HW 2

Solution 1

Describe Gaussian Mixture Model clustering. Why is it an instance of the Expectation Maximization method? What are its advantages over the K-Means clustering algorithm?

GMM based clustering

GMM is probabilistic model used for clustering. It assumes that the data is generated from a mixture of several Gaussian distributions.

- There are k components.
- Each component is associated with its mean vector.
- Each component generates data from a Gaussian with mean and covariance matrix.
    - Every data point is generated by picking a component at random with a probability calculated using Bayes' theorem.

Expectation (E-step):

- Estimation of probabilities of data points belonging to a cluster.

Maximization (M-step):

- Parameters (mean, covariance, weight) are re-estimated for every component at based on probabilities computed in E-step.

Iteration:

- E-step and M-step are repeated until convergence, where the change in parameters falls below a certain threshold.

Advantages of GMM over K-Means

- Flexible cluster shapes.
    - Can form elongated ellipsoidal clusters with different orientations and variances.
    - K-Means assumes spherical clusters.
- Cluster overlaps possible.
    - GMM provides soft assignments, probability of data points belonging to each cluster.
    - K-Means provides hard assignments, assigning each data point to a single cluster.
- Robust to outliers.
    - GMM is less sensitive to outliers as it considers the probability of a data point belonging to each cluster.
    - K-Means assigning is done solely based on the nearest centroid.
- Cluster Sizes and Variations.
    - GMM can handle clusters with different sizes and shapes, making it more suitable for complex datasets where clusters may not be well-separated.

Solution 2

Describe a multivariate Gaussian along with its parameters (μ and Σ). What is the geometric interpretation of these two parameters? List some interesting properties of the eigenvalues and eigenvectors of any covariance matrix.

A multivariate normal distribution – a probability distribution that describes a joint distribution described by multiple random variables.

Characterized by 2 main parameters:

- Mean vector (μ):
  - Column vector containing means for each individual random variable in multivariate distribution.
- Covariance matrix (Σ):
  - Real, Positive semi definite matrix
  - describes the relationships between different variables in the multivariate distribution.
  - diagonal elements represent the variances of individual variables.
  - off-diagonal elements represent the covariances between pairs of variables.

Geometric Significance

- Mean vector (μ):
  - Defines the center of the multivariate Gaussian distribution in the d-dim space.
  - serves as the point around which the distribution is centered.
- Covariance matrix (Σ):
  - Real, Positive semi definite matrix
  - Provides information about the spread, shape, and orientation of the distribution in the d-dimensional space, describes how the variables are correlated or independent.
  - In a 2D or 3D space, you can visualize the ellipsoid as an ellipse or an ellipsoid respectively.

Properties of Eigenvalues & Eigenvectors of Σ

- Eigenvalues
  - Variances of the data along the corresponding eigenvector directions
  - Larger eigenvalues indicate higher variance in the data along those directions.
- Eigenvectors
  - Indicate the directions in which the data has the most spread or variance.
  - Form an orthonormal basis for the space spanned by the data.
    - Are orthogonal to each other.
    - Has unit length of magnitude.
- Eigenvalues and Eigenvectors of the covariance matrix provide information about the directions and magnitudes of the principal axes of the ellipsoid.
- The orientation of the ellipsoid in space is determined by the eigenvectors.

Solution 3

Describe the Principal Component Analysis algorithm for dimensionality reduction along with the time complexity of each of its steps. How does it compare against FastMap representation-wise, efficiency-wise, and quality-wise?

- Unsupervised technique for extracting variance structure from high dimensional data.
- An orthogonal projection or transformation of data in low dimensional space such that the variance of projected data is maximized.

Algorithm

- Standardize data.
  - Data is centered by subtracting the mean of each feature.
  - Scaled by dividing by the standard deviation.
  - Ensures that each feature has zero mean and unit variance.
  - Time Complexity: O(ND)
- Compute Covariance Matrix:
  - Captures the relationships and variances between different features.
  - Time Complexity: $O(ND^2)$
- Compute Eigenvectors and Eigenvalues:
  - Eigenvectors represent the principal components.
  - Eigenvalues indicate the amount of variance along each component.
  - Time Complexity: $O(D^3)$
- Sort Eigenvalues:
  - The eigenvalues are arranged in descending order.
  - Determines the order of importance of the principal components.
  - Time Complexity: $O(DlogD)$
- Select Principal Components:
  - Based on the sorted eigenvalues, a subset of the eigenvectors (principal components) is chosen to retain most of the variance in the data.
  - Time Complexity: O(D)
- Transform Data:
  - The data is projected onto the selected principal components. This results in a lower-dimensional representation of the original data.
  - Time Complexity: O(NDD')
- Overall Time Complexity: $O(ND^2 + D^3)$

Where,

N = Number of data points
D = Number of features of data points
D' = Number of features of projections or principal components

Comparison with FastMap

	PCA	**FastMap**
Representation-wise	Linear transformation of points in geometric space	Maps complex objects into Euclidean space preserving pairwise distance
Efficiency-wise	$O(ND^2 + D^3)$ Linear projection when num features is large	O(ND) Pairwise dissimilarity is available and linear transformation is not possible
Quality-wise	Provides probably optimal solution	Doesn't guarantee an optimal solution

Solution 4

What are the steps in the Perceptron Learning algorithm? What do we do when a constraint is violated in any iteration? Should the learning rate for updating the weights be high or low; why? What happens if we try running the Perception Learning algorithm on data that do not have linearly separable positive and negative labels?

Algorithm

- Initialize weights.
    - Start with random or zero weights.
- For each training example
    - Compute the dot product of feature vector and weight.
    - Apply the activation function.
    - If no constraints are violated, then return the weight vector.
    - Else update the weights.
- Update the weights.
    - Pick any violated constraint and use the above step to update the weights.
- Repeat above 2 steps until a stopping criterion is met.

Violation of a Constraint

- When a constraint is violated in any iteration, indicates current prediction for a training example is incorrect.
- Updates its weights to correct the prediction using the Perceptron update rule.

Learning Rate

Learning rate is to be relatively low when a constraint is violated.

- High Learning Rate:
    - The algorithm may overshoot and fail to converge. It may oscillate or even diverge.
- Low Learning Rate:
    - The algorithm may take a long time to converge or get stuck in a local minimum.

Linear Separability

- If the data is not linearly separable, algorithm will not converge.
- May keep updating the weights indefinitely as it is unable to find a hyperplane that perfectly separates the classes.

Solution 5

What is the Constraint Satisfaction Problem (CSP)? Pick a problem of interest in Data Science which can be solved efficiently using CSP search techniques. Describe the problem and the application of CSP techniques on it. Elaborate on one of these techniques.

A Constraint Satisfaction Problem (CSP) is a mathematical problem defined by

- Set of variables
- Domain of possible values for each variable
- Set of constraints that restrict the allowable combinations of values for these variables.

The goal in a CSP is to find an assignment of values to all the variables such that all constraints are satisfied.

Sudoku Puzzle Solver

- **Variables**:
  - Each variable represents a cell in the Sudoku grid.
  - There are 81 variables in total (9 rows x 9 columns).
- **Domain**:
  - Each variable can hold a set of possible values – {1, 2, 3, 4, 5, 6, 7, 8, 9}.
- **Constraints**:
  - **Row Constraint**: Each number can only appear once in each row.
  - **Column Constraint**: Each number can only appear once in each column.
  - **Box (or Sub-Grid) Constraint**: Each number can only appear once in each 3x3 sub-grid.

CSP Techniques

- Constraint Propagation
  - Elimination of values based on known values in the same row, column, or sub-grid.
  - Elimination of hidden singles based on cells with only one possible value.
  - Can reduce the domain of variables.
- Backtracking Algorithm
  - Involves systematically filling in cells, checking constraints, and look-back if a conflict is encountered.
- Minimum Remaining Values (MRV) Heuristic
  - Select the variable with the fewest remaining possible values in its domain.
  - Can help in reducing the branching factor.
- Least Constraining Value (LCV) Heuristic
  - Prioritize values that constrain the search space the least.
  - If assigning a certain value allows for more flexibility in neighboring cells, that value is preferred.

Elaboration on Constraint Propagation:

- For same row, column and sub-grid
  - If a cell already contains a value, remove it from the domains of the cells.
  - If a value is assigned to a cell, remove it from the domains of the cells.
- By iteratively applying these constraints
  - Prunes search space.
  - Reduces the possibilities and making the solving process more efficient.
  - Helps in identifying hidden singles and making logical deductions about the values in the puzzle.