AI_Project_Neural_Network

Release 1.0

Bogna Lew, Bartosz Strzelecki, Krzysztof Nazar

May 24, 2022

CONTENTS:

CHAPTER

ONE

AI_PROJECT_NEURAL_NETWORK

1.1 blacklist module

This file includes arrays used in get_data functions

1.2 ff module

ff.**create_model**(*inputData*)

This functions creates neural network model.

Parameters

inputData vectorized_dt (text array transformed to document-term matrix)

Returns

model neural network model with layers ready to train

ff.**keras**(*inputData*)

This functions creates and trains neural network model.

Parameters

inputData vectorized_dt (text array transformed to document-term matrix)

Returns

model trained neural network model

ff.**keras_classify**(*model*)

This functions prints two classification reports:

- for training data.
- for testing data

Parameters

model trained neural network model

ff.**keras_permutations**(*inputData*)

This functions creates KerasClassifier and trains on training data. Permutation importance is calculated for each word in the model

Parameters

inputData trained neural network model

ff.**keras_shap**(*model*)

This functions plots SHAP value for each word on a graph.

Parameters

model trained neural network model

1.3 func module

func.**getFeatureNamesFromTFIDFVectorizer**(*arr*)

This functions removes useless words from the text.

Convert a collection of raw documents to a matrix of TF-IDF features. TF-IDF –> TF – term frequency, IDF – inverse document frequency. It is a numerical statistic that is intended to reflect how important a word is to a document.

Parameters:

arr An iterable which generates either str, unicode or file objects.

Returns:

featureNames feature names from vectorizer_dt got from texts basing on TF-IDF method

vectorizer_dt text array transformed to document-term matrix

func.**get_data**()

This functions imports the data basing on json file.

Returns

vectorizer_dt TF-IDF is a numerical statistic that is intended to reflect how important a word is to a document

labels save info about isAntiVaccine

featureNames feature names from vectorizer_dt got from texts basing on TF-IDF method

func.**get_important**(*vocab*)

This function gets only important words from the texts. The important words(vocab) are searched in the texts. The words are saved to the 'temp' array only if they exist in the provided data(texts).

Parameters:

vocab A list of important words (returned from permutationImportance() function)

Returns:

vectorizer_dt text array transformed to document-term matrix

featureNames feature names from vectorizer_dt got from texts basing on TF-IDF method

func.**removeUnnededWords**(*text*)

This functions removes useless words from the text.

Parameters:

text a text to analyze

Returns:

converted_text converted text

1.4 main module

1.5 metrics module

metrics.**permutationImportance**(*model*)

This functions calculates permutation importance for provided model.

Depending on using "takeXMostSignificantFeatures(impArray, namesArray, ss, numberOfFeatures=10)" or "takeAllFeatures(impArray, namesArray, ss)" the essential data will be returned from this function.

Parameters:

model data model, for example svc(C-Support Vector Classifier fitted to training data)

Returns:

impArray array containing the importance of each word

namesArray array containing all words

metrics.**plotImportant**(*namesArray*, *impArray*)

This functions creates a plot of importances of provided words.

Parameters:

impArray array to store value of importance of words

namesArray array to store value of words

metrics.**takeAllFeatures**(*impArray*, *namesArray*, *sortedWords*)

This functions takes all the important features.

Parameters:

impArray array to store value of importance of words

namesArray array to store value of words

sortedWords array with words sorted by the importance value

metrics.**takeXMostSignificantFeatures**(*impArray*, *namesArray*, *sortedWords*, *numberOfFeatures*)

This functions takes numberOfFeatures from the beginning and numberOfFeatures from the end of the set of words.

Parameters:

impArray array to store value of importance of words

namesArray array to store value of words

sortedWords array with words sorted by the importance value

numberOfFeatures integer value

1.6 som module

som.**plotSom**(*som*, *data*)

This function is used to plot som. Colors, sizes and data can be set here.

Parameters:

som trained SOM model

data words in an array

som.**som**(*data*)

This functions creates and trains SOM basing on provided data.

Parameters

data array of importance of words

Returns

vectorizer_dt TF-IDF is a numerical statistic that is intended to reflect how important a word is to a document

labels save info about isAntiVaccine

featureNames feature names from vectorizer_dt got from texts basing on TF-IDF method

som.**somWithClassInfo**(*data*)

SOM with additional column representing if the text is for or against vaccination.

Parameters:

data array of words from vectorizer

Returns:

model trained SOM model

1.7 svm module

svm.**classify**(*model*)

This function performs classification on samples in model, which is svm.SVC (C-Support Vector Classification). After classification confusion matrix and classification report are printed.

Parameters:

model data model –> svm.SVC (C-Support Vector Classification)

svm.**createSvmClassifier**()

This functions creates C-Support Vector Classifier.

Returns:

svc C-Support Vector Classifier fitted to training data

svm.**getImportantData**()

This function creates svm.SVC (C-Support Vector Classification) and performs classification on samples in this model. It gets important names basing on their value of importance.

Returns:

importantData array of texts containing only important words

1.8 variables module

1.9 word module

class word.**Word**(*name*, *importance*)

Bases: object

importance = 0.0

name = ''