Stat4001 Project report – Group 14

Fung Chi In 1155079115 Fung Chun Yin 1155077980 Liu Lok Yi 1155092579

Poon Ling 1155077345 Qin Keqian 1155077950

# House sale price models in Ames, Iowa

Data preparation

As ridge regression and LASSO requires standardized data, so we build our model on both original data and standardized data. The result that result higher pseudo R square is reported. Also, the missing value in ‘LotFrontage’ column is imputed using ‘missForest’ package, a imputation package using random forest model to handle missing data.

Linear Regression Model

*1. Data preparation*

In the linear regression model, we perform the analysis by using three methods. They are stepwise, step backward and step forward model. All models use 5-fold cross validation to train the model. We use the lappy function to change the internal structure of LotShape and OverallCond variables as factor.

*2. Assumptions of Linear Regression*

By using the lm function, we generate a linear regression model. Figure 1 shows the plot of the model, the top-left and bottom-left plots show that the fitted valves along x with no increasing or decreasing trend. And the points in the Normal Q-Q plot lie exactly on the line which means the data is in normal distribution.

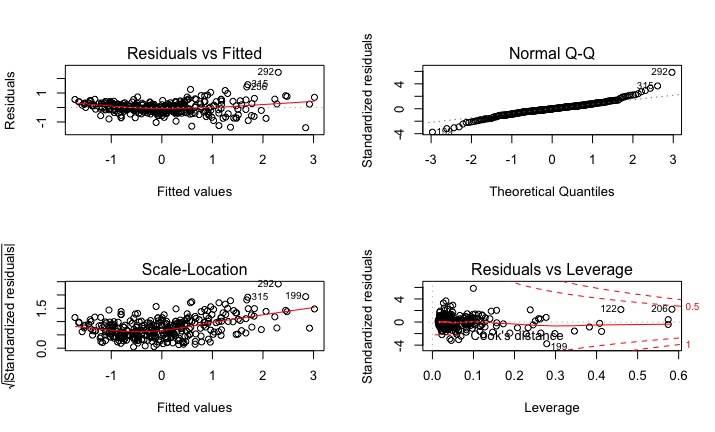


Figure 1: Plots to check the assumption of linear regression model

*3. Model Summary*

* **Model 1: Perform cross-validation with Stepwise Selection**

By using the trainControl with method = “cv” and train function with method = “leapSeq”, we produce a cross validation linear regression model with stepwise function. Table 1 shows the summary of the model.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tested R-square | Bias | Variance | Training Error | Testing Error | CV Error |
| 0.7760851 | 0.9939759 | 0.7202444 | 0.2155495 | 0.2225661 | 0.2847025 |

Table 1: Summary of cross validation linear regression model with stepwise function

In Figure 2, it shows the residual plot of the stepwise model. The data points focus around 0 of x and y axis which means the performance of model prediction is good which the residual is around 0. However, we can recognize there contains some outliers in the datasets. There is a data point locates on the right-top which is an outlier.

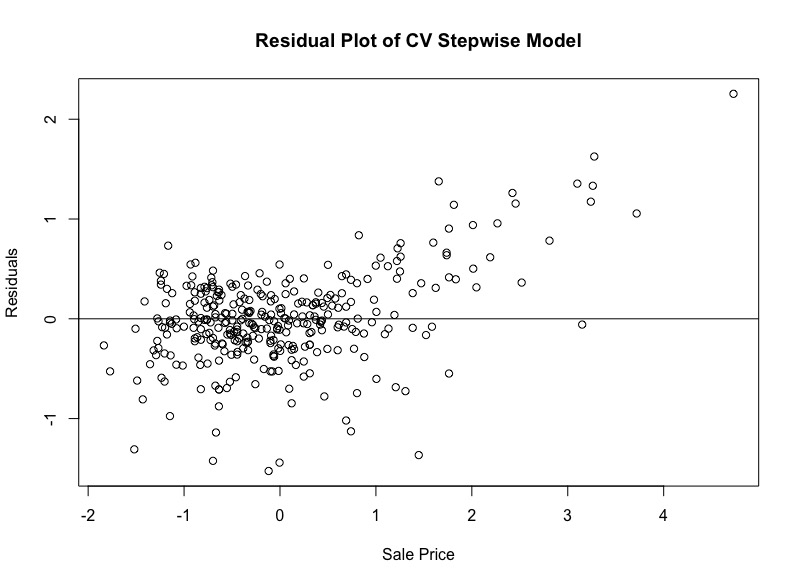


Figure 2: Residual Plot of cross-validation with Stepwise Selection Linear Regression Model

In Figure 3, the left plot shows a histogram of standardize residual of prediction. The plot is normally distributed. However, the right plot shows a u shape plot. So, the model maybe following a quadratic relationship.

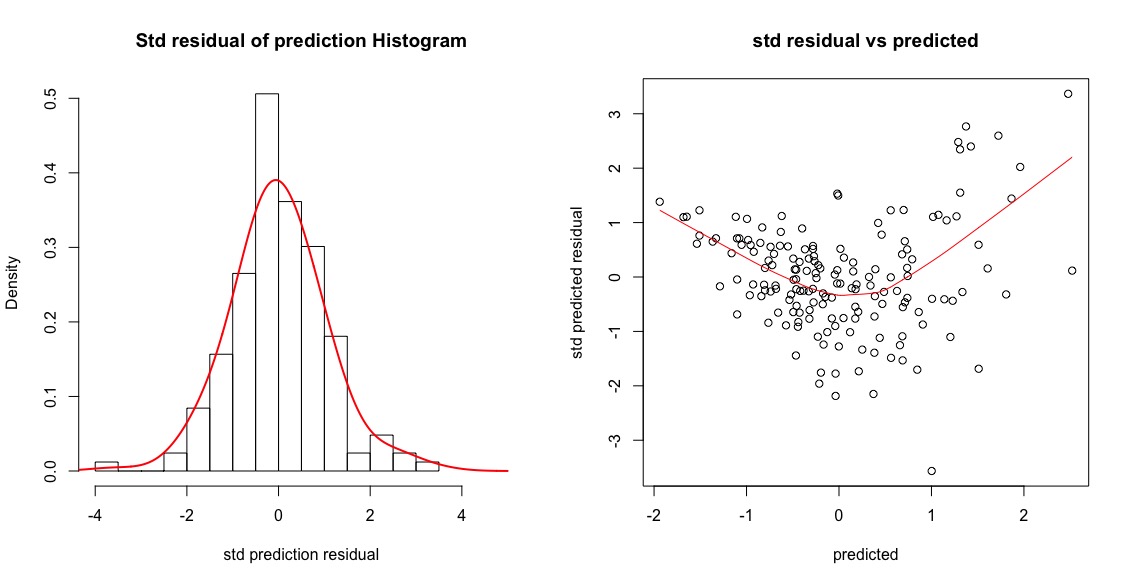


Figure 3: Std residual of prediction Histogram and standardized residual vs predicted plot of Stepwise regression model

In Figure 4, it shows the importance of coefficients of stepwise model. The variables “GrLivArea”, “GarageArea” and “TotalBsmtSF” shows a significant importance of the model. But the categorical variables “LotShape” and “OverallCond” have an unimportant effect on the model.

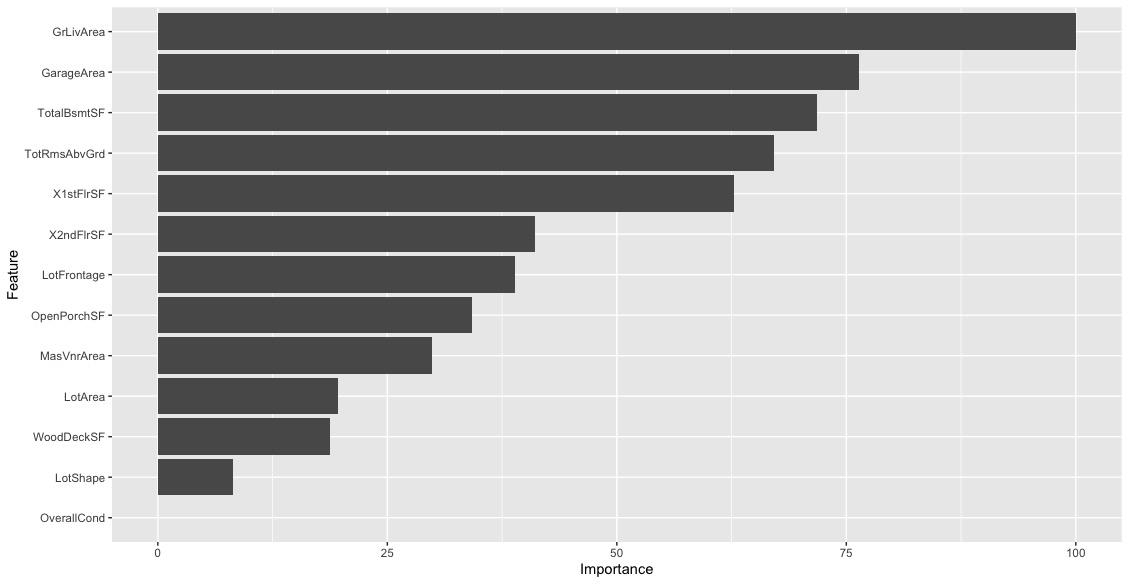


Figure 4: Importance of coefficients of the Stepwise model

* **Model 2: Perform cross-validation with Step backward**

By using the trainControl with method = “cv” and train function with method = “leapBackward”, we produce a 5-fold cross validation linear regression model with step backward function. Table 2 shows the summary of the model.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tested R-square | Bias | Variance | Training Error | Testing Error | CV Error |
| 0.805222 | 0.9939829 | 0.7481322 | 0.185525 | 0.1936047 | 0.2940694 |

Table 2: Summary of cross validation linear regression model with step backward function

In Figure 5, it shows the residual plot of the step backward model. The residual plot is similar as stepwise model, the data points focus around 0. Both stepwise and step backward model have a good performance.

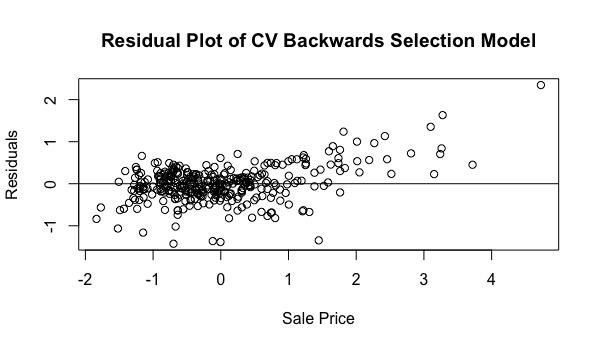


Figure 5: Residual Plot of cross-validation with Step Backwards Selection Linear Regression Model

In Figure 6, the left plot shows a histogram of standardize residual of prediction. The plot is normally distributed which means the standardized residual follow normal distribution. However, the right plot shows a u shape again, which means the model maybe follow a non-linear relationship.

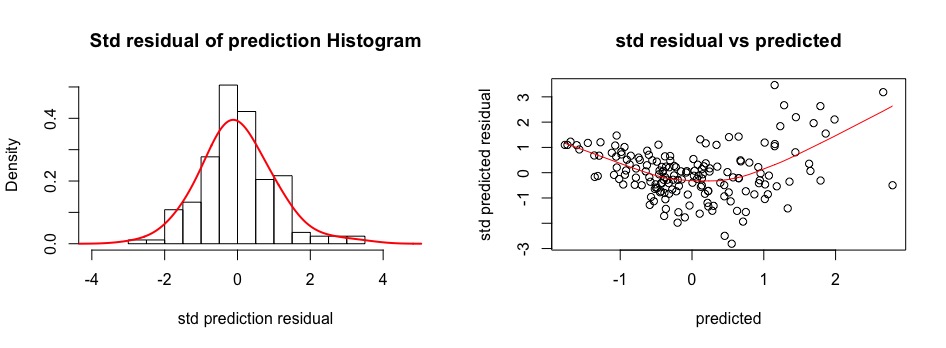


Figure 6: Std residual of prediction Histogram and standardized residual vs predicted plot of step backward regression model

In Figure 4, it shows the importance of coefficients of step backward model. The variables “GrLivArea”, “GarageArea” and “TotalBsmtSF” shows a significant importance of the model. But the categorical variables “LotShape” and “OverallCond” have an unimportant effect on the model. This is similar as the stepwise model.

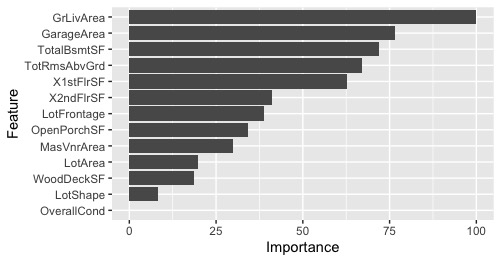


Figure 7: Importance of coefficients of step backward model

* **Model 3: Perform cross-validation with step forward**

By using the trainControl with method = “cv” and train function with method = “leapForward”, we produce a cross validation linear regression model with step forward function. Table 3 shows the summary of the model.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tested R-square | Bias | Variance | Training Error | Testing Error | CV Error |
| 0.7981016 | 0.994051 | 0.7420201 | 0.1836115 | 0.2006822 | 0.2714166 |

Table 3: Summary of cross validation linear regression model with step forward function

In Figure 8, similar as the previous model the residual plot shows a good performance of the Forwards Selection Model due to the residual points focus on 0.

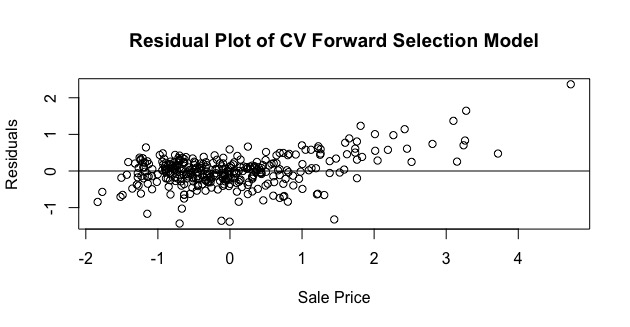


Figure 8: Residual Plot of cross-validation with Step Forward Selection Linear Regression Model

In Figure 9, the left plot shows a histogram of standardize residual of prediction. The plot is normally distributed which means the standardized residual follow normal distribution. However, the right plot shows a u shape again. This kind of pattern shows that the model has room to improve or the data is not perfectly following linear relationship.

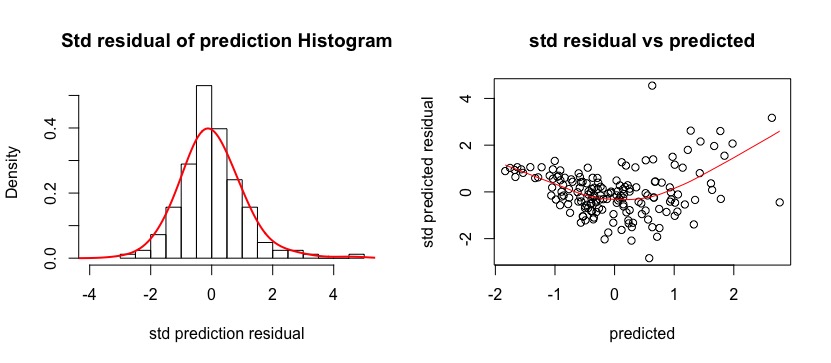


Figure 9: Std residual of prediction Histogram and standardized residual vs predicted plot of step forward regression model

In Figure 10, it shows the importance of coefficients of step forward model. The variables “GrLivArea”, “GarageArea” and “TotalBsmtSF” shows a significant importance of the model. But the categorical variables “LotShape” and “OverallCond” have an unimportant effect on the model. This is similar as the stepwise and step backward model.

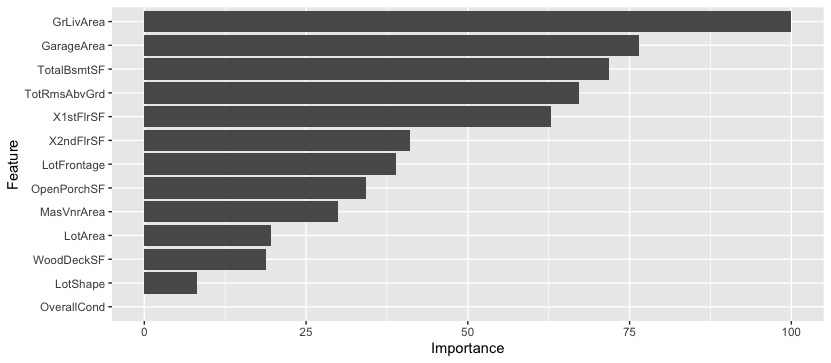


Figure 10: Importance of coefficients of step forward model

*4. Conclusion*

To conclude, step forward linear regression model performs better than the other two models. Since the training error and cv error are the lowest among three models. Ans the tested r-square of the model is the middle one, instead of the lowest one. A model with a higher tested R-square perform better. So, step forward model has the best performance.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Tested R-square | Bias | Variance | Training Error | Testing Error | CV Error |
| Stepwise | 0.7760851 | 0.9939759 | 0.7202444 | 0.2155495 | 0.2225661 | 0.2847025 |
| Step Backward | 0.805222 | 0.9939829 | 0.7481322 | 0.185525 | 0.1936047 | 0.2940694 |
| Step Forward | 0.7981016 | 0.994051 | 0.7420201 | 0.1836115 | 0.2006822 | 0.2714166 |

Table 4: Summary of three linear regression models

Ridge Regression Model

*1. Generate Ridge Regression Model*

In the ridge regression model, we use cv.glmnet function with alpha = 0 to generate our model. We do a 5-fold cross validation to train the model. By using a self-defined grid with size equal to 100, we set lambda = grid in the function. Then, we use plot function to plot the cross-validation error against log(lambda). When log(lambda) increases, the error increases. After training of model, we can get a suitable value for lambda which is 0.1232847. And the best lambda is come from the 91 in the grid. This lambda gives the smallest cross validation error.

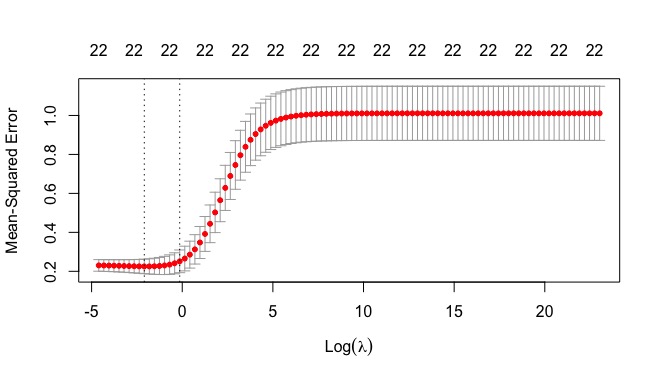


Figure 11: Plot of CV error against Log(lambda) of ridge regression model

*2. Coefficients of the refitted ridge regression model*

To generate a better model, we refit the model with the full dataset. Using the best lambda trained in the train function, we generate a new model. The coefficients of the model show in table 4.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | LotFrontage | LotArea | LotShapeIR2 | LotShapeIR3 | LotShapeReg | OverallCond2 | OverallCond3 |
| 0.050777345 | 0.008612233 | 0.071612152 | 0.246411014 | -0.021164135 | -0.097393687 | -0.526585020 | -0.425865927 |
| OverallCond4 | OverallCond5 | OverallCond6 | OverallCond7 | OverallCond8 | OverallCond9 | MasVnrArea | TotalBsmtSF |
| -0.031664589 | 0.011480505 | -0.065056314 | 0.020113942 | 0.181027496 | 0.455969199 | 0.106539404 | 0.238883697 |
| X1stFlrSF | X2ndFlrSF | GrLivArea | TotRmsAbvGrd | GarageArea | WoodDeckSF |  |  |
| 0.145976158 | 0.141190591 | 0.211954867 | 0.024257499 | 0.248518633 | 0.033472319 |  |  |

Table 5: Coefficient of the ridge regression model

*3. Residual Plot of Ridge Regression Model*

Then, we generate a residual plot of ridge regression model. In figure 12, we cannot see any special pattern in the residual plot that the points are randomly distributes around 0. Also, we can see the linear relationship between the prediction of sale and the real sale in Figure 13. This linear relationship shows that the prediction stability of the model.

|  |  |
| --- | --- |
| Figure 12: Residual Plot of ridge regression model | Figure 13: Sale price prediction vs actual plot of ridge regression model |

Next, we generate two plots. In Figure 14, the plot shows that the standardized residuals of the predictions in the testing dataset is normally distributed with mean close to zero. In Figure 15, the standardized residual vs predicted plot, it shows a u-shape red line because there are some outliers in the dataset.

|  |  |
| --- | --- |
| Figure 14: Std residual of prediction Histogram of ridge regression model | Figure 15: Standardized residual vs predicted plot of ridge regression model |

*4. Summary of Ridge Regression Model*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tested R-Square | Bias | Variance | Training Error | Testing Error | CV error |
| 0.8079794 | 0.9939959 | 0.7038994 | 0.1913637 | 0.1908639 | 0.2251439 |

Table 6: Summary of Ridge Regression Model

LASSO

*1. Generate Ridge Regression Model*

In the LASSO regression model, we use cv.glmnet function with alpha =1 to generate our model. Similar as ridge regression model. We use a 5-fold cross validation method to train the model. Using the same grid in ridge regression model, we set lambda = grid in the function. Then, we use plot function to plot the cross-validation error against log(lambda). When log(lambda) increases, the error increases. However, when log(lambda is larger than 0, the cross-validation error remains unchanged. After training the model, we can get a suitable value for lambda which is 0.0231013. And the best lambda is come from the 97 in the grid. This lambda gives the smallest cross validation error compare with other lambdas we use.

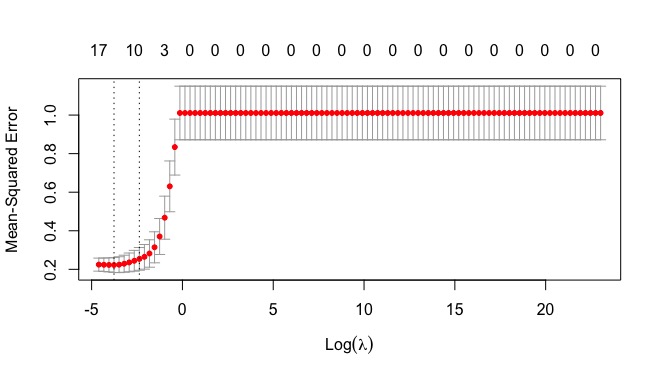


Figure 16: Plot of CV error against Log(lambda) of LASSO regression model

*2. Coefficients of the refitted ridge regression model*

To generate a better model, we refit the model with the full dataset. Using the best lambda trained in the train function, we generate a new model. The coefficients of the model show in table 5.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | LotArea | LotShapeIR2 | LotShapeReg | OverallCond2 | OverallCond3 | OverallCond6 |
| 0.018271476 | 0.060414329 | 0.188261474 | -0.048371216 | -0.337218692 | -0.307991950 | -0.002332798 |
| OverallCond8 | OverallCond9 | MasVnrArea | TotalBsmtSF | GrLivArea | GarageArea | WoodDeckSF |
| 0.096059400 | 0.252142197 | 0.087013260 | 0.271760861 | 0.411692550 | 0.270347278 | 0.013954253 |

Table 7: Coefficient of the LASSO regression model

*3. Residual Plot of Ridge Regression Model*

Then, we generate a residual plot of LASSO model. In figure 17, we can see the data point is normally distributed where it does not concentrate and does not contain any pattern. Also, as showed in figure 18, the relationship of prediction of testing data and actual value is linear related.

|  |  |
| --- | --- |
| Figure 17: Residual Plot of LASSO regression model | Figure 18: Sale price prediction vs actual plot of LASSO regression model |

In the following two figures, they show the characteristics of the predicted value. In figure 19, it shows the histogram of the standardize residual of the prediction. It shows the residual is follow normal distribution. In figure 20, similar as ridge regression, it performs a u shape red line.

|  |  |
| --- | --- |
| Figure 19: Std residual of prediction Histogram of LASSO regression model | Figure 20: Standardized residual vs predicted plot of LASSO regression model |

*5. Summary of LASSO Regression Model*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tested R-Square | Bias | Variance | Training Error | Testing Error | CV error |
| 0.8084832 | 0.9940117 | 0.7151404 | 0.1971728 | 0.1903631 | 0.2224674 |

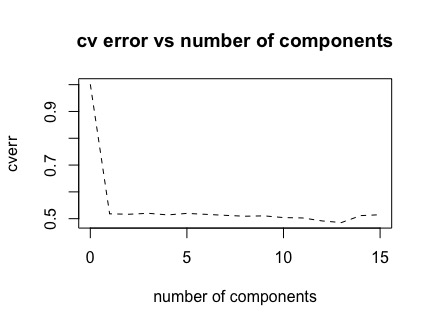
Table 8: Summary of LASSO Regression Model

Principal Components Regression

*1. Number of components*

From the estimation of the pls package using kernel algorithm in R, the reported cross-validation errors of the models are as follow:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | (intercept) | 1 comp | 2 comps | 3 comps | 4 comps | 5 comps | 6 comps | 7 comps | 8 comps |
| RMSE | 1.002 | 0.5181 | 0.5165 | 0.5201 | 0.5141 | 0.5196 | 0.5161 | 0.5124 | 0.5092 |
|  | 9 comps | 10 comps | 11 comps | 12 comps | 13 comps | 14 comps | 15 comps |  |  |
| RMSE | 0.5105 | 0.5041 | 0.5027 | 0.4916 | 0.4856 | 0.5115 | 0.5147 |  |  |



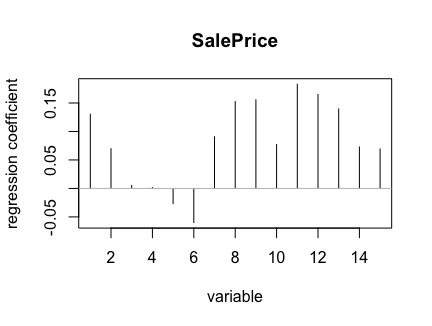
According to the above result, the CV-error has little difference for one or more components included in the model. The second figure has a better visual presentation to the CV-error movement with the number of components included. We can conclude that one component is the appropriate selection. We also used selectNcomp() function in pls package to crosscheck with the selection result. The ‘randomization’ approach is used to return the model with fewest components not significantly worse than the all-components model (Mevik, Wehrens, Liland, Hiemstra, 2019). The function also agrees with the visualization plots.

*2. Regression Coefficients*

A 1-components PCR model is fitted again, the regression coefficients of are shown below:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | LotFrontage | LotArea | LotShapeIR2 | LotShapeIR3 | LotShapeReg | OverallCond | MasVnrArea | TotalBsmtSF | X1stFlrSF |
| Coef. | 0.13030 | 0.06976 | 0.00535 | 0.00147 | -0.02673 | -0.05991 | 0.09074 | 0.15254 | 0.15562 |
|  | X2ndFlrSF | GrLivArea | TotRmsAbvGrd | GarageArea | WoodDeckSF | OpenPorchSF |  |  |  |
| Coef. | 0.07694 | 0.18291 | 0.16400 | 0.13961 | 0.07276 | 0.06923 |  |  |  |

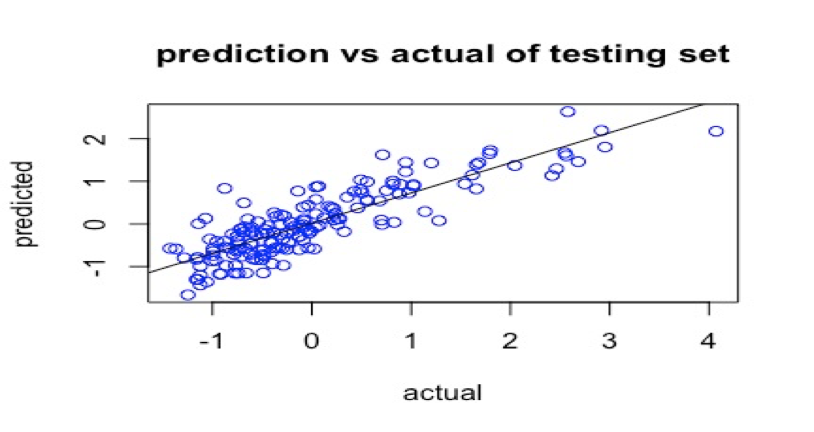
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |



From the figure, ‘LotShapeIR2’, ‘TotalBsmtSF ’ and ‘X2ndFlrSF’ are the top three significant variables. Variables ‘WoodDeskSF’, ‘LotFrontage’, ‘TotRmsAbvGrd’, have coefficient less than 0.05 and are the 3 least significant variables. The others have moderate coefficient. While variable ‘LotShapeReg’ and ‘TotRmsAbvGrd’ have the negative coefficients. The model pseudo R2 equals to 0.7772259, indicating that the model can explain 78% variation of the sale price in training data.

*3. Predictive Power*

First, the linearity is checked by sale price prediction vs actual plot.



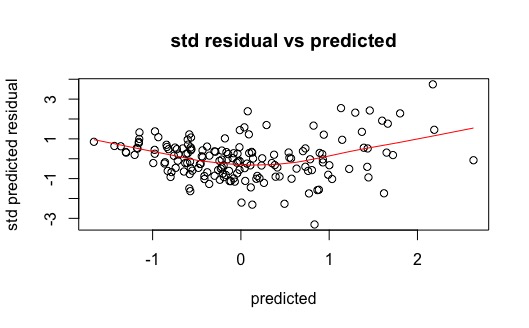
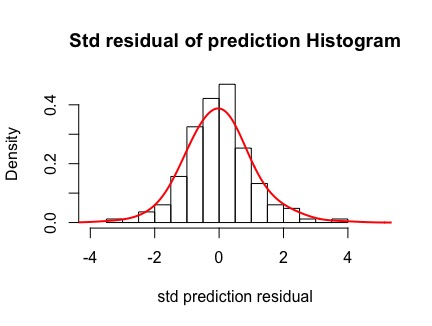
The figure displays the relationship between predicted sale price and actual sale price in the testing dataset. An almost linear relationship can be observed. This indicates the linear model is suitable to explain the variation of sale price.

On the other hand, different indicators for evaluating the performance of the model is calculated, and the results are shown below.

|  |  |  |
| --- | --- | --- |
| Sample Bias | Sample Variance | Tested R2 |
| 0.9943103 | 0.6719595 | 0.7378856 |

From the above table, our model has sample bias larger than sample variance. This indicates our model predict more steadily but less accurate. The tested R sq. indicates the model can explain 74% of the total variation of the sale price in testing dataset.

Next, residual diagnosis plots are obtained as the following.



It is observed from the above two plots that the standardized residuals of the predictions in the testing dataset have mean close to zero. Moreover, the smoother has a U-shape, meaning there probably has a non-linear relation between sale price and the variables and the model is weak in estimating those extreme sale price cases (too expensive or too cheap).

*4. Conclusion*

Summarizing the above result, the model does a good job overall. The coefficients of the variables show Moderately Irregular shape of property, total basement area and the Second-floor square feet are the three most important metrices to the sale price. The linear feet of street connected to the property, Total rooms above grade (does not include bathrooms), and the Wood deck area in square feet are three insignificant metrices to the sale price. From the prediction result we can conclude the model trade off bias for variance, resulting less accurate but more stable prediction. Close ranged pseudo R square (0.78) and tested R square (0.74) is another evidence to support. The U-shaped residual plot suggests the model can be further improved if more detailly job on transformation is spent.

Regression Splines

*1. Find out the best parameter df*

With the help of the caret package in R, the training model with 5-fold cross validation process are fitted. The result is that the best df is 2.

*2. Regression Coefficients*

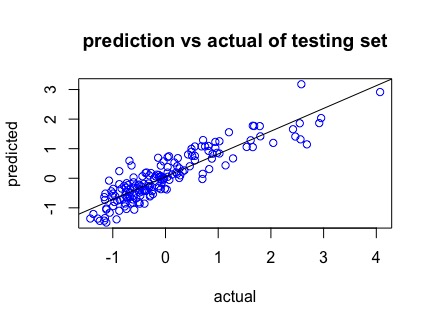
Then, using caret package, the training model with 5-fold cross validation process are fitted. The results are shown below while the ones are significant are summarized in the table.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | LotShapeIR2 | LotShapeIR3 | LotShapeReg | OverallCond | TotRmsAbvGrd | s(WoodDeckSF) | s(OpenPorchSF) | s(MasVnrArea) | s(X2ndFlrSF) | s(LotFrontage) | s(GarageArea) | s(TotalBsmtSF) | s(X1stFlrSF) | s(GrLivArea) |
| -0.420408674 | 0.350293907 | -0.012434696 | -0.106414455 | 0.085279115 | -0.038815835 | 0.020081451 | 0.072433054 | 0.081442569 | 0.234275568 | 0.016949694 | 0.248298832 | 0.307786675 | 0.150221125 | 0.201239453 |

From the figure, ‘s(TotalBsmtSE, df = parm$df)’, ‘s(LotShapeIR2, df = parm$df)’ , ‘s(GarageArea, df = parm$df)’, and ‘s(X2ndFlrSF, df = parm$df)’ are the top four significant variables. Variable ‘s (X1stFlrSF, df = parm$df)’ has the lowest coefficient smaller than 0.05, are the least significant variables. The others have moderate coefficient. The model pseudo R2 equals to 0.8317601, indicating that the model can explain 83% variation of the sale price in training data.

*3. Predictive Power*

First, the linearity is checked by sale price prediction vs actual plot.



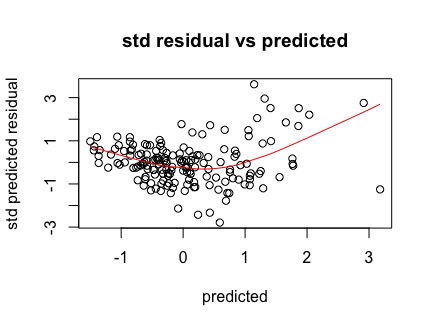
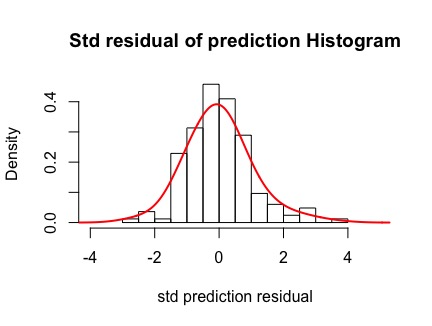
The figure displays the relationship between predicted sale price and actual sale price in the testing dataset. An almost linear relationship can be observed. This indicates the linear model is suitable to explain the variation of sale price.

On the other hand, different indicators for evaluating the performance of the model is calculated, and the results are shown below.

|  |  |  |
| --- | --- | --- |
| Sample Bias | Sample Variance | Tested R2 |
| 0.9966511 | 0. 7274156 | 0.8046241 |

From the above table, our model has sample bias larger than sample variance. This indicates our model predict more steadily but less accurate. The tested R sq. indicates the model can explain 80% of the total variation of the sale price in testing dataset.

Next, residual diagnosis plots are obtained as the following.



It is observed from the above two plots that the standardized residuals of the predictions in the testing dataset have mean close to zero. Moreover, the smoother has a left-skewed U-shape, meaning there probably has a non-linear relationship between sale price and the variables and the model is weak in estimating those extreme sale price cases (too expensive or too cheap).

*4. Conclusion*

Summarizing the above result, the model does well overall. The coefficients of the variables show the Wood deck area in square feet, open porch area in square feet, size of garage in square feet, and total square feet of basement area are the four most important metrices to the sale price. First Floor square feet is the least significant indicator to the sale price. From the prediction result we can conclude the model trade off bias for variance, resulting less accurate but more stable prediction. Close ranged pseudo R square (0.83) and tested R square (0.805) is another evidence to support. The U-shaped residual plot suggests the model can be further improved.

Local Regression

*1. find out the best parameter span*

With the help of the caret package in R, the training model with 5-fold cross validation process are fitted. The results are shown below. The best tuned parameters is span=0.99 and degree = 1 with RMSE 4.7699E-01.

*2. Regression Coefficients*

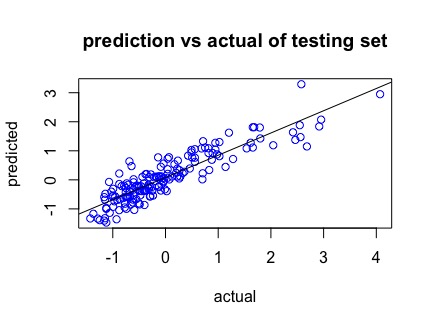
Then, using caret package, the training model with 5-fold cross validation process are fitted. The results are shown below while the ones are significant are summarized in the table.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | LotShapeIR2 | LotShapeIR3 | LotShapeReg | OverallCond | TotRmsAbvGrd | lo(WoodDeckSF) | lo(OpenPorchSF) | lo(MasVnrArea) | lo(X2ndFlrSF) | lo(LotFrontage) | lo(GarageArea) | lo(TotalBsmtSF) | lo(X1stFlrSF) | lo(GrLivArea) | lo(LotArea) |
| -0.41344 | 0.352851 | 0.029029 | -0.10979 | 0.084318 | -0.0356 | 0.020098 | 0.072056 | 0.079331 | 0.240957 | 0.015308 | 0.250206 | 0.301837 | 0.162602 | 0.188548 | 0.059621 |

From the figure, ‘lo(X2ndFlrSF, span = 0.99, degree = 1)’, ‘lo(LotShapeIR2, span = 0.99, degree = 1) ’ , ‘lo(GarageArea, span = 0.99, degree = 1)’, and ‘lo(TotalBsmtSF, span = 0.99, degree = 1)’ are the top four significant variables. Variables ‘lo (WoodDeckSF, span = 0.99, degree = 1)’ and ‘lo(LotShapeIR3, span = 0.99, degree = 1)’, have lower coefficients, are the 2 least significant variables. The others have moderate coefficient. The model pseudo R2 equals to 0.8294882, indicating that the model can explain 83% variation of the sale price in training data.

*3. Predictive Power*

First, the linearity is checked by sale price prediction vs actual plot.



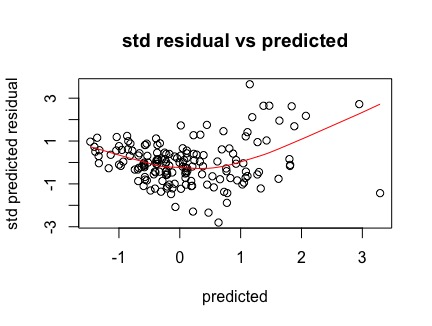
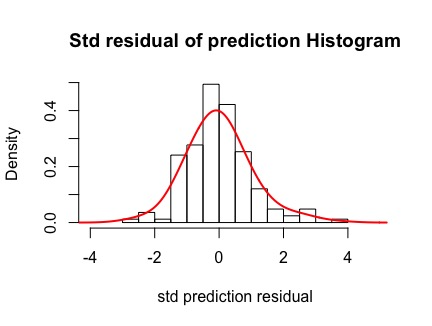
The figure displays the relationship between predicted sale price and actual sale price in the testing dataset. An almost linear relationship can be observed. This indicates the linear model is suitable to explain the variation of sale price.

On the other hand, different indicators for evaluating the performance of the model is calculated, and the results are shown below.

|  |  |  |
| --- | --- | --- |
| Sample Bias | Sample Variance | Tested R2 |
| 1.000542 | 0. 7232044 | 0.797902 |

From the above table, our model has sample bias larger than sample variance. This indicates our model predict more steadily but less accurate. The tested R sq. indicates the model can explain 80% of the total variation of the sale price in testing dataset.

Next, residual diagnosis plots are obtained as the following.



It is observed from the above two plots that the standardized residuals of the predictions in the testing dataset have mean close to zero. Moreover, the smoother has a left-skewed U-shape, meaning there probably has a non-linear relation between sale price and the variables and the model is weak in estimating those extreme sale price cases (too expensive or too cheap).

*4. Conclusion*

Summarizing the above result, the model does a good job overall. The coefficients of the variables show the Wood deck area in square feet, open porch area in square feet, size of garage in square feet, and total square feet of basement area are the four most important metrices to the sale price. First Floor square feet and total rooms above grade (does not include bathrooms) are two less significant metrices to the sale price. From the prediction result we can conclude the model trade off bias for variance, resulting less accurate but more stable prediction. Close ranged pseudo R square (0.83) and tested R square (0.80) is another evidence to support. The U-shaped residual plot suggests the model can be further improved.

Partial Least Square Regression Model

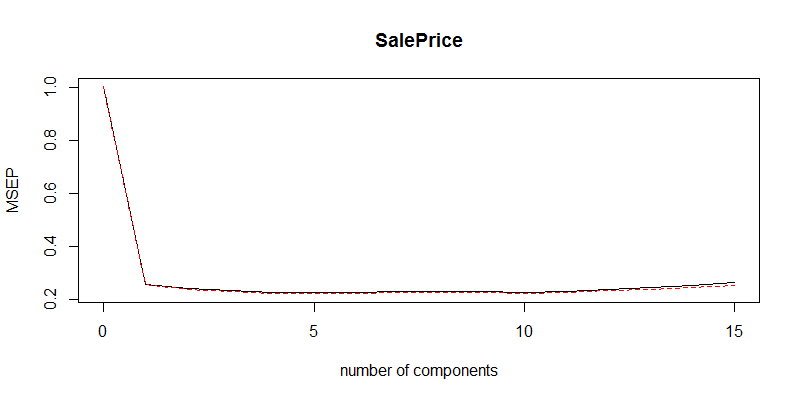
We found that the residual vs prediction plot gave a quadratic shape after we build a PLS model without any transformation. It was therefore we transformed SalePrice to log(SalePrice).

1. Number of components

From the estimation of the pls package using kernel algorithm in R, the reported cross-validation errors of the models before and after transformation are as follow:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | (intercept) | 1 comp | 2 comps | 3 comps | 4 comps | 5 comps | 6 comps | 7 comps | 8 comps | 9 comps | 10 comps |
| RMSE | 1.002 | 0.509 | 0.4935 | 0.4853 | 0.4774 | 0.4756 | 0.4777 | 0.4805 | 0.4802 | 0.4801 | 1.002 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 11 comps | 12 comps | 13 comps | 14 comps | 15 comps |
| RMSE | 0.4792 | 0.4805 | 0.4895 | 0.4984 | 0.5147 |



According to the above result, the CV-error has little difference for 3 or more components included in the model. From above figures has a better visual presentation to the CV-error movement with the number of components included. We can conclude that 3 components is the appropriate selection. We also used selectNcomp() function in pls package to crosscheck with our selection. We used ‘randomization’ approach to return the model with fewest components not significantly worse than the all-components model (Mevik, Wehrens, Liland, Hiemstra, 2019). The function also agrees with our visual inspection.

2. Regression Coefficients

We then fit a 3-components PLS model again, the regression coefficients of are shown below:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | LotFrontage | LotArea | LotShapeIR2 | LotShapeIR3 | LotShapeReg | OverallCond | MasVnrArea | TotalBsmtSF | X1stFlrSF | X2ndFlrSF | GrLivArea |
| Coef. | 0.033978 | 0.048214 | 0.03027 | 0.002609 | -0.05226 | 0.113157 | 0.11222 | 0.194045 | 0.139452 | 0.130083 | 0.214391 |

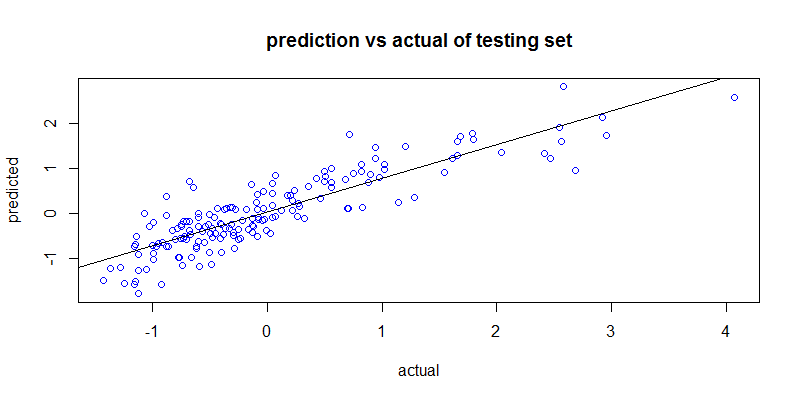
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | TotRmsAbvGrd | GarageArea | WoodDeckSF | OpenPorchSF |
| Coef. | 0.112767 | 0.252062 | 0.055858 | 0.098063 |



From above figures, ‘GarageArea’, ‘GrLivArea’ and ‘TotalBsmtSF’ are the top three significant variables. ‘LotFrontage’, ‘LotArea’, ‘LotShapeIR2’, ‘LotShapeIR3’ have coefficient less than 0.05 and are the 3 least significant variables. The others have moderate coefficient. The model pseudo equals to 0.7906, indicating that the model can explain 79% variation of the sale price in training data.

3. Predictive Power

First, we check if there is non-linearity in sale price prediction vs actual plot.



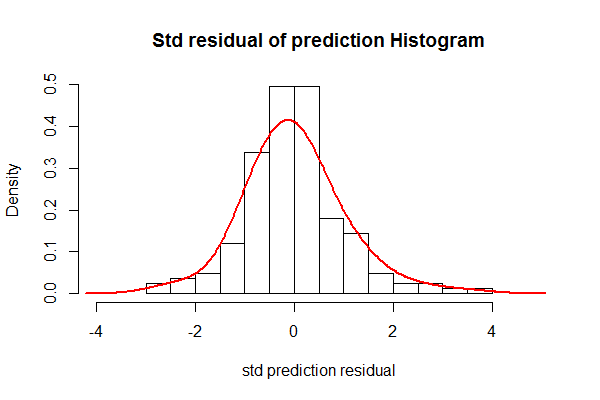
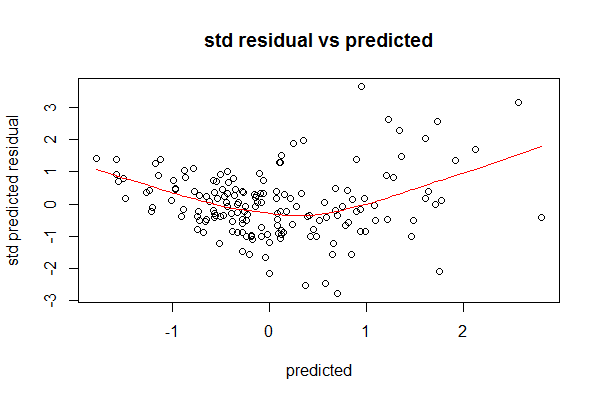
From above figures displays the relationship between predicted sale price and actual sale price in the testing dataset. An almost linear relationship can be observed. This indicates our linear model is suitable to explain the variation of sale price.

On the other hand, we calculate different indicators to evaluate the performance of the model.

|  |  |  |
| --- | --- | --- |
| Sample Bias | Sample Variance | Tested |
| 0.9949974 | 0.727824 | 0.7629896 |

From the above table, our model has sample bias larger than sample variance. This indicates our model predict more steadily but less accurate. The tested R sq. indicates the model can explain 76% of the total variation of the sale price in testing dataset.

Next, we check the sample distribution of the predicted residual.

We can observe from the above two plots that the standardized residuals of the predictions in the testing dataset have mean close to zero. In the standardized residual vs predicted plot, the smoother has a U-shape, meaning there may be a non-linear relation between sale price and the variables. Besides, the model is weak in estimating extreme house price (very expensive or very cheap).

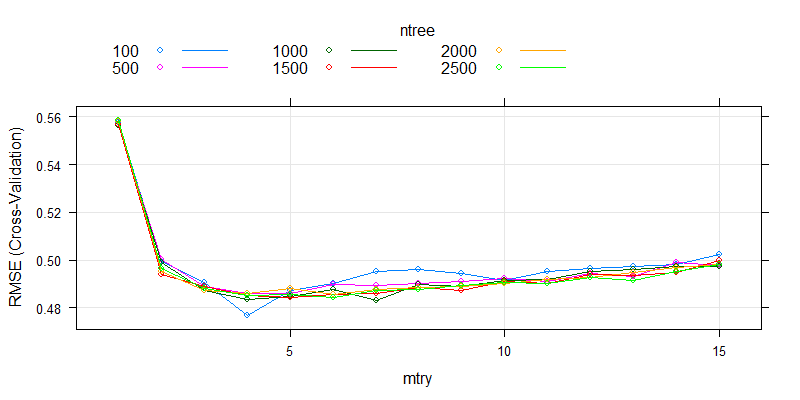
4. Conclusion

Summarizing the above result, the model can predict sale price well overall. The coefficients of the variables show garage area, ground living area and total basement area are the three most important metrices that boost the sale price. The linear feet of street connected to the property, the lot area and the general shape of the property are three insignificant metrices on sale price. From the prediction result we can conclude the model trade off bias for variance, resulting less accurate but more stable prediction. Close ranged pseudo R square (0.79) and tested R square (0.76) is another evidence to prove our point. The U-shaped residual plot suggests the model can be further improved if we spend more time on transformation.

Random Forest

1.Parameters tuning

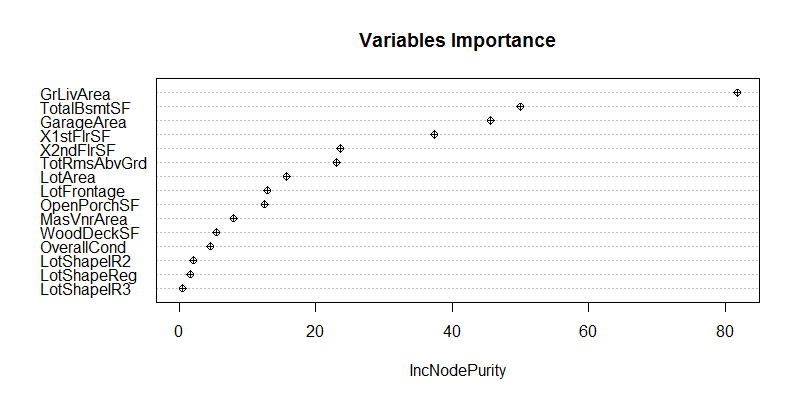
For building random forest, the number of trees and the number of variables sampled as candidates in each spilt are two model parameters we want to maximize. For which, we used caret() library along with randomForest() library to maximize the two parameters using cross-validation.



From above figures, the shapes of RMSE against number of candidates for all number of trees except 100 are almost identical. There is a significant drop in RMSE for 100 trees with 4 candidates each split and attained the lowest RMSE among all. It is therefore the best tune. The corresponding CV-error (RMSE) is 0.476698.

2.Variables Importance

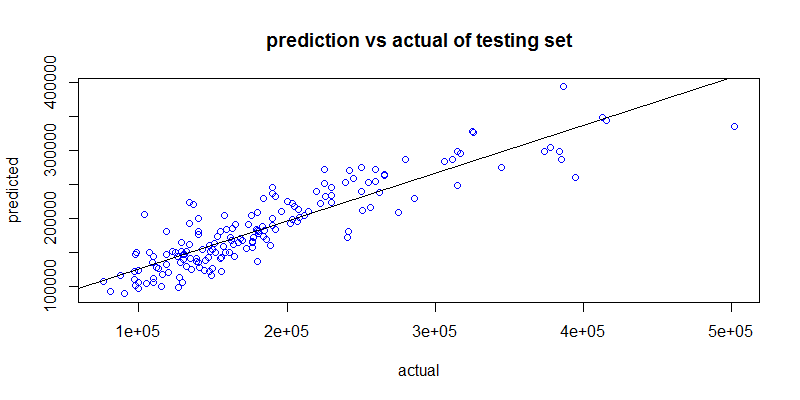
Unlike additive models, random forest model has no coefficients of variables. Instead we can examine the importance of the variables by adding up the increase in RSS when the variable is included for all splits. It can help to identify the effect of the variables on estimation.



From above figures shows the importance plot of the best model we found in the last section. ‘GrLivArea’, ‘TotalBsmtSF’ and ‘GarageArea’ are highly significant variables on sale price estimation. ‘LotShape’ has insignificant effect on the estimation, its dummy variables all have very little increase in RSS when including them. Others have moderate effect on the estimation. The pseudo is 0.782467.

3.Predictive Power

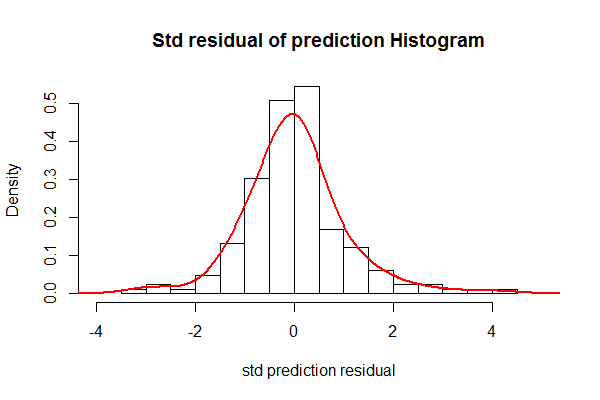
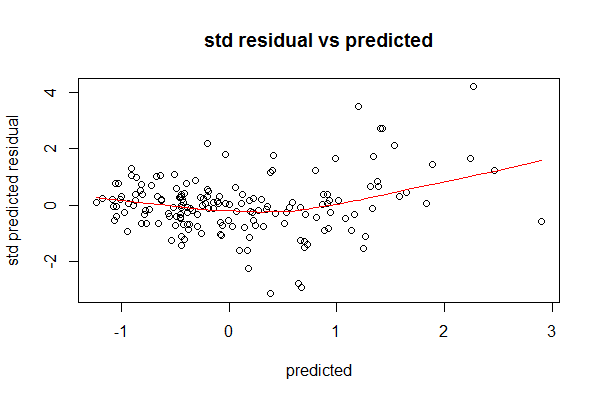
First, we check if there is non-linearity in sale price prediction vs actual plot.



From above figures, the predicted price and the actual price has an overall linear relationship. Indicating the prediction stability of the model.

On the other hand, we also check the metrices as before.

|  |  |  |
| --- | --- | --- |
| Sample Bias | Sample Variance | Tested |
| 0.9980402 | 0.6749213 | 0.7971908 |

From above figures shows a bump on the left tail and the scatter plot shows an elevation in higher predicted sale price. Indicating the model is underestimating the higher sale price houses.

4. Conclusion

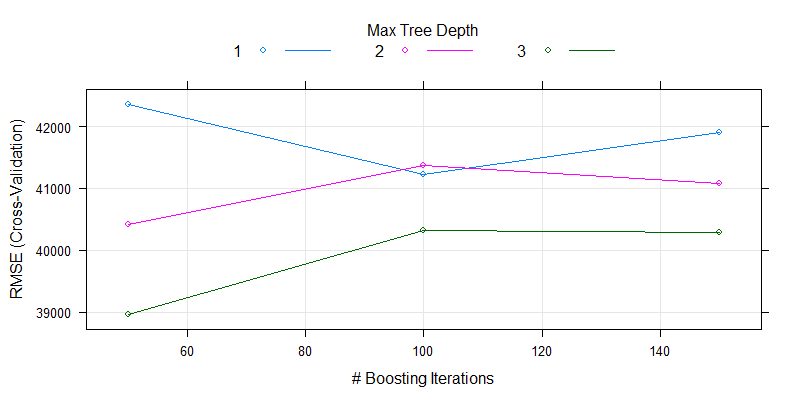
Summarizing the above result, the model can predict sale price well overall. The importance value of the variables shows garage area, ground living area and total basement area, which agrees with previous models, are the three most important metrices that boost the sale price. Unlike others, this model suggests only the shape of the property has little effect on sale price of the house. The model also has good and close pseudo and tested R square (0.78 and 0.80 respectively), indicating its strong and steady predictive power. The tested residuals tend to be larger when predicted price is higher, indicating the model is less favorable when predicting expensive houses. Meanwhile cheap and moderate houses have zero-mean residuals, of which can be predicted using this model.

Boosting

For this method, standardized data drastically reduce the accuracy of the final model, so we use the original data to generate the following result.

1. Parameter Tuning

We used ‘gbm’ to implement boosting to build tree-based model. There are parameters to be tuned in the package. We again used ‘caret’ to help us. We use 5-fold cross-validation to tune.



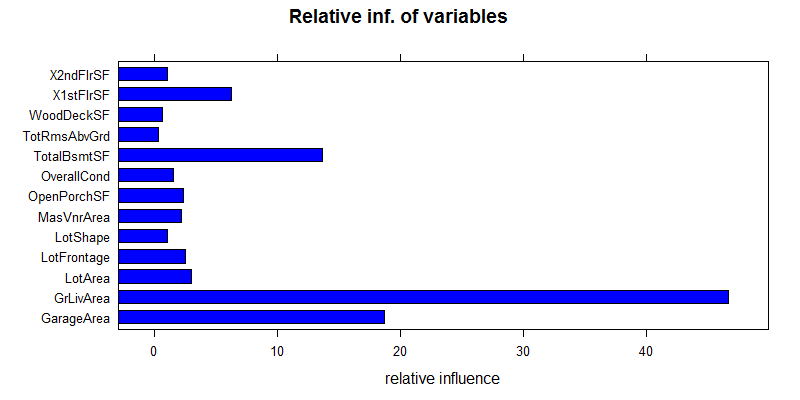
The number of trees (n.trees) and the maximum tree depth (interaction.depth) are tuned. The best set is 50 trees and 3 levels of depth, with C.V. error (RMSE) 38961.16. The full list can be referred to our R notebook.

2.Relative influence

The interpretation of boosting tree model is similar to random Forest. We calculate the relative influence, which is the mean reduction of MSE whenever the variable used to split a node. The larger the reduction, higher the influence of that variable. We again fit the model with the best-tuned parameters to generate following result.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | LotFrontage | LotArea | LotShape | OverallCond | MasVnrArea | TotalBsmtSF | X1stFlrSF | X2ndFlrSF | GrLivArea |
| Infl. | 2.523658 | 3.003611 | 1.040885 | 1.57089 | 2.212193 | 13.66206 | 6.229151 | 1.055044 | 46.67194 |

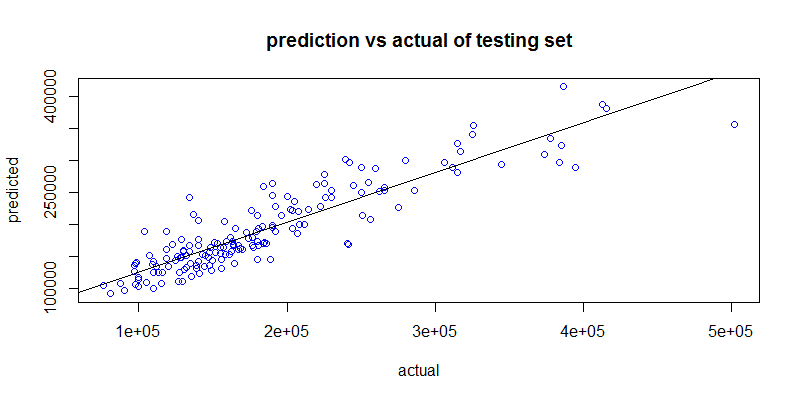
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | TotRmsAbvGrd | GarageArea | WoodDeckSF | OpenPorchSF |
| infl. | 0.305487 | 18.70643 | 0.649242 | 2.369398 |



From above figures shows ‘GrLivArea’ is the most influential variable, while ‘GarageArea’ and ‘TotalBsmtSF’ are the 2nd and 3rd high influence variables. The table shows ‘TotRmsAbvRd’, ‘WoodDeckSF’ and ‘LotShape’ are the three most insignificant variables, while others have moderate relative influence. The model has pseudo equals to 0.8733.

3. Predictive Power

First, we check if there is non-linearity in sale price prediction vs actual plot.



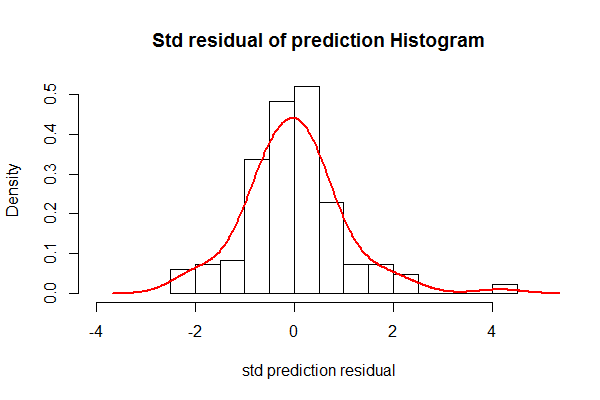
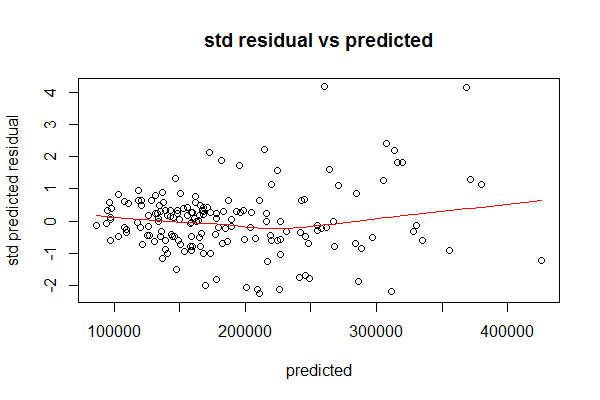
From above figures, the predicted price and the actual price has an overall linear relationship. Indicating the prediction stability of the model.

On the other hand, we calculate different indicators to evaluate the performance of the model.

|  |  |  |
| --- | --- | --- |
| Sample Bias | Sample Variance | Tested |
| 5.99E+09 | 4.43E+09 | 0.8039583 |

From the above table, our model has sample bias larger than sample variance. This indicates our model predict more steadily but less accurate. The tested R sq. indicates the model can explain 80% of the total variation of the sale price in testing dataset.

Next, we check the sample distribution of the predicted residual.

We can observe from the above two plots that the standardized residuals of the predictions in the testing dataset have mean close to zero. In the standardized residual vs predicted plot, the smoother is slightly U-shaped, meaning there may be a bit non-linearity relation between sale price and the variables. Overall, the standardized residuals still have zero mean across the predictions.

4. Conclusion

Summarizing the above result, the model can predict sale price well overall. The relative influence of the variables shows garage area, ground living area and total basement area, which agrees with previous models, are the three most important metrices that boost the sale price. However, this boosting model suggests total rooms above grade, wood deck area and shape of property are the three most insignificant metrices on sale price. The model also has high pseudo and tested R square (0.87 and 0.80 respectively), indicating its strong predictive power. Moreover, the tested residuals show little non-linearity, indicating relatively reliable prediction even if extreme price.

Conclusion

There is one thing that most of the models agree with, which is garage area, ground living area are the two most significant metrices that boost the sale price of the houses. The shape of the property is another factor that most of the models suggest it has very little effect on the sale price of the houses.

Next, we compare the tested R square of the models we built

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Stepwise OLS | Backward OLS | Forward OLS | Ridge | LASSO | PCR | Local reg. | Spline | PLS |
| Tested  R sq. | 0.7760 | 0.8052 | 0.7981 | 0.8080 | 0.8085 | 0.7379 | 0.7979 | 0.8046 | 0.7629896 |

|  |  |  |
| --- | --- | --- |
|  | Random Forest | Boosting |
| Tested  R sq. | 0.7972 | 0.8040 |

The above table suggests backward OLS, LASSO, ridge regression, regression with spline and boosting are the only models that are over 0.8. They all have very high accuracy over this regression problem. Two weakest among all are principle component regression and partial least square. They are actually similar in mechanism so it is not surprise for them to perform poorly at the same time.

Based on tested R square, backward OLS, LASSO, ridge regression, regression with spline and boosting are very similar in magnitude so there is still no definitive choice. It is therefore we need to compare their tested residual plots. All models mentioned above shows U-shape in residual vs predicted plot, except the boosting model shows relatively flat smoother. It is very clear that the mean of the tested residual of the boosting model is steadily close to zero. While other models show non-zero -mean residuals. Consequently, we can conclude that the BOOSTING model is the most reliable one out of all 9 approaches.

# Survival of passengers on Titanic models

Data Description

The dataset contains 500 entries, each represent 1 passenger. The target variable, which we aim to predict, is Survived. The predictor variables include Pclass, Sex, Age, Sibsp, Parch, Fare, Embarked. The table below describes these variables. Some entries contain missing data. There are 102 missing data in Age, and 1 in Embarked.

|  |  |
| --- | --- |
| **Variable** | **Description** |
| Target |  |
| Survived | 1: Survived, 2: Did not survive |
| Predictor |  |
| Pclass | 1: 1st class ticket holder, 2: 2nd class, 3: 3rd class |
| Sex | Male/Female |
| Age | Age in years |
| IsAgeEstimated | Whether the Age is estimated or not |
| Sibsp | Number of siblings and spouses aboard |
| Parch | Number of parents and children aboard |
| Fare | Fare paid by the passenger |
| Embarked | Port of Embarkation, C: Cherbourg, Q: Queenstown, S: Southampton |

Pre-processing

The complete dataset is split into 125 testing data and 375 training data, which is further split into 5 folds of 75 data for cross-validation.

Due to large portion of Age is missing, removing those entries would severely damage the representativeness of the models fitted. Therefore, the missing data were imputed by Random Forest, which were done separately on training dataset and testing dataset without the target variable Survived. Outliers are not removed to preserve representativeness.

Some values in Age were estimated by the creator of this dataset, an additional IsAgeEstimated variable is added to denote that.

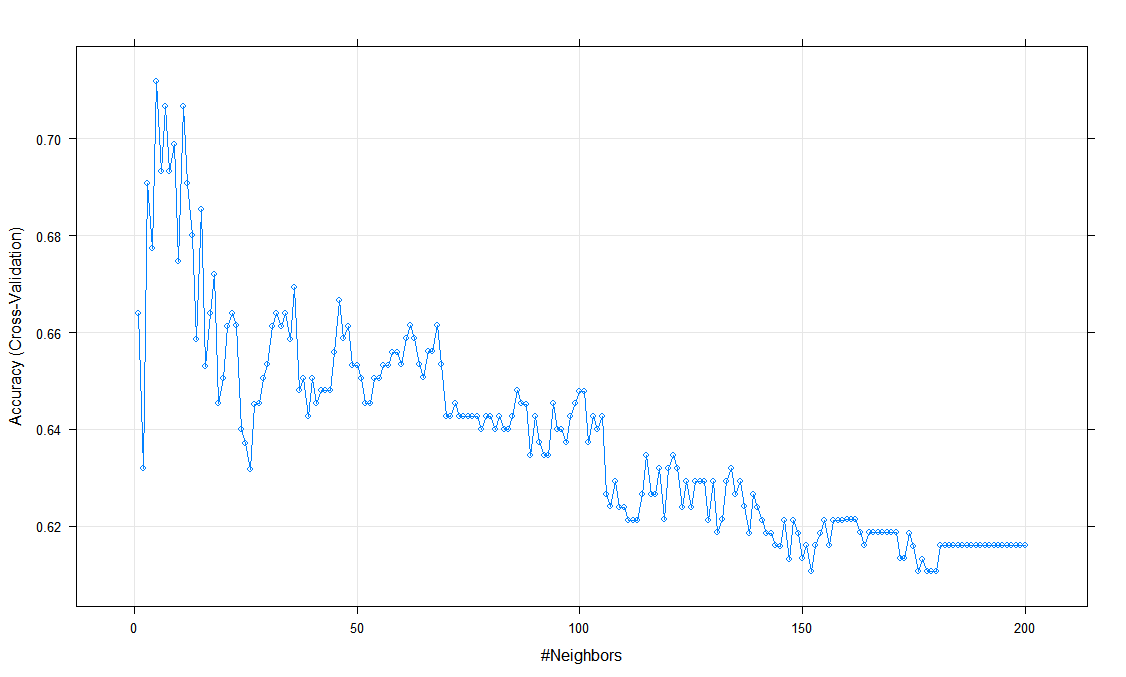
For training data, Sibsp, Parch, and Pclass are standardized using Min-Max scaling to bound the value to be within 0 and 1. Age and Fare are standardized using Z-score scaling to lower the magnitude to be roughly within -3 and 3, so that they are comparable to the scale of other variables.

For testing data, the same variables are standardized using the same methods, but with the minimum, maximum, mean and standard deviation obtained in training data.

KNN Model

*1. Model Selection*

The major parameter of knn model is the number of data points (“neighbors” or “k”) when making prediction. We use 5-folds cross validation to search for the best k number. We search between k=1 to k=200 and we found that k=5 has the highest accuracy, which means it is the best parameter.

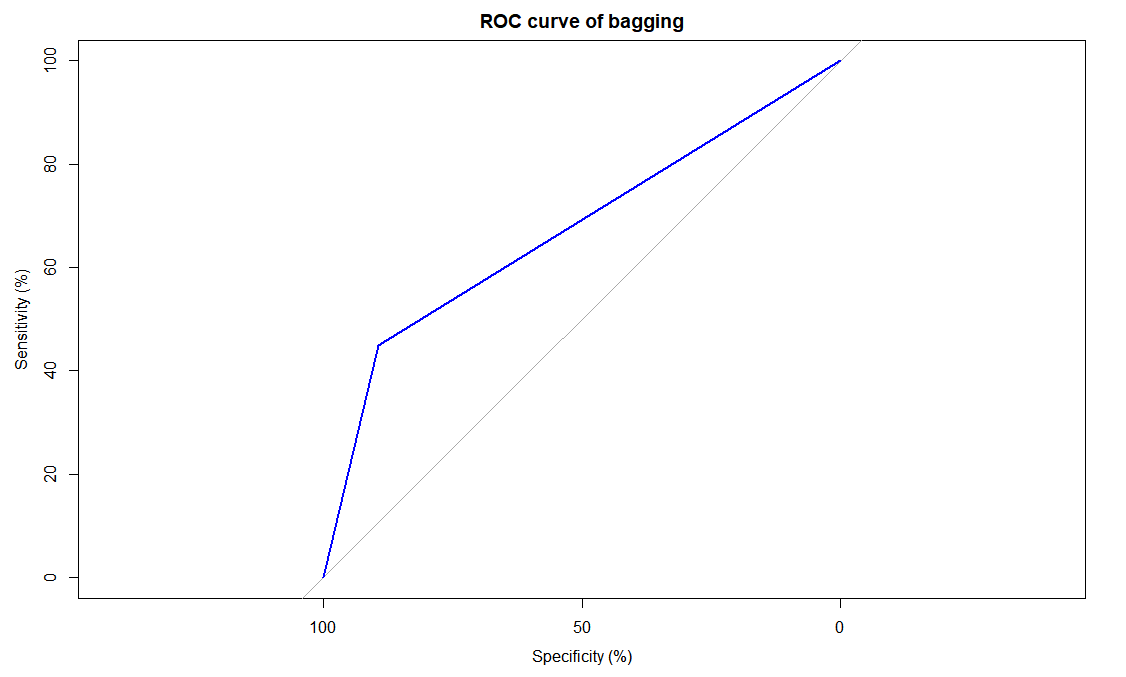
Plot showing the relationship between cross validation error and number of neighbors

*2. prediction result*

The model predicts Survived 22% incorrectly in the training data set, and 28% incorrectly in testing data set. Out of 76 passenger who did not survive in the testing data set, the model correctly predicts 68. Out of 49 passenger who survive in the testing data set, the model correctly predicts 22.

|  |  |  |
| --- | --- | --- |
|  | **Predicted 0** | **Predicted 1** |
| **Real 0** | 68 | 8 |
| **Real 1** | 27 | 22 |

|  |  |
| --- | --- |
|  | **Class Prediction** |
| **CV error** | 0.2881252 |
| **Training error** | 0.216 |
| **Testing error** | 0.28 |
| **AUC** | 0.6719 |



Classification Tree Model

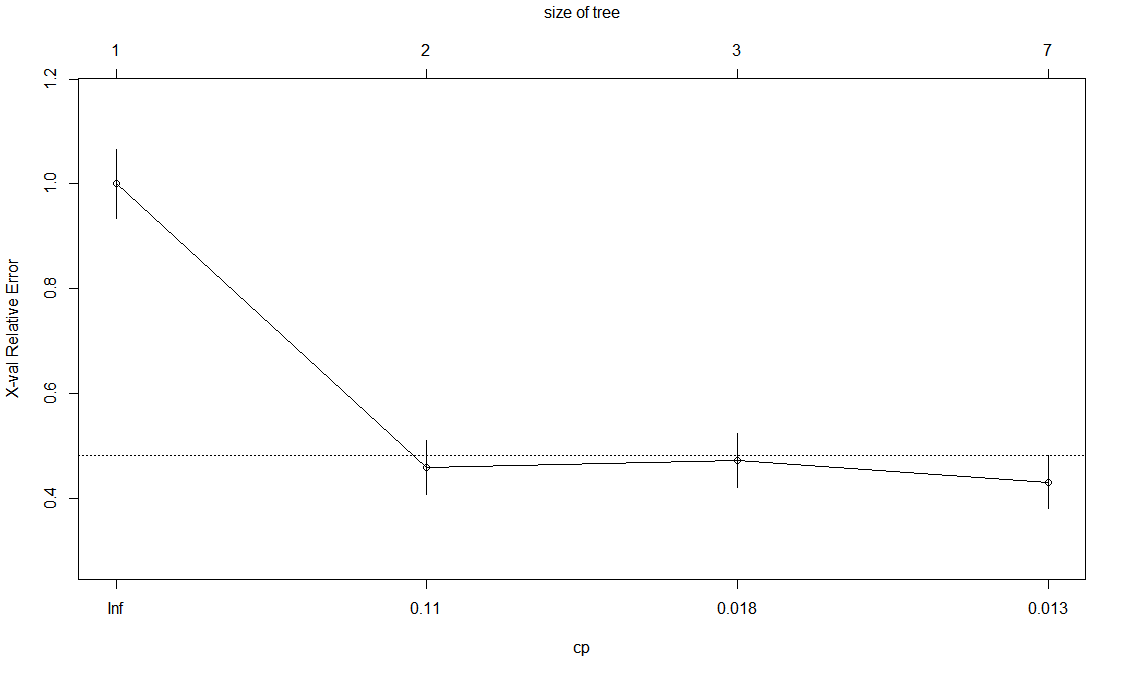
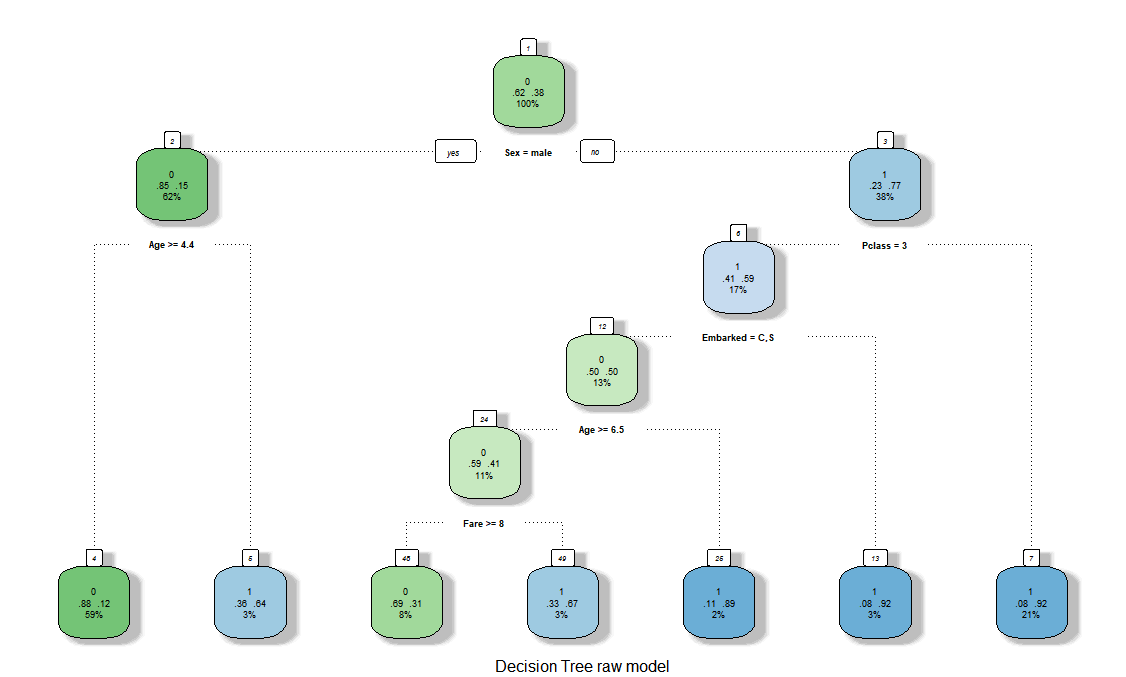
*1. Model Selection*

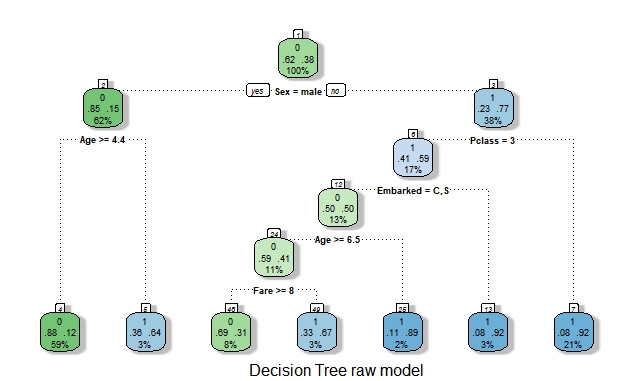
The major parameter of classification tree model is the complexity of the tree model in pruning. We use rpart package and funciton rpart() generate the full decision tree model with 5-folds cross validation. The 5-folds cross validation with generate a “CP value” which is the complexity parameter. We compare the cross validation error and select the model with least cross validation error for pruning the tree model. The cross validation error here is the misclassifcation rate of a model comparing with the tree model with no splits.

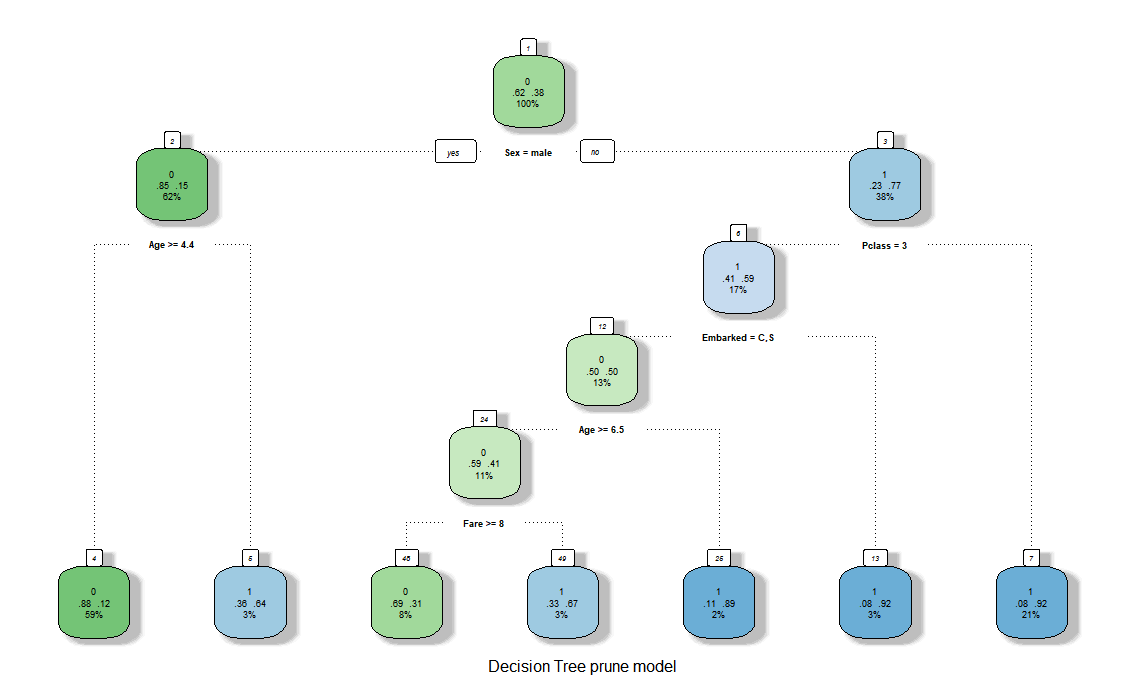
Cross validation error with different complexity table

|  |  |  |
| --- | --- | --- |
| Model complexity | Number of splits | Cross validation error (for complexity) |
| 0.541667 | 0 | 1.00000 |
| 0.020833 | 1 | 0.45833 |
| 0.016204 | 2 | 0.47222 |
| 0.010000 | 6 | 0.43056 |

plot showing Cross validation error with different complexity

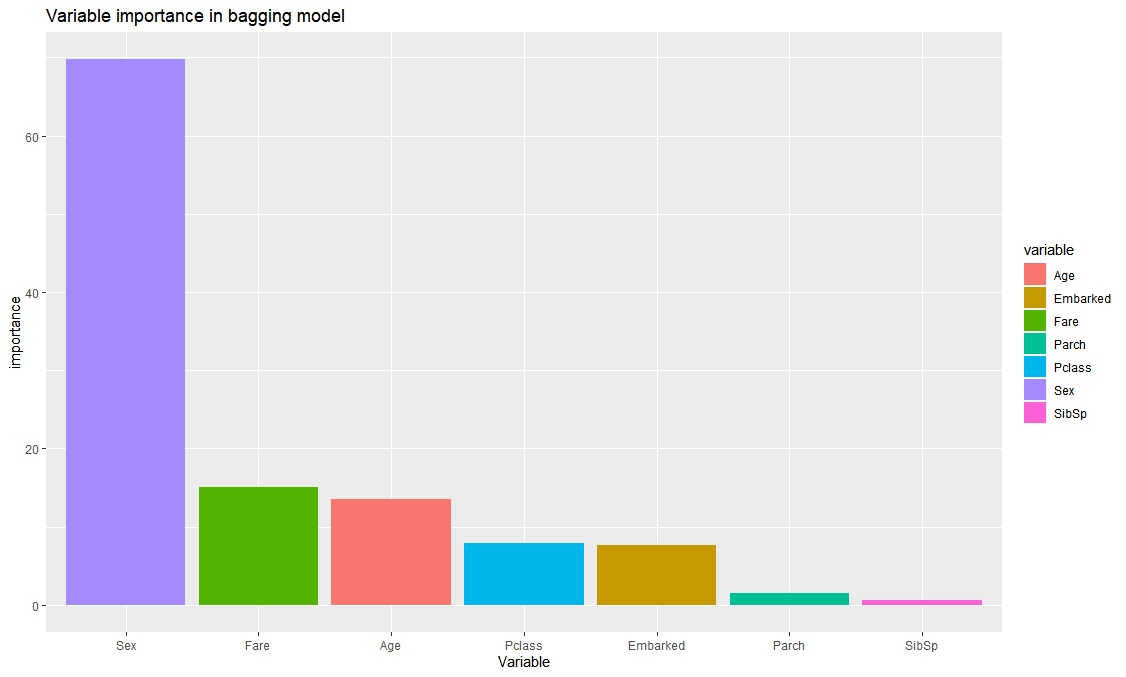
The decision tree model plot (without prune)

The decision tree model plot (without prune)

The decision tree model plot (with prune)

*2. Model Interpretation*

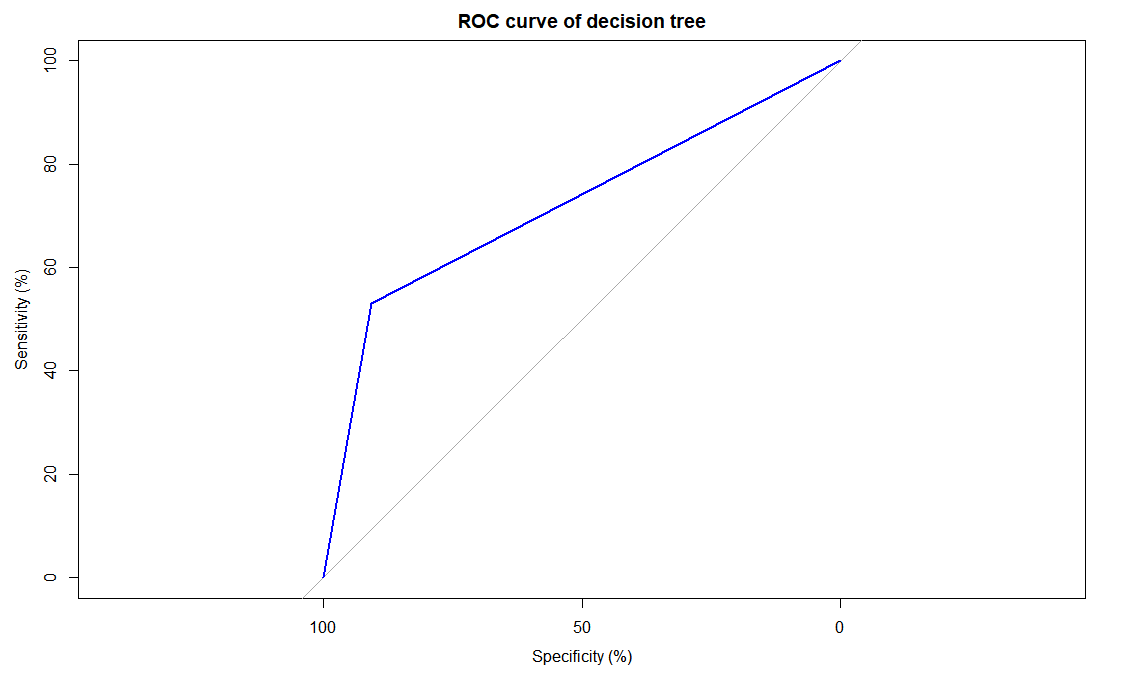
The model above looks like the same decision tree. In fact, for the rpart package, the default parameter of rpart() for cp is 0.01, which is the same as the CP value chosen from the result of cross validation. That means for no pruning is needed for the tree model. As a result, the CV error and the Training error is the same. From the bar chart, we know that the variable “sex” has the highest importance level. Which means it has higher prediction power on survived.

Bar chart showing the importance of the variables in the tree model

*3.Predictive Power*

The model predicts Survived 14% incorrectly in the training data set, and 24% incorrectly in testing data set. Out of 76 passenger who did not survive in the testing data set, the model correctly predicts 69. Out of 49 passenger who survive in the testing data set, the model correctly predicts 26.

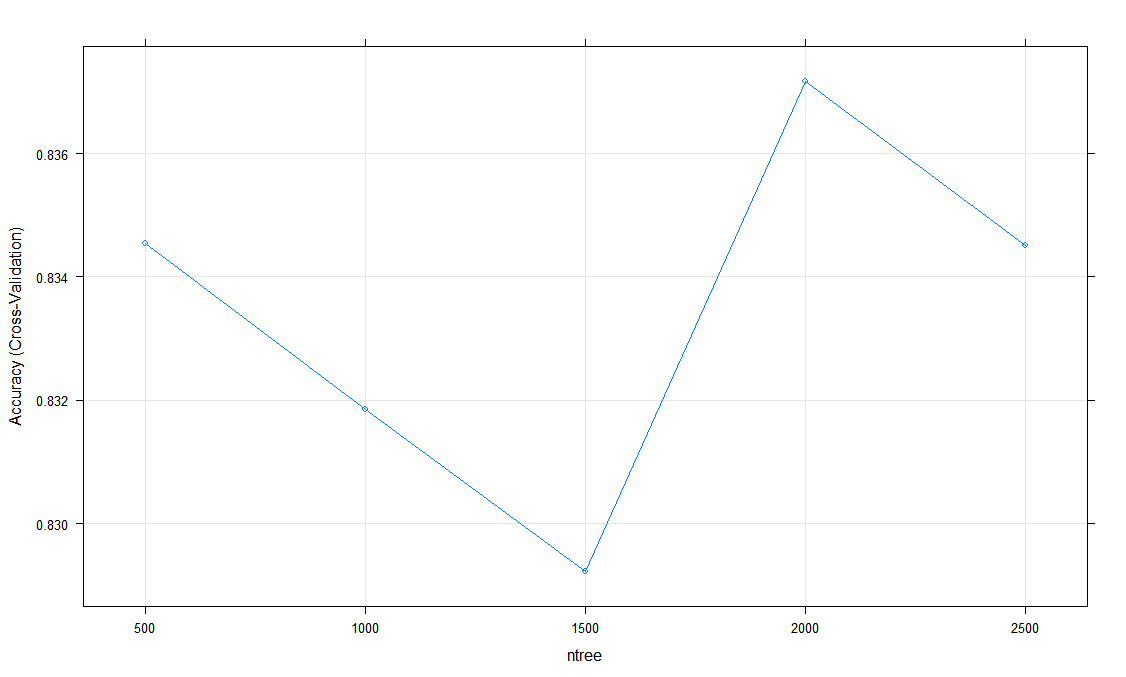
|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Class Prediction** | |
| **CV error** | | 0.1387 | |
| **Training error** | | 0.1387 | |
| **Testing error** | | 0.24 | |
| **AUC** | | 0.7193 | |
|  | **predicted 0** | | **Predicted 1** |
| **real 0** | 69 | | 9 |
| **real 1** | 23 | | 26 |



Bagging Model

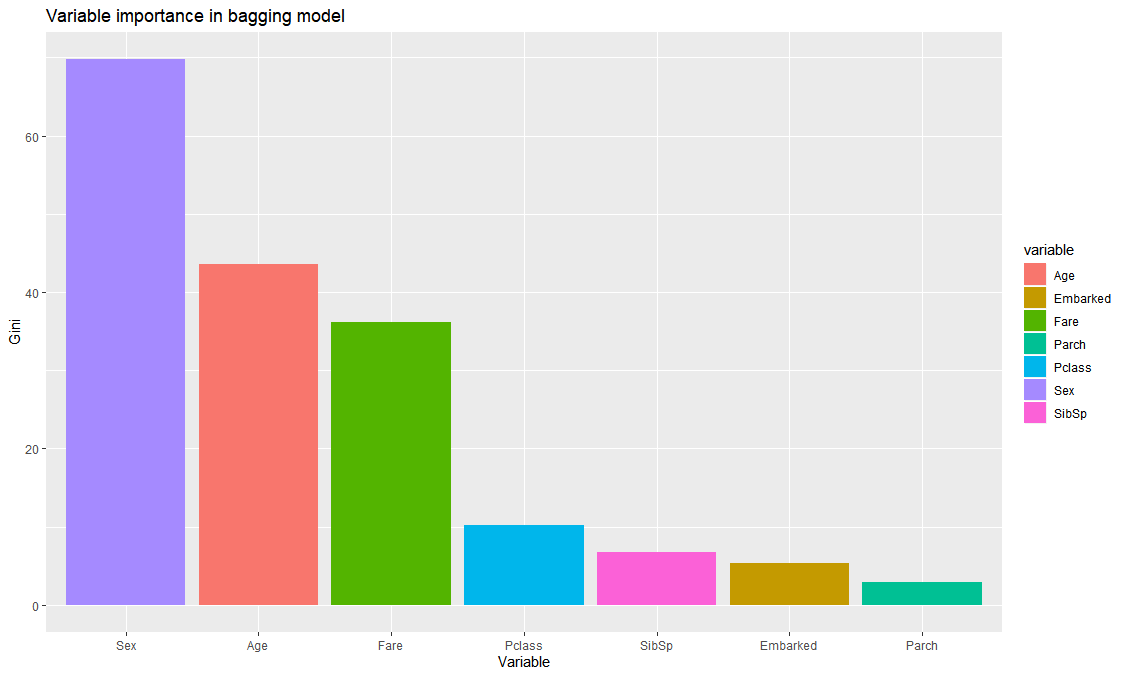
*1. Model Selection*

We apply bagging in random forest. The number of variables used in each node must be fixed as the number of variables in training set. As a result, we tune the number of trees for bagging. We apply 5-folds cross validation to choose the best number of trees between 500 to 2500. Accuracy are used to measure the prediction power of the model. The best model should have the highest accuracy. We found that the model with 2000 trees is the best model with highest accuracy.

The relationship between number of tree and accuracy.

*2. Model Interpretation*

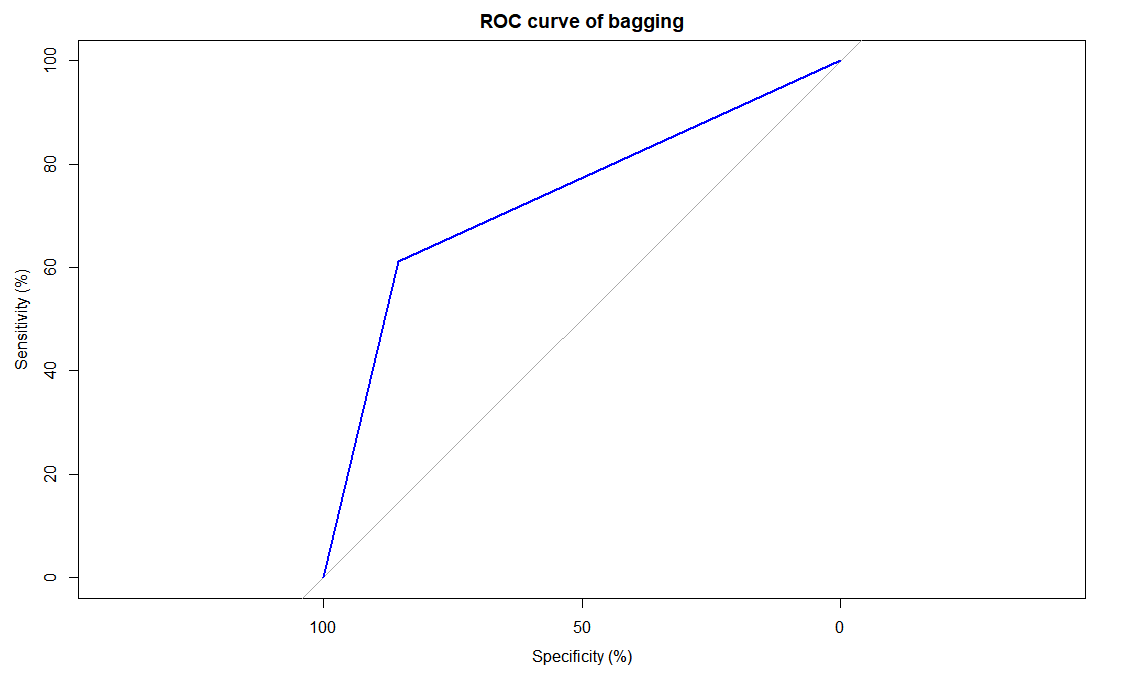
For the bar chart above, the variables “Sex”, “Age” and “Fare” are the most important variables. These 3 variables have higher prediction power on “Survived”

The bar chart showing the importance of variables in bagging model

*3. Prediction Power*

The model predicts Survived 0.8% incorrectly in the training data set, and 24% incorrectly in testing data set. Out of 76 passenger who did not survive in the testing data set, the model correctly predicts 65. Out of 49 passenger who survive in the testing data set, the model correctly predicts 30.

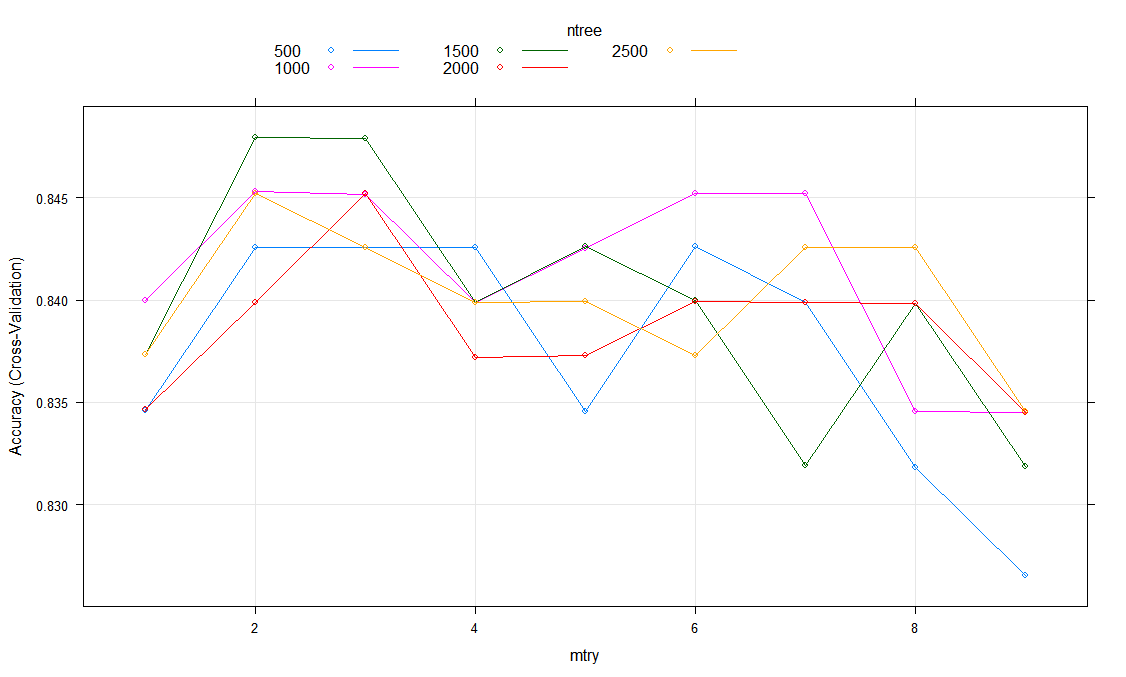
|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Class Prediction** | |
| **CV error** | | 0.1628212 | |
| **Training error** | | 0.008 | |
| **Testing error** | | 0.24 | |
| **AUC** | | 0.7338 | |
|  |  | |  |
|  | **predicted 0** | | **Predicted 1** |
| **real 0** | 65 | | 9 |
| **real 1** | 19 | | 30 |



Random Forest Model

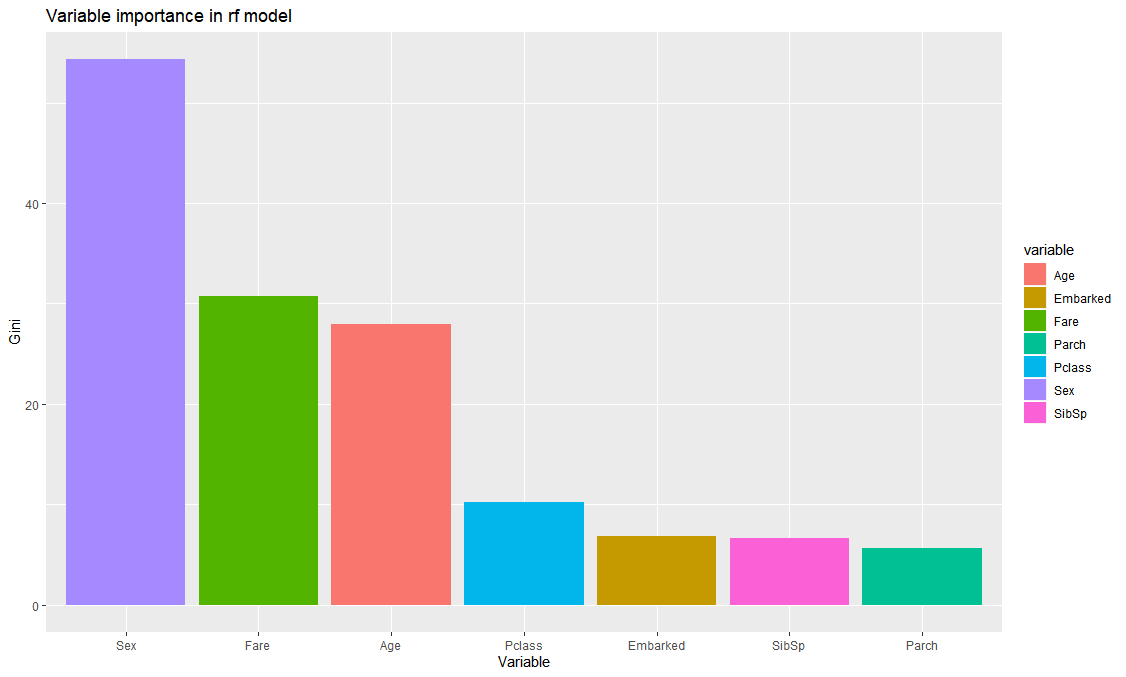
1. Model Selection

The major parameter in random forest are number of variables in each node (mtry) and the number of trees (ntree). We apply 5-folds cross validation to choose the best number of trees between 500 and 2500 and the best number of variables in each nodes from 1 to 7. Accuracy are used to measure the prediction power of the model. The best model should have the highest accuracy. We found that the model with 1500 trees and 2 variables is the best model with highest accuracy.

The plot showing the relationship between number of variable used in each node and the accuracy with different number of trees.

2.Model interpretation

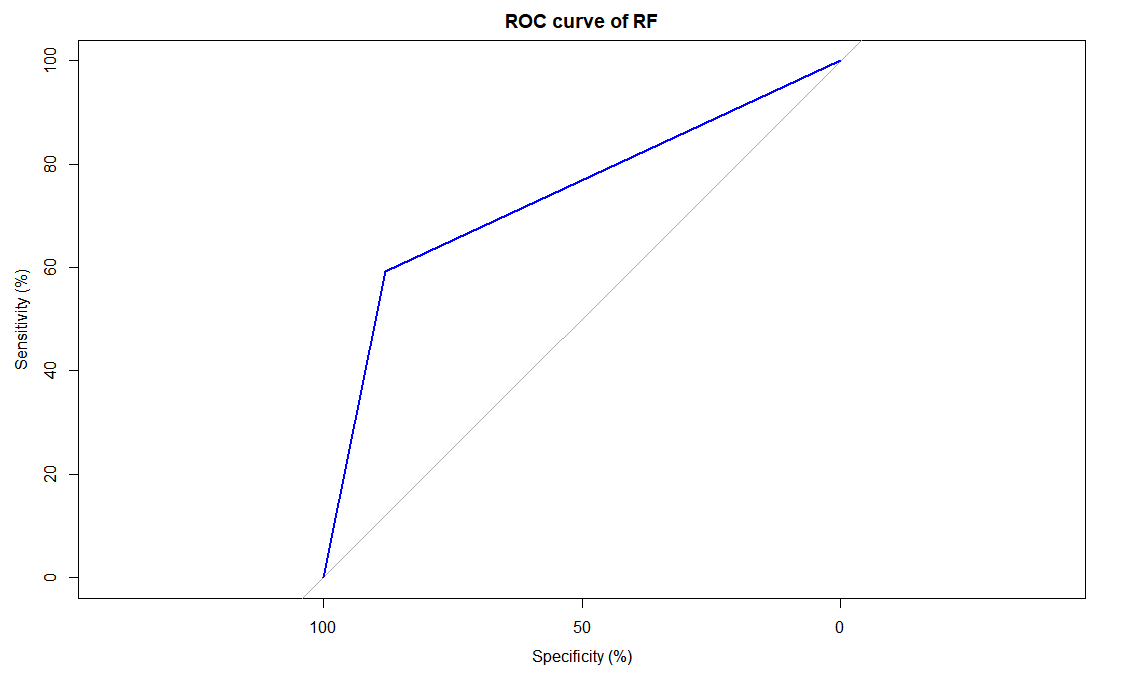
From the bar chart, We know that the variables “Sex”, “Fare”, and “Age” are the most important variables. These 3 variables have the highest prediction power on “Survived”.

The bar chart showing the importance of variables in random forest model

3. Prediction Power

The model predicts Survived 0.064% incorrectly in the training data set, and 24% incorrectly in testing data set. Out of 76 passenger who did not survive in the testing data set, the model correctly predicts 67. Out of 49 passenger who survive in the testing data set, the model correctly predicts 28.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Class Prediction** | |
| **CV error** | | 0.1520465 | |
| **Training error** | | 0.064 | |
| **Testing error** | | 0.24 | |
| **AUC** | | 0.7367 | |
|  | **predicted 0** | | **Predicted 1** |
| **real 0** | 67 | | 9 |
| **real 1** | 21 | | 28 |



Logistic Regression Model

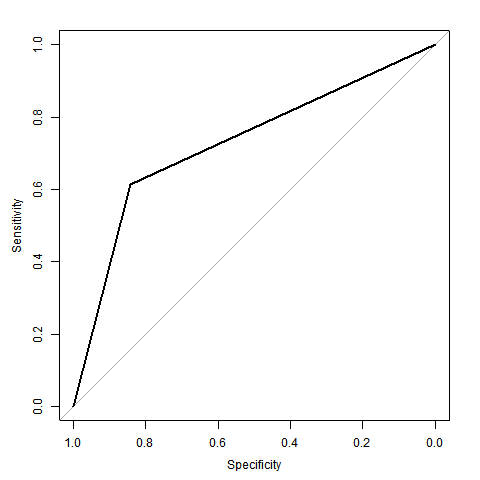
*1. Model interpretation*

The table below estimated the coefficients in the logistic function. Only Sexmale, Pclass, Age are significant variables and have strong confidence in predicting Survived. The model shows that being a male, being older, or holding a lower class ticket would negatively contribute to the probability to survive, while being a female and embarking in Cherbourg positively contribute to the probability to survive.

Paying lower fare, embarking in Southampton, or having more siblings negatively affect the probability to survive, while embarking in Queenstown, or having more parents or kids on board positively affect the probability to survive. However, there is no strong evidence of such effects.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Estimate** | **Std. Error** | **z value** | **Pr(>|z|)** |
| **Sexmale** | -2.978 | 0.305 | -9.749 | 1.86E-22 |
| **(Intercept)** | 5.183 | 0.936 | 5.540 | 3.03E-08 |
| **Pclass** | -1.232 | 0.255 | -4.831 | 1.36E-06 |
| **Age** | -0.037 | 0.013 | -2.983 | 0.003 |
| **EmbarkedQ** | 1.007 | 0.650 | 1.550 | 0.121 |
| **Parch** | 0.132 | 0.218 | 0.604 | 0.546 |
| **Fare** | -0.002 | 0.004 | -0.513 | 0.608 |
| **EmbarkedS** | -0.197 | 0.386 | -0.511 | 0.609 |
| **SibSp** | -0.063 | 0.160 | -0.392 | 0.695 |

*2.Predictive Power*

Since the model has no hyperparameter, cv error is only used to estimate testing error. The model predicts incorrectly 17.1% of the time in the training data set, and 24.8% incorrectly in testing data set. Out of 76 passenger who did not survive in the testing data set, the model correctly predicts 64. Out of 49 passenger who survive in the testing data set, the model correctly predicts 30.

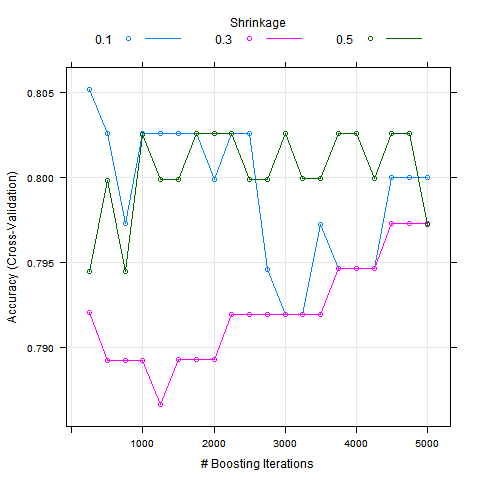
|  |  |  |
| --- | --- | --- |
|  | **Class Prediction** | **Prob. Prediction** |
| **CV error** | 0.184 | 0.132 |
| **Training error** | 0.171 | 0.125 |
| **Testing error** | 0.248 | 0.189 |
| **AUC** | 0.727 | 0.745 |

|  |  |  |
| --- | --- | --- |
|  | **predicted 0** | **Predicted 1** |
| **real 0** | 64 | 12 |
| **real 1** | 19 | 30 |

Boosting Model

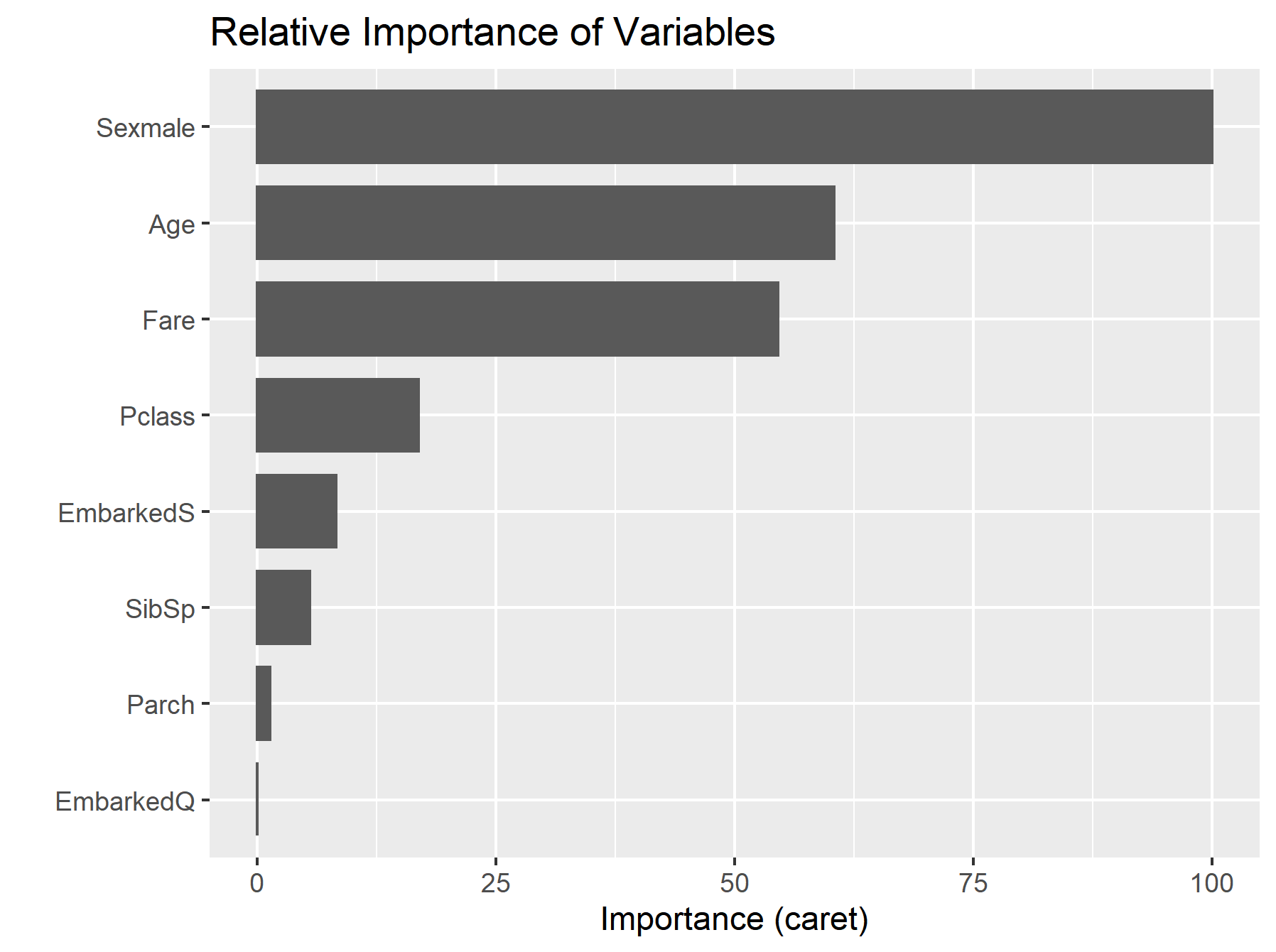
*1. Model Selection*

After Tuning for hyperparameters Shrinkage and Boosting iterations, it is found that Shrinkage = 0.1 and Boosting iterations = 250 maximize accuracy. However, the accuracy of various parameter settings only differs slightly. The performance of the model is not very sensitive towards parameters. The worst parameter setting yields 78.7% accuracy, which is not very different from the 80.5% of the best.

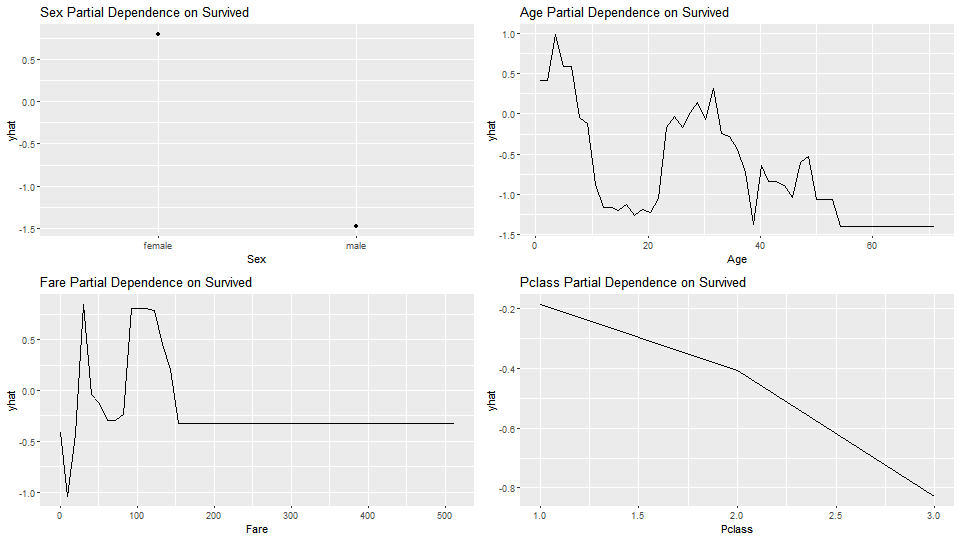


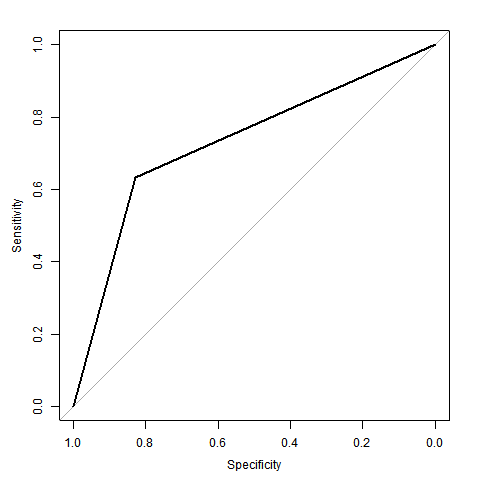
*2 Model Interpretation*

The model found that Sex, Age, Fare, Pclass are the most important variables in predicting whether the passenger survived.



The plots below show these variables’ effect on prediction. Being female increases the probability to survive. Infants, kid, adults are more likely to survive. Passengers who paid a fare around 40 or 100 are more likely to survive. Higher class ticket holders are more likely to survive.

*3.Predictive Power*

Although the model is very predictive in training data set with a misclassification rate of 0.8%, in-sample accuracy is not a reliable estimate of out-sample testing error. The cross validation error of 19.5% misclassification rate is a better estimate for the actual testing data. The model predicts incorrectly 24.8% of the time in testing data set. Out of 76 passenger who did not survive in the testing data set, the model correctly predicts 63. Out of 49 passenger who survive in the testing data set, the model correctly predicts 31.

|  |  |
| --- | --- |
|  | **Class Prediction** |
| **CV error** | 0.195 |
| **Training error** | 0.008 |
| **Testing error** | 0.248 |
| **AUC** | 0.731 |

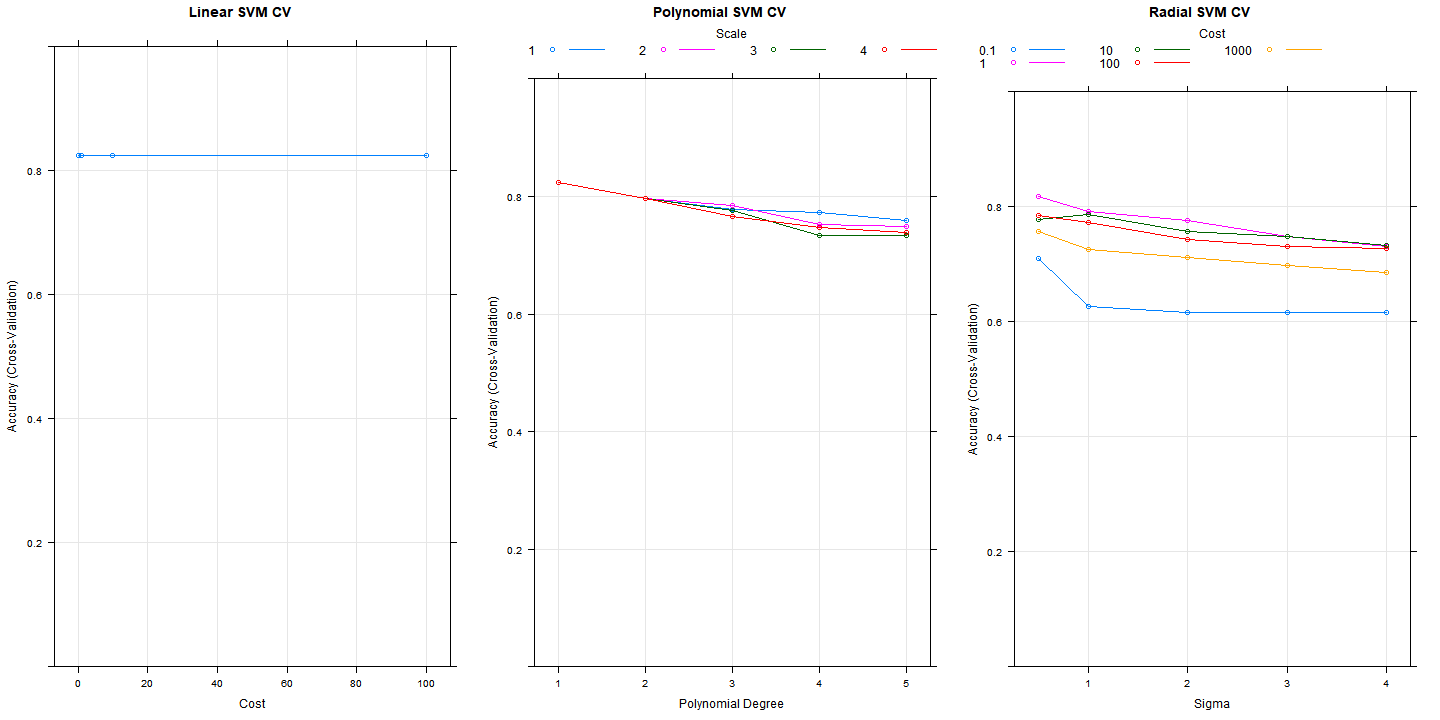
|  |  |  |
| --- | --- | --- |
|  | **predicted 0** | **predicted 1** |
| **real 0** | 63 | 13 |
| **real 1** | 18 | 31 |

Support Vector Machine Model

*1. Model Selection*

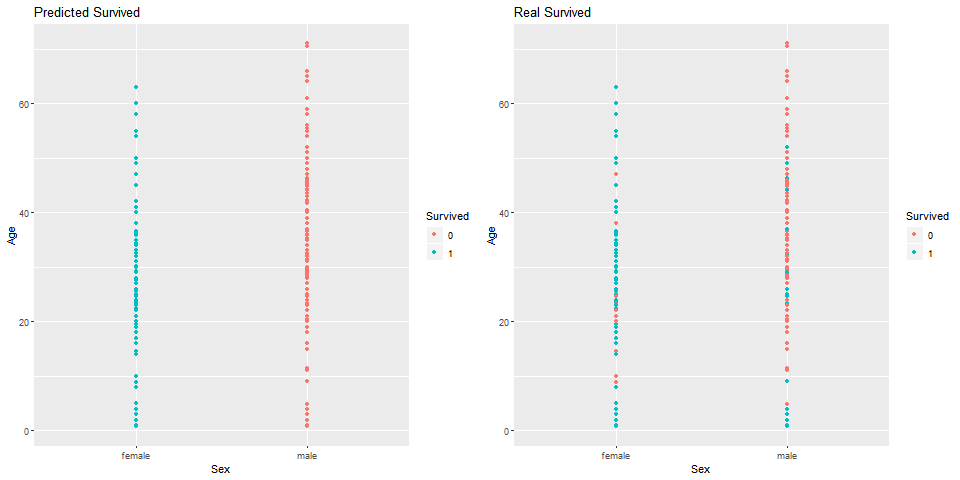
Linear, polynomial, radial kernel with different hyperparameters are cross validated. It is found that the simplest models perform the best. Linear kernel and 1st degree polynomial kernels produce the same accuracy of 82.4%. In view of this, the simplest model would be adopted to fit all training data, that is Linear Kernel with Cost = 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kernel Function** | **Cost** | **scale** | **degree** | **Accuracy** |
| **Linear** | 0.1 | NA | NA | 82.4% |
| **Linear** | 1 | NA | NA | 82.4% |
| **Linear** | 10 | NA | NA | 82.4% |
| **Linear** | 100 | NA | NA | 82.4% |
| **Polynomial** | 1 | 1 | 1 | 82.4% |
| **Polynomial** | 1 | 2 | 1 | 82.4% |
| **Polynomial** | 1 | 3 | 1 | 82.4% |
| **Polynomial** | 1 | 4 | 1 | 82.4% |



*2. Model Interpretation*

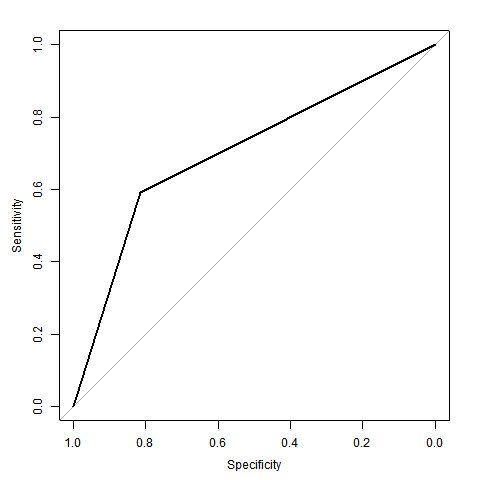
Since Sex is such a strong predictor of Survived, the model simply draws a hyperplane that separate male and females. All males are classified as not Survived and all females Survived.



*3. Predictive Power*

The cross validation error is very close to the training error, both scoring 17.6%. The Testing error is larger, reaching 27.6%. Out of 76 passenger who did not survive in the testing data set, the model correctly predicts 62. Out of 49 passenger who survive in the testing data set, the model correctly predicts 29.

|  |  |
| --- | --- |
|  | **Class Prediction** |
| **CV error** | 17.6% |
| **Training error** | 17.6% |
| **Testing error** | 27.2% |
| **AUC** | 70.4% |



|  |  |  |
| --- | --- | --- |
|  | **predicted 0** | **predicted 1** |
| **real 0** | 62 | 14 |
| **real 1** | 20 | 29 |

Conclusion

All of the classification models we used are reasonably accurate. The best of all, Random Forest, cassfication tree and bagging have the best misclassification rate 24%, while knn are the worst of all. Still, knn scores 28% testing error only. In this data set, blind guessing all passenger to not survived would misclassify 39.2% of the time. In this sense, all models outperform the benchmark of “blind guessing”.

It is worth noting that, Tree-based methods, namely Classification Tree, Random Forest, Bagging, Boosting, achieved very low training error, but their testing errors are nowhere near. This shows that they are overfitted to some extend, especially for Bagging and Boosting which yields training error of 0.8%.

On the other hand, we access the AUC value of each model. AUC tells how well the model distinguish the observation to to different classes. The simplest way to interpret AUC is that the better model, the higher AUC. From the table, random forest has the highest AUC with 0.737 while the knn has the lowest AUC with 0.672. Which means in general case prediction, we expect random forest has the best performance on prediction accuracy while the knn model has the lowest prediction accuracy.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **CV Error** | **Training Error** | **Testing Error** | **AUC** |
| **Random Forest** | 15.2% | 6.4% | 24.0% | 0.737 |
| **KNN** | 28.8% | 21.6% | 28.0% | 0.672 |
| **Classification Tree** | 13.9% | 13.9% | 24.0% | 0.719 |
| **Bagging** | 16.3% | 0.8% | 24.0% | 0.734 |
| **Logistic Regression** | 18.4% | 17.1% | 24.8% | 0.727 |
| **Boosting** | 19.5% | 0.8% | 24.8% | 0.731 |
| **Support Vector Machine** | 17.6% | 17.6% | 27.2% | 0.704 |