Национальный исследовательский университет «МЭИ»

Кафедра управления и интеллектуальных технологий

**Лабораторная работа №3**

# СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ КЛАССИФИКАЦИИ

# ДАННЫХ

Выполнила студентка:

Андриянова К.Б. из группы А-01-20

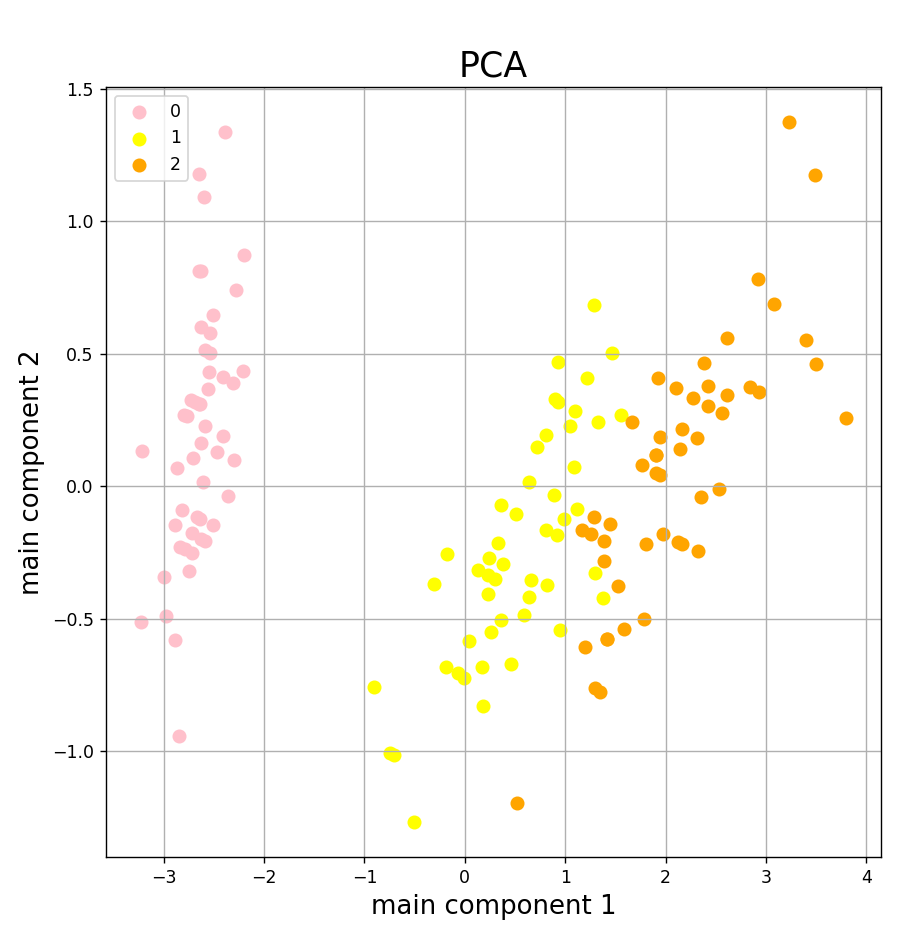
Москва 2022

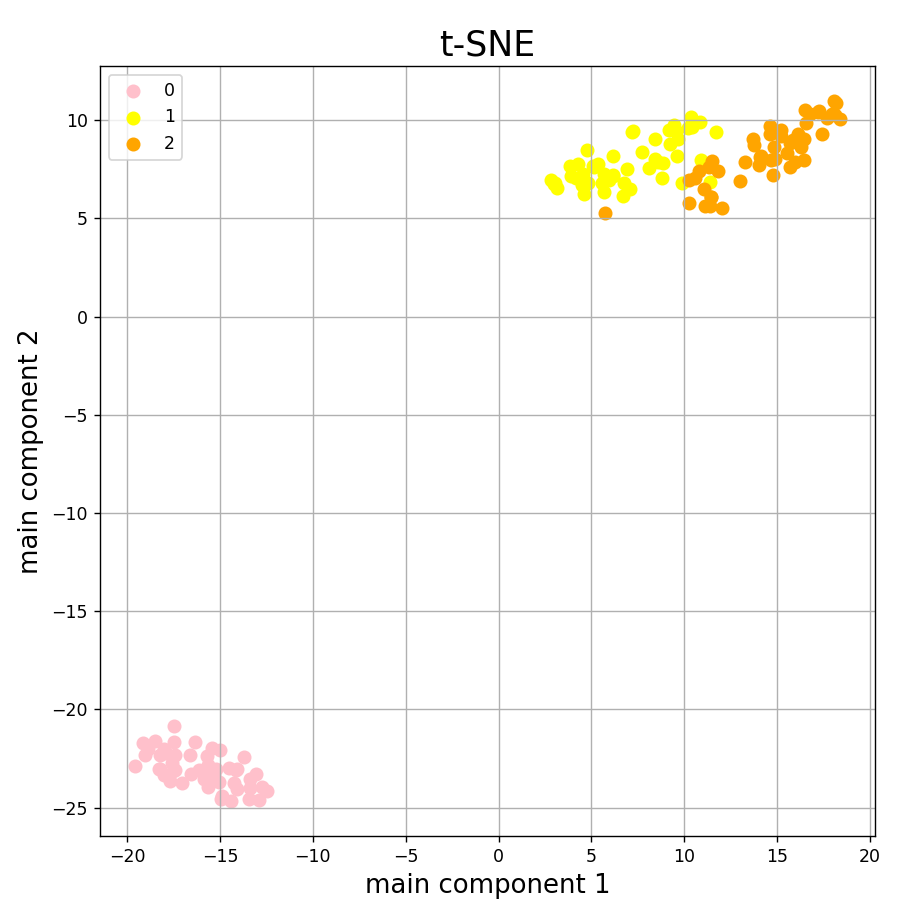
**Описание датасетов**

**1.** Набор данных ***«Iris Data Set»* – *«Ирисы Фишера»*** (см. Л.Р.№1)***.***

**Задание на лабораторную работу**

1. Провести снижение размерности методами МГК (PCA) и t-sne. Сделать предположение о разделимости выборки на классы. В отчёт включить все графики и вывод о разделимости.





Как видно с помощью метода t-sne выбока разбилась точнее, и можнго выделить три кластера. Выборка хорошо поддается классификации.

1. Провести классификацию (с параметрами по умолчанию) (Классификаторы назначаются преподавателем). Проводить обучения классификатора следует с использованием кросс-валидации. Так же требуется разделить выборку на тестовую и обучающую. Так же требуется получить данные о качестве классификации (*Accuracy, Precision, Recall и F1-мера*). После обучения требуется визуализировать тестовую выборку:
2. С исходными метками
3. С метками полученными при классификации

В отчёт включить оба графика. Сделать выводы о качестве классификации.

**KNeighborsClassifier**

(KNeighborsClassifier(),

{'fit\_time': array([0.00300097, 0.00200248, 0.00200033, 0.00200272]),

'score\_time': array([0.00700068, 0.00500607, 0.00399733, 0.0040009 ]),

'test\_accuracy': array([1. , 0.96666667, 0.96666667, 0.93333333]),

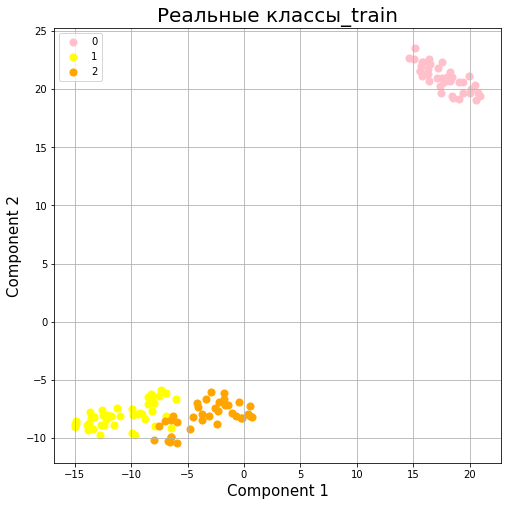
'test\_precision\_weighted': array([1. , 0.97 , 0.97 , 0.93333333]),

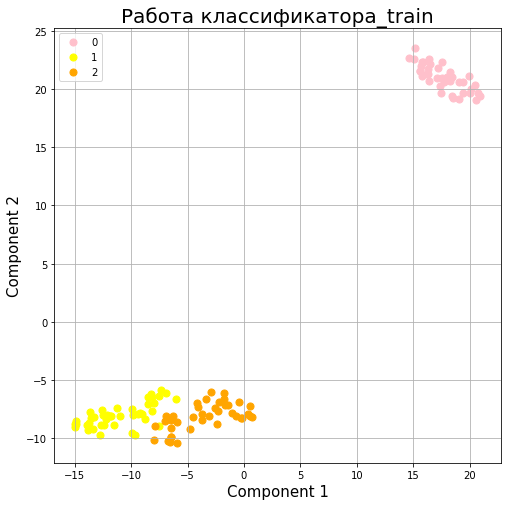
'test\_recall\_weighted': array([1. , 0.96666667, 0.96666667, 0.93333333]),

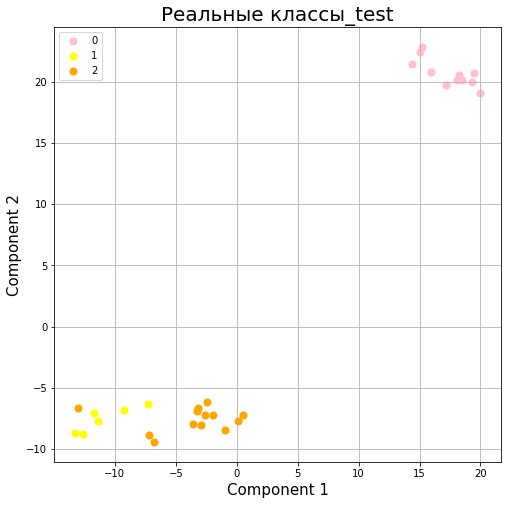
'test\_f1\_weighted': array([1. , 0.96675021, 0.96675021, 0.93333333])},

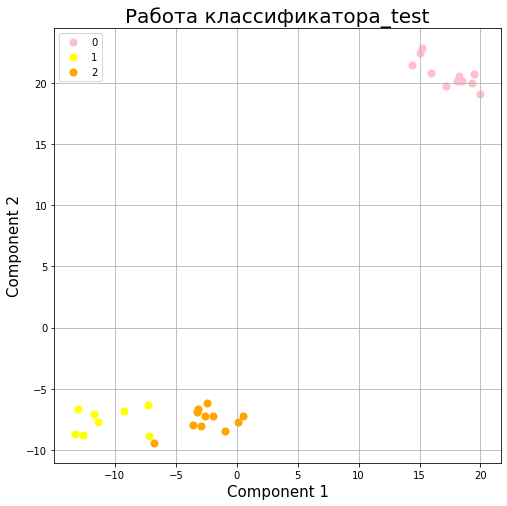
[0.9333333333333333, 0.95, 0.9333333333333333, 0.9353174603174603])

| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9333333333333333 | 0.95 | 0.9333333333333333 | 0.9353174603174603 |









**LogisticRegression**

(LogisticRegression(),

{'fit\_time': array([0.03801036, 0.01300144, 0.01200271, 0.01100183]),

'score\_time': array([0.00300312, 0.00300241, 0.00299954, 0.00300121]),

'test\_accuracy': array([1. , 1. , 1. , 0.93333333]),

'test\_precision\_weighted': array([1. , 1. , 1. , 0.93333333]),

'test\_recall\_weighted': array([1. , 1. , 1. , 0.93333333]),

'test\_f1\_weighted': array([1. , 1. , 1. , 0.93333333])},

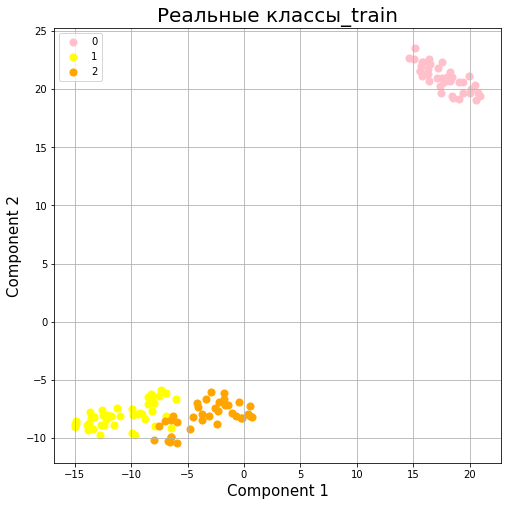
[0.9666666666666667,

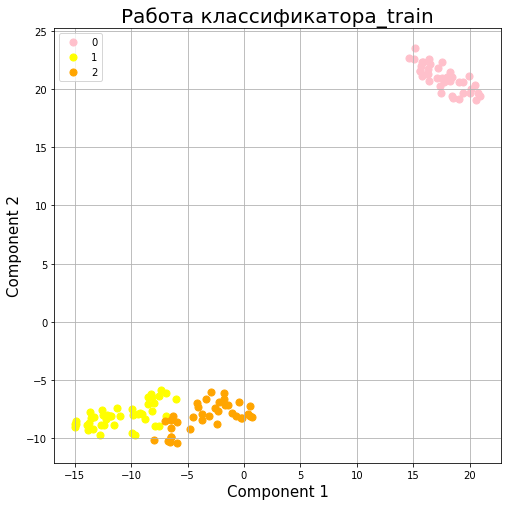
0.9714285714285714,

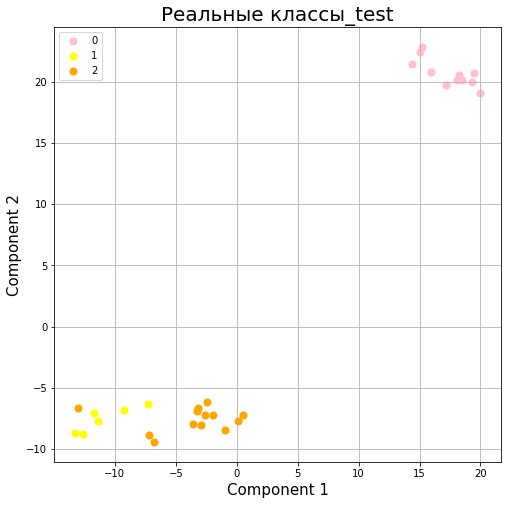
0.9666666666666667,

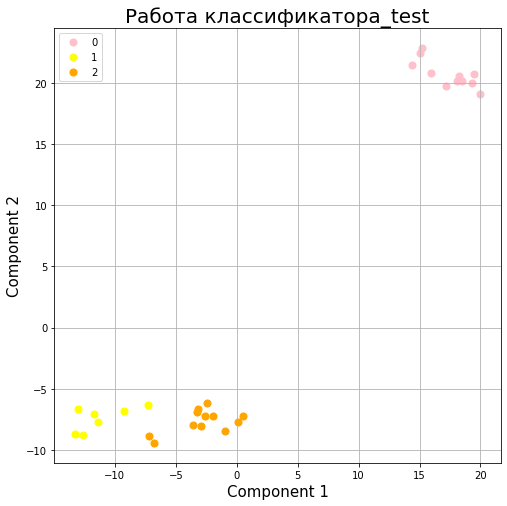
0.9672820512820514])

| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9666666666666667 | 0.9714285714285714 | 0.9666666666666667 | 0.9672820512820514 |









**RandomForestClassifier**

(RandomForestClassifier(),

{'fit\_time': array([0.11242604, 0.09302044, 0.09402895, 0.09101081]),

'score\_time': array([0.01000309, 0.01000214, 0.00900173, 0.01100707]),

'test\_accuracy': array([0.93333333, 0.93333333, 0.93333333, 0.9 ]),

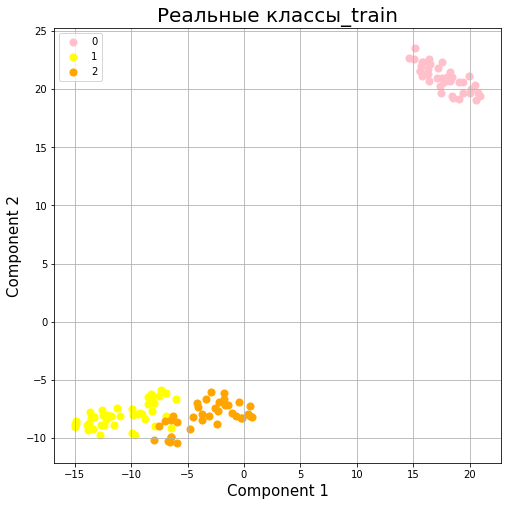
'test\_precision\_weighted': array([0.93333333, 0.94545455, 0.93333333, 0.90185185]),

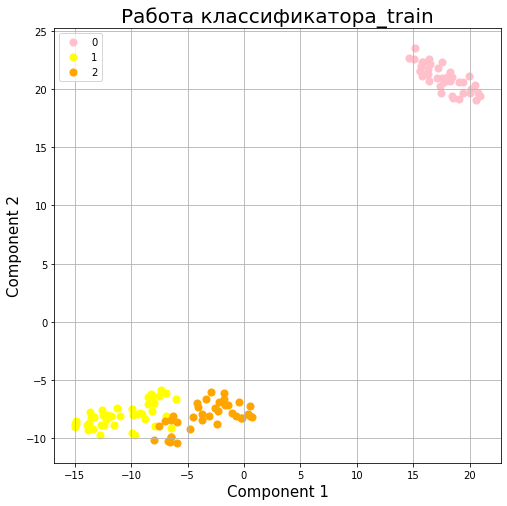
'test\_recall\_weighted': array([0.93333333, 0.93333333, 0.93333333, 0.9 ]),

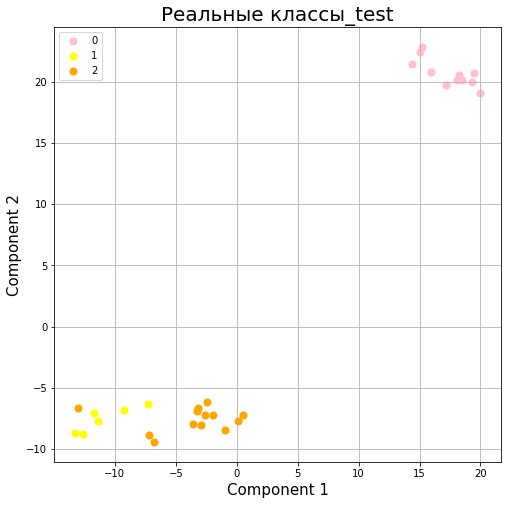
'test\_f1\_weighted': array([0.93333333, 0.93333333, 0.93333333, 0.89954233])},

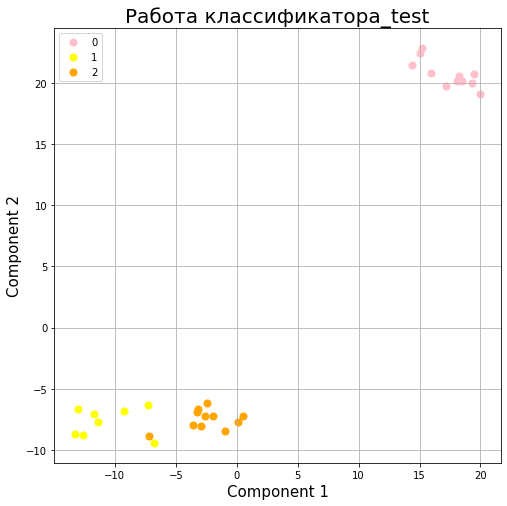
[0.9333333333333333, 0.95, 0.9333333333333333, 0.9353174603174603])

| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9333333333333333 | 0.95 | 0.9333333333333333 | 0.9353174603174603 |









**DecisionTreeClassifier**

(DecisionTreeClassifier(),

{'fit\_time': array([0.00199914, 0.00200558, 0.00200653, 0.00300598]),

'score\_time': array([0.00500345, 0.00300574, 0.00299406, 0.00299549]),

'test\_accuracy': array([0.93333333, 0.83333333, 0.93333333, 0.9 ]),

'test\_precision\_weighted': array([0.93333333, 0.83666667, 0.93333333, 0.90185185]),

'test\_recall\_weighted': array([0.93333333, 0.83333333, 0.93333333, 0.9 ]),

'test\_f1\_weighted': array([0.93333333, 0.83375104, 0.93333333, 0.89954233])},

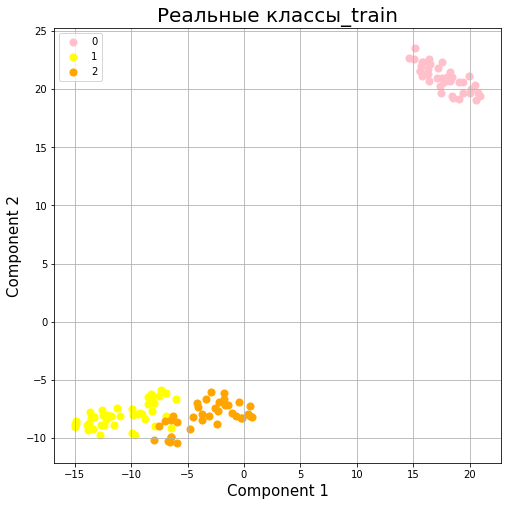
[0.9666666666666667,

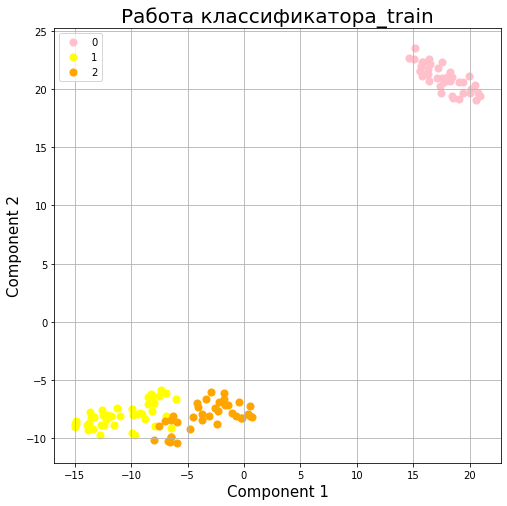
0.9714285714285714,

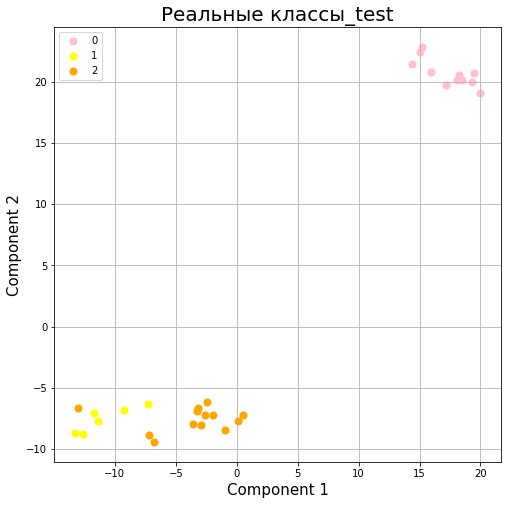
0.9666666666666667,

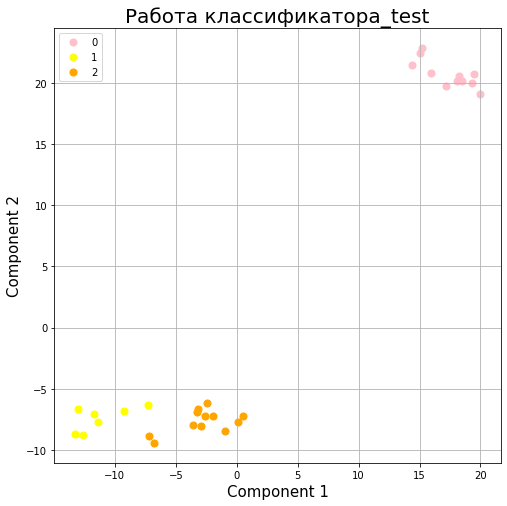
0.9672820512820514])

| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9666666666666667 | 0.9714285714285714 | 0.9666666666666667 | 0.9672820512820514 |









Обобщая результаты полученные ниже можно сказать, что все методы дают очень точные результаты. Лучшими можно назвать классификацию Логистической Регрессией и деревом решений, чуть менее эффективными метод К-соседей и метод случайного леса.

1. Провести настройку параметров классификаторов с использованием gridsearchcv. Результаты настройки сохранить для следующего пункта. (При защите будет уделяться особое внимание параметрам классификации (обязательно знать, что каждый параметр обозначает) и принципу их обучения и работы)

KNeighborsClassifier()

{'algorithm': 'auto', 'metric': 'minkowski', 'n\_neighbors': 4, 'p': 1, 'weights': 'uniform'}

Algorithm – Алгоритм для расчета ближайшего соседа

Metric - Мера близости

n\_neighbors – Количество соседей

weights – Весовая функция используемая для прогноза

LogisticRegression()

{'C': 0.1, 'penalty': 'l2'}

penalty - нормa штрафа

C - обратная сила регуляризации; меньшие значения указывают на более сильную регуляризацию

DecisionTreeClassifier()

{'criterion': 'gini', 'min\_samples\_split': 5, 'splitter': 'random'}

сriterion - Функция для измерения качества разделения

min\_samples\_split - минимальное число образцов для сплита

splitter - минимальное число образцов в листах.

RandomForestClassifier()

{'max\_depth': 20, 'n\_estimators': 201}

n\_estimators - число деревьев в лесу

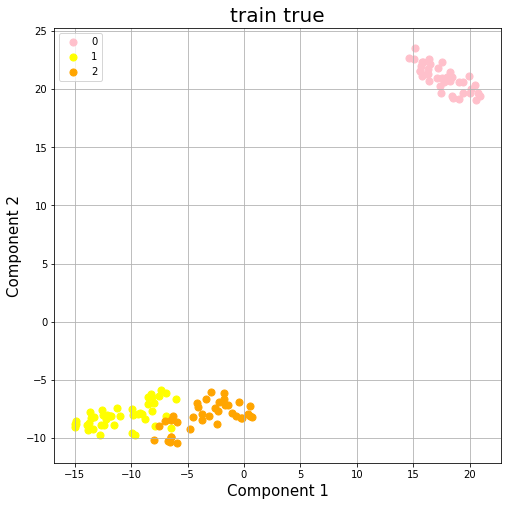
max\_depth - глубина дерева

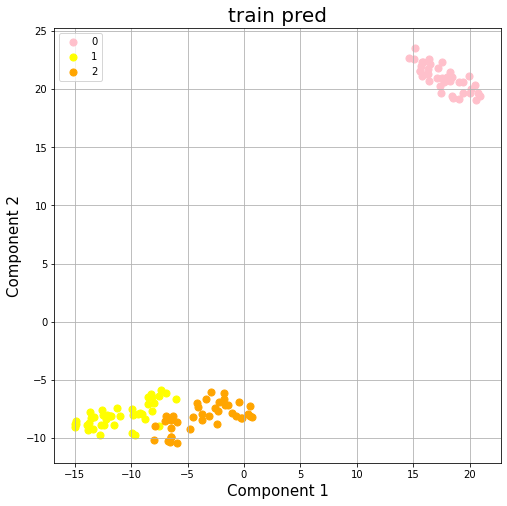
1. Провести классификацию (с параметрами из предыдущего пункта). После обучения требуется визуализировать тестовую выборку:
2. С исходными метками
3. С метками полученными при классификации (пункт 2)
4. С метками полученными при классификации

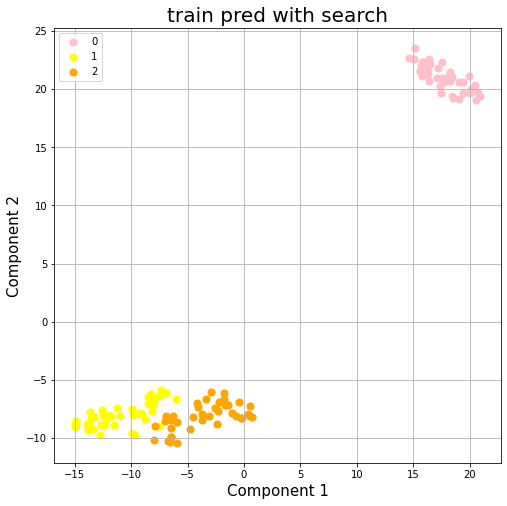
В отчёт включить все графики. Сделать выводы о качестве классификации. Сравнить с результатами 2 пункта.

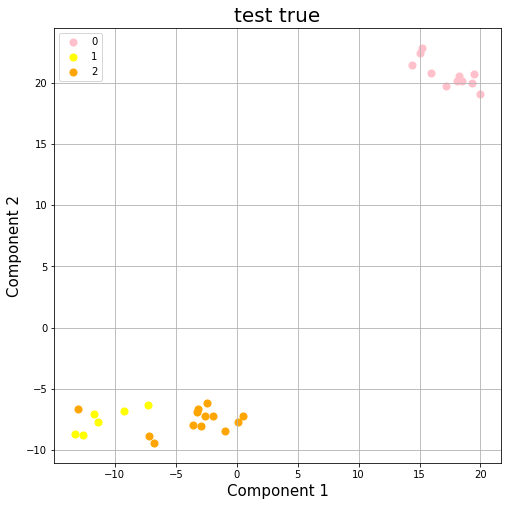
**KNeighborsClassifier()**

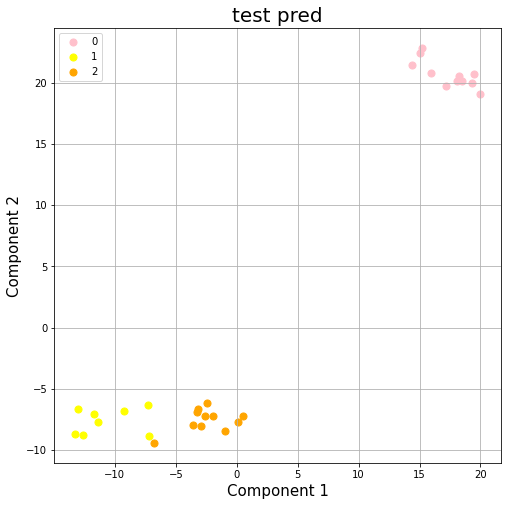
| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9333333333333333 | 0.95 | 0.9666666666666667 | 0.9353174603174603 |

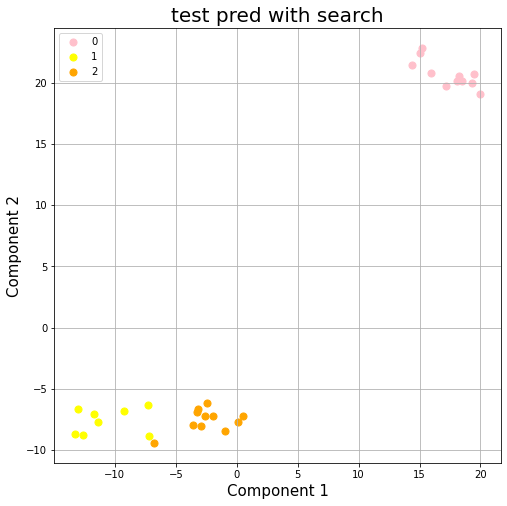






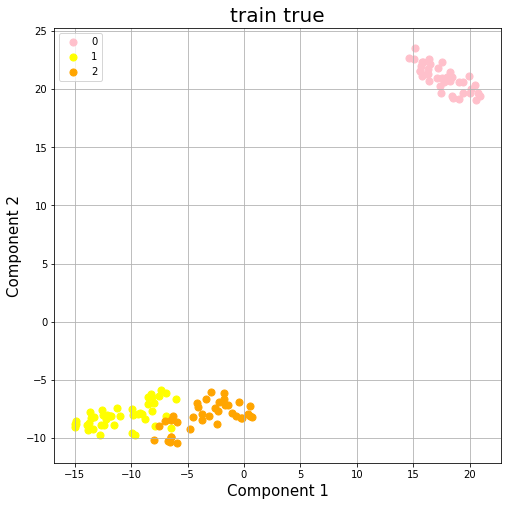


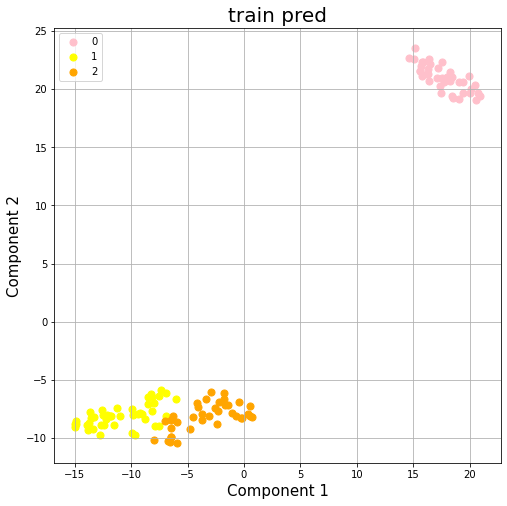


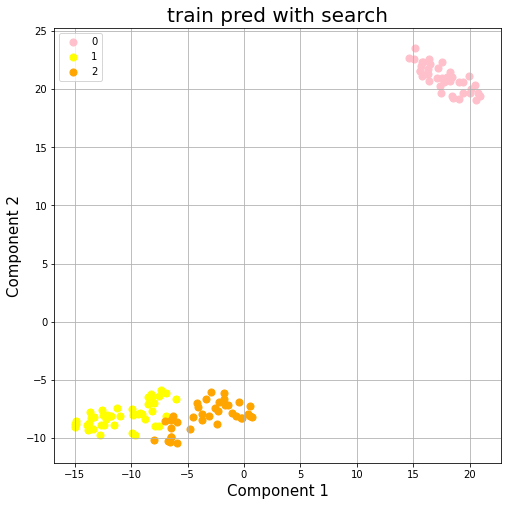


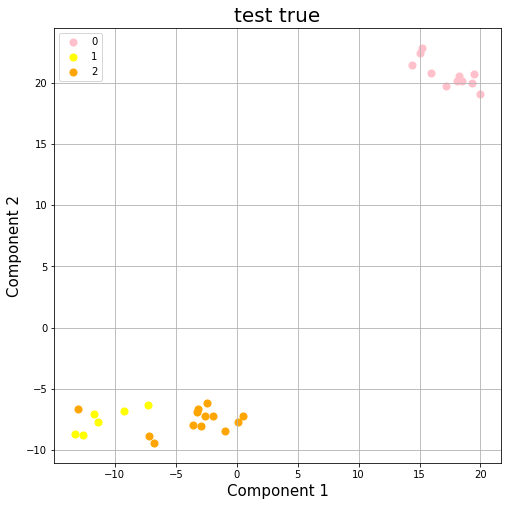
**LogisticRegression**

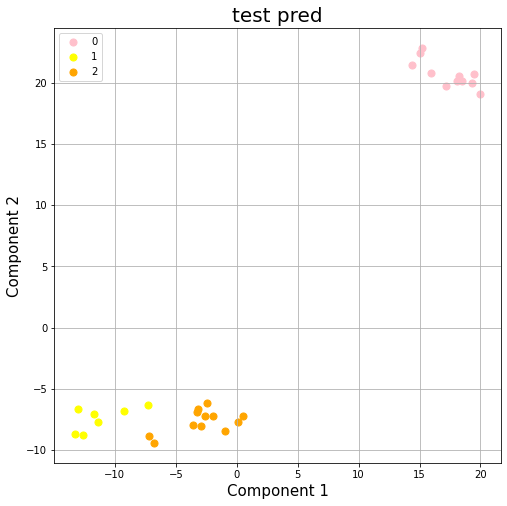
| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9666666666666667 | 0.9714285714285714 | 0.9666666666666667 | 0.9672820512820514 |

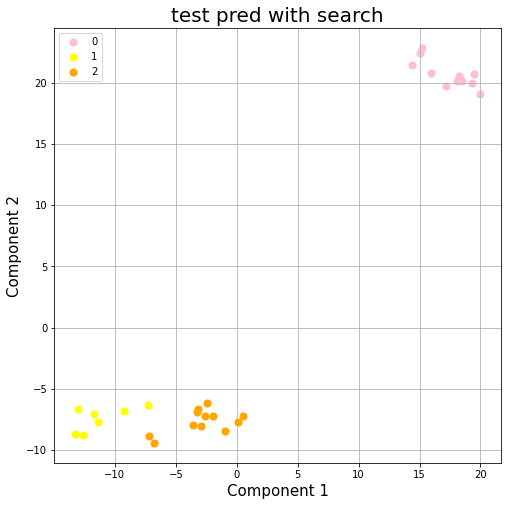






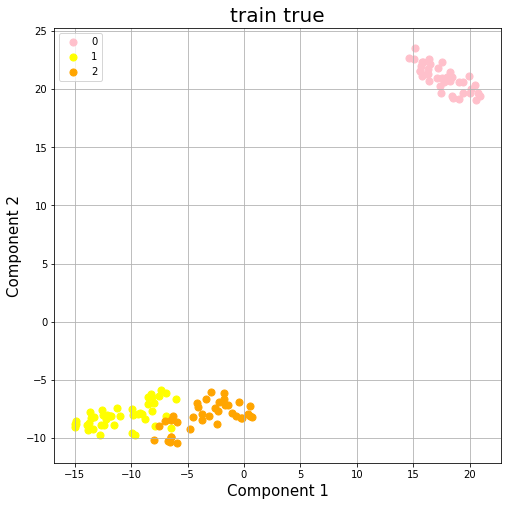


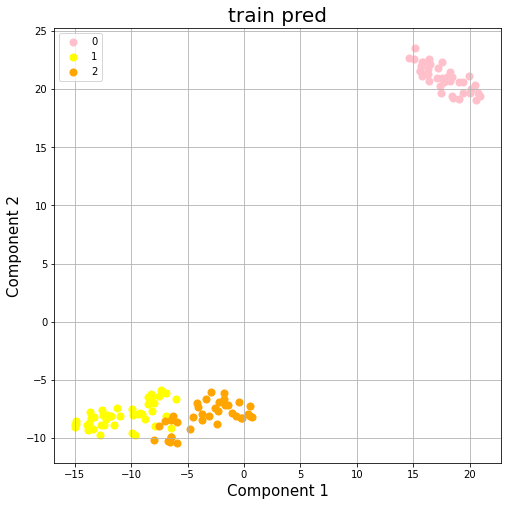


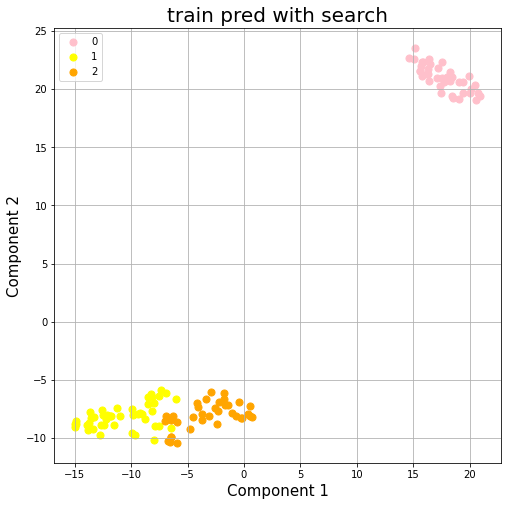


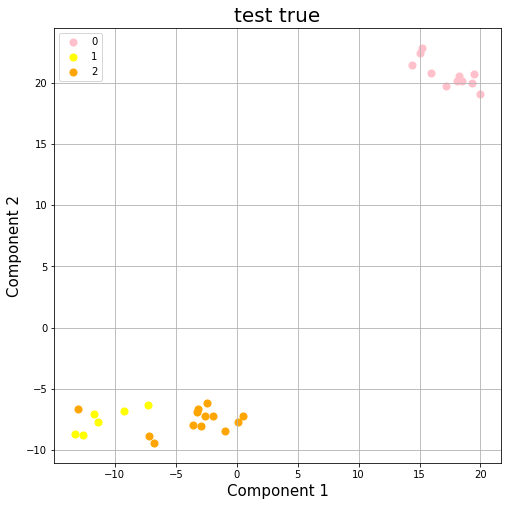
**DecisionTreeClassifier**

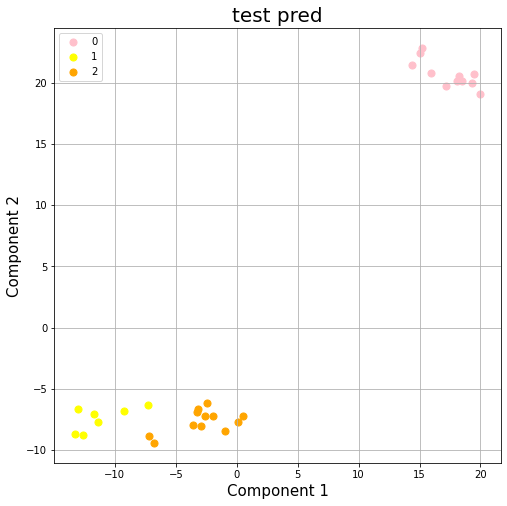
| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9666666666666667 | 0.9714285714285714 | 0.9666666666666667 | 0.9672820512820514 |

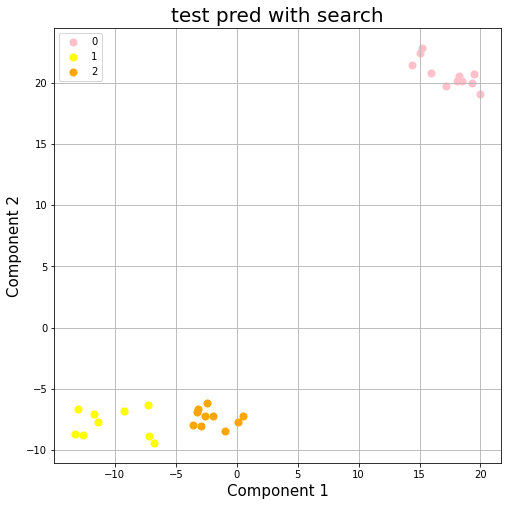






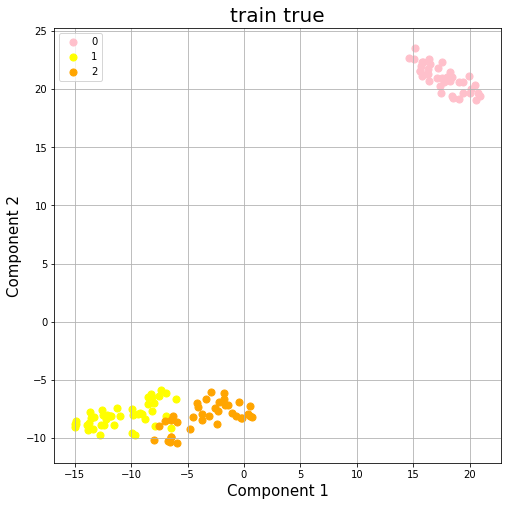


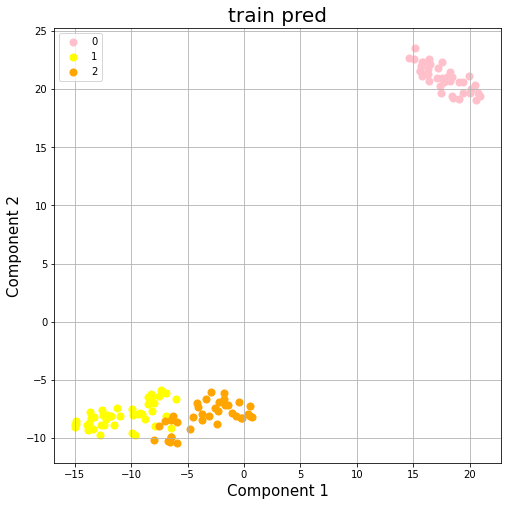


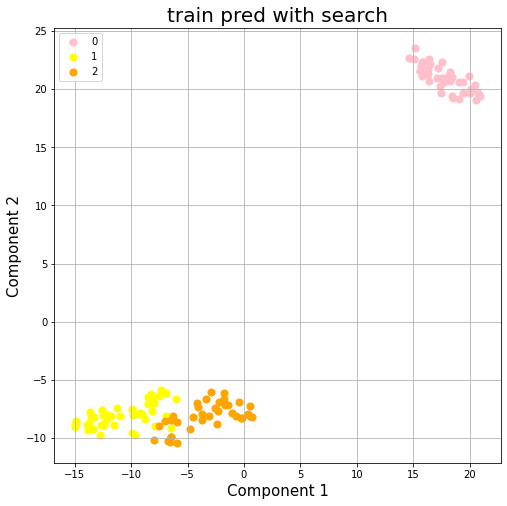


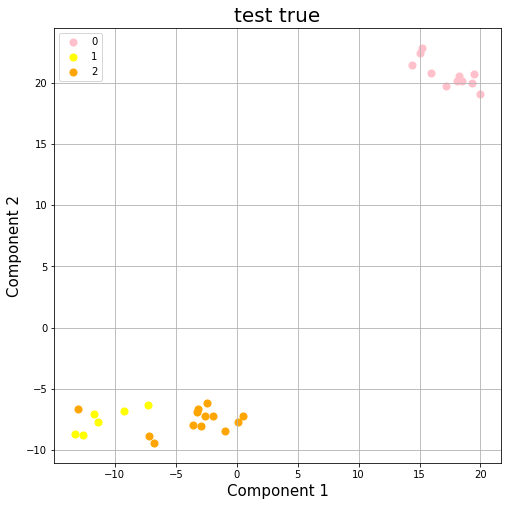
**RandomForestClassifier**

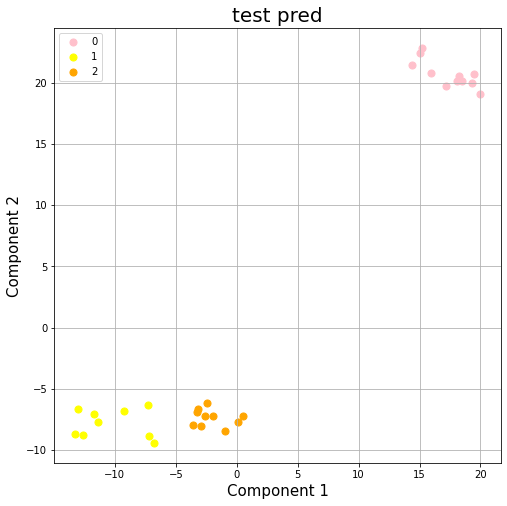
| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9666666666666667 | 0.9714285714285714 | 0.9666666666666667 | 0.9672820512820514 |

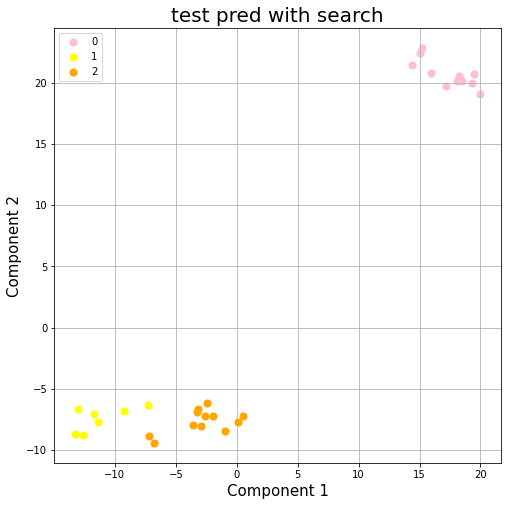






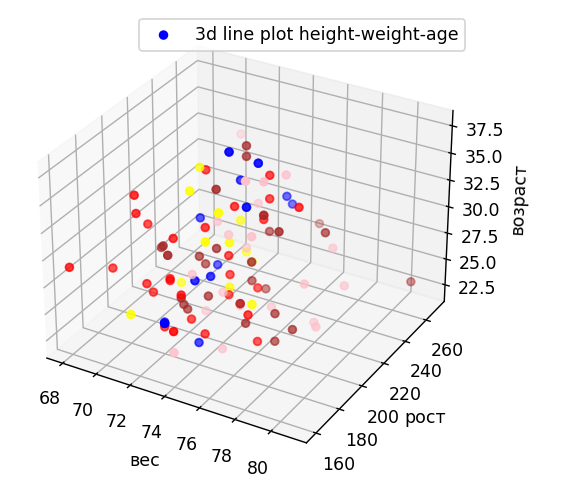






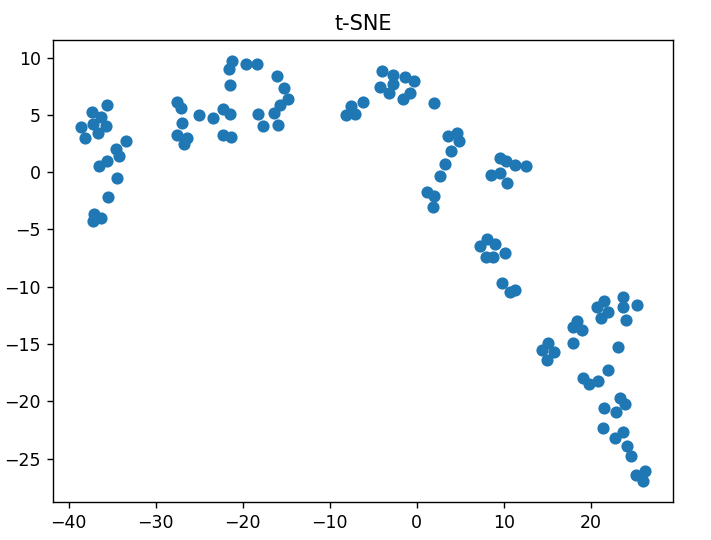
1. Исследовать возможность проведения классификации данных, содержащихся в датасетах: (Датасет назначается преподавателем)

- [Рост-Вес-Возраст-Позиция.sta‎](https://legacy.mpei.ru/owa/attachment.ashx?attach=1&id=RgAAAADa%2bjmVJSDcSLYnQ2XrAEheBwB608uvfhtKTpwU485LJbvQAKwvITQOAAB608uvfhtKTpwU485LJbvQAkno55DCAAAJ&attid0=EABKehY9ntLiTpJ1qqTwRifJ&attcnt=1)



Визуально выделить классы в этой выборке невозможно, они все перемешаны.

Снизили размерность до двух с помощью метода t-sne. Визуально можно выделить несколько кластеров, но точное их количество определить нельзя.



Проведем классификацию с помощью Случайного Леса.

| Precision | Recall | Accuracy | F1 - score |
| --- | --- | --- | --- |
| 0.9597919583916783 | 0.959758754259508 | 0.9597919583916783 | 0.9597219992697565 |

Коэффиценты качества оказались достаточно высокими, следовательно, классификатор хорошо обучен.