Final_Project_Final_Draft

December 13, 2021

ADS-500B-02-FA21

Final Data Science Programming Project

Group 6: Claire Phibbs & Aaron Carr

Dataset: Bank Marketing

Introduction For this final project, the bank_marketing.csv dataset was used to perform a secondary data analysis. This dataset provides information about marketing campaigns from a European bank. The dataset includes variables for age, job, marital status, education level, loan default status (i.e., yes/no), account balance, whether the loan is for housing (i.e., yes/no) or personal (i.e., yes/no), as well as information on the most recent bank outreach campaign, including contact method (i.e., cellular, telephone, unknown, Nan), last day of the month contaced, last month contacted, duration of last contact, number of contacts during current campaign, number of previous contacts (before the current campaign), outcome of the previous campaign, and whether the client has subscribed to a term deposit (i.e., yes/no).
The goals of this project are to import and transform a raw dataset, perform exploratory and descriptive analysis, provide appropriate visualizations, and apply analytic models on the data.
The main question being explored is whether one or more features–including demographics like age and education level, and previous campaign results–can be used to predict whether a bank client will take out a deposit. A secondary question being explored is how factors like age, marital status, education level, account balance, and loan status affect defaulting on a loan from this European bank.
Logistic regression models will be used to explore the relationship between the specified independent varibales and dependent variable for each question.

Import libariaries for data processing & analyses

[1]:
```
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
import copy
from textwrap import wrap

from scipy.stats import chi2_contingency
```

```python
from scipy.stats import mode

from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor as␣
 ↪vif

import rpy2.robjects as robjects
import rpy2.robjects.packages as rpackages
from rpy2.robjects.vectors import StrVector
from rpy2.robjects import pandas2ri
pandas2ri.activate()
from rpy2.robjects.conversion import localconverter
import rpy2.robjects.numpy2ri
from rpy2.robjects.packages import importr

%load_ext rpy2.ipython
%R library(caTools)
%R library(ROCR)
%R library(car)
%R library(rms)
%R library("ggplot2")
%R library(tidyverse)
```

```
R[write to console]: Loading required package: carData

R[write to console]: Loading required package: Hmisc

R[write to console]: Loading required package: lattice

R[write to console]: Loading required package: survival

R[write to console]: Loading required package: Formula

R[write to console]: Loading required package: ggplot2

R[write to console]:
Attaching package: ‘Hmisc’


R[write to console]: The following objects are masked from ‘package:base’:

    format.pval, units
```

```
R[write to console]: Loading required package: SparseM

R[write to console]:
Attaching package: ‘SparseM’


R[write to console]: The following object is masked from ‘package:base’:

    backsolve


R[write to console]:
Attaching package: ‘rms’


R[write to console]: The following objects are masked from ‘package:car’:

    Predict, vif


R[write to console]:   Attaching packages
                          tidyverse 1.3.1

R[write to console]:  tibble  3.1.6
dplyr   1.0.7
  tidyr   1.1.4      stringr 1.4.0
  readr   2.1.1      forcats 0.5.1
  purrr   0.3.4

R[write to console]:   Conflicts
                           tidyverse_conflicts()
  dplyr::filter()    masks
stats::filter()
  dplyr::lag()       masks stats::lag()
  dplyr::recode()    masks car::recode()
  purrr::some()      masks car::some()
  dplyr::src()       masks Hmisc::src()
  dplyr::summarize() masks
Hmisc::summarize()
```

[1]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f9097c67500> [RTYPES.STRSXP]
R classes: ('character',)
['forcats', 'stringr', 'dplyr', 'purrr', …, 'utils', 'datasets', 'methods',
'base']
```

```
[2]: %pwd
```

```
[2]: '/Users/clairephibbs/Desktop'
```

Set global variable values

```
[3]: round_int01 = 4
     unk_str = 'unknown'
```

1. Data Importing & Pre-processing

Null Hyphothesis (H0): There is no relationship between the X variables and the Y variable (i.e., deposit).
Alternative Hyphothesis (Ha): There is a relationship between the X variables and the Y variable (i.e., deposit).
***

Initial data review

```
[4]: # Load dataframe (df) from csv file, print df
     bank_df01 = pd.read_csv('bank_marketing.csv', header=0, sep=';')
     print(bank_df01.head(10)) # display first 10 rows of df

     bank_df01_cols_lst01 = bank_df01.columns.values.tolist() # review column names

     bank_df01_len01 = len(bank_df01)
     print(f'\n Number of df rows = {bank_df01_len01}') # display df length
```

```
    age           job   marital  education default  balance housing loan  \
0  58.0    management   married   tertiary      no     2143     yes   no
1  44.0    technician    single  secondary      no       29     yes   no
2  33.0  entrepreneur   married  secondary      no        2     yes  yes
3  47.0   blue-collar   married    unknown      no     1506     yes   no
4  33.0       unknown    single    unknown      no        1      no   no
5  35.0    management   married   tertiary      no      231     yes   no
6  28.0    management    single   tertiary      no      447     yes  yes
7  42.0  entrepreneur  divorced   tertiary     yes        2     yes   no
8  58.0       retired   married    primary      no      121     yes   no
9  43.0    technician    single  secondary      no      593     yes   no

   contact  day month  duration  campaign  pdays  previous poutcome deposit
0  unknown    5   may       261         1     -1         0  unknown      no
1  unknown    5   may       151         1     -1         0  unknown      no
2  unknown    5   may        76         1     -1         0  unknown      no
3  unknown    5   may        92         1     -1         0  unknown      no
4      NaN    5   may       198         1     -1         0  unknown      no
5  unknown    5   may       139         1     -1         0  unknown      no
```

```
6  unknown    5   may       217          1     -1          0  unknown      no
7  unknown    5   may       380          1     -1          0  unknown      no
8  unknown    5   may        50          1     -1          0  unknown      no
9  unknown    5   may        55          1     -1          0  unknown      no

 Number of df rows = 45211
```

[5]:
```python
print(bank_df01.isnull().sum()) # review dataset for columns w/ null value
```

```
age          1339
job             0
marital         0
education       0
default      1306
balance         0
housing         0
loan            0
contact      1383
day             0
month           0
duration        0
campaign        0
pdays           0
previous        0
poutcome        0
deposit         0
dtype: int64
```

Initial Descriptive statistics

[6]:
```python
data_type = bank_df01.dtypes
print(data_type)
```

```
age          float64
job           object
marital       object
education     object
default       object
balance        int64
housing       object
loan          object
contact       object
day            int64
month         object
duration       int64
campaign       int64
pdays          int64
previous       int64
poutcome      object
```

```
deposit       object
dtype: object
```

[7]:
```
bank_df01.describe()
```

[7]:
```
                age        balance           day      duration      campaign  \
count  43872.000000   45211.000000  45211.000000  45211.000000  45211.000000
mean      40.924781    1362.272058     15.806419    258.163080      2.763841
std       10.610835    3044.765829      8.322476    257.527812      3.098021
min       18.000000   -8019.000000      1.000000      0.000000      1.000000
25%       33.000000      72.000000      8.000000    103.000000      1.000000
50%       39.000000     448.000000     16.000000    180.000000      2.000000
75%       48.000000    1428.000000     21.000000    319.000000      3.000000
max       95.000000  102127.000000     31.000000   4918.000000     63.000000

              pdays      previous
count  45211.000000  45211.000000
mean      40.197828      0.580323
std      100.128746      2.303441
min       -1.000000      0.000000
25%       -1.000000      0.000000
50%       -1.000000      0.000000
75%       -1.000000      0.000000
max      871.000000    275.000000
```

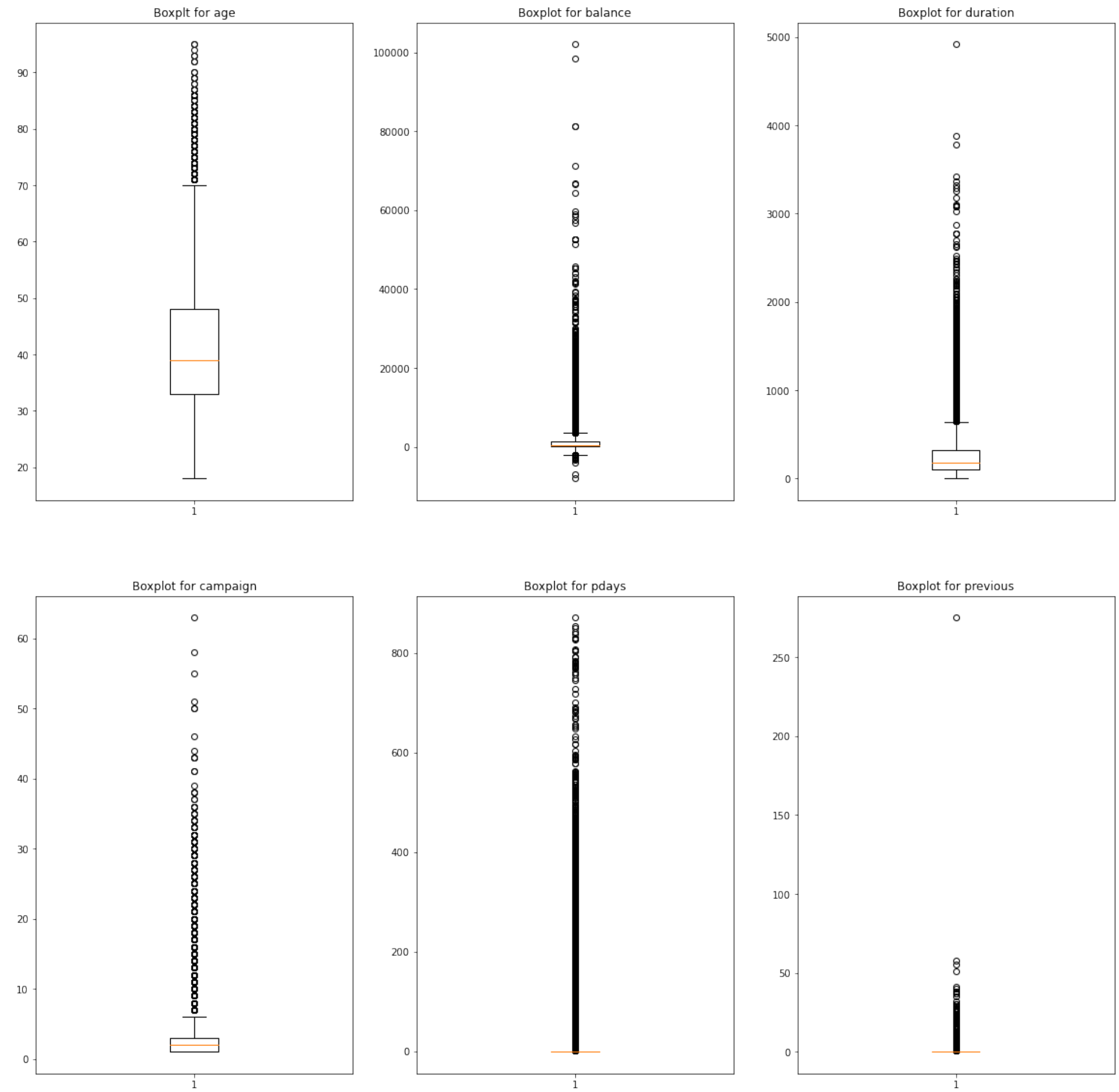
Data Import Explanation & Dataset Characteristics Initial steps included importing the bank_marketing.csv into Jupyter notebook as a pandas dataframe. The initial dataset has 45,211 rows and 17 columns (i.e., `age`, `job`, `marital`, `education`, `default`, `balance`, `housing`, `loan`, `contact`, `day`, `month`, `duration`, `campaign`, `pdays`, `previous`, `poutcome`, and `deposit`).
There are a total of 4,028 missing values: `age` has 1,339 missing values; `default` has 1,306 missing values; and `contact` has 1,383 missing. The dtype command was used to see the different types of variables we are working with.
The dataset contains a large number of categorical variables, including: `job`, `marital`, `education`, `default`, `housing`, `loan`, `contact`, `month`, `poutcome`, and `deposit`. The dataset also contains a few numeric variables: `age`, `balance`, `day`, `duration`, `campaign`, `pdays`, and `previous`. As `day` is ordinal, it will be treated as categorical varibale instead of discrete numerical.
The describe command was also used to display the summary statistics of the numeric variables in the bank_df01 dataset. The summary statistics table shows the mean, standard deviation, min/max, and quartiles (reference table above).

Fill in missing data & transform ambiguous values

Create inital boxplots for features w/ numerical values

[8]:
```
# Plot boxplots for numerical variables
fig, axs = plt.subplots(2, 3, figsize=(20 , 20), sharey=False, sharex=False) #␣
↪set figure fram
```

```
axs[0, 0].boxplot(bank_df01['age'].dropna()) # subplot 1 for full dataset
axs[0, 0].set_title('Boxplt for age')

axs[0, 1].boxplot(bank_df01['balance'].dropna()) # subplot 2 for outlier dataset
axs[0, 1].set_title('Boxplot for balance')

axs[0, 2].boxplot(bank_df01['duration'].dropna()) # subplot 3 for outlier dataset
axs[0, 2].set_title('Boxplot for duration')

axs[1, 0].boxplot(bank_df01['campaign'].dropna()) # subplot 4 for outlier dataset
axs[1, 0].set_title('Boxplot for campaign')

axs[1, 1].boxplot(bank_df01['pdays'].dropna()) # subplot 5 for outlier dataset
axs[1, 1].set_title('Boxplot for pdays')

axs[1, 2].boxplot(bank_df01['previous'].dropna()) # subplot 6 for outlier dataset
axs[1, 2].set_title('Boxplot for previous')

plt.show()
```



Initial visualizaton Check boxplots for numerical variables to indentify spread (varinace), and identify variables with outliers.
Each one of the plots shows outliers for that variable. In order to mitigate their effects on the analyses, there will be removed. Focusing specifically on `previous` and `pdays`, the majority of values are zero based on the lack of variance, while there are also a signifacnt number of outliers. These variables will be removed, and instead a new feature will be created called `previous_contact` (yes=1/no=0) based on whether `pdays` equals -1 (client not previously contacted).
Additional feature transformations are outlined below.

Define function: Fill in missing numerical data based on group by

[9]:
```
def gb_agg_sub(df, t_var=None, gb_vars=[], agg_meth='mean'):
    '''current aggregate methods = sum, mean'''
    if agg_meth == 'sum':
```

```
        df_gpb01 = df.groupby(gb_vars).sum() # create a multi-indexed dataframe
    else:
        df_gpb01 = df.groupby(gb_vars).mean() # create a multi-indexed dataframe
    print(df_gpb01)
    df = pd.merge(df, df_gpb01, how='left', on=gb_vars, suffixes=(None, '_y'))
    df[t_var] = df[t_var].fillna(value=df[t_var + '_y'])
    return df
```

[10]:

```
# Run function to fill in missing age values based on group by
bank_df01 = gb_agg_sub(bank_df01, 'age',  ['marital', 'education'])

# Reset col names
bank_df01 = bank_df01[bank_df01_cols_lst01]

# Remove records where balance is less than zero
bank_df01 = bank_df01.loc[(bank_df01['balance'] >= 0), :]

# Create new feature
bank_df01['previous_contact'] = 0
bank_df01.loc[(bank_df01['pdays'] != -1), 'previous_contact'] = 1

# Save df len for generating % loss later
bank_df01_len02 = len(bank_df01)

# Remove col not used for analysis
bank_df01 = bank_df01.drop(['contact'], axis=1)
bank_df01 = bank_df01.drop(['pdays'], axis=1)
bank_df01 = bank_df01.drop(['previous'], axis=1)

# Fill in `default` w/ "unknown" to transform it below
bank_df01['default'] = bank_df01['default'].fillna(unk_str)

print(bank_df01.head())
```

```
                          age      balance        day    duration  campaign  \
marital  education
divorced primary    51.606849  1137.680851  15.128989  273.042553  2.453457
         secondary  44.168138   902.772647  15.759503  257.512256  2.591829
         tertiary   45.350035  1700.917063  16.254929  263.249490  2.779062
         unknown    49.210843  1417.147929  15.390533  292.674556  2.781065
married  primary    46.556387  1286.655547  15.457682  251.259055  2.906786
         secondary  42.370622  1251.750254  15.880392  255.101670  2.786202
         tertiary   42.262362  1848.779341  16.072464  252.082836  2.883205
         unknown    48.258467  1557.787931  16.018966  251.173276  2.981897
single   primary    36.690821  1131.215709  15.456038  269.594373  2.724502
         secondary  33.169782  1060.546773  15.506725  266.643494  2.569442
         tertiary   33.785899  1643.351210  16.055509  266.518364  2.755217
         unknown    34.744141  1493.657197  15.488636  259.486742  2.579545
```

```
                         pdays  previous
marital  education
divorced primary    38.224734  0.466755
         secondary  42.044760  0.555595
         tertiary   40.233855  0.598912
         unknown    42.668639  0.443787
married  primary    34.699962  0.484941
         secondary  40.452941  0.551634
         tertiary   36.024013  0.635124
         unknown    34.632759  0.462069
single   primary    42.690504  0.535756
         secondary  46.439927  0.606468
         tertiary   43.059057  0.720785
         unknown    39.880682  0.560606
    age           job  marital  education default  balance housing loan  day  \
0  58.0    management  married   tertiary      no     2143     yes   no    5
1  44.0    technician   single  secondary      no       29     yes   no    5
2  33.0  entrepreneur  married  secondary      no        2     yes  yes    5
3  47.0   blue-collar  married    unknown      no     1506     yes   no    5
4  33.0       unknown   single    unknown      no        1      no   no    5

  month  duration  campaign poutcome deposit  previous_contact
0   may       261         1  unknown      no                 0
1   may       151         1  unknown      no                 0
2   may        76         1  unknown      no                 0
3   may        92         1  unknown      no                 0
4   may       198         1  unknown      no                 0
```

Define function: Transform ambigous categorical values

[11]:
```
def cat_mode(df, var=[(None, None, [])]):
    '''create function to transform variable values;
    var input uses "column string", "value to transform", "group_by vars" as␣
 ↪x,y,z tuple'''
    df_sub01 = df.dropna()
    int_start01 = 1
    for i, j, k in var:
        df_sub02 = df_sub01.loc[(df_sub01[i] != j), :]
        df_gpb01 = df_sub02.groupby(k)[i].agg(lambda x: pd.Series.mode(x)[0]).
 ↪to_frame() # create a multi-indexed dataframe; add lambda fx (Stack␣
 ↪Overflow, n.d.)
        print(f'\ndf_gpb01:\n{df_gpb01}')
        df_sub01 = pd.merge(df_sub01, df_gpb01, how='left', on=k,␣
 ↪suffixes=(None, '_z' + str(int_start01)))
        df_sub01.loc[(df_sub01[i] == j) & (df_sub01[i + '_z' +␣
 ↪str(int_start01)].notna()), i] = df_sub01[i + '_z' + str(int_start01)]
        df_sub01 = df_sub01.drop(i + '_z' + str(int_start01), axis=1)
```

```
        int_start01 += 1
    return df_sub01
```

[12]:
```
bank_df01 = cat_mode(bank_df01, [('job', unk_str, ['marital', 'education']),␣
↪('education', unk_str, ['job', 'age']), ('default', unk_str, ['job',␣
↪'marital', 'education'])])

print(f'\nbank_df01:\n{bank_df01.head(10)}') # Review transformed df
```

```
df_gpb01:
                              job
marital  education
divorced primary    blue-collar
         secondary   technician
         tertiary    management
         unknown    blue-collar
married  primary    blue-collar
         secondary  blue-collar
         tertiary    management
         unknown    blue-collar
single   primary    blue-collar
         secondary   technician
         tertiary    management
         unknown        student

df_gpb01:
                  education
job        age
admin.     20.0   secondary
           21.0   secondary
           22.0   secondary
           23.0   secondary
           24.0   secondary
…                       …
unemployed 62.0   secondary
           63.0     primary
           64.0   secondary
           65.0   secondary
           66.0     primary

[606 rows x 1 columns]

df_gpb01:
                                  default
job        marital  education
admin.     divorced primary            no
                    secondary          no
```

```
                      tertiary       no
                      unknown        no
           married    primary        no
…                                 …
unemployed married    secondary      no
                      tertiary       no
           single     primary        no
                      secondary      no
                      tertiary       no

[116 rows x 1 columns]

bank_df01:
    age           job   marital  education default  balance housing loan  day  \
0  58.0    management   married   tertiary      no     2143     yes   no    5
1  44.0    technician    single  secondary      no       29     yes   no    5
2  33.0  entrepreneur   married  secondary      no        2     yes  yes    5
3  47.0   blue-collar   married  secondary      no     1506     yes   no    5
4  33.0       student    single   tertiary      no        1      no   no    5
5  35.0    management   married   tertiary      no      231     yes   no    5
6  28.0    management    single   tertiary      no      447     yes  yes    5
7  42.0  entrepreneur  divorced   tertiary     yes        2     yes   no    5
8  58.0       retired   married    primary      no      121     yes   no    5
9  43.0    technician    single  secondary      no      593     yes   no    5

  month  duration  campaign poutcome deposit  previous_contact
0   may       261         1  unknown      no                 0
1   may       151         1  unknown      no                 0
2   may        76         1  unknown      no                 0
3   may        92         1  unknown      no                 0
4   may       198         1  unknown      no                 0
5   may       139         1  unknown      no                 0
6   may       217         1  unknown      no                 0
7   may       380         1  unknown      no                 0
8   may        50         1  unknown      no                 0
9   may        55         1  unknown      no                 0
```

[13]:
```python
# Re-review dataset for columns w/ null value
print(bank_df01.isnull().sum())
```

```
age                 0
job                 0
marital             0
education           0
default             0
balance             0
housing             0
loan                0
```

```
day                   0
month                 0
duration              0
campaign              0
poutcome              0
deposit               0
previous_contact      0
dtype: int64
```

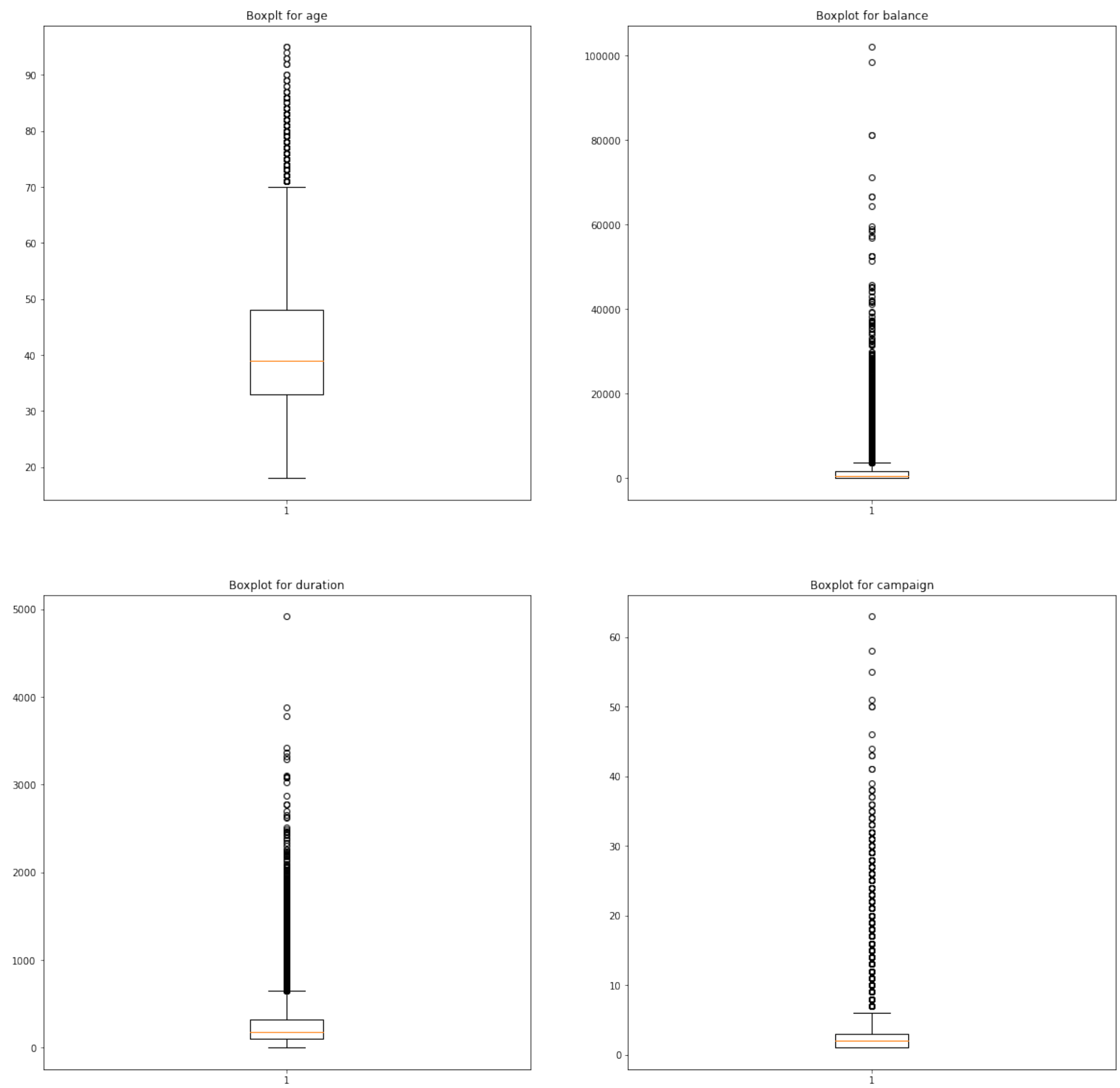
Explanation for filling in missing data In this dataset there are missing values for three of the variables (`age`, `default`, and `contact`). To handle the missing values for the `age` column we have used the groupby command to create a multi-indexed dataframe with the imputed `age` values. Basically, the marital status and education level columns are used to impute a value for `age`. The groupby command groups similar values together and takes a mean. The imputed values for `age` is then calculated by grouping similar attributes together and filling in the age column with the mean from the grouped `marital` and `education` columns. These imputed `age` values are in column `age_y` of the dataset.
To handle the missing values for `contact`, we have decided to just remove the entire column. This decision was made due to the nature of our study questions. Since it is believed that the `contact` variable is not used to predict either default status or deposit decision, it is unnecssary to fill in the missing values or include them in the dataset at all. Instead they have been removed as to not introduce more bias into the dataset by imputing the values ourselves.
The default column also has some missing values. A similar process was performed to fill in the missing data as with `age`, but for this variable the groupby was done using the following variables: `job`, `marital`, `education`.
Also, something to note; we have decided to remove the rows in the balance column, where the balance is less than zero.

2. Data Analysis & Visualizations

Create boxplots for remaining features w/ numerical values

[14]:
```
# Plot boxplots for numerical variables
fig, axs = plt.subplots(2, 2, figsize=(20 , 20), sharey=False, sharex=False) #␣
 ↪set figure fram

axs[0, 0].boxplot(bank_df01['age'].dropna()) # subplot 1 for full dataset
axs[0, 0].set_title('Boxplt for age')

axs[0, 1].boxplot(bank_df01['balance'].dropna()) # subplot 2 for outlier dataset
axs[0, 1].set_title('Boxplot for balance')

axs[1, 0].boxplot(bank_df01['duration'].dropna()) # subplot 3 for outlier␣
 ↪dataset
axs[1, 0].set_title('Boxplot for duration')

axs[1, 1].boxplot(bank_df01['campaign'].dropna()) # subplot 4 for outlier␣
 ↪dataset
```

```
axs[1, 1].set_title('Boxplot for campaign')

plt.show()
```



Interpretation of Boxplots Above, boxplots have been created for four numerical variables in the dataset. This has been done, mainly to show the distribution of the data and to visualize outliers in the dataset. The boxplot for age shows that there is a relatively normal distribution, with outliers steming from the upper whisker. The boxplot for balance displays a highly skewed distribution with many outliers steming from the upper whisker. The balance variable also has a relatively small interquartile range compared to the other variables. The boxplot for duration displays a skewed distribution with many outliers steming from the upper whisker, and a mid-sized range compared to the other variables in the dataset. The boxplot for campaign also has a highly skewed distribution with many outliers steming from the upper whisker. Note that all of the boxplots are displayed ablove.

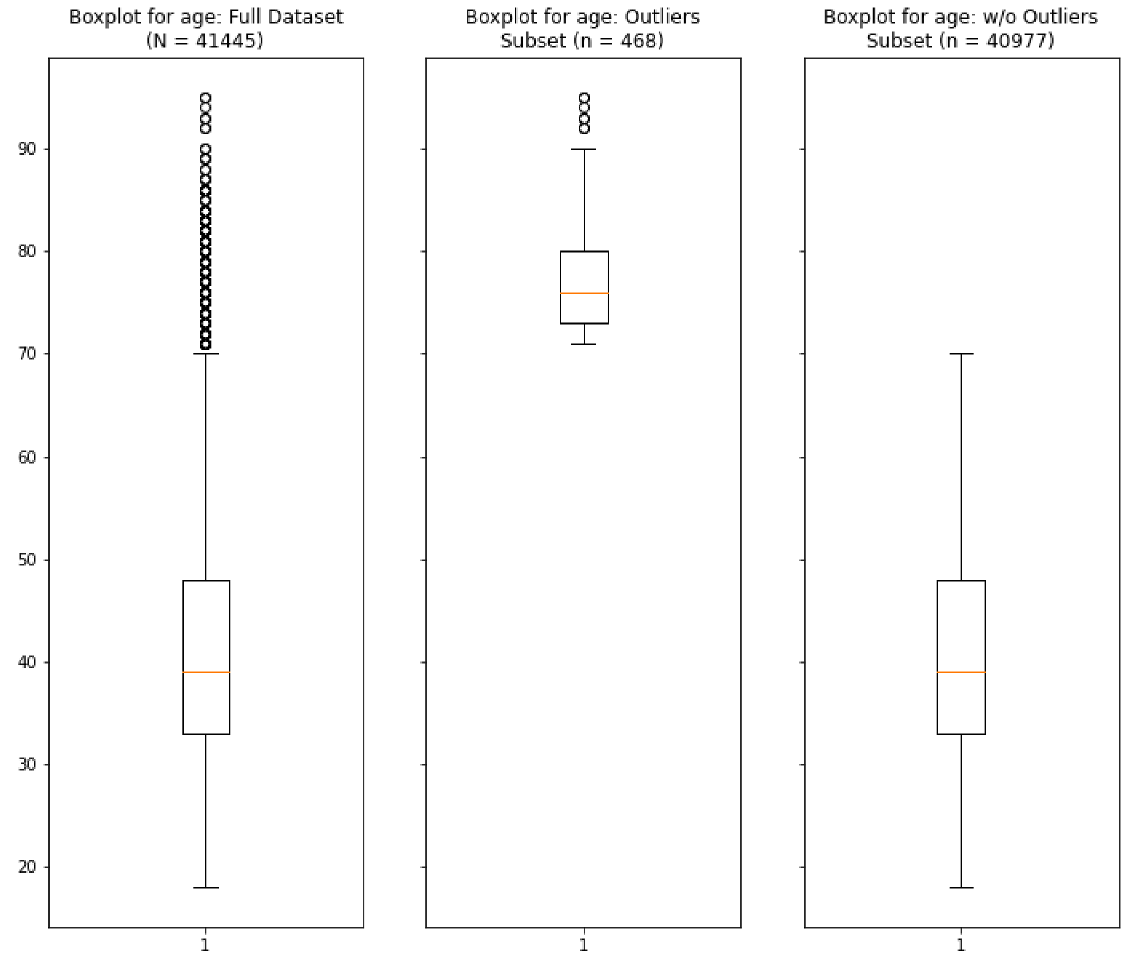
Define function: Remove outliers

[15]:
```
# Create function to generate comparison boxplots
def box_comp(df, var=[(None, 1.5)]):
    '''create function to id outliers & generate compartive boxplots;
    var input uses column string & outlier threshold as x,y tuple'''
    df_sub01 = df.dropna()
    df_sub01['outlier'] = 0
    for i, j in var:
        q3, q1 = np.percentile(df_sub01[i], [75, 25]) # calculate quartiles 1 &␣
↪3
        iqr = q3 - q1 # calculate interquartile range
        print('\nIQR: {}-{} = {}'.format(round(q1, 4),round(q3, 4),round(iqr,␣
↪4))) # display IQR

        iqr_out = iqr * j # calculate outlier threshold
        otlr_low = q1 - iqr_out # calculate lower outlier limit
        otlr_high = q3 + iqr_out # calculate upper outlier limit
        df_sub01_sub1 = df_sub01.loc[(df_sub01[i] < otlr_low) | (df_sub01[i] >␣
↪otlr_high)] # use .loc method to search for records that are outliers;␣
↪assign to new dataframe
        df_sub01_sub2 = df_sub01.loc[(df_sub01[i] >= otlr_low) & (df_sub01[i]␣
↪<= otlr_high)] # use .loc method to search for records that are outliers;␣
↪assign to new dataframe
        df_sub01.loc[(df_sub01[i] < otlr_low) | (df_sub01[i] > otlr_high),␣
↪'outlier'] = 1
        len01 = len(df_sub01)
        len02 = len(df_sub01_sub1)
        len03 = len(df_sub01_sub2)

        fig2, axs = plt.subplots(1, 3, sharey=True, figsize=(12 , 10)) # set␣
↪figure fram
        axs[0].boxplot(df_sub01[i].dropna()) # subplot 1 for full dataset
        axs[0].set_title('\n'.join(wrap(f'Boxplot for {i}: Full Dataset (N =␣
↪{len01})', 30)))
        axs[1].boxplot(df_sub01_sub1[i].dropna()) # subplot 2 for outlier␣
↪dataset
        axs[1].set_title('\n'.join(wrap(f'Boxplot for {i}: Outliers Subset (n =␣
↪{len02})', 30)))
        axs[2].boxplot(df_sub01_sub2[i].dropna()) # subplot 2 for outlier␣
↪dataset
        axs[2].set_title('\n'.join(wrap(f'Boxplot for {i}: w/o Outliers Subset␣
↪(n = {len03})', 30)))
        plt.show()
```

```
        df_sub01.describe()
        print(df_sub01[i].describe()) # descriptive stats for CRIM varibale
        print('\n', df_sub01_sub1[i].describe()) # display descriptive stats of␣
↪data subset
        print('\n', df_sub01_sub2[i].describe()) # display descriptive stats of␣
↪data subset

        print('\nmean {} = {}'.format(i, round(df_sub01[i].mean(), 4))) #␣
↪average age for full dataset
        print('median {} = {}'.format(i, round(df_sub01[i].median(), 4))) #␣
↪median age for full dataset

        print('\noutliers mean {} = {}'.format(i, round(df_sub01_sub1[i].
↪mean(), 4))) # average age for outliers
        print('outliers median {} = {}'.format(i, round(df_sub01_sub1[i].
↪median(), 4))) # median age for outliers
        print('Sub1 Column count = {}'.format(len(df_sub01_sub1.columns))) #␣
↪alternative way to print only number of columns
        print('Sub1 Row count = {}'.format(len(df_sub01_sub1))) # alternative␣
↪way to print only number of rows

        print('\nw/o outliers mean {} = {}'.format(i, round(df_sub01_sub2[i].
↪mean(), 4))) # average age for outliers
        print('w/o outliers median {} = {}'.format(i, round(df_sub01_sub2[i].
↪median(), 4))) # median age for outliers
        print('Sub2 Column count = {}'.format(len(df_sub01_sub2.columns))) #␣
↪alternative way to print only number of columns
        print('Sub2 Row count = {}'.format(len(df_sub01_sub2))) # alternative␣
↪way to print only number of rows

    df_sub01_sub3 = df_sub01.loc[(df_sub01['outlier'] == 0), :]
    len04 = len(df_sub01_sub3)
    for k, l in var:
        fig6 = plt.figure(figsize=(12 , 10)) # set figure fram
        plt.boxplot(df_sub01_sub3[k].dropna()) # subplot 1 for full dataset
        plt.title('\n'.join(wrap(f'Boxplot for {k}: Total Subset (n =␣
↪{len04})', 30)))
        plt.show()

    return df_sub01_sub3
```

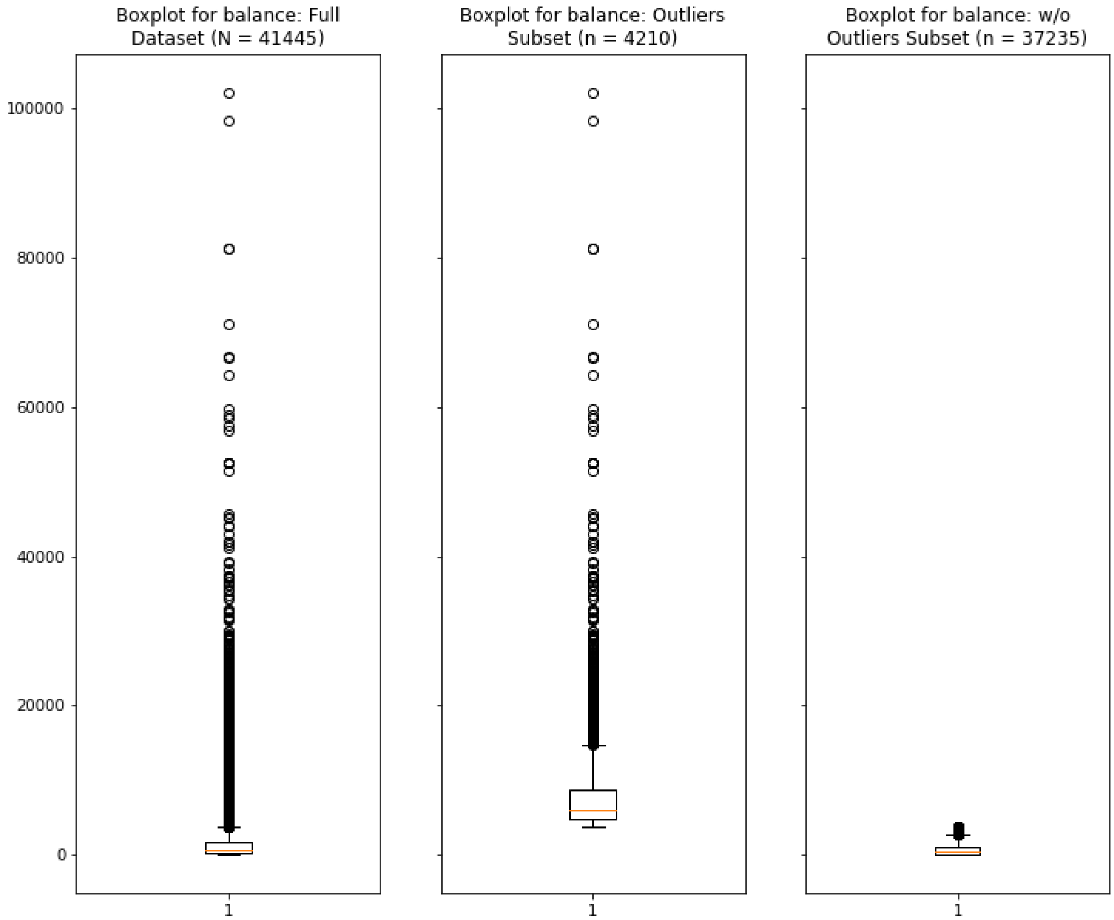
[16]:
```
bank_df01 = box_comp(bank_df01, [('age', 1.5), ('balance', 1.5), ('campaign', 1.
↪5)]) # revise main df to omit records with outliers for age variable
bank_df01 = bank_df01.drop('outlier', axis=1)

bank_df01_len03 = len(bank_df01)
```

```
IQR: 33.0-48.0 = 15.0
```



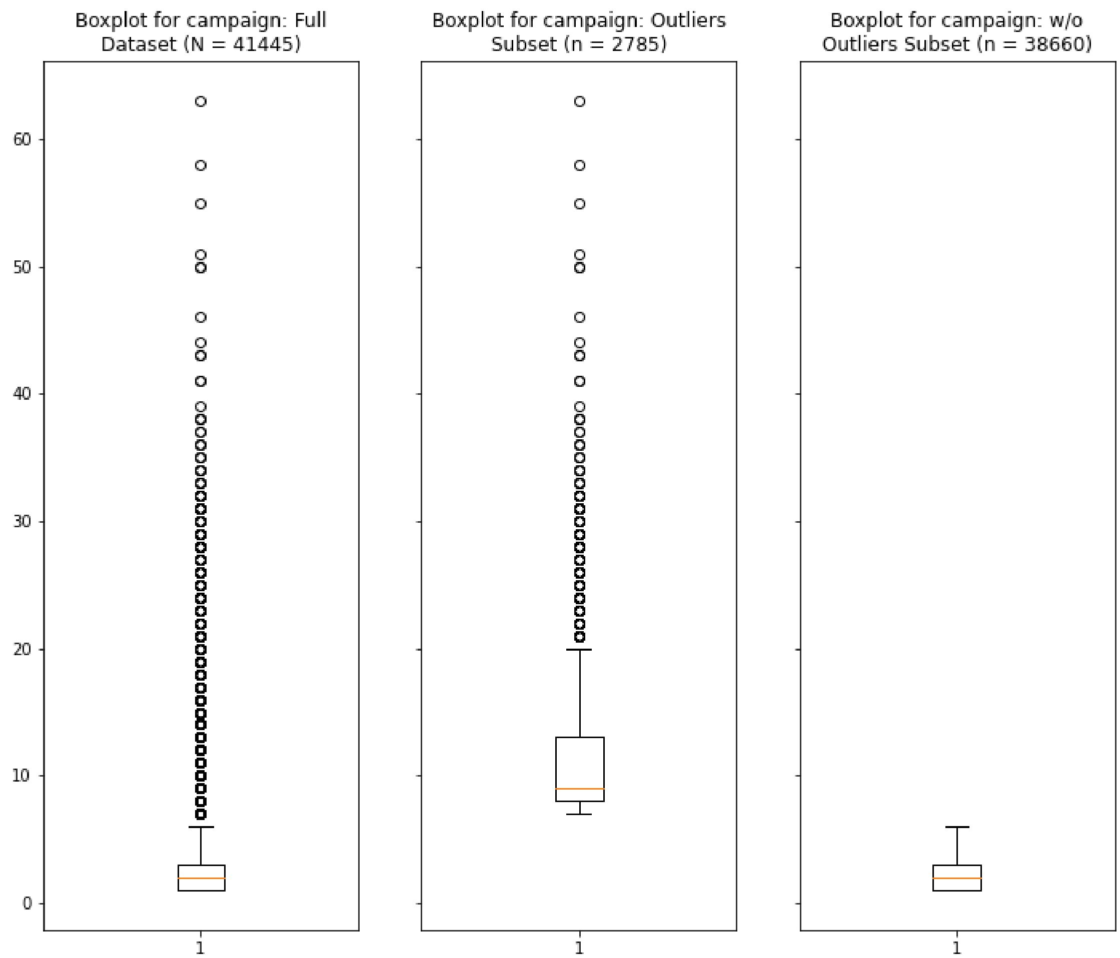

```
count    41445.000000
mean        41.057093
std         10.603407
min         18.000000
25%         33.000000
50%         39.000000
75%         48.000000
max         95.000000
Name: age, dtype: float64

 count    468.000000
mean      76.797009
std        4.841318
min       71.000000
25%       73.000000
```

```
50%       76.000000
75%       80.000000
max       95.000000
Name: age, dtype: float64

 count    40977.000000
mean        40.648906
std          9.934470
min         18.000000
25%         33.000000
50%         39.000000
75%         48.000000
max         70.000000
Name: age, dtype: float64

mean age = 41.0571
median age = 39.0

outliers mean age = 76.797
outliers median age = 76.0
Sub1 Column count = 16
Sub1 Row count = 468

w/o outliers mean age = 40.6489
w/o outliers median age = 39.0
Sub2 Column count = 16
Sub2 Row count = 40977

IQR: 146.0-1596.0 = 1450.0
```



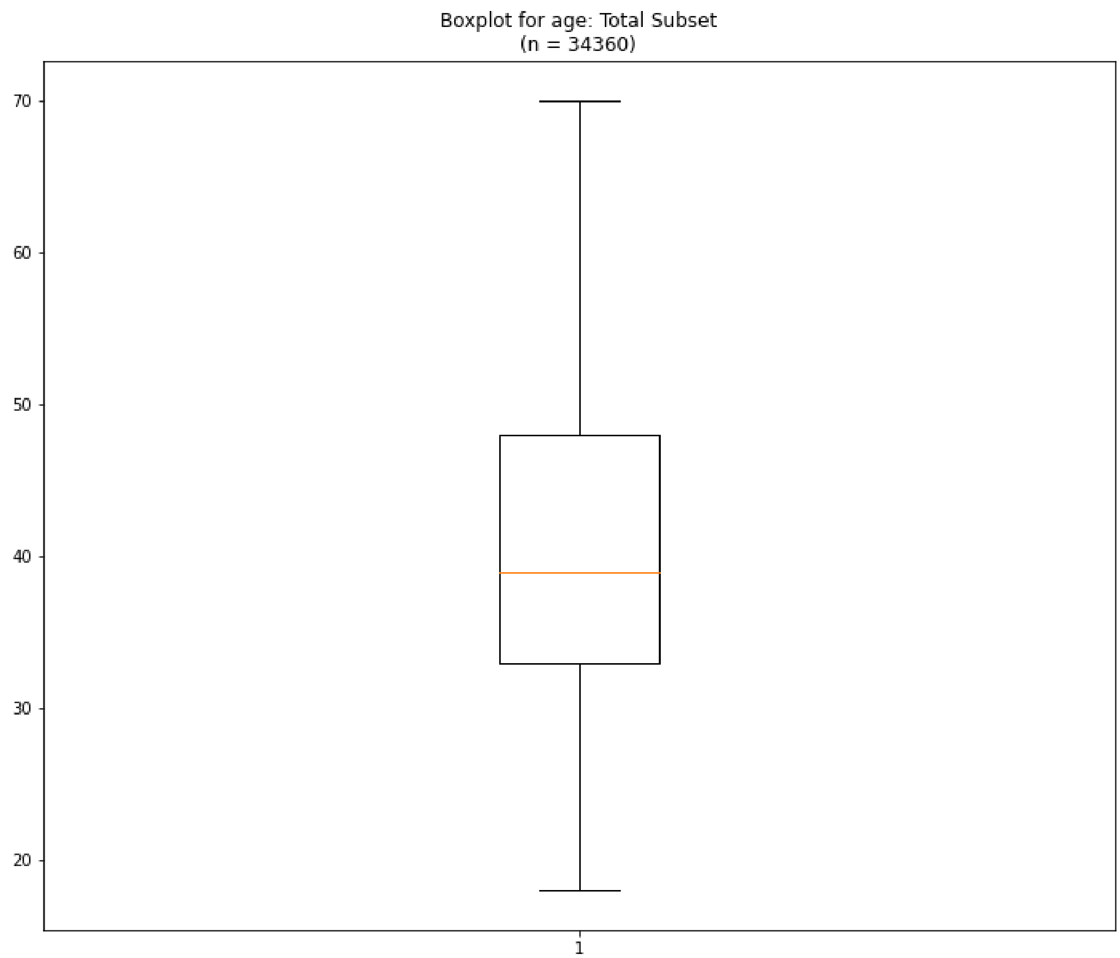

```
count     41445.000000
mean       1514.924744
std        3133.829437
min           0.000000
25%         146.000000
50%         542.000000
75%        1596.000000
max      102127.000000
Name: balance, dtype: float64

 count      4210.000000
mean       8050.544656
std        6501.285121
min        3773.000000
25%        4693.000000
50%        6005.000000
75%        8710.750000
max      102127.000000
```

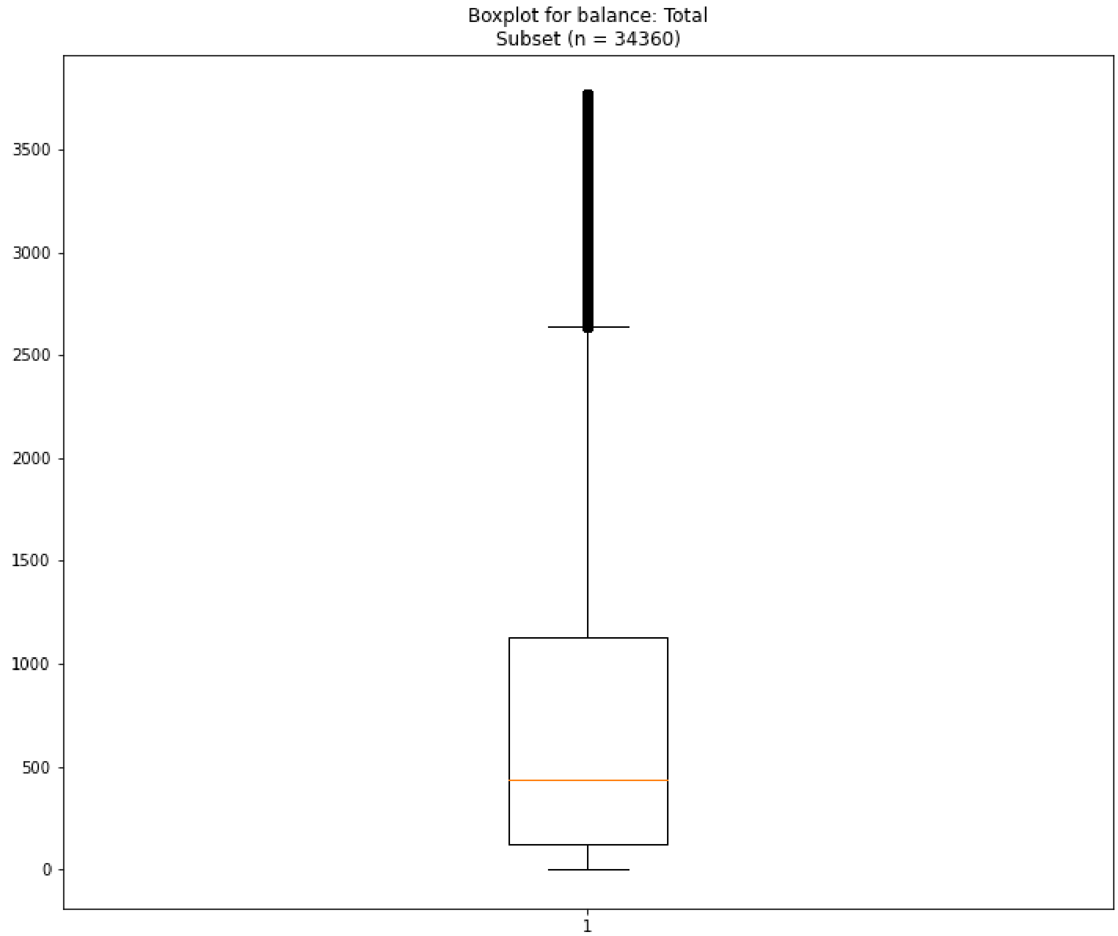
```
Name: balance, dtype: float64

 count    37235.000000
mean       775.970538
std        881.869458
min          0.000000
25%        117.000000
50%        438.000000
75%       1129.000000
max       3771.000000
Name: balance, dtype: float64

mean balance = 1514.9247
median balance = 542.0

outliers mean balance = 8050.5447
outliers median balance = 6005.0
Sub1 Column count = 16
Sub1 Row count = 4210

w/o outliers mean balance = 775.9705
w/o outliers median balance = 438.0
Sub2 Column count = 16
Sub2 Row count = 37235

IQR: 1.0-3.0 = 2.0
```

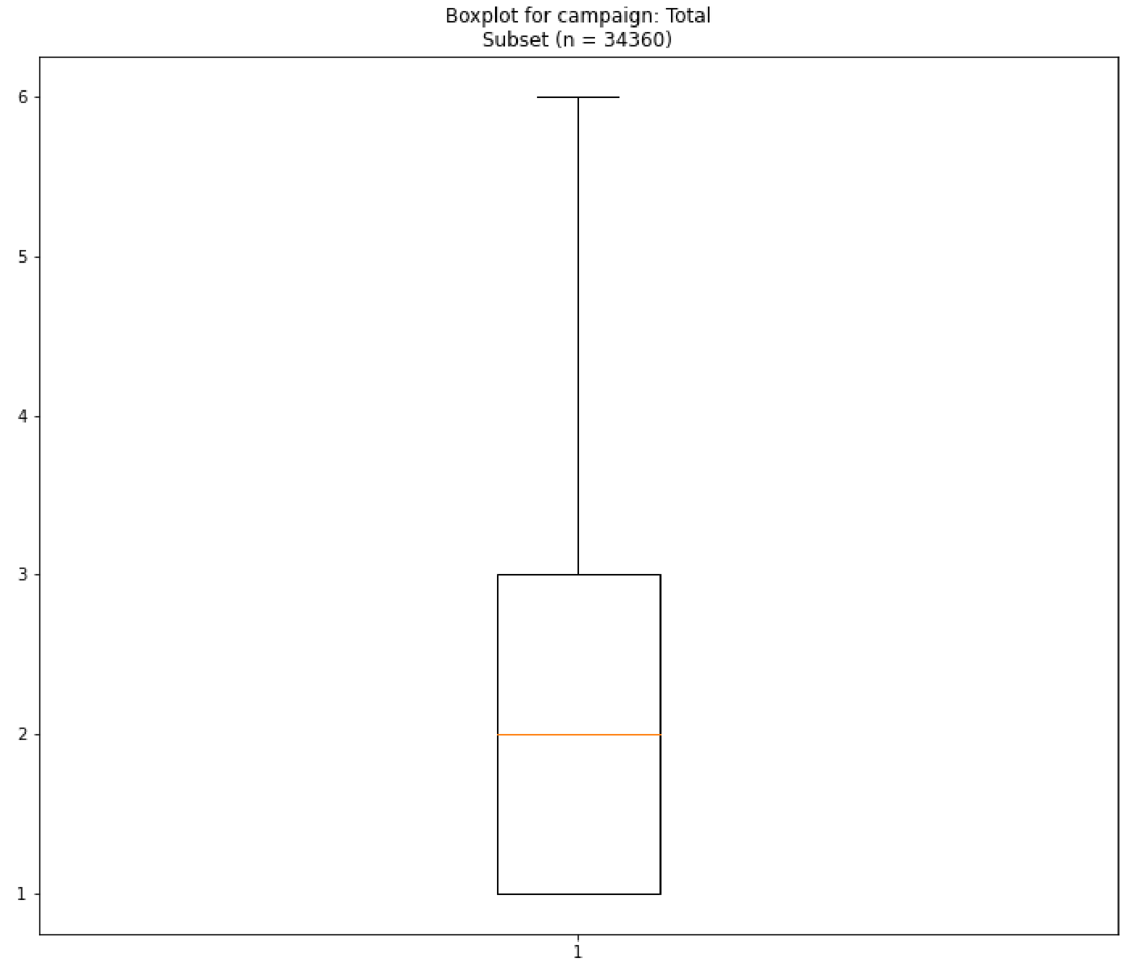



```
count    41445.000000
mean         2.749041
std          3.061182
min          1.000000
25%          1.000000
50%          2.000000
75%          3.000000
max         63.000000
Name: campaign, dtype: float64

 count    2785.000000
mean       11.389946
std         5.951749
min         7.000000
25%         8.000000
50%         9.000000
75%        13.000000
max        63.000000
```

```
Name: campaign, dtype: float64

 count    38660.000000
mean         2.126565
std          1.314740
min          1.000000
25%          1.000000
50%          2.000000
75%          3.000000
max          6.000000
Name: campaign, dtype: float64

mean campaign = 2.749
median campaign = 2.0

outliers mean campaign = 11.3899
outliers median campaign = 9.0
Sub1 Column count = 16
Sub1 Row count = 2785

w/o outliers mean campaign = 2.1266
w/o outliers median campaign = 2.0
Sub2 Column count = 16
Sub2 Row count = 38660
```


Boxplot for age: Total Subset
(n = 34360)
70
60
50
40
30
20
1


Boxplot for balance: Total
Subset (n = 34360)
3500
3000
2500
2000
1500
1000
500
0
1



Removal of Outliers Outliers were removed from `age`, `balance`, and `campaign` based on the function defined above that calculates the IQR for each variable, using a passed-in outlier threshold (e.g., IQR +/- 1 and a half times the IQR), and iterates to assign a flag of 0 or 1 to the total dataset based whether each record is within the outlier range for each applicable variable. This iteration (doing several separate assignments and only one elimination at the end) allows for minimizing how many records are removed from the final dataset, as there may be instances where one record may have outlier values in multiple features. Also, this method ensures that a distribution skew is not introduced through eliminating outliers for one variable, eliminating those records, then proceeding to the next variable–that method would mean that each time the mean may change for each subsequent variable in the (arbitrarily ordered) processing line. The net result is that instead of 7,463 records being eliminated (468 + 4,210 + 2,785), 7,085 were removed.

Initial correlation matrix

[17]:
```
# Generate initial correlation matrix
bank_df01_cols_lst02 = bank_df01.columns.values.tolist() # review column names
print(bank_df01_cols_lst02)
print(len(bank_df01_cols_lst02))
bank_df01.loc[:, bank_df01_cols_lst02].corr() # generate correlation matrix
```

```
['age', 'job', 'marital', 'education', 'default', 'balance', 'housing', 'loan',
'day', 'month', 'duration', 'campaign', 'poutcome', 'deposit',
'previous_contact']
15
```

[17]:

	age	balance	day	duration	campaign
age	1.000000	0.086771	-0.008327	-0.016820	0.039928
balance	0.086771	1.000000	0.021369	0.041036	-0.023945
day	-0.008327	0.021369	1.000000	-0.017647	0.101958
duration	-0.016820	0.041036	-0.017647	1.000000	-0.027469
campaign	0.039928	-0.023945	0.101958	-0.027469	1.000000
previous_contact	-0.018423	0.054869	-0.069389	-0.004868	-0.094269

	previous_contact
age	-0.018423
balance	0.054869
day	-0.069389
duration	-0.004868
campaign	-0.094269
previous_contact	1.000000

Interpretation of Correlation Matrix A correlation matrix is displayed above to show the independent relationship between the numerical variables in the dataset. From the correlation matrix, it can be observed that there is not a strong relationship between any them. The strongest observed relationship is between `campaign` and `day` (r = 0.1020), which is still considered to be very weak.

Scatterplots of numerical variables

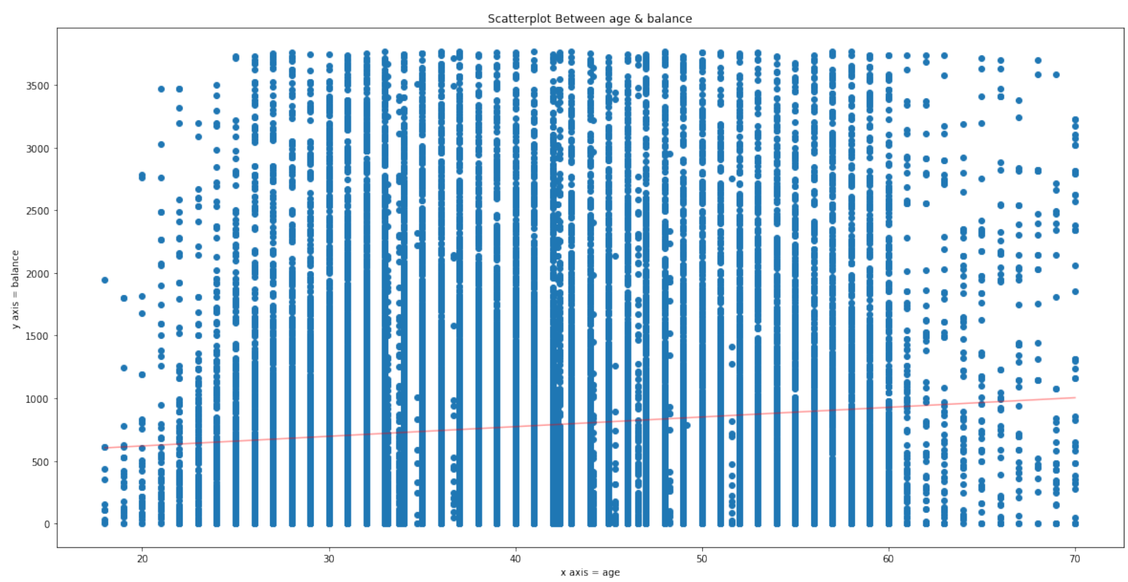
Define function: Produce scatterplots

[18]:
```
def scatr_vars(df, var_x, var_y):
    '''create function to print scatterplots comparing 2 vars'''
    y = df[var_y]
    x = df[var_x]
    x = sm.add_constant(x)
    lr_model = sm.OLS(y, x).fit()
    '''generate correlation coefficient & statsmodel regression table'''
  ␣
 ↪print('<<<<<-------------------------------------------------------------------->>>>>')
```

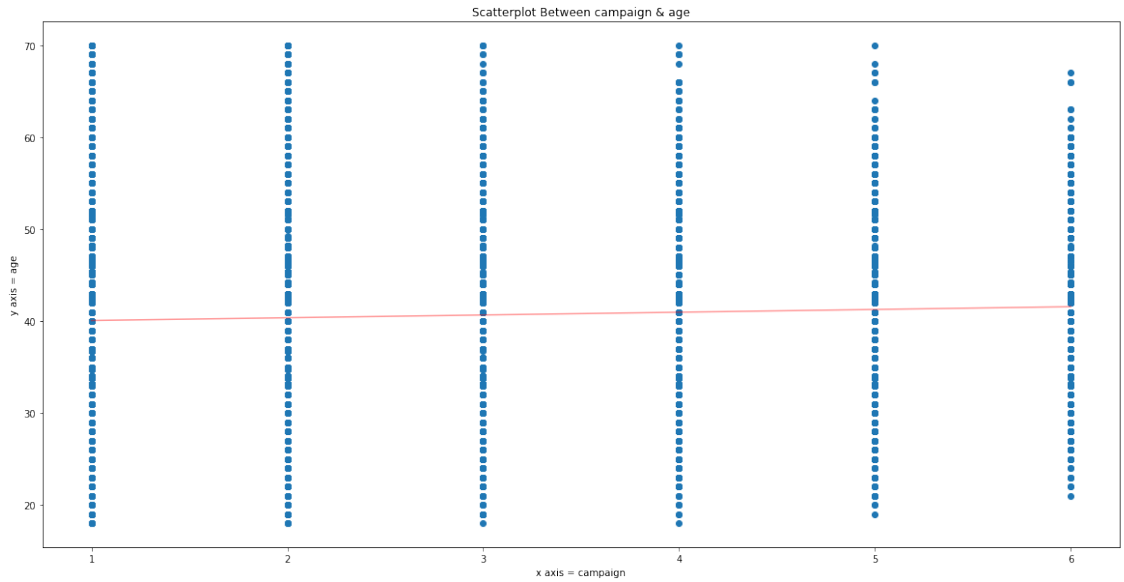
```
    print('r = {}'.format(round(df[var_x].corr(df[var_y]), 4))) # generate␣
↪correlation coefficient
    '''create true regression line'''
    x_prime = np.linspace(x[var_x].min(), x[var_x].max(), 100)
    x_prime = sm.add_constant(x_prime)
    y_hat = lr_model.predict(x_prime)
    '''plot x,y & regression line'''
    fig = plt.figure(figsize = (20 , 10))
    plt.scatter(df[var_x], df[var_y])
    plt.xlabel('x axis = {}'.format(var_x))
    plt.ylabel('y axis = {}'.format(var_y))
    plt.title('Scatterplot Between {} & {}'.format(var_x, var_y))
    plt.plot(x_prime[:,1], y_hat, 'red', alpha=.4)
    plt.show()
```

[19]:
```
# Run scatterplot fx
scatr_vars(bank_df01, 'age', 'balance')
scatr_vars(bank_df01, 'campaign', 'age')
scatr_vars(bank_df01, 'campaign', 'balance')
```

```
<<<<<------------------------------------------------------------------->>>>>
r = 0.0868
```

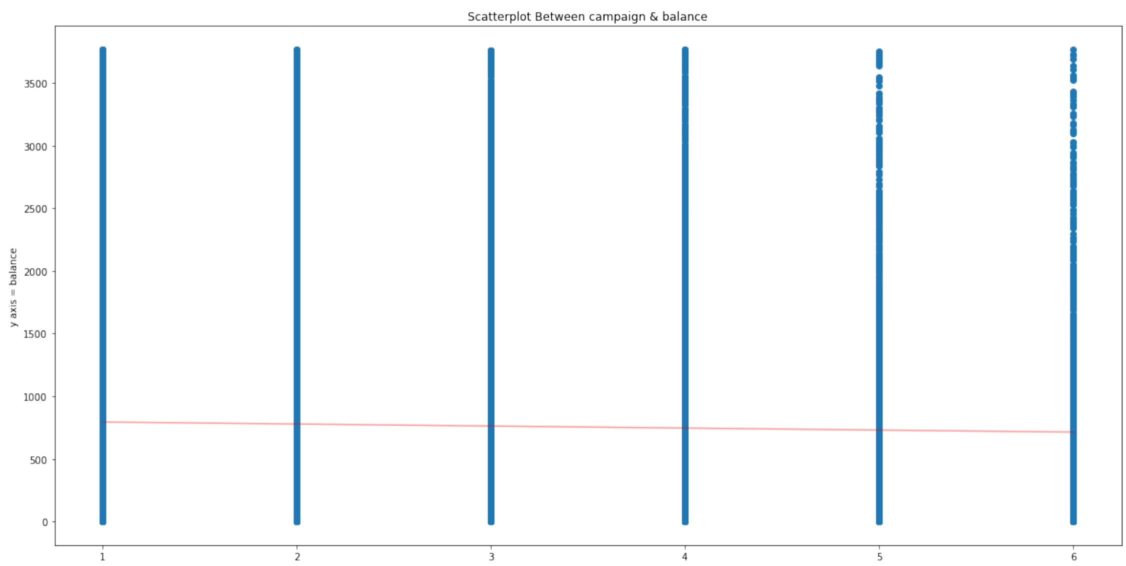



```
<<<<<------------------------------------------------------------------->>>>>
r = 0.0399
```



<<<<<--->>>>>

`r = -0.0239`



Interpretation of scatterplots From reviewing the above scatterplot graphs and along with the outputed correlation coefficient, none of the "natural" numeric varibales (`age`, `balance`, `campaign`) seem to have a linear relationship with any other. The highest is a very weak positive correlation between `age` and `balance` ($r = .0868$). Note, these plots were done using the cleaned and transformed dataset ($n = 34{,}360$).

Define function: Discretize categorical feature values

[20]:
```
# Define functions to convert categorical values to discrete numerical
col_arr_str = '_arr' # suffix to add to col names for array of cat variables

def pcat_list(df, col):
    '''define function to convert cat varibales to discrete numeric for␣
↪correlation'''
    cat_var_col = col
    cat_var_uniq = df[cat_var_col].unique()
    cat_len = len(cat_var_uniq)
    int_start = 0
    cat_dict = {}
    cat_array = np.eye(cat_len, dtype=int)
    for i in cat_var_uniq:
        cat_dict[i] = cat_array[int_start]
        col_name01 = col + '_der'
        col_name02 = col + col_arr_str
        df.loc[(df[col].isin(cat_dict.keys())), col_name01] = df[col]
        int_start += 1
        df[col_name02] = df[col_name01].map(cat_dict)
    return col_name02


def pcat_df(df, col):
    '''define function to map cat varibales to discrete numeric'''
    cat_var_col = col
    cat_var_uniq = df[cat_var_col].unique()
    cat_len = len(cat_var_uniq)
    int_start = 0
    cat_dict = {}
    cat_array = np.eye(cat_len, dtype=int)
    for i in cat_var_uniq:
        cat_dict[i] = int_start
        col_name01 = col + '_der02'
        col_name02 = col + '_num_map'
        df.loc[(df[col].isin(cat_dict.keys())), col_name01] = df[col]
        int_start += 1
        df[col_name02] = df[col_name01].map(cat_dict)
    print(cat_dict)
    return df


def corr_vars(df, num_list=[], cat_list=[], intract={}):
    '''define function to output a correlation matrix for selected num & cat␣
↪vars'''
    df_sub01 = df[num_list]
    cat_list_arr = []
```

```
    col_names01 = []
    col_names02 = []
    col_names03 = []
    merg_list_cont01 = []
    merg_list_cont02 = []
    merg_list_cont03 = []
    col_dict01 = {}
    last_col_list01 = []
    last_col_list02 = []
    '''loop to accumlate cat cols list'''
    for i in cat_list: # iterable in basic list of cat vars
        cat_list_arr.append(i + col_arr_str) # save col name + suffix to list
        col_names02 = list(df[i].unique()) # set list with unique values from␣
↪cat vars cols
        col_names04 = [] # initalize list
        '''loop to add prefix to cat vals'''
        for m in col_names02: # iterabale in list of unique cat vals
            cat_full_name = str(i) + '_' + str(m)
            col_names04.append(cat_full_name)
        for b in intract:
            d_val = intract[b]
            for z in d_val:
                col_names05 = [] # initalize list
                if i == z:
                    for u in col_names04:
                        col_names05.append(str(b) + '_' + str(u) + '_in')
                    merg_list_cont01.extend(col_names04)
                else:
                    pass
                merg_list_cont02.extend(col_names05)
                last_col_list02.extend(col_names05[-1:]) # keep track of cols␣
↪at the end after each pass
                merged_list = [(merg_list_cont02[i], merg_list_cont01[i]) for i␣
↪in range(0, len(merg_list_cont02))]
                col_dict01[b] = merged_list
        col_names03.append(col_names04) # save list of col + cat vals to list
        col_names01.append(col_names02)
    int_start = 0
    '''loop to extend cat vars to new cols'''
    for i in cat_list_arr: # iterable in new cols list
        df02 = df[i].apply(pd.Series) # extend the vals to new cols
        num_list.extend(col_names03[int_start])
        df_sub01 = pd.concat([df_sub01, df02], axis=1) # splice new ext cols to␣
↪orig cols
        df_sub01.columns = num_list
        last_col_list01.extend(num_list[-1:]) # keep track of cols at the end␣
↪after each pass
```

```
        int_start += 1
    '''add new cols for vars w/ interactions'''
    for s in col_dict01:
        d_val02 = col_dict01[s]
        for t, w in d_val02:
            df_sub01.loc[:, t] = df_sub01[s] * df_sub01[w]
    df_sub02 = df_sub01.drop(last_col_list01, axis=1) # drop based on c-1 cats
    df_sub02 = df_sub02.drop(last_col_list02, axis=1) # drop based on c-1 cats
    col_names = list(df_sub01.columns)
    print(f'\nColumn Names:\n{col_names}')

    return df_sub01.loc[:, col_names].corr(), df_sub01, df_sub02
```

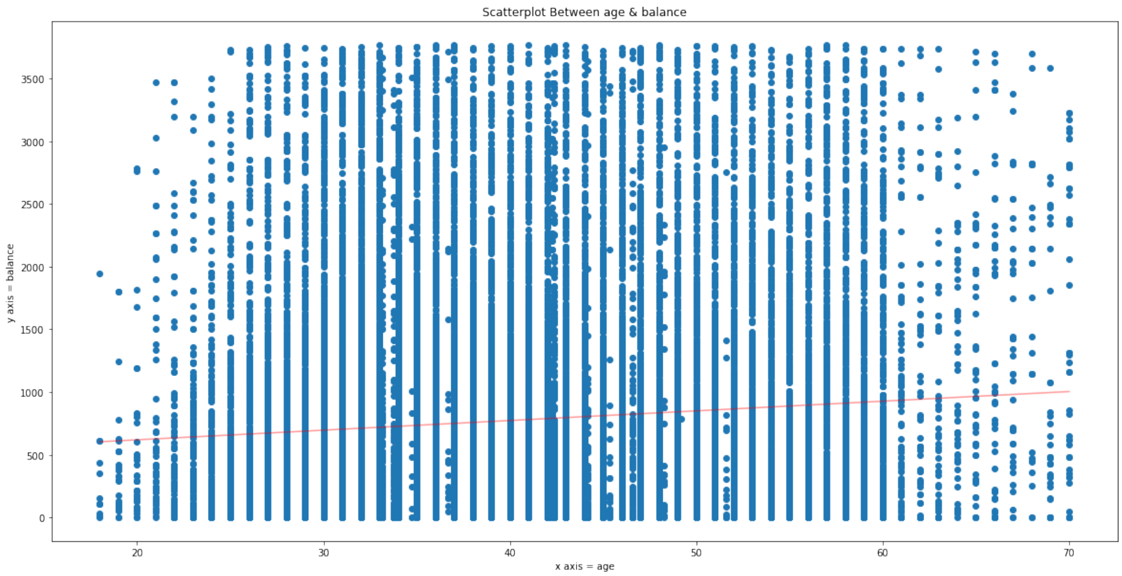
Key variable definitions

Variables for predicting deposit values

[21]:
```
# Load variable lists for functions & analysis
y_cor_vars_m2 = 'deposit'

num_vars_process_m2_01 = [y_cor_vars_m2, 'age', 'balance', 'campaign',␣
 ↪'previous_contact'] # dependent variable & core numerical variables

cat_vars_process_m2_01 = ['job', 'marital', 'education', 'default', 'housing',␣
 ↪'loan', 'day', 'month'] # relevant caterigal varaibles for df02
cat_vars_process_m2_02 = [y_cor_vars_m2, 'job', 'marital', 'education',␣
 ↪'default', 'housing', 'loan', 'day', 'month'] # relevant caterigal varaibles␣
 ↪for df06
```

[22]:
```
# Run scatterplot fx
scatr_vars(bank_df01, 'age', 'balance')
```

```
<<<<<------------------------------------------------------------------->>>>>
r = 0.0868
```



Subset Descriptive statistics

[23]:
```
bank_df01.describe()
```

[23]:

	age	balance	day	duration	campaign
count	34360.000000	34360.000000	34360.000000	34360.000000	34360.000000
mean	40.395460	775.747875	15.408615	261.549447	2.130850
std	9.908841	879.924490	8.265127	256.529050	1.315766
min	18.000000	0.000000	1.000000	0.000000	1.000000
25%	33.000000	119.000000	8.000000	107.000000	1.000000
50%	39.000000	439.000000	15.000000	184.000000	2.000000
75%	48.000000	1128.000000	21.000000	322.000000	3.000000
max	70.000000	3770.000000	31.000000	3881.000000	6.000000

\

	previous_contact
count	34360.000000
mean	0.191473
std	0.393466
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

Dataset branching

[24]:
```
bank_df22 = bank_df01.copy().dropna()

bank_df22_len01 = len(bank_df22)

print(f'Original N = {bank_df01_len01}\nNew n = {bank_df22_len01}\n% Loss =␣
 ↪{round((1-(bank_df22_len01/bank_df01_len01))*100, 2)}%')
```

```
Original N = 45211
New n = 34360
% Loss = 24.0%
```

Final explanation of record cleaning & trimming The final percent loss of records was 24%, reducing the count from 45,211 to 34,360. Considering the large N that was started with, the fact that three variables had missing values, most of the numerical variables had large amounts of data points +/- 1.5 time the IQR, and the remaining count is still large, it has been decided that the percent loss is acceptable for the analyses to follow.

[25]:
```
# Dataframe for consolidate correlation of cat vars
for i in cat_vars_process_m2_02:
    bank_df26 = pcat_df(bank_df22, i)

bank_df28 = bank_df26.copy().dropna()

print(bank_df26.head(10))
print(bank_df28.head(10))
```

```
{'no': 0, 'yes': 1}
{'management': 0, 'technician': 1, 'entrepreneur': 2, 'blue-collar': 3,
'student': 4, 'retired': 5, 'admin.': 6, 'services': 7, 'self-employed': 8,
'unemployed': 9, 'housemaid': 10}
{'married': 0, 'single': 1, 'divorced': 2}
{'tertiary': 0, 'secondary': 1, 'primary': 2, 'unknown': 3}
{'no': 0, 'yes': 1}
{'yes': 0, 'no': 1}
{'no': 0, 'yes': 1}
{5: 0, 6: 1, 7: 2, 8: 3, 9: 4, 12: 5, 13: 6, 14: 7, 15: 8, 16: 9, 19: 10, 20:
11, 21: 12, 23: 13, 26: 14, 27: 15, 28: 16, 29: 17, 30: 18, 2: 19, 3: 20, 4: 21,
11: 22, 17: 23, 18: 24, 24: 25, 25: 26, 1: 27, 10: 28, 22: 29, 31: 30}
{'may': 0, 'jun': 1, 'jul': 2, 'aug': 3, 'oct': 4, 'nov': 5, 'dec': 6, 'jan': 7,
'feb': 8, 'mar': 9, 'apr': 10, 'sep': 11}
    age           job  marital  education default  balance housing loan  day  \
0  58.0    management  married   tertiary      no     2143     yes   no    5
1  44.0    technician   single  secondary      no       29     yes   no    5
2  33.0  entrepreneur  married  secondary      no        2     yes  yes    5
3  47.0   blue-collar  married  secondary      no     1506     yes   no    5
4  33.0       student   single   tertiary      no        1      no   no    5
5  35.0    management  married   tertiary      no      231     yes   no    5
```

```
6  28.0    management    single   tertiary      no       447     yes  yes    5
7  42.0  entrepreneur  divorced   tertiary     yes         2     yes   no    5
8  58.0       retired   married    primary      no       121     yes   no    5
9  43.0    technician    single  secondary      no       593     yes   no    5

  month  …  default_der02  default_num_map housing_der02 housing_num_map  \
0   may  …             no                0           yes               0
1   may  …             no                0           yes               0
2   may  …             no                0           yes               0
3   may  …             no                0           yes               0
4   may  …             no                0            no               1
5   may  …             no                0           yes               0
6   may  …             no                0           yes               0
7   may  …            yes                1           yes               0
8   may  …             no                0           yes               0
9   may  …             no                0           yes               0

   loan_der02 loan_num_map  day_der02 day_num_map  month_der02 month_num_map
0          no            0          5           0          may             0
1          no            0          5           0          may             0
2         yes            1          5           0          may             0
3          no            0          5           0          may             0
4          no            0          5           0          may             0
5          no            0          5           0          may             0
6         yes            1          5           0          may             0
7          no            0          5           0          may             0
8          no            0          5           0          may             0
9          no            0          5           0          may             0

[10 rows x 33 columns]
    age           job   marital  education default  balance housing loan  day  \
0  58.0    management   married   tertiary      no     2143     yes   no    5
1  44.0    technician    single  secondary      no       29     yes   no    5
2  33.0  entrepreneur   married  secondary      no        2     yes  yes    5
3  47.0   blue-collar   married  secondary      no     1506     yes   no    5
4  33.0       student    single   tertiary      no        1      no   no    5
5  35.0    management   married   tertiary      no      231     yes   no    5
6  28.0    management    single   tertiary      no      447     yes  yes    5
7  42.0  entrepreneur  divorced   tertiary     yes        2     yes   no    5
8  58.0       retired   married    primary      no      121     yes   no    5
9  43.0    technician    single  secondary      no      593     yes   no    5

  month  …  default_der02  default_num_map housing_der02 housing_num_map  \
0   may  …             no                0           yes               0
1   may  …             no                0           yes               0
2   may  …             no                0           yes               0
3   may  …             no                0           yes               0
4   may  …             no                0            no               1
```

```
5    may  …              no              0            yes             0
6    may  …              no              0            yes             0
7    may  …             yes              1            yes             0
8    may  …              no              0            yes             0
9    may  …              no              0            yes             0

   loan_der02 loan_num_map  day_der02 day_num_map  month_der02 month_num_map
0          no            0          5           0          may             0
1          no            0          5           0          may             0
2         yes            1          5           0          may             0
3          no            0          5           0          may             0
4          no            0          5           0          may             0
5          no            0          5           0          may             0
6         yes            1          5           0          may             0
7          no            0          5           0          may             0
8          no            0          5           0          may             0
9          no            0          5           0          may             0

[10 rows x 33 columns]
```

Discretize key categorical features

Run correlation & regression functions

[26]:
```python
# Initialize new list
new_col_list02 = []

# Loop using function to populate new col list
for i in cat_vars_process_m2_01:
    nm02 = pcat_list(bank_df22, i)
    new_col_list02.append(nm02)

interact_dict02 = {'balance': ['job', 'education', 'marital'], 'age': ['job',␣
↪'education']}
full_corr01_m2, bank_df22, bank_df23  = corr_vars(bank_df22,␣
↪num_vars_process_m2_01, cat_vars_process_m2_01)
```

```
Column Names:
['deposit', 'age', 'balance', 'campaign', 'previous_contact', 'job_management',
'job_technician', 'job_entrepreneur', 'job_blue-collar', 'job_student',
'job_retired', 'job_admin.', 'job_services', 'job_self-employed',
'job_unemployed', 'job_housemaid', 'marital_married', 'marital_single',
'marital_divorced', 'education_tertiary', 'education_secondary',
'education_primary', 'education_unknown', 'default_no', 'default_yes',
'housing_yes', 'housing_no', 'loan_no', 'loan_yes', 'day_5', 'day_6', 'day_7',
'day_8', 'day_9', 'day_12', 'day_13', 'day_14', 'day_15', 'day_16', 'day_19',
'day_20', 'day_21', 'day_23', 'day_26', 'day_27', 'day_28', 'day_29', 'day_30',
'day_2', 'day_3', 'day_4', 'day_11', 'day_17', 'day_18', 'day_24', 'day_25',
```

```
'day_1', 'day_10', 'day_22', 'day_31', 'month_may', 'month_jun', 'month_jul',
'month_aug', 'month_oct', 'month_nov', 'month_dec', 'month_jan', 'month_feb',
'month_mar', 'month_apr', 'month_sep']
```

Updated correlation matrices

[27]:
```python
# Generate referential correlation matrix w/ full cat mapping (not c-1)
bank_df26_cols_lst = bank_df26.columns.values.tolist() # review column names
print(bank_df26_cols_lst)
print(len(bank_df26_cols_lst))
```

```
['age', 'job', 'marital', 'education', 'default', 'balance', 'housing', 'loan',
'day', 'month', 'duration', 'campaign', 'poutcome', 'deposit',
'previous_contact', 'deposit_der02', 'deposit_num_map', 'job_der02',
'job_num_map', 'marital_der02', 'marital_num_map', 'education_der02',
'education_num_map', 'default_der02', 'default_num_map', 'housing_der02',
'housing_num_map', 'loan_der02', 'loan_num_map', 'day_der02', 'day_num_map',
'month_der02', 'month_num_map', 'job_der', 'job_arr', 'marital_der',
'marital_arr', 'education_der', 'education_arr', 'default_der', 'default_arr',
'housing_der', 'housing_arr', 'loan_der', 'loan_arr', 'day_der', 'day_arr',
'month_der', 'month_arr']
49
```

[28]:
```python
full_corr01_m2
```

[28]:
```
                       age   balance  campaign  previous_contact  \
age               1.000000  0.086771  0.039928         -0.018423
balance           0.086771  1.000000 -0.023945          0.054869
campaign          0.039928 -0.023945  1.000000         -0.094269
previous_contact -0.018423  0.054869 -0.094269          1.000000
job_management   -0.013585  0.036044  0.021515          0.013501
…                      …         …         …                …
month_jan        -0.012415 -0.015060 -0.070767          0.074131
month_feb        -0.017316 -0.009259 -0.014478          0.104390
month_mar        -0.001360  0.036375 -0.021532          0.048408
month_apr        -0.043428  0.035793 -0.063416          0.135404
month_sep        -0.007698  0.024095 -0.043867          0.103887

                  job_management  job_technician  job_entrepreneur  \
age                    -0.013585       -0.052765          0.026035
balance                 0.036044       -0.011867         -0.009629
campaign                0.021515        0.014944          0.003896
previous_contact        0.013501       -0.004871         -0.009351
job_management          1.000000       -0.231325         -0.094912
…                            …              …                 …
month_jan               0.000205        0.006205         -0.005637
month_feb               0.000649       -0.005105          0.000989
month_mar               0.030859       -0.008182         -0.017427
```

```
month_apr           -0.016272       -0.017479         -0.013747
month_sep            0.031482       -0.011981         -0.008315

                 job_blue-collar  job_student  job_retired  …  month_jul  \
age                    -0.008816    -0.212464     0.361657  …   0.018221
balance                -0.011158     0.003434     0.044087  …  -0.063332
campaign                0.012246    -0.025015    -0.013343  …   0.070856
previous_contact       -0.023175     0.044724     0.000856  …  -0.157780
job_management         -0.271782    -0.080587    -0.107254  …  -0.009699
…                              …            …            …  …          …
month_jan              -0.038566     0.010859     0.009309  …  -0.078887
month_feb              -0.038060     0.022638     0.002692  …  -0.106459
month_mar              -0.041332     0.034221     0.017057  …  -0.042260
month_apr               0.032648     0.013513    -0.011944  …  -0.113567
month_sep              -0.040930     0.054017     0.026016  …  -0.047177

                 month_aug  month_oct  month_nov  month_dec  month_jan  \
age               0.078080   0.021641   0.032061   0.004592  -0.012415
balance          -0.022537   0.034468   0.092306   0.020507  -0.015060
campaign          0.176684  -0.061559  -0.078224  -0.009148  -0.070767
previous_contact -0.094027   0.092955   0.083190   0.062494   0.074131
job_management    0.100245   0.015060   0.045658   0.009073   0.000205
…                        …          …          …          …          …
month_jan        -0.074937  -0.024766  -0.058341  -0.013305   1.000000
month_feb        -0.101127  -0.033422  -0.078731  -0.017956  -0.049511
month_mar        -0.040144  -0.013267  -0.031253  -0.007128  -0.019654
month_apr        -0.107880  -0.035654  -0.083988  -0.019155  -0.052817
month_sep        -0.044814  -0.014811  -0.034889  -0.007957  -0.021941

                 month_feb  month_mar  month_apr  month_sep
age              -0.017316  -0.001360  -0.043428  -0.007698
balance          -0.009259   0.036375   0.035793   0.024095
campaign         -0.014478  -0.021532  -0.063416  -0.043867
previous_contact  0.104390   0.048408   0.135404   0.103887
job_management    0.000649   0.030859  -0.016272   0.031482
…                        …          …          …          …
month_jan        -0.049511  -0.019654  -0.052817  -0.021941
month_feb         1.000000  -0.026523  -0.071277  -0.029609
month_mar        -0.026523   1.000000  -0.028294  -0.011754
month_apr        -0.071277  -0.028294   1.000000  -0.031586
month_sep        -0.029609  -0.011754  -0.031586   1.000000

[71 rows x 71 columns]
```

Interpretation of Overall Correlation Matrix The Correlation matrix above was outputed to show the relationship between all of the variables in the dataset, after the imputation and alteration of the categorical variables and the missing values. Due to the discretized categorical variables, the correlation matrix loses its ability to inform on the relationship between two predictor variables. So, while the correlation table still provides a useful measure of the relationship between numerical variables (i.e. `age`, `balance`, `campaign`, …) it not longer provides a useful measurement for the relationship betweent he discretized categorical variables.

3. Data Analytics

Data Legend:
bank_df01–> original bank_marketing.csv imported as a pandas df
bank_df28–> discretized variables
bank_df22–> full discretized dataset with all dummy variables
bank_df23–> dummies; c-1

[29]:
```
bank_df42 = bank_df22.copy()
bank_df42 = pcat_df(bank_df42, 'deposit')
```

```
{'no': 0, 'yes': 1}
```

R & Python integration To get the bank dataset compatible with R, the `deposit` variable outputs have been changed from categorical to binary, with deposit "no" as 0 and deposit "yes" as 1, using the pcat_df function. A new dataframe has been created called bank_df_r from the bank_df42 dataframe. Then, the bank_df_r dataset can be read into R using the RPY2 package.

Develop R Logistic Regression Models

[30]:
```
bank_df_r = bank_df42
print(bank_df_r.head(5))
```

```
  deposit   age  balance  campaign  previous_contact  job_management  \
0      no  58.0     2143         1                 0               1
1      no  44.0       29         1                 0               0
2      no  33.0        2         1                 0               0
3      no  47.0     1506         1                 0               0
4      no  33.0        1         1                 0               0

   job_technician  job_entrepreneur  job_blue-collar  job_student  …  \
0               0                 0                0            0  …
1               1                 0                0            0  …
2               0                 1                0            0  …
3               0                 0                1            0  …
4               0                 0                0            1  …

   month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
0          0          0          0          0          0          0
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
```

```
   month_apr  month_sep  deposit_der02  deposit_num_map
0          0          0             no                0
1          0          0             no                0
2          0          0             no                0
3          0          0             no                0
4          0          0             no                0

[5 rows x 74 columns]
```

[31]:
```
# Calculate ratio of yes/no in deposit var
yes_len = len(bank_df42.loc[(bank_df42['deposit'] == 'yes'), :])
no_len = len(bank_df42.loc[(bank_df42['deposit'] == 'no'), :])
full_len = len(bank_df42)
print(f'Subset len for "yes" = {yes_len}')
print(f'Subset len for "no" = {no_len}')
print(f'Subset len for full dataset = {full_len}')
print(f'yes % = {round((yes_len/full_len)*100, 4)}%')
print(f'no % = {round((no_len/full_len)*100, 4)}%')
```

```
Subset len for "yes" = 4138
Subset len for "no" = 30222
Subset len for full dataset = 34360
yes % = 12.0431%
no % = 87.9569%
```

[32]:
```
# Review value count of dependent var
print(len(bank_df42.loc[(bank_df42[y_cor_vars_m2] == 'yes'), :]))
print(len(bank_df42.loc[(bank_df42[y_cor_vars_m2] != 'yes'), :]))

# Subset default set
bank_df42c = bank_df42.loc[(bank_df42[y_cor_vars_m2] == 'yes'), :]
bank_df42d = bank_df42.loc[(bank_df42[y_cor_vars_m2] != 'yes'), :]
frac_target_42 = .4
n01_m1df42_sample = int((len(bank_df42c) / frac_target_42)) - len(bank_df42c)
bank_df42e = bank_df42d.sample(n = n01_m1df42_sample)
bank_df42f = pd.concat([bank_df42c, bank_df42e])
print(f'\nbank_df42f\nlen = {len(bank_df42f)}\n\ndf:\n{bank_df42f.head(5)}')

bank_df_r2 = bank_df42f
print(bank_df_r2.head(5))
```

```
4138
30222

bank_df42f
len = 10345

df:
    deposit   age  balance  campaign  previous_contact  job_management  \
```

```
74       yes  59.0     2343         1                0              0
77       yes  56.0       45         1                0              0
78       yes  41.0     1270         1                0              0
115      yes  55.0     2476         1                0              0
152      yes  54.0      184         2                0              0

     job_technician  job_entrepreneur  job_blue-collar  job_student  …  \
74                0                 0                0            0  …
77                0                 0                0            0  …
78                1                 0                0            0  …
115               0                 0                0            0  …
152               0                 0                0            0  …

     month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
74           0          0          0          0          0          0
77           0          0          0          0          0          0
78           0          0          0          0          0          0
115          0          0          0          0          0          0
152          0          0          0          0          0          0

     month_apr  month_sep  deposit_der02  deposit_num_map
74           0          0            yes                1
77           0          0            yes                1
78           0          0            yes                1
115          0          0            yes                1
152          0          0            yes                1

[5 rows x 74 columns]
    deposit   age  balance  campaign  previous_contact  job_management  \
74       yes  59.0     2343         1                0              0
77       yes  56.0       45         1                0              0
78       yes  41.0     1270         1                0              0
115      yes  55.0     2476         1                0              0
152      yes  54.0      184         2                0              0

     job_technician  job_entrepreneur  job_blue-collar  job_student  …  \
74                0                 0                0            0  …
77                0                 0                0            0  …
78                1                 0                0            0  …
115               0                 0                0            0  …
152               0                 0                0            0  …

     month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
74           0          0          0          0          0          0
77           0          0          0          0          0          0
78           0          0          0          0          0          0
115          0          0          0          0          0          0
152          0          0          0          0          0          0
```

```
     month_apr  month_sep  deposit_der02  deposit_num_map
74           0          0            yes                1
77           0          0            yes                1
78           0          0            yes                1
115          0          0            yes                1
152          0          0            yes                1

[5 rows x 74 columns]
```

[33]:

```
r_df = robjects.conversion.py2rpy(bank_df_r)
r_df2 = robjects.conversion.py2rpy(bank_df_r2)
```

[34]:

```
%R -i r_df
%R head(r_df)
```

[34]:

```
  deposit   age  balance  campaign  previous_contact  job_management  \
0      no  58.0     2143         1                 0               1
1      no  44.0       29         1                 0               0
2      no  33.0        2         1                 0               0
3      no  47.0     1506         1                 0               0
4      no  33.0        1         1                 0               0
5      no  35.0      231         1                 0               1

   job_technician  job_entrepreneur  job_blue-collar  job_student  …  \
0               0                 0                0            0  …
1               1                 0                0            0  …
2               0                 1                0            0  …
3               0                 0                1            0  …
4               0                 0                0            1  …
5               0                 0                0            0  …

   month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
0          0          0          0          0          0          0
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0

   month_apr  month_sep  deposit_der02  deposit_num_map
0          0          0             no                0
1          0          0             no                0
2          0          0             no                0
3          0          0             no                0
4          0          0             no                0
5          0          0             no                0
```

```
[6 rows x 74 columns]
```

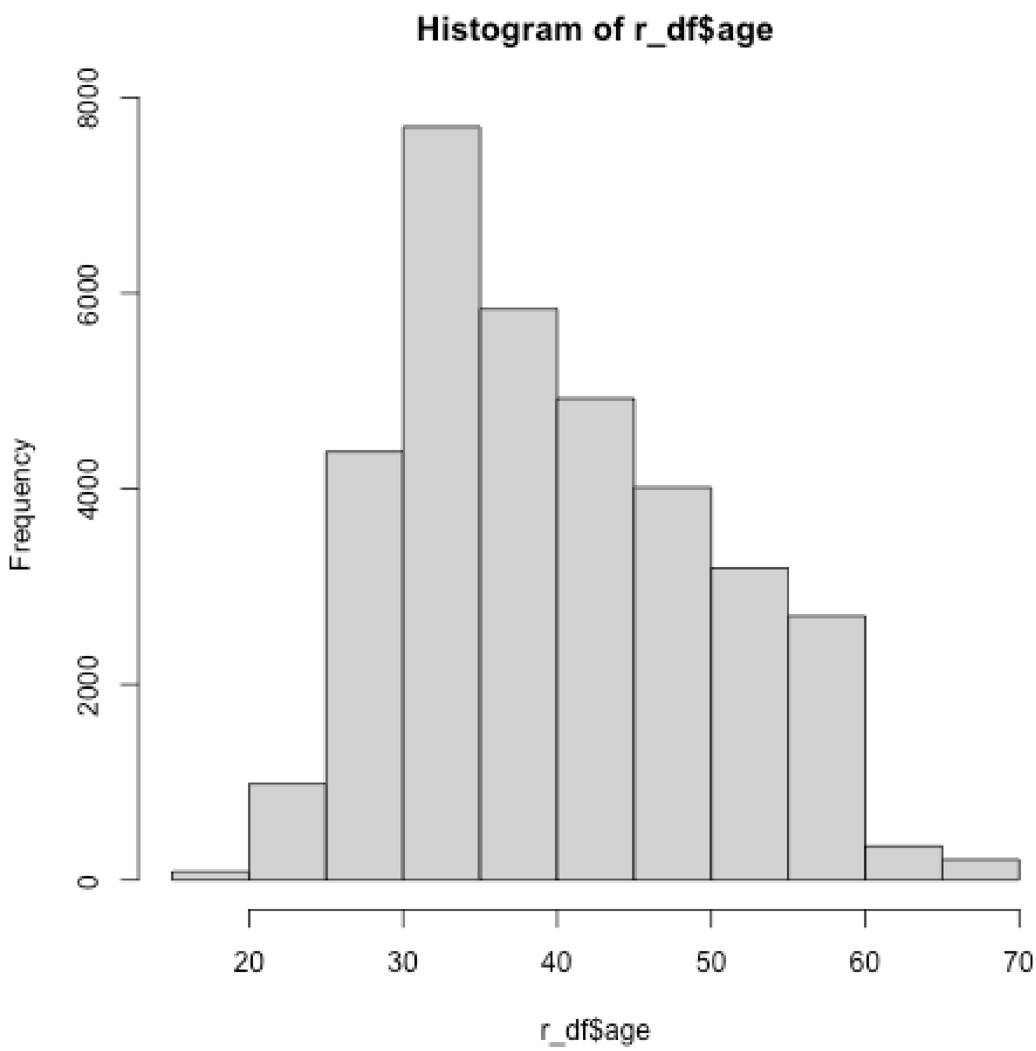
[35]:
```
%R -i r_df2
%R (head(r_df2))
```

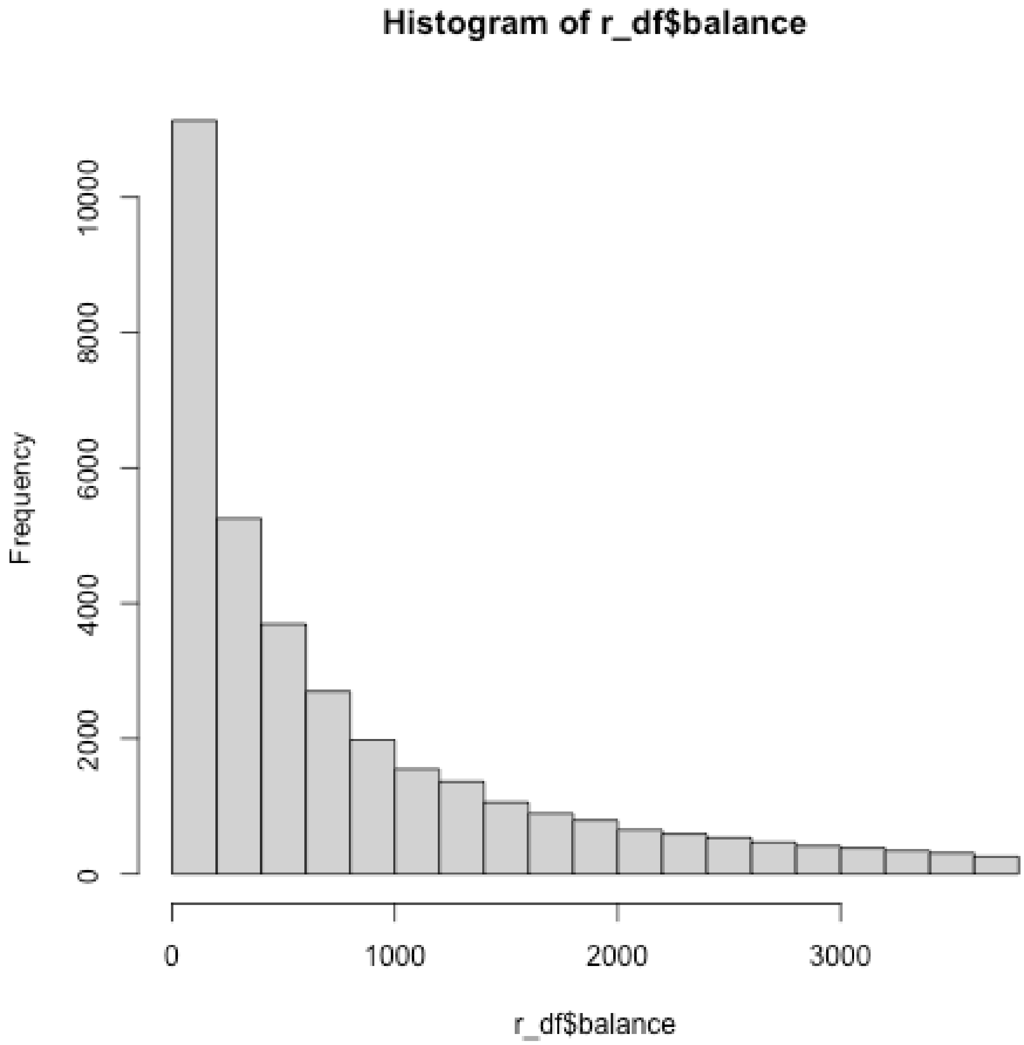
[35]:
```
    deposit   age  balance  campaign  previous_contact  job_management  \
74      yes  59.0     2343         1                 0               0
77      yes  56.0       45         1                 0               0
78      yes  41.0     1270         1                 0               0
115     yes  55.0     2476         1                 0               0
152     yes  54.0      184         2                 0               0
248     yes  42.0        0         2                 0               1

     job_technician  job_entrepreneur  job_blue-collar  job_student  …  \
74                0                 0                0            0  …
77                0                 0                0            0  …
78                1                 0                0            0  …
115               0                 0                0            0  …
152               0                 0                0            0  …
248               0                 0                0            0  …

     month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
74           0          0          0          0          0          0
77           0          0          0          0          0          0
78           0          0          0          0          0          0
115          0          0          0          0          0          0
152          0          0          0          0          0          0
248          0          0          0          0          0          0

     month_apr  month_sep  deposit_der02  deposit_num_map
74           0          0            yes                1
77           0          0            yes                1
78           0          0            yes                1
115          0          0            yes                1
152          0          0            yes                1
248          0          0            yes                1

[6 rows x 74 columns]
```

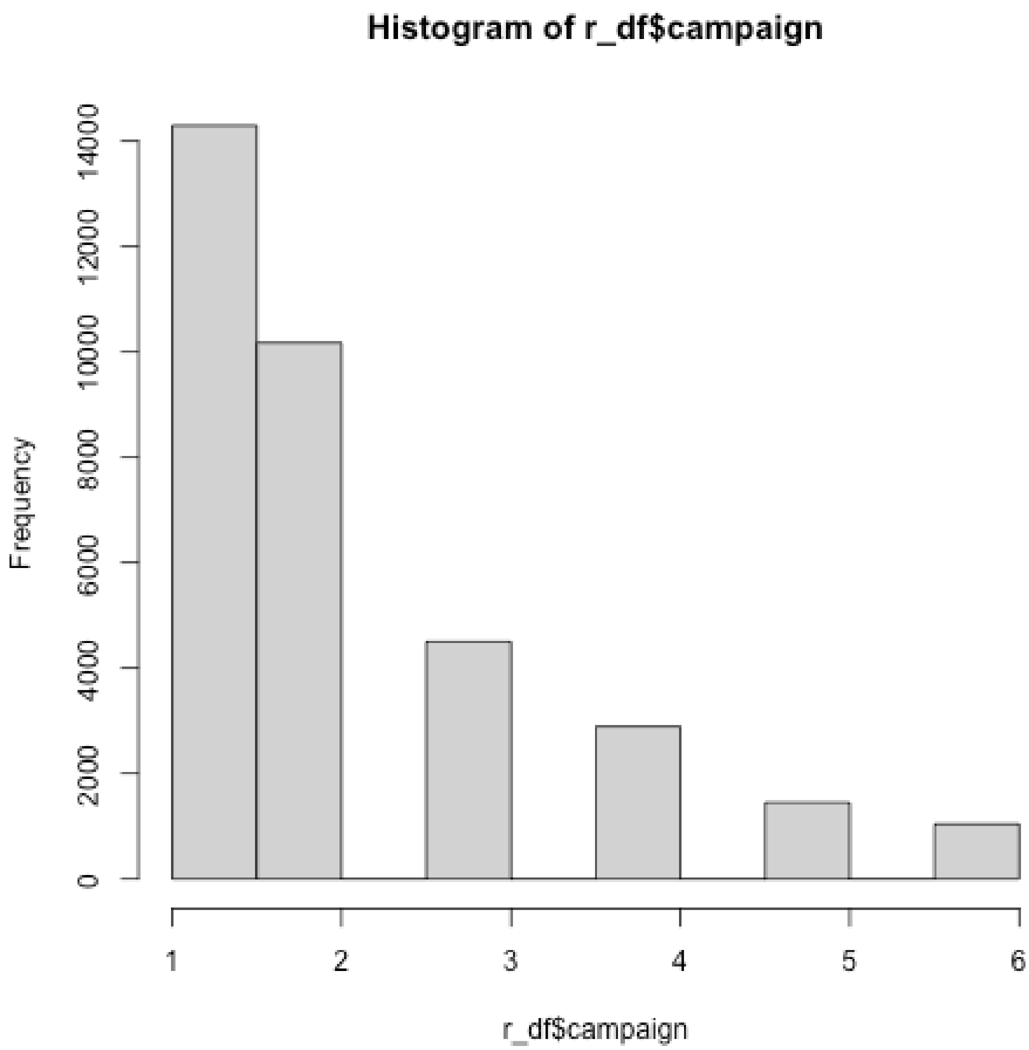
Using RPY2 magic function After creating the bank_df_r dataframe, the robjects.conversion command was used to make the python bank_df_r dataframe into an R compatible dataframe, now called r_df. After the conversion, the dataframe can now be read into R using cell magic (i.e., %R–denoting R codeblock). The r_df dataframe has been printed in R to double check that the file has been read in correctly, with all the variables and outputs as a table, not an array.

[36]:

```
%R hist(r_df$age)
%R hist(r_df$balance)
%R hist(r_df$campaign)
```




Histogram of r_df$balance
Frequency
0
2000
4000
6000
8000
10000
0
1000
2000
3000
r_df$balance




```
[36]: <rpy2.robjects.vectors.ListVector object at 0x7f9070df7cc0> [RTYPES.VECSXP]
      R classes: ('histogram',)
      [Float…, IntSe…, Float…, Float…, StrSe…, BoolS…]
        breaks: <class 'rpy2.rinterface.FloatSexpVector'>
        <rpy2.rinterface.FloatSexpVector object at 0x7f906c9ea4c0> [RTYPES.REALSXP]
        counts: <class 'rpy2.rinterface.IntSexpVector'>
        <rpy2.rinterface.IntSexpVector object at 0x7f90716f7800> [RTYPES.INTSXP]
        density: <class 'rpy2.rinterface.FloatSexpVector'>
        <rpy2.rinterface.FloatSexpVector object at 0x7f90701a5300> [RTYPES.REALSXP]
        mids: <class 'rpy2.rinterface.FloatSexpVector'>
        <rpy2.rinterface.FloatSexpVector object at 0x7f90701a56c0> [RTYPES.REALSXP]
        xname: <class 'rpy2.rinterface_lib.sexp.StrSexpVector'>
        <rpy2.rinterface_lib.sexp.StrSexpVector object at 0x7f90701a57c0>
```

```
[RTYPES.STRSXP]
  equidist: <class 'rpy2.rinterface.BoolSexpVector'>
  <rpy2.rinterface.BoolSexpVector object at 0x7f90701a5d00> [RTYPES.LGLSXP]
```

Interpretation of histograms for numeric variables (age, balance, campaign) The histogram for age displays a slight right skew, so the dataset has a slightly larger number of younger individuals than older individuals. The histogram for balance displays a strong right skew, so the dataset has a much larger number of lower balances than higher balances. The histogram for campaign displays a right skew as well, so most of the campaign values cluster to the left. This means that of the individuals in the dataset, more of them were contacted less frequently during the campaign for the client.

[37]:
```
%R names(r_df)[names(r_df)=="job_blue-collar"] <- "job_blue_collar"
%R names(r_df)[names(r_df)=="job_self-employed"] <- "job_self_employed"
%R head(r_df)
```

[37]:
```
  deposit   age  balance  campaign  previous_contact  job_management  \
0      no  58.0     2143         1                 0               1
1      no  44.0       29         1                 0               0
2      no  33.0        2         1                 0               0
3      no  47.0     1506         1                 0               0
4      no  33.0        1         1                 0               0
5      no  35.0      231         1                 0               1

   job_technician  job_entrepreneur  job_blue_collar  job_student  …  \
0               0                 0                0            0  …
1               1                 0                0            0  …
2               0                 1                0            0  …
3               0                 0                1            0  …
4               0                 0                0            1  …
5               0                 0                0            0  …

   month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
0          0          0          0          0          0          0
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0

   month_apr  month_sep  deposit_der02  deposit_num_map
0          0          0             no                0
1          0          0             no                0
2          0          0             no                0
3          0          0             no                0
4          0          0             no                0
5          0          0             no                0
```

```
[6 rows x 74 columns]
```

[38]:
```
%R sample1 <- floor(0.70 * nrow(r_df))
%R set.seed(1234)
%R train_ind <- sample(seq_len(nrow(r_df)), size=sample1)
%R train_r_df <- r_df[train_ind, ]
%R test_r_df <- r_df[-train_ind, ]

%R sample2 <- floor(0.70 * nrow(r_df2))
%R set.seed(1234)
%R train_ind2 <- sample(seq_len(nrow(r_df2)), size=sample2)
%R train_r_df2 <- r_df2[train_ind2, ]
%R test_r_df2 <- r_df2[-train_ind2, ]
```

[38]:
```
      deposit        age  balance  campaign  previous_contact  job_management  \
77        yes  56.000000       45         1                 0               0
78        yes  41.000000     1270         1                 0               0
115       yes  55.000000     2476         1                 0               0
359       yes  56.000000      830         1                 0               1
361       yes  44.168138      545         1                 0               0
…         …          …      …         …                 …               …
14043      no  29.000000      652         1                 0               0
31482      no  32.000000       75         3                 1               0
17004      no  34.000000      265         2                 0               0
31701      no  31.000000     2882         2                 0               0
25563      no  31.000000      242         1                 0               0

       job_technician  job_entrepreneur  job_blue-collar  job_student  …  \
77                  0                 0                0            0  …
78                  1                 0                0            0  …
115                 0                 0                0            0  …
359                 0                 0                0            0  …
361                 0                 0                0            0  …
…                 …                 …                …            …  …
14043               0                 0                0            1  …
31482               1                 0                0            0  …
17004               0                 0                0            0  …
31701               0                 0                0            1  …
25563               0                 0                1            0  …

       month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
77             0          0          0          0          0          0
78             0          0          0          0          0          0
115            0          0          0          0          0          0
359            0          0          0          0          0          0
361            0          0          0          0          0          0
```

```
...         ...      ...      ...      ...      ...      ...
14043         0        0        0        0        0        0
31482         0        0        0        0        0        0
17004         0        0        0        0        0        0
31701         0        0        0        0        0        0
25563         0        0        0        1        0        0

       month_apr  month_sep  deposit_der02  deposit_num_map
77             0          0            yes                1
78             0          0            yes                1
115            0          0            yes                1
359            0          0            yes                1
361            0          0            yes                1
...          ...        ...            ...              ...
14043          0          0             no                0
31482          0          0             no                0
17004          0          0             no                0
31701          0          0             no                0
25563          0          0             no                0

[3104 rows x 74 columns]
```

[39]:
```
%R train_r_df
```

[39]:
```
      deposit        age  balance  campaign  previous_contact  job_management  \
18177      no  48.258467       59         2                 0               0
40648     yes  38.000000     1199         1                 1               1
21310      no  33.000000      310         2                 0               1
18155      no  31.000000     1836         2                 0               1
24354      no  45.000000      243         1                 1               0
...       ...        ...      ...       ...               ...             ...
2743       no  41.000000      348         2                 0               0
4612       no  31.000000     1599         1                 0               0
34454      no  34.000000     2542         1                 0               0
32036      no  45.000000     1033         1                 1               1
16176      no  35.000000       86         2                 0               0

       job_technician  job_entrepreneur  job_blue_collar  job_student  ...  \
18177               0                 0                0            0  ...
40648               0                 0                0            0  ...
21310               0                 0                0            0  ...
18155               0                 0                0            0  ...
24354               0                 0                1            0  ...
...               ...               ...              ...          ...  ...
2743                0                 0                1            0  ...
4612                0                 0                1            0  ...
34454               0                 0                1            0  ...
```

```
32036                0                  0                 0            0  …
16176                0                  0                 1            0  …

       month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
18177          0          0          0          0          0          0
40648          0          0          0          0          0          0
21310          0          0          0          0          0          0
18155          0          0          0          0          0          0
24354          0          1          0          0          0          0
…            …        …        …        …        …        …
2743           0          0          0          0          0          0
4612           0          0          0          0          0          0
34454          0          0          0          0          0          0
32036          0          0          0          0          0          0
16176          0          0          0          0          0          0

       month_apr  month_sep  deposit_der02  deposit_num_map
18177          0          0             no                0
40648          0          0            yes                1
21310          0          0             no                0
18155          0          0             no                0
24354          0          0             no                0
…            …        …             …               …
2743           0          0             no                0
4612           0          0             no                0
34454          0          0             no                0
32036          0          0             no                0
16176          0          0             no                0

[24052 rows x 74 columns]
```

[40]:
```
%R test_r_df
```

[40]:
```
      deposit   age  balance  campaign  previous_contact  job_management  \
3          no  47.0     1506         1                 0               0
4          no  33.0        1         1                 0               0
7          no  42.0        2         1                 0               0
8          no  58.0      121         1                 0               0
10         no  41.0      270         1                 0               0
…         …    …       …        …                 …              …
41429     yes  68.0     1146         1                 1               0
41437     yes  23.0      113         1                 0               0
41439     yes  25.0      505         2                 0               0
41440     yes  51.0      825         3                 0               0
41444      no  37.0     2971         2                 1               0

       job_technician  job_entrepreneur  job_blue_collar  job_student  …  \
```

```
3                    0                 0                 1            0  …
4                    0                 0                 0            1  …
7                    0                 1                 0            0  …
8                    0                 0                 0            0  …
10                   0                 0                 0            0  …
…                …                …                …              …  …
41429                0                 0                 0            0  …
41437                0                 0                 0            1  …
41439                1                 0                 0            0  …
41440                1                 0                 0            0  …
41444                0                 1                 0            0  …

       month_oct  month_nov  month_dec  month_jan  month_feb  month_mar  \
3              0          0          0          0          0          0
4              0          0          0          0          0          0
7              0          0          0          0          0          0
8              0          0          0          0          0          0
10             0          0          0          0          0          0
…            …        …        …        …        …        …
41429          0          1          0          0          0          0
41437          0          1          0          0          0          0
41439          0          1          0          0          0          0
41440          0          1          0          0          0          0
41444          0          1          0          0          0          0

       month_apr  month_sep  deposit_der02  deposit_num_map
3              0          0             no                0
4              0          0             no                0
7              0          0             no                0
8              0          0             no                0
10             0          0             no                0
…            …        …            …              …
41429          0          0            yes                1
41437          0          0            yes                1
41439          0          0            yes                1
41440          0          0            yes                1
41444          0          0             no                0

[10308 rows x 74 columns]
```

[41]:
```
%R test_df_colum_nums <- colnames(test_r_df)
%R print(test_df_colum_nums)
```

```
 [1] "deposit"            "age"                "balance"
 [4] "campaign"           "previous_contact"   "job_management"
 [7] "job_technician"     "job_entrepreneur"   "job_blue_collar"
[10] "job_student"        "job_retired"        "job_admin."
[13] "job_services"       "job_self_employed"  "job_unemployed"
```

```
[16] "job_housemaid"       "marital_married"     "marital_single"
[19] "marital_divorced"    "education_tertiary"  "education_secondary"
[22] "education_primary"   "education_unknown"   "default_no"
[25] "default_yes"         "housing_yes"         "housing_no"
[28] "loan_no"             "loan_yes"            "day_5"
[31] "day_6"               "day_7"               "day_8"
[34] "day_9"               "day_12"              "day_13"
[37] "day_14"              "day_15"              "day_16"
[40] "day_19"              "day_20"              "day_21"
[43] "day_23"              "day_26"              "day_27"
[46] "day_28"              "day_29"              "day_30"
[49] "day_2"               "day_3"               "day_4"
[52] "day_11"              "day_17"              "day_18"
[55] "day_24"              "day_25"              "day_1"
[58] "day_10"              "day_22"              "day_31"
[61] "month_may"           "month_jun"           "month_jul"
[64] "month_aug"           "month_oct"           "month_nov"
[67] "month_dec"           "month_jan"           "month_feb"
[70] "month_mar"           "month_apr"           "month_sep"
[73] "deposit_der02"       "deposit_num_map"
```

[41]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f907165e940> [RTYPES.STRSXP]
R classes: ('character',)
['deposit', 'age', 'balance', 'campaign', …, 'month_apr', 'month_sep',
'deposit_…, 'deposit_…]
```

Creating Train & Test Subsets First, the column names for job_blue-collar and job_self-employed have been changed to job_blue_collar and job_self_employed, using the tidyverse package in R. This was done to create uniformity in the names of the columns in the dataset, important for modeling the data later. Second, training and testing subsets were created from the r_df dataframe, using the caTools package. The r_df data was split 70/30 for the train_r_df (24,052 rows) and test_r_df (10,308 rows). The last step taken before modeling the deposit variable using logistic regressions was to print the test_df_r column names and corresponding column number. This step was taken to have reference to the columns used in the logistic regression models below.

Model 1: Simple Logistic Regression (deposit ~ previous_contact)

[42]:
```
%R log_r <- glm(deposit_num_map ~ previous_contact, data=train_r_df,␣
 ↪family=binomial)
%R print(log_r)
%R log_r_summary <- summary(log_r)
%R print(log_r_summary)
```

```
Call:  glm(formula = deposit_num_map ~ previous_contact, family = binomial,
    data = train_r_df)

Coefficients:
     (Intercept)  previous_contact
```

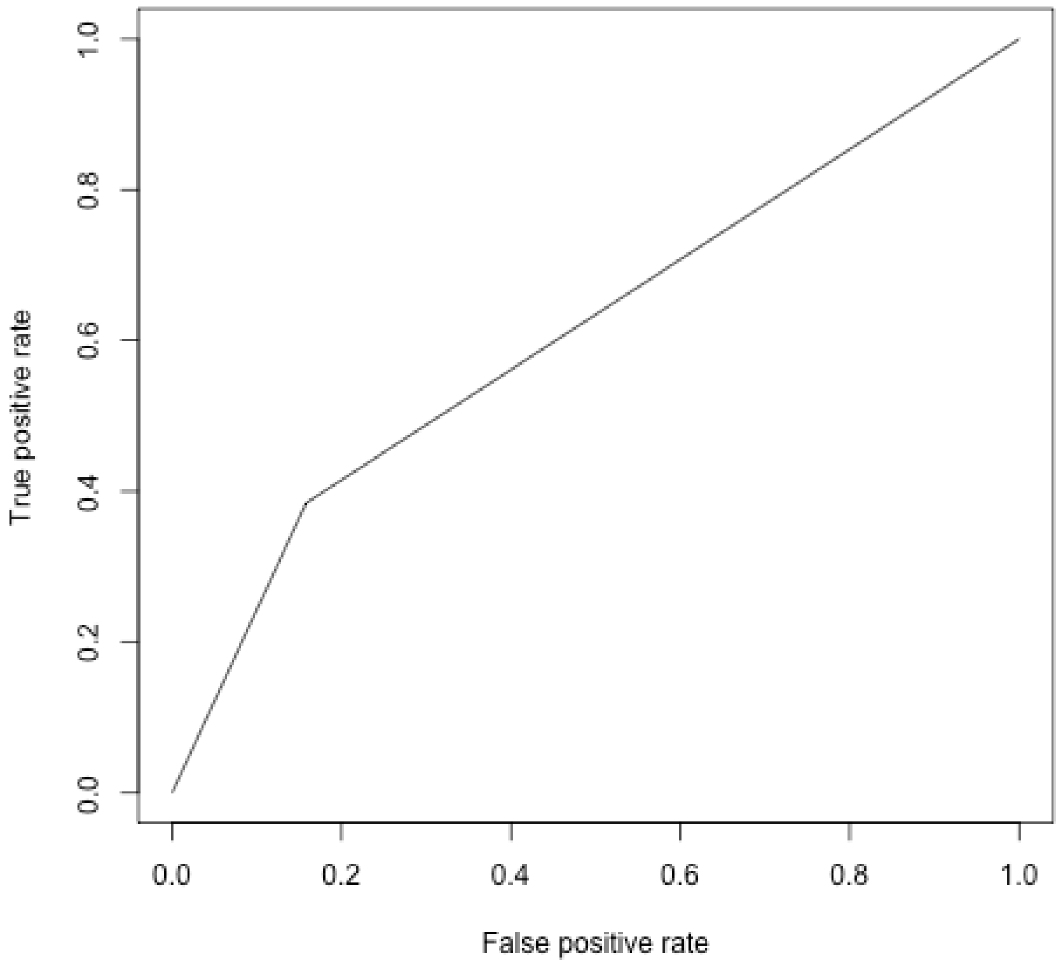
```
          -2.264             1.043

Degrees of Freedom: 24051 Total (i.e. Null);  24050 Residual
Null Deviance:      17650
Residual Deviance: 17100        AIC: 17110

Call:
glm(formula = deposit_num_map ~ previous_contact, family = binomial,
    data = train_r_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.7189  -0.4447  -0.4447  -0.4447   2.1740

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      -2.26421    0.02459  -92.06   <2e-16 ***
previous_contact  1.04288    0.04271   24.42   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 17654  on 24051  degrees of freedom
Residual deviance: 17102  on 24050  degrees of freedom
AIC: 17106

Number of Fisher Scoring iterations: 5
```

[42]:
```
<rpy2.robjects.vectors.ListVector object at 0x7f9070e15f00> [RTYPES.VECSXP]
R classes: ('summary.glm',)
[LangSexpV…, LangSexpV…, ListSexpV…, FloatSexp…, …, FloatSexp…,
IntSexpVe…, FloatSexp…, FloatSexp…]
  call: <class 'rpy2.robjects.language.LangVector'>
  Rlang( glm(formula = deposit_num_map ~ previous_contact, family = binomial,  )
  terms: <class 'rpy2.robjects.Formula'>
  <rpy2.robjects.Formula object at 0x7f9070baefc0> [RTYPES.LANGSXP]
R classes: ('terms', 'formula')
<rpy2.robjects.vectors.ListVector object at 0x7f9070e15f00> [RTYPES.VECSXP]
R classes: ('summary.glm',)
[LangSexpV…, LangSexpV…, ListSexpV…, FloatSexp…, …, FloatSexp…,
IntSexpVe…, FloatSexp…, FloatSexp…]
  deviance: <class 'numpy.ndarray'>
  array([17102.09827662])
…
  contrasts: <class 'numpy.ndarray'>
```

```
array([1.])
df.residual: <class 'numpy.ndarray'>
array([    2, 24050,     2], dtype=int32)
null.deviance: <class 'numpy.ndarray'>
array([[ 0.00060488, -0.00060488],
       [-0.00060488,  0.00182413]])
df.null: <class 'numpy.ndarray'>
array([[ 0.00060488, -0.00060488],
       [-0.00060488,  0.00182413]])
```

Model 1: Receiver Operator Curve & Area Under the Curve

[43]:
```
%R p <- predict(log_r, newdata=subset(test_r_df, select=c(5)), type="response")
%R pr <- prediction(p, test_r_df$deposit_num_map)
%R prf <- performance(pr, measure= "tpr", x.measure="fpr")
%R plot(prf)
%R auc <- performance(pr, measure="auc")
%R auc <- auc@y.values[[1]]
```



[43]: array([0.61308092])

Model 1 Accuracy Measurement

[44]:
```
%R log_r_test <- predict(log_r, newdata=subset(test_r_df, select=c(5)),␣
↪type="response")
%R log_r_test <- ifelse(log_r_test>0.5,1,0)
%R misclasificerror <- mean(log_r_test!=test_r_df$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror))
```

```
[1] "Accuracy 0.878637951105937"
```

[44]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f90702d8b00> [RTYPES.STRSXP]
R classes: ('character',)
```

```
['Accuracy 0.878637951105937']
```

Model 1 Interpretation Model 1 is a simple logistic regression where the R application glm method is used to determine deposit status (i.e., "yes" or "no") as the dependent variable values from `previous_contact` as the independent variables in a training subset (train_r_df) of the full trimmed and cleanded dataset (N = 34,360). The intercept was calculated to be -2.264 and the regression coefficient for `previous_contact` to be 1.043, with a statistically significant p-value <.001. This means that as the number of previous contacts increases, there is a positive increase in the log odds (and by extension probability) of the client subscribing to a term deposit. Model 1 indicates that `previous_contact` is significant in determining the deposit status. The receiver operating curve (ROC) was then plotted to calculate the area under the curve (AUC). The ROC curve plots the true positive rate against the false positive rate, with AUC values closer to 1 indicating a good predictive model. The AUC for model 1 is equal to 0.6131, so the area under the ROC covers about 61% of the rectangle, indicating that model 1 does not have great predictive ability. The accuracy for the model is also reported to measure how well the model developed using the training data performed in predicting values within the test_r_df subset. The accuracy for model 1 is equal to 0.8786, which is high but does not necessarily indicate a well-fitted model due to the low value of the AUC.
In summary, model 1's ability to predict deposit is not great, and this is most likely due to the fact that only predictor variable is included but there is more than one factor affecting if the client decides to subscribe to a term deposit.

Model 2: Simple Logistic Regression (deposit ~ campaign)

[45]:
```
%R log_r_2 <- glm(deposit_num_map ~ campaign, data=train_r_df, family=binomial)
%R print(log_r_2)

%R log_r_2_summary <- summary(log_r_2)
%R print(log_r_2_summary)
```

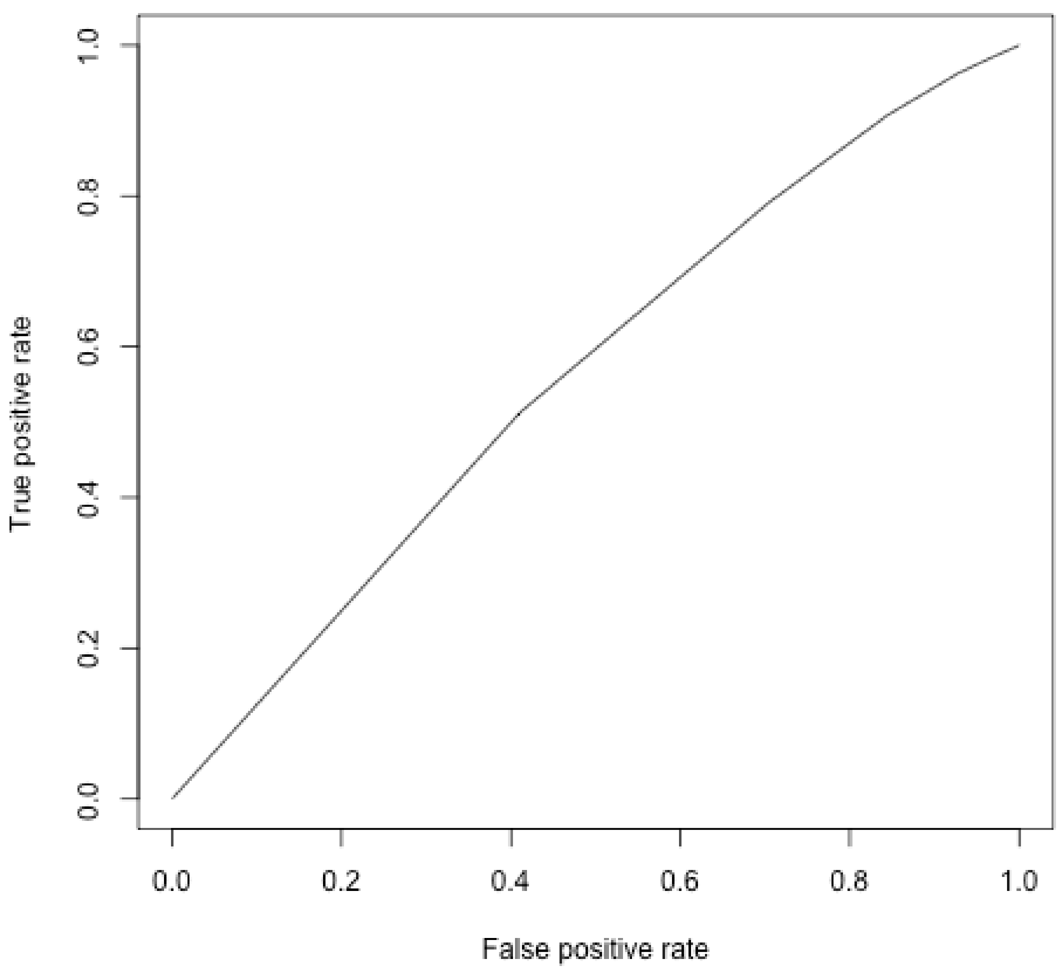
```
Call:  glm(formula = deposit_num_map ~ campaign, family = binomial,
    data = train_r_df)

Coefficients:
(Intercept)     campaign
    -1.6686      -0.1586

Degrees of Freedom: 24051 Total (i.e. Null);  24050 Residual
Null Deviance:      17650
Residual Deviance: 17560        AIC: 17560

Call:
glm(formula = deposit_num_map ~ campaign, family = binomial,
    data = train_r_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
```

```
-0.5462  -0.5462  -0.5072  -0.4365   2.3197

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.66857    0.03807 -43.831   <2e-16 ***
campaign    -0.15859    0.01670  -9.499   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 17654  on 24051  degrees of freedom
Residual deviance: 17556  on 24050  degrees of freedom
AIC: 17560

Number of Fisher Scoring iterations: 5
```

[45]:
```
<rpy2.robjects.vectors.ListVector object at 0x7f9070bb5bc0> [RTYPES.VECSXP]
R classes: ('summary.glm',)
[LangSexpV…, LangSexpV…, ListSexpV…, FloatSexp…, …, FloatSexp…,
IntSexpVe…, FloatSexp…, FloatSexp…]
  call: <class 'rpy2.robjects.language.LangVector'>
  Rlang( glm(formula = deposit_num_map ~ campaign, family = binomial,  )
  terms: <class 'rpy2.robjects.Formula'>
  <rpy2.robjects.Formula object at 0x7f9070bac400> [RTYPES.LANGSXP]
R classes: ('terms', 'formula')
<rpy2.robjects.vectors.ListVector object at 0x7f9070bb5bc0> [RTYPES.VECSXP]
R classes: ('summary.glm',)
[LangSexpV…, LangSexpV…, ListSexpV…, FloatSexp…, …, FloatSexp…,
IntSexpVe…, FloatSexp…, FloatSexp…]
  deviance: <class 'numpy.ndarray'>
  array([17556.16361575])
…
  contrasts: <class 'numpy.ndarray'>
  array([1.])
  df.residual: <class 'numpy.ndarray'>
  array([    2, 24050,     2], dtype=int32)
  null.deviance: <class 'numpy.ndarray'>
  array([[ 0.00144921, -0.00054205],
        [-0.00054205,  0.00027875]])
  df.null: <class 'numpy.ndarray'>
  array([[ 0.00144921, -0.00054205],
        [-0.00054205,  0.00027875]])
```

Model 2: Receiver Operator Curve & Area Under the Curve

[46]:
```
%R log_r_2_test <- predict(log_r_2, newdata=subset(test_r_df, select=c(4)),␣
↪type="response")
%R log_r_2_test <- ifelse(log_r_2_test>0.5,1,0)

%R p_log_r_2 <- predict(log_r_2, newdata=subset(test_r_df, select=c(4)),␣
↪type="response")
%R pr_log_r_2 <- prediction(p_log_r_2, test_r_df$deposit_num_map)
%R prf_log_r_2 <- performance(pr_log_r_2, measure= "tpr", x.measure="fpr")
%R plot(prf_log_r_2)

%R auc_log_r_2 <- performance(pr_log_r_2, measure="auc")
%R auc_log_r_2 <- auc_log_r_2@y.values[[1]]
```



[46]: array([0.56452694])

Model 2 Accuracy Measurement

[47]:
```
%R log_r_2_test <- predict(log_r_2, newdata=subset(test_r_df, select=c(4)),␣
 ↪type="response")
%R log_r_2_test <- ifelse(log_r_2_test>0.5,1,0)
%R misclasificerror_2 <- mean(log_r_2_test!=test_r_df$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror_2))
```

```
[1] "Accuracy 0.878637951105937"
```

[47]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f9097bb5900> [RTYPES.STRSXP]
R classes: ('character',)
['Accuracy 0.878637951105937']
```

Model 2 Interpretation Model 2 is a simple logistic regression where the R application glm method is used to determine deposit status (i.e., "yes" or "no") as the dependent variable values from `campaign` as the independent variable in a training subset (train_r_df) of the full trimmed and cleaned dataset (N = 34,360). The intercept was calculated to be -1.66857 and the regression coefficient for `campaign` to be -0.15859, with a statistically significant p-value <.001. This means that as the number of contacts performed during the campaign increases, there is a decrease in the log odds (and by extension probability) of the client subscribing to a term deposit. Model 2 indicates that `campaign` is significant in determining the deposit status. The ROC was then plotted to calculate the AUC. The AUC for model 2 is equal to 0.5645, so the area under the ROC covers about 56% of the rectangle, indicating that model 2 has a poor predictive ability. The accuracy of model 2 is equal to 0.8786, which is high but does not necessarily indicate a well-fitted model due to the low value of the AUC.
In summary, model 2's ability to predict deposit is poor, and this is most likely due to the fact that only one predictor variable is included in the regression equation, similar to model 1. This second simple logistic regression was performed to better understand which predictor variables have large affects on deposit status, by itself. This information will later be used in the multiple logistic regression models.

Model 3: Simple Logistic Regression (deposit ~ balance)

[48]:
```
%R log_r_3 <- glm(deposit_num_map ~ balance , data=train_r_df, family=binomial)
%R print(log_r_3)

%R log_r_3_summary <- summary(log_r_3)
%R print(log_r_3_summary)
```

```
Call:  glm(formula = deposit_num_map ~ balance, family = binomial, data =
train_r_df)

Coefficients:
(Intercept)      balance
 -2.2050094    0.0002494
```

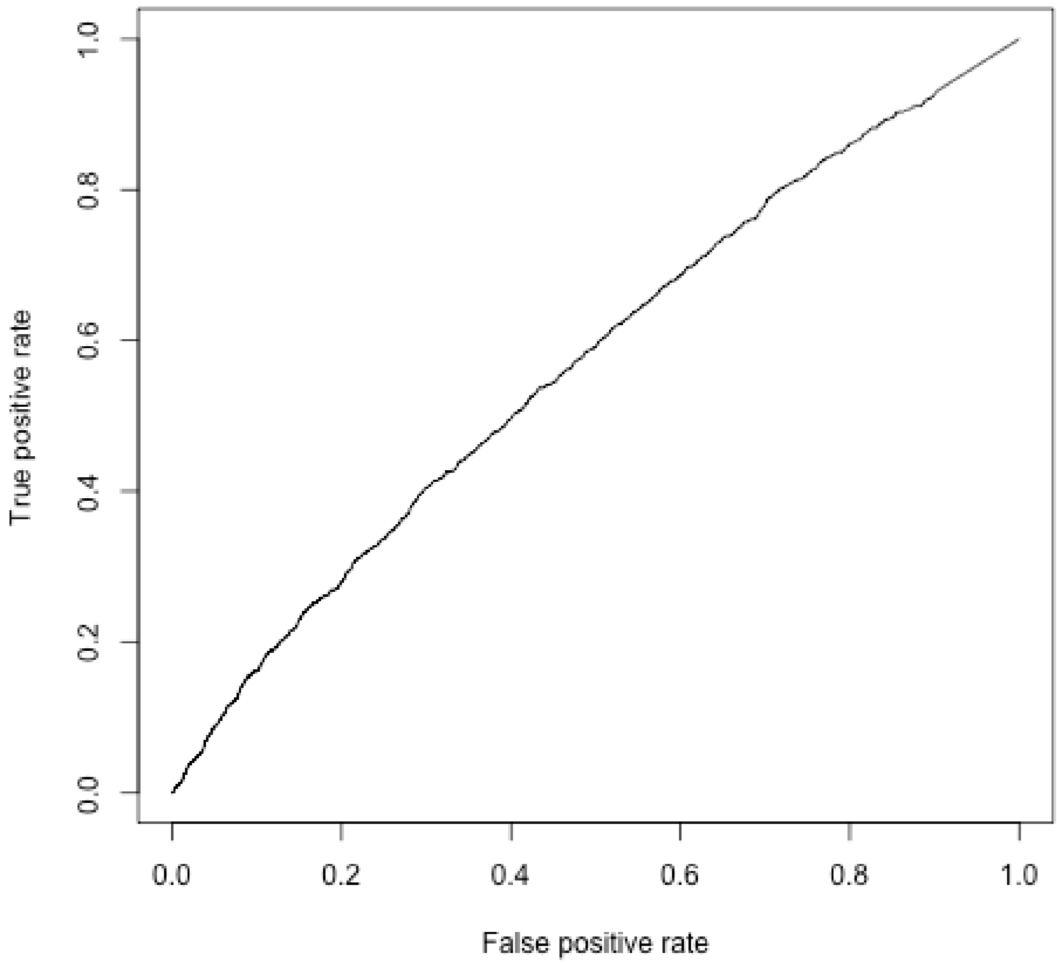
```
Degrees of Freedom: 24051 Total (i.e. Null);  24050 Residual
Null Deviance:      17650
Residual Deviance: 17510        AIC: 17520

Call:
glm(formula = deposit_num_map ~ balance, family = binomial, data = train_r_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.7052  -0.5113  -0.4750  -0.4586   2.1492

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.2050094  0.0275305  -80.09   <2e-16 ***
balance      0.0002494  0.0000204   12.22   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 17654  on 24051  degrees of freedom
Residual deviance: 17513  on 24050  degrees of freedom
AIC: 17517

Number of Fisher Scoring iterations: 4
```

[48]:
```
<rpy2.robjects.vectors.ListVector object at 0x7f9070bae580> [RTYPES.VECSXP]
R classes: ('summary.glm',)
[LangSexpV…, LangSexpV…, ListSexpV…, FloatSexp…, …, FloatSexp…,
IntSexpVe…, FloatSexp…, FloatSexp…]
  call: <class 'rpy2.robjects.language.LangVector'>
  Rlang( glm(formula = deposit_num_map ~ balance, family = binomial, data =
train_r_df) )
  terms: <class 'rpy2.robjects.Formula'>
  <rpy2.robjects.Formula object at 0x7f9070bae540> [RTYPES.LANGSXP]
R classes: ('terms', 'formula')
<rpy2.robjects.vectors.ListVector object at 0x7f9070bae580> [RTYPES.VECSXP]
R classes: ('summary.glm',)
[LangSexpV…, LangSexpV…, ListSexpV…, FloatSexp…, …, FloatSexp…,
IntSexpVe…, FloatSexp…, FloatSexp…]
  deviance: <class 'numpy.ndarray'>
  array([17512.89654839])
…
  contrasts: <class 'numpy.ndarray'>
  array([1.])
  df.residual: <class 'numpy.ndarray'>
```

```
array([    2, 24050,     2], dtype=int32)
null.deviance: <class 'numpy.ndarray'>
array([[ 7.57930217e-04, -3.88030421e-07],
       [-3.88030421e-07,  4.16161787e-10]])
df.null: <class 'numpy.ndarray'>
array([[ 7.57930217e-04, -3.88030421e-07],
       [-3.88030421e-07,  4.16161787e-10]])
```

Model 3: Receiver Operator Curve & Area Under the Curve

[49]:
```
%R log_r_3_test <- predict(log_r_3, newdata=subset(test_r_df, select=c(3)), type="response")
%R log_r_3_test <- ifelse(log_r_3_test>0.5,1,0)

%R p_log_r_3 <- predict(log_r_3, newdata=subset(test_r_df, select=c(3)), type="response")
%R pr_log_r_3 <- prediction(p_log_r_3, test_r_df$deposit_num_map)
%R prf_log_r_3 <- performance(pr_log_r_3, measure= "tpr", x.measure="fpr")
%R plot(prf_log_r_3)

%R auc_log_r_3 <- performance(pr_log_r_3, measure="auc")
%R auc_log_r_3 <- auc_log_r_3@y.values[[1]]
```



[49]: array([0.5701113])

Model 3 Accuracy Measurement

[50]:
```
%R log_r_3_test <- predict(log_r_3, newdata=subset(test_r_df, select=c(3)),␣
 ↪type="response")
%R log_r_3_test <- ifelse(log_r_3_test>0.5,1,0)
%R misclasificerror_3 <- mean(log_r_3_test!=test_r_df$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror_3))
```

```
[1] "Accuracy 0.878637951105937"
```

[50]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f907165a140> [RTYPES.STRSXP]
R classes: ('character',)
```

```
['Accuracy 0.878637951105937']
```

Model 3 Interpretation Model 3 is another simple logistic regression where the R application glm method is used to determine deposit status (i.e., "yes" or "no") as the dependent variable values from `balance` as the independent variable in a training subset (train_r_df) of the full trimmed and cleaned dataset (N = 34,360). The intercept was calculated to be -2.20501 and the regression coefficient for `balance` to be 0.00025, with statistically significant p-value <.001. This means that as the average yearly balance in Euros increases, there is a slight increase in the log odds (and by extension probability) of the client subscribing to a term deposit. Model 3 indicates that `balance` is significant in determining the deposit status. The ROC was then plotted to calculate the AUC. The AUC for model 3 is equal to 0.5701, so the area under the ROC covers about 57% of the rectangle, indicating that model 3 has a poor predictive ability. The accuracy of model 3 is equal to 0.8786, which is high but does not necessarily indicate a well-fitted model due to the low value of the AUC.

In summary, model 3's ability to predict deposit is poor, and this is again most likely due to the fact that only one predictor variable is included in the regression equation, like model 1 and model 2. Now that simple logsitic regressions have been modeled to see which predictor variables have larger individual impacts on the deposit status, more complex multiple logistic regressions can be modeled, illustrating a more realistic relationship between deposit status and multiple predictor variables.

Model 4: Multiple Logistic Regression (deposit ~ age+72 other variables....)

[51]:
```
%R initial_logreg_model <- glm(deposit_num_map ~␣
↪age+balance+previous_contact+campaign+job_management+job_technician\
                                  ␣
↪+job_entrepreneur+job_blue_collar+job_student+job_retired+job_admin.\
                                  +job_services+job_self_employed\
                                  ␣
↪+job_unemployed+job_housemaid+marital_married+marital_single\
                                  ␣
↪+marital_divorced+education_tertiary+education_secondary\
                                  ␣
↪+education_primary+education_unknown+default_yes+default_no\
                                  ␣
↪+housing_yes+housing_no+loan_no+loan_yes+day_1+day_2+day_3\
                                  ␣
↪+day_4+day_5+day_6+day_7+day_8+day_9+day_10+day_11\
                                  ␣
↪+day_12+day_13+day_14+day_15+day_16+day_17+day_18\
                                  ␣
↪+day_19+day_20+day_21+day_22+day_23+day_24+day_25\
                                  ␣
↪+day_26+day_27+day_28+day_29+day_30+day_31+month_may\
                                  +month_jun+month_jul+month_aug+month_oct\
                                  +month_nov+month_dec+month_jan+month_feb\
```

```
                                +month_mar+month_apr+month_sep, data=train_r_df,␣
↪family=binomial)
%R print(initial_logreg_model)

%R initial_logreg_model_summary <- summary(initial_logreg_model)
%R print(initial_logreg_model_summary)

%R ld.vars <- attributes(alias(initial_logreg_model)$Complete)$dimnames[[1]]
%R print(ld.vars)
```

```
Call:  glm(formula = deposit_num_map ~ age + balance + previous_contact +
    campaign + job_management + job_technician + job_entrepreneur +
    job_blue_collar + job_student + job_retired + job_admin. +
    job_services + job_self_employed + job_unemployed + job_housemaid +
    marital_married + marital_single + marital_divorced + education_tertiary +
    education_secondary + education_primary + education_unknown +
    default_yes + default_no + housing_yes + housing_no + loan_no +
    loan_yes + day_1 + day_2 + day_3 + day_4 + day_5 + day_6 +
    day_7 + day_8 + day_9 + day_10 + day_11 + day_12 + day_13 +
    day_14 + day_15 + day_16 + day_17 + day_18 + day_19 + day_20 +
    day_21 + day_22 + day_23 + day_24 + day_25 + day_26 + day_27 +
    day_28 + day_29 + day_30 + day_31 + month_may + month_jun +
    month_jul + month_aug + month_oct + month_nov + month_dec +
    month_jan + month_feb + month_mar + month_apr + month_sep,
    family = binomial, data = train_r_df)

Coefficients:
        (Intercept)                  age              balance
         -1.7846444           -0.0019196            0.0001667
   previous_contact             campaign       job_management
          0.8537772           -0.0990425            0.4308876
     job_technician     job_entrepreneur      job_blue_collar
          0.4175478            0.1698278            0.3632999
        job_student          job_retired           job_admin.
          0.7746094            0.8281621            0.4175947
       job_services    job_self_employed       job_unemployed
          0.3393935            0.3299598            0.7125191
      job_housemaid      marital_married       marital_single
                 NA           -0.1839815            0.0717694
   marital_divorced   education_tertiary  education_secondary
                 NA            0.1275072           -0.0526825
  education_primary    education_unknown          default_yes
         -0.4040860                   NA           -0.2793885
         default_no          housing_yes           housing_no
                 NA           -0.7170751                   NA
            loan_no             loan_yes                day_1
          0.4665961                   NA            0.6858398
```

```
          day_2             day_3             day_4
      0.4696237         0.8705694         0.7251101
          day_5             day_6             day_7
      0.4631163         0.3919635         0.1764790
          day_8             day_9            day_10
      0.4669331         0.3747974         0.9472345
         day_11            day_12            day_13
      0.5057621         0.8908673         1.0429211
         day_14            day_15            day_16
      0.6054207         0.8282876         0.5543519
         day_17            day_18            day_19
     -0.0969308         0.3668931         0.0085761
         day_20            day_21            day_22
     -0.1220134         0.2505664         0.7029496
         day_23            day_24            day_25
      0.7616316         0.3522878         1.0575052
         day_26            day_27            day_28
      0.7423634         0.9953330         0.3616087
         day_29            day_30            day_31
      0.1231890         0.7938139                NA
      month_may         month_jun         month_jul
     -1.6303972        -1.1532576        -1.1702998
      month_aug         month_oct         month_nov
     -1.2924430         0.0230728        -1.1709473
      month_dec         month_jan         month_feb
     -0.1712690        -1.7289570        -1.2245660
      month_mar         month_apr         month_sep
      0.3101207        -0.5631109                NA

Degrees of Freedom: 24051 Total (i.e. Null);  23988 Residual
Null Deviance:      17650
Residual Deviance: 15310        AIC: 15440

Call:
glm(formula = deposit_num_map ~ age + balance + previous_contact +
    campaign + job_management + job_technician + job_entrepreneur +
    job_blue_collar + job_student + job_retired + job_admin. +
    job_services + job_self_employed + job_unemployed + job_housemaid +
    marital_married + marital_single + marital_divorced + education_tertiary +
    education_secondary + education_primary + education_unknown +
    default_yes + default_no + housing_yes + housing_no + loan_no +
    loan_yes + day_1 + day_2 + day_3 + day_4 + day_5 + day_6 +
    day_7 + day_8 + day_9 + day_10 + day_11 + day_12 + day_13 +
    day_14 + day_15 + day_16 + day_17 + day_18 + day_19 + day_20 +
    day_21 + day_22 + day_23 + day_24 + day_25 + day_26 + day_27 +
    day_28 + day_29 + day_30 + day_31 + month_may + month_jun +
    month_jul + month_aug + month_oct + month_nov + month_dec +
    month_jan + month_feb + month_mar + month_apr + month_sep,
```

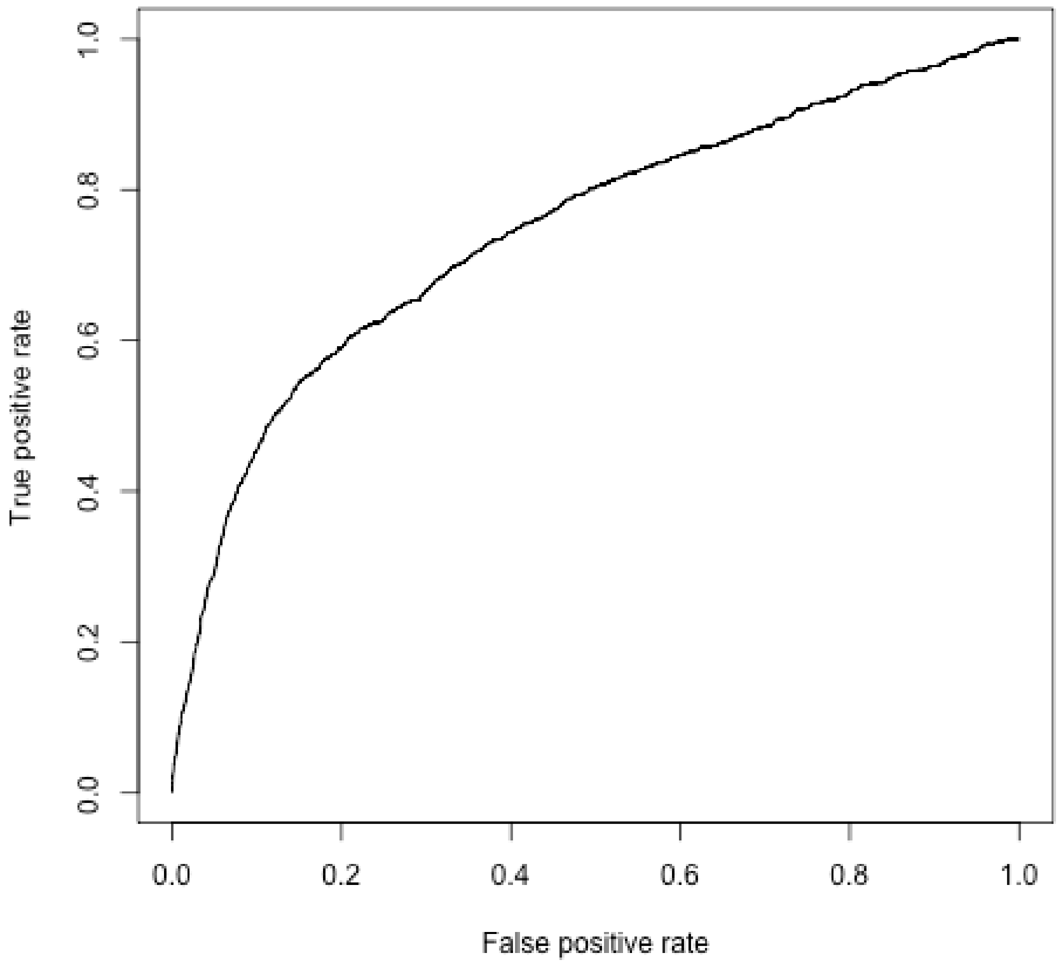
```
    family = binomial, data = train_r_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9522  -0.5032  -0.3758  -0.2761   3.0618

Coefficients: (8 not defined because of singularities)
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)         -1.785e+00  5.823e-01  -3.065 0.002178 **
age                 -1.920e-03  2.697e-03  -0.712 0.476639
balance              1.667e-04  2.266e-05   7.354 1.92e-13 ***
previous_contact     8.538e-01  4.956e-02  17.226  < 2e-16 ***
campaign            -9.904e-02  1.805e-02  -5.486 4.10e-08 ***
job_management       4.309e-01  1.640e-01   2.627 0.008607 **
job_technician       4.175e-01  1.631e-01   2.560 0.010472 *
job_entrepreneur     1.698e-01  2.042e-01   0.832 0.405536
job_blue_collar      3.633e-01  1.622e-01   2.240 0.025120 *
job_student          7.746e-01  1.906e-01   4.064 4.83e-05 ***
job_retired          8.282e-01  1.760e-01   4.706 2.53e-06 ***
job_admin.           4.176e-01  1.671e-01   2.498 0.012474 *
job_services         3.394e-01  1.734e-01   1.957 0.050297 .
job_self_employed    3.300e-01  1.925e-01   1.714 0.086438 .
job_unemployed       7.125e-01  1.871e-01   3.809 0.000140 ***
job_housemaid               NA         NA      NA       NA
marital_married     -1.840e-01  7.052e-02  -2.609 0.009080 **
marital_single       7.177e-02  8.015e-02   0.895 0.370537
marital_divorced            NA         NA      NA       NA
education_tertiary   1.275e-01  4.509e-01   0.283 0.777333
education_secondary -5.268e-02  4.486e-01  -0.117 0.906507
education_primary   -4.041e-01  4.526e-01  -0.893 0.371965
education_unknown           NA         NA      NA       NA
default_yes         -2.794e-01  2.814e-01  -0.993 0.320824
default_no                  NA         NA      NA       NA
housing_yes         -7.171e-01  5.069e-02 -14.145  < 2e-16 ***
housing_no                  NA         NA      NA       NA
loan_no              4.666e-01  7.226e-02   6.457 1.07e-10 ***
loan_yes                    NA         NA      NA       NA
day_1                6.858e-01  3.112e-01   2.204 0.027558 *
day_2                4.696e-01  2.789e-01   1.684 0.092187 .
day_3                8.706e-01  2.783e-01   3.129 0.001756 **
day_4                7.251e-01  2.733e-01   2.653 0.007968 **
day_5                4.631e-01  2.722e-01   1.701 0.088906 .
day_6                3.920e-01  2.743e-01   1.429 0.152943
day_7                1.765e-01  2.770e-01   0.637 0.524108
day_8                4.669e-01  2.704e-01   1.727 0.084172 .
day_9                3.748e-01  2.769e-01   1.354 0.175850
day_10               9.472e-01  2.930e-01   3.233 0.001224 **
day_11               5.058e-01  2.759e-01   1.833 0.066783 .
```

```
day_12              8.909e-01  2.697e-01   3.303 0.000956 ***
day_13              1.043e+00  2.704e-01   3.858 0.000114 ***
day_14              6.054e-01  2.718e-01   2.227 0.025917 *
day_15              8.283e-01  2.715e-01   3.051 0.002284 **
day_16              5.544e-01  2.744e-01   2.020 0.043371 *
day_17             -9.693e-02  2.769e-01  -0.350 0.726321
day_18              3.669e-01  2.720e-01   1.349 0.177301
day_19              8.576e-03  2.843e-01   0.030 0.975938
day_20             -1.220e-01  2.747e-01  -0.444 0.656877
day_21              2.506e-01  2.774e-01   0.903 0.366434
day_22              7.029e-01  2.837e-01   2.477 0.013236 *
day_23              7.616e-01  2.903e-01   2.624 0.008692 **
day_24              3.523e-01  3.253e-01   1.083 0.278836
day_25              1.058e+00  2.836e-01   3.729 0.000192 ***
day_26              7.424e-01  2.888e-01   2.570 0.010161 *
day_27              9.953e-01  2.834e-01   3.512 0.000444 ***
day_28              3.616e-01  2.829e-01   1.278 0.201211
day_29              1.232e-01  2.854e-01   0.432 0.665999
day_30              7.938e-01  2.701e-01   2.939 0.003296 **
day_31                     NA         NA      NA       NA
month_may          -1.630e+00  1.364e-01 -11.956  < 2e-16 ***
month_jun          -1.153e+00  1.398e-01  -8.250  < 2e-16 ***
month_jul          -1.170e+00  1.389e-01  -8.423  < 2e-16 ***
month_aug          -1.292e+00  1.383e-01  -9.344  < 2e-16 ***
month_oct           2.307e-02  1.660e-01   0.139 0.889431
month_nov          -1.171e+00  1.513e-01  -7.740 9.97e-15 ***
month_dec          -1.713e-01  2.383e-01  -0.719 0.472369
month_jan          -1.729e+00  1.831e-01  -9.442  < 2e-16 ***
month_feb          -1.225e+00  1.484e-01  -8.251  < 2e-16 ***
month_mar           3.101e-01  1.822e-01   1.702 0.088757 .
month_apr          -5.631e-01  1.424e-01  -3.954 7.67e-05 ***
month_sep                  NA         NA      NA       NA
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 17654  on 24051  degrees of freedom
Residual deviance: 15312  on 23988  degrees of freedom
AIC: 15440

Number of Fisher Scoring iterations: 5

[1] "job_housemaid"     "marital_divorced"  "education_unknown"
[4] "default_no"        "housing_no"        "loan_yes"
[7] "day_31"            "month_sep"
```

[51]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f9070bb2900> [RTYPES.STRSXP]
R classes: ('character',)
['jo…, 'ma…, 'ed…, 'de…, 'ho…, 'lo…, 'da…, 'mo…]
```

Model 4: Receiver Operator Curve & Area Under the Curve

[52]:
```
%R p_initial_logreg_model <- predict(initial_logreg_model,␣
↪newdata=subset(test_r_df, select=c(2,3,4,5,6,7,8,9,10,11\
                                                                           ␣
↪              ,12,13,14,15,16,17,18\
                                                                           ␣
↪              ,19,20,21,22,23,24,25\
                                                                           ␣
↪              ,26,27,28,29,30,31,32\
                                                                           ␣
↪              ,33,34,35,36,37,38,39\
                                                                           ␣
↪              ,40,41,42,43,44,45,46\
                                                                           ␣
↪              ,47,48,49,50,51,52,53\
                                                                           ␣
↪              ,54,55,56,57,58,59,60\
                                                                           ␣
↪              ,61,62,63,64,65,66,67\
                                                                           ␣
↪              ,68,69,70,71,72)), type="response")
%R pr_initial_logreg_model <- prediction (p_initial_logreg_model,␣
↪test_r_df$deposit_num_map)
%R prf_initial_logreg_model <- performance(pr_initial_logreg_model,␣
↪measure="tpr", x.measure="fpr")
%R plot(prf_initial_logreg_model)

%R auc_initial <- performance(pr_initial_logreg_model, measure="auc")
%R auc_initial <- auc_initial@y.values[[1]]
```



[52]: array([0.74577794])

Model 4 Accuracy Measurement

[53]:
```
%R train_r_df <- r_df[train_ind, ]
%R test_r_df <- r_df[-train_ind, ]

%R initial_logreg_model_test_4 <- predict(initial_logreg_model,␣
↪newdata=subset(test_r_df, select=c(2,3,4,5,6,7,8,9,10\
                                                                     ␣
↪                    ,11,12,13,14,15,16\
                                                                     ␣
↪                    ,17,18,19,20,21,22\
```

```
↪                    ,23,24,25,26,27,28\
↪                    ,29,30,31,32,33,34\
↪                    ,35,36,37,38,39,40\
↪                    ,41,42,43,44,45,46\
↪                    ,47,48,49,50,51,52\
↪                    ,53,54,55,56,57,58\
↪                    ,59,60,61,62,63,64\
↪                    ,65,66,67,68,69,70\
↪                    ,71,72)), type="response")
%R initial_logreg_model_test_4 <- ifelse(initial_logreg_model_test_4>0.5,1,0)

%R misclasificerror_4 <- mean(initial_logreg_model_test_4!
 ↪=test_r_df$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror_4))
```

```
[1] "Accuracy 0.881548311990687"
```

[53]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f906de2a080> [RTYPES.STRSXP]
R classes: ('character',)
['Accuracy 0.881548311990687']
```

Model 4 Interpretation Model 4 presents the first multiple logistic regression where the R application glm method is used to determine the deposit status (i.e., "yes" or "no") as the dependent variable values from all 73 predictors as the independent variables in a training subset (train_r_df) of the full trimmed and cleaned dataset (N = 34,360). The intercept was calculated to be -1.785e+00. The regression coefficients for the 73 independent variables can be found above along with their p-values and significance levels. Many of the independent variables have a significant p-value, indicating that a multiple logistic regression is more appropriate for this data than a simple logistic regression. The ROC was then plotted to calcuate the AUC. The AUC for model 4 is equal to 0.7458, so the area under the ROC covers about 75% of the rectangle, indicating that model 4 has an moderate predictive ability. The accuracy of model 4 is equal to 0.8815, which is high but does not necessarily indicate a well-fitted model due to the low value of the AUC.

In summary, model 4's ability to predict deposit is moderate, most likely due to the fact that there are linearly dependent variables being used, affecting the overall affect on deposit status. The attributes function was used to output the linearly dependent variables (i.e., `job_housemaid`, `marital_divorced`, `education_unknown`, `default_no`, `housing_no`, `loan_yes`, `day_31`, and `month_sep`) illustrated by the "NA" outputs in the model 4 summary table. Given that these variables were found to be linearly dependent, they have been removed for the next model (i.e., model 5).

Model 5: Trimmed Multiple Logistic Regression (deposit ~ age+64 other variables….

```
[54]: %R logreg_model <- glm(deposit_num_map ~␣
 ↪age+balance+previous_contact+campaign+job_management\
                           ␣
 ↪+job_technician+job_entrepreneur+job_blue_collar+job_student\
                            +job_retired+job_admin.+job_services+job_self_employed\
                           ␣
 ↪+job_unemployed+marital_married+marital_single+education_tertiary\
                           ␣
 ↪+education_secondary+education_primary+default_yes+housing_yes\
                           ␣
 ↪+loan_no+day_1+day_2+day_3+day_4+day_5+day_6+day_7+day_8+day_9\
                           ␣
 ↪+day_10+day_11+day_12+day_13+day_14+day_15+day_16+day_17+day_18\
                           ␣
 ↪+day_19+day_20+day_21+day_22+day_23+day_24+day_25+day_26+day_27\
                           ␣
 ↪+day_28+day_29+day_30+month_may+month_jun+month_jul+month_aug\
                           ␣
 ↪+month_oct+month_nov+month_dec+month_jan+month_feb+month_mar\
                            +month_apr, data=train_r_df, family=binomial)

%R logreg_model_summary <- summary(logreg_model)
%R print(logreg_model_summary)

%R vif(logreg_model)
```

```
Call:
glm(formula = deposit_num_map ~ age + balance + previous_contact +
    campaign + job_management + job_technician + job_entrepreneur +
    job_blue_collar + job_student + job_retired + job_admin. +
    job_services + job_self_employed + job_unemployed + marital_married +
    marital_single + education_tertiary + education_secondary +
    education_primary + default_yes + housing_yes + loan_no +
    day_1 + day_2 + day_3 + day_4 + day_5 + day_6 + day_7 + day_8 +
    day_9 + day_10 + day_11 + day_12 + day_13 + day_14 + day_15 +
    day_16 + day_17 + day_18 + day_19 + day_20 + day_21 + day_22 +
    day_23 + day_24 + day_25 + day_26 + day_27 + day_28 + day_29 +
    day_30 + month_may + month_jun + month_jul + month_aug +
    month_oct + month_nov + month_dec + month_jan + month_feb +
    month_mar + month_apr, family = binomial, data = train_r_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
```

```
-1.9522  -0.5032  -0.3758  -0.2761   3.0618

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)    
(Intercept)        -1.785e+00  5.823e-01  -3.065 0.002178 ** 
age                -1.920e-03  2.697e-03  -0.712 0.476639    
balance             1.667e-04  2.266e-05   7.354 1.92e-13 ***
previous_contact    8.538e-01  4.956e-02  17.226  < 2e-16 ***
campaign           -9.904e-02  1.805e-02  -5.486 4.10e-08 ***
job_management      4.309e-01  1.640e-01   2.627 0.008607 ** 
job_technician      4.175e-01  1.631e-01   2.560 0.010472 *  
job_entrepreneur    1.698e-01  2.042e-01   0.832 0.405536    
job_blue_collar     3.633e-01  1.622e-01   2.240 0.025120 *  
job_student         7.746e-01  1.906e-01   4.064 4.83e-05 ***
job_retired         8.282e-01  1.760e-01   4.706 2.53e-06 ***
job_admin.          4.176e-01  1.671e-01   2.498 0.012474 *  
job_services        3.394e-01  1.734e-01   1.957 0.050297 .  
job_self_employed   3.300e-01  1.925e-01   1.714 0.086438 .  
job_unemployed      7.125e-01  1.871e-01   3.809 0.000140 ***
marital_married    -1.840e-01  7.052e-02  -2.609 0.009080 ** 
marital_single      7.177e-02  8.015e-02   0.895 0.370537    
education_tertiary  1.275e-01  4.509e-01   0.283 0.777333    
education_secondary -5.268e-02  4.486e-01  -0.117 0.906507    
education_primary  -4.041e-01  4.526e-01  -0.893 0.371965    
default_yes        -2.794e-01  2.814e-01  -0.993 0.320824    
housing_yes        -7.171e-01  5.069e-02 -14.145  < 2e-16 ***
loan_no             4.666e-01  7.226e-02   6.457 1.07e-10 ***
day_1               6.858e-01  3.112e-01   2.204 0.027558 *  
day_2               4.696e-01  2.789e-01   1.684 0.092187 .  
day_3               8.706e-01  2.783e-01   3.129 0.001756 ** 
day_4               7.251e-01  2.733e-01   2.653 0.007968 ** 
day_5               4.631e-01  2.722e-01   1.701 0.088906 .  
day_6               3.920e-01  2.743e-01   1.429 0.152943    
day_7               1.765e-01  2.770e-01   0.637 0.524108    
day_8               4.669e-01  2.704e-01   1.727 0.084172 .  
day_9               3.748e-01  2.769e-01   1.354 0.175850    
day_10              9.472e-01  2.930e-01   3.233 0.001224 ** 
day_11              5.058e-01  2.759e-01   1.833 0.066783 .  
day_12              8.909e-01  2.697e-01   3.303 0.000956 ***
day_13              1.043e+00  2.704e-01   3.858 0.000114 ***
day_14              6.054e-01  2.718e-01   2.227 0.025917 *  
day_15              8.283e-01  2.715e-01   3.051 0.002284 ** 
day_16              5.544e-01  2.744e-01   2.020 0.043371 *  
day_17             -9.693e-02  2.769e-01  -0.350 0.726321    
day_18              3.669e-01  2.720e-01   1.349 0.177301    
day_19              8.576e-03  2.843e-01   0.030 0.975938    
day_20             -1.220e-01  2.747e-01  -0.444 0.656877    
day_21              2.506e-01  2.774e-01   0.903 0.366434    
```

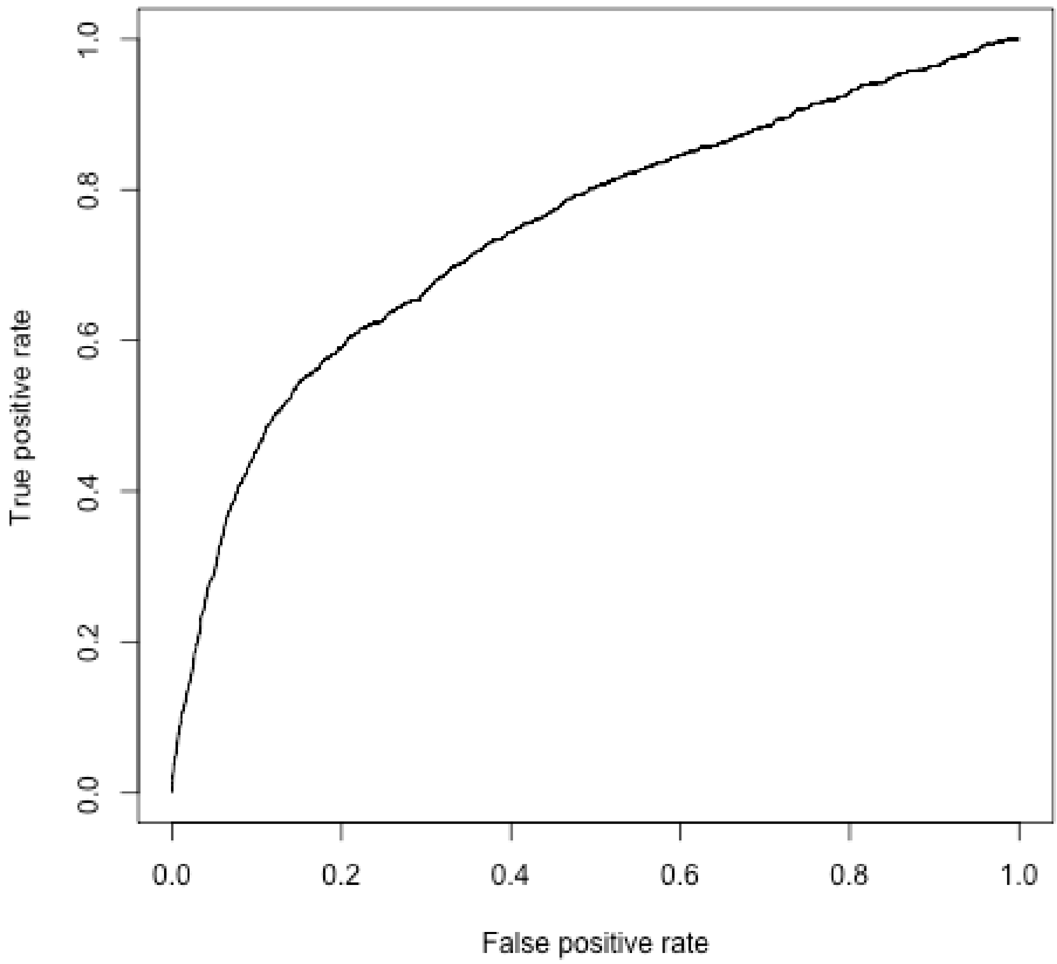
```
day_22               7.029e-01  2.837e-01   2.477 0.013236 *
day_23               7.616e-01  2.903e-01   2.624 0.008692 **
day_24               3.523e-01  3.253e-01   1.083 0.278836
day_25               1.058e+00  2.836e-01   3.729 0.000192 ***
day_26               7.424e-01  2.888e-01   2.570 0.010161 *
day_27               9.953e-01  2.834e-01   3.512 0.000444 ***
day_28               3.616e-01  2.829e-01   1.278 0.201211
day_29               1.232e-01  2.854e-01   0.432 0.665999
day_30               7.938e-01  2.701e-01   2.939 0.003296 **
month_may           -1.630e+00  1.364e-01 -11.956  < 2e-16 ***
month_jun           -1.153e+00  1.398e-01  -8.250  < 2e-16 ***
month_jul           -1.170e+00  1.389e-01  -8.423  < 2e-16 ***
month_aug           -1.292e+00  1.383e-01  -9.344  < 2e-16 ***
month_oct            2.307e-02  1.660e-01   0.139 0.889431
month_nov           -1.171e+00  1.513e-01  -7.740 9.97e-15 ***
month_dec           -1.713e-01  2.383e-01  -0.719 0.472369
month_jan           -1.729e+00  1.831e-01  -9.442  < 2e-16 ***
month_feb           -1.225e+00  1.484e-01  -8.251  < 2e-16 ***
month_mar            3.101e-01  1.822e-01   1.702 0.088757 .
month_apr           -5.631e-01  1.424e-01  -3.954 7.67e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 17654  on 24051  degrees of freedom
Residual deviance: 15312  on 23988  degrees of freedom
AIC: 15440

Number of Fisher Scoring iterations: 5
```

[54]:
```
array([  1.82210608,   1.0459243 ,   1.16742293,   1.09241671,
        11.08192882,   8.5140153 ,   2.29808011,   7.81182321,
         3.4493586 ,   3.96840472,   6.49645025,   4.73660678,
         2.79999737,   2.94589627,   2.75075594,   3.26224728,
       105.83616367, 111.75536608,  43.8628911 ,   1.01078421,
         1.37590841,   1.04880457,   2.92085607,   5.84394749,
         5.71021506,   6.83522237,   7.18762208,   6.24805456,
         5.25794533,   6.46678345,   5.32143786,   3.50568081,
         5.52491221,   7.34163136,   7.28551165,   6.34811701,
         6.49629499,   5.97739312,   5.90828079,   6.91969207,
         4.85094933,   6.51825213,   5.62355665,   4.24862651,
         3.66280014,   2.35908853,   4.23074222,   4.06998215,
         4.62782032,   4.87043542,   4.54784861,   7.21724154,
         6.59864769,   4.52249339,   4.85810901,   5.38180454,
         2.13333708,   3.77939021,   1.35342786,   2.24611726,
```

```
        3.67145029,    1.80526709,    4.32004536])
```

Model 5: Receiver Operator Curve & Area Under the Curve

```
[55]: %R p_logreg_model <- predict(logreg_model, newdata=subset(test_r_df,␣
       ↪select=c(2,3,4,5,6,7,8,9\
                                                                                          ␣
       ↪,10,11,12,13,14\
                                                                                          ␣
       ↪,15,17,18,20,21\
                                                                                          ␣
       ↪,22,25,26,28,30\
                                                                                          ␣
       ↪,31,32,33,34,35\
                                                                                          ␣
       ↪,36,37,38,39,40\
                                                                                          ␣
       ↪,41,42,43,44,45\
                                                                                          ␣
       ↪,46,47,48,49,50\
                                                                                          ␣
       ↪,51,52,53,54,55\
                                                                                          ␣
       ↪,56,57,58,59,61\
                                                                                          ␣
       ↪,62,63,64,65,66\
                                                                                          ␣
       ↪,67,68,69,70,71)), type="response")
      %R pr_logreg_model <- prediction (p_logreg_model, test_r_df$deposit_num_map)
      %R prf_logreg_model <- performance(pr_logreg_model, measure="tpr", x.
       ↪measure="fpr")
      %R plot(prf_logreg_model)

      %R auc_logreg <- performance(pr_logreg_model, measure="auc")
      %R auc_logreg <- auc_logreg@y.values[[1]]
```



[55]: array([0.74577794])

Model 5 Accuracy Measurement

[56]:
```
%R trim_logreg_model_test_5 <- predict(logreg_model, newdata=subset(test_r_df,␣
↪select=c(2,3,4,5,6,7,8,9\
                                                                              ␣
↪        ,10,11,12,13,14\
                                                                              ␣
↪        ,15,17,18,20,21\
                                                                              ␣
↪        ,22,25,26,28,30\
```

```
↪        ,31,32,33,34,35\
↪        ,36,37,38,39,40\
↪        ,41,42,43,44,45\
↪        ,46,47,48,49,50\
↪        ,51,52,53,54,55\
↪        ,56,57,58,59,61\
↪        ,62,63,64,65,66\
↪        ,67,68,69,70,71))\
                                        , type="response")
%R trim_logreg_model_test_5 <- ifelse(trim_logreg_model_test_5>0.5,1,0)
%R misclasificerror_5 <- mean(trim_logreg_model_test_5!
↪=test_r_df$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror_5))
```

```
[1] "Accuracy 0.881548311990687"
```

[56]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f906cac90c0> [RTYPES.STRSXP]
R classes: ('character',)
['Accuracy 0.881548311990687']
```

Model 5 Interpretation Model 5 presents a trimmed multiple logistic regression where the R application glm method is used to determine the deposit status (i.e., "yes" or "no") as the dependent variable values from 65 predictors as the independent variables in a training subset (train_r_df) of the full trimmed and cleaned dataset (N = 34,360). The variables that exhibited perfect linear dependence have been removed for model 5. The intercept was calculated to be -1.785e+00. The regression coefficients for the 65 independent variables can be found above along with their p-values and significance levels. Many of the independent variables have a significant p-value, indicating that a multiple logistic regression is still appropriate.
The VIF's for the independent variables were also calculated to show the linear dependency of each variable in model 5. The ROC was then plotted to calcuate the AUC. The AUC for model 5 is equal to 0.7458, so the area under the ROC covers about 75% of the rectangle, indicating that model 5 has a moderate predictive ability. The accuracy of model 5 is equal to 0.8815, which is high but does not necessarily indicate a well-fitted model due to the low value of the AUC.
In summary, model 5's ability to predict deposit is moderate, most likely due to the fact that there are still variables being used that exhibit some degree of linear dependence, affecting the overall affect on deposit status. To mitigate this issue in model 5, model 6 removes the variables with a calculated VIF>2.5.

Model 6: Trimmed Multiple Logistic Regression (default~age+balance+campaign+previous_contact+job_entrepreneur+default_yes+housing_yes+loan_no+mon

[57]:
```
#excluing variables with a vif>2.5#
%R trim_logreg_model <- glm(deposit_num_map ~ age + balance + campaign +␣
↪previous_contact\
                            + job_entrepreneur + default_yes + housing_yes +␣
↪loan_no\
                            + month_dec + month_mar, data=train_r_df,␣
↪family=binomial())
%R print(trim_logreg_model)

%R trim_logreg_model_summary <- summary(trim_logreg_model)
%R print(trim_logreg_model_summary)

%R vif(trim_logreg_model)
```

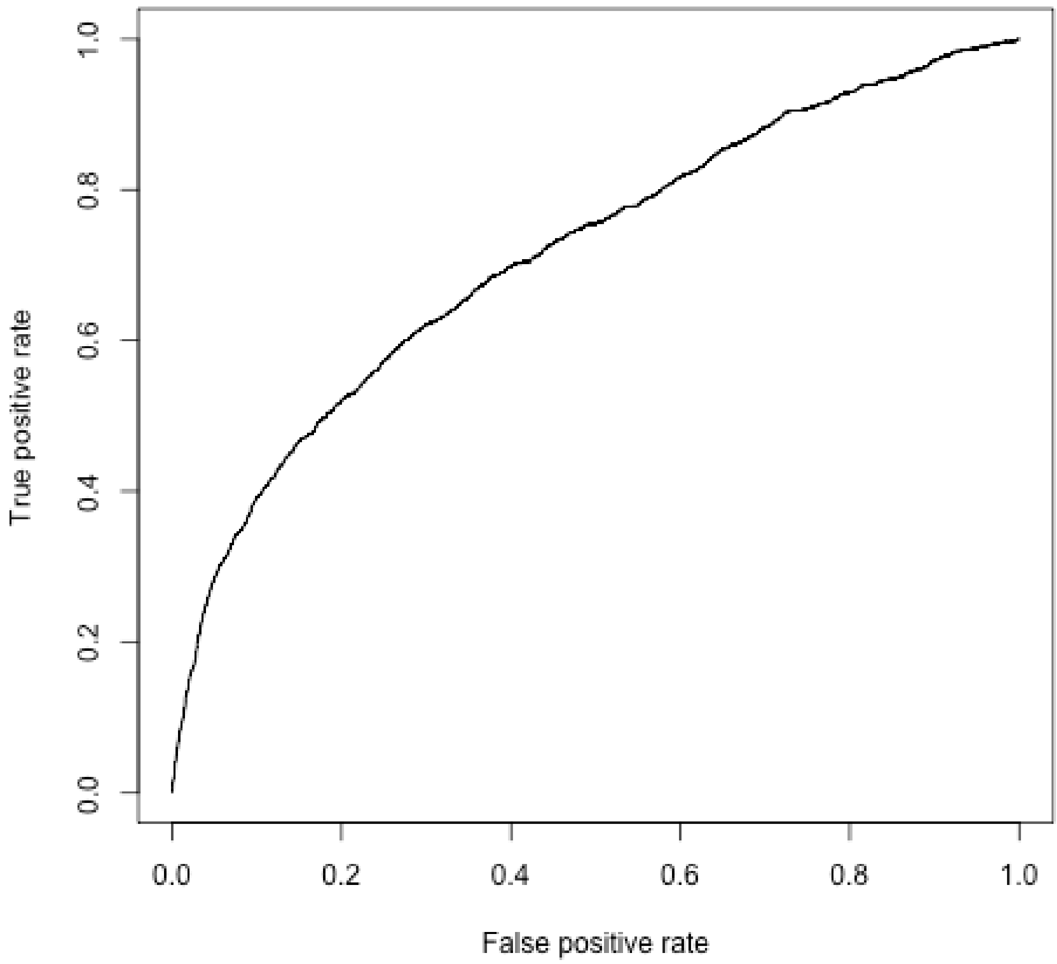
```
Call:  glm(formula = deposit_num_map ~ age + balance + campaign +
previous_contact +
    job_entrepreneur + default_yes + housing_yes + loan_no +
    month_dec + month_mar, family = binomial(), data = train_r_df)

Coefficients:
     (Intercept)               age           balance          campaign
      -1.8099373        -0.0095659         0.0001928        -0.1268168
previous_contact  job_entrepreneur       default_yes       housing_yes
       1.0624858        -0.2926793        -0.4192193        -0.9990470
         loan_no         month_dec         month_mar
       0.5327027         0.9255254         1.5944802

Degrees of Freedom: 24051 Total (i.e. Null);  24041 Residual
Null Deviance:      17650
Residual Deviance: 16060        AIC: 16080

Call:
glm(formula = deposit_num_map ~ age + balance + campaign + previous_contact +
    job_entrepreneur + default_yes + housing_yes + loan_no +
    month_dec + month_mar, family = binomial(), data = train_r_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8291  -0.5451  -0.3907  -0.3119   2.8975

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.810e+00  1.150e-01 -15.742  < 2e-16 ***
age              -9.566e-03  2.030e-03  -4.713 2.44e-06 ***
```

```
balance           1.928e-04  2.164e-05   8.911  < 2e-16 ***
campaign         -1.268e-01  1.726e-02  -7.349 2.00e-13 ***
previous_contact  1.062e+00  4.522e-02  23.496  < 2e-16 ***
job_entrepreneur -2.927e-01  1.333e-01  -2.196   0.0281 *
default_yes      -4.192e-01  2.809e-01  -1.492   0.1356
housing_yes      -9.990e-01  4.333e-02 -23.056  < 2e-16 ***
loan_no           5.327e-01  7.016e-02   7.593 3.13e-14 ***
month_dec         9.255e-01  2.048e-01   4.520 6.19e-06 ***
month_mar         1.594e+00  1.361e-01  11.713  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 17654  on 24051  degrees of freedom
Residual deviance: 16057  on 24041  degrees of freedom
AIC: 16079

Number of Fisher Scoring iterations: 5
```

[57]:
```
array([1.0272732 , 1.0165188 , 1.01336106, 1.03922092, 1.00221962,
       1.00563596, 1.04795243, 1.00666635, 1.00663339, 1.00621266])
```

Model 6: Receiver operator Curve & Area Under the Curve

[58]:
```
%R p_trim_logreg_model <- predict(trim_logreg_model, newdata=subset(test_r_df,
 ↪select=c(2,3,4,5,8,25,26\

 ↪        ,28,67,70)), type="response")
%R pr_trim_logreg_model <- prediction (p_trim_logreg_model,
 ↪test_r_df$deposit_num_map)
%R prf_trim_logreg_model <- performance(pr_trim_logreg_model, measure="tpr", x.
 ↪measure="fpr")
%R plot(prf_trim_logreg_model)

%R auc_trim_6 <- performance(pr_trim_logreg_model, measure="auc")
%R auc_trim_6 <- auc_trim_6@y.values[[1]]
```



[58]: array([0.71454489])

Model 6 Accuracy Measurement

[59]:
```
%R trim_logreg_model_test_6 <- predict(trim_logreg_model,␣
↪newdata=subset(test_r_df, select=c(2,3,4,5,8,25,26,28\
                                                                   ␣
↪              ,67,70)), type="response")
%R trim_logreg_model_test_6 <- ifelse(trim_logreg_model_test_6>0.5,1,0)
%R misclasificerror_6 <- mean(trim_logreg_model_test_6!
↪=test_r_df$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror_6))
```

```
[1] "Accuracy 0.880481179666279"
```

[59]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f907165e480> [RTYPES.STRSXP]
R classes: ('character',)
['Accuracy 0.880481179666279']
```

Model 6 Interpretation Model 6 presents another trimmed multiple logistic regression where the R application glm method is used to determine the deposit status (i.e., "yes" or "no") as the dependent variable values from `age`, `balance`, `campaign`, `previous_contact`, `job_entrepreneur`, `default_yes`, `housing_yes`, `loan_no`, `month_dec`, and `month_mar` as the independent variables in a training subset (train_r_df) of the full trimmed and cleaned dataset ($N = 34{,}360$). The variables that exhibited a VIF>2.5 have been removed for model 6. The intercept was calculated to be -1.810e+00.

The regression coefficient for `age` is -0.00957 with a statistically significant p-value <.001. This means that as age increases, there is a slight decrease in the log odds (and by extension probability) of the client subscribing to a term deposit. Model 6 indicates that `age` is significant in determining the deposit status.

The regression coefficient for `balance` is 0.00019 with a statistically significant p-value <.001. This means that as average yearly balance in Euros increases, there is a slight increase in the log odds of the client subscribing to a term deposit. Model 6 indicates that `balance` is significant in determining the deposit status.

The regression coefficient for `campaign` is -0.12680 with a statistically significant p-value <.001. This means that as campaign increases, there is a decrease in the log odds of the client subscribing to a term deposit. Model 6 indicates that `campaign` is significant in determining the deposit status.

The regression coefficient for `previous_contact` is 1.0620 with a statistically significant p-value <.001. This means that based on whether there was previous contact there is a moderate increase in the log odds of the client subscribing to a term deposit. Model 6 indicates that `previous_contact` is significant in determining the deposit status.

The regression coefficient for `job_entrepreneur` is -0.29270 with a statistically significant p-value <.05. This means that when a person's job is entrepreneur, there is a slight decrease in the log odds of the client subscribing to a term deposit. Model 6 indicates that `job_entrepreneur` is significant in determining the deposit status.

The regression coefficient for `default_yes` is -0.41920 with a p-value of 0.1356. This means that if you default, there is a increase in the log odds of the client subscribing to a term deposit. Model 6 indicates that `default_yes` is not significant in determining the deposit status.

The regression coefficient for `housing_yes` is -0.99900 with a statistically significant p-value <.001. This means that people with a housing loan, decrease the log odds of the client subscribing to a term deposit. Model 6 indicates that `housing_yes` is significant in determining the deposit status.

The regression coefficient for `loan_no` is 0.53270 with a statistically significant p-value <.001. This means that people without a personal loan, increase the log odds of the client subscribing to a term deposit. Model 6 indicates that `loan_no` is significant in determining the deposit status.

The regression coefficient for `month_dec` is 0.92550 with a statistically significant p-value <.001. This means that people who were last contacted about the deposit in December, increase the log odds of the client subscribing to a term deposit. Model 6 indicates that `month_dec` is significant in determining the deposit status.

The regression coefficient for `month_mar` is 1.5940 with a statistically significant p-value <.001. This means that people who were last contacted about the deposit in March, increase the log odds of the client subscribing to a term deposit. Model 6 indicates that `month_mar` is significant in determining the deposit status.

The ROC was then plotted to calcuate the AUC. The AUC for model 6 is equal to 0.7145, so the area under the ROC covers about 71% of the rectangle, indicating that model 6 has a moderate predictive ability. The accuracy of model 6 is equal to 0.8805, which is high but does not necessarily indicate a well-fitted model due to the low value of the AUC.
In summary, model 6's ability to predict deposit is moderate, most likely due to including the `default_yes` variable. Since this variable was not significant, it will be removed for model 7, to more accurately depict a logistic model for deposit status.

Model 7: Trimmed Multiple Regression (deposit~age+balance+camapgin+previous-contact+job_entrepreneur+housing_yes+loan_no+month_dec+month_mar)

[60]:
```
# Final model!!
%R trim_logreg_model_7 <- glm(deposit_num_map ~ age +balance + campaign +␣
↪previous_contact\
                              + job_entrepreneur + housing_yes + loan_no +␣
↪month_dec\
                              + month_mar, data=train_r_df, family=binomial())
%R print(trim_logreg_model_7)

%R trim_logreg_model_7_summary <- summary(trim_logreg_model_7)
%R print(trim_logreg_model_7_summary)

%R vif(trim_logreg_model_7)
```

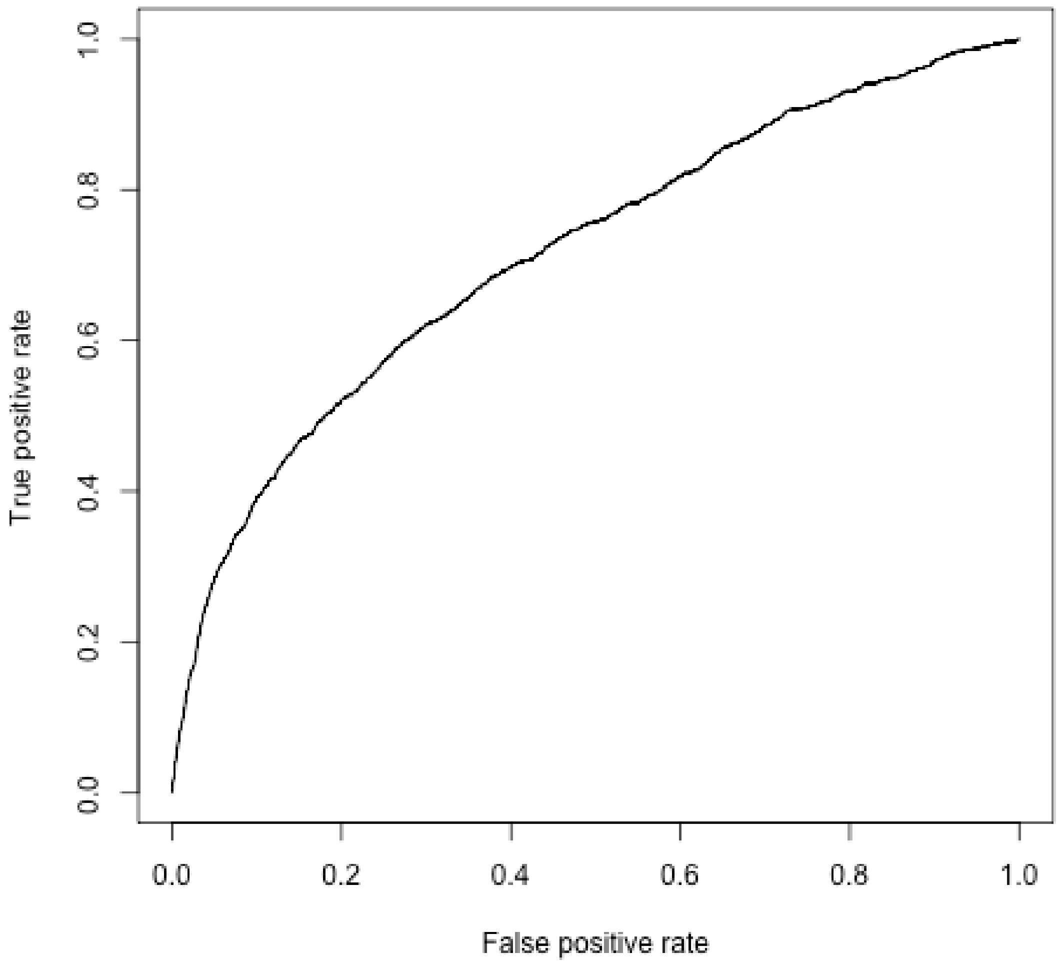
```
Call:  glm(formula = deposit_num_map ~ age + balance + campaign +
previous_contact +
    job_entrepreneur + housing_yes + loan_no + month_dec + month_mar,
    family = binomial(), data = train_r_df)

Coefficients:
     (Intercept)               age           balance          campaign
      -1.8209708        -0.0095439         0.0001948        -0.1267321
previous_contact  job_entrepreneur       housing_yes           loan_no
       1.0644249        -0.2937912        -0.9977237         0.5373725
       month_dec         month_mar
       0.9277300         1.5968905

Degrees of Freedom: 24051 Total (i.e. Null);  24042 Residual
Null Deviance:      17650
Residual Deviance: 16060        AIC: 16080

Call:
glm(formula = deposit_num_map ~ age + balance + campaign + previous_contact +
    job_entrepreneur + housing_yes + loan_no + month_dec + month_mar,
    family = binomial(), data = train_r_df)

Deviance Residuals:
```

```
    Min       1Q   Median       3Q      Max
-1.8319  -0.5447  -0.3914  -0.3125   2.9007

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.8209708  0.1147669 -15.867  < 2e-16 ***
age              -0.0095439  0.0020300  -4.701 2.58e-06 ***
balance           0.0001948  0.0000216   9.019  < 2e-16 ***
campaign         -0.1267321  0.0172572  -7.344 2.08e-13 ***
previous_contact  1.0644249  0.0452100  23.544  < 2e-16 ***
job_entrepreneur -0.2937912  0.1332428  -2.205   0.0275 *
housing_yes      -0.9977238  0.0433240 -23.029  < 2e-16 ***
loan_no           0.5373725  0.0701023   7.666 1.78e-14 ***
month_dec         0.9277300  0.2048106   4.530 5.91e-06 ***
month_mar         1.5968905  0.1361512  11.729  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 17654  on 24051  degrees of freedom
Residual deviance: 16060  on 24042  degrees of freedom
AIC: 16080

Number of Fisher Scoring iterations: 5
```

[60]:
```
array([1.02723503, 1.01387427, 1.01338015, 1.03878423, 1.00223027,
       1.04761069, 1.00536454, 1.00659537, 1.00610882])
```

Model 7: Receiver Operator Curve & Area Under the Curve

[61]:
```
%R p_trim_logreg_model_7 <- predict(trim_logreg_model_7,␣
 ↪newdata=subset(test_r_df, select=c(2,3,4,5,8,26,28,67\
                                                                ␣
 ↪            ,70)), type="response")
%R pr_trim_logreg_model_7 <- prediction (p_trim_logreg_model_7,␣
 ↪test_r_df$deposit_num_map)
%R prf_trim_logreg_model_7 <- performance(pr_trim_logreg_model_7,␣
 ↪measure="tpr", x.measure="fpr")
%R plot(prf_trim_logreg_model_7)

%R auc_trim_7 <- performance(pr_trim_logreg_model_7, measure="auc")
%R auc_trim_7 <- auc_trim_7@y.values[[1]]
```



[61]: array([0.71499417])

Model 7 Accuracy Measurement

[62]:
```
%R trim_logreg_model_test_7 <- predict(trim_logreg_model_7,␣
↪newdata=subset(test_r_df, select=c(2,3,4,5,8,26\
                                                                          ␣
↪                 ,28,67,70))\
                                      , type="response")
%R trim_logreg_model_test_7 <- ifelse(trim_logreg_model_test_7>0.5,1,0)
%R misclasificerror_7 <- mean(trim_logreg_model_test_7!
↪=test_r_df$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror_7))
```

```
[1] "Accuracy 0.880481179666279"
```

[62]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f90701a55c0> [RTYPES.STRSXP]
R classes: ('character',)
['Accuracy 0.880481179666279']
```

Model 7 Interpretation Model 7 presents the "best" trimmed multiple logistic regression where the R application glm method is used to determine the deposit status (i.e., "yes" or "no") as the dependent variable values from `age`, `balance`, `campaign`, `previous_contact`, `job_entrepreneur`, `housing_yes`, `loan_no`, `month_dec`, and `month_mar` as the independent variables in a training subset (train_r_df) of the full trimmed and cleaned dataset (N = 34,360). The `default_yes` variable has been removed for model 7. The intercept was calculated to be -1.8210.
The regression coefficient for `age` is -0.00954 with a statistically significant p-value <.001. This means that as age increases, there is a slight decrease in the log odds (and by extension probability) of the client subscribing to a term deposit. Model 7 indicates that `age` is significant in determining the deposit status.
The regression coefficient for `balance` is 0.00019 with a statistically significant p-value <.001. This means that as average yearly balance in Euros increases, there is a slight increase in the log odds (and by extension probability) of the client subscribing to a term deposit. Model 7 indicates that `balance` is significant in determining the deposit status.
The regression coefficient for `campaign` is -0.1267 with a statistically significant p-value <.001. This means that as campaign increases, there is a decrease in the log odds of the client subscribing to a term deposit. Model 7 indicates that `campaign` is significant in determining the deposit status.
The regression coefficient for `previous_contact` is 1.06442 with a statistically significant p-value <.001. This means that based on whether there was previous there is a moderate increase in the log odds of the client subscribing to a term deposit. Model 7 indicates that `previous_contact` is significant in determining the deposit status.
The regression coefficient for `job_entrepreneur` is -0.29379 with a statistically significant p-value <.05. This means that when a person's job is entrepreneur, there is a slight decrease in the log odds of the client subscribing to a term deposit. Model 7 indicates that `job_entrepreneur` is significant in determining the deposit status.
The regression coefficient for `housing_yes` is -0.99772 with a statistically significant p-value <.001. This means that people with a housing loan, decrease the log odds of the client subscribing to a term deposit. Model 6 indicates that `housing_yes` is significant in determining the deposit status.
The regression coefficient for `loan_no` is 0.53737 with a statistically significant p-value <.001. This means that people without a personal loan, increase the log odds of the client subscribing to a term deposit. Model 7 indicates that `loan_no` is significant in determining the deposit status.
The regression coefficient for `month_dec` is 0.92773 with a statistically significant p-value <.001. This means that people who were last contacted about the deposit in December, increase the log odds of the client subscribing to a term deposit. Model 7 indicates that `month_dec` is significant in determining the deposit status.
The regression coefficient for `month_mar` is 1.59689 with a statistically significant p-value <.001. This means that people who were last contacted about the deposit in march, increase the log odds of the client subscribing to a term deposit. Model 7 indicates that `month_mar` is significant in determining the deposit status.
The ROC was then plotted to calcuate the AUC. The AUC for model 7 is equal to 0.7150, so the area under the ROC covers about 72% of the rectangle, indicating that model 7 has a moderate predictive ability. The accuracy of model 7 is equal to 0.8805, which is high but does not necessarily

indicate a well-fitted model due to the low value of the AUC.
In summary, model 7 is the best of the 8 models produced in its ability to predict deposit status. So, when `age`, `balance`, `camapaign`, `previous_contact`, `job_entrepreneuer`, `housing_yes`, `loan_no`, `month_dec`, and `month_mar` are used to predict deposit status, it produces the best of the models, since all the predictors have a significant effect on deposit status. Something to note: While this model produces the best results compared to our other models, model 7 still doesn't have great predictive ability of deposit status overall.

****Model 7b: Trimmed Multiple Regression** (deposit ~ age+balance+campaign+previous-contact+job_entrepreneuer+housing_yes+loan_no+month_dec+month_mar)

[63]:
```
#final model on subset where "yes" ~40% of sample!!#
%R trim_logreg_model_7b <- glm(deposit_num_map ~ age +balance + campaign +␣
↪previous_contact\
                              + job_entrepreneur + housing_yes + loan_no +␣
↪month_dec\
                              + month_mar, data=train_r_df2, family=binomial())
%R print(trim_logreg_model_7b)

%R trim_logreg_model_7b_summary <- summary(trim_logreg_model_7b)
%R print(trim_logreg_model_7b_summary)

%R vif(trim_logreg_model_7b)
```

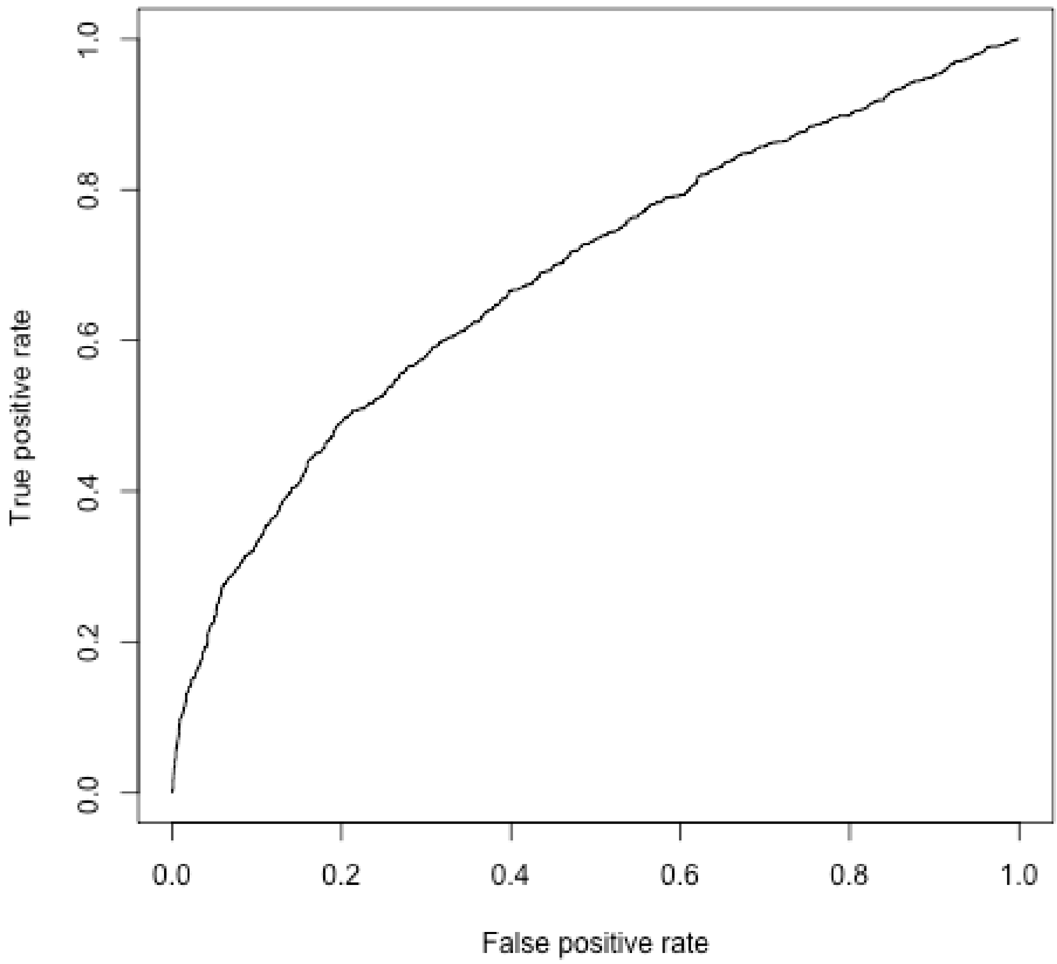
```
Call:  glm(formula = deposit_num_map ~ age + balance + campaign +
previous_contact +
    job_entrepreneur + housing_yes + loan_no + month_dec + month_mar,
    family = binomial(), data = train_r_df2)

Coefficients:
     (Intercept)               age           balance          campaign
      -0.2812157        -0.0100724         0.0002102        -0.1200527
previous_contact  job_entrepreneur       housing_yes           loan_no
       1.0906623        -0.1958035        -0.9645969         0.5495512
       month_dec         month_mar
       1.2974934         1.8187300

Degrees of Freedom: 7240 Total (i.e. Null);  7231 Residual
Null Deviance:      9766
Residual Deviance: 8666        AIC: 8686

Call:
glm(formula = deposit_num_map ~ age + balance + campaign + previous_contact +
    job_entrepreneur + housing_yes + loan_no + month_dec + month_mar,
    family = binomial(), data = train_r_df2)

Deviance Residuals:
```

```
    Min       1Q   Median       3Q      Max
-2.4883  -0.9542  -0.6714   1.1337   2.2702

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)      -2.812e-01  1.377e-01  -2.042   0.0412 *
age              -1.007e-02  2.499e-03  -4.031 5.56e-05 ***
balance           2.102e-04  2.803e-05   7.499 6.42e-14 ***
campaign         -1.201e-01  2.101e-02  -5.714 1.10e-08 ***
previous_contact  1.091e+00  5.939e-02  18.366  < 2e-16 ***
job_entrepreneur -1.958e-01  1.643e-01  -1.192   0.2334
housing_yes      -9.646e-01  5.297e-02 -18.211  < 2e-16 ***
loan_no           5.496e-01  8.117e-02   6.770 1.29e-11 ***
month_dec         1.297e+00  3.052e-01   4.252 2.12e-05 ***
month_mar         1.819e+00  2.226e-01   8.170 3.09e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 9766.2  on 7240  degrees of freedom
Residual deviance: 8665.9  on 7231  degrees of freedom
AIC: 8685.9

Number of Fisher Scoring iterations: 4
```

[63]:
```
array([1.03101614, 1.0150116 , 1.01176488, 1.02455333, 1.00183201,
       1.03680486, 1.0054378 , 1.00216052, 1.0019654 ])
```

Model 7b: Receiver Operator Curve & Area Under the Curve

[64]:
```
%R p_trim_logreg_model_7b <- predict(trim_logreg_model_7b,
 ↪newdata=subset(test_r_df2, select=c(2,3,4,5,8,26,28\

 ↪                ,67,70))\
                                      , type="response")
%R pr_trim_logreg_model_7b <- prediction (p_trim_logreg_model_7b,
 ↪test_r_df2$deposit_num_map)
%R prf_trim_logreg_model_7b <- performance(pr_trim_logreg_model_7b,
 ↪measure="tpr", x.measure="fpr")
%R plot(prf_trim_logreg_model_7b)

%R auc_trim_7b <- performance(pr_trim_logreg_model_7b, measure="auc")
%R auc_trim_7b <- auc_trim_7b@y.values[[1]]
```



[64]: array([0.68676548])

Model 7b Accuracy Measurement

[65]:
```
%R trim_logreg_model_test_7b <- predict(trim_logreg_model_7b,␣
↪newdata=subset(test_r_df2, select=c(2,3,4,5,8\
                                                                      ␣
↪                    ,26,28,67,70))\
                                          , type="response")
%R trim_logreg_model_test_7b <- ifelse(trim_logreg_model_test_7b>0.5,1,0)
%R misclasificerror_7b <- mean(trim_logreg_model_test_7b!
↪=test_r_df2$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror_7b))
```

```
[1] "Accuracy 0.680090206185567"
```

[65]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f906b56b700> [RTYPES.STRSXP]
R classes: ('character',)
['Accuracy 0.680090206185567']
```

Model 8: Trimmed Multiple Logistic Regression (deposit ~ balance+campaign+previous_contact)

[66]:
```
%R trim_logreg_model_8 <- glm(deposit_num_map ~ balance + campaign +␣
 ↪previous_contact, data=train_r_df\
                                , family=binomial())
%R print(trim_logreg_model_8)

%R trim_logreg_model_8_summary <- summary(trim_logreg_model_8)
%R print(trim_logreg_model_8_summary)

%R vif(trim_logreg_model_8)
```

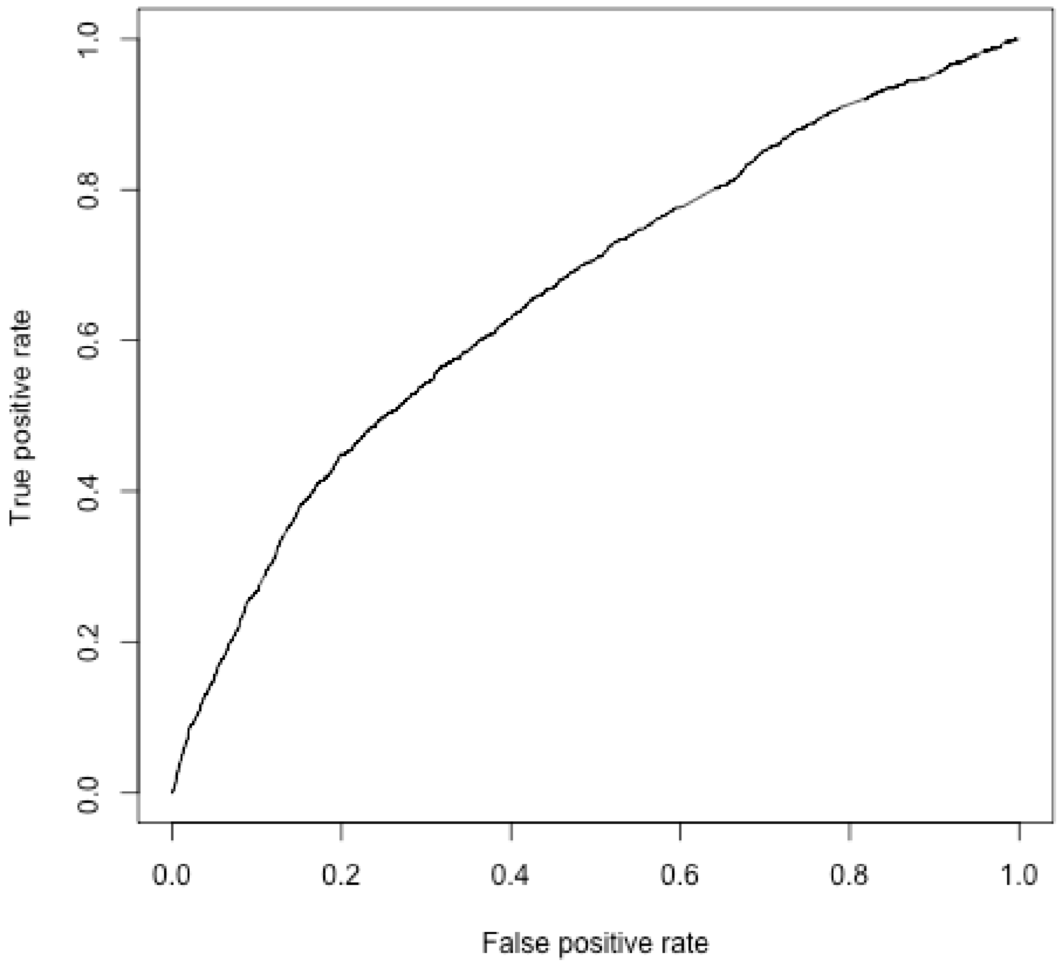
```
Call:  glm(formula = deposit_num_map ~ balance + campaign + previous_contact,
    family = binomial(), data = train_r_df)

Coefficients:
     (Intercept)           balance          campaign  previous_contact
      -2.1954078         0.0002268        -0.1214765         0.9878815

Degrees of Freedom: 24051 Total (i.e. Null);  24048 Residual
Null Deviance:      17650
Residual Deviance: 16930        AIC: 16940

Call:
glm(formula = deposit_num_map ~ balance + campaign + previous_contact,
    family = binomial(), data = train_r_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9810  -0.4966  -0.4363  -0.3942   2.4399

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)      -2.195e+00  4.646e-02  -47.25   <2e-16 ***
balance           2.268e-04  2.091e-05   10.85   <2e-16 ***
campaign         -1.215e-01  1.687e-02   -7.20    6e-13 ***
previous_contact  9.879e-01  4.313e-02   22.91   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 17654  on 24051  degrees of freedom
Residual deviance: 16933  on 24048  degrees of freedom
AIC: 16941

Number of Fisher Scoring iterations: 5
```

[66]: array([1.00115477, 1.00893585, 1.0096618])

Model 8: Receiver Operator Curve & Area Under the Curve

[67]:
```
%R p_trim_logreg_model_8 <- predict(trim_logreg_model_8,␣
 ↪newdata=subset(test_r_df, select=c(3,4,5)), type="response")
%R pr_trim_logreg_model_8 <- prediction (p_trim_logreg_model_8,␣
 ↪test_r_df$deposit_num_map)
%R prf_trim_logreg_model_8 <- performance(pr_trim_logreg_model_8,␣
 ↪measure="tpr", x.measure="fpr")
%R plot(prf_trim_logreg_model_8)

%R auc_trim_8 <- performance(pr_trim_logreg_model_8, measure="auc")
%R auc_trim_8 <- auc_trim_8@y.values[[1]]
```



[67]: array([0.66261589])

Model 8 Accuracy Measurement

[68]:
```
%R trim_logreg_model_test_8 <- predict(trim_logreg_model_8,␣
↪newdata=subset(test_r_df, select=c(3,4,5)), type="response")
%R trim_logreg_model_test_8 <- ifelse(trim_logreg_model_test_8>0.5,1,0)
%R misclasificerror_8 <- mean(trim_logreg_model_test_8!
↪=test_r_df$deposit_num_map)
%R print(paste('Accuracy', 1-misclasificerror_8))
```

```
[1] "Accuracy 0.878637951105937"
```

[68]:
```
<rpy2.robjects.vectors.StrVector object at 0x7f9070bacd80> [RTYPES.STRSXP]
R classes: ('character',)
['Accuracy 0.878637951105937']
```

Model 8 Interpretation Model 8 presents a trimmed multiple logistic regression where the R application glm method is used to determine the deposit status (i.e., "yes" or "no") as the dependent variable values from `balance`, `campaign`, and `previous_contact` as the independent variables in a training subset (train_r_df) of the full trimmed and cleaned dataset (N = 34,360). Model 8 looks at the relationship between three of the numerical predictors and deposit status. The intercept was calculated to be -2.1950.
The regression coefficient for `balance` is 0.00023 with a statistically significant p-value <.001. This means that as average yearly balance in Euros increases, there is a slight increase in the log odds (and by extension probability) of the client subscribing to a term deposit. Model 8 indicates that `balance` is significant in determining the deposit status.
The regression coefficient for `campaign` is -0.12150 with a statistically significant p-value <.001. This means that as campaign increases, there is a decrease in the log odds of the client subscribing to a term deposit. Model 8 indicates that `campaign` is significant in determining the deposit status.
The regression coefficient for `previous_contact` is 0.09879 with a statistically significant p-value <.001. This means that based on whether there was previous contact there is a slight increase in the log odds of the client subscribing to a term deposit. Model 8 indicates that `previous_contact` is significant in determining the deposit status.
The ROC was then plotted to calcuate the AUC. The AUC for model 8 is equal to 0.6626, so the area under the ROC covers about 66% of the rectangle, indicating that model 8 has a weak predictive ability. The accuracy of model 8 is equal to 0.8786, which is high but does not necessarily indicate a well-fitted model due to the low value of the AUC.
In summary, model 8's ability to predict deposit is weak, most likely due to the fact that there are excluded variables that have an affect on deposit status. This model was created to show how soley the numeric variables affect deposit status.

Overall Results & Summary Several models were developed to reduce the number of features to a manageable amount and attempt to find the model with the highest chance of accurately predicting the binary outcome of the `deposit` variable (dependent feature). Models 1-3 were simple logistic regression models comparing one response variable to one predictor variable. As outlined under each model above, none of them were sufficiently predictive to be contenders for the final and most useful model. Models 4-8 used multiple independent variables with the idea that their combinations would be better at predicting the binary result than any one of them alone. Model 4 was the largest model and included 73 independent variables. The large number was due to a large number of categorical variables that were discretized by generating dummy variables, and moreover, several of the nominal variables had several individual categories (e.g., after transforming "unknown" values for job and education, they had 11 and 4 categories respectively). Model 4 was run to see if all variables included would generate a good predictive model, but after doing so, it only had an accuracy of 88.15%, which was only slightly higher than the overall ratio of "yes" in the full cleaned and trimmed dataset (87.96%; N = 34,360). For model 5, the number of predictor variables was reduced to 65 by eliminating those variables (relative to model 4) that were determined to be perfectly linearly dependent on other independent variables (namely, `job_housemaid`, `marital_divorced`, `education_unknown`, `default_no`, `housing_no`, `loan_yes`, `day_31`, `month_sep`), as indicated by a regression coefficient of "NA". Module 6 was reduced relative to model 5 down to 10 independent variables by eliminating any variables with a variance inflation factor (VIF) greater than 2.5. This was done to further avoid possible multi-collinearity with other independent variables. Model 6 had a *p*-value of .1356 for `default_yes`, so it was eliminated for model 7 and 7b–as these latter models were determined to be the best from our analyses, there are discussed further below. For one last model (model 8), solely the "naturally" numerical variables were used to see if they alone would provide a sufficient model. It was found that the accuracy was 87.86% and AUC was 0.6626 which did not make for a good model.

Final Models Model 7 used the following predictor variables along with the corresponding regression coefficients: `age` (-0.0095), `balance` (0.0002), `campaign` (-0.1267), `previous_contact` (1.0644), `job_entrepreneur` (-0.2938), `housing_yes` (-0.9977), `loan_no` (0.5374), `month_dec` (0.9277), and `month_mar` (1.5969). Intercept = -1.8210. After reviewing all of the models, it was determined that model 7 had the best combination of accuracy (88.05%), AUC (0.7150), and interpretability. Though the AUC was closer to 0.5 than 1.0, it was somewhat close to the models with the best overall AUC (models 4 & 5 AUC = 0.7458). Obviously this does not make it good, per se, but given the predictability of all models, it was mid- to high-level relatively.

In terms of interpreting the model, as age increased there was a slight reduction in the probability of being able to predict deposit as "yes" and with an increase in balance, probability rose a little; both coefficients are slight. An increase in the number of contacts during the campaign actually decreased the probability, which makes sense if considering that people might be frustrated at being contacted more and would then be less likely to do business with that bank. However, whether the person did have any previous contact ("yes" or "no") did seem to have relatively high positive impact. Also interesting was that whether or not the person's job was entrepreneur had a negative affect—it should be noted that since only one job type was included, the only interpretation for this model that can be made is based on whether someone was an entrepreneur or not (i.e., a binary distinction, such that all other job types were lumped together into a "not-entrepreneur" category). This could be because as the job of entrepreneur there may be large fluctuations in income due to whether their current ventures are going well or not. Whether someone had a housing loan or didn't have a personal loan also had an impact on the probability of predicting deposit as "yes". This makes sense from the angle that a housing loan indicates a substantial outlay of income (thereby increasing probability that additional funds are not available to make a deposit), but a personal loan may be the opposite in effect. Lastly, whether the contact month was December, March, or other has had an impact. One possibility is that those were high volume banking months based on holiday bonuses received (in December) as well as preparation and saving for summer vacations (in March), and that contact in those months would remind people to open deposit accounts close to when they actually needed a specific place to keep funds.

Model 7b was an attempt to mitigate the fact that the "yes" value in the full trimmed and cleaned dataset represented only 12.04% of the sample. As one additional step, the records with "yes" were pulled out, the n was divided by a fractional signifier (.4 in this case) to achieve a total desired n such that the ratio of "yes" to "no" was 40%. The set that contained only "no" values was then randomly sampled to pull out the number of records such that they would represent 60% of the total sample (1 - .4). Then using that dataset (n = 10,345) the same process that model 7 went through was performed: namely, it was split into a train and test subsets, and then a model was created using the same independent variables, as follows with regression coefficients: `age` (-0.0148), `balance` (0.0002), `campaign` (-0.1324), `previous_contact` (1.034*), `job_entrepreneur` (-0.3484), `housing_yes` (-0.9395), `loan_no` (0.5553), `month_dec` (1.296*), and `month_mar` (1.874*). Intercept = -0.0847.* Rounding due to RPY2 output in scientific notation. The intercept decreased significantly, and the two month-related coefficients increased relatively significantly. However, the accuracy and AUC actually decreased (68.14% and 69.39% respectively) relative to model 7, so in the end, model 7b was worse at predicting the probability of the dependent variable compared to model 7.

Strengths & possible limitations of the current analyses There were several strengths to the dataset. The size of the dataset (N = 45,211) meant that there would be sufficient data to develop a regression model using both a training set and a testing set. Additionally, both Python and R were used, which provided the ability to utilize each program for its strengths; Python's numpy and pandas libarries were used for the pre-processing due their speed and efficient, and R was used to generate the logistic regression models and generate several of the corresponding visualizations. With the trimming models used, multicollinearity was sufficiently addressed to avoid interaction or interdependence between predictor variables.
There were also several factors that may have contributed a bias to the results. Initially a heavy cleaning of the data was required, which removed 20.86% of the data and also required filling in missing values (e.g., for age), and transforming ambigous values (e.g., changing "unknown" value in job to other values based on the modes of grouped data); taking a slightly different approach my have resulted in different outcomes. Another limitation may have been the available ratio of categorical values in some of the columns–for instance, if the number of yes's and no's in binary variable column were not even, it may have meant that the model again did not have sufficient compartive sample for each value to develop a robust predictive model. Lastly, there were a lot of variables to choose from initially to build the model–most of which may be a predictor of the dependent variables–but also several of them were categorical variables (some of which had several categories) that had to be discretized to be used in the model building process, all of which meant that including all of the dummy variables there were more than 70 to choose from to develop the model.

Future directions Explore addition classification methods to obtain a better predictive model that was achieved using logistic regression.

References:

* Coban, H. (2019). Here's how I used Python to build a regression model using an e-commerce dataset. https://searchengineland.com/heres-how-i-used-python-to-build-a-regression-model-using-an-e-commerce-dataset-326493
* Devore, J. L. (2016). *Probability and statistics.* (9th ed.). Cengage Learning.
* Mukala, M. M. (2012). A guide to appropriate use of correlation coefficient in medical research. Malawi Medical Journal, 24(3), 69-71.
* Real Python. (n.d.). Linear regression in python. https://realpython.com/linear-regression-in-python/
* Real Python. (n.d.). Logistic regression in python. https://realpython.com/logistic-regression-python/
* Shah, C. (2020). *A hands-on introduction to data science.* Cambridge University Press.
* Stack Overflow. (n.d.). GroupBy pandas dataFrame and select most common value. https://stackoverflow.com/questions/15222754/groupby-pandas-dataframe-and-select-most-common-value
* UCLA: Statistical Consulting Group. Introduction to SAS.

https://stats.idre.ucla.edu/sas/modules/sas-learning-moduleintroduction-to-the-features-of-sas/
(accessed December 12, 2021).
***