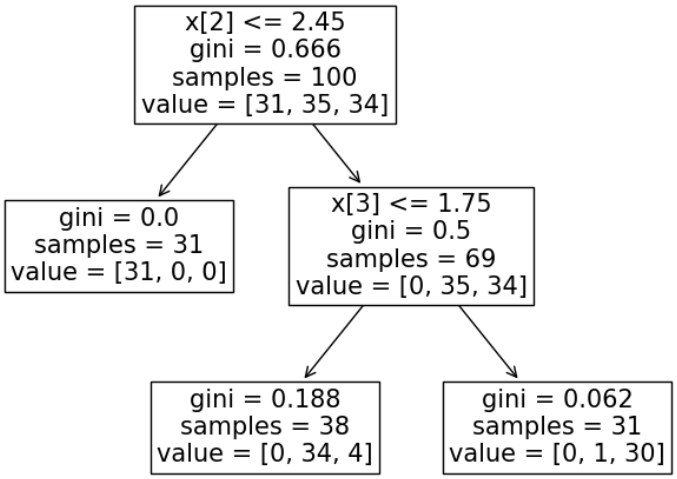
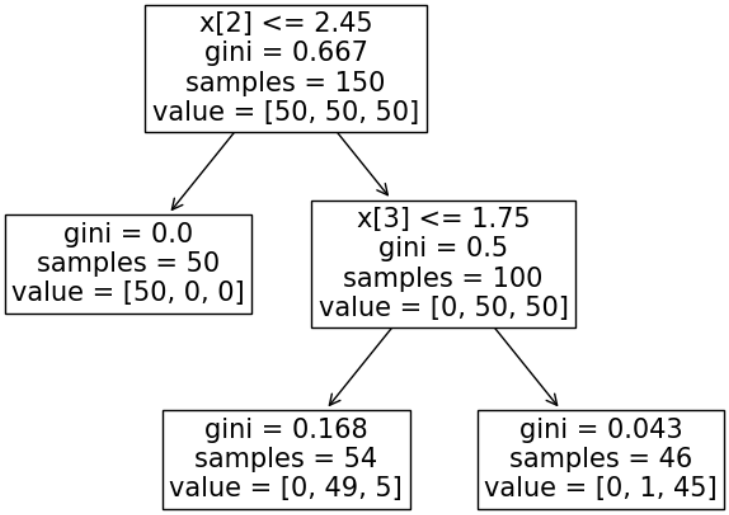
Задача 1.
Можно обучать классификаторы на тренировочной выборке и сравнивать их структуру со структурой при обучении на полной выборке.
Например, для решающего дерева: если решающие правила близки друг к другу, то это говорит о близости статистических характеристик тестовой выборки и полного набора данных, и говорит о репрезентативности разбиения.
На примере датасета iris библиотеки sklearn:

```
iris = load_iris()
X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42, test_size=0.33)
# загружаем датасет


clf = DecisionTreeClassifier(max_leaf_nodes=3, random_state=42)
clf.fit(X_train, y_train)
fig, ax = plt.subplots()
tree.plot_tree(clf, ax = ax, fontsize = 15)
plt.show()
# обучаем дерево и рисуем его структуру
```




```
clf_check = DecisionTreeClassifier(max_leaf_nodes=3, random_state=42)
clf_check.fit(X, y)
fig, ax = plt.subplots()
tree.plot_tree(clf_check, ax = ax, fontsize = 15)
plt.show()
# обучаем другое дерево на полном наборе данных и рисуем его структуру
```



Как видно из примера, решающие правила получились идентичными. Это свидетельствует о том, что разбиение на train и test было репрезентативно.

Задача 2.

Будем обозначать компоненты вектора x как $t_0[x]$, $t_1[x]$, …, $t_5[x]$ – обозначают 6 контролируемых параметров (транзакции) и $p_0[x]$, $p_1[x]$, …, $p_3[x]$ – обозначают 4 фиксированных параметров, характеристик клиента.

Обозначим так же центральные (средние) значения параметров для k-ого кластера. $t_i^{(k)}[\text{cent}] = \frac{1}{N[obj \in k-cluster]} \sum_{obj \in\, k-cluster} t_i[obj], \quad p_i^{(k)}[cent] = \frac{1}{N[obj \in k-cluster]} \sum_{obj \in\, k-cluster} p_i^k[obj]$

Введем параметр ε, который определяет неразличимые с практическое точки зрения квадраты расстояний в параметрическом пространстве $\varepsilon = \sum_{i=0}^{5} \Delta t_i^2 + \sum_{i=0}^{3} \Delta p_i^2$, где $\Delta t_i, \Delta p_i$ – параметры которые определяют границу пренебрежимо малых различий по каждому признаку (они в наших руках: например, можно установить $\Delta p_0 = 1$ день для параметра возраста, если мы хотим пренебречь различиями в возрасте двух людей при разнице в возрасте менее 1 дня и т.д.)

Поскольку мы можем приближаться к центру k-ого кластера только по подпространству t_i , задача оптимизации будет выглядеть так:

$$\sum_{i=0}^{5} \left(t_i[x] - t_i^{(2)}[cent]\right)^2 \to min_{\{t_i\}i=0,1,\dots 5}$$

$$\forall k \in \{1,3,4\}:$$

$$\sum_{i=0}^{5} (t_i[x] - t_i^{(k)}[cent])^2 + \sum_{i=0}^{3} \left(p_i[x] - p_i^k[cent]\right)^2 \geq \sum_{i=0}^{5} \left(t_i[x] - t_i^{(2)}[cent]\right)^2 + \sum_{i=0}^{3} (p_i[x] - p_i^2[cent])^2 + \varepsilon$$

$$\forall i \in \{0,1,2,3\}: p_i[x] = const$$

то есть мы будем минимизировать Евклидово расстояние до кластера 2 в подпространстве транзакций при условиях, что:

- центр кластера 2 должен являться ближайшим (по Евклидовой метрике 10-мерного пространства), то есть квадрат расстояния от объекта x до центра кластера 2 в пространстве полного набора параметров меньше, чем квадрат расстояния до центра любого другого

кластера, не менее чем на ε (то есть должно быть значимое для нас по смыслу превышение расстояний для остальных кластеров)
- все параметры $p_i[x]$ остаются фиксированными

Задача 3.

Один random forest из 1000 деревьев для задачи классификации лучше, чем два random forest'a из 500 деревьев каждый при одинаковых деревьях.
Проиллюстрируем проблему второго классификатора на примере задачи бинарной классификации:
Предположим, что первые 500 деревьев и вторые 500 деревьев проголосовали существенно по-разному (например, из-за того, что преимущественно обучились на разных группах признаков): например, из первой группы 499 деревьев проголосовало за класс A и 1 дерево за класс B, а из второй группы -- 251 дерево за класс B и 249 за класс A.
Из соображений здравого смысла нужно проголосовать за класс A, поскольку незначительный дисбаланс в сторону класса B (величиной в 2 голоса) во втором random forest несравнимо мал по сравнению с дисбалансом в сторону класса A (величиной в 498 голосов) в первом random forest.
Тем не менее, при использовании двух random forest'ов в такой ситуации не получится выдать предсказание: один лес проголосует за A, а другой за B, и их голоса будут равноправны.
В случае одного random forest в аналогичной ситуации такой проблемы не возникнет: у класса A будет перевес в 496 голосов и ответ будет надежно определен.

Заметим, что в случае задачи регрессии, если усреднять предсказания двух лесов как среднее арифметическое, и в каждом лесу усреднять показания деревьев также как среднее арифметическое, из-за равного количества деревьев в лесах такой проблемы не возникнет:

$$\frac{1}{2}\left[\frac{1}{500}\sum_{i<500} y_i + \frac{1}{500}\sum_{i\geq 500} y_i\right] = \frac{1}{1000}\sum_i y_i$$

ответы лесов будут в точности совпадать.

Задача 4

1. Сделаем кластеризацию всех клиентов из датасета по всем их признакам (не включая флажок дефолта/отсутствия дефолта: примем, что дефолту соответствует 1, а отсутствию дефолта 0) на n кластеров, обучив алгоритм k-means на тестовой выборке (число соседей и число кластеров -- параметры).
2. Для каждого элемента тренировочной выборки посчитаем вероятность дефолта следующим образом: $p = \frac{1}{N}\sum_i y_i$, где сумма берется по элементам кластера, к которому отнесен элемент тренировочной выборки, а N - число элементов тестовой выборки в этом кластере. Если ни у кого из тестовых клиентов кластера дефолта не было , p=0 , если у всех был дефолт, то p=1 , и т.д.
3. Вычислим суммарную квадратичную ошибку по всем элементам тестовой выборки: $error = \sum_j \left(p_j - y_{jtrue}\right)^2$. На основании наименьшей ошибки выберем оптимальную пару параметров кластеризации (число соседей и число кластеров).
4. K-means с оптимальным числом соседей и кластеров используем для предсказания вероятности дефолта для новых клиентов. Будем сначала классифицировать нового клиента, а затем вычислять вероятность по формуле из пункта №2.

Задача 5

Датасет обозначим как (X, y) , где y - известные данные по доходу, X- прочие персональные данные. При необходимости предобработаем данные (отобразим на отрезок [0,1] и/или отнормируем).

1. Многократно обучим подходящий по качеству алгоритм кластеризации без учителя (например, k-means) только на данных по доходу, используя все доступные данные по

доходу y, при этом будем варьировать параметр числа кластеров от 1 до числа элементов тренировочной в выборке N. На выходе получим N разных обученных классификаторов, которые умеют группировать клиентов по доходу.

2. Разделим датасет (X,y) на train и test.
3. Для каждого из полученных на шаге №1 классификаторов сконвертируем доходы y_{train} и y_{test} в дискретные discrete y_{train} и discrete y_{test} исходя из того, к какому классу они относятся. Получим, что данные по доходу превратились в дискретный признак (есть N способов это сделать), и клиенты разделились на дискретные группы по доходу (этих групп: одна для первого классификатора, две для второго, и так далее).
4. Для каждого из k-го способа конвертации из шага №3 обучим новый классификатор с учителем или без учителя на выборке train разбивать клиентов на k классов.
5. Для каждого из N новых классификаторов, обученных на шаге №4, проверим, введем метод вычисления непрерывной величины - дохода ранее неизвестного клиента X_{new} - по следующей схеме:
    a. Сначала определяем с помощью классификатора стандартным методом predict кластер по доходу discrete y_{pred}, в котором лежит неизвестный клиент.
    b. Затем считаем доход следующим образом: вычисляем взвешенную сумму доходов для клиентов из тренировочной выборки, относящихся к этому же кластеру, с весами, которые обратно пропорциональны Евклидовым расстояниям от нового клиента до этих клиентов из тренировочной выборки:

$$y_{pred}(X_{new}) = \left[\sum_{i:discrete\ y_{train}^{(i)} = discrete\ y_{pred}} y_{train}^{(i)} \left\| X_{new} - X_{train}^{(i)} \right\|^{-1} \right] \div \left[\sum_{i:discrete\ y_{train}^{(i)} = discrete\ y_{pred}} \left\| X_{new} - X_{train}^{(i)} \right\|^{-1} \right]$$

6. С помощью этого метода предсказания дохода сравниваем между собой качество предсказаний наших N классификаторов на выборке test по удобной метрике, например по среднему квадрату ошибки (MSE). Используем далее только классификатор с номером $k*$, у которого метрика получилась наилучшей, то есть кластеризуем клиентов по доходу на оптимальное число групп.
7. Для предсказания дохода неизвестного клиента распределяем его в один из $k*$ кластеров по доходу с помощью классификатора с номером $k*$, а затем подставляем его вектор признаков в формулу из пункта №5.b.

Можно по аналогии оптимизировать не только число кластеров по доходу, но и другие дискретные параметры кластеризации: в этом случае на шаге №6 нужно будет выбирать оптимальный классификатор из бОльшего числа моделей, а все остальные идеи останутся прежними.