Clustering for Clasification

Reuben Evans

Contents

1 Introduction **3**

2 Description **4**
- 2.1 Motivation . . . 4
- 2.2 Idea . . . 4
- 2.3 The Clusterers . . . 5
  - 2.3.1 First K . . . 5
  - 2.3.2 Simple K Means . . . 6
  - 2.3.3 Farthest First . . . 6
  - 2.3.4 Bisecting K Means . . . 6
  - 2.3.5 Expectation Maximization (EM) . . . 6

3 Experiments **7**
- 3.1 Algorithms . . . 7
  - 3.1.1 Algorithms for Nominal Datasets . . . 7
  - 3.1.2 Algorithms for Numeric Datasets . . . 7
- 3.2 Datasets . . . 8
- 3.3 Experiment One . . . 8
  - 3.3.1 Nominal Datasets . . . 8
  - 3.3.2 Numeric Datasets . . . 8
- 3.4 Experiment Two . . . 8
  - 3.4.1 Nominal Datasets . . . 10
  - 3.4.2 Numeric Datasets . . . 10
- 3.5 Experiment Three . . . 10
- 3.6 Experiment Four . . . 10
- 3.7 Experiment Five . . . 10

4 Related Work **11**
4.1 Data Squishing 11
4.2 Instance Selecction 11

5 Conclusions **12**
5.0.1 Future Work 12

Chapter 1

Introduction

Chapter 2

Description

This section details the problem and an algorithm that addresses this problem

2.1 Motivation

This section describles the reasons behind this research. What makes this algorithm of interest

Dataset too large, cannot fit in memory Classifier to comples speed up by reducing the size of input

2.2 Idea

The classifier handles nominal values using the nominal to binary filter to convert them into a number of binary attributes. Only numeric and binary attributes are used in the filtering process. All attributes are normalized, to prevent different attributes having more weight than others in distance calculations.

For a numeric class the data is clustered directly producing exactly as many clusters as specified by the user in the C parameter to the classifier. For all other types of classes the data is separated sets with all the same class and then the clustering process is use on all sets resulting in C clusters for each posable class value. The clusters are then built, when building the clusters if there are less Instances than the number of classes then the instances are returned as the clusters. Otherwise the instances are randomized and a

number of instances equal to the desired number o clusters are taken from the start of the dataset as cluster centers.

Each of the remaining instances is then taken in turn and merged with the closest cluster centre. The closeness is determined by measuring the relative squared euclidean distance between the instance and each cluster centre. The cluster centre is then updated so that each of it's attribute values is the sum of the weight adjusted attribute values of the cluster centre and the instance, so if the centre had a weight of three and the instance had a weight of one then the resulting attribute value would be three quarters the value of the centre plus one quarter the value of the instance. The weight of the instance is added to the weight of the centre to create the new centre weight.

When all instances have been merged into the clusters the set of clusters is passed to the classifier specified by the user to build the model.

ClustersForClasses Separate Instances into separate collections so each collection has only one class value Use ClustersForData on each collection set to create the clusters Merge resulting cluster sets from ClustersForData into one large set of clusters Return clusters

2.3 The Clusterers

This section describes the properties required in a clusterer for it to be used with the algorthim described by the previous section. The key property required in a clusterer is that it can be asked to make a set number of clusters. While it doesn't have to be exact the clusterer must be able to get close to the requested number. It must also be possible to obtain a meta instance for each cluster. Some clusterers provide a cluster centriod that can be used as the meta instance to reprresent that cluster others mush hhave one generated. The way the Clustered Meta Instance Algorthin generates these is by taking the average values of each attribute for alll the memmbers of the cluster.

2.3.1 First K

First K is a very naive clusterer focused on speed. This clusterer declares the first K instances encountered to be the cluster centers, each subsequent instance in the dataset is then merged with the closest cluster centre by the Euclidian distance. This results in clusters that are certainly not tthe best

clusters thatt could have been obtained from the data however it allows the creation of clusters in a single pass through that data which results in a clusterer that is linear in the number of clusters and instances

ClustersForData IF number of instances is less than or equal to number of classes return instances as clusters randomize instances Set first C instances as cluster centers For Each Instance above C MergeWithClosest Return clusters

MergeWithClosest Set minD = relativeSquaredEuclideanDistance from instance to first cluster for each cluster IF relativeSquaredEuclideanDistance from instance to cluster ¡ minD minD = relativeSquaredEuclideanDistance from instance to cluster Set Total Weight = Weight(minD) + Weight(instance) w0 = Weight(mnD)/total w1 = Weight(instance)/total for each attribute in cluster with minD attributeValue = w0* attributeValue + w1 * instanceAttributeValue Weight(minD) = Weight(minD) + Weight(instance)

2.3.2 Simple K Means

It's just simple K means

2.3.3 Farthest First

Fatherest firsat is a Varient off K means that placces each cluster centre in turn at the point furrthest frrom the existing cluster centres. This point must lie within the data area. This greatly spped up the clustering in mostt cases since less reassignment and adjustment is needed

2.3.4 Bisecting K Means

Creates Bisecting Regions applies Simple K means algorthim with two clusters recursively.

2.3.5 Expectation Maximization (EM)

Model based clusterer

Chapter 3

Experiments

This section discusses the three experiments I have conducted in order to test my system

3.1 Algorithms

This section describes the four different algorithms we used for testing and why each of them was chosen. The Experiments use two algorrthims for each Class type a simple one and a more complex one. Two algorthins were chosen because earlier testing had shhown that some clusterers worked better with more complex clusterers, while others were better forr the simple clusterers.

3.1.1 Algorithms for Nominal Datasets

The experiments using nominal datasets are conducted with two different algorithms: Naive Bayes and Logistic Regression

3.1.2 Algorithms for Numeric Datasets

The experiments using numeric datasets are conducted with two different algorithms: Linear Regression and M5 Model Trees

Table 3.1: Algorithms Used in testing

	Nominal	Numeric
Simple	Naive Bayes	Linear Regression
Complex	Logistic Regression	M5 Model Trees

3.2 Datasets

The three experiments use twenty different datasets. Ten Nominal Datasets and Ten numeric ones.

3.3 Experiment One

This experiment tests all the clusterers against several of the smaller datasets. This experiment is the only one with all the clusterers since several of them are too expencive and are not practical on the larger datasets.

3.3.1 Nominal Datasets

The Following Four Nominal datasets were used by this experiment: mushroom, opdigits, hypothroid, kr-vs-kp

3.3.2 Numeric Datasets

Only Kin8nm and ailerons are small enough to be used for this experiment. These two datasets are dissimilar enough to provide a reasonable check of the clusterers performance on numeric data.

3.4 Experiment Two

This experiment takes two of the clusterers First K and Bisecting K Means and compares them to random selection. This experiment uses all twenty datasets described above. These cclusterers were shown in earlier testing to be considerably faster than the others since they do much less work in their efforts to cluster the data.

Table 3.2: Datasets Used in these experiments

Datset Name	Size	Attributes	Classes
pendigits	10992	17	10
letter	20000	17	26
mushroom	8124	23	2
opdigits	5620	65	10
hypothroid	3772	30	4
kr-vs-kp	3196	37	2
Waveform21	40000	21	3
Waveform40	40000	40	3
Agrawal	40000	10	2
2Dplanes	40768	11	Numeric
ailerons	13750	41	Numeric
kin8nm	8192	9	Numeric
house-8L	22784	9	Numeric
mv	40768	11	Numeric
fried	40768	11	Numeric
ColorHistogram	68040	33	Numeric
LayoutHistogram	66615	33	Numeric
CoocTexture	68040	17	Numeric
ColorMoments	68040	10	Numeric

3.4.1 Nominal Datasets

All Nominal datasets were used.
Arrg here be results!! - eventually

3.4.2 Numeric Datasets

All Numeric datasets were used..

3.5 Experiment Three

This experiment looks at the noise reduction effects of a small amount of summarization over using the full dataset. This experiment used the first K clusterer. Small amounts of summarization were compared to the base classifier performance.

3.6 Experiment Four

Even weightts

3.7 Experiment Five

Dropping small cllusters

Chapter 4

Related Work

This section describes how the Clustered Meta Instance Classifier fits in with other work.

4.1 Data Squishing

Data squishing techniques are the most similar to the Clustered Meta Instance Classifier.

4.2 Instance Selecction

Instance Selection uses real instances rather than meta instances

Chapter 5

Conclusions

.

5.0.1 Future Work

Streaming stuff