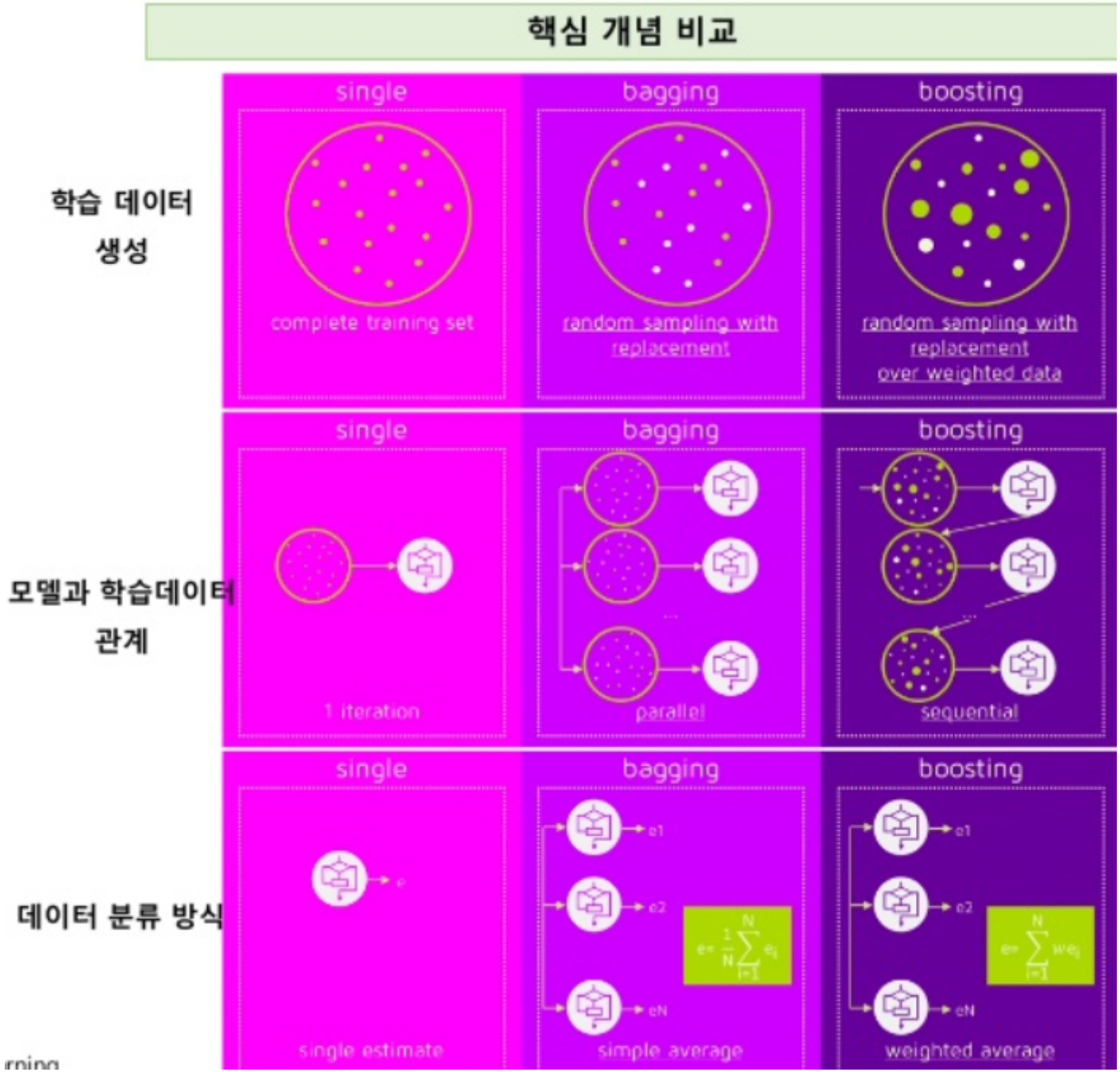
Xgboost의 소개

Bagging : 랜덤 포레스트

Boosting : Xgboosting 기법

특징 비교

비교	Bagging	Boosting
특징	병렬 앙상블 모델 (각 모델은 서로 독립적)	연속 앙상블 (이전 모델의 오류를 고려)
목적	Variance 감소	Bias 감소
적합한 상황	복잡한 모델 (High variance, Low bias)	Low variance, High bias 모델
대표 알고리즘	Random Forest	Gradient Boosting, AdaBoost
Sampling	Random Sampling	Random Sampling with weight on error


핵심 개념 비교
single
bagging
boosting
학습 데이터 생성
complete training set
random sampling with replacement
random sampling with replacement over weighted data
모델과 학습데이터 관계
1 iteration
parallel
sequential
데이터 분류 방식
single estimate
simple average
weighted average

boosting algorithm 의 원리

- 성능이 약한 학습기를 여러개 사용해서, 단 각각의 약한 학습기는 다른 내부 구조와 결과를 산출해야 한다. 하나의 좋은 학습 결과를 산출한다. 약한 학습기는 하나 성능이 좋지 않지만, 여러 학습기의 결과를 합치면 꽤 성능이 좋은 학습기가 만들어진다.

부스팅의 설명

- Bagging의 변형으로, 모델이 잘 예측하지 못하는 부분을 개선을 시키려고 함.
- Boosting은 이전 모델들에 의해 나온 오차들을 교정하기 위해 새로운 모델들을 추가하는 하나의 앙상블 테크닉이다.
- 모델들은 더 이상 향상이 없을 때까지 순차저으로 더해진다.

```
Y = M(x) + error                  # M :학습기, Y : 예측 확률
error를 상세히 분류 (G + error2)  # error > error2
Y = M(x) + G(x) + error2           # M :학습기, Y : 예측 확률
error2를 상세히 분류 (H + error3)  # error2 > error3
Y = M(x) + G(x) + H(x) + error3    # M :학습기, Y : 예측 확률
```

- Boosting은 이전 모델들이 예측하지 못한 Error 데이터에 가중치를 부여하여, 다음 모델들이 더 잘 예측하도록 한다.

하나의 학습 모델(M)을 사용했을 때보다 성능은 높다. 단 M, G, H는 분류기의 성능이 다르다. 하지만 같은 가중치(1,1,1)을 가지고 있다.

모델 앞에 비중(weights)을 두고, 머신러닝을 이용하여 최적의 비중을 찾는다.

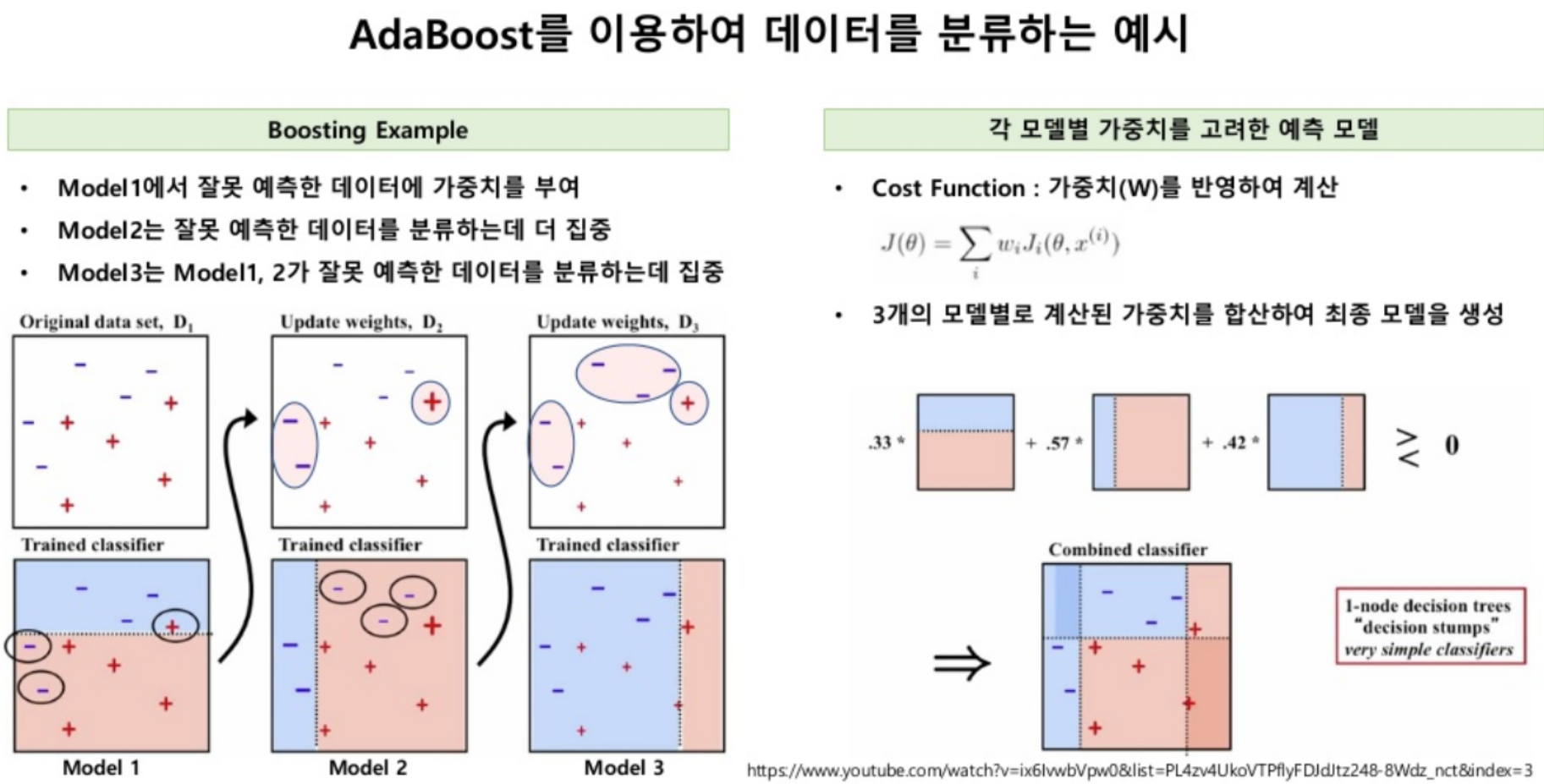
이것은 더 좋은 성능을 갖게 될 것이다.

```
Y = a * M(x) + b * G(x) + c * H(x) + error4
```

알고리즘	특징	비고

알고리즘	특징	비고
AdaBoost	다수결을 통한 정답 분류 및 오답에 가중치 부여	-
GBM	Loss Function의 gradient를 통해 오답에 가중치 부여	-
Xgboost	GBM대비 성능 향상, 시스템 자원 효율적 활용	-
Light GBM	Xgboost 대비 성능향상 및 자원소모 최소화, 대용량 학습 가능	-

AdaBoost



Gradient boosting

- AdaBoost 기본 개념과 동일하다.
- 가중치(D)를 계산하는 방식에서 Gradient Descent를 이용하여 최적의 파라미터를 찾는다.
- 이전 모델들의 오차 또는 잔차들을 예측하는 새로운 모델을 생성한다.
- 그 후, 최종 예측을 하기 위해 이전 모델과 새로운 모델을 합친다.
- 새로운 모델들을 더할 때, 오차를 최소화시키기 위해 gradient descent 알고리즘을 사용한다. 따라서 gradient boosting이라 한다.

Gradient Boosting 고려사항

- Boosting은 과적합 가능성이 높아, 적절한 시점에 멈춰야 한다.

기본 설정

- subsampling : 하나의 트리를 학습할 때 모든 학습 데이터를 사용하는 것이 아닌, 학습 데이터 중에서 랜덤하게 추출한 일부 데이터만을 사용.
- min child weight : 트리의 제일 말단 노드가 가져야할 수(observations)의 최소값을 제한한다. 이 수보다 적은 수의 말단 노드가 갖는 분할은 무시된다. 특이값 때문에 발생하는 변동폭을 줄여 성능 향상시킴.

XGBoost

- 지도 학습 문제을 위해 사용된다.
- XGBoost는 eXtreme Gradient Boosting 의 약자
- Tianqi Chen에 의해 만들어졌다. 많은 개발자의 도움으로 발전되었다.
- Kaggle competition 에서 가장 잘 나가는 알고리즘 중의 하나이다.

XGBoost 장점

- Gradient Boost의 학습 성능은 좋지만, 수행시간/연산시간 측면에서 부족한 부분이 많았다.이를 엄청나게 개선했다.
- Decision Tree를 구성할 때 병렬 처리 기법을 사용한다. (수행 시간 비약적인 향상)
- 휼륭한 그래디언트 부스팅 라이브러리

- Gradient Boost는 경사 하강법을 사용해서 성능을 개선한 Boosting 기법이다.
- 병렬 처리를 사용한다. 학습과 분류가 빠르다.
- 욕심쟁이(Greedy-algorithm)를 사용한 자동 가지치기가 가능. 따라서 과적합(Overfitting)잘 일어나지 않는다.
- xgboost 분류기 아래에 다른 알고리즘을 붙여 앙상블 학습이 가능하다.

모델 특징

- XGB 모형은 Regularization 기능과 함께 R과 scikit-learn의 실행을 지원
- 학습률(Learning rate)를 포함한 gradient boosting machine인 Gradient Boosting
- L1과 L2 regularization이 있는 Regularized Gradient Boosting

In [ ]:
```
1
```

In [ ]:
```
1
```

시스템 특징

- 학습하는 동안 모든 CPU 코어를 사용하여 나무를 생성 (병렬화)
- 메모리에 맞지 않는 엄청 큰 데이터 셋을 위한 Out-of-Core Computing
- 하드웨어의 최상을 사용을 위해, 데이터 구조와 알고리즘에 대한 Cache Optimization

알고리즘 특징

- 결측치를 자동으로 처리하는 Sparse Aware
- 나무 병렬화를 지원하는 Block Structure
- 새로운 데이터에 대해 이전 학습한 모델의 성능을 더 향상시키기 위한 Continued Training

성능

- R, Python, Spark, H2O 같은 것보다 더 빠르다
- 참고 : http://datascience.la/benchmarking-random-forest-implementations/ (http://datascience.la/benchmarking-random-forest-implementations/)

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

ref

- https://gentlej90.tistory.com/87 (https://gentlej90.tistory.com/87)
- Tianqi Chen이 만들면서 생긴 뒷이야기
- https://homes.cs.washington.edu/~tqchen/2016/03/10/story-and-lessons-behind-the-evolution-of-xgboost.html (https://homes.cs.washington.edu/~tqchen/2016/03/10/story-and-lessons-behind-the-evolution-of-xgboost.html)
- gradient algorithm youtube : https://www.youtube.com/watch?v=wPqtzj5VZus (https://www.youtube.com/watch?v=wPqtzj5VZus)
- Xgboost 예제 코드 : https://github.com/dmlc/xgboost/tree/master/demo (https://github.com/dmlc/xgboost/tree/master/demo)
- Xgboost 튜토리얼 가이드 : https://xgboost.readthedocs.io/en/latest/ (https://xgboost.readthedocs.io/en/latest/)
- boosting 기법 이해 : https://www.slideshare.net/freepsw/boosting-bagging-vs-boosting (https://www.slideshare.net/freepsw/boosting-bagging-vs-boosting)

In [ ]:
```
1
```

In [1]:
```
1 library(tidyverse)
2 library(lubridate)
```

```
Warning message:
"package 'tidyverse' was built under R version 3.4.4"-- Attaching packages --------------------------------------- tidyverse 1.2.1
--
√ ggplot2 3.1.0     √ purrr   0.2.5
√ tibble  1.4.2     √ dplyr   0.7.8
√ tidyr   0.8.2     √ stringr 1.3.1
√ readr   1.1.1     √ forcats 0.3.0
Warning message:
"package 'ggplot2' was built under R version 3.4.4"Warning message:
"package 'tibble' was built under R version 3.4.4"Warning message:
"package 'tidyr' was built under R version 3.4.4"Warning message:
"package 'readr' was built under R version 3.4.4"Warning message:
"package 'purrr' was built under R version 3.4.4"Warning message:
"package 'dplyr' was built under R version 3.4.4"Warning message:
"package 'stringr' was built under R version 3.4.4"Warning message:
"package 'forcats' was built under R version 3.4.4"-- Conflicts ------------------------------------------ tidyverse_conflicts() -
-
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
Warning message:
"package 'lubridate' was built under R version 3.4.4"
Attaching package: 'lubridate'

The following object is masked from 'package:base':

    date
```

데이터 불러오기

In [3]:
```
1 train <- read.csv("D:/dataset/Bike/biketrain.csv")
2 test <- read.csv("D:/dataset/Bike/biketest.csv")
```

In [4]:
```
glimpse(train)
glimpse(test)
tr_idx <- 1:nrow(train)
y <- log1p(train$count)

colnames(train)
colnames(test)
```

```
Observations: 10,886
Variables: 12
$ datetime   <fct> 2011-01-01 00:00:00, 2011-01-01 01:00:00, 2011-01-01 02:...
$ season     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ holiday    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
$ workingday <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
$ weather    <int> 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3,...
$ temp       <dbl> 9.84, 9.02, 9.02, 9.84, 9.84, 9.84, 9.02, 8.20, 9.84, 13...
$ atemp      <dbl> 14.395, 13.635, 13.635, 14.395, 14.395, 12.880, 13.635, ...
$ humidity   <int> 81, 80, 80, 75, 75, 75, 80, 86, 75, 76, 76, 81, 77, 72, ...
$ windspeed  <dbl> 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 6.0032, 0.0000, ...
$ casual     <int> 3, 8, 5, 3, 0, 0, 2, 1, 1, 8, 12, 26, 29, 47, 35, 40, 41...
$ registered <int> 13, 32, 27, 10, 1, 1, 0, 2, 7, 6, 24, 30, 55, 47, 71, 70...
$ count      <int> 16, 40, 32, 13, 1, 1, 2, 3, 8, 14, 36, 56, 84, 94, 106, ...
Observations: 6,493
Variables: 9
$ datetime   <fct> 2011-01-20 00:00:00, 2011-01-20 01:00:00, 2011-01-20 02:...
$ season     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ holiday    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
$ workingday <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ weather    <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2,...
$ temp       <dbl> 10.66, 10.66, 10.66, 10.66, 10.66, 9.84, 9.02, 9.02, 9.0...
$ atemp      <dbl> 11.365, 13.635, 13.635, 12.880, 12.880, 11.365, 10.605, ...
$ humidity   <int> 56, 56, 56, 56, 56, 60, 60, 55, 55, 52, 48, 45, 42, 45, ...
$ windspeed  <dbl> 26.0027, 0.0000, 0.0000, 11.0014, 11.0014, 15.0013, 15.0...
```

'datetime' 'season' 'holiday' 'workingday' 'weather' 'temp' 'atemp' 'humidity' 'windspeed' 'casual' 'registered' 'count'

'datetime' 'season' 'holiday' 'workingday' 'weather' 'temp' 'atemp' 'humidity' 'windspeed'

In [5]:
```
bike <- train %>% select(-casual, -registered, -count) %>%
  bind_rows(test) %>%
  mutate(year = year(datetime),
         month = month(datetime),
         yday = yday(datetime),
         mday = mday(datetime),
         wday = wday(datetime),
         qday = qday(datetime),
         week = week(datetime),
         hour = hour(datetime),
         am = am(datetime) %>% as.integer(),
         pm = pm(datetime) %>% as.integer()) %>% select(-datetime)
```

```
Warning message in bind_rows_(x, .id):
"Unequal factor levels: coercing to character"Warning message in bind_rows_(x, .id):
"binding character and factor vector, coercing into character vector"Warning message in bind_rows_(x, .id):
"binding character and factor vector, coercing into character vector"Warning message:
"package 'bindrcpp' was built under R version 3.4.4"
```

In [10]:
```
1 glimpse(bike)
```

```
Observations: 17,379
Variables: 19
$ season     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ holiday    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
$ workingday <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
$ weather    <int> 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3,...
$ temp       <dbl> 9.84, 9.02, 9.02, 9.84, 9.84, 9.84, 9.02, 8.20, 9.84, 13...
$ atemp      <dbl> 14.395, 13.635, 13.635, 14.395, 14.395, 12.880, 13.635, ...
$ humidity   <int> 81, 80, 80, 75, 75, 75, 80, 86, 75, 76, 76, 81, 77, 72, ...
$ windspeed  <dbl> 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 6.0032, 0.0000, ...
$ year       <dbl> 2011, 2011, 2011, 2011, 2011, 2011, 2011, 2011, 2011, 20...
$ month      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ yday       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ mday       <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ wday       <dbl> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,...
$ qday       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ week       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ hour       <int> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16...
$ am         <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0,...
$ pm         <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1,...
$ h_dvi      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3,...
```

In [7]:
```
bike <- bike %>% mutate(h_dvi = ifelse(hour <= 8, 1,
                                       ifelse(hour == 9, 2,
                                       ifelse(hour >= 10, 3, 4))))

tr <- bike[tr_idx, ]
set.seed(0)
tr_index <- sample(nrow(tr)*0.7)
val_index <- sample(nrow(tr[tr_index,])*0.9)

library(xgboost)
```

```
Attaching package: 'xgboost'

The following object is masked from 'package:dplyr':

    slice
```

In [17]:
```
print(nrow(tr[-tr_index,]))              # 30%
print(nrow(tr[tr_index,][val_index,]))   # 70% 중의 90%
print(nrow(tr[tr_index,][-val_index,]))  # 70% 중의 10%
```

```
[1] 3266
[1] 6858
[1] 762
```

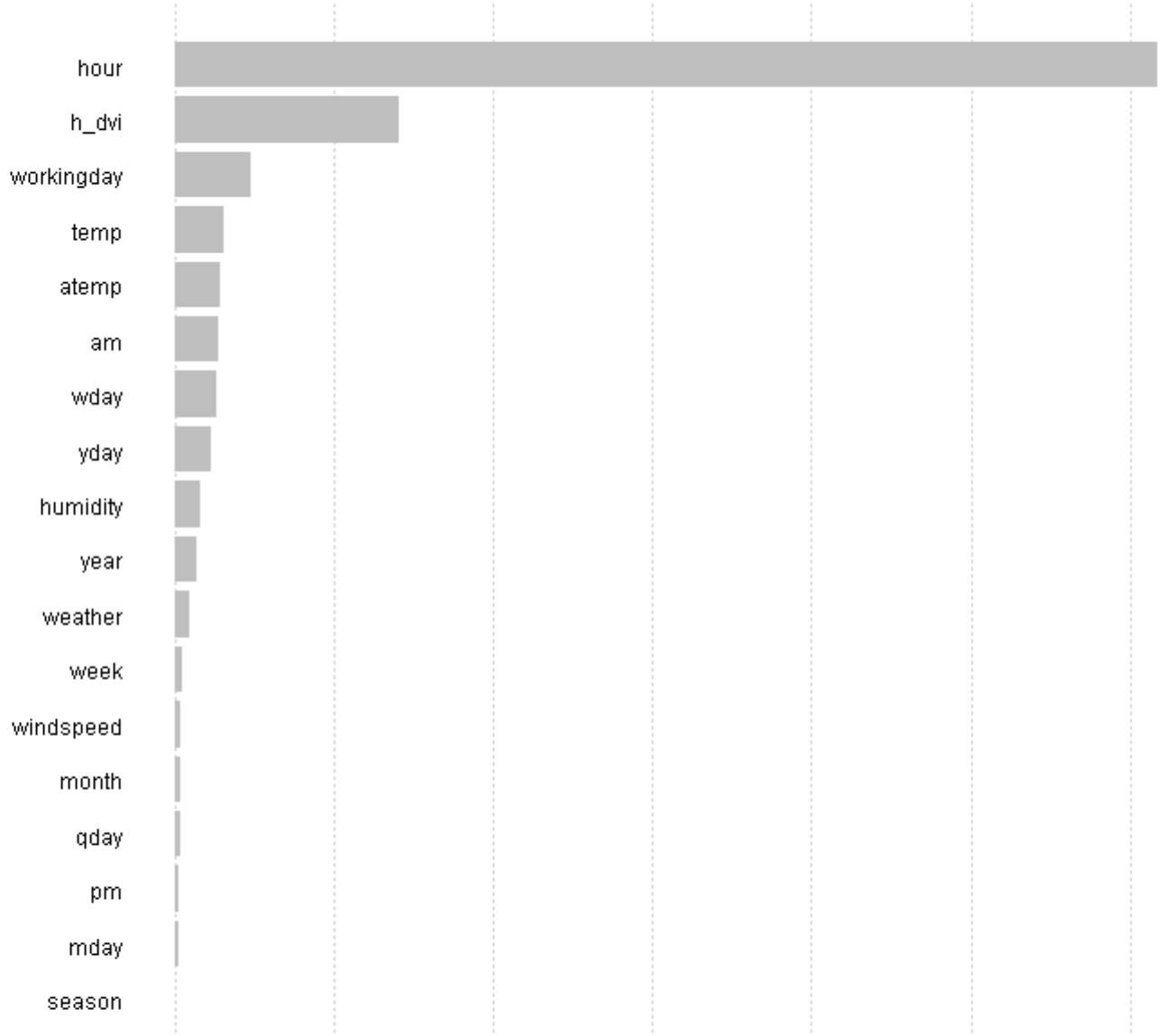
In [18]:
```
dtest <- xgb.DMatrix(data = data.matrix(tr[-tr_index,]))

dtrain <- xgb.DMatrix(data = data.matrix(tr[tr_index,][val_index,]),
                      label = y[tr_index][val_index])

dval <- xgb.DMatrix(data = data.matrix(tr[tr_index,][-val_index,]),
                    label = y[tr_index][-val_index])
```

In [19]:
```
p <- list(booster = "gbtree",          #
          eval_metric = "rmse",        # 평가 메트릭
          nthread = 8,                 # 사용할 thread 수
          eta = 0.05,
          max_depth = 18,              # 나무의 depth
          min_child_weight = 11,       #
          gamma = 0,
          subsample = 0.8,
          colsample_bytree = 0.7,
          alpha = 2.25,
          lambda = 0,
          nrounds = 5000)              # 총 학습
```

In [22]:
```
m_xgb <- xgb.train(p, dtrain, p$nrounds, list(val = dval),
                   print_every_n = 100, early_stopping_rounds = 200)
```

```
[1]     val-rmse:3.971663
Will train until val_rmse hasn't improved in 200 rounds.

[101]   val-rmse:0.313513
[201]   val-rmse:0.290043
[301]   val-rmse:0.287418
[401]   val-rmse:0.286348
[501]   val-rmse:0.285923
[601]   val-rmse:0.286075
Stopping. Best iteration:
[482]   val-rmse:0.285866
```

In [23]:
```
1 xgb.importance(feature_names = colnames(dtrain), m_xgb) %>% xgb.plot.importance()
```



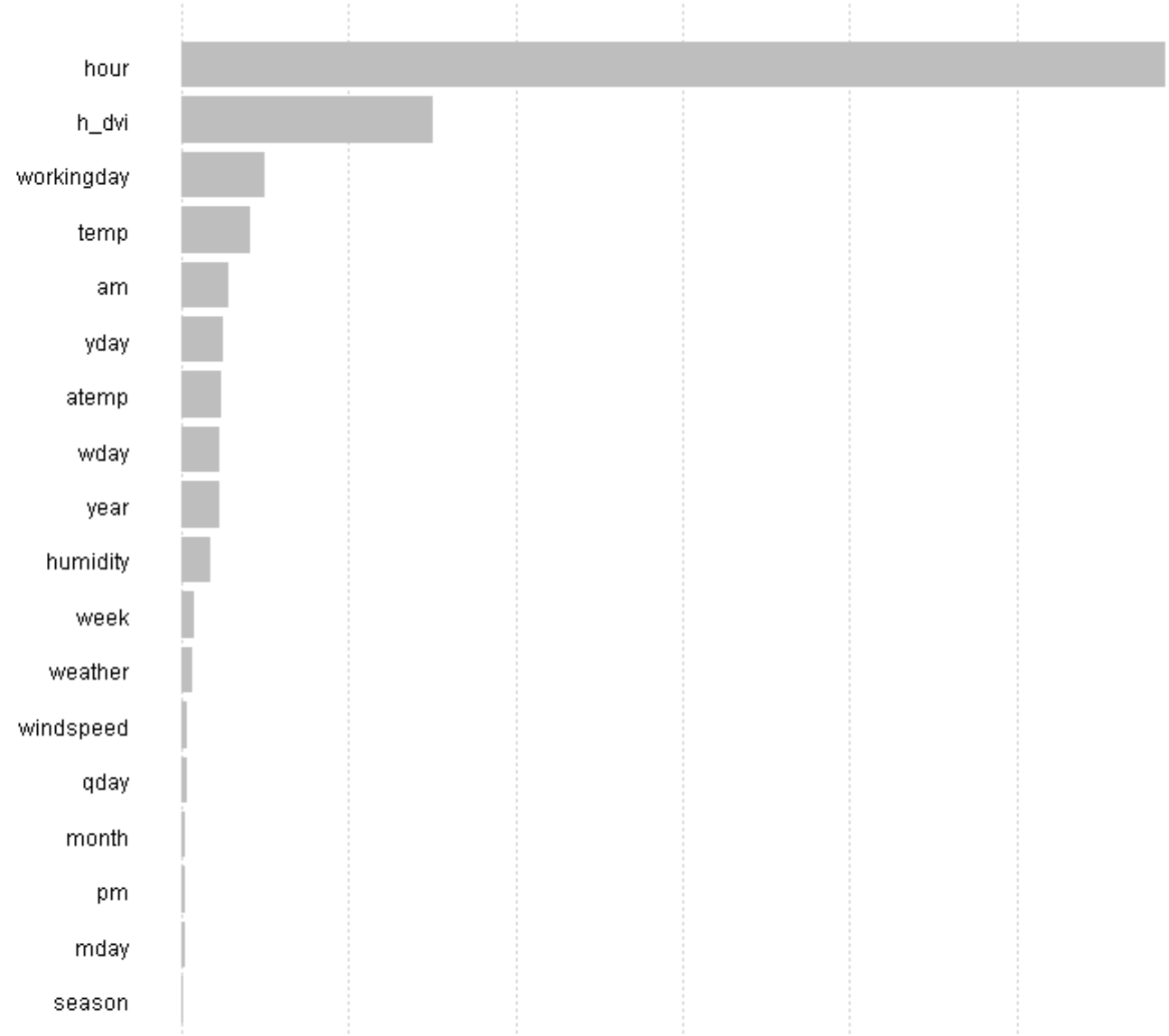
In [24]:

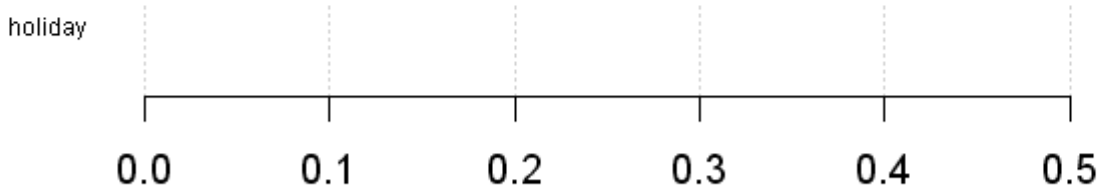
```
realtest <- xgb.DMatrix(data = data.matrix(bike[-tr_idx,]))
# preT <- expm1(predict(m_xgb, realtest))
# range(preT)
sub1 = read.csv("D:/dataset/Bike/sampleSubmission.csv")
# sub1$count <- preT
# write_csv(sub1, "../SubM/predT.csv")

```

In [25]:
```
index <- sample(nrow(tr) * 0.9)
train <- xgb.DMatrix(data = data.matrix(tr[index,]),
                     label = y[index])
val <- xgb.DMatrix(data = data.matrix(tr[-index,]),
                   label = y[-index])
f_xgb <- xgb.train(p, train, p$nrounds, list(val = val),
                   print_every_n = 50, early_stopping_rounds = 200)


```

```
[1]    val-rmse:4.380038
Will train until val_rmse hasn't improved in 200 rounds.

[51]   val-rmse:0.595021
[101]  val-rmse:0.325431
[151]  val-rmse:0.302403
[201]  val-rmse:0.297082
[251]  val-rmse:0.295871
[301]  val-rmse:0.295620
[351]  val-rmse:0.295436
[401]  val-rmse:0.295376
[451]  val-rmse:0.295545
[501]  val-rmse:0.295275
[551]  val-rmse:0.295170
[601]  val-rmse:0.294676
[651]  val-rmse:0.294598
[701]  val-rmse:0.294471
[751]  val-rmse:0.294315
[801]  val-rmse:0.294224
[851]  val-rmse:0.294318
[901]  val-rmse:0.294538
[951]  val-rmse:0.294486
Stopping. Best iteration:
[784]  val-rmse:0.294100
```

In [26]:
```
1 xgb.importance(feature_names = colnames(dtrain), f_xgb) %>% xgb.plot.importance(top_n = 35)
```





In [27]:
```
pred <- expm1(predict(f_xgb, realtest))
sub1$count <- pred
write_csv(sub1, paste0("D:/dataset/Bike/xgb_Bike_0131_", round(f_xgb$best_score, 5), ".csv"))
```

In [ ]:
```

```

In [ ]:
```

```