N-tuple Classifier

Robert M. Haralick

Computer Science, Graduate Center
City University of New York

Solving Complex Computational Problems

- Break global problem into smaller subproblems
- Each of which can be solved independently
- Optimally solve the subproblems
- Combine the solutions to the subproblems to obtain the solution to the global problem

Decompositions

- Maximize the Dependencies within each of the smaller problems
- Maximize the Independence between each of the smaller problems

Decompositions

- Recursive Decomposition
- Data Decomposition
- Functional Decompositions
- Search Space Decompositions

Decompositions and Optimality

- Sometimes the Solution to the decomposed problem is optimal
- Sometimes the Solution to the decomposed problem is sub-optimal
- The Solution obtained by decomposition can be close to optimal

The Subspace Classifier

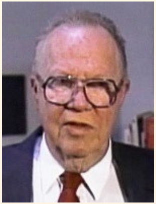
Definition

A Subspace Classifier is one that projects the measurement tuple to one or more subspaces where the projected tuple is processed and then the processed projected tuples are combined in a way to form an assigned classification.

It is typical for the projection operators to be orthogonal projection operators. It is not unusual for the projection operators to be axis aligned.

N-Tuple Method - Bledsoe and Browning -

(a) Bledsoe

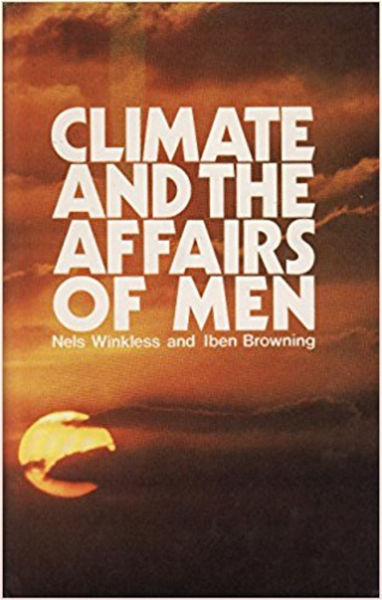


(b) Browning

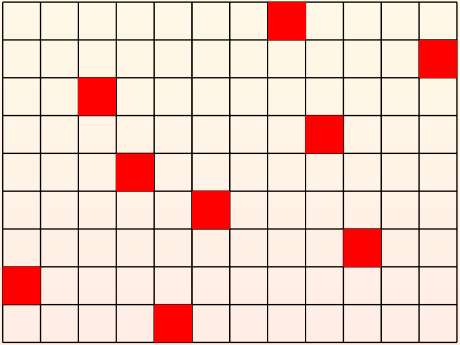
- Developed For Printed Character Recognition
- Each character is contained in an image of $I \times J$ pixels
- Each pixel is a binary 1 or a binary 0
- Designed for table lookup hardware

W.W. Bledsoe and I. Browning, *Pattern Recognition and Reading by Machine*, **Proceeding Eastern Joint Computer Conference**, Boston, 1959, 232-255.

Climate and the Affairs of Men



N-Tuple Method



N Randomly Chosen Pixel Positions

N-Tuple Method

- A small number of pixel positions are randomly selected
- Have multiple sets of such randomly selected pixel positions
- Each of the pixel positions has been thresholded and contains a binary 0 or a binary 1
- Concatenate all the binary values to form a binary number
- Use this number to access an address in a memory array
- For each character class
  - Have as many memory arrays as there are different randomly selected position sets

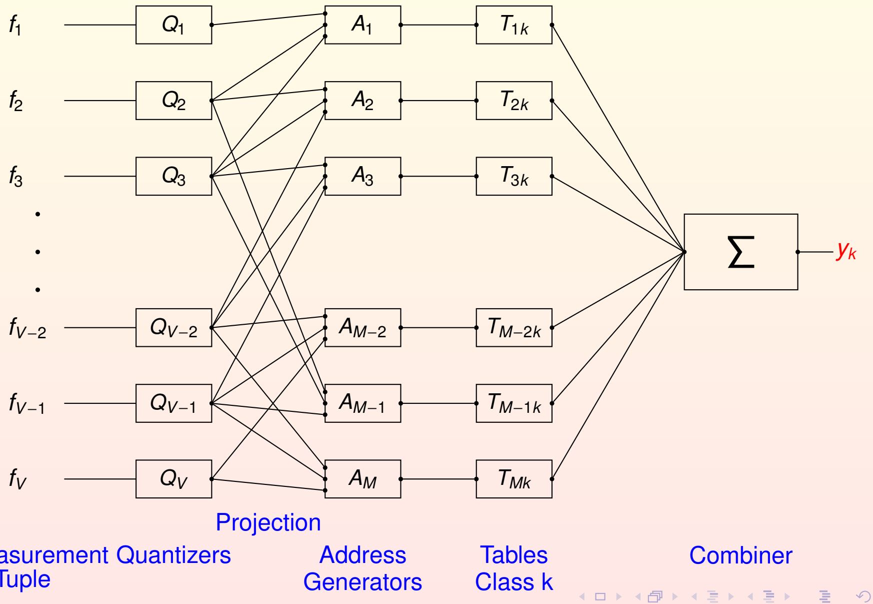
N-Tuple Method

- M pattern sets of N randomly selected pixel positions
- A printed character produces M N-digit binary numbers $b_1, \ldots, b_M$
- K character classes
- T_{mk} lookup table for pattern set m and class k
- $T_{mk}(b_m)$ holds the fraction of times a character in the training set of class k has the binary number b_m for the m^{th} pattern set
- Compute
  - $y_k = \prod_{m=1}^{M} T_{mk}(b_m)$
  - $y_k = \sum_{m=1}^{M} T_{mk}(b_m)$
  - $y_k = \sum_{m=1}^{M} log T_{mk}(b_m)$
- Assign the character to unique class k^*, if there is one, for which $y_{k^*} > 0$ is highest
- Otherwise reserve decision

N-Tuple Method Alternative

- M pattern sets of N randomly selected pixel positions
- A printed character produces M binary numbers $b_1, \ldots, b_M$
- K character classes
- T_m lookup table for pattern set m
- $T_m(b_m)$ holds the set of classes associated with the binary number b_m for the m^{th} pattern set
- Compute
  - $Y = \cap_{m=1}^{M} T_m(b_m)$
  - Assign the character to unique class k^*, if there is one, where $k^* \in Y$ and $|Y| = 1$
  - Otherwise reserve decision

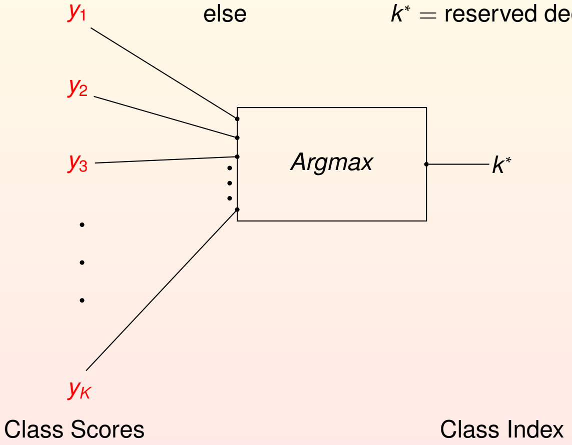
The N-tuple Calculation for Class k



The N-tuple Class Index Generator

if $y_{k^*} > y_i + \epsilon,\ i \neq k^*),\ k^* = Argmax\{y_1, y_2, \ldots, y_K\}$

else $\qquad k^* =$ reserved decision



The Need For The Indexed Tuple

Consider the five dimensional measurement vector (a, b, c, d, e) where

- a is the value produced by feature f_1
- b is the value produced by feature f_2
- $\vdots$
- e is the value produced by feature f_5

The Need For the Indexed Tuple

- Project the measurement vector (a, b, c, d, e) to the third and fifth feature
- The resulting tuple is (c, e).
- But now we have lost from which features c and e came.

In the database world, every value comes from a field and the connection between field and value is never lost.

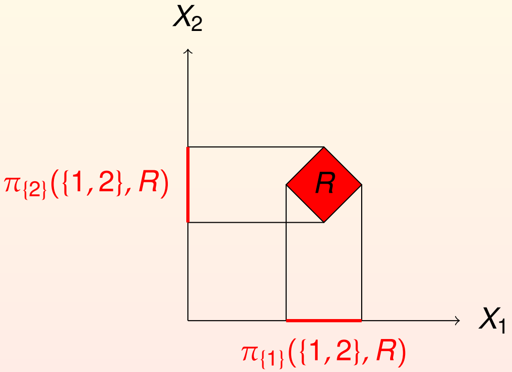
The Indexed Tuple

- Index Sets serve as Field Names
- The tuple (a, b, c, d, e) is written as $(\{1, 2, 3, 4, 5\}, (a, b, c, d, e))$
- (c, d) is written as $(\{3, 4\}, (c, d)$
- (a, b, e) is written as $\{1, 2, 5\}, (a, b, e))$
- A tuple list R=<(a,b,e),(q,r,s),(t,x,z)> is written as $(\{1, 2, 5\}, R)$
  - First component is an index set for the features
  - Second component is a set of tuples
  - Each component of a tuple is the value of the corresponding indexed features

The Tuple Projection Operator

- Suppose that S is a tuple list with respect to the index set I
  - (I, S)
- Let $J \subset I$.
- The projection of (I, S) from the space indexed by I to the subspace indexed by J
  - $\pi_J(I, S) = (J, R)$

Projection



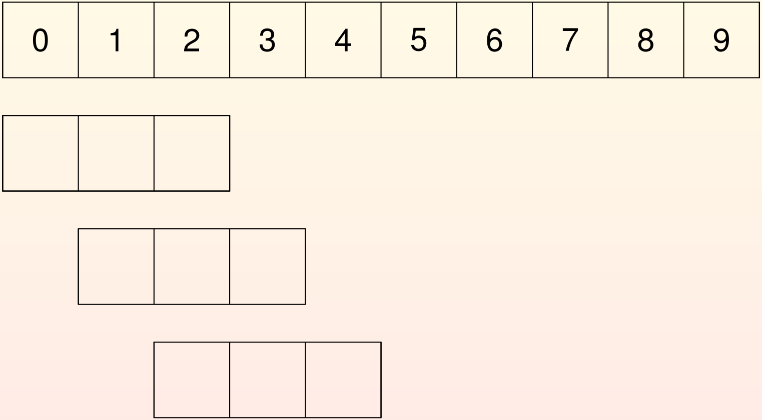
N-tuple Method: 2 Class Case

- Index Set $I = \{1, \ldots, V\}$
- $f_1, \ldots, f_V$ are the V quantized features
- $L_1, \ldots, L_V$ are the corresponding range sets
  - $X_v \in L_v$, $v = 1, \ldots, V$
  - Measurement Space $\mathcal{M} = \times_{i \in I} L_i$
- $< (I, x_1), \ldots, (I, x_Z) \mid x_z \in \mathcal{M} >$ Training Set for Class 1
- $< (I, y_1), \ldots, (I, y_Z) \mid y_z \in \mathcal{M} >$ Training Set for Class 2
- $J_1, \ldots, J_M \subset I$ are the M pattern sets
- $\pi_{J_m}(I, x_z) = (J_m, u_z)$, $u_z \in \times_{j \in J_m} L_j$
- $\pi_{J_m}(I, y_z) = (J_m, w_z)$, $w_z \in \times_{j \in J_m} L_j$

N-tuple Method

- Tables For Each Index Set and Class
  - $T_{m1}(J_m, u) = |\{z \mid (J_m, u) = \pi_{J_m}(I, x_z)\}|/Z$
  - $T_{m2}(J_m, w) = |\{z \mid (J_m, w) = \pi_{J_m}(I, y_z)\}|/Z$
- Scores For Each Class
  - $S_k(I, q) = \sum_{m=1}^{M} T_{mk}(\pi_{J_m}(I, q))$
- Identification
  - Assign class 1 if $S_1(I, q) > S_2(I, q) + \epsilon$
  - Assign class 2 if $S_2(I, q) > S_1(I, q) + \epsilon$
  - Otherwise Assign reserve decision

Scanning N-tuple Classifier



S.M. Lucas and A. Amiri, *Recognition of Chain-coded Handwritten Characters With the Scanning N-Tuple Method*, **Electronics Letters**, vol. 31, no. 24, 1995, pp. 2088-2089.

Scanning N-tuple Classifier Index Sets

$$
\begin{aligned}
J_0 &= \{0, 1, 2\} \\
J_1 &= \{1, 2, 3\} \\
&\vdots \\
J_9 &= \{7, 8, 9\}
\end{aligned}
$$

N-tuple Subspace Classifier Summary

$\mathcal{M}$	Measurement Space
K	Number of classes
$\mathcal{J}$	Collection of M Index Sets
$\mathcal{T}$	Collection of MK tables
T	Classification Function

The one stage subspace classifier $\mathcal{C}$ using quantized features can be written as a 5-tupleZ

$$\mathcal{C} = (\mathcal{M}, \mathcal{J}, \mathcal{T}, K, M, T)$$

Universal Approximator Conjecture

The N-tuple Subspace Classifier is a kind of universal approximator.

Conjecture

Let $\mathcal{M} = \bigtimes_{v=1}^{V} L_d$ *be the V-dimensional measurement space. Let* $f : \mathcal{M} \rightarrow \{0, 1\}$ *be a given classification function associating every measurement tuple with a class index* 0 *or a* 1. *Let P be a probability distribution on* $\mathcal{M}$. *If f is 'zzz' simple, then for every* $\epsilon > 0$, *there exists* $K << V$ *and* $M < \begin{pmatrix} V \\ K \end{pmatrix}$ *and a two class N-tuple subspace classifier* $\mathcal{C} = (\mathcal{M}, \mathcal{J}, \mathcal{T}, K, M, T)$ *such that*

$$P(\{x \in \mathcal{M} \mid f(x) \neq T(x)\} < \epsilon$$

Neural Net and N-tuple Comparison

- Neural Networks
  - Neural Net signal lines take floating point values
  - Each unit computes a weighted linear combination of its input
    - The weights are initially set at random
  - The linear combination is input to an activation function
    - The activation function has bounded output
    - And is non-linear
  - There can be multiple units in any one layer
  - The original form of the neural network had only one inner layer
  - Layers can be cascaded
  - The geometry of the cascading is hand designed
  - There is an iterative training algorithm that optimizes the weights for a given data set

Neural Net and N-tuple Comparison

- N-tuple Classifier
  - The first stage in any N-tuple Classifier unit is a quantization
  - Each unit has a table look up memory to produce an output from the quantized values which are used to form an address
  - The values in the table lookups are determined in one pass through the data
  - There can be multiple units in any one layer
  - The original form of the N-tuple classifier had only one inner layer
  - Layers can be cascaded
  - The geometry of the cascading is initiated at random
  - There is an iterative algorithm for optimizing the index sets defining the projections
    - Therefore the geometry of the cascading is automated
  - There is an iterative algorithm for optimizing the quantization functions

Neural Net and N-tuple Classifier Comparison

- The Neural Network
  - A choice of activation function must be made for each unit in the Neural Network
  - Usually the same activation function is employed in each unit, but the theory does not require this
  - The activation function has parameters which must be set by design
  - The activation function is non-linear
  - The activation function bounds and compresses the unit's output
- The N-tuple Classifier
  - The quantization function is different for the lines on each layer
  - The quantization function is non-linear
  - The quantization function bounds and compresses the values so that they can be used for form an address to the unit's memory

Neural Network and N-tuple Classifier Comparison

Modulo the quantization

- The N-tuple classifier can do everything the Neural Network does
- But the N-tuple classifier is more general

N-tuple vs Decision Tree

Any decision tree can be put in N-tuple form.

Definition

A Decision tree is a classifier whose structural form is a tree.

- Each node of the tree corresponds to a mutually exclusive subset of measurement space
- The nodes of the tree are either decision nodes or leaf nodes
- At each decision node of the tree a distinction is made that partitions its subset of measurement space
- Each leaf node is associated with an assigned class

Advantages

- Understandable rules
- Quick On-line computation
- Continuous or categorical variables.
- Provide a clear indication of which dimensions are most relevant for accurate classification

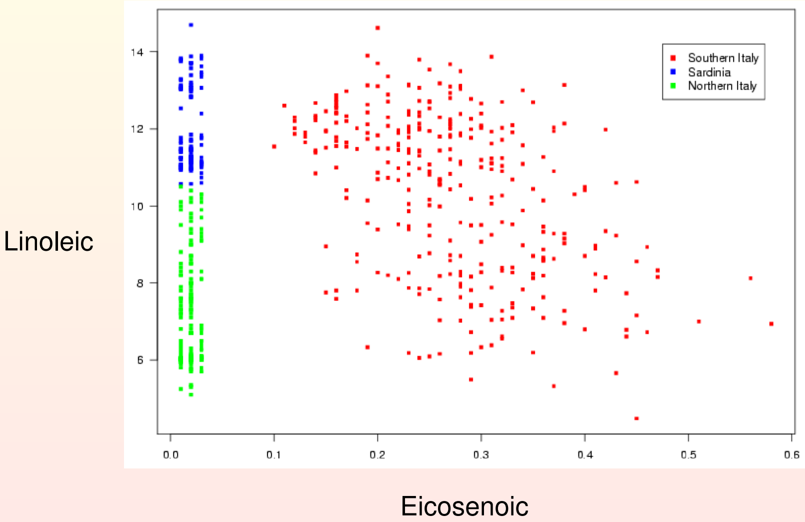
Disjunction of Conjunctions

- On any branch down the tree, the decision region is specified by the conjunction of the constraints of the nodes in the branch
- There are many branches, each of which represents a disjunction of these conjunctions

Where Did the Olives Come From?

- Classes
  - Northern Italy
  - Southern Italy
  - Sardinia
- Fatty Acid Measurements
  - Eicosenoic: x_1
  - Linoleic: x_2

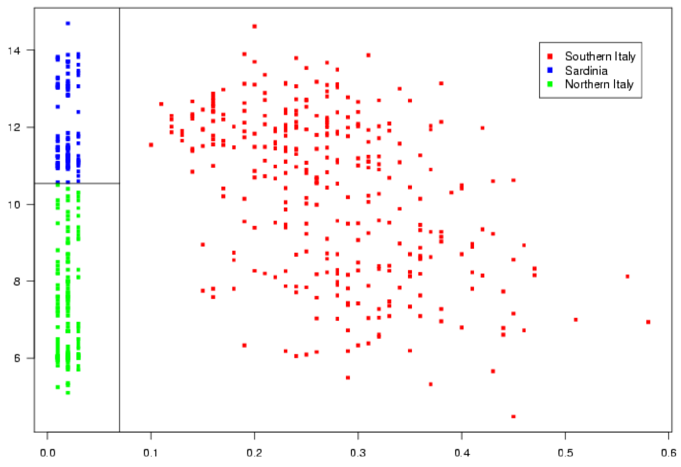
Olives



Linoleic
14
12
10
8
6
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Southern Italy
Sardinia
Northern Italy
Eicosenoic

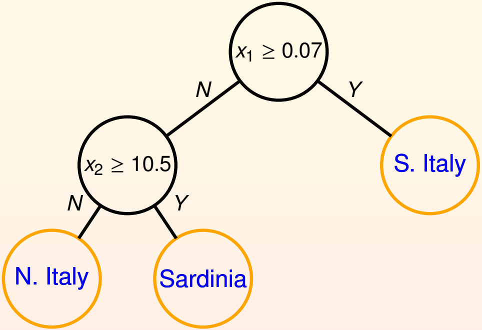
Olives

Linoleic

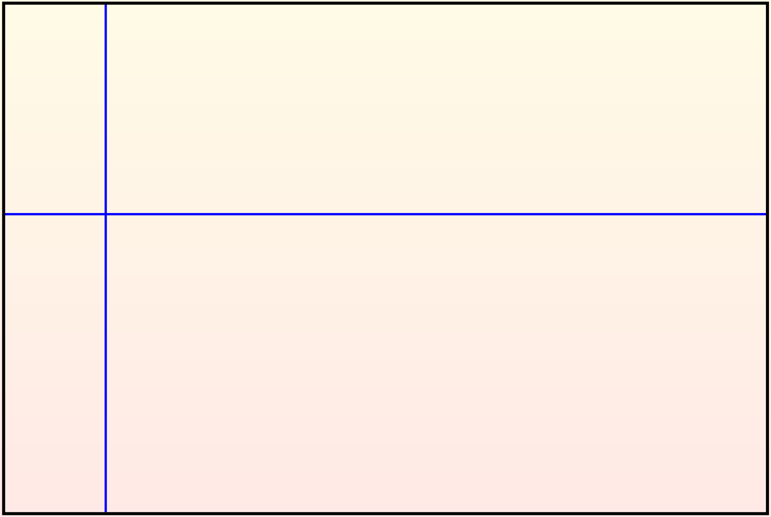


Eicosenoic

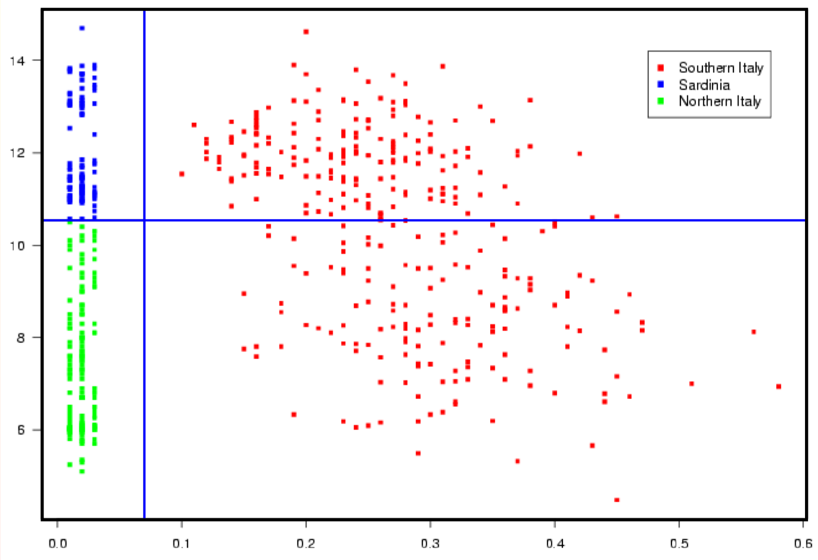
Decision Tree



Binary Quantization

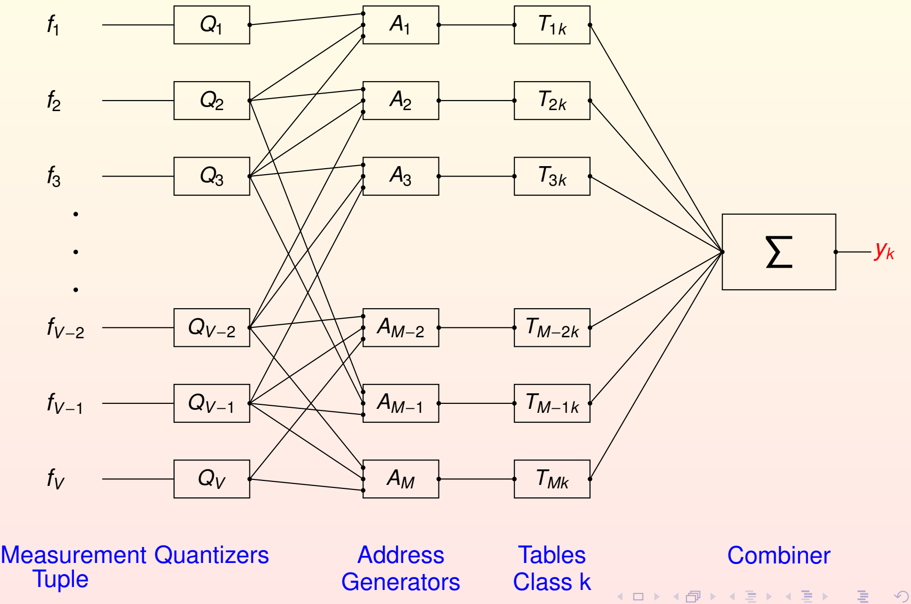


Olives

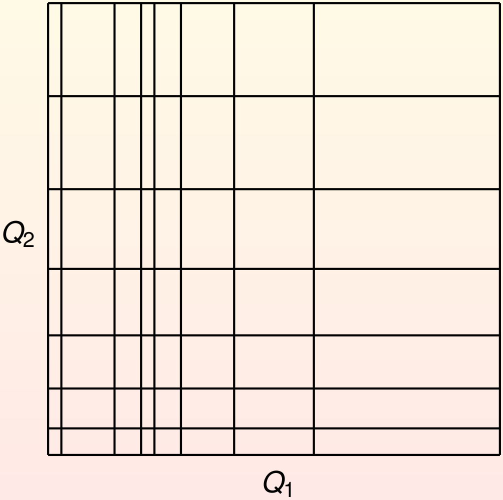


Southern Italy
Sardinia
Northern Italy
14
12
10
8
6
0.0
0.1
0.2
0.3
0.4
0.5
0.6

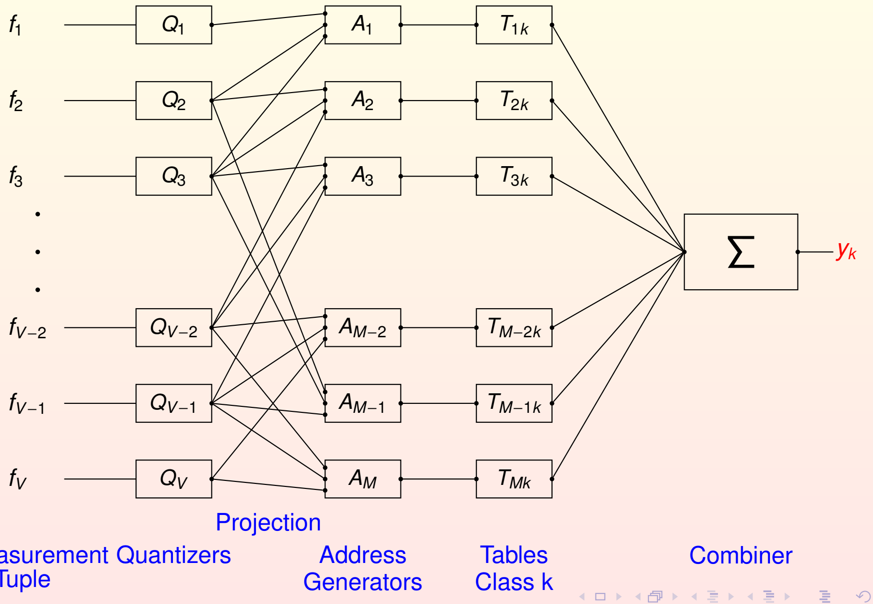
The N-tuple Calculation for Class k



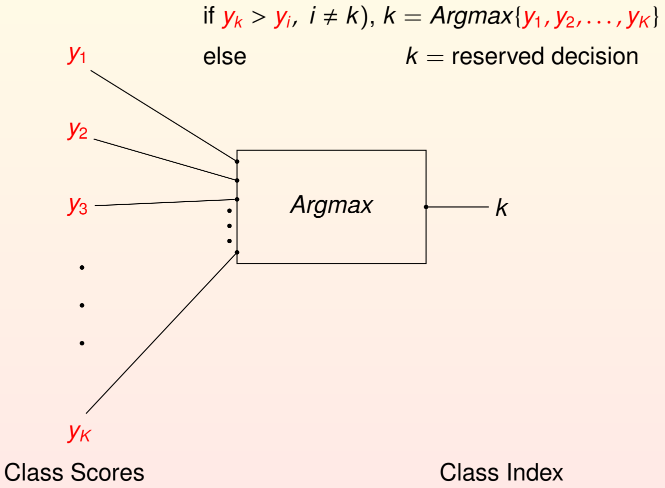
Non-uniform Quantization



The N-tuple Calculation for Class k



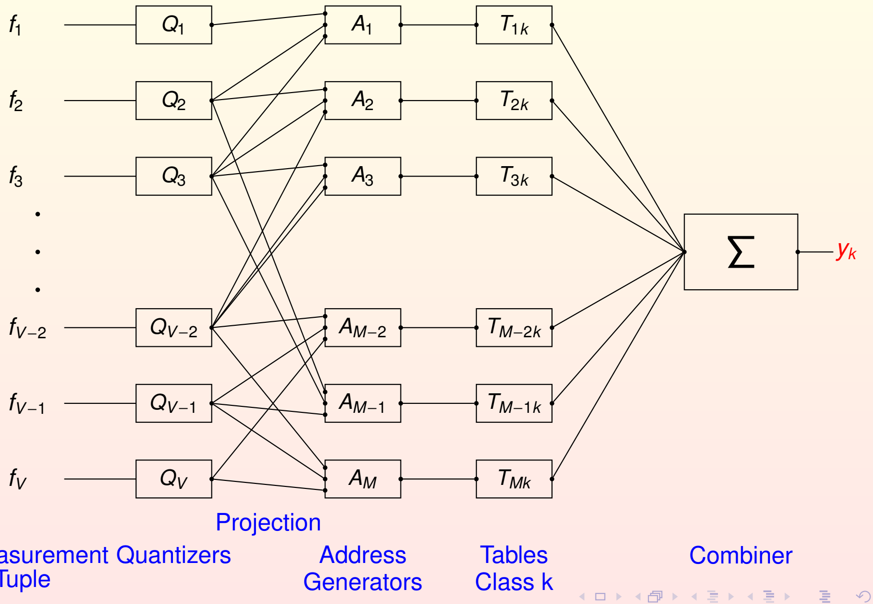
The N-tuple Class Index Generator



Optimizing the N-tuple Classifier

- Quantization
  - Optimize the number of quantized levels for each feature
  - Find the Optimal Quantizer boundaries
- Projections
  - Find the Optimal Index Sets
- Tables
  - Find the Optimal Values for all Table Entries
- Combiner
  - Find the Optimal way to Combine Scores
- Class Index
  - Optimize the way the Class Index is Determined

N-tuple Subspace Method Using Probability of Class

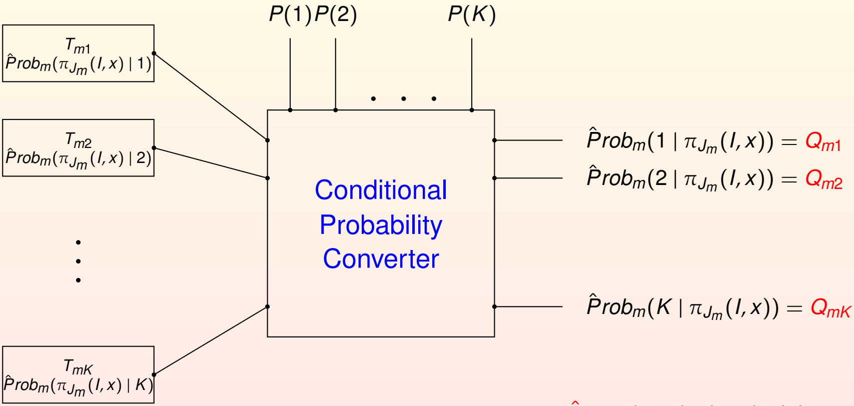


Conditional Probability of Class Given Projected Tuple

$$\begin{aligned} T_{mk}(J_m, u) &= \hat{P}rob((J_m, u) \mid k) \\ \hat{P}rob(J_m, u) &= \sum_{k'=1}^{K} \hat{P}rob((J_m, u) \mid k')P(k') \\ \hat{P}rob(k \mid (J_m, u)) &= \frac{\hat{P}rob((J_m, u) \mid k)P(k)}{\sum_{k'=1}^{K} \hat{P}rob((J_m, u) \mid k')P(k')} \end{aligned}$$

$$\begin{aligned} T_{mk}(\pi_{J_m}(I, x)) &= \hat{P}rob(\pi_{J_m}(I, x) \mid k) \\ \hat{P}rob(k \mid \pi_{J_m}(I, x)) &= \frac{\hat{P}rob(\pi_{J_m}(I, x) \mid k)P(k)}{\sum_{k'=1}^{K} \hat{P}rob(\pi_{J_m}(I, x) \mid k')P(k')} \end{aligned}$$

Discrete Bayes Rule in Subspace Indexed by J_m

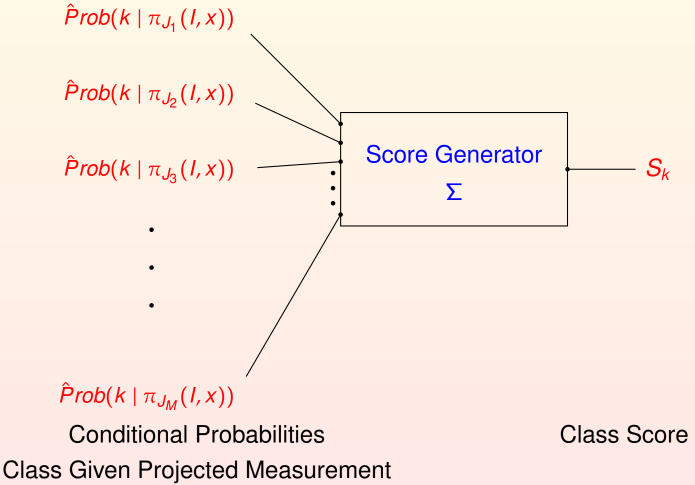


$$Q_{mk} = \hat{P}rob_m(k \mid \pi_{J_m}(I,x)) = \frac{\hat{P}rob_m(\pi_{J_m}(I,x) \mid k)P(k)}{\sum_{k'=1}^{K} \hat{P}rob_m(\pi_{J_m}(I,x) \mid k')P(k')}$$

Assign to class k^* when $Q_{mk^*} > Q_{mk}, k \neq k^*$

Score Generator

$$S_k = \sum_{m=1}^{M} \hat{P}(k \mid \pi_{J_m}(I, x))$$



Class Assignment

- If $S_{k*} > S_k,\ k \neq k^*$
  - Assign class k^* to (I, x)
- Else
  - Assign Reserve Decision to (I, x)

N-tuple Subspace Classifier With Bleaching

- Bleaching Threshold b
- $S_k(I, x) = |\{m \in [1, M] \mid T_{mk}(\pi_{J_m}(I, x) \geq b\}|$
- If $S_{k^*}(I, x) > S_k(I, x), k \neq k^*$
  - Assign class k^* to (I, x)
- Else
  - Assign class Reserve Decision to (I, x)

Hardware N-tuple Method Diagram