Додаток А

Текст програми

```
import tensorflow as tf
import numpy as np
import pandas as pd
from tensorflow import keras
from tensorflow.keras import layers
import coremltools
import tfcoreml


file_path = "data/heart_disease.csv"
dataframe = pd.read_csv(file_path)

dataframe.shape
dataframe.head()

validation_df = dataframe.sample(frac=0.2, random_state=1337)
train_df = dataframe.drop(validation_df.index)

print("Train samples: %d\n Validation samples: %d"
      % (len(train_df), len(validation_df)))


def df_to_ds(df):
    frame = df.copy()
    labels = frame.pop("target")
    set = tf.data.Dataset.from_tensor_slices((dict(frame), labels))
    set = ds.shuffle(buffer_size=len(frame))
    return set


train_ds = df_to_ds(train_df)
validation_ds = df_to_ds(validation_df)

for x, y in train_ds.take(1):
    print("Input:", x)
    print("Target:", y)

train_ds = train_ds.batch(32)
val_ds = validation_ds.batch(32)

from tensorflow.keras.layers.experimental.preprocessing import Normalization
from tensorflow.keras.layers.experimental.preprocessing import CategoryEncoding
from tensorflow.keras.layers.experimental.preprocessing import StringLookup


def encode_numerical(feature, name, dataset):
    norm_layer = Normalization()

    ds = dataset.map(lambda x, y: x[name])
    ds = ds.map(lambda x: tf.expand_dims(x, -1))

    norm_layer.adapt(ds)

    new_feature = norm_layer(feature)
    return new_feature


def encode_string(feature, name, dataset):
    index = StringLookup()
```

```
    ds = dataset.map(lambda x, y: x[name])
    ds = ds.map(lambda x: tf.expand_dims(x, -1))

    index.adapt(ds)

    new_feature = index(feature)

    encoder = CategoryEncoding(output_mode="binary")

    ds = ds.map(index)

    encoder.adapt(ds)

    encoded_feature = encoder(new_feature)
    return encoded_feature


def encode_integer(feature, name, dataset):
    encoder = CategoryEncoding(output_mode="binary")

    ds = dataset.map(lambda x, y: x[name])
    ds = ds.map(lambda x: tf.expand_dims(x, -1))

    encoder.adapt(ds)

    new_feature = encoder(feature)
    return new_feature


sex = keras.Input(shape=(1,), name="sex", dtype="int64")
cp = keras.Input(shape=(1,), name="cp", dtype="int64")
fbs = keras.Input(shape=(1,), name="fbs", dtype="int64")
restecg = keras.Input(shape=(1,), name="restecg", dtype="int64")
exang = keras.Input(shape=(1,), name="exang", dtype="int64")
ca = keras.Input(shape=(1,), name="ca", dtype="int64")

thal = keras.Input(shape=(1,), name="thal", dtype="string")

age = keras.Input(shape=(1,), name="age")
trestbps = keras.Input(shape=(1,), name="trestbps")
chol = keras.Input(shape=(1,), name="chol")
thalach = keras.Input(shape=(1,), name="thalach")
oldpeak = keras.Input(shape=(1,), name="oldpeak")
slope = keras.Input(shape=(1,), name="slope")

all_inputs = [
    sex,
    cp,
    fbs,
    restecg,
    exang,
    ca,
    thal,
    age,
    trestbps,
    chol,
    thalach,
    oldpeak,
    slope,
]

# Integer categorical features
```

```
sex_encoded = encode_integer(sex, "sex", train_ds)
cp_encoded = encode_integer(cp, "cp", train_ds)
fbs_encoded = encode_integer(fbs, "fbs", train_ds)
restecg_encoded = encode_integer(restecg, "restecg", train_ds)
exang_encoded = encode_integer(exang, "exang", train_ds)
ca_encoded = encode_integer(ca, "ca", train_ds)

# String categorical features
thal_encoded = encode_string(thal, "thal", train_ds)

# Numerical features
age_encoded = encode_numerical(age, "age", train_ds)
trestbps_encoded = encode_numerical(trestbps, "trestbps", train_ds)
chol_encoded = encode_numerical(chol, "chol", train_ds)
thalach_encoded = encode_numerical(thalach, "thalach", train_ds)
oldpeak_encoded = encode_numerical(oldpeak, "oldpeak", train_ds)
slope_encoded = encode_numerical(slope, "slope", train_ds)

all_features = layers.concatenate(
    [
        sex_encoded,
        cp_encoded,
        fbs_encoded,
        restecg_encoded,
        exang_encoded,
        slope_encoded,
        ca_encoded,
        thal_encoded,
        age_encoded,
        trestbps_encoded,
        chol_encoded,
        thalach_encoded,
        oldpeak_encoded,
    ]
)
x = layers.Dense(32, activation="relu")(all_features)
x = layers.Dense(32, activation="relu")(x)
x = layers.Dense(32, activation="relu")(x)
x = layers.Dense(32, activation="relu")(x)
x = layers.Dense(32, activation="relu")(x)
output = layers.Dense(1, activation="sigmoid")(x)
model = keras.Model(all_inputs, output)
model.compile("rmsprop", "binary_crossentropy", metrics=["accuracy"])

import matplotlib.pyplot as plt

callbacks_list = [
    keras.callbacks.ModelCheckpoint(
        filepath='best_model.{epoch:02d}-{val_loss:.2f}.tf',
        monitor='val_loss', save_best_only=True),
    keras.callbacks.EarlyStopping(monitor='acc', patience=1)
]

history = model.fit(train_ds, epochs=30, validation_data=val_ds,
callbacks=callbacks_list)

# summarize history for accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
```

```
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

sample = {
    "age": 26,
    "sex": 1,
    "cp": 1,
    "trestbps": 155,
    "chol": 266,
    "fbs": 1,
    "restecg": 2,
    "thalach": 155,
    "exang": 0,
    "oldpeak": 1.6,
    "slope": 2,
    "ca": 0,
    "thal": "normal",
}

input_dict = {name: tf.convert_to_tensor([value]) for name, value in sample.items()}
predictions = model.predict(input_dict)

print(
    "This particular patient had a %.1f percent probability "
    "of having a heart disease, as evaluated by our model.
    % (100 * predictions[0][0],)
)
```

ДОДАТОК Б

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	93 с.
	Інші документи	
2.	Презентаційні матеріали	__ плакатів

.	.	Прізвище та ініціали.	Підп.	Дата			
Розробив		Скорик І.С.			Розробка і дослідження застосування нейронних мереж при обробці і аналізі медичних даних	Шифр групи	Код напр./спец.
Перевірив		Філатов В.О.				СШІм-19-21	122
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ	
Затв.		Філатов В.О.					