# Artificial Intelligence-Based Analysis of Software Requirements

# Yazilim Gereksinimlerinin Yapay Zeka Tabanli Analizi

*Jordi Hako Nintcheu İzmıi katip Çelebi üniversitesi, Yazılım mühendisliği yüksek lisans, İzmir, Türkiye*

***Abstract*—This paper explores the application of artificial intelligence (AI) techniques in analysing software requirements. The study focuses on 2 main objectives. The first is using AI to distinguish between functional and non functional software requirements. The second is leveraging AI to accurately recognize the differences between AI-generated and human-written software requirements for software development. Throughout the study, a series of Natural language processing tools and strategies are used and compared to find out which could possibly render the best results. The results shows that the best accuracy in classifying software requirements as AI-generated or Human-generated is obtained by converting our dataset features using the Tf-idf or One-hot text to vector transform techniques and training the Logistic Regression, Random forest, Decision Tree or Support vector machine (SVM) algorithm on these. The combination of any of these data transforms and any of these models turns out to produce the best possible f1-scores.**

***Keywords — Artificial Intelligence, Software Requirements, Requirements Analysis, Natural Language Processing, Machine Learning.***

***Özetçe*—Bu makale, yazılım gereksinimlerini analiz etmede yapay zeka (YZ) tekniklerinin uygulanmasını incelemektedir. Çalışma, iki ana hedefe odaklanmaktadır. Birincisi, YZ kullanarak yazılım geliştirme için YZ tarafından üretilen ve insan tarafından yazılmış yazılım gereksinimleri arasındaki farkları doğru bir şekilde tanımaktır. İkincisi ise, YZ kullanarak işlevsel ve işlevsel olmayan yazılım gereksinimlerini ayırt etmektir. Çalışma boyunca, hangi doğal dil işleme araçlarının ve stratejilerinin en iyi sonuçları verebileceğini bulmak için çeşitli doğal dil işleme araçları ve stratejileri kullanılmış ve karşılaştırılmıştır. Sonuçlar, yazılım gereksinimlerini YZ tarafından üretilmiş veya insan tarafından üretilmiş olarak sınıflandırmada en iyi doğruluğun, veri kümesi özelliklerini Tf-idf veya One-hot'a dönüştürüp, Lojistik Regresyon, Rastgele Orman, Karar Ağacı veya Destek Vektör Makinesi (SVM) üzerinde eğiterek elde edildiğini göstermektedir. Bu veri dönüşümlerinden herhangi birinin ve bu modellerden herhangi birinin kombinasyonu, mümkün olan en iyi f1-puanlarını üretmektedir.**

***Anahtar Kelimeler — Yapay Zeka, Yazılım Gereksinimleri, Gereksinim Analizi, Doğal Dil İşleme, Makine Öğrenimi.***

## I. Introduction

Software requirements analysis is a critical phase in the software development lifecycle, ensuring that the final product meets user needs and expectations. Traditional methods often involve manual processing, which can be time-consuming and prone to errors. With the advent of artificial intelligence, there is a potential to automate and enhance this process. Furthermore, determining whether text is AI generated or human generated is another emerging necessity. With the increasing accessibility of large language models (LLMs), accurately evaluating human knowledge becomes more and more difficult. This is because most people (students at the front line), tend to rely more on these LLMs for their compulsory academic task, hence making little or no effort to improve their personal knowledge. This paper first investigates an AI-based approach to analysing software requirements, aiming to improve accuracy and efficiency in determining whether they are functional or non functional. Then, it determines if these software requirements are Human-generated or AI-generated. This research has two primary contributions. The first includes the development of a novel AI-based framework for requirements analysis and the demonstration of its effectiveness through empirical evaluation. The second includes the development of a novel AI-based framework to distinguish AI generated text from Human-written text.

## II. Related Work

Previous research has explored various AI techniques for software requirements analysis. One main concern in such research has been the acquisition of a viable datasource for analysis. [1] Some other researchers have attempted to extract software requirements from Tacit's knowledge base. Nonetheless, no matter the method of acquisition of these data, creating a reliable data set for analysis in a language like Turkish isn’t always the easiest task. [2] Some work has also been done in providing useful information for creating datasets for natural language processing in Turkish. Natural language processing (NLP) has been widely used to parse and understand requirements written in natural language. These natural languages have involved several languages from English, to Arabic, to Chinese. These researches do not come without challenges. When it comes to the Turkish language, which is

the language used in this paper, [3] some work has been done in identifying the challenges related to natural language processing in the Turkish language, summarising information about natural language processing techniques, systems and various sources developed for Turkish. Machine learning algorithms have also been employed to classify and prioritise

```
In [4]: df.describe()
Out[4]:
```

| | Gereksinim | Gereksinim Tipi | Yazar |
|---|---|---|---|
| count | 400 | 400 | 400 |
| unique | 342 | 2 | 2 |
| top | Bakım: Uygulamanın kolayca güncellenebilir olm... | Fonksiyonel | İnsan |
| freq | 9 | 200 | 200 |

requirements. This section reviews the state-of-the-art in AI applications for requirements analysis, highlighting the strengths and limitations of existing approaches. Comparisons are made to position the proposed method within the context of current research.

## III. Methodology

The proposed AI-based framework for software requirements analysis involves several key steps:

| Gereksinim | Gereksinim Tipi | Yazar |
|---|---|---|
| kullanıcı TC kimlik veya yabancı kimlik numarasıyla giriş yapabilmelidir | Fonksiyonel | İnsan |
| kullanıcı bakiyesi ana ekranda görebilmelidir | Fonksiyonel | İnsan |
| Kullanıcı başka kullanıcılerin, TCsi, emailleri, ibani veya adresini kullanarak para gönderebilmelidir | Fonksiyonel | İnsan |
| Yeni kullanıcı, eDevlet hesabıyla ya da email adresini kullanarak ve yeni şifre oluşturarak kayıt olabilmelidir | Fonksiyonel | İnsan |
| Kullanıcı western union dan para gönderip teslim alabilmelidi | Fonksiyonel | İnsan |
| Kullanıcı ana ekranda yaptığı son on işlemleri görebilmelidir | Fonksiyonel | İnsan |
| Kullanıcı var olan hesaplara bir menü butonuna tıklayarak ulaşabilmeli, yeni hesap bir artı butonuna basarak oluşturabilmelidir | Fonksiyonel | İnsan |
| Kullanıcı hesaba özel bir barkod kullanarak iban bilgilerini paylaşabilmelidir | Fonksiyonel | İnsan |
| Uygulamanın iki modu olmalıdır. Biri gündüz, bir gece modudur. Kullanıcı bir buton basarak bu iki modun arasında geçebilmelidir | Fonksiyonel | İnsan |
| Kullanıcı anlık kur görebilir, üç farklı döviz çeşidi satın alıp satabilmelidir. Bunlar USD, EUR, GBP | Fonksiyonel | İnsan |
| Kullanıcıların hesap bilgilerini görüntüleme: Hesap bakiyeleri, özetler, kart bilgileri ve diğer finansal detaylar görüntülenebilir olmalıdır. | Fonksiyonel | ChatGPT |
| Para transferi yapabilme: Kullanıcılar, başka hesaplara para transferi yapabilmeli ve transfer detaylarını belirleyebilmelidir. | Fonksiyonel | ChatGPT |
| Fatura ödemelerini gerçekleştirebilme: Kullanıcılar, fatura ödemeleri için gerekli bilgileri girerek ödeme yapabilmelidir. | Fonksiyonel | ChatGPT |
| Banka kartı ve kredi kartı işlemlerini yönetebilme: Kart limitleri, harcama detayları, kart iptal ve yenileme işlemleri yönetilebilmelidir. | Fonksiyonel | ChatGPT |

### A. Data Collection

Software requirements were gathered using two strategies. The first was manual entry, and the second, was using ChatGPT. A total of 400 data entries were obtained. This data was collected across 10 different software scenarios. First, 10 different software scenarios were thought-of and included in a word document. Each scenario was written in detail. Next, for every scenario, 10 functional and 10 non functional software requirements were written by both a human and then ChatGPT 3.5. This totaled to around 20 functional requirements (10 by human and 10 by AI), and 20 non-functional requirements (10 by human and 10 by AI). The resulting dataset was exported and stored as a CSV for data preprocessing and model training.

Dataset first lines

### B. Data Import and Analysis

Popular data analysis functions from the [4] Pandas python library, such as [5] read_csv(), head(), describe() and info() were used to import and analyse the dataset. This provided a better understanding of the dataset's content and the data distribution. It also gave important information like the frequency of occurrence and the datatype of our data.

Dataset describe() function output

### C. Preprocessing

The collected data was cleaned and preprocessed to remove noise and ensure consistency. Techniques such as tokenization, stop-word removal, and stemming were applied. First, using the Natural Language Toolkit python library(NLTK), the stopwords and wordnet lists were downloaded. Then, the imported text was preprocessed by first setting every letter to lowercase, then removing all digits and non-word characters, then tokenizing, followed by removing the stopwords and finally lemmatizing. The lemmatized text was then added to the original dataset.

### D. Features extraction

The processed software requirements were converted into numeric vectors for the AI models convenience. The popular methods used in this research were [6] Term frequency-inverse document frequency (TF-IDF), Count Vectorization, Word2Vec, Doc2Vec and One-Hot encoding. The transforming of these software requirements into these vector representations resulted into 5 different data transforms to be used for classification.

### E. Model Evaluation

5 different classification AI models were evaluated with the 5 available data transforms. These models were, Logistic regression, Random forest, Decision tree, Support Vector Machine and K nearest neighbours. The evaluation was done for the 2 different classification targets. The first target was classifying the software requirements as functional or non-functional, and the second was classifying them as AI or Human generated. Evaluation metrics such as precision, recall, F1-scores and confusion matrices were used and the results were stored in a dictionary for analysis.

### F. Analysis Process

The AI models were trained on a labelled dataset and used to analyse new requirements. The results were printed to the screen for review purposes. For better understanding of these results, plotting techniques like, confusion matrix heat maps, and histogram plottings were used.

## IV. Experiments and Results

Experiments were conducted using a dataset of software requirements from both humans and ChatGPt. This data was further sub-categorised as functional and non-functional. The experimental setup included a high-performance computing environment to train and evaluate the AI models. An appropriate looping approach was used to cross-train all the models with the various transformed datasets. Evaluation metrics such as precision, recall, and F1-score and confusion

metrics were used to assess model performance. The results demonstrated two things. First, a high level of accuracy in identifying and classifying requirements as functional or non-functional, with the AI models outperforming traditional manual methods. Second, a high level of accuracy in identifying and classifying requirements as AI-generated or Human-written. Overall, the TF-IDF and One-Hot data transforms shown to perform best when coupled with one of the Logistic Regression, Random Forest, Decision Tree or Support vector machine algorithms for both targets. Detailed results are presented in tables and figures, illustrating the effectiveness of the proposed approach.

TARGET 1 - TF-IDF

| **Target 1 model results with TF-IDF** | **Model Name** | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.8017 | 0.8017 | 0.8017 | 0.8017 | 0.7876 |
| Recall | 0.6375 | 0.6375 | 0.6375 | 0.6375 | 0.5875 |
| F1-Score | 0.6044 | 0.6044 | 0.6044 | 0.6044 | 0.5341 |

Target 1's results with TF-IDF.

TARGET 1 - COUNT VECTOR

| **Target 1 model results with TF-IDF** | **Model Name** | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.8017 | 0.8017 | 0.8017 | 0.8017 | 0.3164 |
| Recall | 0.6375 | 0.6375 | 0.6375 | 0.6375 | 0.5625 |
| F1-Score | 0.6044 | 0.6044 | 0.6044 | 0.6044 | 0.4049 |

Target 1's results with Count Vector

TARGET 1 - WORD2VEC

| **Target 1 model results with TF-IDF** | **Model Name** | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.1914 | 0.6581 | 0.6187 | 0.4787 | 0.3164 |
| Recall | 0.4375 | 0.6 | 0.5750 | 0.4875 | 0.5625 |
| F1-Score | 0.2663 | 0.5873 | 0.5653 | 0.4812 | 0.4049 |

Target 1's results with Word2Vec

TARGET 1 - DOC2VEC

| **Target 1 model results with TF-IDF** | **Model Name** | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.1914 | 0.5829 | 0.5387 | 0.4666 | 0.5149 |
| Recall | 0.4375 | 0.55 | 0.5125 | 0.45 | 0.4875 |
| F1-Score | 0.2663 | 0.5432 | 0.5067 | 0.4479 | 0.4778 |

Target 1's results with Doc2Vec

TARGET 1 - ONE-HOT

| **Target 1 model results with TF-IDF** | **Model Name** | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.8017 | 0.8017 | 0.8017 | 0.8017 | 0.7876 |
| Recall | 0.6375 | 0.6375 | 0.6375 | 0.6375 | 0.5875 |
| F1-Score | 0.6044 | 0.6044 | 0.6044 | 0.6044 | 0.5341 |

Target 1's results with One-Hot

TARGET 2 - TF-IDF

| **Target 1 model results with TF-IDF** | **Model Name** | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.8095 | 0.8095 | 0.8095 | 0.8095 | 0.7534 |
| Recall | 0.6875 | 0.6375 | 0.6875 | 0.6875 | 0.5250 |
| F1-Score | 0.6568 | 0.6568 | 0.6568 | 0.6568 | 0.3745 |

Target 's results with TF-IDF

TARGET 2 - COUNT VECTOR

| **Target 1 model results with TF-IDF** | **Model Name** | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.8095 | 0.8095 | 0.8095 | 0.8095 | 0.8005 |
| Recall | 0.6875 | 0.6375 | 0.6875 | 0.6875 | 0.6625 |
| F1-Score | 0.6568 | 0.6568 | 0.6568 | 0.6568 | 0.6230 |

Target 's results with Count Vector

TARGET 2 - WORD2VEC

| Target 1 model results with TF-IDF | Model Name | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.2376 | 0.6434 | 0.7120 | 0.6375 | 0.6124 |
| Recall | 0.4875 | 0.6375 | 0.6875 | 0.6375 | 0.6125 |
| F1-Score | 0.3195 | 0.6355 | 0.6803 | 0.6366 | 0.6115 |

Target 's results with Word2Vec

TARGET 2 - DOC2VEC

| Target 1 model results with TF-IDF | Model Name | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.2376 | 0.6264 | 0.5620 | 0.4480 | 0.5368 |
| Recall | 0.4875 | 0.6250 | 0.5625 | 0.45 | 0.5375 |
| F1-Score | 0.3195 | 0.6221 | 0.5614 | 0.4479 | 0.5364 |

Target 's results with Doc2Vec

TARGET 2 - ONE-HOT

| Target 1 model results with TF-IDF | Model Name | | | | |
|---|---|---|---|---|---|
| | ***Logistic Regression*** | ***Random Forest*** | ***SVM*** | ***Decision Tree*** | ***KNN*** |
| Precision | 0.8095 | 0.8095 | 0.8095 | 0.8095 | 0.7534 |
| Recall | 0.6875 | 0.6375 | 0.6875 | 0.6875 | 0.5250 |
| F1-Score | 0.6568 | 0.6568 | 0.6568 | 0.6568 | 0.3745 |

Target 's results with One-Hot

## V. DISCUSSION

The results indicate that the AI-based approach significantly improves the efficiency and accuracy of software requirements analysis. This was better reflected in the ability of the models to accurately classify these requirements as functional and non-function, or AI-generated and Human-written. The use of NLP techniques allows for a more comprehensive understanding of natural language requirements, while machine learning algorithms effectively classify and prioritise them. However, the approach is not without limitations. The quality of the results is highly dependent on the quality of the training data, and the models may require retraining for different types of projects. The implications of these findings suggest that AI can greatly enhance the software development process and the recognition of AI-generated content, but further research is needed to address the limitations and explore additional applications.

## VI. CONCLUSION

This paper presented an AI-based framework for analysing software requirements and classifying them into either functional and non-functional, or AI-generated and Human-written, demonstrating significant improvements in accuracy and efficiency over traditional methods. The use of NLP and machine learning techniques proved effective in automating the requirements analysis process and classification. Future research will focus on enhancing the robustness of the models and exploring their application to a wider range of projects. The potential for AI to revolutionise software engineering practices is vast, and this study provides a solid foundation for further exploration.

## KAYNAKLAR


(1) Ünsal, S., Ayas, M., and Medeni, T. D. "TACIT KNOWLEDGE EXTRACTION FOR SOFTWARE REQUIREMENT SPECIFICATION", International Journal of eBusiness and eGovernment Studies, 3(1):157-168, 2011.

(2) Şahin, T., Demir, Ö., and Yıldız, K. "Doğal Dil İşleme Uygulamaları İçin Türkçe Veri Seti Oluşturulması", International Periodical of Recent Technologies in Applied Engineering, 1(2):51-57, 2019.

(3) Tohma, K. and Kutlu,Y. "Challenges Encountered in Turkish Natural Language Processing Studies", 5(3):204-21, 2020.