70.14255
> var(data$Fertility)
[1] 156.0425
> sd(data$Fertility)
[1] 12.4917
>
>
>
> #среднее арифметическое,дисперсия и СКО Education
> mean(data$Education)
[1] 10.97872
> var(data$Education)
[1] 92.45606
> sd(data$Education)
[1] 9.615407

```

Рисунок 1. Результат работы встроенных функций подсчета среднего арифметического, дисперсии и СКО для *Examination, Education, Fertility*.

Приложение 2.

```
library(car)  
library("lmtest")  
  
# Вариант 7  
  
data = na.omit(mtcars)  
  
help(mtcars)
```

Первая часть практической
"Прочитав информацию из набора данных, выполните задачи:

1. Проверьте, что в наборе данных нет линейной зависимости (построить зависимости между переменными, указанными в варианте, и проверить, что R^2 в каждой из них невысокий). В случае, если R^2 большой, один из таких столбцов можно исключить из рассмотрения.
2. Постройте линейную модель зависимой переменной от указанных в варианте регрессоров по методу наименьших квадратов (команда lm пакета lmtest в языке R). Оценить, насколько хороша модель, согласно: 1) R^2 , 2) p-значениям каждого коэффициента.
3. Введите в модель логарифмы регрессоров (если возможно). Сравнить модели и выбрать наилучшую.
4. Введите в модель всевозможные произведения пар регрессоров, в том числе квадраты регрессоров. Найдите одну или несколько наилучших моделей по доле объясненного разброса в данных R^2 .

```
#1 Исследуем линейную зависимость регрессоров

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

vif(model1)#vif=7.7 самый большой у cyl

summary(lm(disp~cyl+hp+drat,data))#тоже зависим и vif=6.2 > 5

summary(lm(cyl~disp+hp+drat,data))#убеждаемся что больше всех линейно зависим
cyl

summary(lm(hp~cyl+disp+drat,data))

summary(lm(drat~disp+hp+cyl,data))
```

#удаляем cyl из исследования

#2 Строим парную регрессию с менее зависимыми между собой регрессорами

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

summary(model2) #1)R^2=75% модель очень хороша

#2)disp,hp близки к 0.05 по p-статистке (mpg в основном описывается через них)

#и также они зависят от mpg отрицательно, у drat одна *, значит он менее сильно
влияет на mpg

# и также он зависит положительно от mpg
```

#3 Вводим в модель логарифмы регрессоров

```
vif(lm(I(log(drat))~I(log(disp))+I(log(hp)),data))

vif(lm(I(log(disp))~I(log(drat))+I(log(hp)),data))

vif(lm(I(log(hp))~I(log(drat))+I(log(disp)),data))
```

#Из vif трех моделей видим что новые столбцы почти линейно независимы
междусобой

```
summary(lm(mpg~I(log(disp))+I((hp))+I((drat)),data))#R^2=81%
summary(lm(mpg~I(log(disp))+I(log(hp))+I((drat)),data))#R^2=82%

model3=lm(mpg~I(log(disp))+I(log(hp))+I(log(drat)),data)
summary(model3)

#Выбираем новую модель так как ее R^2=82% (>75% предыдущей модели)
#Так как log(drat) в новой модели не влияет на mpg (нет звездочек и даже точки),
#не берем его логарифм (здесь можно его вообще убрать)

model4=lm(mpg~I(log(disp))+I(log(hp)),data)
summary(model4)#наилучшая модель R^2=82,9%

#4 Всевозможные пары регрессоров и их квадраты

model5=lm(mpg~I(disp*hp)+drat,data)#R^2=71%
summary(model5)
vif(model5)

model6=lm(mpg~I(disp*drat)+hp,data)#R^2=68%
summary(model6)
vif(model6)
```

```
model7=lm(mpg~I(hp*drat)+disp,data)#R^2=72%
```

```
summary(model7)
```

```
vif(model7)
```

```
model8=lm(mpg~I(disp^2)+I(hp^2)+I(drat^2),data)#R^2=69%
```

```
summary(model8)
```

```
vif(model8)
```

```
model9=lm(mpg~I(disp^2*hp^2)+I(drat),data)#R^2=66%
```

```
summary(model9)
```

```
vif(model9)
```

```
model10=lm(mpg~I(disp^2*hp^2)+I(drat^2),data)#R^2=67%
```

```
summary(model10)
```

```
vif(model10)
```

```
model11=lm(mpg~I(drat^2*hp^2)+I(disp^2),data)#R^2=60%
```

```
summary(model11)
```

```
vif(model11)
```

```
model12=lm(mpg~I(drat^2*hp^2)+I(disp),data)#R^2=70%
```

```
summary(model12)
```

```
vif(model12)
```

```
model13=lm(mpg~I(drat^2*disp^2)+I(hp^2),data)#R^2=56%
summary(model13)
vif(model13)
```

```
model14=lm(mpg~I(drat^2*disp^2)+I(hp),data)#R^2=62%
summary(model14)
vif(model14)
```

```
model15=lm(mpg~I(log(disp)^2)+I(log(hp)^2)+I(log(drat)^2)+I(log(hp)*log(drat)),data)
summary(model15)#R^2=81%
```

```
model15=lm(mpg~I(log(disp))+I(log(hp))+I(log(drat))+I(log(hp)*log(drat)),data)
summary(model15)#R^2=82%
```

```
model16=lm(mpg~I(log(disp))+I(log(hp))+I(log(drat))+I(log(disp)*log(hp)),data)
summary(model16)#R^2=83,8%
```

#Таким образом из 4ого пункат лучше всех

```
# по доле объясненного разброса в данных R^2 = 83,8%
model16=lm(mpg~I(log(disp))+I(log(hp))+I(log(drat))+I(log(disp)*log(hp)),data)
```

"
_____"

#Вторая часть практической

Для зависимости, построенной при решении практического задания №2, оцените:

1. Доверительные интервалы для всех коэффициентов в модели, $p = 95\%$.
2. Сделайте вывод о отверждении или невозможности отвергнуть статистическую гипотезу о том, что коэффициент равен 0.
3. Доверительный интервал для одного прогноза ($p = 95\%$, набор значений регрессоров выбираете сами)."

#1,2) Доверительные интервалы для каждого коэффициента с $p=95\%$ и проверка статистической гипотезы $\text{коэф} = 0$

```
model16=lm(mpg~I(log(disp))+I(log(hp))+I(log(drat))+I(log(disp)*log(hp)),data)  
summary(model16)#R^2=83,8%
```

#Число степеней свободы модели:37-4=33

```
crit=qt(0.975,33)# t-критериq Стьюдента = 2.034515  
coef(model16)
```

```
count_interval <- function(qt,cof,st_er){  
  lw <- cof-qt*st_er  
  up <- cof+qt*st_er  
  newList <- list("lw" = lw, "up" = up)  
  return(newList)  
}
```

```

interval=count_interval(crit,172.068,55.939)

sprintf("[%s , %s]",interval["up"],interval["lw"])]#Интервал для свободного коэф

# Отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале)

interval=count_interval(crit,-25.688,10.445)

sprintf("[%s , %s]",interval["up"],interval["lw"])]#Интервал для log(disp)

# Отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале)

interval=count_interval(crit,-22.972,10.578)

sprintf("[%s , %s]",interval["up"],interval["lw"])]#Интервал для log(hp)

# Отвергаем гипотезу о том, что этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале)

interval=count_interval(crit,-1.978,5.109)

sprintf("[%s , %s]",interval["up"],interval["lw"])]#Интервал для log(drat)

# Этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале)

interval=count_interval(crit,3.781,2.023)

sprintf("[%s , %s]",interval["up"],interval["lw"])]#Интервал для log(disp) * log(hp))

# Этот коэффициент может быть равен 0, на уровне значимости 5% (нет 0 в этом интервале)

#3 Доверительный интервал для одного прогноза

new.data = data.frame(disp = 10, hp = 15, drat=3.5)

```

```
predict(model16, new.data, interval = "confidence")  
#Доверительный интервал для одного прогноза [43.79245, 99.8277]
```

Приложение 3.

```
install.packages("devtools")
devtools::install_github("bdemeshev/rlms")

library("lmtest")
library("rlms")
library("dplyr")
library("GGally")
library("car")
library("sandwich")

data <- read.csv("Downloads/r26i_os26b.csv")
#glimpse(data)
data2 = select(data, vj13.2, v_age, vh5, v_educ, status, vj6.2, v_marst)

#исключаем строки с отсутствующими значениями NA
data2 = na.omit(data2)

#зарплата с элементами нормализации
data2$vj13.2
sal = as.numeric(data2$vj13.2)
sal1 = as.character(data2$vj13.2)
sal2 = lapply(sal1, as.integer)
sal = as.numeric(unlist(sal2))

mean(sal)
```

```
data2["salary"] = (sal - mean(sal)) / sqrt(var(sal))

data2["salary"]
```

#возраст с элементами нормализации

```
age1 = as.character(data2$v_age)

age2 = lapply(age1, as.integer)

age3 = as.numeric(unlist(age2))

data2["age"] = (age3 - mean(age3)) / sqrt(var(age3))

data2["age"]
```

#ПОЛ

```
data2["sex"] = data2$vh5

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

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

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

#образование

```
data2["h_educ"] = data2$v_educ

#data2["h_educ"] = lapply(data2$v_educ, as.character)

data2["higher_educ"] = data2$v_educ

data2["higher_educ"] = 0

data2$higher_educ[which(data2$h_educ == '21')] <- 1

data2$higher_educ[which(data2$h_educ == '22')] <- 1

data2$higher_educ[which(data2$h_educ == '23')] <- 1
```

#населенный пункт

```

data2["status1"] = data2$status
#data2["status1"] = lapply(data2$status, as.character)

data2["status2"] = 0
data2$status2[which(data2$status1=='1')] <- 1
data2$status2[which(data2$status1=='2')] <- 1
data2$status2 = as.numeric(data2$status2)

#продолжительность рабочей недели
dur1 = as.character(data2$vj6.2)
dur2 = lapply(dur1, as.integer)
dur3 = as.numeric(unlist(dur2))
data2["dur"] = (dur3 - mean(dur3)) / sqrt(var(dur3))

#семейное положение
data2["wed"] = data2$v_marst
#data2["wed"] = lapply(data2$v_marst, as.character)

data2$wed1 = 0
data2$wed1[which(data2$wed=='2')] <- 1
data2$wed1 = as.numeric(data2$wed1)

data2["wed2"] = lapply(data2["wed"], as.character)
data2$wed2 = 0
data2$wed2[which(data2$wed=='4')] <- 1
data2$wed2[which(data2$wed=='5')] <- 1
data2$wed2 = as.numeric(data2$wed2)

```

```
data2["wed3"] = data2$v_marst  
data2$wed3 = 0  
data2$wed3[which(data2$wed == '1')] <- 1  
data2$wed3 = as.numeric(data2$wed3)  
  
data2 = na.omit(data2)
```

#учет инфляции: +3.4% к показателям зарплаты 2017 года, +2.52% к показателям 2016 года

```
data2["salary"]  
data2["salary"] = data2["salary"] * 1.034
```

```
data3 = select(data2, salary, age, sex, higher_educ, status2, dur, wed1, wed2, wed3)  
glimpse(data3)
```

#Пункт 1 построение вспомогательной регрессии salary ~ age + sex + higher_educ + status2 + dur + wed1 + wed2 + wed3

```
model_salary = lm(data = data3, salary ~ age + sex + higher_educ + status2 + dur + wed1 + wed2 + wed3)  
summary(model_salary)  
vif(model_salary) #переменные все хорошие , vif < 5
```

#Пункт 2 модели с логарифмами , степенями и произведениями

```
model2 = lm(data = data3, salary~I(log(age+3)) + sex + higher_educ + status2 + dur+
wed1 + wed2 + wed3)
```

```
summary(model2)
```

```
vif(model2)
```

#adj R² =0.025, параметры I(log(age+3))+ higher_educ + dur+wed1+wed2 значимые

```
model3 = lm(data = data3, salary~I(log(age+3)) + sex + higher_educ + status2 +dur+ wed1+
+ wed2 + wed3)
```

```
summary(model3)
```

```
vif(model3)
```

#adj R² =0.0256 ,параметры I(log(age+3))+ higher_educ + dur+wed1+wed2 значимые

```
model5 = lm(data = data3, salary~I(log(age+3)) + sex + higher_educ + status2 +
dur+I(age*dur)+ wed1 + wed2 + wed3)
```

```
summary(model5)
```

```
vif(model5)
```

#adj R² =0.02602 , параметры параметры I(log(age+3))+ higher_educ + dur+wed1+wed2+I(age*dur) значимые

```
model6 = lm(data = data3, salary~I(log(age+3)) + sex + higher_educ + status2 +
I(dur^2)+I(age*dur)+ wed1 + wed2 + wed3)
```

```
summary(model6)
```

```
vif(model6)
```

#adj R² =0,02602 , параметры age + higher_educ + I(dur^2)+I(age*dur)+ wed1 + wed2 значимые

```
model7 = lm(data = data3, salary~I((age+3)^1.5) + sex + higher_educ + status2 + I(dur^2)+I(age*dur)+ wed1 + wed2 + wed3)
```

```
summary(model7)
```

```
vif(model7)
```

#adj R² =0.0258 , параметры I(log(age+3))+higher_educ +wed1+wed2+ I(age*dur)+I(dur^2)- значимые; менее значимые I(age*dur)+wed3

```
model4 = lm(data = data3, salary~I(log(age+3)^(0.1)) + sex + higher_educ + status2 +I(age*dur) +I(dur)+ wed1 + wed2 + wed3)
```

```
summary(model4)
```

```
vif(model4)
```

#adj R² =0.02601, параметры I(log(age+3))+higher_educ +wed1+wed2+ I(dur)- значимые; менее значимые: I(age*dur)

```
model8 = lm(data = data3, salary~I((age+3)^(0.1)) + sex + higher_educ + status2 +I((dur+1)^(0.9))+I(age*dur)+ wed1 + wed2 + wed3)
```

```
summary(model8)
```

```
vif(model8)
```

#adj R² =0.02602 , параметры I(log(age+3)^(0.1))+higher_educ +wed1+wed2+ I((dur+1)^(0.9))- значимые; менее значимые: I(age*dur)

```
model9 = lm(data = data3, (salary+3)^2~I(age+3^(1.9)) + sex + higher_educ + status2 +I((dur+1)^(0.1))+I(age*dur)+ wed1 + wed2 + wed3)
```

```
summary(model9)
```

```
vif(model9)
```

#adj R² =0.025 ,параметры I(log(age+3)^(1.9))+higher_educ +wed1+wed2+ I((dur+1)^(0.1))- значимые; менее значимые: I(age*dur)

```
model10 = lm(data = data3, (salary+3)^(0.1)~I((age+3)^(0.1)) + sex + higher_educ +
status2+I((dur+1)^(0.1))+I(age*dur)+wed1+wed2+wed3)

summary(model10)

vif(model10)
```

#adj R² =0.02603=2,603% , параметры I(log(age+3)^())+higher_educ +I((dur+1)^(0.1))+wed1+wed2 - значимые; менее значимые: I(age*dur)

#Пункт 3

```
#Лучшая модель - model10 ; #adj R2 =0.02603, параметры I(log(age+3)^())+higher_educ +I((dur+1)^(0.1))+wed1+wed2 значимые; менее значимые: I(age*dur)
```

#Пункт 4

```
##Из последней (лучшей model10) модели можно сделать вывод что лучше всех зарабатывают более взрослые респонденты
```

```
# с высшим образованием, более долгой рабочей неделей, (женатые или разведенные)
```

#Пункт 5

```
#Вариант 7. Файл 17. Подмножество С высшим образованием, не из города; женщины, с высшим образованием
```

```
data <- read.csv("Downloads/r17i_os26b.csv")

data2 = select(data, mj13.2, m_age, mh5, m_educ, status, mj6.2, m_marst)
```

```
#исключаем строки с отсутствующими значениями NA
```

```
data2 = na.omit(data2)
```

```
#зарплата с элементами нормализации
```

```
data2$mj13.2
```

```
sal = as.numeric(data2$mj13.2)
```

```
sal1 = as.character(data2$mj13.2)
```

```
sal2 = lapply(sal1, as.integer)
```

```
sal = as.numeric(unlist(sal2))
```

```
mean(sal)
```

```
data2["salary"] = (sal - mean(sal)) / sqrt(var(sal))
```

```
data2["salary"]
```

```
#возраст с элементами нормализации
```

```
age1 = as.character(data2$m_age)
```

```
age2 = lapply(age1, as.integer)
```

```
age3 = as.numeric(unlist(age2))
```

```
data2["age"] = (age3 - mean(age3)) / sqrt(var(age3))
```

```
data2["age"]
```

```
#пол
```

```
data2["sex"] = data2$mh5
```

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

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

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

```

#образование

data2["h_educ"] = data2$m_educ


data2["higher_educ"] = data2$m_educ

data2["higher_educ"] = 0

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


#населенный пункт

data2["status1"] = data2$status

#data2["status1"] = lapply(data2$status, as.character)

data2["status2"] = 0

data2$status2[which(data2$status1=='1')] <- 1
data2$status2[which(data2$status1=='2')] <- 1
data2$status2 = as.numeric(data2$status2)


#продолжительность рабочей недели

dur1 = as.character(data2$mj6.2)

dur2 = lapply(dur1, as.integer)

dur3 = as.numeric(unlist(dur2))

data2["dur"] = (dur3 - mean(dur3)) / sqrt(var(dur3))


#семейное положение

data2["wed"] = data2$m_marst

```

```

data2$wed1 = 0

data2$wed1[which(data2$wed=='2')] <- 1

data2$wed1 = as.numeric(data2$wed1)

data2["wed2"] = lapply(data2["wed"], as.character)

data2$wed2 = 0

data2$wed2[which(data2$wed=='4')] <- 1

data2$wed2[which(data2$wed=='5')] <- 1

data2$wed2 = as.numeric(data2$wed2)

data2["wed3"] = data2$m_marst

data2$wed3 = 0

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

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

data2 = na.omit(data2)

```

#Подмножество : С высшим образованием, не из города

```

data3=subset(data2,higher_educ==1)

data4=subset(data3,status2==0)

data4

```

```

model_top = lm(data = data3, (salary+3)^(0.1)~I((age+3)^(0.1)) + sex + 1 +
status2+I((dur+1)^(0.1))+I(age*dur)+wed1+wed2+wed3)

summary(model_top)

```

```
vif(model_top)#большой vif у wed1 , потому здесь его можно убрать и на результат он не повлияет
```

```
#adj R^2 = 0.06632=6,632% , параметры I((age+3)^(0.1))+I((dur+1)^(0.1))+I(age*dur)  
значимые
```

```
##Вывод : В подмножестве( С высшим образованием, не из города) хорошо зарабатывают более взрослые респонденты
```

```
#с более длинной рабочей неделей
```

```
#Подмножество : женщины с высшим образованием
```

```
data5=subset(data2,higher_educ==1)
```

```
data5=subset(data5,sex==0)
```

```
model_top = lm(data = data5, (salary+3)^(0.1)~I((age+3)^(0.1)) + 0 + 1 +  
status2+I((dur+1)^(0.1))+I(age*dur)+wed1+wed2+wed3)
```

```
summary(model_top)
```

```
vif(model_top)#большой vif у wed1 , потому здесь его можно убрать и на результат он не повлияет
```

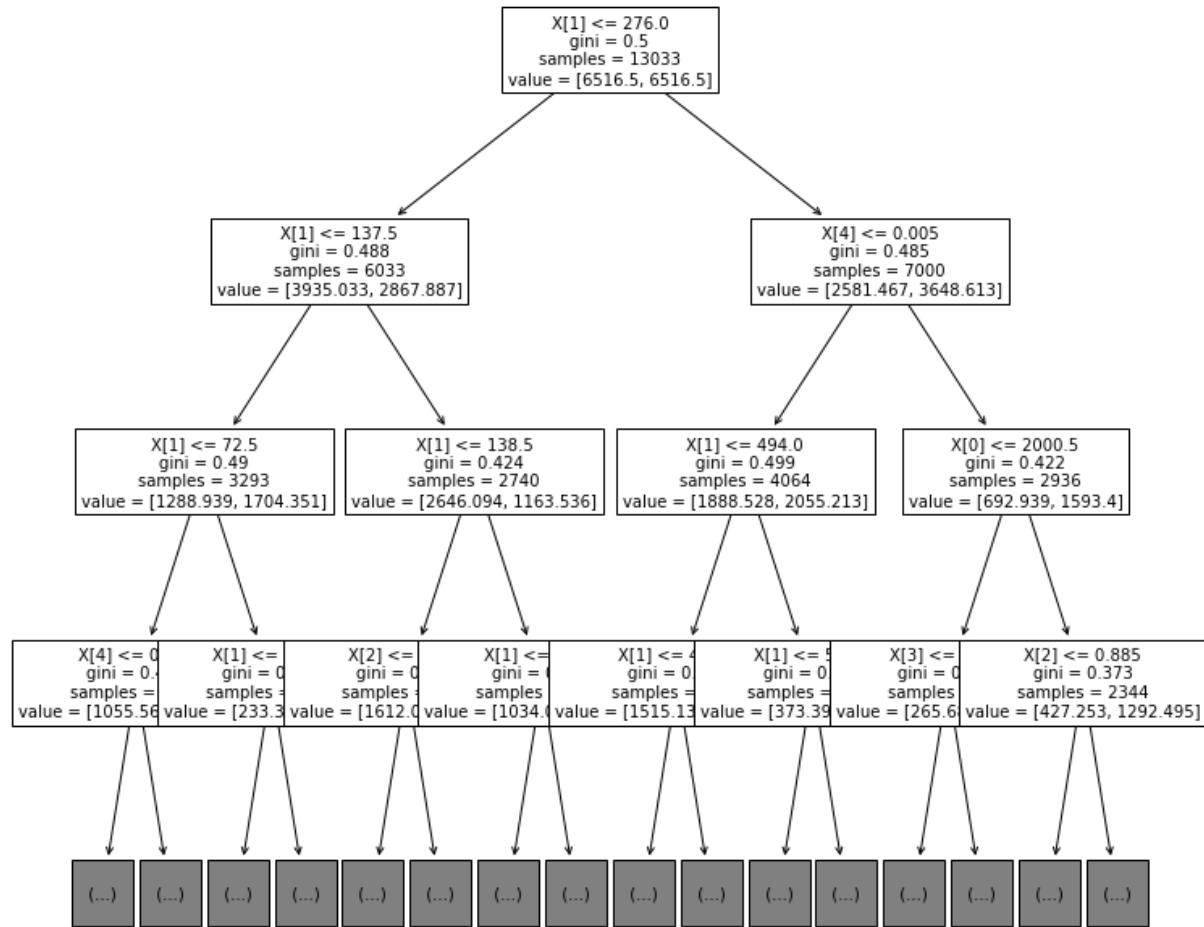
```
#adj R^2 = 0.07919=7,919% , параметры I((age+3)^(0.1))+I((dur+1)^(0.1))+I(age*dur)  
значимые
```

```
##Вывод в подмножестве( С женщины с высшим образованием) хорошо зарабатывают более взрослые респонденты
```

```
#с более длинной рабочей неделей
```

Приложение 4.

DecisionTreeClassifier



```

import numpy as np
import matplotlib as plt
from sklearn import tree
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import OneHotEncoder

import pandas
data =
pandas.read_csv('/home/iv.kem/Downloads/videogames_data/vgsales.csv',
index_col='Rank')

data = data.loc[:, data.columns.isin(['Name', 'Platform', 'Year',
'Genre', 'Publisher', 'NA_Sales',
'EU_Sales', 'JP_Sales', 'Other_Sales', 'Global_Sales'])]
data = data.dropna()

label = LabelEncoder()
dicts = {}

label.fit(data.Publisher.drop_duplicates()) #задаем список значений
для кодирования
dicts['Publisher'] = list(label.classes_)
data.Publisher = label.transform(data.Publisher) #заменяем значения из
 списка кодами закодированных элементов

label.fit(data.Platform.drop_duplicates())#задаем список значений для
кодирования
dicts['Platform'] = list(label.classes_)
data.Platform = label.transform(data.Platform)

data['Genre'] = np.where(data['Genre'] == 'Sports', 0, 1)
data['Genre'] = data['Genre'].astype('int64')

data=data.dropna()
data

```

	Name	Platform
Year \ Rank		
1 2006.0	Wii Sports	26
2 1985.0	Super Mario Bros.	11
3	Mario Kart Wii	26

2008.0									
4									
2009.0									
5									
1996.0									
...									
16596									
2002.0									
16597									
2003.0									
16598	SCORE International Baja 1000: The Official Game								
2008.0									
16599									
2010.0									
16600									
2003.0									

Rank	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	\
1	0	359	41.49	29.02	3.77	8.46	
2	1	359	29.08	3.58	6.81	0.77	
3	1	359	15.85	12.88	3.79	3.31	
4	0	359	15.75	11.01	3.28	2.96	
5	1	359	11.27	8.89	10.22	1.00	
...	
16596	1	269	0.01	0.00	0.00	0.00	
16597	1	241	0.01	0.00	0.00	0.00	
16598	1	21	0.00	0.00	0.00	0.00	
16599	1	8	0.00	0.01	0.00	0.00	
16600	1	544	0.01	0.00	0.00	0.00	

Rank	Global_Sales
1	82.74
2	40.24
3	35.82
4	33.00
5	31.37
...	...
16596	0.01
16597	0.01
16598	0.01
16599	0.01
16600	0.01

[16291 rows x 10 columns]

Проверка продаж на лин. зависимость

```
cov_matrix = np.cov(data['NA_Sales'].values, data['JP_Sales'].values)
print(cov_matrix)
cov_matrix =
np.cov(data['JP_Sales'].values,data['Other_Sales'].values)
print(cov_matrix)
cov_matrix = np.cov(data['JP_Sales'].values, data['EU_Sales'].values)
print(cov_matrix)
cov_matrix = np.cov(data['Other_Sales'].values,
data['EU_Sales'].values)
print(cov_matrix)
cov_matrix = np.cov(data['NA_Sales'].values, data['EU_Sales'].values)
print(cov_matrix)
cov_matrix = np.cov(data['NA_Sales'].values,
data['Other_Sales'].values)
print(cov_matrix)
cov_matrix = np.cov(data['NA_Sales'].values,
data['Global_Sales'].values)
print(cov_matrix)

[[0.6763946  0.11575409]
 [0.11575409  0.09726882]]
 [[0.09726882  0.0172252 ]
 [0.0172252  0.03613149]]
 [[0.09726882  0.06931496]
 [0.06931496  0.25938947]]
 [[0.03613149  0.0703087 ]
 [0.0703087  0.25938947]]
 [[0.6763946  0.32207654]
 [0.32207654  0.25938947]]
 [[0.6763946  0.09919442]
 [0.09919442  0.03613149]]
 [[0.6763946  1.21332847]
 [1.21332847  2.4565688 ]]
```

Делаем вывод что 'Global_Sales' линейно зависим от других продаж, и убираем его из рассмотрения

```
Y = data.loc[:, data.columns.isin(['Genre'])]
X = data.loc[:, 
data.columns.isin(['Year','Publisher','NA_Sales','EU_Sales',
'JP_Sales', 'Other_Sales'])]

X_train = X.sample(frac= 0.8 ,random_state= 0)
X_test= X.drop(X_train. index )

y_train = Y.sample(frac= 0.8 ,random_state= 0)
y_test= Y.drop(y_train. index )

tree1 =
```

```

DecisionTreeClassifier(random_state=241,class_weight='balanced')#max_
leaf_nodes и не дали прироста score
tree1 = tree1.fit(X_train, y_train)

#print("Правильность на обучающем наборе:
{:.3f}".format(tree1.score(X_train, y_train)))
#print("Правильность на тестовом наборе:
{:.3f}".format(tree1.score(X_test, y_test)))

importances = tree1.feature_importances_
importances # важность признаков (делаем вывод что самым важным
является :Publisher)

#data
array([0.20350082, 0.3018054 , 0.19686459, 0.13312733, 0.09516995,
       0.0695319 ])

from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score

predict=tree1.predict(X_test)
report = classification_report(y_test, tree1.predict(X_test))
print(report)

      precision    recall  f1-score   support

          0       0.38      0.40      0.39      489
          1       0.89      0.89      0.89     2769

   accuracy                           0.81      3258
  macro avg       0.64      0.64      0.64      3258
weighted avg       0.82      0.81      0.81      3258

pandas.DataFrame(confusion_matrix(y_test,predict),columns=['Predicted
NO', 'Predicted YES'], index=['Actual NO', 'Actual YES'])

      Predicted NO  Predicted YES
Actual NO           194         295
Actual YES          312         2457

# Finding accuracy, F1, precision and recall
accuracy = accuracy_score(y_test, predict)
print("Accuracy :", accuracy)
precision = precision_score(y_test, predict,average='macro')
print("Precision :", precision)
recall = recall_score(y_test, predict,average='macro')

```

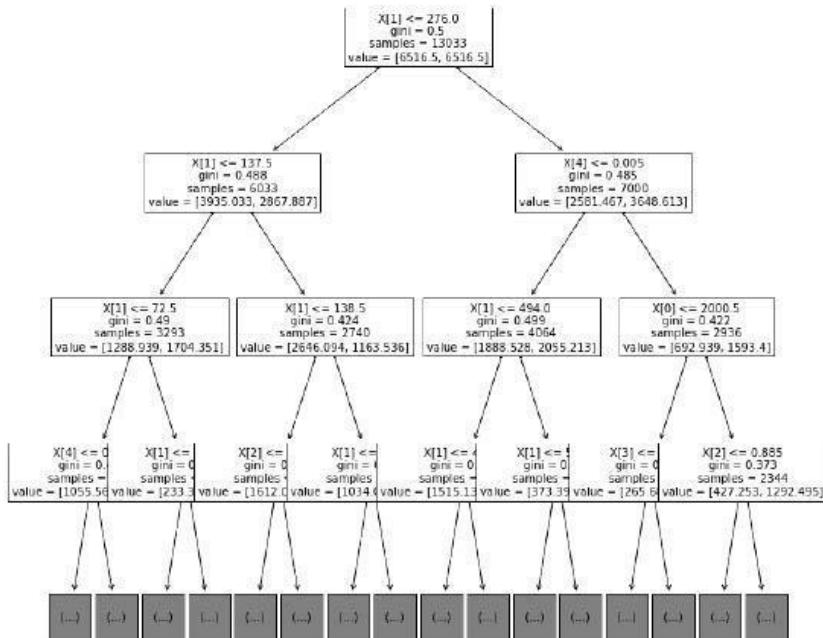
```

print("Recall      : ", recall)
F1_score = f1_score(y_test, predict, average='macro')
print("F1-score   : ", F1_score)

Accuracy   : 0.8136893799877225
Precision  : 0.6381022210221527
Recall     : 0.642025980010945
F1-score   : 0.6400029489960215

plt.figure(figsize=(12,12))
tree.plot_tree(tree1,max_depth=3,fontsize=10)
plt.show()

```



Высокие показатели метрик дают понять нам, что модель хорошо предсказывает классы (процент правильности порядка 64%)

```

import graphviz from sklearn import tree
dot_data = tree.export_graphviz(tree1,
out_file=None) graph = graphviz.Source(dot_data) graph
graph.render('/home/iv.kem/Pictures/data.png')

```

```

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV

from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report

X = data[['Year', 'Publisher', 'NA_Sales', 'Platform',
          'EU_Sales', 'JP_Sales', 'Other_Sales', 'Global_Sales']] #беру
на 1 больше параметр ('Platform') для лучшего результата
y = data['Genre']

X = X.to_numpy()
y = y.to_numpy()

X1_train, X1_test, y1_train, y1_test = train_test_split(X, y,
test_size=0.33, random_state=42, stratify=y)

RF = RandomForestClassifier(n_estimators=100, random_state=42)
RF.fit(X1_train, y1_train)

y_pred = RF.predict(X1_test)

print(classification_report(y1_test,y_pred))

precision    recall   f1-score   support
0            0.64      0.27      0.38      760
1            0.89      0.98      0.93     4617

accuracy                           0.88      5377
macro avg             0.76      0.62      0.65      5377
weighted avg            0.85      0.88      0.85      5377

accuracy = accuracy_score(y1_test, y_pred)
print("Accuracy : ", accuracy)
precision = precision_score(y1_test, y_pred, average='macro')
print("Precision : ", precision)
recall = recall_score(y1_test, y_pred, average='macro')
print("Recall : ", recall)
F1_score = f1_score(y1_test, y_pred, average='macro')
print("F1-score : ", F1_score)

Accuracy : 0.875023247163846
Precision : 0.7641319847186843
Recall : 0.6210986571366688
F1-score : 0.6535865952980463

```

Точность предсказания выросла и стала порядка 65%

```
X1_train, X1_test, y1_train, y1_test = train_test_split(X, y,
test_size=0.33, random_state=42, stratify=y)
RF = RandomForestClassifier()
grid_parametrs = {
    'n_estimators':[50, 100, 150, 200, 250, 300, 350, 400, 450, 500],
    'max_depth':[1,3,5,9,10,12,15,17]
}

grid =
GridSearchCV(RF,param_grid=grid_parametrs,cv=3,scoring='accuracy')
model_grid = grid.fit(X1_train,y1_train)

print('Лучшие гиперпараметры:' + str(model_grid.best_params_))
print('Best score is: ' + str(model_grid.best_score_))

Лучшие гиперпараметры:{'max_depth': 12, 'n_estimators': 100}
Best score is: 0.8702583837273227

grid_parametrs = {
    'n_estimators':[50,60,70,80,90,100,110,120,130,140,150],
    'max_depth':[12]
}

grid =
GridSearchCV(RF,param_grid=grid_parametrs,cv=3,scoring='accuracy')
model_grid = grid.fit(X1_train,y1_train)

print('Лучшие гиперпараметры:' + str(model_grid.best_params_))
pred=grid.predict(X1_test)

Лучшие гиперпараметры:{'max_depth': 12, 'n_estimators': 110}

report = classification_report(y1_test, pred)
print(report)
F1_score = f1_score(y1_test, pred, average='macro')
print("F1-score : ", F1_score)

      precision    recall  f1-score   support
          0       0.84      0.15      0.26      760
          1       0.88      1.00      0.93     4617
   accuracy                           0.88      5377
  macro avg       0.86      0.57      0.60      5377
weighted avg       0.87      0.88      0.84      5377

F1-score : 0.596405386446071
```

Случайный Лес + GridSearch дают худшую точность = 60% > 64% =
точность Решающего Дерева по метрике F1 , так как данные не
сбалансированы. Лучшую точность дает Random Forest без
перебора гиперпараметров (точность порядка 65%)

Приложение 5.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
%matplotlib inline

data=pd.read_csv('/home/iv.kem/Downloads/archive/TravelInsurancePrediction.csv', index_col = "Index")

data
   Age           Employment Type GraduateOrNot
AnnualIncome \
Index

0      31      Government Sector      Yes     400000
1      31 Private Sector/Self Employed      Yes    1250000
2      34 Private Sector/Self Employed      Yes     500000
3      28 Private Sector/Self Employed      Yes     700000
4      28 Private Sector/Self Employed      Yes     700000
...
1982    33 Private Sector/Self Employed      Yes    1500000
1983    28 Private Sector/Self Employed      Yes    1750000
1984    28 Private Sector/Self Employed      Yes    1150000
1985    34 Private Sector/Self Employed      Yes    1000000
1986    34 Private Sector/Self Employed      Yes     500000

   FamilyMembers ChronicDiseases FrequentFlyer
EverTravelledAbroad \
Index

0          No
1          No
```

No			
2	4	1	No
No			
3	3	1	No
No			
4	8	1	Yes
No			
...
1982	4	0	Yes
Yes			
1983	5	1	No
Yes			
1984	6	1	No
No			
1985	6	0	Yes
Yes			
1986	4	0	No
No			
TravelInsurance			
Index			
0	0		
1	0		
2	1		
3	0		
4	0		
...	...		
1982	1		
1983	0		
1984	0		
1985	1		
1986	0		

[1987 rows x 9 columns]

№1 Сколько в наборе данных объектов и признаков? Дать описание каждому признаку, если оно есть.

data.shape

(1987, 9)

Всего в наборе 1987 объектов и 9 признака.

Age - Возраст учащегося. *Employment Type* - Тип занятости.

GraduateOrNot - Выпускник Или Нет. *AnnualIncome* - Годовой доход.

FamilyMembers - Члены семьи. *ChronicDiseases* - Хронические Заболевания. *FrequentFlyer* - Часто летающий пассажир.

EverTravelledAbroad - Когда-либо путешествовал за границу.

TravelInsurance - Страхование путешествий.

№2 Сколько категориальных признаков, какие?

В данном наборе данных 4 категориальных признака. К категориальным признакам относятся: *Employment Type* - Тип занятости. *GraduateOrNot* - Выпускник Или Нет. *FrequentFlyer* - Часто летающий пассажир. *EverTravelledAbroad* - Когда-либо путешествовал за границу.

```
data_nominal=data.loc[:,data.columns.isin(['Employment
Type','GraduateOrNot','FrequentFlyer','EverTravelledAbroad'])]
data_nominal

                           Employment Type GraduateOrNot FrequentFlyer \
Index
0                  Government Sector      Yes        No
1  Private Sector/Self Employed      Yes        No
2  Private Sector/Self Employed      Yes        No
3  Private Sector/Self Employed      Yes        No
4  Private Sector/Self Employed      Yes       Yes
...
1982  Private Sector/Self Employed      Yes       Yes
1983  Private Sector/Self Employed      Yes        No
1984  Private Sector/Self Employed      Yes        No
1985  Private Sector/Self Employed      Yes       Yes
1986  Private Sector/Self Employed      Yes        No

                           EverTravelledAbroad
Index
0                    No
1                    No
2                    No
3                    No
4                    No
...
1982                   ...
1983                   Yes
1984                   No
1985                   Yes
1986                   No

[1987 rows x 4 columns]
```

№3 Столбец с максимальным количеством уникальных значений категориального признака?

```
print(data['Employment Type'].value_counts())
print(data['GraduateOrNot'].value_counts())
print(data['FrequentFlyer'].value_counts())
print(data['EverTravelledAbroad'].value_counts())
```

```
Private Sector/Self Employed    1417
Government Sector                570
```

```

Name: Employment Type, dtype: int64
Yes      1692
No       295
Name: GraduateOrNot, dtype: int64
No      1570
Yes      417
Name: FrequentFlyer, dtype: int64
No      1607
Yes      380
Name: EverTravelledAbroad, dtype: int64

```

Все эти 4 столбца имеют максимальное количество уникальных значений.

№4 Есть ли бинарные признаки?

Да, к таким признакам относятся: *Employment Type* - Тип занятости. *GraduateOrNot* - Выпускник Или Нет. *FrequentFlyer* - Часто летающий пассажир. *EverTravelledAbroad* - Когда-либо путешествовал за границу. *ChronicDiseases* - Хронические Заболевания. *TravelInsurance* - Страхование путешествий.

```

data_binary=data.loc[:,data.columns.isin(['Employment
Type','GraduateOrNot','ChronicDiseases','FrequentFlyer',
'EverTravelledAbroad','TravelInsurance'])]
data_binary

Index          Employment Type GraduateOrNot ChronicDiseases \
0           Government Sector      Yes            1
1   Private Sector/Self Employed    Yes            0
2   Private Sector/Self Employed    Yes            1
3   Private Sector/Self Employed    Yes            1
4   Private Sector/Self Employed    Yes            1
...
1982  Private Sector/Self Employed    Yes            0
1983  Private Sector/Self Employed    Yes            1
1984  Private Sector/Self Employed    Yes            1
1985  Private Sector/Self Employed    Yes            0
1986  Private Sector/Self Employed    Yes            0

Index          FrequentFlyer EverTravelledAbroad TravelInsurance
0             No            No            0
1             No            No            0
2             No            No            1
3             No            No            0
4            Yes            No            0
...
1982           ...           ...           ...
1983           Yes           Yes            1
1984           No            Yes            0
1984           No           No            0

```

```
1985          Yes        Yes      1
1986          No         No       0
```

[1987 rows x 6 columns]

№5 Какие числовые признаки?

К числовым признакам относятся: *Age* - Возраст учащегося.
AnnualIncome - Годовой доход. *FamilyMembers* - Члены семьи.

```
data_numeric=data.loc[:,data.columns.isin(['Age','AnnualIncome','FamilyMembers'])]
data_numeric
```

```
   Age  AnnualIncome  FamilyMembers
Index
0     31      400000           6
1     31     1250000           7
2     34      500000           4
3     28      700000           3
4     28      700000           8
...
1982    33     1500000           4
1983    28     1750000           5
1984    28     1150000           6
1985    34     1000000           6
1986    34      500000           4
```

[1987 rows x 3 columns]

№6 Есть ли пропуски?

```
data.isna().sum()
```

```
Age          0
Employment Type  0
GraduateOrNot  0
AnnualIncome  0
FamilyMembers  0
ChronicDiseases  0
FrequentFlyer  0
EverTravelledAbroad  0
TravelInsurance  0
dtype: int64
```

Пропусков нет.

№7 Сколько объектов с пропусками?

```
print(sum(data.isnull().sum(axis=1)))
```

0

0

№8 Столбец с максимальным количеством пропусков?

Такого столбца нет, так как пропусков в наборе данных нет.

№9 Есть ли на ваш взгляд выбросы? Аномальные значения?
data.describe()

```
          Age  AnnualIncome  FamilyMembers  ChronicDiseases \
count    1987.000000   1.987000e+03   1987.000000   1987.000000
mean     29.650226   9.327630e+05      4.752894     0.277806
std      2.913308   3.768557e+05      1.609650     0.448030
min     25.000000   3.000000e+05      2.000000     0.000000
25%    28.000000   6.000000e+05      4.000000     0.000000
50%    29.000000   9.000000e+05      5.000000     0.000000
75%    32.000000   1.250000e+06      6.000000     1.000000
max    35.000000   1.800000e+06      9.000000     1.000000

TravelInsurance
count      1987.000000
mean       0.357323
std        0.479332
min       0.000000
25%       0.000000
50%       0.000000
75%       1.000000
max       1.000000
```

Здесь нет сильных различий между максимальными значениями и средними, потому можно сделать вывод что вбросов/аномалий в данных нет

№10 Столбец с максимальным средним значением после нормировки признаков через стандартное отклонение?

В ходе нормализации будем нормализовывать числовые признаки, но которые принимают реальные вещественные значения, а не являются описанием какого-либо признака. К таким признакам относятся: -Age - AnnualIncome

```
scale_features_std = StandardScaler()
features_std = scale_features_std.fit_transform(data[['Age']])
data['Age_std']=features_std
features_std=scale_features_std.fit_transform(data[['AnnualIncome']])
data['AnnualIncome_std']=features_std

data_std=data.loc[:,data.columns.isin(['Age_std','AnnualIncome_std'])]
data_std.describe()
```

0

№8 Столбец с максимальным количеством пропусков?

Такого столбца нет, так как пропусков в наборе данных нет.

№9 Есть ли на ваш взгляд выбросы? Аномальные значения?
data.describe()

```
          Age  AnnualIncome  FamilyMembers  ChronicDiseases \
count    1987.000000   1.987000e+03   1987.000000   1987.000000
mean     29.650226   9.327630e+05      4.752894     0.277806
std      2.913308   3.768557e+05      1.609650     0.448030
min     25.000000   3.000000e+05      2.000000     0.000000
25%    28.000000   6.000000e+05      4.000000     0.000000
50%    29.000000   9.000000e+05      5.000000     0.000000
75%    32.000000   1.250000e+06      6.000000     1.000000
max    35.000000   1.800000e+06      9.000000     1.000000

TravelInsurance
count    1987.000000
mean     0.357323
std      0.479332
min     0.000000
25%    0.000000
50%    0.000000
75%    1.000000
max    1.000000
```

Здесь нет сильных различий между максимальными значениями и средними, потому можно сделать вывод что вбросов/аномалий в данных нет

№10 Столбец с максимальным средним значением после нормировки признаков через стандартное отклонение?

В ходе нормализации будем нормализовывать числовые признаки, но которые принимают реальные вещественные значения, а не являются описанием какого-либо признака. К таким признакам относятся: -Age - AnnualIncome

```
scale_features_std = StandardScaler()
features_std = scale_features_std.fit_transform(data[['Age']])
data['Age_std']=features_std
features_std=scale_features_std.fit_transform(data[['AnnualIncome']])
data['AnnualIncome_std']=features_std

data_std=data.loc[:,data.columns.isin(['Age_std','AnnualIncome_std'])]
data_std.describe()
```

	Age_std	AnnualIncome_std
count	1.987000e+03	1.987000e+03
mean	-5.616488e-16	1.477317e-16
std	1.000252e+00	1.000252e+00
min	-1.596603e+00	-1.679482e+00
25%	-5.665868e-01	-8.832207e-01
50%	-2.232480e-01	-8.695957e-02
75%	8.067684e-01	8.420117e-01
max	1.836785e+00	2.301824e+00

Столбец с максимальным средним значением после нормализации с помощью стандартного отклонения - *AnnualIncome_std*

№11 Столбец с целевым признаком?

В наборе данных можно выделить один целевой признак: *TravelInsurance*
- Страхование путешествий.

```
target=data['TravelInsurance']

data['TravelInsurance'].to_numpy()

print(target)

Index
0      0
1      0
2      1
3      0
4      0
 ..
1982    1
1983    0
1984    0
1985    1
1986    0
Name: TravelInsurance, Length: 1987, dtype: int64
```

№12 Сколько параметров попадает в тренировочную выборку при параметрах: *test_size=0.3, random_state=42?*

Выделяем подмножество параметров, удаляем из изначального набора данных параметр *TravelInsurance*. Но для начала сделаем предобработку категориальных признаков, а также бинарных, которые в столбцах содержат строчные типы данных.

Предобработка категориальных признаков.

```
data.rename(columns = {'Employment
Type':'Employment_Type'},inplace=True)
```

```

label = LabelEncoder()
label.fit(data.Employment_Type)
data.Employment_Type= label.transform(data.Employment_Type)

label = LabelEncoder()
label.fit(data.GraduateOrNot)
data.GraduateOrNot = label.transform(data.GraduateOrNot)

label = LabelEncoder()
label.fit(data.FrequentFlyer)
data.FrequentFlyer = label.transform(data.FrequentFlyer)

label = LabelEncoder()
label.fit(data.EverTravelledAbroad)
data.EverTravelledAbroad = label.transform(data.EverTravelledAbroad)

data_nominal=data.loc[:,data.columns.isin(['Employment_Type','Graduate
OrNot','FrequentFlyer','EverTravelledAbroad'])]
data_nominal

      Employment_Type  GraduateOrNot  FrequentFlyer
EverTravelledAbroad
Index

0                  0              1              0
0
1                  1              1              0
0
2                  1              1              0
0
3                  1              1              0
0
4                  1              1              1
0
...
...
1982                1              1              1
1
1983                1              1              0
1
1984                1              1              0
0
1985                1              1              1
1
1986                1              1              0
0

[1987 rows x 4 columns]

data

```

	Age	Employment_Type	GraduateOrNot	AnnualIncome
FamilyMembers \ Index				
0	31	0	1	400000
6				
1	31	1	1	1250000
7				
2	34	1	1	500000
4				
3	28	1	1	700000
3				
4	28	1	1	700000
8				
...
.				
1982	33	1	1	1500000
4				
1983	28	1	1	1750000
5				
1984	28	1	1	1150000
6				
1985	34	1	1	1000000
6				
1986	34	1	1	500000
4				

	ChronicDiseases	FrequentFlyer	EverTravelledAbroad
TravelInsurance \ Index			
0	1	0	0
0			
1	0	0	0
0			
2	1	0	0
1			
3	1	0	0
0			
4	1	1	0
0			
...
...			
1982	0	1	1
1			
1983	1	0	1
0			
1984	1	0	0
0			
1985	0	1	1

```
1  
1986          0          0          0  
0
```

```
      Age_std  AnnualIncome_std  
Index  
0    0.463430      -1.414061  
1    0.463430       0.842012  
2    1.493446      -1.148641  
3   -0.566587      -0.617800  
4   -0.566587      -0.617800  
...  ...  ...  
1982  1.150107      1.505563  
1983  -0.566587      2.169114  
1984  -0.566587      0.576591  
1985  1.493446      0.178461  
1986  1.493446     -1.148641
```

```
[1987 rows x 11 columns]
```

```
one_hot = pd.get_dummies(data['FamilyMembers'],prefix='FamMem')  
# Drop column B as it is now encoded  
data = data.drop('FamilyMembers',axis = 1)  
# Join the encoded df  
data = data.join(one_hot)  
data
```

```
      Age Employment_Type GraduateOrNot  AnnualIncome  
ChronicDiseases \\  
Index  
0      31          0           1      400000  
1      31          1           1      1250000  
0      34          1           1      500000  
1      28          1           1      700000  
1      28          1           1      700000  
4      28          1           1      700000  
1      ...  ...  ...  ...  ...  
...  ...  ...  ...  ...  
1982  33          1           1      1500000  
0      28          1           1      1750000  
1      28          1           1      1150000  
1      34          1           1      1000000
```

0					
1986	34	1	1	500000	
0					
	FrequentFlyer	EverTravelledAbroad	TravelInsurance		
Age_std	\				
Index					
0	0	0	0	0	0.463430
1	0	0	0	0	0.463430
2	0	0	0	1	1.493446
3	0	0	0	0	-0.566587
4	1	0	0	0	-0.566587
...
1982	1	1	1	1	1.150107
1983	0	1	0	0	-0.566587
1984	0	0	0	0	-0.566587
1985	1	1	1	1	1.493446
1986	0	0	0	0	1.493446
	AnnualIncome_std	FamMem_2	FamMem_3	FamMem_4	FamMem_5
FamMem_6	\				
Index					
0	-1.414061	0	0	0	0
1	0.842012	0	0	0	0
1	0.842012	0	0	0	0
0	-1.148641	0	0	1	0
2	-1.148641	0	0	0	0
0	-0.617800	0	1	0	0
3	-0.617800	0	0	0	0
0	-0.617800	0	0	0	0
4	-0.617800	0	0	0	0
0					
...
1982	1.505563	0	0	1	0
0					

```

1983      2.169114      0      0      0      1
0
1984      0.576591      0      0      0      0
1
1985      0.178461      0      0      0      0
1
1986     -1.148641      0      0      1      0
0

          FamMem_7  FamMem_8  FamMem_9
Index
0          0          0          0
1          1          0          0
2          0          0          0
3          0          0          0
4          0          1          0
...
1982      0          0          0
1983      0          0          0
1984      0          0          0
1985      0          0          0
1986      0          0          0

[1987 rows x 18 columns]

parameters=data.loc[:,data.columns.isin(['Age_std','AnnualIncome_std',
'Employment_Type','GraduateOrNot','ChronicDiseases','FrequentFlyer','EverTravelledAbroad','TravelInsurance','FamMem_2','FamMem_3','FamMem_4',
'FamMem_5','FamMem_6','FamMem_7','FamMem_8','FamMem_9'])]
parameters

          Employment_Type  GraduateOrNot  ChronicDiseases  FrequentFlyer
\Index
0          0            1            1            0
1          1            1            0            0
2          1            1            1            0
3          1            1            1            0
4          1            1            1            1
...
1982      1            1            0            1
1983      1            1            1            0

```

1984	1	1	1	0
1985	1	1	0	1
1986	1	1	0	0

	EverTravelledAbroad	TravelInsurance	Age_std	
AnnualIncome_std \ Index				
0	0	0	0.463430	-
1.414061	0	0	0.463430	
1	0	1	1.493446	-
0.842012	0	0	-0.566587	-
2	0	0	-0.566587	-
1.148641	0	0	-0.566587	-
3	0	0	-0.566587	-
0.617800	0	0	-0.566587	-
4	0	0	-0.566587	-
0.617800	0	0	-0.566587	-
...
.				
1982	1	1	1.150107	
1.505563	1	0	-0.566587	
1983	1	0	-0.566587	
2.169114	0	0	-0.566587	
1984	0	0	-0.566587	
0.576591	0	0	-0.566587	
1985	1	1	1.493446	
0.178461	1	0	1.493446	-
1986	0	0	1.493446	-
1.148641	0	0	1.493446	-

	FamMem_2	FamMem_3	FamMem_4	FamMem_5	FamMem_6	FamMem_7	
FamMem_8 \ Index							
0	0	0	0	0	1	0	
0	0	0	0	0	0	1	
1	0	0	0	0	0	0	
0	0	0	1	0	0	0	
2	0	0	1	0	0	0	
0	0	1	0	0	0	0	
3	0	1	0	0	0	0	
0	0	0	0	0	0	0	
4	0	0	0	0	0	0	
1	0	0	0	0	0	0	

```

...
...
1982      0      0      1      0      0      0
0
1983      0      0      0      1      0      0
0
1984      0      0      0      0      1      0
0
1985      0      0      0      0      1      0
0
1986      0      0      1      0      0      0
0

   FamMem_9
Index
0      0
1      0
2      0
3      0
4      0
...
1982      0
1983      0
1984      0
1985      0
1986      0

[1987 rows x 16 columns]

```

Выделение тренировочной и тестовой выборок.

```

x_train, x_test, y_train,
y_test=train_test_split(parameters,target,test_size=0.3,random_state=4
2)
print(y_test.shape)
print(x_test.shape)

(597,)
(597, 16)

```

При заданных параметрах в тренировочную выборку попадает 597 объектов.

№ 13 Между какими признаками наблюдается линейная зависимость (корреляция)?

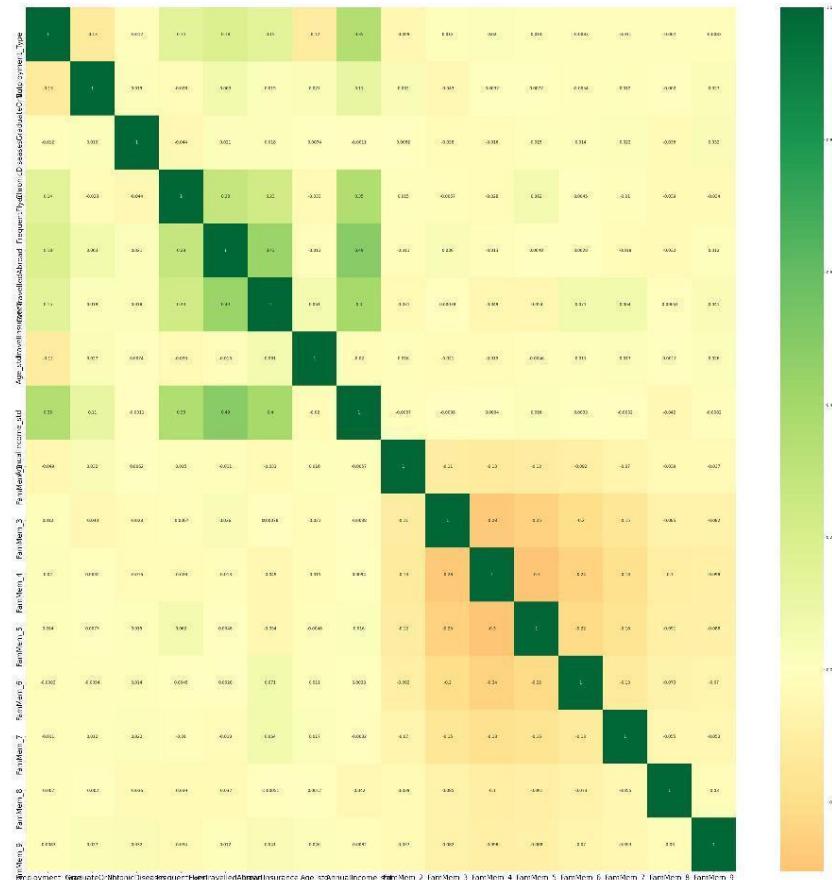
```

plt.figure(figsize=(40,40), dpi= 160)
sns.heatmap(parameters.corr(), xticklabels=parameters.corr().columns,
yticklabels=parameters.corr().columns, cmap='RdYlGn', center=0,
annot=True)

plt.xticks(fontsize=18)

```

```
plt.yticks(fontsize=18)
plt.show()
```



Из таблицы можно сделать вывод, что в этом датасете нет линейно-зависимых параметров.

№ 14 Сколько признаков достаточно для объяснения 90% дисперсии после применения метода PCA?

```
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,parameters.columns)))

1 component: 29.48% of initial variance
0.174 x Employment_Type + 0.028 x GraduateOrNot + 0.000 x
ChronicDiseases + 0.158 x FrequentFlyer + 0.205 x EverTravelledAbroad
+ 0.216 x TravelInsurance + -0.237 x Age_std + 0.894 x
AnnualIncome_std + 0.001 x FamMem_2 + 0.002 x FamMem_3 + -0.007 x
FamMem_4 + 0.015 x FamMem_5 + 0.007 x FamMem_6 + -0.009 x FamMem_7 + -
0.006 x FamMem_8 + -0.003 x FamMem_9
2 component: 25.78% of initial variance
-0.032 x Employment_Type + 0.025 x GraduateOrNot + -0.003 x
ChronicDiseases + 0.031 x FrequentFlyer + 0.042 x EverTravelledAbroad
+ 0.089 x TravelInsurance + 0.968 x Age_std + 0.226 x AnnualIncome_std
+ 0.007 x FamMem_2 + -0.022 x FamMem_3 + -0.002 x FamMem_4 + 0.004 x
FamMem_5 + 0.015 x FamMem_6 + -0.003 x FamMem_7 + -0.003 x FamMem_8 +
0.004 x FamMem_9
3 component: 6.06% of initial variance
0.108 x Employment_Type + -0.022 x GraduateOrNot + -0.038 x
ChronicDiseases + -0.125 x FrequentFlyer + -0.074 x
EverTravelledAbroad + -0.131 x TravelInsurance + 0.009 x Age_std +
0.067 x AnnualIncome_std + -0.012 x FamMem_2 + -0.172 x FamMem_3 +
0.816 x FamMem_4 + -0.488 x FamMem_5 + -0.087 x FamMem_6 + -0.042 x
FamMem_7 + -0.007 x FamMem_8 + -0.008 x FamMem_9
4 component: 5.47% of initial variance
-0.170 x Employment_Type + 0.035 x GraduateOrNot + 0.434 x
ChronicDiseases + 0.006 x FrequentFlyer + 0.228 x EverTravelledAbroad
+ 0.530 x TravelInsurance + -0.023 x Age_std + -0.149 x
AnnualIncome_std + 0.014 x FamMem_2 + 0.307 x FamMem_3 + -0.069 x
FamMem_4 + -0.536 x FamMem_5 + 0.180 x FamMem_6 + 0.078 x FamMem_7 + -
0.001 x FamMem_8 + 0.026 x FamMem_9
5 component: 5.2% of initial variance
-0.176 x Employment_Type + 0.127 x GraduateOrNot + 0.817 x
ChronicDiseases + -0.049 x FrequentFlyer + -0.035 x
EverTravelledAbroad + -0.155 x TravelInsurance + -0.017 x Age_std +
0.076 x AnnualIncome_std + 0.012 x FamMem_2 + -0.387 x FamMem_3 +
0.117 x FamMem_4 + 0.296 x FamMem_5 + -0.010 x FamMem_6 + -0.016 x
FamMem_7 + -0.022 x FamMem_8 + 0.009 x FamMem_9
6 component: 4.99% of initial variance
0.238 x Employment_Type + -0.069 x GraduateOrNot + 0.285 x
ChronicDiseases + -0.262 x FrequentFlyer + -0.213 x
EverTravelledAbroad + -0.549 x TravelInsurance + 0.046 x Age_std +
0.197 x AnnualIncome_std + 0.011 x FamMem_2 + 0.545 x FamMem_3 + -
0.215 x FamMem_4 + -0.229 x FamMem_5 + -0.058 x FamMem_6 + -0.046 x
FamMem_7 + -0.008 x FamMem_8 + -0.001 x FamMem_9
7 component: 4.55% of initial variance
0.753 x Employment_Type + -0.448 x GraduateOrNot + 0.230 x
ChronicDiseases + 0.196 x FrequentFlyer + 0.079 x EverTravelledAbroad
+ 0.205 x TravelInsurance + 0.061 x Age_std + -0.220 x
AnnualIncome_std + -0.031 x FamMem_2 + -0.029 x FamMem_3 + 0.056 x

```

$FamMem_4 + 0.149 \times FamMem_5 + -0.116 \times FamMem_6 + -0.021 \times FamMem_7 +$
 $-0.013 \times FamMem_8 + 0.005 \times FamMem_9$
 8 component: 4.02% of initial variance
 $0.248 \times Employment_Type + -0.055 \times GraduateOrNot + -0.052 \times$
 $ChronicDiseases + -0.232 \times FrequentFlyer + -0.210 \times$
 $EverTravelledAbroad + -0.032 \times TravelInsurance + -0.003 \times Age_std +$
 $0.047 \times AnnualIncome_std + 0.023 \times FamMem_2 + -0.431 \times FamMem_3 + -$
 $0.237 \times FamMem_4 + -0.217 \times FamMem_5 + 0.729 \times FamMem_6 + 0.095 \times$
 $FamMem_7 + 0.020 \times FamMem_8 + 0.018 \times FamMem_9$
 9 component: 3.56% of initial variance
 $-0.212 \times Employment_Type + -0.263 \times GraduateOrNot + 0.049 \times$
 $ChronicDiseases + 0.818 \times FrequentFlyer + -0.052 \times EverTravelledAbroad$
 $+ -0.343 \times TravelInsurance + 0.004 \times Age_std + 0.002 \times$
 $AnnualIncome_std + 0.012 \times FamMem_2 + 0.005 \times FamMem_3 + 0.002 \times$
 $FamMem_4 + -0.186 \times FamMem_5 + 0.239 \times FamMem_6 + -0.030 \times FamMem_7 +$
 $-0.005 \times FamMem_8 + -0.035 \times FamMem_9$
 10 component: 2.86% of initial variance
 $-0.403 \times Employment_Type + -0.818 \times GraduateOrNot + -0.006 \times$
 $ChronicDiseases + -0.284 \times FrequentFlyer + -0.165 \times$
 $EverTravelledAbroad + 0.128 \times TravelInsurance + -0.024 \times Age_std +$
 $0.156 \times AnnualIncome_std + -0.015 \times FamMem_2 + -0.035 \times FamMem_3 + -$
 $0.005 \times FamMem_4 + 0.045 \times FamMem_5 + -0.088 \times FamMem_6 + 0.094 \times$
 $FamMem_7 + 0.004 \times FamMem_8 + 0.000 \times FamMem_9$
 11 component: 2.72% of initial variance
 $0.070 \times Employment_Type + 0.113 \times GraduateOrNot + 0.016 \times$
 $ChronicDiseases + 0.142 \times FrequentFlyer + -0.290 \times EverTravelledAbroad$
 $+ 0.025 \times TravelInsurance + 0.002 \times Age_std + 0.032 \times AnnualIncome_std$
 $+ 0.081 \times FamMem_2 + -0.183 \times FamMem_3 + -0.167 \times FamMem_4 + -0.203 \times$
 $FamMem_5 + -0.383 \times FamMem_6 + 0.786 \times FamMem_7 + 0.040 \times FamMem_8 +$
 $0.029 \times FamMem_9$
 12 component: 2.21% of initial variance
 $-0.008 \times Employment_Type + -0.145 \times GraduateOrNot + -0.043 \times$
 $ChronicDiseases + -0.155 \times FrequentFlyer + 0.828 \times EverTravelledAbroad$
 $+ -0.376 \times TravelInsurance + 0.017 \times Age_std + -0.056 \times$
 $AnnualIncome_std + 0.085 \times FamMem_2 + -0.162 \times FamMem_3 + -0.091 \times$
 $FamMem_4 + -0.098 \times FamMem_5 + -0.027 \times FamMem_6 + 0.265 \times FamMem_7 +$
 $0.001 \times FamMem_8 + 0.026 \times FamMem_9$
 13 component: 1.45% of initial variance
 $0.010 \times Employment_Type + -0.027 \times GraduateOrNot + -0.008 \times$
 $ChronicDiseases + 0.015 \times FrequentFlyer + -0.048 \times EverTravelledAbroad$
 $+ 0.035 \times TravelInsurance + -0.008 \times Age_std + -0.002 \times$
 $AnnualIncome_std + 0.824 \times FamMem_2 + -0.190 \times FamMem_3 + -0.173 \times$
 $FamMem_4 + -0.180 \times FamMem_5 + -0.223 \times FamMem_6 + -0.352 \times FamMem_7 +$
 $0.151 \times FamMem_8 + 0.144 \times FamMem_9$
 14 component: 0.9% of initial variance
 $-0.009 \times Employment_Type + -0.004 \times GraduateOrNot + 0.008 \times$
 $ChronicDiseases + 0.026 \times FrequentFlyer + 0.009 \times EverTravelledAbroad$
 $+ -0.029 \times TravelInsurance + -0.000 \times Age_std + 0.009 \times$
 $AnnualIncome_std + -0.424 \times FamMem_2 + -0.144 \times FamMem_3 + -0.137 \times$
 $FamMem_4 + -0.144 \times FamMem_5 + -0.153 \times FamMem_6 + -0.182 \times FamMem_7 +$

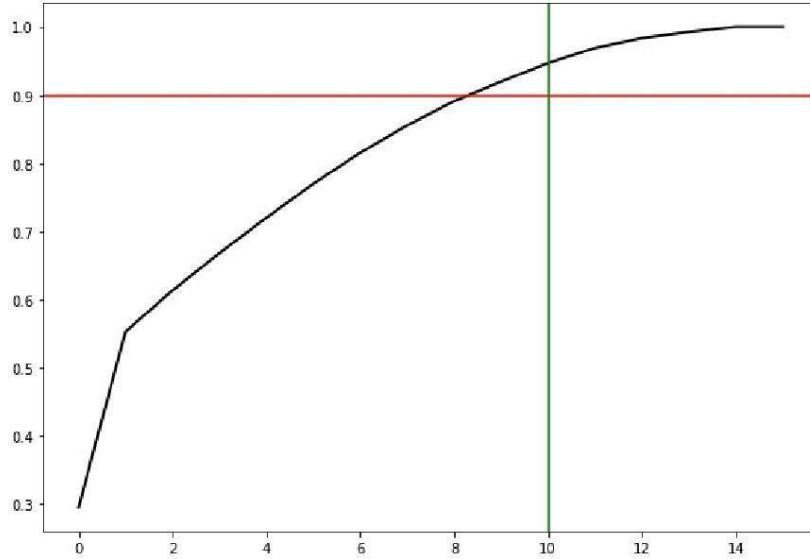
```

0.597 x FamMem_8 + 0.588 x FamMem_9
15 component: 0.75% of initial variance
0.008 x Employment_Type + 0.002 x GraduateOrNot + 0.028 x
ChronicDiseases + -0.020 x FrequentFlyer + 0.023 x EverTravelledAbroad
+ 0.008 x TravelInsurance + 0.005 x Age_std + -0.002 x
AnnualIncome_std + -0.000 x FamMem_2 + 0.001 x FamMem_3 + 0.002 x
FamMem_4 + 0.003 x FamMem_5 + 0.000 x FamMem_6 + 0.003 x FamMem_7 +
0.702 x FamMem_8 + -0.711 x FamMem_9
16 component: 0.0% of initial variance
-0.000 x Employment_Type + -0.000 x GraduateOrNot + 0.000 x
ChronicDiseases + -0.000 x FrequentFlyer + 0.000 x EverTravelledAbroad
+ -0.000 x TravelInsurance + -0.000 x Age_std + 0.000 x
AnnualIncome_std + 0.354 x FamMem_2 + 0.354 x FamMem_3 + 0.354 x
FamMem_4 + 0.354 x FamMem_5 + 0.354 x FamMem_6 + 0.354 x FamMem_7 +
0.354 x FamMem_8 + 0.354 x FamMem_9

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(10, c='g')

```

<matplotlib.lines.Line2D at 0x7f37e00ce4c0>



Для объяснения 90% дисперсии после применения метода РСА достаточно 4 компонент.

№ 15 Какой признак вносит наибольший вклад в первую компоненту?

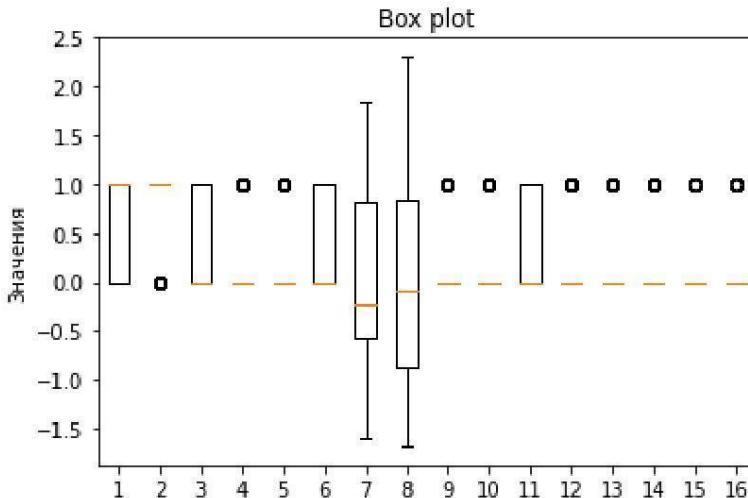
1 component: 29.48% of initial variance
0.174 x Employment_Type + 0.028 x GraduateOrNot + 0.000 x ChronicDiseases + 0.158 x FrequentFlyer + 0.205 x EverTravelledAbroad + 0.216 x TravelInsurance + -0.237 x Age_std + 0.894 x AnnualIncome_std + 0.001 x FamMem_2 + 0.002 x FamMem_3 + -0.007 x FamMem_4 + 0.015 x FamMem_5 + 0.007 x FamMem_6 + -0.009 x FamMem_7 + -0.006 x FamMem_8 + -0.003 x FamMem_9

Признак *AnnualIncome_std* вносит наибольший вклад в первую компоненту.

```
import matplotlib.pyplot as pltz

fig, ax = plt.subplots()
ax.boxplot(parameters)

ax.set_title('Box plot')
ax.set_ylabel('Значения')
plt.show()
```



№ 16 Построить двухмерное представление данных с помощью алгоритма t-SNE. На сколько кластеров визуально, на ваш взгляд, разделяется выборка? Объяснить смысл кластеров.

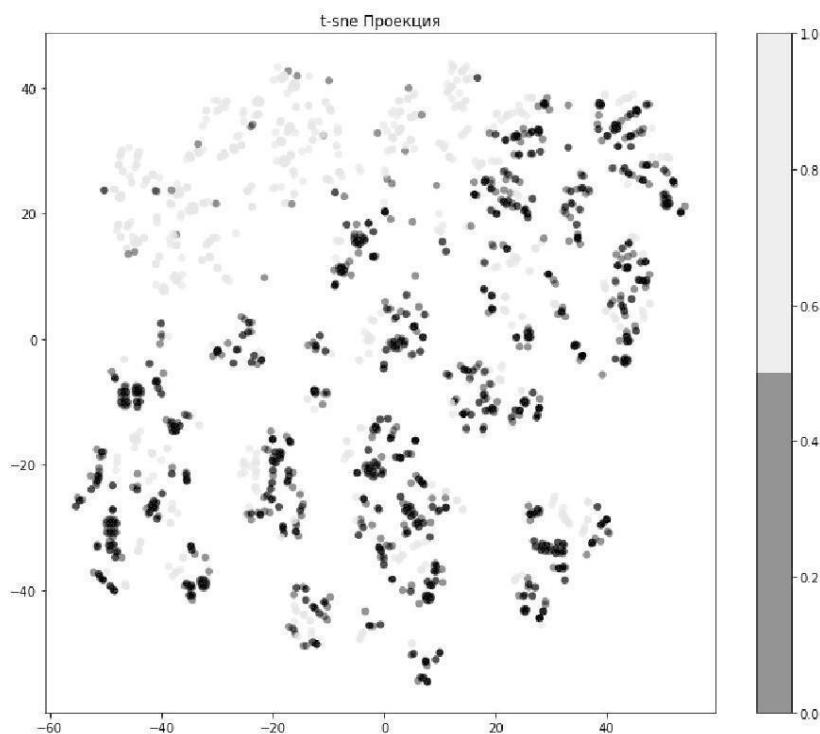
```
from sklearn.manifold import TSNE
tnse=TSNE(random_state=47)
X_tsne=tnse.fit_transform(parameters)
```

```

plt.figure(figsize=(12,10))
plt.scatter(X_tsne[:, 0], X_tsne[:, 1],
c=target, edgecolor='none', alpha=0.7,s=40,
cmap=plt.cm.get_cmap('nipy_spectral',2))
plt.colorbar()
plt.title('t-sne Проекция')
print(X_tsne)

[[ 23.742363      0.19087806]
 [-19.993422     -16.011976   ]
 [ 25.58573       35.545776   ]
 ...
 [-7.8768253     10.66755    ]
 [ 1.3286464     35.985943   ]
 [ 27.694685     33.40452   ]]

```



Выборка разделяется на 2 класса, верхний кластер показывает что объекты покупают страховку, а объекты нижнего кластера - не покупают.

Приложение 6.

```
Кластеризация
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
data =
pd.read_csv('/home/iv.kem/Downloads/archive/TravelInsurancePrediction.
csv', index_col = "Index")

scaler0 = StandardScaler()
data[['Employment Type', 'GraduateOrNot', 'FrequentFlyer',
'EverTravelledAbroad']] = pd.get_dummies(data[['Employment Type',
'GraduateOrNot', 'FrequentFlyer', 'EverTravelledAbroad']],
drop_first=True, dtype = int)
data[['Age', 'AnnualIncome']] =
pd.DataFrame(scaler0.fit_transform(data[['Age', 'AnnualIncome']])),
columns = data[['Age', 'AnnualIncome']].columns
one_hot = pd.get_dummies(data['FamilyMembers'],prefix='FamMem')

# Drop column B as it is now encoded
data = data.drop('FamilyMembers',axis = 1)
# Join the encoded df
data = data.join(one_hot)
data

      Age Employment Type GraduateOrNot AnnualIncome \
Index
0      0.463430          0           1     -1.414061
1      0.463430          1           1      0.842012
2      1.493446          1           1     -1.148641
3     -0.566587          1           1     -0.617800
4     -0.566587          1           1     -0.617800
...
1982   1.150107          1           1      1.505563
1983   -0.566587         1           1      2.169114
1984   -0.566587         1           1      0.576591
1985   1.493446          1           1      0.178461
1986   1.493446          1           1     -1.148641

      ChronicDiseases FrequentFlyer EverTravelledAbroad
TravelInsurance \
Index

0                  1          0           0
0
1                  0          0           0
0
2                  1          0           0
1
3                  1          0           0
0
```

4		1		1		0	
0							
...		
1982		0		1		1	
1							
1983		1		0		1	
0							
1984		1		0		0	
0							
1985		0		1		1	
1							
1986		0		0		0	
0							
	FamMem_2	FamMem_3	FamMem_4	FamMem_5	FamMem_6	FamMem_7	
	FamMem_8	\					
	Index						
0		0		0		1	0
0							
1		0		0		0	1
0							
2		0		1		0	0
0							
3		0		0		0	0
0							
4		0		0		0	0
1							
...	
...							
1982		0		1		0	0
0							
1983		0		0		1	0
0							
1984		0		0		0	1
0							
1985		0		0		0	1
0							
1986		0		1		0	0
0							
	FamMem_9						
	Index						
0		0					
1		0					
2		0					
3		0					
4		0					
...		...					

```

1982      0
1983      0
1984      0
1985      0
1986      0

[1987 rows x 16 columns]

for i in range(data.shape[1] + 1):
    pca2 = PCA(n_components = i)
    pca2.fit(data)
    print(i, sum(pca2.explained_variance_ratio_))

0 0
1 0.29466854545473675
2 0.5512037031435231
3 0.6128567222577291
4 0.6678911989422875
5 0.7204743660054207
6 0.7692441385686428
7 0.8163054165726863
8 0.8569887792070425
9 0.8919322654832452
10 0.9202943430778456
11 0.9468084073916279
12 0.9700359054620751
13 0.9839631105350368
14 0.9927298273367878
15 0.9999999999999998
16 0.9999999999999998

```

Для описания 90% дисперсии достаточно 10 компонент

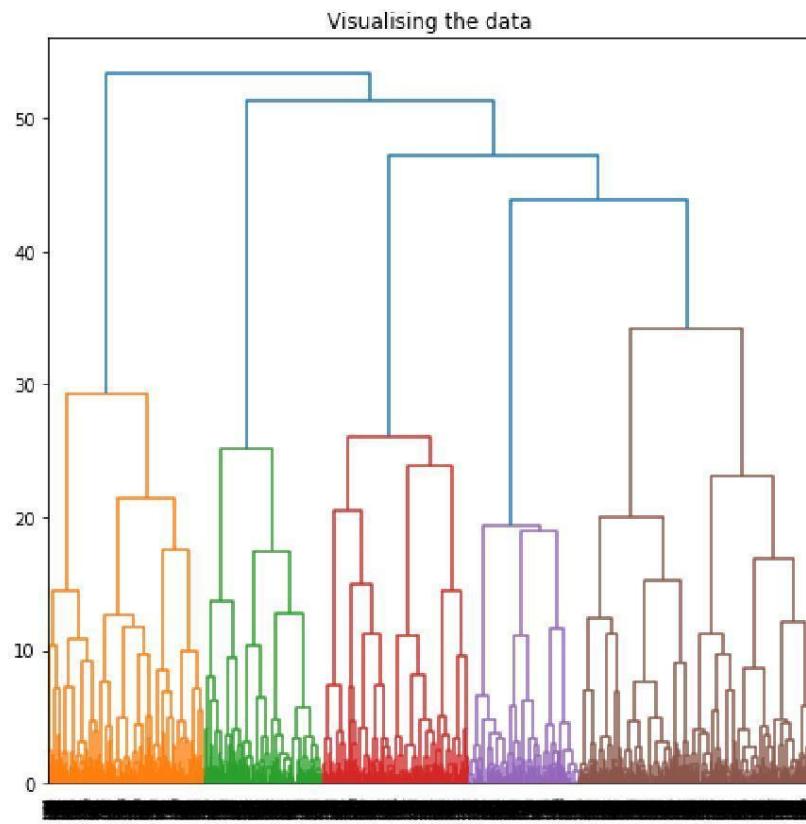
```

#AgglomerativeClustering (Иерархической кластеризация)

from sklearn.cluster import AgglomerativeClustering
from sklearn.metrics import silhouette_score
import scipy.cluster.hierarchy as shc
#снизим размерность данных при помощи метода главных компонент
pca = PCA(n_components=6).fit(data)
pdata = pca.transform(data)

#нормализуем данные
scaler = StandardScaler()
data2 = scaler.fit_transform(pdata)
plt.figure(figsize =(8, 8))
plt.title('Visualising the data')
Dendrogram = shc.dendrogram(shc.linkage(data2, method ='ward'))

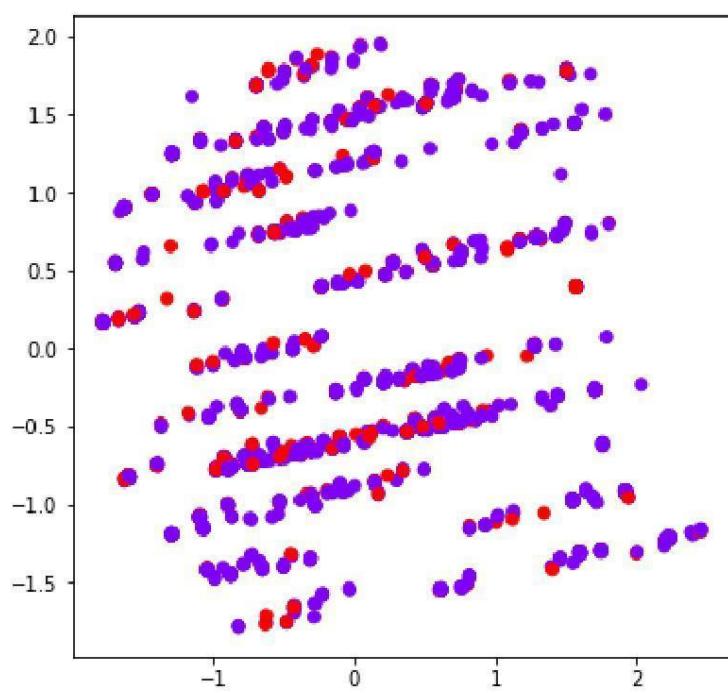
```

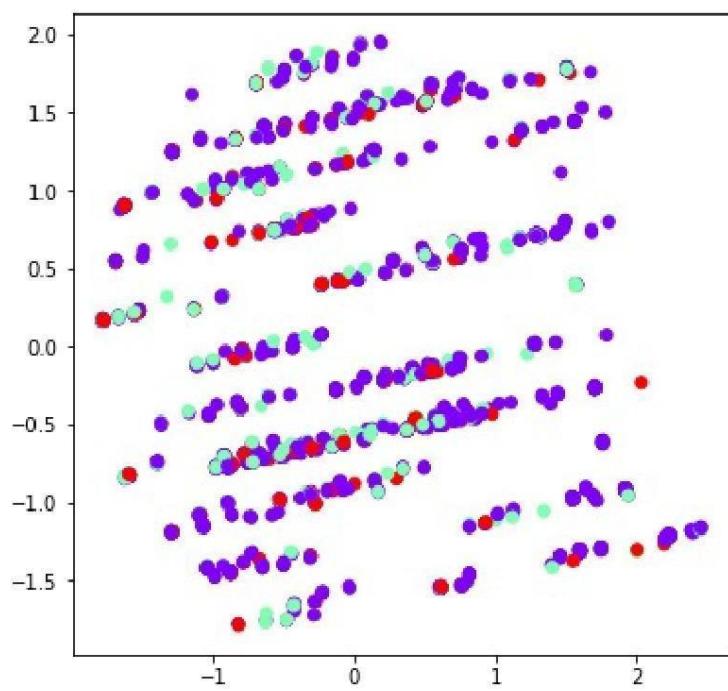


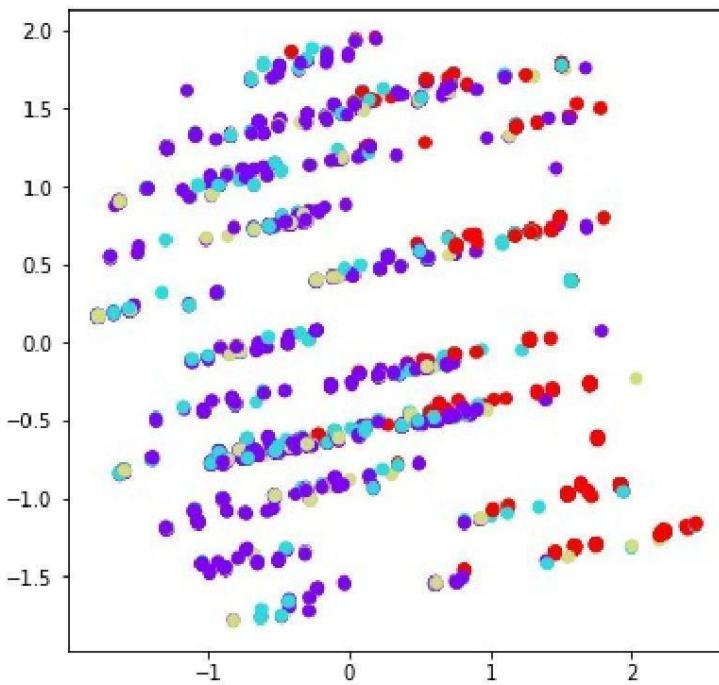
Пробуем отрисовать кластеры

```
for i in range(2,5):
    ac3 = AgglomerativeClustering(n_clusters = i)

    plt.figure(figsize =(6, 6))
    plt.scatter(data2[:,0], data2[:,1],
                c = ac3.fit_predict(data2), cmap ='rainbow')
    plt.show()
```







Можно сделать что чем больше кластеров давать алгоритму, тем менее разделимы они получаются на наших данных

DBSCAN

```
from sklearn.cluster import DBSCAN

#снизим размерность данных при помощи метода главных компонент
pca = PCA(n_components=6).fit(data)
pdata = pca.transform(data)

#нормализуем данные
scaler = StandardScaler()
data2 = scaler.fit_transform(pdata)

#используем dbscan для обнаружения кластеров
dbscan = DBSCAN(eps = 1.65, min_samples=5)
dbscan.fit(data2)
labels = dbscan.labels_

#используем tsne для снижения размерности
```

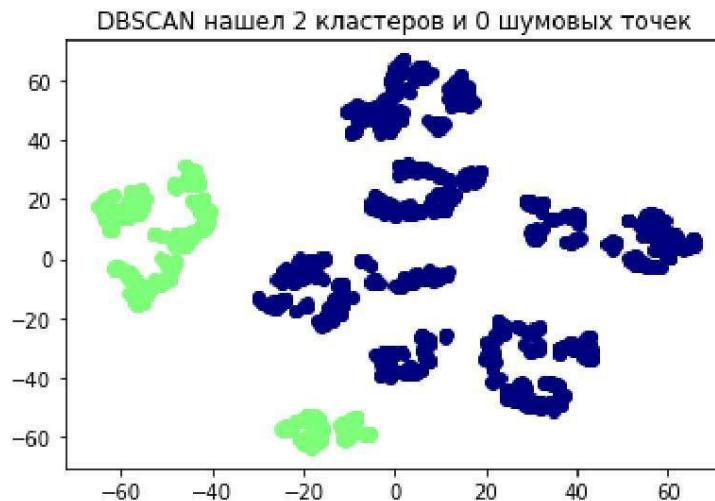
```

tsne = TSNE()
pca_2d = tsne.fit_transform(data2)

#визуализируем кластеры
for label in set(labels):
    if label == -1:
        plt.scatter(pca_2d[labels == label, 0], pca_2d[labels == label, 1], color='blue')
    else:
        plt.scatter(pca_2d[labels == label, 0], pca_2d[labels == label, 1], color=plt.cm.jet(label / np.max(labels + 1)))

plt.title('DBSCAN нашел {} кластеров и {} шумовых точек'.format(len(set(labels)) - (1 if -1 in labels else 0),
np.sum(labels == -1)))
plt.show()

```



K-means

```

pca = PCA(n_components=6).fit(data)
pdata = pca.transform(data)
scaler = StandardScaler()
data2 = scaler.fit_transform(pdata)

kmeans_kwargs = {
    "init": "random",
    "n_init": 10,
    "max_iter": 300,
}

```

```

    "random_state": 42,
}

# A list holds the SSE values for each k
sse = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, **kmeans_kwargs)
    kmeans.fit(data2)
    sse.append(kmeans.inertia_)

plt.style.use("fivethirtyeight")
plt.plot(range(1, 11), sse)
plt.xticks(range(1, 11))
plt.xlabel("Number of Clusters")
plt.ylabel("SSE")
plt.show()

```



Из граффика видно что оптимальное кол-во кластеров - 4

```

kmeans = KMeans(n_clusters=4, random_state = 157)
kmeans.fit(data2)
labels = kmeans.labels_

tsne = TSNE()
pca_2d = tsne.fit_transform(data2)

for label in set(labels):
    if label == -1:
        plt.scatter(pca_2d[labels == label, 0], pca_2d[labels ==

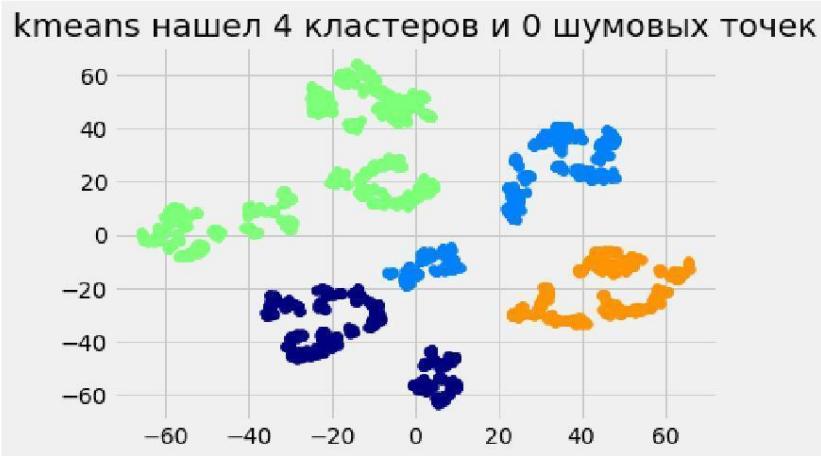
```

```

label, 1], color='blue')
else:
    plt.scatter(pca_2d[labels == label, 0], pca_2d[labels == label, 1], color=plt.cm.jet(label / np.max(labels + 1)))

plt.title('kmeans нашел {} кластеров и {} шумовых
точек'.format(len(set(labels)) - (1 if -1 in labels else 0),
np.sum(labels == -1)))
plt.show()

```



```

label = kmeans.fit_predict(data2)

silhouette_avg = silhouette_score(data2,label)
#print(silhouette_avg)
print(kmeans.cluster_centers_)

[[[-0.02338718  0.01653908  0.12747969 -0.38446642  1.40057273
  0.80884714]
 [ 0.03223941 -0.00598203 -0.97650456  1.29426444  0.19268617 -
 0.70149935]
 [-0.00717411  0.00362513 -0.33309664 -0.52959815 -0.73477452
 0.18495898]
 [ 0.00469648 -0.01965695  1.65835384  0.04514494 -0.32784704 -
 0.50751104]]]

original = scaler.inverse_transform(kmeans.cluster_centers_)
original_pca = pca.inverse_transform(original)
centr = pd.DataFrame(original_pca, columns=data.columns)
res2 = pd.DataFrame(scaler0.inverse_transform(centr[['Age',
'AnnualIncome',]]),
                     columns = ['Age', 'AnnualIncome'])

```

```

print(centr)

      Age Employment Type GraduateOrNot AnnualIncome
ChronicDiseases \
0 0.007864          0.629113    0.912942   -0.001067
0.972436
1 -0.011303         0.685189    0.889088   0.055563
0.271174
2 0.006592          0.756882    0.796469   -0.043795
0.030614
3 -0.009633         0.752306    0.850673   0.027383
0.008222

      FrequentFlyer EverTravelledAbroad TravelInsurance FamMem_2
FamMem_3 \
0 0.071783           0.210861     0.403489  0.052937
0.236327
1 0.309669           0.158524     0.233949  0.050433 -
0.110043
2 0.239356           0.219074     0.437767  0.047907
0.458790
3 0.193985           0.149017     0.280502  0.033298 -
0.072420

      FamMem_4 FamMem_5 FamMem_6 FamMem_7 FamMem_8 FamMem_9
0 0.316191 0.017050 0.197313 0.117420 0.021395 0.041367
1 -0.077586 0.891660 0.115490 0.078440 0.027740 0.023867
2 0.064806 0.078950 0.185096 0.099687 0.037007 0.027757
3 0.952398 -0.057424 0.052043 0.049508 0.026487 0.016110

```

Кластеры:

1. Люди среднего возраста, которые работают в частном секторе, учились в колледже, имеют чуть ниже среднего заработок, количество детей - среднее. У них есть хроническое заболевание. Покупают авиаилеты и путешествуют за рубеж редко, кто-то покупал страховой пакет, а кто-то нет
2. Люди среднего возраста, которые работают в частном секторе, учились в колледже, имеют средний заработок , количество детей - среднее. У них нет болезней. Покупают авиаилеты и путешествуют за рубеж редко, не покупают страховой пакет.
3. Люди немного старше среднего, которые работают в государственном секторе, учились в колледже, имеют заработок ниже среднего, количество детей - среднее. У них нет болезней. Покупают авиаилеты и путешествуют за рубеж редко, большинство не купило страховой пакет.

- Люди среднего возраста, которые работают в частном секторе, учились в колледже, имеют заработок выше среднего, количество детей - среднее. У них нет болезней. Покупают авиаилеты и путешествуют за рубеж чаще обычного (больше половины да чем нет), почти все купили страховой пакет.

Классификация на основе кластеров лучшего алгоритма (K-means)

```

from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn import tree
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import f1_score, confusion_matrix,
classification_report
from sklearn.model_selection import train_test_split
from sklearn import preprocessing

data['cluster'] = labels

cluster_0 = data[data['cluster'] == 0]
cluster_1 = data[data['cluster'] == 1]
cluster_2 = data[data['cluster'] == 2]
cluster_3 = data[data['cluster'] == 3]

data_sel = cluster_0

TravelInsurance = data_sel.loc[:, 
data_sel.columns.isin(['TravelInsurance'])]
data_sel = data_sel.drop('TravelInsurance', axis = 1).drop('cluster',
axis = 1)

x_train, x_test, y_train, y_test = train_test_split(data_sel,
TravelInsurance, test_size= 0.3, random_state=4477)

for i in range(1, 10):
    T = DecisionTreeClassifier(random_state=338, max_depth = i)
    T = T.fit(x_train, y_train)
    print(str(i) + ":" + str(T.score(x_test, y_test)))

1: 0.7364341085271318
2: 0.7674418604651163
3: 0.7751937984496124
4: 0.7906976744186046
5: 0.7984496124031008
6: 0.7984496124031008

```

```
7: 0.8062015503875969  
8: 0.7829457364341085  
9: 0.7441860465116279
```

Лучший результат дает дерево с глубиной 4

```
T = DecisionTreeClassifier(random_state=338, max_depth = 4)  
T = T.fit(x_train, y_train)
```

```
pred = T.predict(x_test)  
report = classification_report(y_test, pred)  
print(report, T.score(x_test, y_test))
```

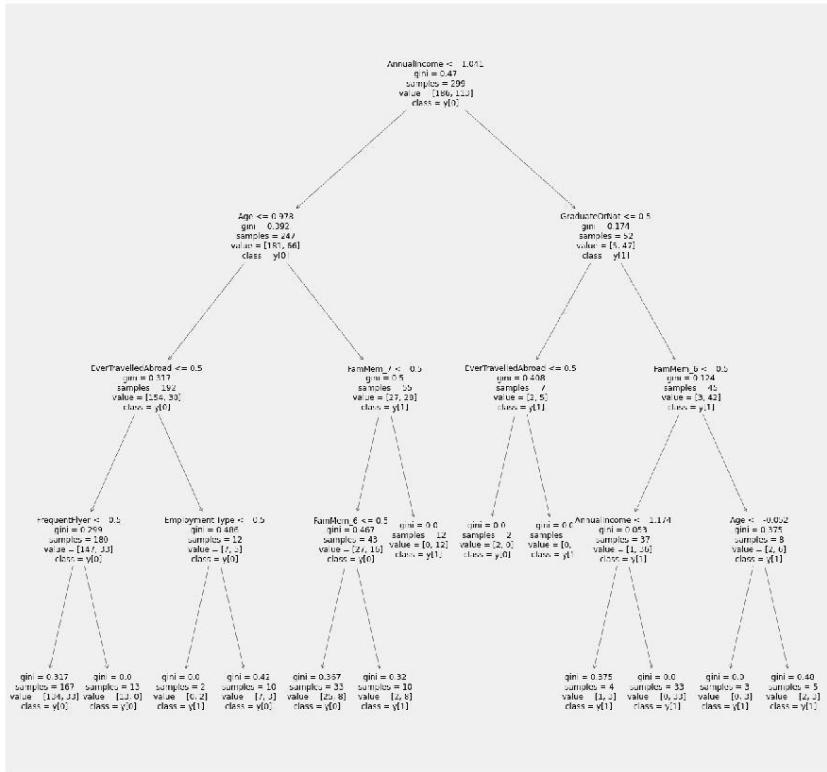
	precision	recall	f1-score	support
0	0.76	0.95	0.85	78
1	0.88	0.55	0.67	51
accuracy			0.79	129
macro avg	0.82	0.75	0.76	129
weighted avg	0.81	0.79	0.78	129
	0.7906976744186046			

Получили модель с правильностью порядка 76% по метрике f1

```
importances = T.feature_importances_  
print(importances)  
  
[0.1382899 0.02543763 0.00905349 0.54925611 0. 0.01466946  
 0.06357554 0. 0. 0. 0. 0.08450275  
 0.11521512 0. 0. ]
```

Самым важным признаком оказался - AnnualIncome

```
plt.figure(figsize=(20,20))  
tree.plot_tree(T,max_depth=4,fontsize=12,class_names=True,feature_name  
s=x_test.columns)  
plt.show()
```



Опишем характеристики некоторых объектов класса 1. $AnnualIncome >= 1.041$, $GraduateOrNot >= 0.5$, Age в окрестности -0.052, $EverTravelledAbroad >= 0.5$.

Опишем характеристики некоторых объектов класса 0. $AnnualIncome <= 1.041$, $Age <= 0.978$ or $Age >= 0.978$, $EverTravelledAbroad <= 0.5$, $FamMem_6 <= 0.5$.

Попробуем RandomForestClassifier + GridSearch

```
RF = RandomForestClassifier()
grid_parametrs = {
    'n_estimators':[20,50,100, 150, 200,250,300,350,400,450,500],
    'max_depth':[1,3,5,9,10,12,15,17]
}
grid = GridSearchCV(RF,param_grid=grid_parametrs,cv=3,scoring='f1')
model_grid = grid.fit(x_train, y_train.values.ravel())
```

```

print('Лучшие гиперпараметры: ' + str(model_grid.best_params_))
print('Best score is: ' + str(model_grid.best_score_))

Лучшие гиперпараметры:{'max_depth': 9, 'n_estimators': 250}
Best score is: 0.7005622871624296

grid_params = {
    'n_estimators':[200, 210, 220, 230, 240, 250, 260, 270, 280, 290, 300],
    'max_depth':[9]
}

grid = GridSearchCV(RF, param_grid=grid_params, cv=3, scoring='f1')
model_grid = grid.fit(x_train, y_train.values.ravel())

print('Лучшие гиперпараметры: ' + str(model_grid.best_params_))
pred=grid.predict(x_test)

report = classification_report(y_test.values.ravel(), pred)
print(report)
F1_score = f1_score(y_test.values.ravel(), pred, average='macro')
print("F1-score : ", F1_score)

Лучшие гиперпараметры:{'max_depth': 9, 'n_estimators': 230}
precision      recall   f1-score   support
0            0.77      0.94      0.84      78
1            0.85      0.57      0.68      51

accuracy                      0.79      129
macro avg          0.81      0.75      0.76      129
weighted avg        0.80      0.79      0.78      129

F1-score : 0.7631417885073104

```

Лучший F1 = 76%. Это также меньше у дерева решений. Попробуем GradientBoostingClassifier

```

from numpy import mean
from numpy import std
from sklearn.datasets import make_classification
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedStratifiedKFold
from matplotlib import pyplot

# evaluate the model
model = GradientBoostingClassifier()
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
n_scores = cross_val_score(model, x_train, y_train, scoring='f1',

```

```

cv=cv, n_jobs=-1, error_score='raise')
print('F1: %.3f (%.3f)' % (mean(n_scores), std(n_scores)))
# fit the model on the whole dataset
model = GradientBoostingClassifier()
model.fit(x_train, y_train.values.ravel())

pred =model.predict(x_test)

report = classification_report(y_test.values.ravel(), pred)
print(report)
F1_score = f1_score(y_test.values.ravel(), pred, average='macro')
print("F1-score : ", F1_score)

F1: 0.653 (0.116)
      precision    recall  f1-score   support
0       0.76     0.92    0.83      78
1       0.82     0.55    0.66      51

accuracy                           0.78      129
macro avg       0.79     0.74    0.75      129
weighted avg    0.78     0.78    0.76      129

F1-score : 0.7455967358041483

```

Получили F1 = 74%, все еще меньше чем у дерева решений (F1=76%), оставляем дерево решений в качестве лучшей модели

Таким образом можно сделать вывод что страховку с большей вероятностью купят люди среднего возраста, которые работают в частном секторе, учились в колледже, имеют заработок выше среднего, количество детей - среднее. У них нет болезней. Покупают авиаперелеты и путешествуют за рубеж чаще обычного (больше половины да чем нет).