Classifying Poems based on Poem Age and Poem Type

Alihan Karatatar **Atakan Yüksel** **Ceren Korkmaz**

Abstract

Advancing science and art lies at the heart of human endeavors for ages. The roads of two areas, which were previously separate from each other, are crossing day by day. Advancing technology has now turned into art-producing artificial intelligence. Art also began to inspire science. In this study, we combined literature, which is a branch of art, with machine learning methods. We worked on the classification of the periods and subjects of the poems in our data set. While doing all this, we tried to choose the most suitable for us by comparing the results of popular machine learning algorithms such as K-nearest neighbors, Random Forest and Logistic Regression. We then experimented with BERT (Bidirectional Encoder Representations from Transformers) yet to achieve similar results.

1. Introduction

You had two options to contribute to the development of civilization in ancient times. You were going to be either an artist or a scientist. As the centuries passed, science and art began to intertwine. In the course of the civilization run that lasted for generations, we now listen to the music composed by the machines and listen to the sounds they create. We watched them draw landscapes that were not even and create human figures.

In a world where civilization has developed so much, we made a small start to make our personal contribution to this age and decided to develop a machine learning algorithm. Our subject was literature that attracted everyone's attention. Since the subject is science, we found it appropriate to work on the age of enlightenment renaissance.

In our project, we are trying to classify text data based on its subject and age. Subject in this case can be Love, Nature and Mythology & Folklore. Age can be either Renaissance or Modern. To achieve this, we tried the following models; K-Nearest Neighbours, Logistic Regression, Support Vector Machines, Naive Bayes, Decision Tree and Random Forest.

We get the best results with Logistic Regression, Support Vector Machines and Random Forest and the worst result with Naive Bayes. The results from classification using BERT are also on par with the best shallow learning models.

2. Related Work

Jasleen Kaur et al.(Kaur & Saini, 2018) created a project to classify 240 poetries according to their theme with the objective to find the best machine learning algorithms. They tried 10 different algorithms which were Adaboost, Bagging, C4.5, Decision Tree, Hyperpipes, K-nearest Neighbour, Naive Bayes, PART, SVM, Voting Feature Interval and ZeroR. According to their work, four algorithms NB, K-Nearest KN), SVM and Hyperpipes performed better than the other algorithms. So, these four algorithms were selected for further experimentation on classifying poems. Selected four were trained and tested on 2034 poems. These 2034 poems were passed through various text pre-processing phases: tokenization, stop word removal, special symbol removal, stemming. With TF weighing of all extracted tokens and using lexical feature, SVM outperformed all other machine learning algorithms with accuracy of 72.04%.

In another study, authors(Can et al., 2012) developed an automatic text categorization by time period and by poets for ottoman poems. For this purpose, they use two fundamentally different machine learning methods: Naive Bayes and Support Vector Machines, and employ 4 style makers: most frequent words, token length, two-word collocations, type lengths. For SVM method they have two different kernel function. Polynomial(p) and Radial-basis-function(rbf) kernels. After designing experiment and testing parts, a two ways analysis of variance (ANOVA) table is conducted order to see classification performances of the tested cases. According to those results most frequent method with SVM-p has best accuracy score for poets. It has 92.80% accuracy. For the classification of time period, again most frequent word with SVM method gives the best accuracy score which is up to 94%. Second place taken by two-word-collocation with SVM. It can be clearly seen that SVM are much better than Naive Bayes.

BERT was created and published by Jacob Devlin(Devlin et al., 2019), and his colleagues from Google in 2018. BERT, which stands for Bidirectional Representations from Transformers, is a language representation model. Unlike recent language representation models, BERT is designed to pre-

train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

3. The Approach

3.1. Dataset

The data we will use can be found on kaggle[1]. The dataset has 573 poems and contains 5 columns; author, content, poem name, age, type where age is either Renaissance or Modern and type is Love, Nature or Mythology & Folklore.

Table 1. Example poem from our dataset

Author	Content	Poem Name	Age	Type
William Shakespeare	Let the bird of loudest lay On the sole Arabian tree Herald sad and trumpet be, To whose sound chaste wings obey. But thou shrieking harbinger, Foul precurrer of the fiend, Augur of the fever's end, To this troop come thou not near. From this session interdict Every fowl of tyrant wing, Save the eagle, feather'd king Keep the obsequy so strict. Let the priest in surplice white, That defunctive music can, ...	The Phoenix and The Turtle	Renaissance	Mythology & Folklore

Table 2. Non-Null Data

COLUMN	NON-NULL COUNT
AUTHOR	573
CONTENT	573
POEM NAME	571
AGE	573
TYPE	573

It can be seen that 2 poems are missing their names. This will not affect our work because we will classify the poems by their age and type, using their names-which are all different- would not help with classification.

Table 3. Data distribution according to type

TYPE	CONTENT
LOVE	326
MYTHOLOGY & FOLKLORE	59
NATURE	188

[1] https://www.kaggle.com/ultrajack/modern-renaissance-poetry

Table 4. Data distribution according to age

AGE	CONTENT
MODERN	258
RENAISSANCE	315

After creating a table according to poem types you can see most of the poems' types are Love. And after distribution according to age, we can say more than half of the belong to Renaissance era.

The distribution visualization of the data is below:

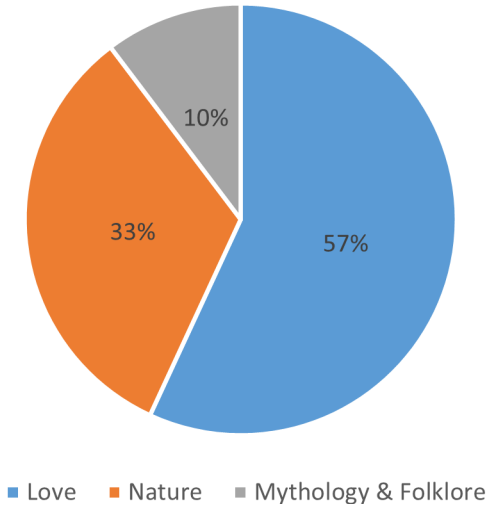


Figure 1. Type Distribution

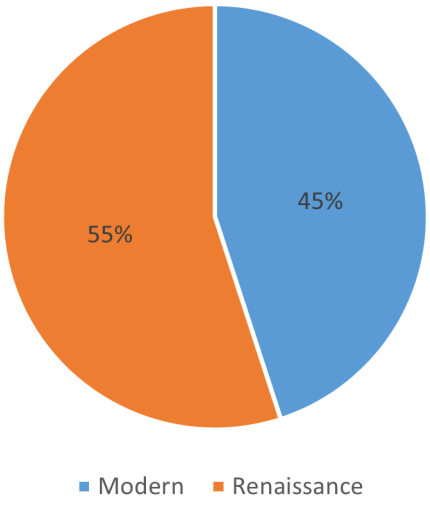


Figure 2. Age Distribution

To feed into our models, we first converted all our poems to lower characters and removed all punctuation. We did not apply stemming for now, as we suspect stemming may not be suitable for poems but we will look further into this in the future. We then used TFIDF to extract features from the corpus. We haven't removed any stopwords yet since the poems are grandiloquent sometimes stopwords can be used to describe the emotions. We will try and see how removing stopwords affect the accuracy and will report on it in further studies. Since we don't have separate data for training and testing, we will randomly split the dataset so that %80 of poems will be used for training and the rest for testing.

3.2. Shallow Learning

3.2.1. K-Nearest Neighbors and Weighted K-Nearest Neighbors

K-Nearest Neighbors is a machine learning technique and algorithm that can be used for both regression and classification tasks. K-Nearest Neighbors examines the labels of a chosen number of data points surrounding a target data point, in order to make a prediction about the class that the data point falls into. K-Nearest Neighbors (KNN) is a conceptually simple yet very powerful algorithm, and for those reasons, it's one of the most popular machine learning algorithms.

Weighed KNN, or W-KNN, differs from standard in the consideration of points. In standard KNN, all points are taken into account equally, whereas in the W-KNN further point weigh less. (effect the final result less.)

Determining the value K can be tricky. If we know the number of labels, or cluster centers from beforehand, that is our K value. If not, using elbow method[2] or silhouette analysis[3] may help determine the number of clusters.

3.2.2. Logistic Regression

Logistic regression is a type of regression analysis. Regression analysis is a type of predictive modeling technique which is used to find the relationship between a dependent variable (usually known as the "Y" variable) and either one independent variable (the "X" variable) or a series of independent variables. When two or more independent variables are used to predict or explain the outcome of the dependent variable, this is known as multiple regression. Regression analysis can be broadly classified into two types: Linear regression and logistic regression.

Logistic regression is a classification algorithm. It is used to predict a binary outcome based on a set of independent variables.

3.2.3. Support Vector Machine

In machine learning, support-vector machines are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. SVM maps training examples to points in space so as to maximise the width of the gap between the two categories. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.(Cortes & Vapnik, 1995)

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

3.2.4. Naive Bayes

In statistics, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with naive independence assumptions between the features (see Bayes classifier). They are among the simplest Bayesian network models, but coupled with kernel density estimation, they can achieve higher accuracy levels.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

3.2.5. Decision Tree

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.

In decision analysis, a decision tree and the closely related influence diagram are used as a visual and analytical deci-

[2]Elbow method increases K and observes sample error rate. As long as the error rate decreases K is increased.

[3]Silhouette analysis can be used to study the separation distance between the resulting clusters.

sion support tool, where the expected values (or expected utility) of competing alternatives are calculated.

3.2.6. Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of over fitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance. (Ho, 1995)

3.3. Deep Learning

We used BERT (Bidirectional Encoder Representations from Transformers) for classification. BERT in it's core, is a model that understands language. It can be used to answer questions, translate etc. Today, it is being used by Google for certain search queries. (Nayak, 2019) Bert has two models, the BASE model, 12 encoders with 12 bidirectional self-attention heads, and LARGE, 24 encoders with 16 bidirectional self-attention heads. (Devlin et al., 2019) To measure the performance of BERT, or any other natural language understanding tasks, the following test are applied. BERT managed to get better results than their competitors.

- **GLUE.** The General Language Understanding Evaluation (GLUE) benchmark is a collection of resources for training, evaluating, and analyzing natural language understanding systems. (Wang et al., 2019)
- **SQuAD.** Stanford Question Answering Dataset is a reading comprehension dataset, consisting of questions posed by crowd workers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable.(Rajpurkar et al., 2018)
- **SWAG.** Given a partial description like "she opened the hood of the car," humans can reason about the situation and anticipate what might come next ("then, she examined the engine"). SWAG (Situations With Adversarial Generations) is a large-scale dataset for this task of grounded commonsense inference, unifying natural language inference and physically grounded reasoning. (Zellers et al., 2018)

BERT is a specific, large transformer masked language model. A language model is a statistical model of the probability of a sentence or phrase. An example is the probability of the sentence P("I love BBM406") is higher than P("BBM406 love I"). A language model trained by removing words and having the model fill in the blank is called a masked language model. This is considered as one of the big contribution of BERT as a way of training language models. (Devlin et al., 2019)

A transformer is a novel architecture that aims to solve sequence-to-sequence tasks while handling long-range dependencies with ease. It relies entirely on self-attention[4] to computer representations of its input and output without using sequence-aligned RNNs or convolution.

The original transformer architecture uses a traditional a sequence-to-sequence model, where you have an encoder that takes your input and turns it into embeddings, and the decoder that takes those embeddings and turns them into a string output.

BERT is a little bit different. It uses multiple encoders and stacks them on top of each other. One way of using the model is to take the embeddings from the multiple encoders and use those as input to a new classifier we train.

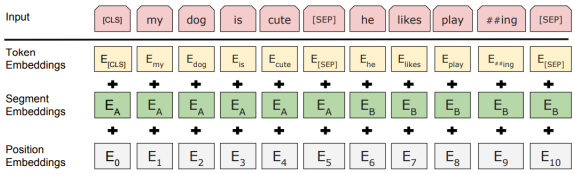


Figure 3. The input representation of BERT (Devlin et al., 2019)

Apart from having word embeddings, BERT also has two unique tokens.

- **[CLS].** defines the class(age or type) of the token (poem).
- **[SEP].** is used to show the end of the tokens.

The way we are using BERT is using our additional classifier to fine tune the original BERT model. So that we are updating the weights the BERT uses.

[4]Self attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence.

4. Experimental Results

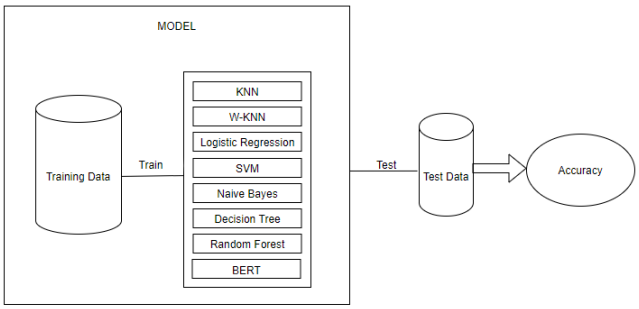


Figure 4. Flowchart of experiments

4.1. K-Nearest Neighbors and Weighted K-Nearest Neighbors

After designing experiment, K-Nearest Neighbors Algorithm (KNN) applied as first algorithm. After that, Weighted K-Nearest Neighbors Neighbors (W-KNN) algorithm is applied. We chose to apply these algorithms first because they are easy to apply at the beginning. K value was chosen as 3 after running test on values and euclidean distance is used as the distance metric. We observed very good accuracy for age classification, for KNN it is %93 and for W-KNN it is %90. Nevertheless, results for type classification is a little bad. They are %60 for KNN and %58 for W-KNN.

4.2. Logistic Regression

Multi-nominal Logistic Regression (LR) has been chosen as second algorithm. LR is easy to implement, interpret, and very efficient to train. It is very fast at classifying unknown records. With solver=”liblinear” and random_state=15 ,it has %74 accuracy for type classification and %93 for age classification.

4.3. Support Vector Machine

Support vector machines are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis.They are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis. SVM are very good when we have no idea about the data. It works well with even unstructured and semi structured data like text, images and trees. It has %71 accuracy for type classification and %96 for age classification using ”rbf” kernel. Different kernels with different parameters have been tested(”linear”, ”poly”) but these gave worse results.

4.4. Naive Bayes

Naive Bayes Classifier (NB) is simple and easy to implement. It does not require as much training data and handles both continuous and discrete data. It is also fast, effective and can be used in real time prediction. In this experiment, it has %65 accuracy for type classification and %75 accuracy for age classification.

4.5. Decision Tree

Decision Tree (DT) algorithm is also easy to read and interpret. Even, without any statistical background knowledge, it is meaningful for readers. Also, it needs to less data cleaning. In this experiment, it has %60 accuracy for type classification and %82 accuracy for age classification.

4.6. Random Forest

Random Forest (RF) is a machine learning method that operates by constructing multiple DT. It reduces over fitting in DT and helps to improve accuracy. It works well both categorical and continuous values. In this experiment, with n_estimators=100, it has %64 accuracy for type classification and %92 accuracy for age classification.

4.7. BERT

The BERT is a neural network-based technique for natural language processing. It uses the Transformer encoder architecture to process each token of input text in the full context of all tokens before and after. Overall, we got our best results using BERT. With learning rate of 2e-5, optimizer AdamW, epoch amount of 25, we got 76% accuracy for Type and 98% for Age.

Figure 5. Age Classification Accuracy with Epochs

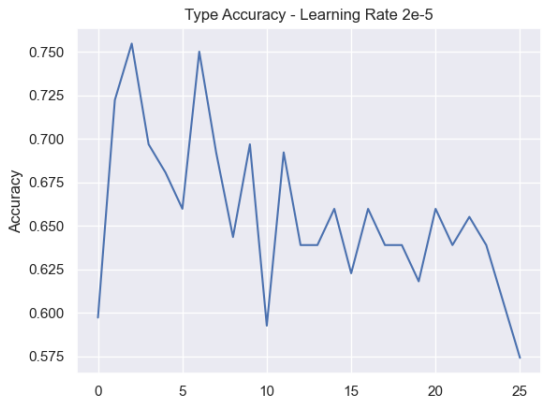
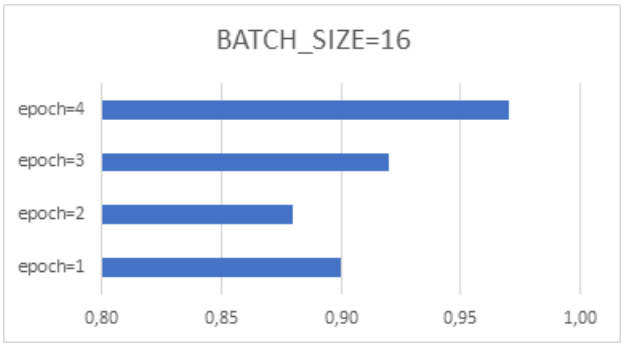
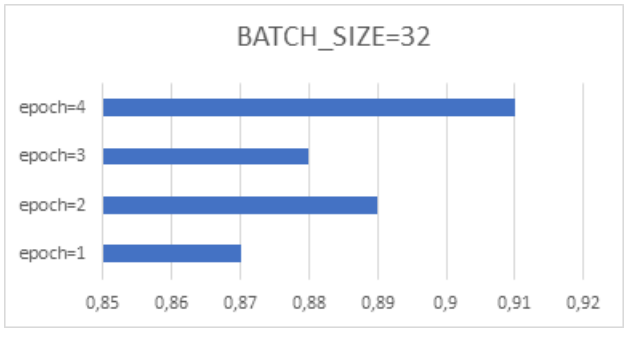
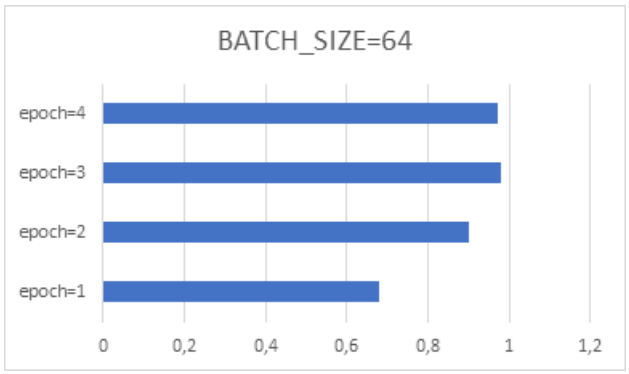


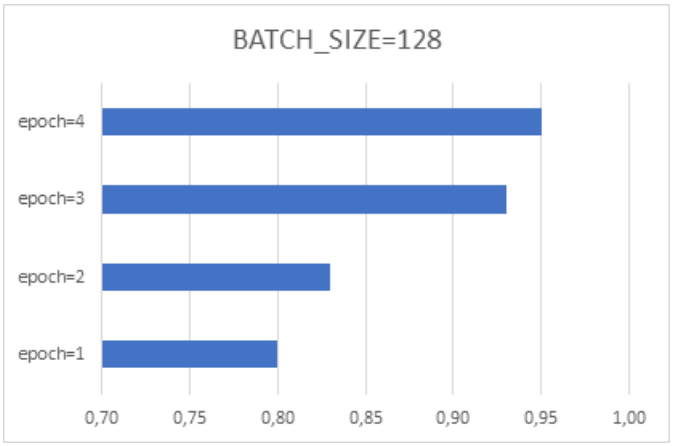
Figure 6. Type Classification Accuracy with Epochs

We repeated our experiment under different conditions. Since, the BERT authors say the best epoch number is four, it has been tried the epoch numbers from 1 to 4 under different batch sizes. Related charts can be seen below.

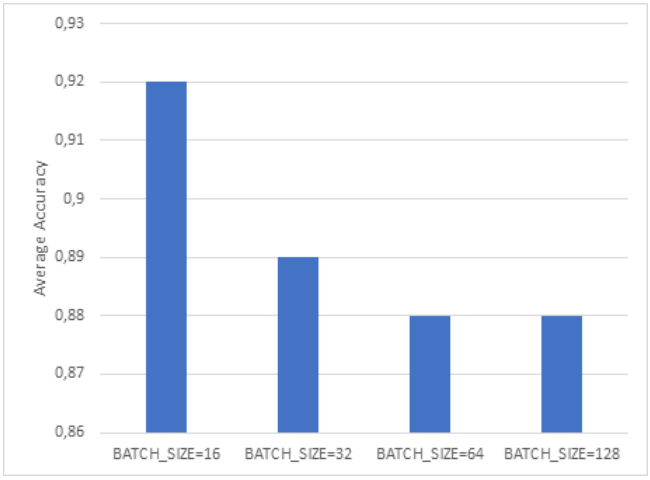






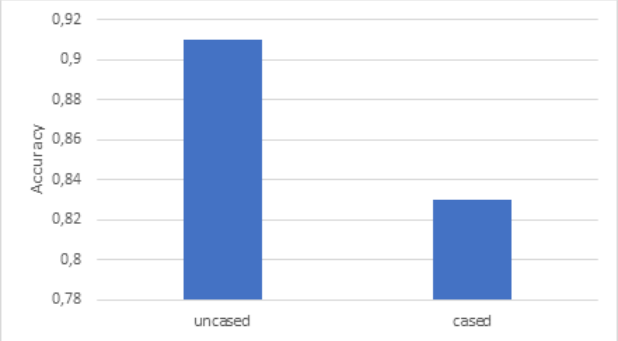


To sum up, chart due to average results is following:



According to the experimental results, the best accuracy score was obtained when the batch size was 16. In addition, a result similar to what the authors said was obtained in the experiment. Despite the changing batch sizes and epoch numbers, the best result was observed when the epoch number was 4. When the batch size was 16 and the epoch number was 4, the BERT algorithm provides %97 accuracy

results.
Case sensitivity is another condition control to diversify test results. With the batch size was 16 and the epoch number was 4, cased and uncased situations were obtained, and results are following:



According to chart above, BERT has more accurate when case sensitivity ignored. When case sensitivity is considered, it is %91 is certain, when it is considered, %83 accuracy score was obtained.

4.8. Tables

Table 5 contain accuracy results for the algorithms we've tested. Every algorithm is trained and tested on same data each iteration and they have been ran 50 times for average results. It can be seen that for type classification the best algorithm is Logistic Regression and for age classification SVM gives the best result. For BERT, learning rate of 2e-5 and epoch number of 25 is used to achieve the best results.

Table 5. Classification accuracies for different methods.

METHOD	TYPE	AGE
KNN	0.60	0.93
W-KNN	0.58	0.90
LOGISTIC REGRESSION	0.74	0.93
SVM	0.71	0.96
NAIVE BAYES	0.65	0.75
DECISION TREE	0.60	0.82
RANDOM FOREST	0.64	0.92
BERT	0.76	0.98

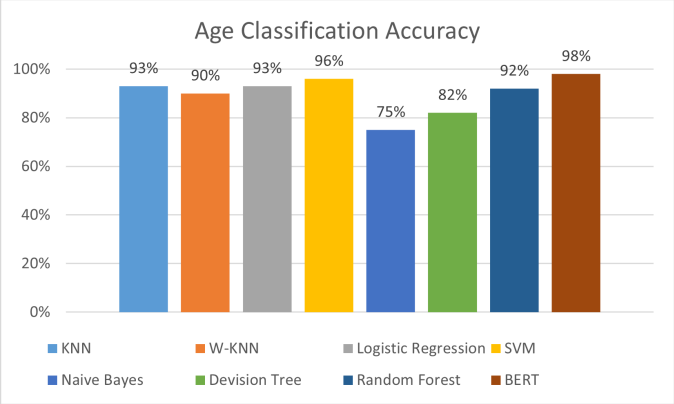


Figure 7. Visualization of accuracies for age classification

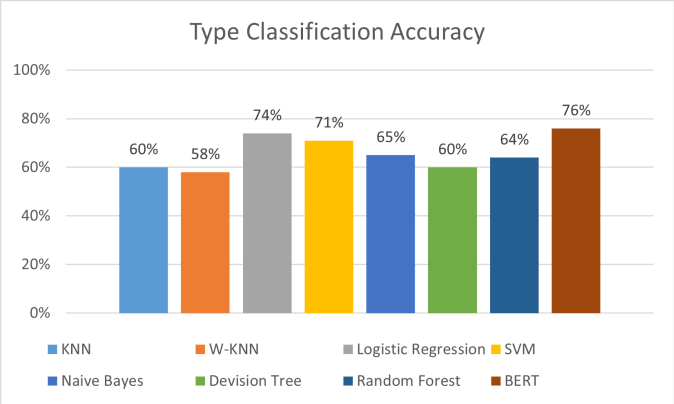


Figure 8. Visualization of accuracies for type classification

5. Conclusions

We thought the accuracy for type classification would be higher using BERT. It was only slightly better than common machine learning algorithms. For age classification on the other hand, even though most of the algorithms gave really good results, BERT was almost perfect with %98 accuracy. This may be because the dataset is better for age classification rather than type.

Our dataset consisted 3 types of poems: Love, Nature and Mythology & Folklore. Not all of the poems written belongs to these categories. This can also be said about age classification. There are lots of poems from different times. We would like to improve our project so that we could classify more types of poetry and get better accuracy as well.

Right now, we make predictions for age and type seperately. We want to create a multi-tasking algorithm so we can combine these two predictions. Maybe we can see how they correlate with each other too.

References

Alpaydin, E. *Introduction to Machine Learning (3rd Edition)*. MIT Press, 2014.

Can, E. F., Can, F., Duygulu, P., and Kalpakli, M. Automatic categorization of ottoman literary texts by poet and time period. 2012.

Cortes, C. and Vapnik, V. Support-vector networks. In *Machine Learning*, pp. 273–297, 1995.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Ho, T. K. Random decision forests, 1995.

III, H. D. *A Course in Machine Learning*.

Kaur, J. and Saini, J. Designing punjabi poetry classifiers using machine learning and different textual features. *The International Arab Journal of Information Technology*, 17(1), 2018.

Nayak, P. Understanding searches better than ever before. Technical report, 2019.

Rajpurkar, P., Jia, R., and Liang, P. Know what you don't know: Unanswerable questions for squad, 2018.

Russell, S. and Norvig, P. *Artificial Intelligence: A Modern Approach 3rd Edition*. Prentice Hall, 2009.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. Glue: A multi-task benchmark and analysis platform for natural language understanding, 2019.

Zellers, R., Bisk, Y., Schwartz, R., and Choi, Y. Swag: A large-scale adversarial dataset for grounded commonsense inference, 2018.