Задание: Выберите набор данных (датасет) для решения задачи классификации или регресии. В случае необходимости проведите удаление или заполнение пропусков и кодирование категориальных признаков. С использованием метода train_test_split разделите выборку на обучающую и тестовую. Обучите следующие ансамблевые модели: одну из моделей группы бэггинга (бэггинг или случайный лес или сверхслучайные деревья); одну из моделей группы бустинга; одну из моделей группы стекинга. Оцените качество моделей с помощью одной из подходящих для задачи метрик. Сравните качество полученных моделей.

In [ ]:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.linear_model import RidgeClassifier
import seaborn as sns
%matplotlib inline
sns.set(style="ticks")
```

In [ ]:

```
# Выберем исходный набор данных и проведём разбиение на обучающую и тестовую выборки
iris = load_iris()

iris_X_train, iris_X_test, iris_y_train, iris_y_test = train_test_split(
    iris.data, iris.target, test_size=0.5, random_state=1)
```

In [ ]:

```
# Обучение модели бэггинга
bg = BaggingClassifier(n_estimators=15, oob_score=True, random_state=1)
bg.fit(iris_X_train, iris_y_train)
```

Out[ ]:

BaggingClassifier(n_estimators=15, oob_score=True, random_state=1)

In [ ]:

```
# Оценка качества модели
accuracy_score(iris_y_test, bg.predict(iris_X_test))
```

Out[ ]:

0.9466666666666667

In [ ]:

```
# Обучение модели бустинга (градиентный спуск)
gb = GradientBoostingClassifier(n_estimators=15, random_state=1)
gb.fit(iris_X_train, iris_y_train)
```

Out[ ]:

GradientBoostingClassifier(n_estimators=15, random_state=1)

In [ ]:

```
# Оценка качества модели
accuracy_score(iris_y_test, gb.predict(iris_X_test))
```

Out[ ]:

0.96

In [ ]:

```
from heamy.estimator import Regressor
from heamy.pipeline import ModelsPipeline
from heamy.dataset import Dataset

dataset = Dataset(iris_X_train, iris_y_train, iris_X_test, iris_y_test)

model_tree = Regressor(dataset=dataset, estimator=DecisionTreeClassifier, name='tree')
model_lr = Regressor(dataset=dataset, estimator=RidgeClassifier,name='lr')
model_rf = Regressor(dataset=dataset, estimator=RandomForestClassifier, parameters={'n_estimators': 50},name='rf')

pipeline = ModelsPipeline(model_tree, model_lr, model_rf)
stack_ds = pipeline.stack(k=10, seed=1)
# модель второго уровня
stacker = Regressor(dataset=stack_ds, estimator=DecisionTreeClassifier)
results = stacker.validate(k=10,scorer=accuracy_score)
```

Metric: accuracy_score
Folds accuracy: [1.0, 1.0, 1.0, 0.875, 1.0, 1.0, 0.8571428571428571, 0.8571428571428571, 0.8571428571428571, 0.8571428571428571]
Mean accuracy: 0.9303571428571429
Standard Deviation: 0.06982576712863271
 [illegible]044