

DAYANANDA SAGAR UNIVERSITY

KUDLU GATE, BANGALORE – 560068



**Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING**

**Special Topic- II (19CS3604) Report
on**

**“DESIGN AND DEVELOPMENT OF A QUESTION
CLASSIFIER BASED ON BLOOM’S TAXONOMY”**

Submitted By

ABHISHEK KUMAR	(ENG19CS0012)
ARYAN KUMAR	(ENG19CS0044)
AMAN THAPA	(ENG19CS0026)

**Under the supervision of
Dr. KIRAN B. MALAGI
Associate Professor, CSE Dept.**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY,**

(2021-2022)

DAYANANDA SAGAR UNIVERSITY

School of Engineering, Kudlu Gate, Bangalore-560068



CERTIFICATE

This is to certify that Mr./Ms. Abhishek Kumar, Aryan Kumar, Aman Thapa bearing USN ENG19CS0012, ENG19CS0044, ENG19CS0026 respectively has satisfactorily completed his Special Topics Project as prescribed by the University for the 6th semester B.Tech. program in Computer Science & Engineering during the year 2021-22 at the School of Engineering, Dayananda Sagar University., Bangalore.

Date: _____

Signature of the guide

Signature of Chairman

Department of Computer Science &
Engineering

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of a task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We are especially thankful to our **Chairman, Dr. Girisha G S**, for providing necessary departmental facilities, moral support and encouragement.

We are very much thankful to **Dr. Kiran B.Malagi**, for providing help and suggestions in completion of this special topics project successfully.

We have received a great deal of guidance and co-operation from our friends and we wish to thank all that have directly or indirectly helped us in the successful completion of this project work.

ABHISHEK KUMAR (ENG19CS0012)
ARYAN KUMAR (ENG19CS0044)
AMAN THAPA (ENG19CS0026)

DECLARATION

We hereby declare that the work presented in this project entitled - “**DESIGN AND DEVELOPMENT OF A QUESTION CLASSIFIER BASED ON BLOOM’S TAXONOMY**”, has been carried out by us and it has not been submitted for the award of any degree, diploma or the project of any other college or university.

ABHISHEK KUMAR (ENG19CS0012)

ARYAN KUMAR (ENG19CS0044)

AMAN THAPA (ENG19CS0026)

TABLE OF CONTENT

S. No.	Contents	Page no
1	Abstract	2
2	Introduction	3
3	Problem Statement	4
4	Literature Survey	5
5	Design	6
6	Functional Requirement	7
7	Methodology	8
8	Code	9-14
9	Result	15-16
10	Conclusion	17
11	References	18

ABSTRACT

When a student is assessed by an exam consisting of questions, the alignment process involves classifying these questions according to the cognitive process categories needed to answer them. This process can be time-consuming if an exam contains many questions, and it can be easy to lose oversight of whether the questions in the assessment are representative of what is taught in the course material.

The task of classifying questions into categories that represent the cognitive processes needed to answer them can be facilitated by providing a classification tool. This tool also gives educators insight by displaying a summary of the different question categories present in a set of questions.

We propose a software solution that uses NLP techniques to classify a course's questions and provides a clear overview of the classes in Bloom's revised taxonomy present in these courses.

INTRODUCTION

Educators can make use of a taxonomy to classify questions used to assess learning objectives into classes to align their courses. Manually classifying each question can be a time consuming task prone to error. It is also easy to lose oversight of whether the questions in the assessment are representative of what is taught in the course material, when the assessment consists of many questions.

Bloom's Revised Taxonomy: Bloom's taxonomy is a taxonomy that classifies educational learning objectives and questions. The taxonomy classifies questions and learning goals in the cognitive process dimension. The classes of the original taxonomy created by Bloom et al. are: Knowledge, Comprehension, Application, Analysis, Synthesis and Evaluation.

PROBLEM STATEMENT

Problem: An examination in the form of a set of questions is often used as a final assessment of a student's performance in a course. Therefore, we have researched whether there is a better way for educators to approach this task while keeping a good overview.

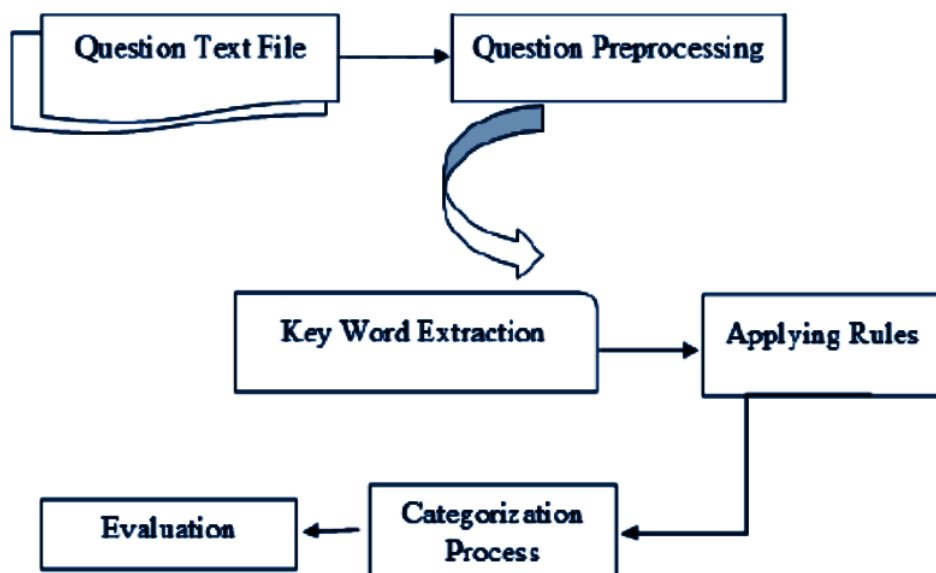
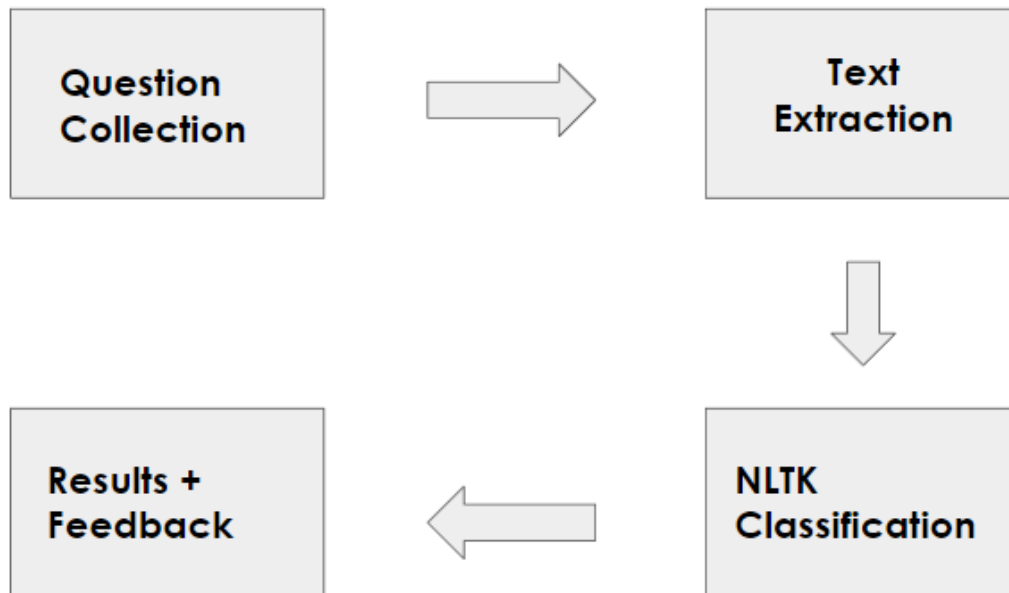
Solution: In this project, we propose a solution to this problem. We have developed a classifier that classifies questions according to Bloom's revised taxonomy using NLP methods. The scope of this project is limited to well-formulated questions in text form of roughly university-level difficulty posed in English. The classifier will not take spelling mistakes into account. We will also only train the classifier on standalone questions i.e. questions that can be answered without an introduction, questions that don't need an accompanying figure or text, and questions that aren't multiple choice.

LITERATURE SURVEY

Author's Name/ Paper Title / Paper	Conference/ Journal Name and year	Technology/ Design	Results shared by author	What you infer
Framework for preparation of quality question paper/ Mr. Kiran B. Malagi , Mrs. Karuna C. Gull , Ms. Veena M	Volume IX Issue V MAY 2020	Provides the process to generate automatic question paper	System has generated Question Paper without repetition of the questions. Compared with university question papers and also are given to the experts from the university for comparison and the level of satisfaction is also high	As the number of questions in the database with different bloom's levels and marks increase, efficiency of the system is still expected to increase.
Exam Questions Classification Based on Bloom's Taxonomy: Approaches and Techniques Karima Makhoulf , Lobna Amouri, Nada Chaabane, and Nahla EL-Haggar	San Francisco State Univ. and June 2021	pre-processing, dimensionality reduction, classification and evaluation	Deciding for the best technique among these steps is not a simple problem as each technique has its benefits and its drawbacks.	Combining various approaches will lead to higher accuracy of Question Classification
EXAM QUESTIONS CLASSIFICATION BASED ON BLOOM'S TAXONOMY Nazila Omar and Dhuha Abdulhadi Abduljabbar	Theoretical and Applied Information Technology and August 2015	three machine learning approaches using a combination voting algorithm	KNN is best in Knowledge, Application, Synthesis and Evaluation, SVM is best in Comprehension and Analysis	The results show that combination can outperform individual classifiers.

DESIGN

Data Flow



FUNCTIONAL REQUIREMENTS

Software requirement:

- Operating System: Windows 8 and above.
- Language: Python
- IDE: VS Code/Pycharm etc.

Hardware requirements:

- Processor: Intel(R)Core (TM) i5-5300U CPU @2.30GHz
- Memory: Powerful enough to run the program.

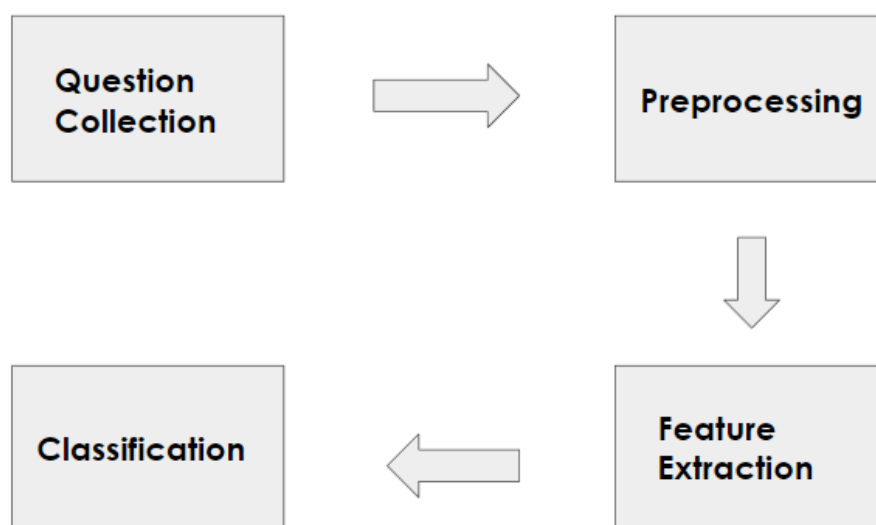
METHODOLOGY

Question Collection- The aim of this study was to build a model to classify questions based on Bloom's Taxonomy from different domains, two open domain datasets had been used to ensure the stability of the proposed method.

Pre-processing- Performing pre-processing for unstructured data is highly recommended in any framework which uses machine learning to reduce the unnecessary, duplicated, irrelevant, and noisy data.

Feature extraction- Feature extraction is a process of transforming the raw input data into a meaningful set of features, in such a way that it can be understandable to the machine learning classifiers.

Classification- Three of the most common supervised machine learning classification algorithms are used, which are the K-Nearest Neighbour (KNN), Logistic Regression (LR), and Support Vector Machine (SVM).



CODE

```
import tkinter
from tkinter import messagebox
def error():
    errorMessage = ""Incorrect formatting of question paper. Please adhere to these guidelines.
```

1. The document has to be in the Microsoft Word 2007/2010 Document (*.docx) format.
2. The headings for each part (part A, part B) must be above the table and must NOT be in a cell. It must be in the text form.
3. The questions for each part (A,B,C) have to be in one continuous table.
4. The or questions have to be separated by a separate cell with the word 'Or' in it.
5. For questions with sub questions the sub questions have to be given in one cell and have to be in the given format.
 - Subquestion 1 (4 marks)
 - Subquestion 2 (2 marks)
 - Subquestion 3 (2 marks)
 - Subquestion 4 (2 marks)The marks have to be given in brackets with the word 'marks' after the number.

6. Please choose the correct mark distribution from the options""

```
# hide main window
root = tkinter.Tk()
root.withdraw()
# message box display
messagebox.showwarning("Error", errorMessage)
try:
    from a2 import *
except:
    error()

from tkinter import *
from tkinter import filedialog, ttk
from ttkthemes import ThemedStyle
# from TkinterDnD2 import DND_FILES, TkinterDnD
import tkinter.messagebox
from ttkbootstrap import Style
```

```
status = True
```

```
markDistFile = open("Mark Distribution.txt", 'r')
kLevelFile = open("K-Level.txt", 'r')
```

```
markDistFile = open("Mark Distribution.txt", 'r')
kLevelFile = open("K-Level.txt", 'r')
```

```

mark_lines = markDistFile.readlines()
K_lines = kLevelFile.readlines()
markDistList = []
kDistList = []
for i in mark_lines:
    markDistList.append(i.strip("\n"))

for i in K_lines:
    kDistList.append(i.strip("\n"))

markDistFile.close()
kLevelFile.close()

LARGEFONT = ("Helvetica", 12, 'bold')

filePath = "
markChoice = '0'
kChoice = '0'
markNumberDict = dict()
dept = "
lower_max_k1 = 0
upper_max_k3 = 0
lower_min_k1 = 0
upper_min_k3 = 0
SMALLFONT = ("Shruti", 11)

class tkinterApp(Tk):
    def __init__(self, *args, **kwargs):
        Tk.__init__(self, *args, **kwargs)
        container = Frame(self)
        container.pack(side="top", fill="both", expand=True)

        container.grid_rowconfigure(0, weight=1)
        container.grid_columnconfigure(0, weight=1)

        self.frames = {}

        for F in (LoginPage, UserPage):
            frame = F(container, self)
            self.frames[F] = frame

            frame.grid(row=0, column=0, sticky='nsew')

        self.show_frame(UserPage)

    def show_frame(self, cont):
        frame = self.frames[cont]
        frame.tkraise()

```

```

class LoginPage(Frame):
    def __init__(self, parent, controller):

        Frame.__init__(self, parent)
        titleLabel = Label(self, text="QN Paper Classification System", pady=50, padx=150,
font=LARGEFONT, background = '#faf3f3',foreground='#136335')
        titleLabel.grid(row=0, column=0, columnspan=3, sticky='nsew')

        loginLabel = Label(self, text="Login as: ", font=SMALLFONT, background =
'#faf3f3',foreground='#136335')
        loginLabel.grid(row=1, column=1, columnspan=1, sticky='n', pady=(20, 25))

        userButton = ttk.Button(self, text="Faculty member", width=6,
                                command=lambda: controller.show_frame(UserPage))
        userButton.grid(row=2, column=1, sticky='nsew')
class UserPage(Frame):
    def __init__(self, parent, controller):
        Frame.__init__(self, parent)

        def uploadAction(self):
            filename = filedialog.askopenfilename()
            browseLb.delete(0, END)
            browseLb.insert(0, filename)

        def drop_inside_entry(event):
            browseLb.delete(0, END)
            browseLb.insert("end", event.data)
        def onEnter(e):
            submitButton['background'] = 'black'
        def submit(event):
            global status
            if status == True:
                global filePath, markChoice, kChoice
                filePath = browseLb.get()
                if (filePath == ""):
                    tkinter.messagebox.showwarning(title="Wrong Input", message="Please upload a
file", )
                elif(filePath[0] == '{' and filePath[-1] == '}'):
                    filePath = filePath[1:-1]
                elif(markChoice == ""):
                    tkinter.messagebox.showwarning(title="Wrong Input", message="Please select
choice of Mark Distribution", )
                elif(kChoice == ""):
                    tkinter.messagebox.showwarning(title="Wrong Input", message="Please select
choice of K Level", )

            elif(filePath == ""):
                tkinter.messagebox.showwarning(title="Wrong Input", message="Please upload a
file", )
                elif(filePath[-4:] != '.docx'):

```

```

tkinter.messagebox.showwarning(title="Wrong Input", message="File must be of
docx type",)
else:
    app.destroy()
else:
    tkinter.messagebox.showwarning(title = "Wrong Formatting", message = "Wrong
formatting of Question Paper")

    # self.destroy()
def getValue():
    global markChoice, kChoice
    markChoice = v.get()
    kChoice = x.get()
    # print("You selected", x.get())
def _on_mouse_wheel(event):
    my_canvas.yview_scroll(-1 * int((event.delta / 120)), "units")

self.configure(bg = '#ffffff')
main_frame = Frame(self)
main_frame.pack(fill=BOTH, expand=1)

my_canvas = Canvas(main_frame, width=600, height=700, )
my_canvas.pack(side=LEFT, fill=BOTH, expand=1)
# Add A Scrollbar To The Canvas
my_scrollbar = ttk.Scrollbar(main_frame, orient=VERTICAL,
command=my_canvas.yview)
my_scrollbar.pack(side=RIGHT, fill=Y)

# Configure The Canvas
my_canvas.configure(yscrollcommand=my_scrollbar.set, bg = 'white')
my_canvas.bind('<Configure>', lambda e:
my_canvas.configure(scrollregion=my_canvas.bbox("all")))
my_canvas.bind_all("<MouseWheel>", _on_mouse_wheel)

# Create ANOTHER Frame INSIDE the Canvas
second_frame = Frame(my_canvas, bg = 'white')

# Add that New frame To a Window In The Canvas
my_canvas.create_window((0, 0), window=second_frame, anchor="nw")

browseLabel = ttk.Label(second_frame, text="Upload a file (.docx only)", style =
'custom.TLabel')
browseLabel.grid(row=0, column=0, padx=10, pady=(10, 5), columnspan = 3)

browseButton = ttk.Button(second_frame, text="Browse files", command=lambda:
uploadAction(self))
browseButton.grid(row=2, column=0, pady=(10, 10), columnspan = 3)

browseLb = ttk.Entry(second_frame)
browseLb.grid(row=1, column=0, ipadx=50, ipady=7, padx=125, columnspan = 3)

```



```

typeQuestionLabel = ttk.Label(second_frame, text = "Select choice of Mark
Distribution", style = 'custom.TLabel')
typeQuestionLabel.grid(row = 3, column = 0, columnspan = 2, padx=80, pady=(10, 15),)

v = StringVar()
x = StringVar()

currRow = 4
variables = []

for i in range(len(markDistList)):
    markRadioButton = ttk.Radiobutton(second_frame, text = markDistList[i], variable = v,
value = str(i+1), command = getValue, style = 'info.TRadiobutton')
    markRadioButton.grid(row=currRow+i, column=0, pady=10, padx=170)
    currRow += len(markDistList)

KlLabel = ttk.Label(second_frame, text="Select choice of K-Level distribution", style =
'custom.TLabel')
KlLabel.grid(row=currRow, column=0, columnspan=2, padx=80, pady=(20, 15))

for i in range(len(kDistList)):
    KLradioButton = ttk.Radiobutton(second_frame, text = kDistList[i], variable =x , value
= str(i+1), command = getValue)
    KLradioButton.grid(row=currRow + i + 2 + len(kDistList), column=0, pady=10,
padx=170)
    currRow += len(kDistList)

submitButton = ttk.Button(second_frame, text="Submit", width=8, style =
'success.TButton', command=lambda: submit(self))
submitButton.grid(row=999, column=0, columnspan = 3, pady=(20, 30))
def on_close():
    global status
    status = False
    app.destroy()

app = tkinterApp()

app.title("Audit System")
app.resizable(False, False)
style = Style(theme = 'minty')

style.configure('custom.TLabel', font = LARGEFONT, foreground = '#f3969a')
app.protocol("WM_DELETE_WINDOW", on_close)
style.configure('Wild.TRadiobutton', background='#f3969a', font = SMALLFONT)

app.mainloop()

```

```

def getKLevel(st):
    x = 0
    y = 0
    l1 = st.split("%")
    print(l1)
    for n, i in enumerate(l1):
        if (len(i) > 0 and n == 0):
            x = int(i[-3:])
        if (len(i) > 0 and n == 1):
            y = int(i[-3:])
    return x, y
def getMarkDict(st):
    st = st.replace(" ", "")
    l1 = st.split(",")
    # print(l1)
    print(l1)
    val = []
    ke = []
    for i in l1:
        for n, j in enumerate(i):
            if (j == 'm'):
                num1 = int(i[0:n])
                print(num1)
                val.append(num1)
            if (j == 'x'):
                num = int(i[n + 1:])
                print(num)
                ke.append(num)
    print(val)
    print(ke)
    markslist = list(zip(ke, val))
    print(markslist)
    markNumberDict = {}
    for i, j in markslist:
        print(i, j)
        markNumberDict[i] = j
    return (markNumberDict)
if(kChoice == '1'):
    lower_max_k1, upper_max_k3 = getKLevel(kDistList[0])
elif(kChoice == '2'):
    lower_max_k1, upper_max_k3 = getKLevel(kDistList[1])
elif(kChoice == '3'):
    lower_min_k1, upper_min_k3 = getKLevel(kDistList[2])
elif(kChoice == '4'):
    lower_min_k1, upper_min_k3 = getKLevel(kDistList[3])

markNumberDict = getMarkDict(markDistList[int(markChoice) - 1])
print(lower_min_k1, lower_max_k1, upper_max_k3, upper_min_k3)

```

RESULT

Marks Distribution:

2 marks x 6, 6 marks x 3, 10 marks x 2
 2 mark x 5, 10 marks x 4
 2 mark x 10, 10 marks x 2, 10 mark x 1
 2 mark x 10, 10 marks x 3

Verbs:

choose define find how label list match name Omit Recall Spell Tell What When Where Which Who Why	Classify Compare Contrast Demonstrate Explain Extend Illustrate Infer Interpret Outline Rephrase Relate Show Summarize Translate	Apply Build Construct Develop Experiment Identify Interview Make use of Model Organize Plan Select Solve Utilize	Analyze Assume Categorize Conclusion Discover Dissect Distinguish Divide Examine Function Inference Inspect List Motive Relationships Simplify Survey Take part Test Theme	Agree Appraise Assess Award Choose Compare Conclude Criteria Criticize Decide Deduct Defend Determine Disprove Estimate Evaluate Importance Influence Judge Justify Mark Measure Opinion	Adapt Change Combine Compile Compose Create Delete Design Discuss Elaborate Formulate Happen Imagine Improve Invent Maximize Minimize Modify Original Originate Predict Propose Solution
--	--	---	---	--	--

GUI:

Upload a file (.docx only)

Select choice of Mark Distribution

☐ 2 marks x 6, 6 marks x 3, 10 marks x 2

☐ 2 mark x 5, 10 marks x 4

☐ 2 mark x 10, 10 marks x 2, 10 mark x 1

☐ 2 mark x 10, 10 marks x 3

Select choice of K-Level distribution


☐ K1/K2 Max: 50% AND K3/K4 Max: 100%

☐ K1/K2 Max: 100% AND K3/K4 Max: 50%

☐ K1/K2 Min: 46% AND K3/K4 Min: 46%

Error Message:

Error



Incorrect formatting of question paper. Please adhere to these guidelines.

1. The document has to be in the Microsoft Word 2007/2010 Document (*.docx) format.
2. The headings for each part (part A, part B) must be above the table and must NOT be in a cell. It must be in the text form.
3. The questions for each part (A,B,C) have to be in one continuous table.
4. The or questions have to be separated by a separate cell with the word 'Or' in it.
5. For questions with subquestions the subquestions have to be given in one cell and have to be in the given format.
 Subquestion 1 (4 marks)
 Subquestion 2 (2 marks)
 Subquestion 3 (2 marks)
 Subquestion 4 (2 marks)
 The marks have to be given in brackets with the word 'marks' after the number.
6. Please choose the correct mark distribution from the options

Result Database:

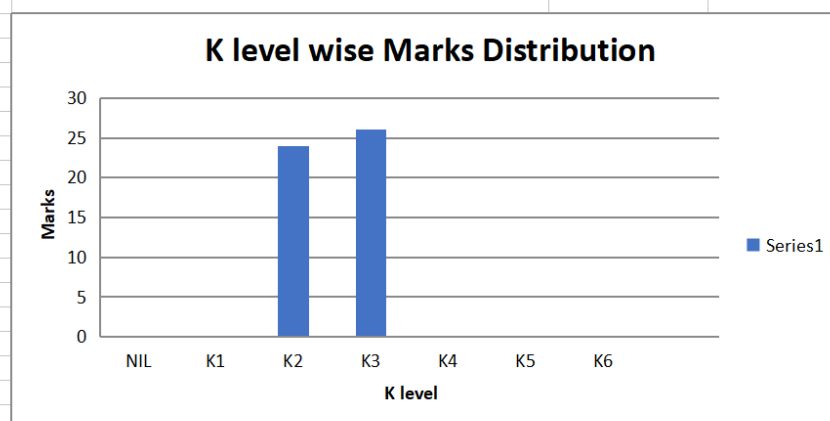
				Dayananda Sagar University	B.E & CSE		
				Degree :	CS1303 – Object Oriented Programming using Java		
				Subject :			
				Date :	20-10-21		
				Faculty :	Prepared By Dr. Kiran B. Malagi		
Q no	Sub Qn	CO	Marks	Question	K Level given by faculty	K Level according to Blooms	Suggested action verbs
1		CO1	2	1.Interpret the output for the following Java program and state the reason for the same. <pre>public class Person { int rno; String name; public static void main(String []args) { Person p1 = new Person(); Person p2 = new Person(); p2=p1; p1.rno=10; p1.name="Raja"; p2.rno=20; p2.name="Rahul"; System.out.println(p1.rno + " " +p1.name); System.out.println(p2.rno + " " +p2.name); } }</pre>	K2	K2	

	%	Marks
K1 / K2 Qns	48	24
K3 / K4 Qns	52	26
Total	100	50

Feedback for the faculty:

Questions Approved

They follow the given guidelines



CONCLUSION

We built a classifier that predicts the correct class out of the six possible taxonomy classes 95% of the time in our testing and outperforms the baseline we set. We have also provided an updatable dataset of verbs and marks distribution according to Bloom's revised taxonomy. We have provided a Result Database to efficiently expand the dataset and review questions by means of an Excel sheet. To showcase the capabilities we have created a GUI. Teachers are able to collect feedback on the correct label of a presented question, this feedback is inserted into the database and will be taken into account by our classifier when the model is retrained.

REFERENCE

Swart AJ. Evaluation of final examination papers in engineering: A case study using Bloom's taxonomy. IEEE Trans Educ. 2010; 10.1109/TE.2009.2014221

A. J. Swart, "Evaluation of Final Examination Papers in engineering: A case study using Bloom's taxonomy". Education, IEEE Transactions on, 53(2), 2010, pp. 257- 264. DOI: 10.1109/TE.2009.2014221.

M. Mohammed and N. Omar, "Question classification based on bloom's taxonomy using enhanced tf-idf," International Journal on Advanced Science, Engineering and Information Technology, vol. 8, no. 4-2, pp. 1679–1685, 2018.