Midterm review

CS589

Summer 2024

TA: Mohammadreza Teymoorianfard

Supervised Learning and k-Nearest Neighbor (k-NN) Algorithm

Supervised Learning Overview:

- Collect a dataset of images and labels.
- Use Machine Learning to train an image classifier.
- Evaluate the classifier on a withheld set of test images.

k-Nearest Neighbor (k-NN) Classifier:

- **Process:**
    - For each test image, find the k nearest images in the training set.
    - The labels of these k nearest images are used to determine the label of the test image.
- **Example Dataset:** CIFAR-10 with 10 labels, 50,000 training images, and 10,000 test images.
- **Distance Metrics:**
    - Common choices are L1 (Manhattan) and L2 (Euclidean) distances.
- **Hyperparameters:**
    - Important to choose the right distance metric and the value of k.
    - Cross-validation is used to tune these hyperparameters effectively.

Linear Classification:

- **Parametric Approach:**
  - Images represented as arrays of numbers.
  - Use a linear function to map these arrays to class scores.
- **Example Calculation:**
  - Given weights (W) and image data, calculate class scores.
  - Use these scores in a loss function to optimize W.

Loss Functions:

- **Multiclass SVM Loss:**
  - Calculate the loss based on the margin between the correct class score and other class scores.
- **Softmax Classifier:**
  - Use softmax function to convert scores to probabilities.
  - Minimize the negative log likelihood of the correct class.

Regularization:

- Prevent the model from having too much flexibility.
- Helps avoid overfitting by adding a penalty term to the loss function.

Sample Midterm Question with Answer

Question: Given the following training examples and their scores calculated with some weights W:

Example	Class Label	Scores (Cat, Frog, Car)
1	Cat	(3.2, 5.1, -1.7)
2	Frog	(4.9, 1.3, 2.0)
3	Car	(-3.1, 2.5, 2.2)

1- Calculate the Multiclass SVM loss for each example.
2- Compute the total training loss as the mean of the individual losses.

Answer:

1. **Multiclass SVM Loss Calculation:**
   - For Example 1 (Cat):

$$\text{Loss}_1 = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

$$= \max(0, 5.1 - 3.2 + 1) + \max(0, -1.7 - 3.2 + 1)$$

$$= \max(0, 2.9) + \max(0, -3.9)$$

$$= 2.9 + 0 = 2.9$$

For Example 2 (Frog):

$$\text{Loss}_2 = \max(0, 4.9 - 1.3 + 1) + \max(0, 2.0 - 1.3 + 1)$$

$$= \max(0, 3.6) + \max(0, 1.7)$$

$$= 3.6 + 1.7 = 5.3$$

For Example 3 (Car):

$$\text{Loss}_3 = \max(0, -3.1 - 2.2 + 1) + \max(0, 2.5 - 2.2 + 1)$$

$$= \max(0, -4.3) + \max(0, 1.3)$$

$$= 0 + 1.3 = 1.3$$

Total Training Loss:

$$L = \frac{1}{N}\sum_{i=1}^{N} \text{Loss}_i$$

$$= \frac{1}{3}(2.9 + 5.3 + 1.3) = \frac{9.5}{3} = 3.17$$

Neural Networks

Question: You are training a neural network with the following configuration:

- Input layer with 3072 neurons (corresponding to 32x32x3 images).
- Hidden layer with 100 neurons using ReLU activation.
- Output layer with 10 neurons (one for each class).

1. Explain the importance of using an activation function like ReLU in the hidden layer.
2. Describe what could happen if we use a zero-centered activation function such as tanh in the hidden layer.
3. Given a small batch of activations at a layer during training, describe how Batch Normalization would be applied.

Answer:

1. **Importance of ReLU Activation:**
   - ReLU (Rectified Linear Unit) computes max(0,x)max(0,x), which introduces non-linearity into the network. This non-linearity allows the network to learn complex patterns and decision boundaries that a linear classifier cannot. ReLU is computationally efficient and helps networks converge faster compared to sigmoid or tanh functions.
2. **Using Tanh Activation:**
   - Tanh squashes input values to the range [-1, 1], making it zero-centered, which is beneficial for having balanced gradients. However, tanh can still suffer from the vanishing gradient problem when neurons become saturated, leading to very small gradients and slow learning.

Answer:

3. Applying Batch Normalization:

- Batch Normalization normalizes the activations of each layer by scaling and shifting them to have zero mean and unit variance. This is done by computing the empirical mean and variance of the activations for each batch and then normalizing each activation:

$$x\_hat = x - \mu / sqrt(\sigma^2 + \epsilon)$$

- where μ is the mean and $\sigma 2$ is the variance of the batch, and ϵ is a small constant to prevent division by zero. This normalization helps improve gradient flow, allows for higher learning rates, and acts as a regularizer, reducing the need for dropout.

Decision Trees

Overview:

- **Definition:** A decision tree is a model that extracts classification rules to classify instances using IF-ELSE statements.
- **Interpretability:** Highly interpretable by humans, mimics human decision-making.

Key Concepts:

- **Components:**
  - Root node
  - Decision nodes
  - Leaf/terminal nodes
- **Process:**
  - Perform tests on attributes of an instance.
  - Make a decision/prediction about the class of the instance.

Information Gain:

- **Entropy:** Measures the homogeneity of a set of instances.
- **Calculation:**
  - Entropy I(D)=−∑pi log2(pi) I(D) = −∑p ilog2(pi)
  - Information Gain: GainA(D)= I(D) − InfoA(D)
  - **Example:** Testing different attributes like weather conditions in predicting if a person will play tennis.

Information Gain Ratio:

- Adjusts Information Gain to account for attributes with many possible values.
- **Calculation:** Gain_Ratio(A,D)=GainA(D) / Split_Info(A)

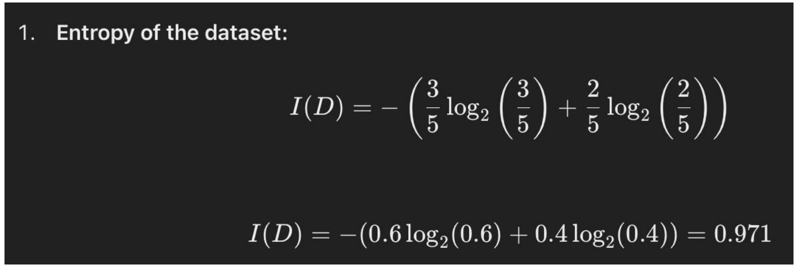
Gini Impurity:

- **Definition:** Measures the impurity of a dataset.
- **Calculation:** Gini(D)=1−∑(pi^2)
- **Example:** Testing attributes like weather to determine the Gini impurity for predicting if a person will play tennis.

Question: Given the following dataset for predicting whether a person will buy a computer based on three attributes: Student, Age, and Credit_Score, construct a decision tree using the ID3 algorithm and calculate the Information Gain for each attribute. Determine which attribute should be tested first.

Instance	Student	Age	Credit_Score	Will_Buy_Computer
1	yes	young	Regular	yes
2	yes	Middle_Age	Excellent	yes
3	no	Young	Excellent	no
4	no	Older	Regular	no
5	yes	Older	Excellent	yes

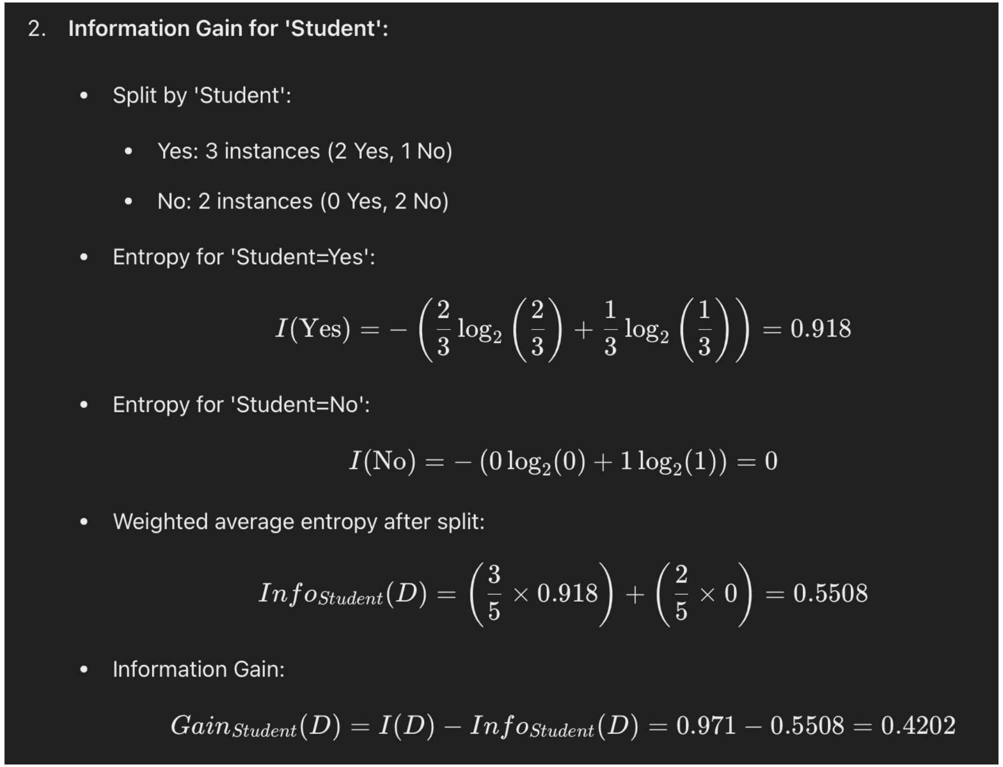
Calculate the entropy of the dataset.
Calculate the Information Gain for the attribute 'Student'.
Based on the calculated Information Gain, determine which attribute should be tested first



1. **Entropy of the dataset:**

$$I(D) = -\left(\frac{3}{5}\log_2\left(\frac{3}{5}\right) + \frac{2}{5}\log_2\left(\frac{2}{5}\right)\right)$$

$$I(D) = -(0.6\log_2(0.6) + 0.4\log_2(0.4)) = 0.971$$



2. **Information Gain for 'Student':**
   - Split by 'Student':
     - Yes: 3 instances (2 Yes, 1 No)
     - No: 2 instances (0 Yes, 2 No)
   - Entropy for 'Student=Yes':

$$I(\text{Yes}) = -\left(\frac{2}{3}\log_2\left(\frac{2}{3}\right) + \frac{1}{3}\log_2\left(\frac{1}{3}\right)\right) = 0.918$$

   - Entropy for 'Student=No':

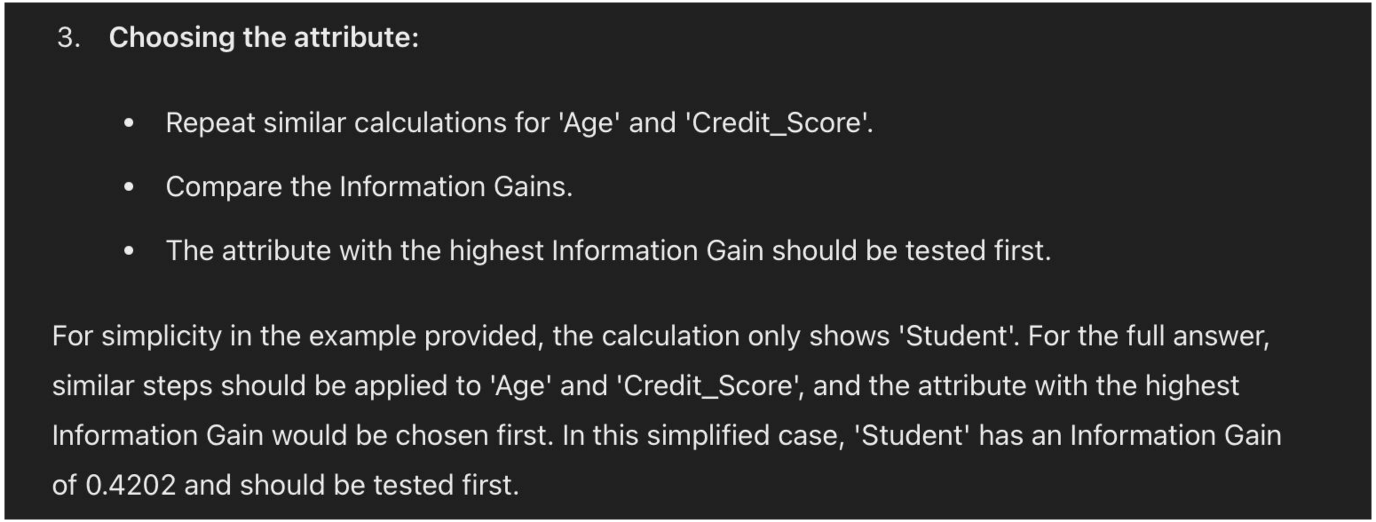
$$I(\text{No}) = -\left(0\log_2(0) + 1\log_2(1)\right) = 0$$

   - Weighted average entropy after split:

$$Info_{Student}(D) = \left(\frac{3}{5} \times 0.918\right) + \left(\frac{2}{5} \times 0\right) = 0.5508$$

   - Information Gain:

$$Gain_{Student}(D) = I(D) - Info_{Student}(D) = 0.971 - 0.5508 = 0.4202$$



3. **Choosing the attribute:**

    - Repeat similar calculations for 'Age' and 'Credit_Score'.
    - Compare the Information Gains.
    - The attribute with the highest Information Gain should be tested first.

For simplicity in the example provided, the calculation only shows 'Student'. For the full answer, similar steps should be applied to 'Age' and 'Credit_Score', and the attribute with the highest Information Gain would be chosen first. In this simplified case, 'Student' has an Information Gain of 0.4202 and should be tested first.

Naive Bayes Classifier:

Definition: A probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions between features.

Formula:

$$P(y_i|x_1, x_2, \ldots, x_n) = \frac{P(y_i)P(x_1, x_2, \ldots, x_n|y_i)}{P(x_1, x_2, \ldots, x_n)}$$

Simplified to:

$$P(y_i|x_1, x_2, \ldots, x_n) \propto P(y_i)\prod_{k=1}^{n} P(x_k|y_i)$$

Application Example:

- **Spam Detection:** Given a new email, calculate the probability of it being spam or not spam based on the occurrence of certain words (features).

Laplace Smoothing:

- **Problem:** Zero probabilities for unseen words in the training set.
- **Solution:** Add one to each count (Laplace smoothing).

Naive Bayes with Numeric Attributes:

- **Discretization:** Convert continuous variables to categorical (e.g., temperature ranges).
- **Assumption of Distribution:** Assume a specific distribution (e.g., Gaussian) for continuous variables and estimate parameters from the training data.

Question: Imagine you are a spam filter and you have to decide if an email is spam or not spam. You are given the following information about emails:

1. Most spam emails contain words like "win", "free", and "prize".
2. Non-spam emails usually contain words like "meeting", "schedule", and "project".

You receive a new email that contains the words "free" and "meeting". Based on your knowledge, what is the most likely classification for this email? Explain your reasoning.

Answer:

1. **Identify Features:** The email contains the words "free" and "meeting".
2. **Consider Known Information:**
   - The word "free" is commonly found in spam emails.
   - The word "meeting" is commonly found in non-spam emails.
3. **Reasoning:**
   - Since the email contains a word ("free") that is highly associated with spam, it might lean towards being classified as spam.
   - However, the presence of the word "meeting," which is associated with non-spam, suggests that it could be a legitimate email.
4. **Conclusion:**
   - Given that "free" is a strong indicator of spam and "meeting" is a strong indicator of non-spam, the classification could go either way.
   - If the filter relies more heavily on words like "free" to classify spam, it might classify this email as spam.
   - Conversely, if it gives more weight to "meeting" as a non-spam indicator, it might classify it as non-spam.

Therefore, the most likely classification depends on how the filter weighs these words. If the filter considers "free" as a stronger indicator, the email will be classified as **spam**. If "meeting" is weighed more heavily, it will be classified as **not spam**.

Ensemble Methods

Wisdom of Crowds:

- **Concept:** Knowledge that emerges from collective decisions is often more accurate than that of any individual, even experts.
- **Four Conditions:**
  1. **Diversity of Opinion:** Each individual has private information.
  2. **Independence:** Individuals' opinions are not influenced by those around them.
  3. **Decentralization:** Decisions are made based on local knowledge without central control.
  4. **Aggregation:** Mechanism to combine private judgments into a collective decision.

Ensemble Learning in Machine Learning:

- **Definition:** Combining multiple models to improve overall performance.
- **Example:** Jelly Beans in a Jar experiment showed collective guesses are more accurate.

Multiple Models:

- Train multiple models independently.
- **Majority Vote:** Aggregate predictions to make final decision.

Conditions for Effective Ensembles:

- **Accuracy:** Individual classifiers should have an error rate less than 50%.
- **Diversity:** Errors made by classifiers should be independent.

Bagging (Bootstrap Aggregating):

- Create multiple bootstrap datasets by sampling with replacement.
- Train a model on each dataset.
- Combine models using majority voting to reduce variance.

Random Forests:

- Train multiple decision trees on different bootstrap datasets.
- Use random subsets of features for splitting nodes.
- Combine predictions using majority voting.

Boosting:

- Sequentially train weak classifiers.
- Each classifier focuses on instances misclassified by previous classifiers.
- Combine predictions to reduce bias and variance.

Question: Imagine you are explaining ensemble methods to a friend using a real-life scenario. You decide to use the example of making a group decision on where to eat dinner. Explain how the principles of ensemble learning (diversity, independence, decentralization, and aggregation) can help the group make a better decision.

Answer:

1. **Diversity of Opinion:**
   - Each person in the group suggests different restaurants based on their unique tastes and preferences. This diversity ensures a wide range of options is considered.
2. **Independence:**
   - Everyone makes their suggestion without being influenced by others. This independence ensures that each opinion is based on personal preference rather than groupthink.
3. **Decentralization:**
   - There is no single person deciding for the group. Instead, each person contributes their suggestion, allowing for a more democratic process.
4. **Aggregation:**
   - The group uses a method to combine all suggestions, such as voting or ranking the restaurants. This aggregation ensures that the final decision reflects the collective preference of the group.

Real-Life Application:

- Each friend writes down their top three restaurant choices independently (independence).
- All choices are collected, ensuring a variety of options (diversity).
- No single person has control over the decision (decentralization).
- The group votes on the options, and the restaurant with the most votes is chosen (aggregation).

This process mirrors ensemble learning, where diverse and independent models are combined to make a more accurate and reliable prediction. In this case, the group's collective decision is likely to be more satisfying to everyone compared to a decision made by a single individual.