Question 10.1

Using the same crime data set uscrime.txt as in Questions 8.2 and 9.1, find the best model you can using

(a) a regression tree model, and

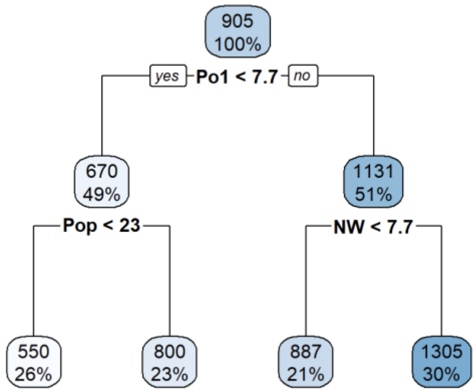
(b) a random forest model.

In R, you can use the tree package or the rpart package, and the randomForest package. For each model, describe one or two qualitative takeaways you get from analyzing the results (i.e., don't just stop when you have a good model, but interpret it too).

(a)

Code:

```
data <- read.table('C:/Users/Susie/Desktop/uscrime.txt', head=TRUE)

# Install packages
install.packages("tree")
install.packages("randomForest")
install.packages("rpart")
install.packages("rpart.plot")

# Load the required libraries
library(rpart)
library(rpart.plot)

# Fit the regression tree model
tree_model <- rpart(Crime ~ ., data = data, method = "anova")

# Plot the tree
rpart.plot(tree_model)

# Summary of the tree model
summary(tree_model)
```



Summary:

rpart(formula = Crime ~ ., data = data, method = "anova")

n= 47

	CP	nsplit	rel error	xerror	xstd
1	0.36296293	0	1.0000000	1.0363101	0.2560318
2	0.14814320	1	0.6370371	0.9214845	0.1991814
3	0.05173165	2	0.4888939	1.0603477	0.2485937
4	0.01000000	3	0.4371622	1.0035757	0.2378527

Variable importance

Po1	Po2	Wealth	Ineq	Prob	M	NW	Pop	Time	Ed	LF	So
17	17	11	11	10	10	9	5	4	4	1	1

Node number 1: 47 observations,   complexity param=0.3629629

mean=905.0851, MSE=146402.7

left son=2 (23 obs) right son=3 (24 obs)

Primary splits:

Po1   < 7.65   to the left, improve=0.3629629, (0 missing)

```
      Po2    < 7.2       to the left,  improve=0.3629629, (0 missing)
      Prob   < 0.0418485 to the right, improve=0.3217700, (0 missing)
      NW     < 7.65      to the left,  improve=0.2356621, (0 missing)
      Wealth < 6240      to the left,  improve=0.2002403, (0 missing)
  Surrogate splits:
      Po2    < 7.2       to the left,  agree=1.000, adj=1.000, (0 split)
      Wealth < 5330      to the left,  agree=0.830, adj=0.652, (0 split)
      Prob   < 0.043598  to the right, agree=0.809, adj=0.609, (0 split)
      M      < 13.25     to the right, agree=0.745, adj=0.478, (0 split)
      Ineq   < 17.15     to the right, agree=0.745, adj=0.478, (0 split)

Node number 2: 23 observations,    complexity param=0.05173165
  mean=669.6087, MSE=33880.15
  left son=4 (12 obs) right son=5 (11 obs)
  Primary splits:
      Pop < 22.5      to the left,  improve=0.4568043, (0 missing)
      M   < 14.5      to the left,  improve=0.3931567, (0 missing)
      NW  < 5.4       to the left,  improve=0.3184074, (0 missing)
      Po1 < 5.75      to the left,  improve=0.2310098, (0 missing)
      U1  < 0.093     to the right, improve=0.2119062, (0 missing)
  Surrogate splits:
      NW   < 5.4       to the left,  agree=0.826, adj=0.636, (0 split)
      M    < 14.5      to the left,  agree=0.783, adj=0.545, (0 split)
      Time < 22.30055  to the left,  agree=0.783, adj=0.545, (0 split)
      So   < 0.5       to the left,  agree=0.739, adj=0.455, (0 split)
      Ed   < 10.85     to the right, agree=0.739, adj=0.455, (0 split)

Node number 3: 24 observations,    complexity param=0.1481432
  mean=1130.75, MSE=150173.4
```

```
  left son=6 (10 obs) right son=7 (14 obs)
  Primary splits:
      NW   < 7.65      to the left,  improve=0.2828293, (0 missing)
      M    < 13.05     to the left,  improve=0.2714159, (0 missing)
      Time < 21.9001   to the left,  improve=0.2060170, (0 missing)
      M.F  < 99.2      to the left,  improve=0.1703438, (0 missing)
      Po1  < 10.75     to the left,  improve=0.1659433, (0 missing)
  Surrogate splits:
      Ed   < 11.45     to the right, agree=0.750, adj=0.4, (0 split)
      Ineq < 16.25     to the left,  agree=0.750, adj=0.4, (0 split)
      Time < 21.9001   to the left,  agree=0.750, adj=0.4, (0 split)
      Pop  < 30        to the left,  agree=0.708, adj=0.3, (0 split)
      LF   < 0.5885    to the right, agree=0.667, adj=0.2, (0 split)

Node number 4: 12 observations
  mean=550.5, MSE=20317.58

Node number 5: 11 observations
  mean=799.5455, MSE=16315.52

Node number 6: 10 observations
  mean=886.9, MSE=55757.49

Node number 7: 14 observations
  mean=1304.929, MSE=144801.8
```

Interpretation for summary:

Based on the variable importance, we can observe that Po1 and Po2 are the most significant predictors of crime rates. Additionally, Wealth and Ineq also play a role, indicating that wealth levels and income inequality are strongly correlated with crime rates. Thus, lower Po1 and Po2 values are associated with lower crime rates. In contrast, higher NW values are linked to higher crime rates.

Interpretation for plotgraph:

This regression tree illustrates the process of predicting crime rates using two key variables: Po1 and NW. The data is first split based on Po1, and then further refined by either Pop or NW.

The left branch shows that lower Po1 and Pop values are typically associated with lower crime rates (550/800).The right branch reveals that higher Po1 and NW values tend to correspond to higher crime rates (887 /1305).

(b)

Code:

```
# Plot the tree
rpart.plot(tree_model)

# Summary of the tree model
summary(tree_model)

# Load the randomForest package
library(randomForest)

# Fit the random forest model
random_forest_model <- randomForest(Crime ~ ., data = data, importance = TRUE)

# Print the model summary
print(random_forest_model)

# Plot variable importance
varImpPlot(random_forest_model)
```

Summary:

```
randomForest(formula = Crime ~ ., data = data, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 5
          Mean of squared residuals: 85214.77
                    % Var explained: 41.79
```

Question 10.2

Describe a situation or problem from your job, everyday life, current events, etc., for which a logistic regression model would be appropriate. List some (up to 5) predictors that you might use.

Problem: whether one can finish one's dues before Thanksgiving

1. The health condition before Thanksgiving (categorical): It can be excellent, a little bit tired, a little bit sick, or sick to death, and one cannot do anything. The worse the health condition is, the less productivity one will obtain, and the slower one will finish the work.
2. The workload due around Thanksgiving (continuous) can be any duration between 0 hours and the maximum time before Thanksgiving. The more workload, the less likely one is to finish it on time.
3. The variety of activities that are attractive to the student and happen before Thanksgiving (continuous): It can be one or multiple events (e.g., a ballet, an amusement park tour, etc.). The more attractive activities are available, the less time and the less likely it is to finish the tasks.
4. The motivation to finish the tasks before Thanksgiving or the potential number of activities that one would access during Thanksgiving (continuous): If one is very eager to go to Disneyland during Thanksgiving or hike in Puerto Rico, then the stronger the wish, the more likely one will try one's best to finish it beforehand.

Question 10.3

1. Using the GermanCredit data set germancredit.txt from http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german / (description at http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29).

   Use logistic regression to find a good predictive model for whether credit applicants are good credit risks or not.

   Show your model (factors used and their coefficients), the software output, and the quality of fit. You can use the glm function in R.

   To get a logistic regression (logit) model on data where the response is either zero or one, use family=binomial(link="logit") in your glm function call.

Answer:

To find a good predictive model with logistic regression, we use GLM Function with binomial distribution.

Code:

First, we load the German credit dataset, then we convert the response variable into a factor (1: good credit risk, 0: bad credit risk).

Next, we set the non- numerical or non- continuous as categorical variables and covert categorical variables into factors.

Then, we fit all variables into logistic regression model.

```
install.packages("readr")

library(readr)

url <- "http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data"

column_names <- c("Status_of_existing_checking_account", "Duration_in_month", "Credit_history", "Purpose",

          "Credit_amount", "Savings_account_bonds", "Present_employment_since",
"Installment_rate_in_percentage_of_disposable_income",

          "Personal_status_and_sex", "Other_debtors_guarantors", "Present_residence_since", "Property", "Age_in_years",

          "Other_installment_plans", "Housing", "Number_of_existing_credits_at_this_bank", "Job",

          "Number_of_people_being_liable_to_provide_maintenance_for", "Telephone", "Foreign_worker", "Credit_risk")

german_credit <- read_delim(url, delim = " ", col_names = column_names)

german_credit$Credit_risk <- factor(ifelse(german_credit$Credit_risk == 1, 1, 0))
```

```
categorical_vars <- c("Status_of_existing_checking_account", "Credit_history", "Purpose",
                      "Savings_account_bonds", "Present_employment_since",
                      "Personal_status_and_sex", "Other_debtors_guarantors",
                      "Property", "Other_installment_plans", "Housing",
                      "Job", "
german_credit[categorical_vars] <- lapply(german_credit[categorical_vars], as.factor)
model_all <- glm(Credit_risk ~ ., data = german_credit, family = binomial(link = "logit"))
summary(model_all)
```

The table 1 shows the factors and their coefficients.

Table 1 Factors and their coefficients

Coefficients: Factors	Estimate
(Intercept)	-0.4005
Status_of_existing_checking_accountA12	0.3749
Status_of_existing_checking_accountA13	0.9657
Status_of_existing_checking_accountA14	1.7120
Duration_in_month	-0.0279
Credit_historyA31	-0.1434
Credit_historyA32	0.5861
Credit_historyA33	0.8532
Credit_historyA34	1.4360
PurposeA41	1.6660
PurposeA410	1.4890
PurposeA42	0.7916
PurposeA43	0.8916
PurposeA44	0.5228
PurposeA45	0.2164
PurposeA46	-0.0363
PurposeA48	2.0590
PurposeA49	0.7401
Credit_amount	-0.0001
Savings_account_bondsA62	0.3577
Savings_account_bondsA63	0.3761
Savings_account_bondsA64	1.3390
Savings_account_bondsA65	0.9467
Present_employment_sinceA72	0.0669

Present_employment_sinceA73	0.1828
Present_employment_sinceA74	0.8310
Present_employment_sinceA75	0.2766
Installment_rate_in_percentage_of_disposable_income	-0.3301
Personal_status_and_sexA92	0.2755
Personal_status_and_sexA93	0.8161
Personal_status_and_sexA94	0.3671
Other_debtors_guarantorsA102	-0.4360
Other_debtors_guarantorsA103	0.9786
Present_residence_since	-0.0048
PropertyA122	-0.2814
PropertyA123	-0.1945
PropertyA124	-0.7304
Age_in_years	0.0145
Other_installment_plansA142	0.1232
Other_installment_plansA143	0.6463
HousingA152	0.4436
HousingA153	0.6839
Number_of_existing_credits_at_this_bank	-0.2721
JobA172	-0.5361
JobA173	-0.5547
JobA174	-0.4795
Number_of_people_being_liable_to_provide_maintenance_for	-0.2647
TelephoneA192	0.3000
Foreign_workerA202	1.3920

Table 2 shows their z-values, p-values and their priority. **Then we remove manually the unsignificant factors and fit the logistic regression model with the priority.**

Table 2 z-values, p-values and their priority

Factors	z value	p value	priority
(Intercept)	-0.369	0.711869	
Status_of_existing_checking_accountA12	1.72	0.0854	.
Status_of_existing_checking_accountA13	2.616	0.008905	**
Status_of_existing_checking_accountA14	7.373	1.66E-13	***
Duration_in_month	-2.997	0.002724	**
Credit_historyA31	-0.261	0.793921	
Credit_historyA32	1.362	0.173348	
Credit_historyA33	1.809	0.07047	.

Credit_historyA34	3.264	0.001099	**
PurposeA41	4.452	8.51E-06	***
PurposeA410	1.918	0.055163	.
PurposeA42	3.033	0.002421	**
PurposeA43	3.609	0.000308	***
PurposeA44	0.686	0.492831	
PurposeA45	0.393	0.694	
PurposeA46	-0.092	0.927082	
PurposeA48	1.699	0.089297	.
PurposeA49	2.216	0.026668	*
Credit_amount	-2.887	0.003894	**
Savings_account_bondsA62	1.25	0.21113	
Savings_account_bondsA63	0.938	0.348476	
Savings_account_bondsA64	2.551	0.010729	*
Savings_account_bondsA65	3.607	0.00031	***
Present_employment_sinceA72	0.157	0.875475	
Present_employment_sinceA73	0.445	0.656049	
Present_employment_sinceA74	1.866	0.06211	.
Present_employment_sinceA75	0.669	0.50341	
Installment_rate_in_percentage_of_disposable_income	-3.739	0.000185	***
Personal_status_and_sexA92	0.713	0.47604	
Personal_status_and_sexA93	2.148	0.031718	*
Personal_status_and_sexA94	0.809	0.418448	
Other_debtors_guarantorsA102	-1.063	0.2877	
Other_debtors_guarantorsA103	2.307	0.021072	*
Present_residence_since	-0.055	0.95592	
PropertyA122	-1.111	0.26663	
PropertyA123	-0.824	0.409743	
PropertyA124	-1.721	0.085308	.
Age_in_years	1.576	0.114982	
Other_installment_plansA142	0.299	0.764878	
Other_installment_plansA143	2.703	0.006871	**
HousingA152	1.89	0.058715	.
HousingA153	1.434	0.151657	
Number_of_existing_credits_at_this_bank	-1.436	0.151109	
JobA172	-0.789	0.43016	
JobA173	-0.847	0.397015	
JobA174	-0.724	0.469086	
Number_of_people_being_liable_to_provide_maintenance_for	-1.062	0.288249	
TelephoneA192	1.491	0.13606	

Foreign_workerA202	2.225	0.026095	*

Based on the results of model, we use confusion matrix to estimate the quality. We set the predicted as predicted factor and credit risk as actual factor. Then we plot the results.

Code:

```
library(caret)

conf_matrix <- confusionMatrix(as.factor(predicted_classes), as.factor(german_credit$Credit_risk))

print(conf_matrix)

fourfoldplot(conf_matrix$table, color = c("#CC6666", "#99CC99"),

        conf.level = 0, margin = 1, main = "Confusion Matrix")
```

The result shows as figure 1.

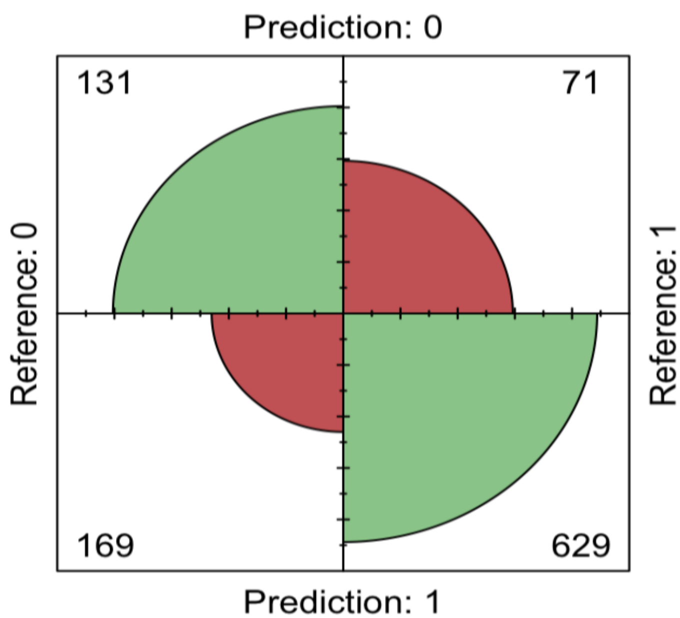


Figure 1 Confusion Matrix

As the figure shows, 131 actual references 0 are predicted as 0, 629 actual references 1 are predicted as 1. The accuracy is 76%. The quality fit is good.

2. Because the model gives a result between 0 and 1, it requires setting a threshold probability to separate between "good" and "bad" answers. In this data set, they estimate that incorrectly identifying a bad customer as good, is 5 times worse than incorrectly classifying a good customer as bad. Determine a good threshold probability based on your model.

To determine a good threshold probability according to the description above, **we need to calculate the cost when incorrect identifying happened in the case that wrong identifying a bad customer is 5 times worse than incorrect identifying a good customer, as the threshold value change.**

First, we calculate the probability of all customers are identified as good, and change the threshold from 0 to1, 0.01. Then we set the **1/5 ratio** that wrong identifying of good over wrong identifying of bad.

Next, we calculate the confusion matrix, then extract FP and FN, with the ratio to calculate the total cost.

Calculate the cost with different thresholds, and find the threshold with the minimum cost, then plot the result.

Last, calculate the confusion matrix and its accuracy with the best threshold.

Code:

```
predictions <- predict(model, type = "response")

# Define a function to calculate the cost based on threshold

calculate_cost <- function(threshold, predictions, actual, cost_fn_fp_ratio = 1/5) {

  predicted_classes <- ifelse(predictions > threshold, 1, 0)


  conf_matrix <- table(Predicted = predicted_classes, Actual = actual)


  FP <- conf_matrix[2, 1]  # Predicted good, actual bad

  FN <- conf_matrix[1, 2]  # Predicted bad, actual good


  total_cost <- FN * cost_fn_fp_ratio + FP


  return(total_cost)

}


thresholds <- seq(0.1, 0.9, by = 0.01)


costs <- sapply(thresholds, calculate_cost, predictions = predictions, actual = german_credit$Credit_risk)
```

```
best_threshold <- thresholds[which.min(costs)]
best_threshold

plot(thresholds, costs, type = "l", col = "blue", lwd = 2,
     xlab = "Threshold", ylab = "Total Misclassification Cost",
     main = "Cost vs. Threshold for Credit Risk Model")
abline(v = best_threshold, col = "red", lty = 2)

predicted_classes_best <- ifelse(predictions > best_threshold, 1, 0)

conf_matrix_best <- table(Predicted = predicted_classes_best, Actual = german_credit$Credit_risk)
print(conf_matrix_best)

accuracy_best <- mean(predicted_classes_best == german_credit$Credit_risk)
print(paste("Accuracy at best threshold:", accuracy_best))
```

The figure shows the best threshold with min cost. **The best threshold is 0.87.**

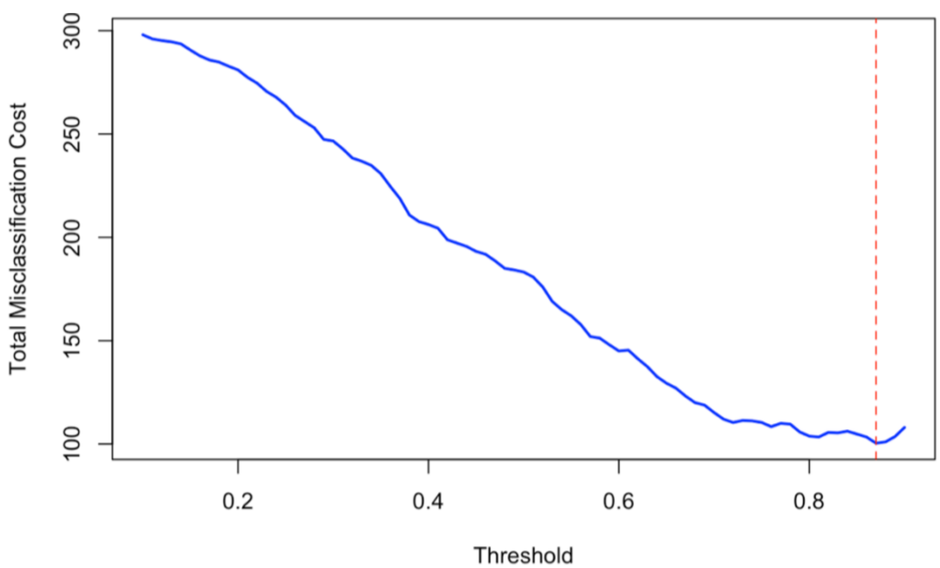


Figure 2 Threshold vs Cost

The figure shows the matrix with the best threshold, the accuracy is **57.4%.** Though the total accuracy is decreasing, the accuracy of bad credit is increasing.

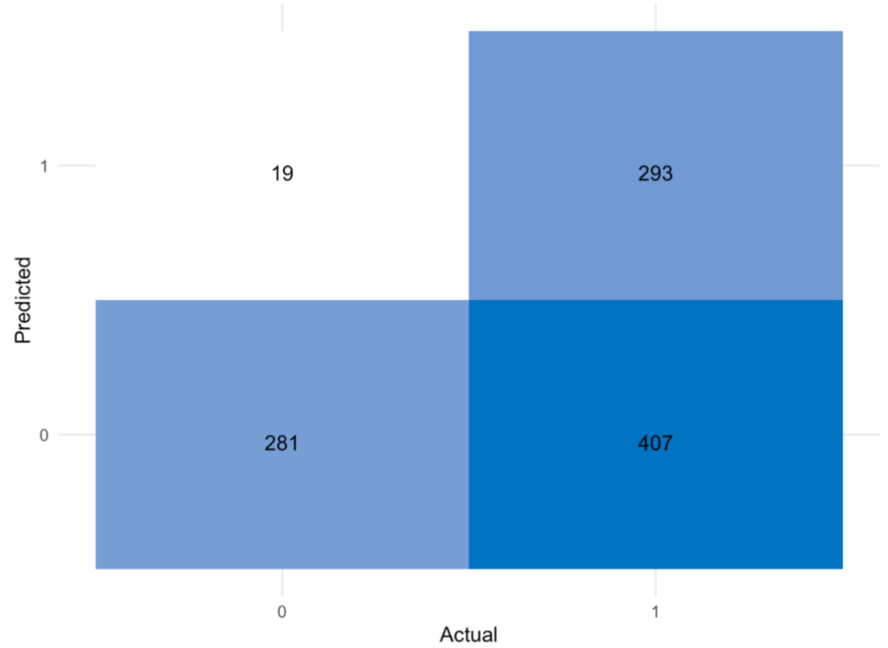


Figure 3 Confusion Matrix