**Міністерство освіти і науки України**

**Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"**

**Факультет інформатики та обчислювальної техніки**

**Кафедра інформатики та програмної інженерії**

**Звіт**

з лабораторної роботи №3 з дисципліни

«Програмування інтелектуальних інформаційних систем»

„Random Forest”

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-11 Головня Олександр Ростиславович*

**Перевірив**

(прізвище, ім'я, по батькові)

*Баришич Л. М.*

Київ 2023

**Завдання**1. Пройти тутор:

<https://www.kaggle.com/code/jhoward/linear-model-and-neural-net-from-scratch#Deep-learning>

2. Побудувати рендом форест звідси:

[**https://www.kaggle.com/code/jhoward/how-random-forests-really-work/**](https://www.kaggle.com/code/jhoward/how-random-forests-really-work/)

2.1. Натрейнити на датасеті звідси**: '/kaggle/input/car-evaluation-data-set/car\_evaluation.csv'**

**Class -** залежна змінна

**Важливо! Незабудьте енкодер**

**encoder = ce.OrdinalEncoder(cols=['buying', 'maint', 'doors', 'persons', 'lug\_boot', 'safety'])**

2.2 Вивести **confusion matrix, auc, Classification report**

3 Зробити буст попередньої моделі XGBoost. Порівняти результати <https://machinelearningmastery.com/random-forest-ensembles-with-xgboost/>

N.B.: catboost російський, тому того, хто спробує здати буст кетбустом, буде внесено в базу “Миротворець”

**Виконання:**

import pandas as pd

#!pip install category\_encoders

from sklearn.model\_selection import train\_test\_split

import category\_encoders as ce

# Load dataset

df = pd.read\_csv('car\_evaluation.csv', header=None)

df.columns = ['buying', 'meant', 'doors', 'persons', 'lug\_boot', 'safety', 'class']

X = df.drop(['class'], axis=1)

y = df['class']

# Split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25)

# Encode

encoder = ce.OrdinalEncoder(cols=['buying', 'meant', 'doors', 'persons', 'lug\_boot', 'safety'])

X\_train = encoder.fit\_transform(X\_train)

X\_test = encoder.transform(X\_test)

print('X\_train.head():', X\_train.head(), sep='\n', end='\n\n')

print('X\_test.head():', X\_test.head(), sep='\n', end='\n')

from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()

rf.fit(X\_train, y\_train)

y\_pred = rf.predict(X\_test)

y\_pred[:5]

from xgboost import XGBRFClassifier

from sklearn.preprocessing import LabelEncoder

label\_encoder = LabelEncoder()

boosted = XGBRFClassifier()

boosted.fit(X\_train, label\_encoder.fit\_transform(y\_train))

y\_pred\_boosted = label\_encoder.inverse\_transform(boosted.predict(X\_test))

y\_pred\_boosted[:5]

from sklearn.metrics import confusion\_matrix, roc\_auc\_score, classification\_report

from sklearn.preprocessing import label\_binarize

import seaborn as sns

import matplotlib.pyplot as plt

plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)

sns.heatmap(confusion\_matrix(y\_test, y\_pred), annot=True, fmt='d', cmap='Blues', cbar=False)

plt.title('Confusion Matrix - Random Forest')

plt.xlabel('Predicted')

plt.ylabel('True')

plt.subplot(1, 2, 2)

sns.heatmap(confusion\_matrix(y\_test, y\_pred\_boosted), annot=True, fmt='d', cmap='Greens', cbar=False)

plt.title('Confusion Matrix - XGBoost')

plt.xlabel('Predicted')

plt.ylabel('True')

plt.tight\_layout()

plt.show()

class\_labels = ['unacc', 'acc', 'good', 'vgood']

y\_test\_encoded = label\_binarize(y\_test, classes=class\_labels)

y\_pred\_encoded = label\_binarize(y\_pred, classes=class\_labels)

y\_pred\_boosted\_encoded = label\_binarize(y\_pred\_boosted, classes=class\_labels)

roc\_auc\_scores = [roc\_auc\_score(y\_test\_encoded[:, i], y\_pred\_encoded[:, i]) for i in range(len(class\_labels))]

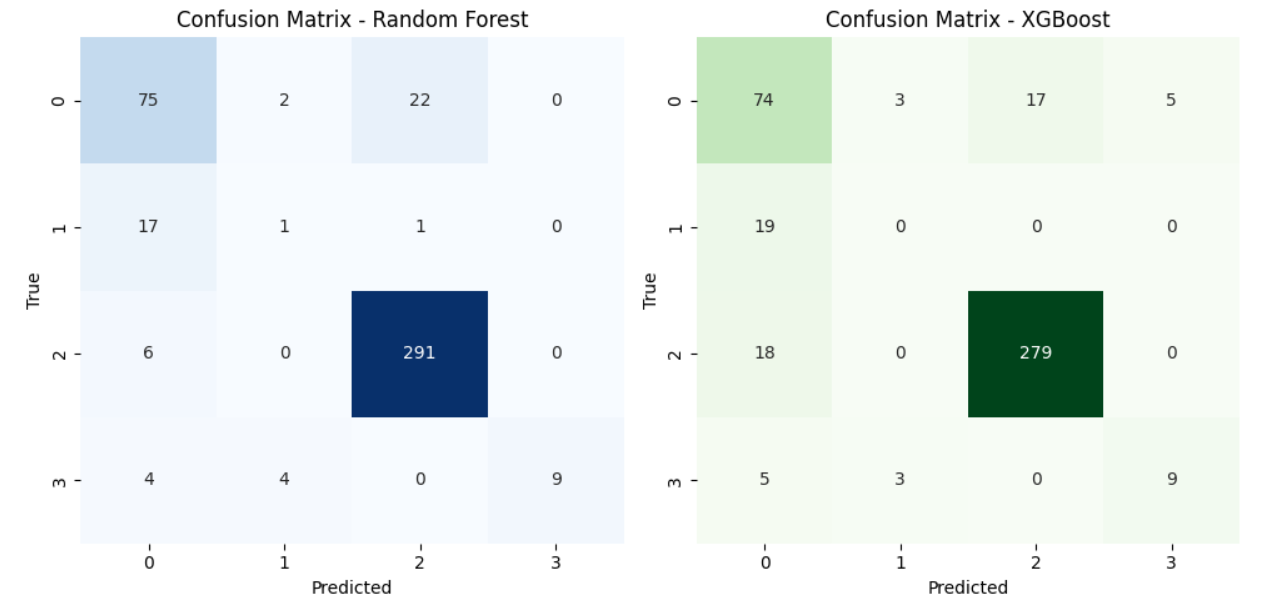
roc\_auc\_scores\_boosted = [roc\_auc\_score(y\_test\_encoded[:, i], y\_pred\_boosted\_encoded[:, i]) for i in range(len(class\_labels))]

print('\nROC AUC scores:\n', pd.DataFrame({'Random Forest':roc\_auc\_scores, 'XGBoost':roc\_auc\_scores\_boosted}).set\_index(pd.Index(class\_labels)), sep='')

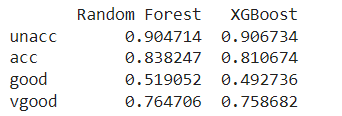
print('\n\nClassification report (Random Forest):\n', classification\_report(y\_test, y\_pred))

print('\nClassification report (XGBoost):\n', classification\_report(y\_test, y\_pred\_boosted))

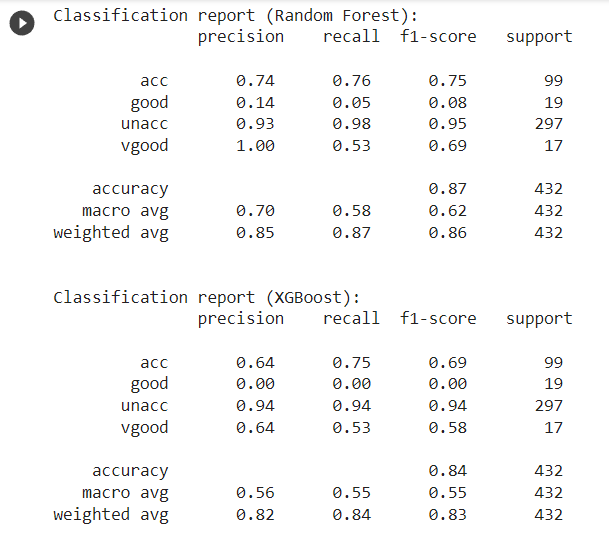
**Confusion Matrix**

****

**Area Under the Curve**

****

**Classification report**

****