Лабораторная работа № 5

Ансамбли 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 две ансамблевые модели. Оцените качество моделей с помощью одной из подходящих для задачи метрик. Сравните качество полученных моделей.

Выполнение:

Импорт библиотек

In [25]:

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_wine
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from operator import itemgetter
from sklearn.metrics import plot_confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

Загрузка данных

In [2]:

```
wine = load_wine()
data = pd.DataFrame(data=np.c_[wine['data'], wine['target']], columns = wine['feature_names']+['target'])
```

In [3]:

```
data.head()
```

Out[3]:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	h
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.

In [4]:

```
data.shape
```

Out[4]:

```
(178, 14)
```

In [5]:

```
data.isnull().sum()
```

Out[5]:

```
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
dtype: int64
```

В данном датасете нет пропусков и категориальных признаков.

Разделение выборки на обучающую и тестовую

In [6]:

```
# С использованием метода train_test_split разделим выборку
X_train, X_test, Y_train, Y_test = train_test_split(wine.data, wine.target, test_size=0.3, random_state=1
```

In [7]:

```
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

Out[7]:

```
((124, 13), (54, 13), (124,), (54,))
```

Ансамблевые модели

In [9]:

```
# Случайный лес
tree = RandomForestClassifier(n_estimators=5, oob_score=True, random_state=10)
tree.fit(X_train, Y_train)
```

```
/Users/kalashnikova/opt/anaconda3/envs/myenv/lib/python3.8/site-
packages/sklearn/ensemble/_forest.py:541: UserWarning: Some inputs do not have OOB scores. This probably
means too few trees were used to compute any reliable oob estimates.
  warn("Some inputs do not have OOB scores. "
/Users/kalashnikova/opt/anaconda3/envs/myenv/lib/python3.8/site-
packages/sklearn/ensemble/_forest.py:545: RuntimeWarning: invalid value encountered in true_divide
  decision = (predictions[k] /
```

Out[9]:

```
RandomForestClassifier(n_estimators=5, oob_score=True, random_state=10)
```

In [10]:

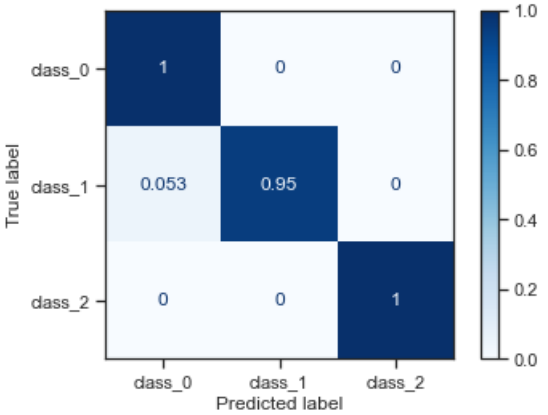
```
y_pred1 = tree.predict(X_test)
```

In [26]:

```
# Для оценки качества моделей будем использовать матрицу ошибок:
plot_confusion_matrix(tree, X_test, Y_test,
                      display_labels=wine.target_names,
                      cmap=plt.cm.Blues, normalize='true')
```

Out[26]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7ff3e2a4ac70>
```



In [12]:

```
# Градиентный бустинг
model_boost = GradientBoostingClassifier(random_state=1)
```

```
model_boost.fit(X_train, Y_train)
```

Out[12]:

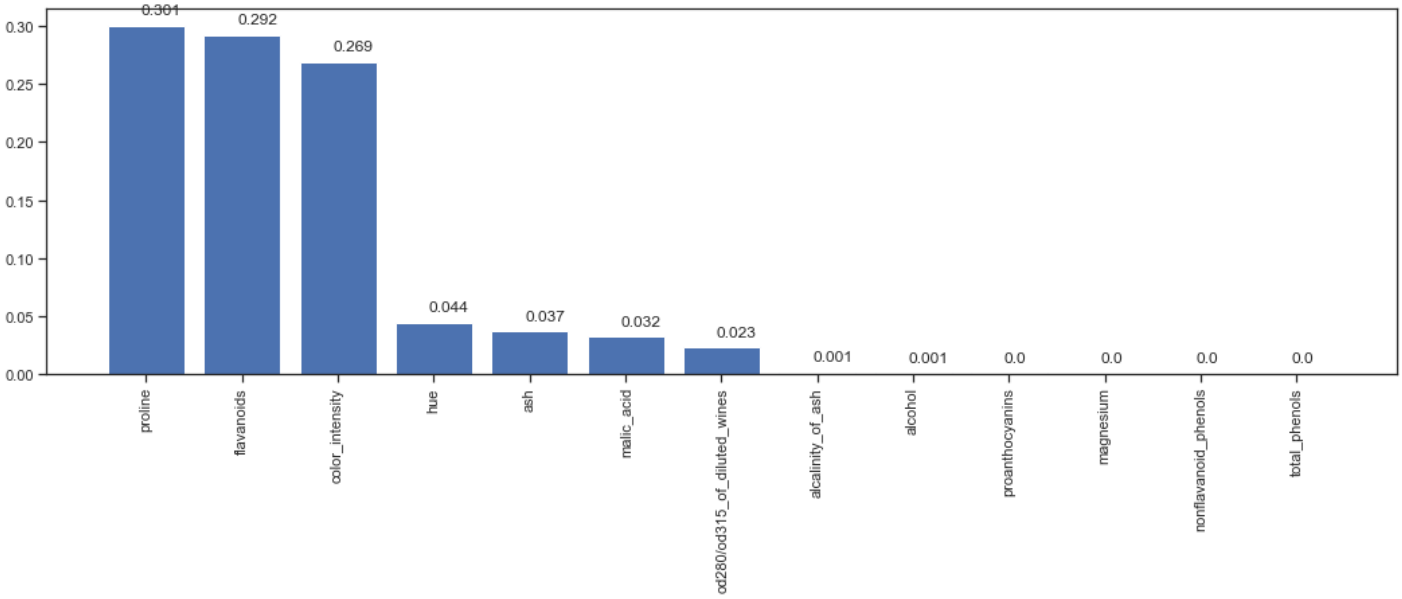
```
GradientBoostingClassifier(random_state=1)
```

In [15]:

```
# Изобразим важность признаков
def draw_feature_importances(tree_model, X_dataset, figsize=(18,5)):
    """
    Вывод важности признаков в виде графика
    """
    # Сортировка значений важности признаков по убыванию
    list_to_sort = list(zip(X_dataset.columns.values, tree_model.feature_importances_))
    sorted_list = sorted(list_to_sort, key=itemgetter(1), reverse = True)
    # Названия признаков
    labels = [x for x,_ in sorted_list]
    # Важности признаков
    data = [x for _,x in sorted_list]
    # Вывод графика
    fig, ax = plt.subplots(figsize=figsize)
    ind = np.arange(len(labels))
    plt.bar(ind, data)
    plt.xticks(ind, labels, rotation='vertical')
    # Вывод значений
    for a,b in zip(ind, data):
        plt.text(a-0.05, b+0.01, str(round(b,3)))
    plt.show()
    return labels, data
```

In [18]:

```
tree_fl, tree_fd = draw_feature_importances(model_boost, data)
```

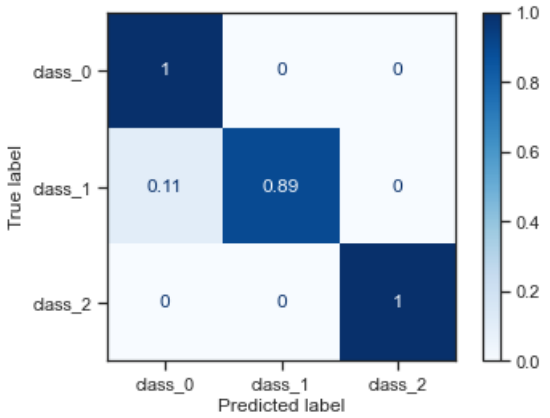


In [27]:

```
plot_confusion_matrix(model_boost, X_test, Y_test,
                      display_labels=wine.target_names,
                      cmap=plt.cm.Blues, normalize='true')
```

Out[27]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7ff3e30fe040>
```



По матрицам ошибок видно, что на данном датасете лучше работает случайный лес, т.к. в градиентном бустинге больше ошибок в первом классе.

In [ ]: