Adzuna Job Analytics

(progress so far)

Data overview

Data Description

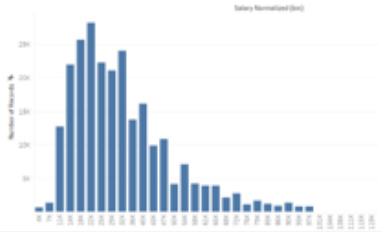
<u>Core job salary dataset</u>

244,768 training records

<u>Location tree</u>

31,763 records of drill-down locations

- For example, *UK~London~North London~Hampstead Garden Suburb*

Variable	Type	Pre-processing task
Id	Nominal	-
Title	Text	- Remove **** and other symbols - Stemming & lemmatization
FullDescription	Text	- Remove **** and other symbols - Remove stopwords - Stemming & lemmatization
LocationRaw	Text	- LocationNormalized is more standardized; LocationRaw, though noisy, provides more information regarding the specific location. Can consider mapping to LocationTree, depending on the algorithm. - Stemming & lemmatization **Note:** According to the competition winner, location mapping provided *no extra gain*; he however suspected it could be because his neural nets learned them effectively.
LocationNormalized	Text	
ContractType	Categorical (full_time, part_time, <blank>)	*179,326 missing values* - One-of-k encoding
ContractTime	Categorical (contract, permanent, <blank>)	*63,905 missing values* - One-of-k encoding
Company	Categorical	*13,454 missing values* - Standardization (e.g. removing "Pte", "Ltd") - One-of-k encoding
Category	Categorical	- One-of-k encoding
SalaryRaw	Categorical	Ignore, use SalaryNormalized instead
SalaryNormalized	Interval	Take the natural log to reduce left skew (note: this has not been implemented yet; forgot) 
SourceName	Categorical	-

Data pre-processing

Salary binning

- Library: binr
- Method: bins.greedy

<u>Binning of salary</u>

- Bins created –
  1. 5,000 – 16,000 (26,125 records)
  2. 16,002 – 20,000 (27,534 records)
  3. 20,002 – 24,000 (27,536 records)
  4. 24,002 – 27,500 (26,093 records)
  5. 27,250 – 31,680 (24,559 records)
  6. 31,681 – 35,000 (24,956 records)
  7. 35,002 – 41,500 (24,713 records)
  8. 41,508 – 50,000 (26,850 records)
  9. 50,027 – 70,000 (25,279 records)
  10. 70,080 – 200,000 (11,123 records)
- Bins made ordinal

Company data cleaning

Library: RecordLinkage
Method: RLBigDataDedup

<u>Word replacement</u>

- Replacing common words (e.g. “the”, “of”, “limited”, “consulting”, “solutions”)
- Motivation being this will affect Jaro-Winkler

<u>Jaro-Winkler de-duplicating</u>

- Identifying and associating duplicates with a Jaro-Winkler distance threshold of 85%

Data pre-processing

Document pre-processing

Library: text2vec
Methods: createIterator, create_vocabulary, create_dtm

<u>Document-term matrix</u>

- Created from merging *Title* and *FullDescription*
- Words small case, and stemmed using *SnowBallC::wordStem*
- tf-idf (term frequency-inverse document frequency) calculated using *TfIdf$new()* method

Treating categorical variables

<u>One-of-k encoding</u>

- Performed on categorical variables
  1. ContractType
  2. ContractTime
  3. Category
  4. SourceName
  5. Location missing
- Construct sparse matrix together with document-term matrix

Modelling

Cross-validated modelling

Library: caret
Methods: createFolds, trainControl

<u>5-fold cross validation</u>

- **Note:** *caret* library provides end-to-end modelling
- Example:

```
tc = trainControl("cv", index = cvIndex, number = 5 ... )

glmnet.fit = train(x.sparse, y,
                   method = "glmnet",
                   trControl = tc,
                   family = "multinomial")
```

Libraries tried

1. glmnet (logit)
2. glmnetcr (logit)
3. biglasso (logit)
4. ordinalNet (ordinal logit)
5. e1071 (SVM)

Issue: processing time

Next steps

1. Try svmlight
2. Add location feature
3. Reduce to 1,000 records