* Entropy = weighted sum of all digits with non zero values regarding the chances of an event occurring

Where H = entropy, P(Ei) = probability of event i and b = encoding base. Entropy reaches max when P(Ei) is the inverse of an integer

* Huffman coding, calculate probabilities of each symbol appearing, then form a tree by combining the symbols that appear least until there is only 2 branches. Then assign codes by adding a 1 to the symbol with lower appearance and 0 to the other
* Avg Huffman code length is P(Ei)\*code length.
* Optimal code length/efficiency is equal to message length times entropy

|  |  |
| --- | --- |
|  |  |

* Numeric coding, calculate probabilities of each message, set them on a scale, then for each successive part of the message, replicate the scale at the width of the value

Chart, scatter chart

Description automatically generated

* To represent a binary number with an interval for example 0101, write as a decimal 0.0101. Then for a lower bound add recurring 0 after it 0.01010, and for the upper bound, add recurring 1 0.01011. Then add by the powers of 2. For the upper bound, use a value of 1 instead of the last 0 that appears before the recurring 1
* A picture containing text

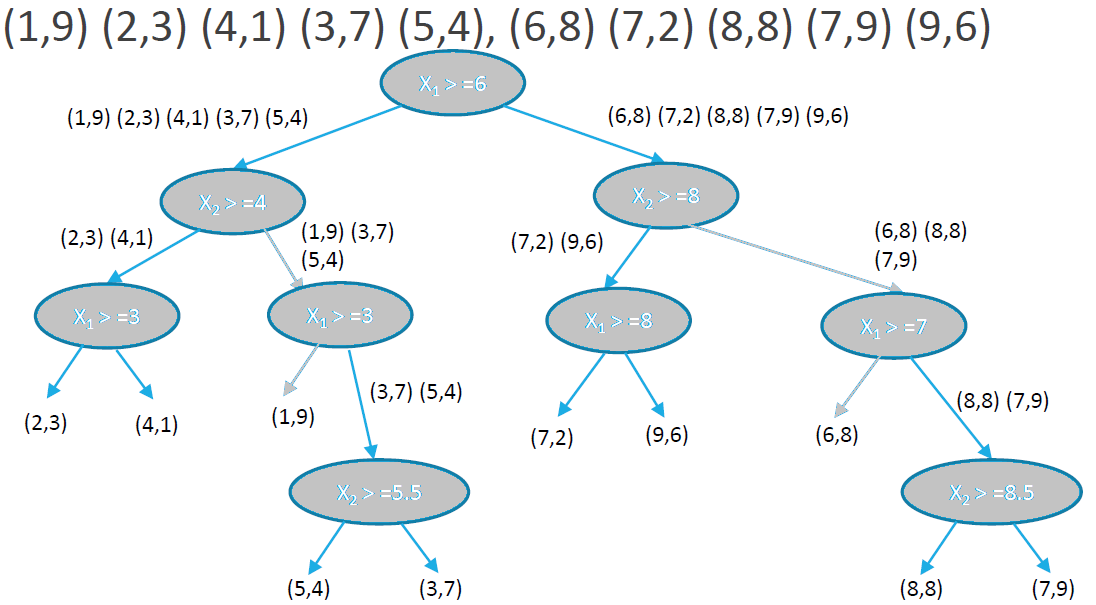
  Description automatically generated
* To find a good, encoded range, convert upper and lower bounds to binary. Truncate to code length and add or remove 1 until the truncated binary is between upper and lower bounds
* Transformation matrices are ordered right to left, last to first. I.e. if T is applied first then R, then the order is RT
* Angle between two vectors v = (a, b, c) and u = (x, y, z),
* Direction normal to two vectors v = (a, b, c) and u = (x, y, z),
* Homogenous matrix to rotate by an angle around an arbitrary normalized direction (x,y,z)
* Text

  Description automatically generated with medium confidence

|  |  |
| --- | --- |
|  |  |

* PCA analysis, calculate mean of all the points, and subtract from the original points to center the values.
* Calculate the covariance matrix by multiplying the matrix of centered points to it’s transpose
* Subtract 𝜆I from the covariance matrix and calculate the determinant of the result. This will lead to a quadratic equation, so find the two possible values of 𝜆
* Then calculate the eigen vectors using the two values of 𝜆 and the following constraints:
* Then form new axes using the two eigen vectors and project the points to the new axes to approximate and group original data
* To form a kd tree of a set of points, draw vertical and horizontal axes alternatingly until all the points are captured.

|  |  |
| --- | --- |
|  |  |



* K-means clustering, given a set of points, choose a number of cluster centers and then divide the points between those centers based on how close they are to the center.
* Then recalculate each center to be equal to the average center between the points it is clustered with. Then re distribute the points again based on the new centers. Repeat this until the centers do not change when recalculated.

A picture containing text

Description automatically generated