430_project

Enguang Fan

5/14/2020

Section 1

This final Project aim to use PCA logistic regression method to predict if the customer will cancel their order(binary response"is_cancelled") based on existing datasets of booking record ranged 2015 to 2017. For comparison, we will include traditional stepwise subset method by AIC/BIC criteria, besides, we will also use regression tree and RandomForest. Notice that I will hide some code in pdf report, you may find them in rmd file.

This data set contains booking information for a city hotel and a resort hotel in Spain, and includes information such as when the booking was made, length of stay, the number of adults, children, and/or babies, and the number of available parking spaces, among other things.

The dataset can be retrieved from

https://www.kaggle.com/jessemostipak/hotel-booking-demand/data#

Data cleaning

We code all catagorical as integer variables since they will devastate the PCA analysis. This will not affect the final result(or just slightly). Interestingly, coding them as factor will not be accepted by PCA package. Since the variable "arrival_date_month" and "arrival_date_week_number" essentially carry the same information(and the latter one is actually with more granularity and therefore better), so it's safe for us to delete arrival_date_month in avoid of collenarity. Some other categorical variables are "meal" "customer_type" "reserved_room_type" "market_segment"

Introduction to some of the variables

Hotel: Resort Hotel or City Hotel

is_canceled: Value indicating if the booking was canceled

lead_time: Number of days that elapsed between the entering date of the booking into the PMS and the arrival date

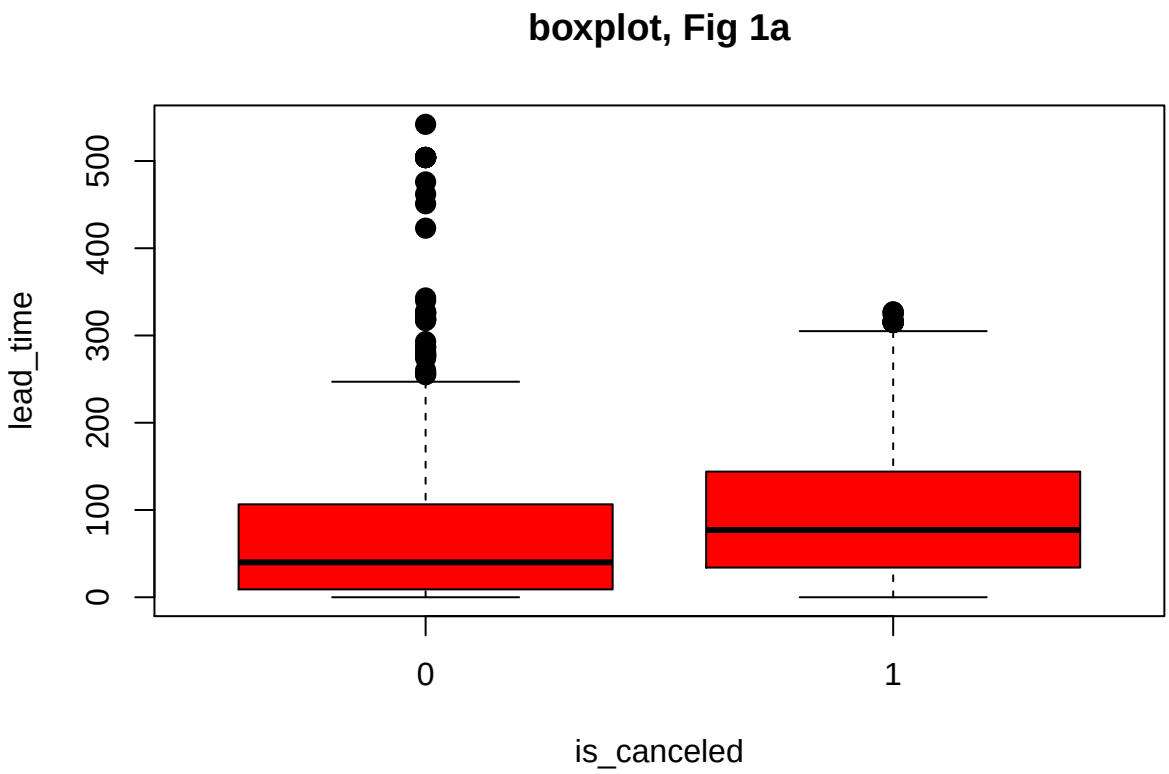
arrival_date_year: Year of arrival date

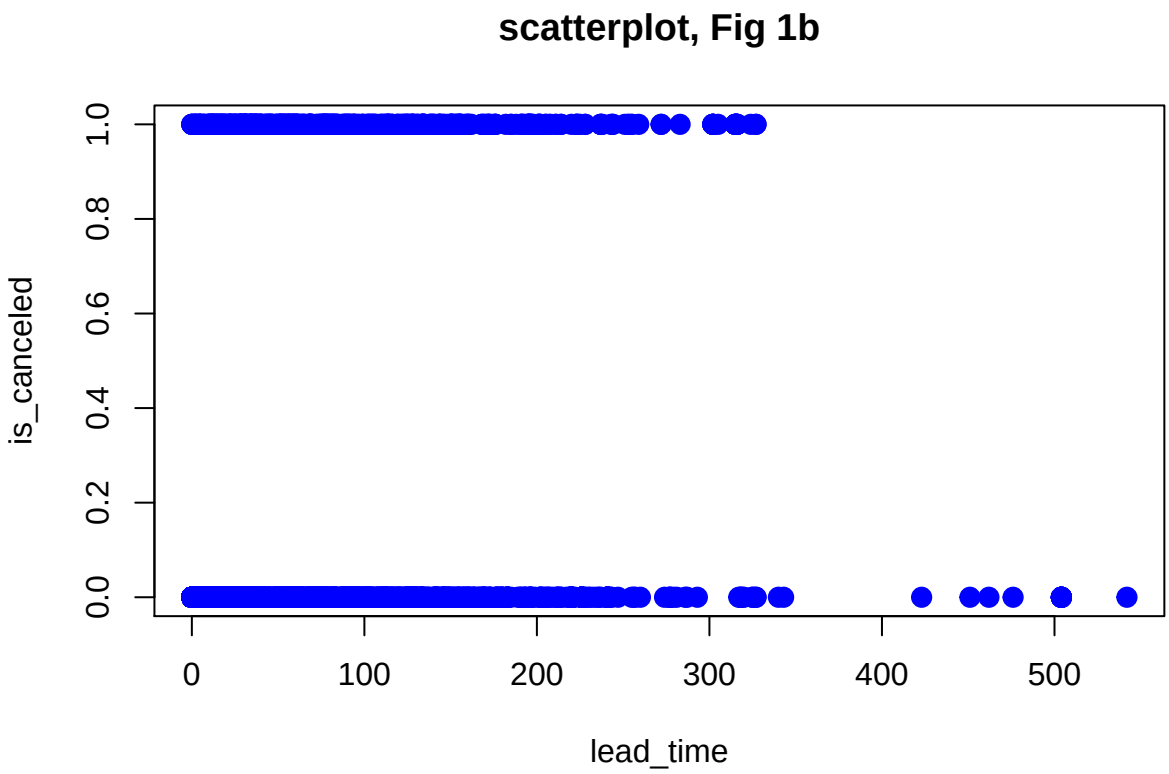
arrival_date_month: Month of arrival date

Section 2

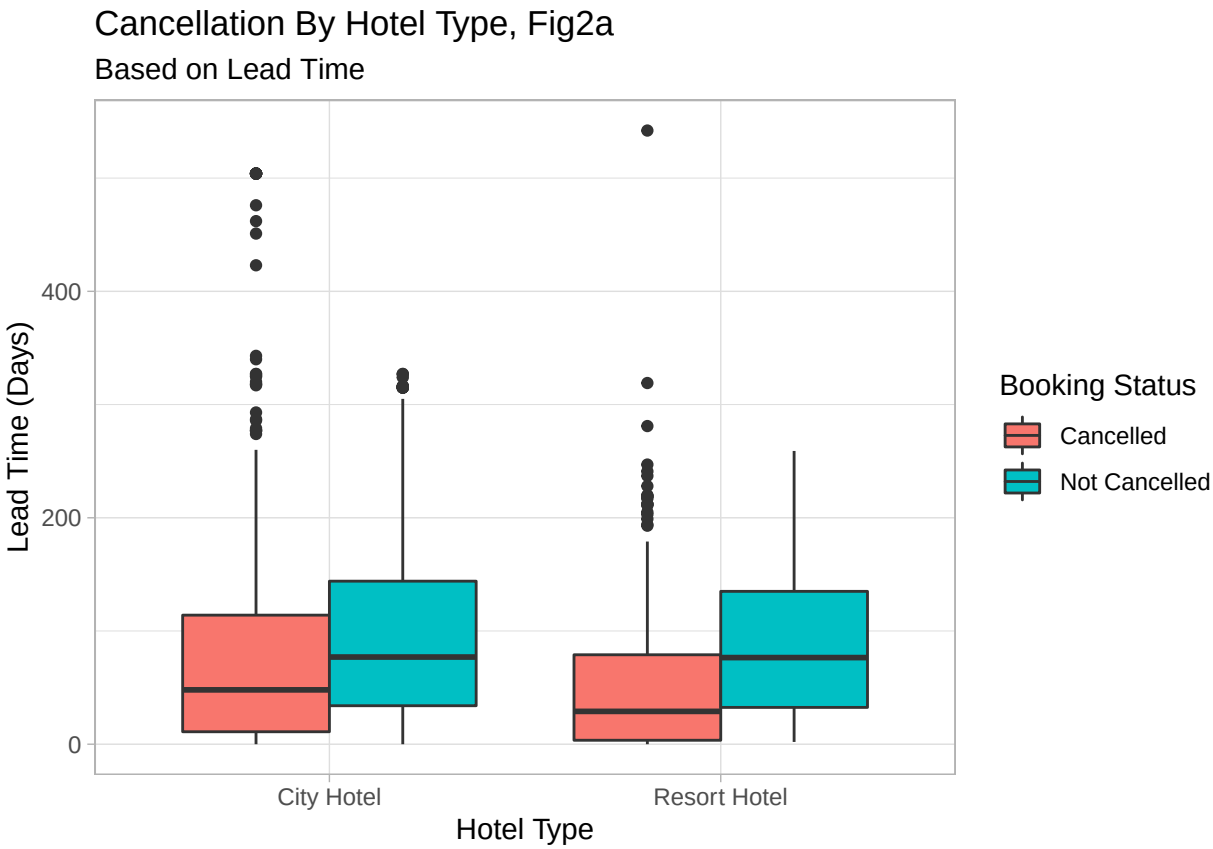
Plots

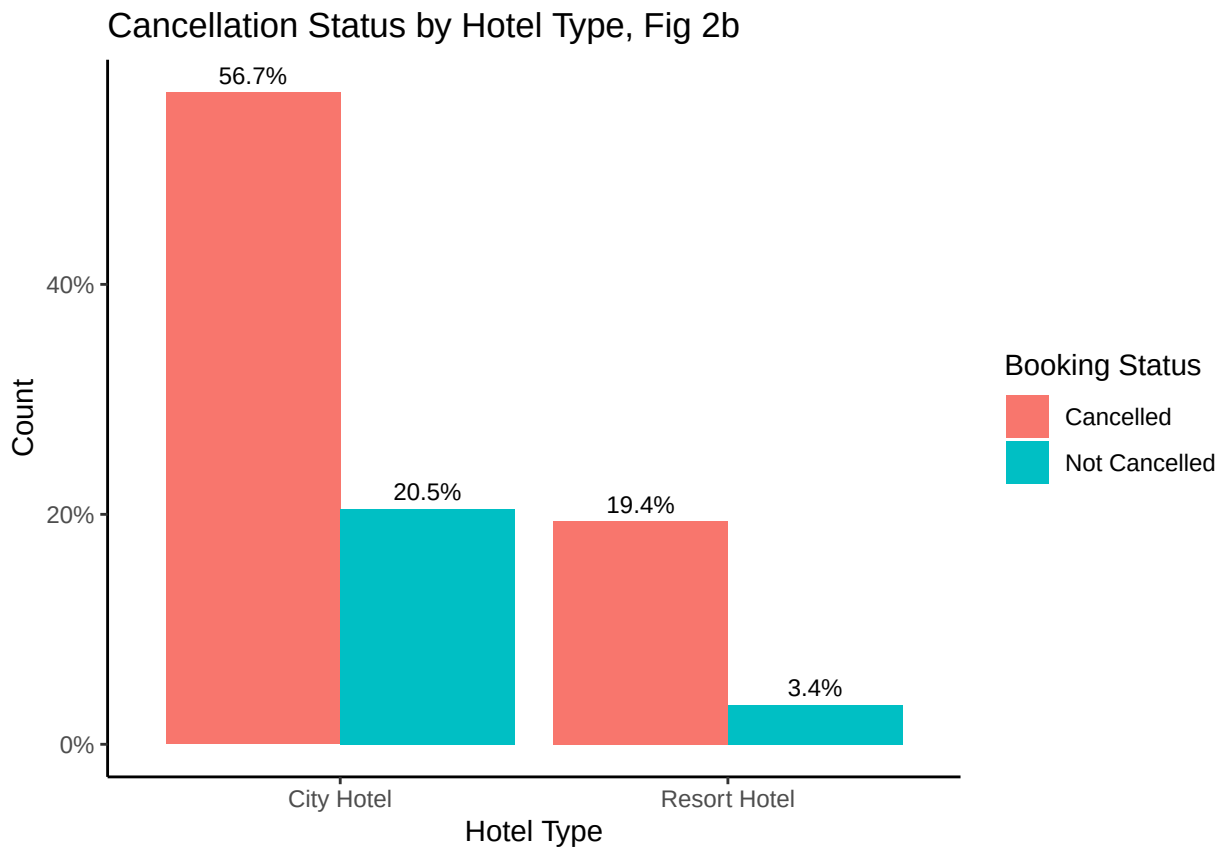
Now let's check the relation between is_canceled with some concerned explanatory variables


boxplot, Fig 1a
lead_time
0
100
200
300
400
500
0
1
is_canceled

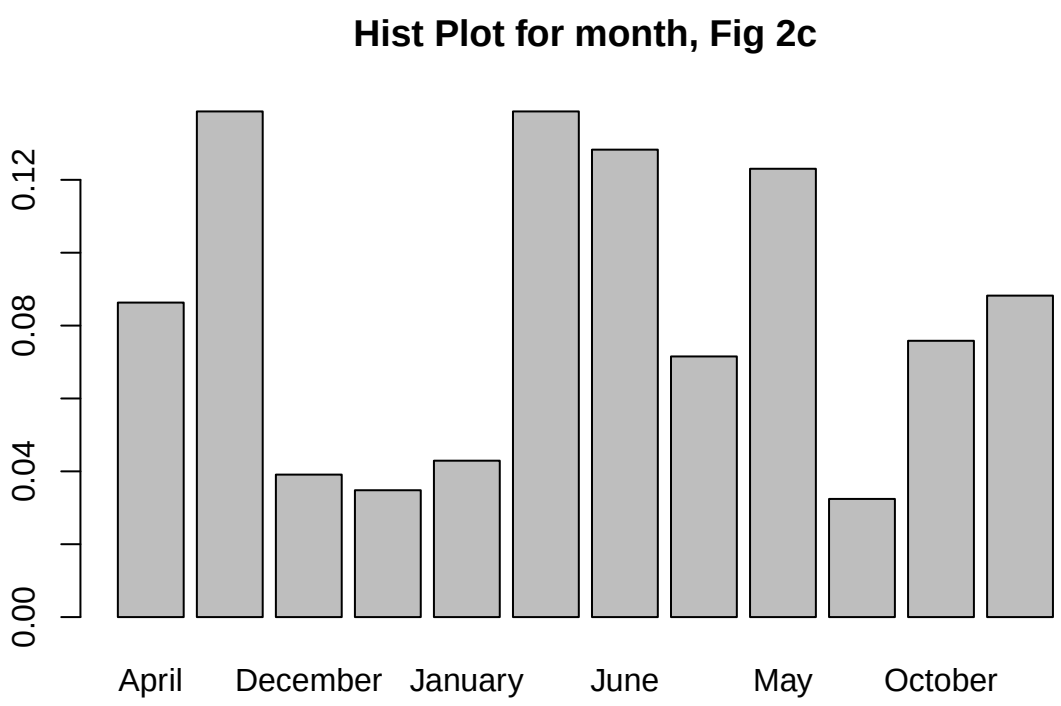


From Figure 1a and 1b, we can see that longer the lead time, bigger the possibility that the order will be cancelled, it makes sense since many people who order in advance eventually decide to cancel it due to some sudden erruptions.

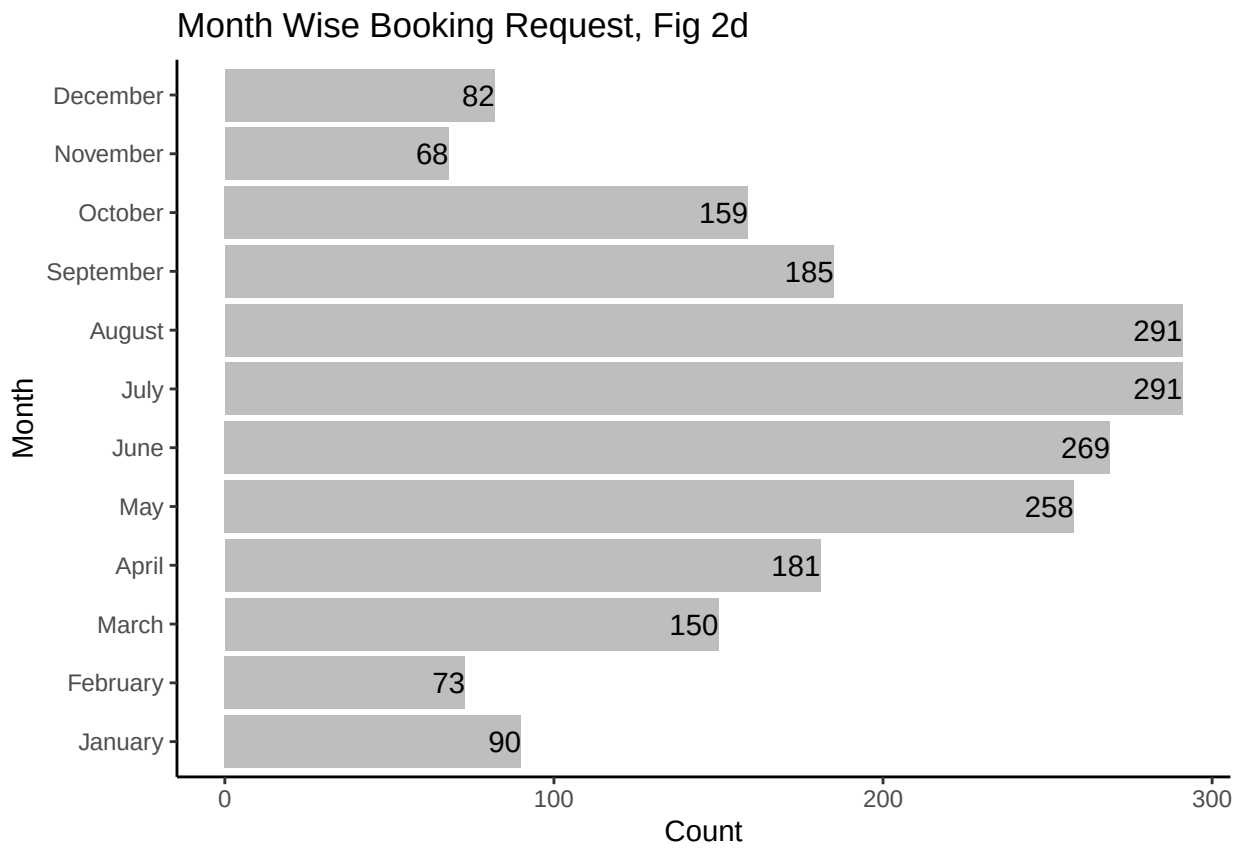

Cancellation By Hotel Type, Fig2a
Based on Lead Time
Lead Time (Days)
400
200
0
City Hotel
Resort Hotel
Hotel Type
Booking Status
Cancelled
Not Cancelled



From Figure 2a and 2b, we can see that longer the lead time, bigger the possibility that the order will be cancelled. Consider the ratio, this issue is more obvious on resort hotel than city hotel. Also, more people choose city hotel over resort hotel. It's reasonable since resort hotel is likely for family vacation and maybe once or twice a year while city hotel for businessman can have more frequent orders if they travel a lot. Besides, City hotel takes nearly three times of likelihood that order will be cancelled compared to resort hotel. An interesting fact is that resort hotel are more likely to be cancelled, this might be explained by much longer lead_time.



Let's make a better one hist plot with ggplot2 package, reordering the month into time order rather than alphebatical order.



Based on Fig 2c and 2d, we conclude that the majority of the ordering present a similar distribution to normal distribution, the peak take place in August and July, whereas lowest amount happened during February to November. This distribution makes sense since the hotels are in Europe, most American tourists went there for vacation with family, which normally happened in summer(August to June). So the weather factor plays major role here.

Section 3: Different models for classification

Firstly we do training test split with ration 0.2 as proportion of test set. After it we get training_set and test_set(find code in Rmd)

Section 3.1 Full logistic model

We would start from a logistic regression model with is_cancelled as response and all other variables as explanatory variable.

```
simple = glm(is_canceled~., data=train_set, family=binomial)
summary(simple)
```

Section 3.2 AIC/BIC selected logistic model

We further use a stepwise AIC/BIC method to find a better subsets of variables which achieve good AIC/BIC

```
stepA = step(simple, scope=list(upper=~., lower=~1))
summary(stepA)
```

It turns out that AIC method give us model glm(formula = is_canceled ~ hotel + lead_time + arrival_date_year + arrival_date_day_of_month + stays_in_weekend_nights + stays_in_week_nights + meal + market_segment + reserved_room_type + customer_type + adr + total_of_special_requests, family = binomial, data = train_set) Which still contains too many predicators, we now try using BIC as criteria.

```
n=dim(hotel_data)[1]
stepB = step(simple, scope=list(upper=~., lower=~1), trace = 0, k=log(n))
summary(stepB)
```

The BIC method gives a much better model in terms of complexity of model. Formula is is_canceled ~ hotel + lead_time + stays_in_week_nights + market_segment + customer_type + adr + total_of_special_requests. We now calculate the training error and testing error by using this model.

To simplify our job, I make my own function for calculating train_error and test_error.

```
#function that calculate train_error
train_error = function(model) {
  phat_test=model$fitted.values;
mypred = (phat_test>0.4)
tb = table(train_set$is_canceled, mypred)
return ((tb[1,2] +tb[2,1])/sum(tb))
}

#function that calculate test_error
test_error = function(model) {
  phat_test=predict(model,test_set);
mypred = (phat_test>0.4)
tb = table(test_set$is_canceled, mypred)
return ((tb[1,2] +tb[2,1])/sum(tb))
}
```

Calculate Training error and Testing error for simple AIC BIC model seperately

```
train_error(simple);
```

```
## [1] 0.2061069
```

```
test_error(simple)
```

```
## [1] 0.2097235
```

```
train_error(stepA);
```

```
## [1] 0.2166031
```

```
test_error(stepA)
```

```
## [1] 0.2087703
```

```
train_error(stepB);
```

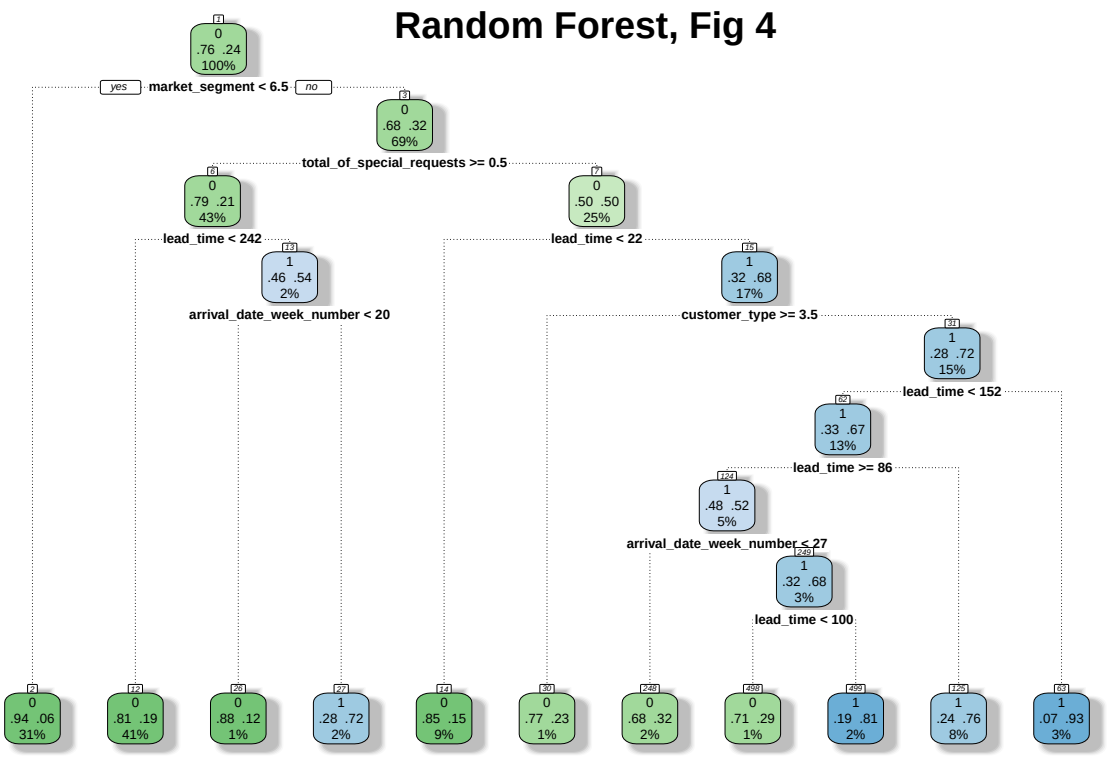
```
## [1] 0.2166031
```

```
test_error(stepB)
```

```
## [1] 0.2030505
```

Section 3.3 RandomForest

We now use random forest method to generate a classification tree and calculate its train and test error. Notice that unlike rpart(), random forest cannot predict binary respone, the predictted value is numerical and we choose 0.5 as threshold for classification.




```
##    train_permutation_table
##      FALSE TRUE
##   0   730   70
##   1    97  151
```

```
training_error_rf = (97+70)/(97+70+730+151)
training_error_rf
```

```
## [1] 0.1593511
```

```
##    test_permutation_table
##     FALSE TRUE
##   0   727   69
##   1    98  155
```

```
training_error_rf = (98+69)/(96+69+727+155)
training_error_rf
```

```
## [1] 0.1595033
```

It seems RandomForest has much lower training and testing rate.

Section 3.4 PCA and PCA regression

We firstly calculate PCA components and give brief explanations.

```
pca=princomp(train_set[,-2],cor=TRUE,scores=TRUE)
summary(pca)
```

```
## Importance of components:
##                            Comp.1    Comp.2     Comp.3     Comp.4     Comp.5
## Standard deviation     1.5380864 1.3493431 1.24694444 1.16164364 1.07523588
## Proportion of Variance 0.1391594 0.1071016 0.09146297 0.07937741 0.06800778
## Cumulative Proportion  0.1391594 0.2462610 0.33772394 0.41710135 0.48510913
##                            Comp.6     Comp.7     Comp.8     Comp.9    Comp.10
## Standard deviation     1.06857957 1.02661119 0.96555565 0.94043837 0.89804968
## Proportion of Variance 0.06716837 0.06199591 0.05484104 0.05202496 0.04744078
## Cumulative Proportion  0.55227750 0.61427341 0.66911446 0.72113942 0.76858020
##                           Comp.11    Comp.12    Comp.13    Comp.14    Comp.15
## Standard deviation     0.86986554 0.84189413 0.82428054 0.77398625 0.71287722
## Proportion of Variance 0.04450977 0.04169328 0.03996697 0.03523851 0.02989376
## Cumulative Proportion  0.81308996 0.85478324 0.89475021 0.92998872 0.95988248
##                           Comp.16    Comp.17
## Standard deviation     0.62848742 0.53572513
## Proportion of Variance 0.02323508 0.01688244
## Cumulative Proportion  0.98311756 1.00000000
```

```
round(pca$loadings[,1:5],2)
```

```
##                           Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## hotel                       0.30   0.13   0.01   0.18   0.42
## lead_time                  -0.10   0.02   0.35  -0.54  -0.08
## arrival_date_year           0.04  -0.55   0.29  -0.03  -0.02
## arrival_date_month         -0.11   0.36  -0.23  -0.04   0.13
## arrival_date_week_number    0.09   0.41  -0.40  -0.20  -0.20
```

```
## arrival_date_day_of_month   0.07  -0.08  -0.07   0.05  -0.32
## stays_in_weekend_nights     0.00   0.21   0.14  -0.29  -0.26
## stays_in_week_nights        0.06   0.24   0.06  -0.35  -0.09
## adults                      0.29  -0.17  -0.09  -0.18   0.06
## children                    0.33  -0.01   0.04  -0.05  -0.32
## babies                      0.02   0.02  -0.01  -0.22   0.56
## meal                       -0.31  -0.28  -0.36  -0.05  -0.19
## market_segment             -0.03  -0.32  -0.36  -0.28   0.01
## reserved_room_type          0.54   0.06   0.07   0.05   0.01
## customer_type              -0.14   0.18   0.51  -0.05  -0.08
## adr                         0.51  -0.11  -0.08  -0.05  -0.18
## total_of_special_requests   0.07  -0.16  -0.09  -0.51   0.28
```

It appears that the first component is concerned with hotel type(city or resort), second component is with the time of move-in(majorly year and month, no affection on arrival_date_day_of_month). The third is with customer type. The fourth component is concerned with lead_time. Fifth component is concerned with the number of babies.

Now we do regression over the first five PCA component and a big model for first 10 components.

```
table(P1mod$fitted.values>0.24,train_set[,"is_canceled"])
```

```
##
##           0   1
##   FALSE 588 185
##   TRUE  212  63
```

```
table(P5mod$fitted.values>0.51,train_set[,"is_canceled"])
```

```
##
##           0   1
##   FALSE 786 234
##   TRUE   14  14
```

```
table(P10mod$fitted.values>0.4,train_set[,"is_canceled"])
```

```
##
##           0   1
##   FALSE 710 147
##   TRUE   90 101
```

Unfortunately, I haven't find a way in R (I know Python could it though) to select proper P threshold for classification, so I brutally try some value here for approximation.

Section 4

Difference of Simple, AIC, BIC RandomForest model and PCA Regression

	simple	AIC	BIC	Random Forest	Pac Regression
Number of predicator	43	28	15	NA	10
training error	0.2061069	0.2166031	0.2166031	0.1593511	0.2259294
testing error	0.2097235	0.2087703	0.2030505	0.1595033	0.1853215

Conclusion: Between simple(contains all variables as predicators), AIC,BIC models we'll see that with the decrease of predicators number, the training error increase drastically and the testing error increase slightly, this corresponds to what we learned from class about complexity of models' impact to training error and testing error(The slight inrease of testing error might due to partition of train/test sample). Lastly, the randomforest provides the smallest training and testing error. Noticeably, The performance of random forest is way better than former three methods. Also the testing error is bigger than training error which corresponds to our suppose. The PCA regression model has better performance than logistic model yet slightlt worse performance than random forest. Overall, for classification purpose, random forest has way better performance of logistic regression and PAC regression, this might due to non-linearity essence of random forest. The essence of PCA regression is some kind of regulazation of our data. Above is a table of the comparison between different methods we use. An interesting issue from the table is that for simple AIC and BIC model that their testing error is slightly smaller than training error which against our knowledge. My explanation to this issue is that we need to sample bigger testing set or change the partition of our current split of train/test set.

Acknowledgement

Some dataset are retrieved from

https://www.kaggle.com/jessemostipak/hotel-booking-demand

Code for Fig 2a, 2b, 2d are modified from

https://www.kaggle.com/anshularya/eda-predictive-analysis-for-hotel-booking-data

ggplot2 code are modified from

https://ggplot2.tidyverse.org/reference/