1. Какими методами машинного обучения можно показать, что разбиение на трейн и тест репрезентативно?

Можно применить t-тест, chi-square тест для сравнения средних и распределений в выборках.
Можно визуализировать данные и сравнить распределение признаков в выборках.

2. Есть кластеризованный датасет на 4 кластера (1, 2, 3, 4). Бизнес аналитики посчитали, что самым прибыльным является кластер 2. Каждый клиент представлен в виде 10-тимертного вектора, где первые 6 значений транзакции, а оставшиеся: возраст, пол, социальный статус (женат (замужем)/неженат (не замужем)), количество детей. Нужно поставить задачу оптимизации для каждого клиента не из кластера 2 так, чтобы увидеть как должен начать вести себя клиент, чтобы перейти в кластер 2.

Добавим в датасет столбец, указывающий принадлежность каждого клиента к конкретному кластеру (1, 2, 3, 4) в качестве таргета.

Обучим модель на этом датасете предсказывать принадлежность к кластеру используя остальные признаки в качестве features.

Проанализируем какие из признаков имеют наибольшее влияние на принадлежность к кластеру 2. Изменение или учёт этих характеристик может повысить вероятность принадлежности к нужному кластеру для клиентов.

3. Что лучше 2 модели случайного леса по 500 деревьев или одна на 1000, при условии, что ВСЕ параметры кроме количества деревьев одинаковы?

У случайного леса рандомность применяется при

1 выборе поднабора данных при обучении нового дерева

2 выборе поднабора фичей, выбранных случайным образом при построении каждого узла

Таким образом если двум моделям случайного леса задать все параметры одинаковыми, включая random_state, то они будут выдавать одинаковый результат на один и тот же вход. Т.е. никакого смысла в этом нет. Это можно проверить запустив код, указанный ниже.

А так как чем больше деревьев, тем лучше для больших наборов данных или данных с большим наборов признаков, то получается один лес на 1000 деревьев лучше. К тому же в официальной документации по алгоритму random forest

algorithm утверждается, что он не переобучается и можно использовать сколь угодно много деревьев.

```
from sklearn.datasets import load_diabetes
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
diabetes = load_diabetes(as_frame = True)


X = diabetes['data']
y = diabetes['target']
# control randomness when splitting
X_train,X_test, y_train, y_test = train_test_split(X, y,test_size = 0.1, random_state
= 42)


for i in range(100):
   # control randomness at initialisation
   rfr_ONE = RandomForestRegressor(random_state = 42)
   rfr_ONE.fit(X_train, y_train)
   y_pred_ONE = rfr_ONE.predict(X_test)


   # control randomness at initialisation
   rfr_TWO = RandomForestRegressor(random_state = 42)
   rfr_TWO.fit(X_train, y_train)
   y_pred_TWO = rfr_TWO.predict(X_test)


   assert (y_pred_ONE == y_pred_TWO).all(), "No they don't"
```

4. В наличии датасет с данными по дефолту клиентов. Как, имея в инструментарии только алгоритм kmeans получить вероятность дефолта нового клиента.

Применить алгоритм kmeans и сформировать кластеры используя наш датасет . Количество кластеров, которые нужно найти можно указать 2 (в самом простом варианте).

После того, как алгоритм обучен на наборе данных и сформировал кластеры он назначает каждому клиенту в датасете свой кластер.

Для нового клиента можно использовать обученную модель для определения, к какому кластеру этот новый клиент будет наиболее близок. Необходимо, чтобы у нового клиента были те же признаки, которые использовались при обучении.

После того, как назначим кластер клиенту надо проанализировать данные о дефолте других клиентов из этого же кластера. Если большинство клиентов из этого кластера ранее столкнулись с дефолтом, то можно предположить, что и для нового клиента вероятность дефолта будет высокая и наоборот. Самый банальный способ - посчитать отношение дефолтных клиентов в кластере ко всем клиентам в кластере.

5. Есть выборка клиентов с заявкой на кредитный продукт. Датасет состоит из персональных данных: возраст, пол и т.д. Необходимо предсказывать доход клиента, который представляет собой непрерывные данные, но сделать это нужно используя только модель классификации.

Для прогнозирования дохода, используя классификацию, можно следовать такой схеме:
Разбиение на категории - преобразовать непрерывные данные о доходе в категории (например, диапазоны доходов).
Классификация - обучение классификатора для предсказания категории дохода.
Постобработка - вычисление предсказания дохода, как среднего или медианы диапазона предсказанной категории.