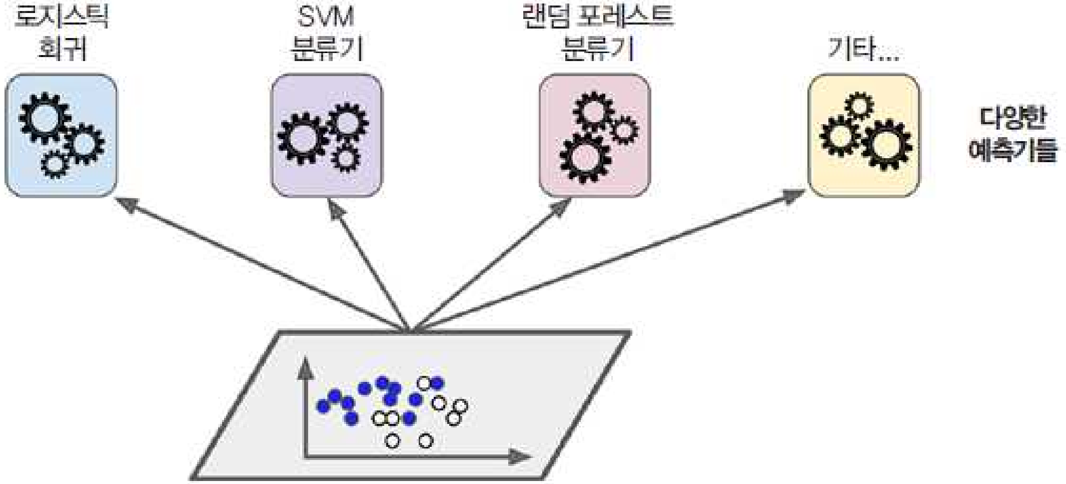
Ensemble Learning

Ensemble Learning

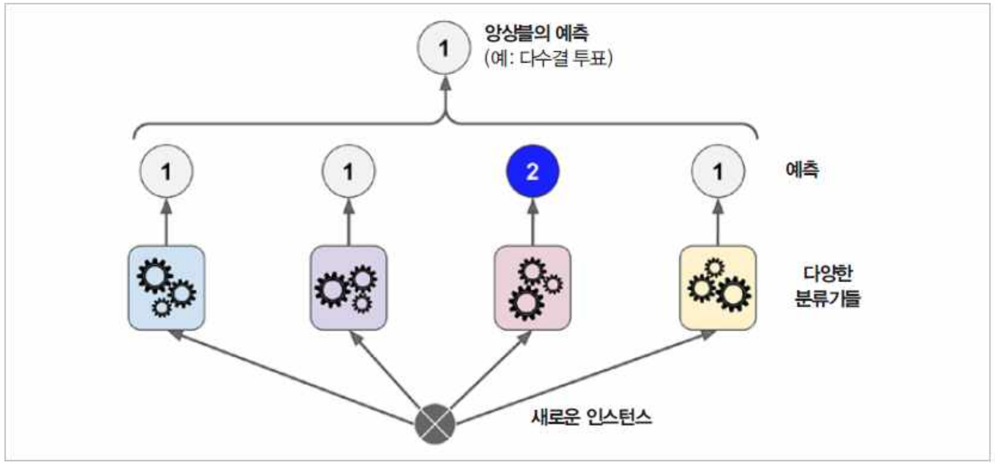
- Ensemble method
  - 여러 개의 분류기 (예: SVM, kNN, Decision tree….) 결과를 수집하여 최종 분류하는 방법



Ensemble Learning

- Hard voting (직접투표)
  - 각 classifier 의 분류 class 를 대상으로 다수결 투표
  - 각 분류기가 상호 독립적일 때 성능 우수
    - 다른 학습 알고리즘
    - 다른 학습 데이터

(예) 정확도
Classifier 1: 0.864, Classifier 2: 0.894, Classifier 3: 0.888
=> Voting Classifier: 0.904

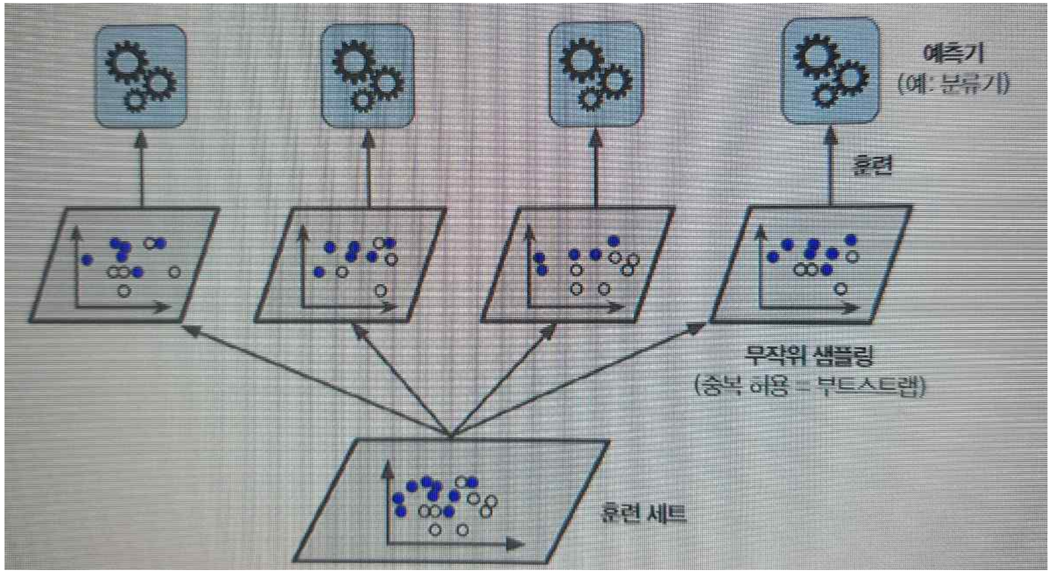


(cf) Soft voting (간접 투표): 각 classifier 의 class 별 확률 값 평균치를 구하여 결정

Bagging

- Ensemble method 의 training set
  - 각 classifier 별로 다른 학습 데이터 (training set) 구성

1) Bagging (Bootstrap aggregating), *Breiman 1996*
   - 중복을 허용하여 샘플링
2) Pasting
   - 중복을 허용하지 않고 샘플링



Bagging

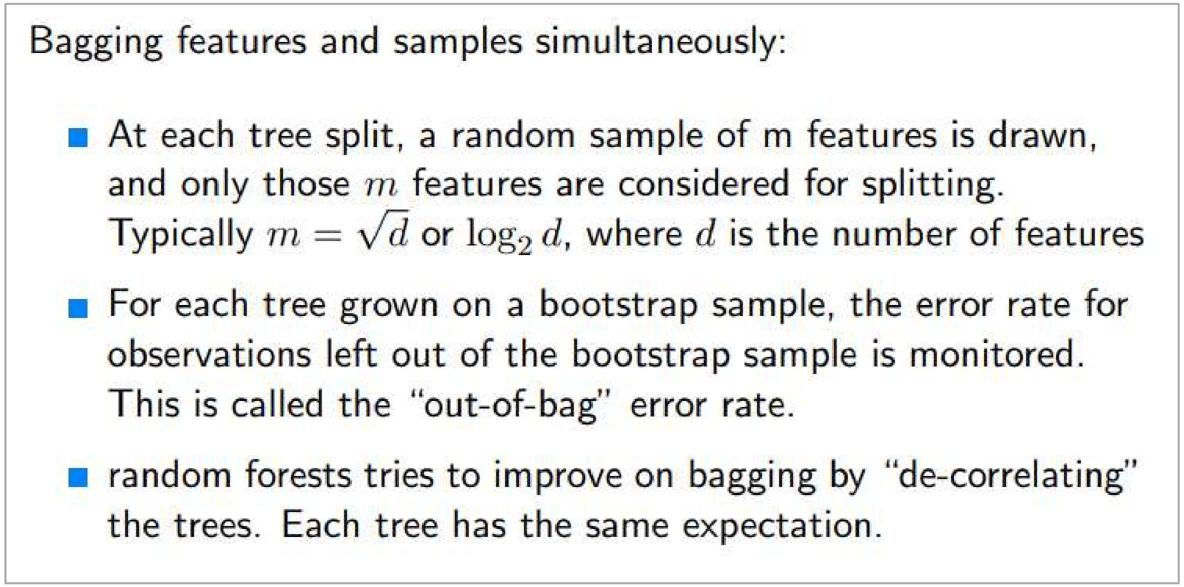
- OOB (Out-of-Bag) sample
  - Bagging 시 전체 샘플 중 한번도 선택되지 않는 훈련 샘플
  - Validation 데이터로 사용 => OOB 평가

- Random patches vs. Random subspaces
  - Training set 에 대하여 data 또는 feature 에 대한 sampling 가능

  1) Random patches method
     - 모든 feature 를 사용하고 data 를 sampling 하는 방식
  2) Random subspaces
     - 모든 data 를 사용하고 feature 를 sampling 하는 방식

Random Forest

- Random Forest
  - Breiman, 1999
  - Bagging 을 적용한 decision tree 의 ensemble 방법
    - Random patch & subspace method 동시 적용

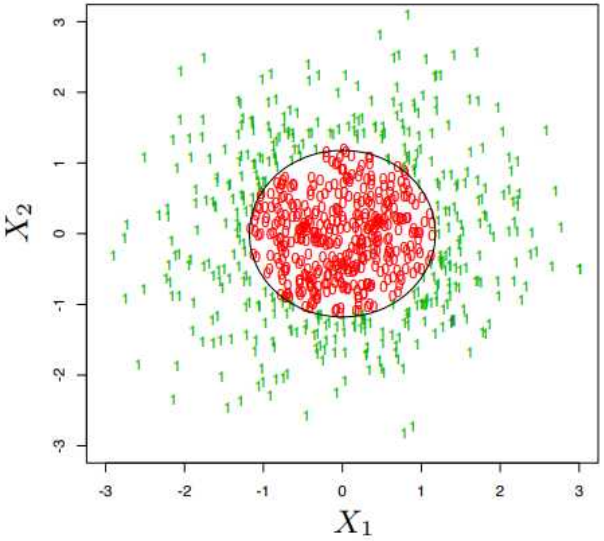


Bagging features and samples simultaneously:

- At each tree split, a random sample of m features is drawn, and only those m features are considered for splitting. Typically $m = \sqrt{d}$ or $\log_2 d$, where d is the number of features
- For each tree grown on a bootstrap sample, the error rate for observations left out of the bootstrap sample is monitored. This is called the "out-of-bag" error rate.
- random forests tries to improve on bagging by "de-correlating" the trees. Each tree has the same expectation.

Random Forest

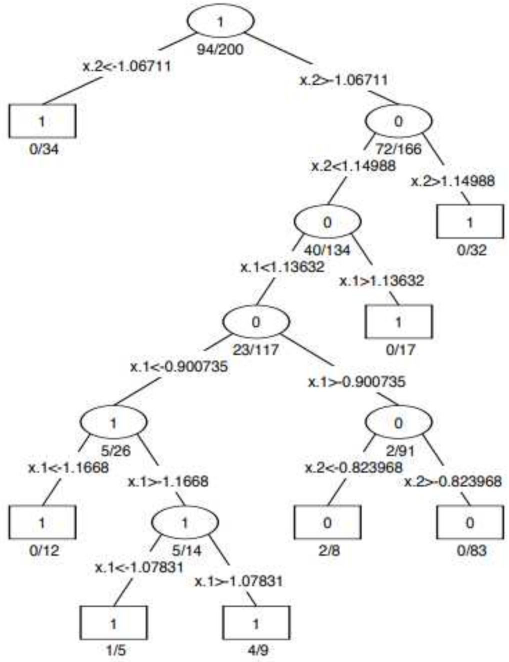
- Example-problem

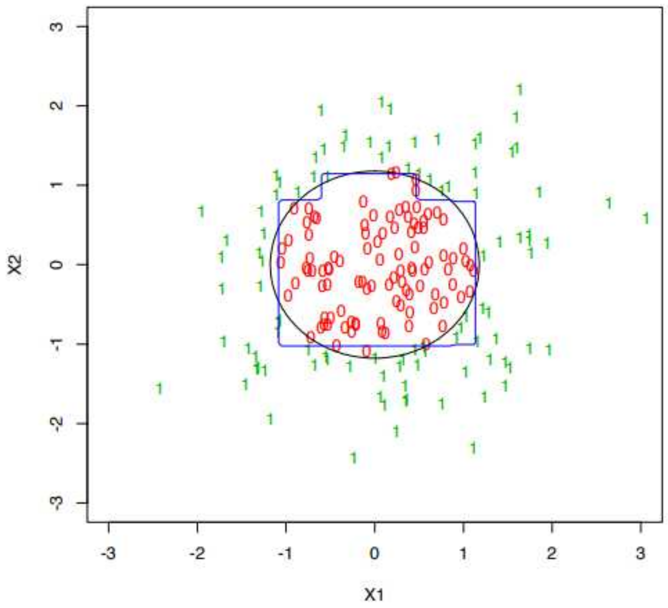


- Nonlinear separable data.
- Optimal decision boundary: $X_1^2 + X_2^2 = 1$.

Random Forest

- Example – decision tree

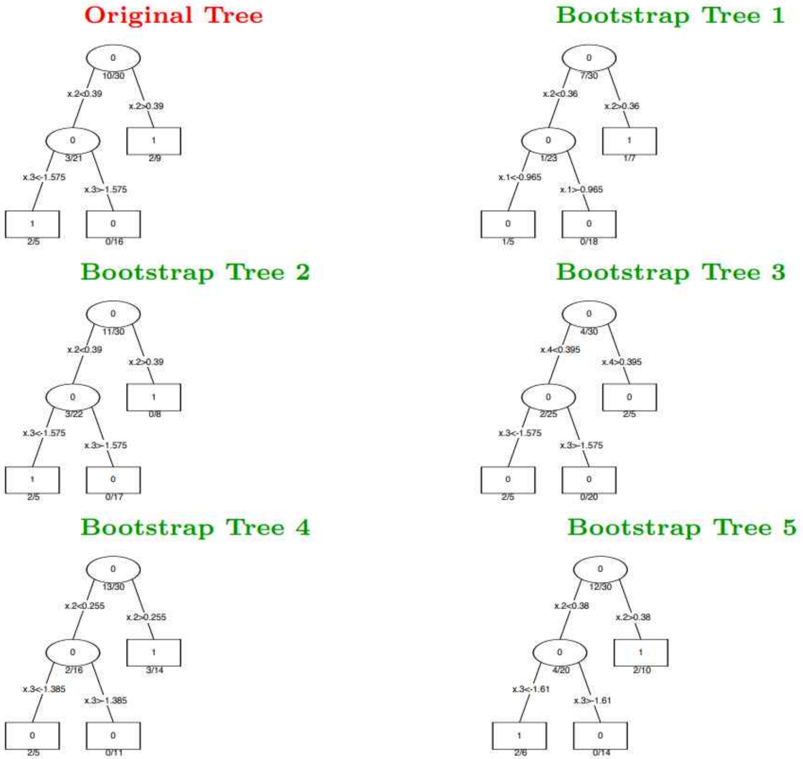




- Sample size: 200
- 7 branching nodes; 6 layers.
- Classification error: 7.3% when $d = 2$; $> 30\%$ when $d = 10$.

Random Forest

- Example – random forest



Original Tree

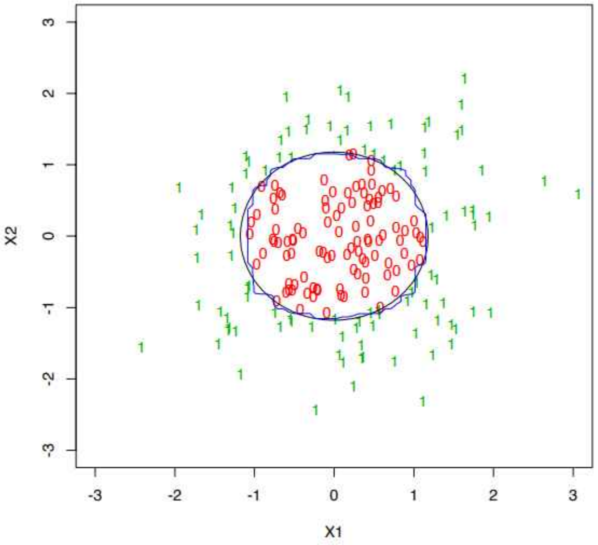
Bootstrap Tree 1

Bootstrap Tree 2

Bootstrap Tree 3

Bootstrap Tree 4

Bootstrap Tree 5

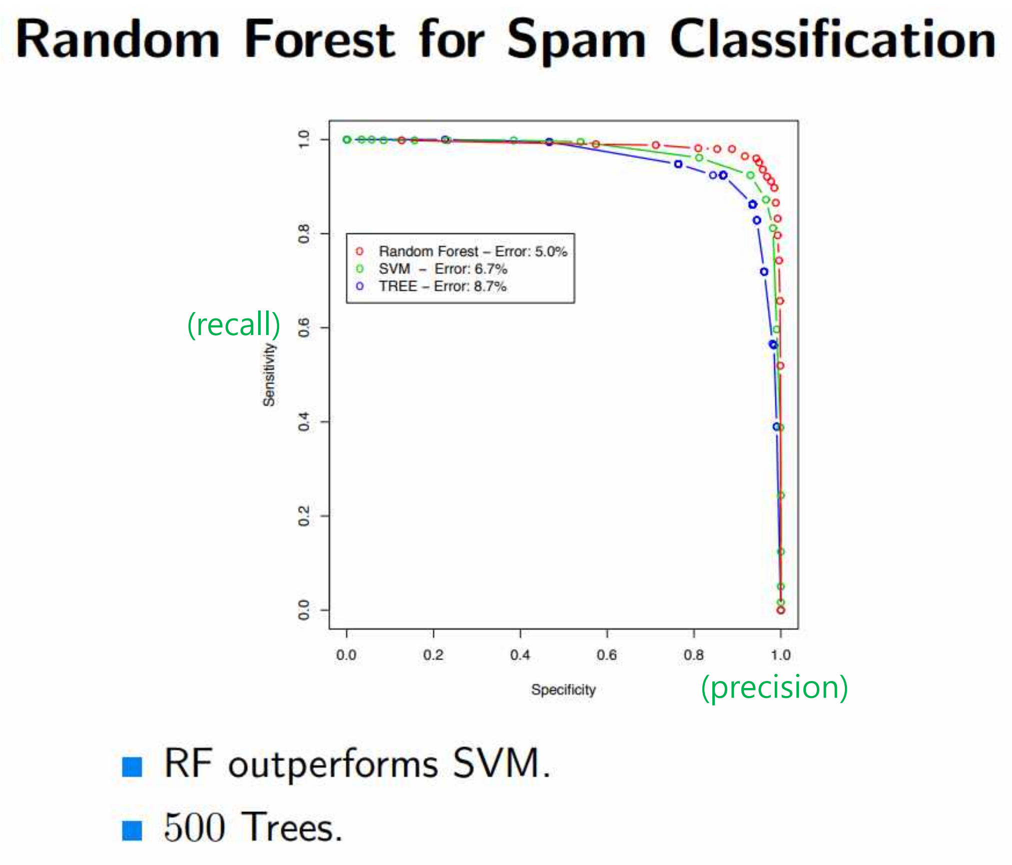


- ■ 2 branching nodes; 2 layers.
- ■ 5 dependent trees.

- ■ A smoother decision boundary.
- ■ Classification error: 3.2% (Single deeper tree 7.3%).

Random Forest

- Performance



Random Forest for Spam Classification

(recall)

(precision)

- RF outperforms SVM.
- 500 Trees.

Random Forest

- Feature importance
  - Feature 또는 variable 의 상대적 중요도 측정
  - 각 노드에서 Impurity 를 얼마나 감소시키는 지 확인가능

(예) IRIS 분류문제

: 꽃잎 길이 (44%), 꽃잎너비 (42%), 꽃받침길이(11%), 꽃받침너비 (2%)

(예) MNIST

: feature = pixel

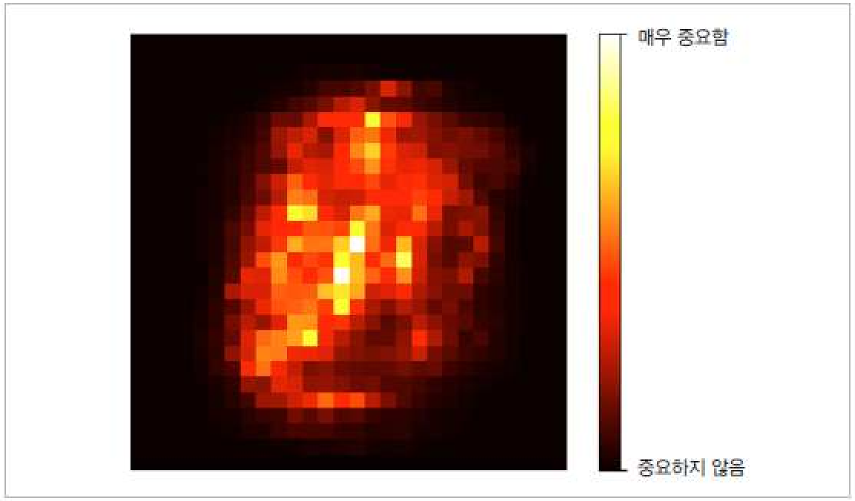
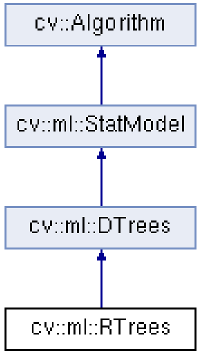
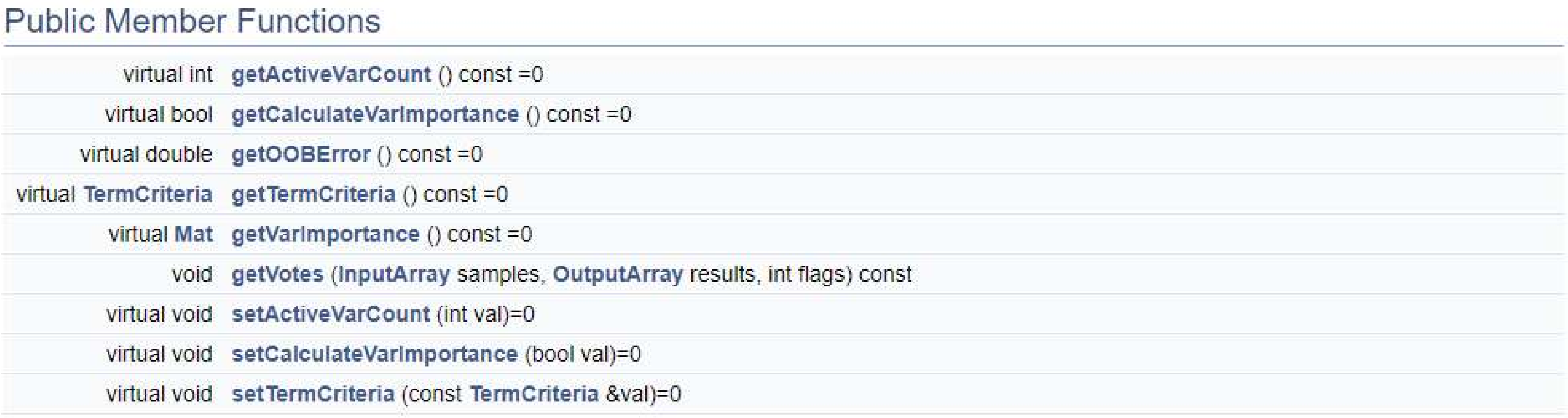


그림 7-6 (랜덤 포레스트 분류기에서 얻은) MNIST 픽셀 중요도

OpenCV

- cv::ml::Rtrees Class





Public Member Functions

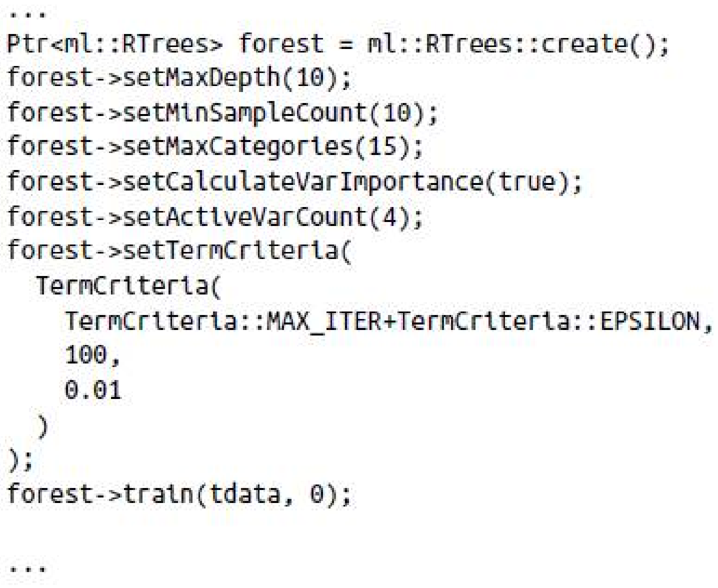
virtual int	getActiveVarCount () const =0
virtual bool	getCalculateVarImportance () const =0
virtual double	getOOBError () const =0
virtual TermCriteria	getTermCriteria () const =0
virtual Mat	getVarImportance () const =0
void	getVotes (InputArray samples, OutputArray results, int flags) const
virtual void	setActiveVarCount (int val)=0
virtual void	setCalculateVarImportance (bool val)=0
virtual void	setTermCriteria (const TermCriteria &val)=0

https://docs.opencv.org/3.4/d0/d65/classcv_1_1ml_1_1RTrees.html

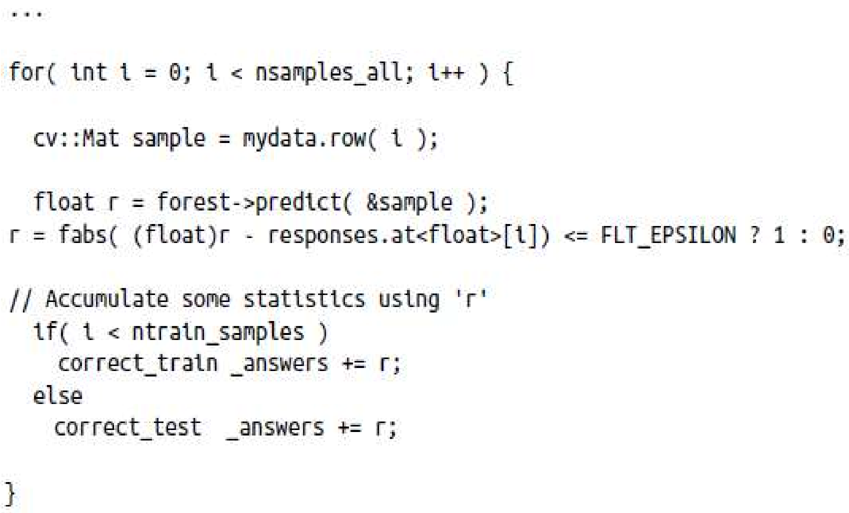
OpenCV

- Sample code

  https://github.com/opencv/opencv/blob/master/samples/cpp/letter_recog.cpp



```
...
Ptr<ml::RTrees> forest = ml::RTrees::create();
forest->setMaxDepth(10);
forest->setMinSampleCount(10);
forest->setMaxCategories(15);
forest->setCalculateVarImportance(true);
forest->setActiveVarCount(4);
forest->setTermCriteria(
    TermCriteria(
        TermCriteria::MAX_ITER+TermCriteria::EPSILON,
        100,
        0.01
    )
);
forest->train(tdata, 0);

...
```



```
...

for( int i = 0; i < nsamples_all; i++ ) {

  cv::Mat sample = mydata.row( i );

  float r = forest->predict( &sample );
r = fabs( (float)r - responses.at<float>[i]) <= FLT_EPSILON ? 1 : 0;

// Accumulate some statistics using 'r'
  if( i < ntrain_samples )
    correct_train _answers += r;
  else
    correct_test  _answers += r;

}
```

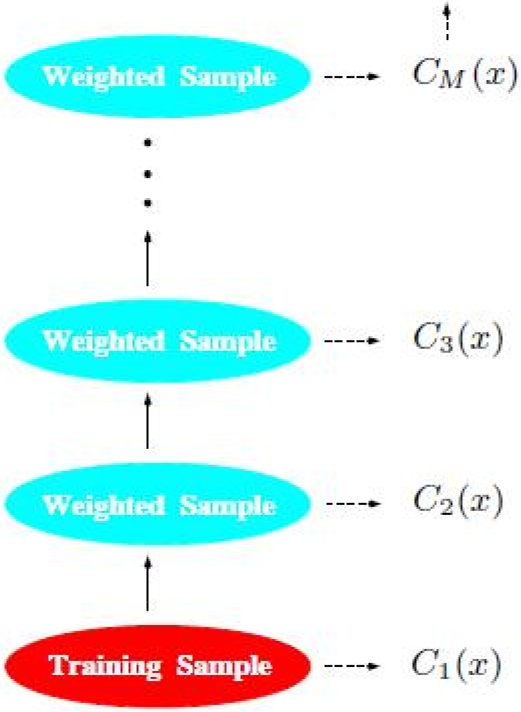
Boosting

- Boosting
  - Freund & Shapire, 1997
  - “약한 분류기를 모아 강한 분류기를 만든다 "
  - Ensemble method
    - Training data 의 weight (가중치) 변경
    - 다수결 투표 시 weight 반영
  - 종류
    - AdaBoost (Adaptive Boosting)
    - Gradient Boosting

	Method	Training data	Voting
Random Forest	Ensemble	Re-sampling	Majority vote
Boosting	Ensemble	Re-weighting	Weighted majority vote

AdaBoost

- Weighted sample
  - $C_m(x)$: Classifier m $(m = 1, \ldots, M)$ 의 training data
  - 같은 샘플에 대해서 Classifier 별로 다른 weight 부여
    - 잘못 분류한 샘플의 weight 를 증가



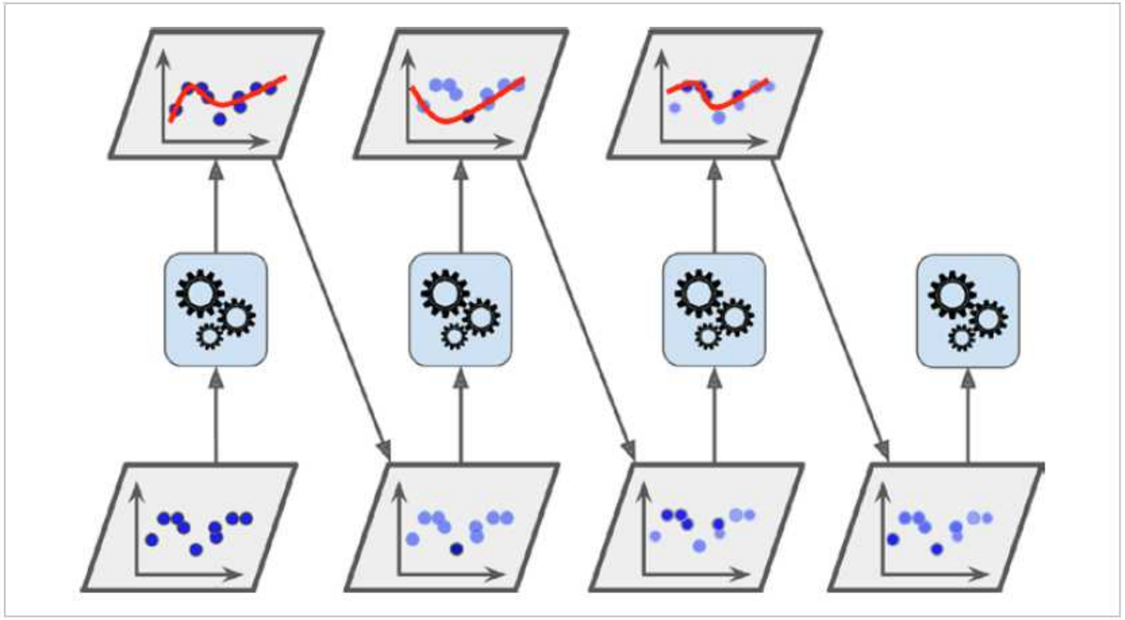
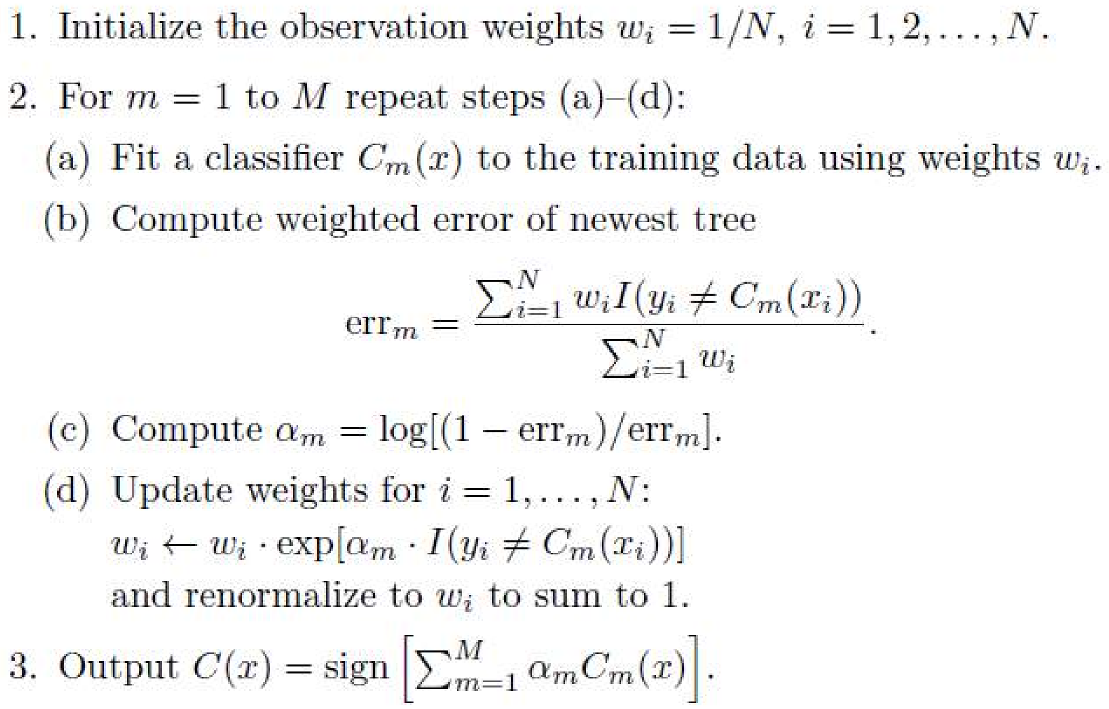


그림 7-7 샘플의 가중치를 업데이트하면서 순차적으로 학습하는 에이다부스트

AdaBoost

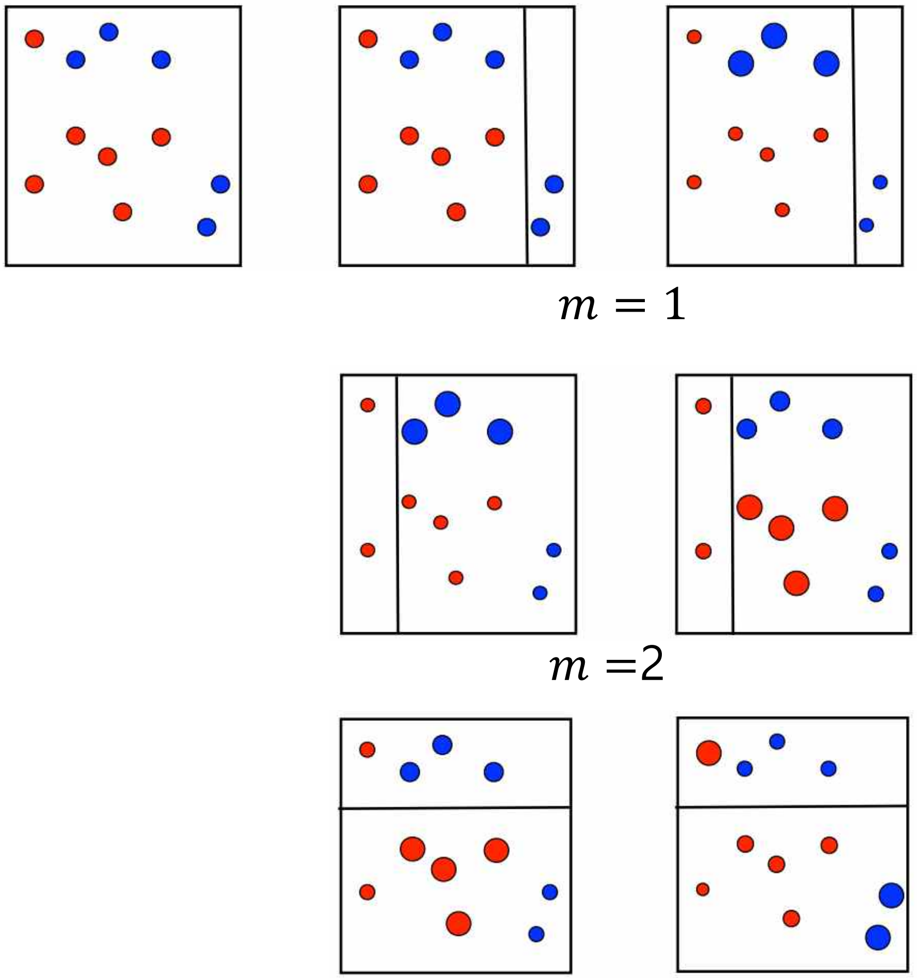
- Algorithm



```
1. Initialize the observation weights $w_i = 1/N,\ i = 1, 2, \ldots, N$.
2. For $m = 1$ to $M$ repeat steps (a)–(d):
    (a) Fit a classifier $C_m(x)$ to the training data using weights $w_i$.
    (b) Compute weighted error of newest tree
            $\mathrm{err}_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq C_m(x_i))}{\sum_{i=1}^{N} w_i}$.
    (c) Compute $\alpha_m = \log[(1 - \mathrm{err}_m)/\mathrm{err}_m]$.
    (d) Update weights for $i = 1, \ldots, N$:
        $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq C_m(x_i))]$
        and renormalize to $w_i$ to sum to 1.
3. Output $C(x) = \mathrm{sign}\left[\sum_{m=1}^{M} \alpha_m C_m(x)\right]$.
```

- w_i's are the weights of the samples.
- err_m is the weighted training error.

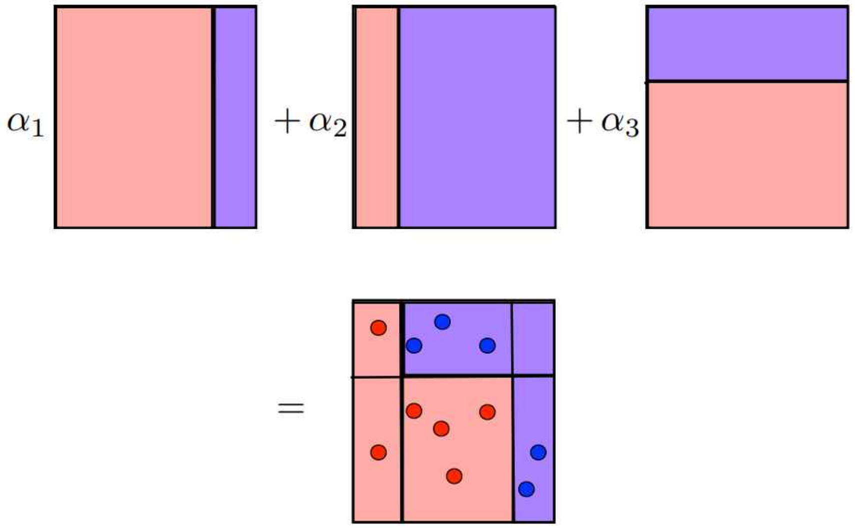
AdaBoost



$m = 1$

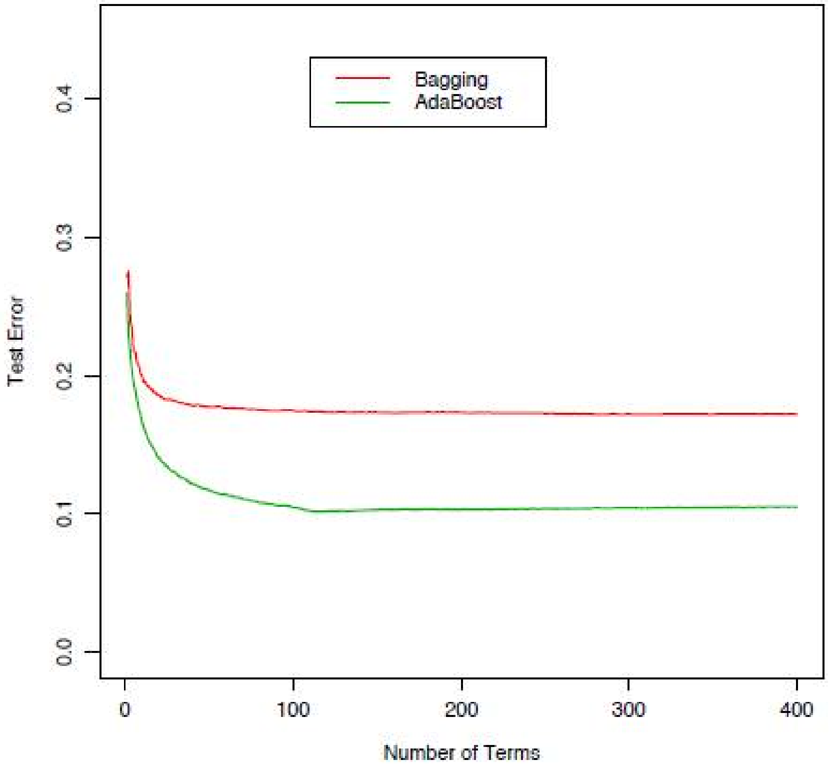
$m = 2$

$m = 3$



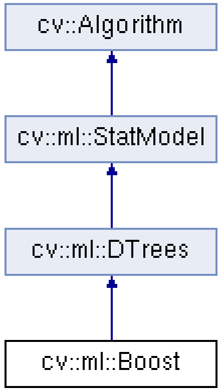
AdaBoost

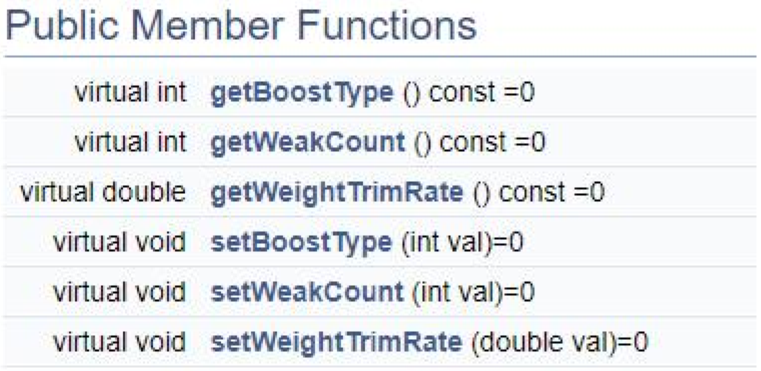
- Performance
  - 구현 용이, 빠른 계산 시간
  - Overfitting 완화
  - Noisy 데이터 영향이 클 수 있음



OpenCV

- cv::ml::Boost Class





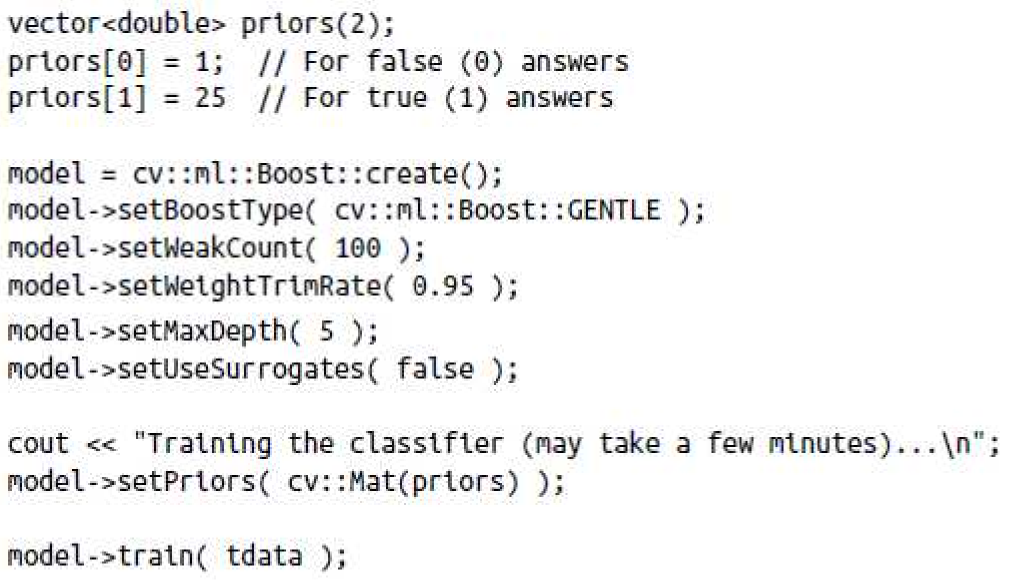
Public Member Functions

virtual int	**getBoostType** () const =0
virtual int	**getWeakCount** () const =0
virtual double	**getWeightTrimRate** () const =0
virtual void	**setBoostType** (int val)=0
virtual void	**setWeakCount** (int val)=0
virtual void	**setWeightTrimRate** (double val)=0

OpenCV

- Sample Code

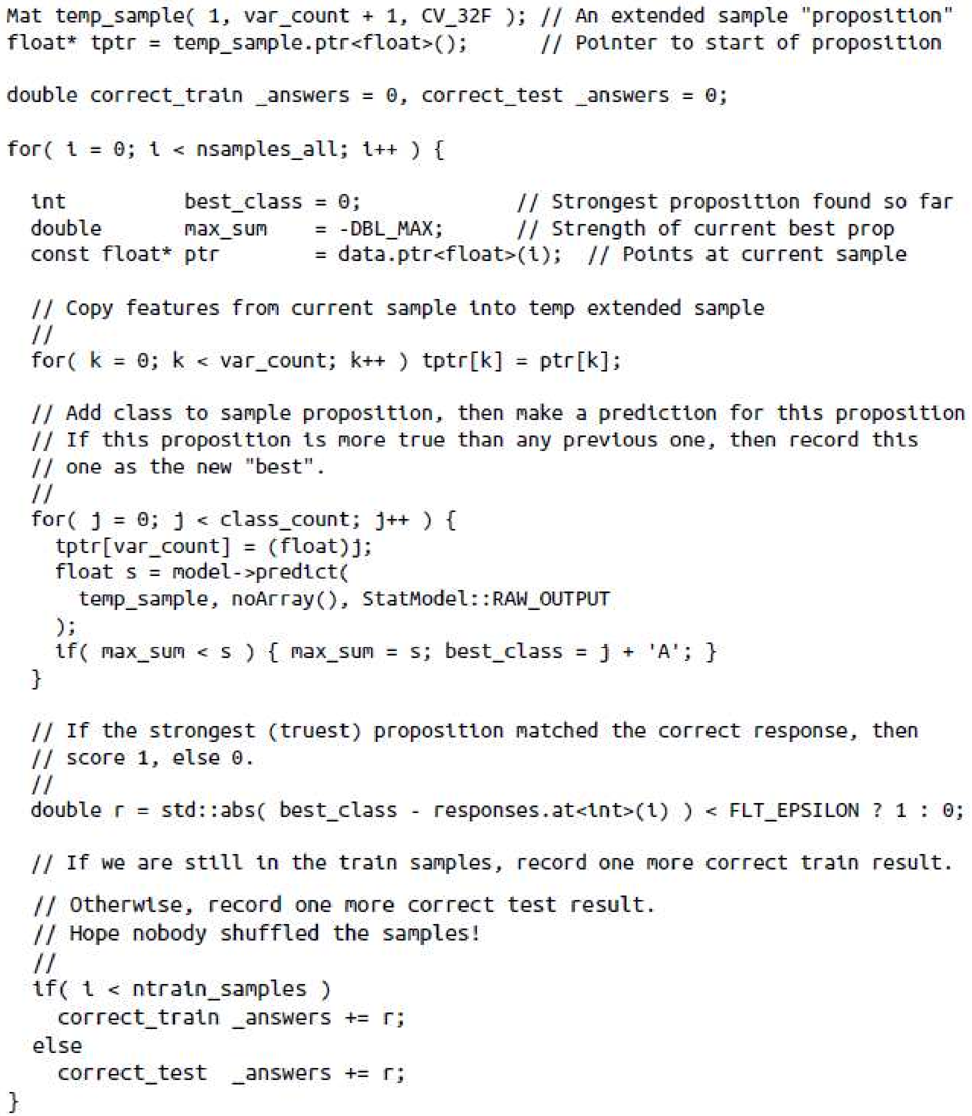
https://github.com/opencv/opencv/blob/master/samples/cpp/letter_recog.cpp



```
vector<double> priors(2);
priors[0] = 1;  // For false (0) answers
priors[1] = 25  // For true (1) answers

model = cv::ml::Boost::create();
model->setBoostType( cv::ml::Boost::GENTLE );
model->setWeakCount( 100 );
model->setWeightTrimRate( 0.95 );
model->setMaxDepth( 5 );
model->setUseSurrogates( false );

cout << "Training the classifier (may take a few minutes)...\n";
model->setPriors( cv::Mat(priors) );

model->train( tdata );
```

OpenCV

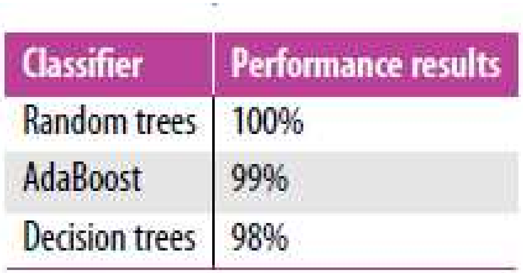
- Sample Code



```
Mat temp_sample( 1, var_count + 1, CV_32F ); // An extended sample "proposition"
float* tptr = temp_sample.ptr<float>();      // Pointer to start of proposition

double correct_train _answers = 0, correct_test _answers = 0;

for( i = 0; i < nsamples_all; i++ ) {

  int          best_class = 0;             // Strongest proposition found so far
  double       max_sum    = -DBL_MAX;      // Strength of current best prop
  const float* ptr        = data.ptr<float>(i);  // Points at current sample

  // Copy features from current sample into temp extended sample
  //
  for( k = 0; k < var_count; k++ ) tptr[k] = ptr[k];

  // Add class to sample proposition, then make a prediction for this proposition
  // If this proposition is more true than any previous one, then record this
  // one as the new "best".
  //
  for( j = 0; j < class_count; j++ ) {
    tptr[var_count] = (float)j;
    float s = model->predict(
      temp_sample, noArray(), StatModel::RAW_OUTPUT
    );
    if( max_sum < s ) { max_sum = s; best_class = j + 'A'; }
  }

  // If the strongest (truest) proposition matched the correct response, then
  // score 1, else 0.
  //
  double r = std::abs( best_class - responses.at<int>(i) ) < FLT_EPSILON ? 1 : 0;

  // If we are still in the train samples, record one more correct train result.
  // Otherwise, record one more correct test result.
  // Hope nobody shuffled the samples!
  //
  if( i < ntrain_samples )
    correct_train _answers += r;
  else
    correct_test  _answers += r;
}
```

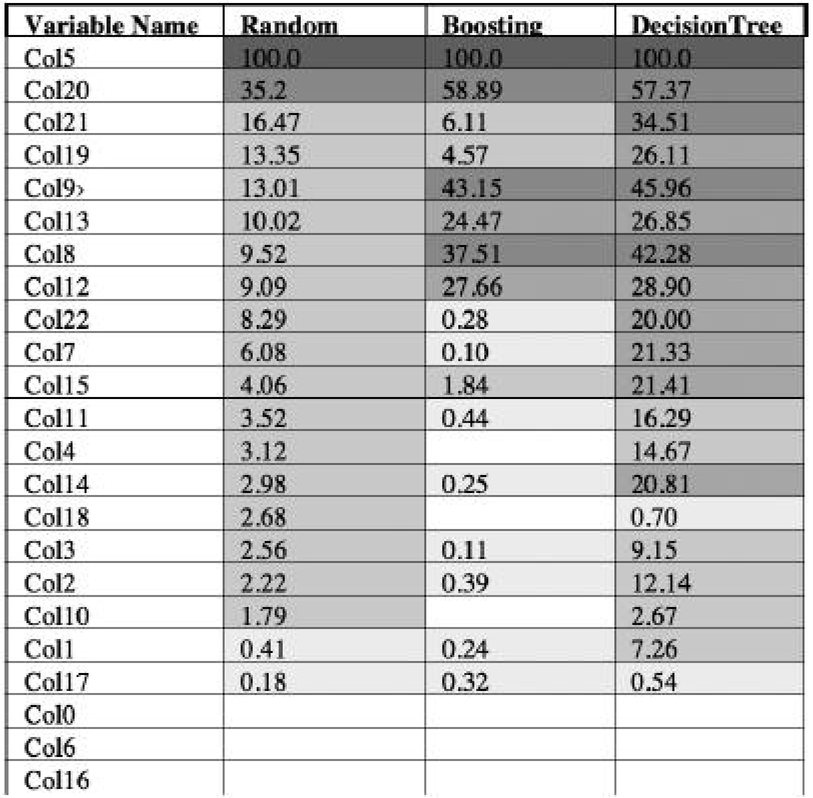
OpenCV

- Mushroom dataset



Classifier	Performance results
Random trees	100%
AdaBoost	99%
Decision trees	98%

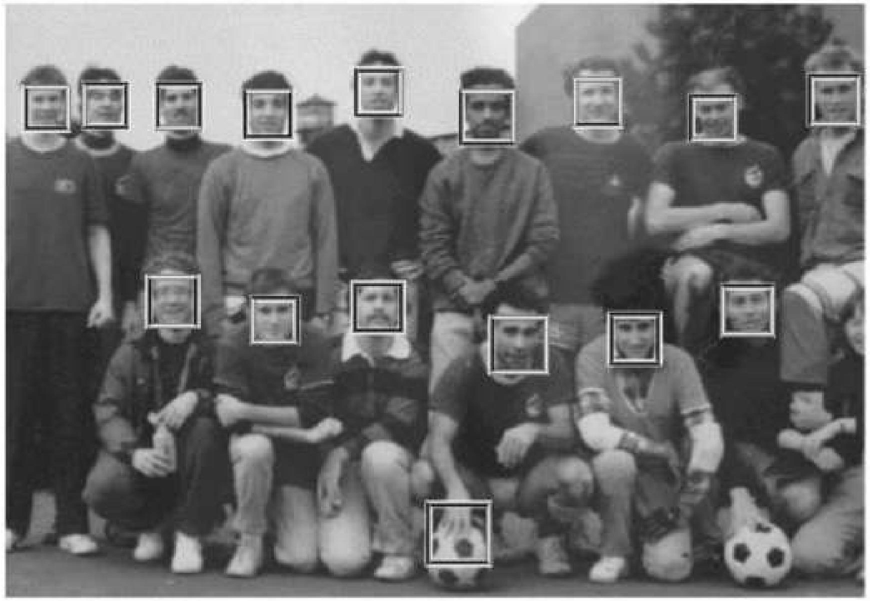
Variable importance



Variable Name	Random	Boosting	DecisionTree
Col5	100.0	100.0	100.0
Col20	35.2	58.89	57.37
Col21	16.47	6.11	34.51
Col19	13.35	4.57	26.11
Col9›	13.01	43.15	45.96
Col13	10.02	24.47	26.85
Col8	9.52	37.51	42.28
Col12	9.09	27.66	28.90
Col22	8.29	0.28	20.00
Col7	6.08	0.10	21.33
Col15	4.06	1.84	21.41
Col11	3.52	0.44	16.29
Col4	3.12		14.67
Col14	2.98	0.25	20.81
Col18	2.68		0.70
Col3	2.56	0.11	9.15
Col2	2.22	0.39	12.14
Col10	1.79		2.67
Col1	0.41	0.24	7.26
Col17	0.18	0.32	0.54
Col0			
Col6			
Col16			

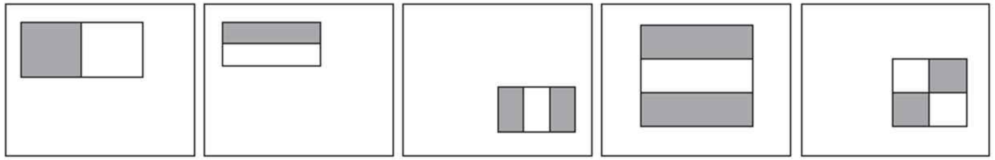
Cascade Classifier

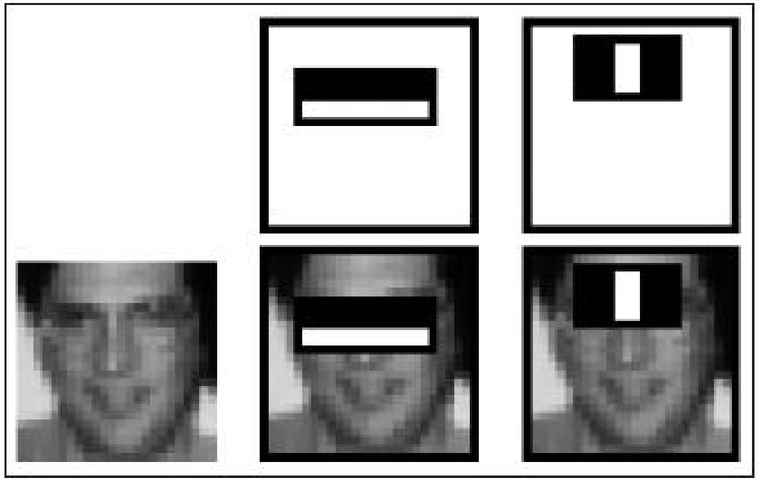
- Face detector
  - P.Viola & M. Jones, 2001
  - Harr-like filter 사용
  - AdaBoost 기반
  - Fast & Accurate (at the time)



Cascade Classifier

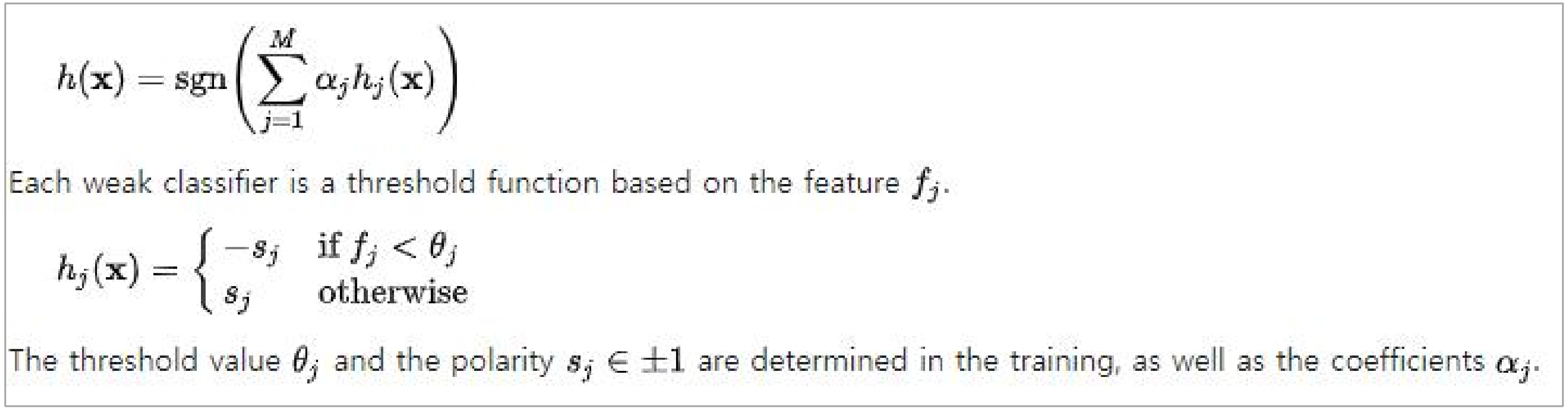
- Harr-like filter (유사 하르 필터)
  - 흑백 사각형이 서로 붙어있는 형태로 구성
  - 흰색영역의 픽셀값을 모두 더하고 검은색 영역의 픽셀값을 모두 빼서 하나의 특징값을 구함
  - 사람 얼굴: 밝은 영역 (이마, 미간, 볼 등) 과 어두운 영역 (눈썹, 볼, 이마 등) 이 정해져 있음 => Harr-like filter 로 특징값 설정 가능





Cascade Classifier

- Ensemble method
  - 각 Harr-like filter 는 weak classifier
  - 여러 개의 weak classifier 를 결합하여 하나의 strong classifier 구성



$$h(\mathbf{x}) = \mathrm{sgn}\left(\sum_{j=1}^{M} \alpha_j h_j(\mathbf{x})\right)$$

Each weak classifier is a threshold function based on the feature f_j.

$$h_j(\mathbf{x}) = \begin{cases} -s_j & \text{if } f_j < \theta_j \\ s_j & \text{otherwise} \end{cases}$$

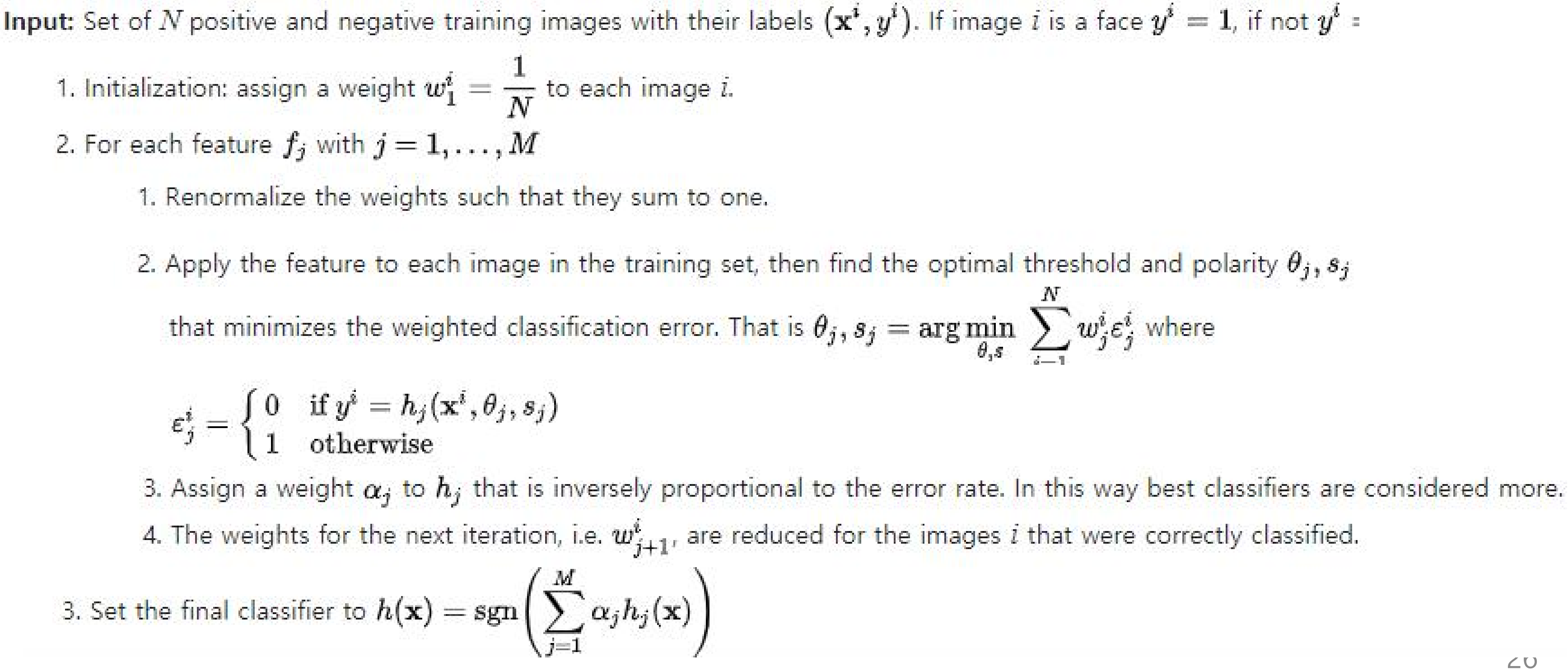
The threshold value θ_j and the polarity $s_j \in \pm 1$ are determined in the training, as well as the coefficients α_j.

Cascade Classifier

- Learning
  - 24x24 영상에서 생성가능한 Harr-like filter: 18만개

    ⇒Face detection 에 유용한 filter 선별 필요함

    ⇒Learning 알고리즘 (AdaBoost) 을 적용하여 6천개 선별함
  - AdaBoost 알고리즘



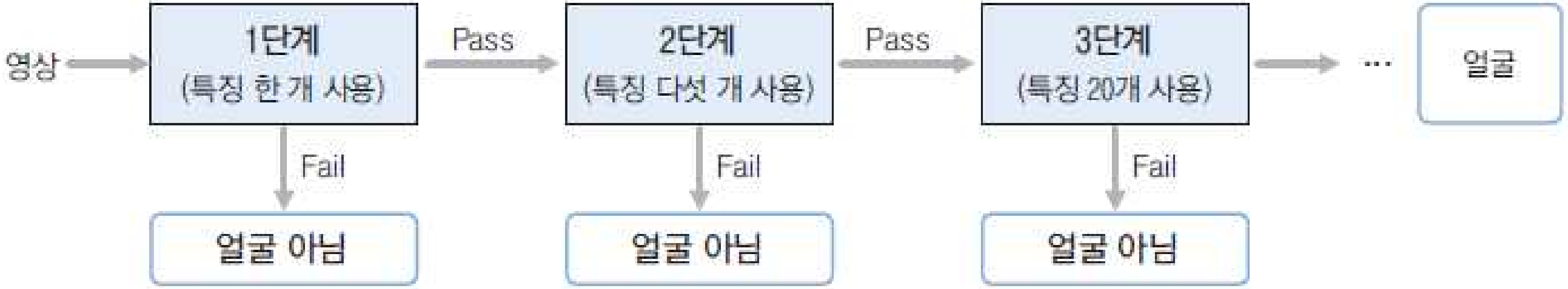
Input: Set of N positive and negative training images with their labels $(\mathbf{x}^i, y^i)$. If image i is a face $y^i = 1$, if not $y^i =$

1. Initialization: assign a weight $w_1^i = \frac{1}{N}$ to each image i.
2. For each feature f_j with $j = 1, \ldots, M$
    1. Renormalize the weights such that they sum to one.
    2. Apply the feature to each image in the training set, then find the optimal threshold and polarity θ_j, s_j that minimizes the weighted classification error. That is $\theta_j, s_j = \arg\min_{\theta, s} \sum_{i=1}^{N} w_j^i \varepsilon_j^i$ where $$\varepsilon_j^i = \begin{cases} 0 & \text{if } y^i = h_j(\mathbf{x}^i, \theta_j, s_j) \\ 1 & \text{otherwise} \end{cases}$$
    3. Assign a weight α_j to h_j that is inversely proportional to the error rate. In this way best classifiers are considered more.
    4. The weights for the next iteration, i.e. w_{j+1}^i, are reduced for the images i that were correctly classified.
3. Set the final classifier to $h(\mathbf{x}) = \text{sgn}\left(\sum_{j=1}^{M} \alpha_j h_j(\mathbf{x})\right)$

Cascade Classifier

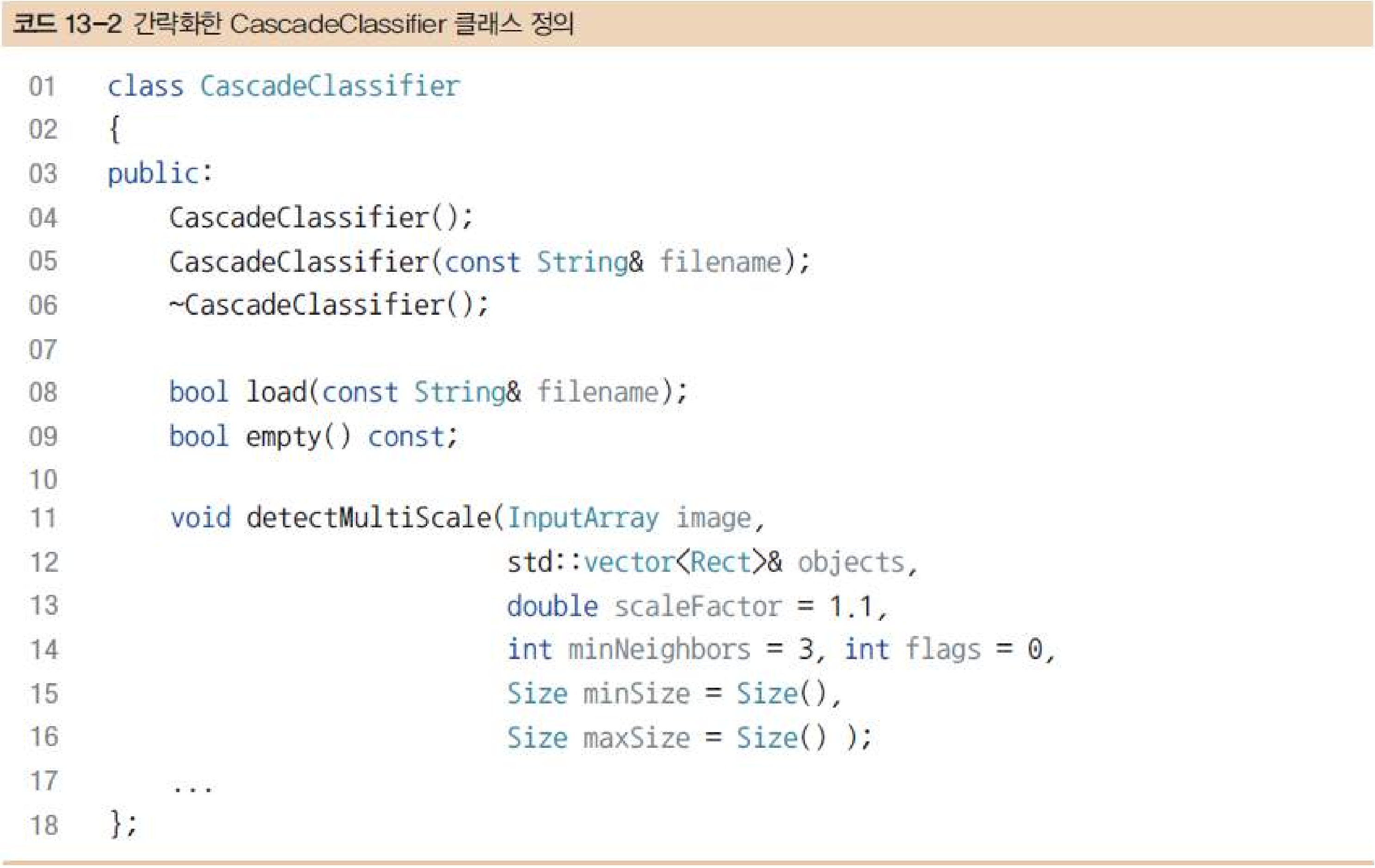
- Cascade architecture
  - 전체 영상에서 얼굴이 아닌 영역을 빠르게 걸러 내는 방식 적용
    - 1단계: Harr-like filter 1개 사용
    - 2단계: Harr-like filter 5개 사용
    - 3단계: Harr-like filter 20개 사용

    ……
  - 15배 이상 빠른 얼굴 검출 가능



OpenCV

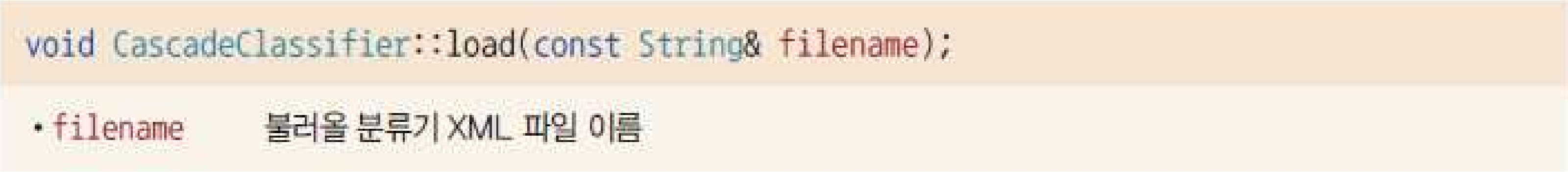
- CascadeClassifier 클래스



코드 13-2 간략화한 CascadeClassifier 클래스 정의

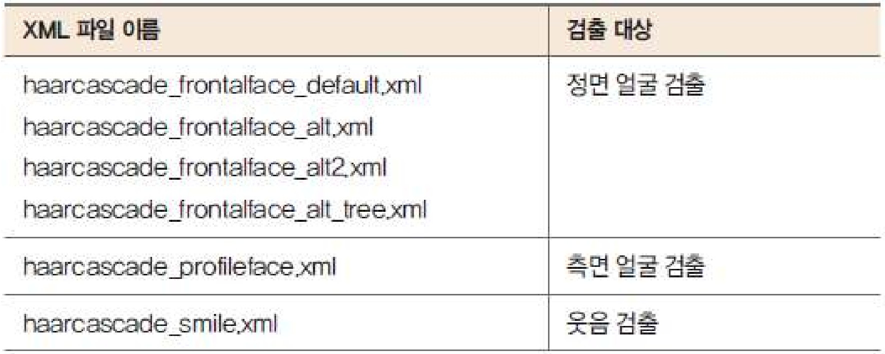
```
class CascadeClassifier
{
public:
    CascadeClassifier();
    CascadeClassifier(const String& filename);
    ~CascadeClassifier();

    bool load(const String& filename);
    bool empty() const;

    void detectMultiScale(InputArray image,
                          std::vector<Rect>& objects,
                          double scaleFactor = 1.1,
                          int minNeighbors = 3, int flags = 0,
                          Size minSize = Size(),
                          Size maxSize = Size() );
    ...
};
```

OpenCV

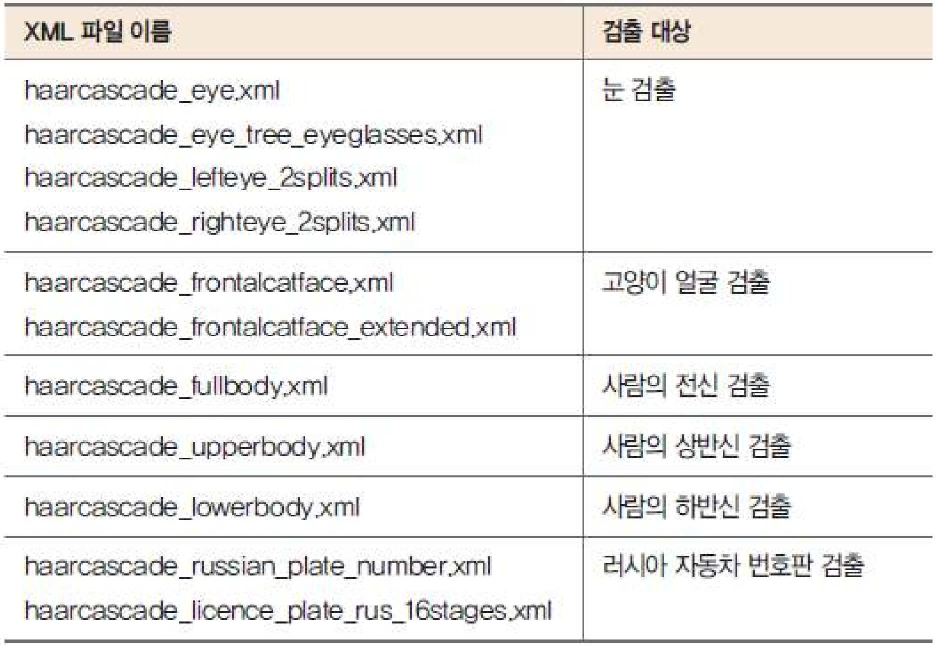


```
void CascadeClassifier::load(const String& filename);
```

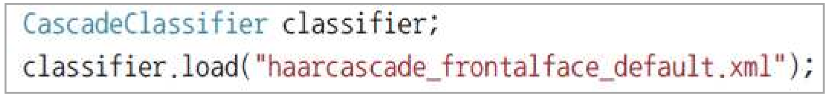
- filename   불러올 분류기 XML 파일 이름



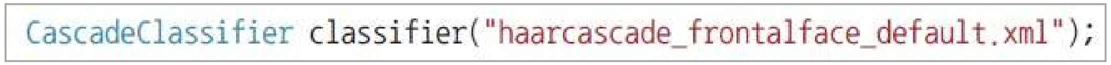
XML 파일 이름	검출 대상
haarcascade_frontalface_default.xml haarcascade_frontalface_alt.xml haarcascade_frontalface_alt2.xml haarcascade_frontalface_alt_tree.xml	정면 얼굴 검출
haarcascade_profileface.xml	측면 얼굴 검출
haarcascade_smile.xml	웃음 검출



XML 파일 이름	검출 대상
haarcascade_eye.xml haarcascade_eye_tree_eyeglasses.xml haarcascade_lefteye_2splits.xml haarcascade_righteye_2splits.xml	눈 검출
haarcascade_frontalcatface.xml haarcascade_frontalcatface_extended.xml	고양이 얼굴 검출
haarcascade_fullbody.xml	사람의 전신 검출
haarcascade_upperbody.xml	사람의 상반신 검출
haarcascade_lowerbody.xml	사람의 하반신 검출
haarcascade_russian_plate_number.xml haarcascade_licence_plate_rus_16stages.xml	러시아 자동차 번호판 검출

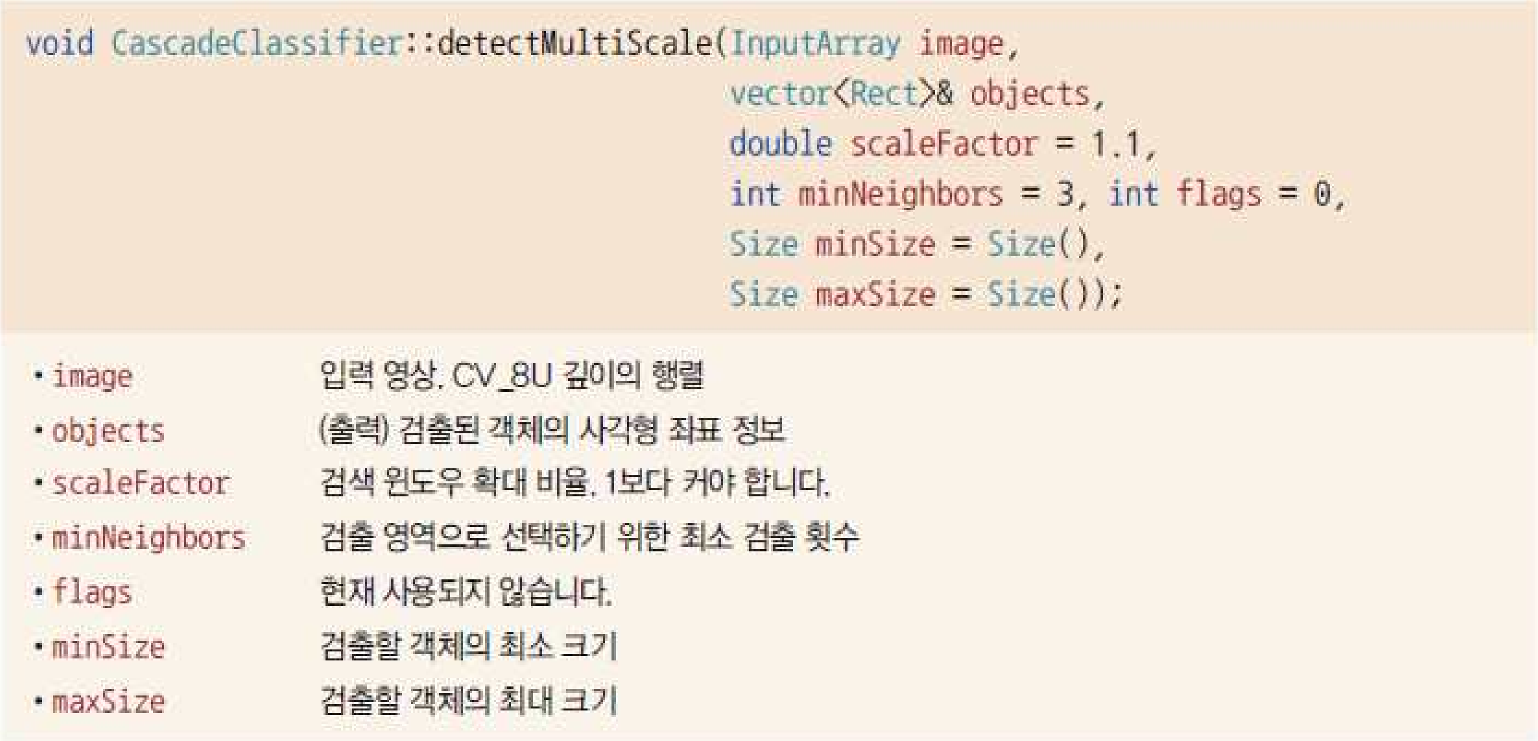


```
CascadeClassifier classifier;
classifier.load("haarcascade_frontalface_default.xml");
```



```
CascadeClassifier classifier("haarcascade_frontalface_default.xml");
```

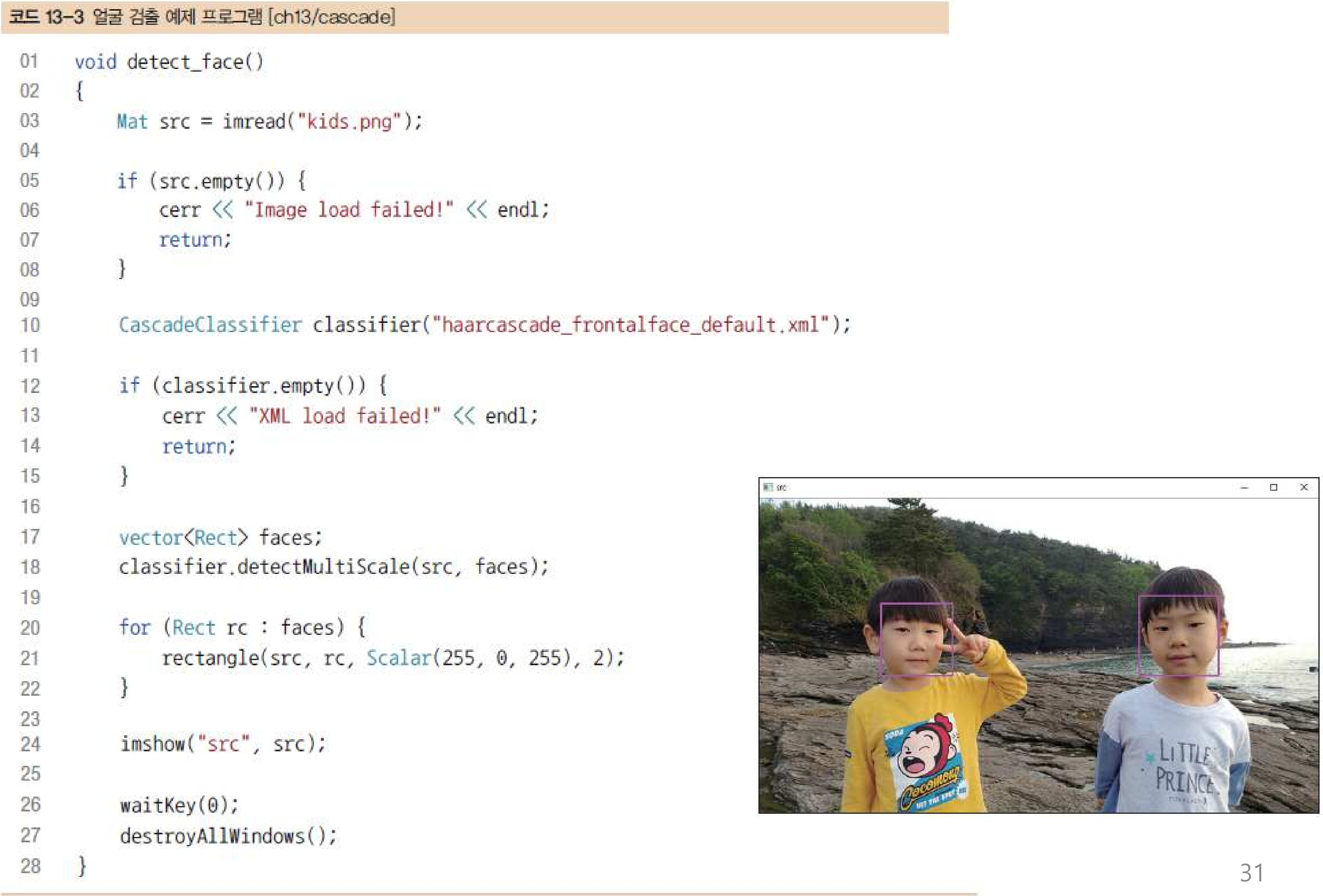
OpenCV



```
void CascadeClassifier::detectMultiScale(InputArray image,
                                         vector<Rect>& objects,
                                         double scaleFactor = 1.1,
                                         int minNeighbors = 3, int flags = 0,
                                         Size minSize = Size(),
                                         Size maxSize = Size());
```

• image	입력 영상. CV_8U 깊이의 행렬
• objects	(출력) 검출된 객체의 사각형 좌표 정보
• scaleFactor	검색 윈도우 확대 비율. 1보다 커야 합니다.
• minNeighbors	검출 영역으로 선택하기 위한 최소 검출 횟수
• flags	현재 사용되지 않습니다.
• minSize	검출할 객체의 최소 크기
• maxSize	검출할 객체의 최대 크기

OpenCV



코드 13-3 얼굴 검출 예제 프로그램 [ch13/cascade]

```cpp
void detect_face()
{
    Mat src = imread("kids.png");

    if (src.empty()) {
        cerr << "Image load failed!" << endl;
        return;
    }

    CascadeClassifier classifier("haarcascade_frontalface_default.xml");

    if (classifier.empty()) {
        cerr << "XML load failed!" << endl;
        return;
    }

    vector<Rect> faces;
    classifier.detectMultiScale(src, faces);

    for (Rect rc : faces) {
        rectangle(src, rc, Scalar(255, 0, 255), 2);
    }

    imshow("src", src);

    waitKey(0);
    destroyAllWindows();
}
```

OpenCV

코드 13-4 눈 검출 예제 프로그램 [ch13/cascade]

```cpp
void detect_eyes()
{
    Mat src = imread("kids.png");

    if (src.empty()) {
        cerr << "Image load failed!" << endl;
        return;
    }

    CascadeClassifier face_classifier("haarcascade_frontalface_default.xml");
    CascadeClassifier eye_classifier("haarcascade_eye.xml");

    if (face_classifier.empty() || eye_classifier.empty()) {
        cerr << "XML load failed!" << endl;
        return;
    }

    vector<Rect> faces;
    face_classifier.detectMultiScale(src, faces);

    for (Rect face : faces) {
        rectangle(src, face, Scalar(255, 0, 255), 2);

        Mat faceROI = src(face);
        vector<Rect> eyes;
        eye_classifier.detectMultiScale(faceROI, eyes);

        for (Rect eye : eyes) {
            Point center(eye.x + eye.width / 2, eye.y + eye.height / 2);
            circle(faceROI, center, eye.width / 2, Scalar(255, 0, 0), 2, LINE_AA);
        }
    }

    imshow("src", src);

    waitKey(0);
    destroyAllWindows();
}
```