# 1 Case Selection

In Case Selection, three methods (knn, parzen window and mkp) are implemented as Matlab Functions respectively. The description is detailed as follows:

**Name:** tj\_knn.m

**Type**: function

**Sample of Use:**

k\_neighbors = tj\_knn(input, k, target)

***Where***

|  |  |  |
| --- | --- | --- |
| Input | : | An m\*n array containing m historical cases with n features respectively. |
| k | : | Number of nearest neighbors need to be found. |
| target | : | An n-vector representing the target case to be predicted. |
| k\_neighbors | : | An k\*n array containing the k nearest cases found. |

**Description**:

It’s the implementation of KNN method. It will calculation the Euclidean Distance between every historical case with the target case by features, and select the most k nearest cases to fill the k\_neighbors array.

**Name:** tj\_parzen\_window.m

**Type**: function

**Sample of Use:**

neighbors = tj\_parzen\_window( input,max\_dis,target )

***Where***

|  |  |  |
| --- | --- | --- |
| Input | : | An m\*n array containing m historical cases with n features respectively. |
| max\_dis | : | Threshold value of the distance |
| target | : | An n-vector representing the target case to be predicted. |
| neighbors | : | An m\*n array containing the m nearest cases found. |

**Description**:

It’s the implementation of Parzen-Window method. It will calculate the Euclidean Distance between every historical case with the target case by features, and filter them by the max\_dis threshold. Finally, those cases fell in the fixed window will be collected and filled in the array of “neighbors”.

**Name:** tj\_parzen\_window.m

**Type**: function

**Sample of Use:**

neighbors = tj\_mkp( input,k,max\_dis,target )

***Where***

|  |  |  |
| --- | --- | --- |
| Input | : | An m\*n array containing m historical cases with n features respectively. |
| k | : | Number of nearest neighbors need to be found. |
| max\_dis | : | Threshold value of the distance |
| target | : | An n-vector representing the target case to be predicted. |
| neighbors | : | An m\*n array containing the m nearest cases found. |

**Description**:

It’s the implementation of MKP method. It will first call tj\_parzen\_window function to select the nearest neighbors. After the function finished, it will count the result to see if larger or equal to k neighbors have been selected. If no, it will call tj\_knn function to select k neighbors. In either case, the selected neighbors will be filled into the array of “neighbors”.

# 2 Case Adaption

In Case Adaption, two methods (mean value and weighted mean value) are implemented as Matlab Functions respectively. The description is detailed as follows:

**Name:** tj\_CaseAdaption\_Mean.m

**Type**: function

**Sample of Use:**

estimation = tj\_CaseAdaption\_Mean( neighbors )

***Where***

|  |  |  |
| --- | --- | --- |
| neighbors | : | An m\*n array containing the selected cases |
| estimation | : | Predicted Value using those neighbors. |

**Description**:

It’s the implementation of Mean Value method. It will simply sum up the efforts of selected cases, and divide it by the number of cases. In other words, the mean value of the efforts of cases is calculated and returned.

**Name:** tj\_CaseAdaption\_Weighted.m

**Type**: function

**Sample of Use:**

estimation = tj\_CaseAdaption\_Weighted( neighbors )

***Where***

|  |  |  |
| --- | --- | --- |
| neighbors | : | An m\*n array containing the selected cases |
| estimation | : | Predicted Value using those neighbors. |

**Description**:

It’s the implementation of Weighted-Mean Value method. It will use the equations detailed as bellows to calculate the estimated cost. The weighted mean value is returned.

(1)

(2)