Name: Ashish Verma Course:CS773 HW#5

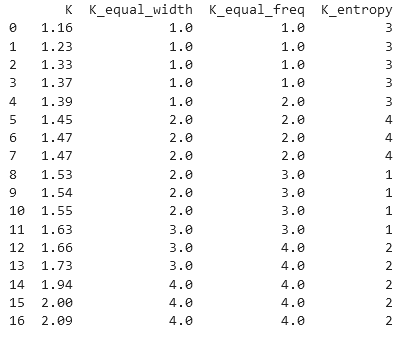
Solution1:

Given

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | P | L | I | K | Ec | C | Ed | Class |
| 31823 | 663 | 223 | 182 | 1.23 | 0.58 | 32274 | 201 | SEKER |
| 27275 | 605 | 220 | 158 | 1.39 | 0.69 | 27604 | 186 | DERMASON |
| 32799 | 654 | 220 | 190 | 1.16 | 0.5 | 33087 | 204 | SEKER |
| 58434 | 981 | 396 | 190 | 2.09 | 0.88 | 59309 | 273 | HOROZ |
| 68513 | 1015 | 359 | 244 | 1.47 | 0.73 | 69406 | 295 | BARBUNYA |
| 85702 | 1107 | 428 | 257 | 1.66 | 0.8 | 86542 | 330 | CALI |
| 137358 | 1365 | 508 | 345 | 1.47 | 0.73 | 138093 | 418 | BOMBAY |
| 41643 | 769 | 295 | 181 | 1.63 | 0.79 | 42233 | 230 | SIRA |
| 68551 | 1025 | 356 | 246 | 1.45 | 0.72 | 69684 | 295 | BARBUNYA |
| 137115 | 1427 | 519 | 337 | 1.54 | 0.76 | 138970 | 418 | BOMBAY |
| 27277 | 605 | 218 | 159 | 1.37 | 0.68 | 27611 | 186 | DERMASON |
| 41646 | 762 | 286 | 186 | 1.53 | 0.76 | 42074 | 230 | SIRA |
| 85666 | 1119 | 436 | 251 | 1.73 | 0.82 | 86305 | 330 | CALI |
| 58454 | 965 | 392 | 196 | 2 | 0.87 | 60280 | 273 | HOROZ |
| 58484 | 956 | 382 | 197 | 1.94 | 0.86 | 59456 | 273 | HOROZ |
| 41646 | 768 | 288 | 186 | 1.55 | 0.76 | 42225 | 230 | SIRA |
| 27267 | 597 | 215 | 162 | 1.33 | 0.66 | 27575 | 186 | DERMASON |

Python Implementation





Manual Calculations

Equal-width binning:

**Range of data**

Max value of K:-2.09

Min value of K:-1.16

**Calculate the range**

Range = Max – Min = 2.09 – 1.16 = 0.93

**Calculate bin width**

Bin width = Range/number of bins = 0.93/4= 0.2325

**Determine Bin Edges**

* Bin 1: [1.16, 1.16 + 0.2325) = [1.16, 1.3925)
* Bin 2: [1.3925, 1.3925 + 0.2325) = [1.3925, 1.625)
* Bin 3: [1.625, 1.625 + 0.2325) = [1.625, 1.8575)
* Bin 4: [1.8575, 1.8575 + 0.2325) = [1.8575, 2.09)

**Assign values of K**

* K=1.23 falls into Bin 1: [1.16, 1.3925)
* K=1.39 falls into Bin 1: [1.16, 1.3925)
* K=1.16 falls into Bin 1: [1.16, 1.3925)
* K=2.09 falls into Bin 4: [1.8575, 2.09)
* K=1.47 falls into Bin 2: [1.3925, 1.625)
* K=1.66 falls into Bin 3: [1.625, 1.8575)
* K=1.47 falls into Bin 2: [1.3925, 1.625)
* K=1.63 falls into Bin 3: [1.625, 1.8575)
* K=1.45 falls into Bin 2: [1.3925, 1.625)
* K=1.54 falls into Bin 2: [1.3925, 1.625)
* K=1.37 falls into Bin 1: [1.16, 1.3925)
* K=1.53 falls into Bin 2: [1.3925, 1.625)
* K=1.73 falls into Bin 3: [1.625, 1.8575)
* K=2.00 falls into Bin 4: [1.8575, 2.09)
* K=1.94 falls into Bin 4: [1.8575, 2.09)
* K=1.55 falls into Bin 2: [1.3925, 1.625)
* K=1.33 falls into Bin 1: [1.16, 1.3925)

Equal frequency binning:

**Sort the data in ascending order**:

* Sorted K values: [1.16, 1.23, 1.33, 1.37, 1.39, 1.45, 1.47, 1.47, 1.53, 1.54, 1.55, 1.63, 1.66, 1.73, 1.94, 2.00, 2.09]

**Determine the number of elements per bin**

* Total number of elements: 17
* Number of bins: 4
* Elements per bin: 17/4=4.25
* Since we can't have a fraction of an element, bins will have either 4 or 5 elements.

**Assign each value to a bin**

* Bin 1: [1.16, 1.23, 1.33, 1.37, 1.39] (first 5 elements)
* Bin 2: [1.45, 1.47, 1.47, 1.53, 1.54] (next 5 elements)
* Bin 3: [1.55, 1.63, 1.66, 1.73] (next 4 elements)
* Bin 4: [1.94, 2.00, 2.09] (last 3 elements)

|  |  |
| --- | --- |
| **K** | **K\_Binned\_EF** |
| 1.16 | 1 |
| 1.23 | 1 |
| 1.33 | 1 |
| 1.37 | 1 |
| 1.39 | 2 |
| 1.45 | 2 |
| 1.47 | 2 |
| 1.47 | 2 |
| 1.53 | 3 |
| 1.54 | 3 |
| 1.55 | 3 |
| 1.63 | 3 |
| 1.66 | 4 |
| 1.73 | 4 |
| 1.94 | 4 |
| 2 | 4 |
| 2.09 | 4 |
|  |  |
|  |  |

Entropy-based discretization:

Sorted K values:

[1.16,1.33,1.37,1.39,1.45,1.47,1.47,1.53,1.54,1.63,1.66,1.73,1.94,2.00,2.09]

Let's calculate the entropy for a specific cut point, K=1.63

Let’s Split the Data(Split 1)

**Left Split (K ≤ 1.63)**:

[1.16,1.33,1.37,1.39,1.45,1.47,1.47,1.53,1.54,1.63]

**Right Split (K > 1.63)**:

[1.66,1.73,1.94,2.00,2.09]

Calculate the entropy for each split

P(SEKER)​=2/11 ,P(DERMASON)=3/​11, P(BARBUNYA)​=2/11, P(BOMBAY)​=2/11​, P(SIRA)​=3​/11

**Entropy(Left)** = -(2/11\*log\_2(2/11) + 3/11\*log\_2 (3/11) +2/11\*log\_2(2/11) +2/11\*log\_2(2/11) + 3/11\*log\_2(3/11)

= -(-2.636)

= 2.636

P(CALI)=2/5​, P(HOROZ)=3/5

**Entropy(Right)** = -(2/6 \* log\_2(2/6) + 3/6\*log\_2(3/6))

=1.028

**Weighted Entropy** = 10/17\*2.636+ 5/17 \*1.028

= 1.8529

**Bin1** : [1.16, 1.63]

**Bin2** : [1.63, 2.09]

Let’s Split the Data(Split 2)

**Left Split (K ≤ 1.53)**:

[1.16,1.33,1.37,1.39,1.45,1.47,1.47,1.53]

**Right Split (K > 1.53)**:

[1.54,1.63,1.66,1.73,1.94,2.00,2.09]

Calculate the entropy for each split

P(SEKER)​=2/9 ,P(DERMASON)=3/​9, P(BARBUNYA)​=2/9, P(BOMBAY)​=1/9​, P(SIRA)​=1​/9

**Entropy(Left)** = −( 2/9\*log\_2(2/9) + 3/9\*log\_2(3/9) +2/9\*log\_2(2/9) +1/9\*log\_2(1/9) + 1/9\*log\_2(1/9))

= -(-1.9056)

= 1.9056

P(BOMBAY)=1/8, P(SIRA)=2/8 , P(CALI)=2/8​, P(HOROZ)=3/8

**Entropy(Right)**= -(1/8\*log\_2(1/8) + 2/8\*log\_2(2/8) + 2/8\*log\_2(2/8) +3/8\*log\_2(3/8))

=-(-1.9056)

=1.9056

**Weighted Entropy** = 9/17\*1.9056+ 8/17 \*1.9056

= 1.9056

**Bin1** : [1.16, 1.53]

**Bin2** : [1.53, 2.09]

Since weighted entropy for Split1 is lower than split2

**Bin1** : [1.16, 1.63]

**Bin2** : [1.63, 2.09]

Is better binning

Since there could be various cut points, we need to find the best cut which have lowest weighted entropy.

**K\_entropy column in above table shows the actual binning based on entropy the minimum value of K split is coming as 1.47 as shown I below table**

|  |  |
| --- | --- |
| **K** | **Information Gain** |
| 1.16 | 0.132 |
| 1.23 | 0.45 |
| 1.33 | 0.437 |
| 1.37 | 0.552 |
| 1.39 | 0.801 |
| 1.45 | 0.746 |
| **1.47** | **0.807** |
| 1.53 | 0.645 |
| 1.54 | 0.734 |
| 1.55 | 0.702 |
| 1.63 | 0.801 |
| 1.66 | 0.597 |
| 1.73 | 0.6 |
| 1.94 | 0.288 |
| 2 | 0.088 |
| 2.09 | 0.072 |

Hence, we can split the bin as

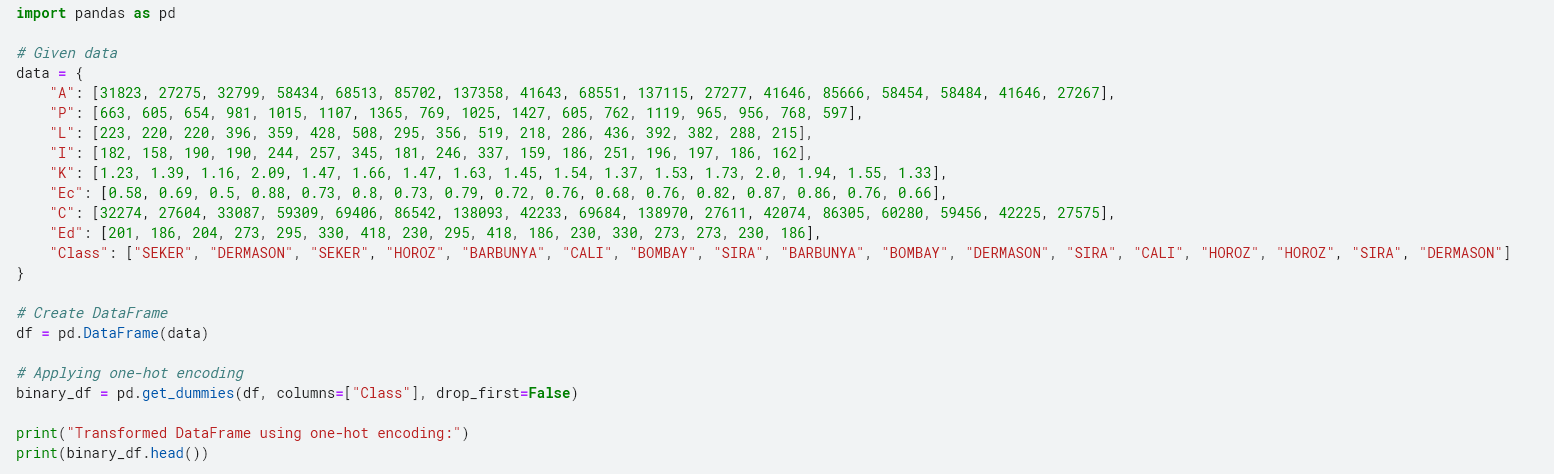
Bin1 [1.16,1.47]

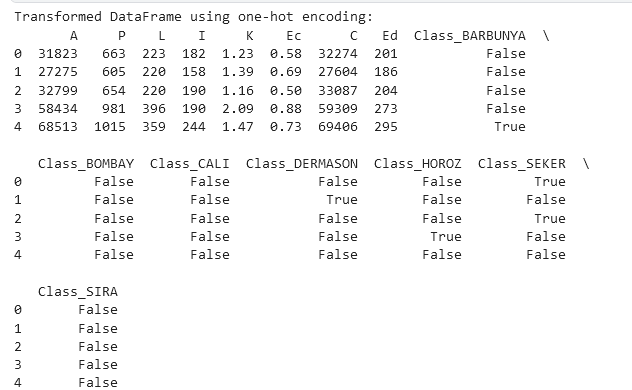
Bin2 [1.47,2.09]

Solution2:-

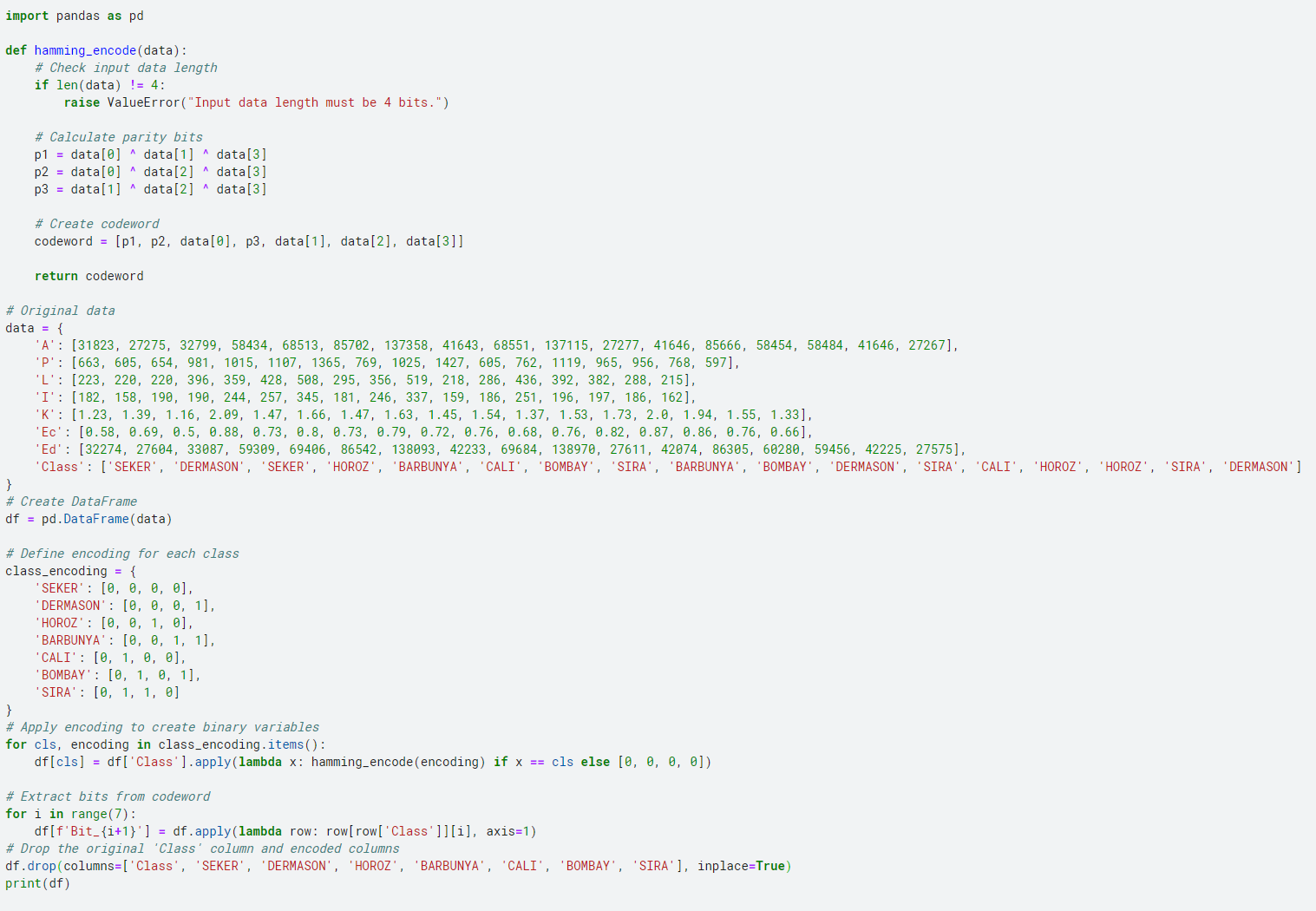
Python Implementation

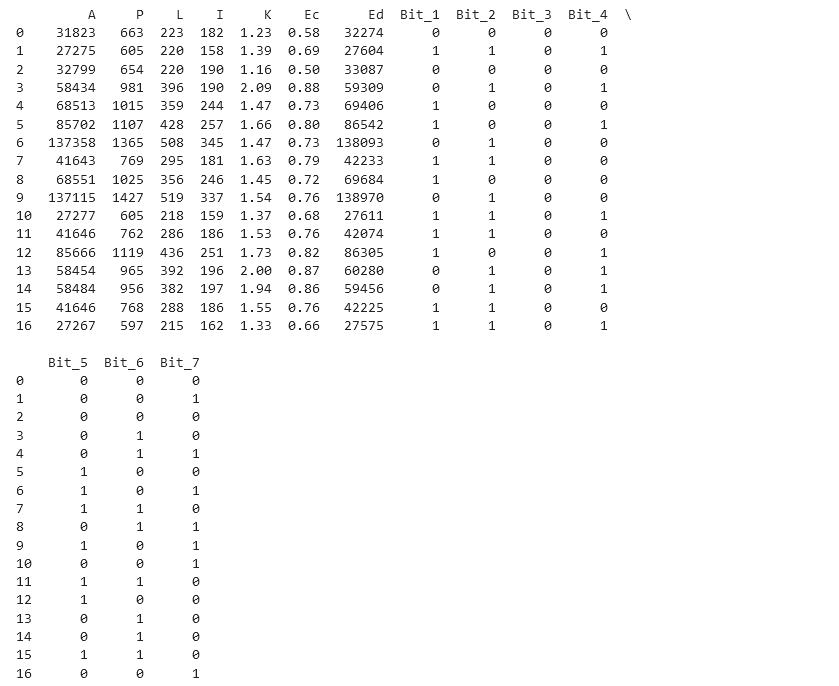
Standard Method



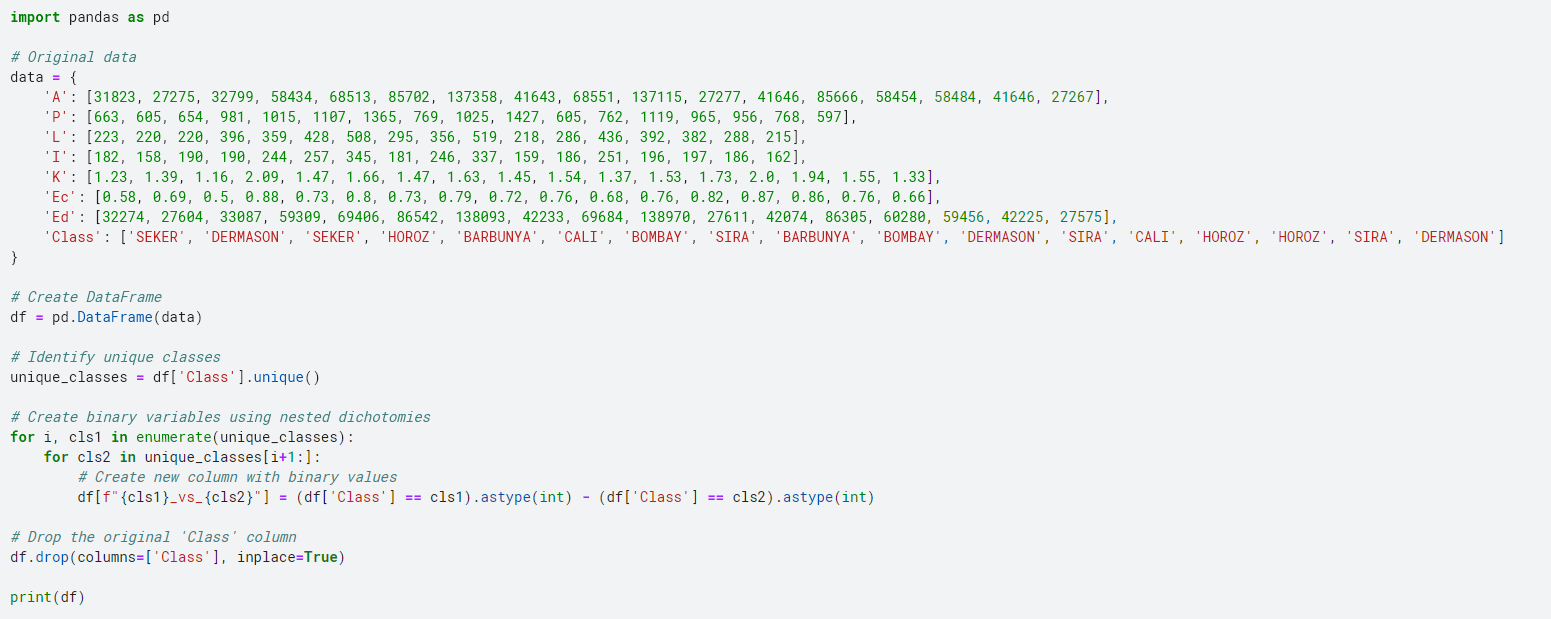


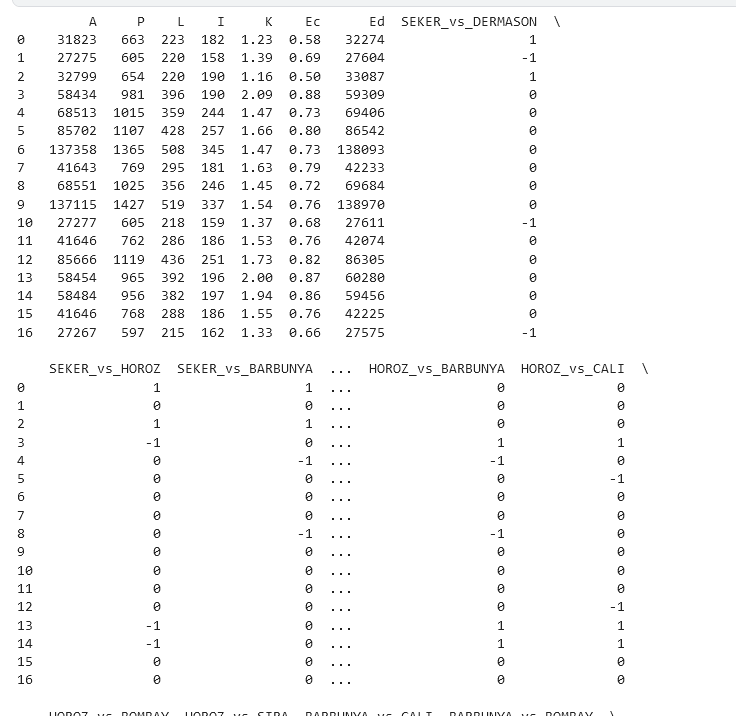
Error Correcting Code using Hamming method

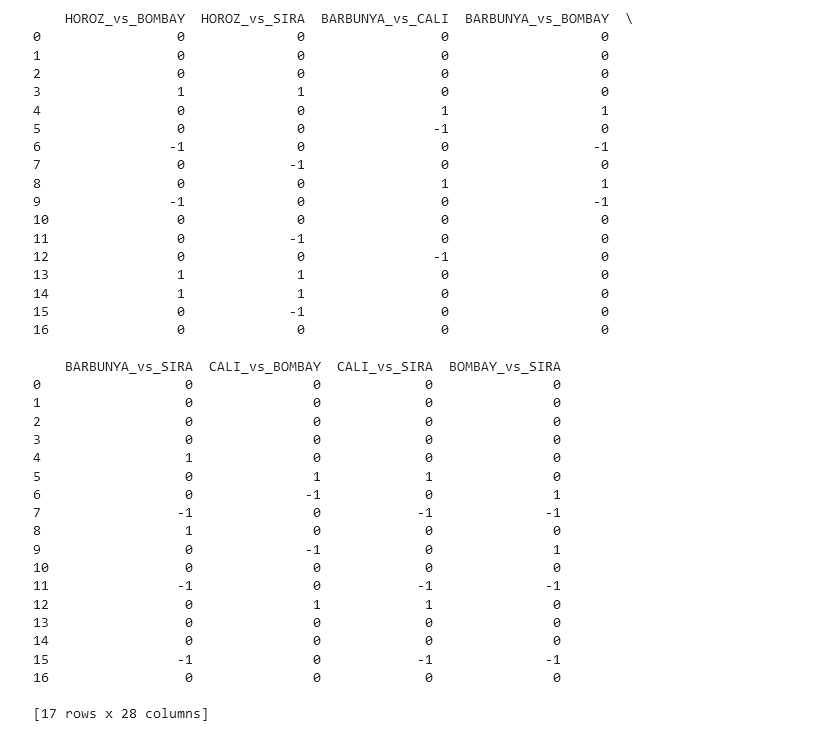




Nested dichotomies







Manual Methods

Standard method:-

**Identify Unique Classes**

Unique classes: SEKER, DERMASON, HOROZ, BARBUNYA, CALI, BOMBAY, SIRA

**Create Binary Variables**

For each unique class, create a new binary variable.

* SEKER: 1 if observation is SEKER, 0 otherwise.
* DERMASON: 1 if observation is DERMASON, 0 otherwise.
* HOROZ: 1 if observation is HOROZ, 0 otherwise.
* BARBUNYA: 1 if observation is BARBUNYA, 0 otherwise.
* CALI: 1 if observation is CALI, 0 otherwise.
* BOMBAY: 1 if observation is BOMBAY, 0 otherwise.
* SIRA: 1 if observation is SIRA, 0 otherwise.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SEKER | DERMASON | HOROZ | BARBUNYA | CALI | BOMBAY | SIRA |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Error-correcting code method

**Identify Unique Classes**

* SEKER
* DERMASON
* HOROZ
* BARBUNYA
* CALI
* BOMBAY
* SIRA

**Assign Binary Codes**

* SEKER: 000
* DERMASON: 001
* HOROZ: 010
* BARBUNYA: 011
* CALI: 100
* BOMBAY: 101
* SIRA: 110

**Encode Data**

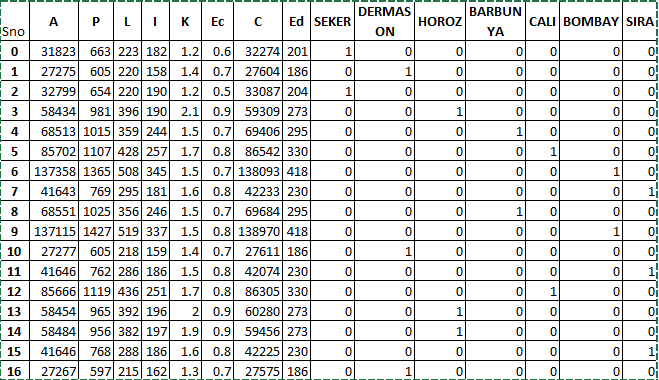
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | P | L | I | K | Ec | C | Ed | Encoded\_Class |
| 31823 | 663 | 223 | 182 | 1.23 | 0.58 | 32274 | 201 | 000 |
| 27275 | 605 | 220 | 158 | 1.39 | 0.69 | 27604 | 186 | 001 |
| 32799 | 654 | 220 | 190 | 1.16 | 0.5 | 33087 | 204 | 000 |
| 58434 | 981 | 396 | 190 | 2.09 | 0.88 | 59309 | 273 | 010 |
| 68513 | 1015 | 359 | 244 | 1.47 | 0.73 | 69406 | 295 | 011 |
| 85702 | 1107 | 428 | 257 | 1.66 | 0.8 | 86542 | 330 | 100 |
| 137358 | 1365 | 508 | 345 | 1.47 | 0.73 | 138093 | 418 | 101 |
| 41643 | 769 | 295 | 181 | 1.63 | 0.79 | 42233 | 230 | 110 |
| 68551 | 1025 | 356 | 246 | 1.45 | 0.72 | 69684 | 295 | 011 |
| 137115 | 1427 | 519 | 337 | 1.54 | 0.76 | 138970 | 418 | 101 |
| 27277 | 605 | 218 | 159 | 1.37 | 0.68 | 27611 | 186 | 001 |
| 41646 | 762 | 286 | 186 | 1.53 | 0.76 | 42074 | 230 | 110 |
| 85666 | 1119 | 436 | 251 | 1.73 | 0.82 | 86305 | 330 | 100 |
| 58454 | 965 | 392 | 196 | 2 | 0.87 | 60280 | 273 | 010 |
| 58484 | 956 | 382 | 197 | 1.94 | 0.86 | 59456 | 273 | 010 |
| 41646 | 768 | 288 | 186 | 1.55 | 0.76 | 42225 | 230 | 110 |
| 27267 | 597 | 215 | 162 | 1.33 | 0.66 | 27575 | 186 | 001 |

Nested dichotomies

**Given classes**

* SEKER
* DERMASON
* HOROZ
* BARBUNYA
* CALI
* BOMBAY
* SIRA

Create Binary variable for each class



We'll partition the classes into two groups in alphabetical order.

To apply nested dichotomies, split the classes into binary classification tasks.

**First Split:**

* Group 1: SEKER, DERMASON, HOROZ
* Group 2: BARBUNYA, CALI, BOMBAY, SIRA

Create a binary column split\_1:

* split\_1 = 1 if the class is in Group 1
* split\_1 = 0 if the class is in Group 2

**Second Split for Group 1:**

* Sub-group 1.1: SEKER
* Sub-group 1.2: DERMASON, HOROZ

Create a binary column split\_1\_1:

* split\_1\_1 = 1 if the class is SEKER
* split\_1\_1 = 0 if the class is DERMASON or HOROZ

**Second Split for Group 2:**

* Sub-group 2.1: BARBUNYA, CALI
* Sub-group 2.2: BOMBAY, SIRA

Create a binary column split\_2:

* split\_2 = 1 if the class is in Sub-group 2.1
* split\_2 = 0 if the class is in Sub-group 2.2

