Assignment Code: DA-AG-015

Boosting Techniques | **Assignment**

Instructions: Carefully read each question. Use Google Docs, Microsoft Word, or a similar tool to create a document where you type out each question along with its answer. Save the document as a PDF, and then upload it to the LMS. Please do not zip or archive the files before uploading them. Each question carries 20 marks.

Total Marks: 200

Question 1: What is Boosting in Machine Learning? Explain how it improves weak learners.

Answer:
Boosting is an ensemble learning technique that combines multiple weak learners to create a strong predictive model. A weak learner is a model that performs slightly better than random guessing.
Boosting works sequentially:
Train the first model on the dataset.
Identify incorrectly predicted samples.
Give higher importance (weights) to misclassified data.
Train the next model focusing more on difficult samples.
Combine all models using weighted voting or summation.
Thus, each new model learns from previous mistakes, improving overall accuracy and reducing bias.

Question 2: What is the difference between AdaBoost and Gradient Boosting in terms of how models are trained?

Answer:

Feature	AdaBoost	Gradient Boosting
Training method	Uses sample weights	Uses gradient descent
Error handling	Focuses on misclassified samples	Minimizes loss function
Learning process	Reweights data	Fits residual errors
Complexity	Simpler	More flexible & powerful
Loss optimization	Not explicit	Explicit loss minimization

Question 3: How does regularization help in XGBoost?

Answer: Regularization in XGBoost prevents overfitting by penalizing complex models.

It uses:
L1 Regularization (alpha) → reduces unnecessary features
L2 Regularization (lambda) → controls large weights

Benefits:
Reduces model complexity
Improves generalization
Prevents overfitting
Produces stable predictions

Question 4: Why is CatBoost considered efficient for handling categorical data?

Answer:

CatBoost handles categorical variables automatically without manual encoding.

Key reasons:
Uses ordered target encoding
Prevents target leakage
Handles categorical features internally
Requires minimal preprocessing
Reduces overfitting
Therefore, it performs well on datasets with many categorical features.

Question 5: What are some real-world applications where boosting techniques are preferred over bagging methods?

Boosting is preferred when high prediction accuracy is required.

Answer:

Examples:
Credit risk prediction
Fraud detection
Medical diagnosis
Customer churn prediction
Recommendation systems
Loan default prediction
Boosting reduces bias and captures complex patterns better than bagging.

Datasets:

- Use `sklearn.datasets.load_breast_cancer()` for classification tasks.
- Use `sklearn.datasets.fetch_california_housing()` for regression tasks.

Question 6: Write a Python program to:

- Train an AdaBoost Classifier on the Breast Cancer dataset
- Print the model accuracy

(*Include your Python code and output in the code box below.*)

Answer:

```
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import accuracy_score
data = load_breast_cancer()
X, y = data.data, data.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
model = AdaBoostClassifier(n_estimators=50)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
```

Question 7: Write a Python program to:

- Train a Gradient Boosting Regressor on the California Housing dataset
- Evaluate performance using R-squared score

(*Include your Python code and output in the code box below.*)

Answer:

```
from sklearn.datasets import fetch_california_housing
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import r2_score
data = fetch_california_housing()
X, y = data.data, data.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
model = GradientBoostingRegressor()
model.fit(X_train, y_train)
pred = model.predict(X_test)
print("R2 Score:", r2_score(y_test, pred))
```

Question 8: Write a Python program to:

- Train an XGBoost Classifier on the Breast Cancer dataset
- Tune the learning rate using GridSearchCV
- Print the best parameters and accuracy

(*Include your Python code and output in the code box below.*)

Answer:

```
from xgboost import XGBClassifier
from sklearn.model_selection import GridSearchCV
model = XGBClassifier(use_label_encoder=False, eval_metric='logloss')
param_grid = {
    'learning_rate':[0.01,0.1,0.2],
    'n_estimators':[50,100]
}
grid = GridSearchCV(model, param_grid, cv=3)
grid.fit(X_train, y_train)
print("Best Params:", grid.best_params_)
print("Best Accuracy:", grid.best_score_)
```

Question 9: Write a Python program to:

- Train a CatBoost Classifier
- Plot the confusion matrix using `seaborn`

(*Include your Python code and output in the code box below.*)

Answer:

```
from catboost import CatBoostClassifier
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
model = CatBoostClassifier(verbose=0)
model.fit(X_train, y_train)
pred = model.predict(X_test)
cm = confusion_matrix(y_test, pred)
sns.heatmap(cm, annot=True, fmt='d')
plt.title("Confusion Matrix")
plt.show()
```

Question 10: You're working for a FinTech company trying to predict loan default using customer demographics and transaction behavior.
The dataset is imbalanced, contains missing values, and has both numeric and categorical features.

Describe your step-by-step data science pipeline using boosting techniques:

- Data preprocessing & handling missing/categorical values
- Choice between AdaBoost, XGBoost, or CatBoost
- Hyperparameter tuning strategy
- Evaluation metrics you'd choose and why
- How the business would benefit from your model

(*Include your Python code and output in the code box below.*)

Answer:

Data Preprocessing
Handle missing values (mean/median).
Encode categorical features (use CatBoost to handle automatically).
Split data into train and test sets.

Use CatBoost or XGBoost because they handle imbalanced data well and give high accuracy.

Tune learning_rate, depth, and n_estimators using GridSearchCV.

4
Use ROC-AUC, Precision, Recall, F1-score (better than accuracy for imbalanced data).

Reduces loan default risk.
Improves credit approval decisions.
Minimizes financial losses.