Universidad Politécnica
de Madrid

Escuela Técnica Superior de
Ingenieros Informáticos

Master in Data Science

Information Retrieval: ML Ranking Practical Work

Rodrigo Castañón Martínez

Javier Arteaga Puell

Dakota James Mellish

Contents

1. Introduction

The objective of this project is to provide relevant results to medical test search queries. This is a challenging task, as medical tests often have very similar and overlapping names, and some countries have their own specific units as well as names. Nevertheless our group remains up for the challenge, and to accomplish this task, we utilize a logistic regression classifier along with pointwise-comparison to score various medical tests and measurements that appear in the dataset.

1.2 Tools

The tools and technologies used in this project include Python, Jupyter Notebook as well as python libraries such as Scikit-learn, NLKT, Pandas and Numpy.

2. Data Overview

2.1 Data description

In this section, we present an overview of the dataset used in this project, highlighting its structure, variables, and format. The data consists of various medical tests and measurements provided by LOINC, whose full name is Logical Observation Identifiers Names and Codes. It is the global standard used in the medical field worldwide.

Table 2.1: Structure of the dataset

Variable	Description	Type	Format
LOINC Num	Unique identifier for medical measurement Example: 1988-5	Categorial	Numeric ID
Long Common Name	Name of medical measurement along with units Example: C reactive protein [Mass/volume] in Serum or Plasma	Categorical	Text
Component	Identifies what is being measured Example: C reactive protein	Categorical	Text
System	Identifies how the item is being measured (through which source) Example: Blood, Serum	Categorical	Text
Property	Identifies the type of measurement or characteristic being observed for a given test or observation Example: Mass concentration, Temp	Categorical	Text

2.2 Dataset expansion

It is worth mentioning that the dataset provided in class only included 70 rows. As part of refining the methodology and expanding the potential results set, the entire LOINC database has been procured. It was obtained legally and downloaded through the LOINC website. The entire dataset comprises 104 thousand records.

2.2 Search Queries

The original search terms provided are “glucose in blood”, “bilirubin in plasma”, and “white blood cells count”. We have added additional terms, “insulin in blood”, “Cancer Ag 125 Pleural Fluid”, “MRA Thigh vessels contrast” and “deoxycortisol in serum” in an effort to add rigor to our project and further demonstrate the capabilities of the logistic regression classifier + pointwise comparison approach.

3. Data Preprocessing

Below we outline each step involved in the preprocessing step of our project

3.1 Initial Relevance Filter

Because of the size of the full LOINC dataset (over 100K records), it was necessary to perform an initial search of the dataset and retrieve a mixture of relevant + non-relevant records. We utilized the query as the foundation for the search. We used the natural language processing NLTK library in Python to filter out irrelevant stop words that have very little value such as “in” “by” and “to”. Each word in a given query is checked for a match in the field *loinc_common_name,* and if a match is found, a score equal to the length of the word. We select all terms having a score of 6 or higher. However, it is important to include non-relevant terms as well for our model to discern, Therefore, we perform a random selection of records equivalent to the quantity of total rows having a relevance score of 6 or higher, keeping the dataset balanced. We now have a more compact dataset to utilize for each query. We can also use the score as a variable for the model.

3.2 Second Relevance Filter

We can also apply a “softer” filter which checks the previous results set for any word matches of the query with respect to the field *loinc_common_name*. The result is a column called *relevance* which returns either a binary indicator of either 1 or 0. This flag will be used as a label to assess relevance of the model.

3.3 Pairwise Comparison Using Cosine Similarity

With the previous results, a pairwise comparison using cosine similarity as well as term frequency inverse document is utilized. We combine the fields *loinc_common_name* with *component* to check for additional terms of relevance included in the component field. The query terms’ frequency are checked against the combined text and this forms the matrix necessary to calculate the cosine similarity measure **W**. We then calculate the cosine similarity using the combined text matrix **W** along with the query term vector to get a score . Results are then sorted from highest to lowest (higher being more relevant). The score can then act as a label for the model. We transform the column to a binary variable using a threshold of .5.

3.4 Min Max Scaling

With the previous results, it is then necessary to transform the variable query score as it has a high degree of variation. We use scikit-learn to implement this transformation.

4. Model Methodology

Below we discuss the model used and the variables to be included

4.1 Model Chosen

We opted to use the logistic regression classifier. This model is a proven workhorse for classification tasks, is quite fast in runtime and performs well with very few features.

4.2 Variables Used

Below we discuss the model used and the variables to be included

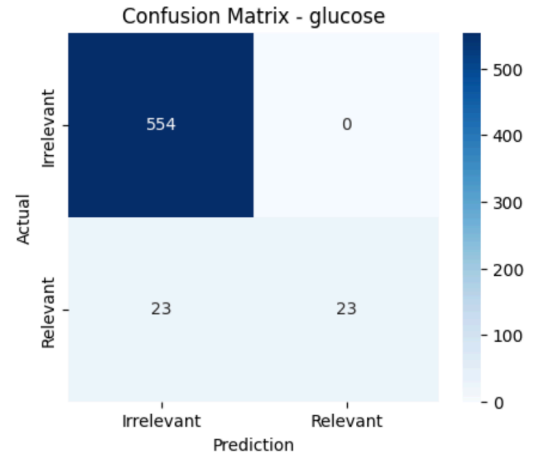
Table 4.1: Variables Included

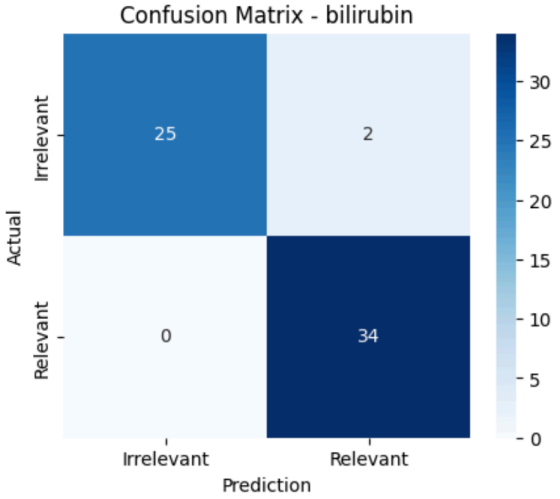
Variable	**Description**	**Type**
Query score	Total character length of term matches in query with loinc_common_name	Float
Relevance	Binary flag that checks for any terms matching query with loinc_common_name	Integer
Score_label	Cosine Similarity score using *loinc_common_name* and *component* fields along with query terms. Threshold of .5 is used to establish 0-1 label	Integer

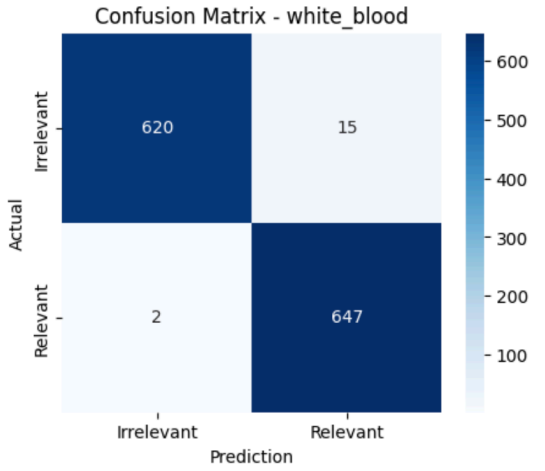
5. Evaluation of Approach

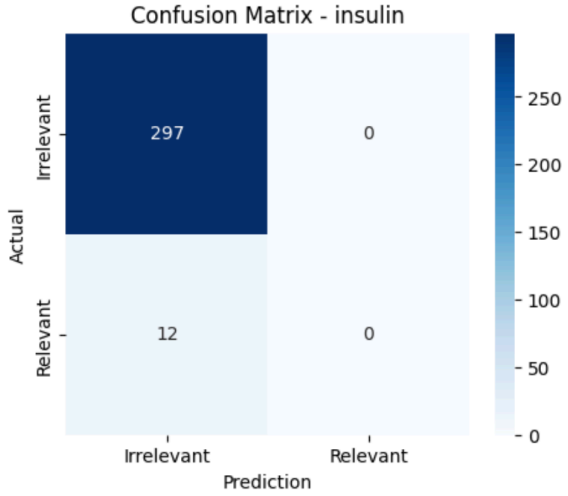
In order to determine if our approach is successful, we will consider each query term on a case by case basis and utilize a confusion matrix, ROC curve as well as consider the probability distribution of relevance based on relevance label. We will also examine the individual accuracy, precision, recall and F1 score of the model for each query.

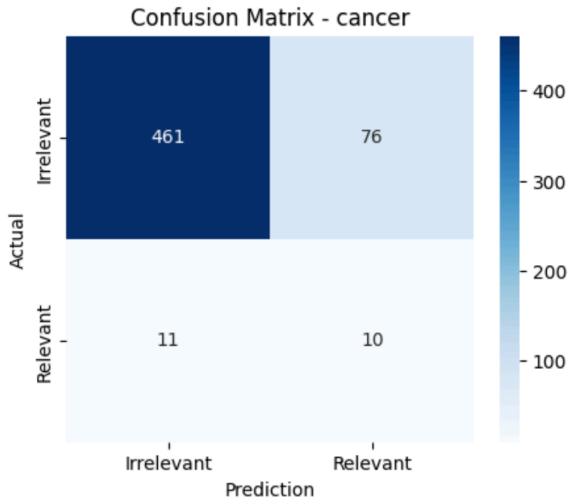
Figure 5.1: Confusion Matrices

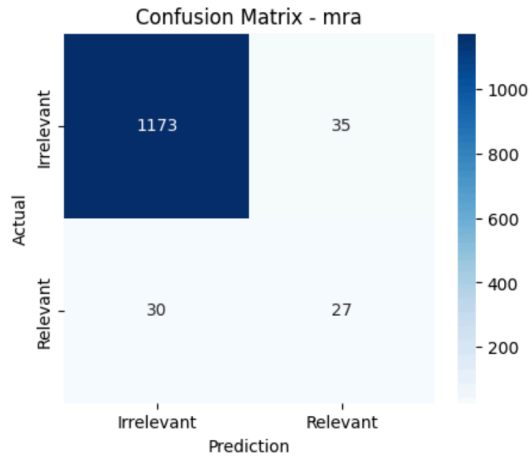

Confusion Matrix - glucose
554
0
23
23
Irrelevant
Relevant
Actual
Prediction
500
400
300
200
100
0


Confusion Matrix - bilirubin
25
2
0
34
Irrelevant
Relevant
Actual
Prediction
30
25
20
15
10
5
0


Confusion Matrix - white_blood
620
15
2
647
Irrelevant
Relevant
Actual
Prediction
600
500
400
300
200
100


Confusion Matrix - insulin
297
0
12
0
Irrelevant
Relevant
Actual
Prediction
250
200
150
100
50
0


Confusion Matrix - cancer
461
76
11
10
Irrelevant
Relevant
Actual
Prediction
400
300
200
100


Confusion Matrix - mra
1173
35
30
27
Irrelevant
Relevant
Actual
Prediction
1000
800
600
400
200

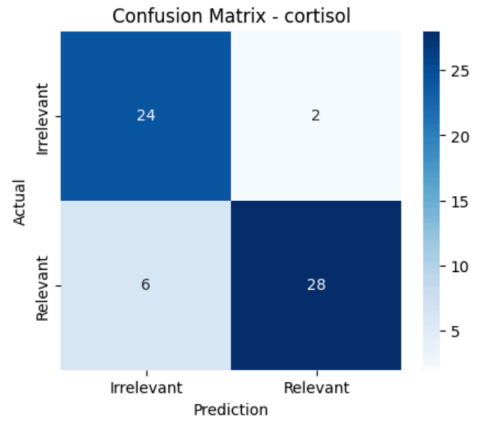
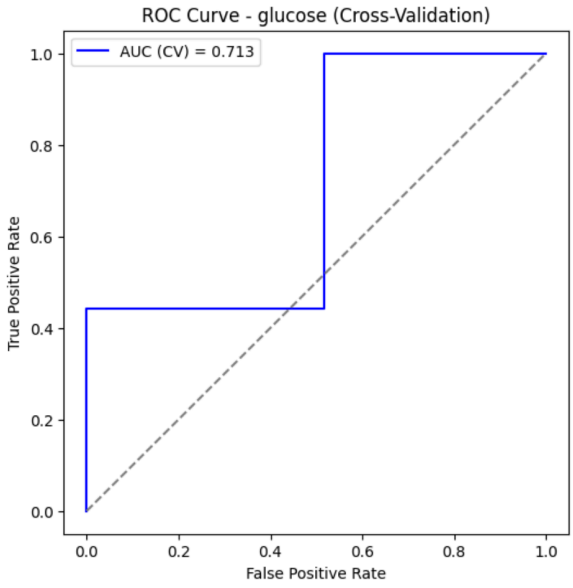
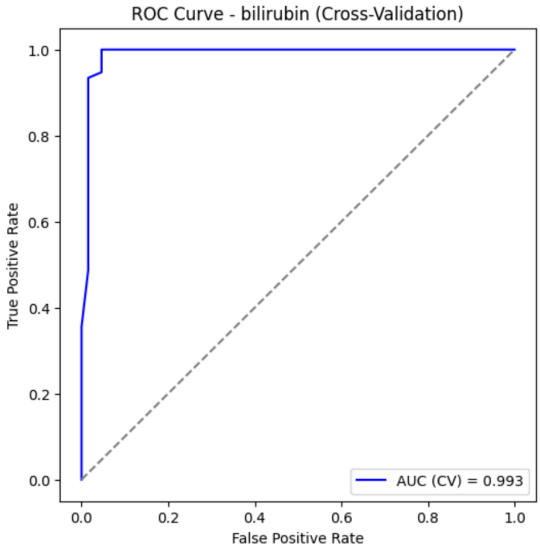
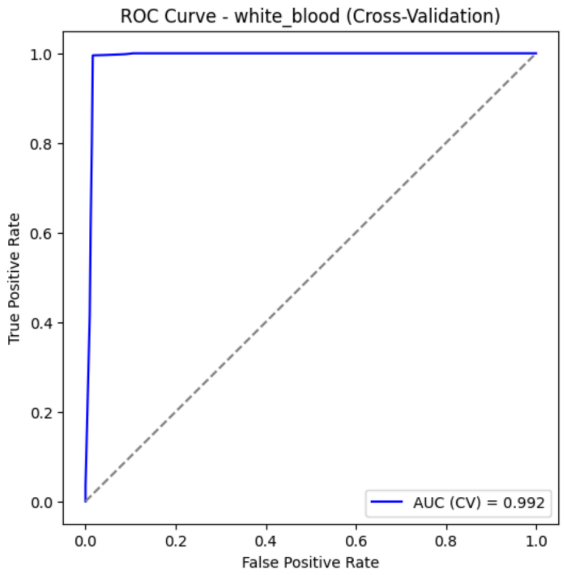
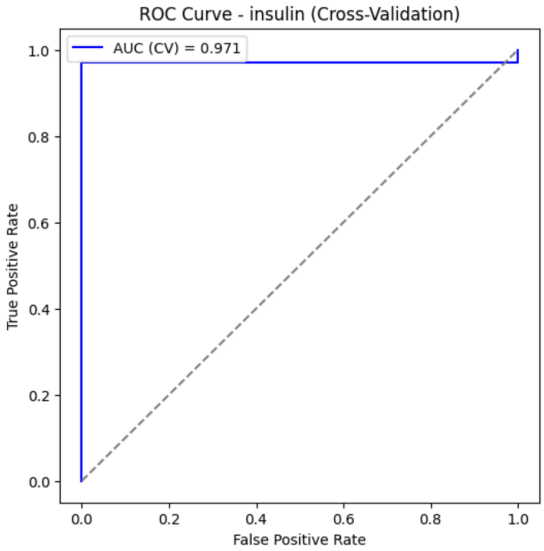


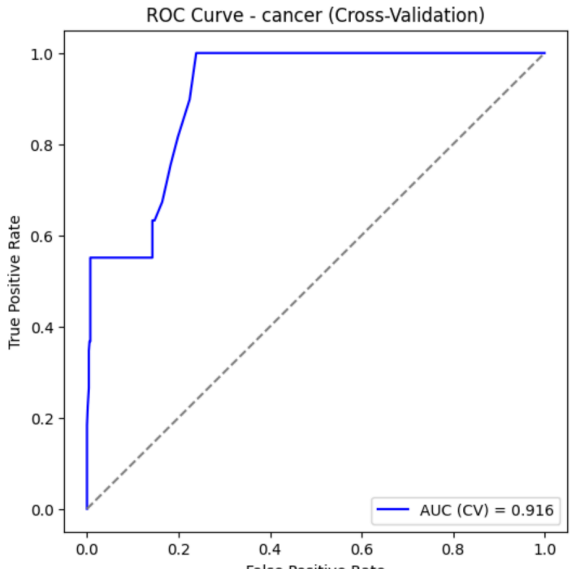
Figure 5.2: ROC Curves

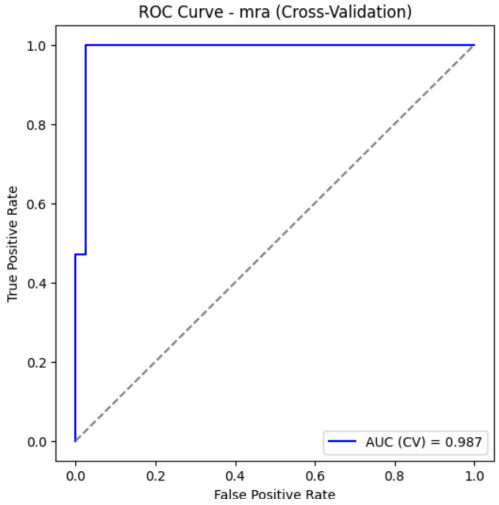





ROC Curve - white_blood (Cross-Validation)
True Positive Rate
False Positive Rate
AUC (CV) = 0.992


ROC Curve - insulin (Cross-Validation)
True Positive Rate
False Positive Rate
AUC (CV) = 0.971


ROC Curve - cancer (Cross-Validation)
True Positive Rate
AUC (CV) = 0.916


ROC Curve - mra (Cross-Validation)
True Positive Rate
False Positive Rate
AUC (CV) = 0.987

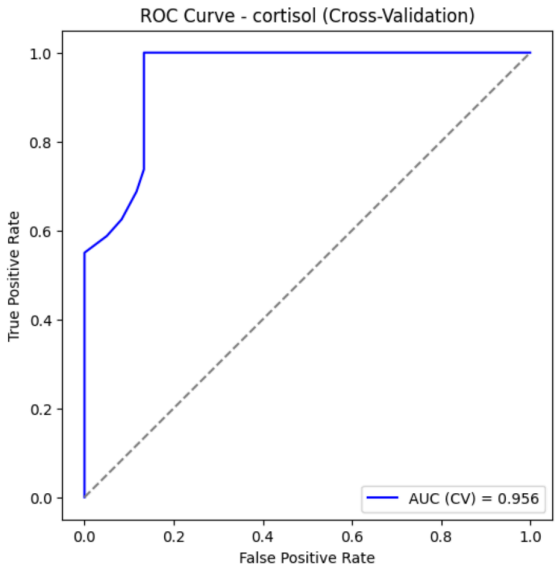
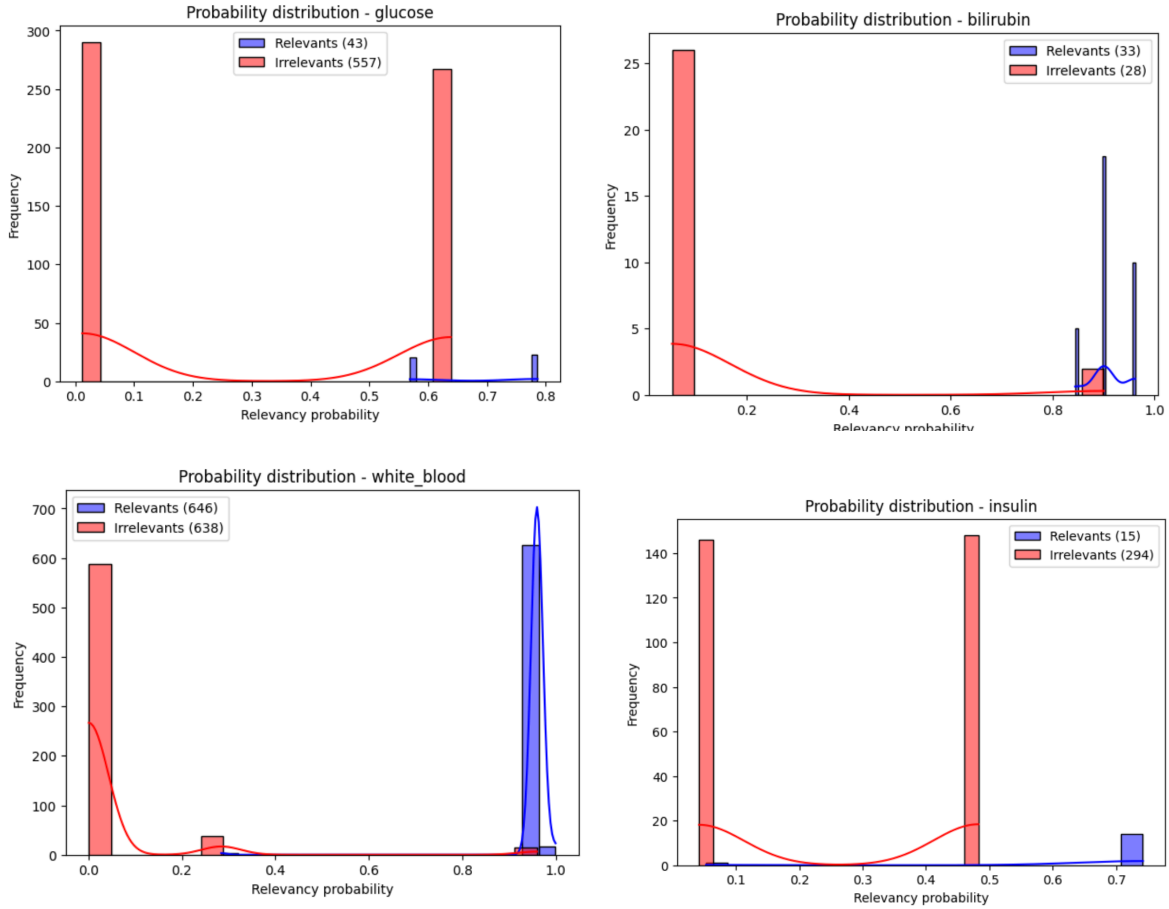
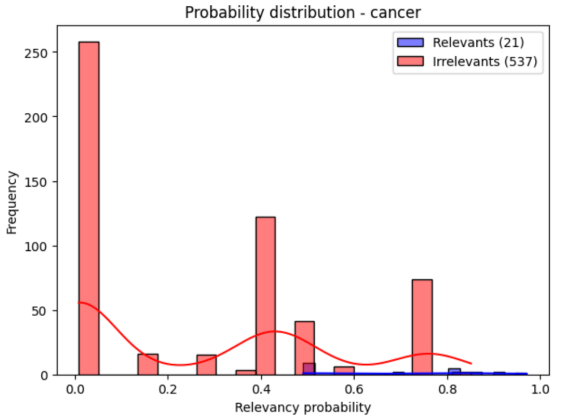
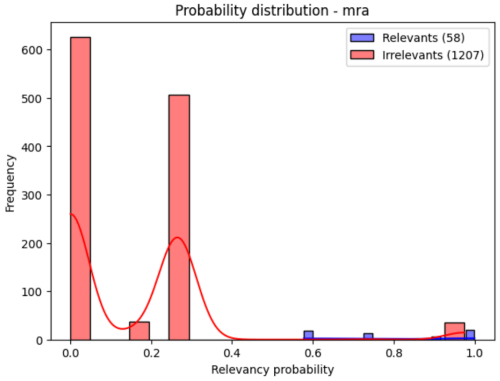


Figure 5.3: Probability Distributions







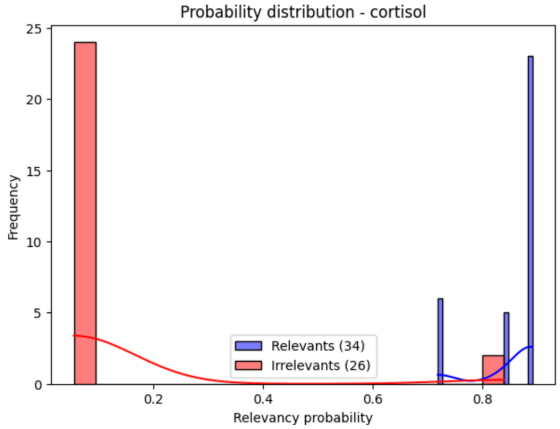


Table 5.1: Classifier Results by Query

Query	Accuracy	Precision	Recall	F1 Score
Glucose in Blood	0.9672	0.9429	1.0000	0.9706
Bilirubin in Plasma	0.9860	0.9787	0.9938	0.9862
White blood cell count	0.9860	0.9787	0.9938	0.9862
Insulin in Blood	0.9579	0.0000	0.0000	0.0000
Cancer AG 125 Pleural Fluid	0.8441	0.1163	0.4762	0.1869

MRA Thigh vessels contrast	0.9502	0.4426	0.4821	0.4615
Deoxycortisol in serum	0.8667	0.9333	0.8235	0.8750

From initial inspection of the confusion matrices, it is worth noting that in 4 out of 7 searches, they are skewed towards non-relevant matches. However, we can see the classifier performs well in the area of specificity, as the model correctly identifies the vast majority of these items as non-relevant. When considering the ROC curves, it is shown that the model delivers near perfect classification performance for most of the queries, but this is largely explained by many "irrelevant" records being correctly classified. For the worst performing query (insulin), the reason the model performs poorly is due to the large concentration of terms which are irrelevant yet score high in the relevancy probability graph. This is likely due to the high amount of potential matches for glucose found in the LOINC dataset. When looking at the table, it is observed that some terms such as bilirubin and glucose offer near-perfect classification, while some terms such as cancer and mra offer a mixed bag between high accuracy but low precision. Overall, model performance is highly subjective to the search term.

6. Future Considerations and Conclusions

Overall, the logistic classifier approach when combined with pointwise comparison is an effective tool when trying to provide relevant results to various query searches. It provides a fast runtime along with a relatively straightforward implementation. There is certainly room for improvement, as more queries could be considered, and the individual results depend a lot on the input provided.

A potential enhancement could be the creation of an algorithm that parses potential queries to test based on the *loinc_common_name* field to provide more robust and general results. Another improvement is that other fields such as *system* and *property* could also be incorporated into the cosine similarity measure. These features could provide additional detail that isn't captured with just *loinc_common_name* and *component*. Another possible improvement is an experimentation of the cosine similarity threshold to determine what is and isn't relevant, as this may help terms such as "insulin in blood", where the model was unable to find relevant results.

Finally, it seems plausible that additional variables could be created for the model based on the presence of partial "sub-word" matches. For example, what happens if a word is mistyped? The classifier will not count this as a match, despite matching all but 1-2 characters of a given word. Overall however, we are quite pleased with results such as the AUC for the different queries, which show a robust model that is capable of providing only the most relevant search results from the LOINC dataset.