STAT4001

Data Mining and Stat Learning

Final group project
Academic Year 2019~2020
Group 13

CHENG Wing Ryan 1155102964
Hung Fan Hin 1155093869
Lee Ka Hin 1155092646
Li Tak Leong 1155094278

Table of Content

1. Introduction

This report is divided into two parts:

1. Home buyers were asked about their dream house, several factors were described. These factors are going to be analyzed by different methods, and be used to predict the final price of the houses.

2. On April 15, 1912, the sinking of the Titanic caused 1502 deaths out of 2224 passengers and crews. Some factors are going to be analyzed, to see whether some groups of people were more able to survival than the others. Several methods are used to predict the survival chance of different passengers.

2. Dream House

a. Data Description

i. Data Structure

```
> str(house)
'data.frame':   500 obs. of  15 variables:
 $ Id          : int  1 2 3 4 5 6 7 8 9 10 ...
 $ LotFrontage : int  65 80 68 60 84 85 75 NA 51 50 ...
 $ LotArea     : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
 $ LotShape    : Factor w/ 4 levels "IR1","IR2","IR3",..: 4 4 1 1 1 1 4 1 4 4 ...
 $ OverallCond : int  5 8 5 5 5 5 5 6 5 6 ...
 $ MasVnrArea  : int  196 0 162 0 350 0 186 240 0 0 ...
 $ TotalBsmtSF : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
 $ X1stFlrSF   : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
 $ X2ndFlrSF   : int  854 0 866 756 1053 566 0 983 752 0 ...
 $ GrLivArea   : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
 $ TotRmsAbvGrd: int  8 6 6 7 9 5 7 7 8 5 ...
 $ GarageArea  : int  548 460 608 642 836 480 636 484 468 205 ...
 $ WoodDeckSF  : int  0 298 0 0 192 40 255 235 90 0 ...
 $ OpenPorchSF : int  61 0 42 35 84 30 57 204 0 4 ...
 $ SalePrice   : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000
```

Figure1

From the structure of "House.xlsx, there are 500 observations and 15 variables. The "SalePrice" is the dependent variable and the others are independent variables. "Id" is omitted in the model fitting and prediction process.

ii. Variables Correlation

Before any model is fitted, it is better to generate a correlation matrix to understand the relationship of the variables. "LotShape", a factor variable, is transformed into an integer variable so as to generate a correlation matrix.

However, there are NAs in several observations. In order to avoid bias, NAs are omitted in the correlation matrix computation. The correlation matrix is shown below.

	Id	LotFrontage	LotArea	LotShape	OverallCond	MasVnrArea	TotalBsmtSF	X1stFlrSF	X2ndFlrSF	GrLivArea	TotRmsAbvGrd	GarageArea	WoodDeckSF	OpenPorchSF	SalePrice
Id	1.0000000000	-0.03960341	0.019225620	0.02086340	0.071972032	-0.04647010	0.01489591	-0.04328554	0.009948307	-0.02910625	-0.01865534	-0.0258788	-0.0006792171	0.012831353	0.002859221
LotFrontage	-0.0396034059	1.00000000	0.377780072	-0.19484521	-0.137566649	0.17650945	0.36780439	0.40971294	0.109801417	0.40574465	0.36281455	0.3812756	0.1119367753	0.142384364	0.427655853
LotArea	0.0192256199	0.37778007	1.000000000	-0.12793624	-0.064787796	0.00328310	0.25692501	0.27375740	-0.003847152	0.19979832	0.14657921	0.1653141	0.0880300601	0.009744772	0.295326480
LotShape	0.0208634036	-0.19484521	-0.127936243	1.00000000	0.109103298	-0.12465004	-0.21365525	-0.19200884	-0.089595078	-0.21714805	-0.14395882	-0.2009072	-0.2099957523	-0.096770773	-0.279053410
OverallCond	0.0719720318	-0.13756665	-0.064787796	0.10910330	1.000000000	-0.17094583	-0.23458147	-0.21113505	-0.003823083	-0.14957113	-0.11217097	-0.1994241	-0.0539569472	-0.092858436	-0.160381132
MasVnrArea	-0.0464701049	0.17650945	0.003283100	-0.12465004	-0.170945826	1.00000000	0.35020178	0.32827133	0.173604532	0.38223293	0.31896337	0.3500395	0.1801592786	0.096806131	0.471017913
TotalBsmtSF	0.0148959133	0.36780439	0.256925013	-0.21365525	-0.234581466	0.35020178	1.00000000	0.83054658	-0.210550697	0.41993786	0.32591923	0.4885616	0.2664995893	0.159767277	0.67085968
X1stFlrSF	-0.0432855423	0.40971294	0.273757396	-0.19200884	-0.211135053	0.32827133	0.83054658	1.00000000	-0.254800931	0.50688374	0.40155650	0.4758677	0.2792270858	0.122028118	0.652023012
X2ndFlrSF	0.0099483065	0.10980142	-0.003847152	-0.08959508	-0.003823083	0.17360453	-0.21055070	-0.25480098	1.000000000	0.69592605	0.61558757	0.1870730	0.0655625535	0.223097678	0.319515343
GrLivArea	-0.0291062517	0.40574465	0.199798319	-0.21714805	-0.149571134	0.38223293	0.41993786	0.50688374	0.695926053	1.00000000	0.84499944	0.5036839	0.2550982852	0.289380645	0.757805017
TotRmsAbvGrd	-0.0186553363	0.36281455	0.146579207	-0.14395882	-0.112170968	0.31896337	0.32591923	0.40155650	0.615587572	0.84499944	1.00000000	0.3915961	0.1448109912	0.243564721	0.607749971
GarageArea	-0.0258787979	0.38127557	0.165314117	-0.20090716	-0.199424142	0.35003955	0.48856156	0.47586765	0.187073008	0.63568388	0.39159610	1.0000000	0.2796213107	0.211059501	0.699490051
WoodDeckSF	-0.0006792171	0.11193678	0.088030060	-0.20999575	-0.053956947	0.18015928	0.26649959	0.27922709	0.065562553	0.25509829	0.14481099	0.2796213	1.0000000000	0.116928642	0.334298216
OpenPorchSF	0.0128313534	0.14238436	0.009744772	-0.09677077	-0.092858436	0.09680613	0.15976728	0.12202812	0.223097678	0.28938064	0.24356472	0.2110595	0.1169286420	1.000000000	0.323630424
SalePrice	0.0028592208	0.42765585	0.295326480	-0.27905341	-0.160381132	0.47101798	0.67085968	0.65202301	0.319515343	0.75780850	0.60774998	0.6994901	0.3542982156	0.323630424	1.000000000

Figure2

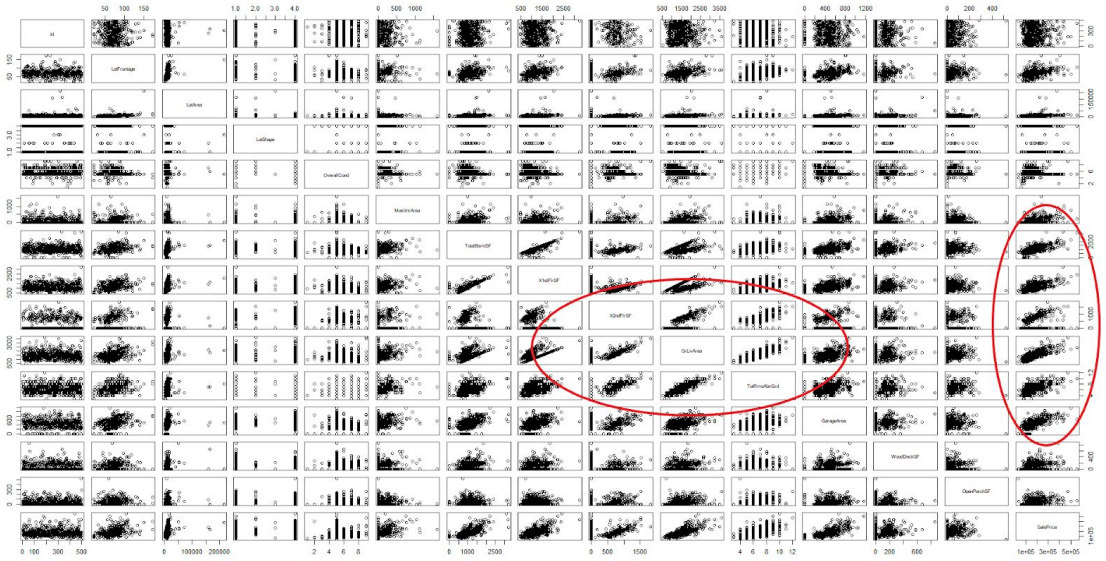


Figure3

After omitting the observations which contain missing value, A correlation matrix can be generated. From the general plot and the correlation matrix, the correlation coefficients between some variables and “SalePrice” are respectively higher. They are “GrLivArea”, “TotalBsmtSF”, “X1stFlrSF”, “TotRmsAbvGrd” and “GarageArea” etc. The correlation coefficients are over 0.6. We hypothesized that those variables will contribute more in model fitting and prediction.

However, the 2 figures also show that “X2ndFlrSF”, “GrLivArea” and “TotRmsAbvGrd” are also highly correlated. The phenomenon may affect the result of regression.

iii. Handling Missing Data

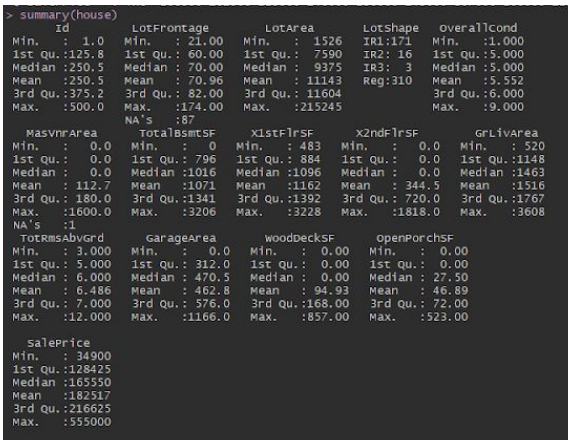


Figure5

From the summary of the dataset "House.xlsx", there are 87 NAs in variable "LotFrontage" and 1 NA in "MasVnrArea. A linear regression model is fitted for "LotFrontage" against other variables without "Id". However, the R-squared of the model is only 0.3451. To simplify the model, backword stepwise selection is applied.

After the simulation, the following model gives the lowest AIC.

Figure6

Another linear model is fitted with the formula of the above model. The model gives 0.3412 R-squared. while the NAs are predicted by the model, the predicted values are rounded off since "LotFrontage" is integer. The NAs values of "LotFrontage" are substituted by the predicted value.

The same method is applied to the NA of "MasVnrArea". A linear regression model is fitted with the formula: "MasVnrArea ~ ." The NA value of "MasVnrArea" is substituted by the predicted value.

b. Modeling

i. *Linear Regression*

Linear regression is using the least square method to minimize the residuals between a dependent variable (usually presented as y) and one or more independent variables (usually presented as x). The model is fitted using linear predictor functions whose unknown coefficients are estimated from the data through Linear regression.

```
numfolds = trainControl( method = "cv", number = 10)
cv_lin_reg = train(SalePrice ~ ., data = house_train,method = 'lm',trControl = numfolds)
```

Figure7

By using 10-folds cross-validation, we analyze the house training data set through linear regression.

```
> summary(cv_lin_reg)

Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-106510  -17523   -2107   17779  181331

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -5.773e+04  1.409e+04  -4.098 5.09e-05 ***
LotFrontage  -1.239e+02  1.078e+02  -1.149  0.25115
LotArea       2.689e-01  1.540e-01   1.747  0.08151 .
LotShapeIR2   2.201e+04  9.696e+03   2.270  0.02379 *
LotShapeIR3   9.215e+03  2.922e+04   0.315  0.75263
LotShapeReg  -7.385e+03  3.927e+03  -1.880  0.06080 .
OverallCond   5.049e+03  1.595e+03   3.166  0.00167 **
MasVnrArea    4.487e+01  1.039e+01   4.318 2.01e-05 ***
TotalBsmtSF   5.172e+01  7.423e+00   6.968 1.40e-11 ***
X1stFlrSF     5.953e+01  2.917e+01   2.041  0.04198 *
X2ndFlrSF     5.167e+01  2.833e+01   1.824  0.06893 .
GrLivArea     2.169e+01  2.781e+01   0.780  0.43590
TotRmsAbvGrd -1.093e+03  2.167e+03  -0.505  0.61415
GarageArea    9.598e+01  1.079e+01   8.891  < 2e-16 ***
WoodDeckSF    1.892e+01  1.559e+01   1.213  0.22576
OpenPorchSF   6.590e+01  2.925e+01   2.253  0.02480 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 35390 on 384 degrees of freedom
Multiple R-squared:  0.7975,    Adjusted R-squared:  0.7895
F-statistic: 100.8 on 15 and 384 DF,  p-value: < 2.2e-16
```

Figure8

Some of the variables including “LotFrontage”, “LotShapeIR3”, “GrLivArea”, “TotRmsAbvGrd” and “WoodDeckSF” are not statistical significantly by linear regression.

Figure9

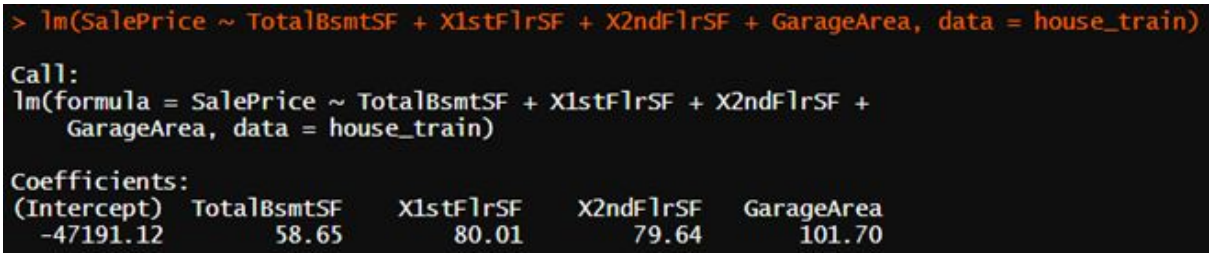


Figure10

The RMSE of each round is shown above. Using Stepwise regression, the AIC is around 9000 on average. The best tune of the linear regression is four, including “TotalBsmtSF", " X1stFlrSF", "X2ndFlrSF" and "GarageArea" through modelling selection.Comparing the AIC and BIC of the lm1(before tuning) and lm2(after tuning), we can see that BIC decreased while AIC increased. One possible is the high cross-validation prediction error.

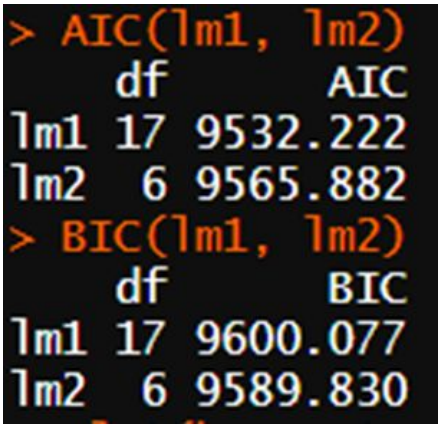


Figure11

cross-validation

```
> print(cv_lin_reg)
Linear Regression

400 samples
 13 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 360, 360, 360, 360, 360, 360, ...
Resampling results:

  RMSE      Rsquared   MAE
  37088.35  0.7864601  27035.66

Tuning parameter 'intercept' was held constant at a value of TRUE
```

Figure12

	RMSE <dbl>
RMSE of 1 st folds	29654.37
RMSE of 2 st folds	33912.28
RMSE of 3 st folds	35389.20
RMSE of 4 st folds	31047.64
RMSE of 5 st folds	42714.44
RMSE of 6 st folds	37140.67
RMSE of 7 st folds	36799.68
RMSE of 8 st folds	39586.24
RMSE of 9 st folds	41019.15
RMSE of 10 st folds	36069.85

Figure13

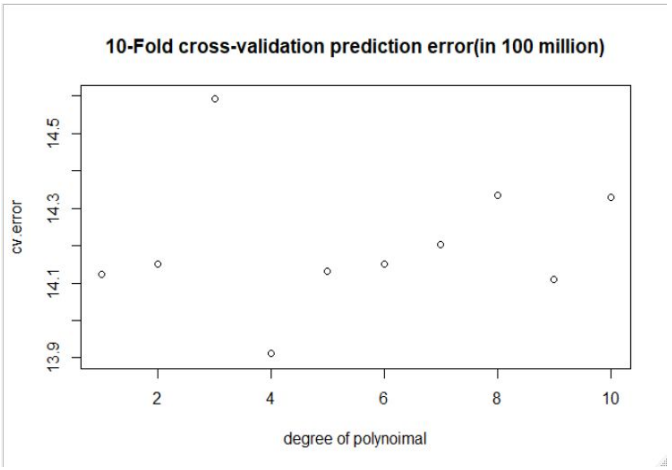


Figure14

A 10-folds cross-validation is applied to find out the best predictive model in 400 samples. Regarding the graph, the root-mean-square error (RMSE) is 37088.35. Also, the cross-validation prediction errors are extremely large, each value is more than 1 billion. The smallest one is around 1.39 billion (the fourth one), but still very large. It is believed that the complexity and diversity of data type contribute to the skyhigh prediction error.

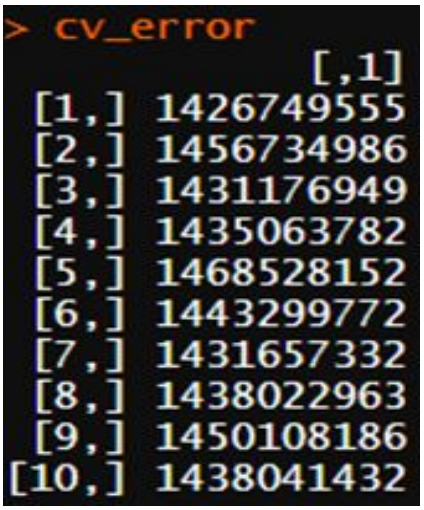


Figure15

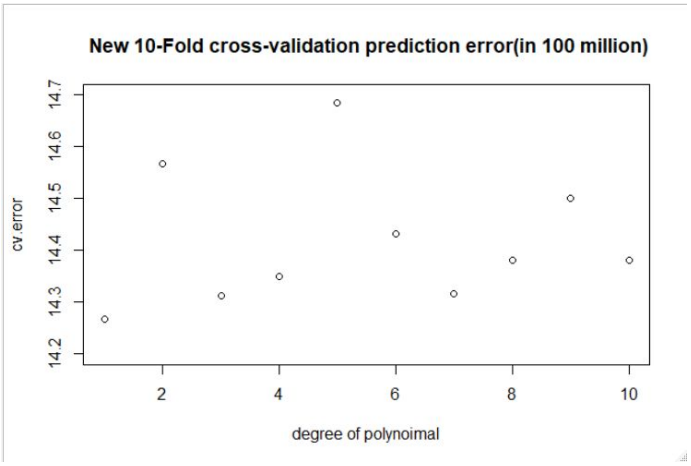


Figure16

Although the cross-validation prediction errors are still large,after the stepwise regression, they are closer to each other, meaning that the model still have rooms of improvement.

Interpretation

In view of the enormous value of cross-validation prediction error and Root-mean-squared Error, It is obvious that Linear Regression cannot predict the result in an accurate and precise way which makes sense as not all variable are collinear with Sale price.

iii. *Ridge Regression*

Introduction

Ridge regression is very similar to linear regression. The difference is that ridge regression implements a penalty lambda on MSE. The higher the lambda will give the weaker the contributes of all coefficients in the model. The selection of turning parameter lambda is the main focus in this section where

$$\text{Error} = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

Grid of Lamda

The best RMSE of ridge regression model is 36 316.

```
> grid = 10^seq(12,-2,length=100)
> grid
  [1] 1.000000e+12 7.220809e+11 5.214008e+11 3.764936e+11 2.718588e+11 1.963041e+11 1.417474e+11 1.023531e+11 7.390722e+10
 [10] 5.336699e+10 3.853529e+10 2.782559e+10 2.009233e+10 1.450829e+10 1.047616e+10 7.564633e+09 5.462277e+09 3.944206e+09
 [19] 2.848036e+09 2.056512e+09 1.484968e+09 1.072267e+09 7.742637e+08 5.590810e+08 4.037017e+08 2.915053e+08 2.104904e+08
 [28] 1.519911e+08 1.097499e+08 7.924829e+07 5.722368e+07 4.132012e+07 2.983647e+07 2.154435e+07 1.555676e+07 1.123324e+07
 [37] 8.111308e+06 5.857021e+06 4.229243e+06 3.053856e+06 2.205131e+06 1.592283e+06 1.149757e+06 8.302176e+05 5.994843e+05
 [46] 4.328761e+05 3.125716e+05 2.257020e+05 1.629751e+05 1.176812e+05 8.497534e+04 6.135907e+04 4.430621e+04 3.199267e+04
 [55] 2.310130e+04 1.668101e+04 1.204504e+04 8.697490e+03 6.280291e+03 4.534879e+03 3.274549e+03 2.364489e+03 1.707353e+03
 [64] 1.232847e+03 8.902151e+02 6.428073e+02 4.641589e+02 3.351603e+02 2.420128e+02 1.747528e+02 1.261857e+02 9.111628e+01
 [73] 6.579332e+01 4.750810e+01 3.430469e+01 2.477076e+01 1.788650e+01 1.291550e+01 9.326033e+00 6.734151e+00 4.862602e+00
 [82] 3.511192e+00 2.535364e+00 1.830738e+00 1.321941e+00 9.545485e-01 6.892612e-01 4.977024e-01 3.593814e-01 2.595024e-01
 [91] 1.873817e-01 1.353048e-01 9.770100e-02 7.054802e-02 5.094138e-02 3.678380e-02 2.656088e-02 1.917910e-02 1.384886e-02
[100] 1.000000e-02
```

Figure17

The grid of lambda is chosen as the above figure. 100 ridge regression models are fitted with the gird.

Cross-validation

```
> sqrt(cv.out$cvm)
  [1] 77324.87 77037.34 76984.44 76949.92 76912.09 76870.64 76825.22 76775.47 76720.98 76661.32 76596.00 76524.52 76446.29 76360.73
 [15] 76267.16 76164.87 76053.10 75931.02 75797.73 75652.27 75493.63 75320.72 75132.36 74927.32 74704.30 74461.93 74198.76 73913.30
 [29] 73604.00 73269.24 72907.41 72516.86 72095.93 71643.00 71156.50 70634.92 70076.88 69481.14 68846.66 68172.64 67458.56 66704.23
 [43] 65909.86 65076.09 64204.02 63295.29 62352.08 61377.13 60373.76 59345.84 58297.80 57234.54 56161.40 55084.04 54008.39 52940.48
 [57] 51886.36 50851.84 49842.54 48863.83 47920.36 47016.30 46155.09 45339.45 44571.35 43852.00 43181.91 42560.91 41988.21 41462.51
 [71] 40982.10 40544.90 40148.61 39790.74 39468.74 39180.02 38922.17 38692.80 38489.10 38308.91 38150.08 38010.63 37888.68 37782.51
 [85] 37690.50 37611.25 37543.80 37486.44 37438.00 37397.62 37364.51 37337.85 37316.13 37300.02 37287.89 37279.58 37274.49 37271.94
 [99] 37271.84 37269.08
>
```

Figure18

The figure shows the RMSE of 100 different lambda in cross validation. A 10-folds cross-validation is applied to find out the best predictive model in 100 models. As shown below, there are 2 recommended lambda values which pointed out by 2 dotted lines. The left one is the lambda with the lowest MSE and the right one is the lambda with MSE above 1 sd of the lowest one. Therefore, the left lambda is chosen to be the best lambda. The value is around 7646.617 and the log value is around 8.942.

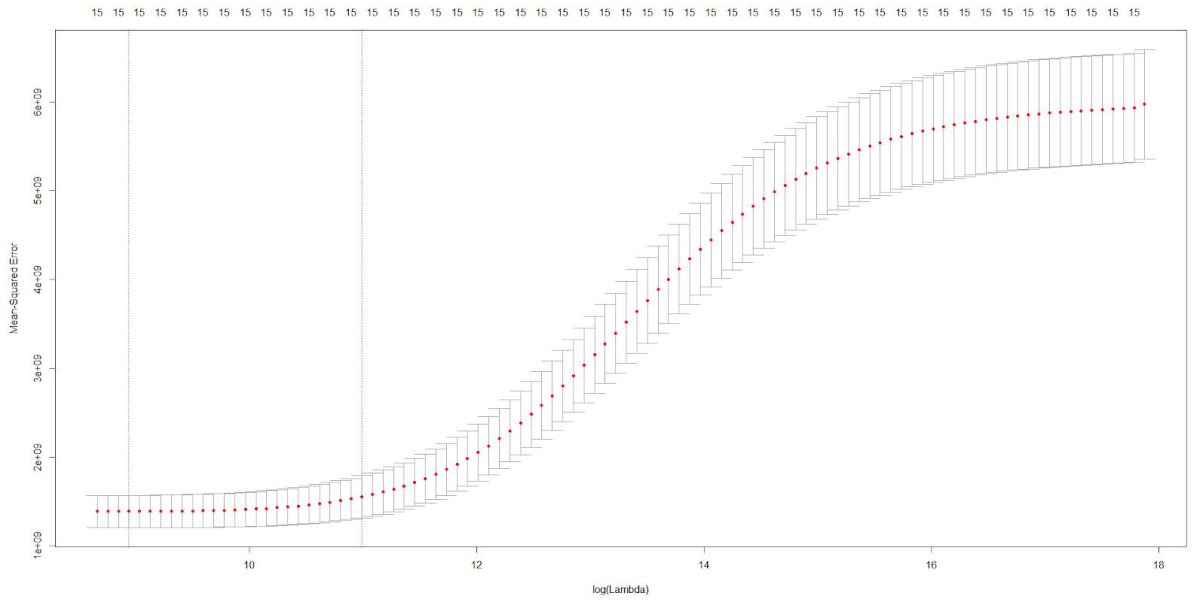


Figure19

Best Model

The best model is fitted with lamda 7646.617 and RMSE 36 316.

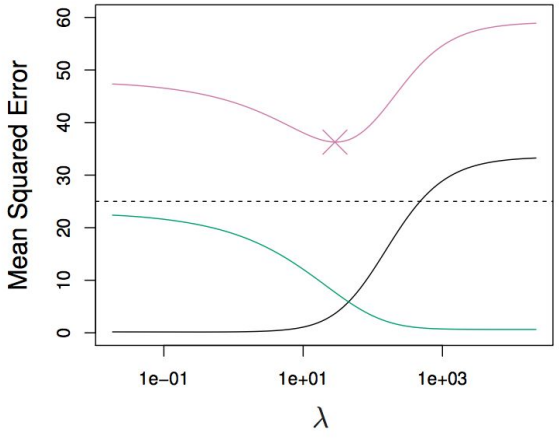


Figure20

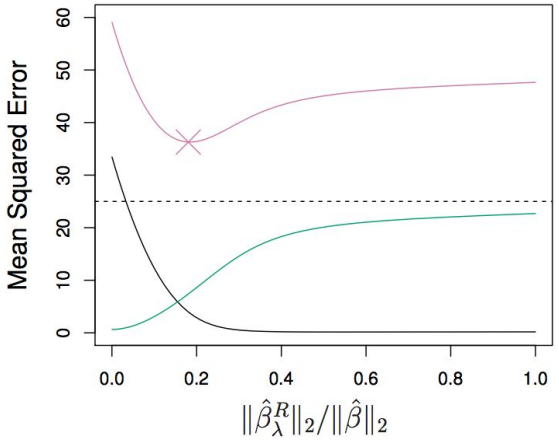


Figure21

Interpretation

In grid, lamda values are arranged in descending order. The 20th lamda value is 49770236 and the 50th lamda value is 11497.57. Therefore, the best lamda 7646.617 is respectively small. The ridge regression is not far from linear regression. The bias and MSE should be in optimum, the red cross in the above figure.

iv. *Lasso Regression*

Introduction

Lasso regression is very similar to ridge regression. Similarly, lasso regression implements a penalty lambda on MSE. The higher the lambda will give the weaker the contributes of all coefficients in the model. The selection of turning parameter lambda is also the main focus in this section where

$$\text{Error} = RSS + \lambda \sum_{j=1}^{p} |\beta_j|$$

Grid of Lamda

The best RMSE of ridge regression model is 35 603.

```
> grid = 10^seq(12,-2,length=100)
> grid
  [1] 1.000000e+12 7.220809e+11 5.214008e+11 3.764936e+11 2.718588e+11 1.963041e+11 1.417474e+11 1.023531e+11 7.390722e+10
 [10] 5.336699e+10 3.853529e+10 2.782559e+10 2.009233e+10 1.450829e+10 1.047616e+10 7.564633e+09 5.462277e+09 3.944206e+09
 [19] 2.848036e+09 2.056512e+09 1.484968e+09 1.072267e+09 7.742637e+08 5.590810e+08 4.037017e+08 2.915053e+08 2.104904e+08
 [28] 1.519911e+08 1.097499e+08 7.924829e+07 5.722368e+07 4.132012e+07 2.983647e+07 2.154435e+07 1.555676e+07 1.123324e+07
 [37] 8.111308e+06 5.857021e+06 4.229243e+06 3.053856e+06 2.205131e+06 1.592283e+06 1.149757e+06 8.302176e+05 5.994843e+05
 [46] 4.328761e+05 3.125716e+05 2.257020e+05 1.629751e+05 1.176812e+05 8.497534e+04 6.135907e+04 4.430621e+04 3.199267e+04
 [55] 2.310130e+04 1.668101e+04 1.204504e+04 8.697490e+03 6.280291e+03 4.534879e+03 3.274549e+03 2.364489e+03 1.707353e+03
 [64] 1.232847e+03 8.902151e+02 6.428073e+02 4.641589e+02 3.351603e+02 2.420128e+02 1.747528e+02 1.261857e+02 9.111628e+01
 [73] 6.579332e+01 4.750810e+01 3.430469e+01 2.477076e+01 1.788650e+01 1.291550e+01 9.326033e+00 6.734151e+00 4.862602e+00
 [82] 3.511192e+00 2.535364e+00 1.830738e+00 1.321941e+00 9.545485e-01 6.892612e-01 4.977024e-01 3.593814e-01 2.595024e-01
 [91] 1.873817e-01 1.353048e-01 9.770100e-02 7.054802e-02 5.094138e-02 3.678380e-02 2.656088e-02 1.917910e-02 1.384886e-02
[100] 1.000000e-02
```

Figure22

The grid of lambda is the same as that of ridge regression model.

Original Model

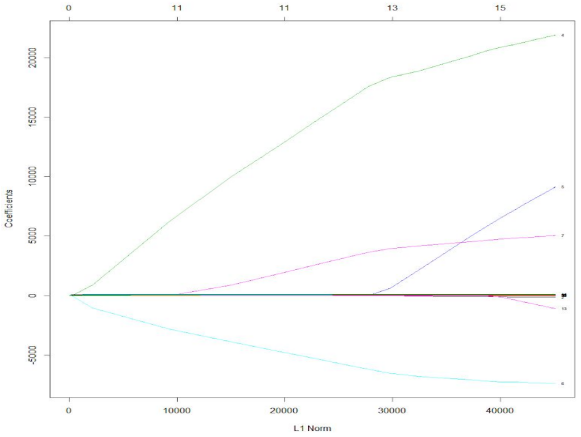


Figure23

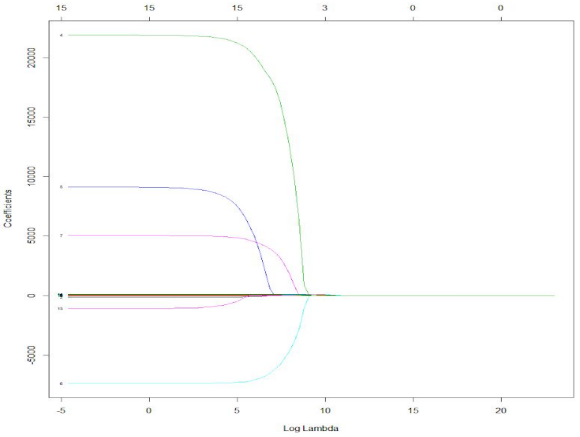


Figure24

Both figures show that the coefficient of lasso regression will decrease if the lambda increases. All coefficient will go to 0 if the log lambda is larger than 9. The figures also show that the 4th and 6th coefficient are firstly to be non-zero if lambda drops. They are corresponding to the coefficients of "LotShapeIR2" and "LotShapeReg". Therefore, the simplest model of lasso should be "SalePrice" against "LotShapeIR2" and "LotShapeReg".

Cross-validation

```
> sqrt(cv.out$cvm)
 [1] 77033.87 73731.66 70407.47 67197.03 63645.82 60149.73 57018.44 54282.00 51900.65 49837.24 48056.91 46527.13 45217.77 44101.20
[15] 43155.01 42359.96 41680.95 41089.20 40573.19 40126.52 39735.46 39398.05 39136.43 38922.78 38725.77 38544.75 38362.39 38194.01
[29] 38033.08 37888.14 37764.61 37641.04 37522.81 37422.23 37342.56 37279.28 37229.34 37190.85 37164.26 37158.84 37170.32 37189.25
[43] 37211.59 37239.54 37275.58 37313.32 37348.82 37384.27 37419.01 37449.28 37471.46 37489.15 37507.25 37527.25 37548.00 37567.48
[57] 37586.86 37607.65 37627.23 37635.90 37639.93 37643.94 37647.32 37652.54 37657.81 37662.70 37667.63 37672.22 37676.38 37680.52
[71] 37684.21 37687.82 37691.13 37693.57 37695.98
```

Figure25

The figure shows the RMSE of 100 different lambda in cross validation. Similar to ridge regression, A 10-folds cross-validation is applied to find out the best predictive model in 100 models. As shown below, there are 2 recommended lambda values which pointed out by 2 dotted lines. Similarly, the left one is the lambda with the lowest MSE and the right one is the lambda with MSE above 1 sd of the lowest one. The left lamda is chosen to be the best lambda. The value is around 1536.384 and the log value is around 7.337.

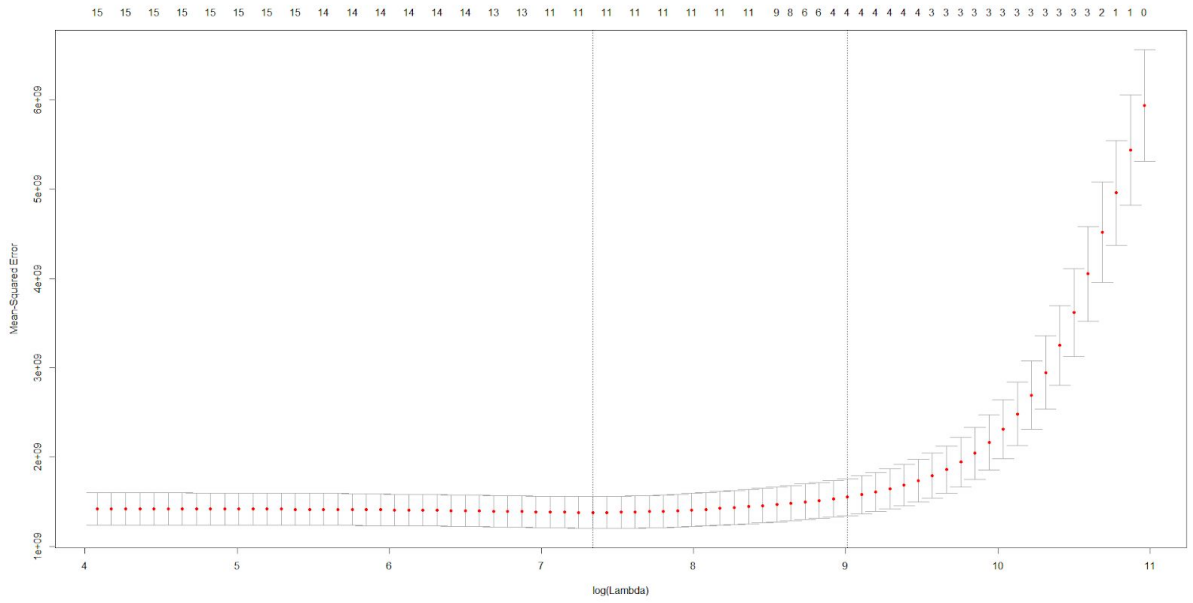


Figure26

Best Model

The best model is fitted with lamda 1536.384 and RMSE 35 603.

Interpretation

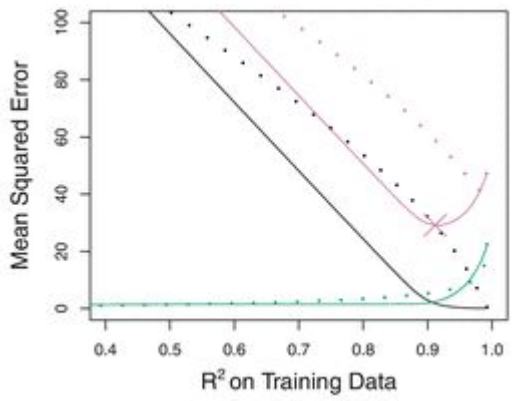


Figure27

As above mentioned, the RMSE from the best ridge regression is 36 316. Lasso regression performs better corresponding to RMSE. In conclusion, lasso gives less bias and variance than ridge in this case.

Remarks: Lasso(Solid), Ridge(Dash)

v. *Regression Spline*

introduction

Spline Regression is a non-parametric regression technique for testing non-linearity in the predictor variables and for modeling nonlinear functions. This regression technique divides the datasets into bins at intervals or points called knots and each bin has its separate fit.

Knots

Adding a greater number of knots can increase the flexibility of the Spline model. The choice of a knot placement depends on the distribution of each variable.

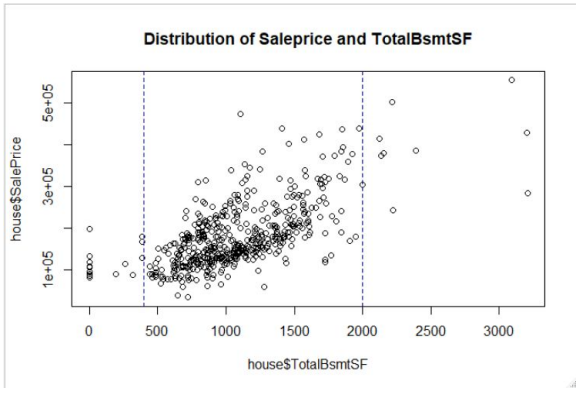


Figure28

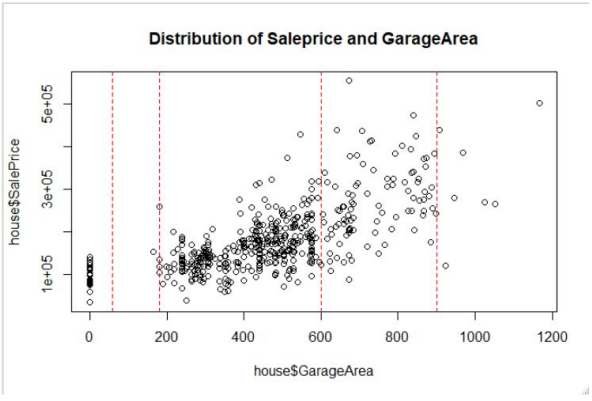


Figure29

Regarding the graph shown above, the distribution is split into several parts respectively with respect to their density.

Result:

```
>spline_reg = lm(SalePrice ~bs(LotFrontage, knots=quantile(house$SalePrice, p=c(0.25,0.5,1))) + bs(LotArea, knots=quantile(house$SalePrice, p=c(0.2,0.4,1))) + bs(MasVnrArea, knots=quantile(house$SalePrice, p=c(0.05,0.25,0.75,1))) + bs(TotalBsmtSF, knots=quantile(house$SalePrice, p=c(0.15,0.7,1))) + bs(X1stFlrSF, knots=quantile(house$SalePrice, p=c(0.7,1))) + bs(X2ndFlrSF, knots=quantile(house$SalePrice, p=c(0.05,0.3,0.7,1))) + bs(GrLivArea, knots=quantile(house$SalePrice, p=c(0.1, 0.75,1))) + bs(TotRmsAbvGrd, knots=quantile(house$SalePrice, p=c(0.3,0.4,0.5,0.6,0.7,0.8,0.9,1))) + bs(GarageArea, knots=quantile(house$SalePrice, p=c(0.05,0.15,0.5,0.75,1))) + bs(WoodDeckSF, knots=quantile(house$SalePrice, p=c(0.05,0.5,1))) + bs(OpenPorchSF, knots=quantile(house$SalePrice, p=c(0.05,0.6,1)))

Residuals:

   Min          1Q  Median      3Q   Max
```

```
-144132  -17520      1485   17529  111400

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 32120 on 360 degrees of freedom

Multiple R-squared:  0.8436,         Adjusted R-squared:  0.8267

F-statistic: 49.79 on 39 and 360 DF,  p-value: < 2.2e-16
```

We hypothesized that H null: the model is useless. Using the R output, the F-statistics shows that F-score is 49.79 and p-value is smaller than 2.2e-16 with 39 and 360 degree of the freedom. Therefore, the H null is rejected. The root-mean-square error (RMSE) is 122699.8. From R-squared value(0.8436 &0.8267), there is a strong relationship between House sale price and other variables. Therefore, 0.84 of the “SalePrice” can be explained and predicted by the regression spline.

Cross-Validation

10-folds cross validation is applied on the spline regression. The figure shows the RMSE of each fold of validation. The best performance of cross validation can give only 27492.74 which is relatively low. However, the code gives a warning message that some independent variables are out of the knots boundary so that the 6th entry are NaN because the knots are splitted by intuition.

RMSE <dbl>
35988.70
41167.51
26519.90
39574.96
NaN
34883.77
30605.46
36230.39
NaN
NaN

(Intercept)	61290.1
bs(LotFrontage, knots = quantile(house$SalePrice, p = c(0.25, 0.5, 1)))1	-16442.5
bs(LotFrontage, knots = quantile(house$SalePrice, p = c(0.25, 0.5, 1)))2	10072.6
bs(LotFrontage, knots = quantile(house$SalePrice, p = c(0.25, 0.5, 1)))3	-14759.2
bs(LotFrontage, knots = quantile(house$SalePrice, p = c(0.25, 0.5, 1)))4	NA
bs(LotFrontage, knots = quantile(house$SalePrice, p = c(0.25, 0.5, 1)))5	NA
bs(LotFrontage, knots = quantile(house$SalePrice, p = c(0.25, 0.5, 1)))6	NA
bs(LotArea, knots = quantile(house$SalePrice, p = c(0.2, 0.4, 1)))1	112889.5
bs(LotArea, knots = quantile(house$SalePrice, p = c(0.2, 0.4, 1)))2	-322589.0
bs(LotArea, knots = quantile(house$SalePrice, p = c(0.2, 0.4, 1)))3	2108438.8
bs(LotArea, knots = quantile(house$SalePrice, p = c(0.2, 0.4, 1)))4	NA
bs(LotArea, knots = quantile(house$SalePrice, p = c(0.2, 0.4, 1)))5	118643.5
bs(LotArea, knots = quantile(house$SalePrice, p = c(0.2, 0.4, 1)))6	NA
bs(MasVnrArea, knots = quantile(house$SalePrice, p = c(0.05, 0.25, 0.75, 1)))1	45807.9
bs(MasVnrArea, knots = quantile(house$SalePrice, p = c(0.05, 0.25, 0.75, 1)))2	-111192.9
bs(MasVnrArea, knots = quantile(house$SalePrice, p = c(0.05, 0.25, 0.75, 1)))3	497335.7
bs(MasVnrArea, knots = quantile(house$SalePrice, p = c(0.05, 0.25, 0.75, 1)))4	41332.4
bs(MasVnrArea, knots = quantile(house$SalePrice, p = c(0.05, 0.25, 0.75, 1)))5	NA
bs(MasVnrArea, knots = quantile(house$SalePrice, p = c(0.05, 0.25, 0.75, 1)))6	NA
bs(MasVnrArea, knots = quantile(house$SalePrice, p = c(0.05, 0.25, 0.75, 1)))7	NA

vi. *Regression tree*

Original model

The original regression tree's summary and illustration is as follow,

```
Regression tree:
tree(formula = SalePrice ~ ., data = house[train, ], method = "class")
Variables actually used in tree construction:
[1] "GrLivArea"   "TotalBsmtSF" "GarageArea"  "OpenPorchSF" "WoodDeckSF"
Number of terminal nodes:  10
Residual mean deviance:  1.122e+09 = 2.694e+11 / 240
Distribution of residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-126200  -18640    2062       0   19060   98140
```

Figure30

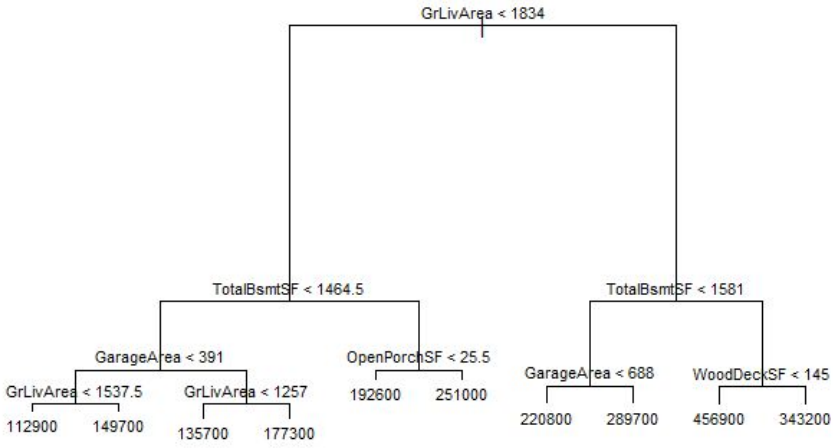


Figure31

And the root-mean-square error (RMSE) is 53338.69.

Cross-validation and tree pruning

The reason for pruning a tree is to prevent overfit problem. And the way to choose the optimal size of tree is to use the training data set to do cross validation, and then figure out which size of tree will produce the least error.

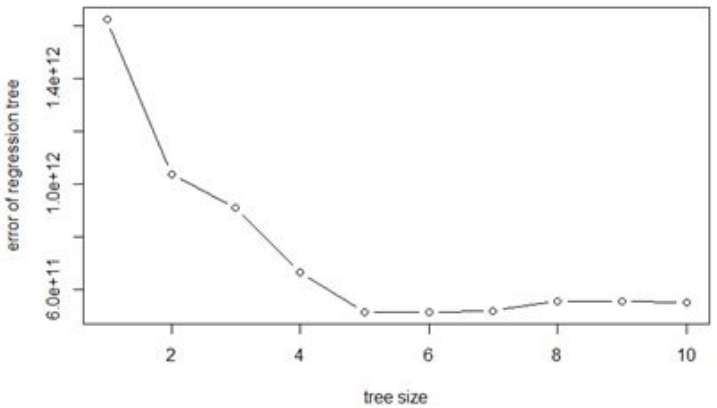


Figure32

A 10 folds cross validation has been done and it is the following graphical presentation. As we can see, a tree size of 5 to 10 produces the least cross validation error. We try to prune the tree until it is a size of 5 regression tree. In this case, the regression tree will be prune until it is a size of 5 regression tree, i.e. a regression tree that only has 5 terminal notes.

However, a size of 5 regression tree seems not to be the best fit of the testing data, the error increased to 55918.44. Thus, with the same approach, we will test all tree size from 5 to 10 as they produce a similar amount of error. The following table represents the RMSE difference of different size of tree. It turns out to be a tree size of 10 still perform the best under this circumstance. As a result, a size of 10 tree is chosen to be our final model.

	tree_result_RMSE <dbl>
RMSE of tree size_5	55918.44
RMSE of tree size_6	57328.92
RMSE of tree size_7	55228.25
RMSE of tree size_8	54135.67
RMSE of tree size_9	53637.72
RMSE of tree size_10	53338.69

Figure33

The below graph shows the performance of different size of tree on predicting the actual price.

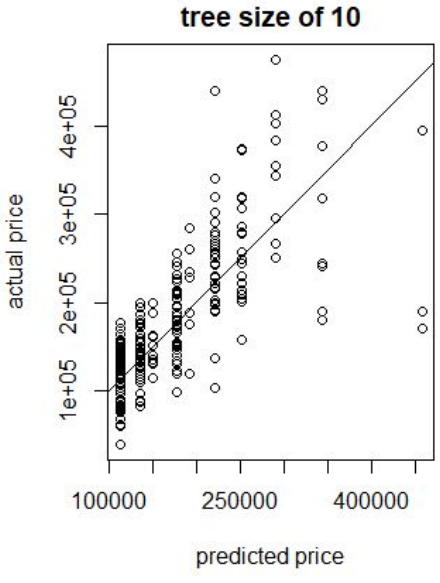


Figure34

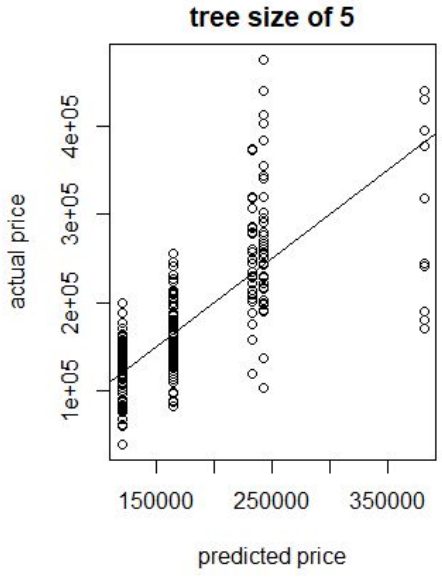


Figure35

Interpretation

We can think of it as a binary question. The first question is, is *GrlivArea* less than 1834? If yes, move to the next branch, is *TotalBsmtSF* less than 1464.5? If yes, move to the next branch. Keep this process until it reaches the terminal node(or sometimes called Leaf). The number that the leaf gives represents the predicted house price. As we can see, *TotalBsmtSF* and *GarageArea* both play an important role in the decision tree, they appeared pretty often and they are usually the parents tree of other sub trees, which means they are the factors that will usually be considered earlier.

Advantage of

using regression tree

Regression tree is easy to visualize and intuitive. Unlike other models, It can be easy to explain to other people or even layman.

vii. *Gradient Boosting*

Introduction

Gradient boosting is an ensemble model of weak learners. The most classic weak learner is decision tree. Gradient boosting ensembles a number of decision trees to make a strong learning of decision tree which can overcome overfitting. Therefore, similar to decision tree, interaction depth, number of trees and shrinkage, etc are the parameters to be discussed.

Simulation

```
> cv_gbm
Stochastic Gradient Boosting

400 samples
 13 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 360, 360, 360, 360, 360, 360, ...
Resampling results across tuning parameters:

  shrinkage  n.trees  RMSE      Rsquared   MAE
  0.001          0    76527.03        NaN  58306.51
  0.001         50    74368.10  0.7113297  56464.46
  0.001        100    72314.03  0.7193518  54705.72
  0.001        150    70375.49  0.7222501  53041.86
  0.001        200    68538.34  0.7254467  51492.49
  0.001        250    66783.82  0.7298841  50007.86
  0.001        300    65103.25  0.7327061  48570.30
  0.001        350    63518.80  0.7371740  47241.30
  0.001        400    62017.53  0.7403071  45970.28
  0.001        450    60604.41  0.7431516  44752.60
  0.001        500    59253.25  0.7458589  43597.29
  0.001        550    57962.39  0.7477290  42486.35
```

Figure36

The training dataset is 80% of the whole data so that there are 400 samples and 13 predictors(features) in the simulation. The grid of gbm is the following: interaction depth is a vector [1 3 5]; number of tree is a sequence from 0 to 2500 separated by 50 each; shrinkage is a vector of [0.01 0.001]; number of minobsinnode is 10.

10-folds cross validation is applied during training. Finally, a gradient boosting model is trained with the formula “SalePrice” against all variables.

Features Importance

An important feature in gbm is the feature importance. The tables above rank the variables based on their relative influence. Relative influence indicates the relative importance of each variable in model training.

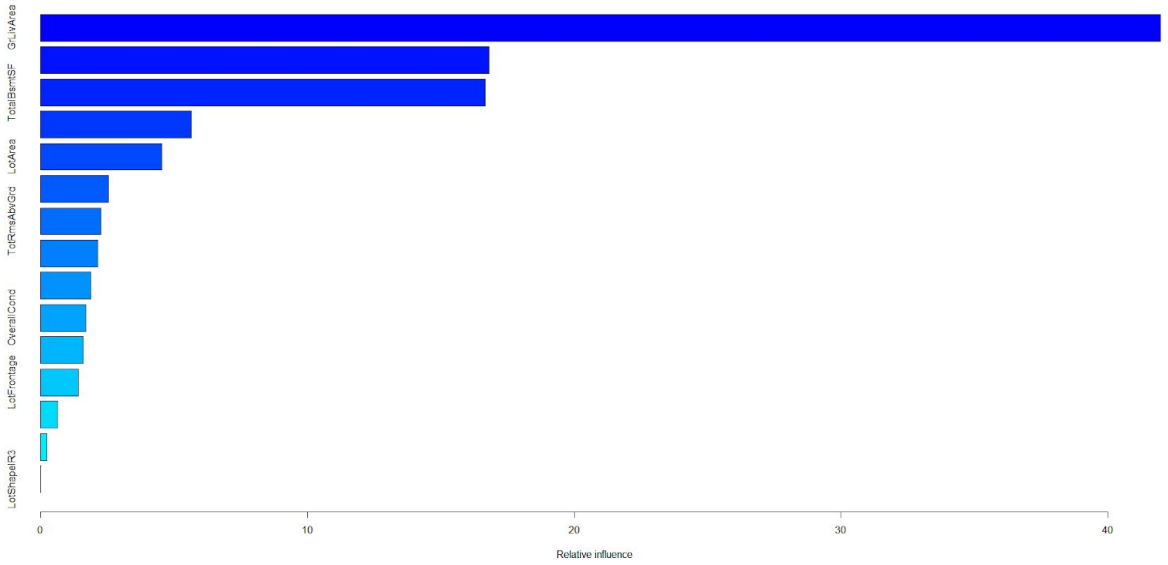


Figure37

var	rel.inf
GrLivArea	41.9818439
GarageArea	16.8188139
TotalBsmtSF	16.6566597
X1stFlrSF	5.6525017
LotArea	4.5336604
MasVnrArea	2.5416157
TotRmsAbvGrd	2.2544892
OpenPorchSF	2.1365754
WoodDeckSF	1.8750724
Overallcond	1.6842118
X2ndFlrSF	1.5909883
LotFrontage	1.4171835
LotShapeReg	0.6324724
LotShapeIR2	0.2239118
LotShapeIR3	0.0000000

Figure38

The figures show that the “GrLivArea” is the most important feature which gives almost 42 of relative influence. Both “GarageArea” and “TotalBsmtSF” are the second most important features which both give almost 17 of relative influence.

“LotShapeReg” and “LotShapeIR2” are the least important features which both give less than 1 of relative influence. “LotShapeIR3” is an unnecessary feature because it gives 0 relative influence in gbm modeling.

Summary of GBM

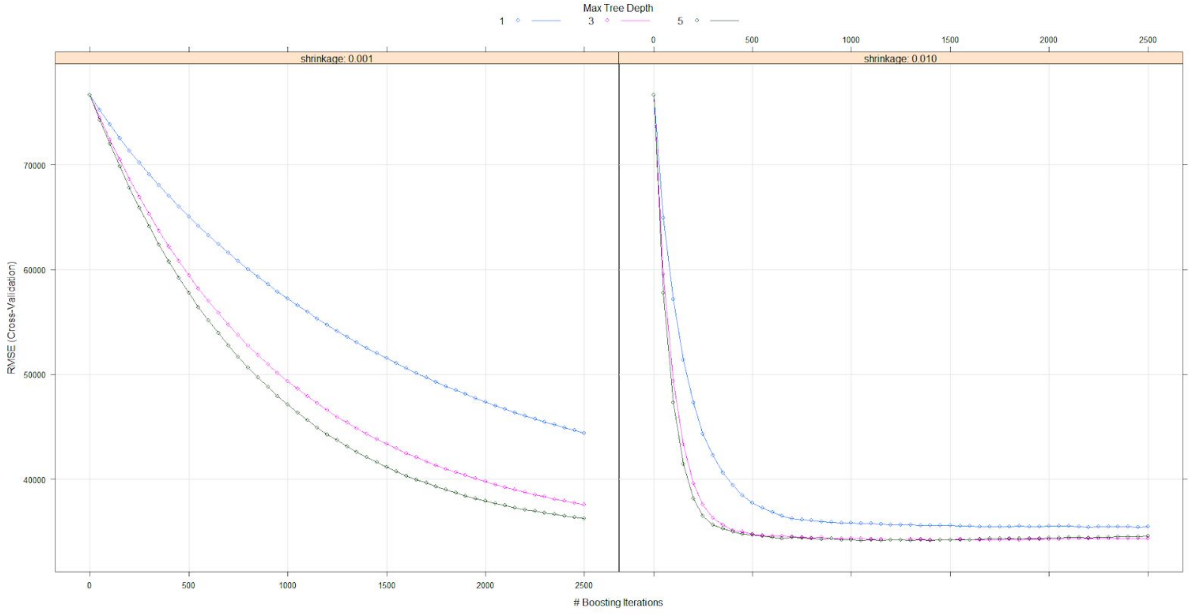


Figure38

The diagram shows the performance of gbm model with different parameters combinations. The figure shows that more iterations(number of trees) give a better model because it is a decreasing trend against RMSE. As shown, the RMSE of models with shrinkage 0.010 sharp drop while the RMSE of models with shrinkage 0.001 flat drop respectively. In general, gbm models with shrinkage 0.010 perform better than that with shrinkage 0.001. As the diagram is shown, 5 maximum tree depth(green line) perform better than 3(pink line) and 1 maximum tree depth(blue line). In conclusion, the best gbm model should be 500-1000 iterations, 0.010 shrinkage and 5 maximum tree depth.

Interpretation and Conclusion

The RMSE of gbm model is finally 35321.91. In some previous attempts, the RMSE can be less than 35000. However, it is hard to control the randomness of splitting when training the model. Nevertheless, gbm model is much better than regression tree as it ensembles 500 to 1000 trees to be a strong learner.

viii. *Principal Component Regression*

Introduction

Principal Component Regression (PCR) is a regression that performed on the component obtained by Principal Component Analysis (PCA). Where PCA is a machine leering technique for reducing the dimension of the variables.

Model

A 10 folds cross validation of PCR has been done and the result is as follows.

```
Data:   X dimension: 400 15
        Y dimension: 400 1
Fit method: svdpc
Number of components considered: 15

VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
CV           81021    42969    43066    41804    41770    40853    40971
adjCV        81021    42703    42822    41659    41591    40763    40980
       7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
CV       40771    40357    40476     40473     39337     38032     36935
adjCV    40810    40293    40377     40385     39178     38023     36810
       14 comps  15 comps
CV        37323     37697
adjCV     37187     37536

TRAINING: % variance explained
                1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
X                 27.97    41.40    52.20    59.96    66.22    72.22    77.89
Saleprice_train   74.50    74.52    74.96    75.27    76.00    76.02    76.24
                8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
X                 83.19    87.81     92.29     95.86     97.70     99.03
Saleprice_train   77.23    77.40     77.43     79.24     80.07     81.40
                14 comps  15 comps
X                  99.97    100.00
Saleprice_train    81.40     81.41
```

Figure39

As we can see, only 10 components can already explain 92.29% of the variance of dependent variables, and 13 components can even explain 99.03% of the variance of dependent variables. The RMSE of the testing data on the original model (a model that has all components) is 37707.86.

This picture shows that when the number of components is about 13, the MSEP is the lowest. Thus, we predict the data with only 13 components, and the RMSE is 37727.87, which is as

good as using all components in the model. However, can it be much better? The second figure represents the RMSE of each component on the test data. We can see that when we only take 11 components, the model performs the best, the RMSE is only 35368.06, which is much better than before.

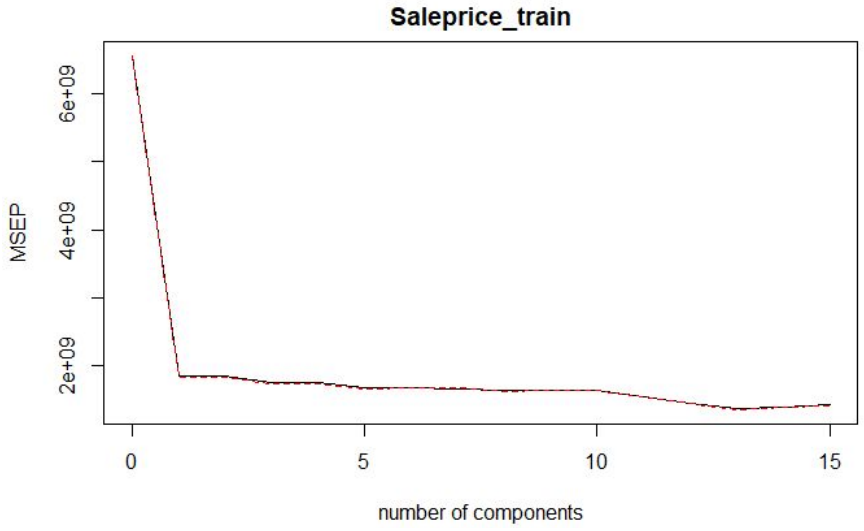


Figure40

	RMSE_table <dbl>
RMSE of 1 comp	41060.87
RMSE of 2 comp	40715.58
RMSE of 3 comp	39265.77
RMSE of 4 comp	39308.34
RMSE of 5 comp	40409.65
RMSE of 6 comp	40454.57
RMSE of 7 comp	41128.57
RMSE of 8 comp	39768.12
RMSE of 9 comp	39684.76
RMSE of 10 comp	39510.17

Figure41

	RMSE_table <dbl>
RMSE of 11 comp	35368.06
RMSE of 12 comp	38671.77
RMSE of 13 comp	37727.87
RMSE of 14 comp	37758.14
RMSE of 15 comp	37707.86

Figure42

interpretation

The reason why taking only 11 components rather than all components will perform better on the test data may due to the reduction of covariance in the dependent variables. Thus, the regression model becomes much efficient, yet the model is hard to be explained or understood. Since sometimes we cannot even explain the loading matrix from PCA, let alone to explain the regressed observed vector of outcomes from PCA. Nonetheless, it is a good way to reduce the noise from the model and the computation need for further machine learning processing.

ix. *Partial Least Squares*

Introduction

Similar to PCR, Partial Least Squares(PLS) has the similar concept to PCR. However, rather than finding the hyperplanes of variance between the dependent and independent variable, PLS project the dependent and independent to a new space in order to cancel out the covariance between the dependent variables.

Model

A 10 folds cross validation of PCR has been done and the result is as follows.

```
Data:   X dimension: 400 15
        Y dimension: 400 1
Fit method: kernelpls
Number of components considered: 15

VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
CV           81021    41129    38225    37557    37470    37223    37269
adjCV        81021    40958    38078    37432    37325    37101    37138
       7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
CV       37272    37285    37332     37500     37627     37657     37699
adjCV    37141    37153    37196     37350     37466     37497     37537
       14 comps  15 comps
CV        37698     37697
adjCV     37537     37536

TRAINING: % variance explained
           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
X            27.91    34.41    42.24    48.16    56.87    64.09     70.1
SalePrice    76.38    80.03    80.86    81.25    81.35    81.39     81.4
           8 comps  9 comps  10 comps  11 comps  12 comps  13 comps  14 comps
X            73.84    78.16      82.8     83.89     88.56     89.82     95.03
SalePrice    81.40    81.40      81.4     81.41     81.41     81.41     81.41
           15 comps
X            100.00
SalePrice     81.41
```

Figure43

As we can see, only 10 components can already explain 82.8% of the variance of the dependent variables, and 13 components can explain 89.82% of the variance of the dependent variables.

The figure below shows when the number of components reaches 2, the MSEP drops dramatically, which also agrees with the RMSE table below. As we can see, when we only take 2 components, the RMSE is 36214.73, which is performing the best along taking other amounts of components to predict the test data. There is also a slight drop when we only use 4 components to predict the test data, the RMSE is also only 36232.74, which is the

second-lowest one. The RMSE of taking all components to predict the test data is 37707.86, we can see that taking 2 or 4 components can already dominate the performance.

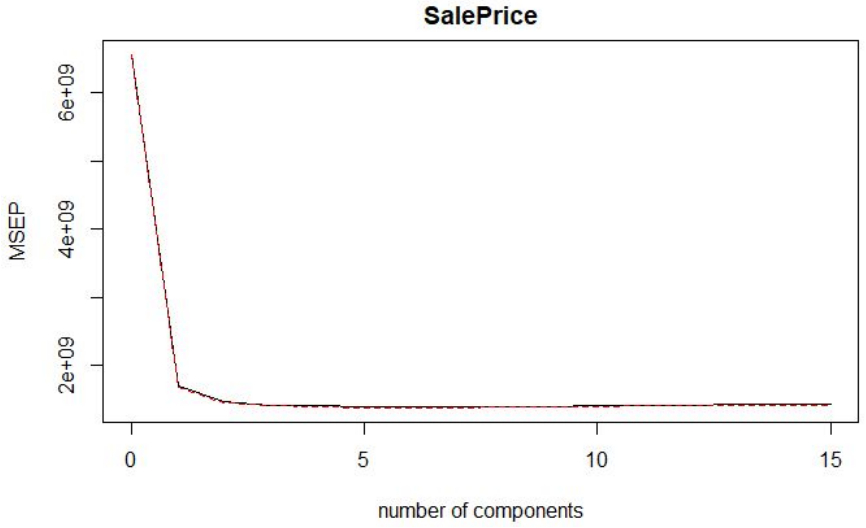


Figure44

Interpretation and conclusion

Again, it is hard to explain the model as we cannot even explain the loading matrix. However, it is not surprising that PLS performs better than PCR, since PLS is known to be work better in terms of small size data. The only problem is using 2 components can only explain 34.41% of variance of the testing data and 4 components can only explain 48.16%, sometimes getting such a result may just be pure luck instead of removing the noise (or removing the covariance between the dependent variables) of the data.

	RMSE_pls <dbl>
RMSE of 1 comp	39426.15
RMSE of 2 comp	36214.73
RMSE of 3 comp	37056.94
RMSE of 4 comp	36232.67
RMSE of 5 comp	37369.65
RMSE of 6 comp	38003.36
RMSE of 7 comp	38024.38
RMSE of 8 comp	37802.67
RMSE of 9 comp	37740.96
RMSE of 10 comp	37741.92

Figure45

	RMSE_pls <dbl>
RMSE of 11 comp	37713.50
RMSE of 12 comp	37746.15
RMSE of 13 comp	37712.56
RMSE of 14 comp	37707.45
RMSE of 15 comp	37707.86

Figure46

Conclusion

Model	RMSE
Linear Regression	37088.35
Ridge Regression	36316
Lasso Regression	35603
Regression Spline	27492
Regression Tree	53338.69
Gradient Boosting	35321.91
Principal Component Regression	35368.06
Partial Least Squares	36214.73

As the table is shown, regression spline has the lowest RMSE in prediction because the knots are split into small pieces where fit the data very well. However, it is very time consuming since we have to define the knots manually. On the other hand, regression tree has the highest RMSE but it is easy to interpret and understand.

3. Titanic

a. Data Description

i. Data Structure

```
> str(Titanic)
'data.frame':   500 obs. of  9 variables:
 $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
 $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
 $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
 $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
 $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
 $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
 $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
 $ Embarked   : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...
```

Figure46

From the structure of "Titanic.xlsx, there are 500 observations and 9 variables. The "Survived" is the dependent variable and the others are independent variables. " PassengerId" is omitted in the model fitting and prediction process.

ii. Variables Correlation

Similar to the "House" data, it is better to generate a correlation matrix to understand the relationship between the variables. "Sex" and "Embarked", which are factor variables, are transformed into an integer variable so as to generate a correlation matrix.

NAs are also omitted in the correlation matrix computation so as to avoid bias. The correlation matrix is shown below.

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
PassengerId	1.0000000000	0.07736386	-0.09640867	0.007219665	0.07517068	-0.10136540	-0.01836310	0.05128369	-0.0006638059
Survived	0.0773638602	1.00000000	-0.26925119	-0.560783075	-0.10967219	-0.02781408	0.10105374	0.20678764	-0.1520622644
Pclass	-0.0964086748	-0.26925119	1.00000000	0.126337925	-0.36619610	0.09719812	0.01337805	-0.58668333	0.2917918384
Sex	0.0072196650	-0.56078307	0.12633792	1.000000000	0.10567939	-0.07712978	-0.17046568	-0.18500424	0.1252571600
Age	0.0751706765	-0.10967219	-0.36619610	0.105679392	1.00000000	-0.35745600	-0.23982165	0.08832065	-0.0713452847
SibSp	-0.1013654039	-0.02781408	0.09719812	-0.077129781	-0.35745600	1.00000000	0.41312014	0.16636411	0.0521966716
Parch	-0.0183630954	0.10105374	0.01337805	-0.170465680	-0.23982165	0.41312014	1.00000000	0.25824915	0.0880073884
Fare	0.0512836865	0.20678764	-0.58668333	-0.185004235	0.08832065	0.16636411	0.25824915	1.00000000	-0.2869565340
Embarked	-0.0006638059	-0.15206226	0.29179184	0.125257160	-0.07134528	0.05219667	0.08800739	-0.28695653	1.0000000000

Figure47

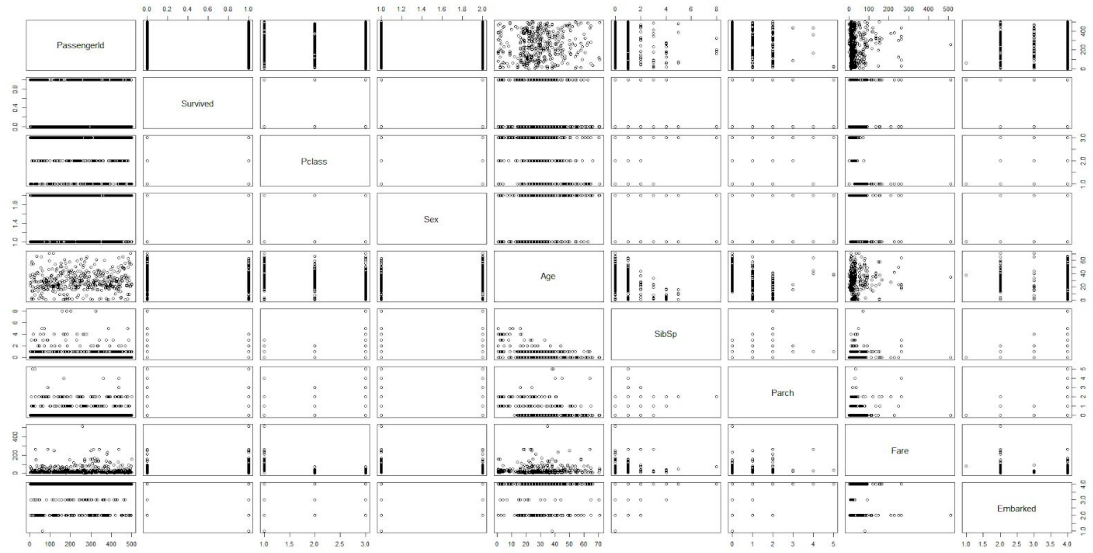


Figure48

From the correlation matrix, "Survived" and "sex" is significantly correlated. We hypothesized that "sex" will contribute more in model fitting and prediction. "Pclass" and "Fare" are significantly correlated which may cause problems in model fitting. On the other hand, the general plot doesn't show a significant correlation between variables.

iii. Handling Missing Data

```
> summary(Titanic)
  PassengerId       Survived         Pclass          Sex           Age            SibSp           Parch            Fare          Embarked
 Min.   :  1.0   Min.   :0.000   Min.   :1.000   female:185   Min.   : 0.75   Min.   :0.000   Min.   :0.00   Min.   :  0.000    : 1
 1st Qu.:125.8   1st Qu.:0.000   1st Qu.:2.000   male  :315   1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.00   1st Qu.:  7.925   C: 92
 Median :250.5   Median :0.000   Median :3.000                Median :28.00   Median :0.000   Median :0.00   Median : 14.456   Q: 45
 Mean   :250.5   Mean   :0.386   Mean   :2.326                Mean   :29.20   Mean   :0.574   Mean   :0.38   Mean   : 31.782   S:362
 3rd Qu.:375.2   3rd Qu.:1.000   3rd Qu.:3.000                3rd Qu.:37.75   3rd Qu.:1.000   3rd Qu.:0.00   3rd Qu.: 30.071
 Max.   :500.0   Max.   :1.000   Max.   :3.000                Max.   :71.00   Max.   :8.000   Max.   :5.00   Max.   :512.329
                                                              NA's   :102
>
```

Figure49

From the summary of the dataset "Titanic.xlsx", there are 102 NAs in variable "Age". A gbm model is fitted for "Age" against other variables. 10-folds cross validation is also applied in the gbm model. Therefore, the missing "Age" can be predicted by the model.

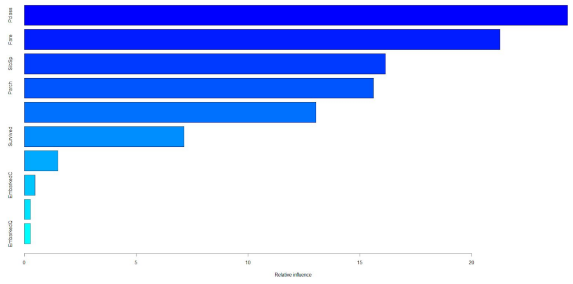


Figure50

Remarks: The gbm model provides feature importance. As the figure is shown, "Pclass" is the most important feature and "EmbarkedQ" is the least important feature for "Age" modeling.

b. Modeling

i. *KNN*

K-Nearest Neighbor algorithm (KNN) is a non-parametric method used for classification. The decision rule of KNN is as follow,

$N_K(x, x_i)y_i = 1$ if x_i is a neighbor of x

$N_K(x, x_i)y_i = 0$ otherwise

The classifier: $C_\lambda(x) = \frac{\sum_{i=1}^{n} K(x,x_i)y_i}{\sum_{i=1}^{n} K(x,x_i)}$

$C_K(x) > 0.5$, predict $y = 1$

$C_K(x) < 0.5$, predict $y = 0$

There are 500 samples with 6 predictors classifies in 2 classes.

```
   k   Accuracy   Kappa
   1   0.752      0.47069169
   2   0.740      0.43962619
   3   0.732      0.42124631
   4   0.698      0.33538732
   5   0.726      0.40418203
   6   0.728      0.41028532
   7   0.738      0.43027625
   8   0.730      0.41288011
   9   0.738      0.43140387
  10   0.736      0.41490399
  11   0.742      0.42993000
  12   0.724      0.39077151
  13   0.730      0.39856725
  14   0.716      0.36415253
  15   0.712      0.35781944
  16   0.702      0.33619645
  17   0.708      0.34221197
  18   0.692      0.30474041
  19   0.690      0.29660555
  20   0.692      0.30330613
  21   0.686      0.29337846
  22   0.678      0.27238873
  23   0.672      0.25959368
  24   0.672      0.25959368
  25   0.650      0.21718437
  26   0.660      0.24340876
  27   0.656      0.23450768
  28   0.626      0.16690427
  29   0.598      0.09348391
  30   0.618      0.13323652
  31   0.626      0.14962893
  32   0.638      0.17348579
  33   0.654      0.20002220
  34   0.636      0.15751662
  35   0.646      0.18324781
  36   0.656      0.20881709
  37   0.654      0.19492945
  38   0.622      0.12970603
  39   0.638      0.15590957
  40   0.650      0.18389046
  41   0.646      0.17456350
  42   0.646      0.17631510
  43   0.636      0.15573451
  44   0.644      0.17428980
  45   0.644      0.17078170
  46   0.634      0.14658260
  47   0.642      0.16345758
  48   0.646      0.17103784
  49   0.648      0.17307222
  50   0.650      0.17688892

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 1.
            Length Class      Mode
learn       2      -none-     list
k           1      -none-     numeric
theDots     0      -none-     list
xNames      18     -none-     character
problemType 1      -none-     character
tuneValue   1      data.frame list
obsLevels   2      -none-     character
param       0      -none-     list
```

Figure51

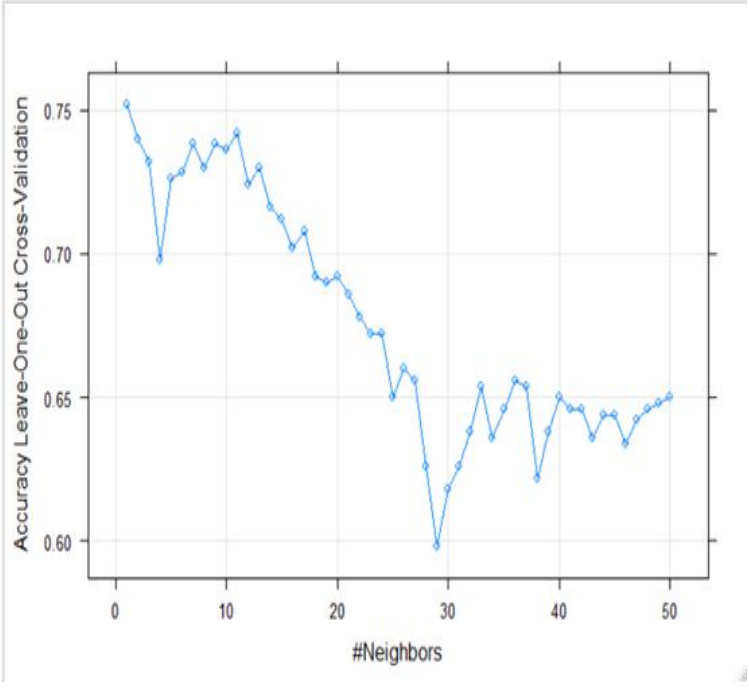


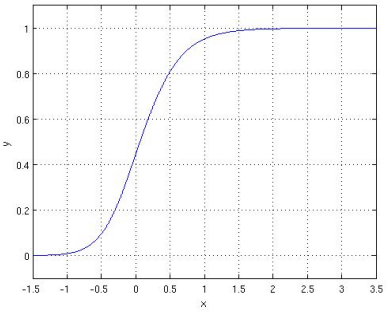
Figure52

From the result and graph, only “k” is a numeric variable. Accuracy was used to select the optimal model using the largest value. The final value used for the model was k = 10.

ii. *Logistic Regression*

introduction

Logistic regression is used to model the probability of a certain class or event, i.e. to predict the output with different categories. The reason for using it is because of the rate of convergence is high. Thus, the model (or the line) can fit the data better.



Model

The original model is as follow,

```
Call:
glm(formula = Survived ~ ., family = binomial, data = tin_train[s,
    ])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0783  -0.5989  -0.3407   0.5292   2.6027

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.544559   0.858007   5.297 1.18e-07 ***
Pclass2     -1.560551   0.537871  -2.901  0.00372 **
Pclass3     -2.795107   0.567045  -4.929 8.25e-07 ***
Sexmale     -3.037595   0.328714  -9.241  < 2e-16 ***
Age         -0.053449   0.013691  -3.904 9.46e-05 ***
SibSp       -0.546952   0.199712  -2.739  0.00617 **
Parch        0.168287   0.243458   0.691  0.48942
Fare        -0.001009   0.005304  -0.190  0.84914
EmbarkedQ    1.580764   0.629724   2.510  0.01206 *
EmbarkedS    0.348967   0.412109   0.847  0.39712
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 481.14  on 359  degrees of freedom
Residual deviance: 296.07  on 350  degrees of freedom
AIC: 316.07

Number of Fisher Scoring iterations: 5
```

Figure53

And the prediction result is as follow,

original model	0	1
0	53	15
1	10	22

Accuracy = 0.75

The RMSE of each fold of cross validation is as follow,

	RMSE <dbl>
1 set of cross validation	0.3658122
2 set of cross validation	0.3538622
3 set of cross validation	0.3689574
4 set of cross validation	0.3486499
5 set of cross validation	0.3589968
6 set of cross validation	0.3604252
7 set of cross validation	0.3582554
8 set of cross validation	0.3512021
9 set of cross validation	0.3643373
10 set of cross validation	0.3553576

Figure54

Decision Rule Cut-Off

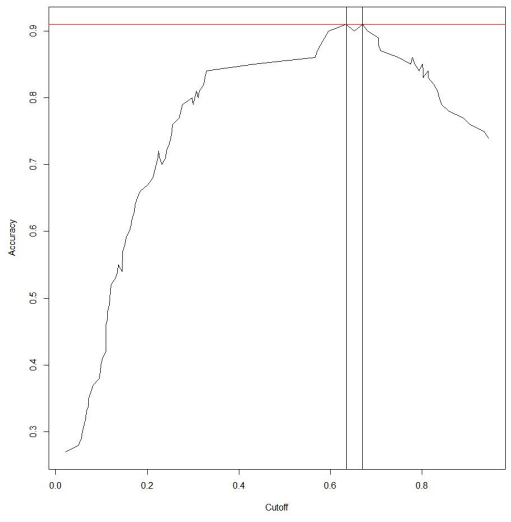


Figure55

However, when we change the cut-off of the model in a correct manner, then the model will become more accurate. In the other words, if we know where the dataset bias to, then we can receive a better model. The cut-off of the original model is 0.5. If the prediction probability is higher than 0.5, the model will classify it as 1 in survived, vice versa. However, the decision rule may not be the best. By using library “ROCR”, the prediction accuracy can be simulated through different cut-off. As the figure is shown, 0.5 cut-off is not the optimum accuracy of the model. In conclusion, the best cut-off is 0.67 which gives 0.91 accuracies.

Using the new cut-off, the prediction on test data get a 0.9 accuracy.

ROC

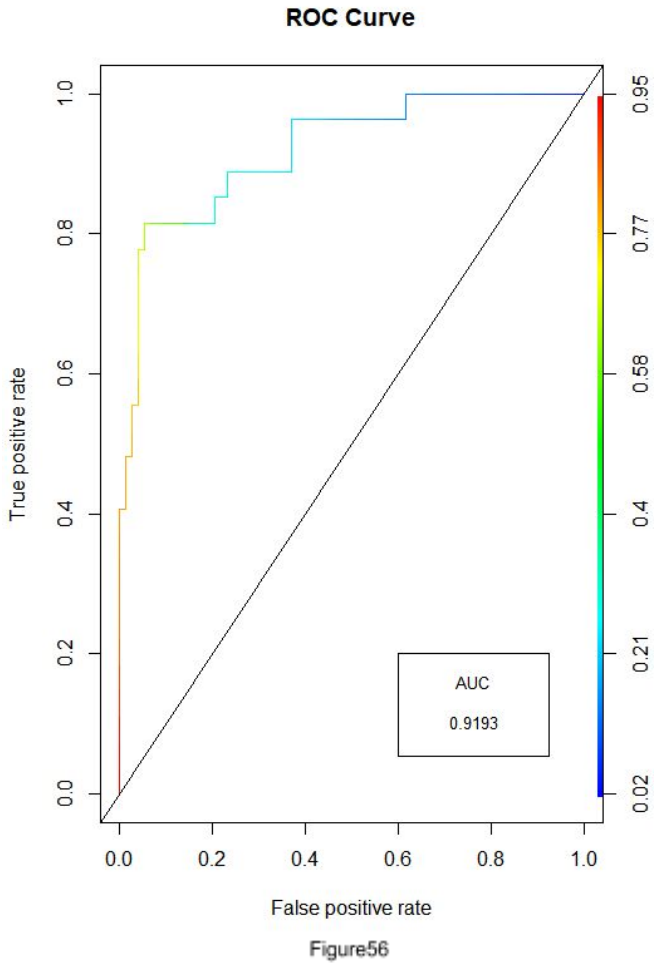


Figure56

The ROC curve shows the trade-off between true positive rate(TPR) and false positive rate(FPR). The ROC is far from the 45-degree diagonal so that the accuracy of the model performs well. On the other hand, AUC is 0.9193. AUC is equivalent to the probability that a randomly chosen TPR is ranked higher than that of FPR. Therefore, the AUC measure mentions that the TPR rate is higher than FPR rate in our model.

Model Selection

Rather than changing the cut-off, we can also drop some useless parameters out. As we can see, some variables are not significant, which means there is room for us to drop some variables that is less useless to prevent overfitting and let the model become much more general. The AIC and the optimized model is as follow,

```
Step:  AIC=328.13
Survived ~ Pclass + Sex + Age + SibSp

         Df Deviance    AIC
<none>        316.13 328.13
- SibSp   1   323.96 333.96
- Age     1   331.14 341.14
- Pclass  2   344.06 352.06
- Sex     1   438.35 448.35
```

Figure57

```
Call:
glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family = binomial,
    data = tin4log_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3608  -0.5668  -0.3918   0.6307   2.4724

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.40918    0.65943   6.686 2.29e-11 ***
Pclass2     -1.27154    0.41837  -3.039 0.002371 **
Pclass3     -2.21180    0.39592  -5.586 2.32e-08 ***
Sexmale     -3.04678    0.29783 -10.230  < 2e-16 ***
Age         -0.04797    0.01261  -3.804 0.000143 ***
SibSp       -0.48668    0.17429  -2.792 0.005231 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 535.00  on 399  degrees of freedom
Residual deviance: 341.28  on 394  degrees of freedom
AIC: 353.28

Number of Fisher Scoring iterations: 5
```

Figure58

And the 10 folds RMSE ,accuracy and ROC are as follow,

	RMSE <dbl>
1 set of cross validation	0.3574021
2 set of cross validation	0.3571269
3 set of cross validation	0.3611478
4 set of cross validation	0.3621071
5 set of cross validation	0.3600803
6 set of cross validation	0.3602852
7 set of cross validation	0.3564025
8 set of cross validation	0.3587751
9 set of cross validation	0.3614368
10 set of cross validation	0.3630320

Figure59

improved model	0	1
0	55	16
1	8	21

Accuracy = 0.76

ROC

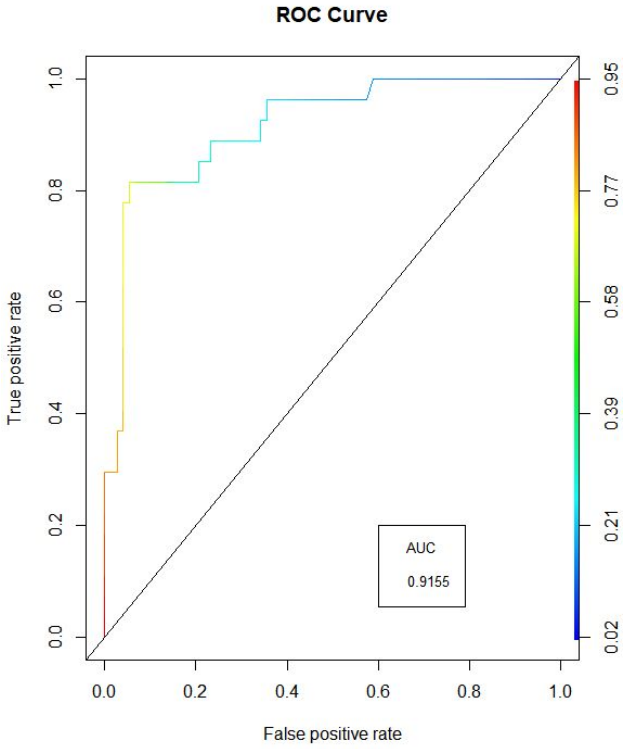


Figure60

As we can see, after the model selection process, the model becomes more accurate and simpler. However, when we take a look at ROC of the second model (figure 60), we can see that the AUC is slightly lesser, which may indicate we may drop too many variables, more investigation may be needed.

Interpretation

The coefficient in the summary of the logistic model indicates the contribution of the number to the model, and since they have a high convergence rate, it is more likely to converge to the favorable result in classification and the result is surprisingly good, about 90% of the result is correctly estimated.

iii. *Classification tree*

introduction

Classification tree is a type of decision tree. The only difference between classification tree and regression tree is, classification tree is used to predict categorical variable, whereas decision tree is used to predict continuous variables.

Model

The summary and the plot of the classification tree is as follow,

```
Classification tree:
tree(formula = Survived ~ ., data = tin_train)
Variables actually used in tree construction:
[1] "Sex"      "Pclass"   "Embarked" "Age"      "Fare"
Number of terminal nodes:  9
Residual mean deviance:  0.7797 = 304.8 / 391
Misclassification error rate: 0.1775 = 71 / 400
```

Figure61

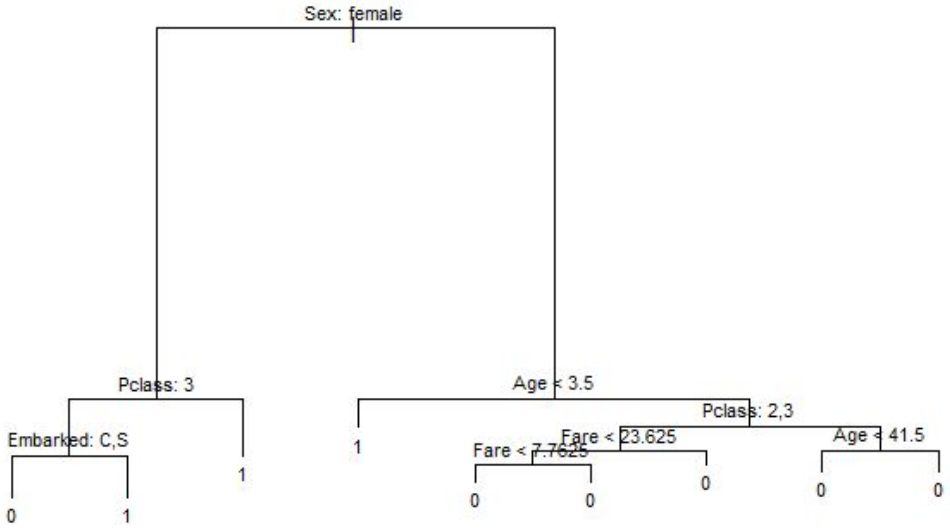


Figure62

And the prediction result is as follow,

Original tree	0	1
0	57	16
1	6	21

Accuracy = 0.78

A classification rate of 0.78 is actually a very outstanding result in terms of only 400 rows of training data and before any tuning of parameters. It is time to see the effect on tree pruning, i.e. get rid of some useless sub trees.

Pruning and model selection

The following graph shows the misclassification error on 10 cv points. As we can see, the misclassification error reaches the lowest on both sizes of 2, 3, 5 trees. Since it is hard to determine which is the best tree, and the following table shows the accuracy of each size of tree.

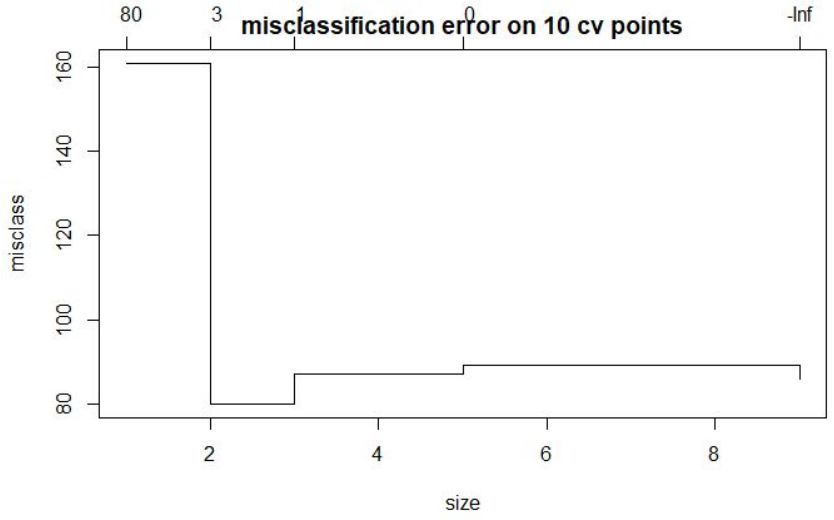


Figure63

tree size	accuracy
2	0.76
3	0.77
5	0.81

Thus, a model with tree size of 5 will be chosen.

Special fact

When we looking at the plot of different sizes of tree, we figure out that *sex* may be the major reason for the survival result.

The graphs below are the tree model of different size.

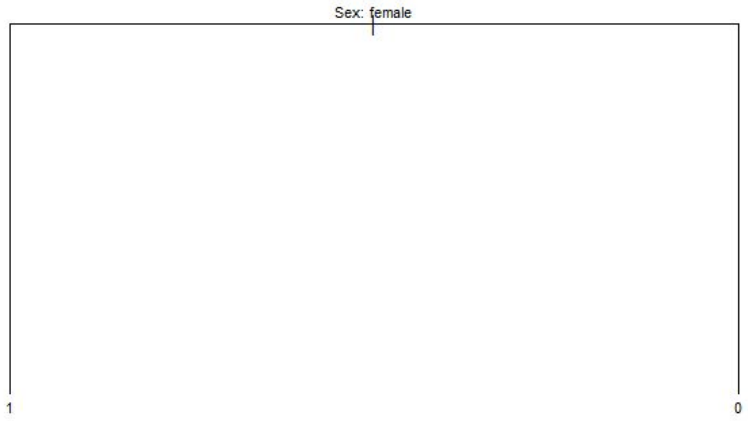


Figure64

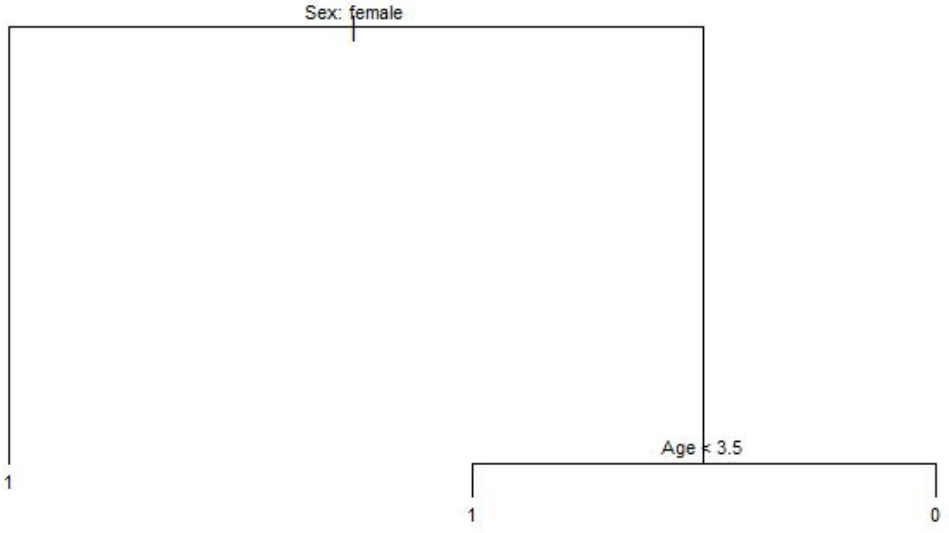


Figure65

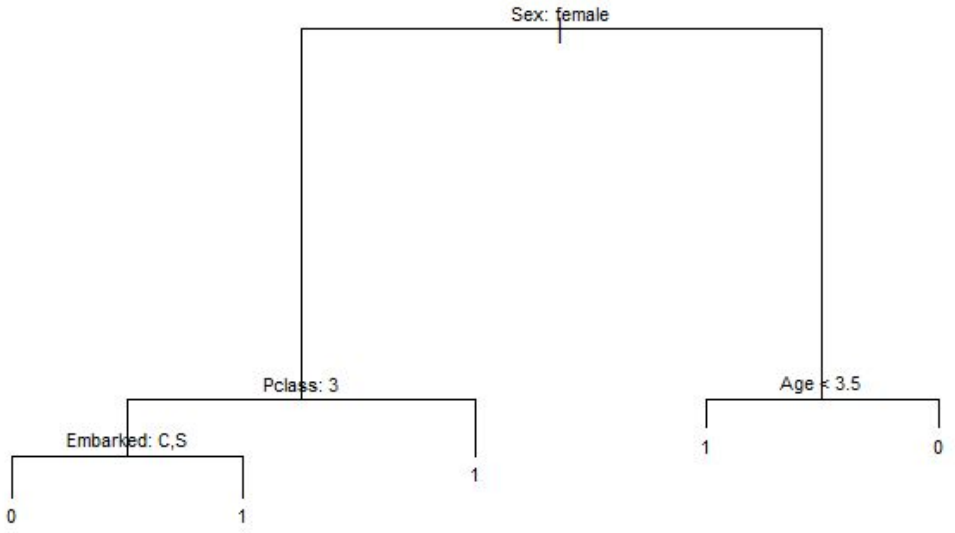


Figure66

We can see that the major factor that the tree consider will always be the *sex*, and more precisely, *Female*. Even a binary tree only classify whether the survivor is Male of female already receive a 0.76 accuracy. We can see how important female is in this dataset.

Conclusion

Classification tree is obviously a strong tool. It performs pretty well even dealing with a small amount of data and receive a 0.81 accuracy, which is pretty impressive. Although it may due to that fact that female is the major factor that affects the survivor, we still cannot deny its power. Furthermore, classification tree is easy to understand and easy to interpret. Tree pruning can even reduce the complexity of the model, making it being more understandable whilst reducing the problem of overfitting.

iv. *Random forest*

Introduction

Random Forest can be considered as a set of decision tree, in other words, it is an ensemble learning method. When we are talking about decision tree, we know that tree pruning can actually increase the accuracy. One reason for this effect occurs is because the overfitting problem has been reduced, as the tree may be too fit to the training data set, i.e. there may be some pattern only occurs in the training dataset. Thus, in order to deal with this problem, rather than pruning tree, training multiples tree and gather all advantage of them may also be a good idea, and that is what we call random forest.

Model

A 10-folds cross validation has been done and the result is as follow,

```
Random Forest

400 samples
  7 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 361, 360, 360, 360, 359, 361, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   1    0.7833927  0.5000134
   2    0.8125453  0.5845067
   3    0.8275281  0.6242803
   4    0.8283615  0.6279940
   5    0.8166114  0.6062560
   7    0.8092120  0.5936528
   9    0.8076053  0.5911707
  10    0.8026673  0.5816914

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 4.
```

Figure67

As we can see, when *mtry* = 3, 4 the model performs the best. This means when 3, 4 and 5 variables randomly sampled as candidates at each split, the models under this condition perform the best. Although the algorithm advise us to select the model with mtry = 4, but since when mtry = 3, 4, the performances are so similar. Thus both parameters will be tested.

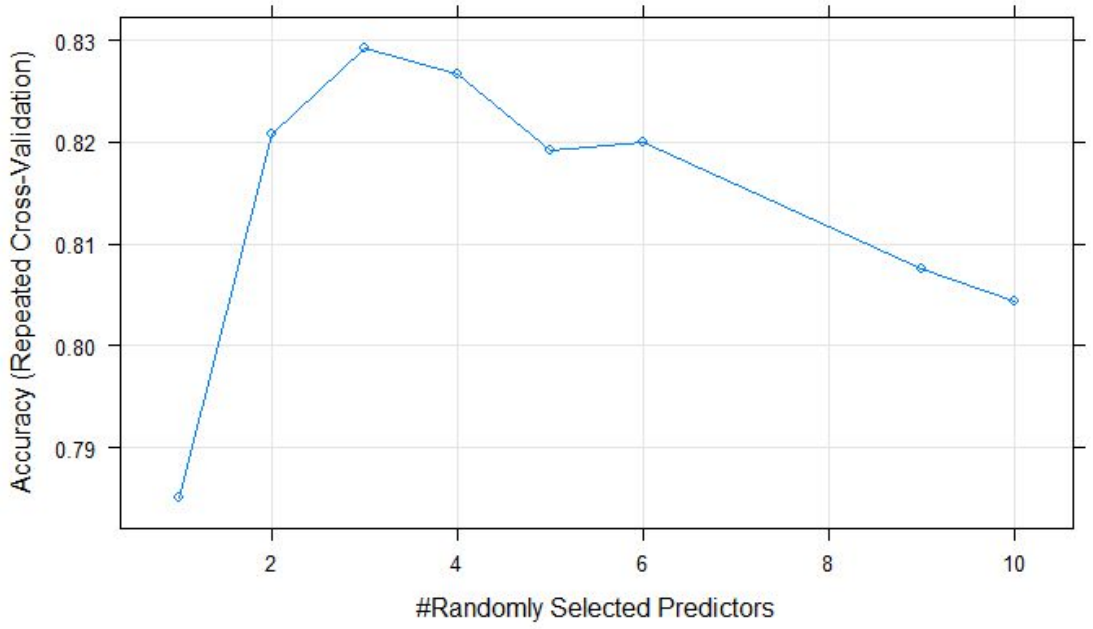


Figure68

Result

Surprisingly, when mrty = 3 and mtry = 4, they both obtain the same result. The confusion matrix is as follow, the accuracy of testing data also agrees with the cross validation result of the training data.

mtry = 3, 4	0	1
0	57	12
1	6	25

Accuracy = 0.82

Interpretation

Different from decision tree, since random forest is an assembly of decision tree, so it is hard to interpret the result obtained from random forest. However, if we can think of random forest is a set of tree, then the concept will be easy for layman to understand.

Comparison with a single classification tree

Not surprisingly, random forest does have better performance compare to classification tree. Although the difference may not be so obvious, it may due to the small size of the dataset. We expect the difference will be larger when the size of the dataset increase.

v. Linear Discriminant Analysis

Introduction

Linear discriminant analysis (LDA) is a supervised dimension reduction statistical method. It projects labeled data into a lower dimension plane, and figure out its characteristic or linear combination, in order to form a linear classifier. It assumes independent variables follows normal distribution, thus when the independent variables are not following normal distribution, the performance may not be too favorable.

Model

The LDA model after 10-folds cross validation is as follow,

```
Call:
lda(x, grouping = y)

Prior probabilities of groups:
   0    1
0.61 0.39

Group means:
    Pclass2   Pclass3   Sexmale      Age     SibSp     Parch     Fare
0 0.1844262 0.6516393 0.8524590 30.68852 0.5737705 0.3196721 22.86294
1 0.2500000 0.4102564 0.2564103 27.93750 0.4743590 0.4038462 38.85823
  EmbarkedC  EmbarkedQ EmbarkedS
0 0.1557377 0.05737705 0.7868852
1 0.2307692 0.14743590 0.6153846

Coefficients of linear discriminants:
                 LD1
Pclass2   -0.805448729
Pclass3   -1.490543754
Sexmale   -2.247040620
Age       -0.028066952
SibSp     -0.225666906
Parch      0.084089834
Fare      -0.001140404
EmbarkedC -0.395744912
EmbarkedQ  0.354568125
EmbarkedS -0.393287193
```

Figure69

Accuracy = 0.78

Fortunately, we do not have to tune the parameter by our self, as the function will help us to tune the parameter, by fitting different folds of data into the normal distribution, then it will automatically receive the optimal mean of each component.

Cross-Validation

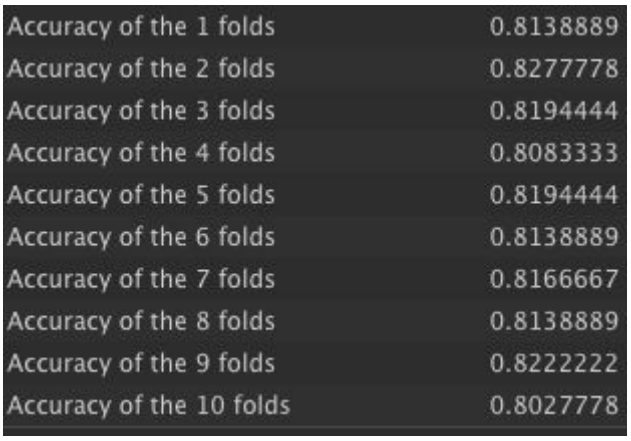


Figure70

10-folds cross-validation is applied to the LDA model. Since LDA is not able to compute RMSE, accuracy of each fold validation is applied. As the figure shown, the accuracy of each fold validation is all over 0.8. The figure shows that LDA model performs stably around 0.8 to 0.82.

Interpretation

The group means in the model means the mean of the respective components such as Age is 30 if getting 0, and 27.9 if getting 1. It follows the 'voting rules', if the data is close to 0, for example, Age is close to 30.688, Fare is close to 22.86, and so on, then the answer is 0; else if the data close to 1, then the answer is 1.

Problem

LDA is a very powerful algorithm and it is not normal that it only has accuracy for 0.78. Thus, an investigation has been done and the result is as follows. As we can see, at least 3 variables are not following normal distribution, as a result, it is letting the accuracy not as favorable as we think.

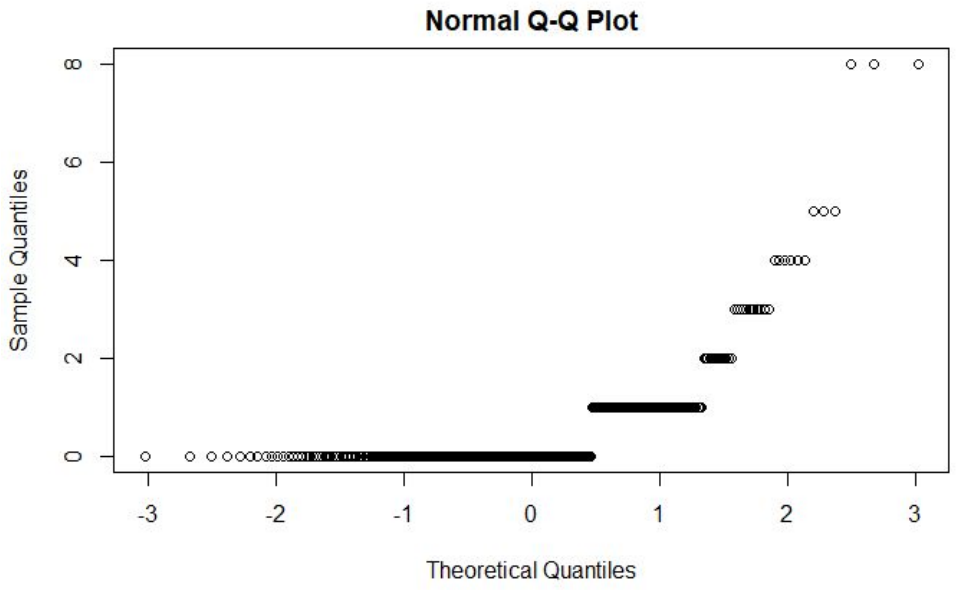


Figure71

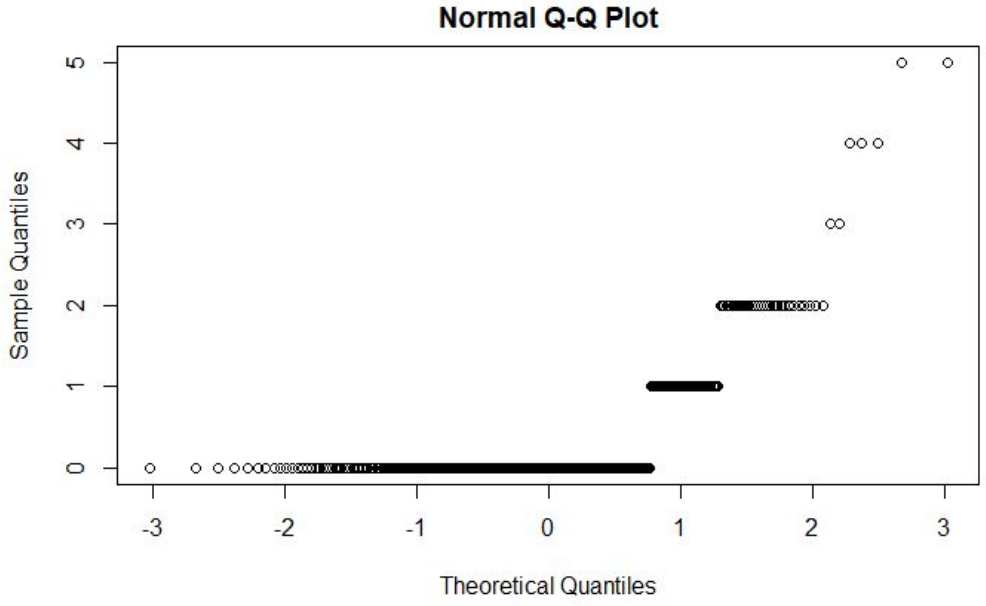


Figure72

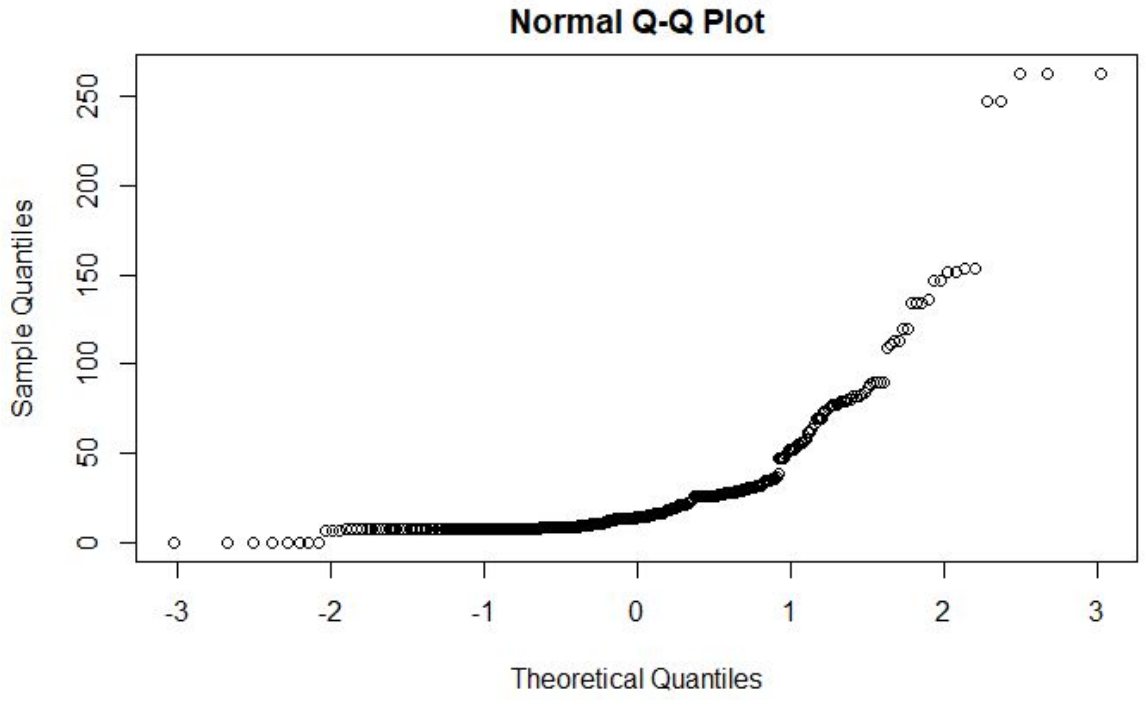


Figure73

Conclusion

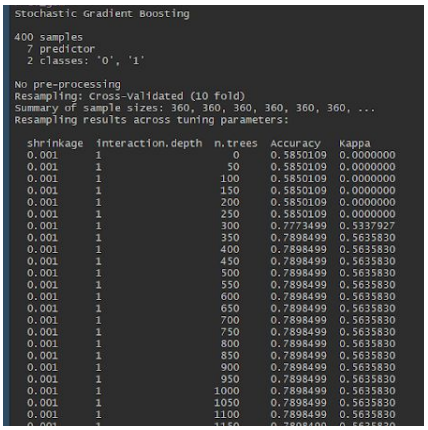
Although LDA is very powerful in other datasets. However, as the amount of data is not enough to follow the law of large numbers, so some variables are not following normal distribution, and it leads to such a non-favourable result. Nevertheless, it still receives an accuracy of 0.78, which is still better than some naive statistical method such as linear regression and shows how powerful it can be in other datasets.

vi. *Gradient Boosting*

Introduction

Gradient boosting is strong learning through ensembling a number of decision trees. Therefore, similar to decision tree, interaction depth, number of trees and shrinkage, etc are the parameters to be discussed. In classification case, the metric will be accuracy.

Simulation



```
Stochastic Gradient Boosting

400 samples
  7 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 360, 360, 360, 360, 360, 360, ...
Resampling results across tuning parameters:

  shrinkage  interaction.depth  n.trees  Accuracy   Kappa
  0.001      1                    0      0.5850109  0.0000000
  0.001      1                   50      0.5850109  0.0000000
  0.001      1                  100      0.5850109  0.0000000
  0.001      1                  150      0.5850109  0.0000000
  0.001      1                  200      0.5850109  0.0000000
  0.001      1                  250      0.5850109  0.0000000
  0.001      1                  300      0.7773499  0.5337927
  0.001      1                  350      0.7898499  0.5635830
  0.001      1                  400      0.7898499  0.5635830
  0.001      1                  450      0.7898499  0.5635830
  0.001      1                  500      0.7898499  0.5635830
  0.001      1                  550      0.7898499  0.5635830
  0.001      1                  600      0.7898499  0.5635830
  0.001      1                  650      0.7898499  0.5635830
  0.001      1                  700      0.7898499  0.5635830
  0.001      1                  750      0.7898499  0.5635830
  0.001      1                  800      0.7898499  0.5635830
  0.001      1                  850      0.7898499  0.5635830
  0.001      1                  900      0.7898499  0.5635830
  0.001      1                  950      0.7898499  0.5635830
  0.001      1                 1000      0.7898499  0.5635830
  0.001      1                 1050      0.7898499  0.5635830
  0.001      1                 1100      0.7898499  0.5635830
```

Figure74

The training dataset is 80% of the whole data so that there are 400 samples and 7 predictors(features) in the simulation where “PassengerId” is omitted. The grid of gbm is the following: interaction depth is a vector [1 3 5]; number of tree is a sequence from 0 to 2500 separated by 50 each; shrinkage is a vector of [0.01 0.001]; number of minobsinnode is 10.

10-folds cross validation is applied during training. Finally, a gradient boosting model is trained with the formula “Survived” against all variables.

Feature Importance

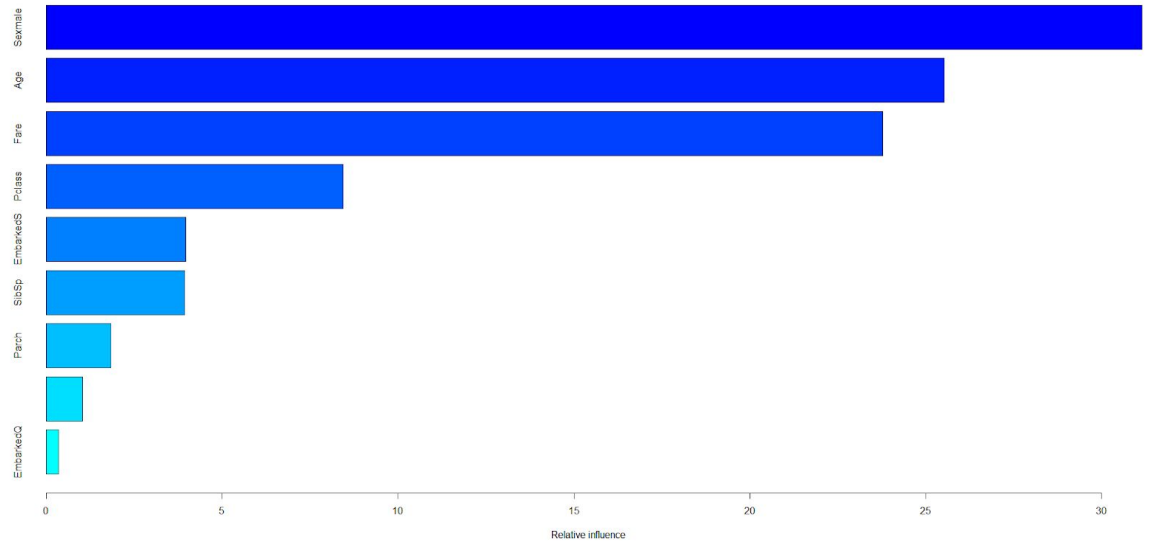


Figure75

var	rel.inf
Sexmale	31.1531575
Age	25.5245054
Fare	23.7782815
Pclass	8.4299316
EmbarkedS	3.9605719
SibSp	3.9340772
Parch	1.8402215
EmbarkedC	1.0253790
EmbarkedQ	0.3538745

Figure76

The figures show that the “sex” is the most important feature which gives around 31 of relative influence. “Age” and “Fare” are the second and third most important features which give almost 25 and 23 of relative influence.

Dummy variable “Embarked” are the least important features which give the lowest relative influence. However, “Embarked” is still necessary for modeling because the relative influence is non-zero.

Summary of GBM

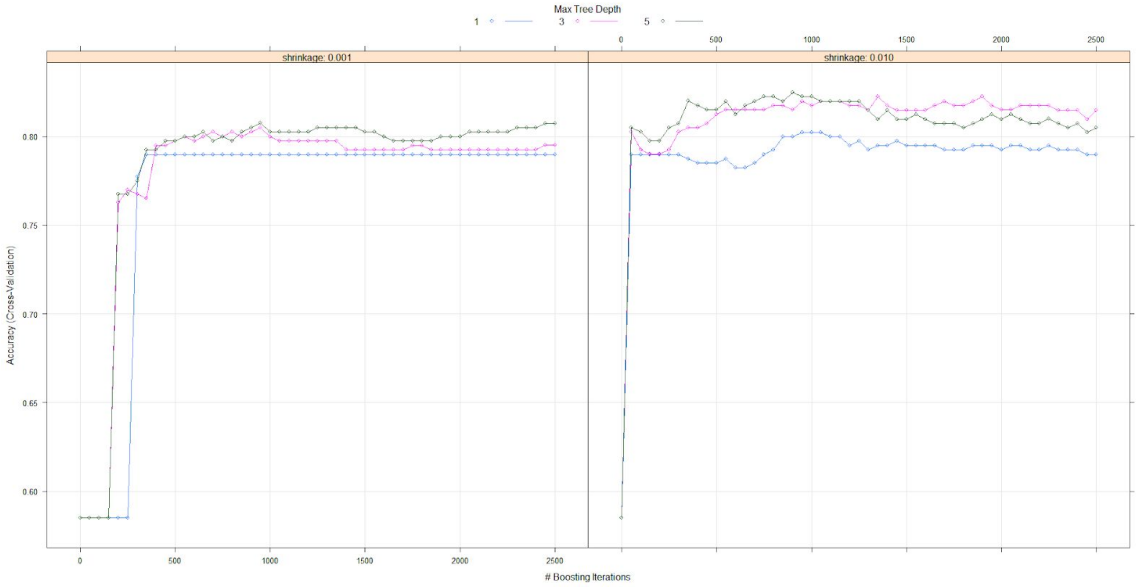


Figure77

The diagram shows the performance of gbm model with different parameter combinations. The figure shows that 500 to 1500 iterations(number of trees) gives better model because it is a sharp increasing trend and becomes flat. As shown, the accuracy of models with shrinkage 0.010 is generally higher than models with shrinkage 0.001. As the diagram is shown, 5(green line) or 3(pink line) maximum tree depth perform better than 1 maximum tree depth(blue line). In conclusion, the best gbm model should be 500-1500 iterations, 0.010 shrinkage, and 5 or 3 maximum tree depth.

Confusion Matrix & Interpretation

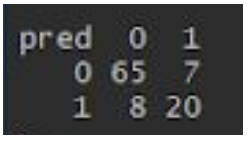


Figure78

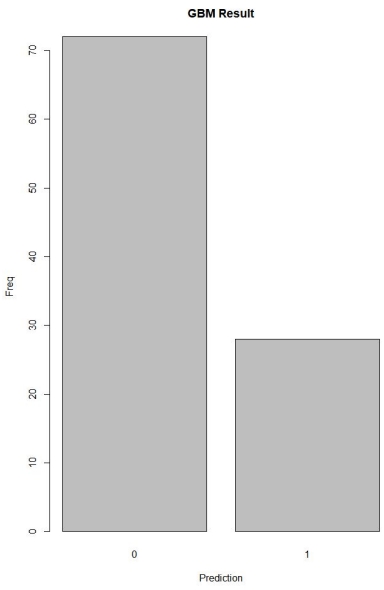


Figure79

The result of GBM is good. From the histogram, the proportion of "0" is much greater than "1". The model predicted 73 "0" and 27 "1". The prediction gets 85 right, out of 100 entries.

Receiver Operating Characteristic Curve(ROC Curve)

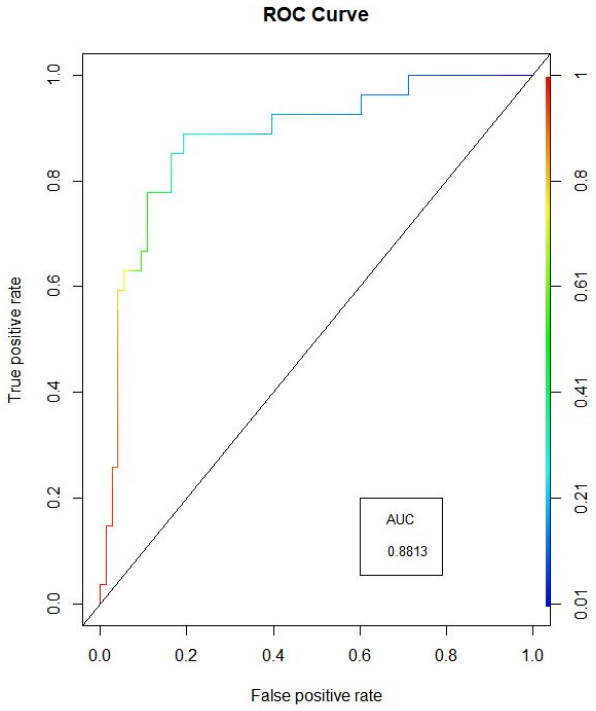


Figure80

The ROC curve shows the trade-off between true positive rate(TPR) and false positive rate(FPR). The ROC is far from the 45-degree diagonal so that the accuracy of the model performs well. On the other hand, AUC is 0.8813. Therefore, the AUC measure mentions that the TPR rate is higher than FPR rate in our model.

Decision Rule Cut-Off

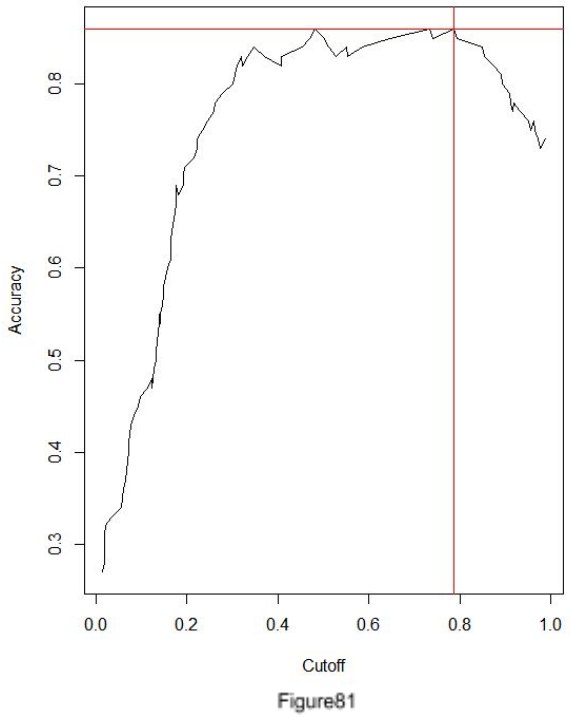


Figure81

The cut-off of the original model is 0.5. If the prediction probability is higher than 0.5, the model will classify it as 1 in survived, vice versa. However, the decision rule may not be the best. By using library "ROCR", the prediction accuracy can be simulated through different cut-off. As the figure is shown, 0.5 cut-off is not the optimum accuracy of the model. In conclusion, the best cut-off is 0.79 which gives 0.86 accuracies.

Using the new cut-off, the prediction on test data gets a 0.84 accuracy.

vii. *Support Vector Machine*

Introduction

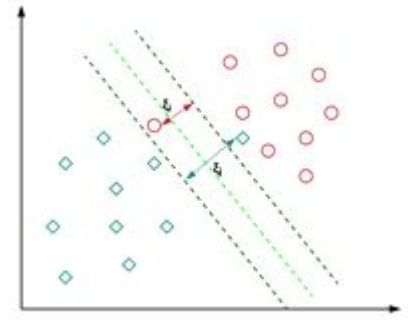


Figure82

Support-vector-machine is a supervised learning model for classification. As shown above, support-vector-machine builds a linear hyperplane(green dash line) to classify squares and circles. 2 black dash lines are the margin of the hyperplane.

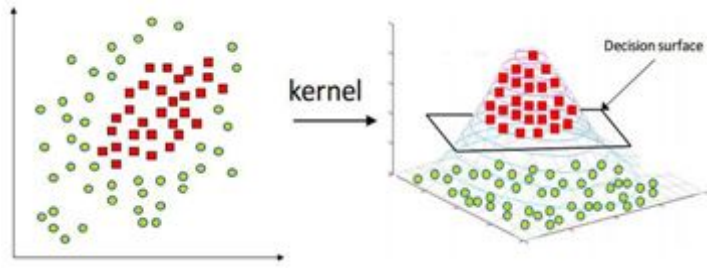


Figure83

Normally, data don't separate clearly. In "Titanic" case, the radial kernel is applied to solve the problem. If the data points are overlapping, the radial kernel will increase the dimensions. As the figure is shown, some hyperplanes fit after a new dimension is defined. The new dimension is defined to be:

$$f(x_1, x_2) = e^{-\frac{\|x_1 - x_2\|}{\sigma}}$$

SVM Model and Cross Validation

```
call:
best.tune(method = svm, train.x = Survived ~ ., data = tin_train, ranges = list(cost = c(0.01, 0.1, 1, 10, 100, 1000, 10000)))


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  214
```

Figure84

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost
    1

- best performance: 0.2025

- Detailed performance results:
   cost  error dispersion
1 1e-02 0.4150 0.06032320
2 1e-01 0.2325 0.05779514
3 1e+00 0.2025 0.06061032
4 1e+01 0.2175 0.07642171
5 1e+02 0.2175 0.06566963
6 1e+03 0.2250 0.05773503
7 1e+04 0.2450 0.05502525
```

Figure85

10-folds cross validation is applied in tuning the SVM model. As shown from the above figure, 0.01 to 10000 cost parameters are input in cross validation. The second figure shows the result that the 3rd row, cost=1, gives the least error and small dispersion. Therefore, 1 cost parameter is the best within 7 choices. On the other hand, the first figure also shows the radial is the best kernel method.

Confusion Matrix & Interpretation

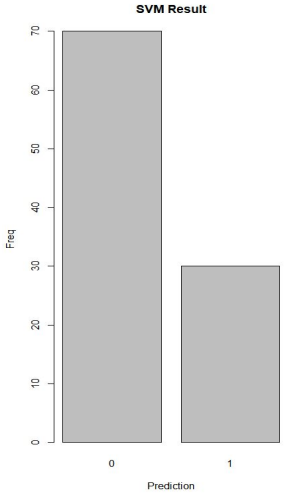


Figure86

The result of SVM is quite good. From the histogram, the proportion of "0" is much greater than "1". The model predicted 73 "0" and 27 "1". The prediction gets 87 right, out of 100 entries. The result indicated that SVM is a strong learner as strong as GBM. However, SVM can't handle NA and the variables must be numeric as a vector.

```
pred  0  1
   0 65  5
   1  8 22
```

Figure87

Conclusion

Model	Accuracy
KNN	0.752
Logistic Regression	0.90
Classification Tree	0.78
Random Forest	0.82
Linear Discriminat Analysis	0.78
Gradient Boosting	0.85
Support Vector Machine	0.87

As the table shown, logistic regression has the highest accuracy in prediction because it performs well in binary classification. However, it is very weak in non-binary classification problem. On the other hand, KNN has the lowest accuracy because it only performs well only in low dimension.

Contribution

CHENG Wing Ryan	25%
Hung Fan Hin	25%
Lee Ka Hin	25%
Li Tak Leong	25%