Team 4 GBUS739 Project

Sai Jaswanth Kumar Kunku,Ravi Teja Adabala,Vaishnavi Putcha,Abhishek Godavarthi

2022-12-17

#Library

```
library(tidyverse)
library(dplyr)
library(skimr)
library(psych)
library(tidymodels)
library(vip)
library(discrim)
library(neuralnet)
library(nnet)
library(fastDummies)
library(rpart)
library(rpart.plot)
library(adabag)
library(ggpubr)
library(ROCR)
library(pROC)
library(randomForest)
library(caret)
library(skimr)
library(e1071)
library(rsample)
library(tm)
library(wordcloud2)
library(lsa)
df=read.csv("zomato.csv")
skim(df)
```

Table 1: Data summary

Name	df
Number of rows	51717
Number of columns	17
Column type frequency:	
character	16
numeric	1
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
url	0	1	180	538	0	51717	0
address	0	1	6	602	0	11495	0
name	0	1	2	287	0	8792	0
online_order	0	1	2	3	0	2	0
book_table	0	1	2	3	0	2	0
rate	0	1	0	6	7775	65	0
phone	0	1	0	32	1208	11875	0
location	0	1	0	29	21	94	0
rest_type	0	1	0	29	227	94	0
dish_liked	0	1	0	134	28078	5272	0
cuisines	0	1	0	86	45	2724	0
approx_cost.for.two.people.	0	1	0	5	346	71	0
reviews_list	0	1	2	1708987	0	22513	0
menu_item	0	1	2	24897	0	9098	0
listed_in.type.	0	1	5	18	0	7	0
listed_in.city.	0	1	3	21	0	30	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
votes	0	1	283.7	803.84	0	7	41	198	16832	

Data Preprocessing(Renaming,dropping and adding columns)

```
df<- df %>% select(-c("url","phone","address","location","menu_item"))

df <- df %>% rename(meal_type=listed_in.type.,
                    locality=listed_in.city.,
                    cost_for_two=approx_cost.for.two.people.,
                    table_booking=book_table,
                    rating=rate)

#New columns
df$total_cuisnes <- str_count(df$cuisines, ",")+1
df$total_dishes_liked <- str_count(df$dish_liked, ",")+1
df$total_reviews <- str_count(df$reviews_list, "\\)")+1

#df<- df %>% select(-c("reviews_list"))
#df$index <- 1:nrow(df)

#Converting characters to numerical
df$rating<-sub("\\/.*", "", df$rating)
df$rating <-as.numeric(df$rating)
df$votes <-as.numeric(df$votes)
df$cost_for_two <-as.numeric(df$cost_for_two)
df$total_cuisnes <-as.numeric(df$total_cuisnes)
df$total_dishes_liked <-as.numeric(df$total_dishes_liked)
#df$rest_type<-as.factor(df$rest_type)
df$online_order<-as.factor(df$online_order)
df$table_booking<-as.factor(df$table_booking)
```

```
df$meal_type<-as.factor(df$meal_type)
df$locality<-as.factor(df$locality)
df$rest_type<-as.factor(df$rest_type)

#df$menu<-str_replace_all(df$menu, "[^[:alnum:]]", " ")
```

```
skim(df)
```

Table 4: Data summary

Name	df
Number of rows	51717
Number of columns	15
Column type frequency:	
character	4
factor	5
numeric	6
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
name	0	1	2	287	0	8792	0
dish_liked	0	1	0	134	28078	5272	0
cuisines	0	1	0	86	45	2724	0
reviews_list	0	1	2	1708987	0	22513	0

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
online_order	0	1	FALSE	2	Yes: 30444, No: 21273
table_booking	0	1	FALSE	2	No: 45268, Yes: 6449
rest_type	0	1	FALSE	94	Qui: 19132, Cas: 10330, Caf: 3732, Del: 2604
meal_type	0	1	FALSE	7	Del: 25942, Din: 17779, Des: 3593, Caf: 1723
locality	0	1	FALSE	30	BTM: 3279, Kor: 2938, Kor: 2836, Kor: 2779

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
rating	10052	0.81	3.70	0.44	1.8	3.4	3.7	4	4.9	
votes	0	1.00	283.70	803.84	0.0	7.0	41.0	198	16832.0	
cost_for_two	7263	0.86	416.63	194.61	40.0	300.0	400.0	550	950.0	
total_cuisnes	0	1.00	2.45	1.27	1.0	2.0	2.0	3	8.0	
total_dishes_liked	0	1.00	3.03	2.64	1.0	1.0	1.0	7	7.0	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
total_reviews	0	1.00	30.71	82.91	1.0	2.0	5.0	14	1870.0	

Data Cleaning

```
#Cleaning the list data
#df$menu_item<-rm_between(df$menu_item, "[", "]", extract=TRUE)

#Merging cities falling under same name
df<- df %>%mutate(locality = recode(locality, 'Koramangala 4th Block' = 'Koramangala', 'Koramangala 5th

#get 1st element from rest_type
df=df %>% separate(rest_type, c("rest_type",NA),sep=",")
df<-df %>% filter(rest_type!="")
unique(df$rest_type)
```

```
##  [1] "Casual Dining"  "Cafe"           "Quick Bites"    "Delivery"
##  [5] "Mess"           "Dessert Parlor" "Bakery"         "Pub"
##  [9] "Takeaway"       "Fine Dining"    "Beverage Shop"  "Sweet Shop"
## [13] "Bar"            "Confectionery"  "Kiosk"          "Food Truck"
## [17] "Microbrewery"   "Lounge"         "Food Court"     "Dhaba"
## [21] "Club"           "Bhojanalya"     "Pop Up"
```

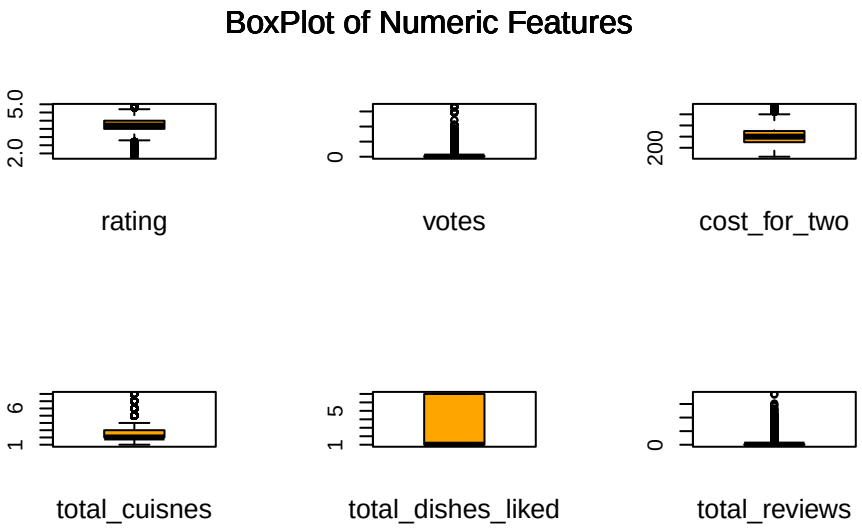
```
#Finding duplicates
sum(duplicated(df))
```

```
## [1] 3909
```

```
df<-unique(df)

#Removing NA values
df<-subset(df, !is.na(cuisines))
df<-subset(df, !is.na(rest_type))
#removing whitespaces
df<-df %>%  mutate(across(where(is.character), str_trim))
#Filling NA values
df <- df %>% group_by(locality) %>% mutate(rating =replace_na(rating, median(rating, na.rm = TRUE))) %>%
df <- df %>% group_by(locality) %>% mutate(cost_for_two =replace_na(cost_for_two, median(cost_for_two, r
```

Finding outliers

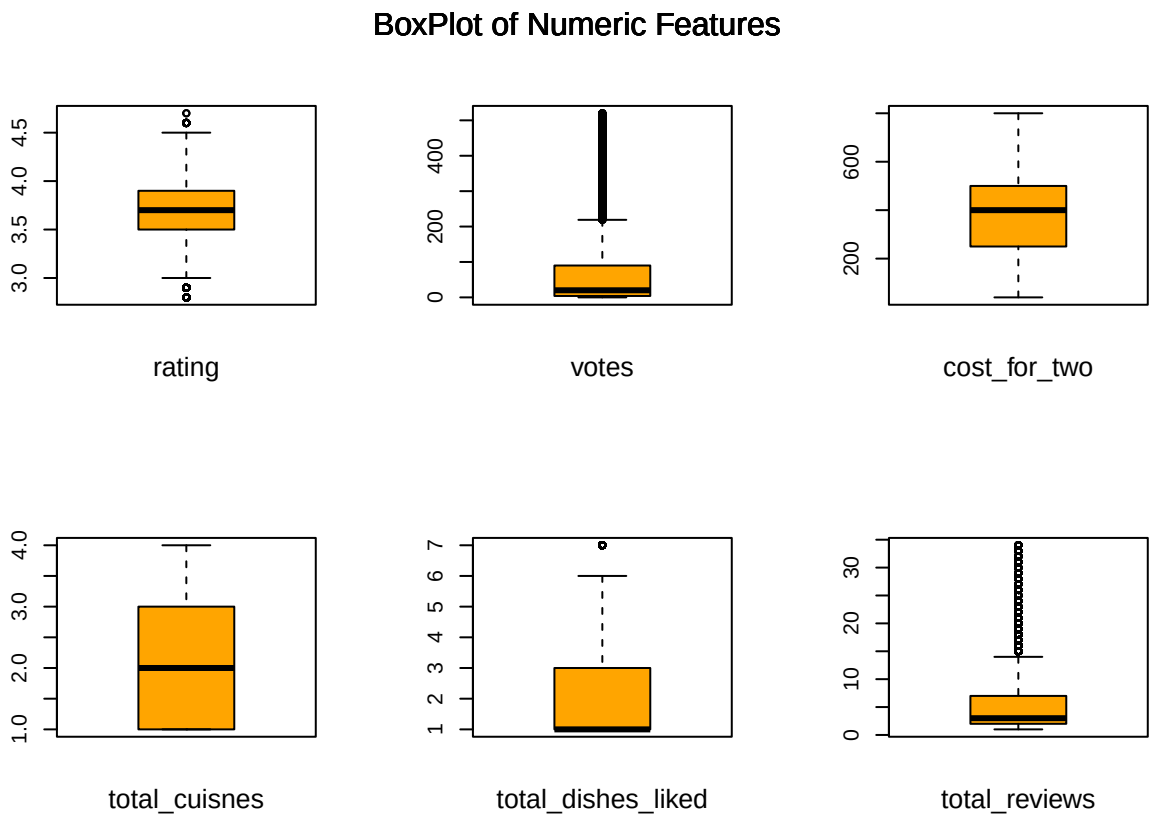
```
numeric_df<-Filter(is.numeric,df)
#EDA
boxrep = par(mfrow = c(2,3))
for ( i in 1:ncol(numeric_df) ) {
  boxplot(numeric_df[[i]],col='orange')
  mtext(names(numeric_df)[i], cex = 0.8, side = 1, line = 2)
  mtext("BoxPlot of Numeric Features", side = 3, line = -1.5, outer = TRUE)
}
```




```
#Outlier Removal
outlier1 <- boxplot(numeric_df$rating, plot=FALSE)$out
outlier2 <- boxplot(numeric_df$votes, plot=FALSE)$out
outlier3 <- boxplot(numeric_df$cost_for_two, plot=FALSE)$out
outlier4 <- boxplot(numeric_df$total_cuisnes, plot=FALSE)$out
outlier5 <- boxplot(numeric_df$total_reviews, plot=FALSE)$out
#outlier5 <- boxplot(numeric_df$total_dishes_liked, plot=FALSE)$out
```

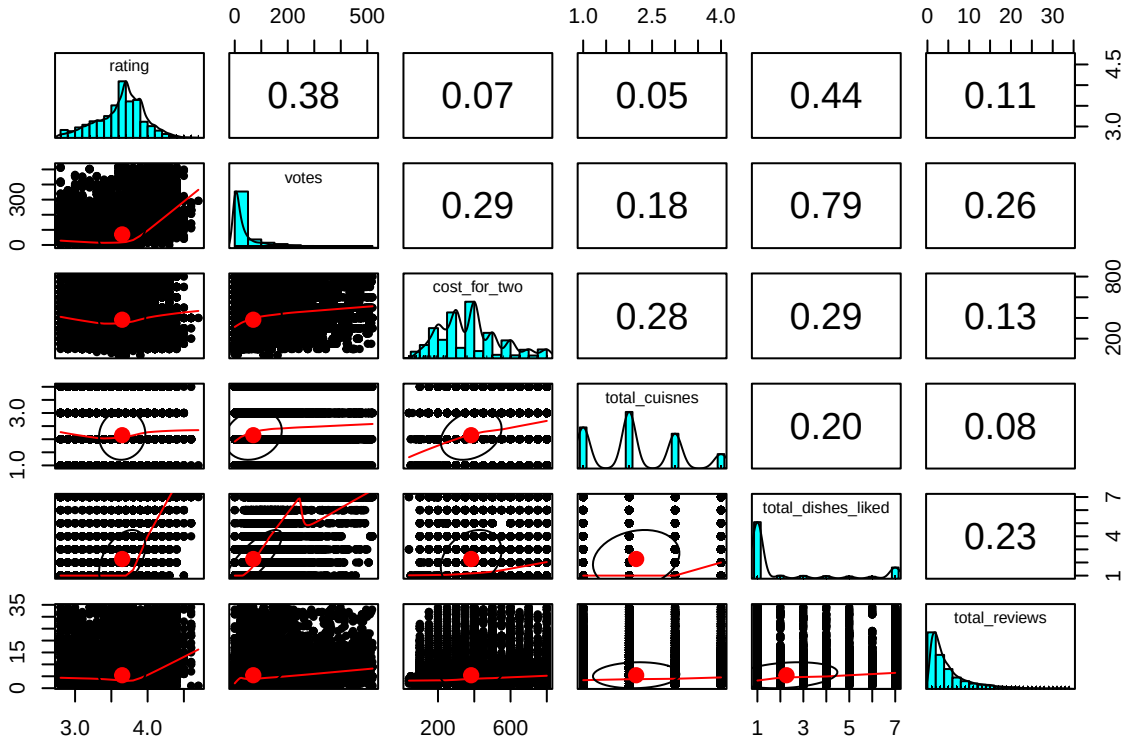
Outlier Removal

```
numeric_df<- as.data.frame(numeric_df[-which(numeric_df$rating %in% outlier1),])
numeric_df<- as.data.frame(numeric_df[-which(numeric_df$votes %in% outlier2),])
numeric_df<- as.data.frame(numeric_df[-which(numeric_df$cost_for_two %in% outlier3),])
numeric_df<- as.data.frame(numeric_df[-which(numeric_df$total_cuisnes %in% outlier4),])
numeric_df<- as.data.frame(numeric_df[-which(numeric_df$total_reviews %in% outlier5),])
#numeric_df<- as.data.frame(numeric_df[-which(numeric_df$total_dishes_liked %in% outlier5),])

boxrep = par(mfrow = c(2,3))
for ( i in 1:ncol(numeric_df) ) {
  boxplot(numeric_df[[i]],col='orange')
  mtext(names(numeric_df)[i], cex = 0.8, side = 1, line = 2)
  mtext("BoxPlot of Numeric Features", side = 3, line = -1.5, outer = TRUE)
}
```



Correlations

```
pairs.panels(numeric_df)
```

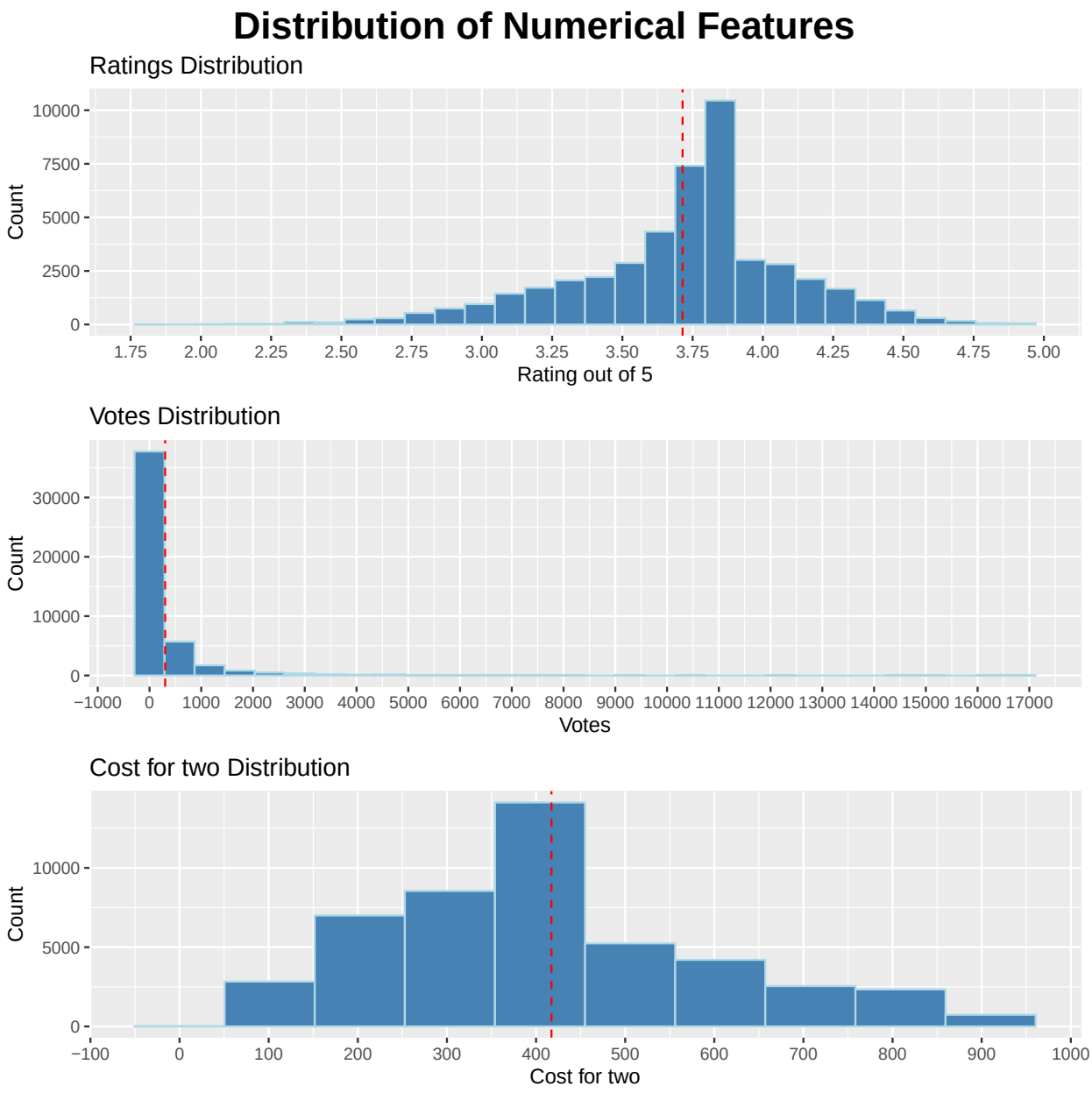


#Univariate Analysis for Caterogical Variables

```
gg1=df %>% group_by(online_order) %>% summarise(count=n()) %>% ggplot()+geom_bar(aes(x=online_order,y=co
gg2=df %>% group_by(table_booking) %>% summarise(count=n()) %>% ggplot()+geom_bar(aes(x=table_booking,y=

fig1=ggarrange(gg1, gg2,
          ncol = 2, nrow = 1)
annotate_figure(fig1,top = text_grob("Distribution of restaurants with online order and table booking ca
```

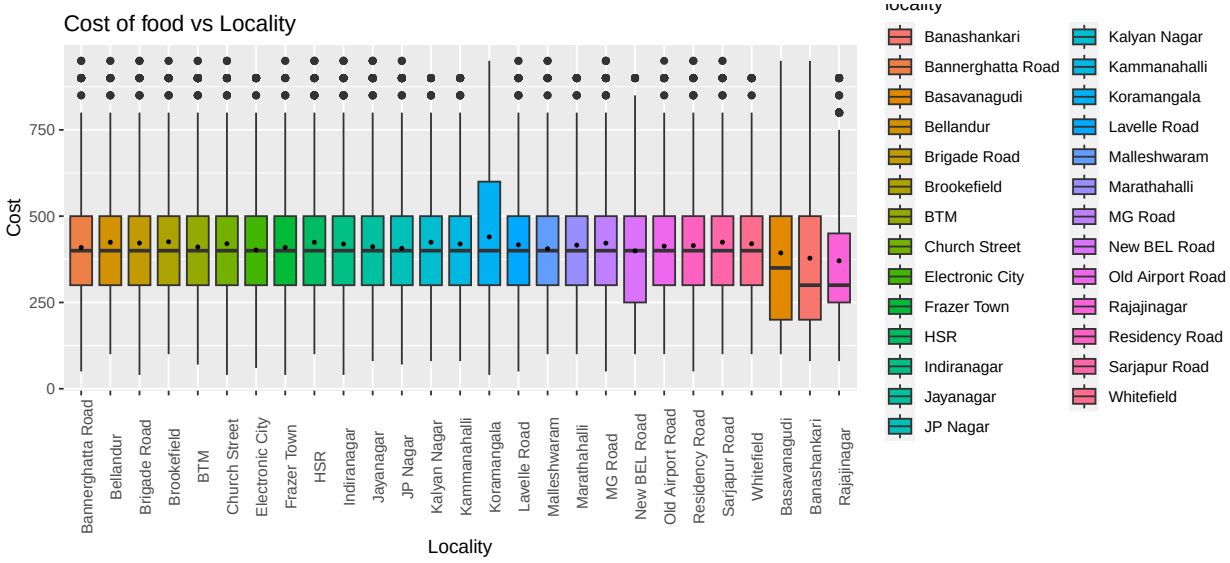
#Univariate Analysis for Numerical Variables

```
#Ratings count
g1=ggplot(df,aes(x = rating))+geom_histogram(bins =30,color='lightblue',fill='steelblue')+xlab("Rating
             col = "red",lty=2)

#Votes count
g2=ggplot(df,aes(x = votes))+geom_histogram(bins =30,color='lightblue',fill='steelblue')+xlab("Votes")+
             col = "red",lty=2)

#Cost for two count
g3=ggplot(df,aes(x = cost_for_two))+geom_histogram(bins =10,color='lightblue',fill='steelblue')+xlab("C
             col = "red",lty=2)

fig1=ggarrange(g1,g2,g3,
          ncol = 1, nrow = 3)

annotate_figure(fig1,top = text_grob("Distribution of Numerical Features",face = "bold", size = 20))
```

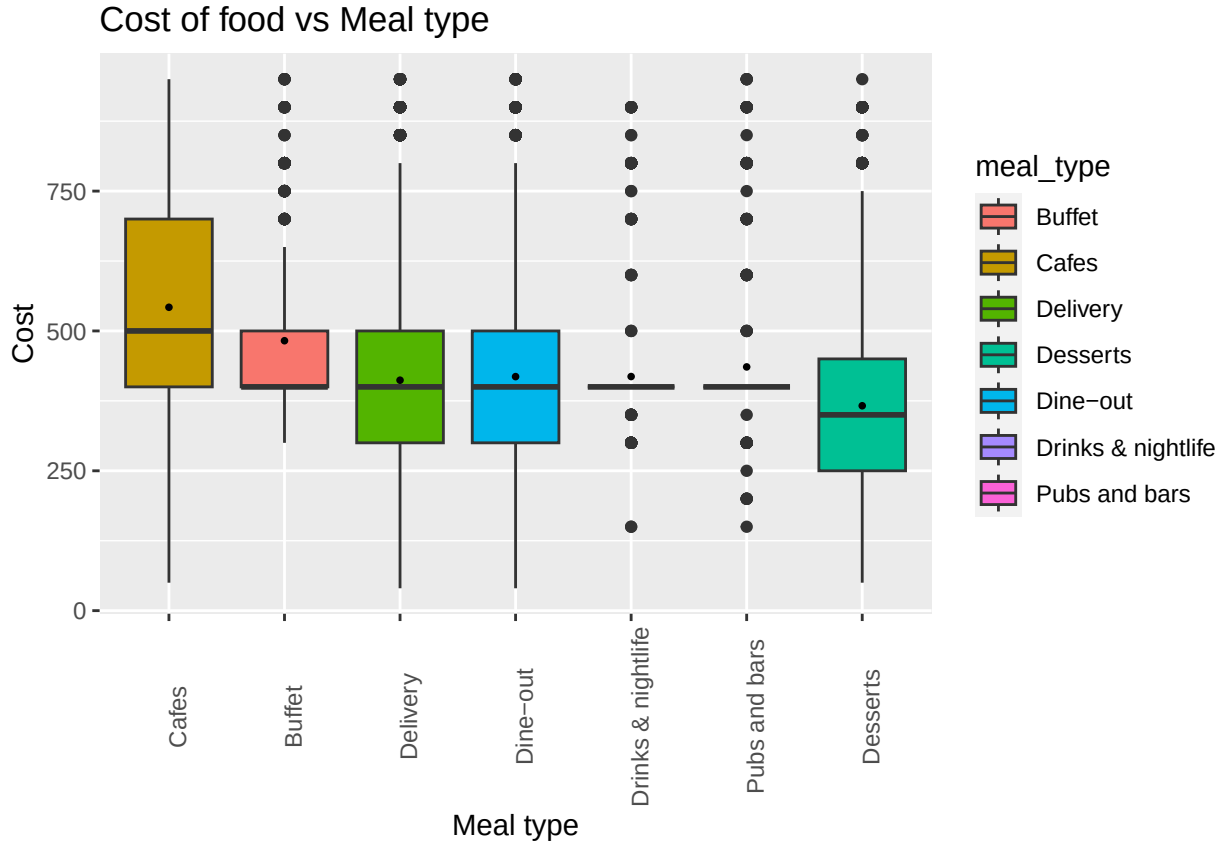


How the cost of food varies by locality and meal type?

```
 ggplot(df,aes(x=reorder(locality, -cost_for_two, FUN = median),
                          y=cost_for_two,
                          fill=locality)) +
geom_boxplot()+ stat_summary(fun=mean, colour="black", geom="point",shape=16,
                             size=1, show.legend=FALSE) +
  labs(title = "Cost of food vs Locality",
       y = "Cost",
       x = "Locality")+theme(axis.text.x = element_text(angle = 90))
```

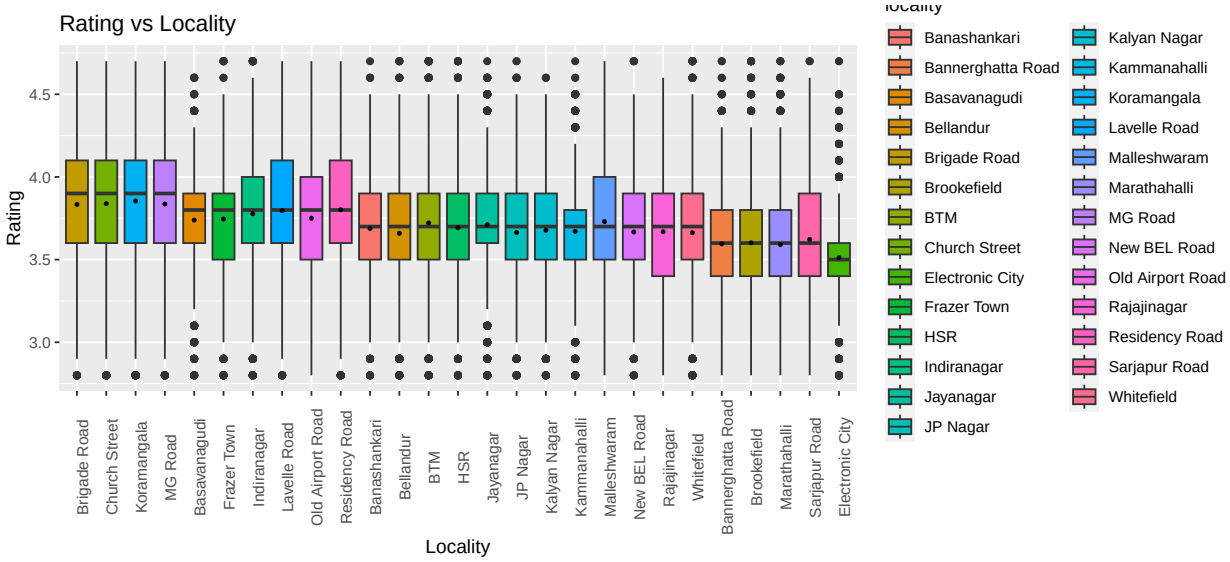



```
ggplot(df,aes(x=reorder(meal_type, -cost_for_two, FUN = median),y=cost_for_two,fill=meal_type)) +
geom_boxplot()+ stat_summary(fun=mean, colour="black", geom="point",shape=16,
                             size=1, show.legend=FALSE) +
  labs(title = "Cost of food vs Meal type",
       y = "Cost",
       x = "Meal type")+theme(axis.text.x = element_text(angle = 90))
```



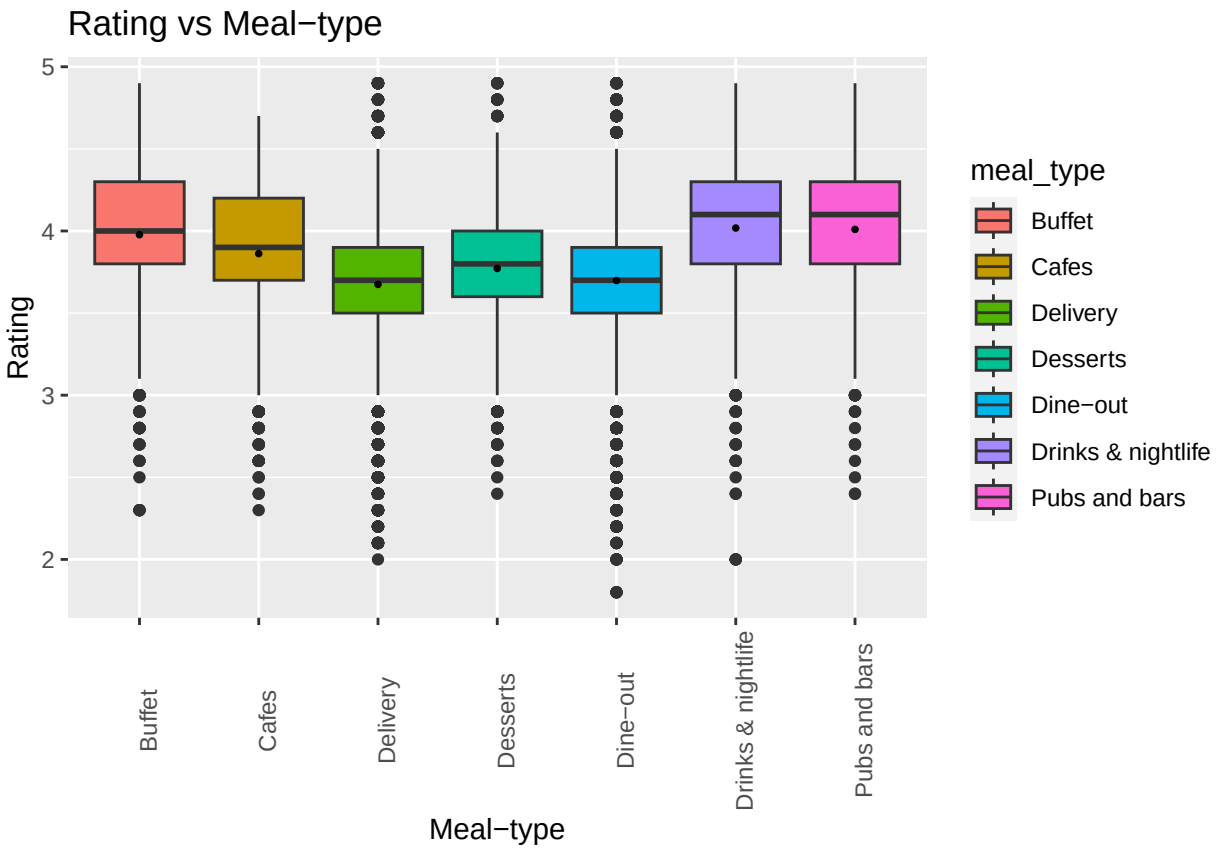
Rating vs Locality

```
#Removal of Outliers
  x<-df
  outliers <- boxplot(x$rating, plot=FALSE)$out
  x<- as.data.frame(x[-which(x$rating %in% outliers),])

ggplot(x,aes(x=reorder(locality, -rating, FUN = median),
                          y=rating,
                          fill=locality)) +
geom_boxplot()+ stat_summary(fun=mean, colour="black", geom="point",shape=16,
                             size=1, show.legend=FALSE) +
  labs(title = "Rating vs Locality",
        y = "Rating",
        x = "Locality")+theme(axis.text.x = element_text(angle = 90))
```



Rating to Meal Type

```
ggplot(df,aes(x=meal_type,y=rating,fill=meal_type)) +
geom_boxplot()+ stat_summary(fun=mean, colour="black", geom="point",shape=16,
                             size=1, show.legend=FALSE) +
  labs(title = "Rating vs Meal-type",
        y = "Rating",
        x = "Meal-type")+theme(axis.text.x = element_text(angle = 90))
```

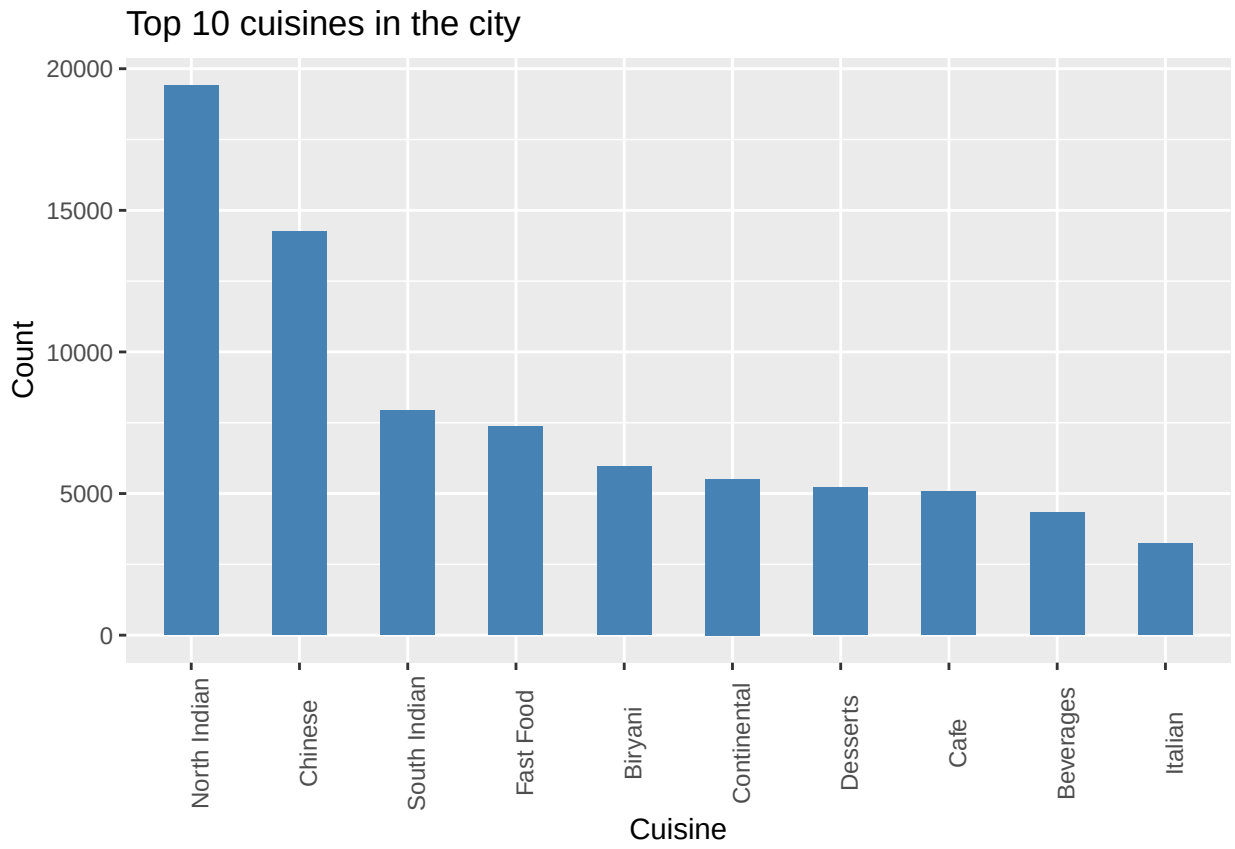


Which localities have highest online orders

```
ggplot(df, aes(locality, fill = online_order)) +
  geom_bar(position = "fill") +
  scale_y_continuous(labels = scales::percent)+
scale_fill_discrete(breaks=c('Yes', 'No'))+
  labs(title = "Online order vs locality",
       y = "Online order",
       x = "locality")+theme(axis.text.x = element_text(angle = 90))
```

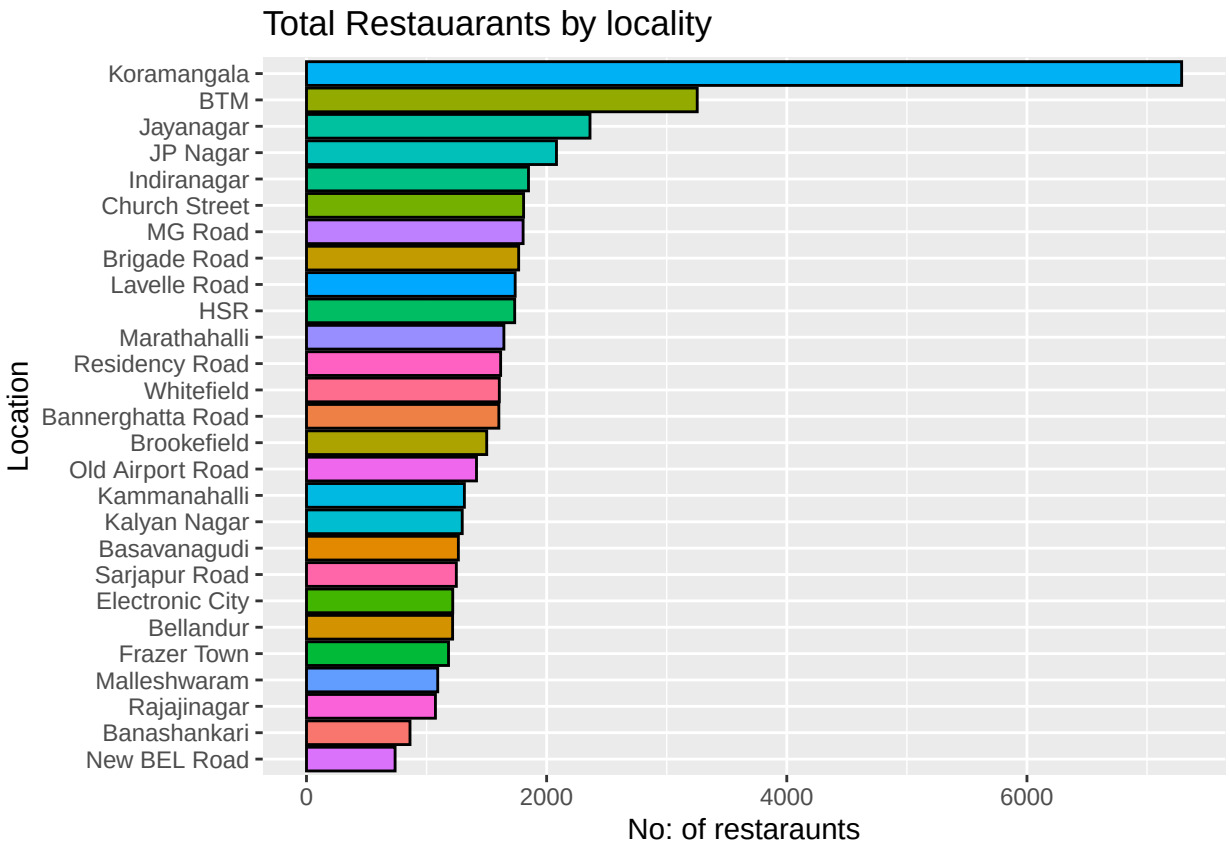
#Which cuisines are most preferred by Banglore people

```
cusines_city<-df %>% select(locality,cuisines) %>%
    mutate(cuisines = strsplit(as.character(cuisines), ",")) %>%
    unnest(cuisines)

 cusines_city$cuisines<-str_trim(cusines_city$cuisines)

cusines_city<- cusines_city %>% group_by(cuisines) %>% summarise(count=n()) %>%  arrange(desc(count))

cusines_city <- cusines_city[with(cusines_city,order(-count)),]
cusines_city <- cusines_city[1:10,]

ggplot(data=cusines_city, aes(x=reorder(cuisines,-count), y=count)) +
  geom_bar(stat="identity", width=0.5,fill="steelblue")+
  labs(title = "Top 10 cuisines in the city",
        y = "Count",
        x = "Cuisine")+theme(axis.text.x = element_text(angle = 90))
```



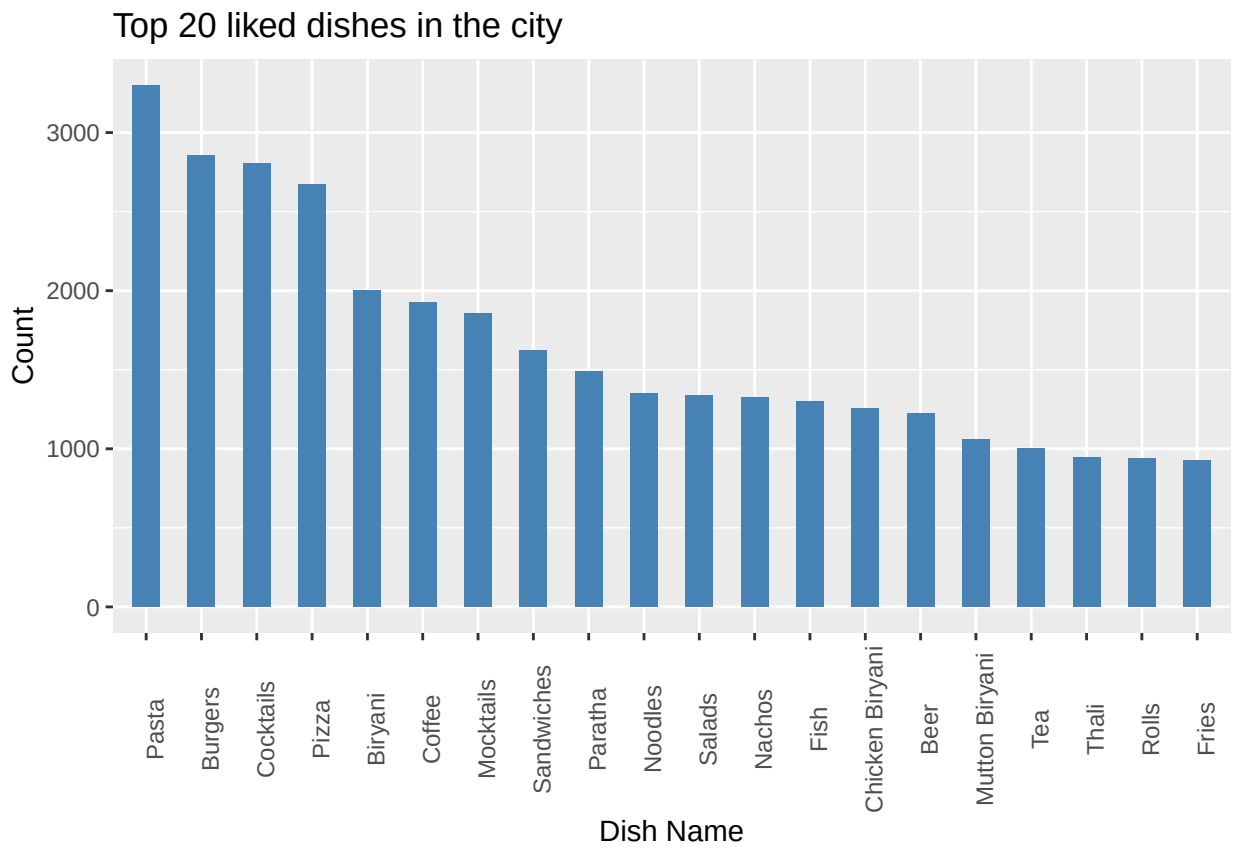
Which location has highest no of restaurants

```
restaunts_count<-df %>% group_by(locality) %>% summarise(count=n())

ggplot(restaunts_count,aes(x=reorder(locality,count),y=count,fill=locality))+
geom_bar(stat = "identity",color="black")+
labs(x="Location",y="No: of restaraunts",
     title="Total Restauarants by locality")+coord_flip()+theme(legend.position = "none")
```



What are the most popular food preferences in the city

```
dishes_list<-df %>% select(locality,dish_liked) %>%
    mutate(dish_liked = strsplit(as.character(dish_liked), ",")) %>%
    unnest(dish_liked)

 dishes_list$dish_liked<-str_trim(dishes_list$dish_liked)

dishes_list<- dishes_list %>% group_by(dish_liked) %>% summarise(count=n()) %>%  arrange(desc(count))

dishes_list <- dishes_list[with(dishes_list,order(-count)),]
dishes_list <- dishes_list[1:20,]

ggplot(data=dishes_list, aes(x=reorder(dish_liked,-count), y=count)) +
  geom_bar(stat="identity", width=0.5,fill="steelblue")+
  labs(title = "Top 20 liked dishes in the city",
        y = "Count",
        x = "Dish Name")+theme(axis.text.x = element_text(angle = 90))
```




```
model_df<-df %>% select(online_order,table_booking,meal_type,total_cuisnes,total_dishes_liked,cost_for_
                        total_reviews)
skim(model_df)
```

Table 8: Data summary

Name	model_df
Number of rows	47581
Number of columns	9
Column type frequency:	
factor	3
numeric	6
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
online_order	0	1	FALSE	2	Yes: 28122, No: 19459
table_booking	0	1	FALSE	2	No: 41271, Yes: 6310
meal_type	0	1	FALSE	7	Del: 23665, Din: 16344, Des: 3310, Caf: 1657

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
total_cuisnes	0	1	2.47	1.28	1.0	2.0	2.0	3	8.0	
total_dishes_liked	0	1	3.12	2.67	1.0	1.0	1.0	7	7.0	
cost_for_two	0	1	417.28	181.52	40.0	300.0	400.0	500	950.0	
votes	0	1	302.26	833.86	0.0	8.0	46.0	213	16832.0	
rating	0	1	3.71	0.40	1.8	3.5	3.7	4	4.9	
total_reviews	0	1	32.86	85.64	1.0	2.0	5.0	15	1870.0	

Linear regression with only numerical variables

```
set.seed(314)
zomato_split <- initial_split(numeric_df, prop = 0.60, strata = cost_for_two)
zomato_training <- zomato_split %>%  training()
zomato_test <- zomato_split %>% testing()

## Step 2. Feature Engineering
zomato_recipe <- recipe(cost_for_two ~ ., data = zomato_training)

## Step 3. Specify a Model
lm_model <- linear_reg() %>% set_engine('lm') %>% set_mode('regression')

# View object properties
lm_model
```

```
## Linear Regression Model Specification (regression)
##
## Computational engine: lm
```

```
lm_sum = lm(cost_for_two ~ ., data = zomato_training) #Create a linear regression with two variables
summary(lm_sum)
```

```
##
## Call:
## lm(formula = cost_for_two ~ ., data = zomato_training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -431.60 -111.94  -24.72   91.74  508.23
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        370.88704   13.70470  27.063  < 2e-16 ***
## rating             -33.53635    3.78131  -8.869  < 2e-16 ***
## votes                0.23167    0.01636  14.158  < 2e-16 ***
## total_cuisnes       40.13791    1.17569  34.140  < 2e-16 ***
## total_dishes_liked   9.95677    0.81376  12.235  < 2e-16 ***
## total_reviews        1.57823    0.20439   7.721  1.2e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 154.6 on 20362 degrees of freedom
## Multiple R-squared:  0.1454, Adjusted R-squared:  0.1452
## F-statistic: 693.1 on 5 and 20362 DF,  p-value: < 2.2e-16
```

```
## Step 4. Create a Workflow
zomato_workflow <- workflow() %>% add_model(lm_model) %>% add_recipe(zomato_recipe)


## Step 5. Execute the Workflow
zomato_fit <- zomato_workflow %>% last_fit(split = zomato_split)

# Obtain performance metrics on test data
zomato_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric .estimator .estimate .config
##   <chr>   <chr>          <dbl> <chr>
## 1 rmse    standard     155.    Preprocessor1_Model1
## 2 rsq     standard       0.152 Preprocessor1_Model1
```

Linear regression with numerical and categorical variables along with Feature Engineering

```
set.seed(314)
zomato_split <- initial_split(model_df, prop = 0.60, strata = cost_for_two)
zomato_training <- zomato_split %>%  training()
zomato_test <- zomato_split %>% testing()

## Step 2. Feature Engineering
zomato_recipe <- recipe(cost_for_two ~ ., data = zomato_training) %>%
                step_YeoJohnson(all_numeric(), -all_outcomes()) %>%
                step_normalize(all_numeric(), -all_outcomes()) %>%
                step_dummy(all_nominal(), - all_outcomes())

## Step 3. Specify a Model
lm_model <- linear_reg() %>% set_engine('lm') %>% set_mode('regression')

lm_sum2 = lm(cost_for_two ~ ., data = zomato_training) #Create a linear regression with two variables
summary(lm_sum2)
```

```
##
## Call:
## lm(formula = cost_for_two ~ ., data = zomato_training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -485.61 -117.91  -32.88   92.80  621.20
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 462.474117  12.960513  35.683  < 2e-16 ***
## online_orderYes              24.572377   2.143926  11.461  < 2e-16 ***
## table_bookingYes              7.744258   3.561499   2.174 0.029680 *
## meal_typeCafes               86.410775   9.121602   9.473  < 2e-16 ***
## meal_typeDelivery           -10.645652   7.656410  -1.390 0.164411
## meal_typeDesserts           -29.207031   8.393973  -3.480 0.000503 ***
## meal_typeDine-out            -5.883856   7.618215  -0.772 0.439919
```

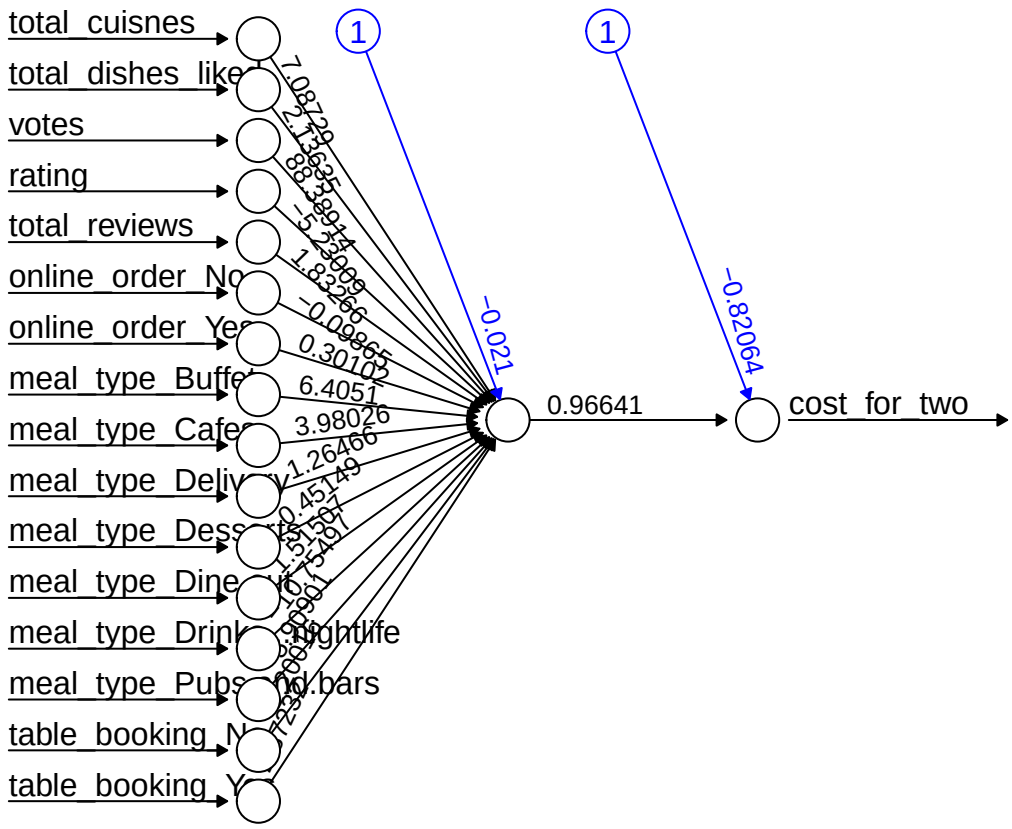
```
## meal_typeDrinks & nightlife -45.906311   9.854960  -4.658  3.2e-06 ***
## meal_typePubs and bars      -37.853874  10.856445  -3.487 0.000490 ***
## total_cuisnes                26.426810   0.819967  32.229  < 2e-16 ***
## total_dishes_liked           21.480912   0.494240  43.463  < 2e-16 ***
## votes                         0.001764   0.001470   1.200 0.230185
## rating                      -50.187750   3.005530 -16.698  < 2e-16 ***
## total_reviews                 0.011158   0.012537   0.890 0.373481
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 164.4 on 28532 degrees of freedom
## Multiple R-squared:  0.1777, Adjusted R-squared:  0.1773
## F-statistic: 474.3 on 13 and 28532 DF,  p-value: < 2.2e-16
```

```
## Step 4. Create a Workflow
zomato_workflow <- workflow() %>% add_model(lm_model) %>% add_recipe(zomato_recipe)


## Step 5. Execute the Workflow
zomato_fit <- zomato_workflow %>% last_fit(split = zomato_split)

# Obtain performance metrics on test data
zomato_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric .estimator .estimate .config
##   <chr>   <chr>          <dbl> <chr>
## 1 rmse    standard     164.    Preprocessor1_Model1
## 2 rsq     standard       0.188 Preprocessor1_Model1
```

Neural Network

```
zomato_training <- dummy_cols(zomato_training, select_columns = c('online_order','rest_type','meal_type
zomato_test <- dummy_cols(zomato_test, select_columns = c('online_order','rest_type','meal_type','locali

my_metrics <-metric_set(accuracy, sens, spec, f_meas, roc_auc)

zomato_training_params <- preProcess(zomato_training, method=c("range"))
zomato_training_normalized <- predict(zomato_training_params, zomato_training)
zomato_test_params <- preProcess(zomato_test, method=c("range"))
zomato_test_normalized <- predict(zomato_test_params, zomato_test)

names(zomato_training_normalized) <- make.names(names(zomato_training))
names(zomato_test_normalized) <- make.names(names(zomato_test))

## single hidden layer Neural Network with 1 nodes
set.seed(10)
nn1 <- neuralnet(cost_for_two~ ., data = zomato_training_normalized,linear.output = F,hidden = 1)
plot(nn1,rep="best")
```




```
test_pred <- neuralnet::compute(nn1, zomato_test_normalized)$net.result

rsq <- function (x, y) cor(x, y) ^ 2
r2<-rsq(zomato_test_normalized$cost_for_two, test_pred)
rmse<-sqrt(mean((zomato_test_normalized$cost_for_two-test_pred)^2))

print(paste("RMSE:",round(rmse,3),"R2:",round(r2,3)))
```

```
## [1] "RMSE: 0.181 R2: 0.182"
```

Classification:Factoring the target Variable & Data Splitting

```
model_df <- model_df %>%
  mutate(rating = case_when(
    rating>=3.5 ~ "High",
    TRUE ~ "Low"))

model_df$rating<-factor(model_df$rating)

set.seed(1)
#Test/Train Split
zomato_split <- initial_split(model_df, prop = 0.70,strata = rating)
zomato_training <- zomato_split %>% training()
zomato_test <- zomato_split %>% testing()

zomato_recipe <- recipe(rating ~ ., data = zomato_training) %>%
```

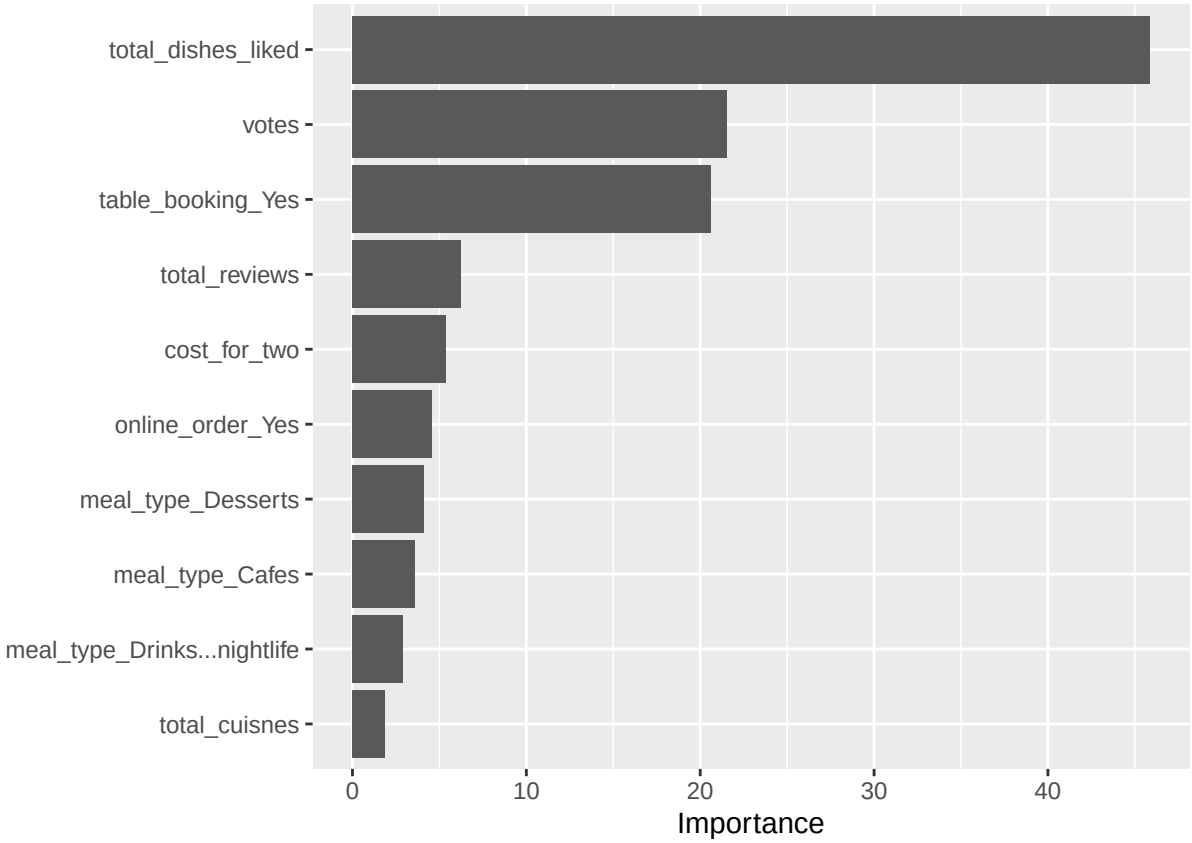
```
                  step_YeoJohnson(all_numeric(), -all_outcomes()) %>%
                  step_normalize(all_numeric(), -all_outcomes()) %>%
                  step_dummy(all_nominal(), -all_outcomes())

zomato_recipe %>%
  prep(training = zomato_training) %>%
  bake(new_data = NULL)
```

```
## # A tibble: 33,306 x 14
##    total~1 total~2 cost_~3  votes total~4 rating onlin~5 table~6 meal_~7 meal_~8
##      <dbl>   <dbl>   <dbl>  <dbl>   <dbl> <fct>    <int>   <int>   <int>   <int>
##  1   0.611  1.31     1.82   1.31    0.607 High         1       1       0       0
##  2   0.611  1.31     1.82   1.32    0.738 High         1       0       0       0
##  3   0.611  1.31     1.82   1.39    1.08  High         1       0       0       0
##  4  -0.197 -0.858   -0.568  0.323   1.34  High         0       0       0       0
##  5  -0.197  0.0910   1.03   0.605  -0.548 High         0       0       0       0
##  6  -1.43   1.31     1.03   0.851  -0.908 High         1       0       0       0
##  7   1.20  -0.858    1.82  -0.660  -0.548 High         0       0       0       0
##  8   0.611  1.31     1.03   1.89    1.41  High         1       1       1       0
##  9   0.611  1.31     1.03   0.560   0.494 High         1       1       1       0
## 10   0.611  1.31     1.82   1.39    1.16  High         1       0       1       0
## # ... with 33,296 more rows, 4 more variables: meal_type_Desserts <int>,
## #   meal_type_Dine.out <int>, meal_type_Drinks...nightlife <int>,
## #   meal_type_Pubs.and.bars <int>, and abbreviated variable names
## #   1: total_cuisnes, 2: total_dishes_liked, 3: cost_for_two, 4: total_reviews,
## #   5: online_order_Yes, 6: table_booking_Yes, 7: meal_type_Cafes,
## #   8: meal_type_Delivery
```

```
zomato_folds <- vfold_cv(zomato_training, v = 5)

#Metrics
my_metrics <-metric_set(accuracy, sens, spec, f_meas, roc_auc)
```

Model 1:Logistic Regression

```
#Model Spec
logistic_model <- logistic_reg() %>%
                  set_engine('glm') %>%
                  set_mode('classification')

zomato_wf <- workflow() %>%
             add_model(logistic_model) %>%
             add_recipe(zomato_recipe)

zomato_logistic_fit <- zomato_wf %>% fit(data = zomato_training)

zomato_trained_model <- zomato_logistic_fit %>% extract_fit_parsnip()

vip(zomato_trained_model)
```




```
last_fit_model <- zomato_wf %>% last_fit(split = zomato_split,metrics = my_metrics)

metrics<-last_fit_model %>% collect_metrics()

metrics
```

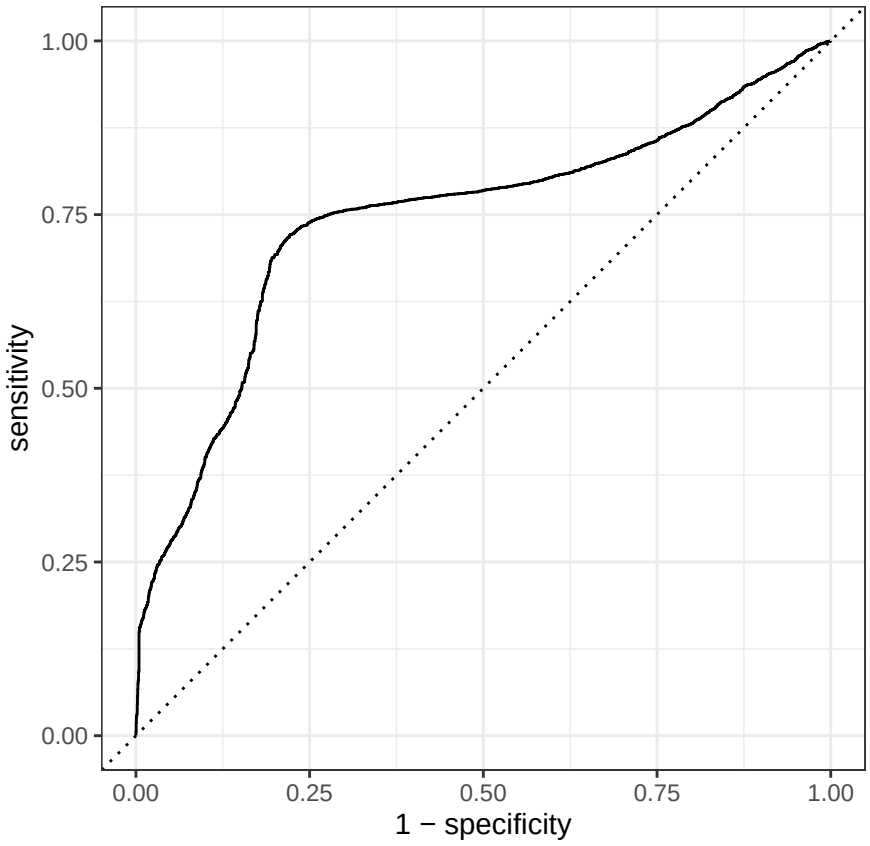
```
## # A tibble: 5 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary        0.770  Preprocessor1_Model1
## 2 sens     binary        0.974  Preprocessor1_Model1
## 3 spec     binary        0.0489 Preprocessor1_Model1
## 4 f_meas   binary        0.869  Preprocessor1_Model1
## 5 roc_auc  binary        0.742  Preprocessor1_Model1
```

```
last_fit_results <- last_fit_model %>%  collect_predictions()

last_fit_results
```

```
## # A tibble: 14,275 x 7
##    id               .pred_class  .row .pred_High .pred_Low rating .config
##    <chr>            <fct>       <int>      <dbl>     <dbl> <fct>  <chr>
##  1 train/test split High            9      0.926   0.0740  High   Preprocessor1~
##  2 train/test split High           10      0.932   0.0678  High   Preprocessor1~
##  3 train/test split High           11      0.933   0.0674  High   Preprocessor1~
##  4 train/test split High           13      0.996   0.00442 High   Preprocessor1~
##  5 train/test split High           14      0.921   0.0787  High   Preprocessor1~
```

```
##  6 train/test split High          16      0.954   0.0456  High   Preprocessor1~
##  7 train/test split High          21      0.942   0.0582  Low    Preprocessor1~
##  8 train/test split High          23      0.634   0.366   High   Preprocessor1~
##  9 train/test split High          24      0.871   0.129   High   Preprocessor1~
## 10 train/test split High          25      0.664   0.336   High   Preprocessor1~
## # ... with 14,265 more rows
```

```
last_fit_results %>%
  roc_curve(truth = rating, estimate = .pred_High) %>%
  autoplot()
```



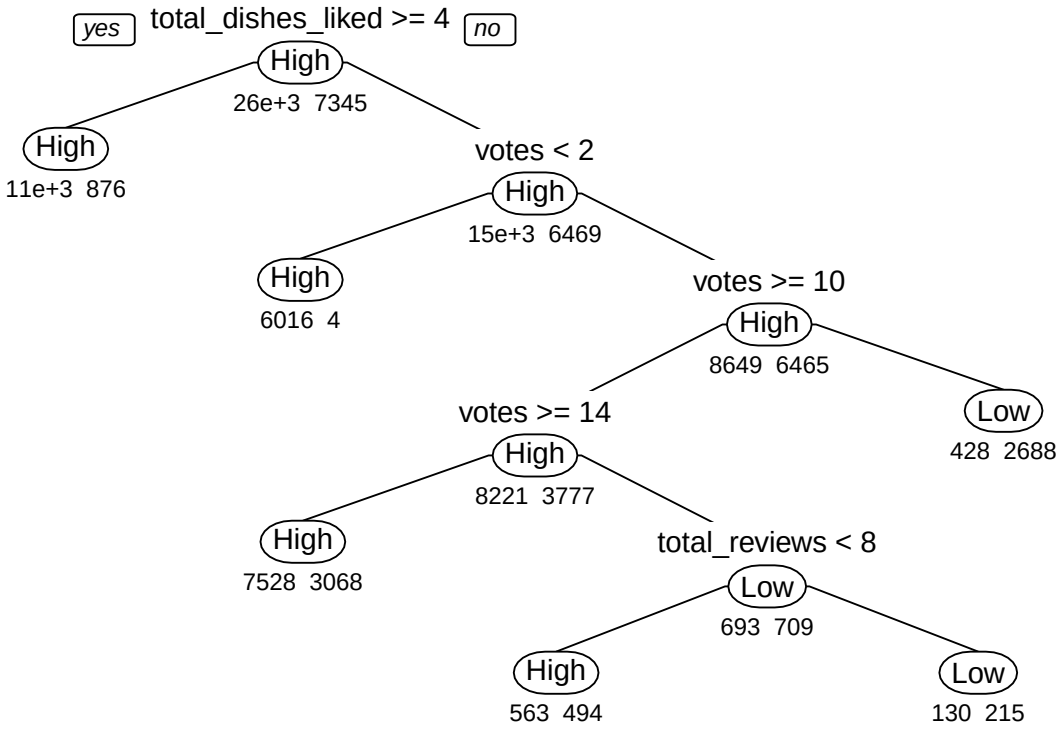

```
conf_mat(last_fit_results, truth = rating, estimate = .pred_class)
```

```
##           Truth
## Prediction  High   Low
##       High 10843  2994
##       Low    284   154
```

Model 2:Decision tree

```
# classification tree
fullygrown_ct <- rpart(rating ~ ., data = zomato_training, method = "class",control = rpart.control(cp =
## Pruned Decision tree
mincp <- fullygrown_ct$cptable[which.min(fullygrown_ct$cptable[,"xerror"]),"CP"]
# Prune the tree with optimal cp
pruned_ct <- prune(fullygrown_ct, cp = mincp )
pruned_ct_point_pred_train <- predict(pruned_ct,zomato_test,type = "class")
```

```
prp(pruned_ct, type = 1, extra = 1, under = TRUE, split.font = 1, varlen = 0)
```




```
# generate confusion matrix
confusionMatrix(pruned_ct_point_pred_train, as.factor(zomato_test$rating))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  High   Low
##       High 10902  1913
##       Low    225  1235
##
##                Accuracy : 0.8502
##                  95% CI : (0.8443, 0.856)
##     No Information Rate : 0.7795
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4607
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9798
##             Specificity : 0.3923
##          Pos Pred Value : 0.8507
##          Neg Pred Value : 0.8459
##              Prevalence : 0.7795
```

```
##          Detection Rate : 0.7637
##    Detection Prevalence : 0.8977
##       Balanced Accuracy : 0.6860
##
##        'Positive' Class : High
##
```

```
## Pruned Decision tree-AUC ROC
tree.preds <- predict(pruned_ct, zomato_test, type="prob")[, 2]
tree.roc <- roc(zomato_test$rating, tree.preds)
```

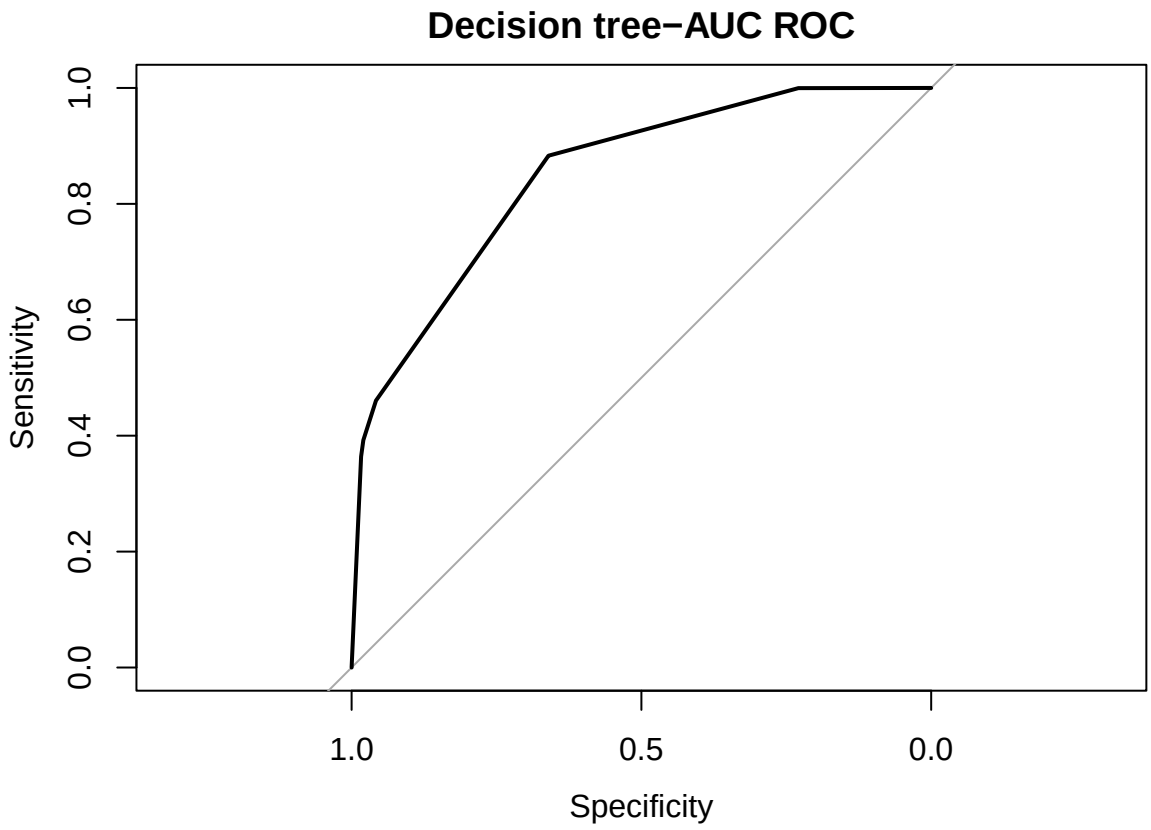
```
## Setting levels: control = High, case = Low
```

```
## Setting direction: controls < cases
```

```
print(tree.roc)
```

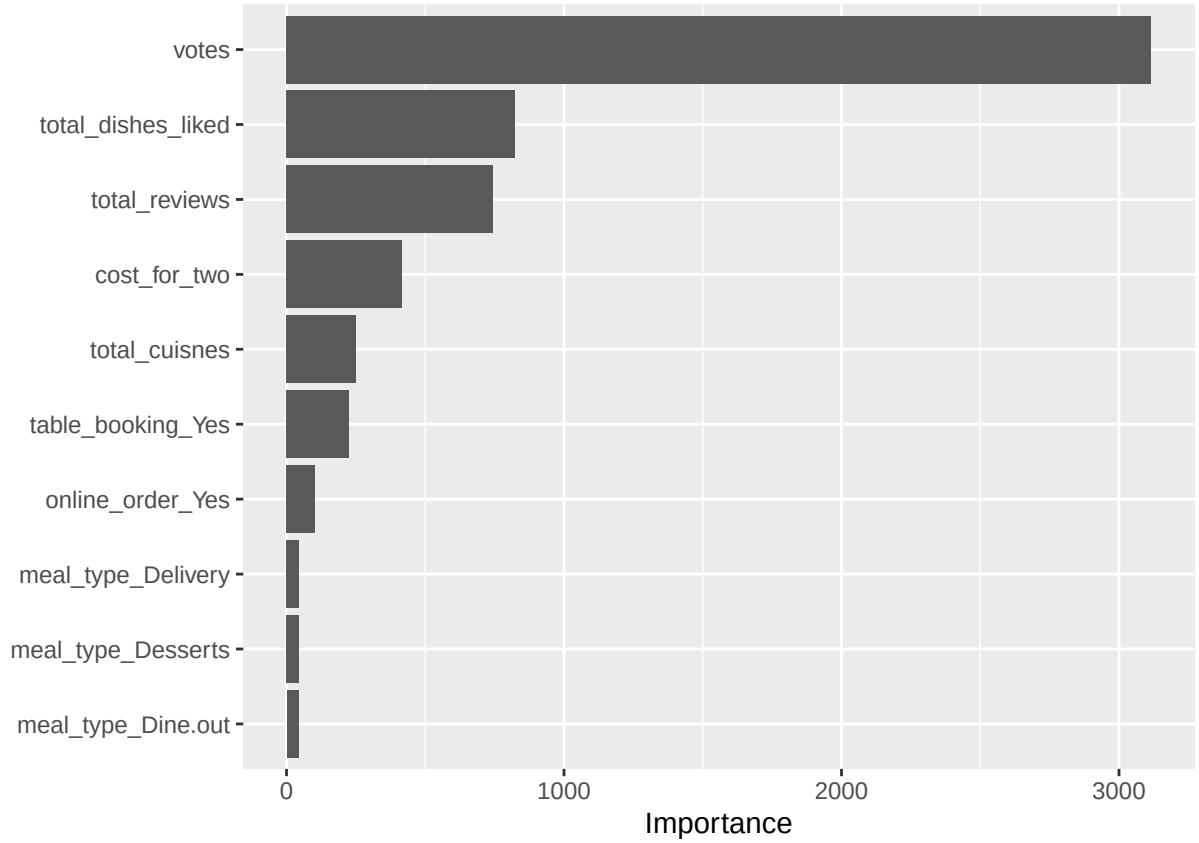
```
##
## Call:
## roc.default(response = zomato_test$rating, predictor = tree.preds)
##
## Data: tree.preds in 11127 controls (zomato_test$rating High) < 3148 cases (zomato_test$rating Low).
## Area under the curve: 0.8487
```

```
plot(tree.roc,main="Decision tree-AUC ROC")
```



Model 3 :Random Forest

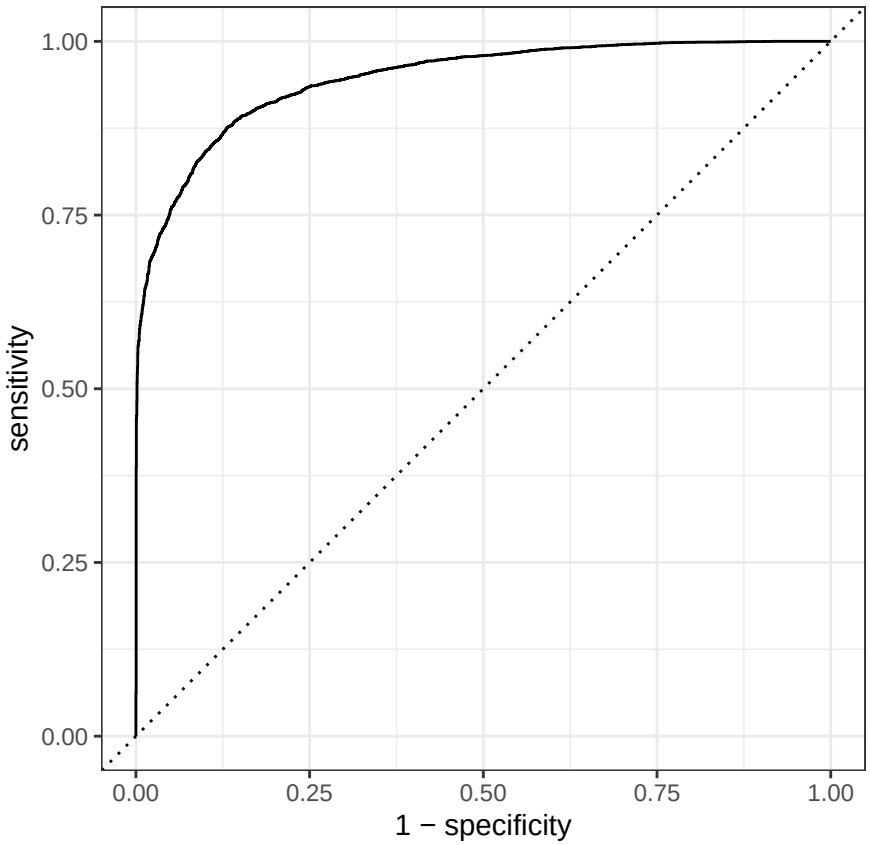
```
rf_model <- rand_forest(mtry = 3,
                        trees = 300,
                        min_n = 8) %>%
            set_engine('ranger', importance = "impurity") %>%
            set_mode('classification')

rf_workflow <- workflow() %>%
               add_model(rf_model) %>%
               add_recipe(zomato_recipe)

set.seed(10)

rf_grid <- grid_random(mtry() %>% range_set(c(2, round(sqrt(ncol(zomato_training))))),
                       trees(),
                       min_n(),
                       size = 3)

set.seed(10)

rf_tuning <- rf_workflow %>%  tune_grid(resamples = zomato_folds,rid = rf_grid)

best_rf <- rf_tuning %>% select_best(metric = 'roc_auc')
final_rf_workflow <- rf_workflow %>% finalize_workflow(best_rf)
rf_last_fit <- final_rf_workflow %>% last_fit(split = zomato_split,metrics=my_metrics)
zomato_rf_fit <- final_rf_workflow %>% fit(data = zomato_training)

rf_trained_model <- zomato_rf_fit %>% extract_fit_parsnip()

vip(rf_trained_model)
```




```
metrics<-rf_last_fit %>% collect_metrics()
metrics
```

```
## # A tibble: 5 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.877 Preprocessor1_Model1
## 2 sens     binary         0.978 Preprocessor1_Model1
## 3 spec     binary         0.519 Preprocessor1_Model1
## 4 f_meas   binary         0.925 Preprocessor1_Model1
## 5 roc_auc  binary         0.946 Preprocessor1_Model1
```

```
rf_last_fit %>% collect_predictions() %>% roc_curve(truth = rating, estimate = .pred_High) %>% autoplot
```




```
rf_last_fit %>% collect_predictions() %>% conf_mat(truth = rating, estimate = .pred_class)
```

```
##           Truth
## Prediction  High   Low
##       High 10883  1513
##       Low    244  1635
```

Model 4:Text Mining

```
data <- read.csv("zomato.csv")
data$label <- ifelse(data$rate>3,1,-1)
data$total_reviews <- str_count(data$reviews_list, "\\)")+1
data <- data %>% filter(total_reviews >1 & total_reviews <5)
data <- data %>% select(reviews_list,label)
corpus <- Corpus(VectorSource(data$reviews_list))
data %>% group_by(label) %>%summarise(count = n())
```

```
## # A tibble: 2 x 2
##   label count
##   <dbl> <int>
## 1    -1  3314
## 2     1 13761
```

```
corpus <- corpus %>%
  tm_map(content_transformer(tolower)) %>%
  tm_map(stripWhitespace) %>%
  tm_map(removePunctuation) %>%
```

```
  tm_map(removeNumbers) %>%
  tm_map(removeWords, stopwords("english")) %>%
  tm_map(stemDocument)

corpus <- tm_map(corpus, removeWords, c("rate","ratedn"))
```

```
find_freq_terms_fun <- function(corpus_in){
doc_term_mat <- TermDocumentMatrix(corpus_in)
freq_terms <- findFreqTerms(doc_term_mat)[1:max(doc_term_mat$nrow)]
terms_grouped <- doc_term_mat[freq_terms,] %>%
    as.matrix() %>%
    rowSums() %>%
    data.frame(Term=freq_terms, Frequency = .) %>%
    arrange(desc(Frequency)) %>%
    mutate(prop_term_to_total_terms=Frequency/nrow(.))
return(data.frame(terms_grouped))
}
```

Visualizing Most Frequent Words

```
positive_freq_terms <- data.frame(find_freq_terms_fun(corpus))
head(positive_freq_terms,5)
```

```
##        Term Frequency prop_term_to_total_terms
## good   good     18302                0.9074322
## place place     17969                0.8909217
## food   food     16455                0.8158560
## order order     13659                0.6772274
## tast   tast     10828                0.5368635
```

```
wordcloud2(positive_freq_terms[,1:2], shape="circle",color="random-dark")
```

Perform TF-IDF and latent semantic analysis

```
ads_tdm <- TermDocumentMatrix(corpus)
review_dtm <- removeSparseTerms(ads_tdm,0.99)
ads_tfidf <- weightTfIdf(review_dtm)
```

Produce a concept matrix

```
ads_lsa_tfidf <- lsa(ads_tfidf, dim =20)
```

convert to data frame

```
Ads_words_df <- as.data.frame(as.matrix(ads_lsa_tfidf$dk))
dim(Ads_words_df)
```

```
## [1] 17075    20
```

```
Ads_df_analysis <- data.frame(ifelse(data$label==-1, 0, 1), Ads_words_df)
names(Ads_df_analysis)[1] <- "label"
```

Test/Train Split

```
set.seed(10)
ads_split <- initial_split(Ads_df_analysis, prop = 0.60)
ads_training <- ads_split %>% training()
ads_valid <- ads_split %>% testing()
```

Logistic Regression

```
reg <- glm(label ~ ., data = ads_training, family = 'binomial')
pred <- predict(reg, newdata = ads_valid, type = "response")
confusionMatrix(table(ifelse(pred>0.5,1,0), ads_valid$label))
```

```
## Confusion Matrix and Statistics
##
##
##        0    1
##   0   66   39
##   1 1263 5462
##
##                Accuracy : 0.8094
##                  95% CI : (0.7999, 0.8186)
##     No Information Rate : 0.8054
##     P-Value [Acc > NIR] : 0.2093
##
##                   Kappa : 0.0654
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.049661
##             Specificity : 0.992910
##          Pos Pred Value : 0.628571
##          Neg Pred Value : 0.812193
##              Prevalence : 0.194583
##          Detection Rate : 0.009663
##    Detection Prevalence : 0.015373
##       Balanced Accuracy : 0.521286
##
##        'Positive' Class : 0
##
```

Findings

From the exploratory data analysis,it can be understood that locality with highest number of restaurants have more online orders with average cost of food.It is also surprising to see that most of the restaurants in the Bangalore area are serving North Indian and Chinese cuisines followed by South Indian cuisines.Based on the output from top foods it is evident that people are preferring fast foods than regular meals.Finally the results from word cloud showed that customers are talking frequently about food,place,order and taste.

When it comes to modeling,we tried to predict cost_for_two by fitting Linear regression with only numerical variables and with both numerical and categorical variables.From the results we interpreted that the simple model is doing better in terms of predicting the cost when compared to model 2.After fitting neural network we noticed that model is over fitting,hence it is not recommended for this data.

To predict the rating of food,we built Decision Tree,Logistic Regression and Random Forest.Among them,Random Forest model yielded great results with an accuracy of 87.6% and roc_auc of 94.5%.The important variables in determining the rating are votes,total dishes liked,total reviews,cost for two and total cuisinesThe reviews in the data were labelled as positive and negative based on rating and text mining is done of the data.The model gave an impressive at classifying the rating with an accuracy of 80.94%.

Recommendations

There are comparatively less restaurants in locality – "New BEL Road" as per Zomato data. Restaurants in this locality can be expanded to appear more frequently in recommendations.

Significantly,there are higher number of restaraunts who are not into online ordering,executives at zomato should try to explain the its importance to them so that it will be helpful in the business development.

Restaurants in "Lavelle Road" has lowest percentage of online orders. So Zomato should concentrate on providing discount or promo codes or offers to those restaurants to grab customer's attraction.

People are more inclined towards North Indian and Chinese cuisines so Zomato should give suggestions to their partnered restaurants to include more of those dishes in their menu.