Data Mining Cup 2014

Approach, problems and results

Benjamin Hoffmann · Daniel Kiertscher · Maik-Peter Jacob · 02.07.2014

{hoffmanb, jacobmai, Daniel.Kiertscher}@fh-brandenburg.de
Our City

- Brandenburg an der Havel ➔ 70 km west of Berlin

- More than 1000 years old, currently ~71k inhabitants

- Famous for:
  - Our lakes, green, and many cultural places (e.g. Cathedral)
  - Birgit Fischer (canoeist who won 8 Olympic gold medals)
  - Venue of 2009 Canoe Sprint European Championships and BUGA 2015 (Federal horticulture show)
About Us

• Master students at University of Applied Sciences in Brandenburg

• Founded in 1992

• 2,920 students

• Department of Informatics and Media (one of three):
  • Master project for 3 semesters about Data Mining taught by Dipl.-Inform. Ingo Boersch

  • Team 1: Daniel Kiertscher (leader) and Maik-Peter Jacob
  • Team 2: Benjamin Hoffmann (leader)
Approach

1. Learn about returns management (spadework)
2. Exploratory analysis
3. Derive / extract new features
4. Create models
5. Measure performance
6. Select model & generate / export the classification

Tools:
R 3.0.3 with constant random seeds: reproducible results

Used R packages: Hmisc, lubridate, data.table, ada, randomForest
Exploratory Analysis – Approach

- Summary statistics
- Value ranges
- Plots:
  - Mosaic plots, histogram / density plot, scatter plots
- Testing assumptions:
  - customer ID ➔ constant salutation, accountDate, state
  - item price change over time?
- Analysis of train and class set
Exploratory Analysis – Results

• itemID / manufacturerID:
  • 3007 different items, 165 different manufacturers
  
  • Only 9 items (13 rows) in CLASS are unknown
    (do not exist in TRAIN)

• Size:
  • 122 different values ("unsized", "l", "10+", "3634", "XXXL", ...)

• Color:
  • Spelling: "blau" = "blue", "brwon" = "brown", "oliv" = "olive"
  
  • Differentiation: "darkblue" != "blue"
Exploratory Analysis – Results

- **Price** equals 0:
  - 1,700 times value “0” in TRAIN (293 times in CLASS)

- On average **8 items/customer** (median: 5 items/customer)

- Potential problem: **New customers**
  - 4,369 of 12,068 customers (**36.2%**) in CLASS do **not exist** in TRAIN

- **Birth date**:  
  - 10.16% missing values  
  - 4,038 times: 19th November 1900  
  - One customer: 19th April 1655
Performance Measure

- **Split** **TRAIN** into **trainings set and test set:**
  - Test set: first and last month (orderDate) ~ 20%

- Stratified **cross validation** (k=3) on trainings set

- Measuring:
  - Resubstitution error
  - **Test error**
  - Out of bag error (oob, exclusively for Random Forests)
Feature Extraction (1)

• Features concerning **different dimensions**:
  
  1. **Group data** by
     - order (orderDate, customerID), customerID, itemID, manufacturerID

  2. **Apply aggregate functions** on different columns
     - numeric: min, max, mean, median, sum
     - nominal: most frequent, rarest, set size

  3. **“Ungroup” data**
     - i.e. insert these features into each row

• Ex.: Group by itemID, calculate mean price & insert into every row
Feature Extraction (2)

- Add **additional information** (from external sources)

- **states:**
  - add population, area, population density, income, ...
  - ranking (converting a nominal feature to numeric)

- **colors:**
  - Convert to RGB and HSV (as far as possible)
  - Ignore problem “colors”:
    - leopard, striped, stained, nature
    - new feature
Feature Extraction (3)

- **Ratios** (73 derived features):
  - Idea:
    - Ratios (if not included) might pose a problem for tree learning algorithms
    - Combining features:
      - row specific values & order/item/... specific values
  - Examples:
    - order item price / mean price of the item
    - customer age / mean age of customer ordering this item
    - order item price / customer age
Feature Extraction (4)

- **Choice order** item:
  - Number of items with the same itemID in a single order (orderDate, customerID) with different sizes / colors

- **Item groups**:
  1. According to the three “bumps” in the itemID histogram
  2. According to their sizes:
     - Group by item and look at all possible sizes
     - (semi-)automatically assign item group, e.g. “s/m/l”, “80-110 (mod 5 == 0)”, “104-176” items
     - Difficult/error-prone for items that are rarely bought
Feature Extraction (5)

- **Package ID** (same order, different delivery date)

- Item/Customer/Manufacturer “returnShipment” rate (mean) including the confidence interval

- **Unused feature ideas** (no influence or too complicated):
  - Temporal distance of order and delivery to public holiday
  - Brute force grouping (automatic feature definition)

In total: **263 features**
Model Creation

• **Focus on Random Forest**

  ✓ Performant implementation in R
  ✓ Copes well with many features (robust)
  ✓ Additional internal error estimation

  – Not very transparent
  – Memory hungry

• A quick comparison test between random forest and AdaBoost favoured RF
Model Creation – Team 2’s Idea

- **Two models** (random forests, nodesize=100, ntree=100)
  - One for *well-known* customers
    - All features **including** customer returnShipment rate and confidence interval
    - Ideally includes more transaction history
  - One for **not well-known** customers
    - All features **excluding** customer returnShipment rate and confidence interval
    - No Transaction history present

*well-known customers: > 2 orders*
Model Creation – Team 1’s Idea

• **Three models** (all random forests)

1. **M1**: One Random Forest classifier, all features excluding customer return shipment rate (nodesize=100, ntree=200)

2. **M2**: Two Random Forests (team 2’s approach)

3. **M3**: One random forest with hand chosen features (e.g. no return shipment rates)

⇒ Simple **Majority vote** returns the final result
## Setup Performances

<table>
<thead>
<tr>
<th>Setup</th>
<th>CV success</th>
<th>Test success</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>M2</strong>: Two random forest</td>
<td>72.78%</td>
<td>67.70%</td>
</tr>
<tr>
<td><strong>M1</strong>: One random forest</td>
<td>69.33%</td>
<td>67.25%</td>
</tr>
<tr>
<td><strong>M3</strong>: One random forest + chosen features</td>
<td>69.02%</td>
<td>64.95%</td>
</tr>
</tbody>
</table>

- **MC (combined model) = M1 + M2 + M3**
Exported Classification

voting difference in percent
final models
of the two teams

M1

M2

M3

MC

18.7%
3.9%
12.0%
14.8%
8.1%
22.9%
Most Important Features (mean decrease gini, RF)

- **Choice order** (same item, different sizes) ratio
- **Account age**
- **Package ID:**
  - Package number / number of total packages
- **Price:**
  - sum of entire order
  - Max sum spent for one order for each customer
- **State** (poverty ranking)
- **Delivery time:**
  - Ratio:
    - delivery time / average delivery time of the same item
  - Weekday
- **Item return shipment rate** (lower/upper boundary, mean)
Problems

- Huge amount of data
  - Memory limit reached during cross validation (8 GB)

- Data issues:
  - Missing values
  - Colors / sizes hard to make sense of
  - Huge differences in size of customer transaction history
  - Missing information about items (item groups, item description, item rating, ...)

- Time constraint (as usual)
  - Reuse last year’s code
Expectations

- We only used 0 or 1 as predictions (no values in between)

- Team 1:
  - Majority voting
  - Exported classification close to one created with a setup that had an approximate 67% (= 16,526 points) test accuracy

- Team 2:
  - One model consisting of two random forests
  - Approx. 68% test accuracy (= 16,025 points)

- Since our test set was harder than their test set, we expect slightly better performances! (assumption)
Keys To Success

- Reproducibility

- Outstanding features (but do not miss simple ones!)

- Competitive learning algorithm

- Reliable estimation of model error for selection → permanently improved baseline

- Weekly team meetings with retrospective and prospective discussions

- Master the tools and keep on the watch for useful libraries
Thank you for your attention!

Any questions?

Benjamin Hoffmann · Daniel Kiertscher · Maik-Peter Jacob · 02.07.2014

{hoffmanb, jacobmai, Daniel.Kiertscher}@fh-brandenburg.de