DHA SUFFA UNIVERSITY

Department of Computer Science

Final-Term Semester Examination – Spring 2024

Course Code: CS-3102L **Course Title: Artificial Intelligence Lab**

Class/Section: BSCS-6C **Time Allowed: 1.5 Hour**

Date: 25-06-2024 **Max Marks: 40**

Course Instructor: Mr. Sagar

Module: A

Student's Name: Mirza Asfanyar Baig **Reg. No: Cs211087**

Note:

1. Attempt **all** questions.

Submission Instructions:

1. Edit in your given .doc file and submit as pdf naming convention cs234477_BSCS-6C_A

Q1. Write a block of code ensures that each unvisited neighbor of the current node is added to the open-set, its parent is recorded, and the cost to reach it (g[m]) is calculated and stored. This is essential for the A* algorithm to explore the graph efficiently and reconstruct the shortest path.

```
def update_neighbors(n, open_set, closed_set, parents, g):

#get neighbors gets adjacent neighbors of the nodes
 for (m, weight) in get_neighbors(n):

      if m not in open_set and m not in closed_set:
        open_set.add(m) # this ensures that each unvisited neighbor node is
added to the open set
          parents[m] = n
          g[m] = g[n] + weight
      else:
          if g[m] > g[n] + weight:
              # Update g[m] according to the values
              g[m] = g[n] + weight
              # recording of its parents
              parents[m] = n

              # If m is in the closed set, it is removed and added to the otehr
              if m in closed_set:
                  closed_set.remove(m)
                  open_set.add(m)
```

Q2. Write a piece of code tokenize this sentence: **My friend and I didn't buy anything on our trip** and show the Pos-tagging by using spacy library.

```
import spacy
nlp = spacy.load('en_core_web_sm')

string = "My friend and I didn't buy anything on our trip."
print(string)

doc = nlp(string)

for token in doc:
   print(f"{token.text}  :-----: {token.pos_}")
```

Q3. Write a code to perform following operations.

1. Separate X and Y assume in Dataset 9 (A, B, C, D, E, F, G, H, Z) features.
2. Feature Scaling
3. Train Test Split
4. Model Train and Test

```
#assuming we have a df with columns  A B C D ....

X = df[['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'Z']]
Y = df['Y']

# feature scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_scaled_df = pd.DataFrame(X_scaled, columns=X.columns)

# train
X_train, X_test, Y_train, Y_test = train_test_split(X_scaled, Y, test_size=0.2)

#print to verify

#Model Train and Model Test

model = LogisticRegression()
model.fit(X_train, Y_train)

Y_pred = model.predict(X_test)

conf_matrix = confusion_matrix(Y_test, Y_pred)
print("\nConfusion Matrix:\n", conf_matrix)
```

DHA SUFFA UNIVERSITY

Department of Computer Science

Final-Term Semester Examination – Spring 2024

Q4. Write a function to perform following operations on Textual Dataset.

1. Tokenize the Uncleaned Corpus
2. Remove Stop Words
3. Perform Stemming
4. Convert cleaned corpus into the model understandable form.

```
#1. Tokenize the Uncleaned Corpus
texts = [
   "This is the first document. It contains some text.",
   "Here is another document with different text.",
   "And here is the third document with more text."
]# imagine a long array

tokenized_texts = []
# nltk is imported and relevant set is downloaded
   for text in texts:
      # Tokenize the uncleaned corpus
      tokens = word_tokenize(text)
      tokenized_texts.append(tokens)

   print(tokenized_texts)

# 2 filter out stop words
#say stopword is downloaded from library
stop_words = set(stopwords.words('english'))
words = [w for w in words if w in stop_words]
print(words)

# 3 stemming
stemmed_texts = []
#tokens are already made in #1
   for text in texts:
      stemmed_tokens = [ps.stem(token) for token in tokens]
      stemmed_texts.append(stemmed_tokens)
   print(stemmed_texts)

#4
#this is all done using the relevant libraries
cleaned_corpus = []

   for text in texts:
     cleaned_text = ' '.join(tokens)
     cleaned_corpus.append(cleaned_text)

vectorizer = CountVectorizer()
X = vectorizer.fit_transform(cleaned_corpus).toarray()
print(X)   #X is now pre processed!
```