Turkish News Text Categorization using Machine Learning

Taha Küçükkatırcı, Yaşar Fatih Enes Yalçın, Haluk Alper Karaevli
Group Aluk

Introduction & Motivation

Automating the classification is the first step in many NLP applications. The aim of our project is **to predict the topic of news** automatically in our **Turkish news dataset**. To achieve the goal, we used 3 different methods:

- Naive Bayes
- Support Vector Machine (SVM)
- Neural Network

As a dataset, we used a dataset provided by Yildiz Technical University. You may reach the dataset via the following link:

www.kemik.yildiz.edu.tr/?id=28

In the dataset, there are 41992 news in 13 topics. Some of the topics are:

- Spor
- Siyaset
- Teknoloji
- Ekonomi
- ...

Figure 1: Wordcloud for 'Spor' category

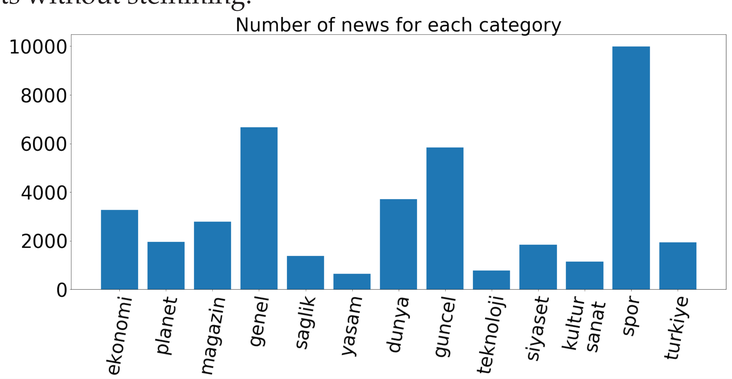
Data & Preprocessing

We did number of operations in preprocessing:

- Label each news with one-hot 13-length array with corresponding topic.
  (e.g.) [0,0,0,0,1,0,0,0,0,0,0,0,0] may mean 'Siyaset'
- Exclude all non-alphanumeric characters like &, %, $, #, _, \.
- Lower all letters.

In neural network model we also excluded words that are seen only in one news, one time. We did this extra work for word-to-vec method to run.

Finally, we have used **stemming**. We have also shown the results without stemming.



Naive Bayes

In Naive Bayes method, there is a **naïve assumption of independence** of words in a given news.

For each category, we have founded the frequency for each unique word. For example, "Galatasaray" may occur in "Spor" category for 1000 times, but not much in "Ekonomi" category.

$$P('ekonomi'|'galatasaray') = 0.000034 \quad (1)$$

Then, when a new news comes, we calculate the probability for each unique word in the news. For example, if the news text is "Fener gol yedi.", we calculate:

$$P('spor'|'Fener','gol','yedi') \propto P('spor') \prod_{i=1}^{3} P(word_i|'spor') \quad (2)$$

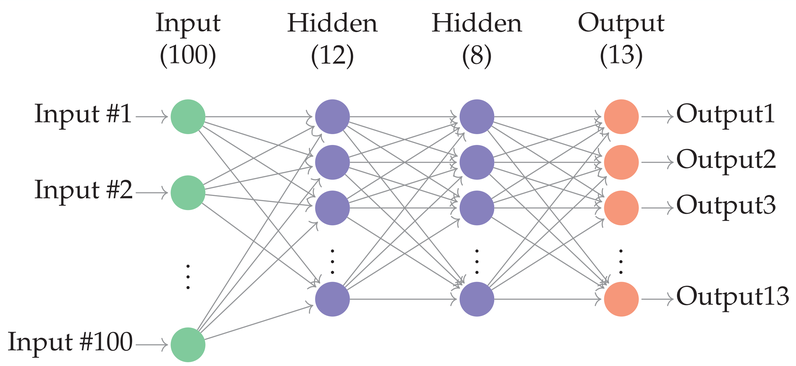
for each category. To overcome underflow, we **summed the log probabilities**. To overcome the 0-product problem, we used **Laplace smoothing** method.

SVM

Our SVM method **tokenizes** each news into words and applies regression to classify it. We used hinge loss function in SVM.

Since SVM outputs data with the margins between the seperated classes **as far apart as possible**, we thought that it would work well on our problem.

Neural Network



We used Word2Vec library to vectorize all the words in all news. Vector of a word is 100-dimensional. Then, for each news, we took the mean of the word vectors. This can be considered as **a vanilla doc-to-vec** method. So, for the MLP, we gave input 100-dimensional document vector.

We used **ReLU** in hidden layers and **Softmax** in the output layer. Then we made the maximum valued output node into value 1, and others to 0.

Results

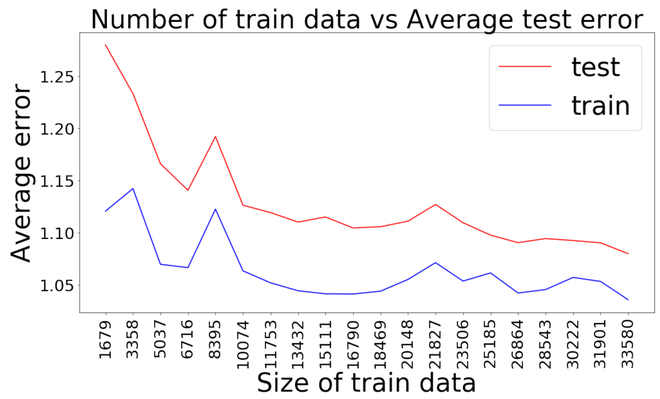
In our dataset, the labeling is not very well done. For example, "Türkiye", "Siyaset", "Güncel", "Genel" categories are all **one-within-the-others**. So, along with testing with all categories, we picked 7 categories and tested our methods on them. These 7 categories are 'Ekonomi', 'Kültür-Sanat', 'Magazin', 'Sağlık', 'Siyaset', 'Spor', 'Teknoloji'.

For neural network method, our **batch size** is always **10** and **number of epochs** is **20**. Accuracies below are all test accuracies, train accuracies are always slightly better. We used **k-fold** method in testing for $k = 5$.

Method	Stemming	Categories	Accuracy
NB	No	7	%89.68
NB	Yes	7	%91.76
SVM	No	7	%92.95
ANN	No	7	%92.83
ANN	**Yes**	**7**	**%93.27**

NB	No	13	%56.06
NB	Yes	13	%56.12
SVM	No	13	%60.26
ANN	No	13	%61.12
ANN	**Yes**	**13**	**%61.17**

Increasing number of data converges in terms of error. So, bigger dataset would probably work similarly.



Conclusion & Future Work

Our work revealed some significant outcomes about the problem, however it is still very open to improvement. Methods do not differ significantly in terms of accuracy, so one may choose the fastest method to use.