Supplement

Supplement A. SIGTIERRAS approach

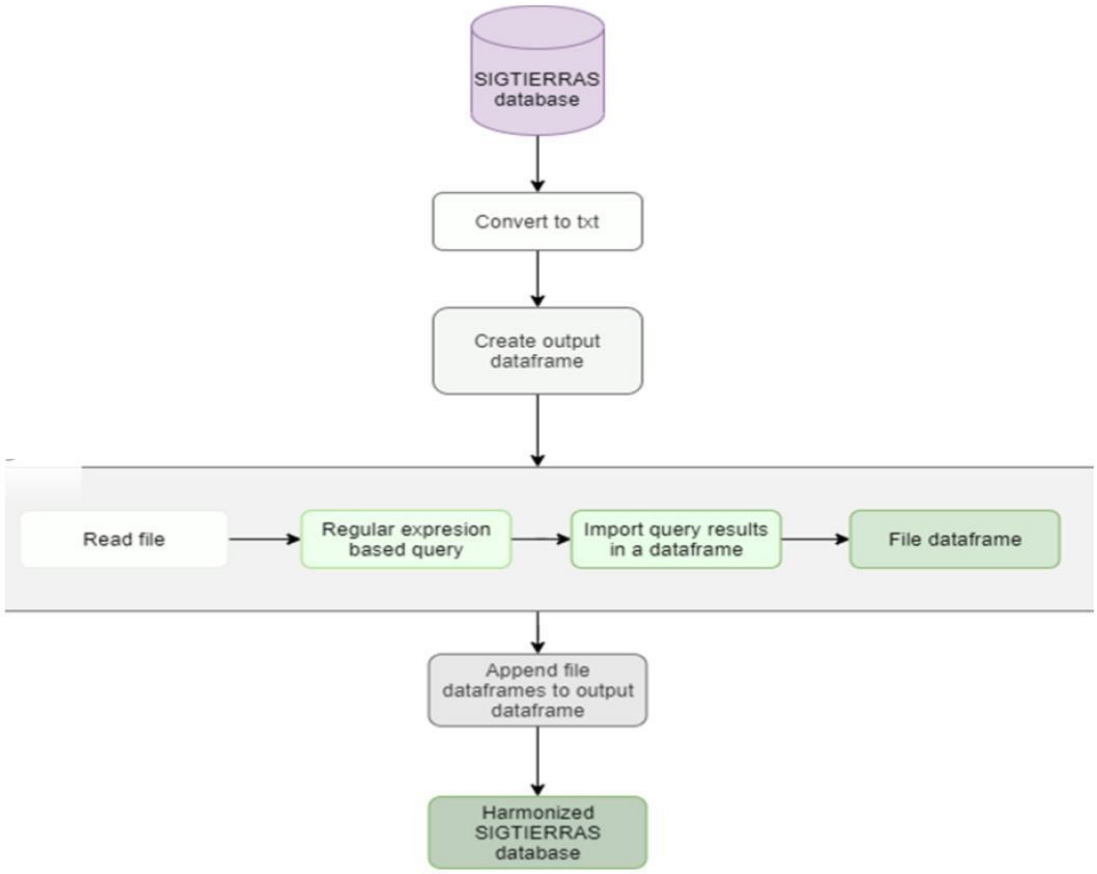

Figure S1. Workflow steps to extract information from SIGTIERRAS´s files.

Supplement B. Script files of SIGTIERRAS

```
# Author: Fernando Bezares Sanfelip
# Email: fernando.bezares@cesefor.com | fernandobezares95@hotmail.es
import re
import glob, os
import subprocess

# Ecuador soil database generation project
# Pass list of PDFs to R

##############################################

# FILES OF SIGTIERRAS

##############################################

# Call the R script that will convert to txt
# indicate the .pdf
pdf_files=
glob.glob(os.path.join('C:\\Users\\fbezares\\Documents\\01_Proyectos\\06_Almeria\\SIGTIERRAS\\samp
le','*.pdf'))
print(pdf_files)

# first position indicate where the Rscript.exe is
# second position indicate Converter_pdf_txt_python.R
commands = ['C:\\Program Files\\R\\R-
4.0.2\\bin\\x64\\Rscript.exe','C:\\Users\\fbezares\\Documents\\01_Proyectos\\06_Almeria\\SIGTIERRAS\
\Conversor_pdf_txt_python.R']

subprocess.call(commands + pdf_files, shell=True)
print(commands+pdf_files)

# we indicate where the txt files are (same folder as pdf)
txt_files =
glob.glob(os.path.join('C:\\Users\\fbezares\\Documents\\01_Proyectos\\06_Almeria\\SIGTIERRAS\\samp
le','*.txt'))
print(txt_files)
# open the soil csv file
```

```
encabezado_perfiles_Sig = 'PerID; ID_nac; COORD_X; COORDY; altitud_m ;Clas_SGST;
Clas_GGST; Clas_OST; RTS; RHS; Prof_efec; Geología;Geoforma; pend_local; Uso Tierra; Vegetación;
Rocosidad; Pedre; Tipo_erosion; Grado_erosion; Drena; fecha_inte'
# create file to write files to
arch = open('BD_SUELOS_PERFILES_SIGTIERRAS.csv','a')
arch.write('{}\n'.format(encabezado_perfiles_Sig))
arch.close()
# Go through the txt to apply the regular expressions
for i,txt_file in enumerate(txt_files):
        texto = open(txt_file,'r')
        string ="
        for i in texto:
                string=string+i
        arch=open('BD_SUELOS_PERFILES_SIGTIERRAS.csv','a')

        # regular expressions
        ID_nac=re.findall(r"(?<=[C,c][ó,o]digo  [F,f]icha\" \")[^\"]+",string)[0].strip()
        coord_x=re.findall(r"(?<=[X,x]:\"  \")[^\"]+",string)[0]
        coord_y=re.findall(r"(?<=[Y,y]:\"  \")[^\"]+",string)[0]
        altitud_m=re.findall(r"(?<=[A,a]ltitud:\"  \")[^\"]+",string)[0]
        fecha_inte=re.findall(r"(?<=[F,f]echa descripci[ó,o]n)[^\n]+",string)[0]
        Geología=re.findall(r"(?<=[F,f]ormaci[ó,o]n  [G,g]eol[ó,o]gica:\"  \")[^\"]+",string)[0]
        pend_local=re.findall(r"(?<=[P,p]endiente  [L,l]ocal:\" \")[^\%]+",string)[0]
        Geoforma=re.findall(r"(?<=[G,g]eoforma:\" \")[^\"]+",string)[0]
        Uso_Tierra=re.findall(r"(?<=[u,U]so  de la [T,t]ierra:\" \")[^\"]+",string)[0]
        Tipo_erosion=re.findall(r"(?<=[C,c]ategor[í,i]a\"  \")[^\"]+",string)[0]
        Grado_erosion=re.findall(r"(?<=[G,g]rado\"  \")[^\"]+",string)[0]
        rts=re.findall(r"(?<=[R,r][é,e]gimen  de temperatura del\"\s\")[^\"]+",string)[0]
        rhs=re.findall(r"(?<=[R,r][é,e]gimen  de humedad del suelo\:\"\s\")[^\"]+",string)[0]
        Prof_efec=re.findall(r"(?<=[v,V]alor:\"  \")[^\"]+",string)
        if len(Prof_efec)<2:
                Prof_efec=re.findall(r"(?<=[v,V]alor:\"  \")[^\"]+",string)[0]
        else:
                Prof_efec=re.findall(r"(?<=[v,V]alor:\"  \")[^\"]+",string)[1]

        Vegetacion=re.findall(r"(?<=[v,V]egetaci[ó,o]n:\"  \")[^\"]+",string)[0]
        Rocosidad=re.findall(r"(?<=Abundancia \(\%\)\" \")[^\"]+",string)[0]
        Pedre=re.findall(r"(?<=Abundancia  \(\%\)\" \")[^\"]+",string)[1]
        Drena=re.findall(r"(?<=[P,p]ermeabilidad  [D,d]renaje [N,n]atural:\" \")[^\"]+",string)[0]

```

```
# extraction of local slope values

if '>' in pend_local:
        p=pend_local.split('>')[1].split('-')
        pend_local=(float(p[1])+float(p[0]))/2
elif '>' not in pend_local:
        p=pend_local.split('(')[1].split('-')
        pend_local=(float(p[1])+float(p[0]))/2

# extraction of stoniness values
if '<' in Pedre:
        Pedre=Pedre.split('<')[1]
elif Pedre=='-':
        Pedre='0'
elif '-'in Pedre:
        pe=Pedre.split('-')

        if pe ==["","",""]:
                Pedre='NAN'
        else:
                int_sup=float(pe[1])
                int_inf=pe[0].split(' ')[1]

                Pedre=(int_sup+float(int_inf))/2
else:
        Pedre=Pedre.split(' ')[1]

# extraction of rockiness values

if '<' in Rocosidad:
        Rocosidad = Rocosidad.split('<')[1]
elif Rocosidad == '-':
        Rocosidad = '0'
elif '-'in Rocosidad:
        pe=Rocosidad.split('-')
        if pe ==["","",""]:
                Pedre='NAN'
        else:
                int_sup=float(pe[1])
                int_inf=pe[0].split(' ')[1]
```

```
                    Rocosidad=(int_sup+float(int_inf))/2
        else:
            Rocosidad=Rocosidad.split(' ')[1]

        # exception handling for vegetation and land use
        if Vegetacion == '-' or Vegetacion == ' ':
            Vegetacion = 'NAN'
        elif Uso_Tierra == '-' or Uso_Tierra == ' ':
            Uso_Tierra = 'NAN'
        else:
            pass









        # classification for horizons
        Orden=re.findall(r"(?<=Orden final:\" \")[^\"]+",string)[0]# orden
        Sistema_clas=re.findall(r"(?<=Clasificaci[ó,o]n Final\s\()[^\)]+",string)[0]
        gran_grupo=re.findall(r"(?<=Gran grupo final:\" \")[^\"]+",string)[0]# gran grupo
        Subgrupo=re.findall(r"(?<=Subgrupo final:\" \")[^\"]+",string)[0]# subgrupo

        # Horizons code
        codigo_horizonte=re.findall(r"(?<=[S,s][í,i]mbolo: )[^\"]+",string)

        # roots
        raices_vf=re.findall(r"(?<=VF \([M,m]uy [F,f]ina\)[\",\s])[^\"]+",string)
        raices_f=re.findall(r"(?<=F \([F,f]ina\)[\",\s]{3})[^\"]+",string)
        raices_mf=re.findall(r"(?<=M \([M,m]edia\)[\",\s]{3})[^\"]+",string)
        raices_g=re.findall(r"(?<=C \([G,g]ruesa\)[\",\s]{3})[^\"]+",string)

        # Depending on the code, they will have different root values

        for i in range(len(codigo_horizonte)):

            ch=codigo_horizonte[i]
            r_vf=raices_vf[i]
            r_f=raices_f[i]
```

```
                    r_m=raices_mf[i]
                    r_g=raices_g[i]

            # indicate if there are roots in the horizon
                    if r_vf!='Ninguna' or r_f!='Ninguna'or r_m!='Ninguna'or r_g!='Ninguna':
                            raices='1'
                    else:
                            raices='0'
                    # 'CODIGO; HORIZONTE; X; Y; SISTEMA CLASIFICACION; ORDEN; GRAN
GRUPO; SUBGRUPO; FECHA; RAICES'
                    #contenido_horizontes= ID_nac+';'+ch+';'+coord_x+';'+ coord_y +';'+ Sistema_clas
+';'+Orden+';'+gran_grupo+';'+Subgrupo+';'+fecha_inte+';'+raices
                    #arch_hori.write('{}\n'.format(contenido_horizontes))



            contenido=' '+';'+ID_nac+';'+coord_x+';'+ coord_y
+';'+altitud_m+';'+Subgrupo+';'+gran_grupo+';'+Orden+';'+  rts +';'+rhs+';'+ Prof_efec+';'+ Geología +';'+
Geoforma +';'+str(pend_local)+';'+ Uso_Tierra +';'+
Vegetacion+';'+str(Rocosidad)+';'+str(Pedre)+';'+Tipo_erosion+';'+Grado_erosion+';'+
Drena+';'+fecha_inte

            arch.write('{}\n'.format(contenido))
            arch.close()
            texto.close()
            os.remove(txt_file)

#(?<=VF \([M,m]uy [F,f]ina\)[\",\s])[^\"]+
#(?<=\"1\" )\"[^\"]+[^\d]+[^\"]+[^\d]+[^\"]+[^\d]+[^\"]+
#\"[a-zA-Z]{1,3}[a-z0-9]*\"
```

Supplement C. IEE approach

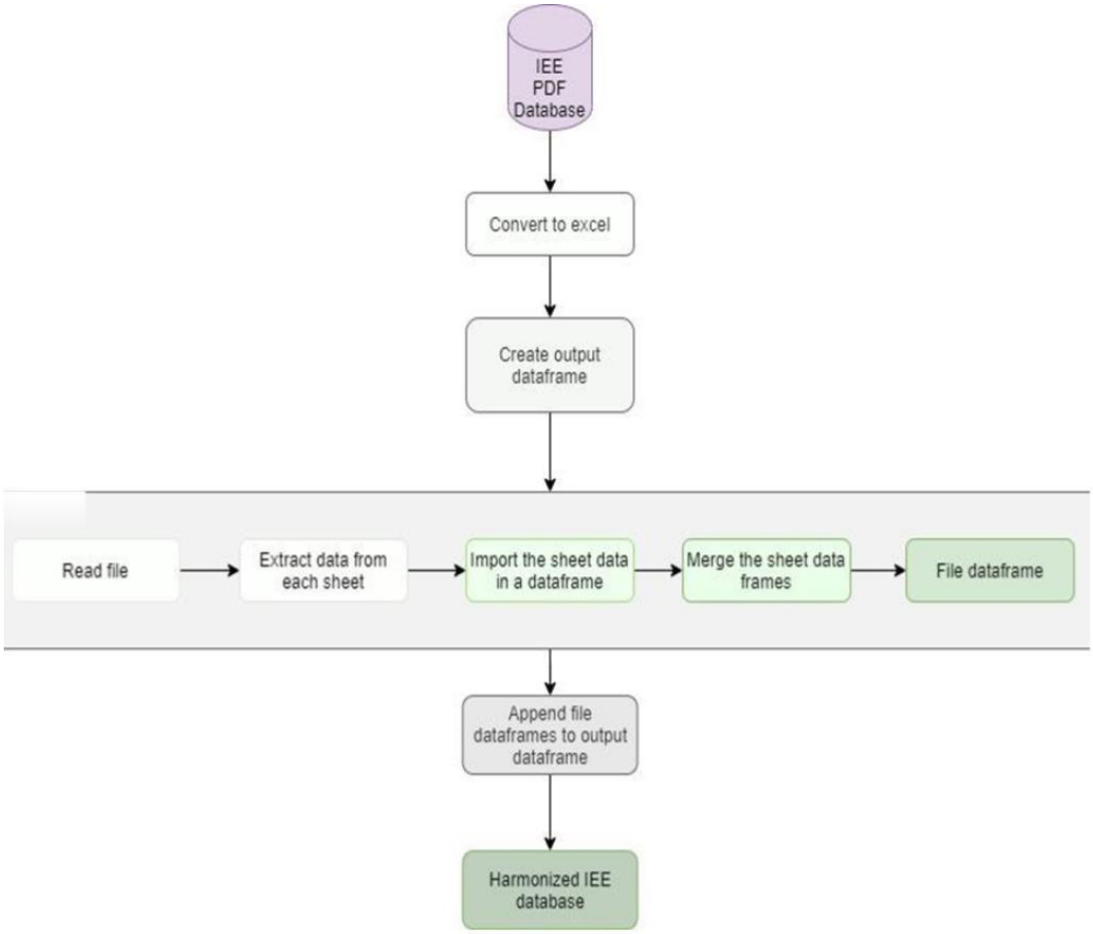

Figure S2. Workflow steps to extract information from IEE's files.

Supplement D. Script files of IEE

```
# Author: Fernando Bezares Sanfelip
# Email: fernando.bezares@cesefor.com | fernandobezares95@hotmail.es

## DESCRIPTION ##

# The script aims to extract in the most automated way the tabular and text information from a soil-backup-pdf-format database files converted to excel.
# Each excel file corresponds to a soil profile where profile descritpion and tabular data are provided.

# The logic of the script is to generate a dataframe for each sheet in the excel file. Each sheet corresponds to a table with chemical or physical description
# for each table the target columns will be identified and pasted in a dictionary, that will form the dataframe. Due to many inconsistences on the origin of the data,
# fields were not always the same, in the same position or with the same name. After converting pdf to excel some names or rows were usually pasted together and therefore
# it was hard to find a systematic rule to split them. Therefore, the functions extract_info and igualar_longitud were desinged to overcome the most frequent cases of
# conversion failures between pdf and excel.

# After creating a dataframe for each table, all the data frames were merged in a common dataframe for the excel file by the dataframe 2, that holds the information
# of the horizons. Finally, the file dataframe is pasted in a general dataframe that holds teh information for all the excel files.

# please bare in mind that the goal of this script is to show an example of how data can be extracted from excel or pdf. Therefore, many rules are taylored just for the
# our data and it is not suitable for other datasets.

# ----------------------------------------------------------------------------------------------------------------------------------
----------------------------------------

# Import libaries
from openpyxl import Workbook
from openpyxl import load_workbook
import pandas as pd
import glob, os, re

# import data
```

```
excel_files=
glob.glob(os.path.join('C:\\Users\\fbezares\\Documents\\01_Proyectos\\06_Almeria\\IEE\\sample','*.xlsx')
)

# DATAFRAME WHERE WE WILL ACCUMULATE ALL THE FINAL INFORMATION AND
WILL BE CONVERTED TO EXCEL.
print(excel_files)
df_final=pd.DataFrame({})

def extract_info(name,colu,referencia,diccionario,d_ref,n_complemento=''):
        ''' looks for information within each column of text. First it checks if the column has a name,
otherwise it passes. Then, it checks if it has line breaks, if it does,
        it generates a list with the information taking into account these line breaks, then it makes sure
that the length of the elements introduced in the dictionary are the same when generating the dataframe.'''
        if colu[0]==None:
                pass
        elif colu[0]==name:
                if len(colu)==1:
                        pass
                # When all the information is in a box, check for line break characters.
                elif '\n' in str(colu[1]):
                        col=colu[1].split('\n')

                        g=[x for x in col]

                        # create a list with the information separated by the character "\n" now check
for blank spaces
                        for k, i in enumerate(g):
                                if r" " in i:
                                        s=i.split(' ')#

                                        s=s[::-1] # reverse the list order
                                        g.pop(k) # pop the element that is joined
                                        for j in s:
                                                g.insert(k,j) # We insert the separated value, in the
copy of the list, where the value with the joined data was before it.
                                else:
                                        pass
                        g=col
                        col.insert(0,name)
```

```
                                print(name, col)

                                if d_ref.get(referencia)==None:
                                        print(d_ref.get(referencia))
                                else:
                                        while len(col)<(len(d_ref.get(referencia))+1):
                                                col.append(None)

                                diccionario.update({name+n_complemento:col[1:]})

                        else:
                                diccionario.update({name:list(colu[1:])})
                pass

def igualar_longitud(diccionario,campo):
        '''Seeks to equalise the length of the dictionary fields once the horizon column is included.'''
        for k,i in  enumerate(diccionario.items()):
                print(i)

                if i==None:
                        pass
                elif len(i[1])==len(diccionario.get(campo)):
                        pass
                else:
                        while len(i[1])<len(diccionario.get(campo))and k!=0:
                                val=i[1].append(None)



for i,file in enumerate(excel_files):

        ## iterate over the sheets
        ## save and create the document with the stored information
        ## access the information in a file
        try:
                wb2 = load_workbook(file)
        except Exception as e:
                pass
```

```
358
359             ws=wb2.active
360
361             d={
362                     'Horizonte/ Capa':[None],
363                     'Profundidad (cm)':[None],
364                     'Arena':[None],
365                     'Limo':[None],
366                     'Arcilla':[None],
367                     'Clase textural':[None],
368                     'Da\n3\n(g/cm )':[None]
369                     }
370
371             # Create the dictionaries that will be used for the data frames
372             dp={}
373             d2={}
374             d3={}
375             d4={}
376             d5={}
377             d6={}
378             d_rp={}
379
380             # dataframe where the data will be pasted in each iteration
381             dff2=pd.DataFrame({'Horizonte/ Capa':[None]})
382
383
384             for sheet in wb2:
385
386                     # check table 1
387                     if sheet.title=='Table 1':
388                             # check postion A3 for profile name
389                             c = sheet['A3'].value
390
391                             print(c)
392                             # alternatively check for A2
393                             if c=='' or c==None:
394                                     c=sheet['A2'].value
395
396                             else:
397                                     pass
```

```
398
399                     #  extract the code by splitting the string by the separators identified
400                     if ':' in c:
401                             perfil=c.split(':')[1]
402                     else:
403                             perfil=c.split('.')[1]
404
405                     # try to include in the dataframe, if not it will  generate an log document
406                     try:
407                             dfp=pd.DataFrame(dp)
408                     except Exception as e:
409                             log=open('descripcion_errores.txt','a')
410                             log.write('error  {} en {}'.format(e,sheet.title))
411                             log.close()
412
413             # Table 4
414             if sheet.title=='Table  4':
415
416                     # check in position A2 if not, then check from the file name
417                     c = sheet['A2'].value
418
419                     if c==''  or c==None:
420                             pass
421                     else:
422                             perfil=file.split('\\')[-1]
423                             perfil=perfil.split('.')[0]
424
425                     # Paste in the dataframe
426                     try:
427                             dfp=pd.DataFrame(dp)
428                     # generate a log file with the name of the file and error
429                     except Exception as e:
430                             log=open('descripcion_errores.txt','a')
431                             log.write('error  {} en {}'.format(e,sheet.title))
432                             log.close()
433
434
435
436
437             # table 5 where the descrption for the field is included
```

```
438                     if sheet.title=='Table 5':
439                             raiz=[]
440                             Pedregosidad=[]
441
442                             for col in sheet.iter_cols(min_row=1,values_only=True):
443
444                                     # if the column is empy then write none in the roots and stoniness
445                                     if col[0]==None:
446                                             d_rp.update({'Raices':[None]})
447                                             d_rp.update({'Pedregosidad':[None]})
448                                             df_rp=pd.DataFrame(d_rp)
449
450                                             pass
451                                     # If the column is not empty then
452                                     elif col[0]=='Horizonte/ Capa' or col[0]=='Horizonte o capa' or
453 col[0]=='Horizonte':
454                                             if col[1]== None:
455                                                     pass
456
457                                             # include the column in the data frame
458                                             elif '\n' in str(col[1]):
459                                                     columna=col[1].split('\n')
460                                                     print('Horizonte/ Capa', columna)
461
462                                                     # if there are white spaces then
463                                                     if ' ' in columna[1]:
464                                                             # select the first value before the split
465 character and extract the information
466                                                             colum=columna[1].split(' ')
467                                                             colum.insert(0,columna[0])
468                                                             colum.insert(0,'Horizonte/ Capa')
469                                                             print('Horizonte', colum)
470                                                             d_rp.update({'Horizonte/ Capa':colum[1:]})
471                                                     else:
472                                                             # extract the information
473                                                             col=[x for x in columna]
474                                                             col.insert(0,'Horizonte/ Capa')
475                                                             print('Horizonte', col)
476                                                             d_rp.update({'Horizonte/ Capa':col[1:]})
477                                             else:
```

```
                                        d_rp.update({'Horizonte/ Capa':list(col[1:])})

                        # if the column is Caracteristicas descritas
                        elif 'Características'in col[0]:

                                if col[1] == None:
                                        pass

                                else:
                                        # makes a list of the column values and for each
position it applies regular expressions
                                        lista=list(col[1:])

                                        for i in lista:
                                                i=str(i)
                                                # to extract the roots
                                                raices=
re.findall(r"(?<=[R,r]a[i,í]ces)[^\;,\,]+",i)

                                                # reclassify in 1 or 0
                                                if len(raices)!=0:
                                                        raiz.append('1')
                                                else:
                                                        raiz.append('0')

                                                # Extract pedregosidad

        pedrego=re.findall(r"(?<=[F,f]ragmentos)[^\;]+",i)
                                                print(pedrego)

                                                # extract the abundance of the rocks using
regular expressions
                                                if len(pedrego)!=0:

        ninguno=re.findall(r"(?<=[N,n]ingun[o,a])[^\;]+",pedrego[0])
                                                        mpoco=re.findall(r"(?<=[M,m]uy
poc[o,a])[^\;]+",pedrego[0])

        comun=re.findall(r"(?<=com[ú,u]n)[^\;]+",pedrego[0])
```

```
517
518             much=re.findall(r"(?<=much[o,a])[^\;]+",pedrego[0])
519
520             abundante=re.findall(r"(?<=abundante)[^\;]+",pedrego[0])
521
522             dominante=re.findall(r"(?<=dominante)[^\;]+",pedrego[0])
523
524
525
526                                                             # reclass the rock abundance
527     values in numbers
528                                                             if len(ninguno)==0:
529                                                                     Pedregosidad.append('0')
530                                                             elif len(mpoco)!=0:
531                                                                     Pedregosidad.append('1')
532                                                             elif len(comun)!=0:
533
534             Pedregosidad.append('3.5')
535                                                             elif len(much)!=0:
536                                                                     print('piedra')
537                                                                     Pedregosidad.append('10')
538                                                             elif len(abundante)!=0:
539                                                                     Pedregosidad.append('10')
540                                                             elif len(dominante)!=0:
541                                                                     Pedregosidad.append('60')
542
543                                                                     else:
544                                                                     Pedregosidad.append('0')
545
546                                             # paste all the information in the dictionary
547                                             d_rp.update({
548                                                     'Raices':raiz,
549                                                     'Pedregosidad':Pedregosidad})
550                                             print(d_rp)
551
552                                             # paste all in the dataframe if not write the error
553                                             try:
554                                                     df_rp=pd.DataFrame(d_rp)
555                                             except Exception as e:
556                                                     df_rp=pd.DataFrame({'Horizonte/ Capa':[None]})
```

```
                                        log=open('descripcion_errores.txt','a')
                                        log.write('error  {} en {}'.format(e,sheet.title))
                                        log.close()
# Extract info from table 7
elif sheet.title=='Table 7': # physical description
        sheet.delete_rows(2)# drop problematic row
        print('\n\tTABLA 7:')
        for i, col in enumerate(sheet.iter_cols(min_row=1,values_only=True)):
                print(col)

                # check the column 'Horizonte/ Capa' and extract the information
                if col[0]=='Horizonte/ Capa':
                        if col==('Horizonte/ Capa',):
                                pass
                        elif '\n' in str(col[1]):
                                columna=col[1].split('\n')
                                print('Horizonte/ Capa', columna)

                                if ' ' in columna[1]:

                                        colum=columna[1].split(' ')
                                        colum.insert(0,columna[0])
                                        colum.insert(0,'Horizonte/ Capa')
                                        print('Horizonte', colum)
                                        d2.update({'Horizonte/ Capa':colum[1:]})
                                else:

                                        col=[x for x in columna]
                                        col.insert(0,'Horizonte/ Capa')
                                        print('Horizonte', col)
                                        d2.update({'Horizonte/ Capa':col[1:]})
                        else:
                                d2.update({'Horizonte/ Capa':list(col[1:])})
```

```
597
598                                     # extract the different features from the table based on the information
599     of the 'Horizonte/ Capa' field
600                                     extract_info('Profundidad (cm)',col,'Horizonte/ Capa',d2,d2)
601                                     extract_info('Arena',col,'Horizonte/ Capa',d2,d2)
602                                     extract_info('Limo',col,'Horizonte/ Capa',d2,d2)
603                                     extract_info('Arcilla',col,'Horizonte/ Capa',d2,d2)
604                                     #extract_info('Clase textural',col,'Horizonte/ Capa',d2,d2)
605                                     extract_info('Da (g/cm',col,'Horizonte/ Capa',d2,d2)
606                                     extract_info('Porosidad (%)',col,'Horizonte/ Capa',d2,d2)
607                                     extract_info('CC',col,'Horizonte/ Capa',d2,d2)
608                                     extract_info('PMP',col,'Horizonte/ Capa',d2,d2)
609
610
611                                     # create dataframe df2 from dictionary d2 this will hold the
612     information of the Horizons
613                                     try:
614                                         df2=pd.DataFrame(d2)
615                                         df2.insert(0,'ID_Nac',perfil)
616                                     except Exception as e:
617                                         df2=pd.DataFrame({'Horizonte/ Capa':[None]})
618                                         log=open('descripcion_errores.txt','a')
619                                         log.write('error {} en {}'.format(e,sheet.title))
620                                         log.close()
621
622
623
624                                     print(d2)
625
626
627                         # Chemical composition
628                         elif sheet.title=='Table 9':
629                             sheet.delete_rows(2)
630                             print('\n\tTABLA 9:')
631                             for col in sheet.iter_cols(min_row=1,values_only=True):
632                                 print(col)
633
634
635                                 # Extract information from the field 'Horizonte/ Capa'
636                                 if col[0]=='Horizonte/ Capa':
```

```
637                                     if col==('Horizonte/ Capa',):
638                                         pass
639                                     elif '\n' in str(col[1]):
640                                         columna=col[1].split('\n')
641                                         print('Horizonte/ Capa', columna)
642
643                                         g=[x for x in columna]
644
645
646                                         for k, i in enumerate(columna):
647                                             if r'' '' in i:
648                                                 s=i.split(' ')#
649
650                                                 s=s[::-1] # reverse the list
651                                                 g.pop(k) # pop the element that are
652 joined
653                                                 for j in s:
654                                                     g.insert(k,j) # we insert
655 the separated value in the copy of the list, in the value where the joined data was
656                                             else:
657                                                 pass
658
659                                         colum=g
660                                         colum.insert(0,'Horizonte/ Capa')
661                                         print('Horizonte', colum)
662                                         d3.update({'Horizonte/ Capa':colum[1:]})
663
664
665                                     else:
666                                         d3.update({'Horizonte/ Capa':list(col[1:])})
667
668
669
670
671                             # extract the features from the table based on a reference field 'Horizonte/
672 Capa'
673
674                                 extract_info('pH', col,'Horizonte/ Capa',d3,d3)
675                                 extract_info('N',col,'Horizonte/ Capa',d3,d3)
676                                 extract_info('P',col,'Horizonte/ Capa',d3,d3)
```

```
                    extract_info('K',col,'Horizonte/ Capa',d3,d3)
                    extract_info('Ca',col,'Horizonte/ Capa',d3,d3)
                    extract_info('Mg',col,'Horizonte/ Capa',d3,d3)
                    extract_info('S',col,'Horizonte/ Capa',d3,d3)
                    extract_info('Zn',col,'Horizonte/ Capa',d3,d3)
                    extract_info('Cu',col,'Horizonte/ Capa',d3,d3)
                    extract_info('Fe',col,'Horizonte/ Capa',d3,d3)
                    extract_info('Mn',col,'Horizonte/ Capa',d3,d3)
                    extract_info('B',col,'Horizonte/ Capa',d3,d3)



            # try to match long of the elements in the dataframe and paste to df3
            try:
                    igualar_longitud(d3,'Horizonte/ Capa')
                    df3 = pd.DataFrame(d3)
            except Exception as e:
                    df3 = pd.DataFrame({'Horizonte/ Capa':[None]})
                    log = open('descripcion_errores.txt','a')
                    log.write('\nArchivo:  {} {}.\n Error {}'.format(file,sheet.title,e))
                    log.close()
                    print(df3)



# Second table of chemical composition
elif sheet.title == 'Table 10':
            print('\n\tTABLA 10:')
            for col in sheet.iter_cols(min_row=1,values_only=True):
                    print(col)

                    # extract info from the Horizon field
                    if col[0] =='Horizonte/ Capa':
                            if col[1] == None:
                                    pass
                            elif '\n' in str(col[1]):
                                    columna=col[1].split('\n')
```

```
                                        print('Horizonte/ Capa', columna)

                                        g=[x  for x in columna]


                                        for k, i in enumerate(columna):
                                                if r" " in i:
                                                        s=i.split(' ')#

                                                        s=s[::-1] # reverse the list
                                                        g.pop(k) # extraxct the element of
the lsit that is joint
                                                        for j in s:
                                                                g.insert(k,j) # insert the
separated value in the copy of the list in the value where the data was together
                                                else:
                                                        pass

                                        colum=g
                                        colum.insert(0,'Horizonte/ Capa')
                                        print('Horizonte', colum)
                                        d4.update({'Horizonte/ Capa':colum[1:]})


                                else:
                                        d4.update({'Horizonte/ Capa':list(col[1:])})


                        # extract information from each of the columnso of interest in the
layer using as reference the dictionary 4
                        extract_info('CE\n(ds/m)',col,'Horizonte/ Capa',d4,d4,'(ds/m)')
                        extract_info('MO(%)',col,'Horizonte/ Capa',d4,d_ref=d4)
                        extract_info('CO(%)',col,'Horizonte/ Capa',d4,d_ref=d4)
                        extract_info('Nitrógeno Total(%)',col,'Horizonte/ Capa',d4,d_ref=d4)
                        extract_info('C / N',col,'Horizonte/ Capa',d4,d_ref=d4)
                        extract_info('Ca/Mg',col,'Horizonte/ Capa',d4,d_ref=d4)
                        extract_info('Mg/K',col,'Horizonte/ Capa',d4,d_ref=d4)
                        extract_info('[Ca+Mg]/K',col,'Horizonte/ Capa',d4,d_ref=d4)
                        extract_info('Al+H',col,'Horizonte/ Capa',d4,d_ref=d4)
```

```
                                        # try convert the dictionary into a dataframe
                                        try:
                                                igualar_longitud(d4,'Horizonte/ Capa')
                                                df4=pd.DataFrame(d4)

                                        # otherwise convert rise a log error file
                                        except Exception as e:
                                                df4=pd.DataFrame({'Horizonte/ Capa':[None]})
                                                log=open('descripcion_errores.txt','a')
                                                log.write('\nArchivo:  {} {}.\n Error
{}'.format(file,sheet.title,e))
                                                log.close()


                        # Third table of checmial composition
                        # this one qas morestable so the columns could be extracted directly
                        elif sheet.title=='Table 11':
                                sheet.delete_rows(2)
                                print('\n\tTABLA 11:')
                                for col in sheet.iter_cols(min_row=1,values_only=True):
                                        print(col)
                                        if col[0]=='Horizonte/ Capa':
                                                d5.update({'Horizonte/ Capa':list(col[1:])})
                                        if col[0]=='Na':
                                                d5.update({'Na_t9':list(col[1:])})
                                        elif col[0]=='K':
                                                d5.update({'K_t9':list(col[1:])})
                                        elif col[0]=='Ca':
                                                d5.update({'Ca_t9':list(col[1:])})
                                        elif col[0]=='Mg':
                                                d5.update({'Mg_t9':list(col[1:])})
                                        elif col[0]=='Suma Bases':
                                                d5.update({'Suma Bases':list(col[1:])})
                                        try:
                                                if 'CIC' in col[0]:
                                                        d5.update({'CIC':list(col[1:])})
                                                elif 'SB' in col[0]:
                                                        d5.update({'SB(%)':list(col[1:])})
```

```
                                except Exception as e:
                                        if col[0]=='CIC':
                                                d5.update({'CIC':list(col[1:])})
                                        elif col[0]=='SB':
                                                d5.update({'SB(%)':list(col[1:])})

                                # convert into a dataframe
                                try:
                                        df5=pd.DataFrame(d5)
                                # rise a log exception
                                except Exception as e:
                                        df5=pd.DataFrame({'Horizonte/ Capa':[None]})
                                        log=open('descripcion_errores.txt','a')
                                        log.write('\nArchivo:  {} {}.\nError
{}'.format(file,sheet.title,e))
                                        log.close()

                # table for saline chemical determinations
                elif sheet.title=='Table 12':
                        sheet.delete_rows(2)
                        print('\n\tTABLA 12:')
                        # values are extracted directly from the column values and converted into a
directionary, then they are turned into a dataframe
                        for col in sheet.iter_cols(min_row=1,values_only=True):
                                print(col)
                                if col[0]=='Horizonte/ Capa':
                                        d6.update({'Horizonte/ Capa':list(col[1:])})
                                if col[0]=='pH':
                                        d6.update({'pH_sal':list(col[1:])})
                                if col[0]=='Na':
                                        d6.update({'Na_sal':list(col[1:])})
                                elif col[0]=='K':
                                        d6.update({'K_sal':list(col[1:])})
                                elif col[0]=='Ca':
                                        d6.update({'Ca_sal':list(col[1:])})
                                elif col[0]=='Mg':
                                        d6.update({'Mg_sal':list(col[1:])})
                                elif col[0]=='Suma':
                                        d6.update({'Suma_Bases_sal':list(col[1:])})
                                elif col[0]=='Cl':
```

```
                                        d6.update({'Cl':list(col[1:])})
                                elif col[0]=='SB(%)':
                                        d6.update({'SB(%)':list(col[1:])})
                                elif col[0]=='CO 3':
                                        d6.update({'CO3':list(col[1:])})
                                elif col[0]=='SO 4':
                                        d6.update({'SO4':list(col[1:])})
                                elif col[0]=='RAS':
                                        d6.update({'RAS':list(col[1:])})
                                elif col[0]=='PSI':
                                        d6.update({'PSI':list(col[1:])})
                                try:
                                        if 'C.E'in col[0]:
                                                d6.update({'CE_sal':list(col[1:])})
                                except Exception as e:
                                        if col[0]=='C.E':
                                                d6.update({'CE_sal':list(col[1:])})


                                # convert into data frame or
                                try:
                                        df6=pd.DataFrame(d6)
                                except Exception as e:
                                        df6=pd.DataFrame({'Horizonte/ Capa':[None]})
                                        log=open('descripcion_errores.txt','a')
                                        log.write('\nArchivo: {} {}.\n Error
{}'.format(file,sheet.title,e))
                                        log.close()


        # Merge the data frame with the horizon information from the file (df2) with the dataframe the
whole file (dff2)
        # the horizons obtained in the df2 will be the reference field for joining the other dataframes
(df3,df4,df5,df6,df_rp)
        print(d2,d3,d4,d5,d6)
        try:
                t=df2.merge(dff2, on='Horizonte/ Capa', how='left')
        except Exception as e:
                log=open('descripcion_errores.txt','a')
                log.write('\nArchivo: {} {}.\n Error {}'.format(file,sheet.title,e))
```

```
                log.close()
                #print(t)
        # Merge the data frames that correspond to the information extracted from each layer.
        # If the length of the dataframe is zero, joining that dataframe is discarded.
        for i in [df3,df4,df5,df6,df_rp]:
                print(i)
                if len(i)==0:
                        pass
                else:
                        try:
                                t=t.merge(i, on='Horizonte/ Capa', how='left')
                        except Exception as e:
                                t=''
                                log=open('descripcion_errores.txt','a')
                                log.write('\nArchivo: {}.\nFallo en merge: {}\n'.format(file,e))
                                log.close()


        # merge all the dataframes into the daframe with all the fields.
        # the dataframe for exporting the data
        try:
                df_final=df_final.append(t)
                log_completos=open('estado_archivos_completos.txt','a')
                log_completos.write('Archivo: {}\n tabla 9 \n{}\ntabla10\n{}\n'.format(file,df3,df4))
                log_completos.close()
        except Exception as e:
                log=open('descripcion_errores.txt','a')
                log.write('\nArchivo {}.Error en append: {}\n'.format(file,e))
                log.close()
                log_resumen=open('archivos_problematicos.txt','a')
                log_resumen.write('Archivo: {}\n'.format(file))
                log_resumen.close()



# Create a excel file from the final dataframe
df_final.to_excel("BD_IEE_Merge_1.xlsx")  # doctest: +SKIP
```