Living Costs and Food Survey (LCF) Project Summary

1. Introduction

The Big Data team was approached by Social Survey Division (SSD) about the possibility of using commercial data and/or data science methods to help improve the processing of the Living Costs and Food Survey (LCF).

LCF data are collected through CAPI interviews and paper diaries of expenditure for two-week periods which include written entries and receipts. Coding the diary is manually intensive and can take up to eight hours for a single diary, with a mean time of around four hours. Each item must be coded to COICOP 1 (a classification of each product into categories such as 'meat') and have an associated price, quantity and weight (volume). A significant proportion of coding effort is spent filling in missing information, most commonly the product weight. Although the customer (DEFRA) only requires amounts to be completed for half of the survey respondents, the additional time taken to find the correct amounts (usually via internet searches outside of Blaise) is a large contributing factor in diary processing delays.

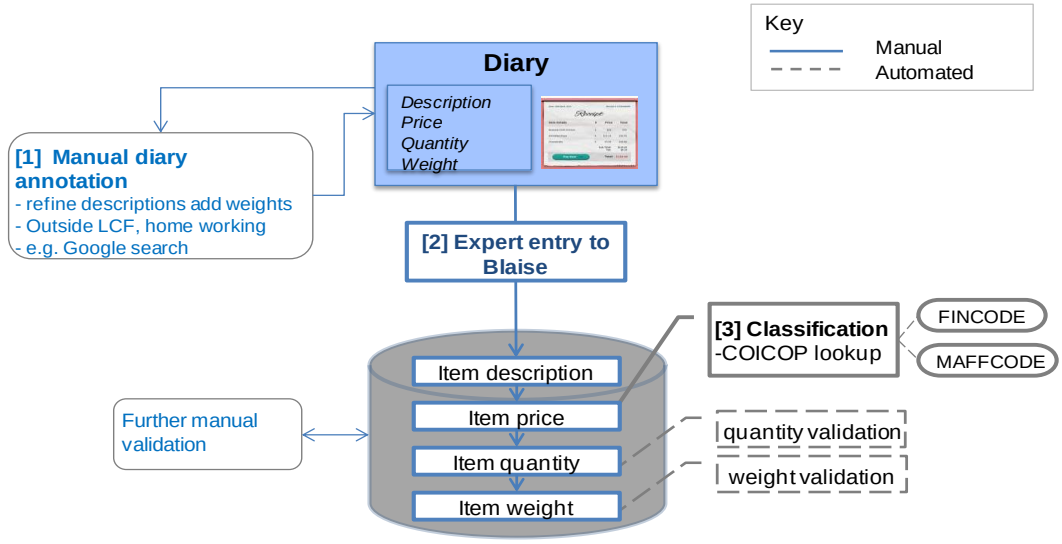

Fig.1 LCF Diary Process

In order to facilitate the LCF diary process, two prototypes were developed by the Big Data Team in consultation with Social Survey Division, Surveys and Life Events Processing and the end user DEFRA. The proposed solutions harness information from clean historic LCF diary data to help complete missing product quantity information (i.e. amount, volume or weight purchased) at the point of data entry.

2. Using historical data to create a lookup

A solution which should integrate easily into the current system and coders' work flow was piloted using flat look-up functions already available in Blaise. The goal was to give the coder an option to choose an amount from a list of matching or very similar items previously entered within the Blaise environment (eliminating the need for an internet search on a different machine or browser).

The list needs to be populated with items which are the closest match in terms of the known information about the item; this could include shop name, price, item's description and COICOP (if available). Blaise has functionality to acquire relevant data for a drop-down list from one or more fixed look-up files stored locally.

Since there are over 200 thousand unique items in the historical 2015 data, having only one file would result in an unmanageable drop-down menu. Hence this prototype is focused on creating a series of smaller flat look-up files based on historic data using the previously entered values (from diary entry). Pre-processed historic data are split into smaller groups (i.e. into separate look-up files) and ordered in a way that is relevant to coders' needs and optimal for efficient look-up (i.e. similar items next to each other).

As far as evaluation is concerned, the LCF coders are the most suitable people to evaluate the performance of this prototype. Therefore, user testing will take place shortly to get feedback. Meanwhile as a proxy to evaluate the accuracy of the lists in providing a correct match, two measures were calculated. The proportion of items for which the correct amount of the product:

1. is available in the look-up file within proximity of the cursor,
2. is the most prevalent value in the look-up file in proximity of the cursor.

For 82% of items, the correct amount of the product is available in the look-up file within proximity of the cursor (within 10 lines). Also for 42% of items, the correct amount of the product equals to the value which appears most frequently in the proximity of the cursor in the look-up file.

3. Using a SOLR-based indexing solution

The restriction of the above solution is that the look-up list and the items' order are fixed and can't be flexibly adjusted based on the input. Ideally an interactive custom list should be created in real-time that would be limited to the items which closely match the information available in the diary entry / receipt. Although preferable, this requires additional functionality not currently present in BLAISE. A solution using SOLR-based search server was proposed. Future integration into Blaise would be in a form of micro-service communication.

SOLR is an open source, Lucene-based search engine library providing scalable enterprise indexing and search technology. Initially records created from historical LCF data are indexed so that they could be

retrieved quickly based on requested criteria. By default, SOLR uses a modified TF-IDF[1] method to calculate a similarity score between the query and all available historical LCF data. SOLR offers highly customizable similarity (matching) functions.

An interactive web application prototype was created to simulate the way coders would use BLAISE accessing a micro-service. The functionality includes automatic COICOP classification based on an item's description as well as creating a small list of items which closely match the new item's description to help with the quantity entry process.

In the initial testing, SOLR based automated COICOP classification achieved 98% accuracy. We compared it with three other types of classifiers (machine learning algorithms) trained to automatically assign a COICOP code based on a product description and their accuracy was similar or worse. The most powerful aspect of this technology however is that it is also scalable i.e. the response time will remain very short even when the amount of documents, against which each query is run, is very large.

4. Conclusion

Two different promising approaches were introduced in order to assist the LCF diary input process with very encouraging initial results. Both options will be assessed by conducting user testing in order to get feedback from the LCF coders and the wider Social Survey Division team to identify the most suitable solution for long term improvements in coding time and efficiency.

[1] TF-IDF (short for **term frequency–inverse document frequency**) is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval.