PREDICTING CLIENT SUBSCRIPTION TO BANK TERM DEPOSITS - A BINARY CLASSIFICATION

A Capstone Project - by Chamila Wijayawardhana

About the project

- This is a binary classification problem where the goal is to predict if a client will subscribe to a term deposit or not.
- The data is related to direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls.
- Initial analysis of the data revealed there is a significant class imbalance with a **88 : 12 split** of the majority class (‘no’ to term deposit) and the minority class (‘yes’ to term deposits).
- Hence this problem is identified as a **binary classification problem with a class imbalance.**

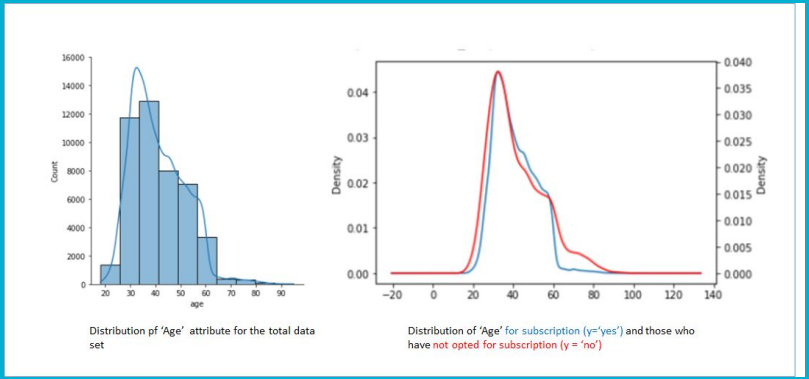
Feature Variables

- Data set comprises of 16 input variable and 45211 instances
- Are of 3 different types
  1. Related bank client data
  2. Related to the last contact of the current campaign
  3. Related to current campaign and the previous campaigns
- Consists of both “Numerical” and ‘Categorical’ variables

#	Varaible	Description	Data Type
1	age	Age	Numeric (int64)
2	job	Type of Job	Categorical - Ordinal (object)
3	marital	Marital status	Categorical - Nominal (object)
4	education	Education	Categorical - Ordinal (object)
5	default	Has credit in default?	Categorical (object)
6	balance	*No information provided	Numeric (int64)
7	housing	Has housing loan?	Categorical (object)
8	loan	Has personal loan	Categorical (object)
9	contact	Contact communication type	Categorical (object)
10	day	day of the last contact month	Numeric (int64)
11	month	last contact month of year	Categorical (object)
12	duration	last contact duration, in seconds	Numeric (int64)
13	campaign	Number of contacts performed during this campaign and for this client	Numeric (int64)
14	pdays	Number of days that passed by after the client was last contacted from a previous campaign	Numeric (int64)
15	previous	Number of contacts performed before this campaign and for this client	Numeric (int64)
16	poutcome	Outcome of the previous marketing campaign	Categorical (object)

Findings from Descriptive Statistics

Following numerical variables are analyzed - : (a) Age, (b) Balance, (c) Campaign, (d) Pdays and (e) Previous.



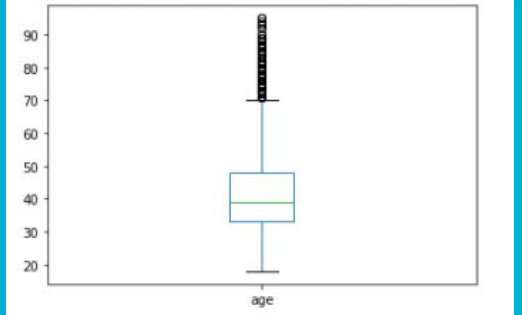
Distribution pf 'Age' attribute for the total data set

Distribution of 'Age' for subscription (y='yes') and those who have not opted for subscription (y = 'no')

The **numerical attributes** that were analyzed **do not have 'null' values.**

```
data['age'].plot(kind='box')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f15b99
```



- Analyses of the **'age' variable** for the population and for the clients who have subscribed for (y = 'yes') and not subscribed y='no') revealed that they have similar probability distributions
- Analyses of the **'balance' attribute** revealed that clients who subscribed to the term deposit (y = 'yes') either **had minus or lower account balances**.
- Analyses of the '**pdays' and 'previous' attributes** revealed that **majority of the clients (82%) have not been contacted in the previous campaign**

All of the variables have a significant **number of 'outliers'**

Exploring Categorical Variables

- Following categorical variables were identified from the data set : **(a) Job, (b) Marital, (c) Education, (d) Default, (e) Housing, (f) Loan (g) Contact, (h) Month (i) poutcome**
- Analysis of these attributes revealed there are a significant **number of "unknown" values** associated. These values are **treated as null values**.
- For the **'job'** and **'education'** attributes, these null values are **replaced with the highest frequency category** of each of the attributes.
- The '**poutcome'** attribute (Outcome of the previous marketing campaign) has **82% of unknown values** and hence will **not be considered** for building the machine learning model.

Data Pre-processing / Feature Engineering

Treatment of numerical variables

All the **numerical variables** were **standardized on Z-score normalization** as the probability distributions of all these variables are skewed.

Treatment of categorical variables

(a) Replacement of null values - Null values in **'job'** and **'education'** are **imputed** by the **most frequently occurring category** for each attribute.

(b) Grouping - There are **11 different job categories** as listed in the data set and they are **grouped into 3 categories** as follows, Job_category1 - {unemployed, students, housemaids} , Job_category 2 - {bluecollar, technician, services, admin, retired}, Job_category 3 - {management, self employed,entrepreneur}

(c) Encoding - All of the selected **categorical variables are one-hot encoded** for building the machine learning model

Consideration correlation of attributes with the target variable, following numerical and categorical attributes were selected for the initial model building :

```
'age','balance','campaign','previous', 'job_group', 'education',
'default', 'housing', 'loan'
```

Model selection and Training

- As there is significant **class imbalance** in this classification problem, with with class 0 (y = no) and class 1 (y = yes) having 88 : 12 percent split, **‘Random Forest’ algorithm** is primarily selected for building the machine learning model as its ensemble properties may assist in negating the the imbalance.
- Model was trained on a 70 : 30 split of training and testing data.
- As the problem is that of a class imbalance the training data set was **resampled using the Random Undersampling technique** for the majority class.
- However this resampling did not improve the model performance for both the classes and for further processing in the pipeline the original train test split was retained.

Hyperparameter tuning and selecting the best model

For the purpose of selecting the best model (parameters) , multiple Random Forest classifiers with the below listed hyper parameters were tested.

- N_estimators: The number of trees in the forest
- The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

Model Name	N_estimators	Maximum depth
rf1	100	None
rf2	100	10
rf3	500	5
rf4	500	10
rf5	500	15

- Alternatively a **grid search based cross validation** was also performed on the model to select/validate the optimal parameters.

Model Evaluation

As this is a **binary classification problem** with a **class imbalance**, model **accuracy will not be considered** as an evaluation metric. Only the (a) model precision, (b) f1 score and the (c) ROC AUC (Area under the ROC Curve) metrics are considered for evaluating the best model.

From manual hyperparameter tuning, the evaluation table was obtained.

	model_name	model	accuracy	precision	f1_score	roc_auc
0	lgr1	LogisticRegression(n_jobs=3, verbose=1)	0.880861	0.235294	0.827428	0.692686
1	rf1	(DecisionTreeClassifier(max_features='auto', r...	0.865600	0.348587	0.843375	0.661394
2	rf2	(DecisionTreeClassifier(max_depth=10, max_feat...	0.882704	0.524476	0.837052	0.718848
3	rf3	(DecisionTreeClassifier(max_depth=5, max_featu...	0.882262	1.000000	0.827149	0.715145
4	rf4	(DecisionTreeClassifier(max_depth=10, max_feat...	0.882409	0.510490	0.836642	0.720152
5	rf5	(DecisionTreeClassifier(max_depth=15, max_feat...	0.880271	0.462857	0.844886	0.704867

By cross validation with grid search

```
#Best Model Parametrs
print(gs_model.best_params_)

{'max_depth': 10, 'n_estimators': 100}
```

Random Forest Classifier - 'rf2' is selected as the best model.

Confusion matrix

	precision	recall	f1-score	support
0	0.89	0.99	0.94	11966
1	0.46	0.05	0.08	1598
accuracy			0.88	13564
macro avg	0.68	0.52	0.51	13564
weighted avg	0.84	0.88	0.84	13564

```
[[11882    84]
 [ 1525    73]]
```

Model performs well on the majority class, but performs rather poorly on the minority class

Model deployment and Explaining the machine learning model

As part of this exercise following components were developed which can be used to create **an inference pipeline** for the purpose of model deployment

1. Pre-processing function
2. Machine Learning model (which is saved using “Joblib” and “pickle”)
3. Predict / Scoring function
4. Post processing function

Additionally the following component can be further developed to create an end to end machine learning solution.

1. A user interface to get input data
2. API that takes in user inputs and invoke the inference pipeline (listed above)
3. An output user interface that displays the output from the inference pipeline. The output can also be further improved by analysing the output with LIME or SHAP and giving end users the reasons / factors that contributed to the predicted output.

The solution can be deployed either locally or on cloud infrastructure, either on virtualized or containerized hardware depending on the particular client requirements.