Text Analysis by Natural Text Processing

Functions:

fit\_in\_batches(classifier, x\_train, y\_train, batch\_size): takes the classifier that we want to fit, the data we want to train, the data we make the train accordingly, and sizes of the batches. If the classifier has the partial.fit attribute, it divides the both datas into batches and fits those batches to make the fitting process faster, if not whole that is fitted simultaneously. Returns the classifier.

pre\_steps(data, vectorizer, compressor, classifier): takes the pulled data, vectorizes the data with the given vectorizer process. If any compressor is given,vectorized data is fitted and transformed accordingly. By k-folding, the compressed and target data is splitted into k=10 groups and we obtain train and test data. Then by using the fit\_in\_batches function we fit the given classifier. Return a matrix that has train data, test data, target train data, target test data and classifier as columns, in order.

K-Fold Cross Validation: Splits the data into groups and leaves one group for evaluation while training. Iterates k times until each group used for testing. Parameters:

n\_splits=10 to decide the number of folds,

random\_state=42 to control the ordering of the indices and the randomness of the each fold,

shuffle=True to shuffle the data before the splitting.

We create a list that includes every possible combination that we have

CountVectorizer, TfidfVectorizer as vectorizers,

IncrementalPCA, LDA as compressors or no compressor and

SGDClassifier, LogisticRegression, SVC as classifiers.

Model Details:

In order to analyse a text, we obtain the word distribution of the documents in form of matrices.

Count Vectorization: We assign the number of times a word is used to each cell of a term-document matrix, where each row represents a word and each column represents a document from the collected data. Simply, it counts the times a word appears in a document sp that the model is based on the word frequency, where is a document is represented by a count vector (Jurafsky & Martin, 2024).

Term Frequency Inverse Document Frequency (TF-IDF Vectorization): After calculating the term frequency, like in count vectorization, the occurrence of a word is divided by the total number of words in that document. With that we do not consider whether that a word have a high frequence because the document is longer than others. Then to understand the semantic meaning of a word, IDF gives higher values that is frequent in a document but less frequent considering all other documents (Yilmaz, Clarke, Messnarz, & Reiner, 2021, pp. 292–293).

To simplify the vectorized data set and not lose the important parts, we will apply dimensionality reduction methods.

Incremental Principal Component Analysis (IPCA): PCA depends on the variance of initial variables, thus it first performs standardization not to lost variables with smaller ranges. Then it computes the covariance matrix to inspect the relation between the variables, especially if there is a highly related variables, there may be disposable information. To determine the principle components of the data, that include the most of the information with minimal correlation, PCA calculates the eigenvectors and eigenvalues of the covariance matrix where the eigenvectors with higher eigenvalues are more significant principle components. Feature vector is created with those components that we want to keep as its columns. Lastly, it returns modified data with principle components, the multiplication of the transpose of the original data and the transpose of the feature vector (Smith, 2002).

Incremental PCA is another version of PCA for large data sets that don’t fit into memory. It performs PCA on batches of data. Parameters:

n\_components=50 to determine the number of components to keep,

batch\_size=1000 to determine the number of samples in each batch.

Linear Discriminant Analysis (LDA): LDA transforms the data into a lower dimensional space when training samples and their class labels are given. If the dimensions are bigger than the number of samples or if classes are non-linearly separable, then LDA fails LDA calculates the between-class variance and within-class variance, the distance between the means of different classes and the samples of each class, respectively. Maximizing the between-class variance and minimizing the within-class variance, data is transformed into the lower dimensional space (Tharwat, Gaber, Ibrahim, & Hassanien, 2017). Parameter:

n\_components=2 to determine the number of components for dimensionality reduction.

To categorize the documents into the corresponding target class, we use classifiers.

Stochastic Gradient Descent Classifier (SGDClassifier): Starting from a random point, gradient descent moves the point in the direction of the negative gradient, the magnitude of the movement depends on the learning rate also which determines the convergence of the algorithm. SGD selects a random observation and divides it into batches, computes the gradient of each batch and continues with the same process. Parameters:

loss="hinge" to use linear SVM for loss function,

penalty="l2" to use l2 as regularization term for linear SVM model,

alpha=1e-3 the constant that multiplies the regularization term,

random\_state=42 to shuffle the data and determine the observation number for each batch,

max\_iter=5 to determine the maximum number of iterations,

tol=None refers to no tolerance for stopping criterion, which is when the improvement of the objective function is less than the specified tolerance for the number of consecutive iterations.

LogisticRegression: Logistic Regression gives the possibility of the categorical output, that can be 0 or 1. It is fitted to an Sigmoid Function, so that the logistic function returns the likelihood of the predictions of the model, then accepts the predictions that are above the specified threshold (usually 0.5) and choose the best fit (Arya, 2022). Parameter:

max\_iter=1000 to choose the maximum number of iteration for find the best fit, for convergence.

Support Vector Classifier (SVC): Support Vector Machine (SVM) finds the best space, takes features as dimentions, that separate data points to respective classes. It separates the points through a hyperplane, where the plane denotes a boundary between the different classes (Kanade , 2022). This model Works fine with linearly separable data. For non-linear data sets it kernels to transform the points to a separable form. Parameters:

kernel='linear' to choose the Kernel type,

random\_state=  to have reproducible results and control the randomness.

In the for loop, we reinitialize the classifier so that we don’t have a classifier that is fitted in another group and add the obtained matrix in a results list and record to execution time for each combination.

For each combination, we calculate the cross validation score for each fold, mean score and the confidence interval of that model to obtain the its accuracy. After the calculation of each model we calculate the overall mean score and 95% confidence interval.

We create a dictionary of model properties, execution time, average cross-validation score of the folds, mean score and the confidence interval of that model to create an excel file with the information.

[1] Jurafsky, D., & Martin, J. H. (2024). Vector Semantics and Embeddings. In *Speech and Language Processing (3rd ed. draft)* (pp. 7, 12). Retrieved from <https://web.stanford.edu/~jurafsky/slp3/6.pdf>

[2] Yilmaz, M., Clarke, P., Messnarz, R., & Reiner, M. (2021). *Systems, Software and Services Process Improvement* (pp. 292–293). Springer Nature.

[3] Smith, L. (2002). *A tutorial on Principal Components Analysis Chapter 1*. Retrieved from <https://www.iro.umontreal.ca/~pift6080/H09/documents/papers/pca_tutorial.pdf>

[4] Tharwat, A., Gaber, T., Ibrahim, A., & Hassanien, A. E. (2017). Linear discriminant analysis: A detailed tutorial. *AI Communications*, *30*(2), 169–190. <https://doi.org/10.3233/aic-170729>

[5] Stojiljković, M. (n.d.). Stochastic Gradient Descent Algorithm With Python and NumPy – Real Python. Retrieved from Real Python website: <https://realpython.com/gradient-descent-algorithm-python/#:~:text=Stochastic%20gradient%20descent%20is%20an>

[6] Arya, N. (2022, April 4). Logistic Regression for Classification. Retrieved from KDnuggets website: <https://www.kdnuggets.com/2022/04/logistic-regression-classification.html>

[7] Kanade , V. (2022). All You Need to Know About Support Vector Machines. Retrieved July 24, 2024, from Spiceworks Inc website: https://www.spiceworks.com/tech/big-data/articles/what-is-support-vector-machine/#\_002