"$SO1$ Challenge"

Wuilloud Jair

03.08.2013

0.1 Task1 :

I went for python as I find that it allows for quick/relatively efficients solution to that kind of problems. Since this library is optimised, reasonably fast and allows for vectorisation (simple vector/matrix operations), I am also using numpy.

The use of numpy should make the code efficient enough while my program should work on extended (more customers, times, alphas/betas) data samples without any changes. This however under the assumption that the input file structure remains the same.

0.2 Task2 :

General Strategy/ Basic Steps/ Ideas : Details concerning substeps are provided below. The steps marked with a * could be skipped.

1. Preparatory steps :
   - Identify products of dataset2.
   - Identify categories in dataset1.
   - For datasets 1 and 2, identify and store buying occurences and the relevant details.
   - Choose a model for the probability of buying a particular product within a category.
2. Generate the hypotheses (and their statistics) :
   - for each dataset1 categories, calibrate the model parameters under the assumption that the dataset2 product belongs to that category.
   - *specialise/refine the model dealing with special cases.
   - *specialise/refine the model using other informations (time/price/...)
3. Test the hypotheses :
   - For each product of dataset2, compare the buying probabilities found with the chosen model under each hypotheses against the ones found in the market.
   - Construct a measure for the hypotheses predictive power.
   - Classify the hypotheses measuring their predictive quality
   - *eventually, extend the efficiency discussion to alternative models or strategies.
4. Draw the conclusions based on the testing step.

Details on the Preparatory Steps :

dataset2 products They are identified through their EAN.

dataset1 categories They are identified through transactional data category field.

Buying occurence and Data structure The data gathered about the buying behaviour can be structured naturally given the transactional data fields. However, we are intending to filter the data mainly through the Category and Location/Unit times/Values fields.

Model Let say that we focus on the logit model as in Task1.

Details on the hypotheses generation :

General Strategy Let's take one particular product within the dataset2. If I assume that it belongs to a particular category of dataset1, I can then calibrate the α, β's with a supervised learning algorithm. I would think of logistic or a kind of linear regression at this point. Because I wrote both algorithms recently, they were added in attachment. I would surely found the appropriate algorithm in the book you have sent me though.

As a summary, for each combinaisons of dataset2 product and dataset1 category, and for each submodels I am willing to consider, I have a set of calibrated parameters. This way, given a dataset2 product, for each category, I can use Unit/Values information gathered about a particular location to generate a probability for the product to be bought, which I write

$$P(Prod \mid Category, Location, \{Values\}, \{Unit\}),$$

whereas $\{Values\}$, $\{Unit\}$ are resp. the particular Prices and Units sets found in that particular location.

***Improved probability**

- Actually, there is more information contained within the data and one could think of adding for instance a Bayesian analysis on the top of the one described previously. This could help to get some information out of the price/customer/buytime distribution :
  $P(Category|Price)$, $P(Category|customer)$, ...,
  or even $P(Category|Price, Buytime)$, ...
  This way, one could try to refine the General strategy on the following manners :
  - Additive approach :
    $P_{tot} = \alpha P(Prod \mid Category, Location, \{Values\}, \{Unit\}) + (1-\alpha) P_{Altern.}$.

  - Multiplicative approach :
    $P_{tot} = \alpha P(Prod \mid Category, Location, \{Values\}, \{Unit\}) \times P_{Altern.}/\hat{P}_{Altern.}$,
    with $\hat{P}_{Altern.}$ the averaged probability for some alternative model.
- Eventually, it might be possible to consider more refined probabilities. For instance, if enough statistics was gathered, why not consider probabilities of the form
  $P(Prod \mid Category, Location, \{Values\}, \{Unit\}, Customer, ...)$. In that case, one would need sufficient data for calibrating the model parameters.

Details on the hypotheses testing :

Test sample statistics From the selling data, for each location and each category, for each particular product of dataset2, one can compute the products that were bought within a category :

$$p_{product \in Category} = \frac{p_{prod \in Category}}{p_{prods \in Category}},$$

with $prods = \bigcup_k prod_k$, $\forall prod_k \in Category$.

Testing with ROC surface Testing a ML model can be done on a test sample for instance using the ROC AUC metric. I have implemented the version described in Fawcett, 2005, An Introduction to ROC Analysis, which is available on the internet. My implementation is also available as attachment.

Model Classifiaction The best model is simply the one having the highest ROC AUC metric value.

Details on the Conclusion : After the sequence specified above, for each product of dataset2 one would have (hopefully) identified the best Category. The best category is the one for which the Logit model gives the highest ROC AUC metric value, after parametrisation of the model parameters.