```
classdef CohortFiles
    %COHORTFILES Files for Normal, TBI, and Stroke Cohorts
    %   List of functions to better use cohort files and read medical
    %   records
    %
    %   Some functions require the Text Analytics Toolbox
    %
    %   Authors:
    %       Michael Caiola (Michael.Caiola@fda.hhs.gov)
    %       Meijun Ye (Meijun.Ye@fda.hhs.gov)
    %
    %   Disclaimer: This software and documentation (the "Software") were
    %   developed at the Food and Drug Administration (FDA) by employees of
    %   the Federal Government in the course of their official duties.
    %   Pursuant to Title 17, Section 105 of the United States Code,
    %   this work is not subject to copyright protection and is in the
    %   public domain. Permission is hereby granted, free of charge, to any
    %   person obtaining a copy of the Software, to deal in the Software
    %   without restriction, including without limitation the rights to
    %   use, copy, modify, merge, publish, distribute, sublicense, or sell
    %   copies of the Software or derivatives, and to permit persons to
    %   whom the Software is furnished to do so. FDA assumes no
    %   responsibility whatsoever for use by other parties of the Software,
    %   its source code, documentation or compiled executables, and makes
    %   no guarantees, expressed or implied, about its quality,
    %   reliability, or any other characteristic. Further, use of this code
    %   in no way implies endorsement by the FDA or confers any advantage
    %   in regulatory decisions. Although this software can be
    %   redistributed and/or modified freely, we ask that any derivative
    %   works bear some notice that they are derived from it, and any
    %   modified versions bear some notice that they have been modified.
    %   The Software is not intended to make clinical diagnoses or to be
    %   used in any way to diagnose or treat subjects for whom the EEG is
    %   taken.

    properties
        AllFiles
        NormalFiles
        Normal_idx
        NormalRecord
        TBIFiles
        TBI_idx
        TBIRecord
        StrokeFiles
        Stroke_idx
        StrokeRecord
        UnknownFiles
        Unknown_idx
        UnknownRecord
        A
    end
    methods
        function obj = CohortFiles(A)
            %COHORTFILES Construct an instance of this class
            %   Detailed explanation goes here
            if isstring(A)
                if endsWith(A,".xlsx")
                    A = ConvertToTable(A);
                else

                end
            end

            obj.A = A;
            obj.AllFiles = A.Filename;
            obj.Normal_idx = or(A.Category=="Normal",A.Category=="HEA");
            obj.NormalFiles = A.Filename(obj.Normal_idx);
            obj.NormalRecord = A.("Full Note");
            obj.TBI_idx= A.Category=="TBI";
            obj.TBIFiles = A.Filename(obj.TBI_idx);
            obj.TBIRecord = A.("Full Note");
            obj.Stroke_idx = or(A.Category=="Stroke",A.Category=="STR");
            obj.StrokeFiles = A.Filename(obj.Stroke_idx);
            obj.StrokeRecord = A.("Full Note");
            obj.Unknown_idx = A.Category=="Unknown";
            obj.UnknownFiles = A.Filename(obj.Unknown_idx);
            obj.UnknownRecord = A.("Full Note");
        end

        function y = NumFiles(obj)
            y = [sum(obj.A.Category == "Normal") sum(obj.A.Category == "TBI") sum(obj.A.↙
Category == "Stroke")];
            disp("Number of Total files: " + height(obj.A));
            disp("Number of Normal files: " + y(1));
            disp("Number of TBI files: " + y(2));
            disp("Number of Stroke files: " + y(3));
        end

        function y = NumSubjects(obj)
            y = [numel(unique(obj.A.Subject(obj.A.Category == "Normal"))),...
                numel(unique(obj.A.Subject(obj.A.Category == "TBI"))),...
                numel(unique(obj.A.Subject(obj.A.Category == "Stroke")))];
            disp("Number of Total subjects: " + numel(unique(obj.A.Subject)));
            disp("Number of Normal subjects: " + y(1));
            disp("Number of TBI subjects: " + y(2));
            disp("Number of Stroke subjects: " + y(3));
        end

        function outputArg = method1(obj,inputArg)
            %METHOD1 Summary of this method goes here
            %   Detailed explanation goes here
            outputArg = obj.Property1 + inputArg;
        end

        function [inT,inA] = Index(obj,T)
            T = [T.Subject,T.Session];
            B = [obj.A.Subject,obj.A.Session];
            [inT,inA] = ismember(T,B,'rows');
        end

        function files = Compare(obj,T,varargin)
            files = {zeros(height(T),length(varargin)),zeros(height(T),length(varargin)),↙
zeros(height(T),length(varargin))};
            C = T.Category;
            T = [T.Subject,T.Session];
            B = [obj.A.Subject,obj.A.Session];
            for k = 1:length(varargin)
                files{1}(:,k) = and(ismember(T,B(obj.Normal_idx,:),'rows'), C == varargin↙
{k});
                files{2}(:,k) = and(ismember(T,B(obj.TBI_idx,:),'rows'), C == varargin↙
{k});
                files{3}(:,k) = and(ismember(T,B(obj.Stroke_idx,:),'rows'), C == varargin↙
{k});
                %files{1}(:,k) = all([ismember(T.Subject,obj.A.Subject(obj.Normal_idx)) ,↙
ismember(T.Session,obj.A.Session(obj.Normal_idx)) , T.Category == varargin{k}],2);
                %files{2}(:,k) = all([ismember(T.Subject,obj.A.Subject(obj.TBI_idx)) ,↙
ismember(T.Session,obj.A.Session(obj.TBI_idx)) , T.Category == varargin{k}],2);
                %files{3}(:,k) = all([ismember(T.Subject,obj.A.Subject(obj.Stroke_idx)) ,↙
ismember(T.Session,obj.A.Session(obj.Stroke_idx)) , T.Category == varargin{k}],2);
            end
            tit = ["Normal", "TBI", "Stroke"];
            for i = 1:3
                figure;
                X = categorical(string(varargin));
                X = reordercats(X,string(varargin));
                b = bar(X,sum(files{i}));
                %b.FaceColor = "flat";
                %b.CData(i,:) = [.5 0 .5];
                title("Number of files from " + inputname(2) +" Table in " + tit(i) + " "↙
+ inputname(1))
                xlabel("Comparison Table")
            end
        end

        function files = Compare1(obj,T,varargin)
            for i = length(varargin): -1: 1
                files{i} = zeros(height(obj.A),length(varargin));
            end
            C = T.Category;
            T = [T.Subject,T.Session];
            B = [obj.A.Subject,obj.A.Session];
            for k = 1:length(varargin)
```

```
                files{k}(:,1) = and(ismember(B,T(C == varargin{k},:),'rows'), obj.A.↙
Category == "Normal");
                files{k}(:,2) = and(ismember(B,T(C == varargin{k},:),'rows'),   obj.A.↙
Category == "TBI");
                files{k}(:,3) = and(ismember(B,T(C == varargin{k},:),'rows'), obj.A.↙
Category == "Stroke");
            end
            tit = ["Normal", "TBI", "Stroke"];
            for i = 1:length(varargin)
                figure;
                X = categorical(string(tit));
                X = reordercats(X,string(tit));
                b = bar(X,sum(files{i}));
                %b.FaceColor = "flat";
                %b.CData(i,:) = [.5 0 .5];
                title("Number of files from " + inputname(1) +" Table in " + varargin{i}↙
+ " " + inputname(2))
                xlabel("Comparison Table")
            end
        end
        function PlotWordClouds(obj,thresh)
            if nargin <2
                thresh = 50;
            end
            pat = regexpPattern('[A-Z\s]*[A-Z]+:');
            wc = wordCloudCounts(obj.A.("Full Note"));
            wc = wc{1:thresh,1};

            x = erase(obj.A.("Full Note")(obj.A.Category=="Normal"),pat); % Remove↙
capital headings
            x = erase(x,wc);
            x = erase(x," ic ");
            x = erase(x,"BPM");
            x = erase(x,"Digital");
            x = erase(x,"Hz");
            x = erase(x,"epileptiform");
            x = replace(x,regexpPattern('\s[a-z][\s\*\)\.\,]')," ");
            x = replace(x,"  "," ");

            y = erase(obj.A.("Full Note")(obj.A.Category=="TBI"),pat);
            y = erase(y,wc);
            y = erase(y," ic ");
            y = erase(y,"BPM");
            y = erase(y,"Digital");
            y = erase(y,"Hz");
            y = erase(y,"epileptiform");
            y = replace(y,regexpPattern('\s[a-z][\s\*\)\.\,]')," ");
            y = replace(y,"  "," ");

            z = erase(obj.A.("Full Note")(obj.A.Category=="Stroke"),pat);
            z = erase(z,wc);
            z = erase(z," ic ");
            z = erase(z,"BPM");
            z = erase(z,"Digital");
            z = erase(z,"Hz");
            z = erase(z,"epileptiform");
            z = replace(z,regexpPattern('\s[a-z][\s\*\)\.\,]')," ");
            z = replace(z,"  "," ");

            figure;
            wordcloud(x)
            title("Normal")
            figure;
            wordcloud(y)
            title("TBI")
            figure;
            wordcloud(z)
            title("Stroke")
        end
    end
end

function T = ConvertToTable(txt)
T = readtable(txt);
filename = "";
session = [];
categ = "";
for i = 1:height(T)
    s = split(string(T.Location(i)),'\');
    %s = split(s(end),"_");
    filename(i) = s(end);
    session(i) = str2double(T.Session{i}(end-2:end));
    categ(i) = categorical(missing);
end
T.Filename = filename';
T.Session = session';
T.Category = categ';
T = renamevars(T,"Notes","Full Note");
end
```