CS-513: KDDM Final Project

Analysis of Bank Marketing Data

What is Bank Term deposit?

"Bank term deposit refers to the money held back by the bank, which is deposited by a client, in order to safe guard their money for various reasons."

About project

Clients can subscribe to bank term deposits depending upon their needs, and period of time. Once that term ends, the client gets back benefits against the money withheld by the bank. Benefits may include interest amount, safer option of saving, creating deposits to use in future as per need, etc.

About project

Therefore, we want to predict if a certain client is going to subscribe for bank term deposit or not, using certain given details. Given, a set of features we want to predict the target variable as true, which is subscriber or false which is non-subscriber.

Dataset in use

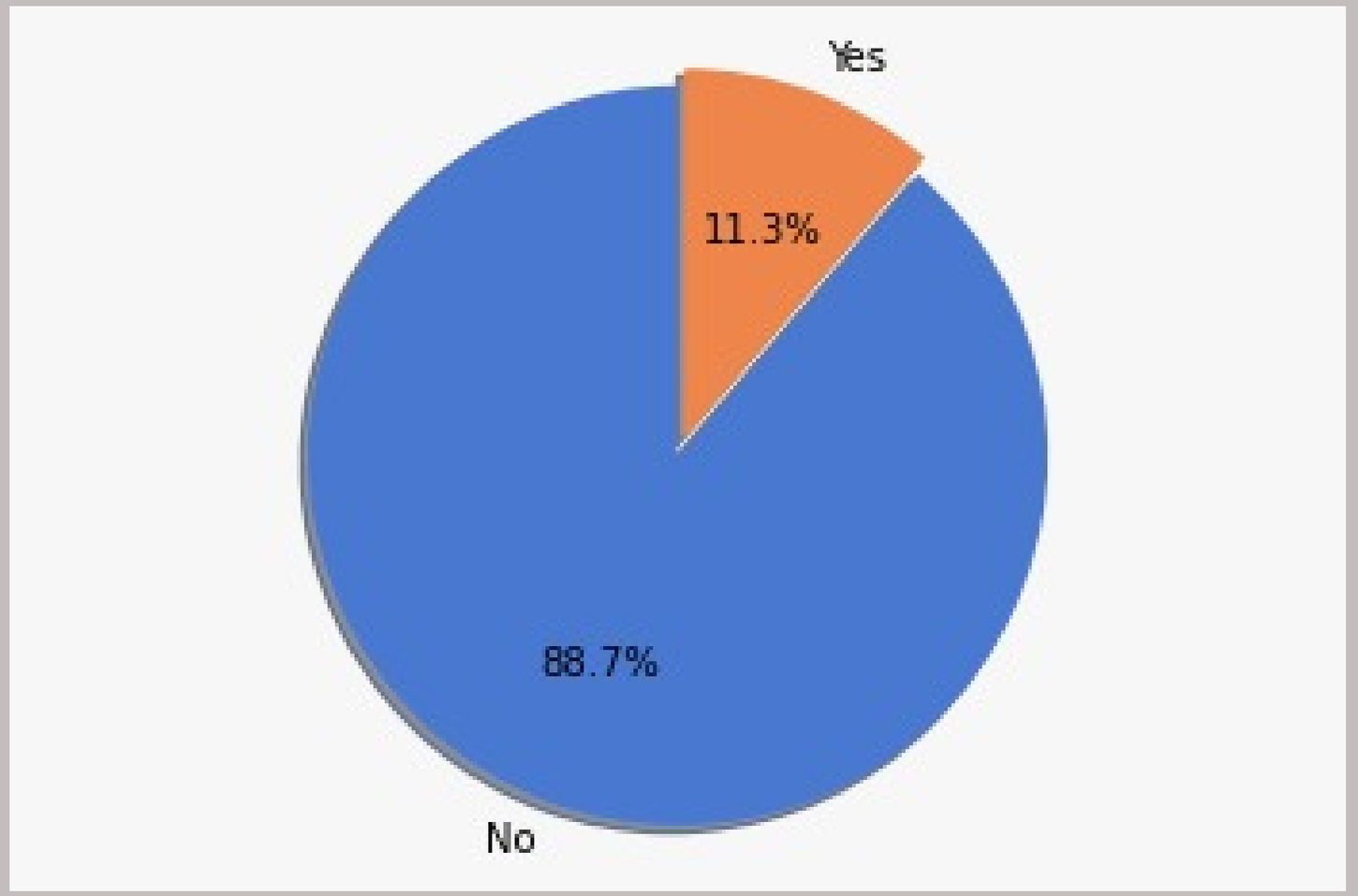
The dataset that we have used is based on phone calls made by a Portuguese banking institution during a direct marketing campaign. Our dataset comprises 21 features in the form of columns (including our desired target column).

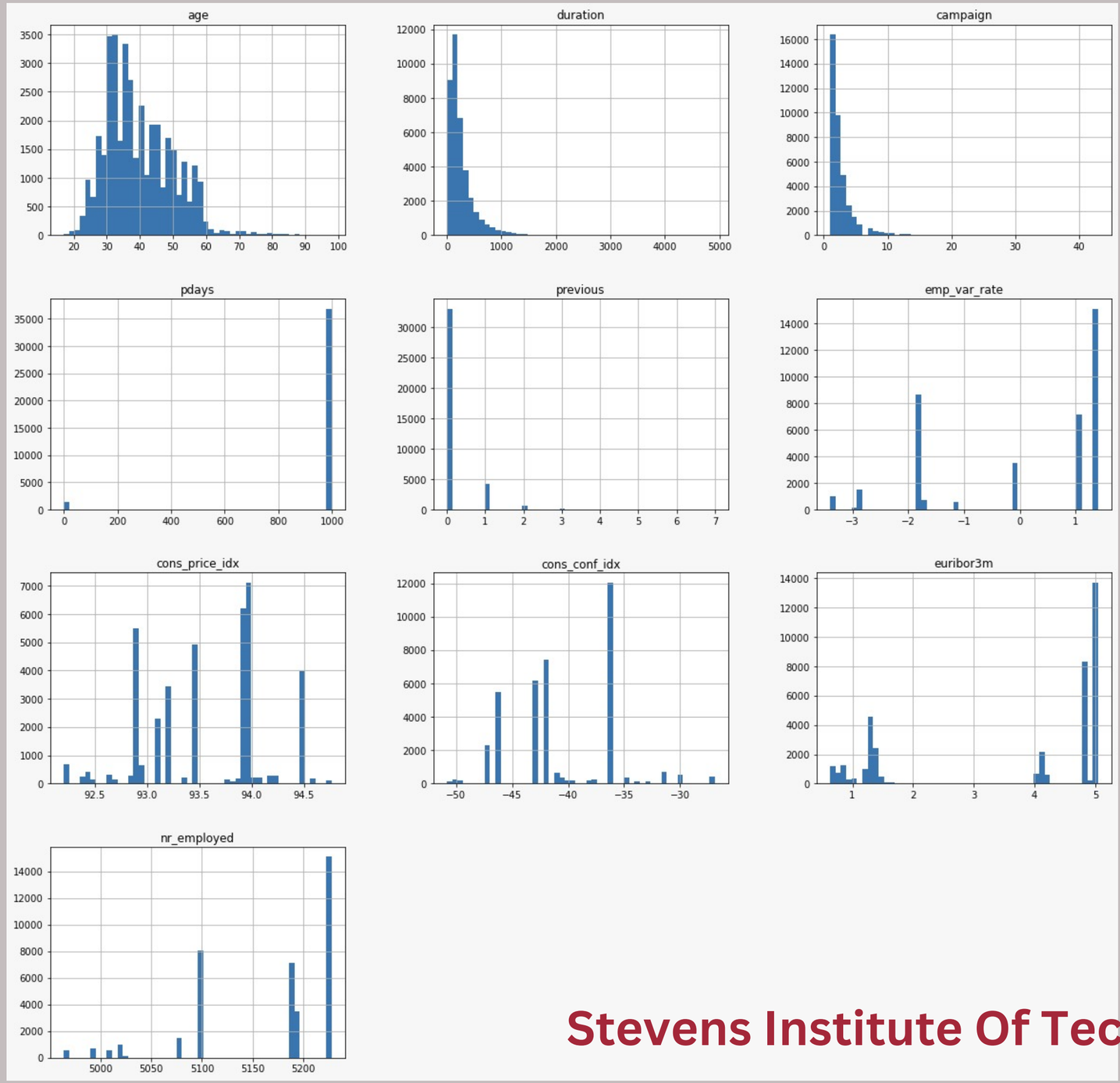
Before data pre-processing :

- Total no of rows - 41,190
- Total no of columns - 21

Exploratory Data Analysis (EDA)

Column Name	Unique Values
jobs	12
marital_counts	4
education_counts	8
default_counts	3
housing_counts	3
loan_counts	3
contact_counts	2
month_counts	10
day_of_week_counts	5
campaign_counts	42
pdays_counts	27
previous_counts	8
poutcome_counts	3
emp_var_rate_counts	10
cons_price_idx_counts	26
cons_conf_idx_counts	26
nr_employed_counts	11
y_counts	2




age
duration
campaign
pdays
previous
emp_var_rate
cons_price_idx
cons_conf_idx
euribor3m
nr_employed

Data Pre-Processing and Normalization

We pre-processed our data and replaced NaN values with a mode of the 'default' column, and for other NaN values, we dropped those rows so as to avoid excessive manual manipulation of data.
We have normalized the data using Min-Max normalization and Z-score normalization.
After data pre-processing and normalization

- Total no of rows - 38,245
- Total no of columns - 21

Algorithms implemented

01 Decision Trees in R

02 Decision Trees in Python

03 Naïve Bayes in R

04 Naïve Bayes in Python

05 K-nearest neighbor in R

06 K-nearest neighbor in Python

07 Random Forest in R

08 Random Forest in Python

Decision Trees

A decision tree partitions the training set until each partition consists dominantly of examples from one class. This algorithm is a class discriminator. The data is partitioned and determined by a non-leaf node of the tree which contains a split point that is a test on one or more attributes.
We are using a Decision tree classifier in python and Classification and regression tree(CART) in R.

Decision Tree Results

Decision Trees				
	R		Python	
	Min-Max Normalized	Z-Score Normalized	Min-Max Normalized	Z-Score Normalized
Error Rate	0.08873006188	0.08873006188	0.1085933415	0.1079832665
Accuracy	91.12699381	91.12699381	89.14066585	89.20167335
Precision	0.599086758	0.599086758	0.4983579639	0.4972462628
Recall	0.5311740891	0.5311740891	0.4887278583	0.512987013
F1 - Measure	0.5630901288	0.5630901288	0.493495935	0.5049940072

Naive Bayes

Naïve Bayes is based on the Bayes theorem and is a probabilistic machine learning algorithm, used in a wide variety of classification tasks. This is also well-known to make quick predictions on a huge dataset and is relatively simple to execute. It is based on the assumption of independence among predictors.
We are using e1071 library in R and using sklearn.naive bayes library in python

Naive Bayes Results

Naïve Bayes				
	R		Python	
	Min-Max Normalized	Z-Score Normalized	Min-Max Normalized	Z-Score Normalized
Error Rate	0.1295214852	0.1295214852	0.120097612	0.2896112951
Accuracy	87.04785148	87.04785148	87.9902388	71.03887049
Precision	0.428733674	0.428733674	0.4463705309	0.2551657857
Recall	0.6113360324	0.6113360324	0.3221266615	0.826459144
F1 - Measure	0.5040053405	0.5040053405	0.3742052679	0.3899394162

K-Nearest Neighbors

KNN algorithm is based on a supervised learning method that takes a non-parametric approach widely used for classification and regression. In the classification phase, k is a constant defined by the user, and an unlabeled vector is classified by assigning the label which is most frequent among the k training samples nearest to that query point.
We are using kknn library in R and using math library in python for KNeighborsClassifier.

K-Nearest Neighbors Results

K-Nearest Neighbors				
	R		Python	
	Min-Max Normalized	Z-Score Normalized	Min-Max Normalized	Z-Score Normalized
Error Rate	0.101882517	0.1009238278	0.1142583232	0.105281506
Accuracy	89.8117483	89.90761722	88.57416768	89.4718494
Precision	0.539614561	0.5690376569	0.458781362	0.5290519878
Recall	0.4054706356	0.4217054264	0.2023715415	0.4102766798
F1 - Measure	0.463022508	0.4844167409	0.2808557323	0.4621549421

Random Forest

Random forest is an umbrella term for learning methods for classification, regression, and other tasks that are executed by creating a multitude of decision trees at training time. For classification, the class selected by most trees is usually the output of the random forest. For regression, the mean or average prediction of the individual trees is returned.
We are using randomforest library in R and using RandomForestClassifier in python

Random Forest Results

Random Forest				
	R		Python	
	Min-Max Normalized	Z-Score Normalized	Min-Max Normalized	Z-Score Normalized
Error Rate	0.08533077661	0.08428484267	0.09159839637	0.08549764685
Accuracy	91.46692234	91.57151573	90.84016036	91.45023531
Precision	0.632231405	0.637014314 9	0.6572827417	0.6828978622
Recall	0.4955465587	0.5044534413	0.4105504587	0.4460822343
F1 - Measure	0.5556059918	0.5630366019	0.5054117647	0.5396527452

Accuracy of algorithms

Algorithm	R with min max normalization	R with Z-score normalization	Python with min max normalization	Python with Z-score normalization
Decision Trees	91.1	91.1	89.1	89.2
Naive Bayes	87.05	87.05	87.9	71.03
KNN	89.8	89.9	88.6	89.5
Random Forest	91.5	91.6	90.8	91.4

Conclusion

Good data mining models are based on the correlation and the quantity of the dataset. We have kept all the columns and refined the data by processing and normalizing the data.

The results on the same dataset using different models may be diverse. Choosing the programming language influences accuracy.

The best accuracy on our dataset is:

Random Forest in R with Z-score normalization.

Future Scope

- Try to find the right form of bank term deposit based on the information about the customer.
- Try new algorithms on our dataset, and analyze the difference between supervised learning and unsupervised learning.
- Collect more data and attribute of the customer base of bank to train the model to increase accuracy.

Thank You