

Московский авиационный институт
(Национальный исследовательский университет)

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Машинное обучение»

Лабораторная работа № 1, 2

Студент: Камеш Михаил

Группа: М80-307Б-18

Преподаватель: Ахмед Самир Халид

Москва, 2021

Постановка задания

1) Найти себе набор данных (датасет) для следующей лабораторной работы и проанализировать его. Выявить проблемы набора данных, устранить их. Визуализировать зависимости, показать распределение некоторых признаков. Реализовать алгоритмы К ближайших соседа с использованием весов и Наивный Байесовский классификатор и сравнить с реализацией библиотеки sklearn.

2) Необходимо реализовать алгоритмы машинного обучения. Применить данные алгоритмы на наборы данных, подготовленных в первой лабораторной работе. Провести анализ полученных моделей, вычислить метрики классификатора. Произвести тюнинг параметров в случае необходимости. Сравнить полученные результаты с моделями реализованными в scikit-learn. Аналогично построить метрики классификации. Показать, что полученные модели не переобучились. Также необходимо сделать выводы о применимости данных моделей к вашей задаче.

Описание алгоритмов

1. Алгоритм n-ближайших соседей (KNN)

Для классификации каждого из объектов тестовой выборки необходимо последовательно выполнить следующие операции:

А) Вычислить расстояние до каждого из объектов обучающей выборки

Б) Отобрать k объектов обучающей выборки, расстояние до которых минимально

С) Класс классифицируемого объекта — это класс, наиболее часто встречающийся среди k ближайших соседей

При взвешенном способе во внимание принимается не только количество попавших в область определённых классов, но и их удалённость от нового значения. Для каждого класса j определяется оценка близости:

$$Q_j = \sum_{i=1}^n \frac{1}{d(x, a_i)^2},$$

где $d(x, a_i)$ — расстояние от нового значения x до объекта a_i .

У какого класса выше значение близости, тот класс и присваивается новому объекту.

2. Гауссовский Наивный Байесовский классификатор

Основная идея — построить классификатор в предположении того, что функция $p(X_i, C_j)$ известна для каждого класса и равна плотности многомерного нормального (гауссовского) распределения:

$$p(x_i | c_j) = \frac{1}{\sqrt{2\pi\sigma_{i,j}^2}} e^{-\frac{1}{2}\left(\frac{x_i - \mu_{i,j}}{\sigma_{i,j}}\right)^2} \text{ for } i = 1, 2 \text{ and } j = 1, 2, 3$$

где μ - среднее значение, а σ - стандартное отклонение, которое мы должны оценить по данным. Это означает, что мы получаем одно среднее значение для каждого признака i в паре с классом c .

3. Логистическая регрессия

Логистическая регрессия применяется для прогнозирования вероятности возникновения некоторого события по значениям множества признаков. Для этого вводится так называемая *зависимая переменная* y , принимающая лишь одно из двух значений — как правило, это числа 0 (событие не произошло) и 1 (событие произошло), и множество *независимых переменных* (также называемых признаками, предикторами или регрессорами) — вещественных x , на основе значений которых требуется вычислить вероятность принятия того или иного значения зависимой переменной. Как и в случае линейной регрессии, для простоты записи вводится фиктивный признак $x_0 = 1$

Делается предположение о том, что вероятность наступления события $y = 1$ равна:

$$\mathbb{P}\{y = 1 \mid x\} = f(z),$$

где $z = \theta^T x = \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n$, x и θ — векторы-столбцы значений независимых переменных и параметров (коэффициентов регрессии) — вещественных чисел θ , соответственно, а $f(z)$ — так называемая *логистическая функция* (иногда также называемая сигмоидом или логит-функцией):

$$f(z) = \frac{1}{1 + e^{-z}}.$$

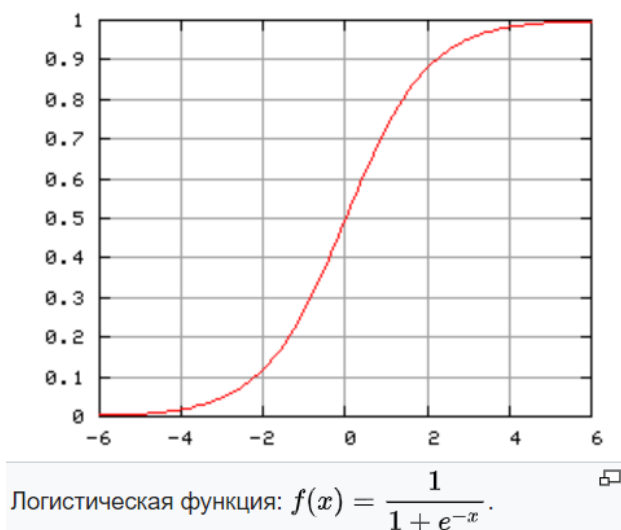
Так как y принимает лишь значения 0 и 1, то вероятность принять значение 0 равна:

$$\mathbb{P}\{y = 0 \mid x\} = 1 - f(z) = 1 - f(\theta^T x).$$

Для краткости функцию распределения при заданном x можно записать в таком виде:

$$\mathbb{P}\{y \mid x\} = f(\theta^T x)^y (1 - f(\theta^T x))^{1-y}, \quad y \in \{0, 1\}.$$

Фактически, это есть распределение Бернулли с параметром, равным $f(\theta^T x)$.



4. Дерево решений

Дерево решений — в основном жадное, нисходящее, рекурсивное разбиение. Энтропия — это мера случайности или неопределенности. Уровень энтропии колеблется от 0 до 1. Для меры энтропии используют примесь Джини. Узел чистый, если все его выборки принадлежат одному и тому же классу, в то время как узел с множеством выборок из разных классов будет иметь Джини ближе к 1.

$$G = 1 - \sum_{k=1}^n p_k^2$$

Каждый узел делит выборку таким образом, что примесь Джини у детей (точнее, среднее значение Джини у детей, взвешенных по их размеру) сводится к минимуму. Рекурсия останавливается, когда, достигается максимальная глубина, или когда нет разделения, которое может привести к двум детям, чище, чем их родитель.

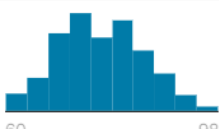
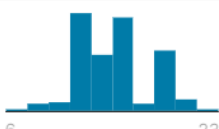

5. Случайный лес

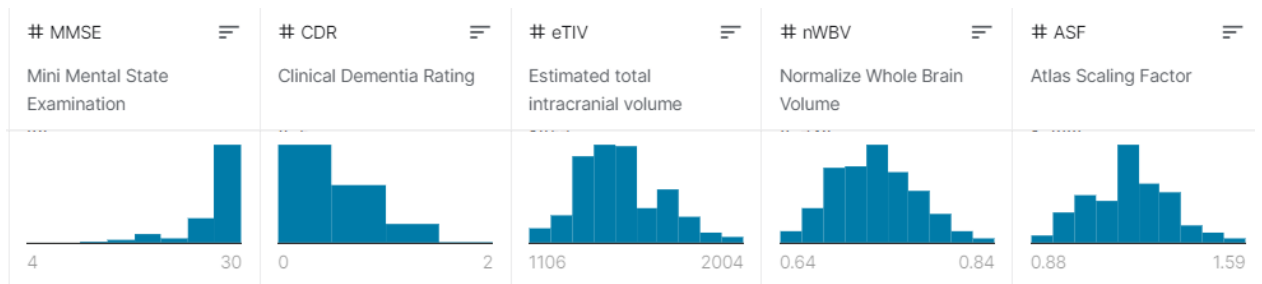
RF (random forest) — это множество решающих деревьев. В задаче регрессии их ответы усредняются, в задаче классификации принимается решение голосованием по большинству. Все деревья строятся независимо по следующей схеме:

- Выбирается подвыборка обучающей выборки размера `samplesize` (м.б. с возвращением) — по ней строится дерево (для каждого дерева — своя подвыборка).
- Для построения каждого расщепления в дереве просматриваем `max_features` случайных признаков (для каждого нового расщепления — свои случайные признаки).
- Выбираем наилучший признак и расщепление по нему (по заранее заданному критерию). Дерево строится, как правило, до исчерпания выборки (пока в листьях не останутся представители только одного класса), но в современных реализациях есть параметры, которые ограничивают высоту дерева, число объектов в листьях и число объектов в подвыборке, при котором проводится расщепление.

Используемый датасет

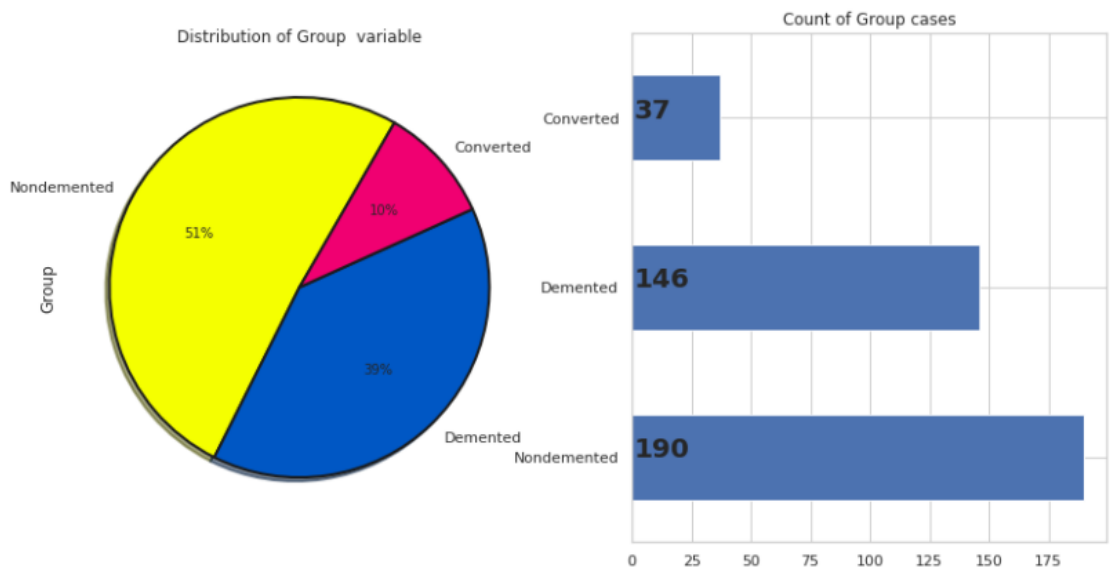
Признаки болезни альцгеймера (взяты с kaggle)

▲ Group		▲ M/F		# Age		# EDUC		# SES	
Class		Male - Female		Age		Years of Education		Socioeconomic Status / 1-5 1 - Low 5 - High	
Nondemented	51%	F	57%						
Demented	39%	M	43%						
Other (37)	10%								

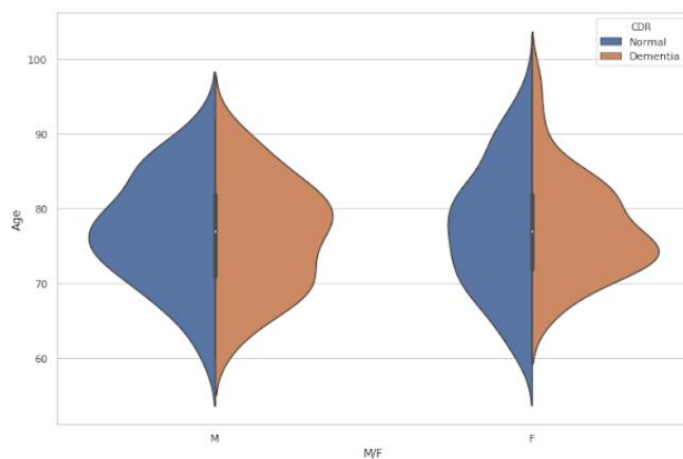


Group --> Class
 Age --> Age
 EDUC --> Years of Education
 SES --> Socioeconomic Status / 1-5
 MMSE --> Mini Mental State Examination
 CDR --> Clinical Dementia Rating
 eTIV --> Estimated total intracranial volume
 nWBV --> Normalize Whole Brain Volume
 ASF --> Atlas Scaling Factor

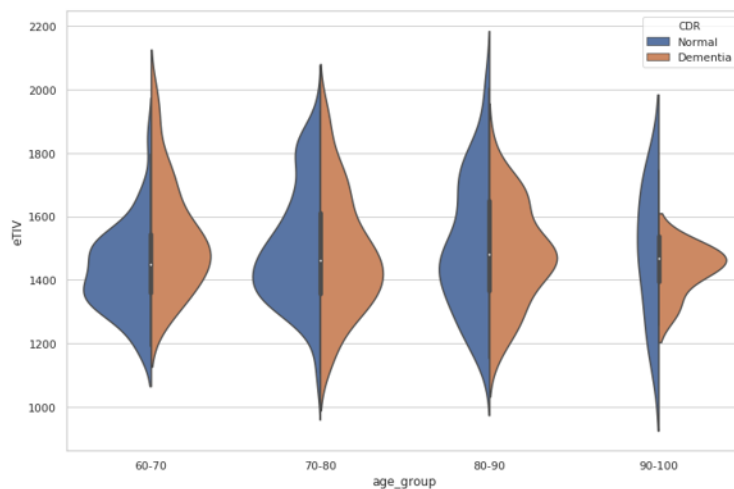
В данном датасете отсутствуют пропущенные значения, поэтому сразу можно выделить два значимых параметра – психическое состояние (MMSE) и объем мозга (eTIV). Разбиение на тестовые и тренировочные данные происходит случайным образом. Также разделим данные на две части.



Проанализируем, в какие периоды деменция прогрессирует в мужчинах и женщинах.



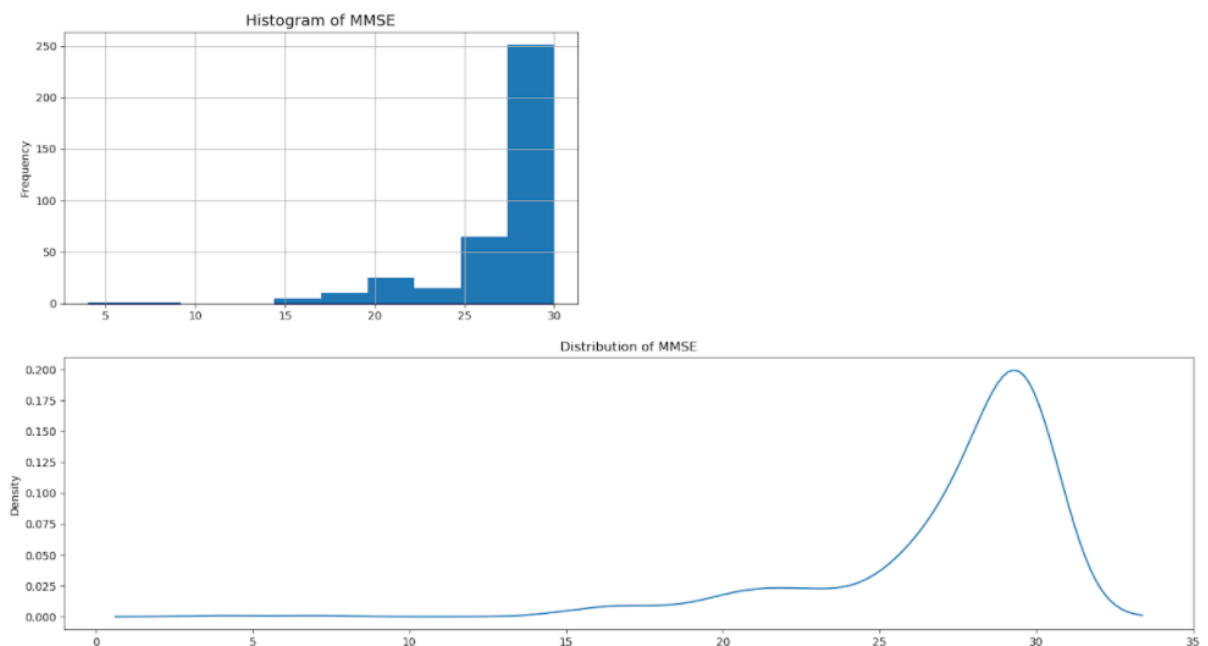
У мужчин наибольшее количество случаев деменции регистрируется в возрасте около 80 лет, в то время как у женщин пик деменции в 75 лет. Еще график показывает, что у мужчин деменция может начаться, когда они моложе 60, в то время как деменция у женщин обычно начинается после 65 лет.



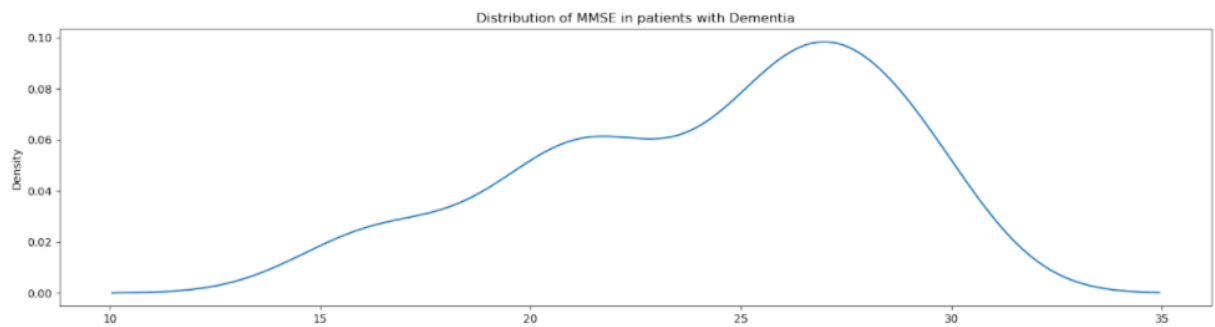
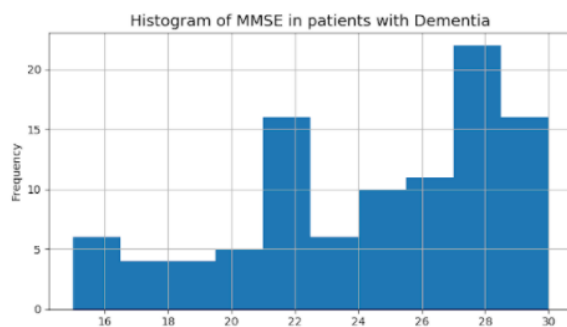
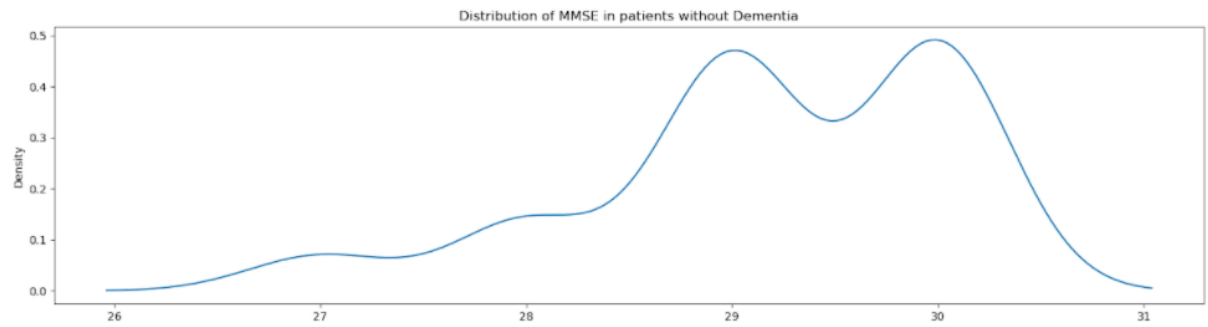
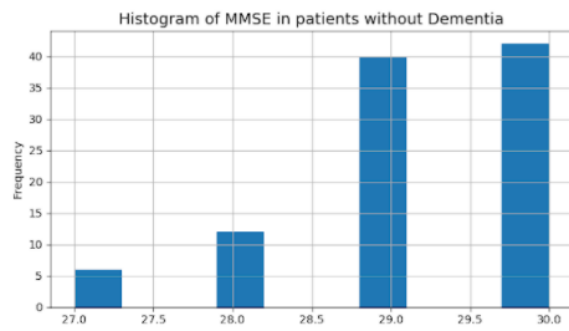
Из графика выше можно понять, что у пациентов с деменцией объем мозга с возрастом значительно отличается от здоровых людей, и с средним он значительно меньше.

Для анализа данных возьмем две группы пациентов по 100 человек – с альцгеймером и без в качестве даты.

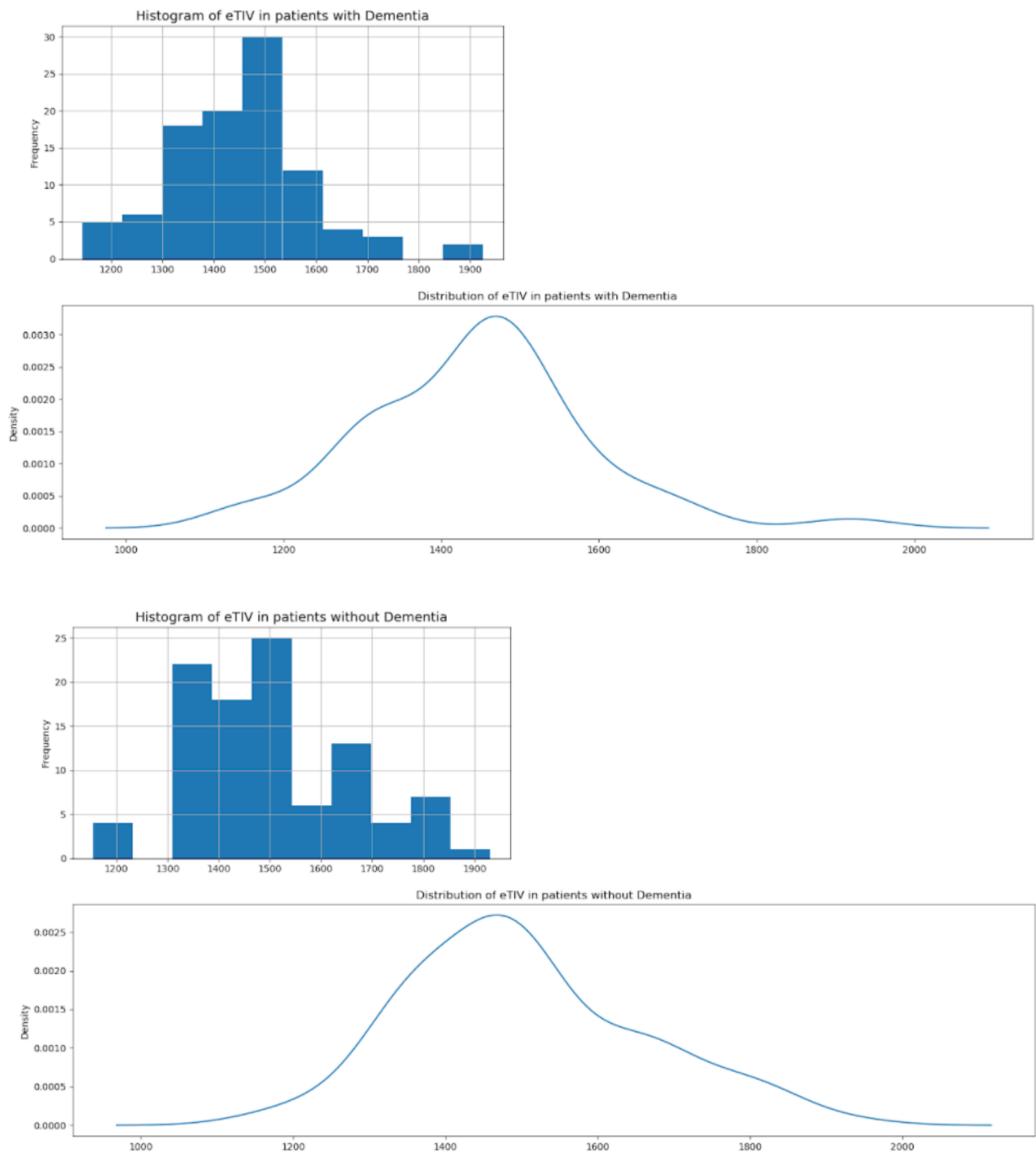
Построим распределение психического состояния пациентов до соответствующего разделения.



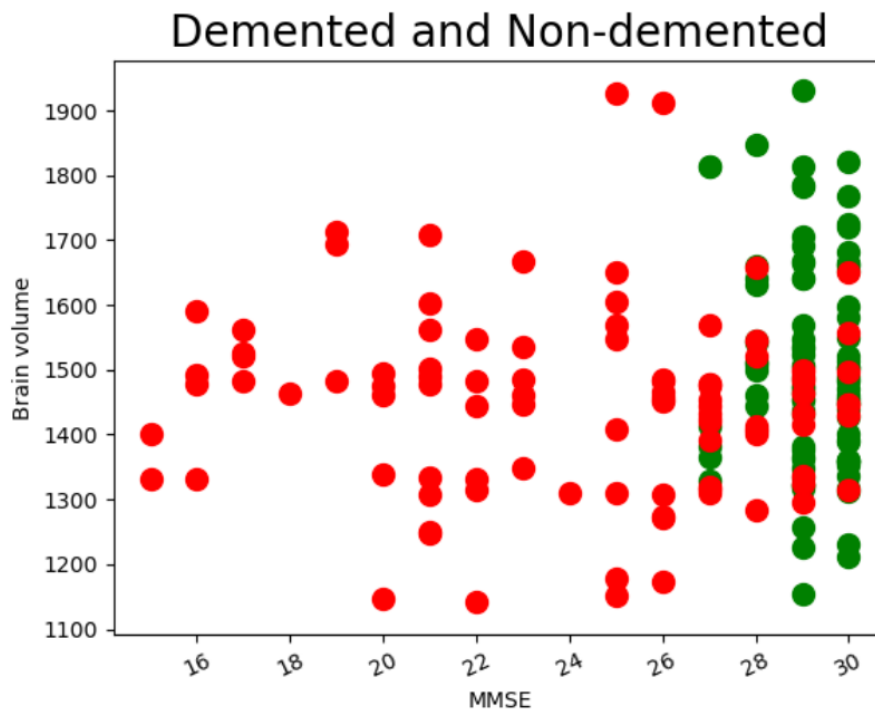
Если произвести разделения и построить соответствующие графики для пациентов с и без деменции, то можно заметить, что психическое состояние больных людей значительно хуже здоровых в среднем.



Выше мы установили, что объем мозга у пациентов с деменцией меньше, чем у здоровых. Проиллюстрируем это на разделенных данных.



Возьмем оба набора данных – психическое состояние и объем мозга для дальнейшего анализа данных. Ниже приведена модель, на которой происходит анализ. Красным показаны больные пациенты, а зеленым – здоровые.



Как видно, данные значительно различаются для двух групп и их можно довольно успешно разделить между собой.

Результаты:

Лабораторная 1

```
1.1 Custom KNeighborsClassifier
Custom KNN accuracy = 0.8484848484848485
Custom KNN precision = 0.8
Custom KNN recall = 0.8571428571428571

1.2 SKlearn KNN
SKlearn KNN accuracy = 0.8787878787878788
SKlearn KNN precision = 0.8125
SKlearn KNN recall = 0.9285714285714286

2.1 Custom Naive Bayes classifier
Custom NB accuracy = 0.8181818181818182
Custom NB precision = 0.75
Custom NB recall = 0.8571428571428571

2.2 SKlearn Naive Bayes classifier
SKlearn NB accuracy = 0.8181818181818182
SKlearn NB precision = 0.75
SKlearn NB recall = 0.8571428571428571
```

Лабораторная 2

```
1.1: SKlearn Logistic Regression
SKlearn log accuracy: 0.9642857142857143
SKlearn log precision = 1.0
SKlearn log recall = 0.9285714285714286

1.2: Custom Logistic Regression
Custom log accuracy: 0.9642857142857143
Custom log precision = 1.0
Custom log recall = 0.9285714285714286
Custom log train accuracy: 0.8197674418604651

2.1: SKlearn DecisionTreeClassifier
SKlearn Dtree accuracy: 0.8928571428571429
SKlearn Dtree precision = 1.0
SKlearn Dtree recall = 0.9285714285714286

2.2: Custom Decision Tree Classifier
Custom Dtree accuracy: 0.8928571428571429
Custom Dtree precision = 0.8666666666666667
Custom Dtree recall = 0.9285714285714286
Custom Dtree train accuracy: 0.9069767441860465

3.1: SKlearn Random Forest Classifier
SKlearn RF accuracy: 0.9285714285714286
SKlearn RF precision = 0.875
SKlearn RF recall = 1.0

3.2: Custom Random Forest Classifier
Custom RF accuracy: 0.9285714285714286
Custom RF precision = 0.875
Custom RF recall = 1.0
Custom RF train accuracy: 0.9651162790697675
```

Выводы:

В результате выполнения работы были изучены различные алгоритмы машинного обучения. При повторной генерации тестовых данных вручную имплементированные алгоритмы иногда допускают на несколько ошибок больше, чем sklearn аналоги, но в общем работают с одинаковой точностью.

По применимости алгоритмов больше всего подходят логическая регрессия из-за разделения аргументов на 2 класса и маленького размера датасета.