Business Analytics Home Work 3

John Deblase, Sekhar Mekala, Sonya Hong

Saturday, October 15, 2016

Project requirements

The main goal of this project is to perform data analysis of the crime data related to a major city and predict if a neighborhood's crime rate is above the median crime rate of the city, based on a given set of inputs. We are given two data sets: the *training data set* and the *test data set*. The training data has input variables along with the observed response variable. We will use the training data set to train our model, and the predictions obtained on the test data will be submitted as a project deliverable.

Data Exploration

The "target" variable will be the dependent variable, and the remaining variables will be independent variables for our predictive models. The variables significance is given in Figure-1.

Figure-1: Training data set's variables

- zn: proportion of residential land zoned for large lots (over 25000 square feet) (predictor variable)
- indus: proportion of non-retail business acres per suburb (predictor variable)
- chas: a dummy var. for whether the suburb borders the Charles River (1) or not (0) (predictor variable)
- nox: nitrogen oxides concentration (parts per 10 million) (predictor variable)
- rm: average number of rooms per dwelling (predictor variable)
- age: proportion of owner-occupied units built prior to 1940 (predictor variable)
- dis: weighted mean of distances to five Boston employment centers (predictor variable)
- rad: index of accessibility to radial highways (predictor variable)
- tax: full-value property-tax rate per $10,000 (predictor variable)
- ptratio: pupil-teacher ratio by town (predictor variable)
- black: $1000(B_k - 0.63)^2$ where B_k is the proportion of blacks by town (predictor variable)
- lstat: lower status of the population (percent) (predictor variable)
- medv: median value of owner-occupied homes in $1000s (predictor variable)
- target: whether the crime rate is above the median crime rate (1) or not (0) (response variable)

The target variable can have two different values: 0 and 1. The crime rate above the city's median crime rate is represented as 1 and below city's median crime rate is represented as 0.

In the test data set, we have the the same set of variables (given in Figure-1), except the "target" variable. Our goal is to predict this variable's value in the test data as accurately as possible. We will use 5 fold cross validation technique to estimate our models accuracy using the training data.

A summary of all the variables in the training data is given below:

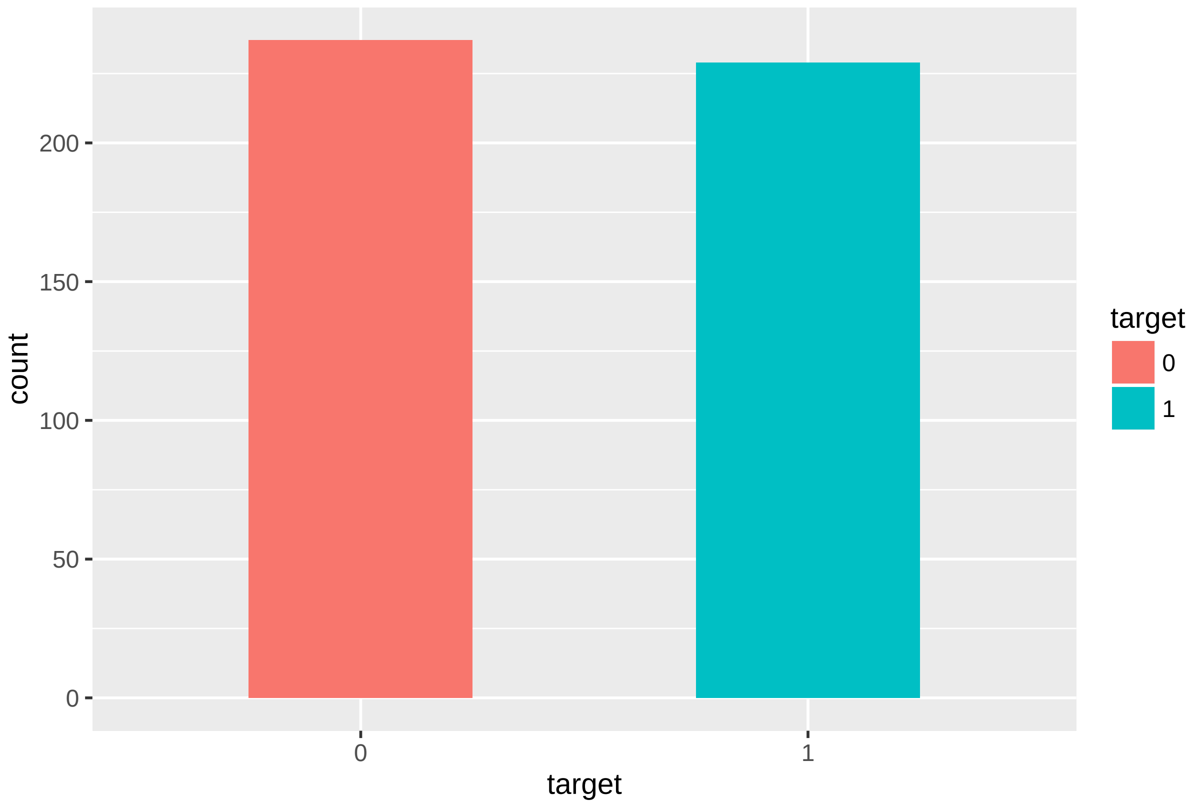
Figure 2: Summary of training data set

```
##        zn             indus             chas              nox
##  Min.   :  0.00   Min.   : 0.460   Min.   :0.00000   Min.   :0.3890
##  1st Qu.:  0.00   1st Qu.: 5.145   1st Qu.:0.00000   1st Qu.:0.4480
##  Median :  0.00   Median : 9.690   Median :0.00000   Median :0.5380
##  Mean   : 11.58   Mean   :11.105   Mean   :0.07082   Mean   :0.5543
##  3rd Qu.: 16.25   3rd Qu.:18.100   3rd Qu.:0.00000   3rd Qu.:0.6240
##  Max.   :100.00   Max.   :27.740   Max.   :1.00000   Max.   :0.8710
##        rm             age              dis              rad
##  Min.   :3.863   Min.   :  2.90   Min.   : 1.130   Min.   : 1.00
##  1st Qu.:5.887   1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00
##  Median :6.210   Median : 77.15   Median : 3.191   Median : 5.00
##  Mean   :6.291   Mean   : 68.37   Mean   : 3.796   Mean   : 9.53
##  3rd Qu.:6.630   3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00
##  Max.   :8.780   Max.   :100.00   Max.   :12.127   Max.   :24.00
##       tax           ptratio         black            lstat
##  Min.   :187.0   Min.   :12.6   Min.   :  0.32   Min.   : 1.730
##  1st Qu.:281.0   1st Qu.:16.9   1st Qu.:375.61   1st Qu.: 7.043
##  Median :334.5   Median :18.9   Median :391.34   Median :11.350
##  Mean   :409.5   Mean   :18.4   Mean   :357.12   Mean   :12.631
##  3rd Qu.:666.0   3rd Qu.:20.2   3rd Qu.:396.24   3rd Qu.:16.930
##  Max.   :711.0   Max.   :22.0   Max.   :396.90   Max.   :37.970
##       medv           target
##  Min.   : 5.00   Min.   :0.0000
##  1st Qu.:17.02   1st Qu.:0.0000
##  Median :21.20   Median :0.0000
##  Mean   :22.59   Mean   :0.4914
##  3rd Qu.:25.00   3rd Qu.:1.0000
##  Max.   :50.00   Max.   :1.0000
```

From the summary information displayed above, we can conclude that we do not have any NA values (unavailable data) in the training data set. The "target" and "chas" variables were inputted as numeric, while they are categorical. We will transform the "target" variable as a categorical variable, and leave the "chas" variable as numeric, treating it as a dummy variable representing whether the suburb faces Charles River.

Below is a distribution of the target variable.

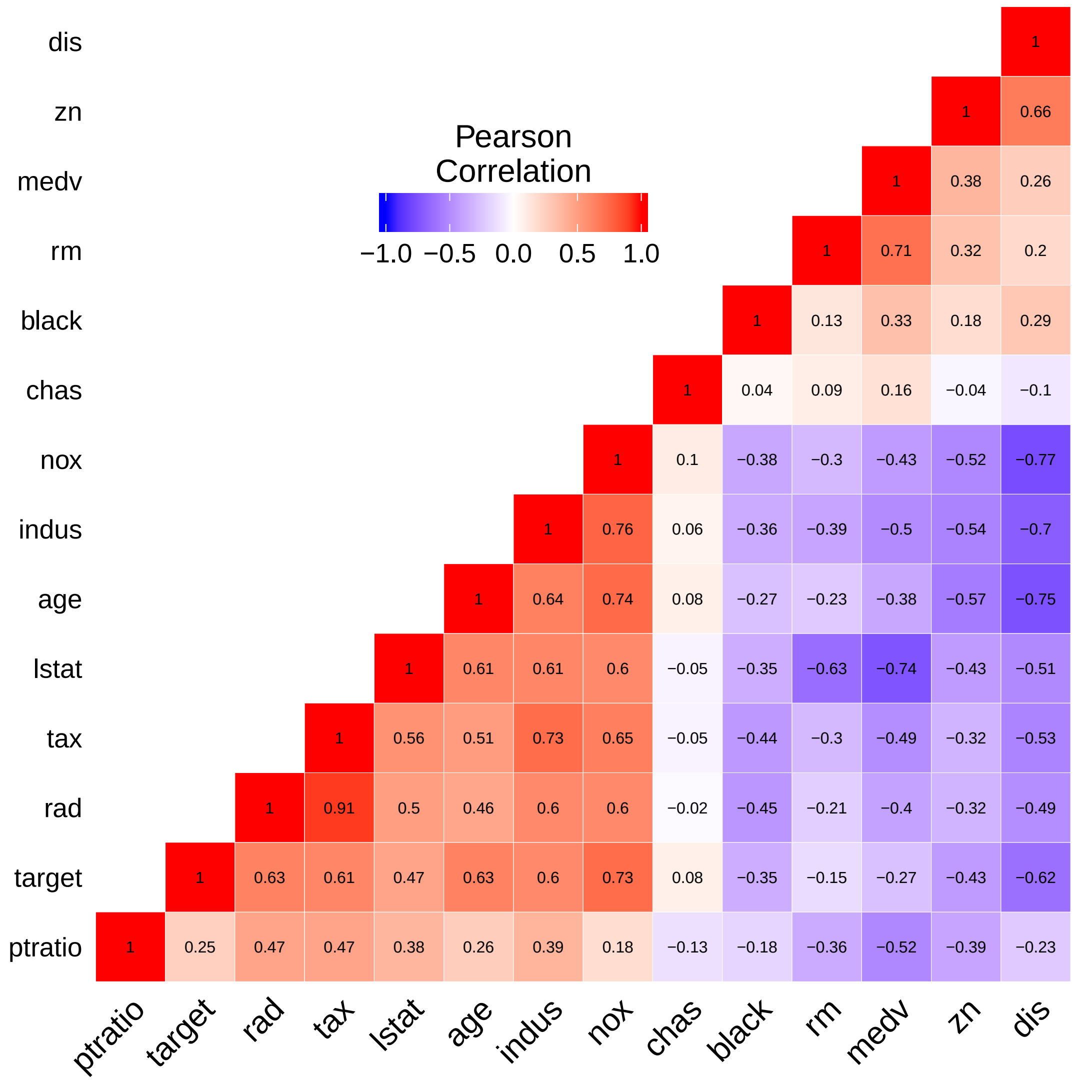
Figure 3: Bar chart of the target variable



The bar chart in Figure-3 shows the target variable as having almost equal distribution between “0” and “1”, and the data is not imbalanced. This also suggests that we can safely use the probability threshold as 0.5 to determine if an observation is positive or negative.

The following heat map shows the correlation between the variables of the test data set.

Figure 4: Heat map showing correlations



From Figure-4, we can identify that the target variable is strongly associated with the "nox" variable, the Nitrogen Oxide (NO_2) Concentration. The relationship between NO_2 concentration and crime rate seems peculiar until we see that "nox" is also strongly correlated to "indus", "dis", "age" variables also. So "nox" variable seems to be indirectly representing the relationship of "indus", "age" and "dis" variables with the "target" variable. Due to the correlation between "nox", "indus", "tax", "dis" and "age" variables we might consider using only one of these variables or combining all of them together to form a new variable in our models.

The target variable is negatively correlated with the "dis" variable, a measure related to the distance to 5 employment centers. The correlation of other predictor variables is not strong with the target variable. While building the models, we can identify and exclude the unnecessary variables in the variable selection process. This process would also elminate the correlated variables.

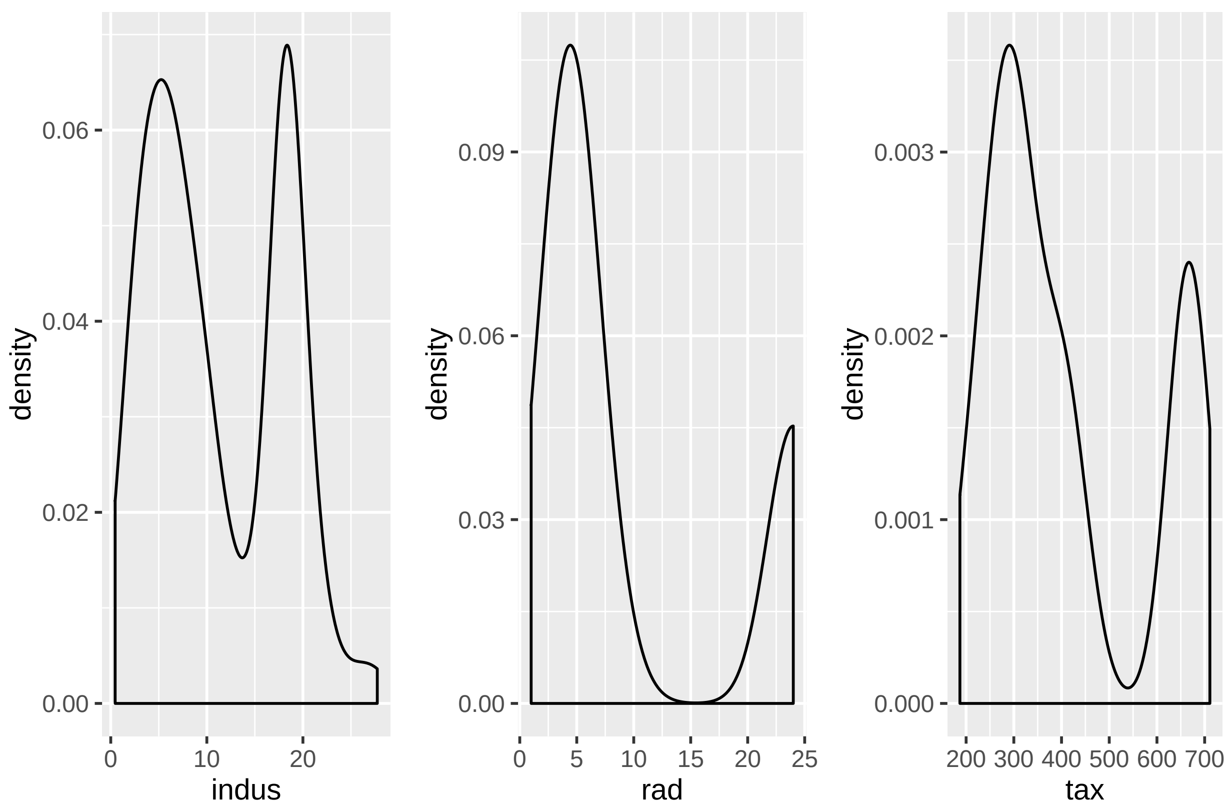
Data Preparation

From Figure-2 we identified that none of the variables in the training data set have unavailable data, so no observations need to be eliminated.

The density plots of all the predictor variables are given in Figure-A.1 (in Appendix-A). From Figure-A.1, we can identify that three variables "indus", "rad" and "tax" have bimodal distributions. Ideally we would like these potential predictor variables to be normally distributed in order to improve our model's accuracy.

For convenience, we are showing the density plots "tax", "indus" and "rad" variables in Figure-5 given below. For all variable's density plots, please refer to Appendix-A, Figure-A.1.

Figure 5: Density plots of indus, rad and tax variables



We will create three dummy variables,"indus_dummy", "rad_dummy" and "tax_dummy", to logically divide the values into 2 groups in "indus", "rad" and "tax" variables respectively. This will separate the data in these variables into 2 groups, so that each group will have an approximately normal distribution. This transformation will also need to be applied to the test data as well. The dummy variables will have the following intrepretation:

- If *indus* variable's value is less than 13, then *indus_dummy* will have a 0, else 1
- If *rad* variable's value is less than 15, then *rad_dummy* will have a 0, else 1
- If *tax* variable's value is less than 550, then *tax_dummy* will have a 0, else 1

A set of sample rows from the transformed training data set are displayed below:

Table 1: Sample records from training data after adding dummy variables (continued below)

zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat
0	19.58	0	0.605	7.929	96.2	2.046	5	403	14.7	369.3	3.7
0	19.58	1	0.871	5.403	100	1.322	5	403	14.7	396.9	26.82
0	18.1	0	0.74	6.485	100	1.978	24	666	20.2	386.7	18.85
30	4.93	0	0.428	6.393	7.8	7.035	6	300	16.6	374.7	5.19
0	2.46	0	0.488	7.155	92.2	2.701	3	193	17.8	394.1	4.82
0	8.56	0	0.52	6.781	71.3	2.856	5	384	20.9	395.6	7.67

medv	target	indus_dummy	rad_dummy	tax_dummy
50	1	1	0	0
13.4	1	1	0	0
15.4	1	1	1	1
23.7	0	0	0	0
37.9	0	0	0	0
26.5	0	0	0	0

We will not be doing any further transformations on the other variables in order to keep the model as simple and interpretable as possible. We can revisit the need of transformations, if we get poor cross validation results.

Model building

We will build three models with the following criteria, and select the model that has the least 5 fold cross validation error. All the three models are based on the *logistic regression*.

Model-1: Build a model with all the predictor variables (except the 3 new dummy variables), use backward variable selection with the *stepAIC()* function of MASS package, and finally build the model with just the necessary variables obtained from the variable selection method.

Model-2: Build a model with dummy variables created for tax, rad and indus. Again use *stepAIC()* function to choose the necessary variables, and finally build a model with just the necessary variables.

Model-3: Using the variables identified in *Model-1*, use a quadratic model, and select the required variables using the *stepAIC()* function.

All three models will be evaluated using the 5-fold cross validation technique, and the model that gets the least cross validation error will be finally used to predict the crime rate of test observations.

Building *Model-1*:

Using the *glm()* function of MASS library, we obtained the following logistic model for classification:

Table:2 Coefficients and P-Values of Model-1

	Estimate	Std_err	z-value	p-value
(Intercept)	-36.8395209	7.0287258	-5.2412801	0.0000002
zn	-0.0617200	0.0344101	-1.7936591	0.0728676
indus	-0.0725805	0.0485463	-1.4950783	0.1348940
chas	1.0323518	0.7596271	1.3590244	0.1741389
nox	50.1595127	8.0495028	6.2313802	0.0000000
rm	-0.6921451	0.7414309	-0.9335262	0.3505484
age	0.0345215	0.0138827	2.4866573	0.0128950
dis	0.7657946	0.2344072	3.2669407	0.0010872
rad	0.6630152	0.1651346	4.0149992	0.0000594
tax	-0.0065933	0.0030643	-2.1516854	0.0314221
ptratio	0.4422171	0.1322340	3.3442000	0.0008252
black	-0.0130936	0.0066798	-1.9601834	0.0499744
lstat	0.0475714	0.0545078	0.8727446	0.3828023
medv	0.1997343	0.0710223	2.8122739	0.0049193

We can observe that only the following coefficients have a p-value of less than 1%.

Table-3: Coefficients of Model-1 which are having a p-value of less than 0.01

	Estimate	Std_err	z-value	p-value
(Intercept)	-36.8395209	7.0287258	-5.241280	0.0000002
nox	50.1595127	8.0495028	6.231380	0.0000000
dis	0.7657946	0.2344072	3.266941	0.0010872
rad	0.6630152	0.1651346	4.014999	0.0000594
ptratio	0.4422171	0.1322340	3.344200	0.0008252
medv	0.1997343	0.0710223	2.812274	0.0049193

We cannot depend on the p-value of the coefficients and eliminate the variables that have a big p-value, due to the chance of a Type-1 error. For this reason we will use the backward selection method using the *stepAIC()* function, which produces the following model.

$$Model_1 = \frac{e^{f_1(x)}}{1 + e^{f_1(x)}}$$

where,

$$f_1(x) = -36.364060 - 0.059091zn - 0.067194indus + 1.217987chas +$$
$$48.468529nox + 0.031961age + 0.703484dis + 0.620491rad - 0.006392tax +$$
$$0.385927ptratio - 0.012935black + 0.119666medv$$

We can observe that *stepAIC()* function has included some of the high p-valued variables (see Table-2) in the final model. This might be due to the fact that some of the variables in Table-2 have high p-values just by chance. The *stepAIC()* function has excluded two variables: "rm" and "lstat". These 2 variables have very high p-values, and they are indeed not significant to predict the target value.

From *Model-1*, we can infer that "zn", "indus", "tax" and "black" have negative coefficients. This means a one unit increase in any of these variables (keeping all other variables constant) will result in a decrease in the probability that $target = 1$. The "nox" variable's coefficient is so large that a one unit increase in "nox" value (keeping all other variables constant) will result in a significant increase in the probability that $target = 1$.

Building *Model-2*:

Our second model is produced using the same process as above, but uses the dummy variables created above:

$$\frac{e^{f_2(x)}}{1 + e^{f_2(x)}}$$

where

$$f_2(x) = -38.39452 - 0.06064zn - 1.45342indus\_dummy + 1.55134chas + 52.01717nox$$

$$-1.15061rm + 0.04066age + 0.92588dis + 0.54515rad - 4.43821tax\_dummy + 0.43975ptratio - 0.01328black + 0.24761medv$$

Model-2 does not have the "indus" variable and "tax" variable, but it has the dummy variables related to them("indus_dummy" and "tax_dummy" respectively). Also a new variable "rm" has been included in *Model-2*, while this variable is not included in *Model-1*. The coefficient of "rm" variable is negative, and a one unit increase in "rm" (keeping all other variables constant) will result in decrease of the probability that the target variable assumes 1. The "nox" variable has a higher coefficient, similar to the first model.

Building *Model-3*:

Model-3 is a logistic regression using the variables of the first model all raised to the second power. Building a logistic model, and using the same backwards selection method, gives us the following quadratic model:

$$\frac{e^{f_3(x)}}{1 + e^{f_3(x)}}$$

where

$$f_3(x) = 4.927 + 235.067nox + 83.840nox^2 + 25.218age + 14.879age^2 + 9.926dis - 43.337dis^2$$
$$494.385rad + 54.792rad^2 - 388.883tax - 136.723tax^2 + 38.181ptratio - 4.147ptratio^2 + 9.933black$$
$$-45.661black^2 + 34.286medv + 16.939medv^2$$

We can see that all the variables in *Model-1* were also included in *Model-3* (by the varible selection function *stepAIC()*), except the "chas" variable. All the variables are also raised to the power 2, and the coefficients of the variables are not significantly different. In *Model-1* and *Model-2* the coefficients of "nox" are significantly higher than the other variables coefficients.

Model selection

In summary we have obtained the following 3 models:

Model-1

$$Model_1 = \frac{e^{f_1(x)}}{1 + e^{f_1(x)}}$$

where,

$$f_1(x) = -36.364060 - 0.059091zn - 0.067194indus + 1.217987chas +$$
$$48.468529nox + 0.031961age + 0.703484dis + 0.620491rad - 0.006392tax +$$
$$0.385927ptratio - 0.012935black + 0.119666medv$$

Model-2

$$\frac{e^{f_2(x)}}{1 + e^{f_2(x)}}$$

where

$$f_2(x) = -38.39452 - 0.06064zn - 1.45342indus\_dummy + 1.55134chas + 52.01717nox$$
$$-1.15061rm + 0.04066age + 0.92588dis + 0.54515rad - 4.43821tax\_dummy + 0.43975ptratio - 0.01328black + 0.24761medv$$

Model-3

$$\frac{e^{f_3(x)}}{1 + e^{f_3(x)}}$$

where

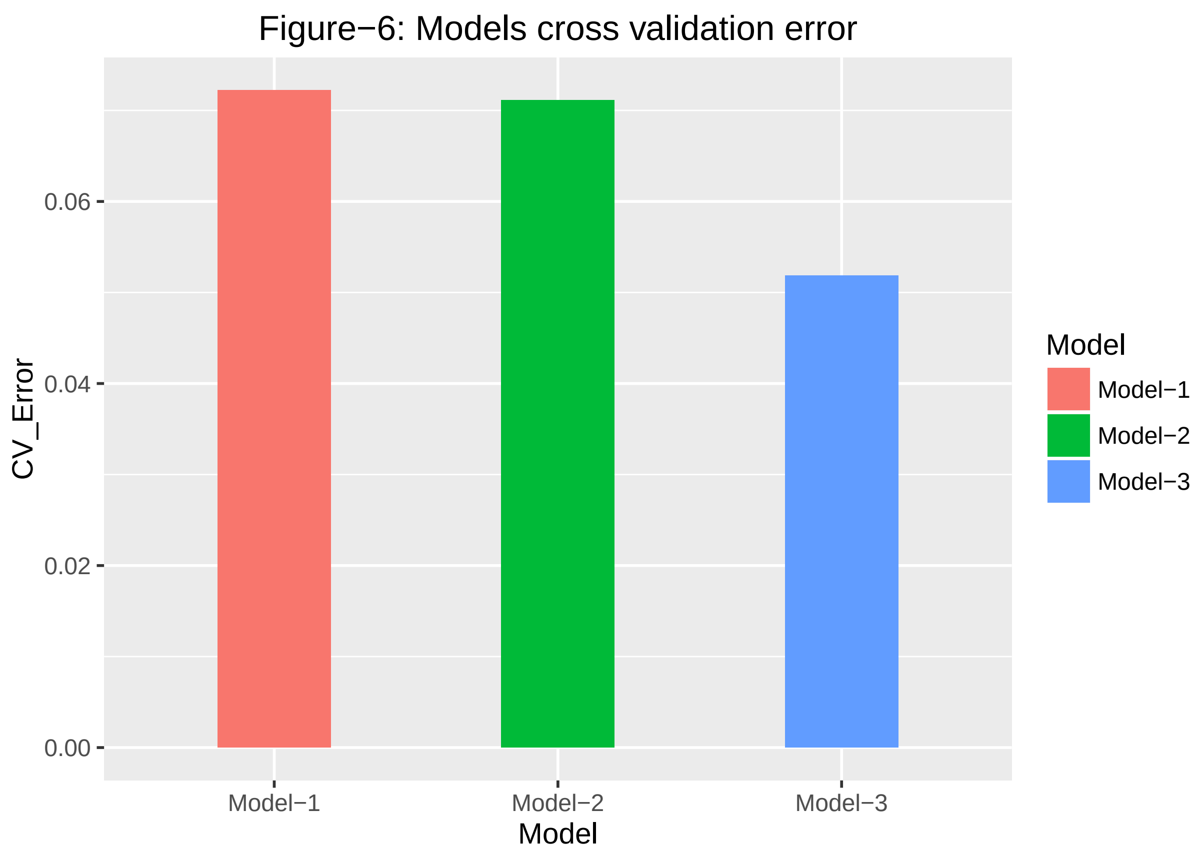
$$f_3(x) = 4.927 + 235.067nox + 83.840nox^2 + 25.218age + 14.879age^2 + 9.926dis - 43.337dis^2$$
$$494.385rad + 54.792rad^2 - 388.883tax - 136.723tax^2 + 38.181ptratio - 4.147ptratio^2 + 9.933black$$
$$-45.661black^2 + 34.286medv + 16.939medv^2$$

We will evaluate the performance of these 3 models first using the 5 fold cross validation technique on the training data using the *cv.glm()* function of "boot" library. The model with the lowest CV error will then be further analyzed using *confusionMatrix()* and *plot.roc()* to make sure the model's accuracy metrics and the ROC curve are within acceptable thresholds. Specifically if the selected model's accuracy is below 0.85, only then will we consider models that had inferior CV error rates.

Using the *cv.glm()* function, we obtained the following cross validation errors given below in Figure-6.

Table-4 Cross validation errors

Model	CV_Error
Model-1	0.0722081
Model-2	0.0711133
Model-3	0.0518364



The above plot shows that *Model-3* has the lowest cross validation error and we will therefore select this model to be checked for accuracy metrics. Specifically if this model's accuracy metric had been below 0.85, we would have considered building more models by transforming independent variables and/or trying other methods such as LASSO and Ridge regression.

The performance of *Model-3* is now further evaluated using the following metrics:

- Confusion matrix
- Accuracy
- Classification error rate
- Precision
- Sensitivity
- Specificity
- F1 score
- AUC

The confusion matrix and resulting metrics for the predictions obtained using *Model-3* are displayed below:

Figure-7: Confusion matrix and other performance metrics of *Model-3*

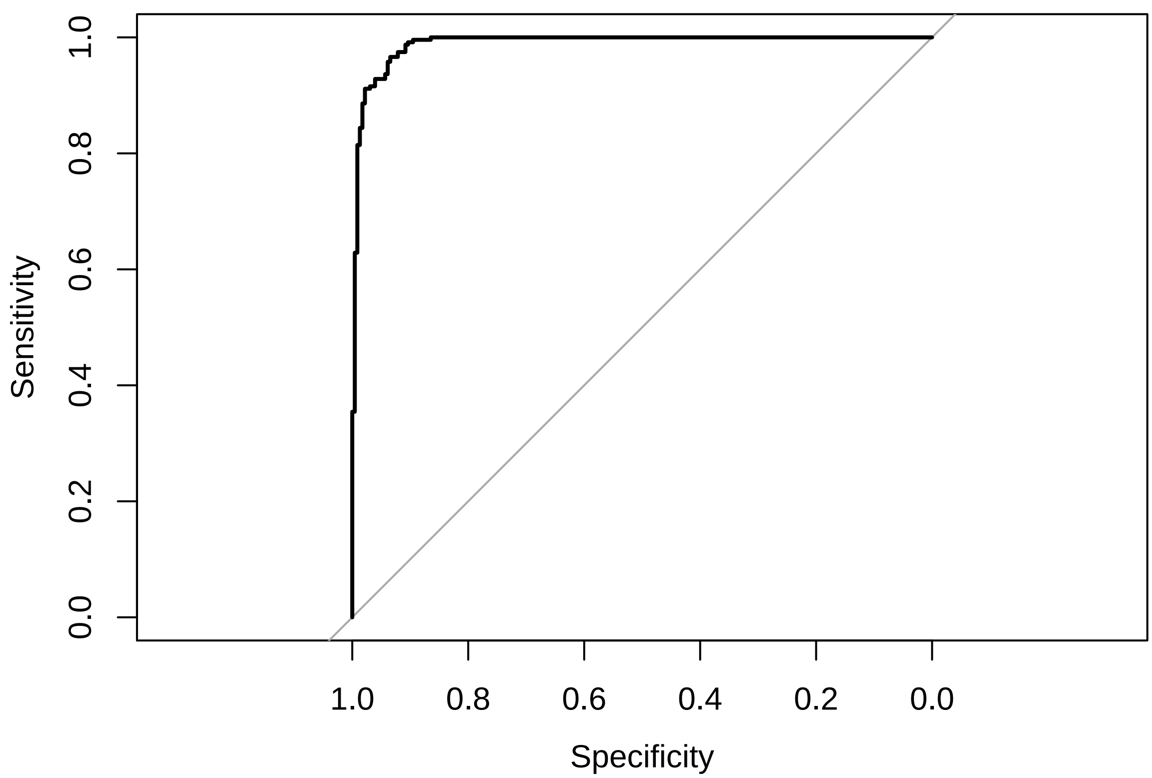
```
## Confusion Matrix and Statistics
##
##          actual
## predicted   0   1
##         0 229  16
```

```
##          1   8 213
##
##                 Accuracy : 0.9485
##                   95% CI : (0.9243, 0.9667)
##      No Information Rate : 0.5086
##      P-Value [Acc > NIR] : <2e-16
##
##                    Kappa : 0.8969
##   Mcnemar's Test P-Value : 0.153
##
##              Sensitivity : 0.9301
##              Specificity : 0.9662
##           Pos Pred Value : 0.9638
##           Neg Pred Value : 0.9347
##               Prevalence : 0.4914
##           Detection Rate : 0.4571
##     Detection Prevalence : 0.4742
##        Balanced Accuracy : 0.9482
##
##         'Positive' Class : 1
##
```

The accuracy of *Model-3* is 0.9485, which is well above our threshold at 0.85. The F1 Score is 0.9466667 meaning that the balance between precision and sensitivity is excellent.

Model-3 has obtained an AUC of 98.93, and the curve is displayed in Figure-8.

Figure 8: AUC for Model−3



```
##
## Call:
## roc.default(response = train_df$target, predictor = prob, levels = rev(levels(as.factor(train_df$targ
##
## Data: prob in 229 controls (train_df$target 1) > 237 cases (train_df$target 0).
## Area under the curve: 0.9893
```

Model-3 is finally used to estimate the probability of *target* variable being 1 on the test data, and the predicted output, along with the estimated probabilities are written to a file named "test_result.csv", which has been submitted along with this write-up.

Future work

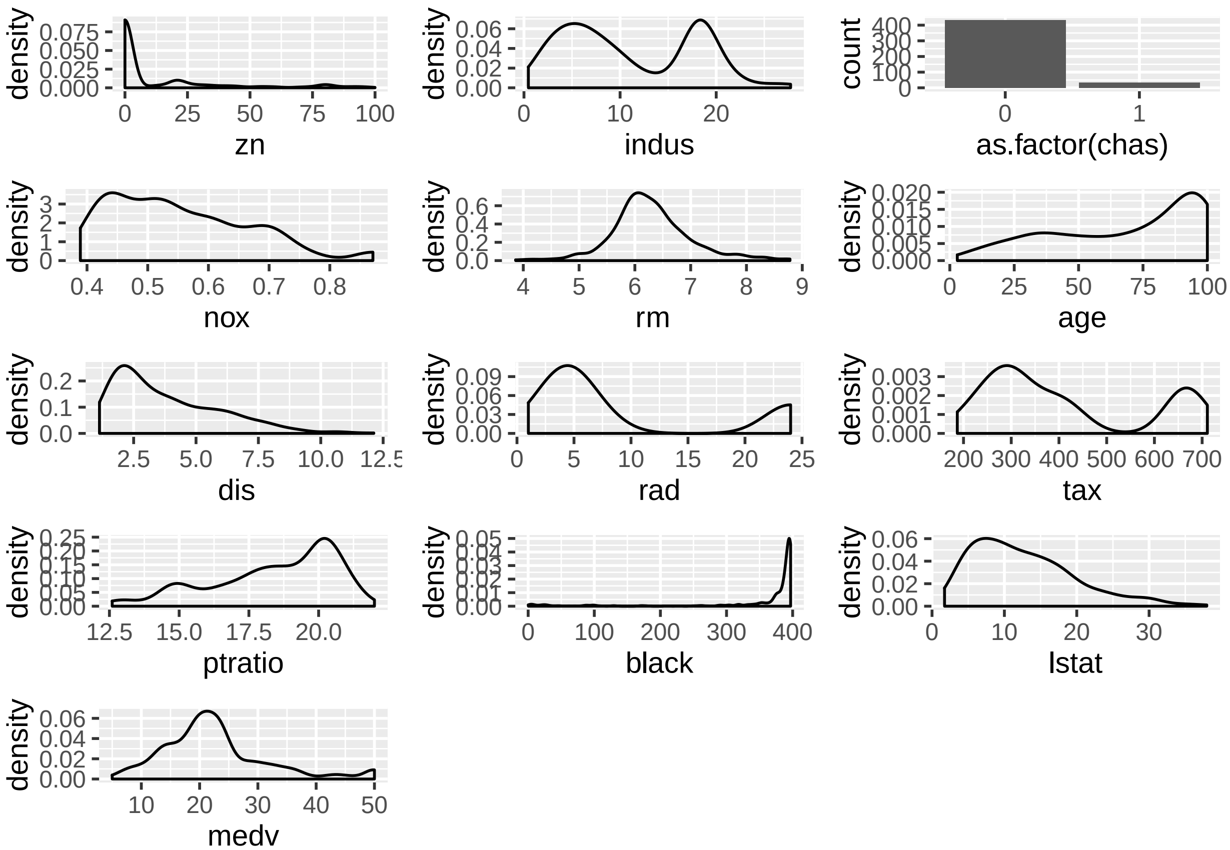
We obtained an accuracy of 94.85%, which suggests that the logistic regression with quadratic variables is a good model to perform predictions. However we would like to test the following in future:

1. Implement and test non-parametric methods like KNN, random forests and SVM to perform classification.
2. Implement and test Linear Discriminant Analysis.
3. Implement and test Quadratic Discriminant Analysis.
4. The coefficient of "nox" variable in *Model-1* and *Model-2* is pretty high when compared to other variables coefficient. We would like to try using LASSO and Ridge regression to diminish the variable coefficients and test the performance of the resulting models.

Appendix-A

The density plots of all the independent variables are displayed below. We did not transform any of these variables:

Figure A-1: Density plots of all the independent variables



Appendix-B

We used the following R code to implement and test the models:

```
library(knitr)
library(ggplot2)
library(reshape2)
library(gridExtra)
library(boot)
library(pander)
library(gridExtra)
library(MASS)
library(caret)
library(pROC)
```

Data Prep

```
setwd("C:/Users/Sekhar/Documents/R Programs/Business Analytics/HW3")

train_df <- read.csv("crime-training-data.csv")
test_df <- read.csv("crime-evaluation-data.csv")
#head(train_df)
#head(test_df)
```

```
summary(train_df)
#summary(test_df)
```

```
train_df$target <- as.factor(train_df$target)

ggplot(data=train_df,aes(target,fill=target))+
  geom_bar(width=0.5)+
  labs(title="Figure 3: Bar chart of the target variable")
```

Data Exploration

```
g1 <- ggplot(data=train_df,aes(indus))+
geom_density()

g2 <- ggplot(data=train_df,aes(rad))+
geom_density()

g3 <- ggplot(data=train_df,aes(tax))+
geom_density()

grid.arrange(g1,g2,g3, ncol=3,
             top="Figure 5: Density plots of indus, rad and tax variables")
```

Dummy Variable Creation

```
test_df$target <- rep(0,nrow(test_df))
test_df$target  <- as.factor(test_df$target )
test_df$indicator <- rep("Test",nrow(test_df))

train_df$indicator <- rep("Train",nrow(train_df))

df <- rbind(train_df,test_df)

df$indus_dummy <- ifelse(df$indus<13,0,1)
df$rad_dummy <- ifelse(df$rad<15,0,1)
df$tax_dummy <- ifelse(df$tax<550,0,1)
train_df <- df[df$indicator == "Train",-15]
test_df <- df[df$indicator == "Test",-15]

#kable(head(train_df))

pander(head(train_df), split.table = 120,
style = 'rmarkdown',
caption="Sample records from training data after adding dummy variables")
```

Model Building

```
glm.fit1 <- glm(data=train_df,target~zn+indus+chas+nox+rm+age+dis+
                  rad+tax+ptratio+black+lstat+medv,family="binomial")
display_df <- as.data.frame(summary(glm.fit1)$coefficients)
names(display_df) <- c("Estimate","Std_err", "z-value","p-value")

kable(display_df)

#pander(display_df, split.table = 120,
#style = 'rmarkdown',
#caption="Coefficients and P-Values of Model-1")
#stepAIC(glm.fit1)
```

```
display_df[display_df[,4] <= 0.01,]
```

```
glm.fit3 <- glm(data=train_df,target~
                  poly(zn,2)+poly(indus,2)+poly(nox,2)+poly(age,2)+poly(dis,2)+
                  poly(rad,2)+poly(tax,2)+chas+
                  poly(ptratio,2)+ poly(black,2)+poly(medv,2)
                ,family="binomial")
#summary(glm.fit3)

#stepAIC(glm.fit3)
#cv.glm(train_df,glm.fit3,K=5)$delta[1]
```

Evaluation

```r
set.seed(100)
cv_1 <- cv.glm(train_df,glm.fit1,K=5)$delta[1]
cv_2 <- cv.glm(train_df,glm.fit2,K=5)$delta[1]
cv_3 <- cv.glm(train_df,glm.fit3,K=5)$delta[1]

#ggplot(train_df, aes(x = factor(cyl), y = mmpg))+
#  geom_bar(stat = "identity")
display_df <- data.frame(Model=c("Model-1","Model-2","Model-3"),
                         CV_Error=c(cv_1,cv_2,cv_3))
kable(display_df)

ggplot(data=display_df,aes(x=Model,y=CV_Error,fill=Model))+
  geom_bar(stat="identity",width=0.4)+
  labs(title="Figure-6: Models cross validation error")
```

```r
actual <- train_df$target
prob <- predict(glm.fit3,type="response")
predicted <- ifelse(prob>=0.5,1,0)

conf_matrix <- table(predicted,actual)
#print(conf_matrix)
confusionMatrix(conf_matrix,positive = "1")
```

```r
roc_obj = roc(response=train_df$target,predictor=prob,
              levels=rev(levels(as.factor(train_df$target))))

plot.roc(roc_obj,main="Figure 8: AUC for Model-3")
```

Test Data Evaluation

```r
test_df <- read.csv("crime-evaluation-data.csv")
prob <- predict(glm.fit3,test_df,type="response")
target <- ifelse(prob>=0.5,1,0)

test_df$probability <- prob
test_df$target <- target

write.csv(test_df,file="test_result.csv")
```