ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

Project Final Report

CENG 408
Innovative System Design and Development I

P2018-06

Multi-label Classification of News Text

Can Koral ADALI 201411002
Bihter ÖZUÇAK 201411048
Miray PADIR 201411050
Kardelen YILDIRIM 201311060

Advisor: *Erdoğan DOĞDU*
Co-Advisor: Roya *CHOUPANI*

Table of Contents

List of Figures

Abstract

Media organizations and news sites publish hundreds of articles and news articles from various agencies and sources, as well as editors and news reporters. While most of the news belong to more than one topic, determining which topics belong to the news provides great convenience to the readers on the internet. But reading each news carefully and determining the headlines individually increases the workload of editors. In this project, it is aimed to classify the topics of Turkish news texts with the help of web based application, Multi-label Classification of Text Documents, which will be developed using artificial intelligence. For this purpose, we aim to develop artificial intelligence models using deep learning methods such as LDA2Vec and Word2Vec, and to develop a system that successfully identifies the topics of news texts that they wish to classify by using these models. The workloads of the people who use this tool will not only ease, but the titles of the texts will be determined automatically, with little errors and very quickly.

Keywords:
Multi-label classification, Multi-class classification, word embedding, lda2vec, python

Özet

Medya kuruluşları ve haber siteleri, gün içinde ellerine çeşitli ajanslardan ve kaynaklardan ulaşan, aynı zamanda editörlerin ve haber muhabirlerinin yazdıkları yüzlerce makale ve haberi yayınlamaktadırlar. Günümüzde çoğu haber birden fazla konu başlığına ait olmakla birlikte, haberin hangi konu başlıklarına ait olduğunu belirlemek, internet üzerindeki okuyucular için büyük bir kolaylık sağlamaktadır. Fakat her haberi dikkatlice okuyup, konu başlıklarını tek tek belirlemek editörlerin iş yükünü arttırmaktadır. Bu projede yapay zeka kullanılarak geliştirilecek olan web tabanlı uygulama, "Multi-label Classification of Text Documents" (Dokümanların Çok-Etiketli Sınıflandırılması) sayesinde Türkçe haber metinlerinin konu başlıklarının kolaylıkla sınıflandırması amaçlanmaktadır. Bu amaçla LDA2Vec ve Word2Vec gibi derin öğrenme metotlarını kullanarak yapay zeka modelleri ortaya çıkarmayı, bu modelleri kullanarak sınıflandırılmak istenilen haber metinlerininin konu başlıklarını ilgi düzeylerine göre başarılı bir şekilde belirleyen bir sistem geliştirmeyi hedefliyoruz. Bu aracı kullanacak kişilerin iş yükleri hafiflediği gibi, metinlerin konu başlıklarının otomatik, az hatalı ve çok hızlı bir şekilde belirlenmesi mümkün olacaktır.

Anahtar Kelimeler:
Çok-Etiketli Sınıflandırma, Çok-Sınıflı Sınıflandırma, Word Embedding, Lda2vec, Python

1. Introduction

By the development of technology, there are a lot of information on the internet and day-by-day there is huge amount of increase in number. Every information/data needs to be classified by the subject of them. Our research covers what are the types of classification, how to classify them using the multi-label classification algorithms and usage cases of classification. In this paper, we study and mention that we're going to use word embedding with lda2vec as an algorithm for classifying news text as multi-labels. Moreover, we are going to build a web-interface with PHP so that, user can upload a news text and get the result of it as categories. We identify and state that there are many algorithms and methods can be used for classification.

1.1. Problem Statement

Nowadays, there are a lot of informations on Internet and every information has their own classification which can be related with medical, marketing, news and many more. Every information has and needs some kind of classification and it needs to be classified for quick access and preventing the data loss. The classification has been widely studied and it has more than one way to classify an information with computer which can be generalizable by one main topic which is Machine-learning.

Based on literature, "Machine-learning systems are used to identify objects in images, transcribe speech into text, match news items, posts or products with users' interests, and select relevant results of search. Increasingly, these applications make use of a class of techniques called deep learning." [1]

This study focuses on news text multilabel classification. News have different subjects and some news can have more than one subject that needs to be classified. So that every news text needs multilabel classification. For our study, we choose using Python language which is mainly used for deep learning. We decided to use Word Embedding with lda2vec which can be done with Python and build a web-interface with PHP which helps user to upload a news text and gets the results.

1.2. What Is Classification, Multiclass And Multilabel Classification?

The classification is a very important topic in data analysis. Every record in a dataset has group of attributes and one of them is class. Classification can be done by creating a model from objects that has been classified. The reason behind this is classifying the unknown objects as right as possible. “There are many classification approaches for extracting knowledge from data such as divide-and conquer [2] and separate-and-conquer [3].” [4]

In machine learning, the multiclass classification is the main problem because application of classification requires a difference between classes and many classes are similar. As an example to this situation can be given as handwritten “character recognition” [5][6], “part-of-speech tagging” [7][8], “speech recognition” [9] and “text categorization” [10][11].

There are three common approaches for multiclass classification. Those are: OvA (One versus all), AvA (All versus all) and WTA (Winner take all). OvA, directly uses binary classifiers to encode and assumes that there is a separator between found class and another classes. AvA, assumes there is a separator between every two classes. “WTA is an expressive classifier [12], it has limited expressivity when trained using the OvA assumption since OvA assumes that each class can be easily separated from the rest.” [13]

Multilabel classification is another very important topic for data analysis which is the main problem under the machine learning branch. “Multi-label classification is nothing but the variants of classification problem in which different target labels should be allocated to every instance. Multi-label classification is different from the multiclass classification. In general, multi-label classification is defined as problem of searching model which maps the input to binary vectors, rather than outputs in scalars.” [14]

There are two methods for solving multilabel classification problem. Those are: problem transformation and algorithm adaptation. “In problem transformation approaches, multi-label classification problem is transformed to binary classification problems set and this can be further processed through single class classifiers. In algorithm adaptation approaches, algorithms are adapted in order to perform the multi-label classification directly.” [14]

1.3. Algorithms of Multi-label Classification

There are several ways to classify a text with Python language. Those are mainly Word Embedding, Text / NLP, SVM, Deep Neural Networks, BR, CC, CDN and LC.

1. Word Embedding
    - Count Vectors
    - TF-IDF Vectors
        - Word Level TF-IDF
        - N-gram Level TF-IDF
        - Character Level TF-IDF
    - LDA
    - Word2vec
    - lda2vec
2. Natural Language Processing (NLP)
3. Support Vector Machine (SVM)
4. Deep Neural Networks
    - Convolutional Neural Network (CNN)
    - Recurrent Neural Network (RNN)
    - Long Short Term Model (LSTM)
5. Binary Relevance (BR)
6. Classifier Chains (CC)
7. Conditional Dependency Network (CDN)
8. Label Combination (LC)

These are explained below and have their own code snippet as an example usage of these methods.

1.3.1. Word Embedding

Word embedding most known ways that transform text to numeric symbols. For instance Google, Amazon etc. using word embedding method. According to researches about word embedding: “Word embeddings are global representations of word properties learned from the context distribution of words. Words are usually ambiguous and belong to multiple classes, e.g., multiple part-of-speech tags or multiple meanings. A good word embedding should represent all information about the word, including its multiple classes. “[15]

1.3.1.1. Count Vectors

Count Vector is a matrix notation of the data set in which each row represents a document from the corpus, where each column represents a term from the corpus and each cell represents the frequency number of a given term in a given document. [17]

1.3.1.2. TF-IDF Vectors

The TF-IDF score represents the importance of the term and the whole corpus. TF-IDF score consists of two periods: the first term the normalized Term Frequency (TF), the second term Inverse Document Frequency (IDF), the logarithm of the number of documents in the corpus is calculated by dividing the number of documents with a specific term. [17]

1.3.1.2.1. Word Level TF-IDF

Matrix representing tf-idf scores of each term in different documents. [17]

1.3.1.2.2. N-gram Level TF-IDF

N-grams are the use of N terms. This Matrix represents the tf-idf scores of N-grams. [17]

1.3.1.2.3. Character Level TF-IDF

The matrix represents the tf-idf points of the character level n-grams in the corpus. [17]

1.3.1.3. TF-IDF Vectors

Latent Dirichlet Allocation is the coordinate transformation technique which is the development of principal component analysis. It aims to find a new coordinate system by minimizing the ratio of (scattering in class)/(scattering between classes).

1.3.1.4. Word2vec

Word2Vec is an algorithm toolkit that allows you to calculate the distance between words in vector. [18]

1.3.1.5. Lda2vec

Lda2vec designed from word2vec and LDA. The brief explanation about lda2vec from inventor is: "The lda2vec model tries to mix the best parts of word2vec and LDA into a single framework. word2vec captures powerful relationships between words, but the resulting vectors are largely uninterpretable and don't represent documents. LDA on the other hand is quite interpretable by humans, but doesn't model local word relationships like word2vec. We build a model that builds both word and document topics, makes them interpretable, makes topics over clients, times, and documents, and makes them supervised topics." [17]

Defining the model is simple and quick (code block as showing how it works):

```
model = LDA2Vec(n_words, max_length, n_hidden, counts)
model.add_component(n_docs, n_topics, name='document id')
model.fit(clean, components=[doc_ids])
```

While visualizing the feature is similarly straightforward:

```
topics = model.prepare_topics('document_id', vocab)
prepared = pyLDAvis.prepare(topics)
pyLDAvis.display(prepared)
```

1.3.1.6. Binary Relevance

"Employing independent classifiers in a series of various decisions is the continuation to the single label problem. In the multi-label literature, this approach often called binary relevance for case of binary labels. Binary Relevance (BR) is a well-known and the most popular transformation method that learns q binary classifiers; one for each possible labels in L. As illustrated in Figure 1a, BR converts a multi-label classification problem into several different single-label binary classification problems according to the one vs. all strategy. Each binary classifier is responsible for predicting the association of a single label [18].” [19]

1.3.1.7. Classifier Chains

“J. Read et al. [20] proposed Classifier Chains (CC) that contains q binary classifiers like BR, but includes previous predictions as feature attributes. Classifiers are connected along a chain where each classifier deals with the binary relevance problem associated with label L, see Figure 1 (b). The attribute space of each link in the chain is extended with the 0/1 label relevance of all previous classifiers; therefore building a classifier chain. This method improves prediction performance and can be applied with any type of base classifier.” [19]

1.3.1.8. Conditional Dependency Network

“Conditional Dependency Network (CDN) is a fully connected bidirectional graphical model, which ensures an intuitive representation for the dependencies between multiple label variables, and a well-integrated structure for efficient model training using binary classifiers and label predictions using Gibbs sampling inference [19]. CDN can effectively exploit the label dependency to improve multi-label classification performance. Moreover, it allows a very simple training procedure, while its representation naturally facilitates a simple Gibbs sampling inference on the test instances. It can also incorporate a wide range of simple classification algorithms, including both probabilistic classifiers and non-probabilistic classifiers. The graphical model of CDN was represented on Figure 1 (d).” [19]

1.3.1.9. Label Combination or Label Powerset

“Label Combination (LC) is an alternative paradigm to BR (Binary Relevance) method, is also known as “Label Power set” [20]. LC uses all label sets as single labels, i.e. each label set becomes a single class label within a single label problem. Therefore, the set of single labels represents all different label subsets in the multilabel training data. As a graphical model for this approach was illustrated by Figure 1 (c).” [19]

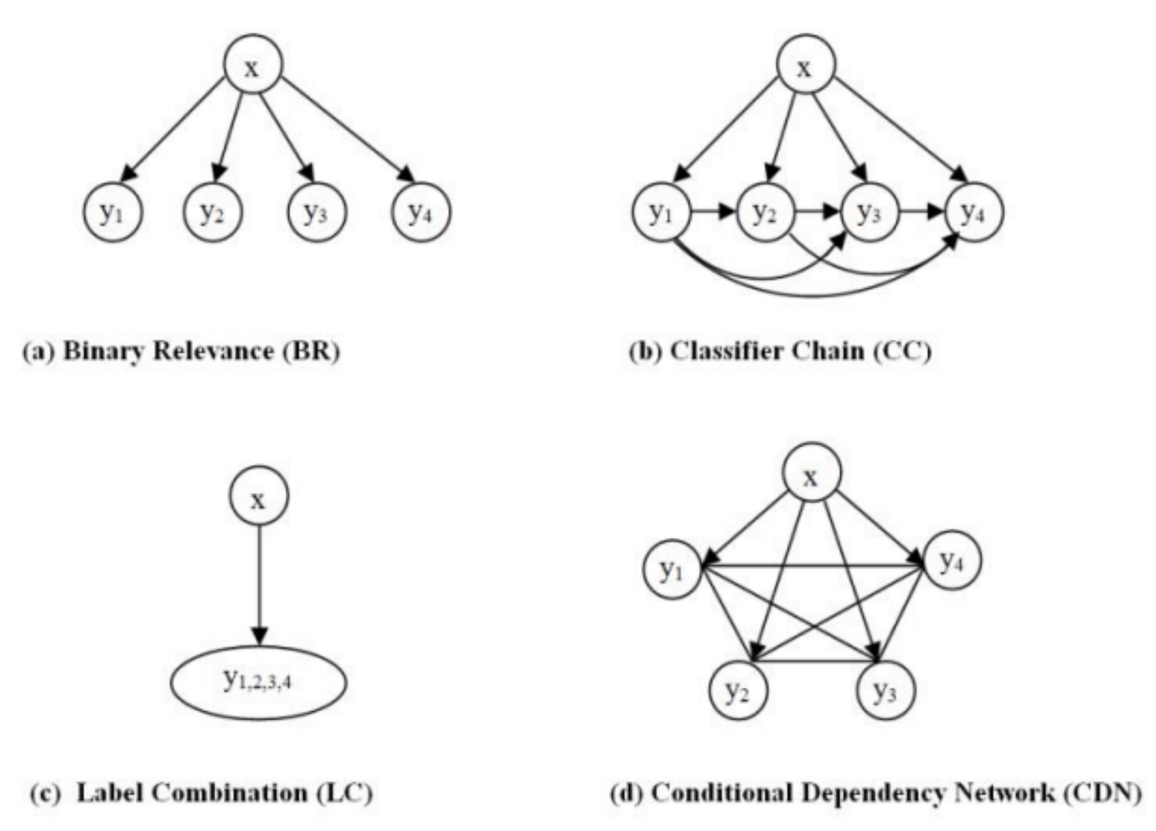

x
y_1
y_2
y_3
y_4
(a) Binary Relevance (BR)
(b) Classifier Chain (CC)
$y_{1,2,3,4}$
(c) Label Combination (LC)
(d) Conditional Dependency Network (CDN)

Figure 1: Several multi-label methods depicted as directed/undirected graphical models [21,22]

Class	*Name*	*Computational complexity*	*Advantage(s)*	*Disadvantage(s)*
BR	Binary Relevance	O(m)	- Computationally efficient, simple and fast.	- It does not consider the relationship among the class variables -Class imbalance
CC	Classifier Chains	O(m)	- Achieves higher predictive performance, - It can work with any type of base classifier.	- It cannot utilize available unlabeled data for classification
CDN	Conditional Dependency Network	O(m)	- It is effective to exploit the dependencies among multiple labels.	- The inference is more expensive and may not scale to large labels. - The convergence rate for the network is very slow.
LC	Label Combination or Label Powerset	O(2^m)	- It considers the relationship among the class labels	- It creates exponentially many classes, so computationally very expensive and complex. - It leads to overfitting of the training data.

Table 1: Comparison of several multi-label classification methods [19]

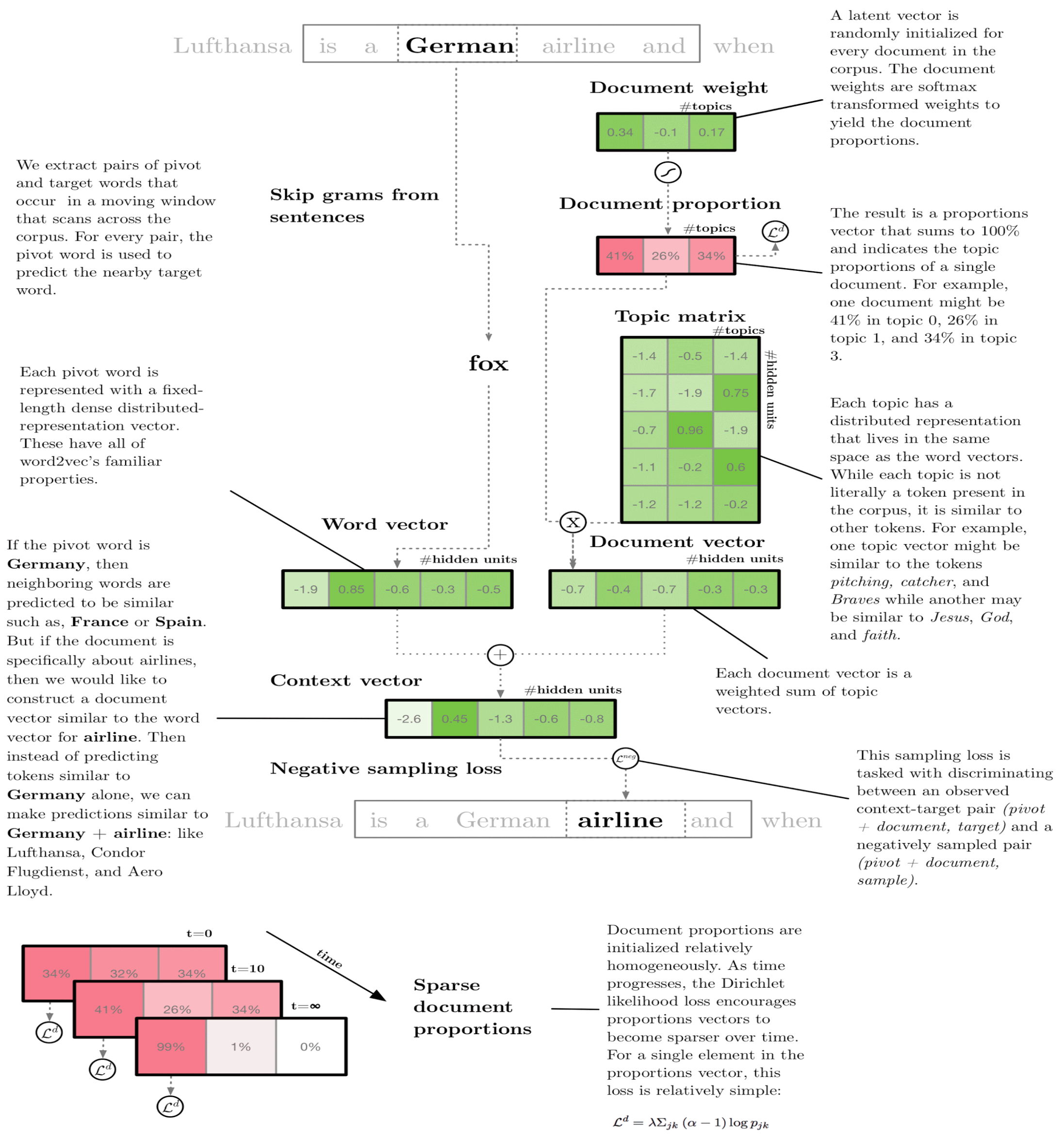

Lufthansa is a German airline and when
A latent vector is randomly initialized for every document in the corpus. The document weights are softmax transformed weights to yield the document proportions.
Document weight
#topics
0.34 -0.1 0.17
We extract pairs of pivot and target words that occur in a moving window that scans across the corpus. For every pair, the pivot word is used to predict the nearby target word.
Skip grams from sentences
Document proportion
#topics
41% 26% 34%
The result is a proportions vector that sums to 100% and indicates the topic proportions of a single document. For example, one document might be 41% in topic 0, 26% in topic 1, and 34% in topic 3.
Topic matrix
#topics
#hidden units
-1.4 -0.5 -1.4
-1.7 -1.9 0.75
-0.7 0.96 -1.9
-1.1 -0.2 0.6
-1.2 -1.2 -0.2
fox
Each pivot word is represented with a fixed-length dense distributed-representation vector. These have all of word2vec's familiar properties.
Each topic has a distributed representation that lives in the same space as the word vectors. While each topic is not literally a token present in the corpus, it is similar to other tokens. For example, one topic vector might be similar to the tokens *pitching*, *catcher*, and *Braves* while another may be similar to *Jesus*, *God*, and *faith*.
Word vector
#hidden units
-1.9 0.85 -0.6 -0.3 -0.5
Document vector
#hidden units
-0.7 -0.4 -0.7 -0.3 -0.3
If the pivot word is **Germany**, then neighboring words are predicted to be similar such as, **France** or **Spain**. But if the document is specifically about airlines, then we would like to construct a document vector similar to the word vector for **airline**. Then instead of predicting tokens similar to **Germany** alone, we can make predictions similar to **Germany** + **airline**: like Lufthansa, Condor Flugdienst, and Aero Lloyd.
Context vector
#hidden units
-2.6 0.45 -1.3 -0.6 -0.8
Each document vector is a weighted sum of topic vectors.
Negative sampling loss
This sampling loss is tasked with discriminating between an observed context-target pair *(pivot + document, target)* and a negatively sampled pair *(pivot + document, sample)*.
Lufthansa is a German airline and when
t=0
34% 32% 34%
t=10
41% 26% 34%
t=∞
99% 1% 0%
time
Sparse document proportions
Document proportions are initialized relatively homogeneously. As time progresses, the Dirichlet likelihood loss encourages proportions vectors to become sparser over time. For a single element in the proportions vector, this loss is relatively simple:
$\mathcal{L}^d = \lambda \Sigma_{jk} (\alpha - 1) \log p_{jk}$

Figure 2: Lda2vec Explanation [19]

1.3.2. Natural Language Processing (NLP)

The main idea of this engineering discipline is interested with the design and realization of computer systems, which is to analyze, understand, interpret and produce a natural language. For now this is subcategory of artificial intelligence especially deep learning have affected the machine learning and linguistics.

1.3.3. Support Vector Machine (SVM)

Support vector machine (SVM), is a very simple and effective methods used in classification. The aim is to obtain the optimal separation hyperplane that will separate the classes from each other. In other words, it is to maximize the distance between support vectors of different classes. Laura A. explained in the article: "SVM is a machine learning method built on strong statistical theories. SVMs are a new technique suitable for binary classification tasks, which is related to and contains elements of non-parametric applied statistics, neural networks and machine learning." [23] However, this method, which has been used since the 90s is not going to be among the methods we will use because it is old.

1.3.4. Deep Neural Networks

1.3.4.1. Convolutional Neural Network (CNN)

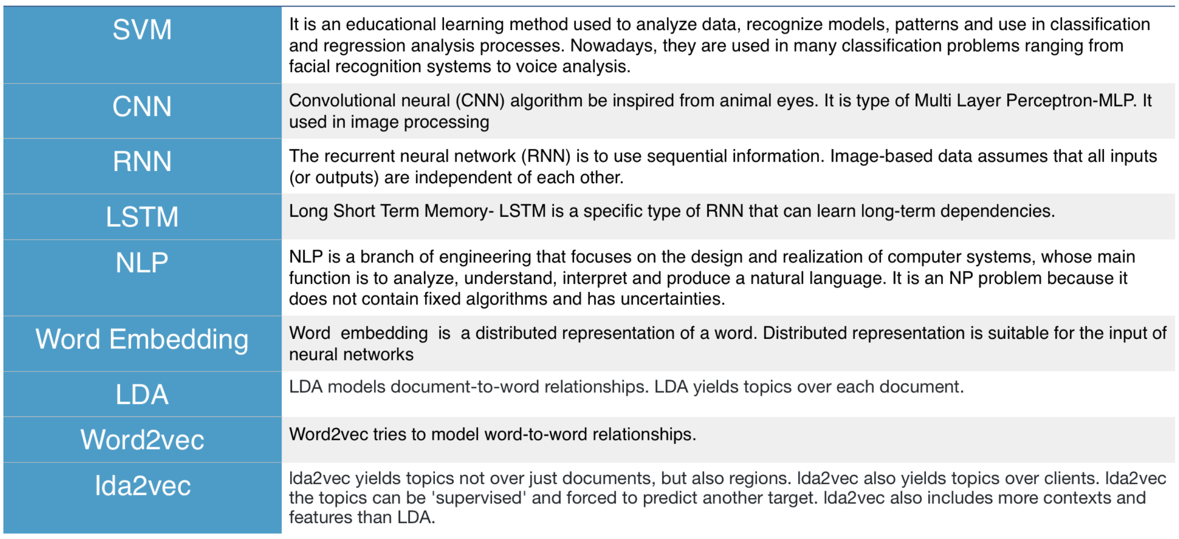
Convolutional neural (CNN) algorithm be inspired from animal eyes. It is type of Multi Layer Perceptron-MLP. CNN algorithms are applied in many different fields such as natural language processing (NLP), biomedical, especially in the field of image and sound processing.

1.3.4.2. Recurrent Neural Network (RNN)

"The main idea of RNN is use repetitive information. In RNN we think that all inputs or outputs are independent of each other. RNNs are called recurrent because the output depends on previous calculations.And this shows we can think that they have memory."[24]

1.3.4.3. Long Short Term Model (LSTM)

Long Short Term Memory-LSTM is a new type of RNN that can learn long-term dependencies. And it called LSTM for prevent to long term dependencies problems. LSTM ability to memories what have been learned up to now.



SVM	It is an educational learning method used to analyze data, recognize models, patterns and use in classification and regression analysis processes. Nowadays, they are used in many classification problems ranging from facial recognition systems to voice analysis.
CNN	Convolutional neural (CNN) algorithm be inspired from animal eyes. It is type of Multi Layer Perceptron-MLP. It used in image processing
RNN	The recurrent neural network (RNN) is to use sequential information. Image-based data assumes that all inputs (or outputs) are independent of each other.
LSTM	Long Short Term Memory- LSTM is a specific type of RNN that can learn long-term dependencies.
NLP	NLP is a branch of engineering that focuses on the design and realization of computer systems, whose main function is to analyze, understand, interpret and produce a natural language. It is an NP problem because it does not contain fixed algorithms and has uncertainties.
Word Embedding	Word embedding is a distributed representation of a word. Distributed representation is suitable for the input of neural networks
LDA	LDA models document-to-word relationships. LDA yields topics over each document.
Word2vec	Word2vec tries to model word-to-word relationships.
lda2vec	lda2vec yields topics not over just documents, but also regions. lda2vec also yields topics over clients. lda2vec the topics can be 'supervised' and forced to predict another target. lda2vec also includes more contexts and features than LDA.

Figure 3: Comparison of models

1.4. TOOLS

1.4.1. Scikit-Learn

1.4.1.1. Classifiers Training

1.4.1.1.1. Pipeline

“Scikit-learn provides a pipeline utility to help automate machine learning workflows. Pipelines are very common in Machine Learning systems, since there is a lot of data to manipulate and many data transformations to apply. So user can utilize pipeline to train every classifier.” [25]

1.4.1.1.2. OneVsRest multi-label strategy

“The Multi-label algorithm accepts a binary mask over multiple labels. The result for each prediction will be an array of 0s and 1s marking which class labels apply to each row input sample.” [25]

1.4.1.1.3. Naive Bayes

“OneVsRest strategy can be used for multi-label learning, where a classifier is used to predict multiple labels for instance. Naive Bayes supports multi-class, but we are in a multi-label scenario, therefore, we wrap Naive Bayes in the OneVsRestClassifier.” [25]

```
# Define a pipeline combining a text feature extractor with multi label classifier
NB_pipeline = Pipeline([
          ('tfidf', TfidfVectorizer(stop_words=stop_words)),
          ('clf', OneVsRestClassifier(MultinomialNB(
             fit_prior=True, class_prior=None))),
        ]) for category in categories:
  print('... Processing {}'.format(category))
  # train the model using X_dtm & y
  NB_pipeline.fit(X_train, train[category])
  # compute the testing accuracy
  prediction = NB_pipeline.predict(X_test)
  print('Test accuracy is {}'.format(accuracy_score(test[category], prediction)))
```

Results Based on referenced database [26]:

```
... Processing toxic
Test accuracy is 0.9191401279933155
... Processing severe_toxic
Test accuracy is 0.9900112041626312
... Processing obscene
Test accuracy is 0.9514802787747584
... Processing threat
Test accuracy is 0.9971135038644866
... Processing insult
Test accuracy is 0.9517271501547694
... Processing identity_hate
Test accuracy is 0.9910556600011394
```

1.4.1.1.4. LinearSVC

```
SVC_pipeline = Pipeline([
          ('tfidf', TfidfVectorizer(stop_words=stop_words)),
          ('clf', OneVsRestClassifier(LinearSVC(), n_jobs=1)),
        ]) for category in categories:
  print('... Processing {}'.format(category))
  # train the model using X_dtm & y
  SVC_pipeline.fit(X_train, train[category])
  # compute the testing accuracy
  prediction = SVC_pipeline.predict(X_test)
  print('Test accuracy is {}'.format(accuracy_score(test[category], prediction)))
```

Results Based on referenced database [26]:

```
... Processing toxic
Test accuracy is 0.9599498661197516
... Processing severe_toxic
Test accuracy is 0.9906948479842003
... Processing obscene
Test accuracy is 0.9789019920621356
... Processing threat
Test accuracy is 0.9974173455629617
```

1.4.1.1.5. Logistic Regression

```
LogReg_pipeline = Pipeline([
            ('tfidf', TfidfVectorizer(stop_words=stop_words)),
           ('clf', OneVsRestClassifier(LogisticRegression(solver='sag'), n_jobs=1)),
        ]) for category in categories:
  print('... Processing {}'.format(category))
  # train the model using X_dtm & y
  LogReg_pipeline.fit(X_train, train[category])
  # compute the testing accuracy
  prediction = LogReg_pipeline.predict(X_test)
  print('Test accuracy is {}'.format(accuracy_score(test[category], prediction)))
```

Results Based on referenced database [26]:

```
... Processing toxic
Test accuracy is 0.9548415275641391
... Processing severe_toxic
Test accuracy is 0.9910556600011394
... Processing obscene
Test accuracy is 0.9761104464573956
... Processing threat
Test accuracy is 0.9973793653506523
... Processing insult
Test accuracy is 0.9687612753755294
```

1.4.2. StarSpace

StarSpace is an ambitious model that aims to solve issues related to the extensive asset embedding. Created by Facebook AI Research (FAIR) and open source. it’s expands on FAIR’s previous text embedding library fastText. StarSpace intends to be a straight-forward and efficient strong baseline, that is, the first model you’d train for a new dataset or a new problem.Because FastText does not fully support the multi-tagged classification, StarSpace can be a good alternative. In addition, StarSpace's documentation is not very comprehensive. This is the reason why this article is written first. [27]

1.5. Related Studies/Works About Multi-label Classification

1.5.1. News Text

In this study, they found a solution to the language problem which is one of the problems of text classification. Generally developed solutions have been in English. There are not many studies on Arabic texts. Therefore, they have worked on Arabic text classification. This study talks about three techniques. According to Mohammed A., there is three popular techniques deploy to classify data. “These three well-known techniques are applied on Arabic data set. A comparative study is made between these three techniques. Also this study used fixed number of documents for all categories of documents in training and testing phase. The result shows that the Support Vector machine gives the best results.” [28] Thanks to this work, we can see that Support Vector Machine (SVM) is an old but still used method.

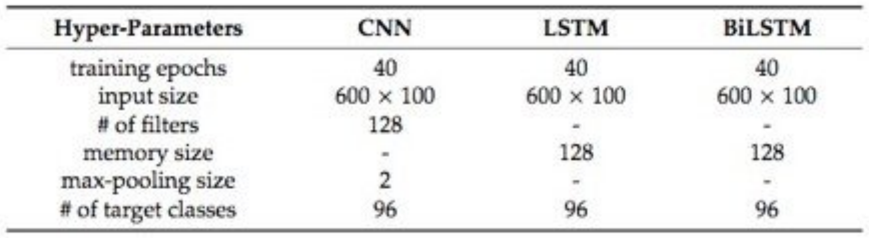
In another study about news classification of Van Meeuwen et al ,ASD Media searches for a solution to classify news and articles from two different datasets. They explain their goal as follows: “The goal is to find out if it’s possible to use a machine learning (ML) approach to TC to construct a classification system that can be used in a semi-automatic setting. Two main challenges of the cases are that news articles are potentially labeled with multiple categories (multi-label) and the dataset is very imbalanced.” [29]

According to ASM media, they used two different data sets whose names were 'ASD' and 'GEW'. Also The ‘ASD’ dataset holds news articles which is more multi- label and also around 4.3 times larger than the ‘GEW’ dataset. “Each news article can be labeled with multiple

categories, thereby making it a multi-label text classification (MTC) problem. This makes it a more challenging problem than single-label classification (SC) problems.”[29] The aim of this exploration is to figure out if it’s possible to design a classification system that can classify these news articles with rational performance. “For analytical purposes we restricted ourselves to classifiers which are easy to interpret by humans and therefore decided to use Decision Trees (DTs). We experimented with various settings and techniques to find out which setup for a classification system is the best suited for each dataset.” [29]
After the experiments using two sets of data, they selected the best performance. But then they continued to take a look at the two techniques that caused them trouble: the classifier chains (CC) and the hierarchical top-down classification (HTC). “We found out that including the source of a news article as a feature for our DT learning algorithm was not a trivial task. There are many different sources which occur with a very low frequency in the dataset, each source having news articles labeled with different categories. For one dataset, the ‘GEW’ dataset, including the source as a nominal (categorical) feature, improved the performance. However for the other dataset, the ‘ASD’ dataset, the performance decreased. The ‘GEW’ dataset has one single source which covered around 40% of the data.” [29]

In study of Hu et al., their aim is: ”We propose a hierarchical feature extraction model (HFEM) for multi-label mechanical patent classification, which is able to capture both local features of phrases as well as global and temporal semantics.” [30]

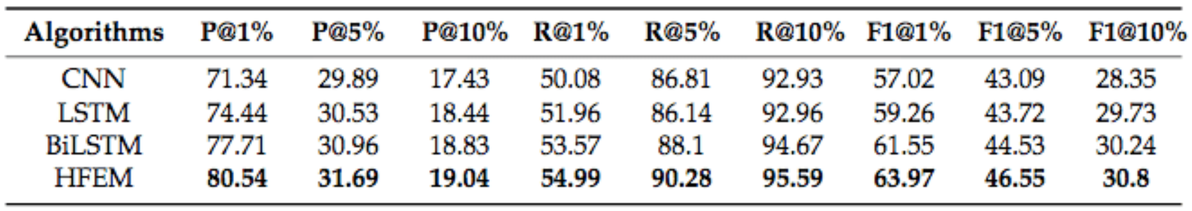
According to this study ,hyper-parameters for baseline methods are listed in Table 4. “For each baseline method, the training epoch was fixed to 40, and the number of input words was set to 150 when only taking one section from the entire patent document. The number was set to 600 when the entire text was used by the model, and finally, a fully-connected layer with sigmoid activation function was connected to 96 categories from the IPC label matrix.” [30]



Hyper-Parameters	CNN	LSTM	BiLSTM
training epochs	40	40	40
input size	600 × 100	600 × 100	600 × 100
# of filters	128	-	-
memory size	-	128	128
max-pooling size	2	-	-
# of target classes	96	96	96

Table 2: Hyper-parameters for baseline methods

"In addition, report the performance of these four models for nine evaluation indicators in Table 5. HFEM obtained the best performance in predicting one label for each patent document as well as in predicting five and ten labels. The experimental results demonstrate and verify the feasibility and effectiveness of our HFEM model for mechanical patent classification." [30]



Algorithms	P@1%	P@5%	P@10%	R@1%	R@5%	R@10%	F1@1%	F1@5%	F1@10%
CNN	71.34	29.89	17.43	50.08	86.81	92.93	57.02	43.09	28.35
LSTM	74.44	30.53	18.44	51.96	86.14	92.96	59.26	43.72	29.73
BiLSTM	77.71	30.96	18.83	53.57	88.1	94.67	61.55	44.53	30.24
HFEM	**80.54**	**31.69**	**19.04**	**54.99**	**90.28**	**95.59**	**63.97**	**46.55**	**30.8**

Table 3: Results of various models using the narrative text as input

1.6. Summary

In this project, we talked about how to classify text documentation. And researches shows that there are several ways to classify text documentation. These methods based on machine learning, semantic processing, word2vec. But recently deep learning methods getting more affected the works. In developing world, methods getting old too fast. All developer's decreasing problems with a new method. As a result of this researches we will use lda2vec and word embedding algorithms. These are the best algorithms and minimize the risks for multi-label classification.

2. Software Requirements Specification

2.1. Introduction

2.1.1. Purpose

This study focuses on news text multilabel classification. News have different subjects and some news can have more than one subject that needs to be classified. So that every news text needs multilabel classification. For our study, we choose using Python language which is mainly used for deep learning. We decided to use Word Embedding with lda2vec which can be done with Python and build a web-interface with PHP which helps user to upload a news text and gets the results.

2.1.2. Scope of Project

Nowadays, there are a lot of informations on Internet and every information has their own classification which can be related with medical, marketing, news and many more. Every information has and needs some kind of classification and it needs to be classified for quick access and preventing the data loss. The classification has been widely studied and it has more than one way to classify an information with computer which can be generalizable by one main topic which is Machine-learning.

2.1.3. Glossary

Term	**Definition**
Python	Programming Language
lda2vec	A method tries to combine best parts of word2vec and LDA into a single framework.
Word2vec	Word2Vec is a method that helps you with calculating the distance between words in vector.
Admin	Person who can access to system.
User	Person who can use the system.

2.1.4. Overview of Document

In this documentation, we mentioned and explained why we need Multilabel Classification for news texts. We pointed out what is the problem and what are the solutions for this problem. The requirement part of this documentation provides what are the functions of our system and shows use cases of each function which will be in web interface.

2.2. Overall Description

2.2.1. Product Perspective

The purpose of Multilabel Classification for news texts is to help decrease of the human work. The system can help everyone who is trying to label their news texts by doing this progress automatically. The system has two parts: Admin Part and Public Part.

Admin part is for using the repo and create a model by using methods (lda2vec, word2vec, etc.) with their respective parameters. Admin can manage repositories by adding, deleting and browsing them. Admin also can manage models by training new models, deleting models and browsing models. Public Part is for usability for everyone who has news texts and wants to label them. Everyone can upload a news text to the system (temporarily) and get the result of it as labels.

2.2.2. User Characteristics

2.2.2.1. Admin

- Admin must have a knowledge about how word embedding and lda2vec methods work.
- Admin must have enough authority to manage user data on the web interface.
- Admin must have enough authority to manage repo on the web interface.
- Admin must have enough authority to manage model on the web interface.

2.2.2.2. Member

- Member must have a news text that can be used by system.
- Member must have knowledge about how to use the system.

2.3. Requirement Specification

2.3.1. External Interface Requirements

2.3.1.1. User Interfaces

The user interface will be on website. Which can be usable from different platforms by any internet browser.

2.3.1.2. Hardware Interfaces

There is no external hardware interface requirement.

2.3.1.3. Software Interfaces

Software presented in this SRS does not need any other software interface than the operating system itself.

2.3.1.4. Communications Interfaces

There is no external communication interface requirement.

2.3.2. Functional Requirements

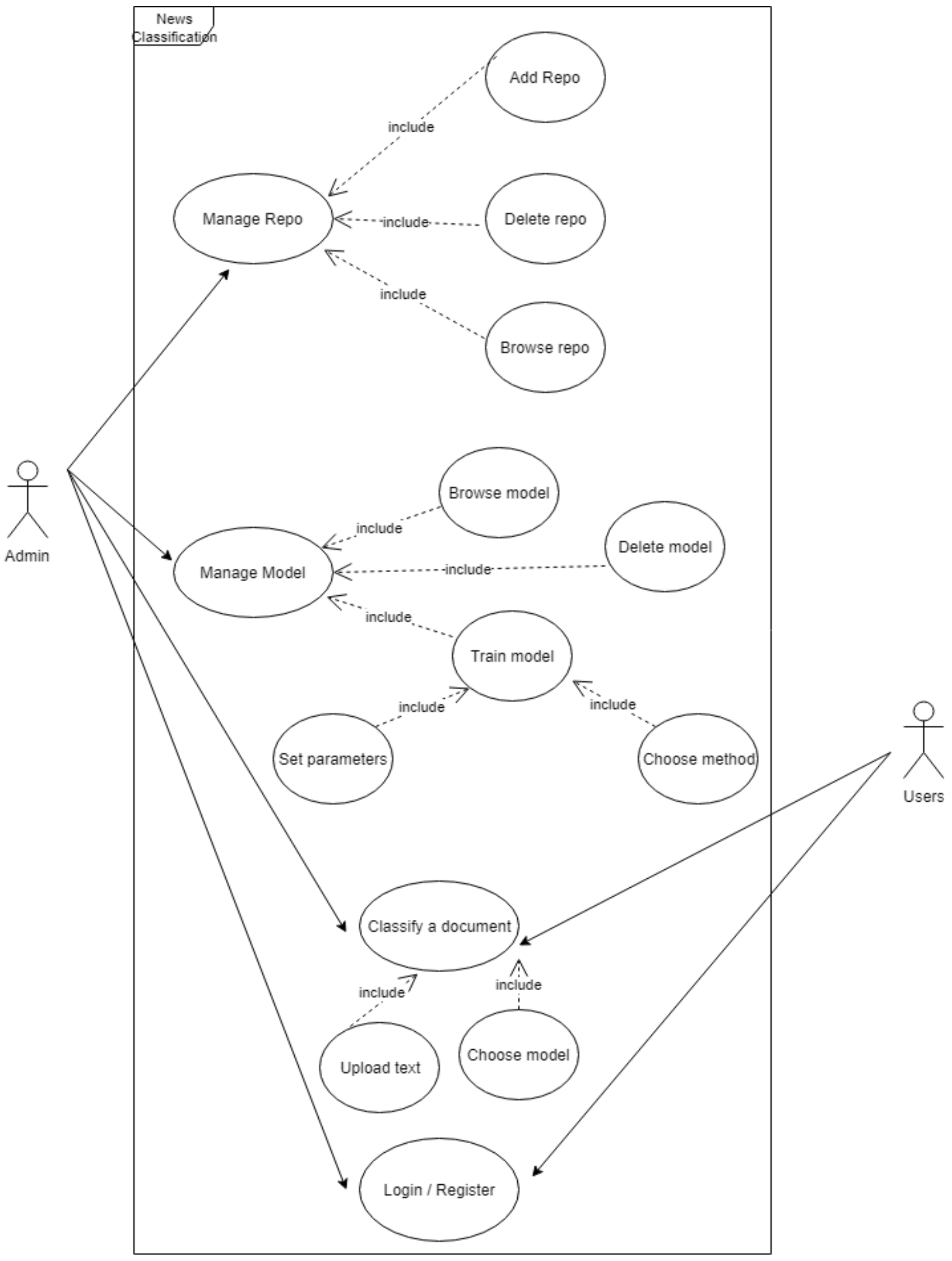


Figure 4: News classification use case

2.3.2.1. Login System Use Case

Use Case:

- Admin or Member can login to the system with correct username and password.
- If username or password is incorrect, user can try to re-login to system.
- If user selects change password button, the system asks his/her email that is already attached to that admin account.
- System should send an email to that mail address for reset/changing the password.
- User can press logout button and logout from the system.

Diagram:

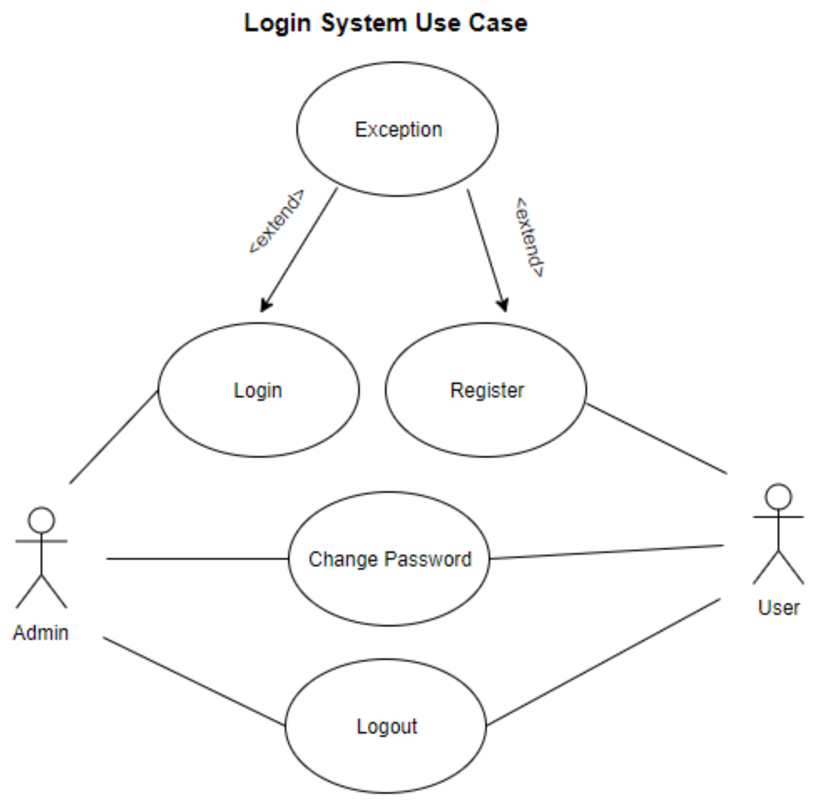


Figure 5: Login System Use Case

Brief Description:

As shown in Figure 5, there are 3 functions that can be usable by both admin and member. User can login to the system, change his/her password and logout from the system.

2.3.2.2. Manage Repository Use Case

Use Case:

- Admin clicks the add repo button to he/she can upload new repository.
- Admin clicks the delete repo button to he/she can remove old repository.
- Admin clicks the browse repo button to system admin can see all repository in the system.
- If adding repository enrolled before system does not add new repo.
- If new repository doesn't exist in the system, the system will add the repository.

Diagram:

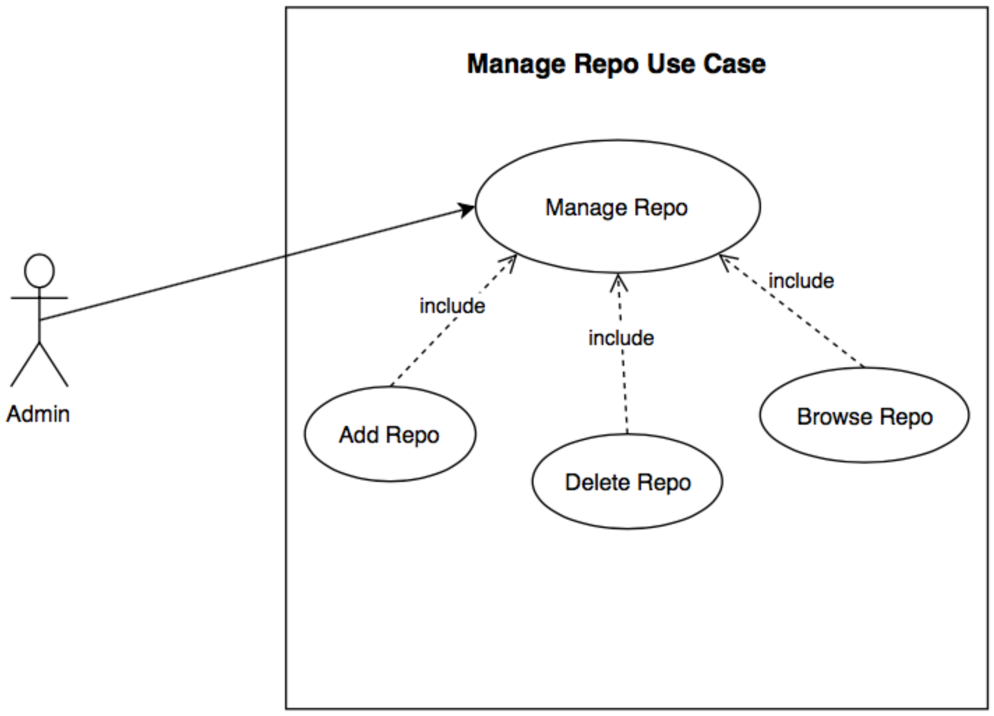


Figure 6: Manage Repository Use Case

Brief Description:
In Manage Repository Use Case diagram(Figure 6) shows the functions which can be used by admin. Admin can be add document repository, delete repository and browse repository.

2.3.2.3. Manage Model Use Case

Use Cases:

- Admin clicks on the train model button, he/she can choose method for create model for classifying text.
- In train model admin can also set parameters for model.
- Admin clicks the browse model button to system, he/she can see compare old document. If the document is existing then give the result of label.
- Admin clicks on the delete model button to he/she can delete model.

Diagram:

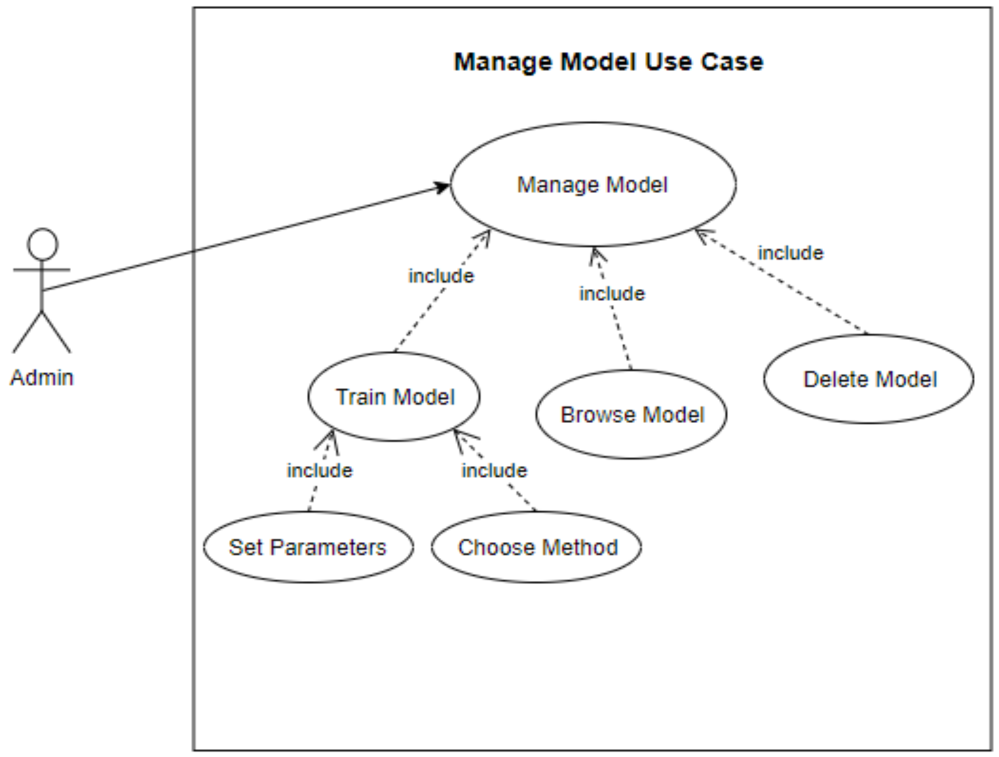


Figure 7: Manage Model Use Case

Brief Description:

In Manage Model Use Case diagram(Figure 7) shows the functions which can be used by admin. Admin can train new documents, compare old documents or delete models.

2.3.2.4. Classify a Document Use Case

Use Cases:

- User clicks the Upload Text button and can upload a document (.txt) from their computer.
- User can select a model.

Diagram:

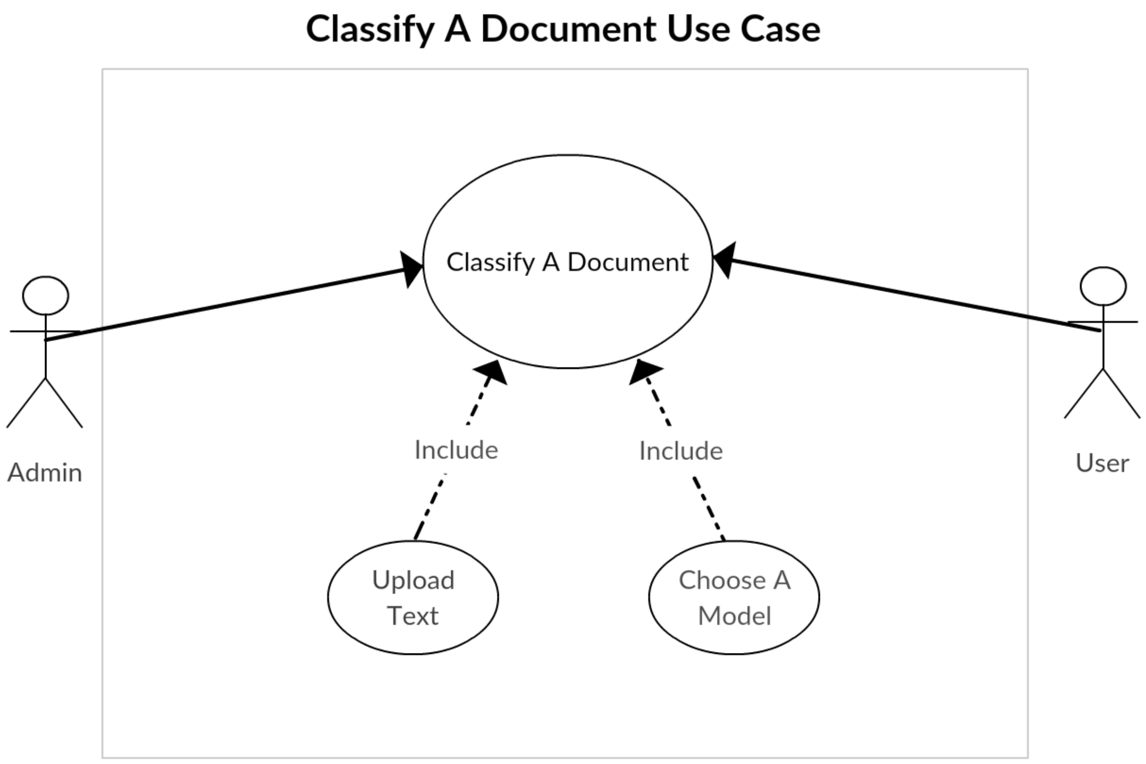


Figure 8: Classify A Document Use Case

Brief Description:
In Classify a Document Use Case diagram (Figure 8) shows that the function can upload and choose a model. With this method, they will be able to upload documents and classify the texts they upload. When the user clicks on this button, two option can be chosen.

2.3.3. Performance Requirement

Application must run without any latency more than progress for reading the document and classify it with different labels.

2.3.4. Software System Attributes

2.3.4.1. Portability

- Multilabel classification should work on every browser that can run Javascript and PHP.

2.3.4.2. Performance

- This will be changed due to process time of labeling the news text document.

2.3.4.3. Usability

- Multilabel classification accept document formats of .txt and .json .

2.3.4.4. Adaptability

- Every news text should be labeled by the respective labels that is already trained.

2.3.4.5. Scalability

- User get information when results are came out.

2.3.5. Safety Requirement

Multilabel classification does not have any interaction with any body movement or action. So, it does not require any safety requirement.

3. Software Design Description

3.1. Introduction

3.1.1. Purpose

The purpose of this Software Design Document (SDD) is providing the details of the project named as “Multi label Classification of News Text”.

The target audience is the people who are working with news text documents. This web based application will help them with multi labeling the each text document that is given/uploaded to the system. Each label will have a percentage and this will show that how much is the given labels related with the uploaded document.

The purpose of Multi label Classification of News Text project is to design and implement a web interface that can multi label the news text documents with a given model. The project will include two kinds of user type which are member and admin. An admin can upload a repo and create models with methods that can be used for the classification. A member can classify a text document/s with giving a model to multi label the document/s.

For a better explanation of the project, this SDD includes diagrams such as database, activity, class and block diagram.

3.1.2. Scope

This document includes brief and in-depth description about the design of project which is named as Multi label Classification of News Text using Python.

Python language used for writing a script that runs on a web server when news text document/s classification happens. The website will be implemented for users to use which will be designed with PHP along with HTML and CSS. PHP is a scripting language that can be used for server-side scripting like in this project.

MySQL will be used for managing the connection of database. MySQL is an open source database management system. As the Python script needs a web server, MySQL requires a web server to keep the database. And both accessible from remote control.

3.1.3. Glossary

Term	**Definition**
Python	Programming Language
lda2vec	A method tries to combine best parts of word2vec and LDA into a single framework.
User	All the members and admins
Admin	People who can manage the system.
Member	People who can use the system.

3.1.4. Overview of document

The remaining chapters and their contents are listed below.

Section 1 included Describe the Project. Section 2 is the Architectural Design of the project. It contains class diagram, database diagram and activity diagram of the system and describes architectural design of project. Section 3 displays and explains the block diagram of the system, which is designed according to use cases in SRS document. Section 4 included User Interface Design of the project. In this section, we have shown user interfaces of the components of the project.

3.1.5. Motivation

We are a group of senior students in computer engineering department who are interested in multi-label classification. In this project we focused on classification of news texts in deep learning. There are many methods and techniques available for labeling a text document but most of the current work usually works on English text. As a group, we aimed to apply deep learning methods for the multi-label classification of news text documents. We will classify Turkish news texts, which is also new with this project. We decided to use Python language because it is useful and suitable for our project.

3.2. Architectural Design

3.2.1. Design Approach (Scrum)

Since every project needs a design approach we have decided to use scrum design approach. Scrum is very basic and easy to adapt for people in the team. Every process is like little tasks that can be done by sprints. Scrum helps with solving big complex problems by parting them to each team members by tasks. We did discuss the definition of the works needs to be done every week by meetings, what we need to do and share what we done in the end of every sprint. Because of these features, we did decide to use scrum by designing approach.

3.2.1.1. Class Diagram

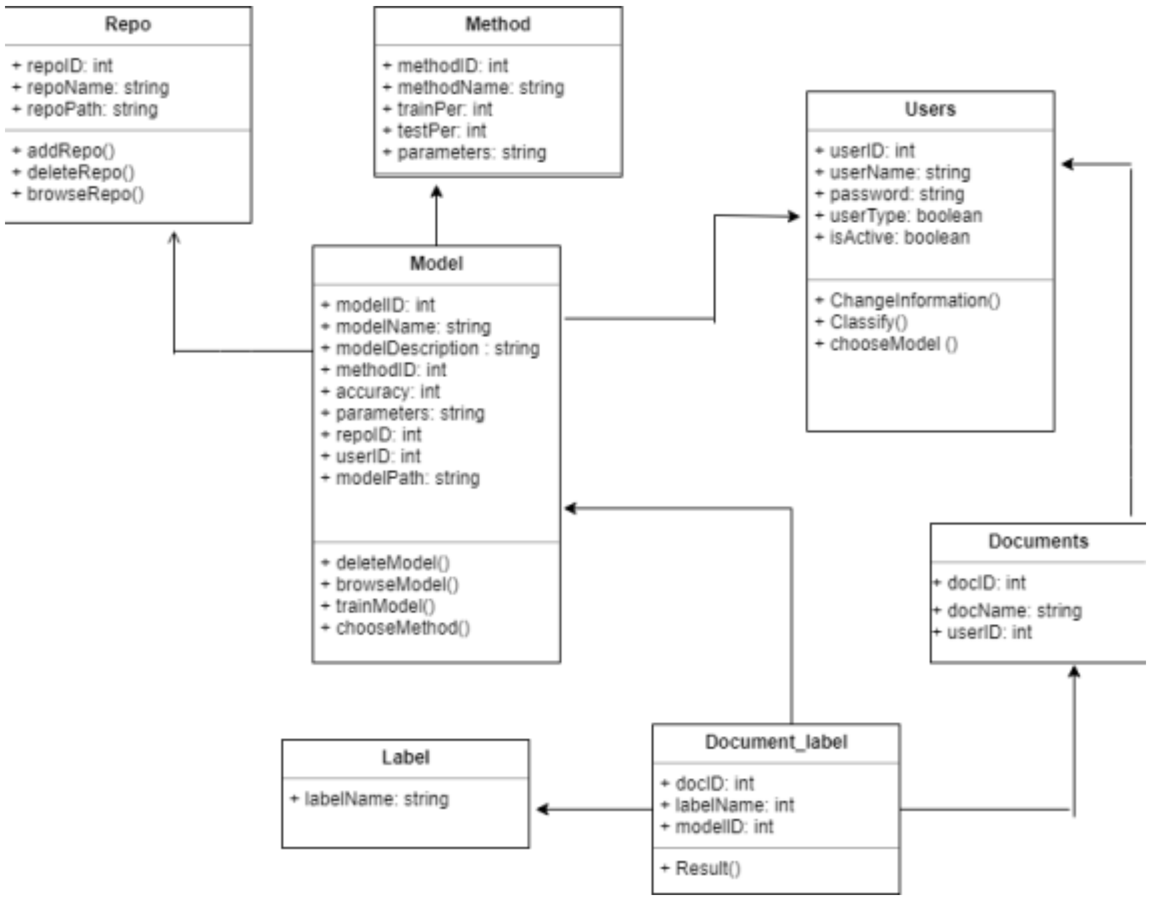
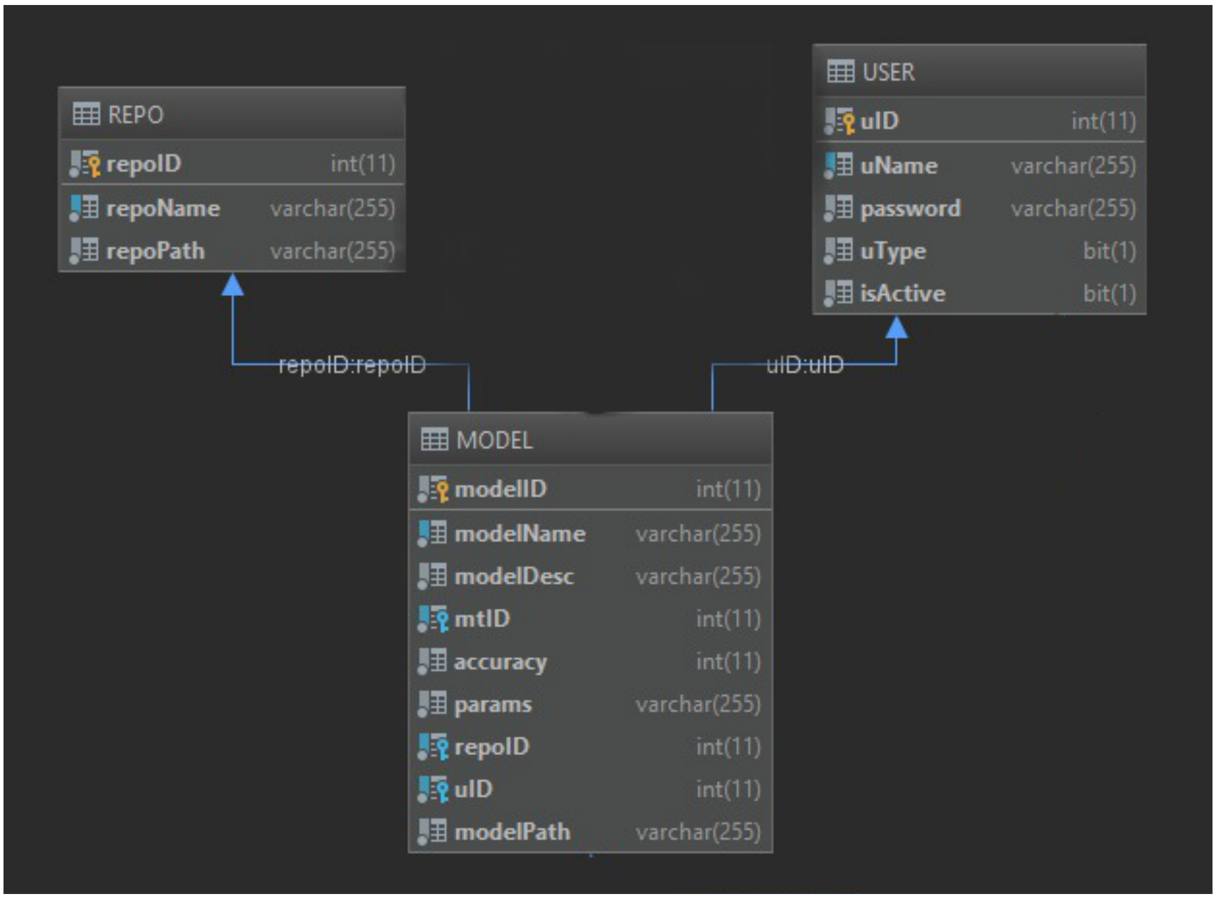


Figure 9: Class Diagram of the Project

Figure 9 displays classes used in this system. The repo class include files. These files can be deleted or added. The method class is for the training a model which can be done by selecting the percentages of the method and the parameters related to the method. Model class includes model deletion, train model, browse model and choose method functions. The document label class contains the result function. It shows the document with which model applied and labels of the document.

3.2.1.2. Database Diagram



REPO	
repoID	int(11)
repoName	varchar(255)
repoPath	varchar(255)

USER	
uID	int(11)
uName	varchar(255)
password	varchar(255)
uType	bit(1)
isActive	bit(1)

MODEL	
modelID	int(11)
modelName	varchar(255)
modelDesc	varchar(255)
mtID	int(11)
accuracy	int(11)
params	varchar(255)
repoID	int(11)
uID	int(11)
modelPath	varchar(255)

Figure 10: Database Diagram of the Project

3.2.1.3. Activity Diagram

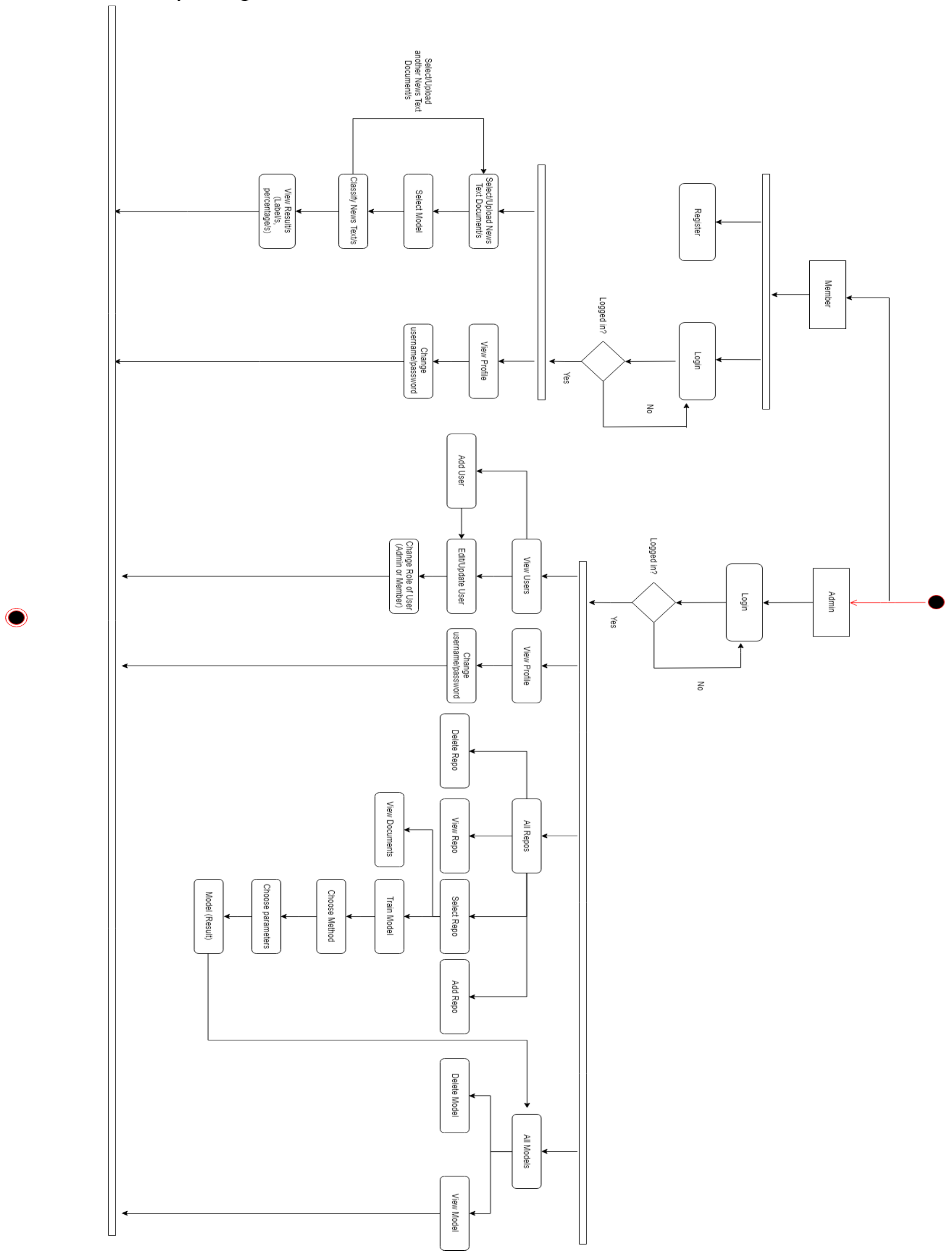


Figure 11: Activity Diagram of the Project

Figure 11 shows that how the scenario generation works as an activity diagram. When the member or admin login to the system, it controls whether the users is a member of the system or not. If the user is a member or admin, system opens member’s or Admin’s interface with all buttons. Member or admin selects a button which he/she wants and then system generates functions related to these buttons. When he/she finished their request to the system, they can exit from the system.

3.2.2. Architecture Design of The Project

3.2.2.1. User Management

Summary: This system can be used by admin or member. Admin or member can login to the system, register, update their password and username and logout from the system.

Actor: Admin, Member

Precondition: User must enter the website.

Basic Sequence:

1. User must register if s/he does not have an account.
2. User must login to the system by entering his/her username and password.
3. User can update his/her username or password by clicking the profile button from the menu.
4. Admin can activate or delete an user account by selecting the checkbox of the user’s row and applying the action.
5. Admin can define a new admin or member to the system by selecting add user from administration menu and selecting the user’s role (Admin or Member).
6. Admin can edit user’s detail such as their role (Admin or Member) by selecting the respective user from the row and clicking on role dropdown.
7. User can logout from the system by selecting logout button from the menu.

Exception: Database connection error can be occurred.

Post Conditions: None

Priority: Low

3.2.2.2. Repo Management

Summary: This system can be used by admin. Admin can add, delete and browse repos.

Actor: Admin

Precondition: Admin must be logged into the system.

Basic Sequence:

1. Admin can add new repo to the system by clicking/selecting add new repo button from the repo page.
2. Admin can delete existing repo from the system by selecting the checkbox of the row and applying the delete repo action from the repo page.
3. Admin can browse existing repo in the system by clicking/selecting repo button the repo page.

Exception: Database connection error can be occurred.

Post Conditions: None

Priority: Medium

3.2.2.3. Model Management

Summary: This system can be used by admin. Admin can delete, train and choose method , browse model.

Actor: Admin

Precondition: Admin must be logged into the system.

Basic Sequence:

1. Admin can delete model from the system by selecting the checkbox of the model row and apply the delete action.
2. Admin can train the model by giving the method with parameters by clicking/selecting train model button from the model page.
3. Admin can browse existing model in the system by clicking/selecting model button from the menu.

Exception: Database connection error can be occurred.

Post Conditions: None

Priority: Medium

3.2.2.4. Classify System

Summary: This system can be used by admin or member. Admin or member can upload a new text file in the system, choose a model and classify the text.

Actor: Admin, Member

Precondition: User must be logged into the system.

Basic Sequence:

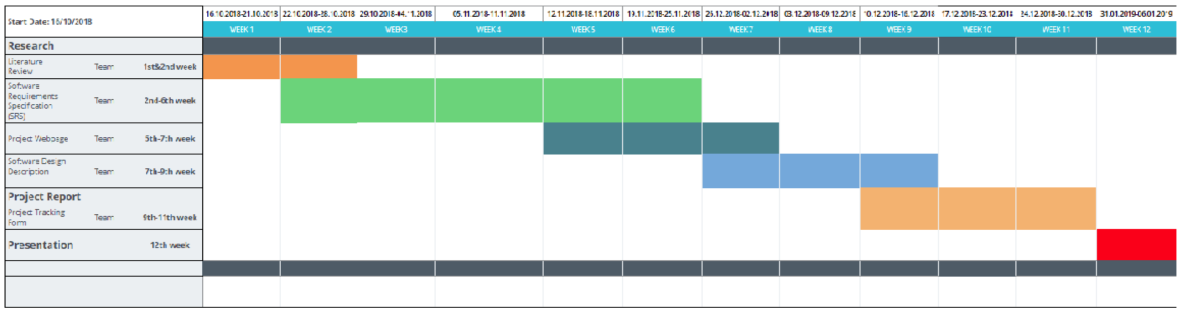
1. Member or Admin can upload a new news text file by dragging/selecting/clicking select file from Classify page.
2. Member or Admin can choose a model by selecting the models from dropdown box.
3. Member or Admin can click/select Classify Document button and see the labels that are related with the document, used model and percentage.
4. Member or Admin can return the Classify Page by selecting/clicking the classify a new document.

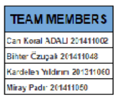
Exception: Database connection error can be occurred.

Post Conditions: None

Priority: High

3.2.3. Work Load Table





TEAM MEMBERS
Can Koral ADALI 201411002
Bihter Özçupak 201411048
Kardelen Yıldırım 201311060
Miray Padır 201411050

Figure 12: Gantt Chart of Work Plan

3.3. Use Case Realization

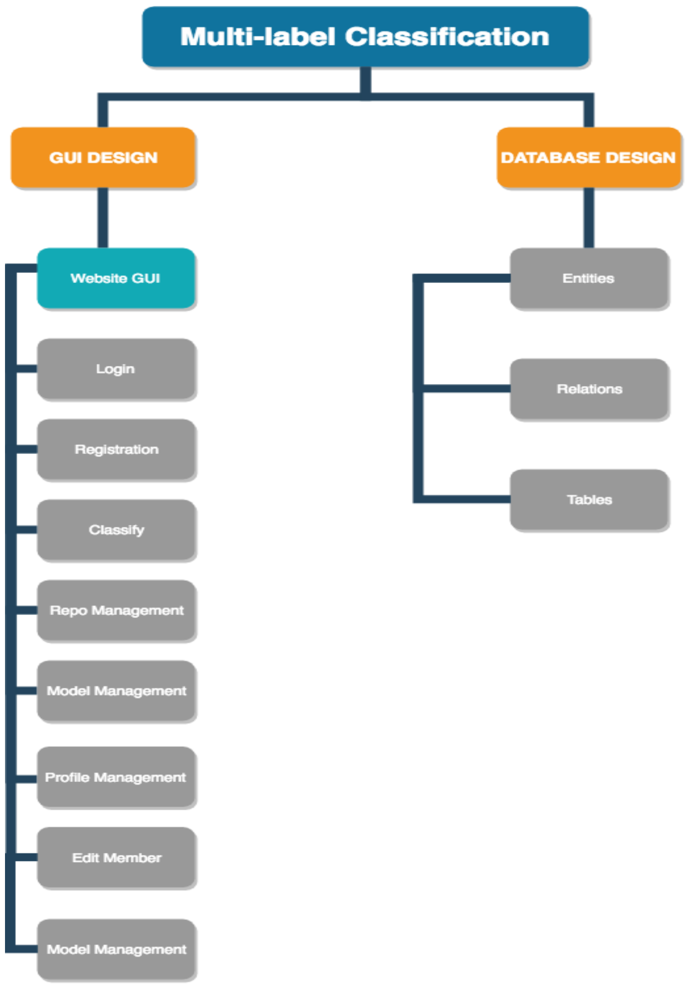


Figure 13: Use Case Relation

3.3.1. Description of the Use Case Relation

In figure 12, all designed system is displayed in block diagram. Diagram includes two main component and their sub-systems.

3.3.1.1. GUI Design

GUI design is responsible for interaction between the admin and the member. There are two sub-system in this design which is Website GUI. This sub-system are divided into other sub-system. This sub-system consist of Login, Registration, Classify, Repo Management, Model Management. GUI design includes ease of use and simplicity.

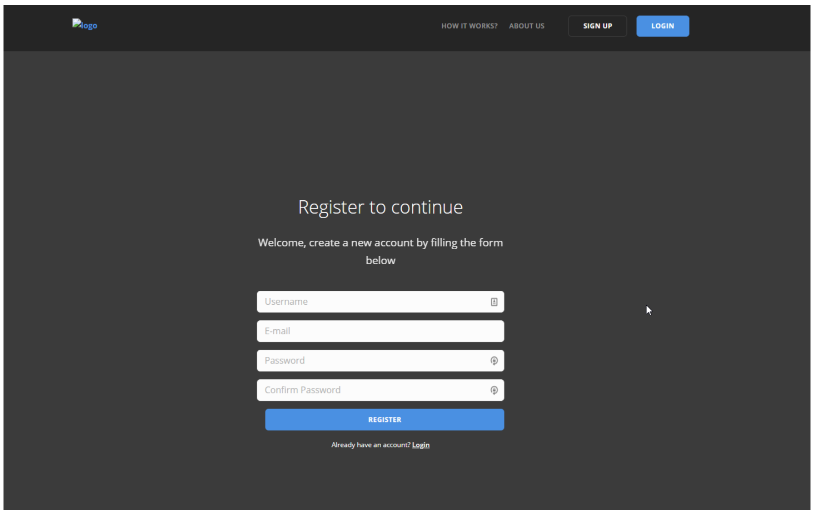
3.3.1.2. Database Design

Database design is responsible for managing data which read and write from the system. There are three type of Database in the system which are Entities, Relations and Tables. In the database, users information, label names, documents, models, methods and repository information will be kept.

3.4. Web Interface design

3.4.1. Overview of Web Interface

3.4.1.1. Login



Register to continue

Welcome, create a new account by filling the form below

Already have an account? Login

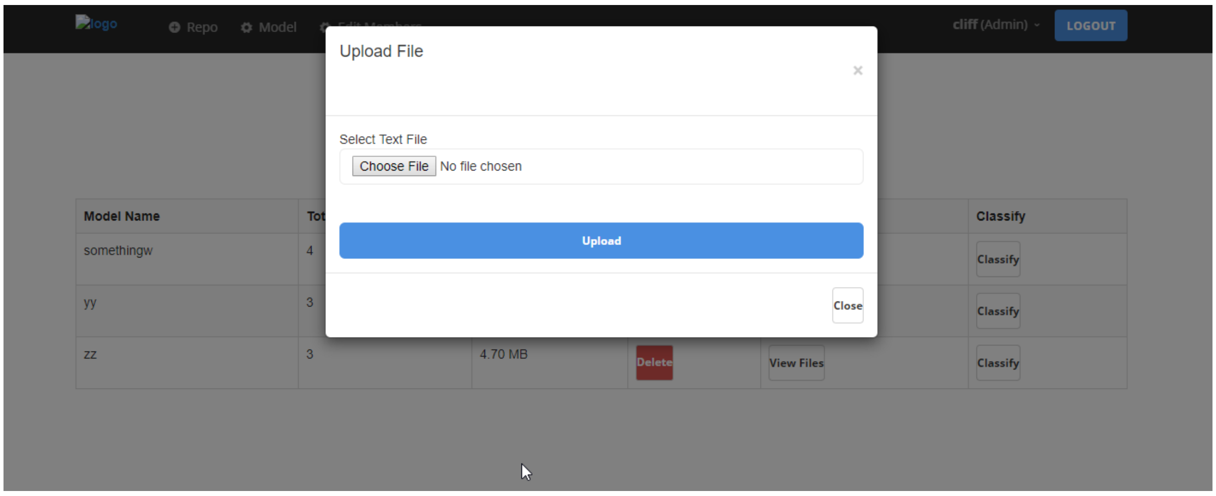
Figure 14: Login page of the Website

In our project, all users have to login to system by entering email and password by clicking login button. There is not specific design for admin to login, all users can login by using same page. If users do not have registration, the system redirect to Sign Up page by clicking 'Sign Up' text.

3.4.1.2. Sign Up

In our project, all users need to register to log in. Then their memberships are approved by admin and they can 'Login Page' log into the system.

3.4.1.3. Classify Page



Model Name	Tot				Classify
somethingw	4				Classify
yy	3				Classify
zz	3	4.70 MB	Delete	View Files	Classify

Figure 18: Admin Classify page

Our project have two type user and interfaces but all type of users (members and admins) can use same classification page as login page.First of all users can click or drag text files to select file area for uploading news texts. After user must select a model from dropdown box for to start the classification process. If no model selection is made or no text document is selected and Classify Document button is clicked, a warning message is displayed. If classification process done successfully, the user will be redirected to Result page.

3.4.1.5. Classify Result Page

In our project, result page of classification shown with classified text, text's labels, their percentage and used model. If user want to classify another document, they can click Classify Another Document button.

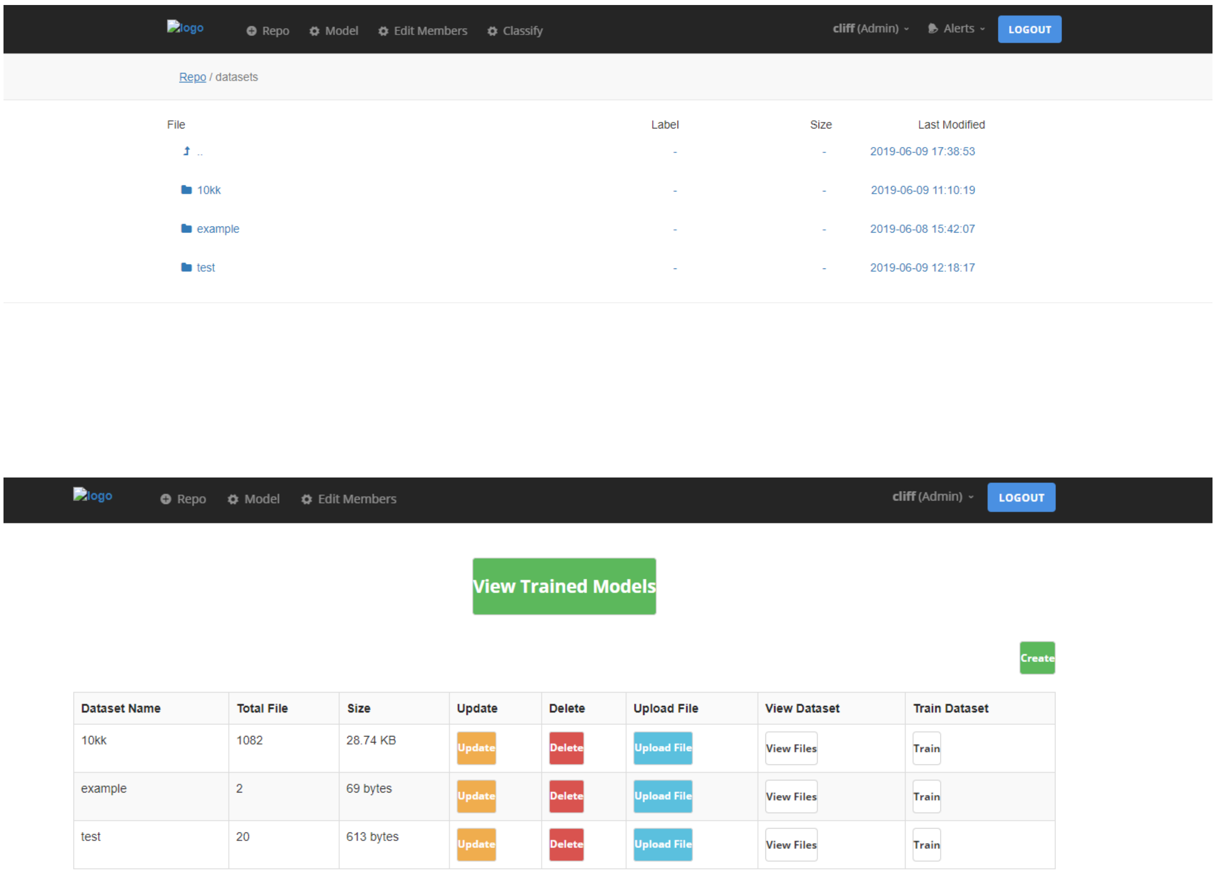
3.4.1.6. My Document Page

In our project, all users can access their previous classified documents. This page list that previous text's names and text's IDs. This page is not editable.

3.4.1.7. My Document Page for Admin

In our project, only admin can edit members. In this page, admin accepts request and activated members for login.

3.4.1.8.Repo Management Page



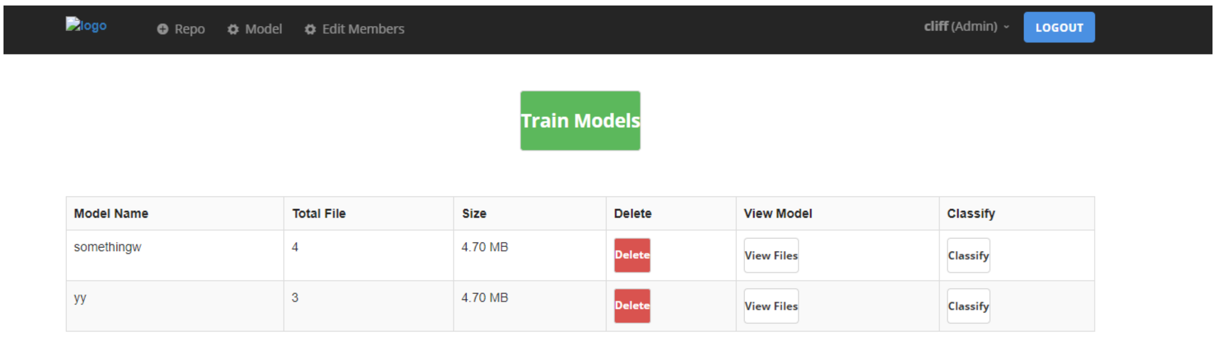
File	Label	Size	Last Modified
..	-	-	2019-06-09 17:38:53
10kk	-	-	2019-06-09 11:10:19
example	-	-	2019-06-08 15:42:07
test	-	-	2019-06-09 12:18:17

Dataset Name	Total File	Size	Update	Delete	Upload File	View Dataset	Train Dataset
10kk	1082	28.74 KB	Update	Delete	Upload File	View Files	Train
example	2	69 bytes	Update	Delete	Upload File	View Files	Train
test	20	613 bytes	Update	Delete	Upload File	View Files	Train

Figure 25: Repo Management Page

In our project, only admin can access repo management page. This page lists existing repos with repo id, repo name and document count. In this page, admin can add new repo. Also admin can select repos and delete them.

3.4.1.9. Model Management Page



Model Name	Total File	Size	Delete	View Model	Classify
somethingw	4	4.70 MB	Delete	View Files	Classify
yy	3	4.70 MB	Delete	View Files	Classify

Figure 26: Model Management Page

In our project, only admin can access some special pages. One of those is Model Management page. In this page, Admin can train models using methods, can delete existing models and can see all of models list with their parameters and method names.

4.TEST PLAN

4.1 INTRODUCTION

4.1.1 Version Control

Version No	**Description of Changes**	**Date**
1.0	First Version	June 09,2019

4.1.2 Overview

The use case of Multi-Label Classification of News Text users namely participant and system administrator which has been determined in SRS document will be tested.

4.1.3 Scope

This document includes the test plan of use cases, test design specifications and test cases correspond to test plan.

4.1.4 Terminology

Acronym	**Definition**
UWP	User Web Page
AWP	Administrator Web Page

4.2 FEATURES TO BE TESTED

This section lists and gives a brief description of all the major features to be tested. For each major feature there will be a Test Design Specification added at the end of this document.

4.2.1 User Web Page

In this project, User Interface constituents are used. The UI part is divided into 8 parts which are Register, Login, List Repo, Model, View Files of Model,Classify and Logout. Every part of the UI also includes smaller parts and includes test of functions of UI components which are used in this project such as panel, button, search, etc. The participant shall sign in to account, if there is no account, they will be registered. Then the participant shall fill in the own information. Then participant who wants to reach anything, clicks on the button .If participant want to see the Repos, click on the list Repo button and the system will show all registered Repos on the screen.

4.2.2 Admin Web Page

This part includes test cases and test plan of Administrator Interface. It includes Login, Edit Repo,Remove Repo,Repo Upload,Repo Create,Train Model,File Name Delete,Repo Train,Train and Delete Model, Edit User Information and Exit. The admin shall edit Repo and edit user information(Remove, change information). Testing of the stated requirements will occur in this document.

4.3 ITEM PASS/FAIL CRITERIA

Describe the general rule to use to decide when a test case passes and when it fails.

4.3.1 Exit Criteria

Describe under what conditions the testing of the product is considered successful. Some examples are:

· 100% of the test cases are executed

- 95% of the test cases passed

· All High and Medium Priority test cases passe

4.4 References

[1] Multi Label Classification of News Text SRS, Available:"https://github.com/CankayaUniversity/ceng-407-408-Multi-class-Classification-of-News-Text-1-/wiki/Software-Requirements-Specification-(SRS)"

[2] Multi Label Classification of News Text SDD, Available:"https://github.com/CankayaUniversity/ceng-407-408-Multi-class-Classification-of-News-Text-1-/wiki/Software-Design-Document"

4.5 TEST DESIGN SPECIFICATIONS

4.5.1 Admin Web Page

4.5.1.1 Subfeatures to be tested

4.5.1.1.1 Register (AWP.RG)

The admin registers to login to the system.

4.5.1.1.2 Login (AWP.LG)

The admin has to login the web application system by entering username and password.

4.5.1.1.3 Repo Dataset (AWP.RD)

When the admin clicks the repo button, the enrolled datasets are displayed.

4.5.1.1.4 Model Button (AWP.MB)

When the admin clicks the model button, the information of the datasets are displayed. Admin his/her can update, delete dataset, upload new file to folder, view dataset and train dataset.

4.5.1.1.5 Repo Train(AWP.RTR)

When admin clicks the train button, the screen opens and enters the parameters (Model name, vector dim, labels, test ratio and epoch). Then admin clicks the train button and shows the result on the screen. Turns off the screen by clicking the close button.

4.5.1.1.6 Repo Upload(AWP.RUP)

When admin clicks the upload button, a screen opens. When the admin clicks the choose file button, the admin selects the file wants to upload and then uploads the file to the system by click the upload button.

4.5.1.1.7 Repo Delete Button (AWP.RDL)

Admin can delete repo. Admin, who added the repo, can remove existing repos.

4.5.1.1.8 Repo Update Button (AWP.RUD)

Admin can update repo. Admin, who added the repo, update existing repos information's.

4.5.1.1.9 Repo Create Button (AWP.RCR)

Admin can create new repo. When admin click that button, he/she can upload new repo files to system.

4.5.1.1.10 View Trained Models Button (AWP.VTM)

If admin click view trained models button, it redirecting to model page.

4.5.1.1.11 Train Models Button (AWP.TM)

If admin click train models button, it redirecting to repo page.

4.5.1.1.12 Model Delete Button (AWP.DM)

Admin can delete model. When user click to model delete button, he/she can delete existing model.

4.5.1.1.13 View Files of Model (AWP.MVF)

The participant can view files that using. Admin who wants to see the file names, click this button and see the model list using by the system.

4.5.1.1.14 File Name Delete (AWF.MDL)

The participant can remove the models. Admin who want to remove these models, can delete from the system.

4.5.1.1.15 Classify (AWP.CS_1)

The participant can classify their documents. Admin who want to classify documents, click and see model information first.

4.5.1.1.16 Classify (AWP.CS_2)

The participant can classify their documents. Admin who want to classify document, click and then document upload file page opens.

4.5.1.1.17 Choose File (AWP.CHF)

The participant can add text documents. Admin who want to add document ,selects the document set from the computer and loads files into the system.

4.5.1.1.18 Classify (AWP.CS_3)

The participant can classify their documents. Admin who want to classify document, clicks the classify button, then source code works on background and shows the classification result on new opened page. The system saved this result on database.

4.5.1.1.19 Close(AWP.CL)

The participant can close the page. Admin who want to close page, can exit from the current page and return to the previous page.

4.5.1.1.20 Logout (AWP.LGO)

The participant can leave this system and account with logout button.

4.5.1.2 Test Cases

Here list all the related test cases for this feature

TC ID	Requirements	Priority	Scenario Description
AWP.RG.01	3.2.1	H	Enter a valid username,email and password.
AWP.RG.02	3.2.1	H	Enter a invalid username and email.

TC ID	Requirements	Priority	Scenario Description
AWP.LG.01	3.2.1	H	Enter a valid username and password.
AWP.LG.02	3.2.1	H	Enter a invalid username and password.

TC ID	Requirements	Priority	Scenario Description
AWP.RD.01	3.2.2	M	Displays the datasets store in the system and the information of the dataset.

TC ID	Requirements	Priority	Scenario Description
AWP.MB.01	3.2.3	H	The dataset stored in the system and the page where some operations are performed are opened(e.g. update, delete, upload file, view files, train).

TC ID	Requirements	Priority	Scenario Description
AWP.RTR.01	3.2.3	H	After entering all parameters, the system trains.
AWP.RTR.02	3.2.3	H	If there are missing parameters, the system does not train.

TC ID	Requirements	Priority	Scenario Description
AWP.RUP.01	3.2.2	H	Saves the file to the system after uploading it as zip format.
AWP.RUP.02	3.2.2	H	If the file is not uploaded in zip format, it will not be saved to the system.

TC ID	Requirements	Priority	Scenario Description
AWP.RDL.01	3.2.2	H	Select existing repo after selecting click "Delete" button and existing repo will remove.
AWP.RDL.02	3.2.2	H	Select existing repo after selecting click "Delete" button and existing repo will not be remove.

TC ID	Requirements	Priority	Scenario Description
AWP.RUD.01	3.2.2	H	Select existing repo after selecting click "Update button " will update repo.
AWP.RUD.02	3.2.2	H	Select existing repo after selecting click "Update button " will not update repo.

TC ID	Requirements	Priority	Scenario Description
AWP.RCR.01	3.2.2	H	Select “Create button” after selecting it will create new repo.
AWP.RCR.02	3.2.2	H	Select “Create button” after selecting it will not create new repo.

TC ID	Requirements	Priority	Scenario Description
AWP.VTM	3.2.3	M	Select “view trained models button” after selecting it will redirecting model page.

TC ID	Requirements	Priority	Scenario Description
AWP.TM	3.2.2	M	Select “train models button” after selecting it will redirecting repo page.

TC ID	Requirements	Priority	Scenario Description
AWP.DM.01	3.2.3	H	Select existing model after selecting click “Delete button” and existing model will remove.
AWP.DM.02	3.2.3	H	Select existing model after selecting click “Delete button” and existing model will not remove.

TC ID	Requirements	Priority	Scenario Description
AWP.MVF	3.2.3	H	Model’s contents list, which mean that model’s used methods files.
AWF.MDL.01	3.2.3	H	Select created model by system who log in this system, After selecting model will delete.

TC ID	Requirements	Priority	Scenario Description
AWF.MDL.02	3.2.3	H	Select created model by user who not log in this system, After selecting model will not be remove.
AWP.CS_1.01	3.2.4	H	Click on the button to see the model information.

TC ID	Requirements	Priority	Scenario Description
AWP.CS_2.01	3.2.4	H	Click on the button to see the document upload page.
AWP.CS_3.01	3.2.4	H	Click on the button to see classification results.

TC ID	Requirements	Priority	Scenario Description
AWP.CS_3.02	3.2.4	H	Click on the button and system not automatically classified these text documents and results will not be displayed.
AWP.CHF.01	3.2.4	H	Select document from computer. After selecting documents,will be uploaded by the system.

TC ID	Requirements	Priority	Scenario Description
AWP.CHF.02	3.2.4	H	Select document from computer. After selecting documents, will not be uploaded by the system.
AWP.CL	3.2	H	Click on the button to exit the current page.
AWP.LGO	3.2	H	Click on the button to leave from system.

4.5.2 User Web Page

4.5.2.1.1 Register (AWP.RG)

The users registers to login to the system.

4.5.2.1.2 Login (AWP.LG)

The users has to login the web application system by entering username and password.

4.5.1.1.3 Repo Dataset (AWP.RD)

When the users clicks the repo button, the enrolled datasets are displayed.

4.5.1.1.4 View Files of Model (AWP.MVF)

The participant can view files that using. Users who wants to see the file names, click this button and see the model list using by the system.

4.5.1.1.5 Classify (AWP.CS_1)

The participant can classify their documents. Users who want to classify documents, click and see model information first.

4.5.1.1.6 Classify (AWP.CS_2)

The participant can classify their documents. Users who want to classify document, click and then document upload file page opens.

4.5.1.1.7 Choose File (AWP.CHF)

The participant can add text documents. Users who want to add document ,selects the document set from the computer and loads files into the system.

4.5.1.1.8 Classify (AWP.CS_3)

The participant can classify their documents. Users who want to classify document, clicks the classify button, then source code works on background and shows the classification result on new opened page. The system saved this result on database.

4.5.1.1.9 Close(AWP.CL)

The participant can close the page. Users who want to close page, can exit from the current page and return to the previous page.

4.5.1.1.10 Logout (AWP.LGO)

The participant can leave this system and account with logout button.

4.5.2.2 Test Cases

Here list all the related test cases for this feature

TC ID	**Requirements**	**Priority**	**Scenario Description**
AWP.RG.01	3.2.1	H	Enter a valid username,email and password.
AWP.RG.02	3.2.1	H	Enter a invalid username and email.

TC ID	Requirements	Priority	Scenario Description
AWP.LG.01	3.2.1	H	Enter a valid username and password.
AWP.LG.02	3.2.1	H	Enter a invalid username and password.

TC ID	Requirements	Priority	Scenario Description
AWP.RD.01	3.2.2	M	Displays the datasets store in the system and the information of the dataset.

TC ID	Requirements	Priority	Scenario Description
AWP.MVF	3.2.3	H	Model’s contents list, which mean that model’s used methods files.

TC ID	Requirements	Priority	Scenario Description
AWP.CS_1.01	3.2.4	H	Click on the button to see the model information.

TC ID	Requirements	Priority	Scenario Description
AWP.CS_2.01	3.2.4	H	Click on the button to see the document upload page.
AWP.CS_3.01	3.2.4	H	Click on the button to see classification results.

TC ID	Requirement s	Priority	Scenario Description
AWP.CS_3.02	3.2.4	H	Click on the button and system not automatically classified these text documents and results will not be displayed.
AWP.CHF.01	3.2.4	H	Select document from computer. After selecting documents,will be uploaded by the system.

TC ID	Requirements	Priority	Scenario Description
AWP.CHF.02	3.2.4	H	Select document from computer. After selecting documents, will not be uploaded by the system.
AWP.CL	3.2	H	Click on the button to exit the current page.
AWP.LGO	3.2	H	Click on the button to leave from system.

4.6 DETAILED TEST CASES

4.6.1 AWP.RG.01

TC_ID	**AWP.RG.01**
Purpose	Enter a valid username,email and password.
Requirements	3.2.1
Priority	High.
Estimated Time Needed	3 sec

Dependency	Register user test cases pass.
Setup	An user should be created.
Procedure	[A01] Go to register page.
	[A02] Click the “Register” button.
	[A03] Enter username, email, password.
	[V01] Observe that the registered is successful and the admin page appears.

4.6.2 AWP.RG.02

TC_ID	**AWP.RG.02**
Purpose	Enter a invalid username and email.
Requirements	3.2.1
Priority	High.
Estimated Time Needed	3 sec

Dependency	Register user test cases should pass.
Setup	An user should be created.
Procedure	[A01] Go to register page.
	[A02] Click the "Register" button.
	[A03] Enter username, email, password.
	[A04]Enter a invalid username and email.
	[V01] Observe that the registered is successful and the admin page appears.

4.6.3 AWP.LG.01

TC_ID	**AWP.LG.01**
Purpose	Enter a valid username and password.
Requirements	3.2.1
Priority	High.

Estimated Time Needed	3 sec
Dependency	Login Page should be opened.
Setup	An user should be created.
Procedure	[A01] Go to Login page.
	[A02] Enter a valid username.
	[A03] Enter a valid password.
	[A04] Click on the “Login” button.
	[V01] Observe that the Login is successfully and the user interface page appears.

4.6.4 AWP.LG.02

TC_ID	**AWP.LG.02**
Purpose	Enter a invalid username and password.
Requirements	3.2.1

Priority	High.
Estimated Time Needed	3 sec
Dependency	Login Page should be opened.
Setup	An user should be created.
Procedure	[A01] Go to Login page.
	[A02] Enter a valid username.
	[A03] Enter a invalid password.
	[A04] Click on the “Login” button.
	[V01] Observe that the Login is unsuccessful and "The email/password is wrong, try again." message, the Login page refreshes.

4.6.5 AWP.RD.01

TC_ID	**AWP.RD.01**
Purpose	Displays the datasets store in the system and the information of the dataset.
Requirements	3.2.2
Priority	Medium.
Estimated Time Needed	3 sec
Dependency	List Dataset test cases should pass.
Setup	An user should be created.
Procedure	[A01] Go to home page.
	[A02] Click “Repo” button.
	[V01] Observe that show the Dataset for information and needed buttons.

4.6.6 AWP.MB.01

TC_ID	**AWP.MB.01**
Purpose	The dataset stored in the system and the page where some operations are performed are opened(e.g. update, delete, upload file, view files, train).
Requirements	3.2.3
Priority	High.
Estimated Time Needed	3 sec
Dependency	List Dataset test cases should pass.
Setup	An user should be created.
Procedure	[A01] Go to home page.
	[A02] Click "Model" button.
	[V01] Observe that show the Dataset for information.

4.6.7 AWP.RTR.01

TC_ID	**AWP.RTR.01**
Purpose	After entering all parameters, the system trains.
Requirements	3.2.3
Priority	High.
Estimated Time Needed	3 sec
Dependency	Train dataset test cases should pass.
Setup	An user should be created.
Procedure	[A01] Go to home page.
	[A02] Click “Model” button.
	[A03] Click “Train” button.

	[A04] Enter the parameters.
	[V01] Observe that show the train.

4.6.8 AWP.RTR.02

TC_ID	**AWP.RTR.02**
Purpose	If there are missing parameters, the system does not train.
Requirements	3.2.3
Priority	High.
Estimated Time Needed	3 sec
Dependency	Train dataset test cases should pass.
Setup	An user should be created.
Procedure	[A01] Go to home page.

	[A02] Click “Model” button.
	[A03] Click “Train” button.
	[A04] Enter the missing parameters.
	[V01]Observe that "Missing parameters" message and go back to train.

4.6.9 AWP.RUP.01

TC_ID	**AWP.RUP.01**
Purpose	Saves the file to the system after uploading it as zip format.
Requirements	3.2.2
Priority	High.
Estimated Time Needed	3 sec
Dependency	Upload file test cases should pass.

Setup	An user should be created.
Procedure	[A01] Go to home page.
	[A02] Click “Model” button.
	[A03] Click “Upload” button.
	[A04] Upload files in zip format.
	[V01]Observe that "Successfully" message .

4.6.10 AWP.RUP.02

TC_ID	**AWP.RUP.02**
Purpose	If the file is not uploaded in zip format, it will not be saved to the system.
Requirements	3.2.2
Priority	High.

Estimated Time Needed	3 sec
Dependency	Upload file test cases should pass.
Setup	An user should be created.
Procedure	[A01] Go to home page.
	[A02] Click “Model” button.
	[A03] Click “Upload” button.
	[A04] Upload files in a format other than zip format.
	[V01]Observe that "The file you upload must be in zip format." message and go back upload page .

4.6.11 AWP.RDL.01

TC ID	AWP.RDL.01
Purpose	Select existing repo after selecting click “Delete” button and existing repo will remove.
Requirements	3.2.2

Priority	High
Estimated Time Needed	3 sec
Dependency	Delete repo test cases should pass.
Setup	Login to the system as admin
Procedure	[A01] Go to repo page.
	[A02] Admin must select existing repo and then click "Delete" button.
	[V01] Observe that the delete is successful and selecting repos removed.

4.6.12 AWP.RDL.02

TC ID	AWP.RDL.02
Purpose	Select existing repo after selecting click "Delete" button and existing repo will remove.
Requirements	3.2.2
Priority	High
Estimated Time Needed	3 sec
Dependency	Delete repo test cases should pass.
Setup	Login to the system as admin

Procedure	[A01] Go to repo page.
	[A02] Admin must select existing repo and then click “Delete” button.
	[V01] Observe that the delete is fail and selecting repos not removed.

4.6.13 AWP.RUD.01

TC ID	AWP.RUD.01
Purpose	Select existing repo after selecting click “Update” button and existing repo will update.
Requirements	3.2.2
Priority	High
Estimated Time Needed	3 sec
Dependency	Update repo test cases should pass.
Setup	Login to the system as admin
Procedure	[A01] Go to repo page.
	[A02] Admin must select existing repo and then click “Update” button.
	[V01] Observe that the update is successful and selecting repos updated.

4.6.14 AWP.RUD.02

TC ID	AWP.RUD.02
Purpose	Select existing repo after selecting click “Update” button and existing repo will update.
Requirements	3.2.2
Priority	High
Estimated Time Needed	3 sec
Dependency	Update repo test cases should pass.
Setup	Login to the system as admin
Procedure	[A01] Go to repo page.
	[A02] Admin must select existing repo and then click “Update” button.
	[V01] Observe that the update is fail and selecting repos not updated.

4.6.15 AWP.RCR.01

TC ID	AWP.RCR.01
Purpose	Select “Create button” after selecting it will create new repo.
Requirements	3.2.2

Priority	High
Estimated Time Needed	3 sec
Dependency	Create repo test cases should pass.
Setup	Login to the system as admin
Procedure	[A01] Go to repo page.
	[A02] Admin must click “Create” button.
	[A03] Add new repos.
	[V01] Observe that the create button is successful.

4.6.16 AWP.RCR.02

TC ID	AWP.RCR.02
Purpose	Select “Create button” after selecting it will create new repo.
Requirements	3.2.2
Priority	High
Estimated Time Needed	3 sec
Dependency	Create repo test cases should pass.

Setup	Login to the system as admin
Procedure	[A01] Go to repo page.
	[A02] Admin must click “Create” button.
	[A03] Add new repos.
	[V01] Observe that the create button is fail.

4.6.17 AWP.VTM

TC ID	AWP.VTM
Purpose	Select “view trained models button” after selecting it will redirecting model page.
Requirements	3.2.3
Priority	Medium
Estimated Time Needed	3 sec
Dependency	View trained models test cases should pass.
Setup	Login to the system
Procedure	[A01] Go to repo page.
	[A02] Admin must click “View trained models” button.
	[A03] After click model page opens.

	[V01] Observe that model page come up successfully.

4.6.18 AWP.TM

TC ID	AWP.TM
Purpose	Select “train models button” after selecting it will redirecting repo page.
Requirements	3.2.2
Priority	Medium
Estimated Time Needed	3 sec
Dependency	Train models test cases should pass.
Setup	Login to the system
Procedure	[A01] Go to repo page.
	[A02] Admin must click “Train models” button.
	[A03] After click repo page opens.
	[V01] Observe that repo page come up successfully.

4.6.19 AWP.DM.01

TC ID	AWP.DM.01
Purpose	Select existing model after selecting click “Delete button” and existing model will remove.
Requirements	3.2.3
Priority	Medium
Estimated Time Needed	3 sec
Dependency	Model delete test cases should pass.
Setup	Login to the system
Procedure	[A01] Go to model page.
	[A02] Admin must select existing model and then click “Delete” button.
	[V01] Observe that the delete is successful and selecting models removed.

4.6.20 AWP.DM.02

TC ID	AWP.DM.02
Purpose	Select existing model after selecting click “Delete button” and existing model will remove.
Requirements	3.2.3

Priority	Medium
Estimated Time Needed	3 sec
Dependency	Model delete test cases should pass.
Setup	Login to the system
Procedure	[A01] Go to model page.
	[A02] Admin must select existing model and then click "Delete" button.
	[V01] Observe that the delete is fail and selecting models not removed.

4.6.21 AWP.MVF

TC_ID	**AWP.MVF**
Purpose	Admin will select and see the model list using by the system.
Requirements	3.2.3
Priority	High.
Estimated Time Needed	3 sec
Dependency	System Administrator Web Page should be opened.

Setup	An Admin should be created.
Procedure	[A01] Go to the model Page.
	[A02] Select a model.
	[A03] Click to View File button of selected model.
	[V01] Observe that the View File is successful and all created models are listed.

4.6.22 AWF.MDL.01

TC_ID	**AWF.MDL.01**
Purpose	Admin who want to remove these models, can remove from the system.
Requirements	3.2.3
Priority	High.
Estimated Time Needed	3 sec
Dependency	System Administrator Web Page should be opened.

Setup	An Admin should be created.
Procedure	[A01] Go to the model Page.
	[A02] Select a model.
	[A03] Click to View File button of selected model.
	[A04] Select a file and click to Remove button.
	[V01] Observe that the remove is successful and selected models removed.

4.6.23 AWF.MDL.02

TC_ID	**AWF.MDL.02**
Requirements	3.2.3
Priority	High.
Estimated Time Needed	3 sec
Dependency	System Administrator Web Page should be opened.

Setup	An Admin should be created.
Procedure	[A01] Go to the model Page.
	[A02] Select a model.
	[A03] Click to View File button of selected model.
	[A04] Select a file and click to Remove button.
	[V01] Observe that the remove is unsuccessful and selected models not removed.

4.6.24 AWP.CS_1.01

TC_ID	**AWP.CS_1.01**
Purpose	Admin who want to classify documents, click and see model information first.
Requirements	3.2.4
Priority	High.
Estimated Time Needed	3 sec

Dependency	System Administrator Web Page should be opened.
Setup	An Admin should be created.
Procedure	[A01] Go to the model Page.
	[A02] Select a model.
	[A03] Click to Classify button of selected model.
	[V01] Observe that the Model Information page come up.

4.6.25 AWP.CS_2.01

TC_ID	**AWP.CS_2.01**
Purpose	Admin who want to classify document, click and then document upload file page opens.
Requirements	3.2.4
Priority	High.
Estimated Time Needed	3 sec

Dependency	System Administrator Web Page should be opened.
Setup	An Admin should be created.
Procedure	[A01] Go to the Model Page.
	[A02] Select a model.
	[A03] Click to Classify button of selected model.
	[A04] Click to Classify button again after come up model information page.
	[V01] Observe that the Document Upload page come up successfully.

4.6.26 AWP.CS_3.01

TC_ID	**AWP.CS_3.02**
Purpose	Admin who want to classify document, clicks the classify button, then source code works on background and not shows the classification result on new opened page. The system not saved this result on database.
Requirements	3.2.4
Priority	High.

Estimated Time Needed	2 min
Dependency	System Administrator Web Page should be opened.
Setup	An Admin should be created.
Procedure	[A01] Go to the Model Page.
	[A02] Select a model.
	[A03] Click to Classify button of selected model.
	[A04] Click to Classify button again after come up model information page.
	[A05] After Uploaded a .zip file, click Classify button.
	[V01] Observe that the Classify button is successful and results come up without error.

4.6.27 AWP.CS_3.02

TC_ID	**AWP.CS_3.02**
Purpose	Admin who want to classify document, clicks the classify button, then source code works on background and shows the classification result on new opened page. The system saved this result on database.
Requirements	3.2.4
Priority	High.
Estimated Time Needed	2 min
Dependency	System Administrator Web Page should be opened.
Setup	An Admin should be created.
Procedure	[A01] Go to the Model Page.
	[A02] Select a model.
	[A03] Click to Classify button of selected model.
	[A04] Click to Classify button again after come up model information page.

	[A05] After Uploaded a .zip file, click Classify button.
	[V01] Observe that the Classify button is successful and results come up without error.

4.6.28 AWP.CHF.01

TC_ID	**AWP.CHF.01**
Purpose	Admin who want to add document ,selects.zip files from the computer and loads files into the system.
Requirements	3.2.4
Priority	High.
Estimated Time Needed	2 min
Dependency	System Administrator Web Page should be opened.
Setup	An Admin should be created.
Procedure	[A01] Go to the Model Page.
	[A02] Select a model.

	[A03] Click to Classify button of selected model.
	[A04] Click to Classify button again after come up model information page.
	[A05] After Uploaded a .zip file
	[V01] Observe that the Uploading Document is successful and the system ready for classification .

4.6.29 AWP.CHF.02

TC_ID	**AWP.CHF.02**
Purpose	Admin who want to add document ,selects files with any format from the computer and loads files into the system.
Requirements	3.2.4
Priority	High.
Estimated Time Needed	2 min
Dependency	System Administrator Web Page should be opened.

Setup	An Admin should be created.
Procedure	[A01] Go to the Model Page.
	[A02] Select a model.
	[A03] Click to Classify button of selected model.
	[A04] Click to Classify button again after come up model information page.
	[A05] After Uploaded a any format file
	[V01] Observe that the Uploading Document is unsuccessful and the system ready is not for classification .

4.6.30 AWP.CL.01

TC_ID	**AWP.CL**
Purpose	Admin who want to close page, can exit from the current page and return to the previous page.
Requirements	3.2
Priority	High.

Estimated Time Needed	2 min
Dependency	System Administrator Web Page should be opened.
Setup	An Admin should be created.
Procedure	[A01] Go to the any page.
	[A02] Click Close button.
	[V01] Observe that the Close button is successful and the current page close properly.

4.6.31 AWP.LGO.01

TC_ID	**AWP.LGO**
Purpose	Admin who want to leave this system and account with logout button at any time.
Requirements	3.2
Priority	High.
Estimated Time Needed	2 min
Dependency	System Administrator Web Page should be opened.
Setup	An Admin should be created.
Procedure	[A01] Go to the any page.
	[A02] Click Logout button.
	[V01] Observe that the Logout button is successful and the current account close properly.

5. Conclusion

In this project, a web application developed which aims to easily determine the multiple texts of Turkish news texts with a classification model formed from datasets with millions of news. This application aims to make it more informed and effective for the best user experience in the use of news sites. As a result of the classification, the application will determine the topics according to the relevance of the text. As a result of this percentage, the user can publish the news according to the level of interest of the classified news under the headings.

6.Solution

Our solution based on two processes. One of them is train model based on given datasets. The other one is using this trained models on given news text documents. In detail, we accomplished the training task by using deep learning. Training is performed by using Magpie library which is using Google's, Tensorflow and Keras modelling algorithm. Finally we used this technique for our modelling and then used PHP for Web Application to interpret giving news text.

Acknowledgement

We would like to express our deep and sincere gratitude to our project supervisors, Prof. Dr. Erdoğan DOĞDU and Dr. Roya CHOUPANI, for giving us to the opportunity to do this project and providing invaluable guidance and many advices throughout this project. Their visions, sincerities, wisdoms and motivations have deeply inspired us and affected us in many ways.

References

1. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.
2. J.R. Quinlan. Generating production rules from decision trees. In Proceedings of the lOth Int. Joint Conferences on Artificial Intelligence, pp. 304-307, Morgan Kaufmann, 1987
3. J. Furnkranz. Separate-and-conquer rule learning. Technical Report TR-96-25, Austrian Research Institute for Artificial Intelligence, Vienna, 1996.
4. Thabtah, F., Cowling, P., & Peng, Y. (2005, January). MCAR: multi-class classification based on association rule. In *Computer Systems and Applications, 2005. The 3rd ACS/IEEE International Conference on* (p. 33). IEEE.
5. D. Lee and H. Seung. Unsupervised learning by convex and conic coding. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, Advances in Neural Information Processing Systems, volume 9, page 515. The MIT Press, 1997.
6. Y. Le Cun, B. Boser, J. Denker, D. Hendersen, R. Howard, W. Hubbard, and L. Jackel. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1:pp 541, 1989.
7. E. Brill. Some advances in transformation-based part of speech tagging. In AAAI, Vol. 1, pages 722–727, 1994.
8. Y. Even-Zohar and D. Roth. A sequential model for multi class classification. In EMNLP-2001, the SIGDAT Conference on Empirical Methods in Natural Language Processing, pages 10–19, 2001.
9. F. Jelinek. Statistical Methods for Speech Recognition. The MIT Press, Cambridge, Massachusetts, 1998
10. C. Apte, F. Damerau, and S. M. Weiss. Automated learning of decision rules for text categorization. Information Systems, 12(3):233–251, 1994.
11. I. Dagan, Y. Karov, and D. Roth. Mistake-driven learning in text categorization. In EMNLP-97, The Second Conference on Empirical Methods in Natural Language Processing, pages 55–63, 1997.
12. Har-Peled, S., Roth, D., & Zimak, D. (2003). Constraint classification for multiclass classification and ranking. In Advances in neural information processing systems (pp. 809-816).
13. T. Hastie and R. Tibshirani. Classification by pairwise coupling. In NIPS-10, The 1997 Conference on Advances in Neural Information Processing Systems, pages 507–513. MIT Press, 1998.
14. W. Maass. On the computational power of winner-take-all. Neural Computation, 12(11):2519–2536, 2000
15. Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 384–394. Association for Computational Linguistics.

16. Ronan Collobert, Jason Weston, Leon Bottou, Michael ´ Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. The Journal of Machine Learning Research, 12:2493–2537.
17. Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 151–161. Association for Computational Linguistics.
18. Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual nlp. In Proc. of CoNLL 2013.
19. Levy, O., & Goldberg, Y. (2014). Dependency-based word embeddings. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (Vol. 2, pp. 302-308).
20. Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States, pages 3111–3119.
21. code.google.com/p/word2vec/
22. https://www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/
23. https://radimrehurek.com/gensim/models/word2vec.html
24. https://github.com/cemoody/lda2vec
25. Otter, D. W., Medina, J. R., & Kalita, J. K. (2018). A Survey of the Usages of Deep Learning in Natural Language Processing. arXiv preprint arXiv:1807.10854.
26. Auria, L., & Moro, R. A. (2008). Support vector machines (SVM) as a technique for solvency analysis.
27. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. 1998. Gradient-based learning applied to document recog- nition. *In Proceedings of the IEEE*, 86(11):2278– 2324, November.
28. W. Yih, X. He, C. Meek. 2014. Semantic Parsing for Single-Relation Question Answering. *In Proceed- ings of ACL 2014*.
29. Y. Shen, X. He, J. Gao, L. Deng, G. Mesnil. 2014. Learning Semantic Representations Using Convolu- tional Neural Networks for Web Search. *In Proceed- ings of WWW 2014*.
30. N. Kalchbrenner, E. Grefenstette, P. Blunsom. 2014. A Convolutional Neural Network for Modelling Sen- tences. *In Proceedings of ACL 2014*.
31. R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuglu, P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Ma- chine Learning Research 12:2493–2537*.
32. Y. Kim, "Convolutional Neural Networks for Sentence Classification," Aug. 2014
33. Zhou, C., Sun, C., Liu, Z., & Lau, F. (2015). A C-LSTM neural network for text classification. arXiv preprint arXiv:1511.08630.
34. Mohammad, A. H., Alwada'n, T., & Al-Momani, O. (2018). Arabic text categorization using support vector machine, Naïve Bayes and neural network. GSTF Journal on Computing (JoC), 5(1).