School Connect Online

SCO INTERNATIONAL ARTIFICIAL INTELLIGENCE OLYMPIAD

CLASS 9 OFFICIAL QUESTION PAPER

A comprehensive document for schools, teachers, parents, and students

Designed from Class 9 AI pathways and aligned with SCO's guided preparation, practice, reporting, and future-ready AI literacy.

- age-fit AI, machine learning, NLP, Python, ethics, and application-based reasoning for Class 9 learners globally
- professionally formatted question paper with compact question-number tags, answer key, and explanations
- ready for school, teacher, parent, and student use as a website/PDF learning resource

AI	Machine Learning	NLP	Python	Applications
Data	Logic	Fairness	Capstone	Practice

Paper Overview

Exam Name	SCO International Artificial Intelligence Olympiad
Class / Grade	Class 9 / Grade 9
Question Paper Set	Set S
Total Questions	50 multiple-choice questions
Duration	60 minutes
Question Type	Objective type with one correct answer
Sections	General Questions and Achievers Section
Use	Website sample paper, classroom practice, and guided preparation

Candidate Instructions

- Read every question carefully before marking the answer.
- There is only one correct answer for each multiple-choice question.
- Calculators, mobile phones, and digital aids are not allowed unless specifically permitted by the examination authority.
- Use the question paper for rough reading only; answers must be marked as instructed by the invigilator or online system.
- Passages, code snippets, and case details are part of the question block and should be read fully before choosing an option.
- This document includes the answer key and explanations for guided learning and website sample-paper use.

School Connect Online

Section A: General Questions

Q.1 Which of the following best describes a "hyperplane" in the context of Support Vector Machines (SVM) used for classification?

A) A curved boundary that adapts to complex data distributions.

B) A line (or plane in higher dimensions) that separates data points belonging to different classes.

C) A cluster centroid representing the average position of data points.

D) A node in a decision tree that splits the data.

Answer: B

Explanation: In SVM, a hyperplane is the decision boundary that separates the data into two distinct classes. It is typically a straight line (in 2D) or a flat subspace (in higher dimensions) that maximizes the margin between the classes.

Q.2 In decision tree algorithms, what is the primary purpose of the Gini Index?

A) To measure the variance in the dataset.

B) To compute the entropy of a node.

C) To assess the impurity or purity of a node.

D) To determine the number of clusters in the data.

Answer: C

Explanation: The Gini Index is a metric used to evaluate the impurity of a node. A lower Gini Index indicates a purer node, meaning that most data points in the node belong to a single class.

Q.3 Which clustering algorithm requires the number of clusters (k) to be specified before the analysis?

A) DBSCAN

B) Hierarchical clustering

C) k-Means clustering

D) Mean Shift clustering

Answer: C

Explanation: k-Means clustering requires the user to predefine the number of clusters (k). The algorithm then partitions the data into k clusters by minimizing the sum of squared distances from each data point to the cluster centroid.

Q.4 In classification, the term "overfitting" refers to:

A) A model that generalizes well to unseen data.

B) A model that is too simple and underperforms.

C) A model that performs exceptionally well on training data but poorly on new, unseen data.

D) A model that has too few parameters.

Answer: C

Explanation: Overfitting occurs when a model learns the details and noise in the training data to the extent that it negatively impacts the model's performance on new data. It means the model is too tailored to the training data and fails to generalize.

Q.5 Which of the following techniques is commonly used to prevent overfitting in decision trees?

A) Increasing the depth of the tree

B) Pruning the tree

C) Adding more features

D) Using k-Means clustering

Answer: B

Explanation: Pruning a decision tree involves removing parts of the tree that provide little power in predicting target variables. This helps prevent overfitting by simplifying the model.

Q.6 In clustering, what does the term "centroid" refer to?

A) The data point farthest from the cluster center

B) The average (mean) position of all data points in a cluster

C) An outlier that does not belong to any cluster

D) The boundary that separates clusters

Answer: B

Explanation: The centroid is the mean position of all points in a cluster. In k-Means clustering, each cluster is represented by its centroid, and data points are assigned to the cluster with the nearest centroid.

Q.7 Which of the following is considered a disadvantage of decision trees?

A) They are easily interpretable.

B) They can overfit if not properly pruned.

C) They require minimal preprocessing of data.

D) They work with both numerical and categorical data.

Answer: B

Explanation: Although decision trees are intuitive and easy to interpret, they can overfit if the tree grows too complex. Pruning is often necessary to improve their generalization to unseen data.

Q.8 Which algorithm is primarily associated with unsupervised learning?

A) Naive Bayes

B) k-Nearest Neighbors (k-NN)

C) k-Means clustering

D) Decision Trees

Answer: C

Explanation: k-Means clustering is an unsupervised learning algorithm, meaning it groups data based on inherent patterns without using labeled outcomes.

Q.9 In classification tasks, what is the purpose of a confusion matrix?

A) To visualize the decision boundary of an algorithm

B) To show the distribution of data points in clusters

C) To summarize the performance of a classification model by comparing predicted labels with actual labels

D) To calculate the overall variance in the data

Answer: C

Explanation: A confusion matrix is a tool used to evaluate the performance of a classification model by displaying the counts of correct and incorrect predictions across different classes.

Q.10 What is the primary purpose of using entropy in decision tree algorithms like ID3?

A) To calculate the average distance between data points

B) To measure the uncertainty or impurity of a node

C) To determine the optimal number of clusters

D) To standardize the features in the dataset

Answer: B

Explanation: Entropy is used in decision trees to quantify the amount of disorder or impurity within a node. By selecting the attribute that minimizes entropy (or maximizes information gain), the algorithm chooses the best split at each node.

Q.11 Tokenization in NLP refers to:

A) Converting text into lowercase letters only.

B) Splitting text into meaningful units like words or sentences.

C) Removing punctuation from text.

D) Translating text from one language to another.

Answer: B

Explanation: Tokenization is the process of breaking down a piece of text into smaller units, such as words, phrases, or sentences. It is a fundamental step in NLP used to prepare text for further analysis.

Q.12 Which Python library is most commonly used for basic NLP tasks like tokenization, stemming, and part-of-speech tagging?

A) TensorFlow

B) NLTK

C) Matplotlib

D) OpenCV

Answer: B

Explanation: NLTK (Natural Language Toolkit) is one of the most popular Python libraries for performing various NLP tasks such as tokenization, stemming, lemmatization, and POS tagging. TensorFlow is more focused on machine learning and deep learning, Matplotlib is used for plotting, and OpenCV is used for computer vision.

Q.13 What is the primary purpose of stemming in NLP?

A) To correct spelling errors in text.

B) To reduce words to their root form by removing suffixes.

C) To translate words into another language.

D) To identify named entities in text.

Answer: B

Explanation: Stemming reduces words to their base or root form, often by chopping off inflections and suffixes. For example, "running", "runs", and "ran" might be reduced to "run." This helps in normalizing text data for analysis.

Q.14 How does lemmatization differ from stemming?

A) Lemmatization uses a dictionary to find the proper base form of a word, while stemming uses simple rules to cut off suffixes.

B) Lemmatization increases the length of the word, whereas stemming shortens it.

C) Lemmatization is used for numerical data, while stemming is used for textual data.

D) Lemmatization and stemming are identical processes.

Answer: A

Explanation: Lemmatization transforms words to their canonical form (lemma) using vocabulary and morphological analysis, ensuring that the resulting word is valid. In contrast, stemming applies rule-based approaches that may not result in a valid word.

Q.15 Which of the following best describes word embeddings in NLP?

A) A method for generating random text data.

B) A technique that transforms words into numerical vectors capturing semantic relationships.

C) A process to segment text into sentences.

D) A technique used to compress text files.

Answer: B

Explanation: Word embeddings convert words into dense numerical vectors where semantically similar words are positioned close together in vector space. This representation is key to many modern NLP applications.

Q.16 In a typical Python implementation of word2vec using the gensim library, which function is used to train the model on a list of tokenized sentences?

A) Word2Vec.build_vocab()

B) Word2Vec.train()

C) Word2Vec(sentences)

D) Word2Vec.initialize()

Answer: C

Explanation: In gensim, you typically create a word2vec model by passing a list of tokenized sentences directly to the Word2Vec() constructor, which then internally builds the vocabulary and trains the model. Further training can be done with the train() method if needed.

Q.17 Consider the following Python code snippet. What does it output?

```
import nltk
nltk.download('punkt')
from nltk.tokenize import word_tokenize
text = "Hello, world! Welcome to NLP."
tokens = word_tokenize(text)
print(tokens)
```

A) ["Hello", "world", "Welcome", "to", "NLP"]

B) ["Hello,", "world!", "Welcome", "to", "NLP."]

C) ["Hello", ",", "world", "!", "Welcome", "to", "NLP", "."]

D) ["Hello world", "Welcome to NLP"]

Answer: C

Explanation: The word_tokenize function from NLTK splits the input text into words and punctuation. Therefore, punctuation marks are treated as separate tokens, resulting in the output ["Hello", ",", "world", "!", "Welcome", "to", "NLP", "."].

Q.18 Which method is used in NLP for part-of-speech (POS) tagging?

A) K-Means Clustering

B) Hidden Markov Models (HMM)

C) Support Vector Machines (SVM)

D) Principal Component Analysis (PCA)

Answer: B

Explanation: Hidden Markov Models (HMMs) are widely used in POS tagging to predict the sequence of tags for a given sentence. They consider the likelihood of a tag given the previous tag, making them effective for sequential data like text.

Q.19 What is the primary purpose of Named Entity Recognition (NER) in NLP?

A) To split text into sentences

B) To identify and classify key information (like names, dates, locations) in text

C) To remove stop words from text

D) To calculate the frequency of words

Answer: B

Explanation: Named Entity Recognition (NER) is an NLP task that identifies and categorizes key elements in text (such as names of people, organizations, locations, dates, etc.), helping extract structured information from unstructured data.

Q.20 Stop words in NLP are:

A) Rare words that are highly informative

B) Commonly used words that are often removed before processing

C) The first words of every sentence

D) Words that denote negative sentiment only

Answer: B

Explanation: Stop words are common words (such as "the", "is", "in") that usually do not contribute significant meaning to the text analysis. Removing them can help reduce noise and improve the efficiency of NLP models.

Q.21 Consider an AI-powered hiring system that uses historical employment data. Which of the following best describes how algorithmic bias might manifest in this scenario, and what is a primary ethical concern associated with it?

A) The system treats every candidate equally because it uses aggregate data; the main ethical concern is data privacy.

B) The system may inherit past hiring biases (e.g., gender or racial discrimination) present in the historical data, leading to unfair treatment of underrepresented groups; the primary ethical concern is perpetuating systemic inequality.

C) The system randomly selects candidates, eliminating bias; the primary ethical concern is that randomness may decrease efficiency.

D) The system is fully transparent, so any bias is easily correctable; the primary ethical concern is the overreliance on automated processes.

Answer: B

Explanation: Historical data often reflects existing societal biases. When an AI system is trained on such data, it can learn and reproduce those biases, leading to unfair outcomes. The ethical concern is that these biased decisions may reinforce systemic inequality and discrimination.

Q.22 Which of the following best differentiates between "disparate treatment" and "disparate impact" in the context of AI fairness, and why is understanding this distinction ethically important?

A) Disparate treatment involves intentional discrimination against a group, whereas disparate impact refers to policies that inadvertently harm a group; understanding this helps in designing both fair algorithms and compliant policies.

B) Disparate treatment is a statistical measure, while disparate impact is a legal concept; this distinction is primarily administrative and not ethically significant.

C) Disparate treatment applies to unsupervised learning, while disparate impact applies to supervised learning; this is crucial for choosing the right machine learning model.

D) Both terms refer to the same phenomenon and are used interchangeably; the ethical importance is minimal.

Answer: A

Explanation: Disparate treatment is intentional discrimination, whereas disparate impact occurs when a neutral policy disproportionately affects a particular group. Recognizing the difference is essential for both ethical AI design and legal compliance, ensuring fairness in automated decision-making.

Q.23 When developing an AI model, what role does the "fairness constraint" play, and which technique is often used to implement this constraint in a machine learning algorithm?

A) It ensures that the model uses the entire feature set equally; feature scaling is the common technique.

B) It ensures that the model's predictions are not biased toward any particular group; techniques like reweighing or fairness-aware regularization are often employed.

C) It guarantees that the model always achieves perfect accuracy; cross-validation is the common method.

D) It ensures the model does not overfit the training data; early stopping is typically used.

Answer: B

Explanation: Fairness constraints are applied to ensure that the AI model's outcomes do not favor one group over another. Techniques such as reweighing training data or adding fairness-aware regularization terms help adjust the model to meet specified fairness criteria.

Q.24 In ethical AI design, why is it crucial to incorporate transparency and interpretability, and which method is commonly used to enhance model interpretability?

A) They are not crucial as long as the model performs well; boosting is a common method for improved performance.

B) Transparency and interpretability help stakeholders understand and trust the model's decisions; methods like LIME (Local Interpretable Model-agnostic Explanations) are frequently used.

C) They guarantee that the model is free from any bias; principal component analysis (PCA) is used to enhance interpretability.

D) They are only required for regulatory compliance and have no practical value; random forest feature importance is the sole method used.

Answer: B

Explanation: Transparency and interpretability allow users and stakeholders to understand how the model arrives at its decisions, which is critical for trust, accountability, and identifying potential biases. Techniques such as LIME provide local explanations for individual predictions.

Q.25 Which of the following scenarios is most indicative of sample bias, and what ethical challenge does it present in AI systems?

A) An AI model trained on data that is too diverse, making it hard to converge; the challenge is increased computational cost.

B) An AI system that uses outdated data, leading to errors in prediction; the ethical challenge is irrelevant as long as the data is historical.

C) An AI model developed using data from a single demographic group, which then performs poorly on other groups; the ethical challenge is that it can lead to unfair, exclusionary outcomes.

D) An AI system that uses synthetic data for training; the challenge is the lack of real-world applicability.

Answer: C

Explanation: Sample bias occurs when the training data does not adequately represent the diversity of the real-world population. This leads to models that perform poorly for underrepresented groups, creating ethical challenges by perpetuating inequality and unfair treatment.

Q.26 In the context of AI ethics, what is meant by "algorithmic fairness," and which of the following metrics is commonly used to measure it?

A) Algorithmic fairness means the algorithm is 100% accurate; accuracy is the only metric needed.

B) It refers to the fairness of resource allocation during model training; GPU utilization is the key metric.

C) It implies that the algorithm's decisions do not systematically disadvantage any particular group; metrics like Equal Opportunity Difference or Statistical Parity Difference are used.

D) It means that the algorithm is transparent; model complexity is the metric to monitor.

Answer: C

Explanation: Algorithmic fairness ensures that decisions made by AI systems do not systematically disadvantage specific groups. Metrics such as Equal Opportunity Difference (which measures differences in true positive rates) or Statistical Parity Difference (which compares outcomes across groups) help quantify fairness.

Q.27 A company uses an AI system to predict creditworthiness. Which of the following steps is most effective for mitigating bias that might arise from historical loan data?

A) Increase the number of features without modifying the dataset; more data always reduces bias.

B) Remove sensitive attributes (like race or gender) from the dataset; however, this alone may not fully mitigate bias due to proxy variables.

C) Use a simple linear regression model; simplicity ensures fairness.

D) Only focus on improving the model's accuracy; bias is automatically reduced when accuracy increases.

Answer: B

Explanation: Removing sensitive attributes can help reduce direct bias; however, it's important to recognize that proxy variables may still carry bias. This approach, often combined with fairness-aware techniques, is a common step in mitigating bias in sensitive applications like credit scoring.

Q.28 Which of the following best describes the concept of "counterfactual fairness" in AI, and how is it typically assessed?

A) A model is counterfactually fair if its predictions change significantly when sensitive attributes are altered; it is assessed by measuring prediction variability.

B) A model is counterfactually fair if its predictions remain the same in hypothetical scenarios where sensitive attributes are altered; it is assessed using counterfactual reasoning and simulation.

C) Counterfactual fairness ensures the model's decisions are 100% accurate; it is assessed through error rate comparisons.

D) It refers to the fairness of the training process itself; it is assessed by analyzing the training algorithm's complexity.

Answer: B

Explanation: Counterfactual fairness requires that the model's predictions would remain unchanged in a counterfactual scenario where the sensitive attribute (e.g., race, gender) is different. This is typically evaluated using counterfactual reasoning, which simulates these alternative scenarios to test for fairness.

Q.29 In an effort to design an ethical AI system, developers decide to use "adversarial debiasing." Which statement best describes this approach?

A) It involves using adversarial examples to make the model robust against hacking.

B) It incorporates an adversarial network that attempts to predict sensitive attributes from the model's outputs, and the main model is trained to minimize this prediction accuracy, thereby reducing bias.

C) It trains two independent models, one for prediction and one for bias detection, without interaction.

D) It applies a simple threshold on the predictions to equalize outcomes across groups.

Answer: B

Explanation: Adversarial debiasing uses an adversarial network that tries to infer sensitive attributes from the primary model's outputs. The primary model is then trained to prevent this, thereby reducing the encoding of bias in its predictions. This technique helps in making the model more fair.

Q.30 Which ethical principle is most directly related to ensuring that AI systems provide equitable outcomes for all users, and which strategy best supports this principle?

A) Transparency; using deep neural networks to improve accuracy.

B) Accountability; enforcing strict data encryption methods.

C) Justice; employing fairness-aware algorithms and regular audits to detect and mitigate bias.

D) Autonomy; allowing users to modify algorithmic parameters.

Answer: C

Explanation: The principle of justice in AI ethics emphasizes fairness and equity in outcomes. Implementing fairness-aware algorithms, along with regular audits to detect and mitigate bias, is essential to uphold this principle and ensure that the system treats all users equitably.

Q.31 Palindromic Word Extraction

Examine the following code that extracts palindromic words (ignoring case) from a given sentence. What is the output when the code is executed?

```
import re
def extract_palindromes(text):
words = re.findall(r'\b\w+\b', text.lower())
return [word for word in words if word == word[::-1] and len(word) > 1]
text = "Madam Arora teaches malayalam and civic duties."
print(extract_palindromes(text))
```

A) ['madam', 'malayalam', 'civic']

B) ['madam', 'arora', 'malayalam', 'civic']

C) ['arora', 'malayalam', 'civic']

D) ['madam', 'arora', 'civic']

Answer: **B**

Explanation: The code first converts the text to lowercase and extracts all words. "madam", "arora", "malayalam", and "civic" are all palindromes (each reads the same forwards and backwards) and have more than one letter. Hence, the output is ['madam', 'arora', 'malayalam', 'civic'].

Q.32 Frequent Same-Letter Word

Consider the following function that returns the most frequent word from a list that starts and ends with the same letter. What will be printed?

```
def frequent_same_letter_word(words):
freq = {}
for word in words:
if word[0].lower() == word[-1].lower():
freq[word] = freq.get(word, 0) + 1
return max(freq, key=freq.get) if freq else None
words = ["level", "deed", "test", "level", "refer", "test", "deed", "rotor", "level", "noon"]
print(frequent_same_letter_word(words))
```

A) "level"

B) "deed"

C) "test"

D) "rotor"

Answer: **A**

Explanation: The function counts words that begin and end with the same letter. "level" appears 3 times, "deed" and "test" appear 2 times each, while others appear less. Hence, the most frequent is "level".

Q.33 Finding Anagrams

Given the following code that groups words into anagrams, what is the output when it is executed?

```
from collections import defaultdict
def find_anagrams(words):
d = defaultdict(list)
for word in words:
key = ''.join(sorted(word))
d[key].append(word)
return [group for group in d.values() if len(group) > 1]
words = ["listen", "silent", "enlist", "google", "gooogle", "evil", "vile", "live"]
print(find_anagrams(words))
```

A) [['listen', 'silent', 'enlist'], ['evil', 'vile', 'live']]

B) [['google'], ['gooogle']]

C) [['listen', 'silent'], ['evil', 'vile']]

D) [['listen', 'silent', 'enlist'], ['evil', 'vile', 'live'], ['google', 'gooogle']]

Answer: A

Explanation: The code groups words by their sorted character sequence. "listen", "silent", and "enlist" share the same sorted key, as do "evil", "vile", and "live". "google" and "gooogle" have different sorted keys, so they do not form a group with more than one element.

Q.34 Naive Bayes Score Calculation

Consider the following code that computes a simplified Naive Bayes score for a given text using predefined probabilities. What is the printed output (rounded to three decimals)?

```
import math
def naive_bayes_predict(text, prior, likelihood):
score = math.log(prior)
for word in text:
if word in likelihood:
score += math.log(likelihood[word])
return score
# For class A:
prior_A = 0.6
likelihood_A = {"good": 0.5, "bad": 0.1, "excellent": 0.2, "poor": 0.05}
text = ["good", "excellent", "bad"]
print(round(naive_bayes_predict(text, prior_A, likelihood_A), 3))
```

A) -5.116

B) -5.000

C) -4.500

D) -6.000

Answer: A

Explanation: Calculation steps:
Initial score = log(0.6) ≈ -0.511
Add log(0.5) ≈ -0.693 for "good"
Add log(0.2) ≈ -1.609 for "excellent"
Add log(0.1) ≈ -2.303 for "bad"
Total ≈ -0.511 - 0.693 - 1.609 - 2.303 = -5.116.

Q.35 Text Cleaning and Word Count

Review the following code that cleans a text string and counts the frequency of each word. What is the output of the code?

```
import re
from collections import Counter
def clean_and_count(text):
cleaned = re.sub(r'[^a-zA-Z0-9\s]', '', text).lower()
words = cleaned.split()
return Counter(words)
text = "Hello!!! This is a test... Test, test; hello?"
print(clean_and_count(text))
```

A) Counter({'hello': 3, 'test': 3, 'this': 1, 'is': 1, 'a': 1})

B) Counter({'test': 3, 'hello': 2, 'this': 1, 'is': 1, 'a': 1})

C) Counter({'hello': 2, 'test': 2, 'this': 1, 'is': 1, 'a': 1})

D) Counter({'hello': 1, 'test': 1, 'this': 1, 'is': 1, 'a': 1})

Answer: B

Explanation: After removing punctuation and converting to lowercase, the string becomes "hello this is a test test test hello". Splitting yields: ["hello", "this", "is", "a", "test", "test", "test", "hello"]. The frequency count is: hello=2, test=3, this=1, is=1, a=1.

Q.36 Decoding a Message with Slicing

Analyze the following code that decodes an encoded message by slicing and reversing segments. What does the function output when executed?

```
def decode_message(message):
part1 = message[::3]
part2 = message[1::3]
part3 = message[2::3]
return part1[::-1] + part2[::-1] + part3[::-1]
encoded = "abcdefghijklmno"
print(decode_message(encoded))
```

A) "mlkjihgfedcba"

B) "mjgdankhebolfic"

C) "cfilonkhbemjgda"

D) "mjgdankhebolfi"

Answer: B

Explanation: part1 = message[::3] yields characters at indices 0,3,6,9,12: "a", "d", "g", "j", "m" → "adgjm".
part2 = message[1::3] yields indices 1,4,7,10,13: "b", "e", "h", "k", "n" → "behkn".
part3 = message[2::3] yields indices 2,5,8,11,14: "c", "f", "i", "l", "o" → "cfilo".
Reversing each gives: "mjgda", "nkheb", and "olfic". Concatenating yields "mjgdankhebolfic".

Q.37 Filter with Lambda and List Comprehension

What is the output of the following code that filters numbers between 1 and 50 which are divisible by 7 but not by 5?

```
data = [x for x in range(1, 51)]
result = list(filter(lambda x: x % 7 == 0 and x % 5 != 0, data))
print(result)
```

A) [7, 14, 21, 28, 35, 42, 49]

B) [7, 14, 21, 28, 42, 49]

C) [14, 28, 42]

D) [7, 21, 35, 49]

Answer: B

Explanation: Multiples of 7 in the range are 7, 14, 21, 28, 35, 42, and 49. However, 35 is divisible by 5 and is filtered out, leaving [7, 14, 21, 28, 42, 49].

Q.38 Recursive Word Reversal

Consider the following recursive function that reverses the order of words in a sentence. What is the output?

```
def reverse_words(s):
if ' ' not in s:
return s
else:
return reverse_words(s[s.find(' ')+1:]) + " " + s[:s.find(' ')]
print(reverse_words("School Connect Olympiad"))
```

A) "Olympiad Connect School"

B) "School Connect Olympiad"

C) "Connect Olympiad School"

D) "Olympiad School Connect"

Answer: A

Explanation: The function recursively moves the first word to the end. Starting with "School Connect Olympiad", the final output becomes "Olympiad Connect School".

Q.39 TF-IDF Calculation

Consider the following simplified TF-IDF calculation. What is the printed output (rounded to three decimals)?

```
import math
docs = ["the cat sat on the mat", "the dog sat on the log", "the cat and the dog played"]
def compute_tf(doc):
words = doc.split()
tf = {}
for word in words:
tf[word] = tf.get(word, 0) + 1
for word in tf:
tf[word] /= len(words)
return tf
def compute_idf(docs):
idf = {}
total_docs = len(docs)
all_words = set(word for doc in docs for word in doc.split())
for word in all_words:
count = sum(1 for doc in docs if word in doc.split())
idf[word] = math.log(total_docs / count)
return idf
idf = compute_idf(docs)
tf_cat = compute_tf(docs[0])["cat"]
tf_dog = compute_tf(docs[1])["dog"]
score = tf_cat * idf["cat"] + tf_dog * idf["dog"]
print(round(score, 3))
```

A) 0.135

B) 0.250

C) 0.405

D) 0.500

Answer: A

Explanation: For document 1: "cat" frequency = 1/6; idf("cat") = log(3/2) ≈ 0.405465.
For document 2: "dog" frequency = 1/6; idf("dog") = log(3/2) ≈ 0.405465.
Total score ≈ (1/6 * 0.405465) + (1/6 * 0.405465) ≈ 0.067578 + 0.067578 = 0.135156, which rounds to 0.135.

Q.40 Cosine Similarity with Bag-of-Words

Given the following code that computes cosine similarity between two sentences using a bag-of-words model, what is the printed output (rounded to three decimals)?

```
import math
from collections import Counter
def cosine_similarity(text1, text2):
vec1 = Counter(text1.lower().split())
vec2 = Counter(text2.lower().split())
intersection = set(vec1.keys()) & set(vec2.keys())
dot_product = sum([vec1[x] * vec2[x] for x in intersection])
norm1 = math.sqrt(sum([v**2 for v in vec1.values()]))
norm2 = math.sqrt(sum([v**2 for v in vec2.values()]))
return dot_product / (norm1 * norm2)
text1 = "the quick brown fox jumps over the lazy dog"
text2 = "the fast brown fox leaps over the lazy hound"
print(round(cosine_similarity(text1, text2), 3))
```

A) 0.500

B) 0.727

C) 0.812

D) 0.900

Answer: B

Explanation: For both texts, the word counts are computed, and the intersection includes "the", "brown", "fox", "over", and "lazy". With "the" appearing twice and others once, the dot product sums to 8. The norms for both texts are √11 (≈3.317). Thus, cosine similarity ≈ 8 / (3.317 × 3.317) ≈ 8 / 11 ≈ 0.727.

Section B: Achievers Section

Q.41 An analyst uses the following code to analyze a customer review. What is the expected output (approximately), and what does the "compound" score represent?

```
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer
nltk.download('vader_lexicon')
sia = SentimentIntensityAnalyzer()
sentence = "The movie was utterly disappointing and a waste of time."
print(sia.polarity_scores(sentence))
```

A) {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}

B) {'neg': 0.541, 'neu': 0.459, 'pos': 0.0, 'compound': -0.7096}

C) {'neg': 0.2, 'neu': 0.8, 'pos': 0.0, 'compound': -0.2000}

D) {'neg': 0.7, 'neu': 0.3, 'pos': 0.0, 'compound': -0.9000}

Answer: B

Explanation: VADER analyzes the sentence and produces scores for negative, neutral, and positive sentiment. In this case, the review is largely negative. The "compound" score is a normalized aggregation of these scores, ranging from –1 (extremely negative) to +1 (extremely positive). Option B reflects a typical output for a strongly negative sentence.

Q.42 Examine the following code snippet. What will be the output, and what does the regex pattern do?

```
import re
text = "Hello!!! Welcome to School Connect Olympiad. Visit us at: www.schoolconnectonline.com"
clean_text = re.sub(r'[^a-zA-Z0-9\s]', '', text)
print(clean_text)
```

A) "Hello Welcome to School Connect Olympiad Visit us at wwwschoolconnectonlinecom"

B) "Hello!!! Welcome to School Connect Olympiad. Visit us at: www.schoolconnectonline.com"

C) "Hello Welcome to School Connect Olympiad Visit us at www schoolconnectonline com"

D) "Hello!!! Welcome to School Connect Olympiad Visit us at www.schoolconnectonline.com"

Answer: **A**

Explanation: The regex pattern [^a-zA-Z0-9\s] matches any character that is not a letter, digit, or whitespace. The re.sub function removes these characters, so all punctuation is stripped. Option A correctly shows the cleaned text after punctuation marks, including dots in the URL, have been removed.

Q.43 Given the following code using spaCy, what is the expected output and what do the entity labels indicate?

```
import spacy
nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")
for ent in doc.ents:
print(ent.text, ent.label_)
```

A)

Apple PERSON

U.K. LOCATION

$1 billion MONEY

B)

Apple ORG

U.K. GPE

$1 billion MONEY

C)

Apple ORG

U.K. LOC

$1 billion CARDINAL

D)

Apple PRODUCT

U.K. GPE

$1 billion QUANTITY

Answer: **B**

Explanation: The spaCy model recognizes "Apple" as an organization (ORG), "U.K." as a geopolitical entity (GPE), and "$1 billion" as a monetary value (MONEY). Option B is the correct interpretation.

Q.44 Consider the code for training a Naive Bayes classifier. What is the expected classification output for the test document, and what does it indicate?

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
docs = ["I love this product", "This is the worst purchase", "Absolutely fantastic", "Not worth the money"]
labels = [1, 0, 1, 0]  # 1: positive, 0: negative
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(docs)
clf = MultinomialNB()
clf.fit(X, labels)
test_doc = ["I hate this product"]
print(clf.predict(vectorizer.transform(test_doc)))
```

A) [1]

B) [0]

C) [1, 0]

D) [0, 1]

Answer: B

Explanation: The test document "I hate this product" contains the word "hate," a strong negative indicator. The classifier, trained on positive and negative reviews, predicts the sentiment as negative (label 0).

Q.45 What does the df.describe() function output when applied to the following DataFrame?

```
import pandas as pd
data = {'Name': ['Alice', 'Bob', 'Charlie', 'David'],
'Score': [85, 92, 78, 90]}
df = pd.DataFrame(data)
print(df.describe())
```

A) It outputs only the mean and standard deviation of the "Score" column.

B) It outputs summary statistics including count, mean, std, min, 25%, 50%, 75%, and max for the "Score" column.

C) It outputs the median and mode of the "Score" column only.

D) It outputs the full DataFrame with descriptive text.

Answer: B

Explanation: The describe() method in Pandas returns summary statistics for numeric columns, including count, mean, standard deviation (std), min, 25th percentile, 50th percentile (median), 75th percentile, and max. Option B accurately describes the output.

Q.46 Consider the following code that computes cosine similarity between text documents. What does the similarity matrix represent, and what insight does it provide?

```
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import CountVectorizer
docs = ["I love programming in Python", "Python programming is fun", "I enjoy long walks on the beach"]
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(docs)
sim_matrix = cosine_similarity(X)
print(sim_matrix)
```

A) A 3x3 matrix where high values indicate identical documents.

B) A matrix where each value represents the Euclidean distance between document vectors.

C) A 3x3 cosine similarity matrix where higher scores indicate more similar word usage between documents.

D) A matrix that only measures document length similarity.

Answer: C

Explanation: The cosine similarity matrix is a 3x3 matrix where each entry represents the cosine similarity between two documents. A higher value (closer to 1) indicates that the documents share similar word distributions, showing how similar they are in content.

Q.47 A school administrator uses the following code to visualize student scores. What does the code produce?

```
import matplotlib.pyplot as plt
scores = [85, 92, 78, 90]
names = ["Alice", "Bob", "Charlie", "David"]
plt.bar(names, scores, color='blue')
plt.xlabel("Students")
plt.ylabel("Scores")
plt.title("Student Scores")
plt.show()
```

A) A line chart displaying student scores over time.

B) A scatter plot showing the distribution of scores.

C) A bar chart with students' names on the x-axis and their scores on the y-axis.

D) A pie chart representing the percentage of total scores.

Answer: C

Explanation: The code uses plt.bar() to create a bar chart where the x-axis labels are student names and the y-axis values are their scores. This visualization allows for a clear comparison of individual performance.

Q.48 Examine the following code that scrapes a webpage. What is its primary purpose?

```
import requests
from bs4 import BeautifulSoup
url = "http://example.com"
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')
titles = soup.find_all('h1')
for title in titles:
print(title.get_text())
```

A) To extract all hyperlinks from the webpage.

B) To retrieve and print the text content of all <h1> elements on the page.

C) To download images from the webpage.

D) To analyze the metadata of the webpage.

Answer: B

Explanation: The code makes an HTTP GET request to the specified URL, parses the HTML using BeautifulSoup, finds all <h1> tags, and prints their text. This is useful for extracting main headings from a webpage.

Q.49 Consider the following recursive function for reversing a string. What is the output when executed with the input "School"?

```
def recursive_reverse(s):
if len(s) == 0:
return s
else:
return recursive_reverse(s[1:]) + s[0]
print(recursive_reverse("School"))
```

A) "loohcS"

B) "loohc"

C) "loohcS "

D) "School"

Answer: A

Explanation: The function works by recursively removing the first character and then concatenating it at the end. For "School", the final reversed string is "loohcS".

Q.50 A data scientist uses the following code to train and evaluate a decision tree classifier on the Iris dataset. What does the printed accuracy score represent?

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
iris = load_iris()
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.3, random_state=42)
clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)
predictions = clf.predict(X_test)
print(accuracy_score(y_test, predictions))
```

A) The percentage of misclassified samples in the training set.

B) The percentage of correctly classified instances in the test set.

C) The overall error rate of the model on the test set.

D) The percentage of correctly classified samples in both the training and test sets.

Answer: B

Explanation: The code uses accuracy_score to compare the true labels (y_test) with the predicted labels (predictions) on the test set. The printed value represents the proportion of test samples that were correctly classified by the decision tree classifier.

Consolidated Answer Key

Q	Ans	Q	Ans	Q	Ans	Q	Ans	Q	Ans
1	B	**2**	C	**3**	C	**4**	C	**5**	B
6	B	**7**	B	**8**	C	**9**	C	**10**	B
11	B	**12**	B	**13**	B	**14**	A	**15**	B
16	C	**17**	C	**18**	B	**19**	B	**20**	B
21	B	**22**	A	**23**	B	**24**	B	**25**	C
26	C	**27**	B	**28**	B	**29**	B	**30**	C
31	B	**32**	A	**33**	A	**34**	A	**35**	B
36	B	**37**	B	**38**	A	**39**	A	**40**	B
41	B	**42**	A	**43**	B	**44**	B	**45**	B
46	C	**47**	C	**48**	B	**49**	A	**50**	B

School Connect Online

SPACE FOR ROUGH WORK