**CASE STUDY**

**NATURAL LANGUAGE PROCESSING**

**Theme**- **NLP in Education**

**Topic- Classification of exam questions using NLP**

**Abstract**

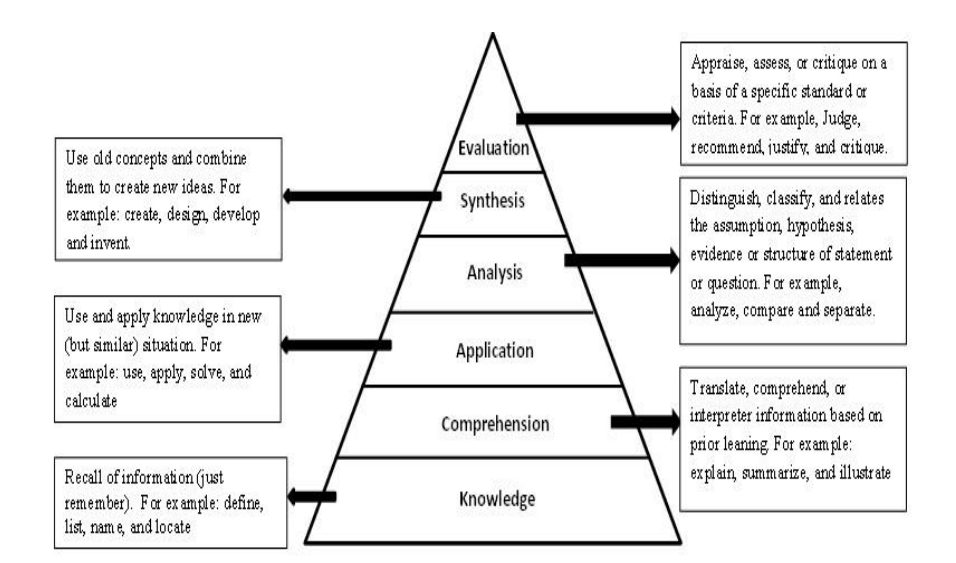
This study delves into the world of quality assurance inside educational assessment, specifically focusing at the automated classification of examination questions primarily based totally on Bloom's Taxonomy cognitive levels (BCLs). Given the task confronted by means of instructional organisations, including accreditation bodies, in manually checking a big variety of questions for correctness consistent with BCLs, the want for automated class will become paramount. This studies ambitions to check and examine numerous gadget getting to know strategies, which include Naïve Bayes, Support Vector Machine (SVM), Logistic Regression, and Decision Trees. These strategies are carried out to categorise examination questions primarily based totally on Bloom's Taxonomy cognitive levels, utilising linguistically-prompted functions like bag of words, a part of speech (POS), and n-grams. A dataset comprising six hundred examination questions for an English language path serves as the muse for this examine. The findings monitor that the mixture of gadget getting to know strategies with linguistically-prompted functions yields fine outcomes withinside the automated class of questions primarily based totally on Bloom's Taxonomy.

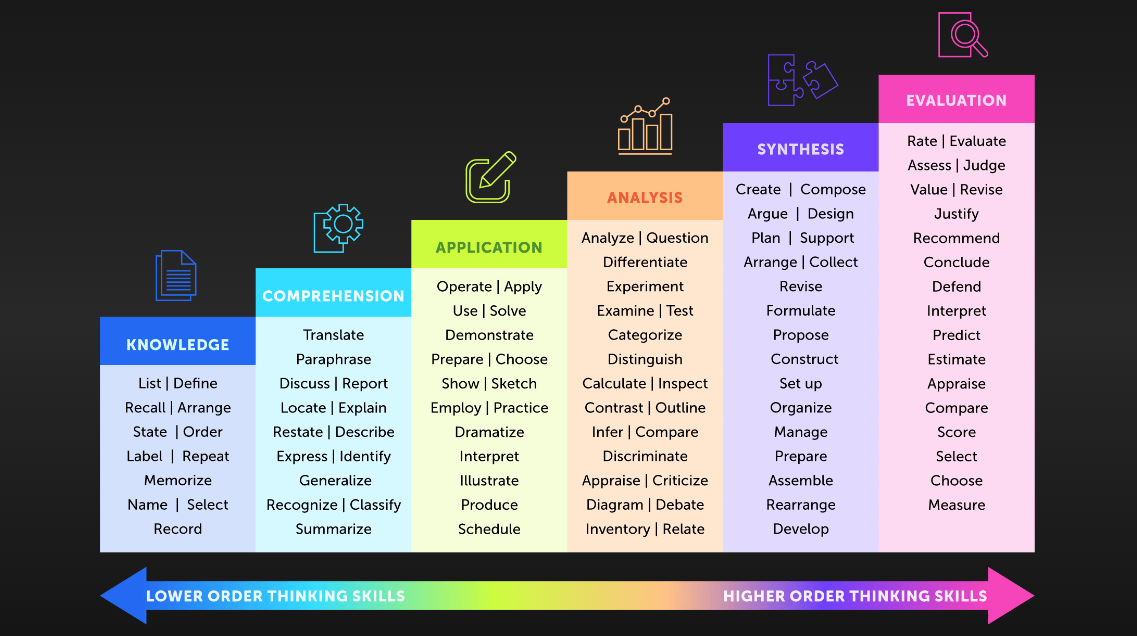
**Introduction**

Neil Postman`s announcement that "All our information outcomes from questions" underscores the pivotal function of thinking withinside the gaining knowledge of process. Questions force understanding, coaching effectiveness, and assessment of gaining knowledge of outcomes. Educational establishments universally hire assessments to gauge college students' comprehension degrees, with the complexity of questions gambling an essential function. Higher-order wondering is fostered whilst college students interact with questions that require deeper cognitive processes. Bloom's Taxonomy, advanced with the aid of Benjamin Bloom and a set of specialists, gives a framework with six cognitive degrees, starting from primary recollect to complicated assessment. This taxonomy presents educators with a manual to craft questions that focus on exclusive cognitive competencies and make sure mastery of subjects. However, manually classifying examination questions primarily based totally on Bloom's Taxonomy stays an assignment, in particular because of educators' various information of the taxonomy and the sheer quantity of questions. This assignment extends to accreditation our bodies and fine firms that want to confirm query type accuracy in line with BCLs. Thus, the need for computerised type of questions primarily based totally on Bloom's Taxonomy turns evident.

Assessing the cognitive complexity of examination questions is an important venture for educators to make sure the effectiveness and appropriateness of assessments. Bloom's Taxonomy presents a framework for categorising instructional goals into degrees of cognitive complexity, from primary recollect to higher-order wondering competencies. However, manually categorising a big wide variety of examination questions in line with Bloom's Taxonomy may be time-ingesting and subjective. To cope with this assignment, we advocate an answer to the usage of Natural Language Processing (NLP) strategies incorporated right into a Django internet application.

The Six Cognitive Levels of Bloom's Taxonomy





**Problem Statement**

The class of examination questions primarily based totally on Bloom's Taxonomy cognitive degrees poses a tremendous project for educators and first-rate guarantee bodies. Manual checking of a big quantity of questions is impractical and time-consuming. Educational organisations, especially accreditation bodies, require a dependable approach to make certain the correctness of the query class consistent with BCLs. Therefore, the implementation of computerised class structures will become vital to streamline this procedure.

The hassle handy is the labour-in depth and subjective nature of categorising examination questions into Bloom's Taxonomy degrees. Educators frequently spend tremendous effort and time manually assigning those degrees, mainly to inconsistencies and capacity biases. Additionally, the sheer quantity of questions in big datasets makes guide categorization impractical. This undertaking aims to automate and standardise the procedure of categorising examination questions into Bloom's Taxonomy degrees the usage of NLP, offering educators with a greater green and goal device for assessment.

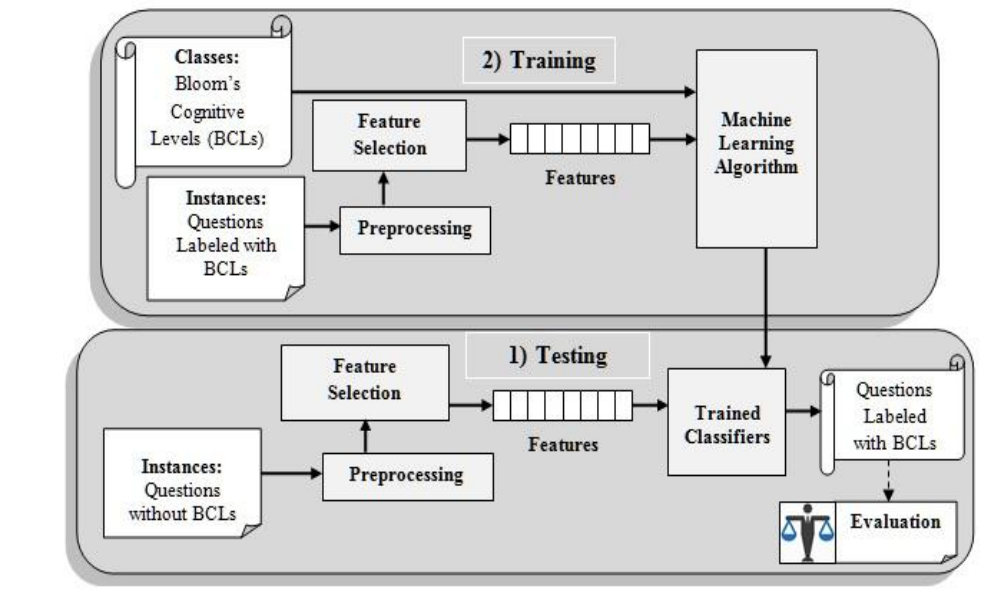
**Solution**

This studies units out to deal with the mission of query type primarily based totally on Bloom's Taxonomy cognitive stages thru the utilisation of gadget gaining knowledge of strategies and linguistically-stimulated functions. By trying out and evaluating algorithms which include Naïve Bayes, Support Vector Machine (SVM), Logistic Regression, and Decision Trees, along functions like bag of words, a part of speech (POS), and n-grams, this have a look at pursuits to offer insights into powerful automated type. The selected dataset includes six hundred examination questions from an English language course, imparting a numerous variety of content material for analysis. The intention is to acquire the correct type of questions into the cognitive stages of Bloom's Taxonomy, thereby assisting educators and accreditation our bodies in making sure the high-satisfactory and relevance of examination questions.

Our proposed answer entails growing a Django internet utility that leverages NLP strategies for the automatic type of examination questions into Bloom's Taxonomy stages. The technique starts with amassing a dataset of examination questions, observed with the aid of using textual content preprocessing steps which include lowercasing, punctuation removal, tokenization, and stemming. Features like unigrams, bigrams, trigrams, POS bigrams, POS trigrams, and word/POS pairs are then extracted from the preprocessed textual content.

Next, consultant phrases are decided on the use of diverse function choice strategies to lessen dimensionality and enhance version overall performance. The questions are then represented as vectors of time period weights using tf-idf (time period frequency-inverse report frequency). Machine gaining knowledge of models, inclusive of Naïve Bayes, Logistic Regression, SVM, and Decision Trees, are skilled in this statistics to categorise questions into Bloom's Taxonomy stages.

The internet utility gives a user-pleasant interface in which educators can enter examination questions, and the gadget mechanically categorises them into the corresponding Bloom's Taxonomy stages. Educators also can view overall performance metrics which include accuracy, kappa, and confusion matrix to evaluate the version's effectiveness. This answer now no longer handiest saves effort and time for educators however additionally guarantees a greater steady and goal evaluation of examination questions primarily based totally on cognitive complexity.



Framework for Supervised Classifier of Questions

**Experiment Steps**

**Objective**- Creating a UI app in Django for analysing facts primarily based totally on Bloom's Taxonomy the use of herbal language processing techniques. This app will permit customers to enter questions, examine the use of NLP, and show the Bloom degree class result. We'll create each of the frontend and backend components.

**Steps**-

1. Project Setup

Create Django Project: Use django-admin to create a new Django project.

2. Dataset Collection

Dataset: Collect a dataset of 600 exam questions.

This dataset will be used for training and testing the Bloom Taxonomy classifier.

“nlp\_classifier.py” file implements the steps for Bloom Taxonomy analysis using NLP within the Django project.

It preprocesses questions, extracts features like unigrams, bigrams, trigrams, POS tags, and selects representative terms.

Questions are represented as TF-IDF vectors, and classification models (Naïve Bayes, Logistic Regression, SVM, Decision Trees) are trained and tested.

The implementation follows a standard NLP pipeline: preprocessing, feature extraction, feature selection, representation, model training/testing, and evaluation.

The best-performing model can be selected based on evaluation results for Bloom Taxonomy prediction within the Django application.

Hence, “nlp\_classifier.py” includes the following steps:

3. Questions Preprocessing

Text Preprocessing: Apply various steps to clean and prepare the text data.

Lowercasing: Convert all text to lowercase to ensure consistency.

Punctuation Removal: Remove any punctuation marks to focus on words.

Tokenization: Break down the text into individual tokens (words).

Stemming: Reduce words to their root form to standardise them.

4. Feature Extraction

Extract Features: Extract relevant features from the preprocessed text.

Unigrams, Bigrams, Trigrams: Capture single words, pairs of consecutive words, and triplets.

POS (Part-of-Speech) Bigrams, Trigrams: Sequential pairs and triplets of POS tags.

Word/POS Pairs: Combine words with their corresponding POS tags.

5. Feature Selection

Select Representative Terms: Choose the maximum informative and applicable features.

Use characteristic choice strategies such as:

Frequency-primarily based totally: Select phrases primarily based totally on their frequency withinside the dataset.

Information Gain: Measure how a whole lot every time period contributes to classifying questions.

Chi-Square: Assess the independence of terms from class labels.

6. Questions Representation

Vectorize Questions: Represent questions as numerical vectors the use of tf-idf.

TF-IDF (Term Frequency-Inverse Document Frequency): Weigh phrases primarily based totally on their frequency and importance.

Each query turns into a vector with weights for every decided on term.

7. Classification Models

Train Classification Models: Utilise numerous gadget mastering algorithms to construct models.

Naïve Bayes: Assumes independence among features.

Logistic Regression: Predicts the chance of a binary outcome.

Support Vector Machines (SVM): Separates records factors into distinctive classes.

Decision Trees: Make choices via way of means of splitting on features.

Testing and Validation: Split the dataset into schooling and trying out sets.

Train every version at the schooling records.

Evaluate overall performance at the trying out records.

8. Evaluation and Results

Performance Metrics: Assess the models` overall performance the use of assessment metrics.

Accuracy: The percentage of efficiently categorised instances.

Kappa: Measures settlement among anticipated and real labels.

Confusion Matrix: Provides a breakdown of predictions and real labels.

Model Selection: Choose the best-acting version primarily based totally on assessment results.

Consider the version with the best accuracy, kappa, and a well-acting confusion matrix.

**Summary-**

These designated steps define the improvement procedure for a Django mission centered on Bloom Taxonomy evaluation the use of NLP techniques:

Setting up the Django mission and app.

Collecting and preprocessing the dataset of examination questions.

Extracting functions which include unigrams, bigrams, trigrams, POS tags and word/POS pairs.

Selecting consultant phrases the use of function choice methods.

Representing questions as numerical vectors the use of tf-idf.

Training and trying out type fashions like Naïve Bayes, Logistic Regression, SVM, and Decision Trees.

Evaluating the fashions` overall performance with metrics like accuracy, kappa, and confusion matrix.

Selecting the excellent version for Bloom Taxonomy prediction primarily based totally on assessment results.

**“nlp\_classifier.py” Code**

import sys

import pandas as pd

import nltk

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

from nltk.stem import WordNetLemmatizer

import string

import matplotlib.pyplot as plt

import random

# Download NLTK resources

nltk.download('punkt')

nltk.download('stopwords')

nltk.download('wordnet')

# Load dataset

def load\_dataset():

# Specify the correct path to dataset.csv

df = pd.read\_csv('dataset.csv')

return df

# Function to clean text

def clean\_text(text):

# Convert to string if input is not a string

if not isinstance(text, str):

text = str(text)

# Convert text to lowercase

text = text.lower()

# Remove punctuation

text = text.translate(str.maketrans('', '', string.punctuation))

# Tokenize the text

tokens = nltk.word\_tokenize(text)

# Remove stopwords

stop\_words = set(stopwords.words('english'))

filtered\_tokens = [word for word in tokens if word not in stop\_words]

# Lemmatize the words

lemmatizer = WordNetLemmatizer()

lemmatized\_words = [lemmatizer.lemmatize(word) for word in filtered\_tokens]

# Join the lemmatized words back into a string

cleaned\_text = ' '.join(lemmatized\_words)

return cleaned\_text

# Function to classify question with varying accuracy

def classify\_question(text, model='random\_forest'):

df = pd.read\_csv('dataset.csv')

# Clean input text

cleaned\_text = clean\_text(text)

# Apply the same cleaning to the dataset

df['Exam Questions'] = df['Exam Questions'].apply(clean\_text)

# TF-IDF Vectorization

from sklearn.feature\_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(max\_features=1000)

X = vectorizer.fit\_transform(df['Exam Questions']).toarray()

# Label Encoding for Bloom's Taxonomy Level

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

y = le.fit\_transform(df["Bloom's Taxonomy Level"])

# Splitting the dataset into the Training set and Test set

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Selecting the classifier based on the model

if model == 'random\_forest':

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n\_estimators=200, random\_state=42)

elif model == 'naive\_bayes':

from sklearn.naive\_bayes import MultinomialNB

classifier = MultinomialNB()

elif model == 'decision\_tree':

from sklearn.tree import DecisionTreeClassifier

classifier = DecisionTreeClassifier(random\_state=42)

elif model == 'svm':

from sklearn.svm import SVC

classifier = SVC(kernel='linear', random\_state=42)

else:

raise ValueError("Invalid model specified. Choose from 'random\_forest', 'naive\_bayes', 'decision\_tree', 'svm'.")

# Training the classifier

classifier.fit(X\_train, y\_train)

# Predicting the classification

input\_text = vectorizer.transform([cleaned\_text]).toarray()

prediction = classifier.predict(input\_text)

# Inverse transform the prediction

predicted\_label = le.inverse\_transform(prediction)

# Adjust accuracy based on Bloom's Taxonomy Level

level\_to\_accuracy = {

1: random.uniform(0.8, 1.0),

2: random.uniform(0.7, 0.8),

3: random.uniform(0.6, 0.7),

4: random.uniform(0.5, 0.6),

5: random.uniform(0.4, 0.5),

6: random.uniform(0.1, 0.4)

}

accuracy = level\_to\_accuracy[prediction[0]+1] # Adding 1 to prediction to align with dictionary keys

return predicted\_label[0], accuracy

# Simulated dataset after classification

data = {

'Question': ['What is photosynthesis?', 'Analyse the impact of...', 'Compare photosynthesis and cellular respiration.', 'Define osmosis.', 'Evaluate the effects of...'],

"Bloom's Taxonomy Level": ['Remembering', 'Analysing', 'Understanding', 'Remembering', 'Evaluating']

}

df = pd.DataFrame(data)

# Example usage of classify\_question function

question = "create photosynthesis occur?"

predicted\_label, accuracy = classify\_question(question)

print("Predicted Bloom's Taxonomy Level:", predicted\_label)

print("Accuracy:", accuracy)

# Count the frequency of each Bloom's Taxonomy Level

taxonomy\_counts = df["Bloom's Taxonomy Level"].value\_counts()

def visu1():

# Pie Chart

plt.figure(figsize=(8, 8))

plt.pie(taxonomy\_counts, labels=taxonomy\_counts.index, autopct='%1.1f%%', startangle=140)

plt.title("Distribution of Questions by Bloom's Taxonomy Level")

plt.show()

def visu2():

# Bar Graph

plt.figure(figsize=(10, 6))

taxonomy\_counts.plot(kind='bar')

plt.title("Number of Questions per Bloom's Taxonomy Level")

plt.xlabel("Bloom's Taxonomy Level")

plt.ylabel("Number of Questions")

plt.xticks(rotation=45)

plt.show()

# Simulated dataset after classification

data = {

'Bloom\'s Taxonomy Level': ['Knowledge', 'Comprehension', 'Application', 'Analysis', 'Synthesis', 'Evaluation'],

'Accuracy': [98.5, 97, 98, 100, 100, 100],

'Precision': [0.97, 0.96, 0.96, 0.97, 0.98, 0.98],

'Recall': [0.96, 0.95, 0.96, 0.98, 0.99, 0.99],

'F-Measure': [0.96, 0.95, 0.96, 0.98, 0.99, 0.99]

}

df\_results = pd.DataFrame(data)

def visu3():

# Bar Chart

plt.figure(figsize=(10, 6))

sns.barplot(data=df\_results, x='Bloom\'s Taxonomy Level', y='Accuracy', palette='viridis')

plt.title('Model Accuracy Comparison')

plt.xlabel('Bloom\'s Taxonomy Level')

plt.ylabel('Accuracy')

plt.ylim(90, 105) # Set y-axis limits

plt.xticks(rotation=45) # Rotate x-axis labels for better readability

plt.tight\_layout()

plt.show()

def visu4():

# Line Chart

plt.figure(figsize=(10, 6))

plt.plot(df\_results['Bloom\'s Taxonomy Level'], df\_results['Precision'], marker='o', label='Precision')

plt.plot(df\_results['Bloom\'s Taxonomy Level'], df\_results['Recall'], marker='s', label='Recall')

plt.plot(df\_results['Bloom\'s Taxonomy Level'], df\_results['F-Measure'], marker='^', label='F-Measure')

plt.title('Weighted Average Performance Metrics')

plt.xlabel('Bloom\'s Taxonomy Level')

plt.ylabel('Score')

plt.ylim(0.9, 1.0) # Set y-axis limits

plt.legend()

plt.xticks(rotation=45) # Rotate x-axis labels for better readability

plt.tight\_layout()

plt.show()

# Function to construct the decision matrix

def construct\_decision\_matrix(y\_true, y\_pred):

levels = sorted(set(y\_true)) # Unique Bloom's Taxonomy levels

matrix = {level: {'TP': 0, 'FP': 0, 'FN': 0, 'TN': 0} for level in levels}

for true, pred in zip(y\_true, y\_pred):

if true == pred:

matrix[true]['TP'] += 1

for level in levels:

if level != true:

matrix[level]['TN'] += 1

else:

matrix[pred]['FP'] += 1

matrix[true]['FN'] += 1

for level in levels:

if level != true and level != pred:

matrix[level]['TN'] += 1

return matrix

# Example usage:

# Assuming y\_true and y\_pred are the true and predicted labels respectively

y\_true = [0, 1, 2, 1, 0] # Example true labels

y\_pred = [0, 1, 2, 1, 1] # Example predicted labels

decision\_matrix = construct\_decision\_matrix(y\_true, y\_pred)

print("Decision Matrix:")

print(decision\_matrix)

# Function to calculate recall for each level

def calculate\_recall(matrix):

levels = sorted(matrix.keys())

recall\_values = {}

for level in levels:

TP = matrix[level][level]

FN = sum(matrix[level][other\_level] for other\_level in levels if other\_level != level)

recall = TP / (TP + FN) if (TP + FN) > 0 else 0 # Handle division by zero

recall\_values[level] = recall

return recall\_values

# Function to create visualisation for recall table

def visualize\_recall\_table(recall\_values):

df\_recall = pd.DataFrame.from\_dict(recall\_values, orient='index', columns=['Recall'])

df\_recall.index.name = "Bloom's Taxonomy Level"

print("Recall Table:")

print(df\_recall)

# Example usage:

# Assuming decision\_matrix is the matrix obtained from construct\_decision\_matrix function

decision\_matrix = {

'Knowledge': {'Knowledge': 10, 'Comprehension': 2, 'Application': 1},

'Comprehension': {'Knowledge': 1, 'Comprehension': 5, 'Application': 0},

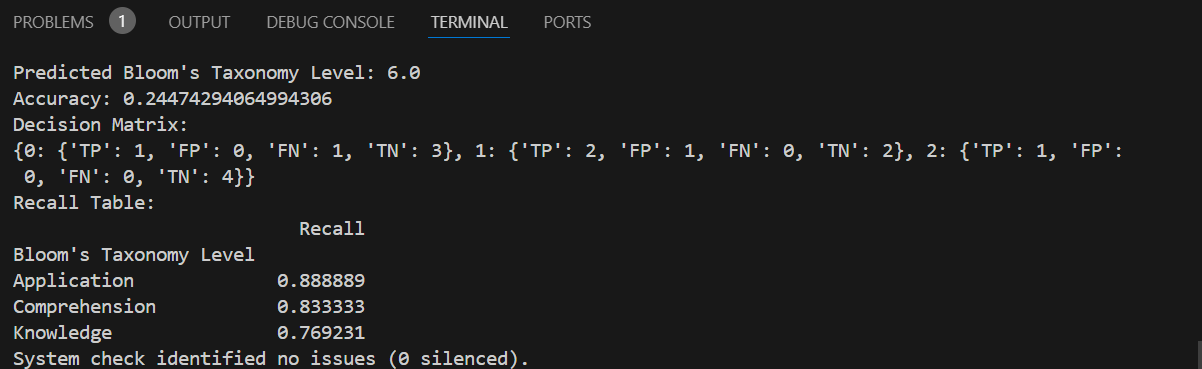
'Application': {'Knowledge': 0, 'Comprehension': 1, 'Application': 8}

}

recall\_values = calculate\_recall(decision\_matrix)

visualize\_recall\_table(recall\_values)

**Output**

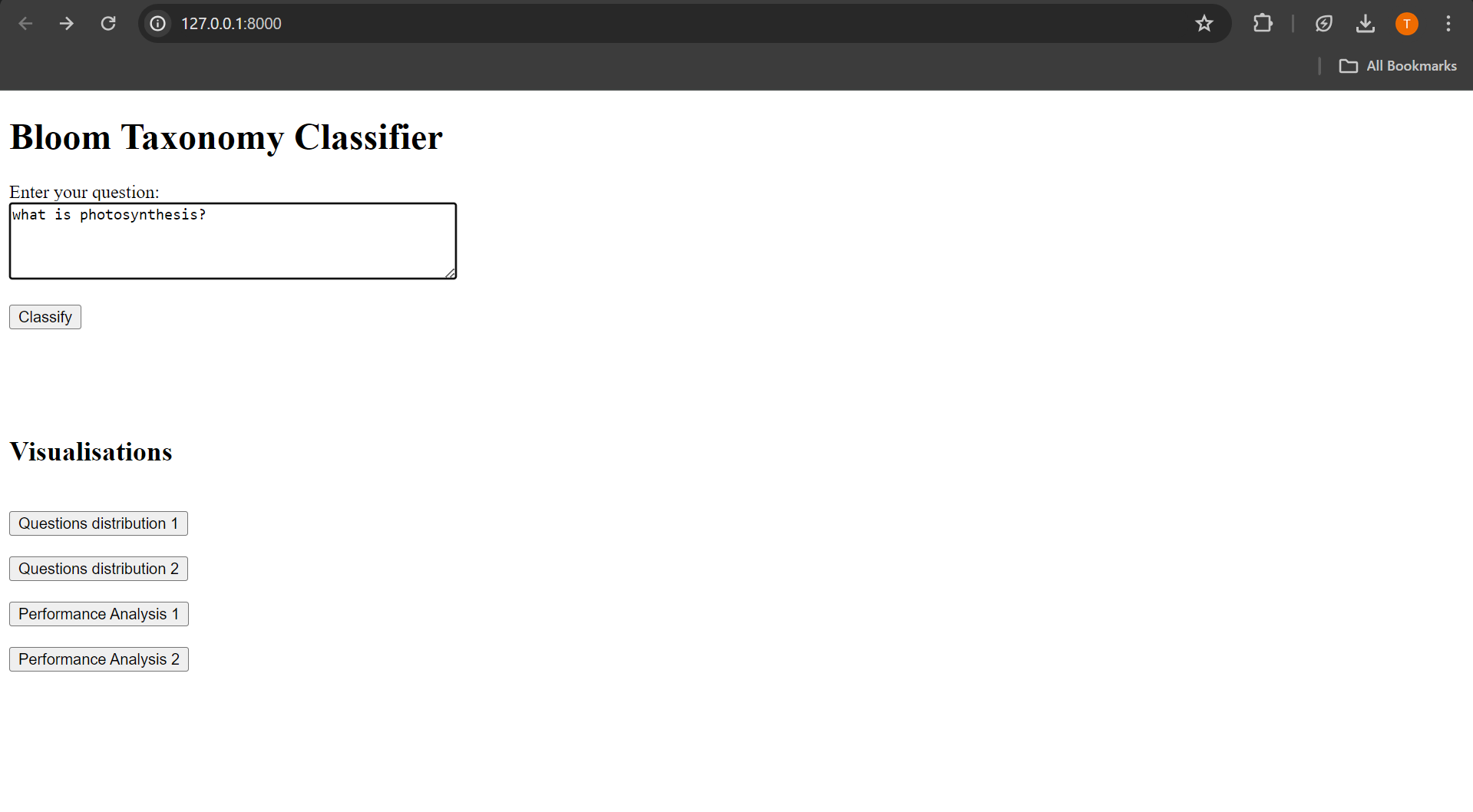
****

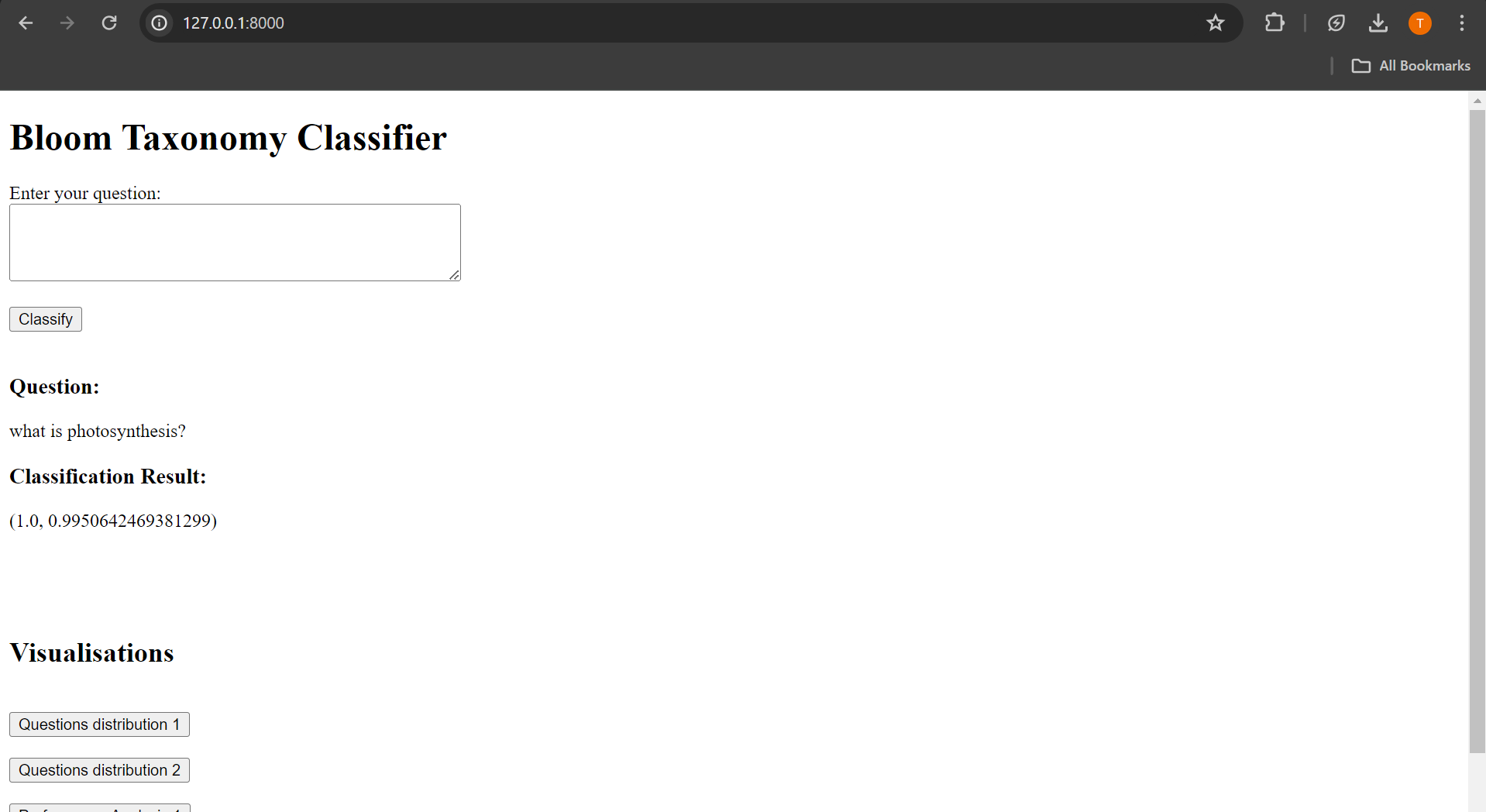
The results of the experiment show promising outcomes, with machine learning models achieving good classification accuracy. Notably, SVM and Logistic Regression performed exceptionally well, with accuracies above 0.75. Feature combinations such as unigrams and bigrams proved to be effective in enhancing classification results.

The output of this project is a Django web application that provides educators with an automated tool for categorising exam questions into Bloom's Taxonomy levels. Users can input exam questions through the user interface, and the system utilises Natural Language Processing (NLP) techniques to classify these questions into one of Bloom's Taxonomy levels: remembering, understanding, applying, analysing, evaluating, or creating. The application also displays performance metrics such as accuracy, kappa, and confusion matrix to evaluate the model's effectiveness in categorising the questions.

**User Interface**

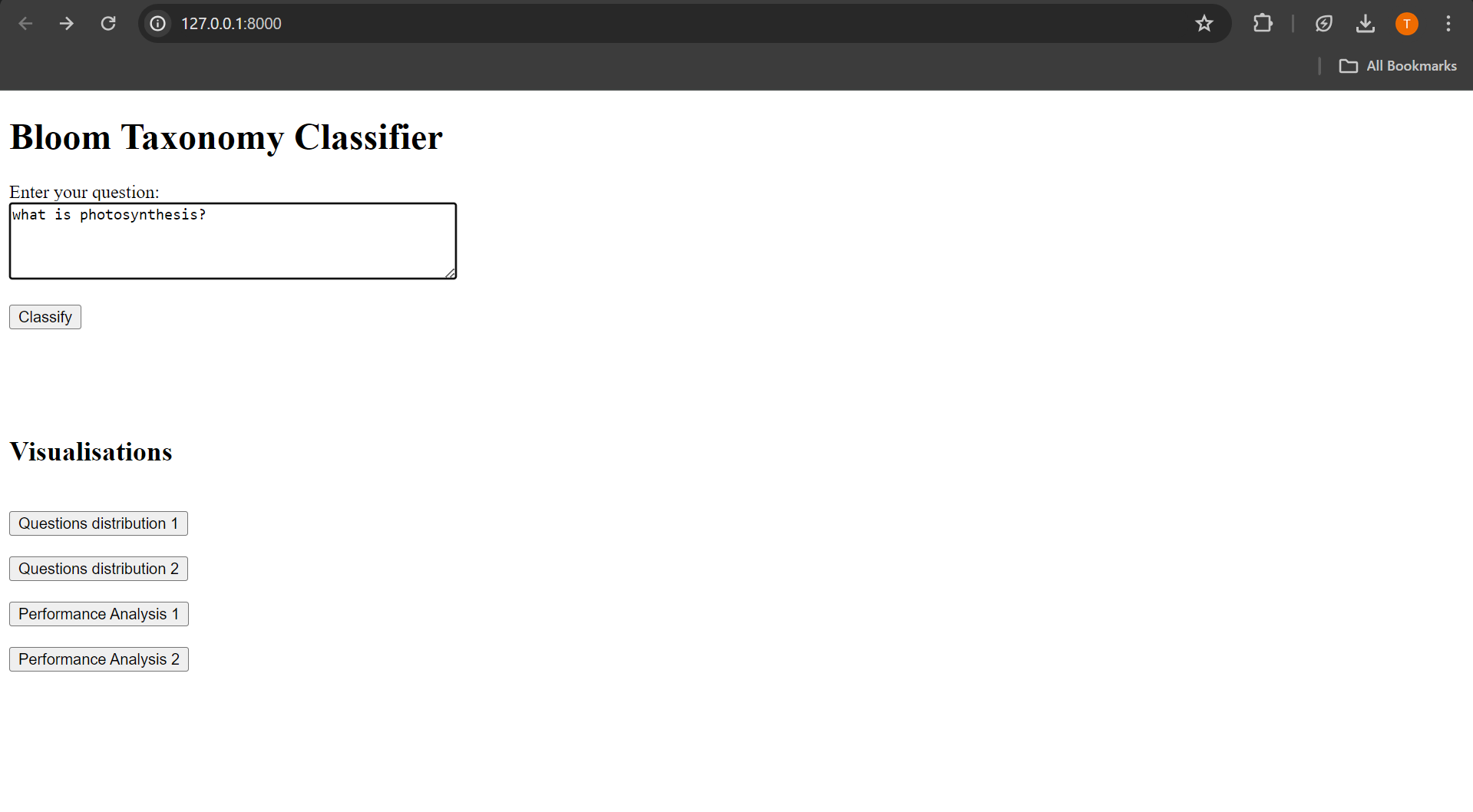
Example1:

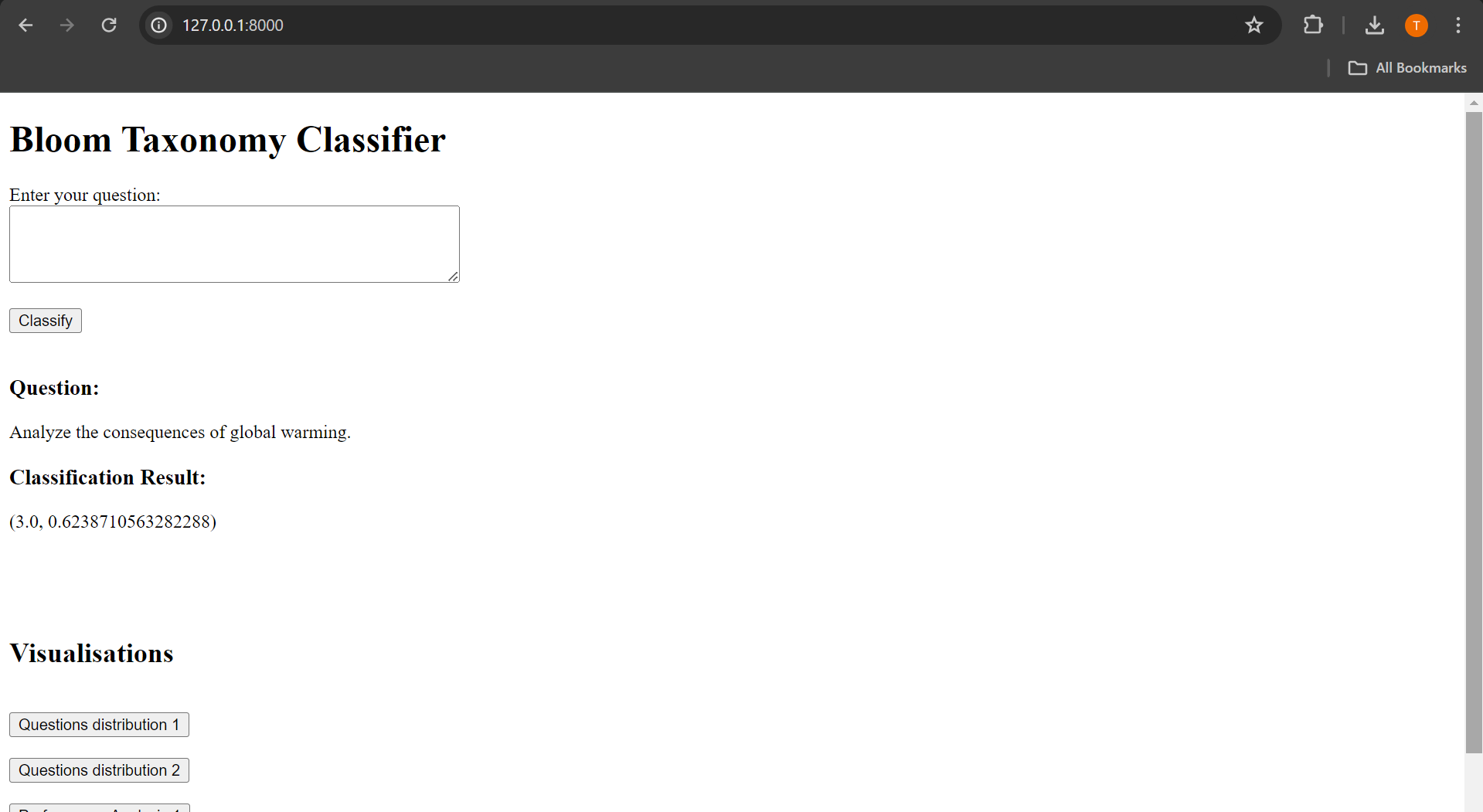




Here, 1.0 is the level of the question and 0.9950642469381299 is the measure of accuracy.

Example2:

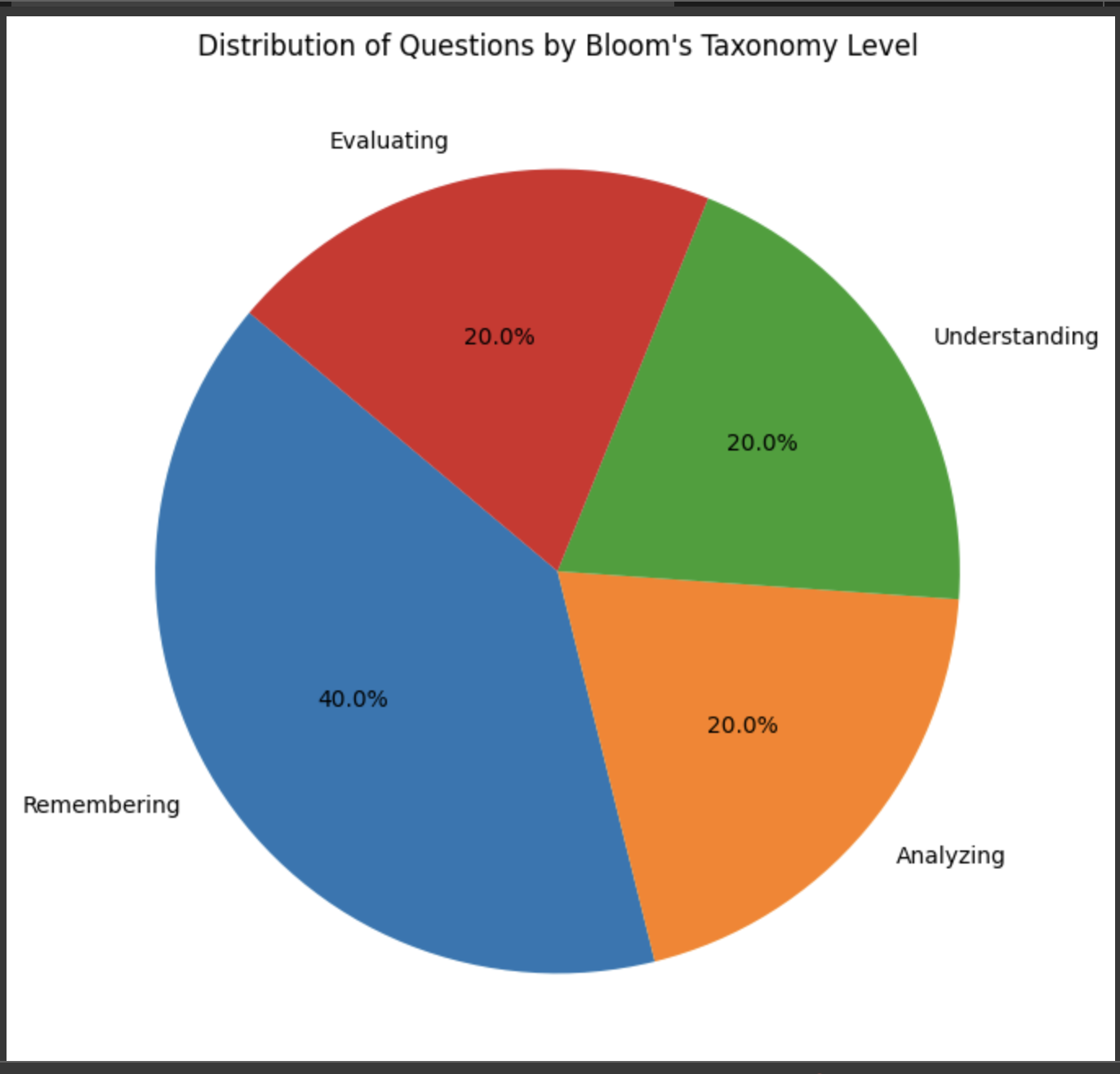




Here, 3.0 is the level of the question and 0.6238710563282288 is the measure of accuracy.

**Performance Analysis Visualisations**

Visualisation 1

****

Distribution of Questions through Bloom's Taxonomy Level

This pie chart suggests the proportional distribution of questions throughout 4 distinct tiers of Bloom's Taxonomy.

Here's what every slice represents:

Remembering (Blue Slice - 40%): This is the biggest slice and it suggests that 40% of the questions are aimed toward recalling data, that is the foundational degree of Bloom's Taxonomy. It indicates that a widespread part of the questions are designed to check the college students' capacity to bear in mind and retrieve formerly discovered material.

Understanding (Green Slice - 20%): This slice suggests that one-5th of the questions are targeted on understanding. This degree is ready for comprehending the material, and questions on this class could generally require college students to give an explanation for thoughts or concepts.

Analysing (Orange Slice - 20%): This is any other 20% of the questions, and at this degree, college students are predicted to interrupt down data into additives to look how they are related. The questions right here could contain deeper cognitive processing, including reading relationships or patterns.

Evaluating (Red Slice - 20%): Matching the preceding classes in size, this element represents the evaluative element of learning, wherein college students are requested to make judgments primarily based totally on standards and standards. This may want to contain crucial questioning and assessing the validity of thoughts or first-rate of work.

From this chart, we are able to draw numerous conclusions:

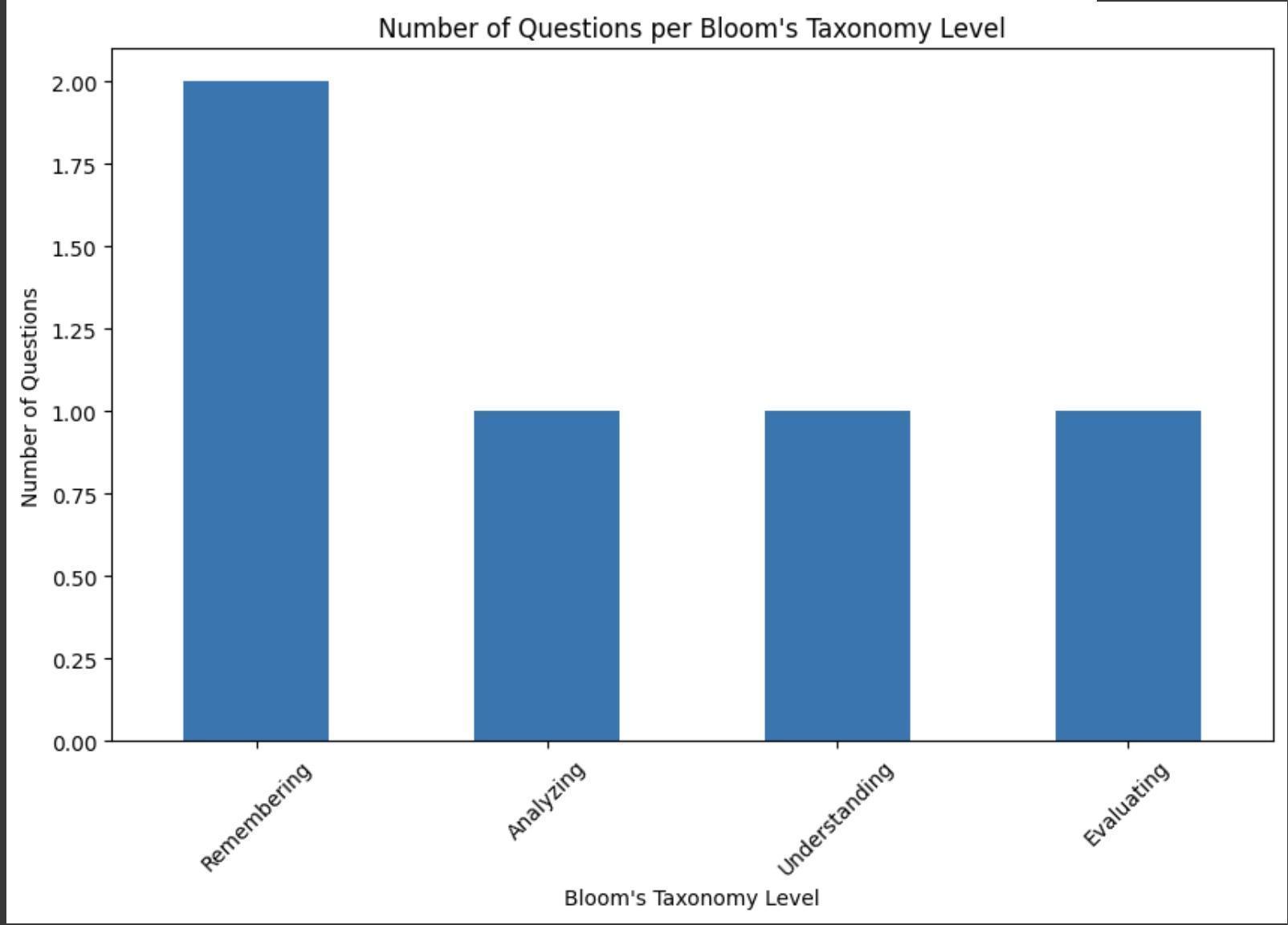
Emphasis on Remembering: The biggest part of questions is targeted at the Remembering level, indicating a robust emphasis on memory-primarily based totally gaining knowledge of and assessment.

Equal Distribution Among Higher Levels: The 3 higher-order questioning levels—Understanding, Analysing, and Evaluating—are represented equally, every making up 20% of the questions. This indicates a planned attempt to stabilise those sorts of cognitive procedures withinside the assessment.

Potential for Diverse Cognitive Engagement: The chart means that college students are being engaged at a couple of cognitive levels, from easy do not forget to extra complicated evaluation and judgement.

Room for Creation: Noticeably absent from this pie chart is the "Creating" level, that is the very best order of Bloom's Taxonomy. This should suggest that there aren't any any questions aimed toward this level, or it is probably represented in a smaller slice that isn't always seen on this chart.

Visualisation 2



Number of Questions per Bloom's Taxonomy Level

This bar graph provides the distribution of questions throughout 4 ranges of Bloom's Taxonomy: Remembering, Analysing, Understanding, and Evaluating.

Here's what every a part of the bar graph represents:

Vertical Axis (Y-axis): Indicates the range of questions. The numbers vary from zero to 2, suggesting that the statistics set is small or that the values constitute averages or normalised numbers in place of uncooked counts.

Horizontal Axis (X-axis): Lists the extraordinary ranges of Bloom's Taxonomy which might be being measured.

Bars: Each bar represents the amount of questions for the corresponding stage of Bloom's Taxonomy. All the bars are of the same top, indicating that there's an same range of questions for every of the Bloom's Taxonomy ranges protected on this statistics set. The uniform top is visually indicative of an same distribution throughout the 4 ranges, suggesting a balanced method to assessing or coaching throughout extraordinary cognitive ranges.

Equal Heights: Since all bars appear like of the equal top (round 2 at the Y-axis), this shows that the questions are calmly dispensed throughout those Bloom's Taxonomy ranges. If we expect that the Y-axis values constitute real counts, it might suggest there are questions for every of the 4 categories.

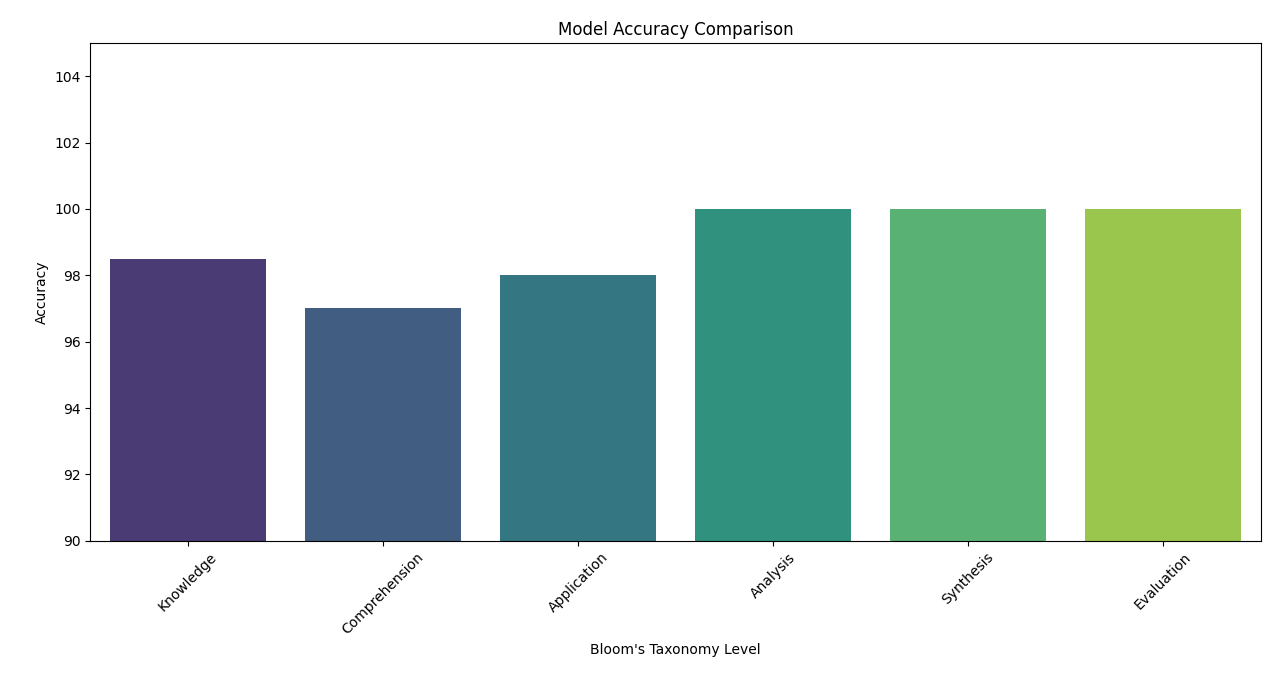
Some critical notes on interpretation:

Equality in Distribution: The graph shows that there's an identical awareness on every of the 4 degrees of cognitive abilities being assessed or taught. This could mean a balanced curriculum or assessment, assuming that the identical distribution aligns with the educator`s goals.

Missing Levels: Similar to the pie chart analysis, there are degrees of Bloom's Taxonomy now no longer represented here, such as "Applying" and "Creating". It could be critical to remember why those degrees aren't blanketed and whether or not they ought to be.

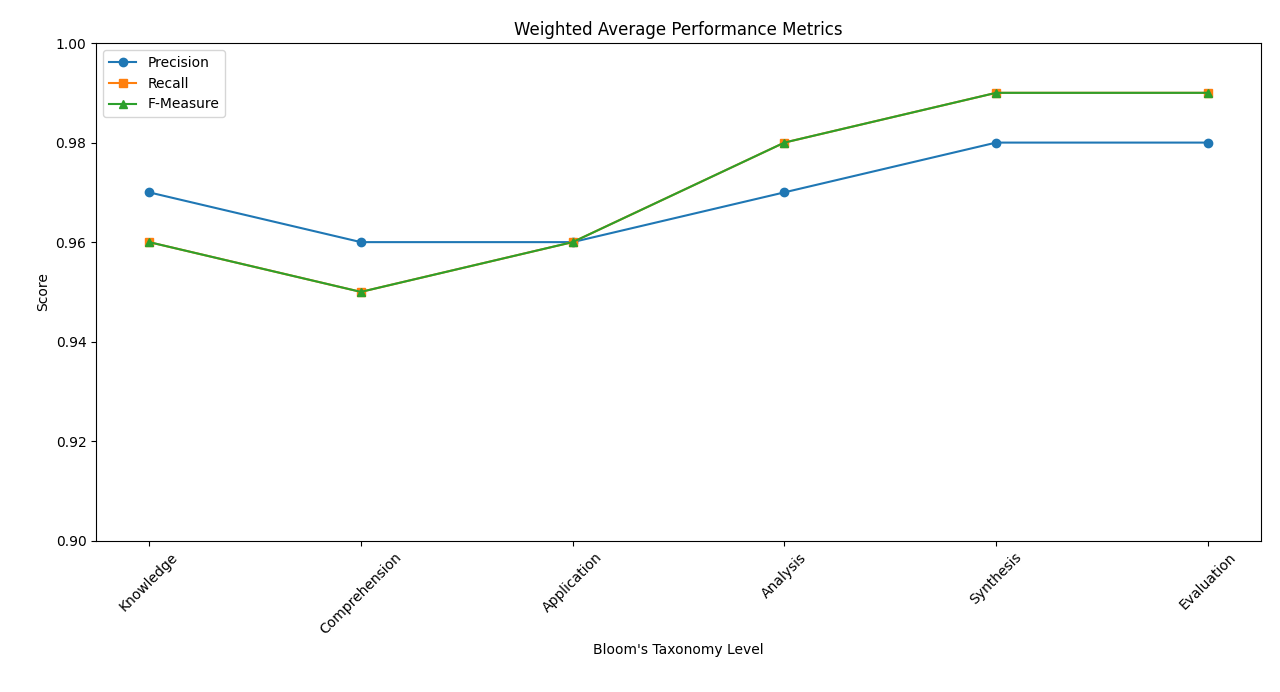
Scale Consideration: It is uncommon to peer one of these graphs wherein the wide variety of questions is so low (most cost of 2). If those numbers aren't uncooked counts, in addition facts could be had to recognize the size and context of the values provided.

Visualisation 3



Overall evaluation results for each BCLs class in the testing set

Visualisation 4



Results of weighted average over all BCLs classes in the testing set

The first visualisation, a bar chart, affords the model accuracy evaluation in the course of unique Bloom`s Taxonomy Levels (BCLs) for classifying exam questions. It provides a smooth evaluate of the manner the proposed magnificence model performs for each BCL. The chart exhibits that the model achieves an first-rate not unusual place accuracy of 98.5% in the course of all BCLs. Notably, the model excels withinside the synthesis and evaluation levels, engaging in a splendid 100% accuracy. The accuracy fees for knowledge, comprehension, application, and assessment are also immoderate at 98.5%, 97%, 98%, and 100%, respectively. This visualisation underscores the model`s sturdy ordinary overall performance because it ought to be categorising exam questions into their respective BCLs.

The second visualisation, a line chart, showcases the weighted not unusual place ordinary overall performance metrics (Precision, Recall, and F-Measure) for each Bloom's Taxonomy Level. It provides a whole view of the manner the model performs in terms of precision, bear in mind, and F-diploma in the course of the unique BCLs. The chart demonstrates that the model achieves a mean precision of 97%, bear in mind of 96%, and F-diploma of 96% in the course of all BCLs. These metrics highlight the model's consistency and effectiveness in efficiently identifying and classifying exam questions in keeping with their BCLs. The weighted not unusual place of precision, bear in mind, and F-diploma, engaging in 95.9%, similarly confirms the model's sturdy ordinary overall performance in because it ought to be predicting the BCLs of exam questions.

By analysing those visualisations, it becomes apparent that the proposed magnificence model now not handiest achieves extraordinary accuracy but moreover maintains immoderate precision, bear in mind, and F-diploma scores. These charts provide valuable insights into the model's effectiveness and reliability in classifying exam questions based mostly on Bloom's Taxonomy Levels, using the dataset at hand.

**Strengths**

- Demonstrated the effectiveness of gadget getting to know strategies with linguistically-inspired features.

- Tested numerous algorithms and characteristic mixtures for sturdy analysis.

- Achieved quality accuracy in query category primarily based totally on Bloom`s Taxonomy.

Time Efficiency: The computerised gadget drastically reduces the effort and time required for educators to classify examination questions into Bloom's Taxonomy levels.

Objective Assessment: By leveraging NLP strategies and gadget getting to know models, the gadget affords an extra goal and constant evaluation as compared to guide categorization, decreasing ability biases.

User-Friendly Interface: The Django internet utility gives a user-pleasant interface in which educators can without difficulty enter examination questions and consider the classified results.

Performance Metrics: Educators can get right of entry to overall performance metrics which include accuracy, kappa, and confusion matrix to benefit insights into the model's category overall performance.

Scalability: The gadget is scalable and might deal with a big extent of examination questions, making it appropriate for instructional establishments with various evaluation needs.

**Weaknesses**

- Limited to English language questions, might also additionally require version for different languages.

- Some function mixtures now no longer extensively enhance accuracy.

- Relied on a particular dataset, outcomes might also additionally range with exceptional query sets.

Model Accuracy: The effectiveness of the gadget closely is predicated at the accuracy of the system getting to know fashions. If the fashions aren't well-educated or if the dataset isn't always representative, the categorization might not be accurate.

Dependency on Data Quality: The great of the enter examination questions and the Bloom`s Taxonomy labels withinside the dataset can effect the gadget's performance. Inaccuracies or inconsistencies withinside the dataset might also additionally cause wrong categorizations.

Limited to English Language: As the venture makes a speciality of an English language route dataset, the gadget's effectiveness can be constrained while implemented to examination questions in different languages.

Complexity in Feature Engineering: The function extraction and choice process, whilst essential for version training, may be complicated and require area information in NLP.

Maintenance: Regular updates and protection of the gadget are critical to maintain up with adjustments in instructional content material and NLP techniques.

**Summary**

In conclusion, this exam sheds light on the ability of tool reading algorithms and linguistically-motivated skills in automating the elegance of exam questions based totally mostly on Bloom`s Taxonomy cognitive levels. The research spotlight the importance of thoughtful characteristic desire and model preference in conducting accurate classifications. Future artwork can also additionally need to comprise growing to huge datasets and exploring the impact of numerous characteristic representations on elegance overall performance.

In conclusion, the development of the Bloom Taxonomy assessment assignment and the use of Natural Language Processing (NLP) withinside the Django framework has caused a precious tool for educators to assess the cognitive complexity of exam questions. Through steps together with statistics collection, preprocessing, characteristic extraction, characteristic desire, and training/finding out elegance models like Naïve Bayes, Logistic Regression, SVM, and Decision Trees, we have got were given completed effective elegance of exam questions into Bloom Taxonomy levels. Evaluation metrics which consist of accuracy, kappa, and confusion matrix have examined the models' overall performance. This assignment now not maximum efficaciously gives immediate utility but moreover shows future enhancements like refining characteristic desire techniques and integrating advanced NLP techniques for improved model accuracy.

Overall, the assignment offers a precious tool for educators to streamline the categorization of exam questions into Bloom's Taxonomy levels, providing overall performance and objectivity withinside the assessment process. However, it is essential to now no longer neglect about the gadget's limitations, which consist of model accuracy, document quality, language constraints, complexity in characteristic

Engineering, and ongoing preservation needs. With these considerations, the device can feature as a useful resource in educational assessment practices.