Домашнее задание Каждый студент получает 3 файла train_датасет.npy, test_датасет_open.npy, test_датасет_closed.npy. Каждый файл содержит словарь с признаками, один из признаков - Class. Также каждый студент получает 2 классификатора из списка (kNN, LogReg, SVM, NeuralNetworks). Задание состоит в подборе параметров классификаторов на обучающей выборке и проверке качества на тестовой выборке. Дополнительные баллы можно получить за хорошие результаты

Требуется.

1. Выполнить подбор параметров моделей на обучающем датасете. Оценить качество модели с помощью метода кросс-валидации
2. Попробовать улучшить модель. Для улучшения можно использовать следующие методы:

- Нормализация данных
- Подбор гиперпараметров моделей
- Уменьшение размерности данных
- Другие методы

Каждый раз проверять качество с помощью кросс-валидации

3. Сделать выводы, какие методы улучшения работают, какие нет. Посчитать accuracy лучших моделей на открытом датасете.
4. На полученных моделях построить ROC кривые для открытого тестового множества. Сравнить полученные модели между собой. Какая лучше работает, сделать предположение почему. Для построения ROC кривой для числа классов больше, чем 2, воспользуйтесь многоклассовый ROC и методом predict_proba

Результат выполнения задания

- jupyter ноутбук Task2_Фамилия выполненными пунктами 1-4.
- файлы Task2_Фамилия_алгоритм1.npy Task2_Фамилия_Имя_алгоритм2.npy с результатами классификации на закрытом тесте, где алгоритм1 - ваш первый метод, алгоритм2 - ваш второй метод

Оценка за задание.

- Пункты 1-4 выполнены, результаты accuracy не выше бейзлайнов для каждого метода - 2.
- Пункты 1-4 выполнены, и хотя бы один результат выше бейзлайна - 2,5.
- Пункты 1-4 выполнены, и хотя бы один результат лучший среди всех работ для данного метода для данного датасета - 2,5 + 1 экстра балл.
- Пункты 1-4 выполнены, и хотя бы один результат лучший среди всех работ для данного датасета - 2,5 + 2 экстра балла.

Пример генерации ответов на закрытом датасете

In [ ]:
```
import numpy as np
from sklearn.naive_bayes import GaussianNB

dataTrain = np.load("C:\\Users\\Алексей\\vscode\\hometask\\Task2\\testYou\\train
dataTestOpen = np.load("C:\\Users\\Алексей\\vscode\\hometask\\Task2\\testYou\\te
dataTestClosed = np.load("C:\\Users\\Алексей\\vscode\\hometask\\Task2\\testYou\\

X_train = []
X_test_open = []
X_test_closed = []
for c in dataTrain:
    if c == 'Class':
        Y_train = dataTrain[c]
    else:
        X_train.append(dataTrain[c].astype(float))
Y_train = np.array(Y_train)
for c in dataTestOpen:
    if c == 'Class':
        Y_test = dataTestOpen[c]
    else:
        X_test_open.append(dataTestOpen[c].astype(float))
Y_test = np.array(Y_test)
for c in dataTestClosed:
    X_test_closed.append(dataTestClosed[c].astype(float))

classes = np.unique(Y_train)
for i,c in enumerate(classes):
    Y_train[np.argwhere(Y_train == c)] = i
    Y_test[np.argwhere(Y_test == c)] = i
Y_train = Y_train.astype(int)
Y_test = Y_test.astype(int)
X_train = np.array(X_train).transpose()
X_test_open = np.array(X_test_open).transpose()
X_test_closed = np.array(X_test_closed).transpose()

gaus = GaussianNB()
gaus.fit(X_train,Y_train)
Y_test_closed = gaus.predict(X_test_closed)
np.save('Летуновский_Алексей_GaussianNB.npy',Y_test_closed)
```