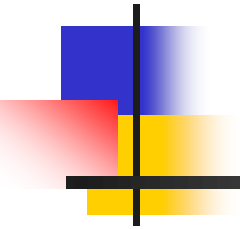


Recap

Types of Data Munging

Record(Row) Operations

filter()

rbind()

arrange()

sample_n()
sample_frac()

group_by() + summarize()

distinct()

Field(Column) Operations

mutate()

select()
rename()

select()

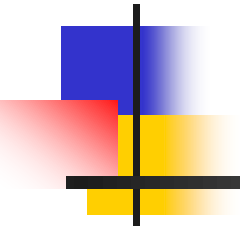
is_na() +
mutate()

inner_join()
left_join()

t()

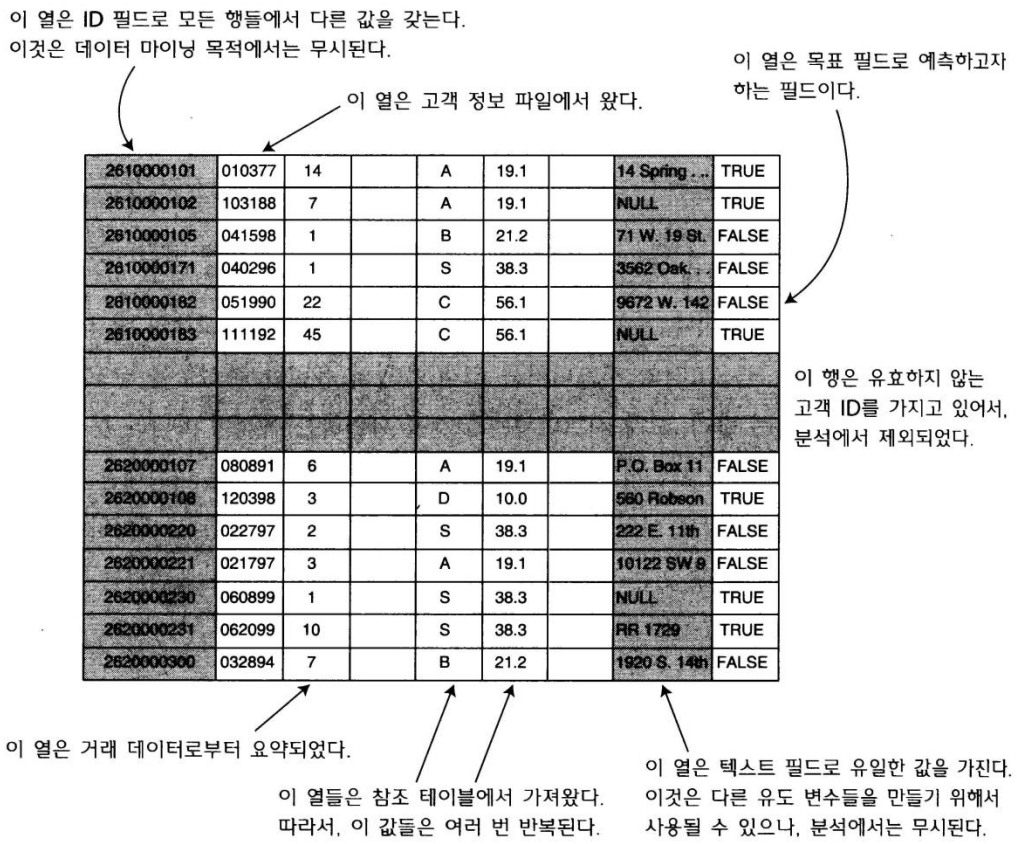
melt()
cast()

cut()



Data Munging with R (2)

데이터마이닝을 위한 데이터 – Customer Signature



2610000101	010377	14		A	19.1		14 Spring . ..	TRUE
2610000102	103188	7		A	19.1		NULL	TRUE
2610000105	041598	1		B	21.2		71 W. 19 St.	FALSE
2610000171	040296	1		S	38.3		3562 Oak. . .	FALSE
2610000182	051990	22		C	56.1		9672 W. 142	FALSE
2610000183	111192	45		C	56.1		NULL	TRUE
2620000107	080891	6		A	19.1		P.O. Box 11	FALSE
2620000108	120398	3		D	10.0		560 Robson	TRUE
2620000220	022797	2		S	38.3		222 E. 11th	FALSE
2620000221	021797	3		A	19.1		10122 SW 8	FALSE
2620000230	060899	1		S	38.3		NULL	TRUE
2620000231	062099	10		S	38.3		RR 1729	TRUE
2620000300	032894	7		B	21.2		1920 S. 14th	FALSE

- 모든 데이터가 하나의 테이블에 존재해야 한다.
- 각 행은 기업과 관련 있는 한 개체 (Ex: 고객)에 대응해야 한다.
- 하나의 값을 갖는 필드는 무시되어야 한다.
- 대부분이 한 값을 갖는 필드도 가급적 무시되어야 한다.
- 각 행마다 다른 값들을 가지는 필드는 무시되어야 한다.
- 예측 모델링을 위해서 목표 필드와 지나치게 높은 상관관계를 갖는 필드는 제거되어야 한다.

파생변수를 생성하는 일반적인 방법

- 한 값으로부터 특징들을 추출한다.
  - 날짜로부터 요일을 계산
  - 신용카드번호로부터 신용카드 발급자를 추출
- 한 레코드 내의 값들을 결합한다.
  - 멤버십 가입일과 첫 구매일로부터 경과를 계산
- 다른 테이블의 부가적인 정보를 참조한다.
  - 우편번호에 따른 인구와 평균가계수입
  - 상품코드에 대한 계층 구조
- 다수 필드 내에 시간 종속적인 데이터를 pivoting한다.
  - 월마다 한 행씩 저장되는 과금 데이터를 각각의 월에 대응하는 필드로 변환
- 거래 레코드들을 요약한다.
  - 년간 총 구매액
- Customer Signature 필드들을 요약한다.
  - 값의 표준화 및 서열화

H백화점 데이터 (1/2)

종류	화일명	형식	필드 수	레코드 수
고객정보	HDS_Customers.tab	TSV	36	49,995
구매정보	HDS_Transactions_MG.tab	TSV	18	1,726,430
카드정보	HDS_Cards.tab	TSV	2	290
직업정보	HDS_Jobs.tab	TSV	3	267
우편번호	mic_engzipcode_DB20050215.xlsx	엑셀	16	48,072

- ✓ 데이터 수집 기간: 2000.05.01 ~ 2001.04.29 (4개 지점)
- ✓ 취급 상품 수: 11,031개 (1,906개 브랜드, 309개 코너)
- ✓ 1인당 년간 구매액: 약 329만원 (최대 1억1479만원)

H백화점 데이터 (2/2)

구매정보	
strcd	지점코드
custid	고객 아이디
goodcd	상품코드
brand	브랜드코드
team	팀코드
part	파트코드
pc	상품군코드
corner	코너코드
tot_amt	구매액
dis_amt	할인액
net_amt	실구매액
inst_mon	할부기간
sales_time	판매시간

고객정보		
custid	고객 아이디	
sex	성별코드	* 0 : 무효값 1 : 남성 2 : 여성
birth	생일	
birth_flg	생일구분코드	* -1 : 기타 1 : 양력 2 : 음력
card_cd	카드 코드	Card 테이블 참조
mrg_date	결혼기념일	
mrg_flg	결혼여부코드	* 0 : 기타 1 : 기혼 2 : 미혼 7 : 기타
h_type1	주거형태코드	* A : 아파트 B : 빌딩 H : 병원 N : 단독주택
		* V : 빌라 X : 기타 Z : 기타(분류안됨)
h_type2	주거현황코드	* 0 : 기타 1 : 본인소유 2 : 배우자소유 3 : 부모소유 4 : 전세
		* 5 : 기타 6 : 기타 7 : 기타 8 : 기타
hobby	취미코드	* 0001 : 등산 0002 : 스포츠 0003 : 여행 0004 : 낚시 0005 : 관람
		* 0006 : 컴퓨터 0007 : 요리 0008 : 서예 0009 : 미술 0010 : 공예
		* 0011 : 음악 0012 : 꽃꽂이 0013 : 수집 0014 : 독서 0015 : 작문
		* 0016 : 바둑 0017 : 기타
job_stype	직업코드	Job 테이블 참조
ent_date	신규가입(카드)일	
mail_flg	청구지구분	* 1 : 자택 2 : 직장
card_str	카드 발급점	
mail_zip1	청구지 우편번호 1	
mail_zip2	청구지 우편번호 2	
home_zip1	자택 우편번호 1	
home_zip2	자택 우편번호 2	
work_zip1	직장 우편번호 1	
work_zip2	직장 우편번호 2	
cus_stype	고객소유형	* 1 : 초우량 1등급 2 : 초우량 2등급 3 : 초우량 3등급 4 : 초우량4등급
		* 5 : 우량 1등급 6 : 우량 2등급 7 : 우량 3등급 8 : 우량 4등급
		* 9 : 고정 1등급 10 : 고정 2등급 11 : 일반 1등급 12 : 일반 2등급
m_srt1	주구매 지점코드	
m_time1	주구매 시간대	* -1 : 알수없음 1~48 : 0시부터 24시 까지 30분간격
autopat	자동이체여부코드	* 0 : 일반청구 1 : 자동이체 2 : 급여이체 4 : 기타

H백화점 데이터로부터 파생 가능한 변수

- 환불행태
- 구매상품 다양성
- 내점일수 & 내점 당 구매건수
- 구매주기
- 요일별 구매패턴
- 연령대
- 기간별 구매 금액 & 횟수
- 구매추세 패턴
- 가격 선호도
- 시즌 선호도
- 상품별 구매 금액/횟수/여부
- 상품별 구매순서
- 주 구매상품
- 휴면/이탈 여부

파생변수 – 환불행태

- H백화점 고객의 환불행태(금액, 건수)에 대한 변수 생성


```
library(dplyr)
library(lubridate)
library(ggplot2)

cs <- read.delim("HDS_Customers.tab", stringsAsFactors=F)
tr <- read.delim("HDS_Transactions_MG.tab", stringsAsFactors=F)

cs.v1 <- tr %>%
  filter(net_amt < 0) %>%
  group_by(custid) %>%
  summarize(rf_amt=sum(net_amt), rf_cnt=n())
```

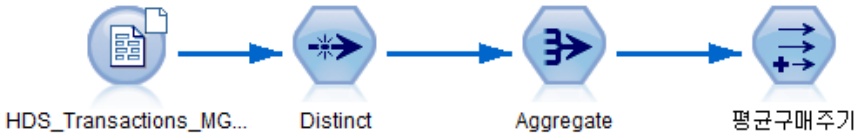
파생변수 – 구매상품 다양성

- H백화점 고객의 구매상품 다양성에 대한 변수 생성


```
cs.v2 <- tr %>%
  distinct(custid, brd_nm) %>%
  group_by(custid) %>%
  summarize(buy_brd=n())
```

파생변수 – 내점일수 & 구매주기

- H백화점 고객의 내점일수와 평균구매주기(Average Purchasing Interval)를 계산

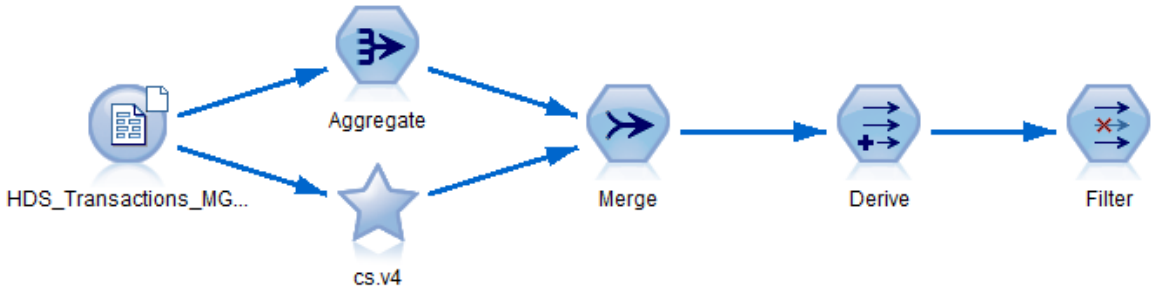



```
# lubridate 패키지 사용법 => http://blog.naver.com/dfdf4912/220623488198
start_date <- ymd(ymd_hms(min(tr$sales_date)))
end_date <- ymd(ymd_hms(max(tr$sales_date)))

cs.v3 <- tr %>%
  distinct(custid, sales_date) %>%
  group_by(custid) %>%
  summarise(visits=n()) %>%
  mutate(API = as.integer(end_date - start_date) / visits)
```

파생변수 – 내점 당 구매건수

- 내점 당 구매건수(Number of Purchases Per Visit) 도출




```
tmp <- tr %>%
  group_by(custid) %>%
  summarise(n=n())

cs.v4 <- inner_join(cs.v3, tmp) %>%
  mutate(NPPV = n / visits) %>%
  select(custid, NPPV)
```

파생변수 – 요일별 구매패턴

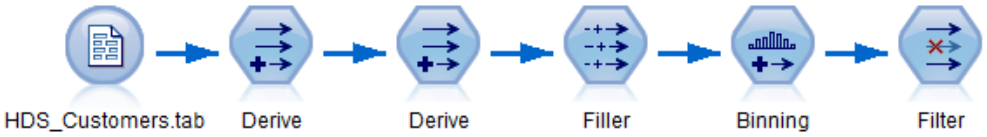
- H백화점 고객의 주중•주말 구매패턴에 대한 변수 생성


```
cs.v5 <- tr %>%
  mutate(wk_amt = ifelse(wday(sales_date) %in% 2:6, net_amt, 0),
         we_amt = ifelse(wday(sales_date) %in% c(1,7), net_amt, 0)) %>%
  group_by(custid) %>%
  summarize_each(funs(sum), wk_amt, we_amt) %>%
  mutate(wk_pat = ifelse(wk_amt >= we_amt * 1.5, "주중형",
                  ifelse(we_amt >= wk_amt * 1.5, "주말형", "유형없음")))

ggplot(cs.v5, aes(wk_pat)) + geom_bar(aes(fill=wk_pat))
```

파생변수 – 나이와 연령대

- 고객의 생일로부터 특정시점의 나이와 연령대를 계산



10대(이하),20대,30대,40대,50대,
60대,70대(이상)으로 구간을 나눔

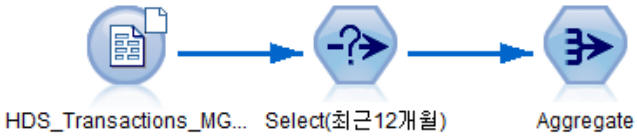
주의) 이상치 처리가 필요함.

```
cs.v6 <- cs %>%
  mutate(age=year('2001-05-01') - year(ymd_hms(birth))) %>%
  mutate(age=ifelse(age < 10 | age > 100, NA, age)) %>%
  mutate(age=ifelse(is.na(age),round(mean(age,na.rm=T)),age)) %>%
  mutate(agegrp=cut(age, c(0,19,29,39,49,59,69,100), labels=F)*10) %>%
  select(custid, age, agegrp)
```

```
cs.v6 <- cs %>% # 위와 동일한 결과를 얻는 다른 표현
  mutate(age=year('2001-05-01') - year(ymd_hms(birth)),
         age=ifelse(age < 10 | age > 100, NA, age),
         age=ifelse(is.na(age),round(mean(age,na.rm=T)),age),
         agegrp=cut(age, c(0,19,29,39,49,59,69,100), labels=F)*10) %>%
  select(custid, age, agegrp)
```

파생변수 – 기간별 구매 금액 & 횟수 (1/2)

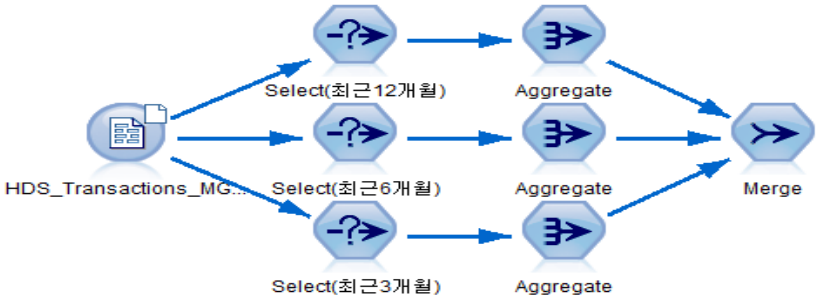
- H백화점 고객의 최근 12개월 구매금액 및 구매횟수에 대한 변수 생성




```
end_date <- ymd(ymd_hms(max(tr$sales_date)))
start_date <- ymd('20010501') - months(12)
cs.v7.12 <- tr %>%
  filter(start_date<=sales_date & sales_date<=end_date) %>%
  group_by(custid) %>%
  summarize(amt12=sum(net_amt), nop12=n())
```

파생변수 – 기간별 구매 금액 & 횟수 (2/2)

- 최근 3개월, 6개월, 12개월 구매 금액 및 횟수 계산 및 병합




```
start_date <- ymd('20010501') - months(6)
cs.v7.06 <- tr %>%
  filter(start_date<=sales_date & sales_date<=end_date) %>%
  group_by(custid) %>%
  summarize(amt6=sum(net_amt), nop6=n())

start_date <- ymd('20010501') - months(3)
cs.v7.03 <- tr %>%
  filter(start_date<=sales_date & sales_date<=end_date) %>%
  group_by(custid) %>%
  summarize(amt3=sum(net_amt), nop3=n())

cs.v7 <- left_join(cs.v7.12, cs.v7.06) %>% left_join(cs.v7.03)

# amt6, nop6, amt3, nop3가 NA이면 0으로 대체하는 코드 삽입
```

Customer Signature 만들기

```
custsig <- cs %>%
    left_join(cs.v1) %>%
    left_join(cs.v2) %>%
    left_join(cs.v3) %>%
    left_join(cs.v4) %>%
    left_join(cs.v5) %>%
    left_join(cs.v6) %>%
    left_join(cs.v7)
```

```
# 파생변수가 NA이면 적절한 값으로 대체하는 코드를 작성해야 함.
```

팀프로젝트 #1 – 10월8일 제출 및 발표

H백화점 데이터를 이용하여 아래와 같은 파생변수를 생성하고,
수업시간에 다룬 파생변수와 함께 고객데이터와 병합하여
최종 Customer Signature를 만드시오.

- 가격 선호도 변수
- 시즌 선호도 변수
- 구매추세 패턴
- 상품별 구매 금액/횟수/여부 변수
- 상품별 구매순서 변수
- 주 구매상품 변수
- 휴면/이탈 가망 변수
  - Ex) If 평균구매주기 < 최종구매경과(현재 – 마지막 구매시점) Then “이탈”