

Estudiantes :Andres Afanador , Sebastian Villegas Rojas.

Idea del codigo: Detectar los bajones de corriente en una señal eléctrica (sag), estos se ven de la siguiente manera:

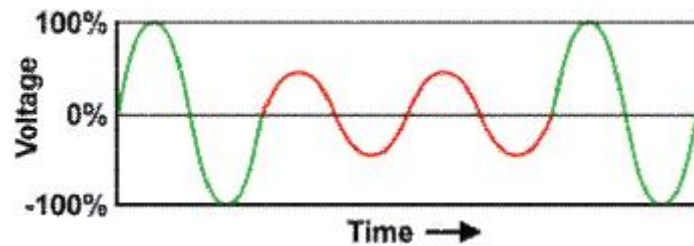


Figura 1. Onda con sag.

Base de datos:

La base de datos fue generada de manera artificial mediante un código que se obtuvo de un paper de investigación , esta base de datos me dejó generar 20000 vectores de ondas con sag, y otros 20000 vectores de ondas sin este problema de calidad de la energía . Para el periodo de prueba se usarán 10000 con sag y 10000 normales.

Estos vectores son todos de una onda de 60 hz y de 10 ciclos , con valores normalizados para facilitar el programa.

Extraccion de características:

En el proceso de la IA por ahora llevamos hasta lo que seria la extraccion de características de la base de datos, estos los definimos con las siguientes funciones:

```
def std(vecstd):
    #Funcion para la desviacion estandar
    return np.std(vecstd)
def mean(meanvec):
    #Funcion para el promedio
    return np.mean(meanvec)
def counth(vect):
    #Funcion para hacer la cuenta de los valores encima de 1
    vect1=abs(vect)
    count1=0
    for i in vect1:
        if abs(i)>1:
            count1=count1+1
    return count1
def countlow(vect):
    #Funcion para contar las veces que el valor es menor a 0.1
    vectlow=abs(vect)
    clow=0
    for i in vectlow:
        if abs(i)<=0.1:
            clow=clow+1
    return clow
def countinter(vint):
    #Funcion para contar las veces que se encuentra en el intervalo 0.1 y 1
    vinter=abs(vint)
    cinter=0
    for i in vinter:
        if abs(i)>0.1 and abs(i)<=1:
            cinter=cinter+1
    return cinter
```

Figura 2. Primeras funciones para extracción de características.

Las funciones std y mean ,son la desviación estándar y el promedio respectivamente de una onda completa, ya que cuando la función es sana estos darán unos valores cercanos pero cuando haya un bajón de tensión, estas tendrán un promedio lejano a cero y una desviación estándar por ende diferente a la de una función sana.

Las otras funciones son counth que cuenta las veces que el valor absoluto de la onda es mayor a 1 , countlow que cuenta las veces que el valor absoluto de la onda está entre 0 y 0.1 , y finalmente countinter que cuenta las veces que la onda está entre 0.1 y 1.

THD:El THD se define como la distorsión armónica de una onda , esta indica que tan distorsionada está la señal con respecto a su frecuencia fundamental que en nuestro caso siempre son 60 hz, definiéndose matemáticamente así:

$$THD_v(\%) = \sqrt{\left(\frac{V_{rms}}{V_1}\right)^2 - 1} \times 100$$

Figura 3.THd de una función.

Donde V1 es la magnitud original con la transformada de fourier y Vrms es :

$$V_{rms} = \sqrt{\sum_{h=0}^{\infty} I_h^2}$$

Ecuación 1.Definición de Vrms.

```
[ ] def RMS(Vrms1):
    Vrms= np.abs(Vrms1)
    return sqrt(sum(np.array(Vrms)**2))
def THD(Vthd):
    Vthd= np.abs(Vthd)
    return sqrt((RMS(Vthd)/list(Vthd)[1])**2-1)*100
print(THD(list(X2)[0:15]))
print(list(np.abs(X2))[0:15])
```

Figura 4.Función RMS y THD.

Esta será importante puesto que nos ayudará a diferenciar una onda con problemas de una dañada , y en nuestro caso se hará para cada ciclo de los 10 ciclos de cada onda , pero tomaremos en cuenta sólo el valor mayor ,el menor, y el promedio del THD para los 10 ciclos devueltos en una lista con el siguiente código:

```

def THDIA(vTHDIA1):
    "Esta funcion me saca el THD promedio de todos los ciclos, el menor y el mayor"
    #Dividimos el vector en su numero de ciclos en una lista
    vTHDIA=list(vTHDIA1)
    interval=[]
    ffi=[]
    THDr=[]
    for i in range(1,m.floor(len(vTHDIA)/53.3)):
        num1=m.floor(53.3*(i-1))
        num2=m.floor(53.3*(i))
        interval=vTHDIA[num1:num2]
        #Hacemos la transformada de fourier y lo añadimos a una lista
        ffi=fftpack.fft(interval)

        #Le aplicamos el THD
        THDr.append(THD(list(ffi)[0:25]))
        #Le aplicamos el LHD que es THD pero considerando solo los primeros 7 frecuencias

    #Vamos a retornar una lista de 6 numeros
    mp=[np.mean(THDr),min(THDr),max(THDr)]
    return mp

```

Figura 4.Función de THD max,mínimo y promedio.

Siguiente paso para el próximo informe:

Para el siguiente paso estaremos entrenando la IA , la cual decidimos que será un SVM y también testeando con la base de datos que tenemos.