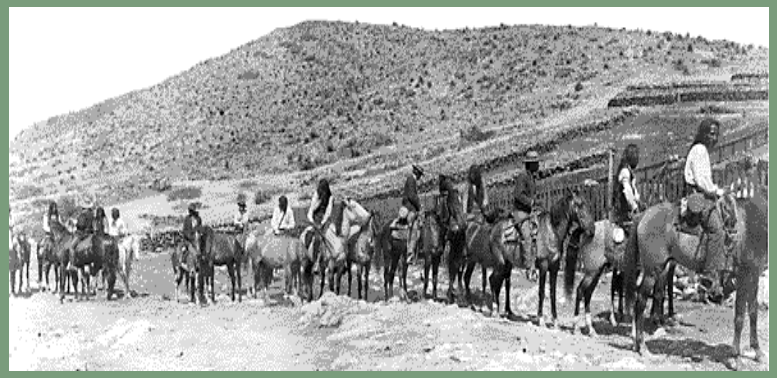
Diabetes Classification

Zhenhuan Cui
Crystal Dong
Juan Du
Lynn Friedmann
Yanxing Zhao

Data Source and Description

- UCI Machine Learning Database
- National Institute of Diabetes and Digestive and Kidney Diseases
- Diabetes of Pima Indian Women
- 768 cases
  - 268 positive for diabetes (1)
  - 500 negative for diabetes (0)

Who are Pima Indians?

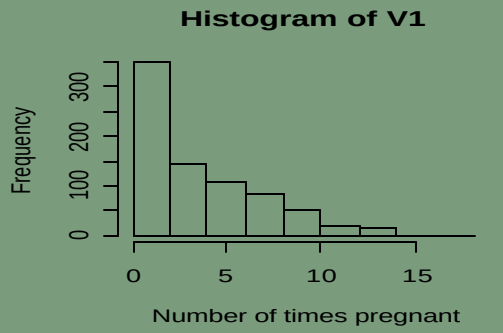


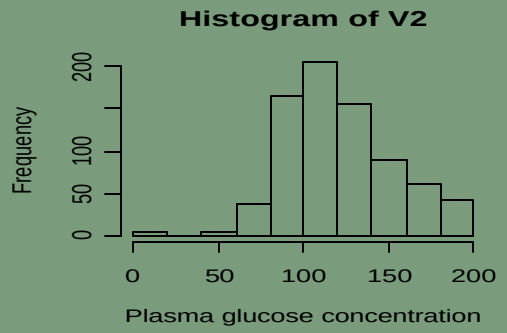
- Gila River Indian Community in Southern Arizona
- Within community marriage 2000+ yrs
- Volunteer for research studies
- Many Pima Indian women are diabetic
- NIH conducts genetic research

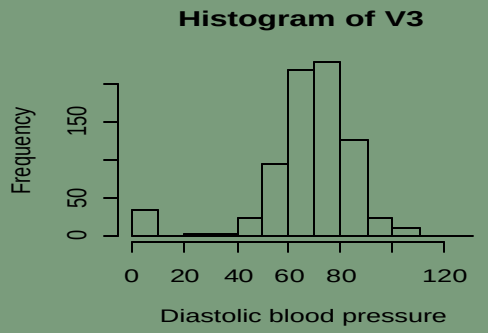
Explanatory Variables

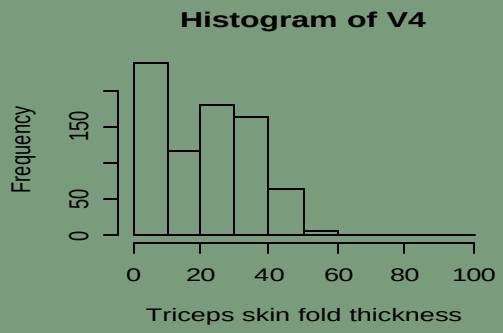
1. “npreg”—Number of times pregnant
2. “glu”—Plasma glucose concentration
3. “bp”—Diastolic blood pressure (mm Hg)
4. “skin”—Triceps skin fold thickness (mm)
5. “insu”—2-Hour serum insulin (mu U/ml)
6. “bmi”—Body mass index (kg/m^2)
7. “ped”—Diabetes pedigree function
8. “age”—Age (years).

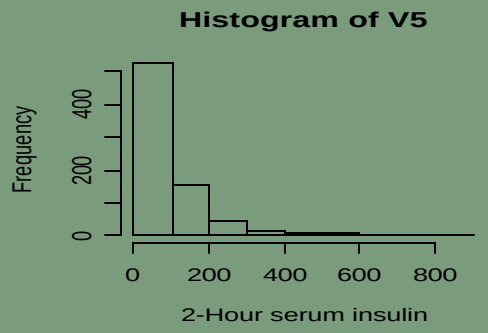
Explanatory variables

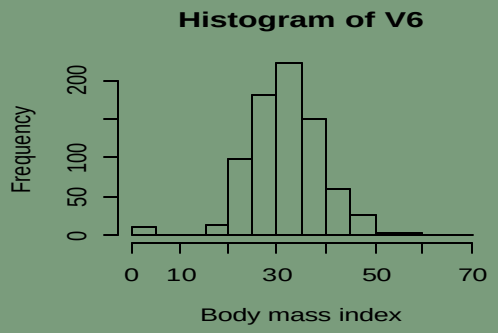


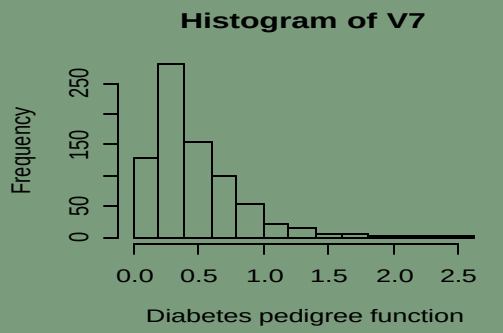


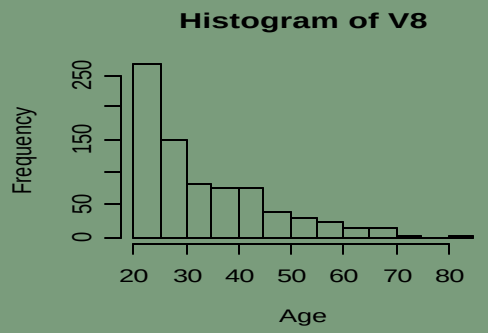










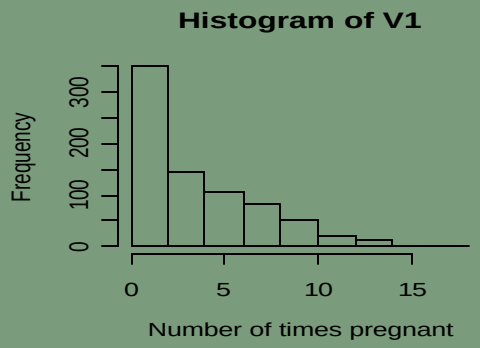


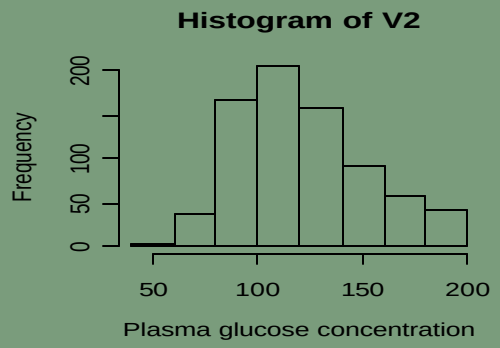
Missing Value

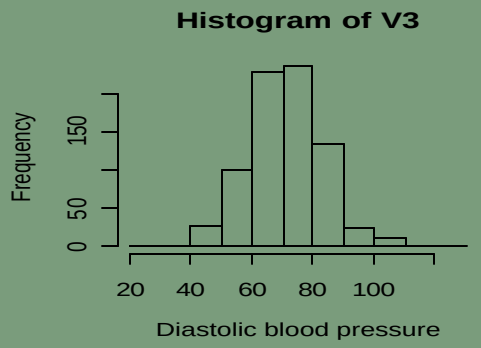
variable	no. missing	% missing
glu	6	0.007813
bp	35	0.045573
skin	227	0.295573
insu	374	0.486979
bmi	11	0.014323

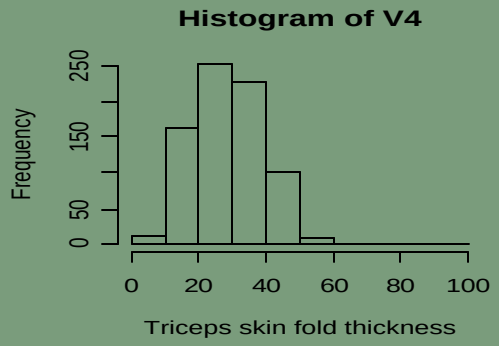
- Nearest neighbor imputation
- Euclidian distance measure

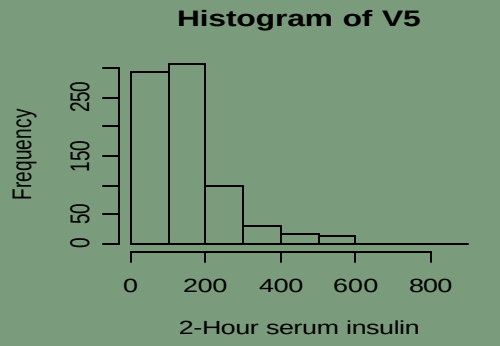
After Imputation

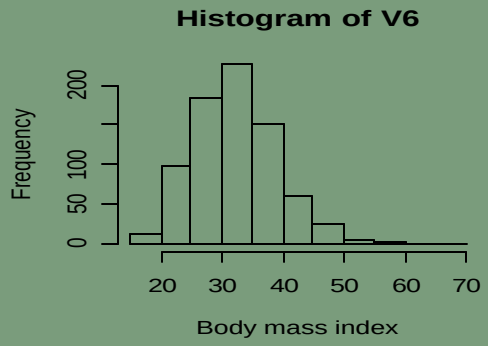


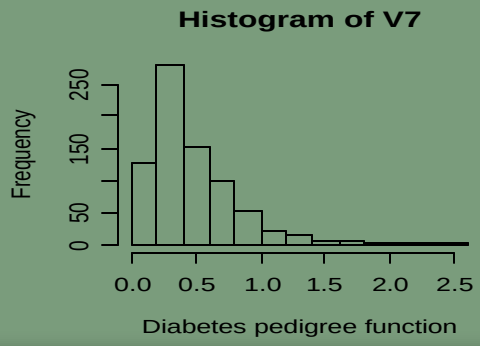


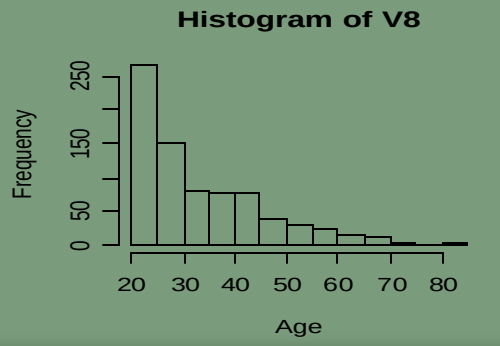




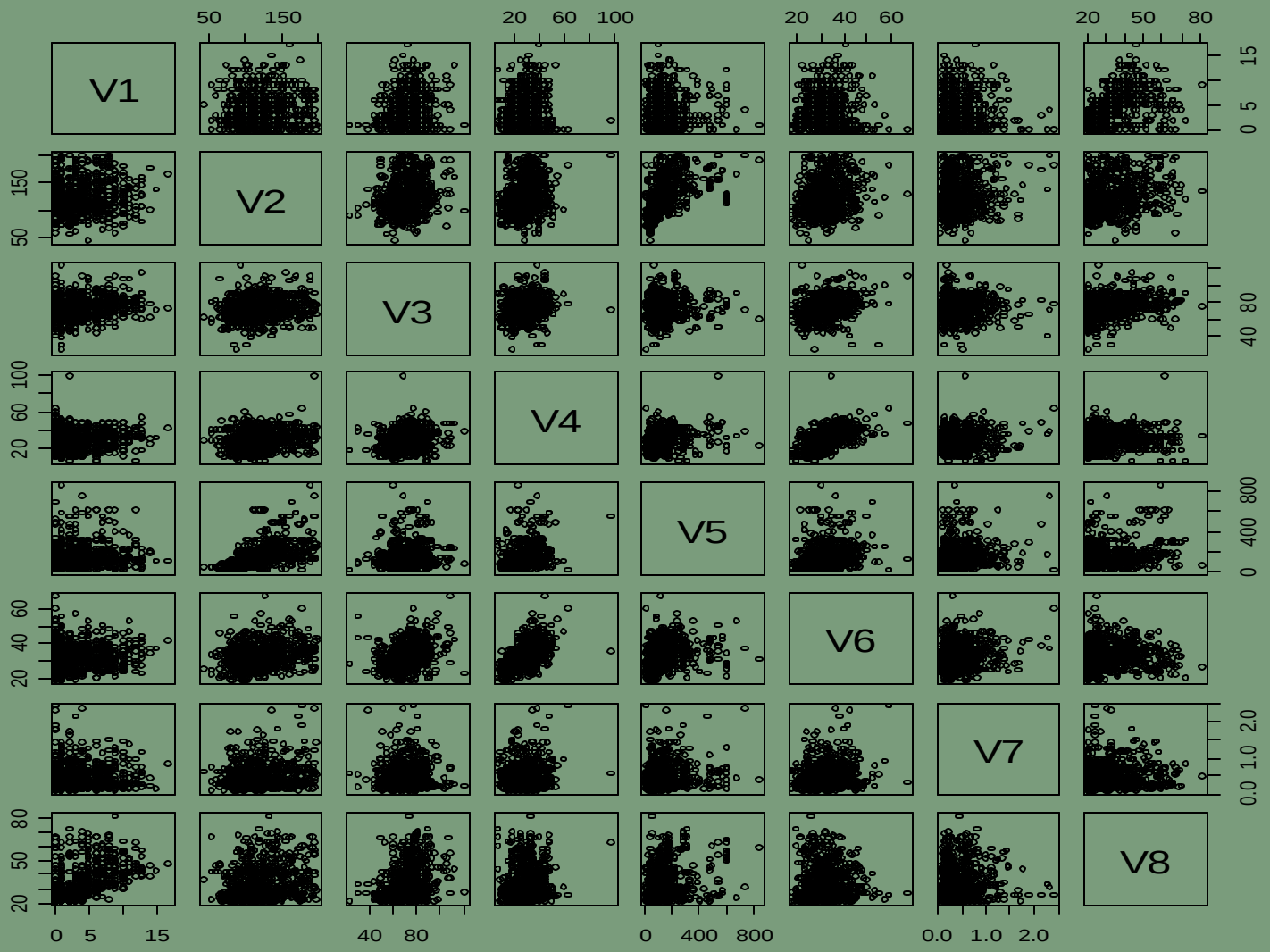




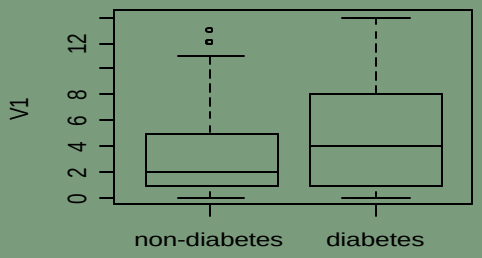


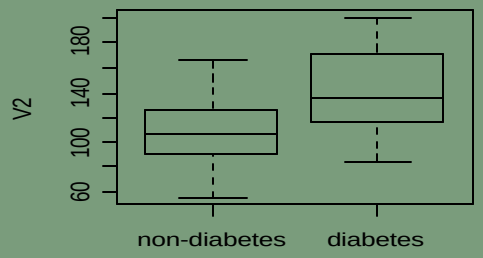


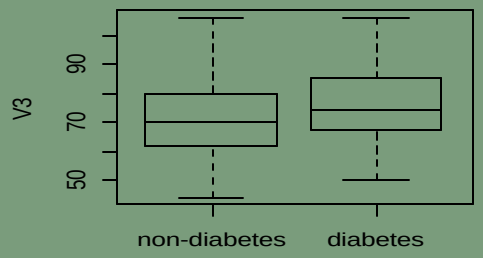
Correlation between Explanatory Variable before and after Imputation

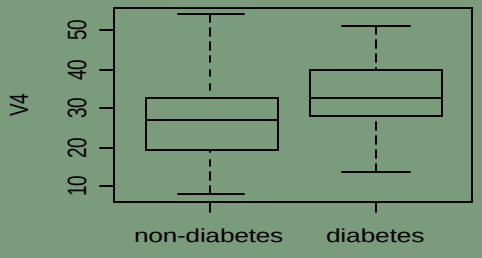


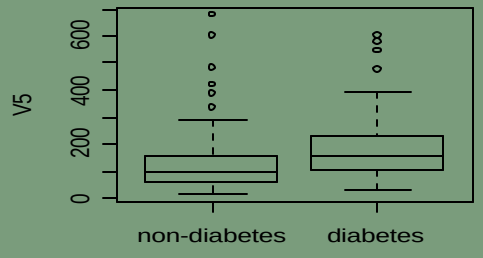
Diabetes explained

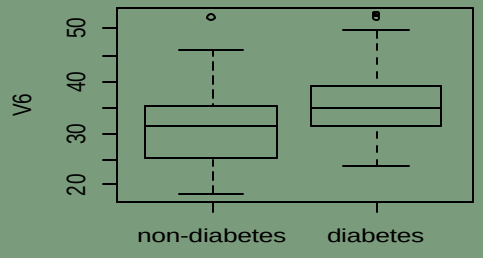


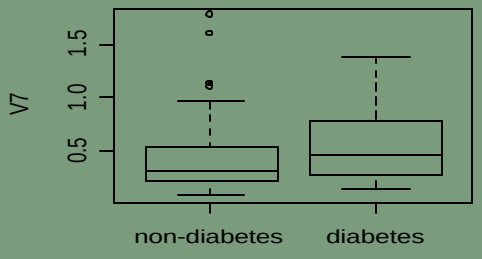


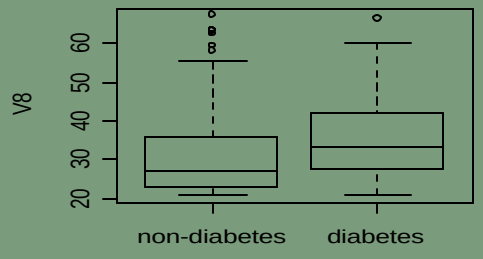












Previous Study

	Error Rate			Error Rate	
Algorithm	Train	Test	Algorithm	Train	Test
LogDisc	0.219	0.223	Bayes	0.239	0.262
Dipol92	0.22	0.224	C4.5	0.131	0.27
Discrim	0.22	0.225	IndCart	0.079	0.271
Smart	0.177	0.232	BayTree	0.008	0.271
Radial	0.218	0.243	LVQ	0.101	0.272
Itrule	0.223	0.245	Kohonen	0.134	0.273
BackProp	0.198	0.248	Ac2	0	0.276
Cal5	0.232	0.25	NewId	0	0.289
Cart	0.227	0.255	Cn2	0.01	0.289
Castle	0.26	0.258	Alloc80	0.288	0.301
QuaDisc	0.237	0.262	KNN	0	0.324
Default	0.35	0.35			

Classification Tree Method

A classification tree classify input space c into classes

At node *m*, the splits are of the form $x_m < t$ versus $x_m > t$

Suppose there are *K* classes and the probability distribution of the tree is $p_{mj}, j = 1.....K$

We define the entropy of the tree at node *m* by:

$$\sum_{j \neq k} p_{mj} p_{mk} = 1 - \sum_{k} p_{mk}^2$$

This is the impurity measure of this node.

At each node, we choose the split that most reduces the average impurity.

p_{mk} can be estimated by $\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I\left(x_m = k\right)$

The tree construction continues until the number of cases reaching each leaf is smaller than 10 or the entropy of each leave is less than 1% of that of the root.

To prevent overfit, we introduce the cost-complexity pruning of the tree.

let R_m denote the entropy of the tree. The size of the tree is the number of leaves.

The complexity measure is then:

$$R_a = R + a\,size$$

a is called complexity parameter (cp)

Thus we can find the optimal trees by a sequence of snip operations on the current tree as we increase the a.

We use 10 time cross validation to choose the degree of pruning. We can find the cross-validation error versus a on the validation set. We look for the value that minimizes the error.

R Algorithm

Step 1: Randomly select 200 obs. from the data set as the training set

Step 2: Fit a classification tree model to the training set

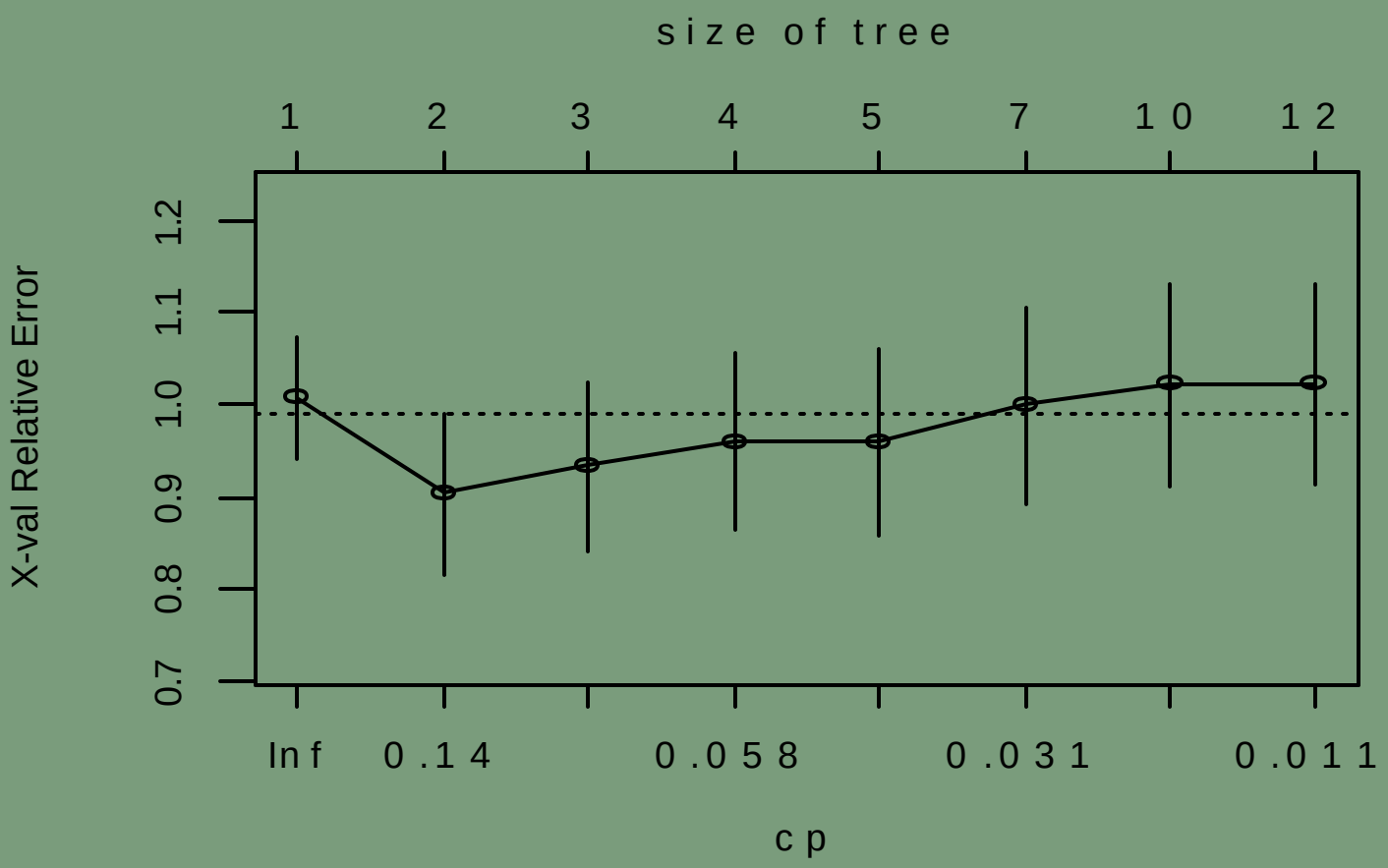
Step 3: Calculate the training error and testing error of the model

Step 4: Repeat above steps 10 times

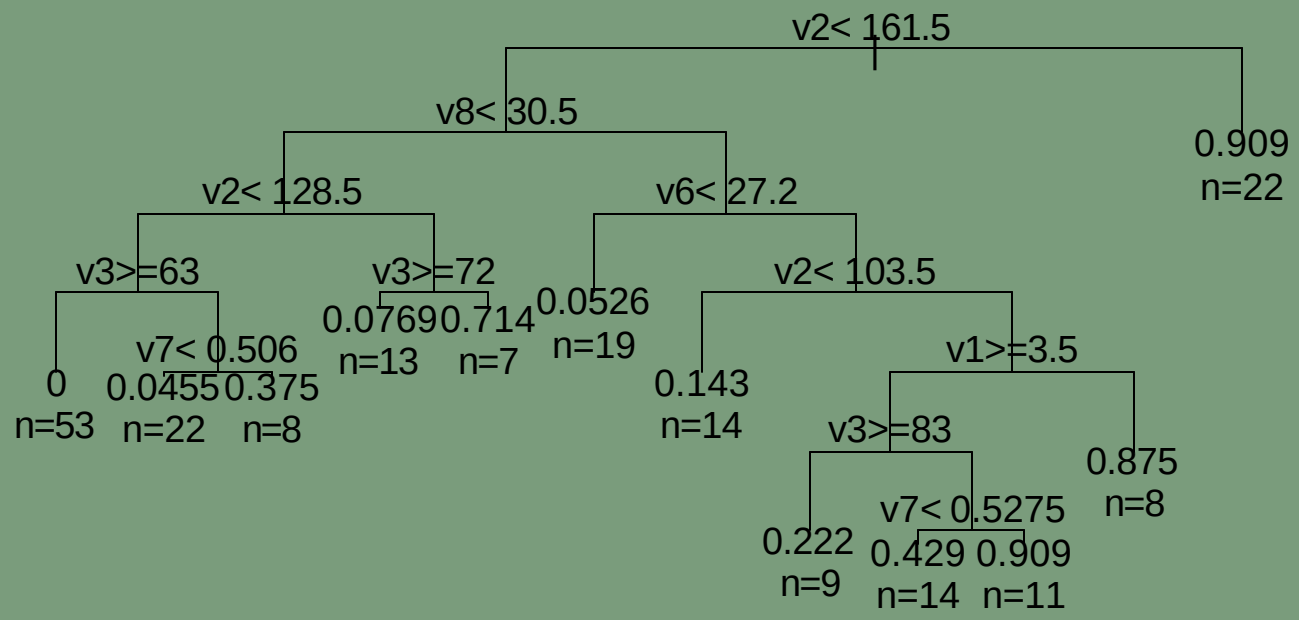
Detail for step 2

rpart(class~v1+v2+v3+v4+v5+v6+v7+v8,diabetes.train,parms=information)
printcp(tree)

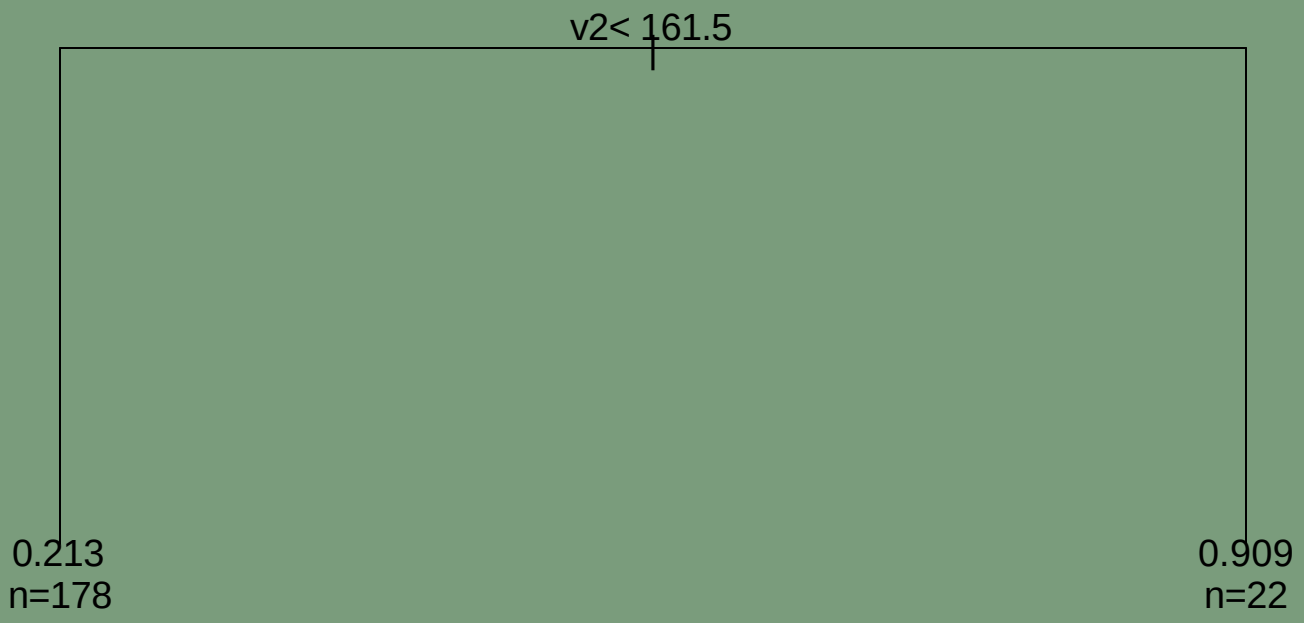
	CP	nsplit	rel error	xerror	xstd
1	0.230067	0	1	1.00729	0.065959
2	0.080424	1	0.76993	0.90268	0.0874
3	0.063554	2	0.68951	0.93321	0.091911
4	0.052181	3	0.62595	0.95973	0.095838
5	0.03485	4	0.57377	0.96028	0.101262
6	0.027123	6	0.50407	0.99911	0.108026
7	0.011871	9	0.4227	1.02195	0.109469
8	0.01	11	0.39896	1.02209	0.109403


size of tree
1
2
3
4
5
7
10
12
X-val Relative Error
0.7
0.8
0.9
1.0
1.1
1.2
Inf
0.14
0.058
0.031
0.011
cp

```
plot(tree, uniform=FALSE, compress=F,minbranch=3,margin=0.14)
text(tree,digits=3,use.n=T)
```



```
tree.pruned<-prune(tree,cp=0.14)
plot(tree.pruned, uniform=FALSE, compress=F,minbranch=3,margin=0.1)
text(tree.pruned,digits=3,use.n=T)
```



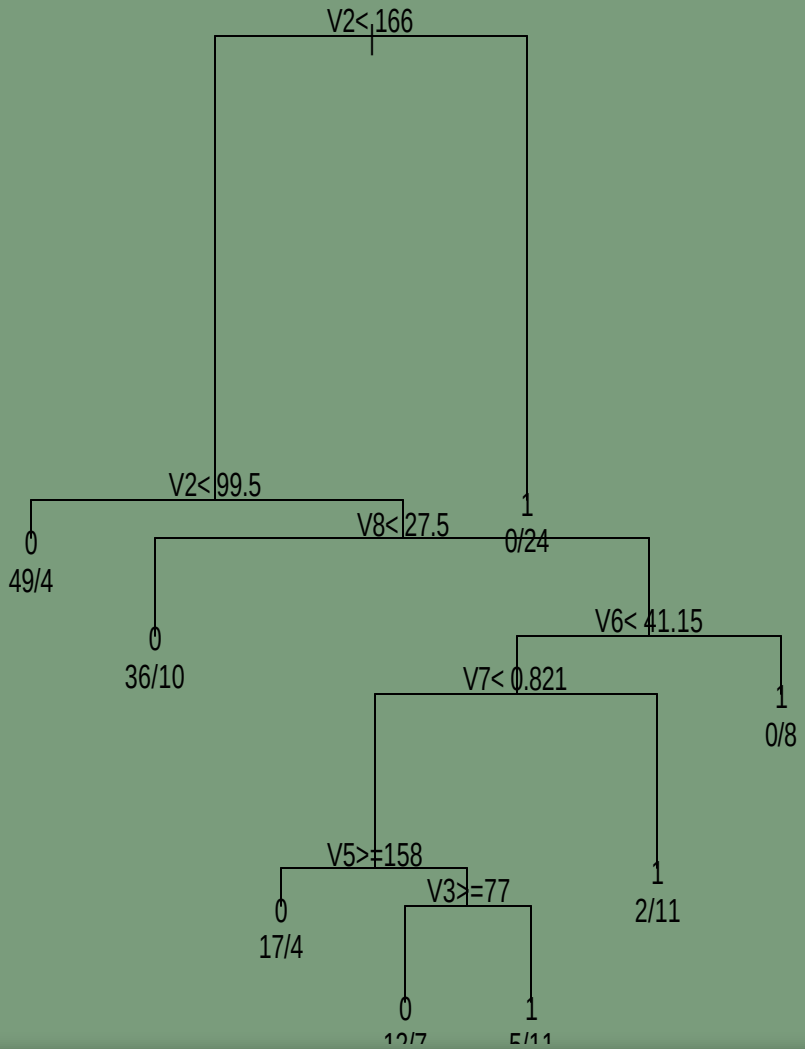
Next, we obtained the predicted probablity of getting diabetes for each observation in the training and testing data.
If the probability > 0.5, then the predicted class is 1.

The training error rate is: # of missclassification/200
The testing error rate is: # of missclassification/568

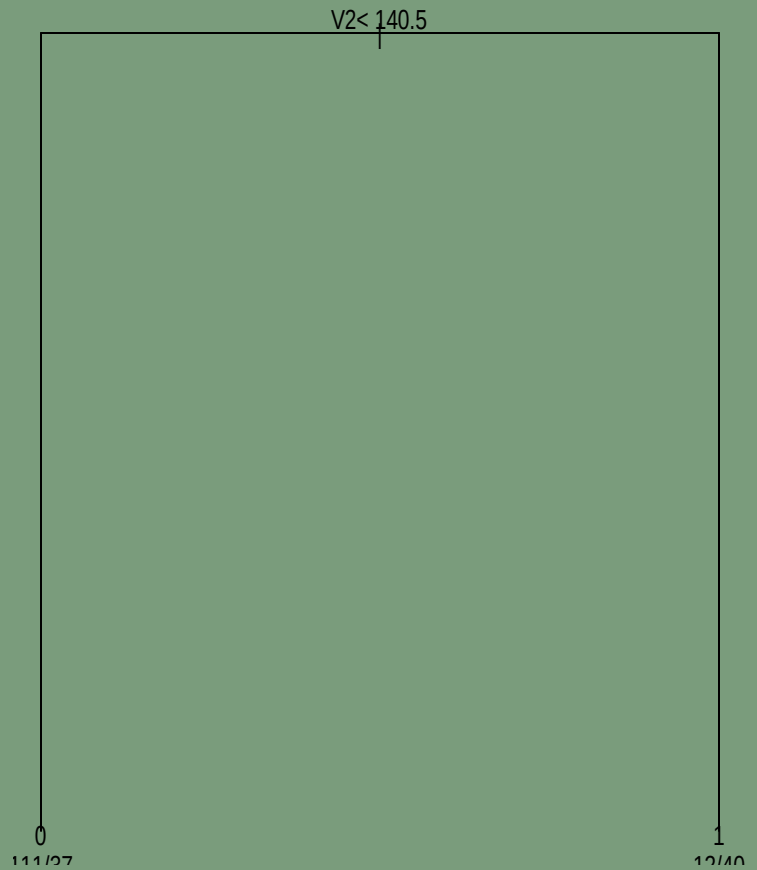
Results

The 10 trees for the 10 repeats

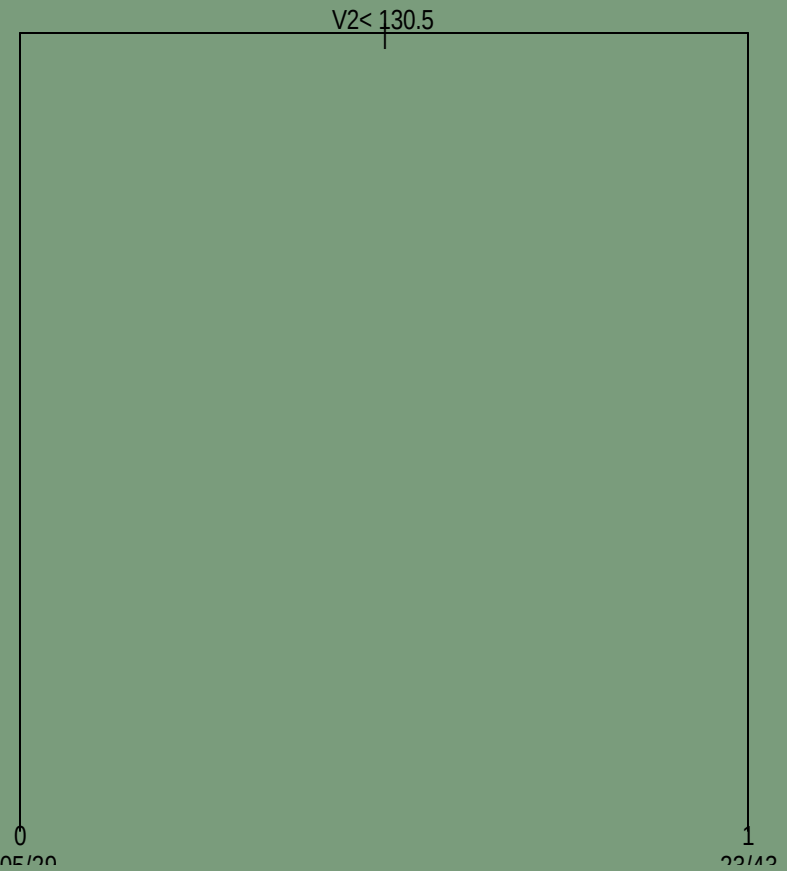
Tree One



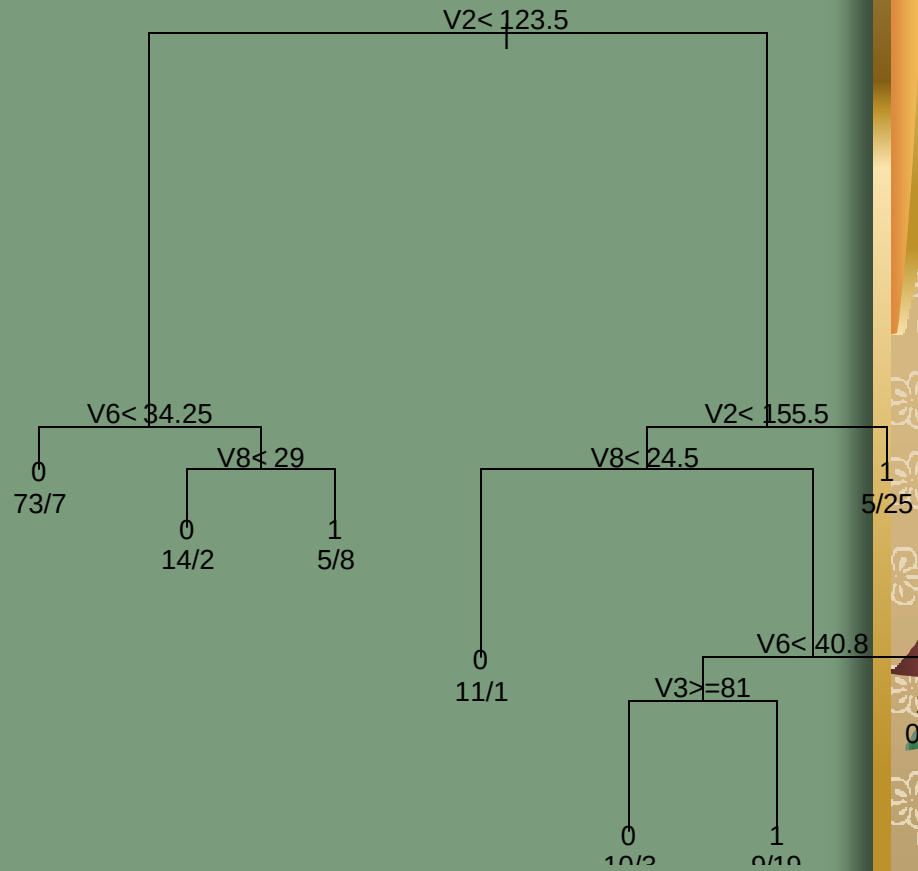
Tree Two



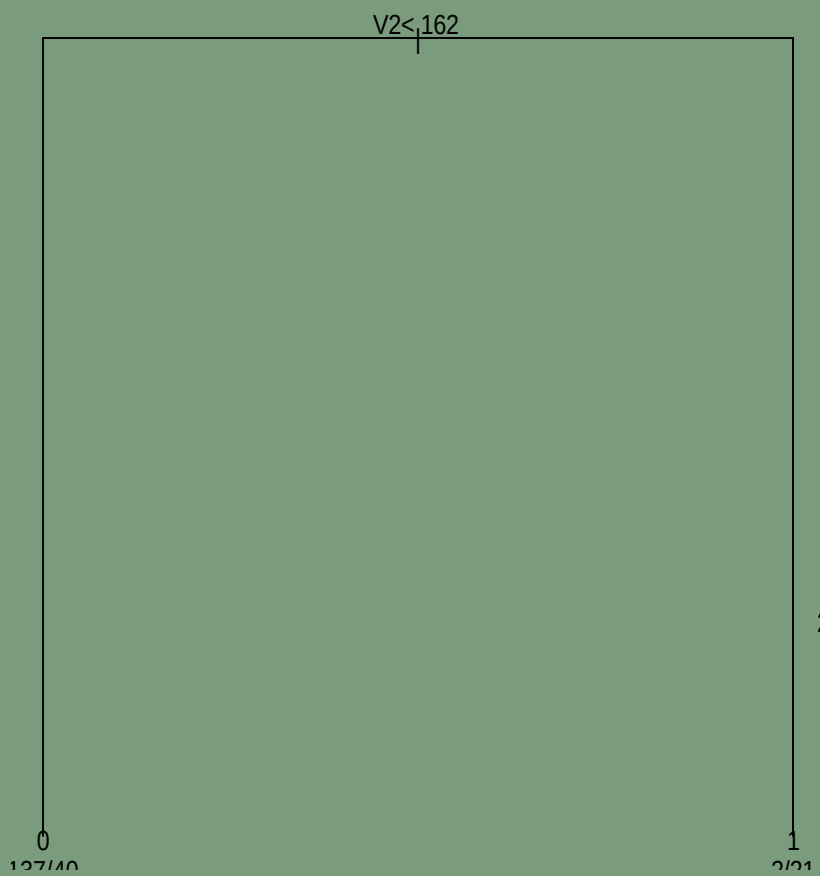
Tree Three



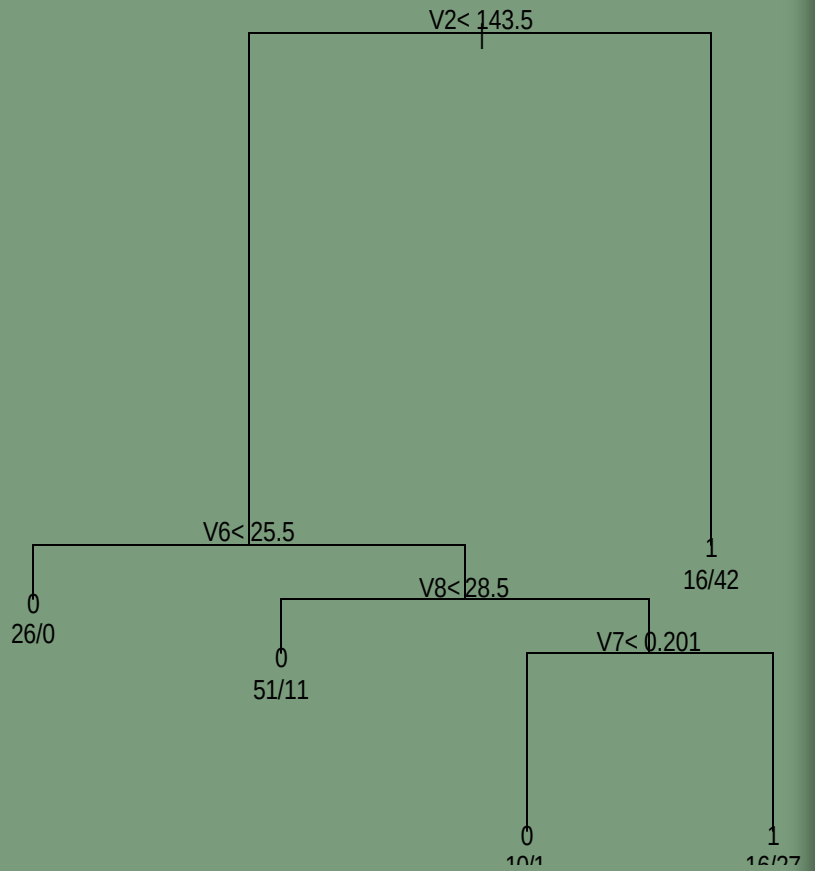
Tree Four



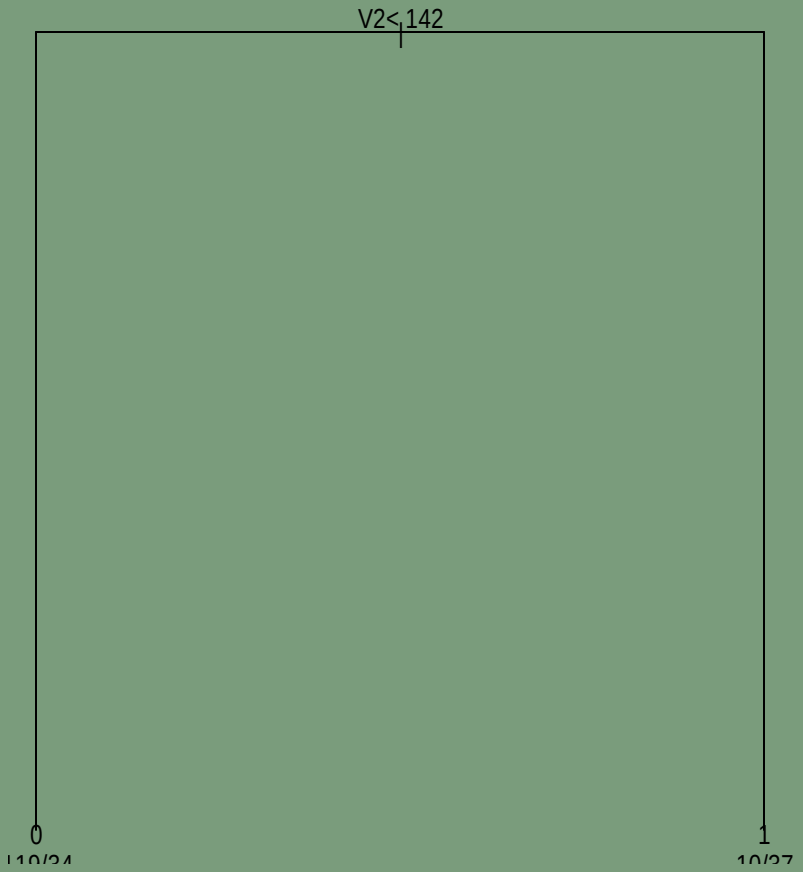
Tree Five



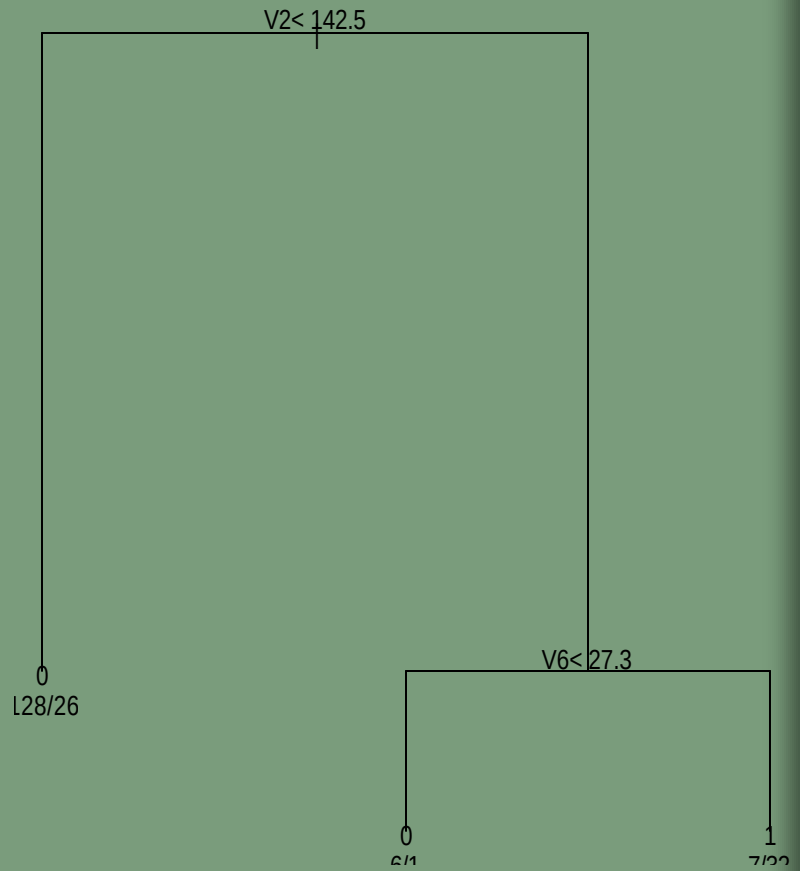
Tree Six



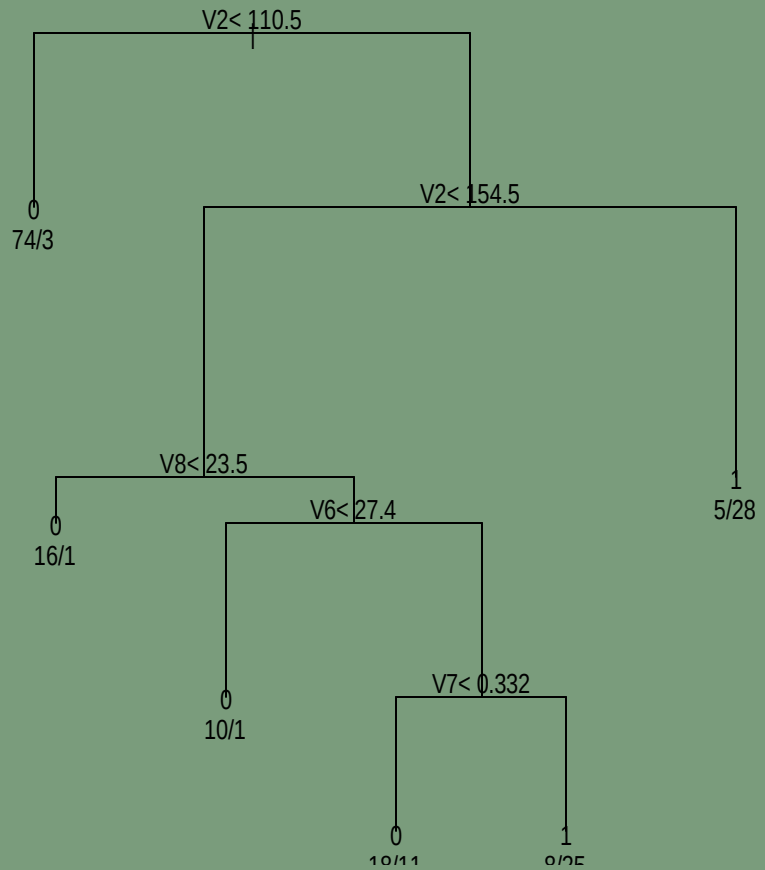
Tree Seven



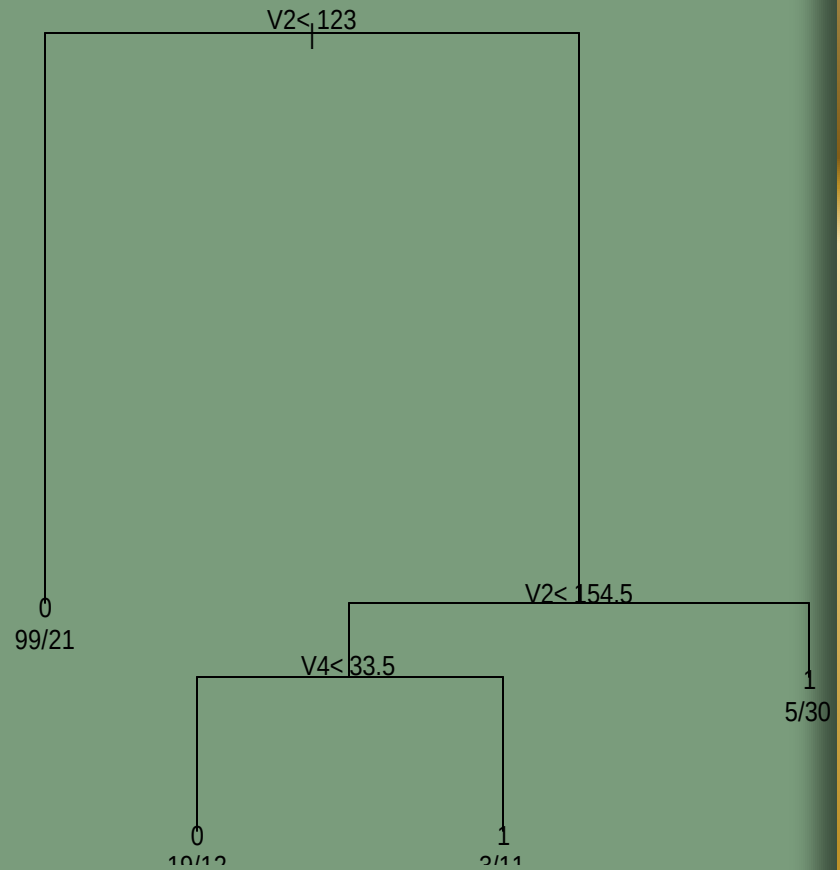
Tree Eight



Tree Nine

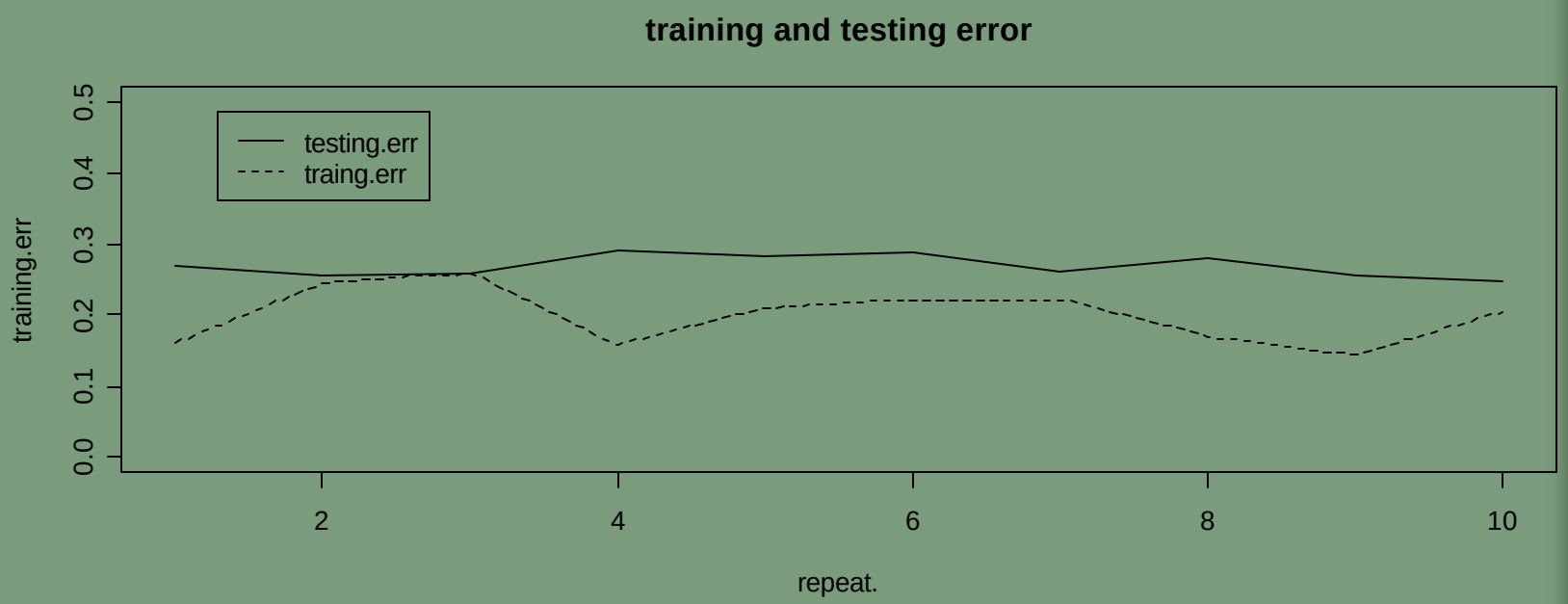


Tree Ten



Error Rate	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
Training Error	0.15994	0.24576	0.25920	0.15936	0.20928
Testing Error	0.26772	0.25555	0.26028	0.29070	0.28327

Error Rate	Tree 6	Tree 7	Tree 8	Tree 9	Tree 10
Training Error	0.22080	0.22080	0.17088	0.14496	0.20544
Testing Error	0.28868	0.26231	0.27989	0.25690	0.24676


training and testing error
testing.err
traing.err
training.err
0.0
0.1
0.2
0.3
0.4
0.5
2
4
6
8
10
repeat.

Logistic Regression

Suppose that Y_i (i=1,…,n) are n independent B(1,p_i) RVs. We let

$$h_i = \log\left(\frac{p_i}{1-p_i}\right) = b_0 + x_i^T b$$

where x_i (i=1,…,n) is some explanatory variable.

Logistic Regression

The model is equivalent to fitting ~ B(1, p_i), where

$$p_i = \frac{e^{b_0 + x_i^T b}}{1 + e^{b_0 + x_i^T b}}$$

This is a generalized linear model for binomial data with a logit link function and $m_i = 1$. We can fit using IWLS (or Fisher scoring)

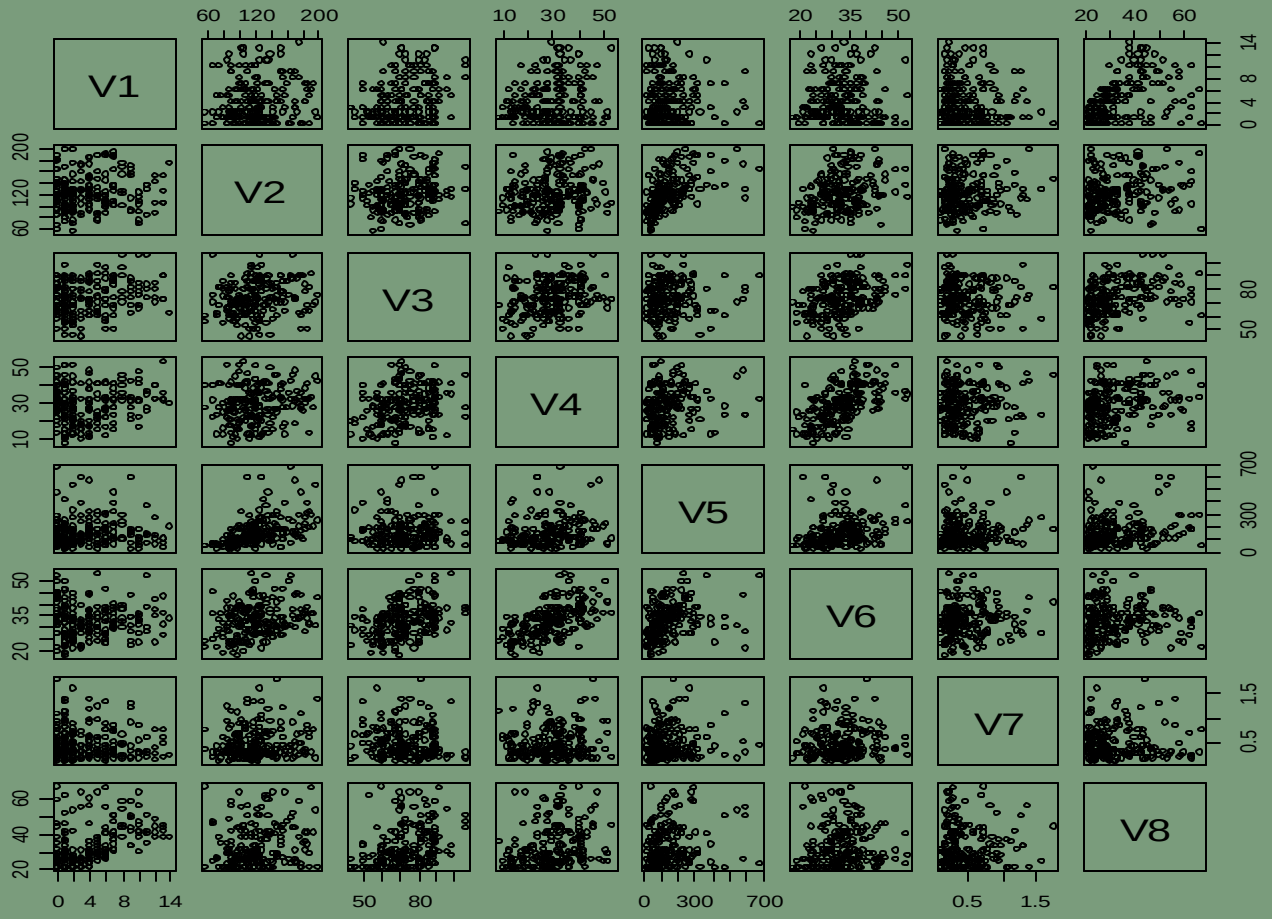
Logistic Regression

Implementation:

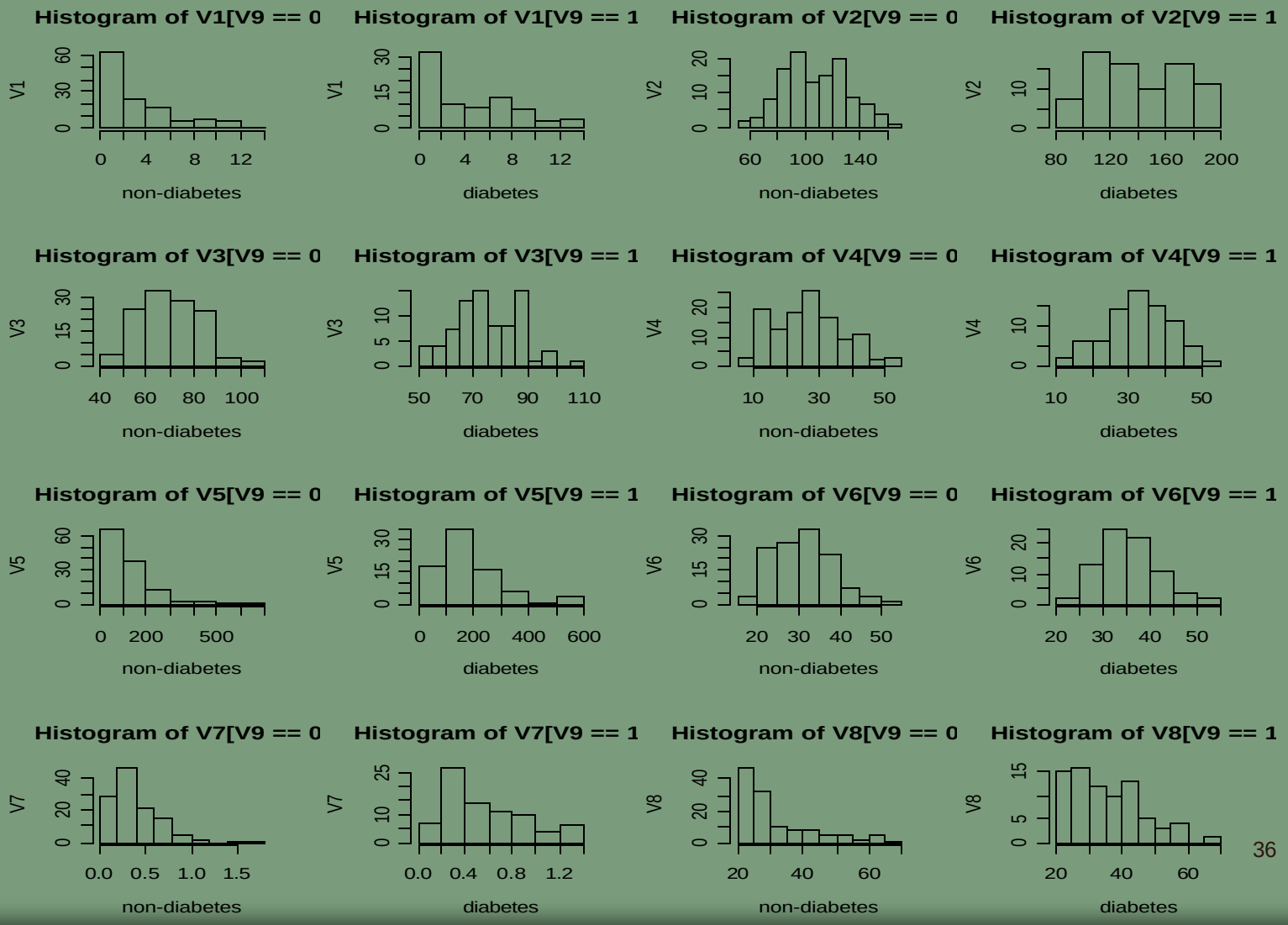
There are totally 768 observations. We randomly select 200 observations for the training set and put the remaining 568 observations into the testing set. We repeat the random split 10 times and implement the classification on each split for training error and testing error. Finally we calculate the mean and std. for those errors.

Logistic Regression

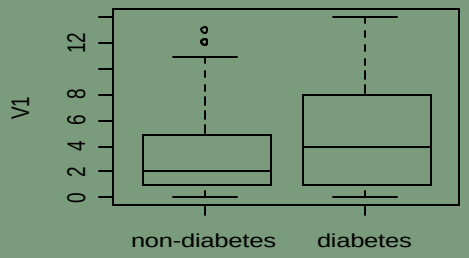
Model Selection:

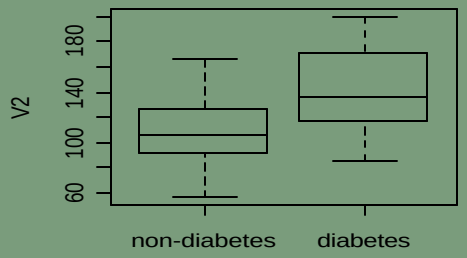


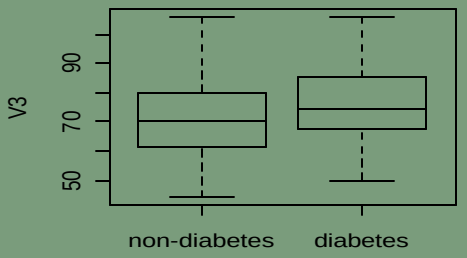
Logistic Regression

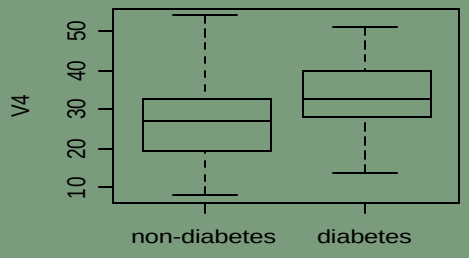


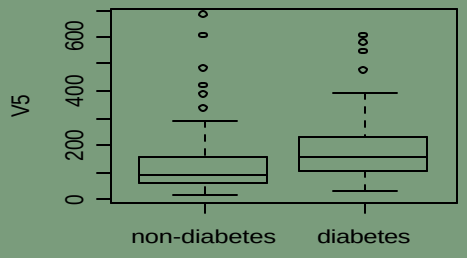
Logistic Regression

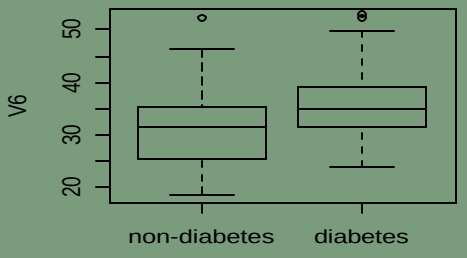


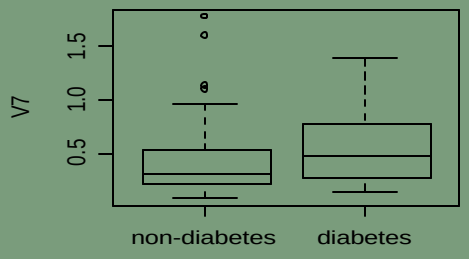


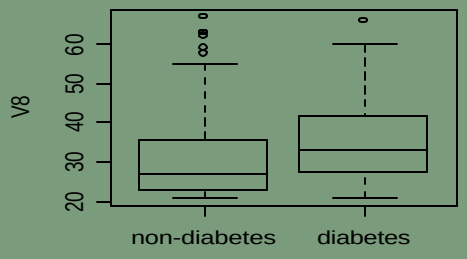












Logistic Regression —Our Final Model

- glm(formula = response ~ npreg + glu + bmi + ped, family = binomial)
- Null deviance: 268.37 on 199 degrees of freedom
- Residual deviance: 182.90 on 195 degrees of freedom
- AIC: 192.90
- Number of Fisher Scoring iterations: 5

Coefficient	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-10.0177	1.500661	-6.676	0.000000
npreg	0.126211	0.052323	2.412	0.015860
glu	0.040658	0.007344	5.536	0.000000
bmi	0.09328	0.030352	3.073	0.002120
ped	1.906364	0.594621	3.206	0.001350

Logistic Regression

Error	1	2	3	4	5	6	7	8	9	10	Ave.	Std.
Training	0.2093	0.2458	0.2496	0.24	0.215	0.2803	0.2074	0.1805	0.2208	0.24	0.229	0.028
Testing	0.2339	0.2238	0.2339	0.2217	0.3644	0.2075	0.2393	0.2481	0.2305	0.2339	0.244	0.0438

Logistic Regression

Support Vector Machine

Our training data consists of N pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, with $x_i \in \Re^p$ and $y_i \in \{-1,1\}$. Define a hyper plane by

$$\{x : f(x) = x^T b + b_0\}$$

where b is a unit vector: $\|b\|=1$.

Support Vector Machine

A classification rule induced by f(x) is :

$$G(x) = sign[x^T b + b_0]$$

For separable case, the hyperplane that creates the biggest margin between the training points for class -1 and 1 is the following optimization problem:

$$\max_{b, b_0, \|b\|=1} C \quad \text{subject to} \quad y_i(x_i^T b + b_0) \geq C, i = 1, ..., N$$

Support Vector Machine

which is equivalent to

$$\min_{b,b_0} \| b \| \quad \text{subject to} \quad y_i(x_i^T b + b_0) \geq 1, i = 1, ..., N$$

For non-separable case, we still maximize ||C||, but allow for some points to be on the wrong side of the margin. Define the slack variables $x = (x_1, x_{2,} ....., x_N)$. The support vector classifier is:

$\min \| b \|$ subject to $y_i(x_i^T b + b_0) \geq 1 - x_i \forall i, x_i \geq 0, \sum x_i \leq$ constant

Support Vector Machine

Computationally, it is convenient to be re-expressed in the equivalent form

$$\min \frac{1}{2} \| b \| + g \sum_{i=1}^{N} x_i$$

subject to $y_i (x_i^T b + b_0) \geq 1 - x_i \forall i, x_i \geq 0$

which is equivalent to the optimization problem:

$$\min_{b, b_0} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + l \| b \|^2$$

with $l = 1/(2g)$

Support Vector Machine

Implementation:

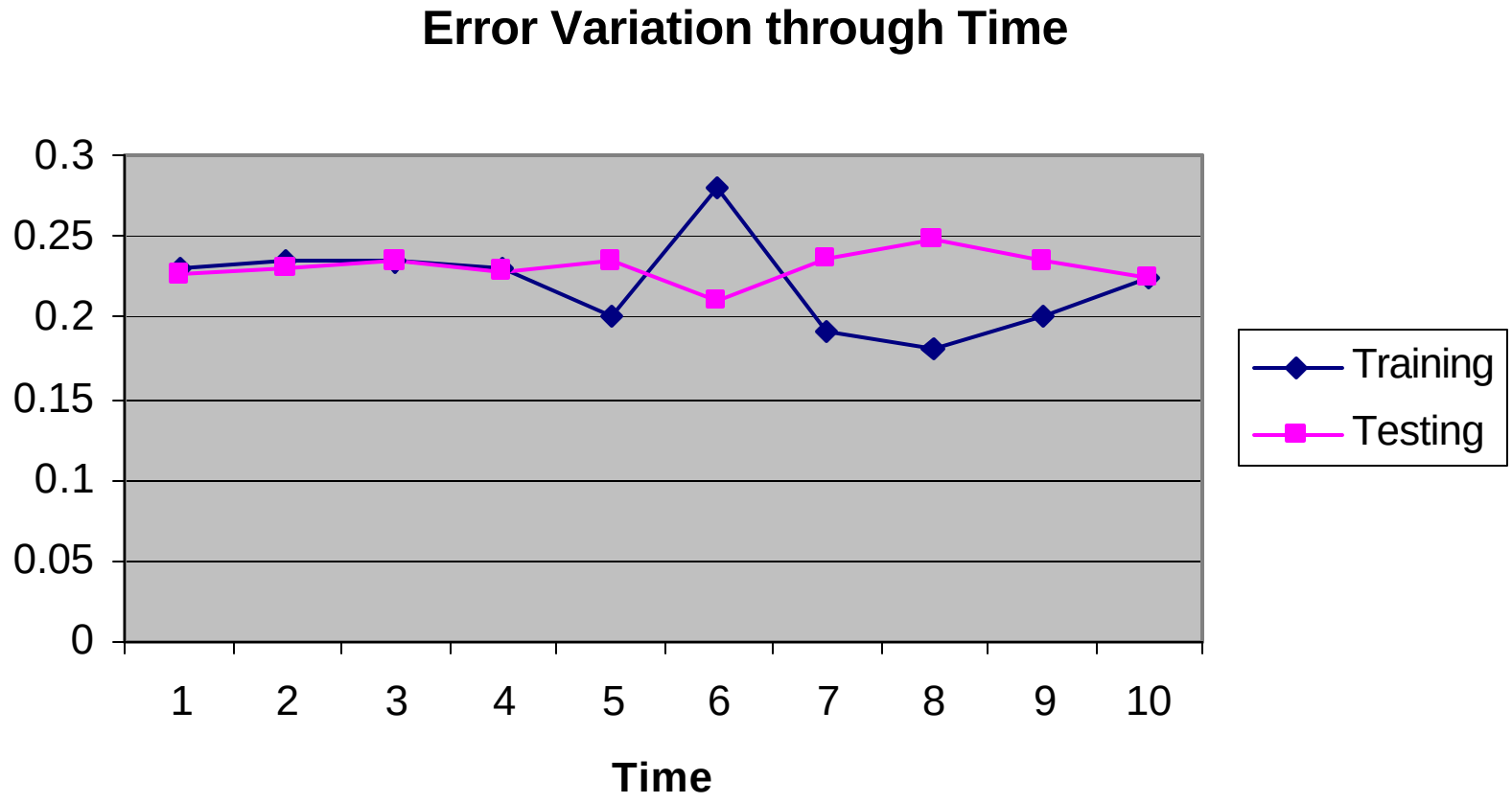
For simplicity, we implement linear support vector machine without standardization of the inputs. We do not conduct model selection for SVM. We used a grid search to choose the tuning parameter C. There are 20 values to choose for C and the 20 values are 0.00125*2^i; i=0, 1, ..., 19.

Therefore, we do not fully take advantage of the predicting potential of the linear support vector machine.

Support Vector Machine

Error	1	2	3	4	5	6	7	8	9	10	Ave.	Std.
Training	0.23	0.235	0.235	0.23	0.2	0.28	0.19	0.18	0.2	0.225	0.221	0.029
Testing	0.227	0.231	0.234	0.229	0.234	0.211	0.236	0.248	0.234	0.225	0.231	0.0094

Support Vector Machine



Relationship between LR & SVM

Loss Function	L(Y, F(X))	Minimizing Function
(-) Binomial Log-likelihood	$\log(1+e^{Yf(X)})$	$f(X)=\log\frac{P(Y=+1\mid X)}{P(Y=-1\mid X)}$
Support Vector Machine	$[1+Yf(X)]_+$	$f(X)=\begin{cases}+1, & \text{if } P(Y=+1\mid X)\geq\frac{1}{2}\\ -1, & \text{Otherwise}\end{cases}$

Relationship between LR & SVM

Linear logistic regression uses

$$\hat{f}(x) = x^T b + b_0$$

to approximate:

$$f(X) = \log \frac{P(Y = +1 \mid X)}{P(Y = -1 \mid X)}$$

Relationship between LR & SVM

while linear support vector machine uses

$$\hat{G}(x) = \text{sign}[\, x^T b + b_0 \,]$$

to estimate

$$f(X) = \begin{cases} +1, & \text{if} \quad P(Y = +1 \mid X) \geq \dfrac{1}{2} \\ -1, & \text{Otherwise} \end{cases}$$

Relationship between LR & SVM

If we use 0-1 loss for misclassification in standard situation, the Bayes rule minimizing the expected loss is:

$$f_B(x) = \begin{cases} +1, & \text{if } \dfrac{p(Y=+1|X)}{1-p(Y=+1|X)} > 1 \\ -1, & \text{Otherwise} \end{cases}$$

Relationship between LR & SVM

In logistic regression, we use MLE, which is consistent and asymptotically efficient. If we choose logit=0 or p=0.5 as the threshold, the result will converge to the Bayes rule asymptotically.

Relationship between LR & SVM

According to Lin(1999), ***if the reproducing kernel Hilbert space is rich enough, the solution to the non-linear support vector machine approaches the Bayes rule as the sample size tends to infinity.*** Hence, we can not guarantee that linear support vector machine (use linear kernel) is a good approximation of the Bayes rule under the standard situation.

Relationship between LR & SVM

In non-standard situation, the Bayes rule minimizing the expectet loss becomes:

$$f_B(x) = \begin{cases} +1, & \text{if } \dfrac{p(Y=+1|X)}{1-p(Y=+1|X)} > \dfrac{c^+}{c^-}\dfrac{p_s^+}{p_s^-}\dfrac{p^-}{p^+} \\ -1, & \text{Otherwise} \end{cases}$$

In this scenario, all we should do to logistic regression is to simply change the threshold of the classification problem and all the conclusions about logistic regression hold in this situation.

Relationship between LR & SVM

While for support vector machine, we have to modify the loss function and redo the quadratic programming procedure to obtain the estimate of the minimizing function. In this non-standard situation, the loss function is changed to:

$$\frac{1}{N}\sum_{i=1}^{N} L(y_i)[(1-y_i f(x_i))_+]$$

where $L(-1) = c^+ p_s^+ p^-$ and $L(1) = c^- p_s^- p^+$. The penalty term remains unchanged. The solution to this regularization problem also tends to the Bayes rule under certain conditions (see Lin (2002) Support Vector Machines for Classification in Nonstandard Situation).

Relationship between LR & SVM

The reason why we have to go through all the steps for SVM is that SVM does not give the whole picture of probability. We loss most of the information when the estimated boundary approaches the step function of the Bayes rule [For linear support vector machine we can retrieve the probability under some conditions (See John Platt (1999) Probalistic Outputs for Support vector machines and comparison to Regularized Likelihood Method)}

Summary of the three methods

Error	1	2	3	4	5	6	7	8	9	10	Ave.	Std.
LR-Training	0.209	0.246	0.250	0.240	0.215	0.280	0.207	0.181	0.221	0.240	0.229	0.028
LR-Testing	0.234	0.224	0.234	0.222	0.364	0.208	0.239	0.248	0.231	0.234	0.244	0.044
CT-Training	0.160	0.246	0.259	0.159	0.209	0.221	0.221	0.171	0.145	0.205	0.200	0.039
CT-Testing	0.268	0.256	0.260	0.291	0.283	0.289	0.262	0.280	0.257	0.247	0.269	0.015
SVM-Training	0.230	0.235	0.235	0.230	0.200	0.280	0.190	0.180	0.200	0.225	0.221	0.029
SVM-Testing	0.227	0.231	0.234	0.229	0.234	0.211	0.236	0.248	0.234	0.225	0.231	0.009

Summary of the three methods