**Prediction of the Success of Bank Telemarketing and Model Comparison**

Zhiying Li(zl2697)

Yuting He(yh3054)

Xiaoming Chen(xc2479)

**1.Introduction**

Marketing selling constitute a typical strategy to enhance business. Companies not only use direct marketing when targeting segments of customers by contacting them to meet a specific goal, but also through various indirect channels, telephone (fixed-line or mobile) being one of the most widely used. Using data driven methods like the machine learning techniques, it is more efficient and effective for the companies to grasp the information of their customers, thus improving their selling process. In this paper, we use data driven methods to predict the success of bank telemarketing and learn which kind of customers are more likely to buy the products.

We analyze a recent and large dataset (45,211 records) from a Portuguese bank. The data were collected from 2008 to 2013, thus including the effects of the global financial crisis that peaked in 2008. We compare four Machine Learning models (Logistic Regression, Random Forest, Neural Network and SVM) using a AUC-ROC and classification metric. We find that the best model (Random Forest) could most accurately predict the possibility of the customers buying the products of the bank.

It is without a shadow of doubt that there are several classification models appropriate for the setting of our problem, such as the classical Logistic Regression (LR), Random Forest (RF), the more recent Neural Networks (NNs) and Support Vector Machines (SVMs). RF, NN and SVM are more flexible (i.e., no a priori restriction is imposed) when compared with classical statistical modeling (e.g., LR) presenting learning capabilities that range from linear to complex nonlinear mappings. Due to such flexibility, RF,NN and SVM tend to provide more accurate predictions, but the obtained models are difficult to be understood by humans, which is a loss of interpretability. In this paper, a binary classification task was modeled using the four machine learning algorithm that was fed with 16 attributes (after a feature selection step), using 4/5 randomly selected customers for training and 1/5 for testing. In the training samples, we cross validated for different models.

**2. Data Description**

Our bank data set is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be (or not) subscribed.

The data contains 45,211 instances and 16 features for each instance. Our response variable is whether the customer will make a deposit or not. We want to build different models to our data in order to illustrate how important the role that different features play in predicting our target problem. What’s more, we will also compare the performances of different models to find the best model that fits our data set.

The response variable is a categorical variable, for which 1 represents that the customer will buy this product and 0 represents that the customer will not buy the product. As for the categorical predictors, the data contains job (types of job), marital (marital status), education, default (whether customer has credit in default), housing(whether customer has housing loan), loan (has personal loan or not), contact (contact communication type), month (last contact month of year) and poutcome (outcome of the previous marketing campaign).

Also, there are seven numerical predictors: age, balance (average yearly balance, in euros), day (last contact day of the month), duration (last contact duration, in seconds), campaign (number of contacts performed during this campaign and for this client), pdays (number of days that passed by after the client was last contacted from a previous campaign, -1 means client was not previously contacted) and previous (number of contacts performed before this campaign and for this client). The predictors are summarized in figure 2-1 below:

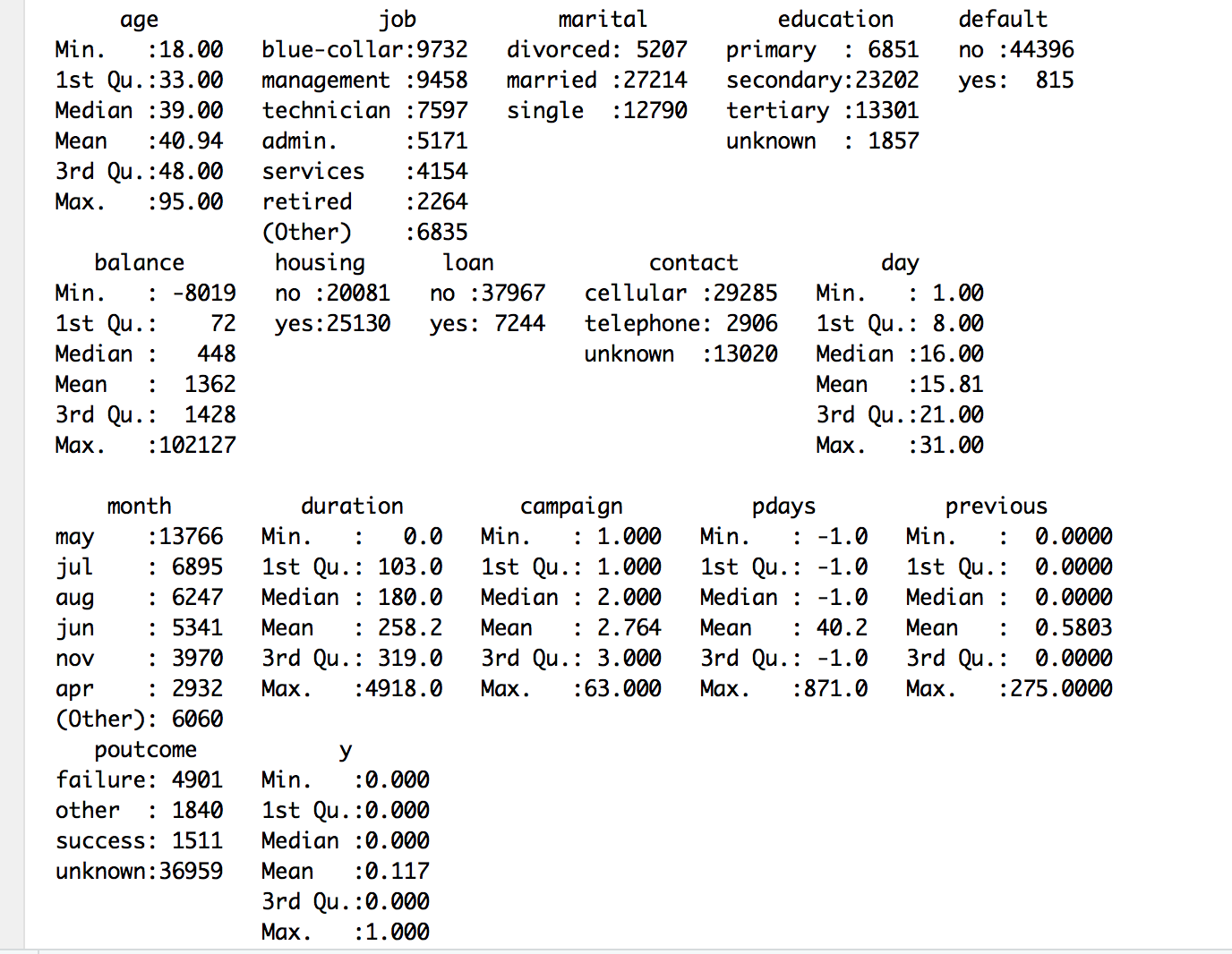


Figure 2-1

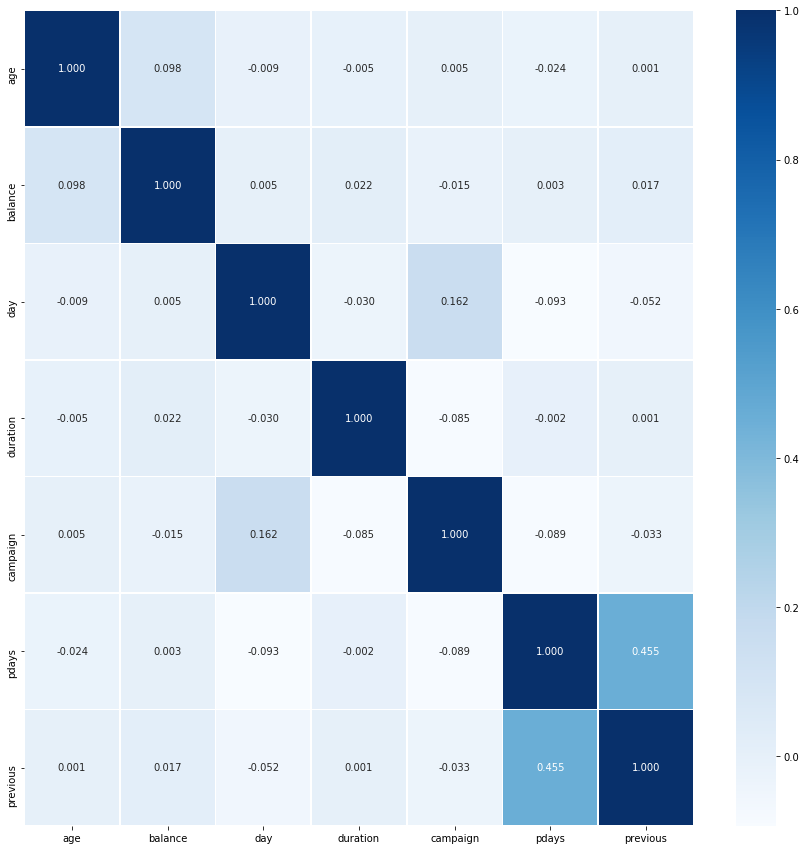


Figure 2-2

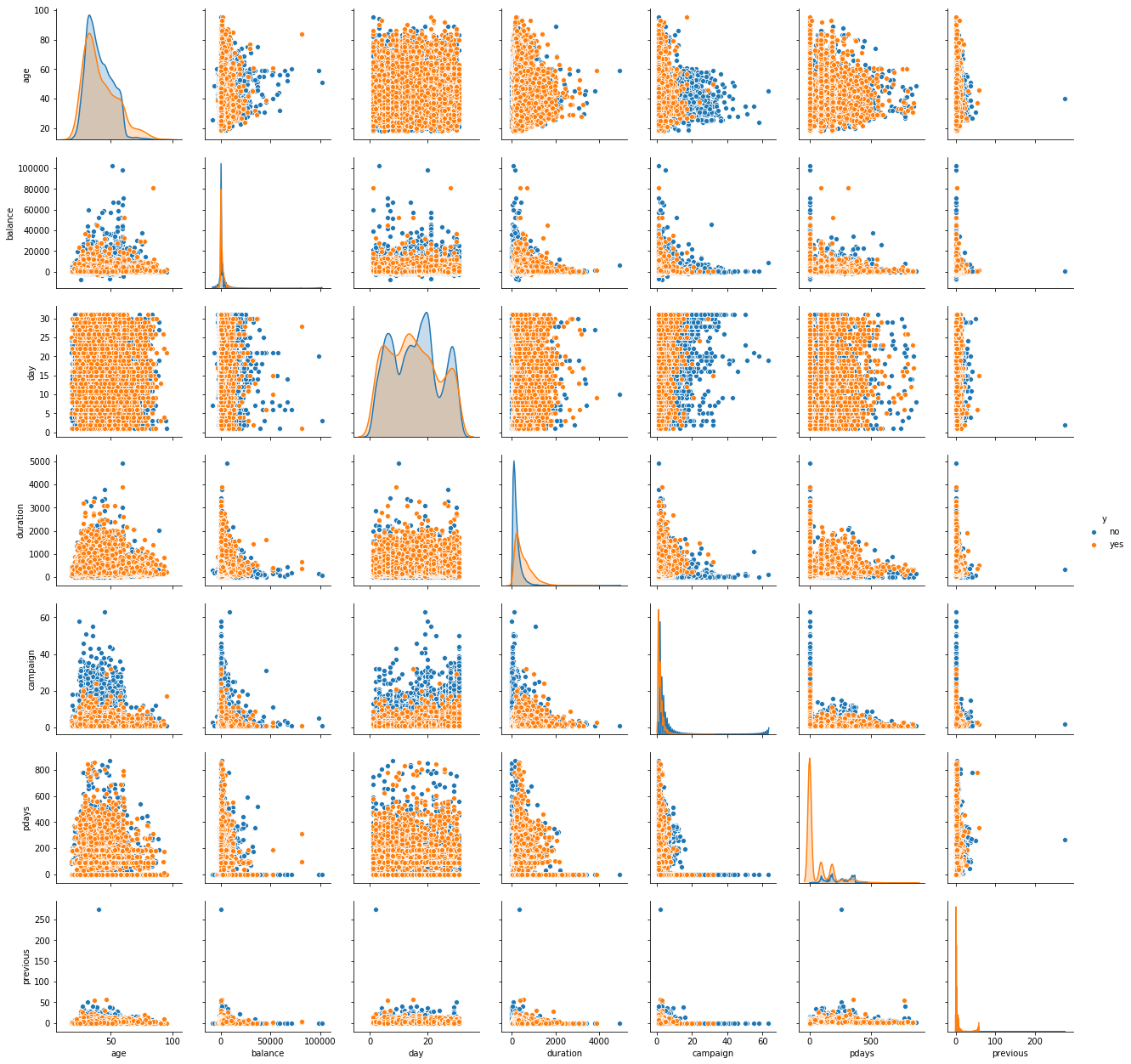


Figure 2-3

Figure 2-2 is a heatmap of the correlation between the numerical variables. The figure indicates that there does exist small correlation between variables. However, the correlations are not very strong except the correlation between p-days and previous, which is 0.455.

Figure 2-3 is the pair plot of numerical predictors based on whether the customers buy the product or not. From the pair plot, we cannot observe obvious correlation between the numerical predictors. In addition, we can observe that the average balance of the people who buy the product is lower than the people who do not buy the product, which accords with the fact since people use their deposit to buy financial product. And the duration of people who buy the product is lower than the people who do not buy the product since the marketing personnels will spend more time to persuade the clients who are not willing to buy the product. The number of contacts performed before this campaign and for those clients who will buy the product is much more than the clients who will not buy the product, since they have more interests in the product and so they are intended to get more information about the product.

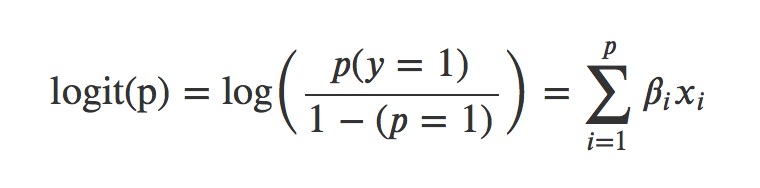
**3. Models**

For our models, we use mainly five supervised learning models: Lasso/Ridge Regression, Logistic Regression, Support Vector Machine, Random Forest and Neural Networks. We will build these models to fit our bank data set to figure out which predictors are significant in the prediction of whether clients of the bank will buy the deposit product. Also, we will evaluate these five models based on their error rates to figure out which model is the best model that can describe the bank data set best.

**3.1 Logistic Regression**

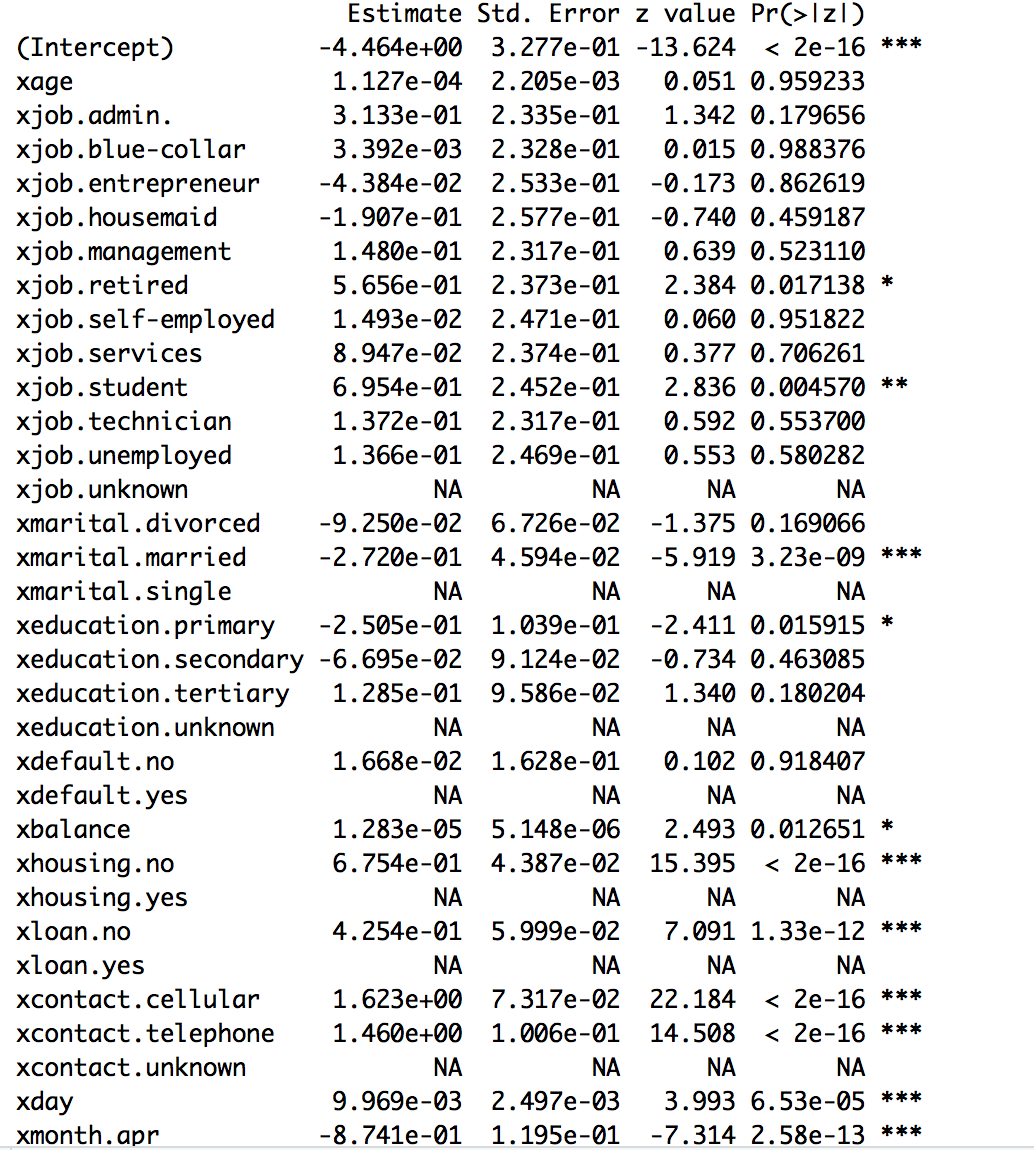
*3.1.1 Logistic Regression Methodology*

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. Mathematically, logistic regression estimates a multiple linear regression function defined as:



*3.1.2 Logistic Regression Analysis*

Figure 3-1 below is the result of fitting logistic regression model. From the p-values of the predictors, we can conclude that the most important features are married status, no housing loan, no personal loan, cellular and telephone contact status, last contact day, last contact month, contact duration, number of contacts and the success outcome status of the last campaign. The predictors age, job, divorced marital status, education, default, pdays, previous and failure outcome status of the last campaign are not significant in the logistic regression model.



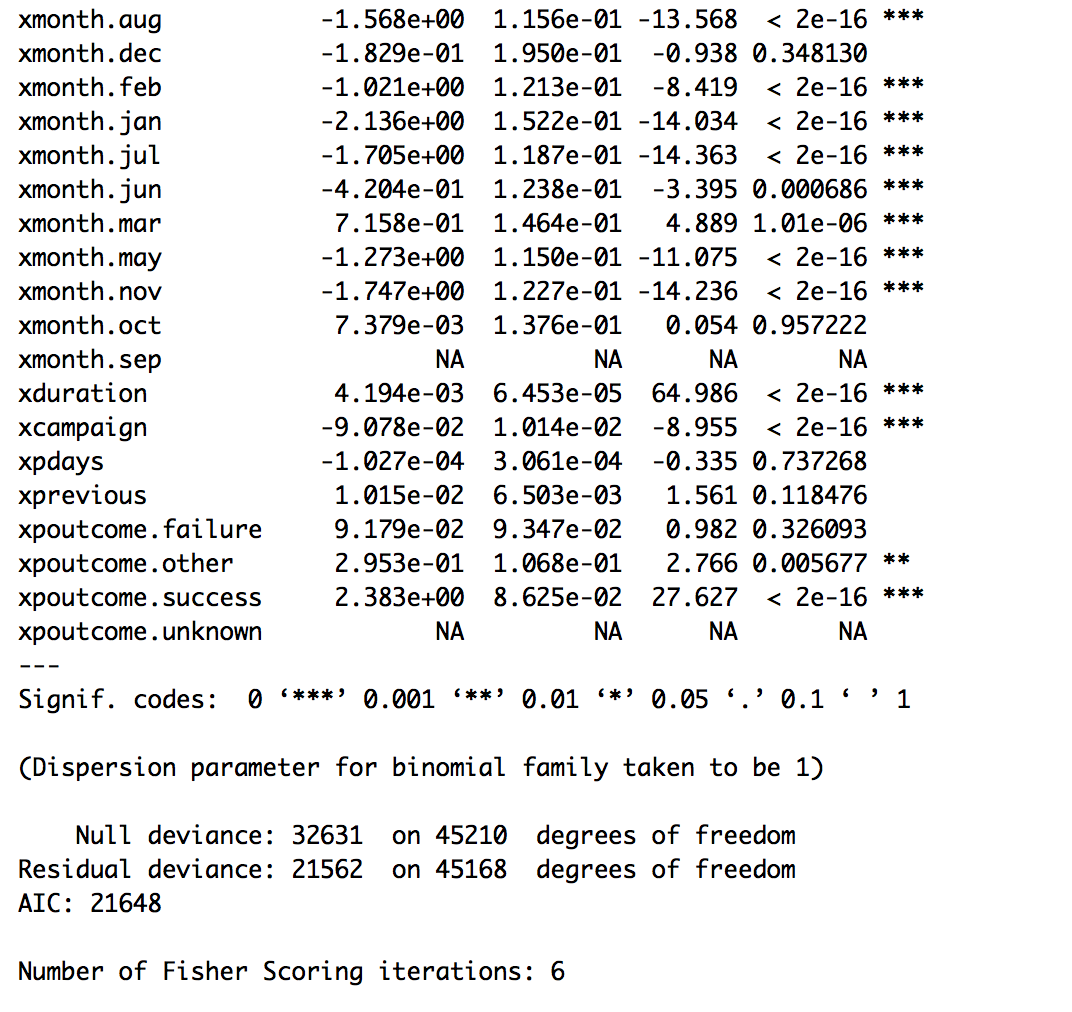


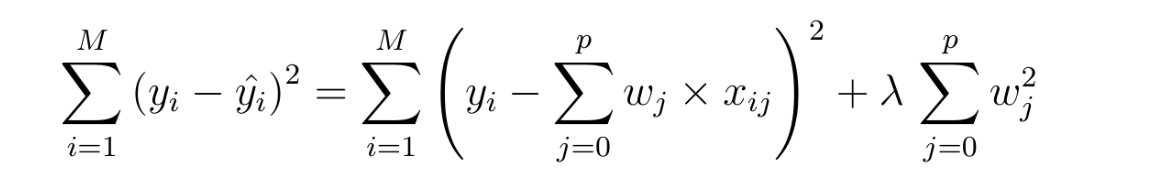
Figure 3-1

**3.2 . Lasso/ Ridge Regression**

*3.2.1 Lasso/ Ridge Regression Methodology*

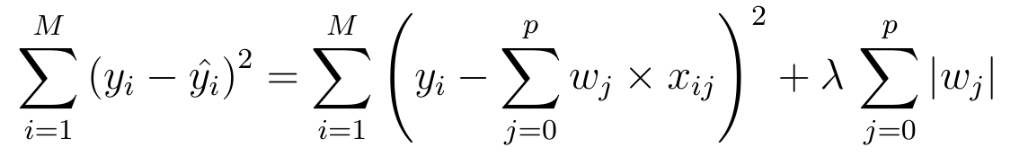
Ridge and Lasso regression are some of the simple techniques to reduce model complexity and prevent overfitting which may result from simple linear regression. The difference between ridge regression, lasso regression and linear regression is that ridge regression and lasso regression add a penalty term on the loss function.

In ridge regression, the cost function is altered by adding a penalty equivalent to square of the magnitude of the coefficients. The object function of ridge regression is to minimize the function:

.

Ridge regression puts constraint on the coefficients (w). The penalty term (lambda) regularizes the coefficients such that if the coefficients take large values the optimization function is penalized. So, ridge regression shrinks the coefficients and it helps to reduce the model complexity and multi-collinearity. The cost function for Lasso (least absolute shrinkage and selection operator) regression can be written as:

.



This type of regularization (L1) can lead to zero coefficients i.e. some of the features are completely neglected for the evaluation of output. So Lasso regression not only helps in reducing overfitting but it can help us in feature selection.

As for the bank data, there does exist collinearity between the features, which increases the variance of estimation. So we construct ridge and lasso regression model instead of building linear regression model in order to stabilize our estimation.

*3.2.2 Lasso/ Ridge Regression Parameter Tuning*

In the object functions of lasso and ridge regression, the penalty coefficient (lambda) need to be chosen in order to achieve the minimum prediction error.

Figure 3-2 is the result of cross validation of the penalty parameter of ridge regression. Here, we choose the lambda which can achieve the maximum AUC (lambda = 0.01391628).

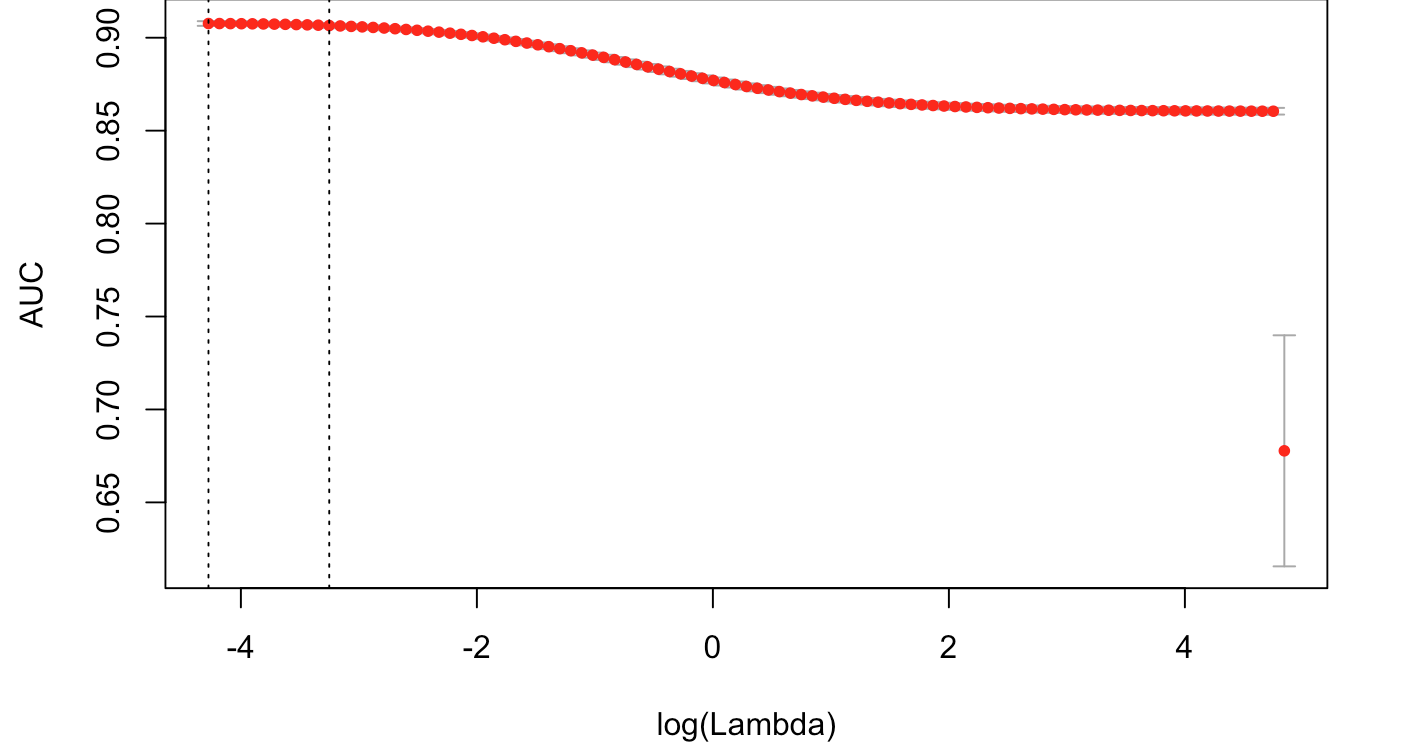


Figure 3-2

Figure 3-3 is the result of cross validation of the penalty parameter of lasso regression. Here, we choose the lambda = 0.003068721. Figure 3-4 is the plot of coefficients against log lambda, it shows that L1 penalty can shrink some coefficients directly to zeros.

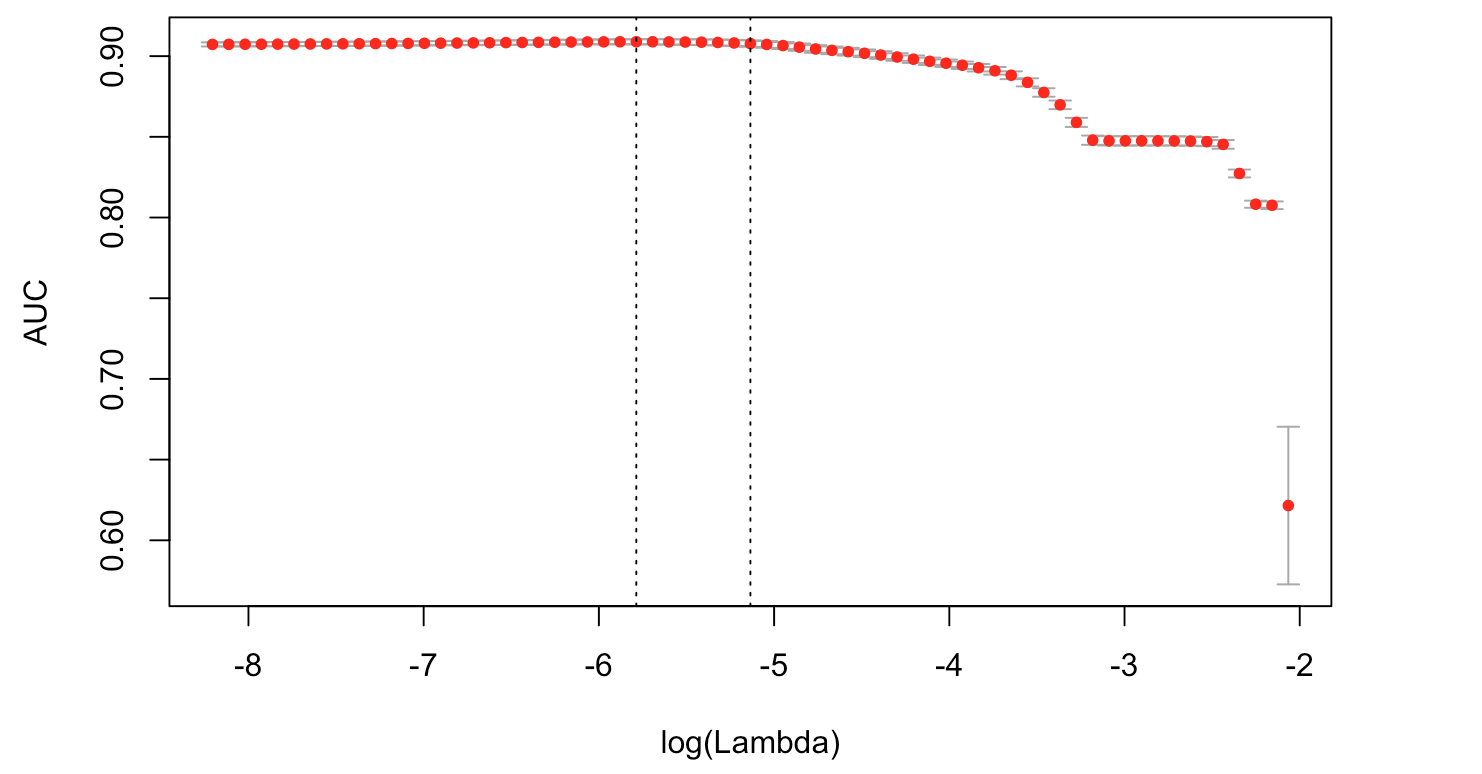


Figure 3-3

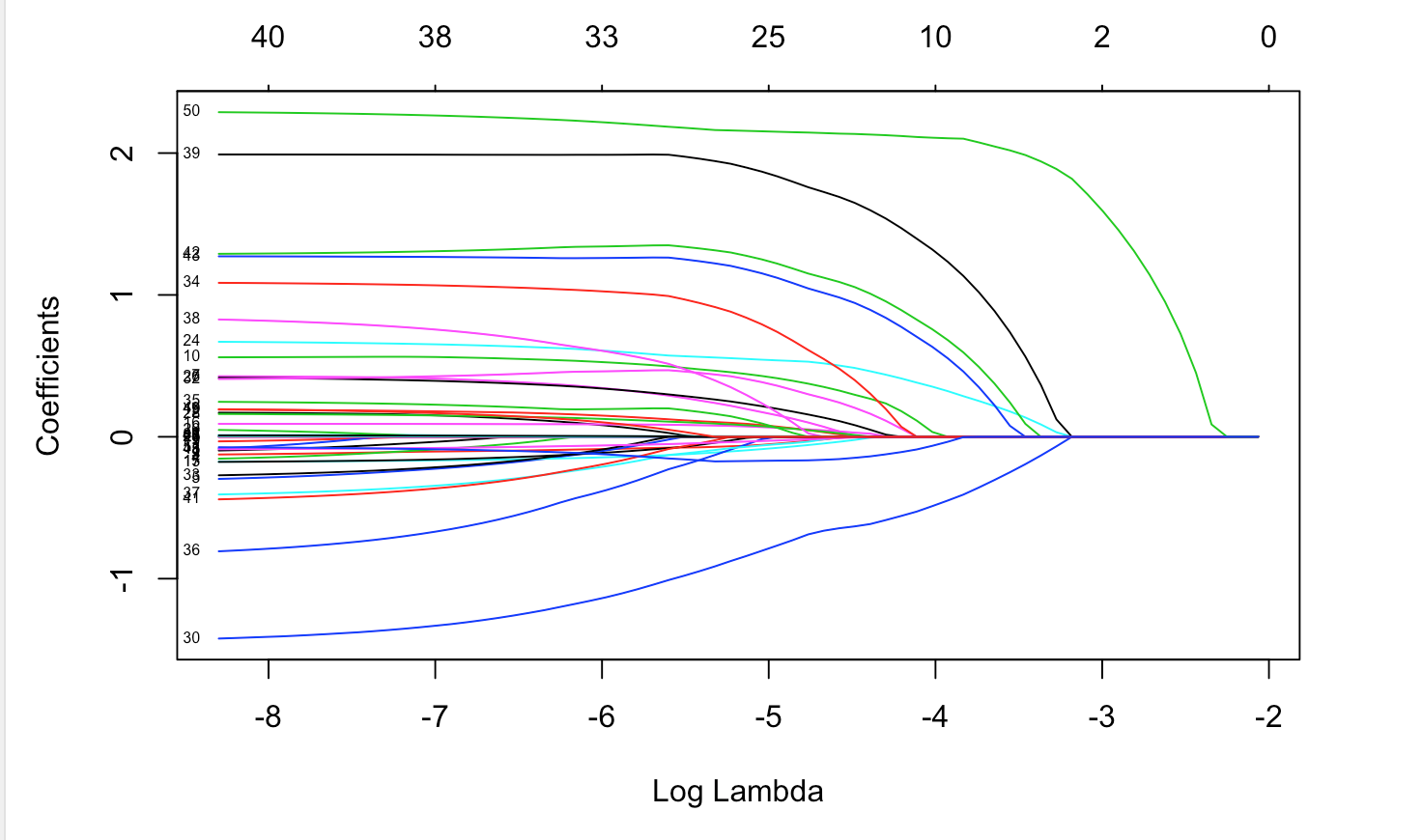


Figure 3-4

*3.2.3 Lasso/ Ridge Regression Analysis*

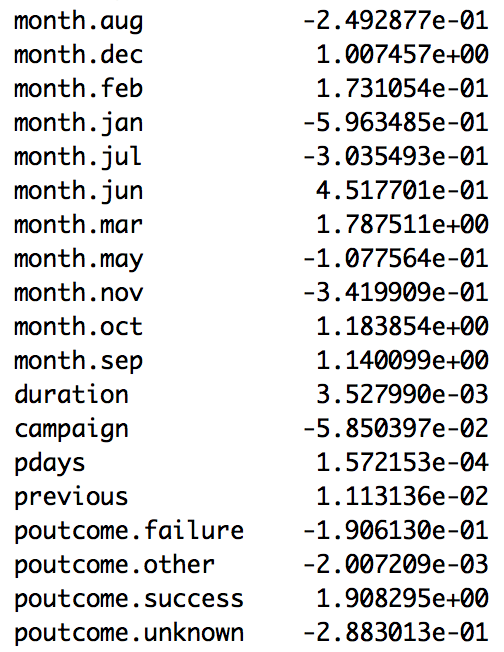
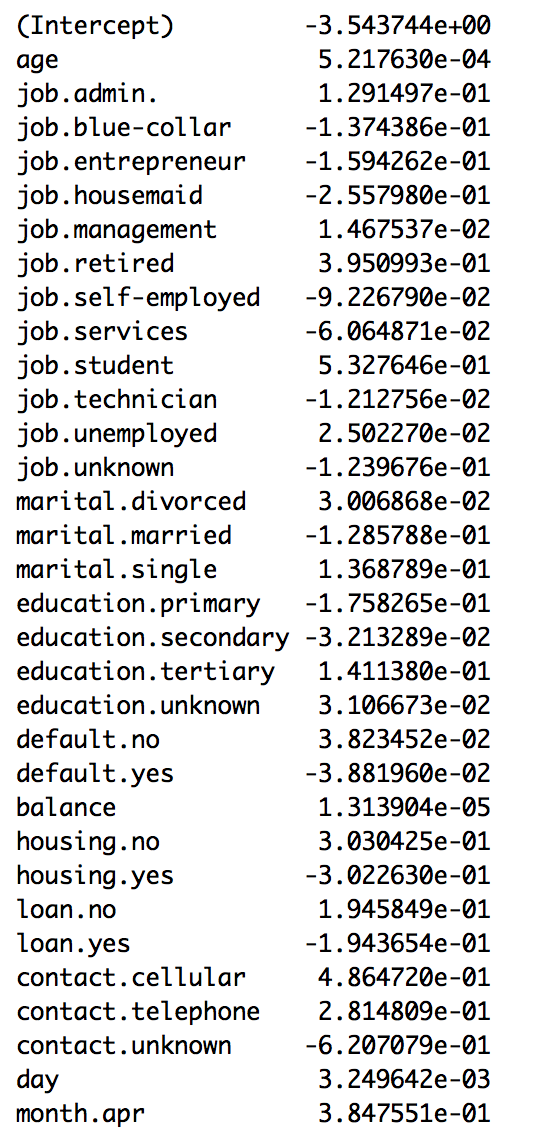


Figure 3-4

Figure 3-4 is the coefficient of ridge regression. The result indicates that dec, mar, oct, sep categories of month and the success category of poutcome have the largest partial effect on the prediction of whether a client will buy product. It means that the last contact happened in these months and the success outcome from the previous campaign have the most positive effect on predicting the client will buy the product. On the contrast, the predictor age has the smallest partial effect on the prediction.

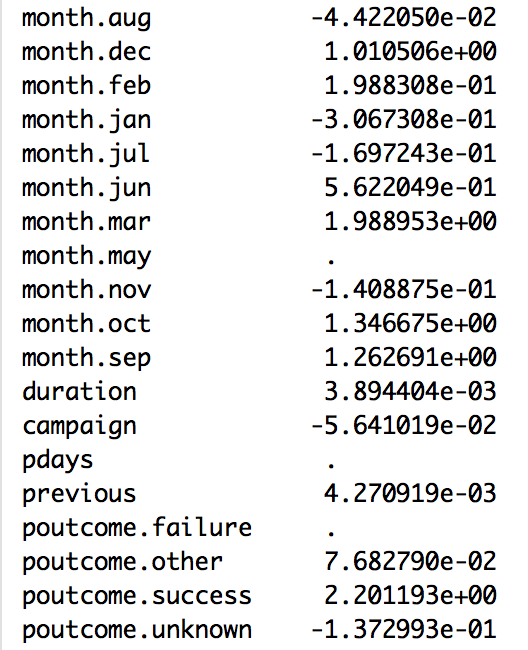
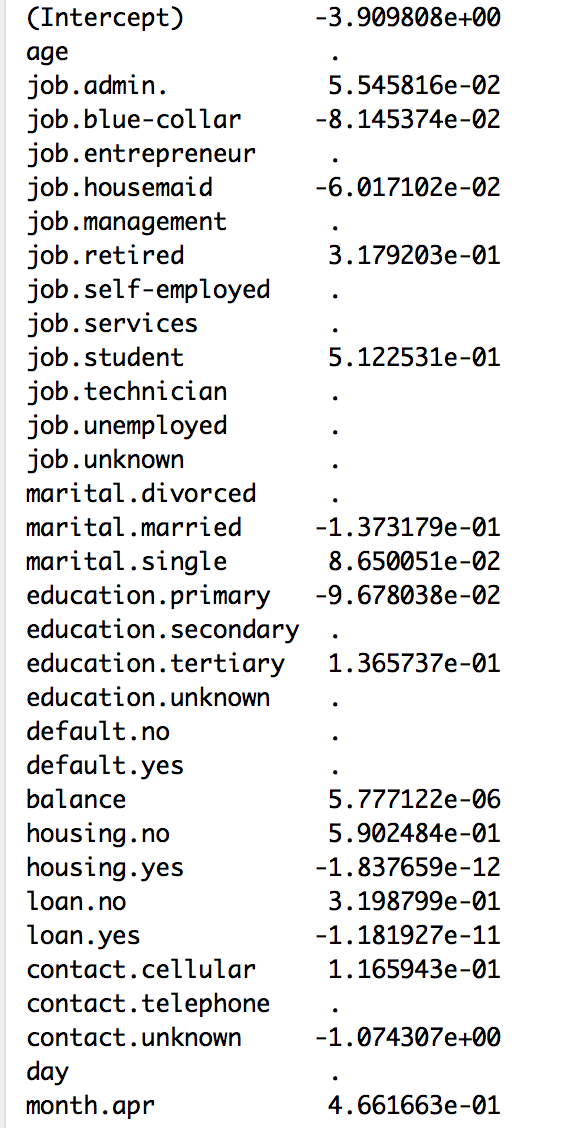


Figure 3-5

Figure 3-5 is the coefficients of lasso regression. The L1 penalty of lasso regression shrinks the coefficients of age, job (entrepreneur, job.management, self-employed, services, technician, unemployed categories), divorced marital status, secondary education status, default, telephone contact status, etc to zeros. In lasso regression, the predictors month and poutcome have the largest partial effect on the prediction, which is the same as the result from ridge regression.

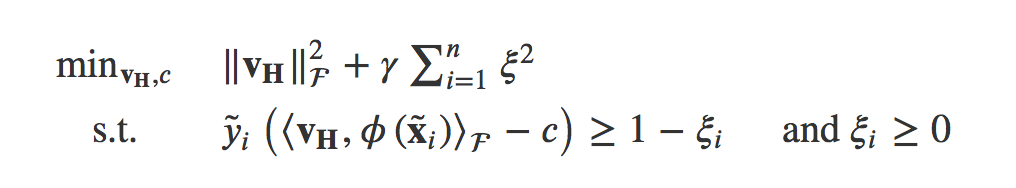
However, since ridge and lasso regression may lead to the predicted response not equal to 0 or 1. So we also built support vector machine model, random forest and neural network model to fit the bank data.

**3.3. Support Vector Machine**

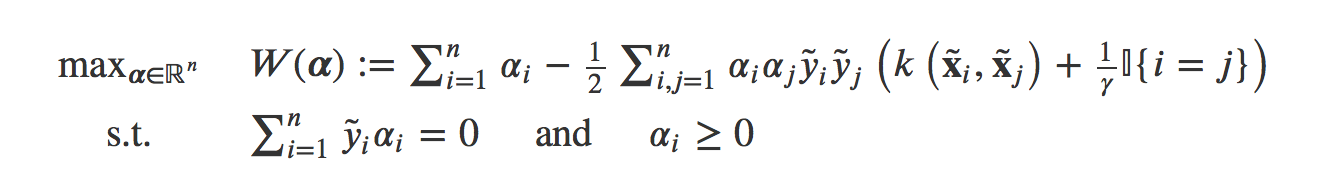
*3.3.1 Methodology*

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. The objective of SVM is to find a hyperplane which maximize the margin separating the two categories of data. It is a classification method of the supervised learning.

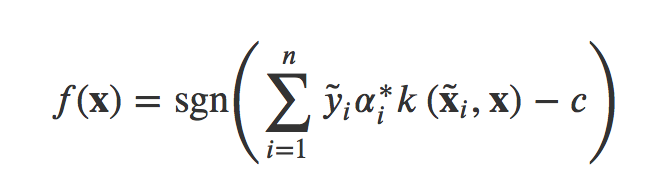
The following formula formally describes SVM.



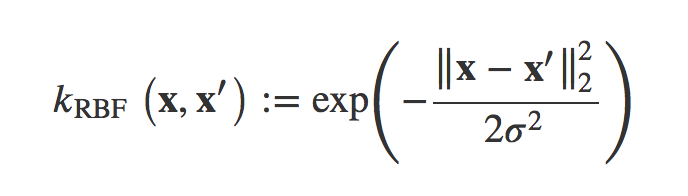
Where is the support vectors, is the penalty term of the misclassified object, is a transforming function of the data to a higher dimensional space. The variables ξi are slack variables. To solve the problem implicitly, we can solve a dual optimization problem:



The classifier is:



Where *k* is the kernel of the SVM. There a many choices of the kernels, such as the linear kernel, the Gaussian kernel, etc. In this paper, we choose the Gaussian kernel:



*3.3.2 Parameter tuning*

Once the model is designed and determined, we will evaluate the parameters using the cross validation. We will choose the appropriate which is the penalty term, the larger , the larger cost of allowing a point on the wrong side of the SVM classifier. The metric we use here is the classification accuracy. Figure 3-6 shows the classification accuracy rate of different penalty terms. We can see all the results appear similar, indicating that the SVM model is quite robust. The best result is 89.13, when is equal to 10.

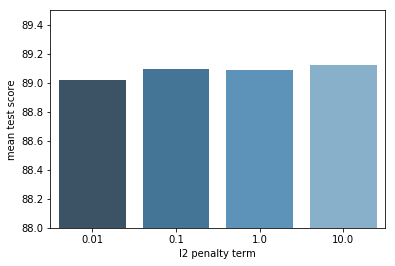


Figure 3-6

**3.4. Random Forest**

Random forests, or random decision forests are an ensemble learning method for classification and regression. It operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

The advantage of Random forests is that it correct for decision trees' habit of overfitting to the training-set, which decreases the variance of the model and thus improve the MSE of the model, whereas the disadvantage is that every time we fit a tree to a bootstrap sample, we get a different tree, and this cause the loss of interpretability.

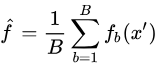
*3.4.1 Methodology*

The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging to tree learners. Given a training set *X* = *x1*, ..., *xn* with responses *Y* = *y1*, ..., *yn*, bagging repeatedly (*B* times) selects a random sample with replacement of the training set and fits trees to these samples:

For *b* = 1, ..., *B*:

1. Sample, with replacement, *n* training examples from *X*, *Y*; call these *Xb*, *Yb*.
2. Train a classification or regression tree *fb* on *Xb*, *Yb*.

After training, predictions for unseen samples *x'* can be made by averaging the predictions from all the individual classification trees on *x'*:



For each individual tree, we use CART based on the split criterion of Gini impurity, which is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.

*3.4.2 Parameter tuning*

As what we have done in the previous model of logistic regression and SVM, we use the cross validation for different hyper-parameters of Random Forest. We choose to test the n\_estimators, which is the number of trees as well as the maximum depth of each sub trees.

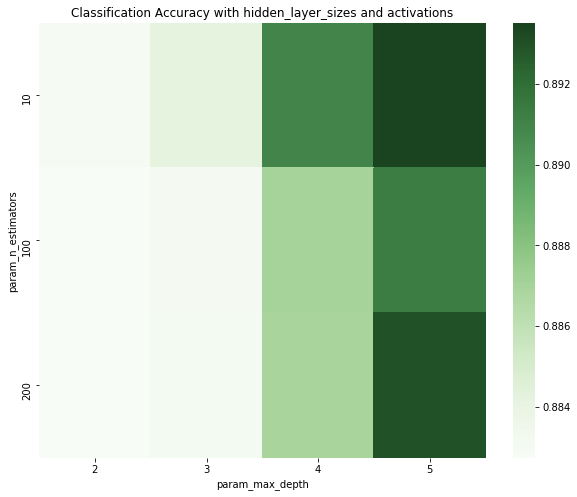


Figure 3-7

Fig 3-7 shows a heatmap of the combination of the number of trees and maximum depth of each tree. The metric we use is still the classification accuracy. The overall model seems to be good since the last accuracy is 0.884. The largest accuracy is 0.895, when the maximum tree depth is 5 and amount of trees is 10.

*3.4.3 Feature importance*

Different with the coefficients or the significance level of the logistic regression which is interpretable, it is hard to give a “coefficient” to the Random Forests mode. However, what we can do is to calculate the feature importance. When a tree is built, the decision about which variable to split at each node uses a calculation of the Gini impurity. For each variable, the sum of the Gini decrease across every tree of the forest is accumulated every time that variable is chosen to split a node. The sum is divided by the number of trees in the forest to give an average. The scale is irrelevant: only the relative values matter.

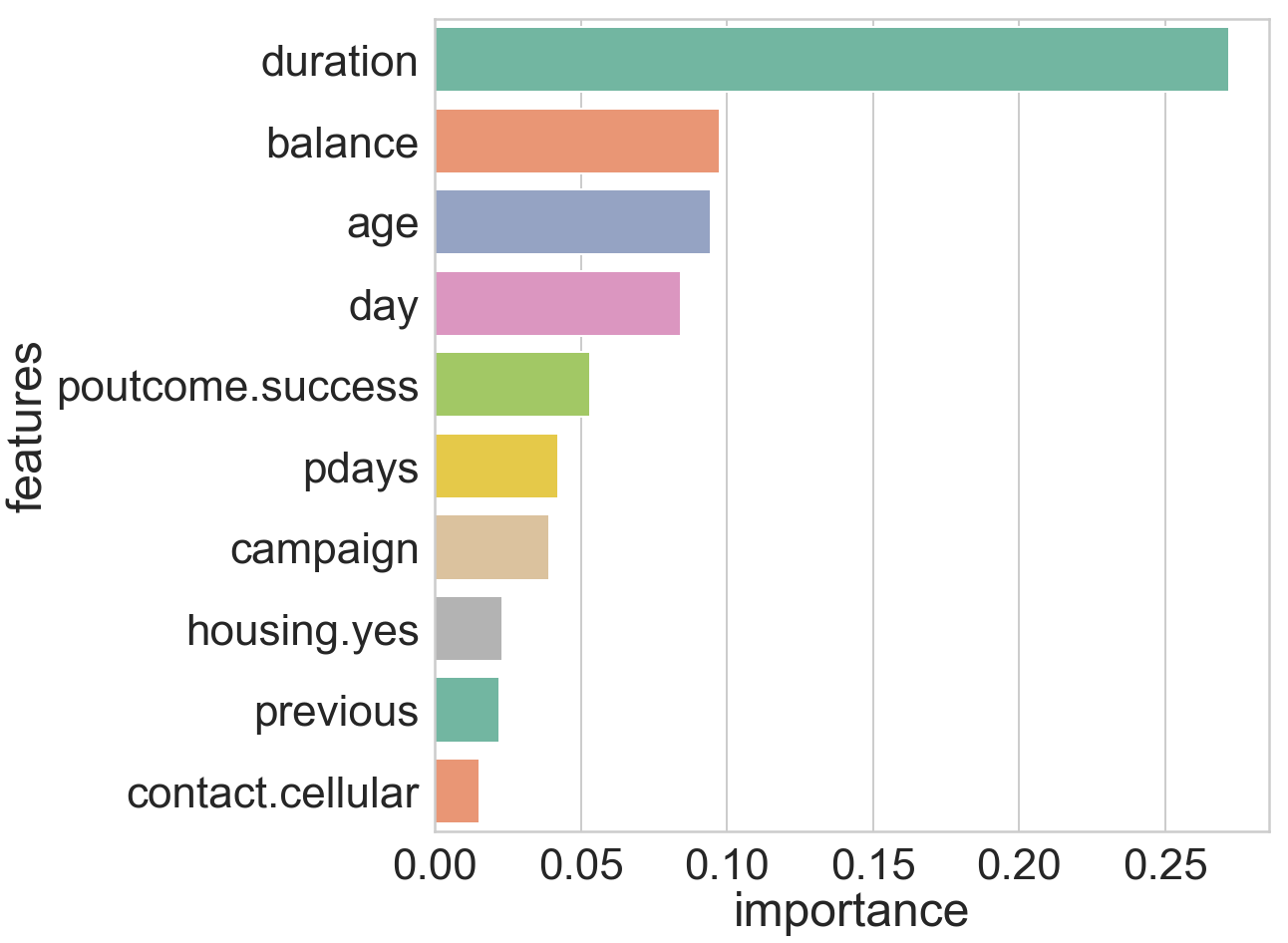


Figure 3-8

Figure 3-8 shows the importance of the features. It should be emphasized that we do not know the “direction” of each variable, but just the magnitude of importance. Amongst the features, duration plays the most significant role, followed by balance, age and day. Duration is the time duration of the sales person contacting with the client. It makes sense that the longer time they talk, the larger possibility the client will buy the product.

**3.5 Neural Network**

A neural network is a supervised learning method which can be applied to both regression and classification problems. The central idea of neural network is to extract linear combinations of the inputs as derived features, and then model the target as a nonlinear function of these features.

*3.5.1 Methodology*

Figure 3-9 shows the structure of a feedforward neural network. There are three layers: the input layer, the hidden layer and the output layer.

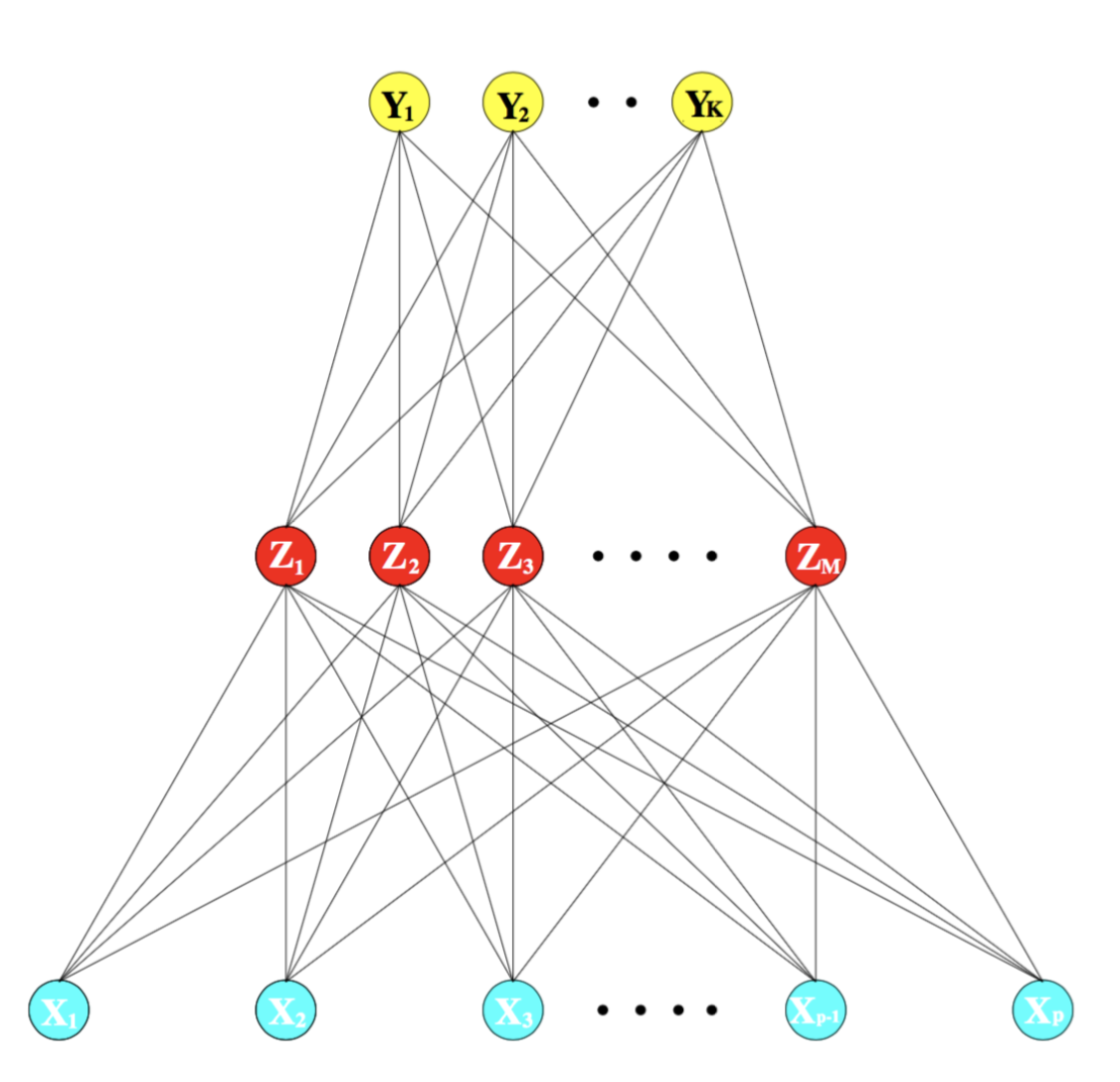
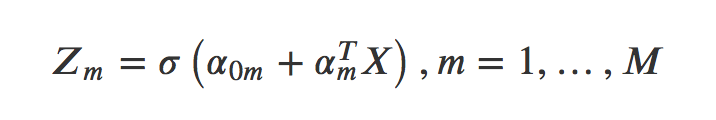


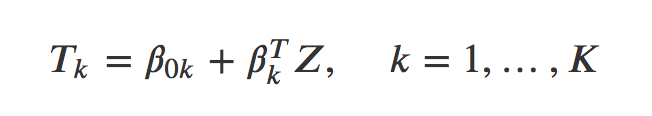
Figure 3-9

A general description of the neural network is:

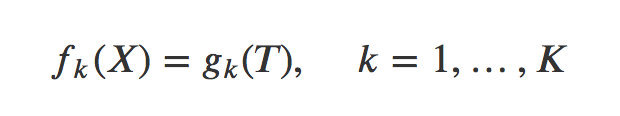
In the input layer, derived features Zm are obtained by applying the activation function σ to linear combinations of the inputs:



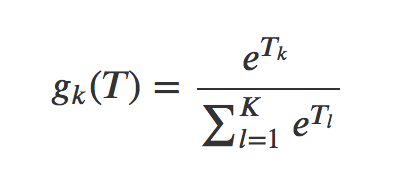
In the hidden layer, the target Tk is modeled as a function of linear combinations of Zm:



In the output layer, the output function gk(T) allows a final transformation of the vector of outputs T:

****

Where is the activation function. There are several choices of the activation function, such as the sigmoid function, the relu activation, and the tanh function, etc. For the K-classification, we use the softmax function:



When fitting the weight of the model, we use the back-propagation and the gradient descent.

*3.5.2 Parameter tuning*

We use 10 fold cross validation to choose the parameters of the hidden layer size and the activation function. From Figure 3-10, we can see that the logistic function, which is a sigmoid function does a better job than the others. The best parameters combination are hidden layer size 50 and logistic activation function.

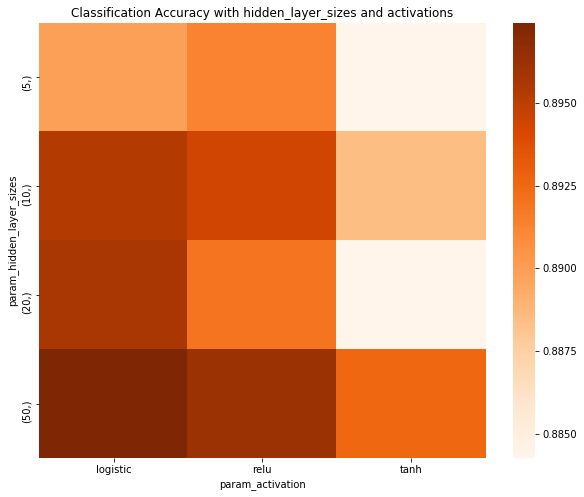


Figure 3-10

**3.6 Model Comparison**

After we have trained the models we compare with each other. In this model comparison process, we split the data into the training data and test data with the percentage of 80% training and 20% test data. We first train the data for each model and use two metrics: ROC-AUC and the classification accuracy.

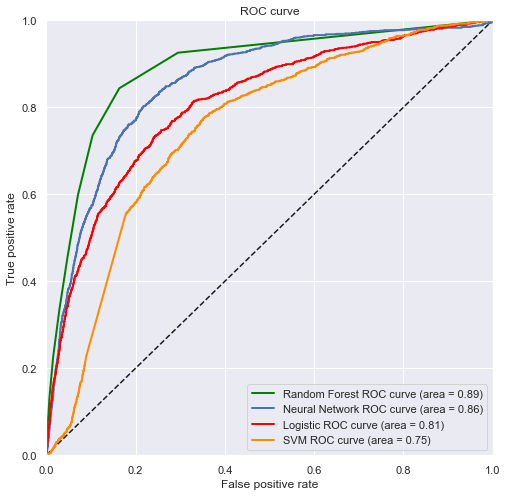
AUC-ROC curve is a performance measurement for classification problem at various thresholds settings. ROC is a probability curve and AUC represents degree or measure of separability. It tells how much model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s.

We generate both the 0-1 prediction for measure the classification accuracy, and the probability for calculating AUC-ROC for the four models: Logistic regression with penalty, Random Forest, SVM and Neural Network. We pick the best performances of the parameters achieved by the cross validation of them relatively. We find that the Random Forest performs the best in both ROC-AUC and Classification Accuracy. The result is shown in Table 3-1 and Figure 3-11.

**Table 3-1**

Comparison of models.(bold denotes the best value).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Metric | Logistic Regression | Random  Forest | SVM | Neural Network |
| ROC-AUC | 0.81 | **0.89** | 0.75 | 0.86 |
| Classification Accuracy | 0.85 | **0.92** | 0.88 | 0.90 |

  
Figure 3-11

**4. Conclusion**

Under this context, we predict the result of a telemarketing phone call to sell long term deposits is a valuable tool to support client selection decisions of bank campaign managers. In this study, we uses a machine learning approach for the selection of bank telemarketing clients. We analyzed a recent and large Portuguese bank dataset, collected from 2008 to 2013,with a total of 45,211 records. The goal was to model the success of subscribing a long-term deposit using attributes that were known.

During the modeling phase, we used 16 relevant features with no scaling of data. In the logistic regression model, we can conclude that the most important features are married status, no housing loan, no personal loan, cellular and telephone contact status, last contact day, last contact month, contact duration, number of contacts and the success outcome status of the last campaign. For example, married status has a negative effect to the model. As an interesting outcome from Random Forest model, the duration of the call, which highly affects the probability of a success contact or the amount that is deposited in the bank, is outstandingly significant. Also the average yearly balance, and age variables.

Finally. four ML models were compared: logistic regression, random forest, neural networks and support vector machines. These models were compared using two metrics, area of the receiver operating characteristic curve (AUC) and the prediction accuracy. The best results were obtained by the RFs, which resulted in an ROC-AUC of 0.89, and Classification Accuracy of 0.92. Such AUC corresponds to a very good discrimination.

**Reference**

Saptashwa. (Sep 26, 2018). Ridge and Lasso Regression: A Complete Guide with Python Scikit-Learn. Retrieved from <https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b>

Ricardo Carvalho. How to use Ridge Regression and Lasso in R. Retrieved from <http://ricardoscr.github.io/how-to-use-ridge-and-lasso-in-r.html>

Paulo Cortez, Data mining with neural networks and support vector machines using

the r/rminer tool, Advances in Data Mining. Applications and Theoretical Aspects,

6171, Springer, 2010, pp. 572–583.

Ian H.Witten, Eibe Frank, Data Mining: Practical Machine Learning Tools and Techniques,

2nd edition Morgan Kaufmann, 2005.

**Appendix**

* Link to the data set: <https://archive.ics.uci.edu/ml/datasets/Bank+Marketing#>
* Group contribution

Yuting He: data description, lasso regression, ridge regression, logistic regression

Zhiying Li: SVM, Random forest, Neural Network, model comparison

Xiaoming Chen: report introduction, report conclusion, poster

**R code**

---

title: "5223project"

author: "Yuting He"

date: "2019/4/27"

output: html\_document

---

```{r setup, include=FALSE}

knitr::opts\_chunk$set(echo = TRUE)

```

```{r}

setwd("~/Desktop")

data <- read.csv("raw\_data.csv")

data$y <- ifelse(data$y == "no", 0, 1)

Y <- data$y

head(data)

summary(data)

```

```{r}

library(dummies)

M <- data[,-ncol(data)]

M <- dummy.data.frame(M, sep = ".")

head(M)

data <- cbind(Y,M)

head(data)

```

##ridge regression

```{r}

library(glmnet)

x <- as.matrix(data[,-1]) # Removes class

y <- as.double(as.matrix(data[, 1])) # Only class

```

```{r}

# Fitting the model (Ridge: Alpha = 0)

set.seed(999)

cv.ridge <- cv.glmnet(x, y, family='binomial', alpha=0, parallel=TRUE, standardize=TRUE, type.measure='auc')

# Results

plot(cv.ridge)

cv.ridge$lambda.min

cv.ridge$lambda.1se

coef(cv.ridge, s=cv.ridge$lambda.min)

```

##lasso

```{r}

cv.lasso <- cv.glmnet(x, y, family='binomial', alpha=1, parallel=TRUE, standardize=TRUE, type.measure='auc')

# Results

plot(cv.lasso)

plot(cv.lasso$glmnet.fit, xvar="lambda", label=TRUE)

cv.lasso$lambda.min

cv.lasso$lambda.1se

coef(cv.lasso, s=cv.lasso$lambda.min)

```

##logistic regression

```{r}

glm.fit <- glm(y ~ x, data = data, family = binomial)

summary(glm.fit)

```

Python code

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Created on Thu Apr 25 19:36:07 2019

@author: lizhiying

"""

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import warnings

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import GaussianNB

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

from sklearn.neural\_network import MLPClassifier

from sklearn.metrics import roc\_auc\_score

from sklearn.metrics import roc\_curve

from sklearn.model\_selection import GridSearchCV

from tqdm import tqdm

####read the dataset####

data = pd.read\_csv('/Users/lizhiying/Desktop/Multivariate/bank/raw\_data.csv')

#----------------------------

'''EDA'''

sns.set\_style("whitegrid")

sns.set\_palette(sns.color\_palette("Paired"))

ax = plt.axes()

sns.countplot(x='y', data=data, ax=ax)

ax.set\_title('Target class distribution')

plt.show()

cols = ['age','balance','day','duration','campaign','pdays','previous','y']

eda = data[cols]

'''Correlation heatmap'''

f,ax = plt.subplots(figsize=(15, 15))

sns.heatmap(eda.corr(), annot=True, linewidths=.5, fmt= '.3f',ax=ax,cmap = 'Blues')

plt.show()

'''Scatter plot'''

sns.pairplot(eda,diag\_kind="kde")

sns.pairplot(eda,hue='y')

'''Paired plot'''

f, axarr = plt.subplots(2, 2, figsize=(15, 15))

sns.boxplot(x='age', y='y', data=data, showmeans=True, ax=axarr[0,0])

sns.boxplot(x='balance', y='y', data=data, showmeans=True, orient = 'h',ax=axarr[0, 1]).set(xscale = "log")

sns.boxplot(x='duration', y='y', data=data, showmeans=True, ax=axarr[1, 0])

sns.boxplot(x='campaign', y='y', data=data, showmeans=True, ax=axarr[1, 1])

axarr[0, 0].set\_title('age')

axarr[0, 1].set\_title('education')

axarr[1, 0].set\_title('duration')

axarr[1, 1].set\_title('campaign')

#plt.tight\_layout()

plt.show()

#-------------------------------------------------------------

'''Model'''

df = pd.read\_csv('/Users/lizhiying/Desktop/Multivariate/bank/processed\_data.csv',sep = ',')

y = df["Y"]

X = df.ix[:,1:44]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

#-------------------------------------------------------------

'''Logistic'''

lr = LogisticRegression(solver = 'lbfgs')

parameters = {'C':[0.01,0.1,1,10]}

clf = GridSearchCV(lr, parameters, cv=10)

clf.fit(X\_train,y\_train)

#clf.cv\_results\_

plot = sns.barplot(parameters.get('C'),clf.cv\_results\_['mean\_test\_score']\*100,palette="Blues\_d")

plot.set\_ylim(88.8,89.3)

plot.set\_ylabel("mean test score")

plot.set\_xlabel("l2 penalty term")

plot.show()

#-------------------------------------------------------------

'''Random Forest'''

rfc = RandomForestClassifier(n\_estimators=100)

rfc.fit(X\_train,y\_train)

# Feature importance

feature\_importances = pd.DataFrame(rfc.feature\_importances\_,index = X\_train.columns,

columns=['importance']).sort\_values('importance', ascending =False)

feature\_importances['features'] = feature\_importances.index.values

f,ax = plt.subplots(figsize=(15, 15))

sns.set\_context("poster",font\_scale = 2)

sns.barplot(y="features", x="importance", data=feature\_importances.iloc[0:10,:],

label="Feature Importance", palette="Set2")

#cross validation

rfc = RandomForestClassifier()

parameters = {'n\_estimators':[10,100,200],'max\_depth':[2,3,4,5]}

clf = GridSearchCV(rfc, parameters, cv=5)

clf.fit(X\_train,y\_train)

pvt = pd.pivot\_table(pd.DataFrame(clf.cv\_results\_),

values='mean\_test\_score', index='param\_n\_estimators', columns='param\_max\_depth')

plt.figure(figsize=(10,8))

plt.title('Classification Accuracy with hidden\_layer\_sizes and activations')

ax = sns.heatmap(pvt,cmap="Greens")

plt.show()

#-----------------------------------------------------

'''SVM'''

svc = SVC(kernel = 'rbf')

parameters = {'C':[0.01,0.1,1]}

clf = GridSearchCV(svc, parameters, cv=5)

clf.fit(X\_train,y\_train)

plot = sns.barplot(parameters.get('C'),clf.cv\_results\_['mean\_test\_score']\*100,palette="Blues\_d")

plot.set\_ylim(88.26,88.28)

plot.set\_ylabel("mean test score")

plot.set\_xlabel("l2 penalty term")

#------------------------------------------------------

''' Neural Network '''

mlp = MLPClassifier(activation='relu', solver='adam')

parameters = {'hidden\_layer\_sizes':[(5,),(10,),(20,),(50,)],'activation' : ["relu","logistic","tanh"]}

clf = GridSearchCV(mlp, parameters, cv=3)

clf.fit(X\_train,y\_train)

pvt = pd.pivot\_table(pd.DataFrame(clf.cv\_results\_),

values='mean\_test\_score', index='param\_hidden\_layer\_sizes', columns='param\_activation')

plt.figure(figsize=(10,8))

plt.title('Classification Accuracy with hidden\_layer\_sizes and activations')

ax = sns.heatmap(pvt,cmap="Oranges")

plt.show()

#----------------------------------------------------

''' AUC ROC'''

lr = LogisticRegression(solver = 'lbfgs',C = 10).fit(X\_train,y\_train)

y\_pred\_lr = lr.predict\_proba(X\_test)[:,1]

fpr\_lr, tpr\_lr, \_ = roc\_curve(y\_test, y\_pred\_lr)

auc\_lr = roc\_auc\_score(y\_test,y\_pred\_lr)

rfc = RandomForestClassifier(n\_estimators=10).fit(X\_train,y\_train)

y\_pred\_rf = rfc.predict\_proba(X\_test)[:,1]

fpr\_rf, tpr\_rf, \_ = roc\_curve(y\_test, y\_pred\_rf)

auc\_rf = roc\_auc\_score(y\_test,y\_pred\_rf)

svc = SVC(kernel = 'rbf',C = 0.01, probability =True).fit(X\_train,y\_train)

y\_pred\_sv = svc.predict\_proba(X\_test)[:,1]

fpr\_sv, tpr\_sv, \_ = roc\_curve(y\_test, y\_pred\_sv)

auc\_sv = roc\_auc\_score(y\_test,y\_pred\_sv)

mlp = MLPClassifier(activation='relu', solver='adam',hidden\_layer\_sizes = (10,)).fit(X\_train,y\_train)

y\_pred\_nn = mlp.predict\_proba(X\_test)[:,1]

fpr\_nn, tpr\_nn, \_ = roc\_curve(y\_test, y\_pred\_nn)

auc\_nn = roc\_auc\_score(y\_test,y\_pred\_nn)

plt.subplots(figsize=(8, 8))

plt.xlim(0, 1)

plt.ylim(0, 1)

plt.plot([0, 1], [0, 1], 'k--')

plt.plot(fpr\_rf, tpr\_rf,color = 'green',label = 'Random Forest ROC curve (area = {0:0.2f})'

''.format(auc\_rf),lw=2)

plt.plot(fpr\_nn, tpr\_nn, label = 'Neural Network ROC curve (area = {0:0.2f})'

''.format(auc\_nn),lw=2)

plt.plot(fpr\_lr, tpr\_lr,color = 'red',label = 'Logistic ROC curve (area = {0:0.2f})'''.format(auc\_lr),lw=2)

plt.plot(fpr\_sv, tpr\_sv ,color = 'darkorange',label = 'SVM ROC curve (area = {0:0.2f})'

''.format(auc\_sv),lw=2)

plt.xlabel('False positive rate')

plt.ylabel('True positive rate')

plt.title('ROC curve')

plt.legend(loc='best')

ax.grid(True)

plt.show()