DAFTAR PUSTAKA

Amin, A., Anwar, S., Adnan, A., Nawaz, M., Howard, N., Qadir, J., & Hawalah, A. (2016). Comparing Oversampling Techniques to Handle the Class Imbalance Problem: A Customer Churn Prediction Case Study. *IEEE Access, 4*, 7940-7957.

Baily, J. (2020, May 27). *HealthTechZone*. Retrieved July 21, 2020, from HealthTechZone: https://www.healthtechzone.com/topics/healthcare/articles/2020/05/27/445511-how-iot-solving-pain-points-healthcare.html

Baker, S. B., Xiang, W., & Atkinson, I. (2017). Internet of Things for Smart Healthcare: Technologies, Challenges, and Opportunities. *IEEE*.

Balitbangkes. (2018). *Hasil Utana Riskesdas 2018.* Jakarta: Kementrian Kesehatan Republik Indonesia.

Brownlee, J. (2019, December 23). *A Gentle Introduction to Imbalanced Classification*. Retrieved September 13, 2020, from Machine Learning Mastery: https://machinelearningmastery.com/what-is-imbalanced-classification/

da Costa, C. A., Paslousta, C. F., Eskofier, B., da Silva, D. B., & da Righi, R. (2018). Internet of Health Things: Toward intelligent vital signs monitoring in hospital wards. *Elsevier B.V.*

DIRJENP2P. (2017, October 26). *Direktorat P2PTM Kementerian Kesehatan RI.* Retrieved from Direktorat P2PTM Kementerian Kesehatan RI: http://p2ptm.kemkes.go.id/dokumen-ptm/kebijakan-dan-strategi-pencegahan-dan-pengendalian-stroke-di-indonesia-dr-lily-sriwahyuni-sulistyowati-mm

Elprocus. (2019, Juni 15). *Elprocus*. Retrieved from Elprocus: https://www.elprocus.com/sound-sensor-working-and-its-applications/

Fernández , A., Garcia, S., & Galar, M. (2018). *Learning from Imbalanced Data Sets 1st ed. 2018 Edition.* Springer.

Goyal, A. (2018, September 10). *Endive Software*. Retrieved July 22, 2020, from https://www.endivesoftware.com/blog/how-is-iot-useful-in-the-healthcare-industry/

Hakim, A. E. (2018, February). Internet of Things (IoT) System Architecture and Technologies.

Huth, A., & Cebula, J. (2012). The Basics of Cloud Computing. *US-CERT*.

Indrawati, d., Sari, d., & Dewi, C. S. (2016). *Care Yourself Stroke.* Jakarta: Penebar Plus.

ITU. (2020, Juli 20). *ITU*. Retrieved July 20, 2020, from ITU: https://www.itu.int/en/ITU-T/gsi/iot/Pages/default.aspx

Julpan, Nababan, E. B., & Zarlis, M. (2015). ANALISIS FUNGSI AKTIVASI SIGMOID BINER DAN SIGMOID. *Jurnal Teknovasi* .

Kamouchi, M., Kumagai, N., Okada, Y., Origasa, H., Yamaguchi, T., & Kitazono, T. (2012). Risk Score for Predicting Recurrence in Patients with Ischemic troke: The Fukuoka Stroke Risk Score for Japanese. *Karger AG, Basel*.

Sharma, N., Singh, A., & Gill, B. (2018). CI-DPF: A Cloud IoT based ramework for Diabetes Prediction. *IEEE Journal*.

Kiyenda, & Argarachmah, B. (2019). Hubungan antara Kadar Kolesterol HDL dan Hipertensi terhadap Kejadian Stroke Iskemik Berulang di TSUD DR Moewardi Surakarta.

Liu, A. Y.-c. (2004). The effect of oversampling and undersampling on classifying imbalanced text datasets. *The University of Texas at Austin*.

Lukman, M. P., & Surasa, H. (2017). Portable Monitoring Penderita Penyakit Jantung Terhadap Serangan Berulang Berbasis GPS dan Android. *Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) KHARISMA Makassar*.

Madakam, S., Ramaswamy, R., & Tripathi, S. (2015). Internet of Things (IoT): A Literature Review. *Journal of Computer and Communications*.

Majumder, A. J., ElSaadany, Y., ElSaadany, M., Ucci, D. R., & Rahman, F. (2017). A Wireless IoT System Towards Gait Detection in Stroke Patients. *IEEE*.

N, N. P., Pebralia, J., Dewi, Y. C., & Hendro. (2015). Studi Penerapan Sensor MLX90614 Sebagai Pengukur Suhu Tinggi secara Non-kontak Berbasis Arduino dan Labview. *Prosiding Simposium Nasional Inovasi dan Pembelajaran Sains 2015 (SNIPS 2015)*.

Nair, V., & Hinton, G. E. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines.

NHLBI. (2019, November 15). *National Heart, Lung, and Blood Institute*. Retrieved from National Heart, Lung, and Blood Institute: https://www.nhlbi.nih.gov/health-topics/stroke

Park, S. J., Subramaniyam, M., Kim, S. E., Hong, S., Lee, J. H., Jo, C. M., & Seo, Y. (2017). Development of the Elderly Healthcare Monitoring System with IoT. *Center for Medical Metrology, Korea Research Institute of Standards and Science (KRISS)*.

Prakash, R., Girish, S. V., & Ganesh, A. B. (2016). Real-Time Remote Monitoring of Human Vital Signs Using Internet of Things (IoT) and GSM Connectivity. *Springer India*.

Putri, D. M. (2017). Mengenal Wemos D1 Mini dalam dunia IoT. *ilmuti.org*.

Riadi, M. (2019, Mei 26). *kajianpustaka*. Retrieved from kajianpustaka: https://www.kajianpustaka.com/2019/05/pengertian-layanan-dan-parameter-quality-of-service-qos.html

Sandeep K.Sood, I. M. (2018). IoT-Fog based Healthcare Framework to Identify and Control Hypertension Attack . *IEEE Internet of Things Journal*.

Santosa, B., & Umam, A. (2018). *Data Mining dan Big Data Analytics Edisi 2.* Yogyakarta: Penebar Media Pustaka.

Sari, I. P. (2015). Faktor-faktor yang berhubungan dengan Terjadinya Stroke berulang pada Penderita Pasca Stroke. *Skripsi*.

Shaji, J. E., Varghese, B., & Varghese, R. (2017). A Health Care Monitoring System with Wireless Body Area Network using IoT . *International Journal of Recent Trends in Engineering & Research (IJRTER)* .

Sugiyono. (2017). *Metode Penelitian Kuantitatif, Kualitatif, dan Kombinasi (Mixed Methods).* Bandung: Alfabeta.

. R. (2015). Hubungan Upaya Penceagahan Terhadap Kejadian Stroke erulang Pada Penderita Stroke. *Digital Repository Universitas Jember*.

P., & Sood, S. K. (2017). Cloud-centric IoT based disease diagnosis ealthcare framework. *Elsevier*.

w3schools. (2019). *w3schools.in*. Retrieved from w3schools.in: https://www.w3schools.in/cloud-computing/how-cloud-computing-works/

WHO. (2020, Juli 1). *World Health Organization*. Retrieved from World Health Organization: https://www.who.int/topics/cerebrovascular_accident/en/

World Health Organization Regional Office for the Western Pacific. (1998). *Nursing Care of The Sick: A guide for Nurses Working in Small Rural Hospital*. Manila: WHO Regional Office for the Western Pacific.

Yahyaie, M., & Tarokh, M. J. (2018). Use IoT to Provide a New Model for Remote Heart Attack Prediction. *Strategic Intelligence Research Lab, K. N. Toosi University of Technology*.

LAMPIRAN

1. *Source code* Wemos D1 Mini dan *pulse sensor*

```
#include <Wire.h>
#include "FirebaseESP8266.h"
#include <ESP8266WiFi.h>

#define FIREBASE_HOST "stroke-25f59.firebaseio.com"
#define FIREBASE_AUTH "py0Ce80Hd7ermav2utY0qw57utiCfoAHrJ******"
#define WIFI_SSID     "thya"
#define WIFI_PASSWORD "********"

FirebaseData firebaseData;
FirebaseData ledData;
FirebaseJson json;

#define pulsePin A0

int rate[10];
unsigned long sampleCounter = 0;
unsigned long lastBeatTime = 0;
unsigned long lastTime = 0, N;
int BPM = 0;
int IBI = 0;
int P = 512;
int T = 512;
int thresh = 512;
int amp = 100;
int Signal;
boolean Pulse = false;
boolean firstBeat = true;
boolean secondBeat = true;
boolean QS = false;

void setup() {
  Serial.begin(9600);

  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);
```

```
void loop() {
  if (QS == true) {
   Serial.println("BPM: "+ String(BPM));
   QS = false;
   } else if (millis() >= (lastTime + 2)) {
     readPulse();
     lastTime = millis();
   }
}
void readPulse() {
  Signal = analogRead(pulsePin);
  sampleCounter += 2;
  int N = sampleCounter - lastBeatTime;

  detectSetHighLow();

  if (N > 250) {
    if ((Signal > thresh) && (Pulse == false) && (N > (IBI/5) * 3 ))
      pulseDetected();
  }

  if (Signal < thresh && Pulse == true) {
    Pulse = false;
    amp = P - T;
    thresh = amp / 2 + T;
    P = thresh;
    T = thresh;
  }

if (N > 2500) {
    thresh = 512;
    P = 512;
    T = 512;
    lastBeatTime = sampleCounter;
    firstBeat = true;
    secondBeat = true;
  }
}

void detectSetHighLow() {
  if (Signal < thresh && N > (IBI / 5) * 3) {
    if (Signal < T) {
      T = Signal;
    }
  }
  if (Signal > thresh && Signal > P) {
    P = Signal;
  }

void pulseDetected() {
  Pulse = true;
    sampleCounter - lastBeatTime;
    eatTime = sampleCounter;
```

```
if (firstBeat) {
    firstBeat = false;
    return;
  }
  if (secondBeat) {
    secondBeat = false;
    for (int i = 0; i <= 9; i++) {
      rate[i] = IBI;
    }   word runningTotal = 0;

  for (int i = 0; i <= 8; i++) {
    rate[i] = rate[i + 1];
    runningTotal += rate[i];
  }

  rate[9] = IBI;
  runningTotal += rate[9];
  runningTotal /= 10;
  BPM = 60000 / runningTotal;
  QS = true;

  if (Firebase.setFloat(firebaseData, "/vitalsigns/heartRate", BPM))
  {
    Serial.println("PASSED");
    Serial.println("PATH: " + firebaseData.dataPath());
    Serial.println("TYPE: " + firebaseData.dataType());
    Serial.println("ETag: " + firebaseData.ETag());
    Serial.println("------------------------------------");
    Serial.println();
  }
  else
  {
    Serial.println("FAILED");
    Serial.println("REASON: " + firebaseData.errorReason());
    Serial.println("------------------------------------");
    Serial.println();
  }
}
```

2. *Source code* Wemos D1 Mini, sensor MLX9014, dan *sound sensor*

```
#include "FirebaseESP8266.h"  // Install Firebase ESP8266 library
#include <ESP8266WiFi.h>
#include <Wire.h>
#include <Adafruit_MLX90614.h>

#define FIREBASE_HOST "stroke-25f59.firebaseio.com"
#define FIREBASE_AUTH "py0Ce80Hd7ermav2utY0qw57utiCfoAHr*******"
#define WIFI_SSID      "thya"
#define WIFI_PASSWORD "********"

Adafruit_MLX90614 mlx = Adafruit_MLX90614();
FirebaseData firebaseData;
FirebaseData ledData;
FirebaseJson json;

int nafas = D5;
int nafasState = 1;
int nafasRate;

void setup() {
  Serial.begin(9600);
  Serial.println("Adafruit MLX90614 test");
  mlx.begin();

  pinMode(nafas, INPUT);
  Serial.println("Nafas :");//lcd.print("Nafas :");

  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);

void loop() {
  float t = mlx.readObjectTempC();
  Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempC());
  Serial.print("*C\tObject = "); Serial.print(t);
  Serial.println("*C");
  Serial.println();
  delay(500);
  nafas=1;
  int time=0;

    (digitalRead(nafas)==LOW){
    fas++;
```

```
  delay(500);
  time+=100;
  } while(time<=1000);
  nafasRate=6*nafas;
  Serial.print("rr = ");
  Serial.print(nafasRate);
  Serial.println(" /mnt");

  if (Firebase.setFloat(firebaseData, "/vitalsigns/temperature",t)) {
    Serial.println("PASSED");
    Serial.println("PATH: " + firebaseData.dataPath());
    Serial.println("TYPE: " + firebaseData.dataType());
    Serial.println("ETag: " + firebaseData.ETag());
    Serial.println("------------------------------------");
    Serial.println();
  }
    else {
    Serial.println("FAILED");
    Serial.println("REASON: " + firebaseData.errorReason());
    Serial.println("------------------------------------");
    Serial.println();
  }

if
(Firebase.setFloat(firebaseData,"/vitalsigns/respiratoryRate",nafasRa
te))
  {
    Serial.println("PASSED");
    Serial.println("PATH: " + firebaseData.dataPath());
    Serial.println("TYPE: " + firebaseData.dataType());
    Serial.println("ETag: " + firebaseData.ETag());
    Serial.println("------------------------------------");
    Serial.println();
  }
  else
  {
    Serial.println("FAILED");
    Serial.println("REASON: " + firebaseData.errorReason());
    Serial.println("------------------------------------");
    Serial.println();
  }
}
```

3. *Source code* aplikasi *mobile* berbasis Android

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    RiskFactors riskFactors;
    String stringGender;
    String stringHypertension;
    String stringHeartDisease;
    String stringDm;
    String stringSmoking;
    Button btnSave;
```

```
    private DatabaseReference databaseReference;
    private EditText editTextName;
    private EditText editTextAge;
    private RadioGroup rgGender;
    private RadioGroup rgHypertension;
    private RadioGroup rgHeartDisease;
    private RadioGroup rgDm;
    private RadioGroup rgSmoking;
    private RadioButton genderOption;
    private RadioButton hypertensionOption;
    private RadioButton hdOption;
    private RadioButton dmOption;
    private RadioButton smokingOption;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    riskFactors = new RiskFactors();
    editTextName = findViewById(R.id.edt_name);
    editTextAge = findViewById(R.id.edt_age);
    rgGender = findViewById(R.id.radioGroup_gender);
    rgHypertension = findViewById(R.id.radioGroup_hyper);
    rgHeartDisease = findViewById(R.id.radioGroup_hd);
    rgDm = findViewById(R.id.radioGroup_dm);
    rgSmoking = findViewById(R.id.radioGroup_smoking);
    btnSave = findViewById(R.id.button_save);
    ImageView imageViewVitalSign =
findViewById(R.id.imageView_vitalSign);
databaseReference = FirebaseDatabase.getInstance().getReference();

btnSave.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (checkAndGetData()) {
            saveData();
        } else {
            Toast.makeText(MainActivity.this, "invalid data",
Toast.LENGTH_SHORT).show();
        }
    }
});
imageViewVitalSign.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (view.getId() == R.id.imageView_vitalSign) {
                Intent moveIntent = new Intent(MainActivity.this,
VitalSignsActivity.class);
                startActivity(moveIntent);
            } else {
                Toast.makeText(MainActivity.this, "error",
ENGTH_SHORT).show();
            }
        }
```

```
private void saveData() {
        databaseReference.child("riskFactors").setValue(riskFactors)
                .addOnSuccessListener(new OnSuccessListener<Void>()
{
                    @Override
                    public void onSuccess(Void aVoid) {
                        Toast.makeText(MainActivity.this, "Data
saved successfully ", Toast.LENGTH_LONG).show();
                    }
                })
                .addOnFailureListener(new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception e) {
                        Toast.makeText(MainActivity.this, "Data
failed to save", Toast.LENGTH_LONG).show();
                    }
                });
    }

    private boolean checkAndGetData() {
        boolean valid = true;
        riskFactors.name = editTextName.getText().toString().trim();
        riskFactors.age = editTextAge.getText().toString().trim();


        rgGender.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(RadioGroup radioGroup, int
checkedId) {
                genderOption = rgGender.findViewById(checkedId);
                stringGender =
genderOption.getText().toString().trim();
                riskFactors.gender = stringGender.equals("Man");
            }
        });

        rgHypertension.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(RadioGroup radioGroup, int
checkedId) {
                hypertensionOption =
rgHypertension.findViewById(checkedId);
                stringHypertension =
hypertensionOption.getText().toString().trim();
                riskFactors.hypertension =
stringHypertension.equals("Yes");
            }
        });

        rgHeartDisease.setOnCheckedChangeListener(new
        oup.OnCheckedChangeListener() {
            @Override
```

```
public void onCheckedChanged(RadioGroup radioGroup, int checkedId) {
                hdOption = rgHeartDisease.findViewById(checkedId);
                stringHeartDisease =
hdOption.getText().toString().trim();
                riskFactors.heartDisease =
stringHeartDisease.equals("Yes");
            }
        });

rgDm.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(RadioGroup radioGroup, int
checkedId) {
                dmOption = rgDm.findViewById(checkedId);
                stringDm = dmOption.getText().toString().trim();
                riskFactors.dm = stringDm.equals("Yes");

            }
        });

        rgSmoking.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(RadioGroup radioGroup, int
checkedId) {
                smokingOption = rgSmoking.findViewById(checkedId);
                stringSmoking =
smokingOption.getText().toString().trim();
                riskFactors.smoking = stringSmoking.equals("Yes");

            }
        });

if (isEmpty(riskFactors.name)) {
            valid = false;
            editTextName.setError("Enter Patient's Name");
        } else {
            editTextName.setError(null);
        }
        if (isEmpty(riskFactors.age)) {
            valid = false;
            editTextAge.setError("Enter Patient's Age");
        } else {
            editTextAge.setError(null);
        }

        return valid;
    }

}
```

VitalSignsActivity.java

```
public class VitalSignsActivity extends AppCompatActivity {

    VitalSigns vitalSigns;
    TextView textHeartRate;
    TextView textRespiratoryRate;
    TextView textTemperature;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_vital_signs);

        vitalSigns = new VitalSigns();

        textHeartRate = findViewById(R.id.textView_heartRate);
        textRespiratoryRate =
findViewById(R.id.textView_respiratoryRate);
        textTemperature = findViewById(R.id.textView_temperature);

        FirebaseDatabase database = FirebaseDatabase.getInstance();
        DatabaseReference reference =
database.getReference("vitalsigns/");

       reference.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot)
{
                String heartRate =
String.valueOf(snapshot.child("heartRate").getValue());
                textHeartRate.setText(heartRate);
                String respiratoryRate =
String.valueOf(snapshot.child("respiratoryRate").getValue());
                textRespiratoryRate.setText(respiratoryRate);
                Double temperature = (Double)
snapshot.child("temperature").getValue();
                textTemperature.setText(new
DecimalFormat("##.##").format(temperature));
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {

            }
        });

    }
}
```

VitalSigns.java

```
class VitalSigns implements Serializable {
ing heartRate;
ing respiratoryRate;
at temperature;
```

```
public VitalSigns() {

    }

    public VitalSigns(String HeartRate, String RespiratoryRate,
Float Temperature) {
        this.heartRate = HeartRate;
        this.respiratoryRate = RespiratoryRate;
        this.temperature = Temperature;
    }

    public String getHeartRate() {
        return heartRate;
    }

    public void setHeartRate(String heartRate) {
        this.heartRate = heartRate;
    }

    public String getRespiratoryRate() {
        return respiratoryRate;
    }

    public void setRespiratoryRate(String respiratoryRate) {
        this.respiratoryRate = respiratoryRate;
    }

    public Float getTemperature() {
        return temperature;
    }

    public void setTemperature(Float temperature) {
        this.temperature = temperature;
    }

    @NonNull
    @Override
    public String toString() {
        return " " + heartRate + "\n" +
                " " + respiratoryRate + "\n" +
                " " + temperature;
    }

}
```

RiskFactors.java

```
public class RiskFactors implements Serializable {
    String name;
    String age;
    lean gender;
    lean hypertension;
    lean heartDisease;
    lean dm;
    lean smoking;
```

```
public RiskFactors() {

    }

    public RiskFactors(String Name, String Age, boolean Gender,
boolean Hypertension, boolean HeartDisease, boolean Dm, boolean
Smoking) {
        name = Name;
        age = Age;
        gender = Gender;
        hypertension = Hypertension;
        heartDisease = HeartDisease;
        dm = Dm;
        smoking = Smoking;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAge() {
        return age;
    }

    public void setAge(String age) {
        this.age = age;
    }

    public boolean isGender() {
        return gender;
    }

    public void setGender(boolean gender) {
        this.gender = gender;
    }

    public boolean isHypertension() {
        return hypertension;
    }

    public void setHypertension(boolean hypertension) {
        this.hypertension = hypertension;
    }

    public boolean isHeartDisease() {
        return heartDisease;
    }


    lic void setHeartDisease(boolean heartDisease) {
        this.heartDisease = heartDisease;
```

```
public boolean isDm() {
        return dm;
    }

    public void setDm(boolean dm) {
        this.dm = dm;
    }

    public boolean isSmoking() {
        return smoking;
    }

    public void setSmoking(boolean smoking) {
        this.smoking = smoking;
    }

    @NonNull
    @Override
    public String toString() {
        return " " + name + "\n" +
                " " + age + "\n" +
                " " + gender + "\n" +
                " " + hypertension + "\n" +
                " " + heartDisease + "\n" +
                " " + dm + "\n" +
                " " + smoking;
    }
}
```

4. *Source code* Pengolahan Data

```
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import random
import seaborn as sns
import warnings
import time

import tensorflow as tf
from tensorflow.keras.layers import *
# Oversampling
from imblearn.over_sampling import RandomOverSampler

warnings.filterwarnings("ignore")

seed = 101
tf.random.set_seed(seed)

.read_csv(r'dataset_fix2.csv', header=0)
.drop(['no', 'bloodPressure'], axis=1)
```

```
## Convert categorical values to numeric
gender = {'L': 0, 'l' : 0, 'P': 1, 'p' : 1}
df['gender'] = df['gender'].map(gender)
```

```
X = df
y = X.pop('STROKE')
```

```
def baseline_model():
    model = tf.keras.Sequential()

model.add(tf.keras.layers.Flatten(input_shape=(len(X.columns),)))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(16, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy',
optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
metrics=['accuracy'])
    return model
```

```
model = baseline_model()
model.fit(X,y)
```

```
EPOCHS = 30
BATCH_SIZE = 64

from sklearn.model_selection import StratifiedKFold
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=seed)

train_acc = []
test_acc = []
history = []
for train_index, test_index in skf.split(X, y):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y[train_index], y[test_index]
    X_train , y_train =
RandomOverSampler(random_state=seed).fit_resample(X_train, y_train)
    model = baseline_model()
    history += [model.fit(X_train, y_train,
validation_data=(X_test,y_test), epochs=EPOCHS,
batch_size=BATCH_SIZE)]

    m = tf.keras.metrics.Accuracy()
    m.update_state(model.predict_classes(X_train), y_train)
    trainScore = m.result().numpy()

    m = tf.keras.metrics.Accuracy()
    m.update_state(model.predict_classes(X_test), y_test)
    tScore = m.result().numpy()

    in_acc.append(trainScore)
    t_acc.append(testScore)
```