1. Загрузите данные load_wine из sklearn.datasets. Из обучающей части исключите объекты класса 2. Обучите случайный лес, задав только гиперпараметры `n_estimators=100` и `random_state=0` . Оцените важность признаков. Укажите название двух наиболее важных признаков.

In [ ]:
```
from sklearn.datasets import load_wine
data = load_wine()
```

1. Загрузите данные load_wine из sklearn.datasets. Из обучающей части исключите объекты класса 2. Отмасштабируйте признаки, используя класс StandardScaler с гиперпараметрами по умолчанию. Обучите случайный лес, задав только гиперпараметры `n_estimators = 100` и `random_state=0` . Оцените важность признаков. Укажите название двух наиболее важных признаков.

In [ ]:
```

```

Ниже приведена неполная реализация класса Bagging который имеет методы `fit` для обучения бэггинга над `DecisionTreeRegressor` и метод `predict` для предсказания. Допишите необходимый код, чтобы реализовать бэггинг.

используемы переменные в коде:

- `self.n_estimators` , `n_estimators` - число используемых деревьев
- `self.regressors` - список объектов класса `DecisionTreeRegressor` , к которым уже был применён метод `fit` Данный список необъодимо заполнить в методе `fit` и использовать для предсказания в методе `predict`
- `ind` - выбранные индексы объектов при бутстрапе

при создании объекта класса `DecisionTreeRegressor` зафиксируйте `random_state=0`

In [ ]:
```
import numpy as np
from sklearn.tree import DecisionTreeRegressor
class Bagging():
  def __init__(self, n_estimators=10):
    self.n_estimators = n_estimators
    self.regressors = []
  def fit(self, x_train, y_train):
    for i in range(self.n_estimators):
      np.random.seed(i)
      ind = np.random.choice(np.arange(x_train.shape[0]), size = x_train.shape[0])
       # your code here
  def predict(self, x_test):
    # your code here
```

Загрузите данные приложенные к заданию

In [ ]:
```
import pandas as pd
from google.colab import files
uploder = files.upload()
```

Выбрать файлы Файл не выбран

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

Положим матрицу объекты-признаки в переменную `X` , а ответы в переменную `y`

In [ ]:
```
X, y = Data.iloc[:, :100], Data.iloc[:, 100]
```

Положим первые 6000 объектов в обучающую часть, остальные объекты в тестовую часть

In [ ]:
```
x_train, y_train = X[:6000], y[:6000]
x_test, y_test = X[6000:], y[6000:]
```

1. Обучите бэггинг на 1 дереве. Оцените качество по метрике MSE на тестовой части. Ответ разделите на 1000 и округлите до целой части по математичестким правилам округления.

In [ ]:

1. Обучите бэггинг на 5 деревьях. Оцените качество по метрике MSE на тестовой части. Ответ разделите на 1000 и округлите до целой части по математичестким правилам округления.

In [ ]:

1. Обучите бэггинг на 100 деревьях. Оцените качество по метрике MSE на тестовой части. . Ответ разделите на 1000 и округлите до целой части по математичестким правилам округления.

In [ ]:

1. Обучите на этих же данных случайный лес, используйте гиперпараметр `n_estimators = 1` , зафиксируйте `random_state=0` . Оцените качество по метрике MSE на тестовой части. . Ответ разделите на 1000 и округлите до целой части по математичестким правилам округления.

In [ ]:

1. Обучите на этих же данных случайный лес, используйте гиперпараметр `n_estimators = 1` , зафиксируйте `random_state=0` . Оцените качество по метрике MSE на тестовой части. . Ответ разделите на 1000 и округлите до целой части по математичестким правилам округления.

In [ ]:

1. Обучите на этих же данных случайный лес, используйте гиперпараметр `n_estimators = 5` , зафиксируйте `random_state=0` . Оцените качество по метрике MSE на тестовой части. . Ответ разделите на 1000 и округлите до целой части по математичестким правилам округления.

In [ ]:

1. Обучите на этих же данных случайный лес, используйте гиперпараметр `n_estimators = 100` , зафиксируйте `random_state=0` . Оцените качество по метрике MSE на тестовой части. . Ответ разделите на 1000 и округлите до целой части по математичестким правилам округления.

In [ ]:

1. Изучите документацию и разберитесь как посчитать Out-of-bag ошибку в RandomForestRegressor. Обучите RandomForestRegressor с гиперпараметром n_estimators=100 на обучающей части, зафиксируйте `random_state=0` . Найдите Out-of-bag ошибку алгоритма. Ответ округлите до сотых.

In [ ]: