In [133]:

```
library(readr)
```

In [134]:

```
wine <- read_csv("C:/Users/admin/Desktop/wineR3_4.csv")
```

```
Parsed with column specification:
cols(
  Flavanoids = col_double(),
  Hue = col_double(),
  OD = col_double(),
  Proline = col_double(),
  Class = col_character()
)
```

In [135]:

```
wine$Class <- as.factor(wine$Class)                              # Convert character column to factor
```

In [136]:

```
library(e1071)
```

In [130]:

```
library(GGally)
```

In [137]:

```
library(ggplot2)
```

In [138]:

```
str(wine)
```

```
tibble [178 x 5] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Flavanoids: num [1:178] 3.06 2.76 3.24 3.49 2.69 3.39 2.52 2.51 2.98 3.15 ...
 $ Hue       : num [1:178] 1.04 1.05 1.03 0.86 1.04 1.05 1.02 1.06 1.08 1.01 ...
 $ OD        : num [1:178] 3.92 3.4 3.17 3.45 2.93 2.85 3.58 3.58 2.85 3.55 ...
 $ Proline   : num [1:178] 1065 1050 1185 1480 735 ...
 $ Class     : Factor w/ 3 levels "Low","Middle",..: 3 3 3 3 3 3 3 3 3 3 ...
 - attr(*, "spec")=
  .. cols(
  ..   Flavanoids = col_double(),
  ..   Hue = col_double(),
  ..   OD = col_double(),
  ..   Proline = col_double(),
  ..   Class = col_character()
  .. )
```

In [139]:

```
head(wine,5)
```

Flavanoids	Hue	OD	Proline	Class
3.06	1.04	3.92	1065	Premium
2.76	1.05	3.40	1050	Premium
3.24	1.03	3.17	1185	Premium
3.49	0.86	3.45	1480	Premium
2.69	1.04	2.93	735	Premium

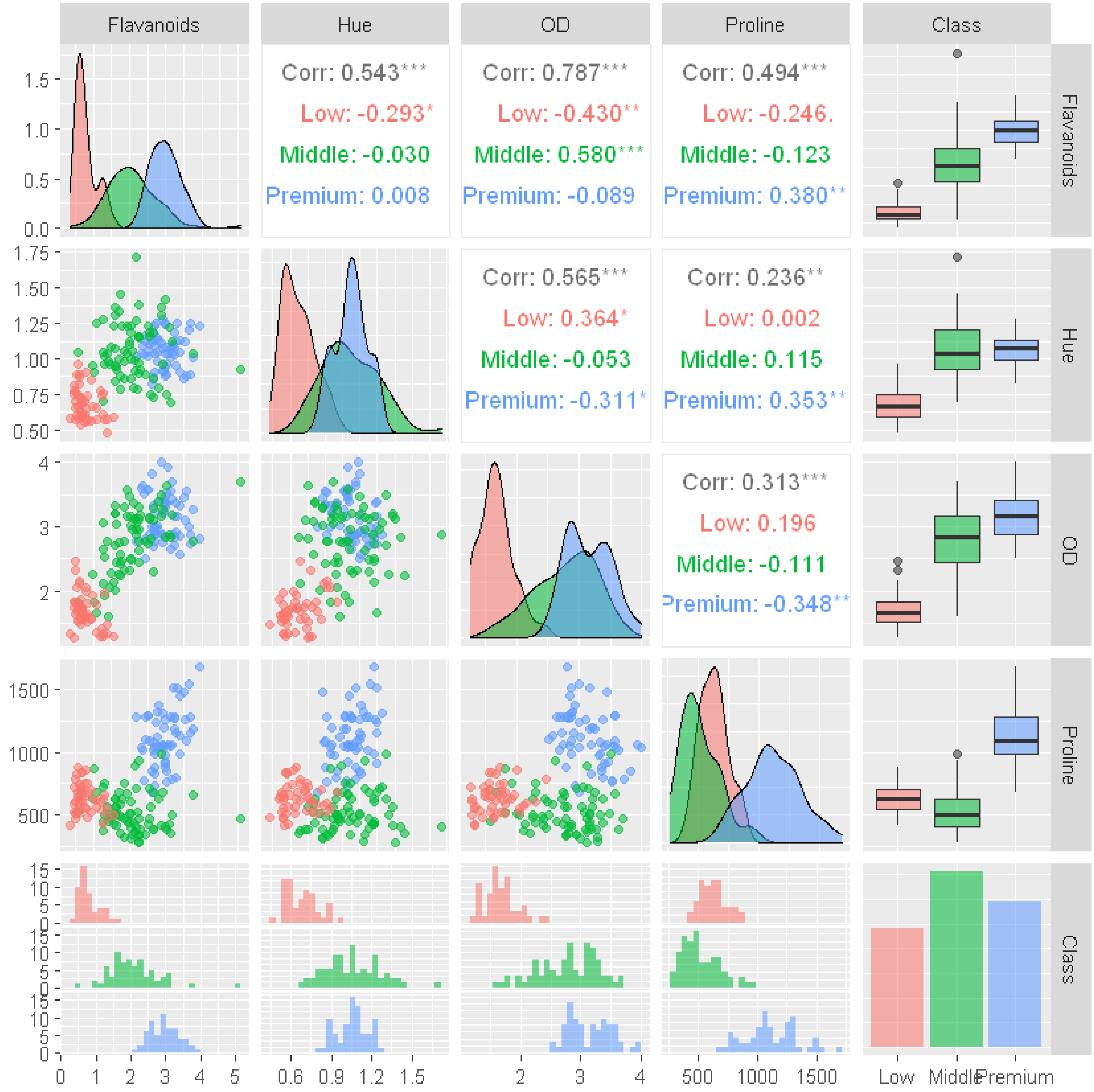
In [140]:

```
# Create SVM Model

#RADIAL

svm_model <- svm(Class ~ ., data=wine, type = "C-classification",
                 kernel="radial") #linear/polynomial/sigmoid
```

In [141]:

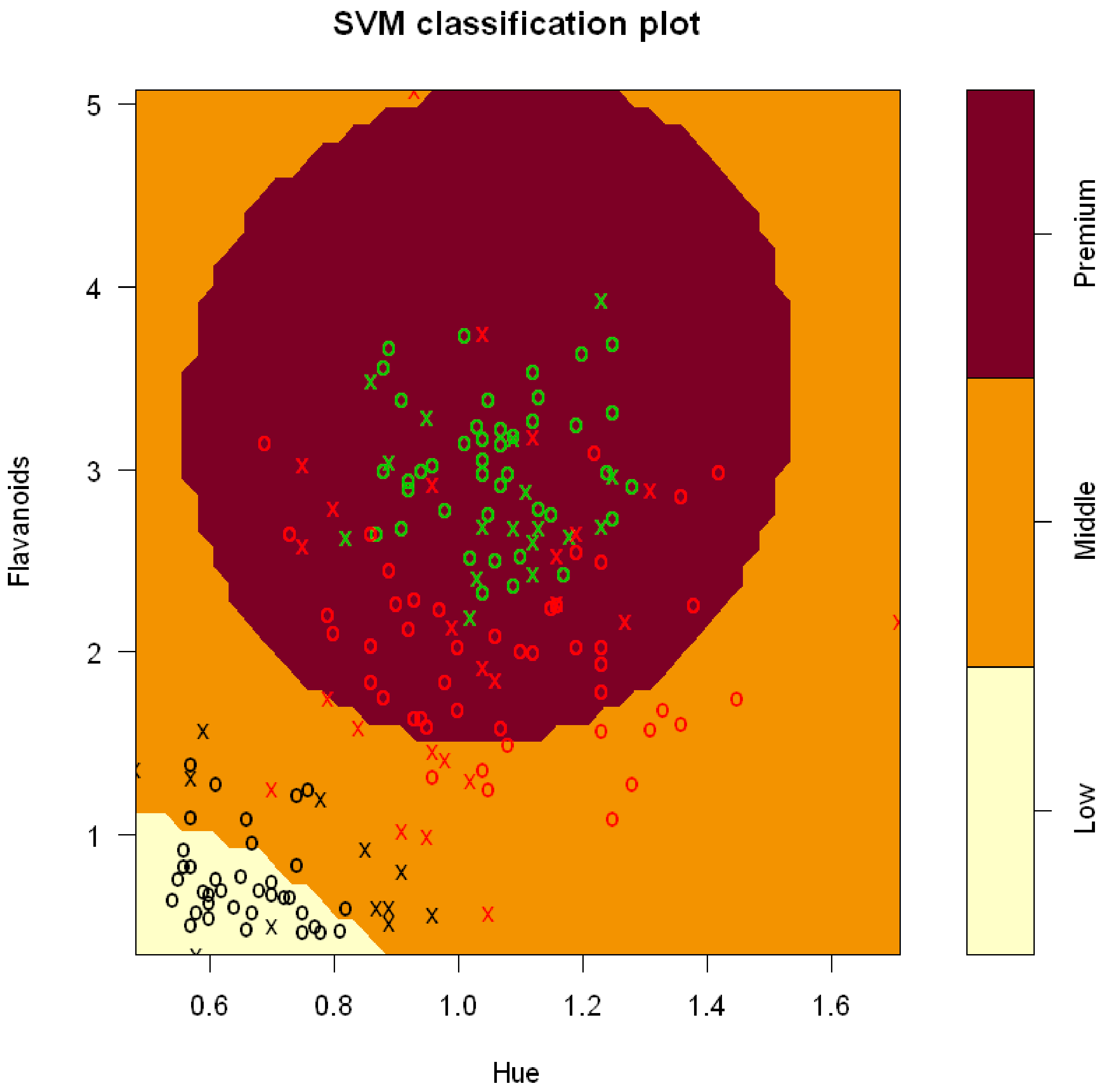
```
ggpairs(wine, ggplot2::aes(colour = Class, alpha = 0.4))
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



In [142]:

```
plot(svm_model, data=wine,
    Flavanoids~Hue,
    slice = list(OD=3, Proline=1000)

    )
```



In [143]:

```
#Predict each Species
#Confusion matrix and missclasscation Error

pred = predict(svm_model,wine)
tab = table(Predicted=pred, Actual = wine$Class)
tab
```

```
          Actual
Predicted Low Middle Premium
  Low      48      2       0
  Middle    0     65       2
  Premium   0      4      57
```

In [144]:

```
1-sum(diag(tab)/sum(tab)) #Missclasification error
```

```
0.0449438202247191
```

In [145]:

```
sum(diag(tab)/sum(tab))  #Accuracy
```

```
0.955056179775281
```

In [146]:

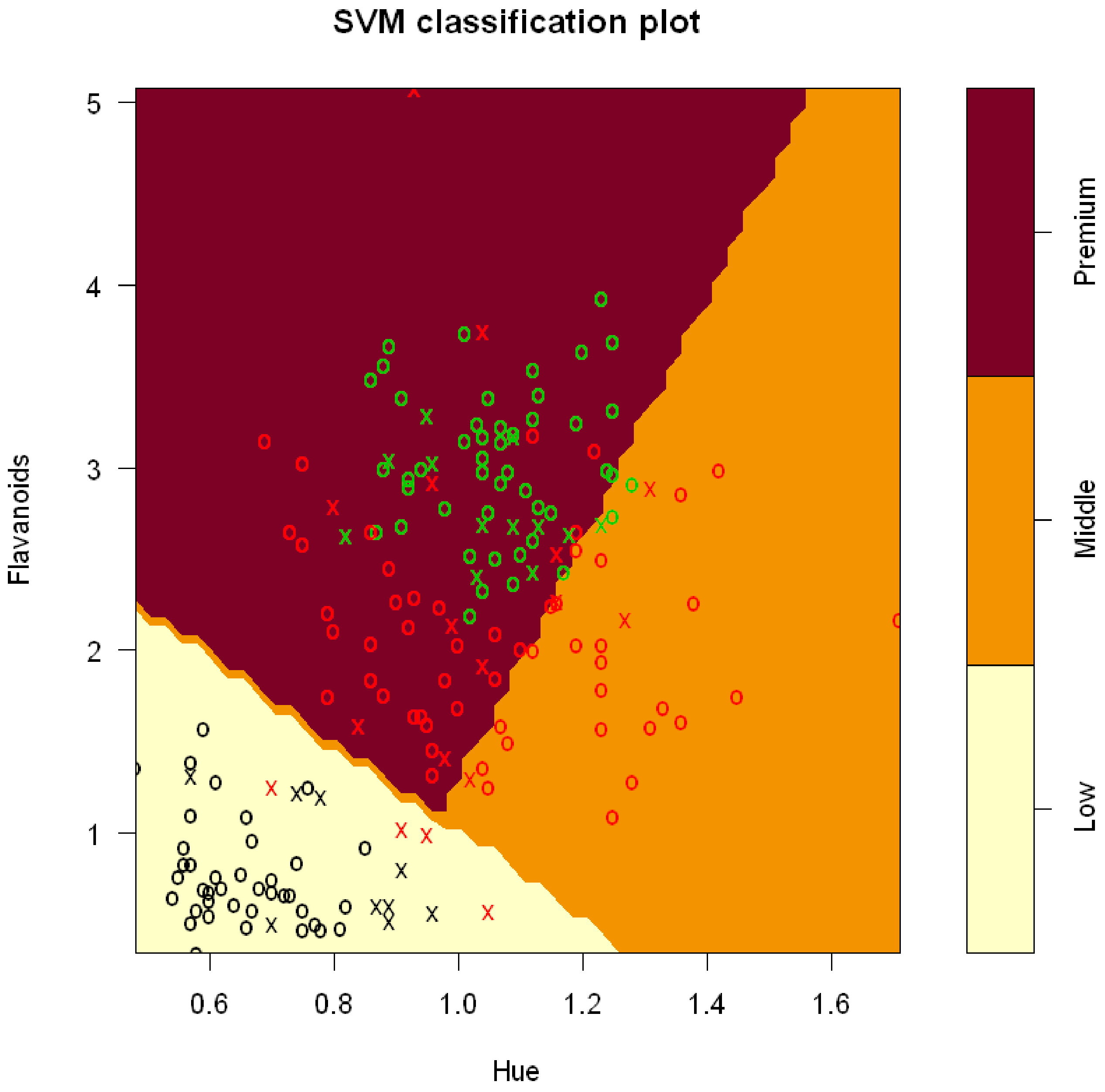
```
#LINEAR
```

In [147]:

```
svm_model <- svm(Class ~ ., data=wine, type = "C-classification",
                 kernel="linear") #linear/polynomial/sigmoid
```

In [148]:

```
plot(svm_model, data=wine,
     Flavanoids~Hue,
     slice = list(OD=2, Proline=1000)

     )
```



In [149]:

```
#Predict each Species
#Confusion matrix and missclassification Error and Accuracy

pred = predict(svm_model,wine)
tab = table(Predicted=pred, Actual = wine$Class)
tab
```

```
        Actual
Predicted Low Middle Premium
  Low      48      3       0
  Middle    0     65       2
  Premium   0      3      57
```

In [150]:

```
1-sum(diag(tab)/sum(tab))
```

```
0.0449438202247191
```

In [151]:

```
sum(diag(tab)/sum(tab))
```

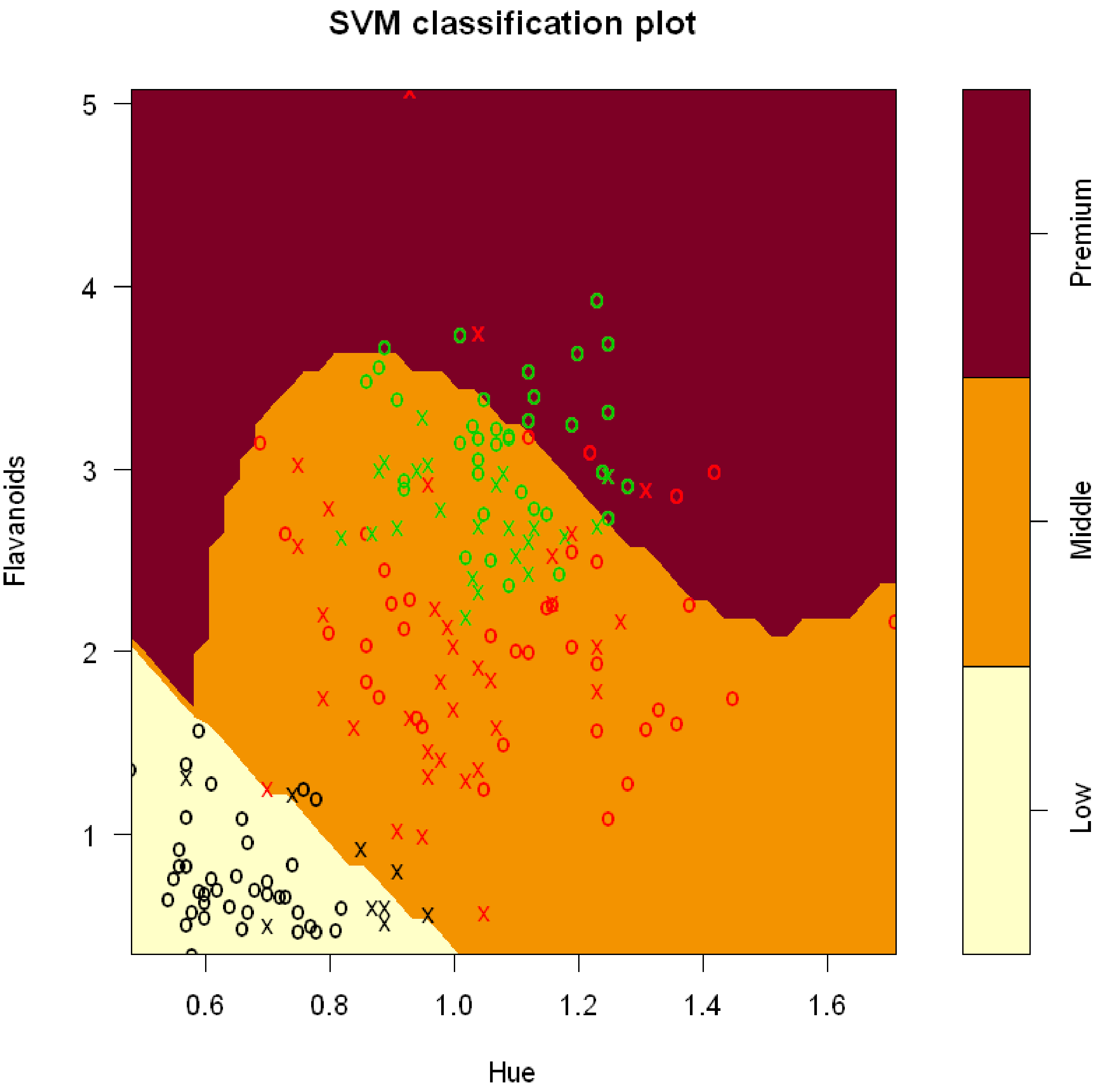
```
0.955056179775281
```

In [152]:

```
#POLYNOMIAL

svm_model <- svm(Class ~ ., data=wine, type = "C-classification",
            kernel="poly") #linear/polynomial/sigmoid
```

In [153]:

```
plot(svm_model, data=wine,
    Flavanoids~Hue,
    slice = list(OD=2, Proline=1000)

    )
```



In [154]:

```
#Predict each Species
#Confusion matrix and missclassification Error and Accuracy

pred = predict(svm_model,wine)
tab = table(Predicted=pred, Actual = wine$Class)
tab
```

```
        Actual
Predicted Low Middle Premium
  Low      47      0       0
  Middle    1     70       5
  Premium   0      1      54
```

In [155]:

```
1-sum(diag(tab)/sum(tab))
```

```
0.0393258426966292
```

In [156]:

```
sum(diag(tab)/sum(tab))
```

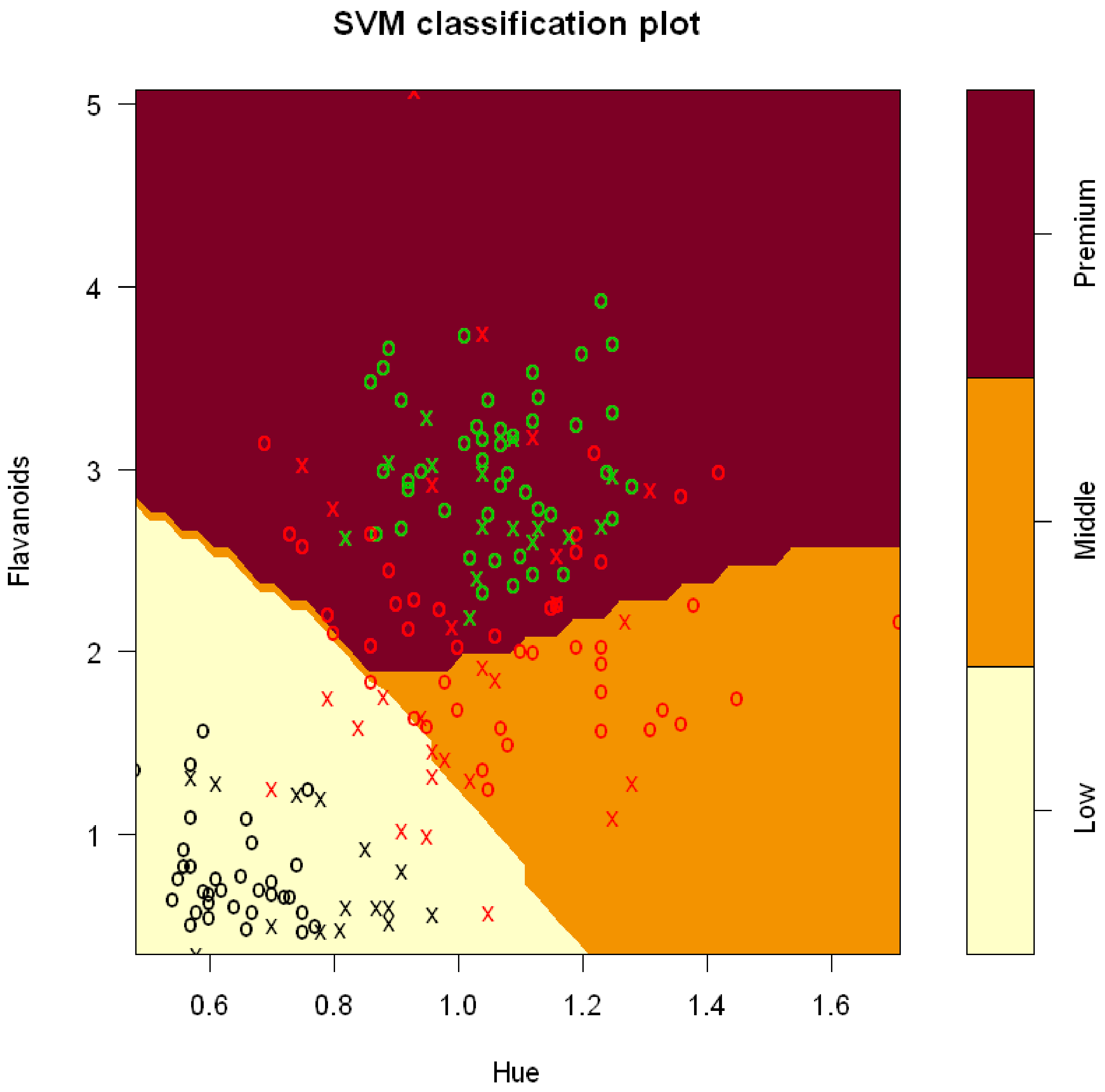
```
0.960674157303371
```

In [157]:

```
#SIGMOID

svm_model <- svm(Class ~ ., data=wine, type = "C-classification",
               kernel="sigmoid") #linear/polynomial/sigmoid
```

In [158]:

```
plot(svm_model, data=wine,
    Flavanoids~Hue,
    slice = list(OD=2, Proline=1000)

    )
```



In [159]:

```
#Predict each Species
#Confusion matrix and missclassification Error and Accuracy

pred = predict(svm_model,wine)
tab = table(Predicted=pred, Actual = wine$Class)
tab
```

```
          Actual
Predicted Low Middle Premium
  Low      48      6       0
  Middle    0     60       2
  Premium   0      5      57
```

In [160]:

```
1-sum(diag(tab)/sum(tab))
```

```
0.0730337078651686
```

In [161]:

```
sum(diag(tab)/sum(tab))
```

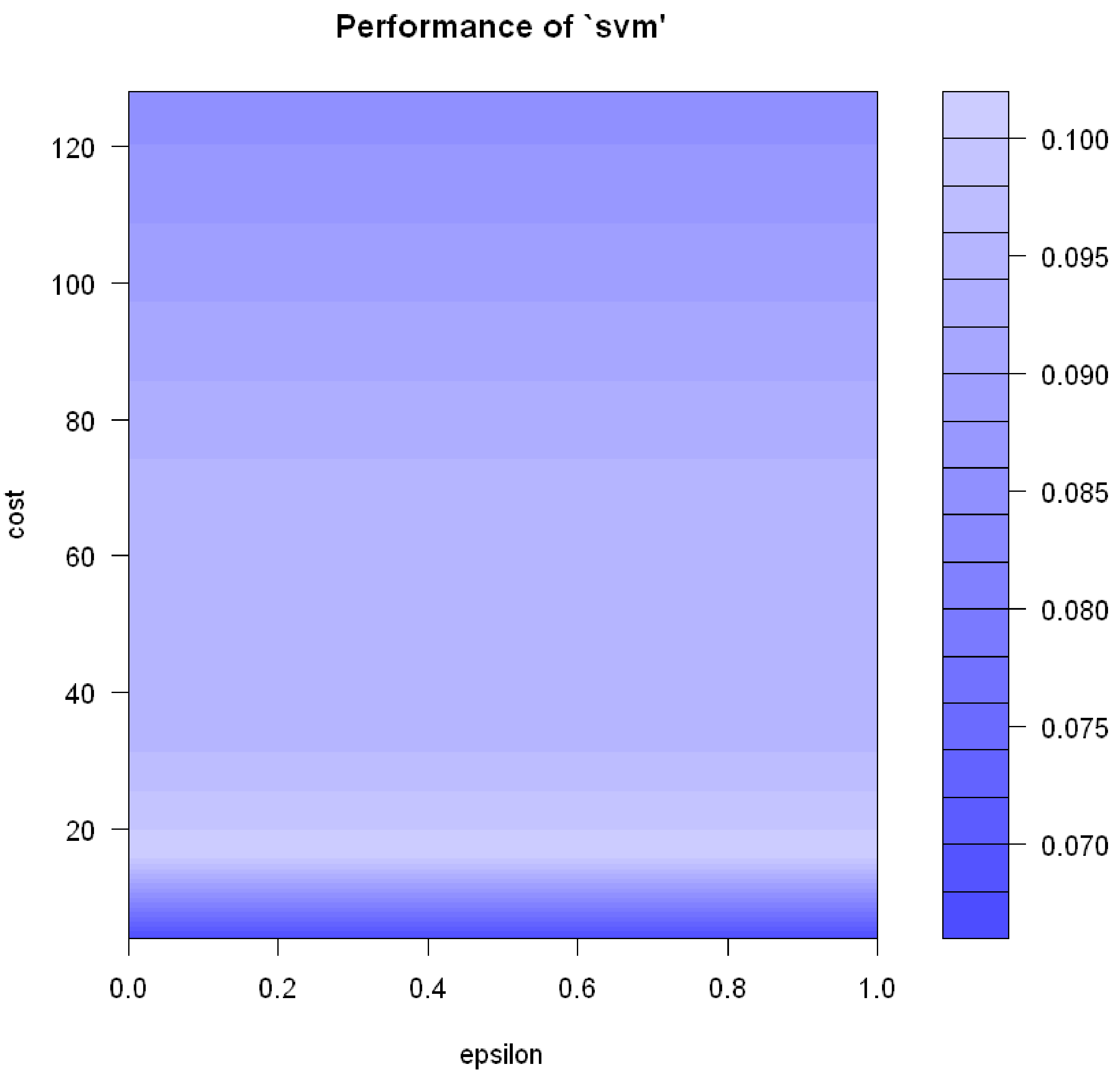
```
0.926966292134831
```

In [162]:

```
#Parameter Tunning
```

In [169]:

```
set.seed(123)
tmodel=tune(type = "C-classification", svm, Class~., data=wine,
    ranges=list(epsilon= seq(0,1,0.1), cost = 2^(2:7)))
plot(tmodel)
```



In [170]:

```
summary(tmodel)
```

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation 

- best parameters:
 epsilon cost
       0    4

- best performance: 0.06732026 

- Detailed performance results:
   epsilon cost      error dispersion
1      0.0    4 0.06732026 0.04460116
2      0.1    4 0.06732026 0.04460116
3      0.2    4 0.06732026 0.04460116
4      0.3    4 0.06732026 0.04460116
5      0.4    4 0.06732026 0.04460116
6      0.5    4 0.06732026 0.04460116
7      0.6    4 0.06732026 0.04460116
8      0.7    4 0.06732026 0.04460116
9      0.8    4 0.06732026 0.04460116
10     0.9    4 0.06732026 0.04460116
11     1.0    4 0.06732026 0.04460116
12     0.0    8 0.07875817 0.04724329
13     0.1    8 0.07875817 0.04724329
14     0.2    8 0.07875817 0.04724329
15     0.3    8 0.07875817 0.04724329
16     0.4    8 0.07875817 0.04724329
17     0.5    8 0.07875817 0.04724329
18     0.6    8 0.07875817 0.04724329
19     0.7    8 0.07875817 0.04724329
20     0.8    8 0.07875817 0.04724329
21     0.9    8 0.07875817 0.04724329
22     1.0    8 0.07875817 0.04724329
23     0.0   16 0.10130719 0.06403895
24     0.1   16 0.10130719 0.06403895
25     0.2   16 0.10130719 0.06403895
26     0.3   16 0.10130719 0.06403895
27     0.4   16 0.10130719 0.06403895
28     0.5   16 0.10130719 0.06403895
29     0.6   16 0.10130719 0.06403895
30     0.7   16 0.10130719 0.06403895
31     0.8   16 0.10130719 0.06403895
32     0.9   16 0.10130719 0.06403895
33     1.0   16 0.10130719 0.06403895
34     0.0   32 0.09575163 0.06002268
35     0.1   32 0.09575163 0.06002268
36     0.2   32 0.09575163 0.06002268
37     0.3   32 0.09575163 0.06002268
38     0.4   32 0.09575163 0.06002268
39     0.5   32 0.09575163 0.06002268
40     0.6   32 0.09575163 0.06002268
41     0.7   32 0.09575163 0.06002268
42     0.8   32 0.09575163 0.06002268
43     0.9   32 0.09575163 0.06002268
44     1.0   32 0.09575163 0.06002268
45     0.0   64 0.09575163 0.05400788
46     0.1   64 0.09575163 0.05400788
47     0.2   64 0.09575163 0.05400788
48     0.3   64 0.09575163 0.05400788
49     0.4   64 0.09575163 0.05400788
50     0.5   64 0.09575163 0.05400788
51     0.6   64 0.09575163 0.05400788
52     0.7   64 0.09575163 0.05400788
53     0.8   64 0.09575163 0.05400788
54     0.9   64 0.09575163 0.05400788
55     1.0   64 0.09575163 0.05400788
56     0.0  128 0.08464052 0.06141801
57     0.1  128 0.08464052 0.06141801
58     0.2  128 0.08464052 0.06141801
59     0.3  128 0.08464052 0.06141801
60     0.4  128 0.08464052 0.06141801
61     0.5  128 0.08464052 0.06141801
62     0.6  128 0.08464052 0.06141801
63     0.7  128 0.08464052 0.06141801
64     0.8  128 0.08464052 0.06141801
65     0.9  128 0.08464052 0.06141801
66     1.0  128 0.08464052 0.06141801
```

In [171]:

```
mymodel=tmodel$best.model
summary(mymodel)
```

```
Call:
best.tune(method = svm, train.x = Class ~ ., data = wine, ranges = list(epsilon = seq(0, 
    1, 0.1), cost = 2^(2:7)), type = "C-classification")


Parameters:
   SVM-Type:  C-classification 
 SVM-Kernel:  radial 
       cost:  4 

Number of Support Vectors:  45

 ( 14 19 12 )


Number of Classes:  3 

Levels: 
 Low Middle Premium
```

In [172]:

```
# Best model
```

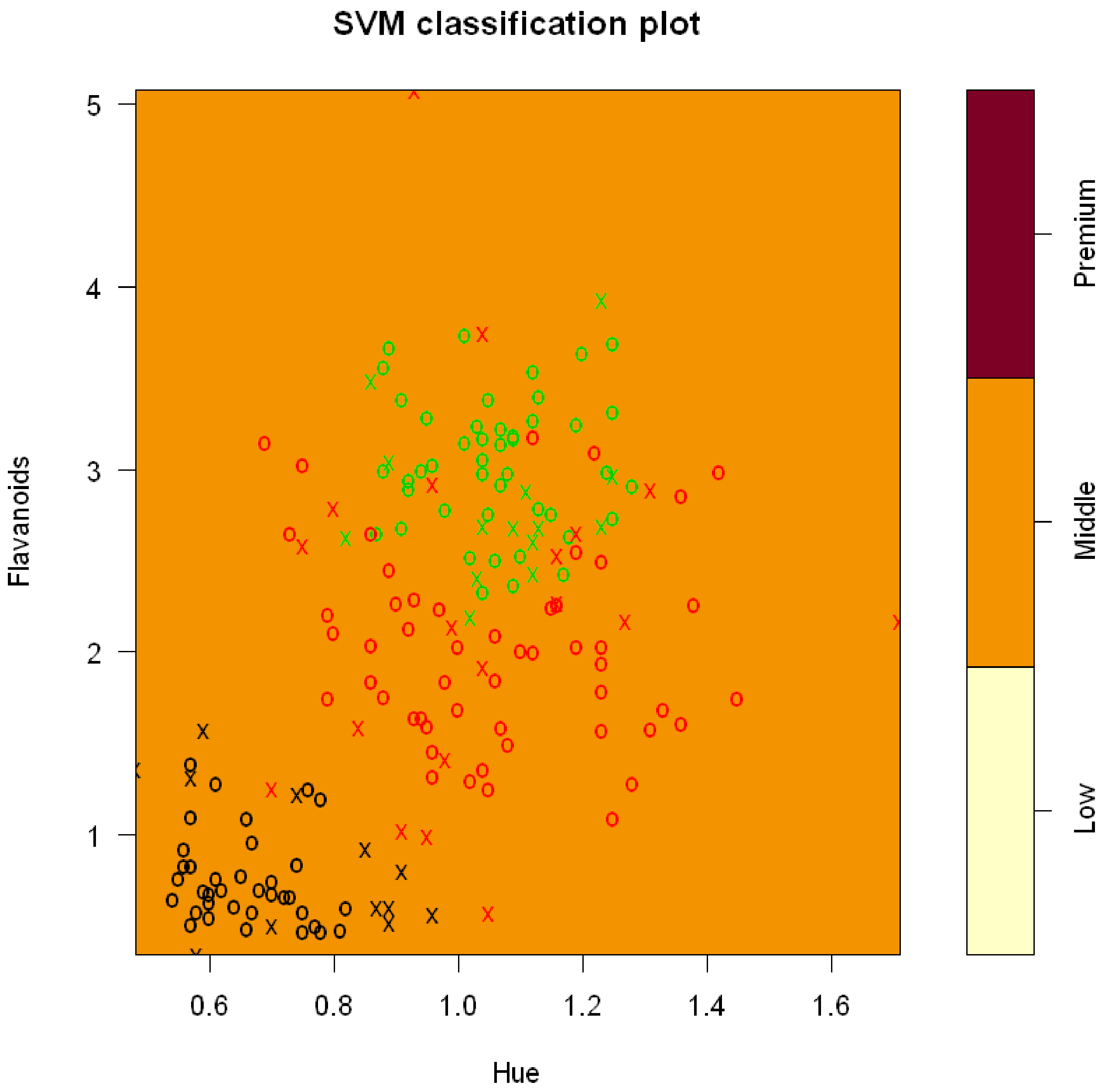
In [173]:

```
mymodel=tmodel$best.model
summary(mymodel)
```

```
Call:
best.tune(method = svm, train.x = Class ~ ., data = wine, ranges = list(epsilon = seq(0, 
    1, 0.1), cost = 2^(2:7)), type = "C-classification")


Parameters:
   SVM-Type:  C-classification 
 SVM-Kernel:  radial 
       cost:  4 

Number of Support Vectors:  45

 ( 14 19 12 )


Number of Classes:  3 

Levels: 
 Low Middle Premium
```

In [174]:

```
# RADIAL model was selected as the best

plot(mymodel, data=wine,
     Flavanoids~Hue,
     slice = list(OD=2, Proline=4)
)
```



In [175]:

```
# Confusion matrix and missclassification rate and accuracy using best parameter
```

In [176]:

```
pred1 = predict(mymodel,wine)
tab1 = table(Predicted=pred1, Actual = wine$Class)
tab1
```

```
         Actual
Predicted Low Middle Premium
  Low      48      1       0
  Middle    0     66       1
  Premium   0      4      58
```

In [177]:

```
1-sum(diag(tab1)/sum(tab1))
```

```
0.0337078651685393
```

In [178]:

```
sum(diag(tab1)/sum(tab1))
```

```
0.966292134831461
```

In [ ]: