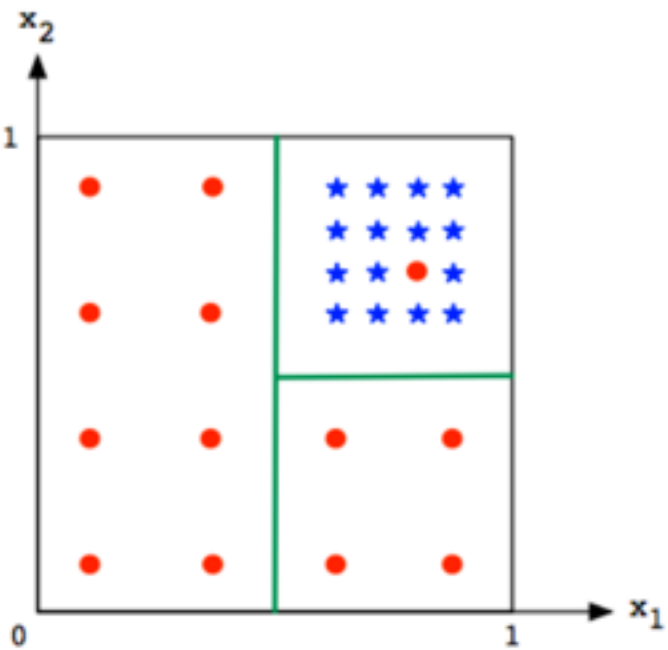
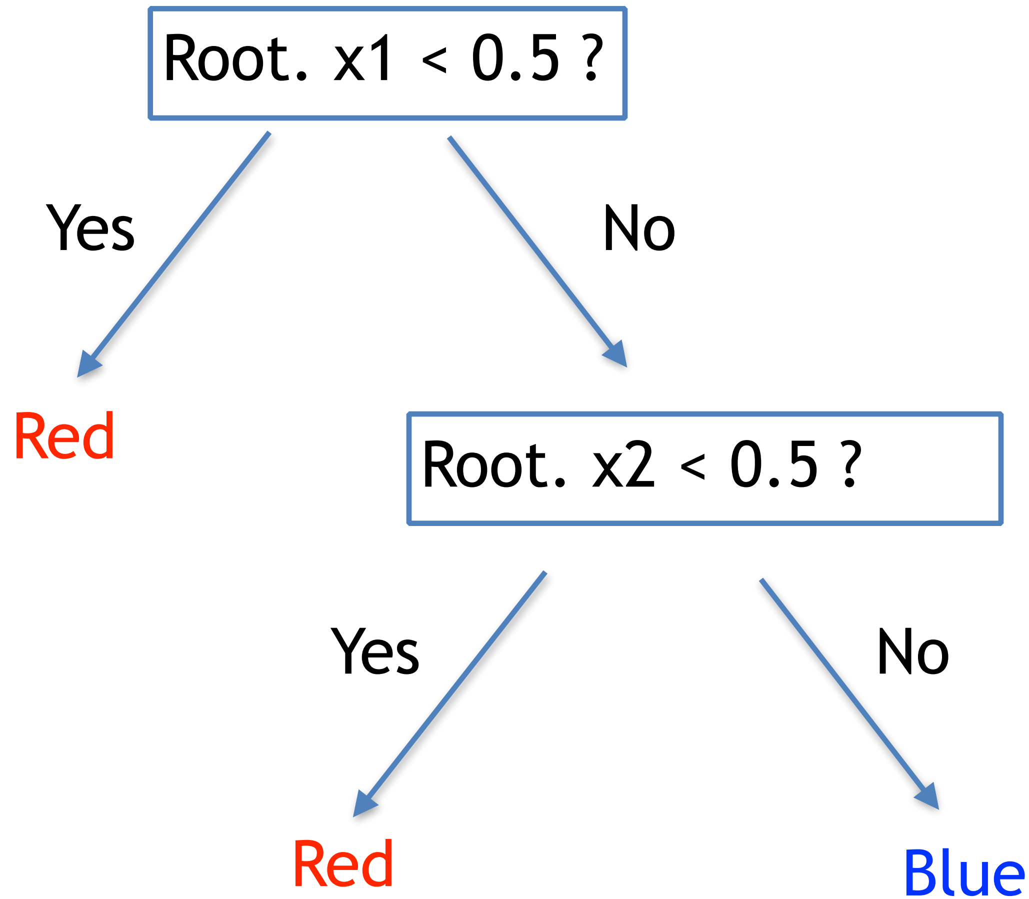
Random Forest

Decision Tree

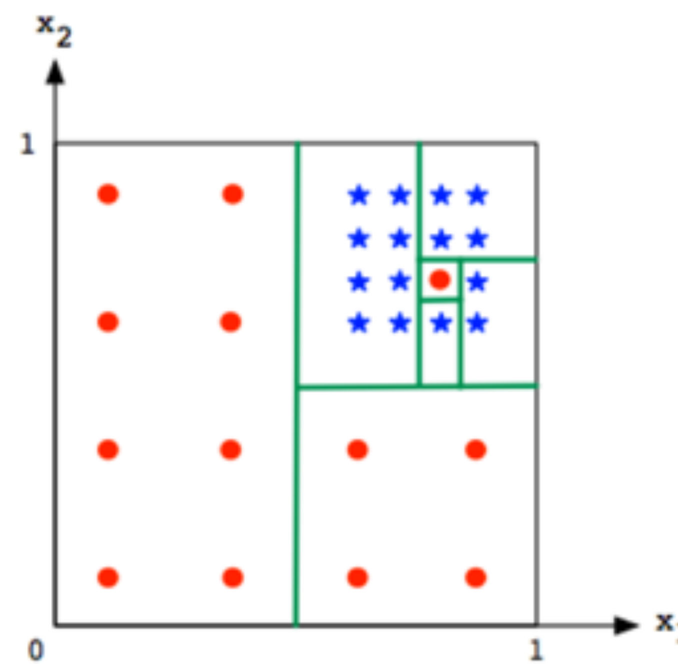
z = (X,Y)
X = (x1,x2)
Y in {0,1}





Decision Tree - Splitting Rule

ID3 Decision Tree: In each node, choose the feature and threshold which will reduce the uncertainty most, i.e., which will minimize the entropy of the data sets in the node after splitting by this rule.



Issue: Overfitting

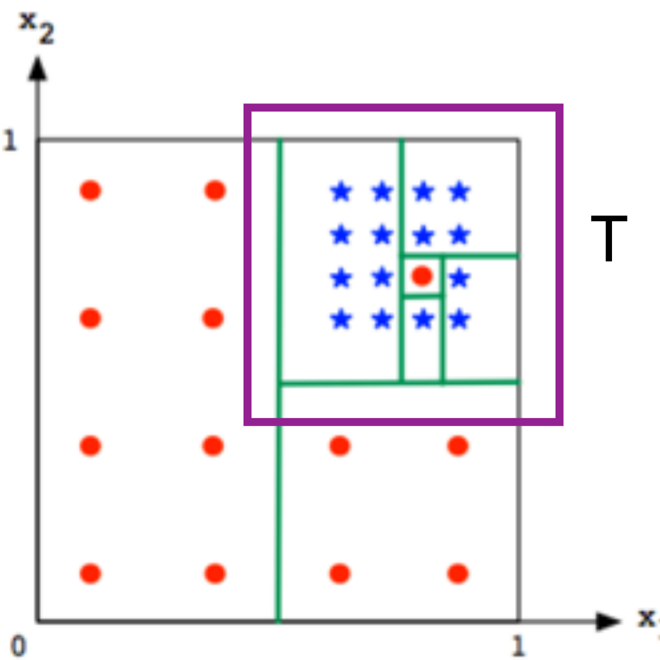
Solution: Prune

Decision Tree - prune

Decision Tree - prune

S = {training}U{test}

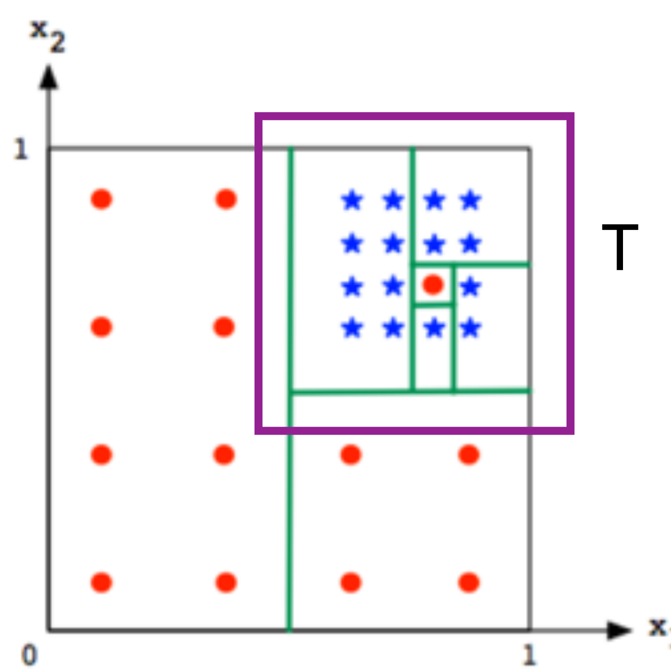
build the tree



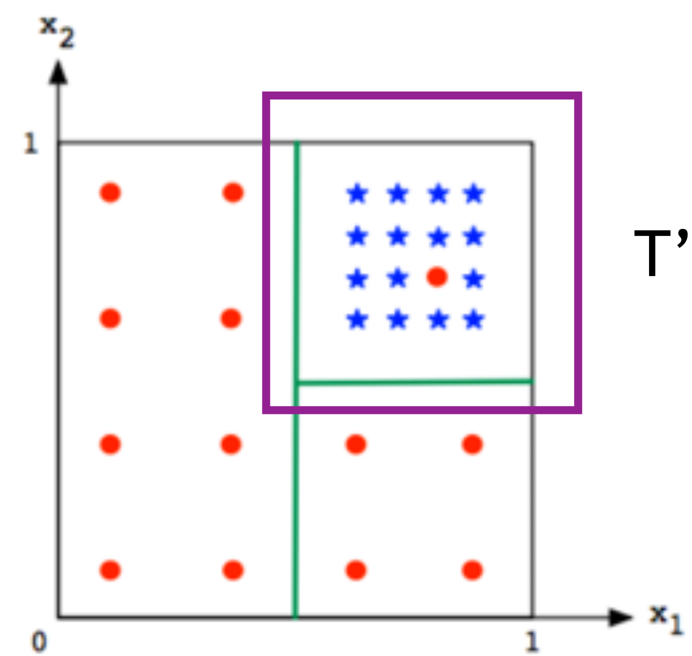
Decision Tree - prune

S = {training}U{test}

build the tree



T



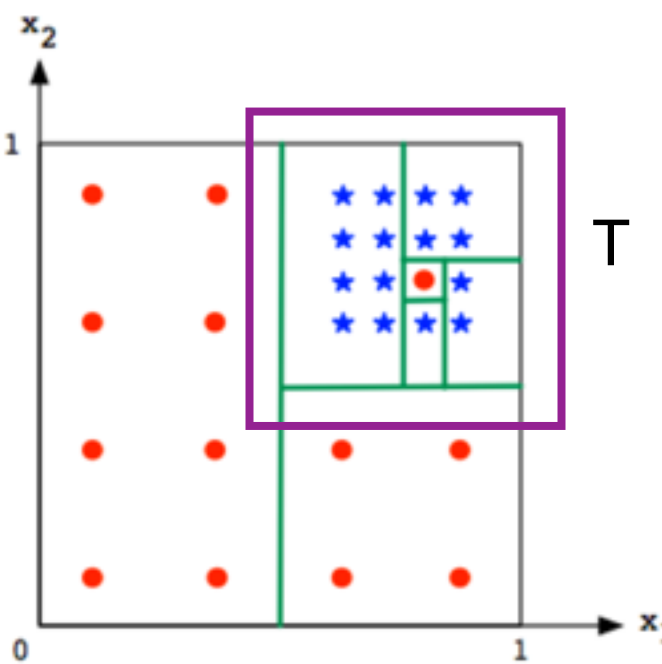
T’

Decision Tree - prune

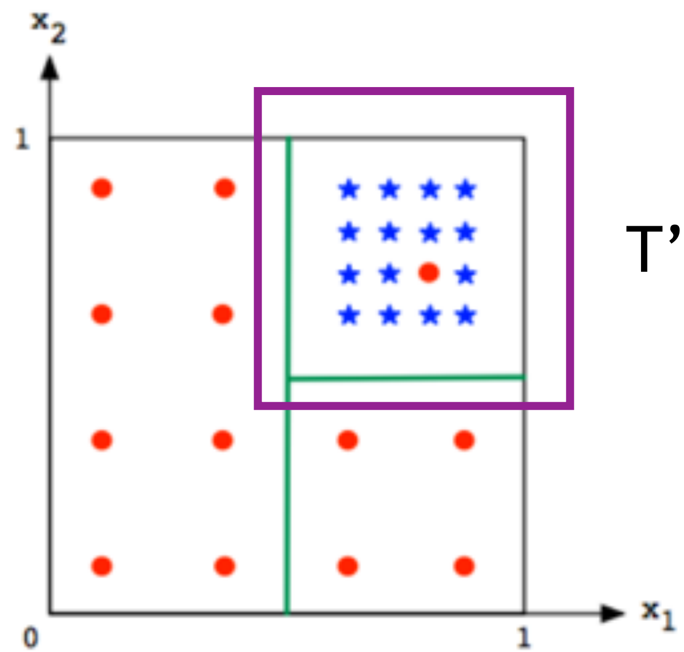
S = {training}U{test}

build the tree

if error(T) > error(T') on test data set, replace T with T'.



T



T'

Decision Tree - drawback

Overfitting the data, low bias, very high variance

Random Forest

Construct a multitude of decision trees at training time and predict the class that is the majority of the classes output by individual tree

Tree bagging

Tree bagging: Bootstrap training samples for each tree.

Original training set: X = X1, X2, X3, …, Xn with response

Y = y1,y2, y3, …, yn. Xi has M features, i.e.,

Xi = (x_i1,x_i2,x_i3,…, x_iM)

For b = 1, … B:

1. sample, with replacement, n training samples, (X1_b,y1_b), (X2_b,y2_b), … ,(Xn_b,yn_b) from X, Y, denoted as

Xb, Yb

2. Train a decision tree Tree_b on Xb, Yb.

Using random features

Using random features

- For each bootstrapped training sample, randomly choose F features out of M features. (F = logM + 1)

Using random features

- For each bootstrapped training sample, randomly choose F features out of M features. (F = logM + 1)

Using random features

- For each bootstrapped training sample, randomly choose F features out of M features. ($F = \log M + 1$)

- Use the F features and bootstrap sample Xb, Yb to grow a decision tree.

Using random features

- For each bootstrapped training sample, randomly choose F features out of M features. (F = logM + 1)

- Use the F features and bootstrap sample Xb, Yb to grow a decision tree.

Using random features

- For each bootstrapped training sample, randomly choose F features out of M features. (F = logM + 1)
- Use the F features and bootstrap sample Xb, Yb to grow a decision tree.
- We get B trees from Tree-bagging. The predicted class is the class with most popular votes from the decision trees.

Tree-bagging prediction

Pros:

1. Small variance without increasing the bias.
2. Bootstrap de-correlates the trees by providing them different training sets and features.

Random Forest Converge

margin function: $mg(\mathbf{X}, Y) = av_k I(h_k(\mathbf{X}) = Y) - \max_{j \neq Y} av_k I(h_k(\mathbf{X}) = j).$

where $h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_K(\mathbf{x}),$ are classifiers.

margin measures the extent to which the average number of votes at X, Y for the right class exceeds the average vote for any other class

generalization error: $PE^* = P_{\mathbf{X},Y}(mg(\mathbf{X}, Y) < 0)$

Theorem 1.2. *As the number of trees increases, for almost surely all sequences* $\Theta_{1,\ldots}PE^*$ *converges to*

$$P_{\mathbf{X},Y}(P_\Theta(h(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_\Theta(h(\mathbf{X}, \Theta) = j) < 0). \qquad (1)$$

Out-of-bag error

Out-of-bag error

- Estimate of generalization error: out-of-bag error

Out-of-bag error

- Estimate of generalization error: out-of-bag error

Out-of-bag error

- Estimate of generalization error: out-of-bag error

- For each training sample xi, aggregate votes over the trees that do NOT have xi in their bootstrap sample. We call these trees as out-of-bag classifiers. Out-of-bag prediction of xi is the class with most votes of out-of-bag classifiers.

Out-of-bag error

- Estimate of generalization error: out-of-bag error

- For each training sample xi, aggregate votes over the trees that do NOT have xi in their bootstrap sample. We call these trees as out-of-bag classifiers. Out-of-bag prediction of xi is the class with most votes of out-of-bag classifiers.

Out-of-bag error

- Estimate of generalization error: out-of-bag error
- For each training sample xi, aggregate votes over the trees that do NOT have xi in their bootstrap sample. We call these trees as out-of-bag classifiers. Out-of-bag prediction of xi is the class with most votes of out-of-bag classifiers.

Out-of-bag error

- Estimate of generalization error: out-of-bag error

- For each training sample xi, aggregate votes over the trees that do NOT have xi in their bootstrap sample. We call these trees as out-of-bag classifiers. Out-of-bag prediction of xi is the class with most votes of out-of-bag classifiers.

- Estimate of generalization error: error rate of the out-of-bag classifiers on the training set.

out-of-bag error - continue

out-of-bag error - continue

- The out-of-bag classifiers (decision tree) for (xi,yi) are {T_i1, T_i2,T_i3}. It means the bootstrapped sample of each tree doesn't contain xi.

out-of-bag error - continue

- The out-of-bag classifiers (decision tree) for (xi,yi) are {T_i1, T_i2,T_i3}. It means the bootstrapped sample of each tree doesn't contain xi.

out-of-bag error - continue

- The out-of-bag classifiers (decision tree) for (xi,yi) are {T_i1, T_i2,T_i3}. It means the bootstrapped sample of each tree doesn’t contain xi.

- The votes of xi from the three tree are T_i1(xi), T_i2(xi), T_i3(xi).Let ‘j’ be the class getting most votes. If the ‘j’ is equal to yi, the prediction is correct, otherwise it’s wrong.

out-of-bag error - continue

- The out-of-bag classifiers (decision tree) for (xi,yi) are {T_i1, T_i2,T_i3}. It means the bootstrapped sample of each tree doesn’t contain xi.

- The votes of xi from the three tree are T_i1(xi), T_i2(xi), T_i3(xi).Let ‘j’ be the class getting most votes. If the ‘j’ is equal to yi, the prediction is correct, otherwise it’s wrong.

out-of-bag error - continue

- The out-of-bag classifiers (decision tree) for (xi,yi) are {T_i1, T_i2,T_i3}. It means the bootstrapped sample of each tree doesn’t contain xi.

- The votes of xi from the three tree are T_i1(xi), T_i2(xi), T_i3(xi).Let ‘j’ be the class getting most votes. If the ‘j’ is equal to yi, the prediction is correct, otherwise it’s wrong.

- We compute the out-of-bag error by computing the error rate of the out-of-bag classifiers on the training data.

Performance of random forest: strength

Strength: measure how accurate the individual classifier a is.

Definition 2.1. The margin function for a random forest is

$$mr(\mathbf{X}, Y) = P_\Theta(h(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_\Theta(h(\mathbf{X}, \Theta) = j)$$

and the strength of the set of classifiers $\{h(\mathbf{x}, \Theta)\}$ is

$$s = E_{\mathbf{X},Y}\, mr(\mathbf{X}, Y).$$

Performance of random forest: correlation

Correlation: measure the dependence between decision trees

$$\hat{j}(\mathbf{X}, Y) = \arg\max_{j \neq Y} P_\Theta(h(\mathbf{X}, \Theta) = j)$$

Definition 2.2. The raw margin function is

$$rmg(\Theta, \mathbf{X}, Y) = I(h(\mathbf{X}, \Theta) = Y) - I(h(\mathbf{X}, \Theta) = \hat{j}(\mathbf{X}, Y)).$$

$$\bar{\rho} = E_{\Theta,\Theta'}(\rho(\Theta, \Theta')sd(\Theta)sd(\Theta'))/E_{\Theta,\Theta'}(sd(\Theta)sd(\Theta'))$$

where $\rho(\Theta, \Theta')$ is the correlation between $rmg(\Theta, \mathbf{X}, Y)$ and $rmg(\Theta', \mathbf{X}, Y)$ holding Θ, Θ' fixed and $sd(\Theta)$ is the standard deviation of $rmg(\Theta, \mathbf{X}, Y)$ holding Θ fixed. Then,

Performance of random forest

High strength and low correlation generate better random forest,i.e., lower generalization error. Both of them can be estimated from data, see Appendix

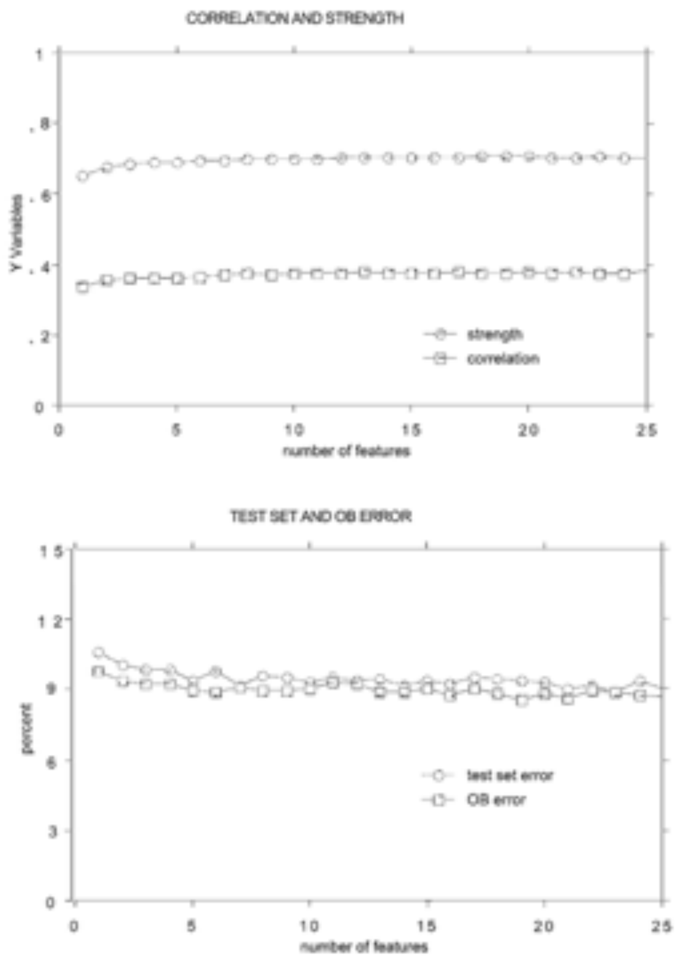


Figure 3. Effect of number of features on satellite data.

Random forest:rank feature importance

Random forest:rank feature importance

- The values of j-th feature of the training data are permuted.

Random forest:rank feature importance

- The values of j-th feature of the training data are permuted.

Random forest:rank feature importance

- The values of j-th feature of the training data are permuted.

- Recompute the out-of-bag error for the permuted training data.

Random forest:rank feature importance

- The values of j-th feature of the training data are permuted.

- Recompute the out-of-bag error for the permuted training data.

Random forest:rank feature importance

- The values of j-th feature of the training data are permuted.
- Recompute the out-of-bag error for the permuted training data.
- Importance score for j-th feature is computed by the difference (percent increase) in out-of-bag error after and before the permutation.

Random forest:rank feature importance

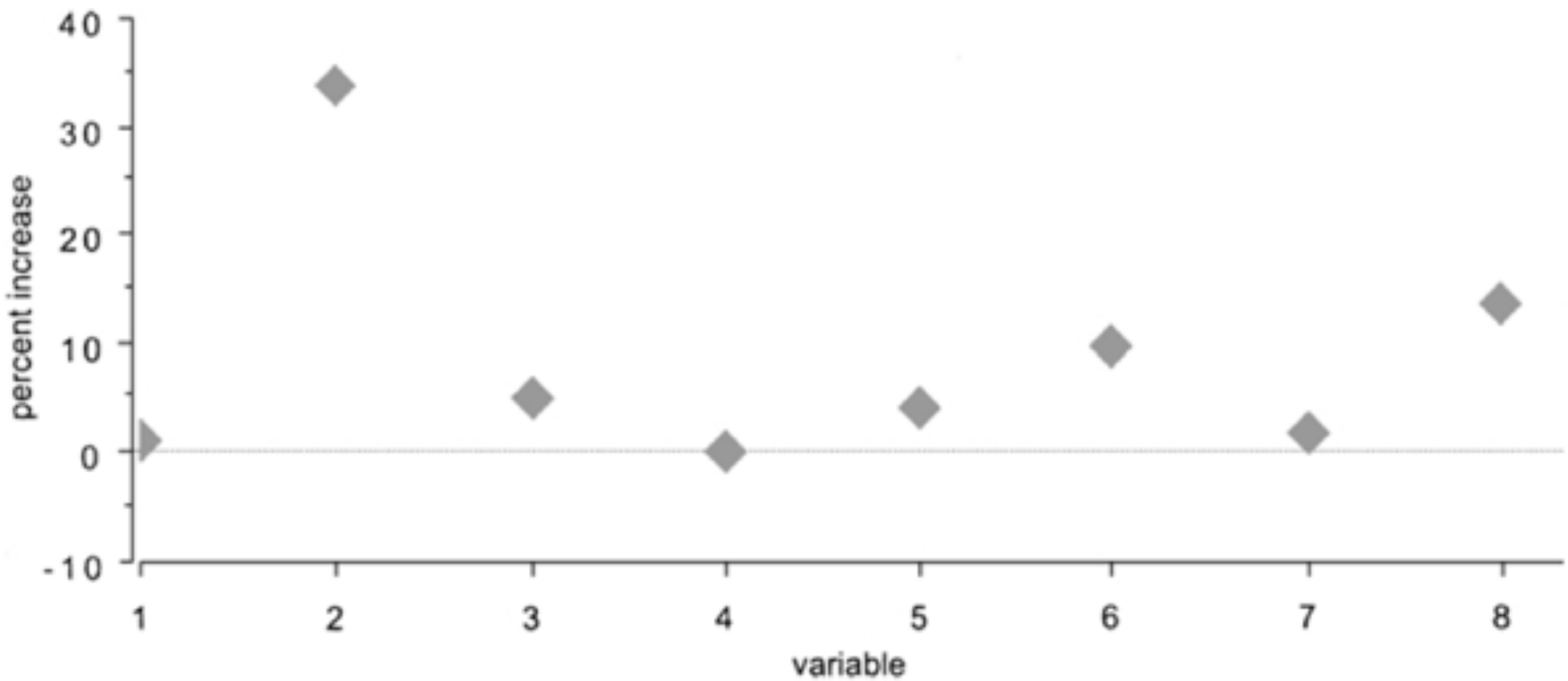


Figure 4. Measure of variable importance—diabetes data.