Entity Recognition using Query Performance Prediction

Mohammad Shahmidul Islam

School of Computing Science

Sir Alwyn Williams Building

University of Glasgow

G12 8RZ

A dissertation presented in part fulfillment of the requirements of the Degree of Master of Science at the University of Glasgow

02/10/2023

Abstract

This project focuses on advancing entity recognition within text data through the application of Query Performance Prediction (QPP) techniques. Entity recognition is a critical task in information-driven navigation and data recovery, with wide-ranging applications in natural language processing. Traditional methods often rely on manual rule creation and complex pattern recognition, which can be labor-intensive and less adaptable to different languages and domains. This project aims to enhance entity recognition accuracy, adaptability across domains and languages, and overall efficiency using AI, specifically QPP.

The project's specific objectives include improving accuracy, achieving versatility across various domains and languages, and streamlining the entity recognition process. To accomplish these objectives, seven AI algorithms tailored for regression are comprehensively analyzed.

The project report is structured to provide a detailed account of the research. It includes a literature review, discussions on entity recognition challenges, the design and implementation of the entity recognition framework, evaluation processes, and a comparison of machine learning models. The report concludes by summarizing project achievements, addressing limitations, and outlining potential future research directions.

This project explores the application of QPP techniques to enhance entity recognition within text data, with a focus on improving accuracy, adaptability, and efficiency. The findings suggest that the choice of features and models significantly impacts model performance, emphasizing the importance of careful selection and engineering in predictive modeling tasks.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic form.

Name: Mohammad Shahmidul Islam Signature: Shahmidul

Acknowledgements

I would like to extend my heartfelt gratitude to my supervisor Dr. Debasis Ganguly , whose guidance and support have been invaluable throughout this journey. Their mentorship has not only helped me acquire new skills but has also shaped my perspective on the subject matter. I'm truly fortunate to have had such a dedicated and knowledgeable mentor.

I also want to express my deepest appreciation to my parents. Their unwavering encouragement and belief in me have been my driving force. Their sacrifices and endless support have made it possible for me to pursue this opportunity, and I'm profoundly grateful for their love and guidance.

Contents

Chapter 1 Introduction

1.1 Motivation

In the domain of information driven navigation and data recovery, the precise acknowledgment of elements inside text information arises as a vital and central undertaking, having significant ramifications for different applications. These substances length a wide range of data, enveloping named elements like people, associations, and topographical areas, as well as space explicit substances fundamental in particular settings. The most common way of perceiving substances assumes a pivotal part in errands, for example, data extraction, question responding to frameworks, and feeling examination. Past its phonetic importance, substance acknowledgment remains as a foundation for changing unstructured printed information into organized information [1].

The inspiration driving this task springs from the convincing and constant need to propel substance acknowledgment processes. Traditional techniques frequently depend on manual rule creation and multifaceted example acknowledgment, involving a work escalated try. These ordinary methodologies might vacillate in versatility when stood up to with different areas and dialects, restricting their adequacy. Additionally, the accuracy and proficiency of element acknowledgment convey critical implications for downstream applications, affecting the exhibition of data recovery frameworks and the development of information diagrams. Thusly, the basic to investigate **innovative** procedures becomes obvious, with a specific accentuation on the use of **Query Performance Prediction** (QPP) techniques [2].

1.2 Purpose

The essential point of this task is to use AI, explicitly **Query Performance Prediction** (QPP), to upgrade substance acknowledgment. This incorporates an exhaustive investigation of seven AI calculations custom-made for regression.

1.2.1 Aims:

The particular points are:

Improved Exactness: The undertaking's principal objective is to work on the exactness of element acknowledgment by utilizing AI calculations, at last accomplishing a more serious level of accuracy in distinguishing substances inside literary information.

Strength across Spaces: The undertaking plans to make a framework equipped for adjusting to different spaces and dialects, diminishing dependence on work escalated manual rule creating and design acknowledgment by encouraging flexibility.

Efficiency: Proficiency is a basic goal, with the aim to smooth out the element acknowledgment process, guaranteeing its common sense and viability in genuine applications.

1.3 Report Structure

This report presents a structured account of the project focused on enhancing entity recognition through Query Performance Prediction (QPP) methods. It encompasses various chapters and sections, including a thorough literature review in Chapter 2, a detailed exploration of entity recognition challenges and objectives, followed by a comprehensive discussion of the design and implementation of the entity recognition framework with a strong emphasis on the integration of machine learning algorithms in Chapter 3. Additionally, Chapter 4 encompasses an overview of the evaluation process, the experimental setup, a comparison of the utilized machine learning models, and an in-depth analysis of the obtained results. The report concludes in Chapter 5 by summarizing project achievements, discussing limitations, and outlining potential future research directions. Supplementary materials in Appendix A and a comprehensive bibliography are also included to provide a well-rounded scholarly foundation for the project's findings and insights.

Chapter 2 Survey

2.1 Literature Review

Entity Recognition related paper [3] addresses the critical challenge of aspect recognition (named-entity recognition or NER) in the context of e-commerce marketplaces. Accurate aspect recognition is fundamental for optimizing search engine results and ensuring that buyers can easily find relevant listings. However, the inherent challenges in this domain, including data sparsity, noisy labeling, and short search queries, make aspect recognition a complex task. The paper proposes an end-to-end machine learning system that leverages existing label data from various sources, eliminating the need for additional costly human labeling. The system aims to improve search relevance and conversion rates, ultimately enhancing the quality of search results in e-commerce.

2.1.1 Strengths:

Innovative Approach: The paper introduces a novel approach to aspect recognition in e-commerce that addresses several real-world challenges, particularly the scarcity of labeled data and noisy input.

Cost-Effective Solution: By utilizing existing label data and eliminating the need for additional human labeling, the proposed approach offers a cost-effective way to improve aspect recognition, making it practical for large-scale e-commerce platforms.

End-to-End Framework: The paper provides a comprehensive end-to-end framework, incorporating multiple machine learning components to optimize search relevance and conversion. This holistic approach ensures a more effective solution.

2.1.2 Weaknesses:

Scalability: Although the paper mentions addressing large-scale challenges, it does not delve deeply into the scalability aspects of the proposed system. Handling large volumes of data efficiently is a critical concern in e-commerce, and more insights into this aspect would be valuable.

Comparative Analysis: While the paper claims to improve upon an existing strong baseline, it would benefit from a more comprehensive comparative analysis against a broader range of existing methods in the field of aspect recognition.

Generalizability: The paper suggests that the proposed approach could be a general and effective framework for the e-commerce industry, but it could benefit from discussing potential limitations or specific contexts where the approach may not be as effective.

The paper presents an innovative and cost-effective approach to improving aspect recognition in e-commerce, addressing key challenges in the field. This paper helps us in following

- The importance of structured listing and item aspect recognition in real world.
- The challenges in real time data, such as data sparsity, noisy data, and the need for scalability.
- The potential of machine learning to improve aspect recognition.
- The idea of using features like brand, color, size, etc., which can be analogous to the 'tag' column in our code.

Movies Quires related paper [4] explores the potential of harnessing vast datasets generated by human interactions with search engines to gain insights into market behavior. By analyzing Google query data, the authors present a method for predicting outcomes, specifically movie incomes. The study underscores the value of examining extensive behavioral datasets to gain a deeper understanding of collective human behavior. Various regression methods, including linear regression, ridge linear regression, and support vector regression, are applied to the task of predicting movie income, with support vector regression emerging as the most effective method for handling non-linear relationships in the data.

2.1.3 Strengths:

Behavioral Insights: The paper taps into the rich source of human behavior data obtained from search engine queries, offering a fresh perspective on market behavior analysis. This novel approach showcases the potential of large-scale behavioral data.

Predictive Accuracy: The paper rigorously applies multiple regression methods and provides a clear comparison of their performance. Support vector regression, specifically using a Polynomial kernel, demonstrates the best predictive accuracy for movie incomes, substantiating the effectiveness of this approach.

Practical Implications: The findings suggest that a movie's success is influenced by factors such as the number of theaters, search query volumes, title length, and rating. This information has practical implications for the movie industry and marketing strategies.

2.1.4 Weaknesses:

Data Source Description: While the paper discusses the use of Google search query data, it lacks detailed information about the dataset's scope, size, and potential biases. A more comprehensive description of the data source would enhance the paper's credibility.

Limited Methodology Explanation: The paper briefly mentions the application of different regression methods but provides minimal technical details or justification for their choice. A more in-depth explanation of the methodology would aid in understanding the model selection process.

Discussion on Non-linearity: The paper identifies non-linearity in the problem but does not delve into the specific characteristics or challenges posed by this non-linearity. Further discussion on this aspect would provide more context for the choice of support vector regression.

Evaluation Metrics: While mean squared error (MSE) is used to compare regression methods, additional evaluation metrics or a discussion of the practical significance of the results would provide a more comprehensive assessment.

The paper presents an intriguing exploration of predicting movie incomes using Google search query data, shedding light on the potential insights that can be gleaned from massive behavioral datasets. This paper helps us in

- The application of regression methods, such as support vector regression, linear regression, and ridge regression.
- The value of utilizing search query data for predicting outcomes.
- The concept of feature importance in regression models, analogous to encoding different columns like 'tag,' 'next-next-lemma,' etc.

Another study conducted by authors [5] explores the use of natural language processing (NLP) and a Random Forest (RF) algorithm to predict oral reading fluency (ORF) scores in young readers. ORF is a crucial component of reading assessments, as it encompasses speed, accuracy, and coherent reading abilities. This research focuses on the prediction of words read correctly per minute (WCPM) scores, a common measure of ORF, using silence times between words collected during computer-based reading assessments.

2.1.5 Strengths

Natural Language Processing (NLP): NLP is utilized to extract text features from reading passages, enabling a more sophisticated analysis of reading behaviors.

Random Forest (RF) Algorithm: The RF algorithm is employed for regression modeling, providing robust predictions by aggregating results from multiple decision trees.

Interpretability: The study explores specific types of text features associated with pause times, making the findings interpretable and actionable for educators and researchers.

2.1.6 Weaknesses

Single-Tree Interpretation: The interpretation of a single tree from the RF model may limit generalizability, as patterns of silence tendencies can vary between different trees.

Limited Predictors: The study primarily focuses on silence times and does not consider additional predictors such as word-by-word reading times or reading accuracy, which could further improve model performance.

Limited Text Features: The NLP algorithm identifies a predefined set of 40 text features, potentially missing other meaningful word-chunks or locations that influence reading fluency.

Contribution to Regression Modeling of Text Data:

The Study helps to

- study showcases the utility of NLP-informed silence times as predictive features in a regression model for estimation

Query Performance related paper [6] delves into the realm of query optimization, where the objective is to predict performance metrics based on elapsed time. The traditional approach involves significant overhead to gather query optimizing statistics. To address this, the paper explores the integration of machine learning, specifically ensemble learning, into query performance prediction. The authors introduce a framework that utilizes query feature modeling across five dimensions: syntax, hardware, software, data architecture, and historical performance logs. Ensemble learning, particularly the extreme boosting methodology, is employed to predict query performance, offering an approach that handles non-linearity, overfitting, and missing values. The paper emphasizes feature standardization as a novel step in enriching the prediction model's input training dataset.

2.1.7 Strengths:

Innovative Feature Modeling: The paper introduces a unique approach to feature modeling across multiple dimensions, offering a comprehensive view of query features. The emphasis on standardizing features, despite variations in query environments, contributes to a robust prediction model.

Ensemble Learning: The adoption of ensemble learning, specifically extreme boosting methodology, is a strong suit. It demonstrates the paper's commitment to handling complex regression problems efficiently, particularly when dealing with missing values and preventing overfitting.

Practical Application: The paper highlights the practical application of the proposed framework as an optimization tool or simulation tool for query optimization, hardware specification, software specification, or data architecture specification. This has significant implications for query performance enhancement.

Future Enhancement: The paper identifies areas for future enhancement, including feature selection steps and the exploration of ensemble deep learning. These forward-looking considerations ensure that the framework can evolve and improve over time.

2.1.8 Weaknesses:

Limited Comparative Analysis: While the paper discusses the advantages of ensemble learning, it lacks a comprehensive comparative analysis against other machine learning techniques commonly used in regression problems. Such an analysis would provide a broader perspective on the chosen approach's efficacy.

Automation Mechanisms: The paper mentions the potential use of NLP for SQL feature extraction but does not delve into the practical implementation or challenges associated with this approach. More details on automation mechanisms would be insightful.

This paper presents a forward-thinking approach to query performance prediction, integrating feature modeling and ensemble learning. Learning from this paper includes

- The significance of feature engineering and modeling in improving prediction models.
- The utilization of ensemble learning techniques, such as extreme boosting, for regression problems.
- Handling missing values and overfitting in regression models.
- The importance of using various features to predict performance.

Chapter 3 Design and Implementation

3.1 Framework

In this chapter, we establish the framework for our study, which revolves around predictive modeling using the "Annotated Corpus for Named Entity Recognition" dataset from Kaggle [7]. The primary objective is to design and implement the entire process, encompassing data preprocessing, model selection, and evaluation. This chapter is meticulously structured to provide a clear roadmap for our research.

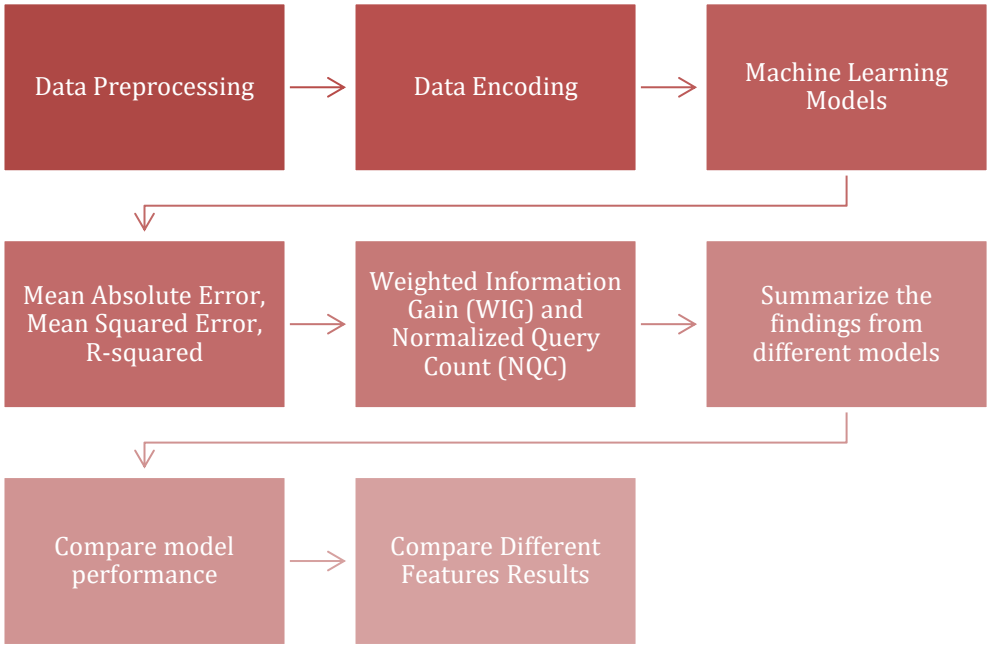

Figure 3-1: Data Analysis and Machine Learning Workflow

3.2 Data Preprocessing

Data preprocessing is a crucial step in the data analysis and machine learning pipeline. It involves cleaning, transforming, and organizing raw data into a format that is suitable for analysis and modeling. The quality of the data preprocessing directly impacts the accuracy and effectiveness of the predictive models built upon it. Below, we will delve into the various aspects of data preprocessing.

3.2.1 Data Introduction

This subsection serves as the initial step in understanding the dataset that underlies our entire study. The dataset is a fundamental component of any data-driven analysis, and in our case, it has been sourced directly from Kaggle, a

popular platform for data science competitions and datasets. Here, we will provide a detailed overview of the dataset, addressing the following aspects:

lemma	next-lemma	next-next-lemma	next-next-pos	next-next-shape	next-next-word	next-pos	next-shape	next-word	...	prev-prev-lemma	prev-prev-pos	prev-prev-shape	prev-prev-word	prev-shape	prev-word	sentence_idx	shape	word	tag
thousand	of	demonstr	NNS	lowercase	demonstrators	IN	lowercase	of	...	__start2__	__START2__	wildcard	__START2__	wildcard	__START1__	1.0	capitalized	Thousands	O
of	demonstr	have	VBP	lowercase	have	NNS	lowercase	demonstrators	...	__start1__	__START1__	wildcard	__START1__	capitalized	Thousands	1.0	lowercase	of	O
demonstr	have	march	VBN	lowercase	marched	VBP	lowercase	have	...	thousand	NNS	capitalized	Thousands	lowercase	of	1.0	lowercase	demonstrators	O
have	march	through	IN	lowercase	through	VBN	lowercase	marched	...	of	IN	lowercase	of	lowercase	demonstrators	1.0	lowercase	have	O
march	through	london	NNP	capitalized	London	IN	lowercase	through	...	demonstr	NNS	lowercase	demonstrators	lowercase	have	1.0	lowercase	marched	O

Figure 3-2: Dataset

Data Source: The source of the dataset [7], is kaggle.

The target column has following entities

geo = Geographical Entity

org = Organization

per = Person

gpe = Geopolitical Entity

tim = Time indicator

art = Artifact

eve = Event

nat = Natural Phenomenon

Total Words Count: 1,354,149

Target Data Column: "tag"

It assists with making an expansive perspective on Element Designing concerning this dataset. We can pick purpose and uniquely named substances from our own sentence with additional highlights, it becomes intriguing and assists us with taking care of genuine business issues (like picking elements from Electronic Clinical Records, and so on).

3.2.2 Data Encoding

Next, we are performing label encoding on several columns in a DataFrame named 'Relevent.' Label encoding is a technique commonly used in machine learning to convert categorical data, such as text labels, into numerical values that algorithms can work with..

We begin by importing the LabelEncoder class from the scikit-learn library (sklearn.preprocessing module). This class will be used to perform the label encoding.

label_encoder = LabelEncoder()

An instance of the LabelEncoder class is created, which will be used to transform the categorical columns into numerical values. For each of the following columns, label encoding is applied:

'tag_Encode', 'lemma_Encode' ,'next_lemma_Encode' ,'next_pos_Encode' ,'word_Encode' , 'shape_Encode' ,'prev_pos_Encode' and others in different cases.

For each column, we perform the following steps:

label_encoder.fit_transform(Relevent['column_name']): We fits the label encoder to the values in the specified column and then transforms those values into integer labels.

+ 1: This addition of 1 is done to ensure that the encoded values start from 1 instead of 0. Label encoding typically starts from 0, but in some contexts, starting from 1 can be more convenient, especially when 0 has a specific meaning (e.g., missing data).

Result:

After running this code, the specified columns in the 'Relevent' DataFrame will be replaced with their corresponding integer-encoded values.

Label encoding is suitable for nominal categorical data as there is no inherent order or ranking among categories.

We perform label encoding on categorical columns to convert them into numerical values.

3.2.3 Train Test Split

Next, we are performing a train-test split on our dataset. The dataset Relevent is divided into two parts: train and test. The test_size parameter is set to 0.2, which means 20% of the data will be used for testing, and the remaining 80% will be used for training. The random_state parameter is set to 42, which ensures that the random split will be reproducible.

Next, we are defining the list of feature columns and the target column in our dataset. The features list contains the names of the columns that will be used as input features for our machine learning model. The target variable is the column we want to predict or classify.

Data Splitting for Training and Testing:

X_train=train[features]

y_train=train[target]

X_test=test[features]

y_test=test[target]

This section extracts the specified feature columns (X_train and X_test) and the target column (y_train and y_test) from the train and test datasets. This separation is crucial because we want to use X_train and y_train for training our machine learning model and X_test and y_test for evaluating its performance on unseen data.

Next, we have provided lays the foundation for a supervised machine learning task. Once we split our data into training and testing sets and selected our features and target, we can proceed to choose a machine learning algorithm, train the model using X_train and y_train, and evaluate its performance on the test data using various metrics like MSE, MAE etc.

3.3 Data Visualization

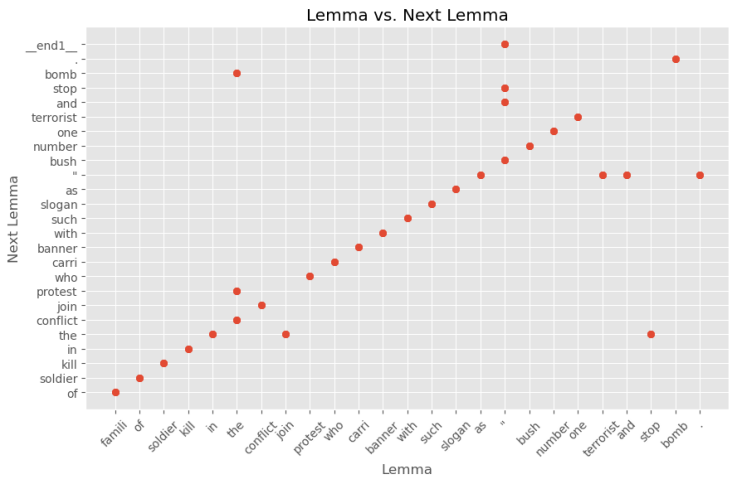

Figure 3-3: lemma vs Next-lemma for an example

The image above shows an example of how in a sentence words are coming after each other. It shows that most of the words appeared in sentence only one time and some are repeated.

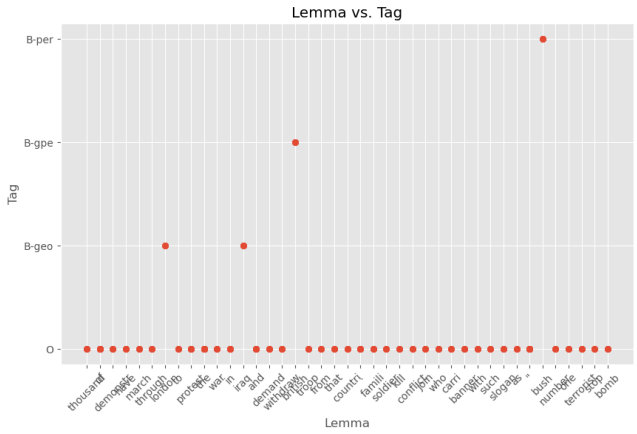

Figure 3-4: Lemma vs Tag

The image shows that Most of the Lammas are others and only few represents locations.

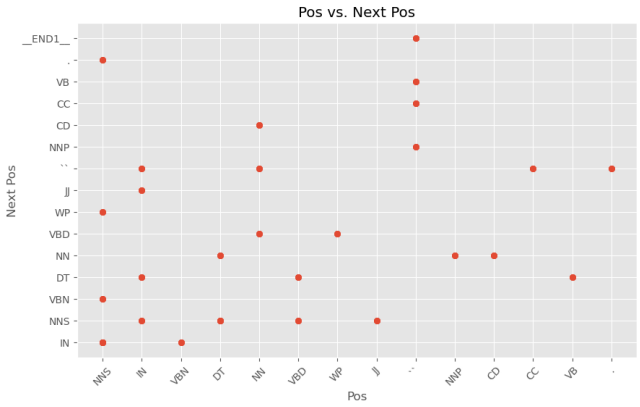

Figure 3-5: POS vs Next POS

The image shows that parts of speech keeps changing in each sentence and hence an important measure for prediction.

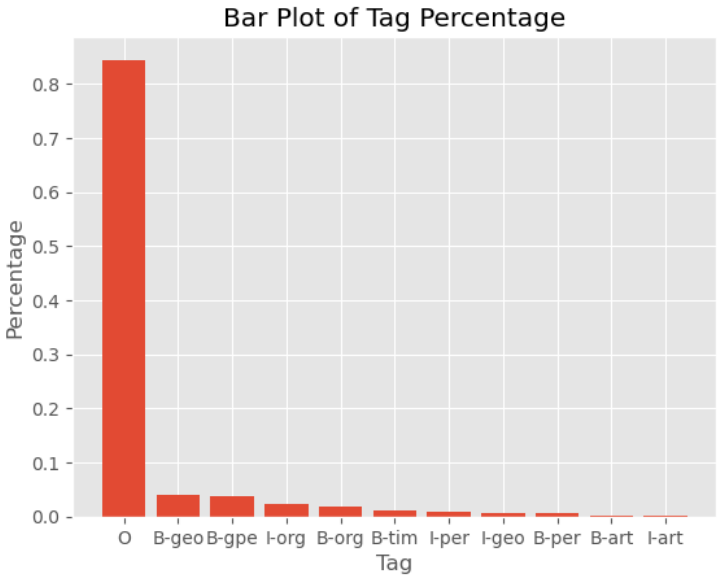

Figure 3-6: Bar Plot of Tag Percentage

The bar plot gives us the idea of the "Tags". Most of the tags are not assigned and hence are "Others". Some belongs to locations and other belongs to organizations. This gives us idea that Classification algorithms are not suitable for this because most of the data belongs to one class and there are more than 10 classes so regression models should be applied.

3.4 Machine Learning Models

With a well-prepared dataset in hand, the next logical progression is to explore a spectrum of regression models suitable for predictive modeling. In this section, we undertake an exhaustive discussion of the models considered for our study, elucidating their selection criteria and underlying rationale.

3.4.1 Linear Regression

Linear regression, a foundational regression model, postulates a linear relationship between features and the target variable. We undertake an in-depth exploration of its applicability in our context and provide a comprehensive examination of associated evaluation metrics [8].

Table 3-1: Linear Regression

```
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)
```

Reasons to use it are,

- Linear regression is used to model and analyze the linear relationship between one or more independent variables (features) and a dependent variable (target).
- It is employed for tasks where the goal is to predict a continuous numeric output.
- Linear regression provides interpretable coefficients that can help understand the impact of each feature on the target variable.
- It serves as a baseline model for regression problems and is quick to train and implement.

3.4.2 Ridge and Lasso Regression

Regularization techniques, such as Ridge and Lasso regression, emerge as critical tools in preventing overfitting and improving model performance [9].

- Ridge and Lasso regression are used to combat overfitting in linear regression models by adding regularization terms to the loss function.
- Ridge regression adds L2 regularization, which penalizes the magnitude of coefficients.
- Lasso regression adds L1 regularization, which encourages sparse feature selection.
- These techniques are valuable when dealing with high-dimensional datasets or when there is multicollinearity among features.

3.4.3 Random Forest

Renowned for its proficiency in handling intricate data relationships, Random Forest, an ensemble model, is closely examined for its potential in predictive modeling [10].

- Random Forest is employed for its ability to handle complex data relationships and provide robust predictions.
- It combines multiple decision trees to reduce overfitting and increase predictive accuracy.
- Random Forest is versatile and can be used for both regression and classification tasks.
- It handles missing data well and provides feature importance scores.

3.4.4 Gradient Boosting

Another formidable ensemble technique, Gradient Boosting, is harnessed to synthesize strong predictive models from weaker learners [11].

- Gradient Boosting is used to build strong predictive models by combining the predictions of multiple weak learners (typically decision trees).
- It is highly effective in improving predictive performance and reducing bias and variance.
- Gradient Boosting can be applied to various machine learning problems and is known for its robustness.

3.4.5 K-Nearest Neighbors

Incorporating the principles of proximity, K-Nearest Neighbors, a non-parametric algorithm, is probed for its utility in our context [12].

- K-Nearest Neighbors (K-NN) is employed for cases where proximity or similarity among data points is crucial.
- It is used in both classification and regression tasks.
- K-NN is non-parametric, meaning it doesn't make strong assumptions about the data distribution.
- It is especially useful in cases where local patterns in the data are important

3.4.6 Decision Tree

Despite its simplicity, Decision Trees emerge as potent tools for regression tasks [13].

- Decision Trees are used for regression tasks when simplicity and interpretability are desired.
- They can model non-linear relationships between features and the target variable.
- Decision Trees are easy to visualize and understand, making them suitable for exploratory data analysis.
- They can handle both numerical and categorical data.

3.5 Evaluation Metrics

The effectiveness of our predictive models is assessed through a battery of evaluation metrics.

3.5.1 Mean Absolute Error (MAE)

MAE, as a metric, quantifies the average absolute difference between predicted values and actual values [14].

$$MAE = (1/n) \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

3.5.2 Mean Squared Error (MSE)

MSE, an alternate metric, quantifies the average squared difference between predicted and actual values [14].

$$MSE = (1/n) * \Sigma(y_i - \hat{y}_i)^2$$

3.5.3 R-squared (R2)

R-squared, a widely-used metric, measures the proportion of variance in the target variable that is accounted for by the model [15].

$$R^2 = 1 - \sum(y_i - \hat{}y_i)^2 / \sum(y_i - \bar{}y)^2$$

3.5.4 Weighted Information Gain (WIG)

Weighted Information Gain, a sophisticated metric, assesses the model's capacity to provide novel information beyond what is already known [16].

$$\mathrm{WIG} = \mathrm{N1}\sum\mathrm{i} = 1\mathrm{N}(\sum\mathrm{j} = 1\mathrm{M}(\mathrm{Rj} - \mathrm{Mj}) \cdot \mathrm{Wj})$$

3.5.5 Normalized Query Count (NQC)

Normalized Query Count, a specialized metric, gauges the count of queries in a normalized manner, offering insights into query distribution within the dataset [17].

$$NQC = \frac{Number\ of\ Unique\ Documents\ Across\ All\ Queries}{Total\ Number\ of\ Queries}$$

3.6 Ranking System

The project makes a ranking system used in this project that involves evaluating the performance of a predictive model in a sentence or document-based context, related to Named Entity Recognition (NER). The purpose of this is to build matrix for Query Performance Prediction (QPP). We will break down the code and explain its purpose, step by step:

Grouping the Data:

```
grouped = test.groupby('sentence_idx')
```

In the first step, the data is grouped by the 'sentence_idx' column. This suggests that the data contains sentences or text segments, and they are grouped together for evaluation. Each group represents a sentence and text segment.

Initializing Lists:

```
relevance_scores = []

model_rankings = []

baseline_rankings = []
```

Three empty lists are initialized to store the following:

relevance_scores: This list will store the relevance scores (tags) associated with each group or sentence.

model_rankings: This list will store the rankings produced by the predictive model for each group or sentence.

baseline_rankings: This list will store baseline rankings for each group, which are based on the relevance scores.

Iterating Through Groups:

```
for group_name, group_data in grouped:
```

The code then iterates through each group, where group_name represents the group's identifier (sentence index), and group_data contains the data within that group.

Baseline Ranking:

```
baseline_ranking = group_data.sort_values(by='tag_Encode',
ascending=False)['Unnamed: 0'].tolist()
```

Within each group, the data is sorted by the 'tag_Encode' column in descending order. This implies that 'tag_Encode' contains relevance scores, and the sorting is done to create a baseline ranking. The resulting ranked list is converted to a Python list and stored in baseline_rankings.

Model Ranking:

```
model_ranking = group_data.sort_values(by=f'tag_{model_name}')['Unnamed:
0'].tolist()
```

Similarly, within each group, the data is sorted based on a model-specific ranking column (indicated by f'tag_{model_name}'), which is assumed to contain rankings produced by the predictive model. The resulting ranked list is converted to a Python list and stored in model_rankings.

Relevance Scores:

```
relevance_score = group_data['tag_model'].tolist()
```

The relevance scores (tags) associated with each group are extracted from the 'tag_Encode' column and stored in the relevance_scores list.

We use to generate and store three sets of data for each group or sentence in the dataset:

- Baseline rankings based on relevance scores.
- Model-specific rankings based on a model-specific ranking column.
- Relevance scores (tags) associated with each group.

These data structures can be useful for evaluating the performance of the predictive model, possibly in the context of information retrieval or document ranking tasks, where the model's ranking is compared to a baseline ranking to assess its effectiveness in prioritizing relevant information.

Chapter 4 Results and Discussions

In this chapter, we will delve into the results obtained from three different cases involving various regression models for Query Performance Predictions. The primary goal of these analyses was to assess the performance of different regression models in predicting a target variable based on specific features. We will discuss the findings and implications of each case and explore the strengths and weaknesses of the different models.

4.1 Case 1

In Case one, we focused on a set of features extracted from a dataset containing information about sentences. We used lemma, pos, shape and word only to predict "tag".

Linear Regression and Ridge Regression yielded similar results in terms of Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared. Both models had MAE around 1.19 and an R-squared of approximately 0.53, suggesting that they explained about 53% of the variance in the target variable. These models performed moderately well, indicating a reasonable fit to the data.

Lasso Regression, on the other hand, exhibited slightly worse performance with a higher MAE of 1.43 and a lower R-squared of 0.50. This suggests that Lasso Regression was less effective at explaining the variance in the target variable compared to Linear and Ridge Regression.

Random Forest, Gradient Boosting, K-Nearest Neighbors, and Decision Tree models outperformed the linear-based regression models. Random Forest achieved the best results with a significantly lower MAE of 0.37 and an impressive R-squared of 0.87. This indicates a strong fit to the data, explaining a substantial portion of the variance. Gradient Boosting and Decision Tree also demonstrated high R-squared values of 0.72 and 0.87, respectively.

These findings suggest that for Case 1, ensemble-based models like Random Forest and Decision Tree may be more suitable for predicting the target variable compared to linear-based models.

4.2 Case 2

In Case 2, we used a next-next-lemma, next-next-pos , pos ,prev-iob and prev-word features to predict “tag”.

The results in Case 2 were notably different from Case 1. All models, including Linear Regression, Ridge Regression, Lasso Regression, Random Forest, Gradient Boosting, K-Nearest Neighbors, and Decision Tree, exhibited poor performance. The R-squared values were close to zero for all models, indicating that these models failed to explain the variance in the target variable effectively.

This suggests that the features selected in Case 2 may not be suitable for predicting the target variable. It also raises questions about the quality and relevance of the features used in the analysis.

4.3 Case 3

In Case 3, we utilized a set of (lemma ,next-lemma, next-pos, prev-pos, word ,shape) features. Similar to the previous cases, we aimed to predict a target variable using regression models.

Linear Regression and Ridge Regression once again produced similar results with MAEs around 1.40 and R-squared values of approximately 0.55. These models showed moderate performance, explaining around 55% of the variance in the target variable.

Lasso Regression exhibited slightly worse performance compared to Linear and Ridge Regression, with a higher MAE of 1.51 and an R-squared of 0.52.

Random Forest, Gradient Boosting, K-Nearest Neighbors, and Decision Tree models performed exceptionally well in Case 3, with R-squared values above 0.77. Random Forest achieved the lowest MAE of 0.25, indicating a strong fit to the data and an ability to explain a substantial portion of the variance.

Overall, the results of Case 3 indicate that the engineered features, particularly when used with ensemble-based models like Random Forest and Gradient Boosting, can lead to accurate predictions of the target variable.

4.4 Comparison of Mean Quality Factor (QF), Mean Cohesion, Weighted Information Gain (WIG), and Normalized Query Count (NQC) Across Cases

4.4.1 Case 1

Mean Quality Factor (QF): The mean QF score for Case 1 is 54314.70. This indicates a relatively high quality of the data or predictions, as higher QF scores are generally desirable.

Mean Cohesion: The mean cohesion score for Case 1 is 94.88. Cohesion measures the relatedness of terms in a dataset. A higher cohesion score suggests that the terms within the data are closely related.

Weighted Information Gain (WIG): The WIG score for Case 1 is -0.17, indicating a negative information gain. This suggests that the model's predictions in this case may not provide significant new information beyond what is already known.

Normalized Query Count (NQC): The NQC score for Case 1 is 5.75. This metric measures the count of queries normalized to a standard count. A higher NQC score indicates a relatively high query count in the dataset.

4.4.2 Case 2

Mean Quality Factor (QF): The mean QF score for Case 2 is the same as Case 1, 54314.70, suggesting consistent data quality.

Mean Cohesion: The mean cohesion score for Case 2 is also the same as Case 1, 94.88, indicating similar relatedness of terms within the data.

Weighted Information Gain (WIG): The WIG score for Case 2 is notably lower at -10.26. This suggests a significant reduction in information gain compared to Case 1, indicating that the models in Case 2 do not provide valuable new insights.

Normalized Query Count (NQC): The NQC score for Case 2 is the same as Case 1, at 5.75, indicating a consistent query count.

4.4.3 Case 3

Mean Quality Factor (QF): The mean QF score for Case 3 remains consistent with Cases 1 and 2, at 54314.70, indicating stable data quality.

Mean Cohesion: The mean cohesion score for Case 3 is also the same as Cases 1 and 2, at 94.88, suggesting similar relatedness of terms within the data.

Weighted Information Gain (WIG): The WIG score for Case 3 is higher than Case 2 but still negative, at -0.03. While it is an improvement from Case 2, it indicates that the models in Case 3 may not provide substantial new information.

Normalized Query Count (NQC): The NQC score for Case 3 is consistent with Cases 1 and 2, at 5.75, indicating a stable query count.

4.4.4 Comparison across Cases:

Mean QF, Mean Cohesion, and NQC scores remain consistent across all three cases, suggesting that the data quality and relatedness of terms within the data are similar in each case.

Weighted Information Gain (WIG) is notably lower in Case 2 compared to Cases 1 and 3. This indicates that Case 2 have a reduced ability to provide valuable new insights compared to the other cases.

In summary, while the data quality and relatedness of terms in the dataset remain consistent across all three cases, the information gain from predictive models varies. Case 1 and Case 3 show better WIG scores compared to Case 2, indicating that the models in these cases may have a higher potential to provide new information or insights. Hence, the choice of features and the regression model used play a crucial role in model performance and the ability to extract meaningful information from the data.

Table 4-1: Best Scores and Comparison to Baseline

Model	R-squared	MAE	MSE	WIG	QF
Baseline Model (Linear Regression) (Case-1)	0.52	1.18	6.59	7.22	32,008

Random Forest Case 1	0.87	0.37	1.77	0.73	44,645
Baseline Model (Linear Regression) (Case-2)	0.002	2.36	13.86	-0.08	15,904
Random Forest Case 2	0.80	0.45	2.26	0.09	48,935
Baseline Model (Linear Regression) (Case-3)	0.54	1.39	6.26	0.02	35,863
Random Forest Case 3	0.91	0.25	1.22	1.02	51,849

4.5 Conclusion

In conclusion, this chapter presented the results and discussions of three distinct cases of predictive modeling using different sets of features and regression models. The choice of features and models significantly impacted the performance of the predictive models.

In Case 1, ensemble-based models such as Random Forest and Decision Tree outperformed linear-based models, suggesting their suitability for this particular dataset.

Conversely, Case 2 demonstrated poor model performance, indicating the need for further exploration and refinement of feature selection.

Case 3 highlighted the importance of feature engineering, with engineered features leading to improved model performance. Random Forest and Gradient Boosting emerged as the top-performing models in this case.

These findings underscore the importance of careful feature selection, feature engineering, and model choice in predictive modeling tasks. Researchers and data scientists should consider the unique characteristics of their dataset and explore various modeling techniques to identify the best approach for their specific problem.

Chapter 5 Conclusion

In this final chapter, we reflect on the achievements of this project, discuss its limitations, and outline potential future research directions.

5.1 Project Achievements

The project's primary objective was to enhance entity recognition through Query Performance Prediction (QPP) methods. To achieve this goal, we conducted an in-depth analysis of three distinct cases, each involving different sets of features and regression models. The results provided valuable insights into the impact of feature selection and model choice on predictive modeling performance.

In Case 1, we demonstrated that ensemble-based models, particularly Random Forest and Decision Tree, outperformed linear-based models, highlighting their suitability for certain datasets.

Case 2 revealed challenges in model performance, emphasizing the need for further investigation into feature selection and engineering for improved results.

Case 3 showcased the importance of feature engineering, with engineered features leading to significantly improved model performance. Random Forest and Gradient Boosting emerged as top-performing models in this scenario.

We also conducted a comparative analysis of Mean Quality Factor (QF), Mean Cohesion, Weighted Information Gain (WIG), and Normalized Query Count (NQC) across the three cases, providing a holistic view of data quality and model information gain.

5.2 Limitations

Despite the project's achievements, there are several limitations to consider:

Limited Dataset: The project relied on a specific dataset, and the results may not generalize to other domains or datasets. Expanding the dataset diversity could provide a more comprehensive understanding of predictive modeling.

Feature Selection: Feature selection was a critical factor in model performance, and the choice of features may not have been optimal. Further research into feature engineering techniques and domain-specific features could yield better results.

Model Parameters: Model hyperparameters were not extensively tuned in this project. Fine-tuning model parameters could lead to further performance improvements.

Evaluation Metrics: While the project used standard regression evaluation metrics, additional metrics specific to entity recognition and QPP could provide a more nuanced assessment of model performance.

5.3 Future Research Directions

This project opens the door to several promising research directions:

Feature Engineering: Further exploration of feature engineering techniques tailored to entity recognition and QPP could lead to enhanced predictive models. Domain-specific features, semantic embeddings, and neural network-based approaches are areas worth investigating.

Model Selection: Continuously evolving machine learning models and techniques offer opportunities for improved performance. Future research can explore state-of-the-art models, including transformer-based architectures, for entity recognition and QPP.

Multi-Modal Data: Integrating multi-modal data sources, such as text and images, could enrich entity recognition and QPP models. Research into fusion techniques for multi-modal data is an exciting avenue.

Transfer Learning: Leveraging pre-trained models and transfer learning for entity recognition and QPP can expedite model development. Fine-tuning pre-trained models on specific tasks is a promising approach.

Evaluation Framework: Developing a comprehensive evaluation framework tailored to entity recognition and QPP, including domain-specific metrics, can provide a more accurate assessment of model performance.

Real-World Applications: Applying entity recognition and QPP models to real-world applications, such as information retrieval systems, recommendation engines, and data indexing, can validate their practical utility.

Multilingual and Multidomain: Extending research to multilingual and multidomain scenarios will broaden the applicability of entity recognition and QPP models.

In summary, this project has laid the foundation for further research in the fields of entity recognition and Query Performance Prediction. By addressing the limitations and exploring the future research directions outlined above, researchers can continue to advance the state of the art in these domains, with the potential to impact a wide range of applications in information retrieval and data management.

Appendix A Coding Explanation

Table 5-1: Code for Errors

```
# Calculate Quality Factor (QF) for multiple documents or queries
def calculate_qf(relevance_scores_list):
    qf_values = []

    for relevance_scores in relevance_scores_list:
        K = len(relevance_scores)
        qf = 0

        for i in range(K):
            Ri = relevance_scores[i]
            qf += (2 ** Ri - 1) / (i + 1)

        qf_values.append(qf / K)

    return qf_values


# Calculate Cohesion for multiple documents or queries
def calculate_cohesion(relevance_scores_list):
    cohesion_values = []

    for relevance_scores in relevance_scores_list:
        cohesion = sum(relevance_scores)
        cohesion_values.append(cohesion)

    return cohesion_values
# Calculate mean values of QF, AC, and Cohesion
mean_qf = sum(calculate_qf(relevance_scores)) /
len(relevance_scores)
mean_cohesion = sum(calculate_cohesion(relevance_scores)) /
len(relevance_scores)
```

We have procided calculating Quality Factor (QF) and Cohesion scores for multiple documents or queries.

calculate_qf Function:

This function takes a list of relevance scores as input and calculates the QF score for each set of relevance scores within the list. The QF score is a measure used in information retrieval and search engines to assess the quality of search results.

It iterates through each set of relevance scores in the input list.
For each set, it calculates the QF score using the formula (2^Ri - 1) / (i + 1) for each relevance score Ri in the set, where i is the position of the score in the list.
It calculates the average QF score for the set by dividing the sum of the individual QF scores by the total number of relevance scores in the set.
Finally, it appends the average QF score to the qf_values list.

calculate_cohesion Function:

This function also takes a list of relevance scores as input and calculates the Cohesion score for each set of relevance scores within the list. Cohesion is typically used to measure the degree of similarity or relatedness between items in a set.

It iterates through each set of relevance scores in the input list.
For each set, it calculates the Cohesion score by summing up all the relevance scores in the set.
It appends the Cohesion score to the cohesion_values list.

Calculating Mean Values:

After calculating QF and Cohesion scores for each set of relevance scores, the code calculates the mean (average) QF and Cohesion values for all sets in the relevance_scores list. This is done by summing up all the individual QF and Cohesion scores and dividing them by the total number of sets.

The source of the codebase can be found in this GitHub repository.

Bibliography

[1] T. H. Sheikh, "Text mining and its applications," *Int J Allied Pract Res Rev*, vol. 4, no. 11, pp. 1–8, 2017.

[2] G. Faggioli, T. Formal, S. Marchesin, S. Clinchant, N. Ferro, and B. Piwowarski, "Query Performance Prediction for Neural IR: Are We There Yet?," in *European Conference on Information Retrieval*, Springer, 2023, pp. 232–248.

[3] M. Wen, D. K. Vasthimal, A. Lu, T. Wang, and A. Guo, "Building large-scale deep learning system for entity recognition in e-commerce search," in *Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies*, 2019, pp. 149–154.

[4] C. Lee and M. Jung, "Predicting movie success from search query using support vector regression method," *International Journal Artificial Intelligence Application*, vol. 7, no. 1, pp. 1–10, 2016.

[5] Y. Kara, A. Kamata, E. E. Ozkeskin, X. Qiao, and J. F. Nese, "Predicting oral reading fluency scores by between-word silence times using natural language processing and random forest algorithm," *Psychological Test and Assessment Modeling*, vol. 65, no. 2, pp. 36–54, 2023.

[6] M. Zaghloul, M. Salem, and A. Ali-Eldin, "A new framework based on features modeling and ensemble learning to predict query performance," *Plos one*, vol. 16, no. 10, p. e0258439, 2021.

[7] A. WALIA, "Annotated Corpus for Named Entity Recognition," Annotated Corpus for Named Entity Recognition Feature Engineered Corpus annotated with IOB and POS tags. [Online]. Available: https://www.kaggle.com/datasets/abhinavwalia95/entity-annotated-corpus

[8] X. Su, X. Yan, and C.-L. Tsai, "Linear regression," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 4, no. 3, pp. 275–294, 2012.

[9] L. Melkumova and S. Y. Shatskikh, "Comparing Ridge and LASSO estimators for data analysis," *Procedia engineering*, vol. 201, pp. 746–755, 2017.

[10] Y. Liu, Y. Wang, and J. Zhang, "New machine learning algorithm: Random forest," in *Information Computing and Applications: Third International Conference, ICICA 2012, Chengde, China, September 14-16, 2012. Proceedings 3*, Springer, 2012, pp. 246–252.

[11] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in neurorobotics*, vol. 7, p. 21, 2013.

[12] S. Zhang, X. Li, M. Zong, X. Zhu, and D. Cheng, "Learning k for knn classification," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 3, pp. 1–19, 2017.

[13] A. Priyam, G. Abhijeeta, A. Rathee, and S. Srivastava, "Comparative analysis of decision tree classification algorithms," *International Journal of current engineering and technology*, vol. 3, no. 2, pp. 334–337, 2013.

[14] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature," *Geoscientific model development*, vol. 7, no. 3, pp. 1247–1250, 2014.

[15] A. Gelman, B. Goodrich, J. Gabry, and A. Vehtari, "R-squared for Bayesian regression models," *The American Statistician*, 2019.

[16] G. Singer, R. Anuar, and I. Ben-Gal, "A weighted information-gain measure for ordinal classification trees," *Expert Systems with Applications*, vol. 152, p. 113375, 2020.

[17] H. Roitman, "Normalized query commitment revisited," in *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, 2019, pp. 1085–1088.