The CLVTools Package

1 Walkthrough for the CLVTools package

1.1 Setup the R environment

Install the stable version from CRAN:

```R
install.packages("CLVTools")
```

Install the development version from GitHub (using the devtools (Wickham, Hester, and Chang 2019) package):

```R
install.packages("devtools")
devtools::install_github("bachmannpatrick/CLVTools", ref = "master")
```

Load the package

```R
library("CLVTools")
```

2 Load sample data provided in the package

As Input data CLVTools requires customers’ transaction history. Every transaction record consists of a purchase date and customer ID. Optionally, the price of the transaction may be included to allow for prediction of future customer spending using an additional Gamma/Gamma model (Fader, Hardie, and Lee 2005b; Colombo and Jiang 1999). Using the full history of transaction data allows for comprehensive plots and summary statistics, which allow the identification of possible issues prior to model estimation. Data may be provided as `data.frame` or `data.table` (Dowle and Srinivasan 2019).

It is common practice to split time series data into two parts, an estimation and a holdout period. The model is estimated based on the data from the estimation period while the data from the holdout period allows to rigorously assess model performance. Once model performance is checked on known data one can proceed to predict data without a holdout period. The length of the estimation period is heavily dependent on the characteristics of the analyzed dataset. We recommend to choose an estimation period that contains in minimum the length of the average inter-purchase time. Note that all customers in the dataset need to purchase at least once during the estimation period, i.e. these models do not account for prospects who have not yet a purchase record.

Some models included in CLVTools allow to model the impact of covariates. These covariates may explain heterogeneity among the customers and therefore increase the predictive accuracy of the model. At the same time we may also identify and quantify the effects of these covariates on customer purchase and customer attrition. CLVTools distinguishes between time-invariant and time-varying covariates. Time-invariant covariates include customer characteristics such as demographics that do not change over time. Time-varying covariates are allowed to change over time. They include for example direct marketing information or seasonal patterns.

For the following example, we use simulated data comparable to data from a retailer in the apparel industry. The dataset contains transactional detail records for every customer consisting of customer id, date of
purchase and the total monetary value of the transaction. The apparel dataset is available in the `CLVTools` package. Use the `data(apparelTrans)` to load it:

```r
data("apparelTrans")

apparelTrans
#> # A tibble: 2,353 × 3
#> # Row.names   Date       Price
#>   <chr>        <date>     <dbl>
#> 1 1            2005-01-03  26.95
#> 2 10           2005-01-03  38.95
#> 3 10           2005-02-25  93.73
#> 4 10           2005-04-05 224.96
#> 5 100          2005-01-03 104.95
#> ---
#> 2349 1221 2006-01-23  62.95
#> 2350 1221 2006-03-09  89.95
#> 2351 1221 2006-05-14  52.95
#> 2352 1222 2005-01-03  5.90
#> 2353 1222 2005-03-03  13.90
```

2.1 Initialize the CLV-Object

Before we estimate a model, we are required to initialize a data object using the `clvdata()` command. The data object contains the prepared transactional data and is later used as input for model fitting. Make sure to store the generated object in a variable, e.g. in our example `clv.apparel`.

Through the argument `data.transactions` a `data.frame` or `data.table` which contains the transaction records, is specified. In our example this is `data.transactions=apparelTrans`. The argument `date.format` is used to indicate the format of the date variable in the data used. The date format in the apparel dataset is given as “year-month-day” (i.e., “2005-01-03”), therefore we set `date.format="ymd"`. Other combinations such as `date.format="dmy"` are possible. See the documentation of `lubridate` (Grolemund and Wickham 2011) for all details. `time.unit` is the scale used to measure time between two dates. For this dataset and in most other cases The argument `time.unit="week"` is the preferred choice. Abbreviations may be used (i.e. “w”). `estimation.split` indicates the length of the estimation period. Either the length of the estimation period (in previous specified time units) or the date at which the estimation period ends can be specified. If no value is provided, the whole dataset is used as estimation period (i.e. no holdout period). In this example, we use an estimation period of 40 weeks. Finally, the three name arguments indicate the column names for customer ID, date and price in the supplied dataset. Note that the price column is optional.

```r
clv.apparel <- clvdata(apparelTrans,
                         date.format = "ymd",
                         time.unit = "week",
                         estimation.split = 40,
                         name.id = "Id",
                         name.date = "Date",
                         name.price = "Price")
```

2.2 Check the clvdata Object

To get details on the `clvdata` object, print it to the console.

```r
clv.apparel
#> CLV Transaction Data
#> # A tibble: 2,353 × 3
#> # Row.names   Date       Price
#>   <chr>        <date>     <dbl>
#> 1 1            2005-01-03  26.95
#> 2 10           2005-01-03  38.95
#> 3 10           2005-02-25  93.73
#> 4 10           2005-04-05 224.96
#> 5 100          2005-01-03 104.95
#> ---
#> 2349 1221 2006-01-23  62.95
#> 2350 1221 2006-03-09  89.95
#> 2351 1221 2006-05-14  52.95
#> 2352 1222 2005-01-03  5.90
#> 2353 1222 2005-03-03  13.90
```
Alternatively the `summary()` command provides full detailed summary statistics for the provided transactional detail. `summary()` is available at any step in the process of estimating a probabilistic customer attrition model with CLVTools. The result output is updated accordingly and additional information is added to the summary statistics. `nobs()` extracts the number of observations. For this particular dataset we observe a total of 250 customers who made in total 2257 repeat purchases. Approximately 26% of the customers are zero repeaters, which means that the only a minority of the customers do not return to the store after their first purchase.

```
summary(clv.apparel)
```

2.3 Estimate Model Parameters

After initializing the object, we are able to estimate the first probabilistic latent attrition model. We start with the standard Pareto/NBD model (Schmittlein, Morrison, and Colombo 1987) and therefore use the
command \texttt{pnbd()} to fit the model and estimate model parameters. \texttt{clv.data} specifies the initialized object prepared in the last step. Optionally, starting values for the model parameters and control settings for the optimization algorithm may be provided: The argument \texttt{start.params.model} allows to assign a vector (e.g. \texttt{c(alpha=1, beta=2, s=1, beta=2)} in the case of the Pareto/NBD model) of starting values for the optimization. This is useful if prior knowledge on the parameters of the distributions are available. By default starting values are set to 1 for all parameters. The argument \texttt{optimx.args} provides an option to control settings for the optimization routine. It passes a list of arguments to the optimizer. All options known from the package \texttt{optimx} (Nash and Varadhan 2011; Nash 2014) may be used. This option enables users to specify specific optimization algorithms, set upper and/or lower limits or enable tracing information on the progress of the optimization. In the case of the standard Pareto/NBD model, \texttt{CLVTools} uses by default the optimization method \texttt{L-BFGS-G} (Byrd et al. 1995). If the result of the optimization is in-feasible, the optimization automatically switches to the more robust but often slower \texttt{Nelder-Mead} method (Nelder and Mead 1965). \texttt{verbose} shows additional output.

```
est.pnbd <- pnbd(clv.data = clv.apparel)
#> Starting estimation...
#> Estimation finished!
est.pnbd
#> Pareto NBD Standard Model
#>
#> Call:  
#> pnbd(clv.data = clv.apparel)  
#> 
#> Coefficients:  
#> r   alpha   s   beta
#> 0.7867  5.3356  0.3574  11.6316
#> KKT1: TRUE
#> KKT2: TRUE
#>
#> Used Options:  
#> Correlation: FALSE
```

Parameter estimates may be reported by either printing the estimated object (i.e. \texttt{est.pnbd}) directly in the console or by calling \texttt{summary(est.pnbd)} to get a more detailed report including the likelihood value as well as AIC and BIC. Alternatively parameters may be directly extracted using \texttt{coef(est.pnbd)}. Also \texttt{loglik()}, \texttt{confint()} and \texttt{vcov()} are available to directly access the Loglikelihood value, confidence intervals for the parameters and to calculate the Variance-Covariance Matrix for the fitted model. For the standard Pareto/NBD model, we get 4 parameters \(r, \alpha, s, \beta\). where \(r, \alpha\) represent the shape and scale parameter of the gamma distribution that determines the purchase rate and \(s, \beta\) of the attrition rate across individual customers. \(r/\alpha\) can be interpreted as the mean purchase and \(s/\beta\) as the mean attrition rate. A significance level is provided for each parameter estimates. In the case of the apparelTrans dataset we observe a an average purchase rate of \(r/\alpha = 0.147\) transactions and an average attrition rate of \(s/\beta = 0.031\) per customer per week. KKT 1 and 2 indicate the Karush-Kuhn-Tucker optimality conditions of the first and second order (Kuhn and Tucker 1951). If those criteria are not met, the optimizer has probably not arrived at an optimal solution. If this is the case it is usually a good idea to rerun the estimation using alternative starting values.

```
#Full detailed summary of the parameter estimates
summary(est.pnbd)
```

If we assign starting parameters and additional arguments for the optimizer we use:

```
est.pnbd <- pnbd(clv.data = clv.apparel,  
  start.params.model = c(r=1, alpha = 2, s = 1, beta = 2),  
  optimx.args = list(control=list(trace=5),
               method="Nelder-Mead")
)```

Parameter estimates may be reported by either printing the estimated object (i.e. \texttt{est.pnbd}) directly in the console or by calling \texttt{summary(est.pnbd)} to get a more detailed report including the likelihood value as well as AIC and BIC. Alternatively parameters may be directly extracted using \texttt{coef(est.pnbd)}. Also \texttt{loglik()}, \texttt{confint()} and \texttt{vcov()} are available to directly access the Loglikelihood value, confidence intervals for the parameters and to calculate the Variance-Covariance Matrix for the fitted model. For the standard Pareto/NBD model, we get 4 parameters \(r, \alpha, s, \beta\). where \(r, \alpha\) represent the shape and scale parameter of the gamma distribution that determines the purchase rate and \(s, \beta\) of the attrition rate across individual customers. \(r/\alpha\) can be interpreted as the mean purchase and \(s/\beta\) as the mean attrition rate. A significance level is provided for each parameter estimates. In the case of the apparelTrans dataset we observe a an average purchase rate of \(r/\alpha = 0.147\) transactions and an average attrition rate of \(s/\beta = 0.031\) per customer per week. KKT 1 and 2 indicate the Karush-Kuhn-Tucker optimality conditions of the first and second order (Kuhn and Tucker 1951). If those criteria are not met, the optimizer has probably not arrived at an optimal solution. If this is the case it is usually a good idea to rerun the estimation using alternative starting values.

```
#Full detailed summary of the parameter estimates
summary(est.pnbd)
```
# Pareto NBD Standard Model

## Call:
```r
pnbd(clv.data = clv.apparel)
```

## Fitting period:
- Estimation start: 2005-01-03
- Estimation end: 2005-10-10
- Estimation length: 40.0000 Weeks

## Coefficients:

| Parameter | Estimate | Std. Error | z-val  | Pr(>|z|) |
|-----------|----------|------------|--------|----------|
| r         | 0.7867   | 0.1324     | 5.942  | 2.82e-09 *** |
| alpha     | 5.3356   | 0.9028     | 5.910  | 3.42e-09 *** |
| s         | 0.3574   | 0.1841     | 1.941  | 0.0523 .   |
| beta      | 11.6316  | 10.6823    | 1.089  | 0.2762    |

---

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## Optimization info:
- LL: -2879.4699
- AIC: 5766.9399
- BIC: 5781.0257
- KKT 1 TRUE
- KKT 2 TRUE
- fevals: 20.0000
- Method: L-BFGS-B

## Used Options:
- Correlation: FALSE

To extract only the coefficients, we can use `coef()`. To access the confidence intervals for all parameters `confint()` is available.

To extract only the coefficients, we can use `coef()`. To access the confidence intervals for all parameters `confint()` is available.

```r
#Extract the coefficients only
coefficients(est.pnbd)
```

```r
r alpha s beta
0.7866688 5.3355963 0.3573849 11.6316385
```

```r
#Alternative: coefficients(est.pnbd.obj)
```

#Extract the confidence intervals
```r
confint(est.pnbd)
```

```r
r alpha s beta
2.5 % 97.5 %
0.5271724 1.0461651
3.566110956 7.1050816
-0.003526077 0.7182959
-9.305344078 32.5686211
```
In order to get the Likelihood value and the corresponding Variance-Covariance Matrix we use the following commands:

```r
# LogLikelihood at maximum
logLik(est.pnbd)
#> 'log Lik.' -2879.47 (df=4)

# Variance-Covariance Matrix at maximum
vcov(est.pnbd)
#>             r     alpha     s    beta
#> r  0.01752937 0.1025851 -0.0056323 -0.5554541
#> alpha 0.102585075 0.8150753 -0.0239615 -2.4694235
#> s -0.005632316 -0.0239615 0.0339081 1.8589624
#> beta -0.555454141 -2.4694235 1.8589624 114.1121797
```

As an alternative to the Pareto/NBD model CLVTTools features the BG/NBD model (Fader, Hardie, and Lee 2005a) and the GGomp/NBD (Bemmaor and Glady 2012). To use the alternative models replace `pnbd()` by the corresponding model-command. Note that the naming and number of model parameters is dependent on the model. Consult the manual for more details on the individual models.

<table>
<thead>
<tr>
<th>Command</th>
<th>Model</th>
<th>Covariates</th>
</tr>
</thead>
<tbody>
<tr>
<td>pnbd()</td>
<td>Pareto/NBD</td>
<td>time-invariant &amp; time-varying</td>
</tr>
<tr>
<td>bgnbd()</td>
<td>BG/NBD</td>
<td>time-invariant</td>
</tr>
<tr>
<td>ggomnbd()</td>
<td>GGom/NBD</td>
<td>time-invariant</td>
</tr>
</tbody>
</table>

To estimate the GGom/NBD model we apply the `ggomnbd()` to the `clv.apparel` object. The GGom/NBD model is more flexible than the Pareto/NBD model, however it sometimes is challenging to optimize. Note that in this particular case providing start parameters is essential to arrive at an optimal solution (i.e. `kkt1`: TRUE and `kkt2`: TRUE).

```r
est.ggomnbd <- ggomnbd(clv.data = clv.apparel, 
                        start.params.model = c(r=0.7, alpha=5, b=0.005, s=0.02, beta=0.001), 
                        optimx.args = list(control=list(trace=5), 
                                            method="Nelder-Mead"))
```

### 2.4 Predicting Customer Behavior

Once the model parameters are estimated, we are able to predict future customer behavior on an individual level. To do so, we use `predict()` on the object with the estimated parameters (i.e. `est.pnbd`). The prediction period may be varied by specifying `prediction.end`. It is possible to provide either an end-date or a duration using the same time unit as specified when initializing the object (i.e `prediction.end = "2006-05-08"` or `prediction.end = 30`). By default, the prediction is made until the end of the dataset specified in the `clvdata()` command. The argument `continuous.discount.factor` allows to adjust the discount rate used to estimated the discounted expected transactions (DERT). The default value is 0.1 (=10%). Probabilistic customer attrition model predict in general three expected characteristics for every customer:

- “conditional expected transactions” (CET), which is the number of transactions to expect form a customer during the prediction period,
- “probability of a customer being alive” (PAlive) at the end of the estimation period and
- “discounted expected residual transactions” (DERT) for every customer, which is the total number of transactions for the residual lifetime of a customer discounted to the end of the estimation period.
If spending information was provided when initializing the clvdata-object, CLVTools provides prediction for

- predicted spending estimated by a Gamma/Gamma model (Colombo and Jiang 1999; Fader, Hardie, and Lee 2005a) and
- the customer lifetime value (CLV).

If a holdout period is available additionally the true numbers of transactions (“actual.x”) and true spending (“actual.spending”) during the holdout period are reported.

To use the parameter estimates on new data (e.g., an other customer cohort), the argument newdata optionally allows to provide a new clvdata object.

```r
c results <- predict(est.pnbd)
## Predicting from 2005-10-11 until (incl.) 2006-07-16 (39.86 Weeks).
print(results)
## PAlive CET DERT predicted.Spending predicted.CLV
## 1: 0.3571358 0.2212240 0.05848369 216.7955 12.67900
## 2: 0.4224409 0.9269543 0.24505345 209.1146 51.24426
## 3: 0.9155010 13.5430229 3.58028916 188.9758 676.58785
## 4: 0.9967760 13.1755612 3.48314547 171.0913 595.93576
## 5: 0.5096716 3.5263202 0.93223249 229.8529 214.27636
## 246: 0.9578990 3.6104399 0.95447073 206.0859 196.70295
## 247: 0.3571358 0.2212240 0.05848369 216.7955 12.67900
## 248: 0.3571358 0.2212240 0.05848369 216.7955 12.67900
## 249: 0.9433972 4.2986317 1.13640393 207.0518 235.29447
## 250: 0.4135069 0.5817466 0.15379292 208.3194 32.03804
```

To change the duration of the prediction time, we use the prediction.end argument. We can either provide a time period (30 weeks in this example):
```
predict(est.pnbd, prediction.end = 30)
```

or provide a date indication the end of the prediction period:
```
predict(est.pnbd, prediction.end = "2006-05-08")
```

### 2.5 Model Plotting

clvdata objects may be plotted using the plot() command. Similar to summary(), the output of plot() adapts to the current modeling step. It provides a descriptive plot of the actual transactional data if the model has not yet been fitted. Once the model has been estimated, plot() provides an aggregated incremental tracking plot of the actual data and the model based on the estimated parameters. The time-span for the
plot may be altered using the `prediction.end` argument by providing either a duration or an end date. By default the plot is generated for the entire time-span of the provided dataset specified in the `clvdata()` command. The dashed line indicates the end of the estimation period. Alternatively cumulative actual and expected transactions can be plotted by setting `cumulative` to `TRUE`. The argument `transactions` disable for plotting actual transactions (`transactions=FALSE`). For further plotting options see the documentation.

```r
plot(est.pnbd)
#> Plotting from 2005-01-03 until 2006-07-16.
```

To plot the cumulative expected transactions 30 time units (30 weeks in this example) ahead of the end of the estimation plot, we use:

```r
plot(est.pnbd, prediction.end = 30, cumulative = TRUE)
```

Alternatively, it is possible to specify a date for the `prediction.end` argument. Note that dates are rounded to the next full time unit (i.e. week):

```r
plot(est.pnbd, prediction.end = "2006-05-08", cumulative = TRUE)
```

### 2.6 Covariates

`CLVTools` provides the option to include covariates into probabilistic customer attrition models. Covariates may affect the purchase or the attrition process, or both. It is also possible to include different covariates for the two processes. However, support for covariates is dependent on the model. Not all implemented models provide the option for covariates. In general, `CLVTools` distinguishes between two types of covariates: time-invariant and time-varying. The former include factors that do not change over time such as customer demographics or customer acquisition information. The latter may change over time and include marketing activities or seasonal patterns.

Data for time-invariant covariates must contain a unique customer ID and a single value for each covariate. It should be supplied as a `data.frame` or `data.table`. In the example of the apparel retailer we use demographic information “gender” as time-invariant and information on the acquisition channel as covariate for both, the purchase and the attrition process. Use the `data("apparelStaticCov")` command to load the
time-invariant covariates. In this example gender is coded as a dummy variable with \texttt{male}=0 and \texttt{female}=1 and channel with \texttt{online}=0 and \texttt{offline}=1.

```
data("apparelStaticCov")
apparelStaticCov
define
  Id Gender Channel
  1: 1 0 0
  2: 10 0 0
  3: 100 1 0
  4: 1000 1 1
  5: 1001 1 0
  ...  
  246: 1219 0 1
  247: 122 0 0
  248: 1220 0 0
  249: 1221 1 1
  250: 1222 1 0
```

Data for time-varying covariates requires a time-series of covariate values for every customer. I.e. if the time-varying covariates are allowed to change every week, a value for every customer for every week is required. Note that all contextual factors are required to use the same time intervals for the time-series. In the example of the apparel retailer we use information on direct marketing (Marketing) as time-varying covariate. Additionally, we add gender as time-invariant contextual factors. Note that the data structure of invariant covariates needs to be aligned with the structure of time-varying covariate. Use \texttt{data("apparelDynCov")} command to load

```
data("apparelDynCov")
apparelDynCov
defining
  Id Cov.Date Marketing Gender Channel
  1: 1 2004-12-26 1 0 0
  2: 1 2005-01-02 1 0 0
  3: 1 2005-01-09 0 0 0
  4: 1 2005-01-16 1 0 0
  5: 1 2005-01-23 2 0 0
  ...  
  20496: 1222 2006-06-18 0 1 0
  20497: 1222 2006-06-25 0 1 0
  20498: 1222 2006-07-02 0 1 0
  20499: 1222 2006-07-09 0 1 0
  20500: 1222 2006-07-16 0 1 0
```

To add the covariates to an initialized \texttt{clvdata} object the commands \texttt{SetStaticCovariates()} and \texttt{SetDynamicCovariates()} are available. The two commands are mutually exclusive. The argument \texttt{clv.data} specifies the initialized object and the argument \texttt{data.cov.life} respectively \texttt{data.cov.trans} specifies the data source for the covariates for the attrition and the purchase process. Covariates are added separately for the purchase and the attrition process. Therefore if a covariate should affect both processes it has to be added in both arguments: \texttt{data.cov.life} and \texttt{data.cov.trans}. The arguments \texttt{names.cov.life} and \texttt{names.cov.trans} specify the column names of the covariates for the two processes. In our example, we use the same covariates for both processes. Accordingly, we specify the time-invariant covariates “Gender” and “Channel” as follows:

```
clv.static<- SetStaticCovariates(clv.data = clv.apparel, 
data.cov.life = apparelStaticCov, 
data.cov.trans = apparelStaticCov, 
names.cov.life = c("Gender", "Channel"), 
names.cov.trans =c("Gender", "Channel"), 
```

9
To specify the time-varying contextual factors for seasonal patterns and direct marketing, we use the following:

```r
clv.dyn <- SetDynamicCovariates(clv.data = clv.apparel, 
data.cov.life = apparelDynCov, 
data.cov.trans = apparelDynCov, 
names.cov.life = c("Marketing", "Gender", "Channel"), 
names.cov.trans = c("Marketing", "Gender", "Channel"), 
name.id = "Id", 
name.date = "Cov.Date")
```

In order to include time-invariant covariates in a time-varying model, they may be recoded as a time-varying covariate with a constant value in every time period.

Once the covariates are added to the model the estimation process is almost identical to the standard model without covariates. The only difference is that the provided object now data for contains either time-invariant or time-varying covariates and the option to define start parameters for the covariates of both processes using the arguments `start.params.life` and `start.params.trans`. If not set, the staring values are set to 1. To define starting parameters for the covariates, the name of the corresponding factor has to be used. For example in the case of time-invariant covariates:

```r
est.pnbd.static <- pnbd(clv.static, 
start.params.model = c(r=1, alpha = 2, s = 1, beta = 2), 
start.params.life = c(Gender=0.6, Channel=0.4), 
start.params.trans = c(Gender=0.6, Channel=0.4))
```

To inspect the estimated model we use `summary()`, however all other commands such as `print()`, `coef()`, `loglike()`, `confint()` and `vcov()` are also available. Now, output contains also parameters for the covariates for both processes. Since covariates are added separately for the purchase and the attrition process, there are also separate model parameters for the two processes. These parameters are directly interpretable as rate elasticity of the corresponding factors: A 1% change in a contextual factor $X^P$ or $X^L$ changes the purchase or the attrition rate by $\gamma_purch X^P$ or $\gamma_life X^L$ percent, respectively (Gupta 1991). In the example of the apparel retailer, we observe that female customer purchase significantly more ($\text{trans.Gender}=1.42576$). Note, that female customers are coded as 1, male customers as 0. Also customers acquired offline (coded as Channel=1), purchase more ($\text{trans.Channel}=0.40304$) and stay longer ($\text{life.Channel}=0.9343$). Make sure to check the Karush-Kuhn-Tucker optimality conditions of the first and second order (Kuhn and Tucker 1951) (KKT1 and KKT1) before interpreting the parameters. If those criteria are not met, the optimizer has probably not arrived at an optimal solution. If this is the case it is usually a good idea to rerun the estimation using alternative starting values.

```r
summary(est.pnbd.static)
```

Analogously, we can estimate the model containing time-varying covariates. We recommend to enable the built-in support for multithreading when estimating more complex models like this. See section Multithreading.

```r
est.pnbd.dyn <- pnbd(clv.dyn, 
start.params.model = c(r=1, alpha = 2, s = 1, beta = 2), 
start.params.life = c(Marketing=0.5, Gender=0.6, Channel=0.4), 
start.params.trans = c(Marketing=0.5, Gender=0.6, Channel=0.4))
```

To inspect the estimated model we use `summary()`, however all other commands such as `print()`, `coef()`, `loglike()`, `confint()` and `vcov()` are also available. Now, output contains also parameters for the covariates for both processes. Since covariates are added separately for the purchase and the attrition process, there are also separate model parameters for the two processes. These parameters are directly interpretable as rate elasticity of the corresponding factors: A 1% change in a contextual factor $X^P$ or $X^L$ changes the purchase or the attrition rate by $\gamma_purch X^P$ or $\gamma_life X^L$ percent, respectively (Gupta 1991). In the example of the apparel retailer, we observe that female customer purchase significantly more ($\text{trans.Gender}=1.42576$). Note, that female customers are coded as 1, male customers as 0. Also customers acquired offline (coded as Channel=1), purchase more ($\text{trans.Channel}=0.40304$) and stay longer ($\text{life.Channel}=0.9343$). Make sure to check the Karush-Kuhn-Tucker optimality conditions of the first and second order (Kuhn and Tucker 1951) (KKT1 and KKT1) before interpreting the parameters. If those criteria are not met, the optimizer has probably not arrived at an optimal solution. If this is the case it is usually a good idea to rerun the estimation using alternative starting values.

```r
summary(est.pnbd.static)
```
To predict future customer behavior we use `predict()`. Note that dependent on the model, the predicted metrics may differ. For example, in the case of the Pareto/NBD model with time-varying covariates, instead of DERT, DECT is predicted. DECT only covers a finite time horizon in contrast to DERT. Time-varying covariates must be provided for the entire prediction period. If the data initially provided in the `SetDynamicCovariates()` command does not cover the complete prediction period, the argument `new.data` offers the ability to supply new data for the time-varying covariates in the from of a `clvdata` object.

### 2.7 Add Correlation to the model

To relax the assumption of independence between the purchase and the attrition process, CLVTools provides the option to specify the argument `use.cor` in the command to fit the model (i.e. `pnbd`). In case of `use.cor=TRUE`, a Sarmanov approach is used to correlate the two processes. `start.param.cor` allows to optionally specify a starting value for the correlation parameter.

```r
est.pnbd.cor <- pnbd(clv.apparel, 
                      use.cor= TRUE)
summary(est.pnbd.cor)
```
The parameter Cor(life,trans) is added to the parameter estimates that may be directly interpreted as a correlation. In the example of the apparel retailer the correlation parameter is not significant and the correlation is very close to zero, indicating that the purchase and the attrition process are independent.

2.8 Advanced Options for Contextual Factors

CLVTools provides two additional estimation options for models containing covariates (time-invariant or time-varying): regularization and constraints for the parameters of the covariates. Both options are included in the command to fit the model (i.e., \texttt{pnbd()}). Support for this option is dependent on the model. They may be used simultaneously.

- The argument \texttt{reg.lambdas} provides the possibility to specify separate $\lambda_{\text{reg}}$ for the two processes (i.e. \texttt{reg.lambdas = c(trans=100, life=100)}. The larger the $\lambda_{\text{reg}}$ the stronger the effects of the regularization. Regularization only affects the parameters of the covariates.

- The argument \texttt{names.cov.constr} implements equality constraints for contextual factors with regards to the two processes. For example the variable “gender” is forced to have the same effect on the purchase as well as on the attrition process. To do so, the option \texttt{names.cov.constr} is available (i.e. \texttt{names.cov.constr=c("Gender")}). To provide starting parameters for the constrained variable use \texttt{start.params.constr}.

To enable regularization for the covariates, we use the following command:

\begin{verbatim}
est.pnbd.reg <- pnbd(clv.static,  
    start.params.model = c(r=1, alpha = 2, s = 1, beta = 2),  
    reg.lambdas = c(trans=100, life=100))
\end{verbatim}

\texttt{#> Starting estimation...}  
\texttt{#> Estimation finished!}  
\texttt{summary(est.pnbd.reg)}  
\texttt{#> Pareto NBD with Static Covariates Model}  
\texttt{#> Call:}  
\texttt{#> pnbd(clv.data = clv.static, start.params.model = c(r = 1, alpha = 2,}  
\texttt{#> s = 1, beta = 2), reg.lambdas = c(trans = 100, life = 100))}  
\texttt{#> Fitting period:}  
\texttt{#> Estimation start 2005-01-03}  
\texttt{#> Estimation end 2005-10-10}  
\texttt{#> Estimation length 40.0000 Weeks}  
\texttt{#> Coefficients:}  
\texttt{#> }  
\texttt{#> Estimate Std. Error z-val Pr(>|z|)}  
\texttt{#> r 7.928e-01 2.117e+00 0.375 0.708}  
\texttt{#> alpha 5.393e+00 1.443e+01 0.374 0.709}  
\texttt{#> s 3.603e-01 2.946e+00 0.122 0.903}  
\texttt{#> beta 1.176e+01 1.711e+02 0.069 0.945}  
\texttt{#> life.Gender -2.448e-05 1.712e-02 -0.001 0.999}  
\texttt{#> life.Channel -1.163e-04 1.712e-02 -0.007 0.995}  
\texttt{#> trans.Gender 5.391e-04 1.712e-02 0.031 0.975}  
\texttt{#> trans.Channel 4.498e-04 1.712e-02 0.026 0.979}  
\texttt{#> Optimization info:}  
\texttt{#> LL -11.5178}  
\texttt{#> AIC 39.0357}  
\texttt{#> BIC 67.2074}
To constrain “Gender” to have the same effect on both processes we use the following command. Note, that the output now only contains one parameter for “Gender” as it is constrained to be the same for both processes.

```r
est.pnbd.constr <- pnbd(clv.static,
  start.params.model = c(r=1, alpha = 2, s = 1, beta = 2),
  start.params.constr = c(Gender=0.6),
  names.cov.constr=c("Gender"))
```

```r
# Starting estimation...
# Estimation finished!
summary(est.pnbd.constr)
# Pareto NBD with Static Covariates Model
#
# Call:
# pnbd(clv.data = clv.static, start.params.model = c(r = 1, alpha = 2,
# s = 1, beta = 2), names.cov.constr = c("Gender"), start.params.constr = c(Gender = 0.6))
#
# Fitting period:
# Estimation start 2005-01-03
# Estimation end 2005-10-10
# Estimation length 40.0000 Weeks
#
# Coefficients:
# Estimate Std. Error  z-val Pr(>|z|)
# r    1.42323   0.27536  5.169  2.36e-07 ***
# alpha 35.45252  8.37875  4.231  2.32e-05 ***
# s     0.27408   0.09527  2.877   0.00402 **
# beta  7.87475  7.67175  1.026   0.30467
# life.Channel -1.68821  0.64231 -2.628  0.00858 **
# trans.Channel 0.40185  0.15102  2.661  0.00779 **
# constr.Gender 1.41567  0.18405  7.692  1.44e-14 ***
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Optimization info:
# LL   -2846.1729
# AIC  5706.3458
# BIC  5730.9960
# KKT 1 TRUE
# KKT 2 TRUE
# fevals 39.0000
# Method L-BFGS-B
```
# Used Options:

- Correlation: FALSE
- Regularization: FALSE
- Constraint covs: TRUE
- Constraint params Gender

2.9 Multithreading: Enable parallel processing for CLVTools

CLVTools supports parallel processing when estimating models containing time-varying covariates using the package `future` (Bengtsson 2020b) and the corresponding `foreach` (Microsoft and Weston 2019) parallel adapter `doFuture` (Bengtsson 2020a). To enable the distribution of the workload across all available cores, use execute the following commands before estimating the model (i.e. calling `pnbd()`):

```r
# disable multithreading for data.table (to avoid nested parallelism)
setDTthreads(1)

library("doFuture")
registerDoFuture()
plan("multisession")
```

To limit the number of parallel processes simultaneously executed, we can specify the number of workers `plan(multisession, workers = 2)`.

Literature


