Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Московский государственный технический университет имени Н.Э. Баумана»

(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления (ИУ5)

О т ч е т

по лабораторной работе №5

Ансамбли моделей машинного обучения

Дисциплина: Технологии машинного обучения

Студент гр. ИУ5-63Б _____ Терентьев В.О.
(Подпись, дата) (Фамилия И.О.)

Преподаватель _____ Гапанюк Ю.Е.
(Подпись, дата) (Фамилия И.О.)

Москва, 2021

1. Цель работы

Цель лабораторной работы: изучение ансамблей моделей машинного обучения.

2. Описание задания

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.

3. Основная часть

Лабораторная работа №5

Ансамбли моделей машинного обучения

Импорт библиотек:

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier, ExtraTreesClassifier
from sklearn.ensemble import GradientBoostingClassifier
from catboost import CatBoostClassifier
from lightgbm import LGBMClassifier
from xgboost import XGBClassifier
#from gmdhpy import gmdh
from sklearn.metrics import roc_auc_score
```

In [2]:

```
def vis_models_quality(array_metric, array_labels, str_header, figsize=(5, 5)):
    fig, ax1 = plt.subplots(figsize=figsize)
    pos = np.arange(len(array_metric))
    rects = ax1.barh(pos, array_metric,
                     align='center',
                     height=0.5,
                     tick_label=array_labels)
    ax1.set_title(str_header)
    for a,b in zip(pos, array_metric):
        plt.text(0.2, a-0.1, str(round(b,3)), color='white')
    plt.show()
```

Загрузка данных:

В качестве набора данных используется готовый набор данных из лабораторной работы №3 [IEEE-CIS Fraud Detection (https://www.kaggle.com/c/ieee-fraud-detection/)](https://www.kaggle.com/c/ieee-fraud-detection/) в котором уже произведена обработка пропусков в данных, кодирование категориальных признаков и масштабирование данных.

In [3]:

```
data = pd.read_csv('D:/Загрузки/IEEE-CIS Fraud Detection/newdone_train.csv', sep=",")
data.drop(['Unnamed: 0'], inplace=True, axis=1)
```

Разделение выборки на обучающую и тестовую:

С использованием метода train_test_split.

In [4]:

```
data_train, data_test, data_y_train, data_y_test = train_test_split(data[data.columns.drop(
del data
```

Обучение моделей:

Модель "Случайный лес"

In [5]:

```
RF = RandomForestClassifier(random_state=1)
RF.fit(data_train, data_y_train)
```

Out[5]:

```
RandomForestClassifier(random_state=1)
```

In [6]:

```
data_test_predicted = {}
data_test_predicted["RF"] = roc_auc_score(data_y_test, RF.predict_proba(data_test)[:,1])
```

Модель "Сверхслучайные деревья"

In [7]:

```
ET = ExtraTreesClassifier(random_state=1)
ET.fit(data_train, data_y_train)
data_test_predicted["ET"] = roc_auc_score(data_y_test, ET.predict_proba(data_test)[:,1])
```

Модели "Градиентный бустинг"

In [8]:

```
GB = GradientBoostingClassifier(random_state=1)
GB.fit(data_train, data_y_train)
data_test_predicted["GB"] = roc_auc_score(data_y_test, GB.predict_proba(data_test)[:,1])
```

In [9]:

```
GB_CB = CatBoostClassifier()
GB_CB.fit(data_train, data_y_train)
data_test_predicted["GB_CB"] = roc_auc_score(data_y_test, GB_CB.predict_proba(data_test)[:,
```

```
Learning rate set to 0.138966
0:      learn: 0.4908057        total: 285ms    remaining: 4m 44s
1:      learn: 0.3625994        total: 403ms    remaining: 3m 21s
2:      learn: 0.2744751        total: 547ms    remaining: 3m 1s
3:      learn: 0.2221522        total: 730ms    remaining: 3m 1s
4:      learn: 0.1877148        total: 961ms    remaining: 3m 11s
5:      learn: 0.1634522        total: 1.18s    remaining: 3m 14s
6:      learn: 0.1467737        total: 1.41s    remaining: 3m 20s
7:      learn: 0.1369372        total: 1.61s    remaining: 3m 20s
8:      learn: 0.1295377        total: 1.84s    remaining: 3m 23s
9:      learn: 0.1241591        total: 2.09s    remaining: 3m 26s
10:     learn: 0.1201690        total: 2.33s    remaining: 3m 29s
11:     learn: 0.1163576        total: 2.52s    remaining: 3m 27s
12:     learn: 0.1139235        total: 2.73s    remaining: 3m 27s
13:     learn: 0.1115860        total: 2.9s     remaining: 3m 24s
14:     learn: 0.1096261        total: 3.15s    remaining: 3m 26s
15:     learn: 0.1087205        total: 3.28s    remaining: 3m 21s
16:     learn: 0.1077087        total: 3.44s    remaining: 3m 19s
17:     learn: 0.1064933        total: 3.65s    remaining: 3m 19s
```

In [10]:

```
LGBM = LGBMClassifier()
LGBM.fit(data_train, data_y_train)
data_test_predicted["LGBM"] = roc_auc_score(data_y_test, LGBM.predict_proba(data_test)[:,1]
```

In [11]:

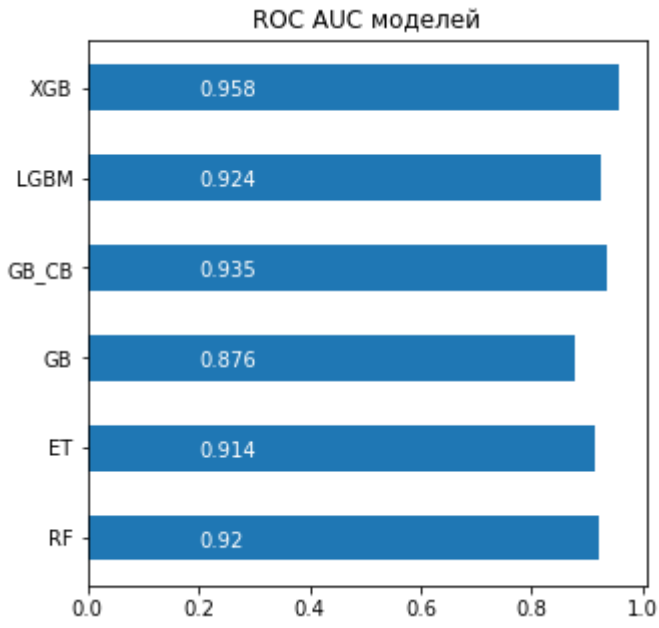
```
XGB = XGBClassifier(n_estimators=500,
                    max_depth=9,
                    learning_rate=0.05,
                    subsample=0.9,
                    colsample_bytree=0.9,
                    tree_method='gpu_hist')
XGB.fit(data_train, data_y_train)
data_test_predicted["XGB"] = roc_auc_score(data_y_test, XGB.predict_proba(data_test)[:,1])
```

```
D:\ProgramData\Anaconda3\lib\site-packages\xgboost\sklearn.py:1146: UserWarn
ing: The use of label encoder in XGBClassifier is deprecated and will be rem
oved in a future release. To remove this warning, do the following: 1) Pass
option use_label_encoder=False when constructing XGBClassifier object; and
2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [n
um_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)

[17:22:24] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_
1.4.0/src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation
metric used with the objective 'binary:logistic' was changed from 'error' to
'logloss'. Explicitly set eval_metric if you'd like to restore the old behav
ior.

D:\ProgramData\Anaconda3\lib\site-packages\xgboost\data.py:112: UserWarning:
Use subset (sliced data) of np.ndarray is not recommended because it will ge
nerate extra copies and increase memory consumption
  warnings.warn(
```

Сравнение качества полученных моделей:

In [14]:

```
vis_models_quality(data_test_predicted.values(), list(data_test_predicted.keys()), 'ROC AUC
```



In [ ]: