University of Victoria, Software Engineering

Adzuna: A Kaggle Data Mining Competition

Jordan Ell, Braden Simpson

March 22, 2013

1 Introduction

2 Data Gathering

The Adzuna competition is all about trying to predict the salaries of jobs posted online from a few choice data points. These predictions will help to bring transparancy to the market while helping potential employees discover the best jobs suited for themselves. This being said, the Adzuna team has provided all challengers of the Kaggle competition with data sets in CSV formats.

There are two main data sets used for this competition. The first set contains the training instances. The training instance contains 11 fields of which 10 are attributes with the 11th being the class. The 10 attributes are as follows. An ID provides a unique identifier for each job. This ID will not be used in the actuall data mining process but will be used to associate jobs to salaries in the final results of this project. A title is a free text field that gives the position that an employeer is looking to fill. A full description is a free text field provided by the job advertisers. This description field holds many of the most interesting potential for future data mining analysis. This field has been give to the competetors stripped of any text that may directly refer to salary. The location raw field is a free text field for the location of the job while the location normalized field is a formal location provided by Adzuna with standard formatting. Contract type provides a categorical look at full time or part time. Contact time provides a categorical view of permanent or contract positions. The company field is a free text field for company names. The job category field is a selection field from 30 standard job categories which employeers pick to their needs. Finally, the source name field provides the email address

of the employer.

The second set of data provided by the Kaggle competition is the validation set of data. This data again comes in the CSV format and again has 11field of which 10 are attributes and 1 as a class. The difference here is that salary has been stripped from the class field. This file is used as a standard testing file for all competetor's data models. This is the file of which all tests will be put against.

There is also an optional file provided by the Kaggle team called the location tree. This file is again a CSV file, however is used for a different purpose. This file is used for the purpose of describing the hierarchial nature of the normalized location field in the training set data. Here the country, direction, city, and riegon are all given in order to give sense to the previously mentioned normalized location data.

For this project, we have simply made use of both the training and validation sets of data. These pieces of data can be found at the Kaggle website.[1]

3 Data Preparation

Our data mining process intends to use the program Vowpal Wabbit (which will be explained further in the section to follow). Vowpal Wabbit, or VW, takes a custom inout format which needed to be accounted for before any data mining could take place. While preparing this custom data format, various other steps were also taken (to be explained) in order to prepare the data for various attemps and algorithms with data mining. We will outline below the type of input format needed, the custom data transformations, as well as data normalization for future data mining efforts.

The input format for VW is as follows:

$$[Label][Importance[Tag]]|NamespaceFeatures|NamespaceFeatures...|NamespaceFeatures \tag{3.1}$$

where Namespace=String[:Value] and Features=(String[:Value])*. Here we can see the imediate differences to the original CSV files that are provided for the Kaggle challenge. The Label is the floating point number that is being attemped to be predicted. The importance tage can be used to give weight to a specific training instance over the ther instances. This can be useful for weighted preditions or higher confidence in a particular instance. Namespace is a value identifier while features are the values associated with the identifier. In order to transform the CSV files to the given VW input format, python scripts were created and run to make the conversion. While these python scipts were used primarily for conversion purposes, they also allowed the tweaking and adjustments of data as it was converted. We will now explain the different steps taken to tweak data

[1] http://www.kaggle.com/c/job-salary-prediction/data

during the conversion.

As it was shown on the Kaggle website, the data given in the training instanced represented a skewed bell curve. In order to get this data into a more acceptable bell curve shape, or normalized (as was needed to VW to run properly, more on this later), two solutions were used, one at a time. The first was simply to take the logarithm of the data set. This creates a more acceptable normalization of the data. That being said, this was the first tweak to be given to the data during the conversion process. The other availible option for creating normalized data is to take the sqaure root of the instances. This also gave a somewhat more normalized view of the data. This tweak was also taken on the data although was preformed at a different time than the logarithm. The results of these two data tweaks can be seen in Table 1 as the different types of normalization transformations.

The second large tweak to be placed on the data during the conversion process was the limiting of free text fields to only keywords. In order to make this tweak, the python library topia.termextract was used to extract keywords from local text fields. This means that every text field was analysed sperately for keywords as opposed to looking at all training instances at once. The keyword threshold for reoccurance was set only to 1 as we decided that job descriptions did not always repeat the most important words at any given rate. This data tweak was only applied to data for particular runs of the data mining process and can be seen in the results in Table 1 for how it was used in relation to other data tweaks and their outcomes.

The final data tweak to be placed on the data during the conversion process was the limiting of attributes to be used during the training instance model creation. Our python scripts allowed us to adjust which attributes were to be used for the model generation. Here we played with one major setting in which we turned off all free text fields. The results of turning off the free text fields can again be seen in Table 1 in relation with the normalization tweaks. A large number of combinations of free text and categorical attributes could be tested at any given time with or without keywords with this program. However, due to the limited time of this project only the few were actually tested.

This conversion of data to be handles by a larger data set data mining program along with the keyword analysis and other data tweaks represent a non trival step in our data mining project.

4 Data Mining

5 Results and Conclusions