



Министерство образования и науки Российской Федерации
**МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ им. Н.Э. БАУМАНА**

**Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления» (ИУ5)**

ДИСЦИПЛИНА: «Технологии машинного обучения»

**Отчет по лабораторной работе №6
«Ансамбли моделей машинного обучения»**

Выполнил:

Студент группы ИУ5-61Б

Кочетков М.Д.

Преподаватель:

Гапанюк Ю.Е.

Москва, 2020 г.

Цель лабораторной работы: изучение ансамблей моделей машинного обучения.

Задание:

1. Выберите набор данных (датасет) для решения задачи классификации или регрессии.
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.

Выполненная работа:

В данной лабораторной работе модели будут строиться для решения задачи классификации. Загрузка и первичный анализ данных. Формирование `DataFrame`:

```
In [2]: wine = load_wine()

In [3]: wine_x_ds = pd.DataFrame(data=wine['data'], columns=wine['feature_names'])

In [4]: from operator import itemgetter

def draw_feature_importances(tree_model, X_dataset, figsize=(15,7)):
    """
    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
```

Разделение данных на обучающую и тестовую выборки. Построение ансамблевой модели «Случайный лес»:

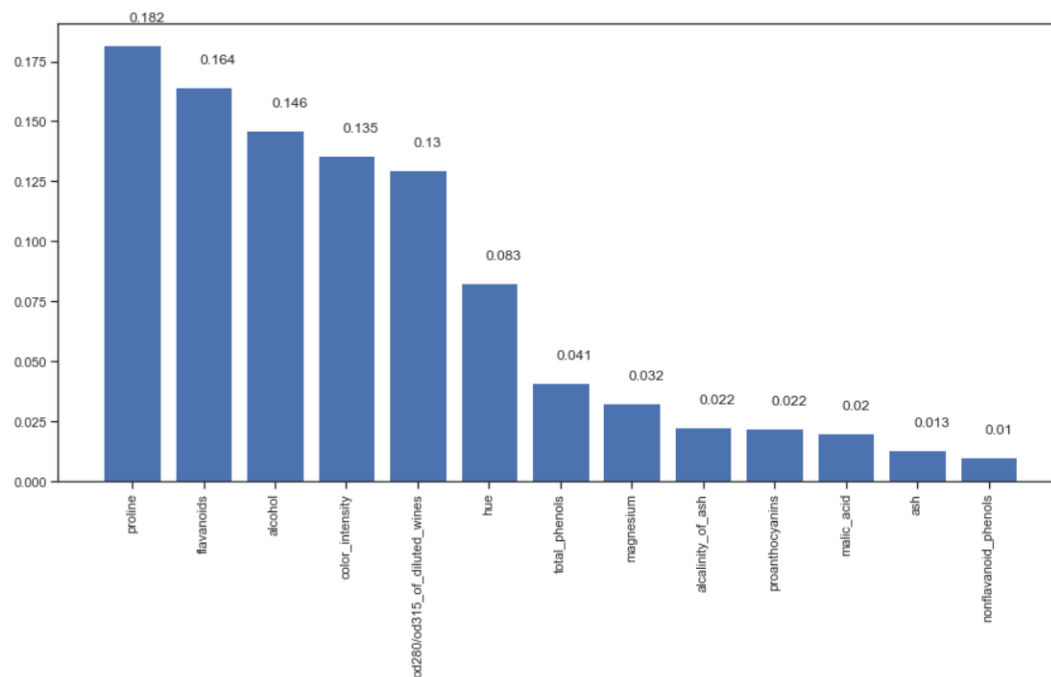
```
In [5]: wine_X_train, wine_X_test, wine_y_train, wine_y_test = train_test_split(
        wine.data, wine.target, test_size=0.15, random_state=1)

In [6]: rf = RandomForestClassifier(n_estimators=1, max_depth=2, random_state=3).fit(wine_X_train, wine_y_train)
        target_rf = rf.predict(wine_X_test)

In [7]: accuracy_score(wine_y_test, target_rf)

Out[7]: 0.8148148148148148

In [8]: # Важность признаков
        wine_rf_cl = RandomForestClassifier(random_state=1)
        wine_rf_cl.fit(wine_x_ds, wine.target)
        _, _ = draw_feature_importances(wine_rf_cl, wine_x_ds)
```



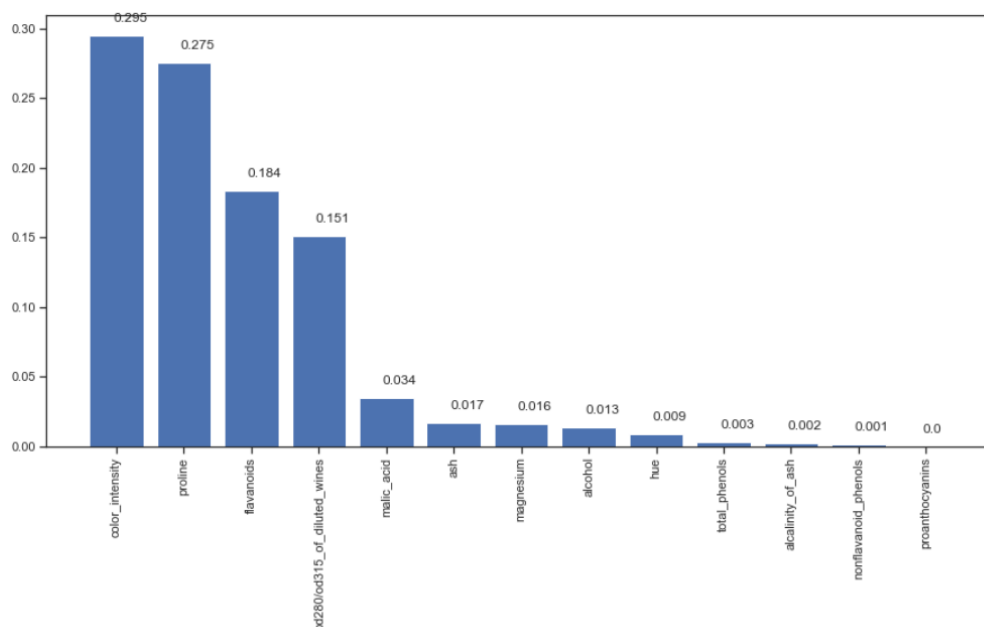
Построение ансамблевой модели «Градиентный бустинг»:

```
In [9]: gb = GradientBoostingClassifier(n_estimators=5, max_depth=2, learning_rate=0.05).fit(wine_X_train, wine_y_train)
        target_gb = gb.predict(wine_X_test)
```

```
In [10]: accuracy_score(wine_y_test, target_gb)
```

```
Out[10]: 0.9259259259259259
```

```
In [11]: # Важность признаков
        wine_gb_cl = GradientBoostingClassifier(random_state=1)
        wine_gb_cl.fit(wine_x_ds, wine.target)
        _ = draw_feature_importances(wine_gb_cl, wine_x_ds)
```



Таким образом, исходя из результатов метрики Ассигасу можно сделать вывод о высоком качестве всех построенных моделей.