DAR Coursework

Oboni Anower | 13923163 | MSc Advanced Computing

1. Statistical learning methods (10%)

For each of parts (a) through (d), indicate whether we would generally expect the performance of a non-parametric statistical learning method to be better or worse than a parametric method. Justify your answer.

(a) The number of predictors p is large, and the number of observations n is small. (2%)

In this scenario, performance of non-parametric methods would generally be better than the parametric methods.

Non-parametric statistical learning methods, such as decision trees or k-nearest neighbours do not make definite assumptions about functional form of the relationship between predictors and the target variable. Instead, they are flexible because they learn patterns directly from the data and adapts itself when left unconstrained.

On the contrary, parametric methods assume that the relationship between predictors and targets maintain linear relationship. Since the Observation is small, this method can struggle accurately estimate the parameters for complex functions due to insufficient data and give large residuals.

(b) The sample size n is large, and the number of predictors p is also large. (2%)

When both the sample size and the number of predictors is large, the performance of non-parametric methods would be worse than parametric methods.

With high-dimensional datasets, non-parametric methods can become computationally rigorous, which may also lead to overfitting. With too many predictors, the modelling will be unable to capture the underlying patterns in the data effectively, thus decreasing the performance. This is also known as the curse of dimensionality.

Since parametric models assume the relationship between predictors and targets are linear, it can generate better explainable models in complex scenarios.

(c) The sample size n is small, and the relationship between the predictors and response is highly linear. (3%)

When sample size is small, but predictor-response relationship is highly linear, undoubtably non-parametric methods will underperform compared to parametric methods.

With larger samples, non-parametric methods offer more robust estimates as they learn patterns directly from the relationships between predictors and responses. When the sample is smaller, methods have less observations available to accurately estimate the underlying relationships. Therefore, with smaller samples, non-parametric methods generate predictions with high variance.

Also, since there is small number of data available, the model will try to capture the noise in the data rather than the true underlying relationship. This causes overfitting, which can indeed create models that perform really well on the training dataset but perform poorly with new, unseen data, making the model less reliable.

For linear models, parametric methods can capture the actual relationship between predictors and their responses. This reduces prediction errors.

(d) The standard deviation of the error terms, i.e. $\sigma = \text{sd}(\varepsilon)$, is extremely high. (3%)

Non-parametric methods will perform better than parametric for extremely high standard deviations.

Cases where the error terms have high standard deviations, it is suggested that the predictor-response relationship is mostly likely to be complex or non-linear. A non-parametric method can flexibly adapt to the training data by capturing the variability in the data, if left unconstrained (no specified functional form). But parametric is linear form, so it is constrained.

In some cases, a very high standard deviation may occur due to the presence of outliers in the data distributions. But non-parametric methods are more robust to these outliers because they aren't influenced by data points that deviate from the normality. On the contrary, parametric methods are sensitive to outliers.

2. Linear regression (20%)

This question involves the `Auto` dataset included in the "ISLR" package.

(a) Use the `lm()` function to perform a simple linear regression with `acceleration` as the response and `cylinders` as the predictor. Use the `summary()` function to print the results. Comment on the output. For example:

```
library(ISLR)
data("Auto")

# Loads Auto datasets
lm.fit <- lm(acceleration ~ cylinders, data = Auto)
summary(lm.fit)

##
## Call:
```

```
## lm(formula = acceleration ~ cylinders, data = Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4778 -1.7428 -0.2428  1.3897  8.7222
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  20.0078     0.4052   49.38   <2e-16 ***
## cylinders    -0.8163     0.0707  -11.54   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.385 on 390 degrees of freedom
## Multiple R-squared:  0.2547, Adjusted R-squared:  0.2528
## F-statistic: 133.3 on 1 and 390 DF,  p-value: < 2.2e-16
```

i. Is there a relationship between the predictor and the response? (3%)

The result in the summary suggests that there is a significant statistical relationship between the number of cylinders and acceleration.

Here, Pr(>|t|) is the P-value, that is <2e-16 meaning it is smaller than 0.05 (very close to zero, but not equal to zero). Thus it rejects null hypothesis, implying that there is an observable linear relationship between the predictor and response.

Th RSE is 2.385, which suggests that any prediction on the observed acceleration values based on the number of cylinders would be off by 2.385 on average when the least squares line is used.

ii. How strong is the relationship between the predictor and the response? (3%)

R-squared value is 0.2547, which is closer to zero than 1. This means that only about 25% of the response's variance can be explained by the variation of predictor.Thus, the relationship is moderately strong.

iii. Is the relationship between the predictor and the response positive or negative? (3%)

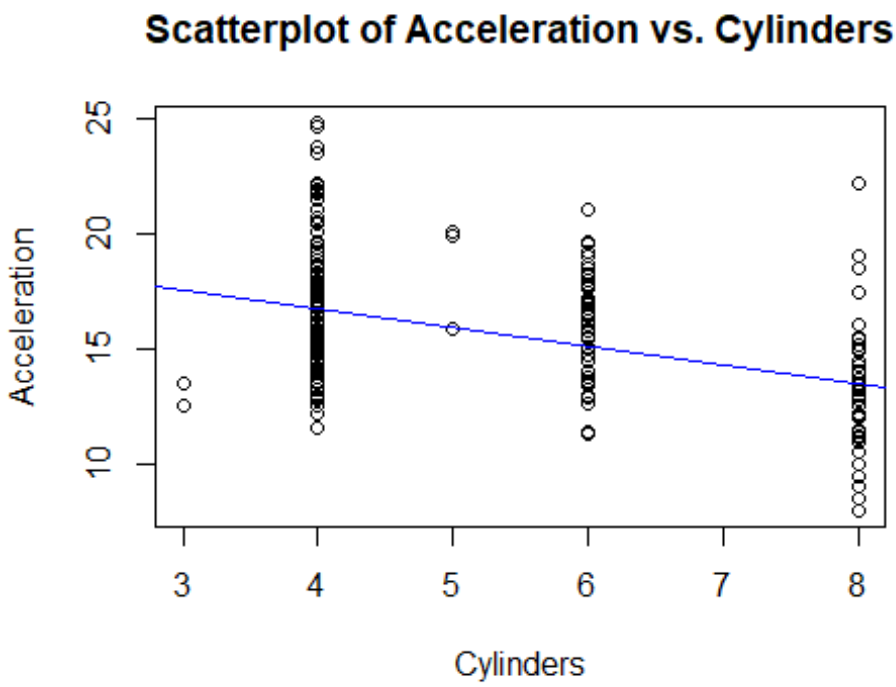
The predictor coefficient (cylinders) is -0.8163. This Negative value suggests that for every additional cylinder in the added to the engine, the acceleration (response) is expected to decrease by 0.8163 units.

iv. What is the predicted `acceleration` associated with an `cylinders` of 3.0? What are the associated 99% confidence and prediction intervals? (3%)

```
c_interval <- predict(lm.fit, data.frame(cylinders=(c(3.0))), interval = "con
fidence", level = 0.99)
print(c_interval)

##        fit      lwr      upr
## 1 17.55906 17.00962 18.10849

p_interval <- predict(lm.fit, data.frame(cylinders=(c(3.0))), interval = "pre
diction", level = 0.99)
print(p_interval)

##        fit      lwr      upr
## 1 17.55906 11.36164 23.75648

cat("\n","Predicted acceleration:", c_interval[1], "\n")

##
##  Predicted acceleration: 17.55906

cat("99% Confidence Interval: from ", c_interval[2], " to ", c_interval[3], "
\n")

## 99% Confidence Interval: from  17.00962  to  18.10849

cat("99% Prediction Interval: from ", p_interval[2], " to ", p_interval[3], "
\n")

## 99% Prediction Interval: from  11.36164  to  23.75648
```

(b) Plot the response and the predictor. Use the `abline()` function to display the least squares regression line. (3%)

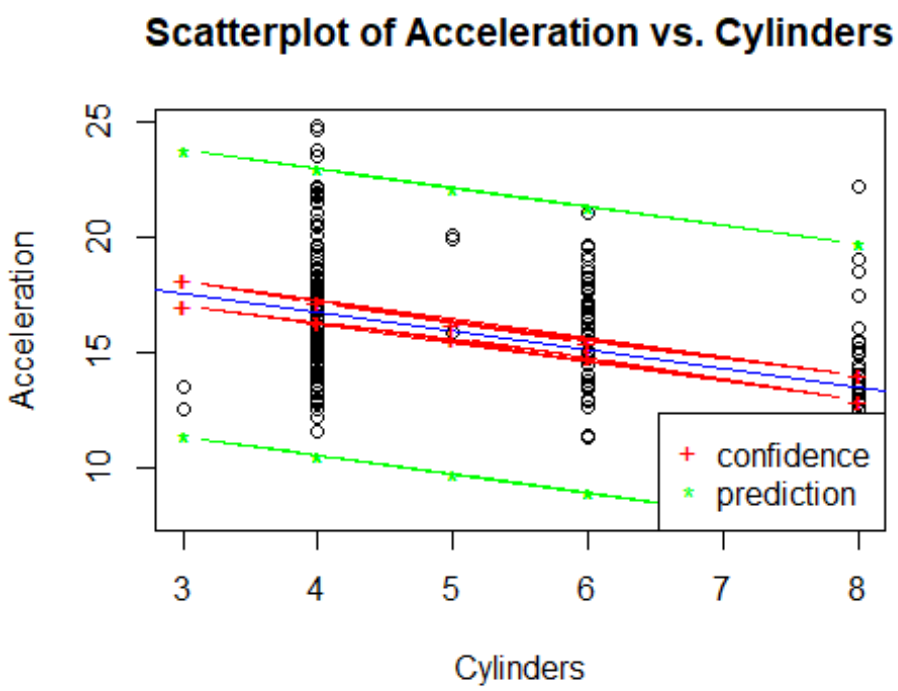
```
plot(Auto$cylinders, Auto$acceleration,
     xlab = "Cylinders", ylab = "Acceleration",
     main = "Scatterplot of Acceleration vs. Cylinders")

# Add the least squares regression line
abline(lm.fit, col = "blue")
```



(c) Plot the 99% confidence interval and prediction interval in the same plot as (b) using different colours and legends. (5%)

```
c_int <- predict(lm.fit, data.frame(cylinders =(Auto$cylinders)), interval =
"confidence", level = 0.99)
p_int <- predict(lm.fit, data.frame(cylinders = (Auto$cylinders)), interval =
"prediction", level = 0.99)

plot(Auto$cylinders, Auto$acceleration,
     xlab = "Cylinders", ylab = "Acceleration",
     main = "Scatterplot of Acceleration vs. Cylinders")

# Add the least squares regression line
abline(lm.fit, col = "blue")

# Add the confidence interval to the plot
lines(Auto$cylinders, c_int[,2], col = "red", type="b", pch="+")
lines(Auto$cylinders, c_int[,3], col = "red", type="b", pch="+")

# Add the prediction interval to the plot
lines(Auto$cylinders, p_int[,2], col = "green", type="b", pch="*")
lines(Auto$cylinders, p_int[,3], col = "green", type="b", pch="*")


# Add legends
```

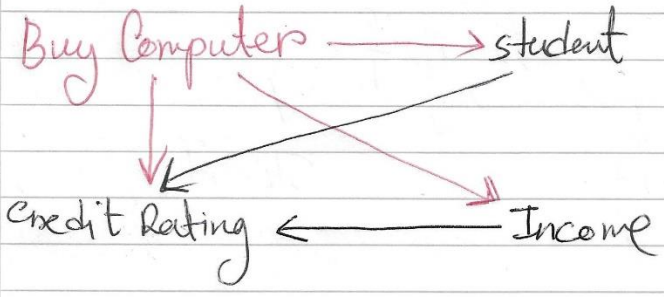
```
legend("bottomright",
pch=c("+","*"),
col=c("red","green"),
legend = c("confidence","prediction"))
```



3. Bayesian networks and naïve Bayes classifiers.(30%)

a) Given a training dataset including 30 observations and a Bayesian network indicating the relationships between 3 features (i.e. Income, Student and Credit Rate) and the class attribute (i.e. Buy Computer), please create the conditional probability tables by hand. (10%)

3a.



Buy Computer (BC)

yes	no
0.4	0.6

total 30
yes = 12 , No = 18
∴ P(yes) = 12/30 = 0.4
P(no) = 1 − 0.4 = 0.6

CPT for Income

P(Income = High | B.C = No) = 7/18 = 0.38
P(Income = Low | B.C = No) = 11/18 = 0.612
P(Income = High | BC = yes) = 5/12 = 0.417
P(Income = Low | BC = yes) = 7/12 = 0.583

Buy Computer	High	Low
yes	0.417	0.583
No	0.38	0.612

CPT for credit rating

⇒ 2 example probability calculations will be included, the rest of the values are shown in the table.

P(Credit rating = Fair | Income = High, Student = True, Buy Computer = No) = 3/4 = 0.75

P(Credit Rating = Excellent | Income = low, Student = False, Buy Computer = yes) = 1/3 = 0.3

Income	Student	Buy Computer	Credit Rating	
			Fair	Excellent
High	True	yes	1/2 = 0.5	0.5
High	True	No	3/4 = 0.75	0.25
High	False	yes	1/3 = 0.33	0.667
High	False	No	1/3 = 0.333	0.667
Low	True	yes	2/4 = 0.5	0.5
Low	True	No	6/8 = 0.75	0.25
Low	False	yes	1/3 = 0.333	0.667
Low	False	NO	2/3 = 0.667	0.333

CPT for Student

$P(\text{Student} = \text{True} \mid BC = No) = 12/18 = 0.667$
$P(\text{Student} = \text{False} \mid BC = No) = 6/18 = 0.333$

$P(\text{Student} = \text{True} \mid BC = yes) = 6/12 = 0.5$
$P(\text{Student} = \text{False} \mid BC = yes) = 6/12 = 0.5$

Buy Computer	Student	
	True	False
yes	0.5	0.5
NO	0.6	0.3

CPT for Credit Rating (CR)

Income = I, Student = S, Buy Computer = B

$P(\text{Credit Rating} = \text{Fair} \mid I = \text{High}, S = \text{True}, BC = yes) = 1/2 = 0.5$
$P(CR = \text{Fair} \mid I = \text{High}, S = \text{True}, BC = No) = 0.5$

b) Make predictions for 2 testing observations by using a Bayesian network classifier. (5%)

3b.

Observation 31

Buy=YES $P($Buy Computer = Yes, Income = Low, Student = True, Credit Rating = Excellent$)$

$= P($Income = Low $|$ Buy Computer = Yes$)$
$\times P($Student = True $|$ Buy Computer = yes$)$
$\times P($Credit Rating = Excellent $|$ Income = Low, Student = True, Buy computer = Yes$)$
$\times P($Buy Computer = yes$)$
$= 0.583 \times 0.5 \times 0.5 \times 0.9$
≈ 0.0583

Buy=NO $P($Buy Computer = No, Income = Low, Student = True, Credit Rating = Excellent$)$

$= P($Income = low $|$ Buy Computer = No$)$
$\times P($Student = True $|$ Buy Computer = No$)$
$\times P($Credit Rating = Excellent $|$ Income = Low, Student = True, Buy Computer = No$)$
$\times P($Buy Computer = No$)$
$= 0.612 \times 0.667 \times 0.25 \times 0.6$
≈ 0.0612

∴ Probability of Buying a Computer is lower than not buying. So Buy Computer = No. Predicted.

3b Observation 32

Buy = Yes

P(Buy Computer = Yes, Income = High, Student = True, Credit Rating = Fair)

= P(Income = High | Buy Computer = Yes)
× P(Student = True | Buy Computer = Yes)
× P(Credit Rating | Income = High, Student = True, Buy Computer = Yes)
× P(Buy Computer = yes)
= 0.497 × 0.5 × 0.5 × 0.4 ≈ 0.0417

Buy = NO

P(Buy Computer = No, Income = High, Student = True, Credit Rating = Fair)

= P(Income = High | Buy Computer = No)
× P(Student = True | Buy Computer = No)
× P(Credit Rating = Fair | Income = High, Student = true, Buy Computer = No)
× P(Buy Computer = No)
= 0.38 × 0.667 × 0.75 × 0.6 ≈ 0.1141

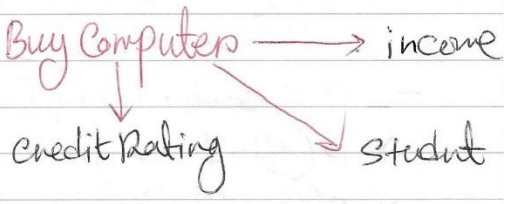
∴ Probability of Not buying a Computer is higher.

∴ Prediction is Buy Computer = No.

c) Based on the conditional independence assumption between features, please create the conditional probability tables by hand. (10%)

3c. CPT tables for Income, Student and Buy Computer is same as 3a.

Buy Computer	
yes	NO
0.4	0.6



	Income	
Buy Computer	High	Low
Yes	0.417	0.583
NO	0.38	0.612

	Student	
Buy Computer	True	False
yes	0.5	0.5
NO	0.667	0.333

Additional table:

	Credit Rating	
Buy Computer	Fair	Excellent
yes	5/12 ≈ 0.417	0.583
no	12/18 ≈ 0.667	0.333

d) Make predictions for 2 testing observations by using a naïve Bayes classifier. (5%)

3d. Observation 31

for Buy Computer = yes class label

$P(\text{Buy Computer} = \text{yes}, \text{Income} = \text{low}, \text{Student} = \text{True}, \text{Credit rating} = \text{Excellent})$

$= P(\text{Income} = \text{Low} \mid \text{Buy Computer} = \text{yes})$
$\times P(\text{Student} = \text{True} \mid \text{Buy Computer} = \text{yes})$
$\times P(\text{Credit Rating} = \text{Excellent} \mid \text{Buy Computer} = \text{yes})$
$\times P(\text{Buy Computer} = \text{yes})$
$= 0.583 \times 0.5 \times 0.583 \times 0.4 \approx 0.068$

for Buy Computer = No class label

$P(\text{Buy Computer} = \text{No}, \text{Income} = \text{low}, \text{Student} = \text{True}, \text{Credit Rating} = \text{Excellent})$

$= P(\text{Income} = \text{Low} \mid \text{Buy Computer} = \text{No})$
$\times P(\text{Student} = \text{True} \mid \text{Buy Computer} = \text{No})$
$\times P(\text{Credit Rating} = \text{Excellent} \mid \text{Buy Computer} = \text{No})$
$\times P(\text{Buy Computer} = \text{No})$
$= 0.612 \times 0.667 \times 0.333 \times 0.6 \approx 0.0816$

∴ In Observation 31, Class label No is greater than class label yes.

Therefore, the prediction is "No".

3d. Observation 32

for Class label yes,

P(Buy Computer = yes, Income = High, Student = Tru
Credit Rating = Fair)

= P(Income = High | Buy Computer = yes)
× P(Student = True | Buy Computer = yes)
× P(Credit Rating = Fair | Buy Computer = yes
× P(Buy Computer = yes)
= 0.417 × 0.5 × 0.417 × 0.4 ≈ 0.0348

For Class label No,

P(Buy Computer = No, Income = High, Student = Tru
Credit Rating = Fair)

= P(Income = High | Buy Computer = No)
× P(Student = True | Buy Computer = No)
× P(Credit Rating = Fair | Buy Computer = No)
× P(Buy Computer = No)
= 0.38 × 0.667 × 0.667 × 0.6 ≈ 0.1014

∴ In Observation 32, Class label "No" is greater than class label "yes".

Thus, the Prediction is "No".

4. Predicting wine quality by using support vector machine classification algorithm. (40%)

a) Download the full wine quality training and testing datasets from Moodle, and use the training dataset to find out the optimal value of hyperparameter C for a linear kernel-based svm. Define the value of the random seed equals 1 and cost = c(0.01, 1, 100). (5%)

```
#installs library for SVM and cross-validation
library(e1071)

# loads Training dataset
# Converts predictor's categorical values to numerical values (1 for "Good" a
nd -1 for "Bad")
train_WineQuality <- read.table("C:/Users/anowe/OneDrive/Documents/WineQualit
y_Training.txt", header = TRUE, sep = ",")
train_WineQuality$quality <- ifelse(train_WineQuality$quality == "Good", 1, -
1)

# hyperparameter tuning
set.seed(1)
cost_val <- c(0.01, 1, 1e2)
tune.out <- tune(svm,
                 quality ~ .,
                 data = train_WineQuality,
                 kernel = "linear",
                 ranges = list(cost = cost_val),
                 )
summary(tune.out)

##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 0.7489535
##
## - Detailed performance results:
##    cost     error dispersion
## 1 1e-02 0.7594945 0.03335106
## 2 1e+00 0.7489535 0.03936566
## 3 1e+02 0.7492469 0.03986006
```

b) Train a svm classifier by using the linear kernel and the corresponding optimal value of hyperparameter C, then make predictions on the testing dataset, report the classification accuracy. (10%)

```r
# converts the target variable from categorical to numerical
test_WineQuality <- read.table("C:/Users/anowe/OneDrive/Documents/WineQuality
_Testing.txt", header = TRUE, sep = ",")
test_WineQuality$quality <- ifelse(test_WineQuality$quality == "Good", 1, -1)

# trains SVM model using full training dataset and Optimal value
optimal_value_C1 <- 1
svmfit.linear.C1 <- svm(quality ~ .,
                 data = train_WineQuality,
                 type = "C-classification",
                 kernel = "linear",
                 cost = optimal_value_C1
                 )
summary(svmfit.linear.C1)

##
## Call:
## svm(formula = quality ~ ., data = train_WineQuality, type = "C-classificat
ion",
##     kernel = "linear", cost = optimal_value_C1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  1710
##
##  ( 855 855 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1

# makes predictions on the test dataset using trained SVM model
# prints classification accuracy of predictions in %
y.predict.linear.C1 <- predict(svmfit.linear.C1, newdata = test_WineQuality)
accuracy_report <- mean(y.predict.linear.C1 == test_WineQuality$quality) * 10
0
print(paste("CLASIFICATION ACCURACY REPORT: ", round(accuracy_report, 2), "%"
))

## [1] "CLASIFICATION ACCURACY REPORT:  68.25 %"
```

c) Use the training dataset to find out the optimal values of hyperparameters C and for an RBF kernel-based svm. Define the value of the random seed equals 1, cost = c(0.01, 1, 100) and gamma=c(0.01, 1, 100). (5%)

```r
# Sets the random seed
# Defines the values of C and gamma to try
set.seed(1)
cost_v <- c(0.01, 1, 100)
gamma_v <- c(0.01, 1, 100)

# hyperparameter tuning using C and gamma
tune.out_C_gamma <- tune(svm,
                quality ~ .,
                data = train_WineQuality,
                kernel = "radial",
                ranges = list(cost = cost_v, gamma = gamma_v)
                )
summary(tune.out_C_gamma)

##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##   100     1
##
## - best performance: 0.4864171
##
## - Detailed performance results:
##    cost gamma     error dispersion
## 1 1e-02 1e-02 0.7796057 0.03493146
## 2 1e+00 1e-02 0.6797668 0.03000791
## 3 1e+02 1e-02 0.6242817 0.05653273
## 4 1e-02 1e+00 1.5108124 0.22355174
## 5 1e+00 1e+00 0.4888902 0.05816749
## 6 1e+02 1e+00 0.4864171 0.05384989
## 7 1e-02 1e+02 1.7312625 0.26626939
## 8 1e+00 1e+02 0.6646390 0.02221481
## 9 1e+02 1e+02 0.6646386 0.02221448
```

d) Train a svm classifier by using the RBF kernel and the corresponding optimal values of hyperparameters C and gamma, then make predictions on the testing dataset, report the classification accuracy. (10%)

```r
# trains SVM model using full training dataset and Optimal value
optimal_C <- 100
optimal_G <- 1
svmfit.radial.C100G1 <- svm(quality ~ .,
                 data = train_WineQuality,
                 kernel = "radial",
                 type = "C-classification",
                 cost = optimal_C,
```

```
                  gamma = optimal_G,
                  probability = TRUE
                  )
summary(svmfit.radial.C100G1)
```

```
##
## Call:
## svm(formula = quality ~ ., data = train_WineQuality, kernel = "radial",
##     type = "C-classification", cost = optimal_C, gamma = optimal_G,
##     probability = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  100
##
## Number of Support Vectors:  2173
##
##  ( 1071 1102 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```
# makes predictions on the test dataset using trained SVM model
# gets classification accuracy of predictions in %
y.predict.radial.C100G1 <- predict(svmfit.radial.C100G1, newdata = test_WineQ
uality)
accuracy_report2 <- mean(y.predict.radial.C100G1 == test_WineQuality$quality)
* 100
print(paste("CLASSIFICATION ACCURACY REPORT: ", round(accuracy_report2, 2), "
%"))
```

```
## [1] "CLASSIFICATION ACCURACY REPORT:  64 %"
```

e) Train a logistic regression model. Then use the testing dataset to conduct an ROC curve analysis to compare the predictive performance of the trained logistic regression model and those two svm classifiers trained by using linear and RBF kernels respectively. (10%)

```
library(ROCR)

train_data = read.table("C:/Users/anowe/OneDrive/Documents/WineQuality_Testin
g.txt", header = TRUE, sep = ",")
# Convert quality to binary
train_data$quality <- ifelse(train_data$quality == "Good", 1, 0)

# Train logistic regression model
```

```r
train.logit_model <- glm(quality ~ .,
                         data = train_data,
                         family = binomial,
                         maxit = 1000
                         )
summary(train.logit_model)
```

```
##
## Call:
## glm(formula = quality ~ ., family = binomial, data = train_data,
##     maxit = 1000)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2384  -0.3963   0.3137   0.5908   2.3279
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)           1.142e+03  3.146e+02   3.629 0.000285 ***
## fixed.acidity        -8.168e-01  2.923e-01  -2.794 0.005201 **
## volatile.acidity     -5.740e+00  1.336e+00  -4.295 1.75e-05 ***
## citric.acid           3.014e+00  1.300e+00   2.318 0.020427 *
## residual.sugar        6.044e-01  1.252e-01   4.828 1.38e-06 ***
## chlorides             4.327e+00  7.999e+00   0.541 0.588586
## free.sulfur.dioxide   9.594e-03  7.784e-03   1.233 0.217749
## total.sulfur.dioxide -6.847e-03  5.231e-03  -1.309 0.190540
## density              -1.146e+03  3.185e+02  -3.597 0.000322 ***
## pH                    9.394e-01  1.635e+00   0.575 0.565602
## sulphates             3.236e+00  1.550e+00   2.088 0.036802 *
## alcohol              -3.974e-01  4.077e-01  -0.975 0.329651
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 520.39  on 399  degrees of freedom
## Residual deviance: 296.99  on 388  degrees of freedom
## AIC: 320.99
##
## Number of Fisher Scoring iterations: 6
```

```r
library(ROCR)

test_data = read.table("C:/Users/anowe/OneDrive/Documents/WineQuality_Testing
.txt", header = TRUE, sep = ",")
# Convert to binary classification (good vs. not good) for logistic regressio
n
test_data$quality <- ifelse(test_data$quality == "Good", 1, 0)
# Check the structure of predicted_linear
```

```r
# Logistic Regression
y.predict.logit <- predict(train.logit_model, newdata = test_data, type = "re
sponse")
p_log <- prediction(y.predict.logit, test_data$quality)
# ROC curve and AUC calculation for Logistic Regression
perf_logit <- performance(p_log, "tpr", "fpr")
auc_logit <- performance(p_log, "auc")
```