1. Постановка задачи

Провести серию экспериментов с построением и тестированием деревьев решений (используя DecisionTreeClassifier и RandomForestClassifier), переразбивая исходное множество данных, заданное в варианте, следующим образом:

Номер эксперимента Размер обучающей выборки Размер тестовой выборки

Номер эксперимента	Размер обучающей выборки	Размер тестовой выборки
1	60 %	40 %
2	70 %	30 %
3	80 %	20 %
4	90 %	10 %

2. Исходные данные

- Датасет: http://archive.ics.uci.edu/ml/datasets/seeds
- Предметная область: семена пшениц
- Задача: определить, к какому из 3х типов относится каждое семя (Kama, Rosa and Canadian)
- Количество записей: 210
- Количество атрибутов: 7
- Атрибуты:

1. area A,
2. perimeter P,
3. compactness C = 4*pi*A/P^2,
4. length of kernel,
5. width of kernel,
6. asymmetry coefficient
7. length of kernel groove.

3. Ход работы

```
import numpy
import pandas
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split

# разделение датасета на тестовую и обучающую выборку
def split_dataset(size):
	ds = pandas.read_csv('seeds_dataset.txt', sep='\t', lineterminator='\n',
header=None).values
	ds_attributes = ds[:, :-1] # атрибуты семени
	ds_class = ds[:, -1].astype(numpy.int64, copy=False) # класс семени
	return train_test_split(ds_attributes, ds_class, test_size=size,
random_state=55)

def main():
	max_size = 0.4
	min_size = 0.1
	step = 0.1
	for size in numpy.arange(min_size, max_size, step):
		data_train, data_test, class_train, class_test =
split_dataset(size)

		decisionForest = DecisionTreeClassifier()
		decisionForest = decisionForest.fit(data_train, class_train)
		decisionAcc = decisionForest.score(data_test, class_test)

		randonForest = RandomForestClassifier()
		randonForest = randonForest.fit(data_train, class_train)
		randomAcc = randonForest.score(data_test, class_test)

		print('Size: ', size)
		print('Decision Tree accuracy: ', round(decisionAcc,10))
		print('Random Tree accuracy: ', round(randomAcc,10))
		print('\n')

main()
```

4. Результаты

Lab_1

Naïve Bayes:

myNBClass Accuracy: 0.9245283018867925

sklNBClass Accuracy: 0.924528301887

K Nearest Neighbors:

myKNClass Accuracy: 0.962264150943

sklKNClass Accuracy: 0.962264150943

Lab_2

Size: 0.1

Decision Tree accuracy: 0.9523809524

Random Tree accuracy: 1.0

Size: 0.2

Decision Tree accuracy: 0.9047619048

Random Tree accuracy: 0.8571428571

Size: 0.3

Decision Tree accuracy: 0.953125

Random Tree accuracy: 0.984375

Size: 0.4

Decision Tree accuracy: 0.9523809524

Random Tree accuracy: 0.9166666667

В ходе проделанной работы были получены приведенные выше результаты. Точность алгоритмов примерно схожая и варьируется в зависимости от размеров выборки. Оба алгоритма показали результат с высокой точностью, при некоторых размерах выборки даже больше, чем у алгоритмов из лабораторной работы №1. Данный факт говорит о том, что использование деревьев решений при определенных размерах выборки является хорошим инструментом для решения поставленной задачи данного датасета.