# Ансамбли моделей машинного обучения

Цель лабораторной работы: изучение ансамблей моделей машинного обучения.

## Задание

1. Выберите набор данных (датасет) для решения задачи классификации или регресии.
2. В случае необходимости проведите удаление или заполнение пропусков и кодирование категориальных признаков.
3. С использованием метода train\_test\_split разделите выборку на обучающую и тестовую.
4. Обучите следующие ансамблевые модели:
   1. одну из моделей группы бэггинга (бэггинг или случайный лес или сверхслучайные деревья);
   2. одну из моделей группы бустинга;
   3. одну из моделей группы стекинга.
5. (+1 балл на экзамене) Дополнительно к указанным моделям обучите еще две модели:
   1. Модель многослойного персептрона. По желанию, вместо библиотеки scikit-learn возможно использование библиотек TensorFlow, PyTorch или других аналогичных библиотек.
   2. Модель МГУА с использованием библиотеки - <https://github.com/kvoyager/GmdhPy> (или аналогичных библиотек). Найдите такие параметры запуска модели, при которых она будет по крайней мере не хуже, чем одна из предыдущих ансамблевых моделей.
6. Оцените качество моделей с помощью одной из подходящих для задачи метрик. Сравните качество полученных моделей.

## Ход работы

### Выбор и загрузка датасета

# %matplotlib inline  
# sns.set(style="ticks")  
  
import pandas as pd  
import numpy as np  
import seaborn as sns  
import matplotlib.pyplot as plt  
# import gmdhpy  
from sklearn.datasets import \*  
from sklearn.preprocessing import MinMaxScaler  
from sklearn.model\_selection import train\_test\_split  
from sklearn.linear\_model import LogisticRegression  
from sklearn import svm, tree  
from sklearn.tree import DecisionTreeClassifier  
# from sklearn.ensemble import RandomForestClassifier  
from sklearn.ensemble import BaggingClassifier  
from sklearn.ensemble import GradientBoostingClassifier  
from sklearn.ensemble import StackingClassifier  
from sklearn.neural\_network import MLPClassifier  
from gmdhpy.gmdh import Classifier  
# from heamy.estimator import Classifier  
# from heamy.pipeline import ModelsPipeline  
# from heamy.dataset import Dataset  
from sklearn.metrics import accuracy\_score  
# from sklearn.metrics import mean\_absolute\_error  
from sklearn.metrics import confusion\_matrix, ConfusionMatrixDisplay  
from operator import itemgetter  
  
def make\_dataframe(ds\_function):  
 ds = ds\_function()  
 df = pd.DataFrame(data= np.c\_[ds['data'], ds['target']],  
 columns= list(ds['feature\_names']) + ['target'])  
 return df  
  
wine = load\_wine()  
  
df = make\_dataframe(load\_wine)

# Первые 5 строк датасета  
df.head()

alcohol malic\_acid ash alcalinity\_of\_ash magnesium total\_phenols   
0 14.23 1.71 2.43 15.6 127.0 2.80 \  
1 13.20 1.78 2.14 11.2 100.0 2.65   
2 13.16 2.36 2.67 18.6 101.0 2.80   
3 14.37 1.95 2.50 16.8 113.0 3.85   
4 13.24 2.59 2.87 21.0 118.0 2.80   
  
 flavanoids nonflavanoid\_phenols proanthocyanins color\_intensity hue   
0 3.06 0.28 2.29 5.64 1.04 \  
1 2.76 0.26 1.28 4.38 1.05   
2 3.24 0.30 2.81 5.68 1.03   
3 3.49 0.24 2.18 7.80 0.86   
4 2.69 0.39 1.82 4.32 1.04   
  
 od280/od315\_of\_diluted\_wines proline target   
0 3.92 1065.0 0.0   
1 3.40 1050.0 0.0   
2 3.17 1185.0 0.0   
3 3.45 1480.0 0.0   
4 2.93 735.0 0.0

df.dtypes

alcohol float64  
malic\_acid float64  
ash float64  
alcalinity\_of\_ash float64  
magnesium float64  
total\_phenols float64  
flavanoids float64  
nonflavanoid\_phenols float64  
proanthocyanins float64  
color\_intensity float64  
hue float64  
od280/od315\_of\_diluted\_wines float64  
proline float64  
target float64  
dtype: object

# Проверим наличие пустых значений  
# Цикл по колонкам датасета  
for col in df.columns:  
 # Количество пустых значений - все значения заполнены  
 temp\_null\_count = df[df[col].isnull()].shape[0]  
 print('{} - {}'.format(col, temp\_null\_count))

alcohol - 0  
malic\_acid - 0  
ash - 0  
alcalinity\_of\_ash - 0  
magnesium - 0  
total\_phenols - 0  
flavanoids - 0  
nonflavanoid\_phenols - 0  
proanthocyanins - 0  
color\_intensity - 0  
hue - 0  
od280/od315\_of\_diluted\_wines - 0  
proline - 0  
target - 0

### Разделение на тестовую и обучающую выборки

y = df['target']  
x = df.drop('target', axis = 1)  
  
scaler = MinMaxScaler()  
scaled\_data = scaler.fit\_transform(x)  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(scaled\_data, y, test\_size = 0.4, random\_state = 0)  
  
print(f"Обучающая выборка:\n{x\_train, y\_train}")  
print(f"Тестовая выборка:\n{x\_test, y\_test}")

Обучающая выборка:  
(array([[0.58157895, 0.64031621, 0.4973262 , ..., 0.27642276, 0.63369963,  
 0.28673324],  
 [0.80789474, 0.28063241, 0.5026738 , ..., 0.62601626, 0.6959707 ,  
 0.87874465],  
 [0.15263158, 0.12055336, 0.71657754, ..., 0.3902439 , 0.72893773,  
 0.28673324],  
 ...,  
 [0.36578947, 0.17193676, 0.44385027, ..., 0.47154472, 0.61904762,  
 0.04778887],  
 [0.75526316, 0.18577075, 0.40641711, ..., 0.3495935 , 0.75457875,  
 0.5042796 ],  
 [0.82368421, 0.34980237, 0.59893048, ..., 0.11382114, 0.16117216,  
 0.2724679 ]]), 43 0.0  
10 0.0  
109 1.0  
73 1.0  
171 2.0  
 ...   
103 1.0  
67 1.0  
117 1.0  
47 0.0  
172 2.0  
Name: target, Length: 106, dtype: float64)  
Тестовая выборка:  
(array([[0.71315789, 0.18379447, 0.47593583, 0.29896907, 0.52173913,  
 0.55862069, 0.54008439, 0.1509434 , 0.38170347, 0.38993174,  
 0.35772358, 0.70695971, 0.55777461],  
 [0.46315789, 0.38142292, 0.59893048, 0.58762887, 0.45652174,  
 0.17241379, 0.21518987, 0.20754717, 0.2681388 , 0.81228669,  
 0. , 0.07326007, 0.14407989],  
 [0.35263158, 0.0770751 , 0.42780749, 0.43298969, 0.18478261,  
 0.86896552, 0.58227848, 0.11320755, 0.46056782, 0.27047782,  
 0.60162602, 0.58608059, 0.10128388],  
 [0.66578947, 0.19565217, 0.58823529, 0.51030928, 0.5 ,  
 0.68275862, 0.51476793, 0.13207547, 0.64353312, 0.42406143,  
 0.40650407, 0.64468864, 0.60057061],  
 [0.53157895, 1. , 0.41176471, 0.56185567, 0.17391304,  
 0.56551724, 0.48734177, 0.32075472, 0.50473186, 0.11262799,  
 0.20325203, 0.67032967, 0.07275321],  
 [0.13947368, 0.25889328, 1. , 0.92268041, 0.5326087 ,  
 0.75862069, 1. , 0.64150943, 0.46056782, 0.40273038,  
 0.36585366, 0.88644689, 0.13338088],  
 [0.79736842, 0.27865613, 0.6684492 , 0.36082474, 0.55434783,  
 0.55862069, 0.45780591, 0.33962264, 0.26498423, 0.32167235,  
 0.47154472, 0.84615385, 0.7253923 ],  
 [0.35 , 0.61067194, 0.54545455, 0.53608247, 0.19565217,  
 0.45517241, 0.12236287, 0.69811321, 0.19873817, 0.54351536,  
 0.06504065, 0.11355311, 0.17261056],  
 [0.32105263, 0.19565217, 0.40641711, 0.43298969, 0.10869565,  
 0.23103448, 0.35654008, 0.45283019, 0.38485804, 0.18088737,  
 0.42276423, 0.6959707 , 0.16547789],  
 [0.27631579, 0.21541502, 0.51336898, 0.40721649, 0.11956522,  
 0.2137931 , 0.24472574, 0.73584906, 0.38801262, 0.09556314,  
 0.48780488, 0.36630037, 0.14407989],  
 [0.61315789, 0.35968379, 0.52941176, 0.48453608, 0.20652174,  
 0.14482759, 0.03375527, 0.45283019, 0.07255521, 0.36860068,  
 0.17886179, 0.43956044, 0.35805991],  
 [0.75 , 0.84980237, 0.46524064, 0.48453608, 0.10869565,  
 0. , 0. , 0.50943396, 0.0851735 , 0.30887372,  
 0.08130081, 0.02197802, 0.09771755],  
 [0.83421053, 0.20158103, 0.5828877 , 0.2371134 , 0.45652174,  
 0.78965517, 0.64345992, 0.39622642, 0.49211356, 0.46672355,  
 0.46341463, 0.57875458, 0.83594864],  
 [0.35263158, 0.06521739, 0.39572193, 0.40721649, 0.19565217,  
 0.87586207, 0.71940928, 0.20754717, 0.48580442, 0.27474403,  
 0.45528455, 0.54945055, 0.2724679 ],  
 [0.67105263, 0.36363636, 0.71122995, 0.71649485, 0.38043478,  
 0.19655172, 0.10548523, 0.49056604, 0.35646688, 0.62969283,  
 0.21138211, 0.19413919, 0.33666191],  
 [0.25526316, 0.03557312, 0.34224599, 0.43298969, 0.17391304,  
 0.49655172, 0.40506329, 0.32075472, 0.32176656, 0.10409556,  
 0.73170732, 0.67765568, 0. ],  
 [0.71842105, 0.15612648, 0.71657754, 0.45876289, 0.67391304,  
 0.67931034, 0.50632911, 0.69811321, 0.29652997, 0.35153584,  
 0.62601626, 0.63369963, 0.68259629],  
 [0.83157895, 0.16798419, 0.59893048, 0.30412371, 0.41304348,  
 0.8 , 0.75738397, 0.35849057, 0.45741325, 0.6331058 ,  
 0.6097561 , 0.56776557, 1. ],  
 [0.42368421, 0.12252964, 0.35294118, 0.31958763, 0.32608696,  
 0.35862069, 0.2257384 , 0.75471698, 0.06624606, 0.38139932,  
 0.40650407, 0.11721612, 0.12268188],  
 [0.73684211, 0.1798419 , 0.6631016 , 0.34020619, 0.26086957,  
 0.50689655, 0.55907173, 0.16981132, 0.59305994, 0.36860068,  
 0.61788618, 0.76923077, 0.70399429],  
 [0.54736842, 0.05335968, 0.18181818, 0.22680412, 0.08695652,  
 0.68965517, 0.59915612, 0.24528302, 0.58990536, 0.34300341,  
 0.5203252 , 0.6996337 , 0.15977175],  
 [0.53157895, 0.1798419 , 0.63636364, 0.3814433 , 0.30434783,  
 0.50689655, 0.44092827, 0.30188679, 0.32492114, 0.25341297,  
 0.5203252 , 0.45421245, 0.58987161],  
 [0.58157895, 0.36561265, 0.80748663, 0.53608247, 0.52173913,  
 0.62758621, 0.49578059, 0.49056604, 0.44479495, 0.25938567,  
 0.45528455, 0.60805861, 0.32596291],  
 [0.38947368, 0.19565217, 0.3315508 , 0.51030928, 0.16304348,  
 0.42068966, 0.33333333, 0.35849057, 0.33753943, 0.14163823,  
 0.45528455, 0.84249084, 0.2810271 ],  
 [0.34210526, 0.07114625, 0.49197861, 0.27835052, 0.33695652,  
 0.36896552, 0.15822785, 0.94339623, 0. , 0.16979522,  
 0.62601626, 0.14652015, 0.28673324],  
 [0.39210526, 0.33399209, 0.43315508, 0.53608247, 0.19565217,  
 0.54137931, 0.407173 , 0.24528302, 0.2555205 , 0.06143345,  
 0.34146341, 0.55311355, 0.03352354],  
 [0.36842105, 0.15612648, 0.4973262 , 0.56185567, 0.17391304,  
 0.60689655, 0.592827 , 0.49056604, 0.42902208, 0.22696246,  
 0.17073171, 0.57509158, 0.05278174],  
 [0.29736842, 0.17193676, 0.50802139, 0.62886598, 0.2173913 ,  
 0.27586207, 0.28481013, 0.56603774, 0.36277603, 0.09982935,  
 0.69105691, 0.36263736, 0.15477889],  
 [0.19210526, 0.38339921, 0.8342246 , 0.48453608, 0.35869565,  
 0.26551724, 0.35654008, 0.88679245, 0.20189274, 0.21501706,  
 0.6097561 , 0.45054945, 0.23466476],  
 [0.72368421, 0.39920949, 0.5026738 , 0.58762887, 0.2173913 ,  
 0.12758621, 0.07172996, 0.52830189, 0.1955836 , 0.70819113,  
 0.17886179, 0.15018315, 0.2403709 ],  
 [0.62105263, 0.20355731, 0.67379679, 0.28350515, 0.25 ,  
 0.64482759, 0.54852321, 0.39622642, 0.32807571, 0.3003413 ,  
 0.35772358, 0.71428571, 0.65406562],  
 [0.83947368, 0.18972332, 0.5026738 , 0.29381443, 0.52173913,  
 0.76551724, 0.56118143, 0.24528302, 0.51104101, 0.43515358,  
 0.37398374, 0.74725275, 0.4935806 ],  
 [0.26578947, 0.70355731, 0.54545455, 0.58762887, 0.10869565,  
 0.3862069 , 0.29746835, 0.54716981, 0.29652997, 0.11262799,  
 0.25203252, 0.47619048, 0.21540656],  
 [0.83684211, 0.65217391, 0.57754011, 0.42783505, 0.44565217,  
 0.64482759, 0.48734177, 0.32075472, 0.26498423, 0.33788396,  
 0.31707317, 0.75457875, 0.57203994],  
 [1. , 0.17786561, 0.43315508, 0.17525773, 0.29347826,  
 0.62758621, 0.55696203, 0.30188679, 0.49526814, 0.33447099,  
 0.48780488, 0.57875458, 0.54707561],  
 [0.53157895, 0.20355731, 0.39572193, 0.32989691, 0.40217391,  
 0.69655172, 0.56118143, 0.28301887, 0.51104101, 0.32081911,  
 0.32520325, 0.76190476, 0.43295292],  
 [0.7 , 0.49802372, 0.63101604, 0.48453608, 0.40217391,  
 0.29310345, 0.0464135 , 0.69811321, 0.12302839, 0.39249147,  
 0.3902439 , 0.2014652 , 0.28673324],  
 [0.43684211, 0.15612648, 0.48128342, 0.52061856, 0.10869565,  
 0.13793103, 0.23628692, 0.8490566 , 0.38170347, 0.15102389,  
 0.3902439 , 0.28937729, 0.15477889],  
 [0.15526316, 0.24703557, 0.49197861, 0.3814433 , 0.30434783,  
 0.70344828, 0.40506329, 0.0754717 , 0.29652997, 0.16808874,  
 0.55284553, 0.61904762, 0.04778887],  
 [0.62368421, 0.62648221, 0.59893048, 0.63917526, 0.34782609,  
 0.28275862, 0.08649789, 0.56603774, 0.31545741, 0.51365188,  
 0.17886179, 0.10622711, 0.33666191],  
 [0.65 , 0.21146245, 0.6684492 , 0.48453608, 0.2826087 ,  
 0.53448276, 0.47890295, 0.28301887, 0.39432177, 0.19112628,  
 0.5203252 , 0.93406593, 0.40442225],  
 [0.71052632, 0.15019763, 0.71657754, 0.61340206, 0.33695652,  
 0.69655172, 0.61392405, 0.30188679, 0.6214511 , 0.37713311,  
 0.57723577, 0.52747253, 0.71825963],  
 [0.33157895, 0.41304348, 0.45989305, 0.3814433 , 0.19565217,  
 0.50689655, 0.40295359, 0.22641509, 0.49842271, 0.07423208,  
 0.54471545, 0.74358974, 0.0085592 ],  
 [0.41315789, 0.11857708, 0.28877005, 0.40721649, 0.19565217,  
 0.16206897, 0.21518987, 0.30188679, 0.29652997, 0.09982935,  
 0.45528455, 0.54945055, 0.20256776],  
 [0.1 , 0. , 0.60962567, 0.53608247, 0.19565217,  
 0.51724138, 0.35232068, 0.54716981, 0.32492114, 0.15358362,  
 0.50406504, 0.38095238, 0.11126961],  
 [0.68684211, 0.46640316, 0.64171123, 0.2371134 , 0.5 ,  
 0.59310345, 0.56751055, 0.0754717 , 0.39432177, 0.32593857,  
 0.3902439 , 0.76556777, 0.40442225],  
 [0.41315789, 0.33992095, 0.44919786, 0.40721649, 0.26086957,  
 0.22068966, 0.06751055, 0.94339623, 0.16719243, 0.49658703,  
 0.20325203, 0.11355311, 0.29743224],  
 [0.24473684, 0.06916996, 0.5026738 , 0.53608247, 0.33695652,  
 0.82758621, 0.37974684, 0. , 0.39116719, 0.16467577,  
 0.41463415, 0.68131868, 0.43366619],  
 [0.32105263, 0.62055336, 0.44919786, 0.40721649, 0.45652174,  
 0.13793103, 0.092827 , 0.30188679, 0.23028391, 0.59129693,  
 0.13821138, 0.26739927, 0.41155492],  
 [0.86052632, 0.23320158, 0.72727273, 0.48453608, 0.54347826,  
 0.62758621, 0.5907173 , 0.37735849, 0.49211356, 0.41979522,  
 0.4796748 , 0.50549451, 0.7146933 ],  
 [0.48684211, 0.44466403, 0.55614973, 0.48453608, 0.36956522,  
 0.11034483, 0.18565401, 0.20754717, 0.13249211, 0.35153584,  
 0.21138211, 0.05494505, 0.17974322],  
 [0.64736842, 0.56324111, 0.44385027, 0.45876289, 0.19565217,  
 0.22068966, 0.02953586, 0.8490566 , 0.14826498, 0.37713311,  
 0.26829268, 0.2014652 , 0.21540656],  
 [0.66578947, 0.1916996 , 0.50802139, 0.28865979, 0.51086957,  
 0.74827586, 0.62236287, 0.39622642, 0.60883281, 0.41382253,  
 0.38211382, 0.77289377, 0.36875892],  
 [0.87105263, 0.18577075, 0.71657754, 0.74226804, 0.30434783,  
 0.62758621, 0.20464135, 0.75471698, 0.72239748, 1. ,  
 0.07317073, 0.25274725, 0.2724679 ],  
 [0.70526316, 0.22134387, 0.53475936, 0.30927835, 0.33695652,  
 0.56206897, 0.53586498, 0.26415094, 0.40378549, 0.21501706,  
 0.51219512, 1. , 0.53994294],  
 [0.31315789, 0.10869565, 0.31016043, 0.43298969, 0.23913043,  
 0.47586207, 0.35864979, 0.49056604, 0.52681388, 0.12116041,  
 0.30894309, 0.64102564, 0.02425107],  
 [0.58947368, 0.69960474, 0.48128342, 0.48453608, 0.54347826,  
 0.21034483, 0.07383966, 0.56603774, 0.29652997, 0.76109215,  
 0.08943089, 0.10622711, 0.39728959],  
 [0.56052632, 0.55928854, 0.42245989, 0.53608247, 0.34782609,  
 0.17931034, 0.0443038 , 0.56603774, 0.2807571 , 0.23208191,  
 0.09756098, 0.15018315, 0.39372325],  
 [0.74473684, 0.15217391, 0.70053476, 0.74226804, 0.17391304,  
 0.67931034, 0.53164557, 0.1509434 , 0.46056782, 0.17918089,  
 0.71544715, 0.69230769, 0.09415121],  
 [0.47894737, 0.5 , 0.65240642, 0.58762887, 0.39130435,  
 0.23103448, 0.05485232, 0.88679245, 0.17350158, 0.3668942 ,  
 0.31707317, 0.30769231, 0.20827389],  
 [0.73947368, 0.66798419, 0.54545455, 0.45876289, 0.20652174,  
 0.28275862, 0.10337553, 0.66037736, 0.36277603, 0.65955631,  
 0.07317073, 0.13553114, 0.14407989],  
 [0.59736842, 0.19367589, 0.4171123 , 0.32989691, 0.26086957,  
 0.48965517, 0.39029536, 0.26415094, 0.29652997, 0.2278157 ,  
 0.43902439, 0.54945055, 0.71825963],  
 [0.53157895, 0.61660079, 0.51336898, 0.61340206, 0.16304348,  
 0.23103448, 0.26371308, 0.90566038, 0.38170347, 0.3003413 ,  
 0.29268293, 0.27106227, 0.16904422],  
 [0.38947368, 0.09881423, 0.47593583, 0.3556701 , 0.16304348,  
 0.35172414, 0.05063291, 0.88679245, 0.26498423, 0.35580205,  
 0.2195122 , 0.08791209, 0.26533524],  
 [0.33157895, 0.13241107, 0.3315508 , 0.27835052, 0.16304348,  
 0.54137931, 0.4556962 , 0.30188679, 0.42902208, 0.13822526,  
 0.6097561 , 0.53846154, 0.10699001],  
 [0.45789474, 0.53162055, 0.3315508 , 0.27835052, 0.10869565,  
 0.22413793, 0.19198312, 0.56603774, 0.13249211, 0.18088737,  
 0.17886179, 0.31135531, 0.06704708],  
 [0.50789474, 0.53557312, 0.52941176, 0.40721649, 0.39130435,  
 0.14137931, 0.07594937, 0.50943396, 0.16719243, 0.34129693,  
 0.16260163, 0.17582418, 0.2831669 ],  
 [0.69473684, 0.10079051, 0.29946524, 0.3814433 , 0.26086957,  
 0.3862069 , 0.30590717, 0.35849057, 0.10094637, 0.21501706,  
 0.6097561 , 0.43589744, 0.2510699 ],  
 [0.56052632, 0.3201581 , 0.70053476, 0.41237113, 0.33695652,  
 0.62758621, 0.61181435, 0.32075472, 0.75709779, 0.37542662,  
 0.44715447, 0.6959707 , 0.64693295],  
 [0.35263158, 0.03952569, 0. , 0. , 0.19565217,  
 0.34482759, 0.04852321, 0.28301887, 0.00315457, 0.05716724,  
 0.46341463, 0.2014652 , 0.17261056],  
 [0.37894737, 0.1541502 , 0.44919786, 0.43298969, 1. ,  
 0.52413793, 0.407173 , 0.35849057, 0.90536278, 0.11262799,  
 0.55284553, 0.4981685 , 0.4700428 ],  
 [0.20526316, 0.27272727, 0.73796791, 0.56185567, 0.69565217,  
 0.2137931 , 0.1371308 , 0.01886792, 0.36277603, 0.10409556,  
 0.38211382, 0.36263736, 0.24750357]]), 54 0.0  
151 2.0  
63 1.0  
55 0.0  
123 1.0  
 ...   
62 1.0  
2 0.0  
59 1.0  
95 1.0  
96 1.0  
Name: target, Length: 72, dtype: float64)

### Бэггинг

# Обучим классификатор на 5 деревьях  
bc = BaggingClassifier(n\_estimators=5, oob\_score=True, random\_state=10)  
bc\_prediction = bc.fit(x\_train, y\_train).predict(x\_test)

/Library/Frameworks/Python.framework/Versions/3.9/lib/python3.9/site-packages/sklearn/ensemble/\_bagging.py:706: UserWarning: Some inputs do not have OOB scores. This probably means too few estimators were used to compute any reliable oob estimates.  
 warn(  
/Library/Frameworks/Python.framework/Versions/3.9/lib/python3.9/site-packages/sklearn/ensemble/\_bagging.py:712: RuntimeWarning: invalid value encountered in divide  
 oob\_decision\_function = predictions / predictions.sum(axis=1)[:, np.newaxis]

### Градиентный бустинг

gb = GradientBoostingClassifier(random\_state=0)  
gb\_prediction = gb.fit(x\_train, y\_train).predict(x\_test)

### Стекинг

# Качество отдельных моделей  
def val\_mae(model):  
 st\_prediction = model.fit(x\_train, y\_train).predict(x\_test)  
# y\_pred = model.predict(boston\_X\_test)  
# result = mean\_absolute\_error(boston\_y\_test, y\_pred)  
 print(model)  
 print('Accuracy score={}'.format(accuracy\_score(y\_test, st\_prediction)))

# Точность на отдельных моделях  
for model in [  
 DecisionTreeClassifier(random\_state=0),  
 svm.SVC(random\_state=0),  
 LogisticRegression(random\_state=0)  
]:  
 val\_mae(model)  
 print('==========================')  
 print()

DecisionTreeClassifier(random\_state=0)  
Accuracy score=0.9166666666666666  
==========================  
  
SVC(random\_state=0)  
Accuracy score=0.9722222222222222  
==========================  
  
LogisticRegression(random\_state=0)  
Accuracy score=0.9583333333333334  
==========================

# Первый уровень - две модели: дерево и метод опорных векторов  
# Второй уровень: логистическая регрессия  
  
estimators = [  
 ('dt', DecisionTreeClassifier(random\_state=0)),  
 ('svc', svm.SVC(random\_state=0))  
]  
  
sc = StackingClassifier(  
 estimators=estimators, final\_estimator=LogisticRegression()  
)  
sc\_prediction = sc.fit(x\_train, y\_train).predict(x\_test)

### Многослойный персептрон

mlp = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden\_layer\_sizes=(5, 2), random\_state=1)  
mlp\_prediction = mlp.fit(x\_train, y\_train).predict(x\_test)

### Модель МГУА (метод группового учёта аргументов)

# mgua = Classifier()  
# mgua\_prediction = mgua.fit(x\_train, y\_train).predict(x\_test)  
# accuracy\_score(y\_test, mgua\_prediction)

### Оценка качества решений

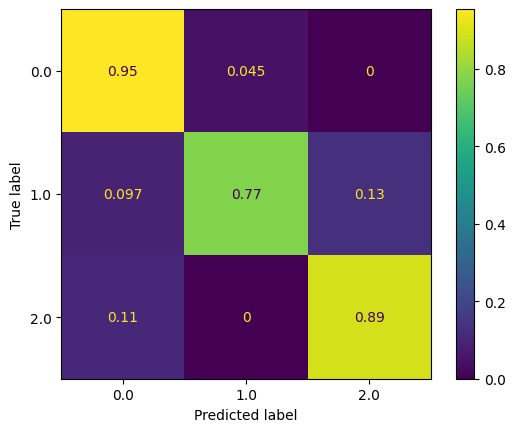
print("Бэггинг: ", accuracy\_score(y\_test, bc\_prediction))  
print("Градиентный бустинг: ", accuracy\_score(y\_test, gb\_prediction))  
print("Стекинг (дерево и метод опорных векторов + логистическая регрессия): ", accuracy\_score(y\_test, sc\_prediction))  
print("Многослойный персептрон: ", accuracy\_score(y\_test, mlp\_prediction))

Бэггинг: 0.8611111111111112  
Градиентный бустинг: 0.9305555555555556  
Стекинг (дерево и метод опорных векторов + логистическая регрессия): 0.9722222222222222  
Многослойный персептрон: 0.9305555555555556

print("Бэггинг: ", accuracy\_score(y\_test, bc\_prediction))  
  
cm = confusion\_matrix(y\_test, bc\_prediction, labels=np.unique(df.target), normalize='true')  
disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=np.unique(df.target))  
disp.plot()

Бэггинг: 0.8611111111111112

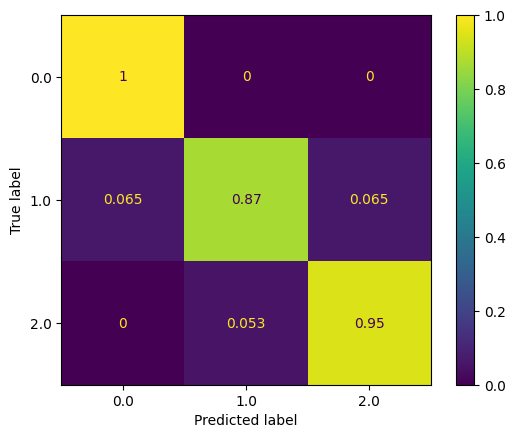
<sklearn.metrics.\_plot.confusion\_matrix.ConfusionMatrixDisplay at 0x7fcd33718910>



print("Градиентный бустинг: ", accuracy\_score(y\_test, gb\_prediction))  
  
cm = confusion\_matrix(y\_test, gb\_prediction, labels=np.unique(df.target), normalize='true')  
disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=np.unique(df.target))  
disp.plot()

Градиентный бустинг: 0.9305555555555556

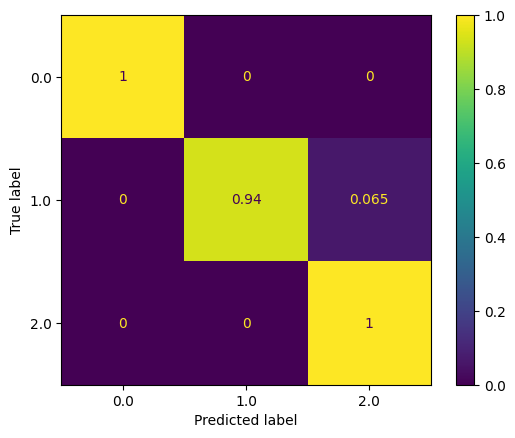
<sklearn.metrics.\_plot.confusion\_matrix.ConfusionMatrixDisplay at 0x7fcd00579dc0>



print("Стекинг (дерево и метод опорных векторов + логистическая регрессия): ", accuracy\_score(y\_test, sc\_prediction))  
  
cm = confusion\_matrix(y\_test, sc\_prediction, labels=np.unique(df.target), normalize='true')  
disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=np.unique(df.target))  
disp.plot()

Стекинг (дерево и метод опорных векторов + логистическая регрессия): 0.9722222222222222

<sklearn.metrics.\_plot.confusion\_matrix.ConfusionMatrixDisplay at 0x7fcd33718370>



print("Многослойный персептрон: ", accuracy\_score(y\_test, mlp\_prediction))  
  
cm = confusion\_matrix(y\_test, mlp\_prediction, labels=np.unique(df.target), normalize='true')  
disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=np.unique(df.target))  
disp.plot()

Многослойный персептрон: 0.9305555555555556

<sklearn.metrics.\_plot.confusion\_matrix.ConfusionMatrixDisplay at 0x7fcd11888d60>

