GBUS 738 - Data Mining
May 14, 2020
Final Project

Sai Santosh Avala
Jian-Hong (Adam) Chen
Thaw Zin (Tiffany) Htwe
Hemalekha Pillarishetty
Haneefa Bhanu Sunkesula
Priscilla Tran

Data Set:

https://archive.ics.uci.edu/ml/datasets/Bank+Marketing

Marketing Campaign Effectiveness in the Banking Industry

In the banking industry, many finance companies seek to monopolize the client pool. Facing such a large amount of competition, the marketing and advertising strategies used are critical for banks to deliver value and capture the target audience. Companies spend a substantial amount of their budget towards marketing efforts which oftentimes results in poor ROI. How can financial companies better allocate their funds and strategies to effectively reach the target market?

Our team plans on analyzing data sets to view how effective or ineffective banks are in their marketing efforts. We will take a look at a timeless marketing effort that has been used (phone calls) to evaluate their approach and performance in persuading clients to subscribe to a product or service of theirs.

Based on the analysis of the data and its different attributes, we can measure successes or failures of the current strategy, and reassess to recommend future campaigns that will deliver more profitable results. The chosen dataset is important as it is a useful model that banking companies can have as a benchmark or example when developing marketing campaigns. Analyzing the results gives us a general idea of how consumers respond to certain types of marketing strategies. And as such, companies could either follow a similar strategy or implement other innovative tactics that would prove more effective in successfully reaching the target audience.

Data Collection

The bank marketing data is downloaded from the UCI Machine Learning Repository which is a collection of databases. It offers numerous datasets open to the public especially for the machine learning community for the empirical analysis of machine learning algorithms. The bank marketing data relates with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone call outreach to customers. Often, more than one method of contact to the same client was required, in order to assess if the product (bank term deposit) would be subscribed or not ('yes' or 'no'). The dataset contains 17 attributes and 45211 instances, and it provides different types of customer data such as age, job, marital status, education and so on. The attribute values are mixed of both numerical and categorical data. The variables names age, job, marital, education, default, balance, housing, loan, contact, day, month, duration, campaign, pdays, previous, and poutcome are independent variables, or x variables, whereas deposit is the dependent variable, or y variable. The output variable y denotes if the client subscribed to a term deposit or not.

Objective of the analysis

The objective of the analysis is to answer which clients are more likely to subscribe for term deposits. We will be able to see how effective the bank institution's marketing efforts are while additionally analyzing the list of attributes to see which would be valuable for future use. Thus, our goal is to build models that will predict if a client will subscribe to a term deposit or not, and use the results as feedback for further marketing planning.

Attribute Information:

Customer-related Attributes:

- Age: Customer Age, which is a numeric type
- Job : gives information about the type of job, it is a categorical data type. Categories include- *'admin.', 'blue-collar', 'entrepreneur', 'housemaid', 'management', 'retired', 'self-employed', 'services', 'student', 'technician', 'unemployed', 'unknown'*
- Marital : gives information about the customer's marital status, it is a categorical data type. Categories include- *'divorced', 'married', 'single', 'unknown' ; ('divorced' means divorced or widowed and 'unknown' means missing value)*

- Education : Education details of the customer, it is a categorical data type. Categories include- *‘primary’, 'secondary', 'tertiary', 'unknown' (unknown refers to missing value).*
- Default:it is a categorical data type. ‘yes’ for the customers who have credit in default, ‘no’ for customers who have no credit in default and ‘unknown’ for missing data.
- Balance: gives information about the client’s account balance; it is a numerical data type.
- Housing: it is a categorical data type. ‘yes’ for customers who have a housing loan, ‘no’ for customers who have no housing loan and ‘unknown’ for missing data.
- Loan:it is a categorical data type. ‘yes’ for customers who have a personal loan, ‘no’ for customers who have no personal loan and ‘unknown’ for missing data.

Campaign-related Attributes:

- Contact: gives information about the type of communication, it is a categorical data type. Categories include- *'Cellular','telephone','unknown'*
- Day: refers to day of the month (1 to 31), it is a numerical data type
- Month: gives details about the last contact month of the year, it is a categorical data type. Categories include- *'jan', 'feb', 'mar',…, 'nov', 'dec'*
- Duration: gives details about last contact duration (in seconds); it is a numeric data type.

Target Variable:

- Deposit: It is a binary data type (Yes or no). It gives the information about whether or not a customer subscribes to a term deposit.

Other Attributes:

- Campaign: gives information about the number of times the customer is contacted during the campaign (includes last contact), it is numeric data type.
- Pdays: gives information about the number of days that passed by after the client was last contacted from a previous campaign (-1 means client was not previously contacted), it is a numeric data type.
- Previous: gives information about the number of contacts performed before this campaign, it is a numeric data type.
- Poutcome: gives information about the outcome of the previous marketing campaign, it is a categorical data type. Categories include- *'failure','other','success','unknown'*

Model Selection

Why do we choose logistics regression?

We chose a binary logistics regression because:

- The variables are not linearly distributed.
- The target variable is a binary variable.
- We have explanatory X-variables that we think are related to the Y-variable.
- There is no multicollinearity among independent variables.

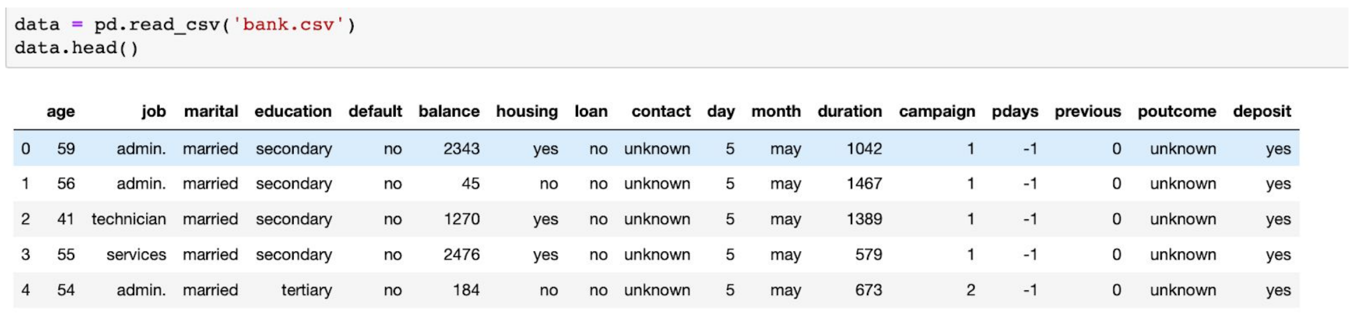
Building the Logistic Regression Model

We've decided to use a logistic regression model because it would be the most efficient method for analyzing relationships between various predictor variables and the independent variable. Furthermore, the term deposit outcome of "yes" or "no" tells us that the target variable is binary.

After encoding the categorical attributes to have a numerical value, we assign the test set and train set to run a logistic regression model. For the test set, we selected the variables we found that they could be good indicators for the outcome variable and assigned the y variable to determine whether or not customers will subscribe to a term deposit.

Data Preprocessing

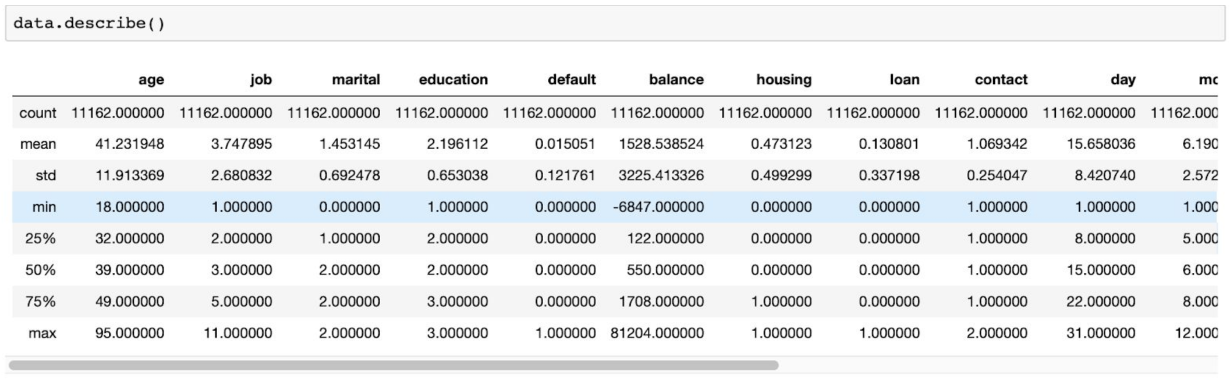
As we review the full dataset for each customer, we see that the spreadsheet does not have the above-listed attributes properly added or categorized (i.e. no "header" available to be read). Also, most of the categorical values have missing values that need to be imputed or deleted accordingly. Furthermore, each of the variables is parsed into various forms: numeric, categorical, and binary. Because of this, we need to find a way to scan through the data and assign what is important. What values do we need to analyze, and why are they helpful for us to know?
We read the data by using pandas as 'pd' and naming the dataframe as 'df.' The code df.head() applies to the first five rows of the dataframe to see how it looks.



```
data = pd.read_csv('bank.csv')
data.head()
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	deposit
0	59	admin.	married	secondary	no	2343	yes	no	unknown	5	may	1042	1	-1	0	unknown	yes
1	56	admin.	married	secondary	no	45	no	no	unknown	5	may	1467	1	-1	0	unknown	yes
2	41	technician	married	secondary	no	1270	yes	no	unknown	5	may	1389	1	-1	0	unknown	yes
3	55	services	married	secondary	no	2476	yes	no	unknown	5	may	579	1	-1	0	unknown	yes
4	54	admin.	married	tertiary	no	184	no	no	unknown	5	may	673	2	-1	0	unknown	yes

Printing basic statistical details

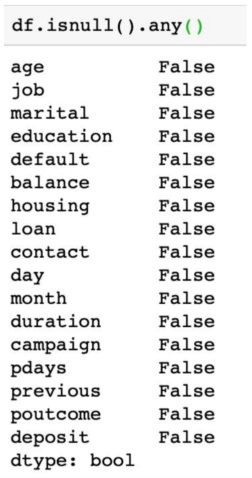


```
data.describe()
```

	age	job	marital	education	default	balance	housing	loan	contact	day	mc
count	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000
mean	41.231948	3.747895	1.453145	2.196112	0.015051	1528.538524	0.473123	0.130801	1.069342	15.658036	6.190
std	11.913369	2.680832	0.692478	0.653038	0.121761	3225.413326	0.499299	0.337198	0.254047	8.420740	2.572
min	18.000000	1.000000	0.000000	1.000000	0.000000	-6847.000000	0.000000	0.000000	1.000000	1.000000	1.000
25%	32.000000	2.000000	1.000000	2.000000	0.000000	122.000000	0.000000	0.000000	1.000000	8.000000	5.000
50%	39.000000	3.000000	2.000000	2.000000	0.000000	550.000000	0.000000	0.000000	1.000000	15.000000	6.000
75%	49.000000	5.000000	2.000000	3.000000	0.000000	1708.000000	1.000000	0.000000	1.000000	22.000000	8.000
max	95.000000	11.000000	2.000000	3.000000	1.000000	81204.000000	1.000000	1.000000	2.000000	31.000000	12.000

Handling Missing Data

First, we need to check for missing data from the dataset. The code df.isnull().any() is to find the columns that have NaN values and returns a boolean value. Since a boolean value is returned as "False" for each attribute, it is clear that the data frame does not have any null values. However, "unknown" categorical values are present in the job and education variables. Because they account for only 4.06% of the overall data, we've decided to replace them with mode.



```
df.isnull().any()
```

```
age          False
job          False
marital      False
education    False
default      False
balance      False
housing      False
loan         False
contact      False
day          False
month        False
duration     False
campaign     False
pdays        False
previous     False
poutcome     False
deposit      False
dtype: bool
```

Handling Data Type

Some attributes need to be converted to numeric data depending on the model that we will be using. The code df.dtypes checks the type of attributes in the data frame. The following return shows that they are mixing categorical and numeric data.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11162 entries, 0 to 11161
Data columns (total 17 columns):
age          11162 non-null int64
job          11162 non-null object
marital      11162 non-null object
education    11162 non-null object
default      11162 non-null object
balance      11162 non-null int64
housing      11162 non-null object
loan         11162 non-null object
contact      11162 non-null object
day          11162 non-null int64
month        11162 non-null object
duration     11162 non-null int64
campaign     11162 non-null int64
pdays        11162 non-null int64
previous     11162 non-null int64
poutcome     11162 non-null object
deposit      11162 non-null object
dtypes: int64(7), object(10)
memory usage: 1.4+ MB
```

Dealing with Categorical data

Many machine learning algorithms do not support categorical data; thus, we need to encode categorical data. We are planning to use a logistics regression model, so any X variables should be integers and the y variables should be binary. For a binary logistics regression, factor level 1 of the dependent variable should represent the desired outcome. Therefore, we've recorded the deposit variables "yes" as 1 and "no" as 0.

```
cleanup = {"marital":     {"married": 2, "single": 1, "divorced" :0},
                 "job": {"management": 1, "blue-collar": 2,"technician": 3, "admin.":4 , "services" : 5, "retired" : 6,
                  "education": {"secondary": 2, "tertiary":3, "primary" :1 , "unknown": 2 },
                   "default": {"no":0 , "yes": 1},
                   "housing": {"no" :0 , "yes" :1},
                   "loan": {"no":0 , "yes":1},
                  "deposit": {"no":0 , "yes":1},
                   "month":{"may":5 , "aug":8, "jul":7, "jun":6, "nov":11, "apr": 4, "feb":2, "oct":10, "jan":1, "sep":
                    "poutcome":{"unknown": 2, "failure":0, "success": 1 , "other": 3},
                    "contact": {"cellular":1, "telephone":2, "unknown":1}}
```

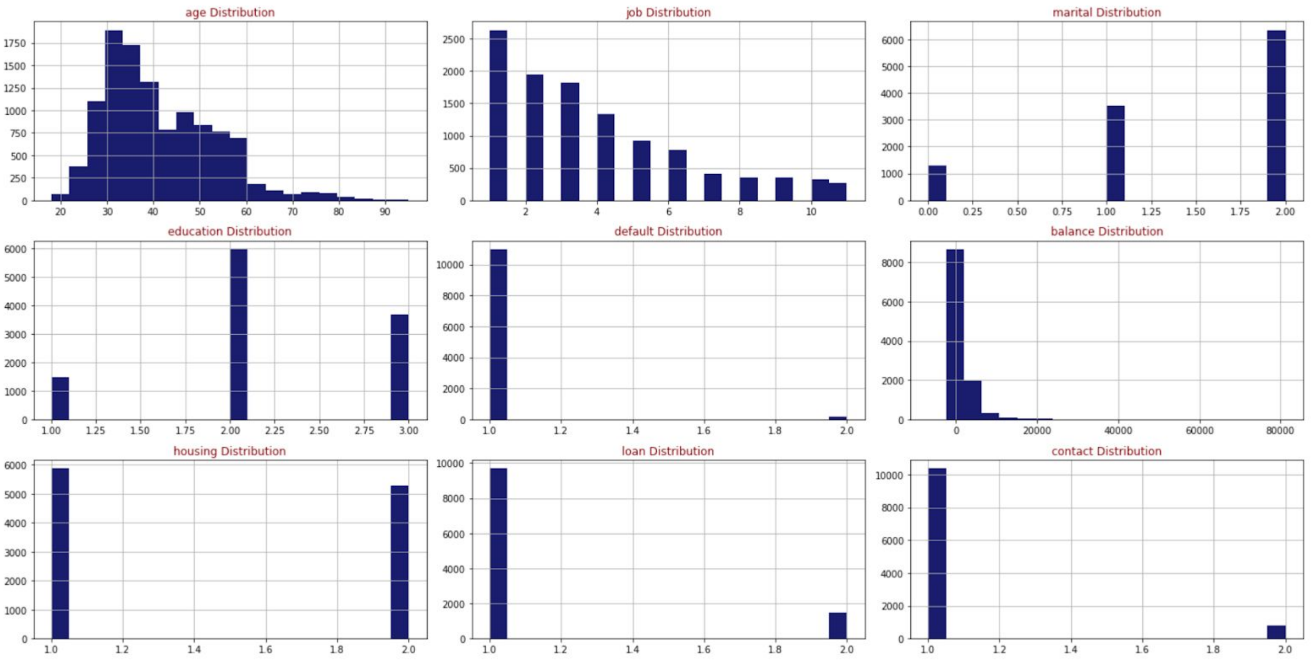
```
data.replace(cleanup, inplace=True)
```

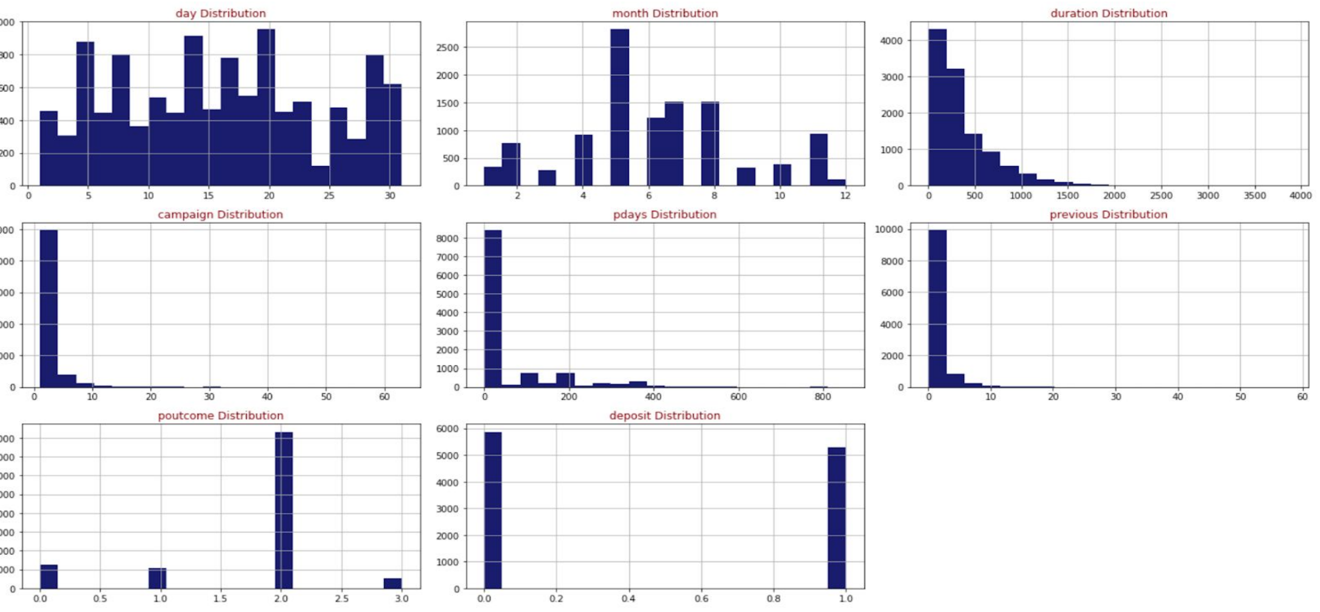
```
data.head()
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	deposit
0	59	4	2	2	0	2343	1	0	1	5	5	1042	1	-1	0	2	1
1	56	4	2	2	0	45	0	0	1	5	5	1467	1	-1	0	2	1
2	41	3	2	2	0	1270	1	0	1	5	5	1389	1	-1	0	2	1
3	55	5	2	2	0	2476	1	0	1	5	5	579	1	-1	0	2	1
4	54	4	2	3	0	184	0	0	1	5	5	673	2	-1	0	2	1

Exploratory Analysis

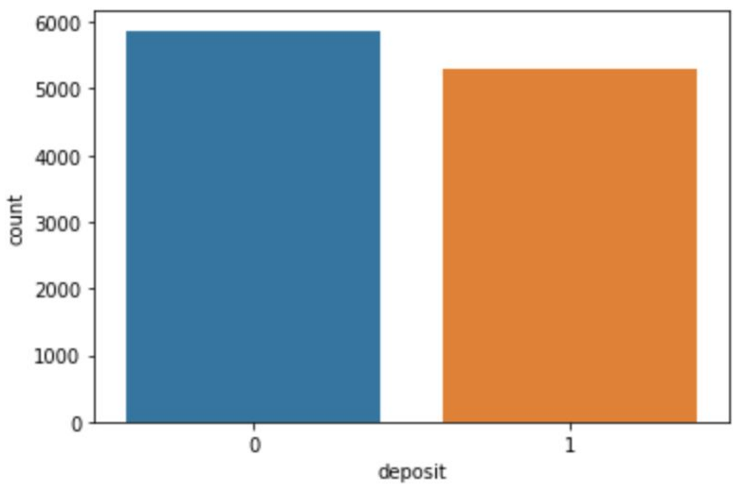
Checking normal distribution:





Check count by deposit

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c265721d0>
```



Checking multicollinearity

```
sn.pairplot(data=data)
```

Splitting the attributes into independent and dependent attributes

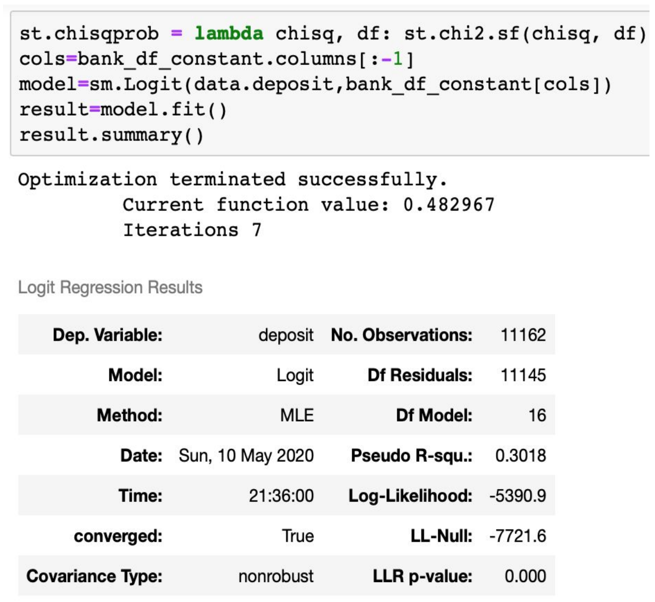
```
# Splitting the attributes into independent and dependent attributes
x = data.iloc[:, :-1].values # attributes to determine dependent variable, Takes all rows of all columns except last co
y = data.iloc[:, -1].values # dependent variable / Class, Takes all rows of the last column
```

<u>Splitting the dataset into training set and test set:</u>

We are using a 7.5/2.5 ratio to distribute train and test sets. Setting the training data set to be 75% and 25% for the test set will avoid model overfitting.

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state= 10)
```

Logistic Regression Model

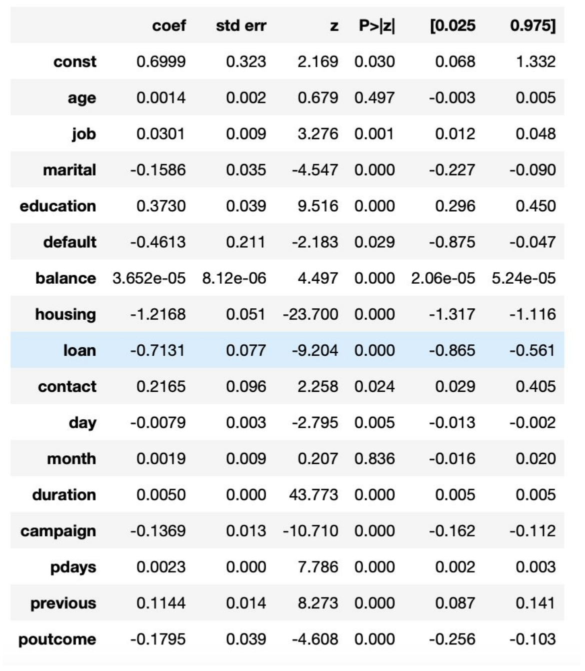


```
st.chisqprob = lambda chisq, df: st.chi2.sf(chisq, df)
cols=bank_df_constant.columns[:-1]
model=sm.Logit(data.deposit,bank_df_constant[cols])
result=model.fit()
result.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.482967
         Iterations 7
```

Logit Regression Results

Dep. Variable:	deposit	**No. Observations:**	11162
Model:	Logit	**Df Residuals:**	11145
Method:	MLE	**Df Model:**	16
Date:	Sun, 10 May 2020	**Pseudo R-squ.:**	0.3018
Time:	21:36:00	**Log-Likelihood:**	-5390.9
converged:	True	**LL-Null:**	-7721.6
Covariance Type:	nonrobust	**LLR p-value:**	0.000



	coef	std err	z	P>|z|	[0.025	0.975]
const	0.6999	0.323	2.169	0.030	0.068	1.332
age	0.0014	0.002	0.679	0.497	-0.003	0.005
job	0.0301	0.009	3.276	0.001	0.012	0.048
marital	-0.1586	0.035	-4.547	0.000	-0.227	-0.090
education	0.3730	0.039	9.516	0.000	0.296	0.450
default	-0.4613	0.211	-2.183	0.029	-0.875	-0.047
balance	3.652e-05	8.12e-06	4.497	0.000	2.06e-05	5.24e-05
housing	-1.2168	0.051	-23.700	0.000	-1.317	-1.116
loan	-0.7131	0.077	-9.204	0.000	-0.865	-0.561
contact	0.2165	0.096	2.258	0.024	0.029	0.405
day	-0.0079	0.003	-2.795	0.005	-0.013	-0.002
month	0.0019	0.009	0.207	0.836	-0.016	0.020
duration	0.0050	0.000	43.773	0.000	0.005	0.005
campaign	-0.1369	0.013	-10.710	0.000	-0.162	-0.112
pdays	0.0023	0.000	7.786	0.000	0.002	0.003
previous	0.1144	0.014	8.273	0.000	0.087	0.141
poutcome	-0.1795	0.039	-4.608	0.000	-0.256	-0.103

Removing month and age variables, since its p-value is higher than 0.05
Using backward elimination (p-value approach)

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.7714	0.306	2.520	0.012	0.171	1.371
job	0.0304	0.009	3.312	0.001	0.012	0.048
marital	-0.1552	0.035	-4.492	0.000	-0.223	-0.087
education	0.3684	0.039	9.562	0.000	0.293	0.444
default	-0.4613	0.211	-2.182	0.029	-0.876	-0.047
balance	3.712e-05	8.08e-06	4.592	0.000	2.13e-05	5.3e-05
housing	-1.2229	0.051	-24.153	0.000	-1.322	-1.124
loan	-0.7141	0.077	-9.217	0.000	-0.866	-0.562
contact	0.2282	0.094	2.416	0.016	0.043	0.413
day	-0.0079	0.003	-2.786	0.005	-0.013	-0.002
duration	0.0050	0.000	43.784	0.000	0.005	0.005
campaign	-0.1369	0.013	-10.718	0.000	-0.162	-0.112
pdays	0.0023	0.000	7.782	0.000	0.002	0.003
previous	0.1146	0.014	8.285	0.000	0.087	0.142
poutcome	-0.1804	0.039	-4.639	0.000	-0.257	-0.104

Model Evaluation Logistic Regression

```
from sklearn.linear_model import LogisticRegression
logreg=LogisticRegression(solver='liblinear')
logreg.fit(x_train,y_train)
y_pred=logreg.predict(x_test)
```

```
sklearn.metrics.accuracy_score(y_test,y_pred)
```

```
0.79648871372268
```

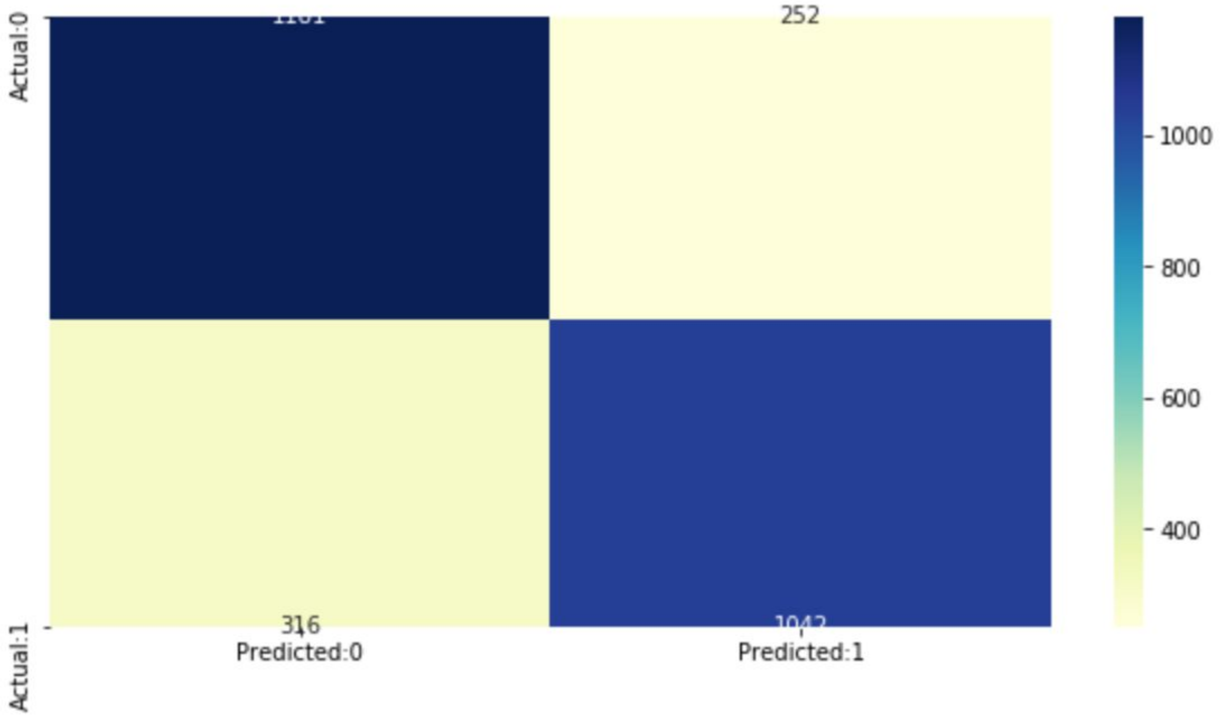
The logistics regression model accuracy is ~80%.

Odds Ratios and Confidence Interval 95%

```
params = np.exp(result.params)
conf = np.exp(result.conf_int())
conf['OR'] = params
pvalue=round(result.pvalues,3)
conf['pvalue']=pvalue
conf.columns = ['CI 95%(2.5%)', 'CI 95%(97.5%)', 'Odds Ratio','pvalue']
print ((conf))
```

```
           CI 95%(2.5%)  CI 95%(97.5%)  Odds Ratio  pvalue
const          1.186934       3.940884    2.162769   0.012
job            1.012496       1.049612    1.030887   0.001
marital        0.800173       0.916232    0.856238   0.000
education      1.340255       1.558724    1.445368   0.000
default        0.416593       0.954207    0.630489   0.029
balance        1.000021       1.000053    1.000037   0.000
housing        0.266578       0.325100    0.294389   0.000
loan           0.420660       0.569930    0.489639   0.000
contact        1.044013       1.511717    1.256285   0.016
day            0.986664       0.997666    0.992150   0.005
duration       1.004787       1.005237    1.005012   0.000
campaign       0.850502       0.894167    0.872061   0.000
pdays          1.001694       1.002836    1.002265   0.000
previous       1.091407       1.152203    1.121393   0.000
poutcome       0.773646       0.901047    0.834920   0.000
```

Confusion Matrix



True Positives: 1042 - *accurately predicted that customers would subscribe to a term deposit*

True Negatives: 1101 - *accurately predicted that customers would NOT subscribe to a term deposit*

False Positives: 252 (*Type I error*) - *inaccurately predicted that customers would subscribe to a term deposit*

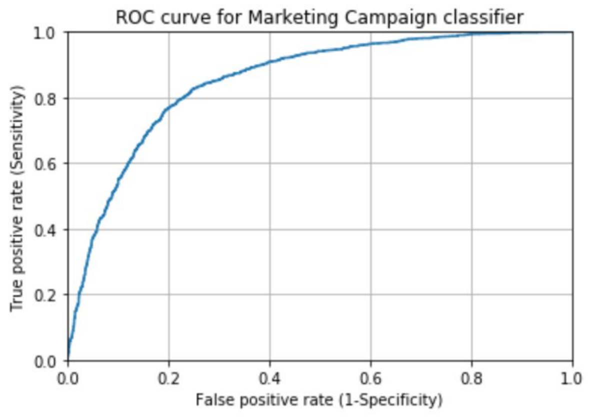
False Negatives: 316 (*Type II error*) - *inaccurately predicted that customers would NOT subscribe to a term deposit*

Accuracy: (TP + TN) / (TP + TN + FP + FN) = 2143 / 2711 = **0.79**

Error: (FP + FN) / (TP + TN + FP + FN) = 568 / 2711 = **0.21**

ROC Curve

```
from sklearn.metrics import roc_curve
fpr, tpr, thresholds = roc_curve(Y_test,y_pred_prob_yes[:,1])
plt.plot(fpr,tpr)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.title('ROC curve for Marketing Campaign classifier')
plt.xlabel('False positive rate (1-Specificity)')
plt.ylabel('True positive rate (Sensitivity)')
plt.grid(True)
```



AUC

```
sklearn.metrics.roc_auc_score(y_test,y_pred_prob_yes[:,1])
```

```
0.8676057828977592
```

AUC is 0.87 which indicates a good model.

To analyze our model's performance and prediction, a ROC curve is used as a metric to evaluate. We follow that by taking a look at the area under the curve (AUC) and see that the higher AUC shows that the model is better at accurately predicting 0s and 1s (true positives and false positives). The AUC score of 0.87 indicates a strong model.

Results and Future Suggestions

Using the banking dataset, we ran a logistic regression model and used a confusion matrix and ROC curve/AUC as metrics to determine the overall accuracy of our predictions. Our logistic regression model and confusion matrix gave us an accuracy of 80% while the AUC score was an 87% accuracy.

The independent variables that we analyzed were job, marital, education, default, balance, month, contact, day, poutcome, housing, loan, duration, campaign, pdays, previous. Since their p-values are all less than the average significance level of 0.05 (5%), we can conclude that they can all be statistically significant to our outcome variable of accepting/denying a term deposit. However, the p-value of month and age attributes are 0.836 and 0.497 respectively, which is greater than the 0.05 significance level. Therefore, we excluded variables month, age and then built the binary logistic regression model with all other independent variables once again.

Taking a closer look using the odds ratio, variables with an odds ratio greater than 1 indicates that as the independent variable increases, a customer is more likely to subscribe to another term deposit with the bank. Categories with a value greater than 1 are: **job**, **education**, **contact**, **balance**, **duration**, **pdays**, and **previous**.

We are able to determine a few different things to recommend to the bank for its future marketing efforts and to better guarantee the likeliness of customers subscribing to term deposits:

- Important categories for this bank to keep in mind are a customer's occupation, education level, account balance, days since the last campaign, and previous contact methods since the last campaign
- For future scope and planning, we would recommend that the bank collects at least 3-4 years' worth of data in order to make more accurate analyses and predictions regarding their marketing campaigns
- This analysis may or may not be applicable towards other global markets as the collected dataset was confined to only one financial institution in Portugal
- This banking institution can continue to further analyze the predictor variables to search for patterns and help identify key target audiences for marketing campaigns such as:
    - Advertising term deposits and other banking services to specific industries such as health industry, pharmaceuticals, etc.
    - Customers with account balances above 'x' amount
    - A/B testing different communications methods to customers to measure efficiency and success rate (i.e. email versus phone call)
- There is no strong evidence indicating that phone calls are a reliable method to use since this is the only strategy that was measured; no useful benchmark to weigh these results against
- The bank can and should be more creative and innovative in how they decide to reach out to customers; strategies can be further tested using similar methods (running logistic regression model) to determine their effectiveness