This is a dataset for email classification on whether or not an email is spam.

Dataset obtained from: <https://archive.ics.uci.edu/dataset/94/spambase> on 29/11/2024

Sources:

* Creators: Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt Hewlett-Packard Labs, 1501 Page Mill Rd., Palo Alto, CA 94304
* Donor: George Forman (gforman at nospam hpl.hp.com) 650-857-7835
* Generated: June-July 1999

**ALL WRITTEN CODE IS MADE USING PYTHON ALONGSIDE THE PANDAS LIBRARY**

**Data can be found in the .data file “spambase.data”, which can be accessed using any text processing apps, such as Notepad.**

A dataset tablization code was written to turn the dataset into a table in csv format for simple and easy access and readability.

**The code can be found in the python file “save\_spambase.py”**

After successfully saving it as “spambase\_raw.csv”, we converted it to an Excel file format (xlsx) for feature preservation.

**A table of the raw dataset can be accessed in the file “spambase\_raw.xlsx”**

Information about the dataset:

* Number of attributes (columns): 58 (57 continuous, 1 nominal class label)
* Number of instances (rows): 4601 (With 1813 OR 39.4% being spam)
* The last column of the dataset is what denotes if the email is spam or not.
* Most attributes indicate whether a specific word or character was frequently occuring in the email.
* 48 of the attributes are continuous values of range [0,100] and type word\_freq\_WORD = percentage of words in the email that match WORD (i.e. how frequently the word “order” was mentioned in the email).
* 6 of the attributes are continuous values of range [0,100] and type word\_freq\_CHAR = percentage of characters in the email that match CHAR (i.e. how frequently the character “!” was mentioned in the email)
* 1 of the attributes are continuous values of range [1,...] and type capital\_run\_length\_average = average length of uninterrupted sequences of capital letters
* 1 of the attributes are continuous values of range [1,...] and type capital\_run\_length\_longest = length of longest uninterrupted sequences of capital letters
* 1 of the attributes are continuous values of range [1,...] and type capital\_run\_length\_total = total number of capital letters
* 1 nominal {0,1} class attribute = demotes whether the email is spam (1) or not spam (0)

Distribution of classes:

* Spam = 1813 (39.4%)
* Non-Spam = 2788 (60.6%)

An inconsistency detection code was written to find all the inconsistencies in the dataset (Missing values, duplicate rows, outliers, etc)

**The code can be found in the python file “Inconsistency Detection.py”**

Here is a summary of the results:

* Missing Values: 0
* Inconsistent Datatypes: 0
* Duplicate Rows: 391
* Outliers Detected: Yes
* Negative Values: No

Here is the result straight from the code output:

Missing Values:

 Series([], dtype: int64)

Data Types:

 word\_freq\_make                float64

word\_freq\_address             float64

word\_freq\_all                 float64

word\_freq\_3d                  float64

word\_freq\_our                 float64

word\_freq\_over                float64

word\_freq\_remove              float64

word\_freq\_internet            float64

word\_freq\_order               float64

word\_freq\_mail                float64

word\_freq\_receive             float64

word\_freq\_will                float64

word\_freq\_people              float64

word\_freq\_report              float64

word\_freq\_addresses           float64

word\_freq\_free                float64

word\_freq\_business            float64

word\_freq\_email               float64

word\_freq\_you                 float64

word\_freq\_credit              float64

word\_freq\_your                float64

word\_freq\_font                float64

word\_freq\_000                 float64

word\_freq\_money               float64

word\_freq\_hp                  float64

word\_freq\_hpl                 float64

word\_freq\_george              float64

word\_freq\_650                 float64

word\_freq\_lab                 float64

word\_freq\_labs                float64

word\_freq\_telnet              float64

word\_freq\_857                 float64

word\_freq\_data                float64

word\_freq\_415                 float64

word\_freq\_85                  float64

word\_freq\_technology          float64

word\_freq\_1999                float64

word\_freq\_parts               float64

word\_freq\_pm                  float64

word\_freq\_direct              float64

word\_freq\_cs                  float64

word\_freq\_meeting             float64

word\_freq\_original            float64

word\_freq\_project             float64

word\_freq\_re                  float64

word\_freq\_edu                 float64

word\_freq\_table               float64

word\_freq\_conference          float64

char\_freq\_;                   float64

char\_freq\_(                   float64

char\_freq\_[                   float64

char\_freq\_!                   float64

char\_freq\_$                   float64

char\_freq\_#                   float64

capital\_run\_length\_average    float64

capital\_run\_length\_longest      int64

capital\_run\_length\_total        int64

class                           int64

dtype: *object*

Number of Duplicate Rows: 391

Outliers detected in each numerical column:

 word\_freq\_make                 90

word\_freq\_address              43

word\_freq\_all                  94

word\_freq\_3d                   13

word\_freq\_our                  81

word\_freq\_over                104

word\_freq\_remove               99

word\_freq\_internet             77

word\_freq\_order               113

word\_freq\_mail                 74

word\_freq\_receive             100

word\_freq\_will                102

word\_freq\_people               89

word\_freq\_report              106

word\_freq\_addresses            99

word\_freq\_free                 69

word\_freq\_business             97

word\_freq\_email               106

word\_freq\_you                  60

word\_freq\_credit               76

word\_freq\_your                 87

word\_freq\_font                 57

word\_freq\_000                 107

word\_freq\_money                32

word\_freq\_hp                   86

word\_freq\_hpl                 105

word\_freq\_george              123

word\_freq\_650                 107

word\_freq\_lab                  64

word\_freq\_labs                 90

word\_freq\_telnet               61

word\_freq\_857                  51

word\_freq\_data                 77

word\_freq\_415                  52

word\_freq\_85                   82

word\_freq\_technology           77

word\_freq\_1999                105

word\_freq\_parts                21

word\_freq\_pm                   69

word\_freq\_direct               61

word\_freq\_cs                   62

word\_freq\_meeting              78

word\_freq\_original            107

word\_freq\_project              51

word\_freq\_re                   64

word\_freq\_edu                  74

word\_freq\_table                28

word\_freq\_conference           49

char\_freq\_;                    31

char\_freq\_(                    49

char\_freq\_[                    30

char\_freq\_!                    44

char\_freq\_$                    62

char\_freq\_#                    24

capital\_run\_length\_average     21

capital\_run\_length\_longest     68

capital\_run\_length\_total       86

class                           0

dtype: int64

After detection was successful, an inconsistency cleaning code was written to get rid of all the inconsistencies, in this case being the duplicate rows and outliers. We used the .drop\_duplicates() to remove duplicates.

Attempting to remove outliers removes a significant amount of the dataset (almost 2000 rows) due to its heavily skewed nature, especially since said outliers are valid variations in the data (aka not anomalies that negatively affected the dataset). So, we are not going to remove outliers.

**The code can be found in the python file “Inconsistency Cleaning.py”**

Here is a summary of the results:

* No. of Rows after removing duplicates = 4210

Results straight from the code output is unnecessary due to its similarity to the summary.

**A table of the pre-processed dataset can be found in the excel file “cleaned\_spambase.xlsx”**

Finally, we write code to convert the dataset back to its original .data format in order to allow our Machine Learning model to train on the dataset.

**The code can be found in the python file “Convert to data file.py”**

Here is a summary of the results:

* The data is successfully saved to .data format

**The data can now be found in the .data file “cleaned\_spambase.data”, which can be accessed using any text processing apps, such as Notepad.**