Bank Marketing Effectiveness Prediction

VINIT LADSE, PRATIKSHA KHARODE, GAURAV BHAKTE

CAPSTONE PROJECT-III

ALMABETTER,BANGLORE

Abstract:

Data from a marketing campaign run by Portugal banking Institution. The campaign’s aim was to increase customers’ subscription rates to fixed-term deposit products, such as CDs. Using knowledge from the course, a number of machine learning algorithms are implemented to answer the question: How can banks successfully market these products in the most efficient way possible and with the highest possible rate of success?

Problem Statements:

The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be ('yes') or not ('no') subscribed. The classification goal is to predict if the client will subscribe a term deposit (variable ‘y’).

Data Summary:

- job : type of job (categorical: 'admin.' , 'blue-collar' , 'entrepreneur' , 'housemaid' , 'management' , 'retired' , 'self employed' , 'services' , 'student' , 'technician' , 'unemployed' , 'unknown')
- marital : marital status (categorical: 'divorced' , 'married' , 'single')
- education : (categorical: 'primary' , 'secondary' , 'tertiary' , 'unknown')
- default: has credit in default? (categorical: 'no' , 'yes')
- housing: has housing loan? (categorical: 'no' , 'yes')
- loan: has personal loan? (categorical: 'no' , 'yes')

- contact: contact communication type (categorical: 'cellular' , 'telephone' , 'unknown')
- month: last contact month of year (categorical: 'jan' , 'feb' , 'mar', ..., 'nov', 'dec')
- poutcome: outcome of the previous marketing campaign (categorical: 'failure' , 'success')

Introduction:

With the startling rise over the last few decades of media and technology which increases the amount of information we have at our fingertips (cell phones, television, Internet, etc.), humans are now more connected than ever. One result of this is that marketing campaigns are growing evermore pervasive in our daily lives.

This glut of advertising has forced businesses to compete for the attention of a populace that has an ever growing amount of distractions. Thus raising the question: How can businesses successfully advertise their products in the most efficient way possible with the highest possible rate of success? We will answer this question in the context of banks advertising fixed term deposit products to their customers.

Using data collected from a previous bank marketing campaign, a number of features centered around the clients, the campaign itself, and general market conditions will be explored. Based on this data, machine learning models will predict which clients will subscribe and what banks can do to increase the rate of subscription.

METHODOLOGY

A. Programming in Python:

Python provides a number of packages and libraries for the convenience of the programmer. The whole project is coded using Python 3. Packages/libraries used are numpy for array manipulation, pandas for dataframe operations, and matplotlib and seaborn for visualization.

The sklearn libraries were also critical in providing packages for machine learning algorithms, tasks, and by giving the user the control to set important attributes of those algorithms as they wished. The dataset is stored in a dataframe and is intensively queried and manipulated using facilities provided by the Python 3 environment. Other data structures such as arrays, lists, and dictionaries are used as needed.

B. Data cleaning and exploratory analysis:

The dataset was provided by the Machine Learning Repository and contained information on 45,211 clients across 17 different features, both categorial (marital status, job type, education, etc.) and numeric (age, number of days since previous contact, etc.). The target variable is a binary “Yes” (client subscribed) or “No” (client did not subscribe). The first step is to load the dataset into a dataframe for easy manipulation and exploration using the pandas package.

The ‘duration’ feature was dropped due to the risk of data leakage. This feature measures the length of the phone call between the bank’s marketing representative and the customer. Since this time cannot be known until after the call has ended (when the outcome for that customer is already known), including it in a predictive model would not provide realistic results. The next step was to explore and clean the categorical variables such as ‘job type,’ ‘marital status,’ ‘education,’ etc.

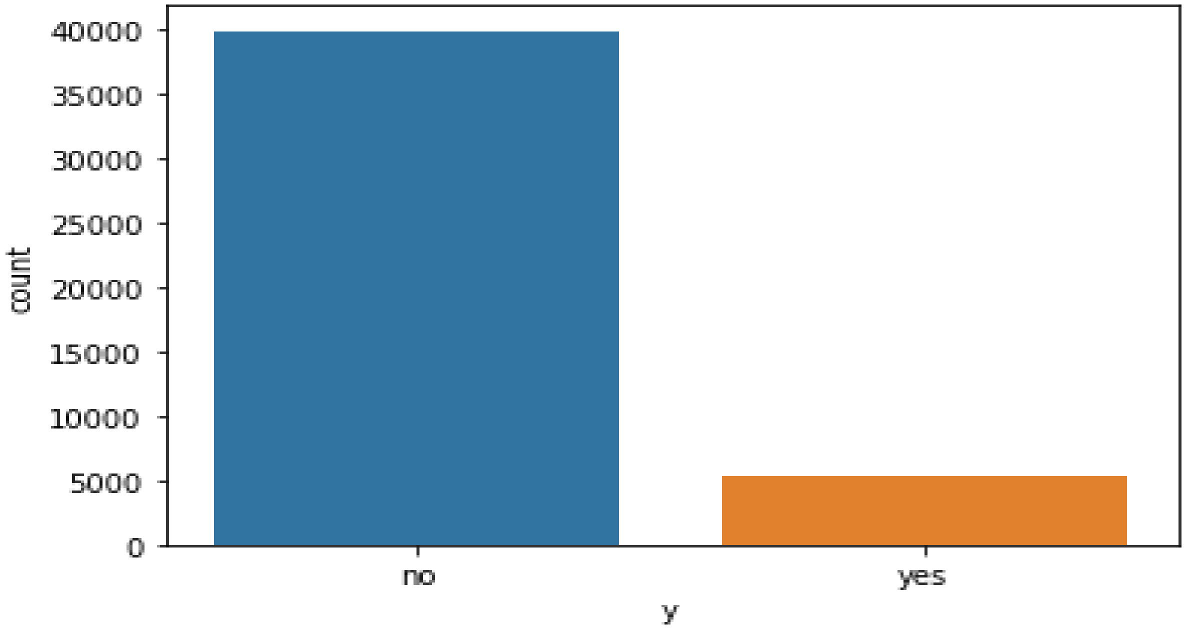
Plots for each were produced that looked at their relative frequency as well as normalized relative frequency. In Python, these graphs are created using the seaborn package. Many of these features contain unknown values so the next question is how to deal with this missing data. Simply discarding these rows would lead to a huge reduction in the amount of data and thus greatly interfere with the results. Instead, these missing values are imputed using other independent variables to infer the missing values. While this does not guarantee that all the missing data will be restored, a majority of it will be. For instance, cross-tabulation between ‘job’ and ‘education’ was used based on the hypothesis that a person’s job will be influenced by their education. Thus, a person’s job is used to predict their education level.

The Python function cross tab was created for this cross-tabulation step. A similar cross-tabulation process was carried out for the ‘house ownership’ and ‘loan status’ features. It’s important to note that in making these imputations, care was taken to ensure the correlations made sense in the real world. If not, the values were not replaced. Throughout this process, dataframes using the pandas package were invaluable. Python provides quickness, ease of modifiability and ease of replacement of values throughout the dataset thanks to this tool.

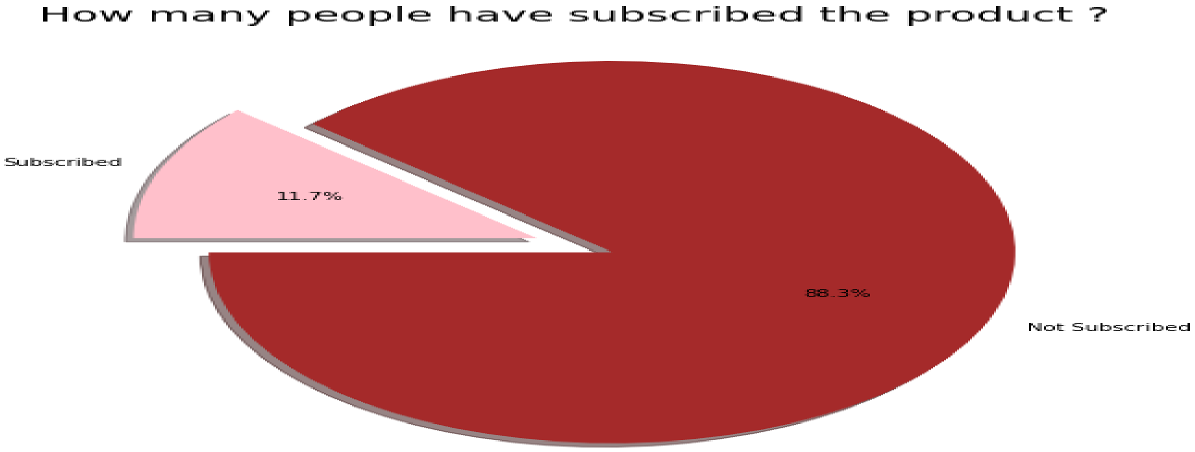
The next task is to deal with missing data among the numerical features. In this particular dataset. It’s quickly noted that while only the ‘pdays’ (number of days since that customer had been contacted from the previous campaign) column contained such values, they made up the majority of the data for this feature. In other words, this column was missing more data than it contained. Further exploration showed that this missingness was due to customers who had not been contacted previously at all.

To deal with this, the numerical feature ‘pdays’ was replaced with a categorical feature based on whether the customer had never been contacted, Finally, a heatmap was created to show us whether there is strong correlation between the target variable and any independent variables.

Target Variable:



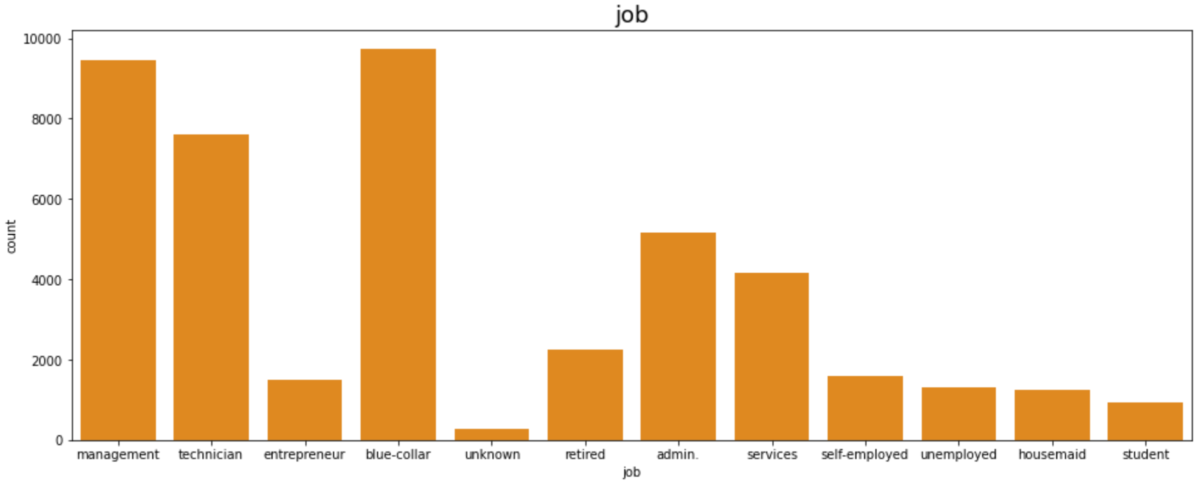
Only 11.7% people have subscribed to our product

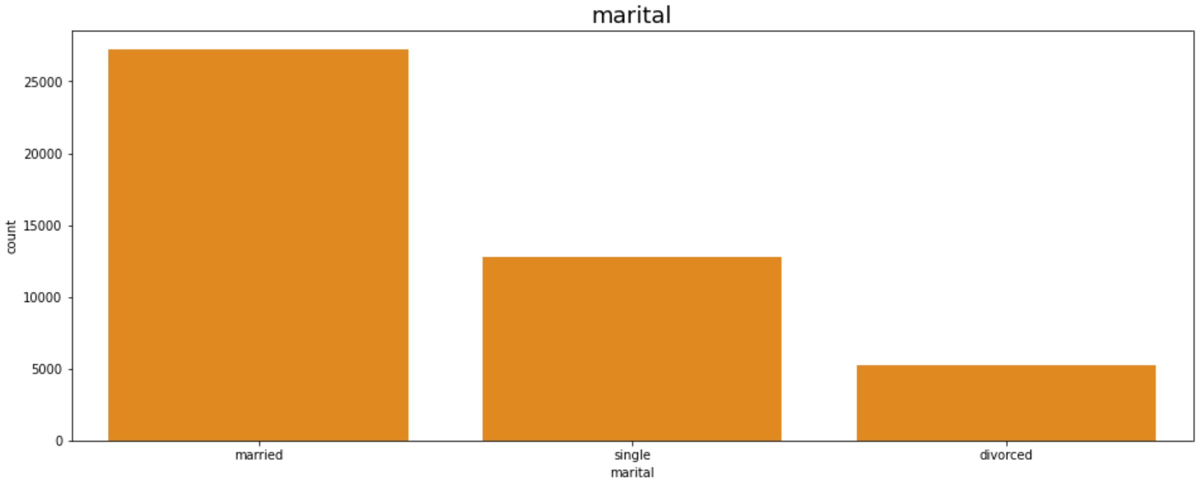


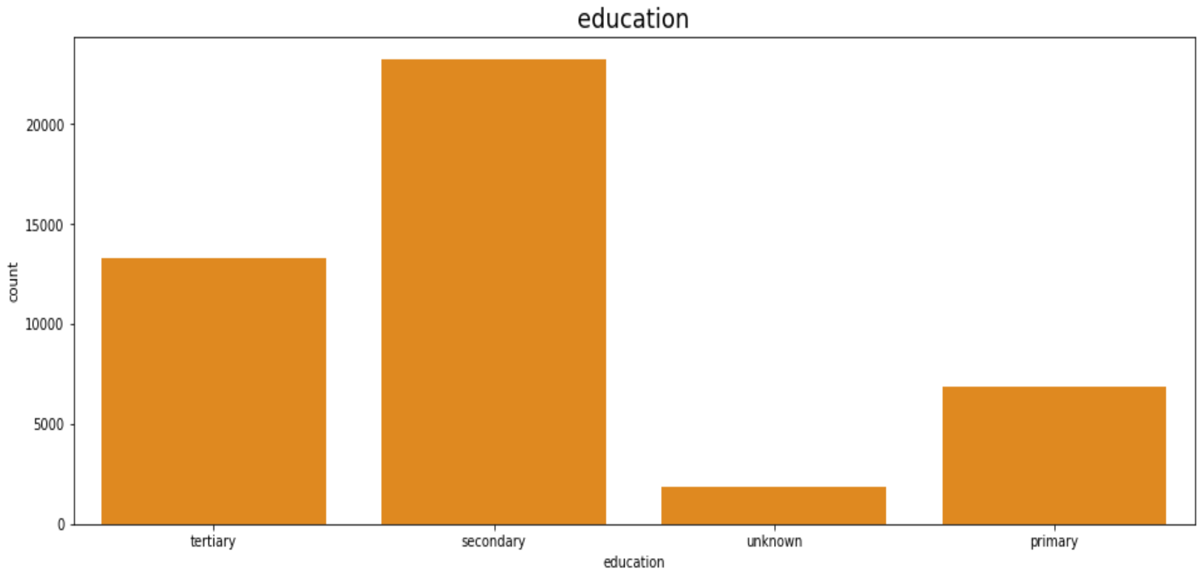
Only 11.7% people have subscribed to our product.

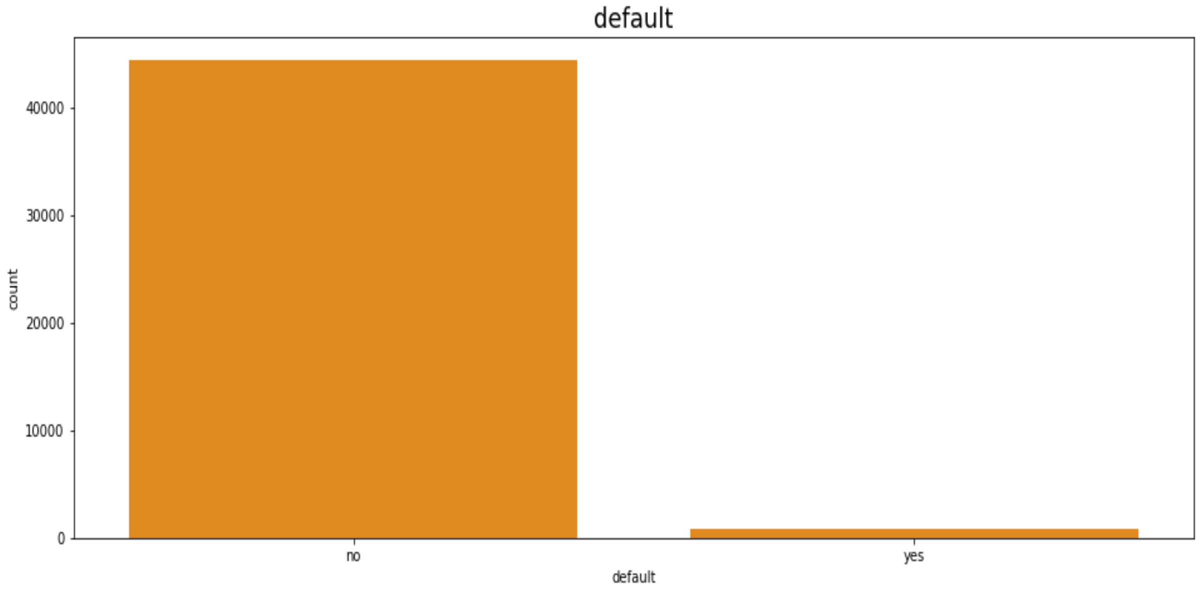
Bar Graph Representation of Each Variable

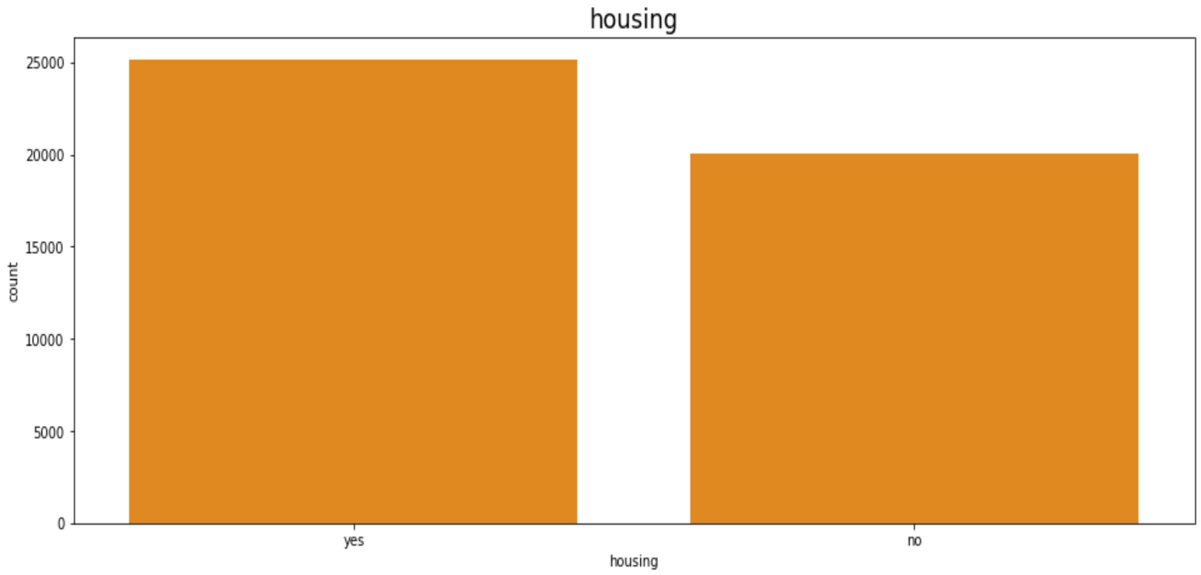
Categorical Features Exploration:

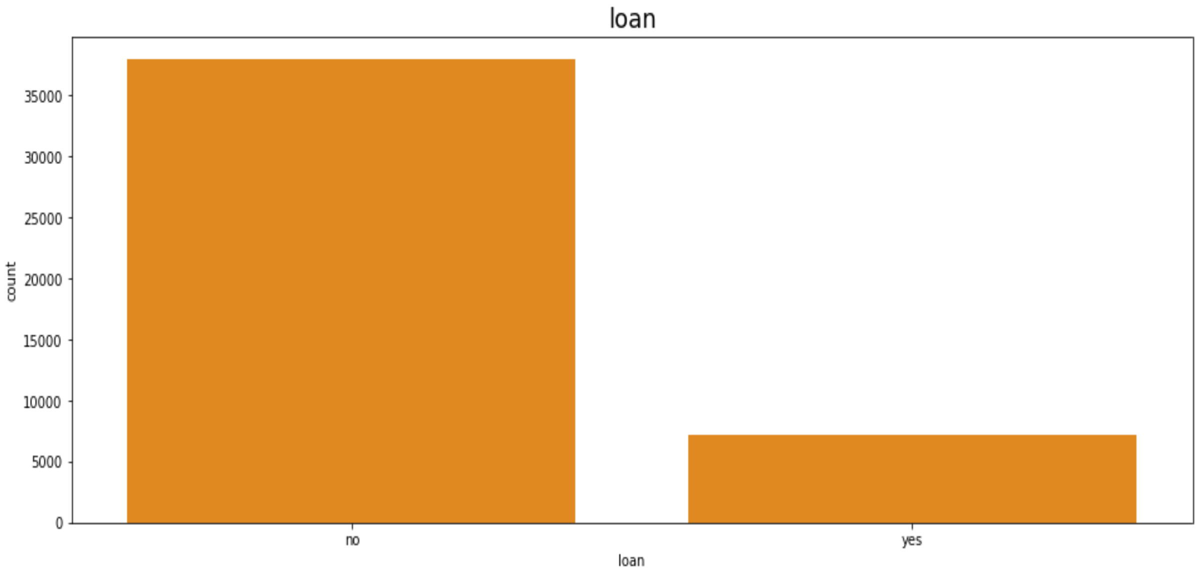


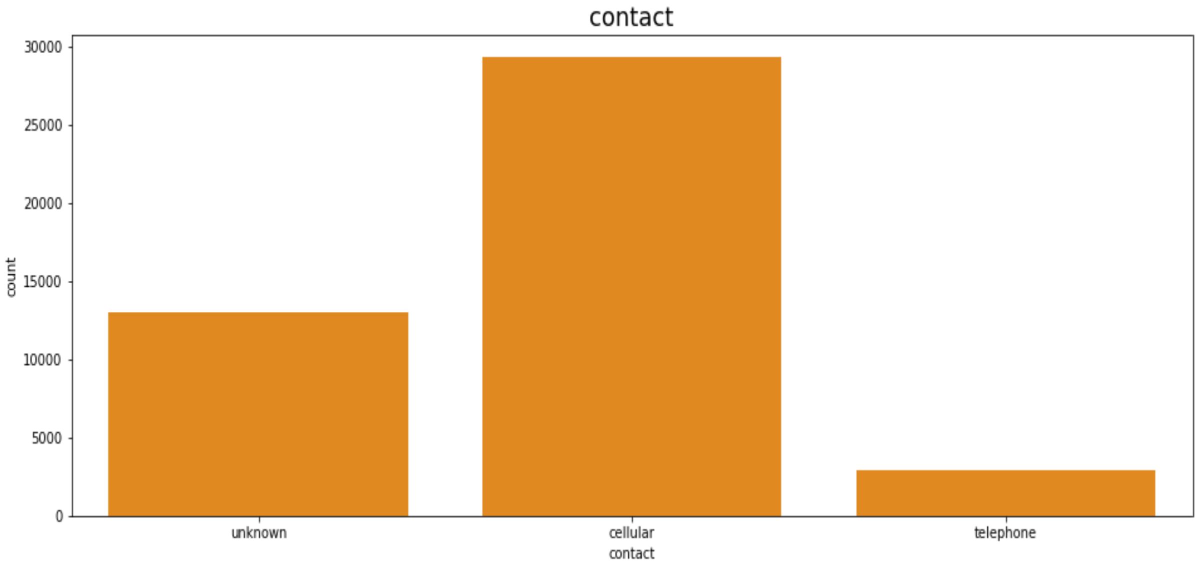


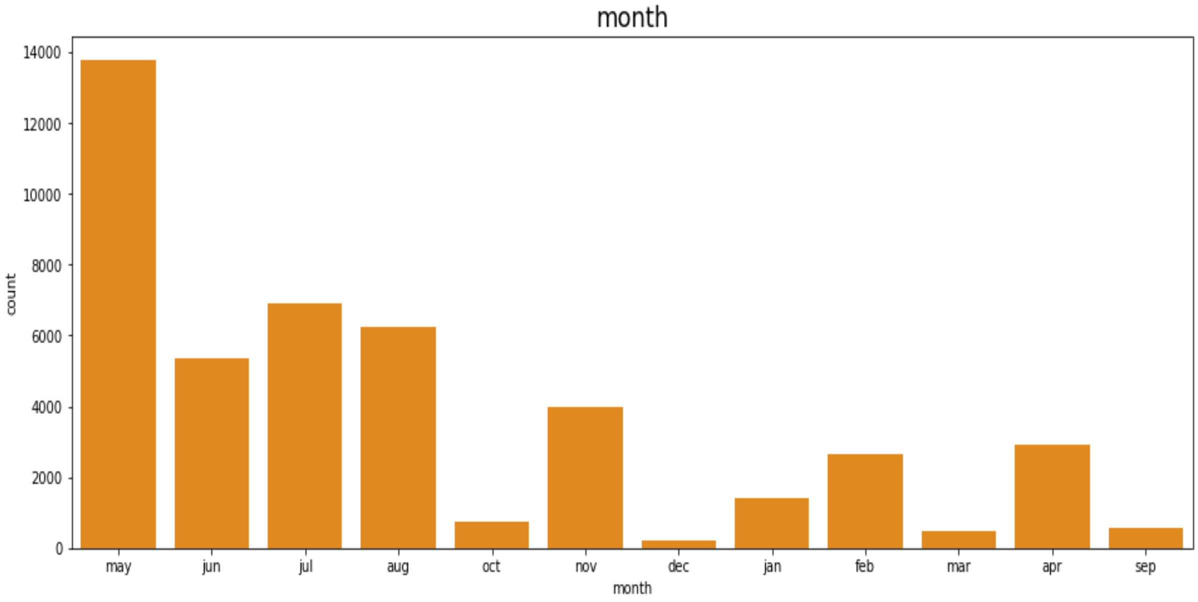

education
count
0
5000
10000
15000
20000
tertiary
secondary
unknown
primary
education

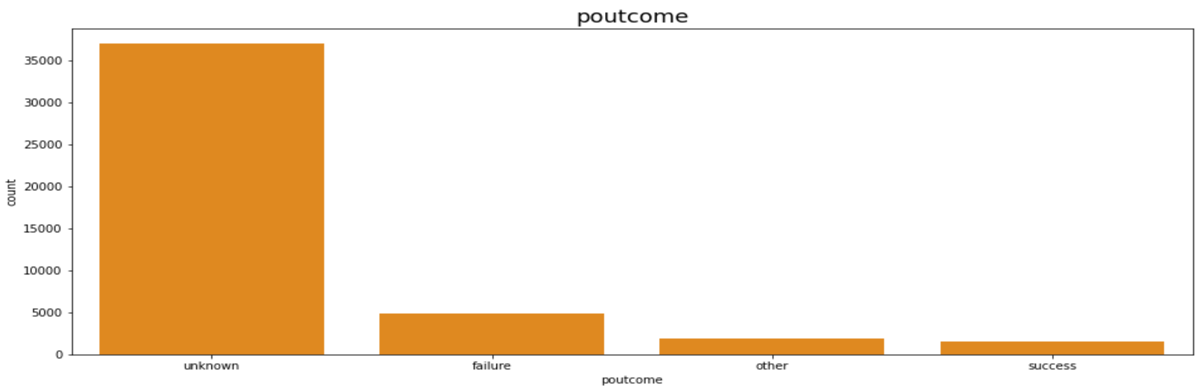

default
count
0
10000
20000
30000
40000
no
yes
default

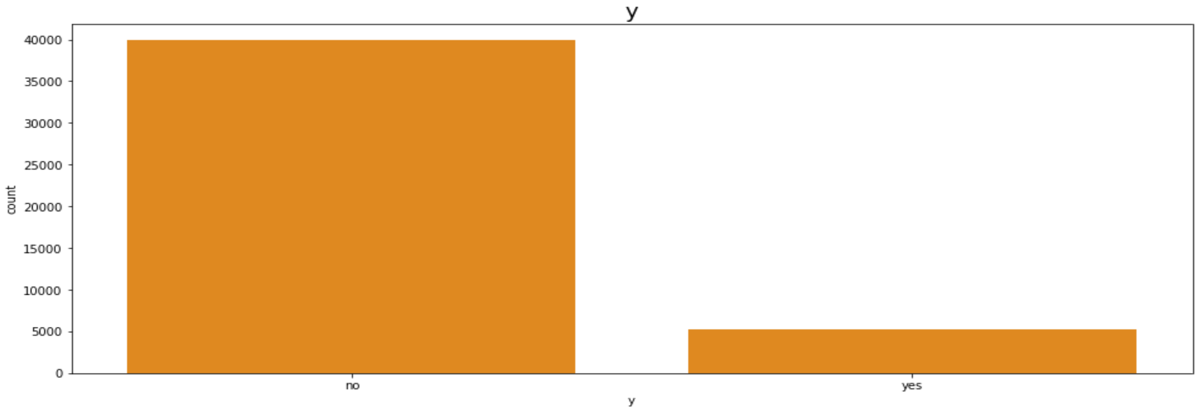

housing
count
0
5000
10000
15000
20000
25000
yes
no
housing


loan
count
0
5000
10000
15000
20000
25000
30000
35000
no
yes
loan

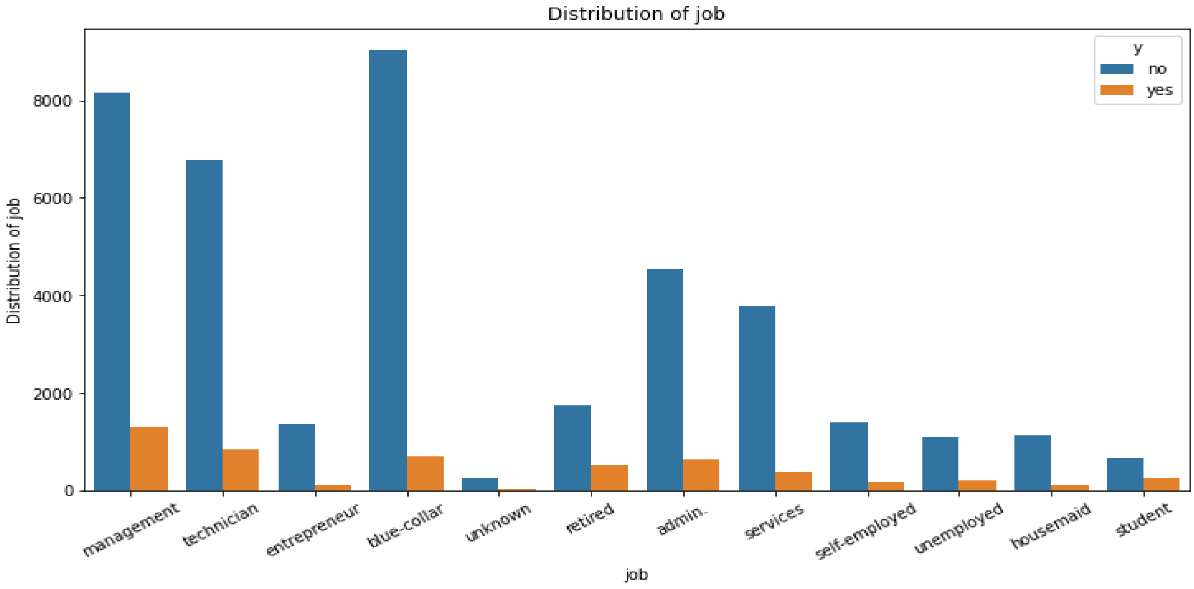

contact
count
0
5000
10000
15000
20000
25000
30000
unknown
cellular
telephone
contact

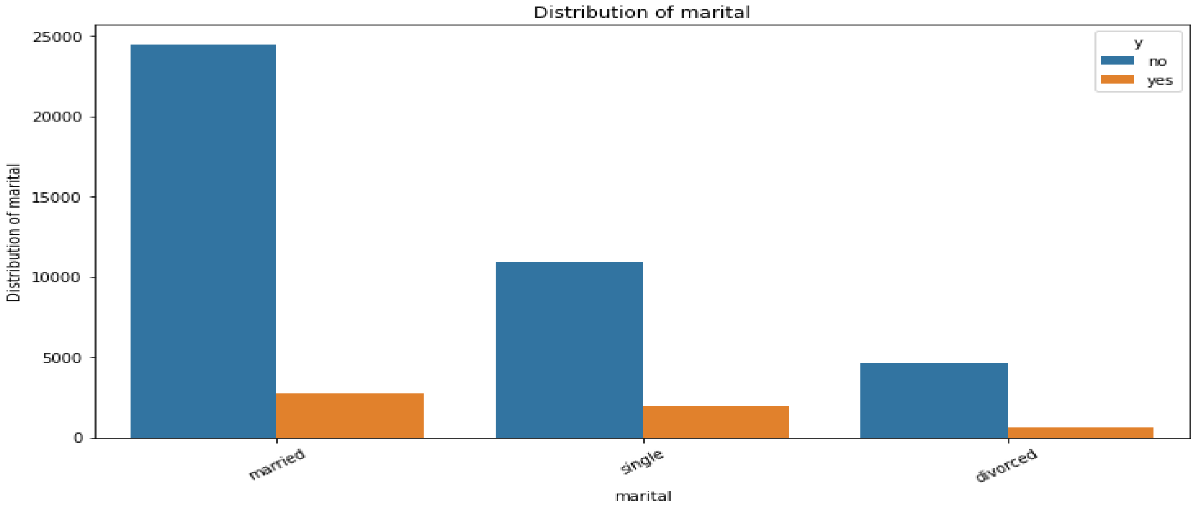

month
count
0
2000
4000
6000
8000
10000
12000
14000
may
jun
jul
aug
oct
nov
dec
jan
feb
mar
apr
sep
month

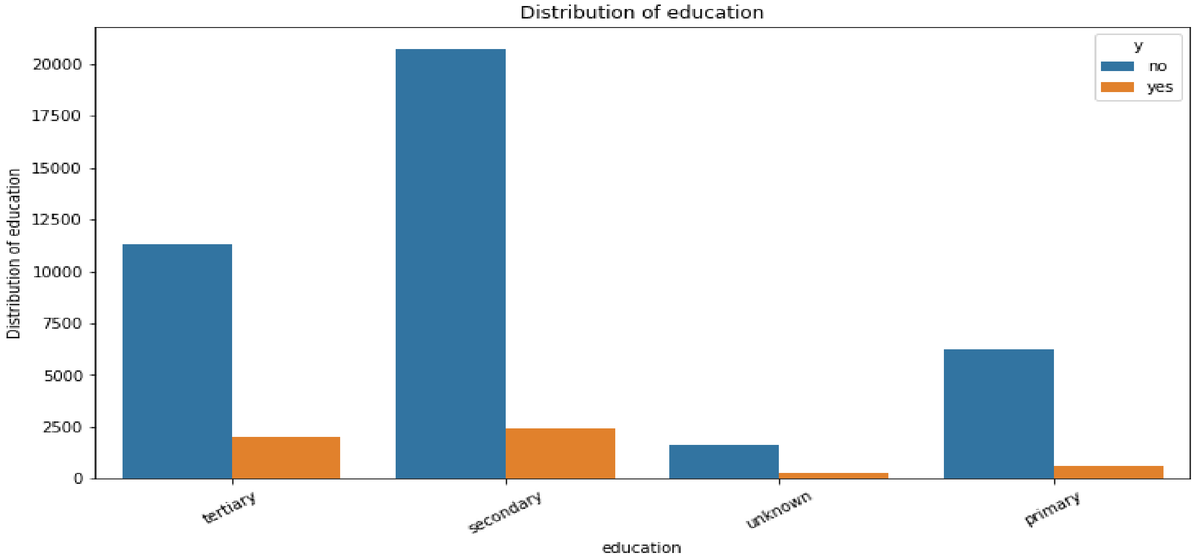


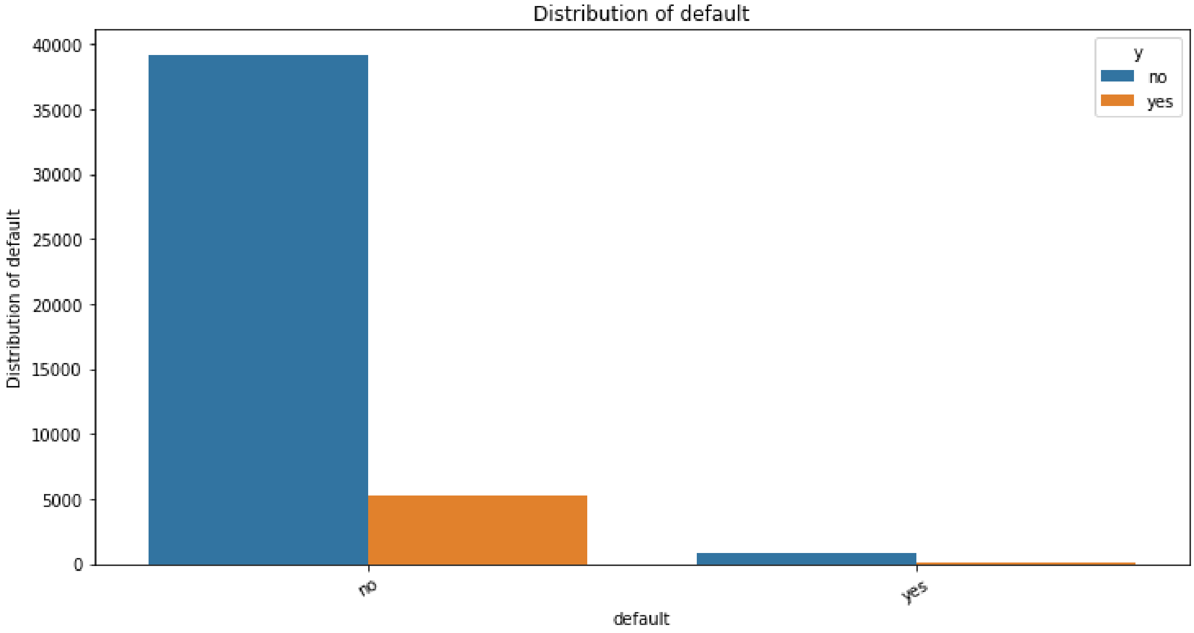


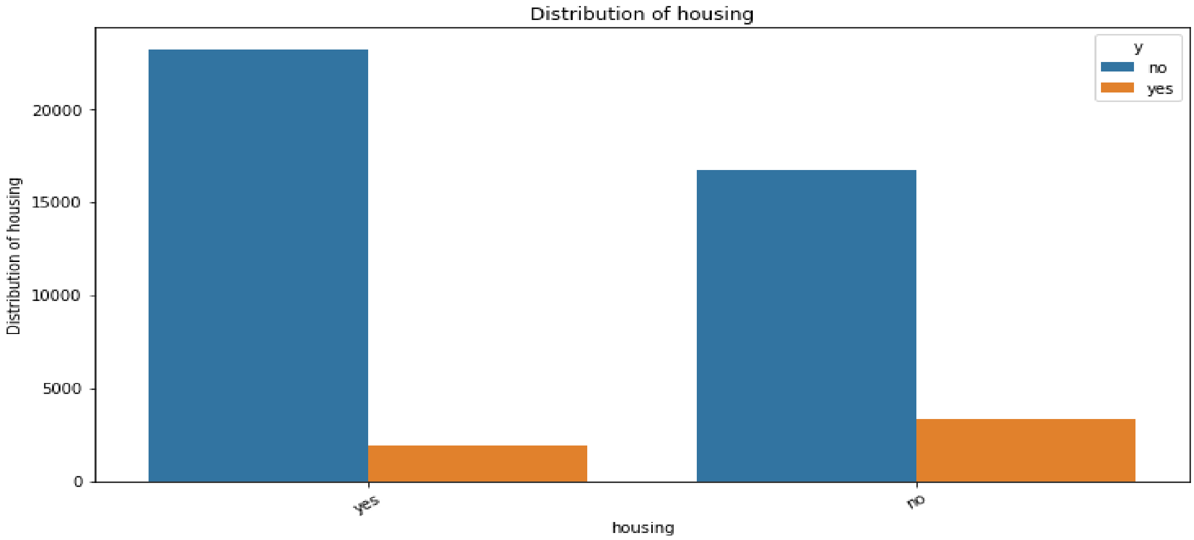
Countplot Distribution Of Categorial Variables

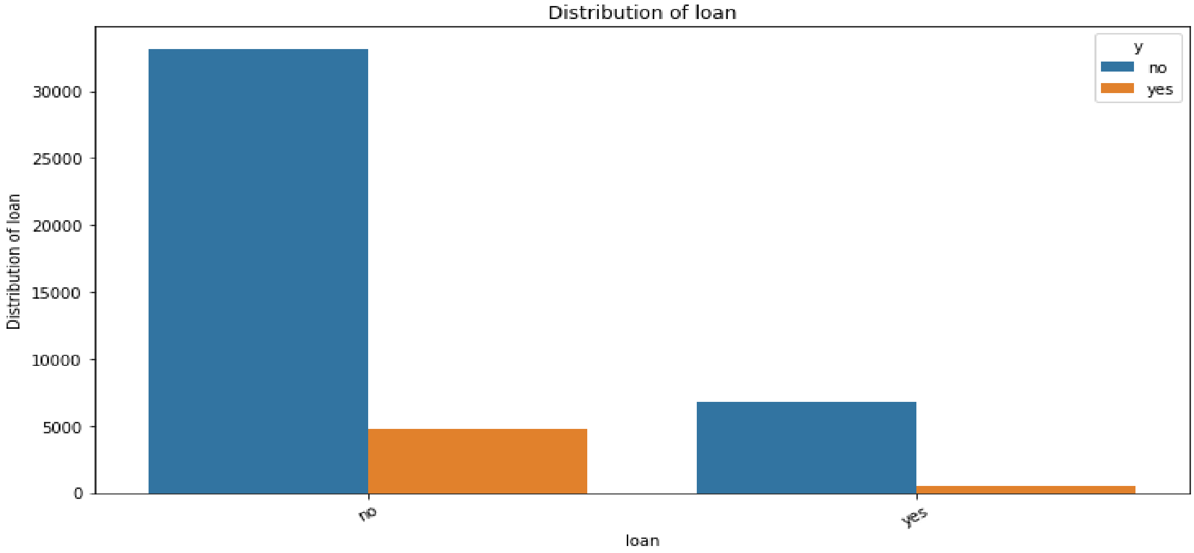


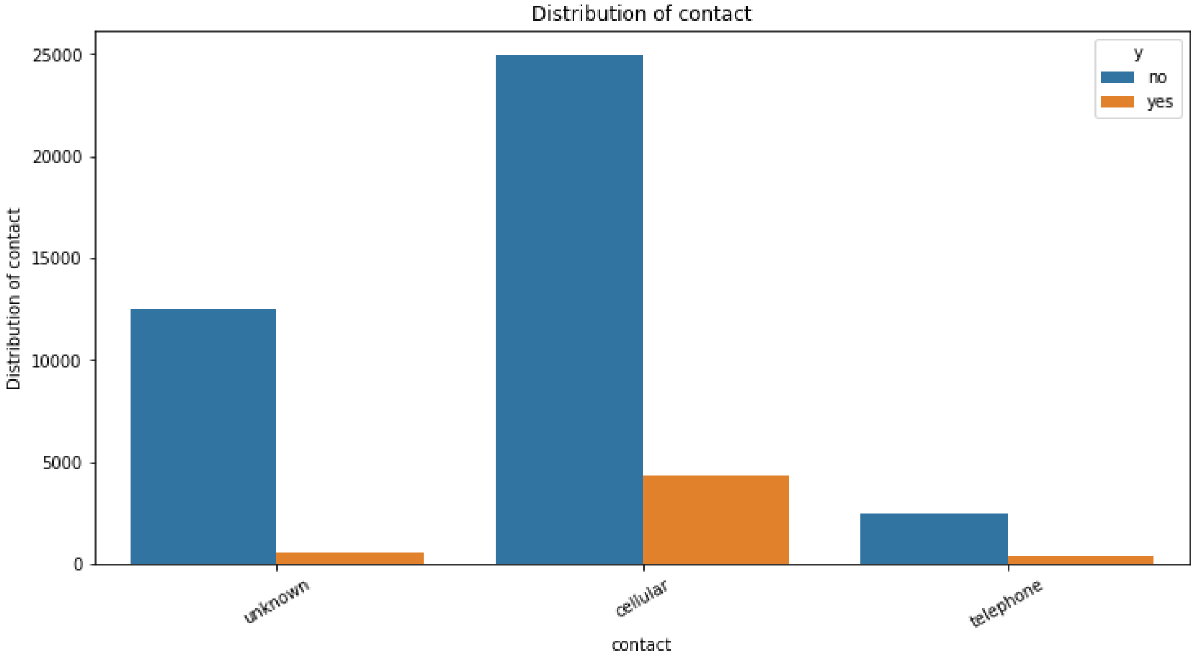

Distribution of marital
Distribution of marital
25000
20000
15000
10000
5000
0
married
single
divorced
marital
y
no
yes

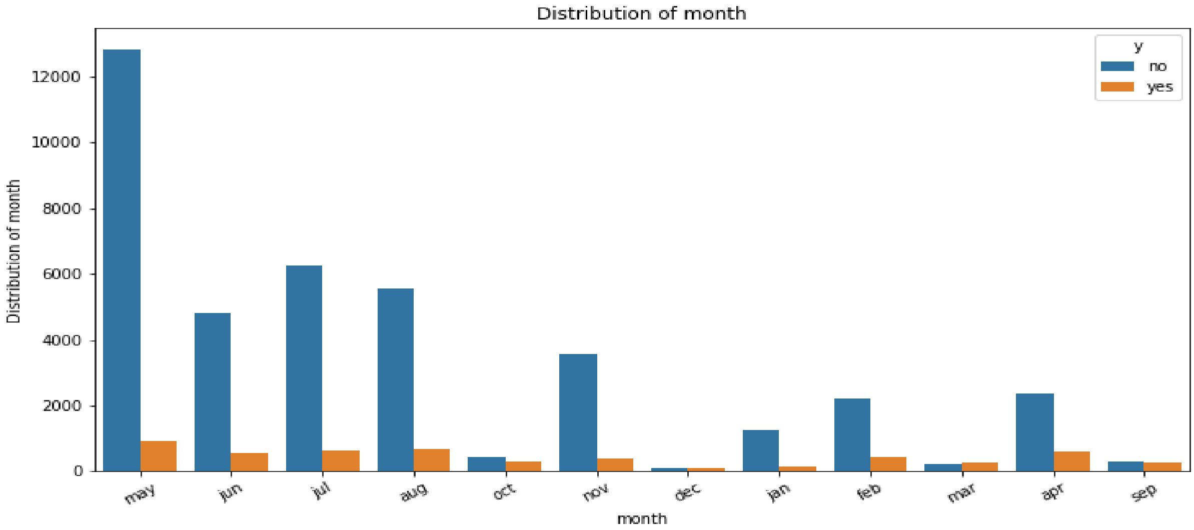

Distribution of education
Distribution of education
20000
17500
15000
12500
10000
7500
5000
2500
0
tertiary
secondary
unknown
primary
education
y
no
yes

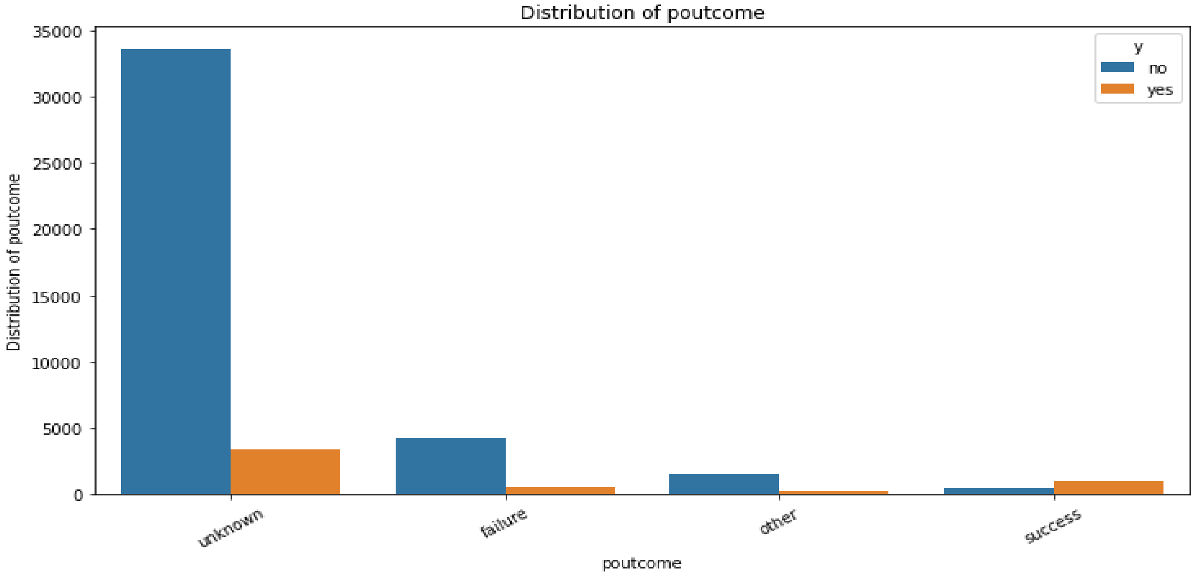

Distribution of default
Distribution of default
40000
35000
30000
25000
20000
15000
10000
5000
0
no
yes
default
y
no
yes

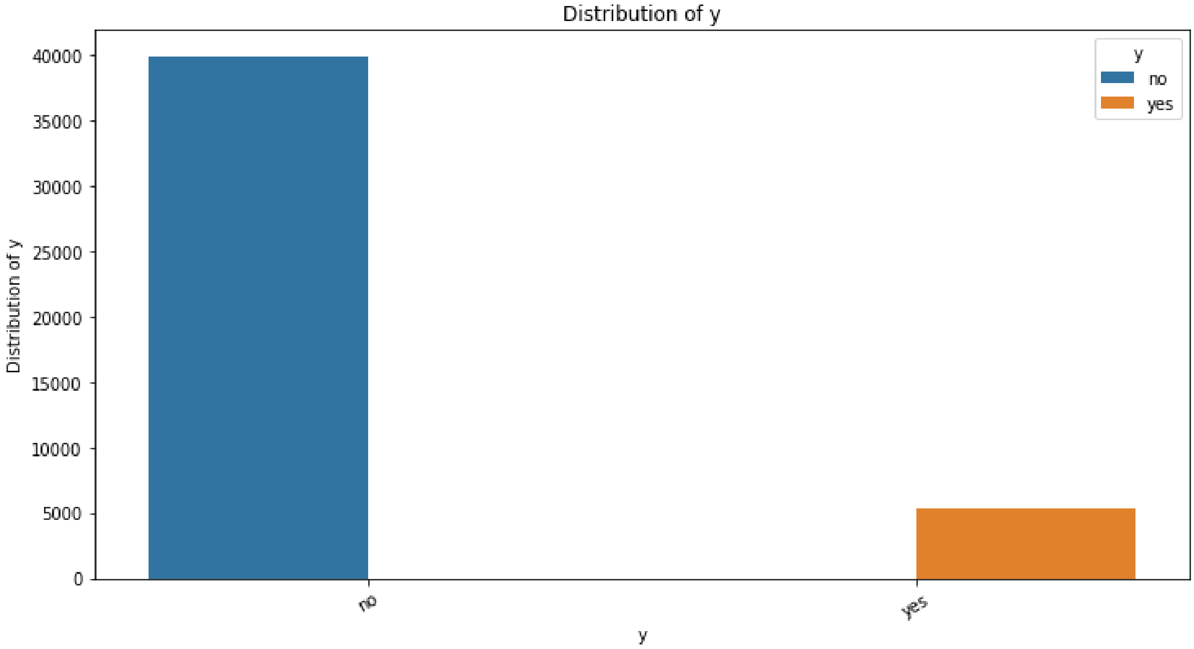

Distribution of housing
y
no
yes
20000
15000
10000
5000
0
Distribution of housing
yes
no
housing


Distribution of loan
y
no
yes
30000
25000
20000
15000
10000
5000
0
Distribution of loan
no
yes
loan


Distribution of contact
y
no
yes
25000
20000
15000
10000
5000
0
Distribution of contact
unknown
cellular
telephone
contact

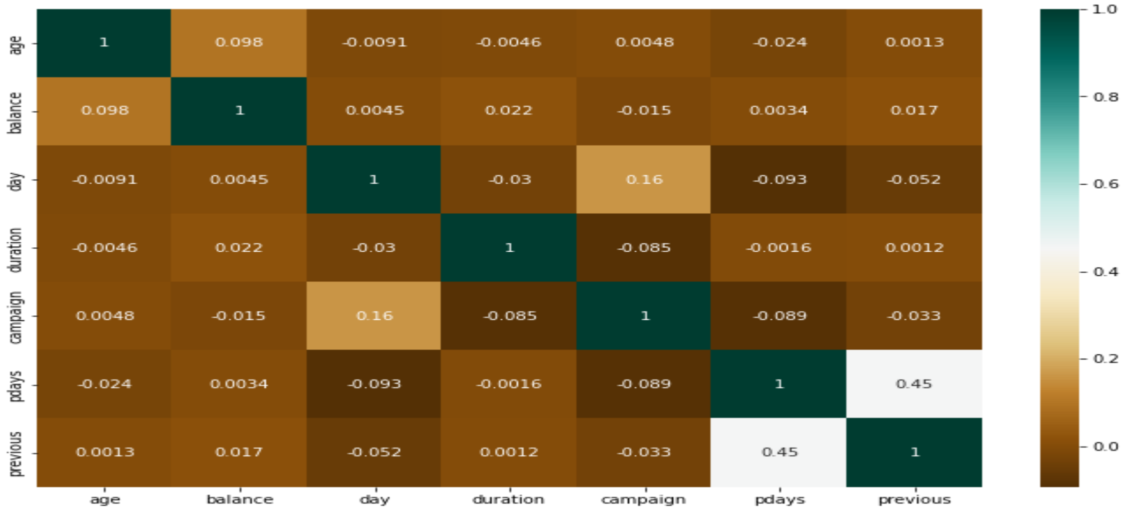

Distribution of month
Distribution of month
month
12000
10000
8000
6000
4000
2000
0
may
jun
jul
aug
oct
nov
dec
jan
feb
mar
apr
sep
y
no
yes


Distribution of poutcome
Distribution of poutcome
poutcome
35000
30000
25000
20000
15000
10000
5000
0
unknown
failure
other
success
y
no
yes


Distribution of y
Distribution of y
y
40000
35000
30000
25000
20000
15000
10000
5000
0
no
yes
y
no
yes

The heatmap is created using Spearman correlation, which measures the degree to which the rankings of each variable (as opposed to the actual values) align, thus minimizing the effect of outliers. Once this is measured, those variables are expected to be significant during the modeling stage.

This graphic was created using Python's seaborn package and the specially written function drawheatmap, which takes a dataframe as an input. The code for this function can be seen in the Google Collab Notebook for this project.



For performing predictive analysis, many well known machine learning models should be fit on training data to learn parameters of the model and then they can be run on test set to get the prediction. Models used in our project are discussed below.

C. Model Building:

The dataset is divided into training data and test data with the intention of using the training data to find the parameters of the particular model being used (fitting the model on the training data) and then applying this to the test data to determine the model's performance and to draw conclusions about its predictive capability.

This can be done with a sklearn.cross validation.train test split function call by specifying split ratio.

Logistic Regression:

Python provides the package sklearn,linear model,LogisticRegression for Logistic Regression. LR a is well known classification model.

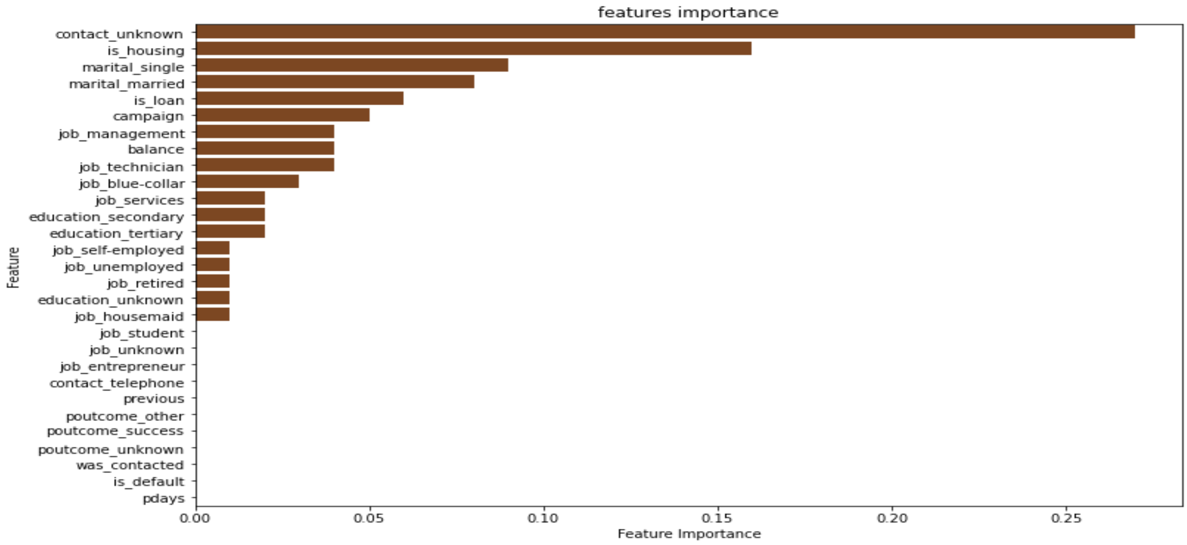
The linear model fits the training data to the equation y = w0 + w1x1 + w2x2 + . . . (where y stands for the target variable, w0 stands for the y intercept, x1, x2, x3, . . . are feature vectors, and w1, w2, w3, . . . are their corresponding weights) while the logistic regression algorithm uses the same decision boundary with bit modifications as shown: P(X) = 1 1+e−y . Logistic regression is used because classification is not exactly a linear function and using linear regression produces an output within [−∞, +∞] while the probability has to be within [0, 1]. The logistic function itself does output the probability of an instance belonging to the positive class. This output probability does indeed have a range of [0, 1], hence overcoming the drawbacks of classification using a linear model.

Decision Trees:

Python provides the package sklearn tree Decision Tree Classifier for the decision tree classifier. Decision trees are a simple yet effective method for classification. Using a tree structure, this algorithm splits the data set based on one feature at every node until all the data in the leaf belongs to the same class.

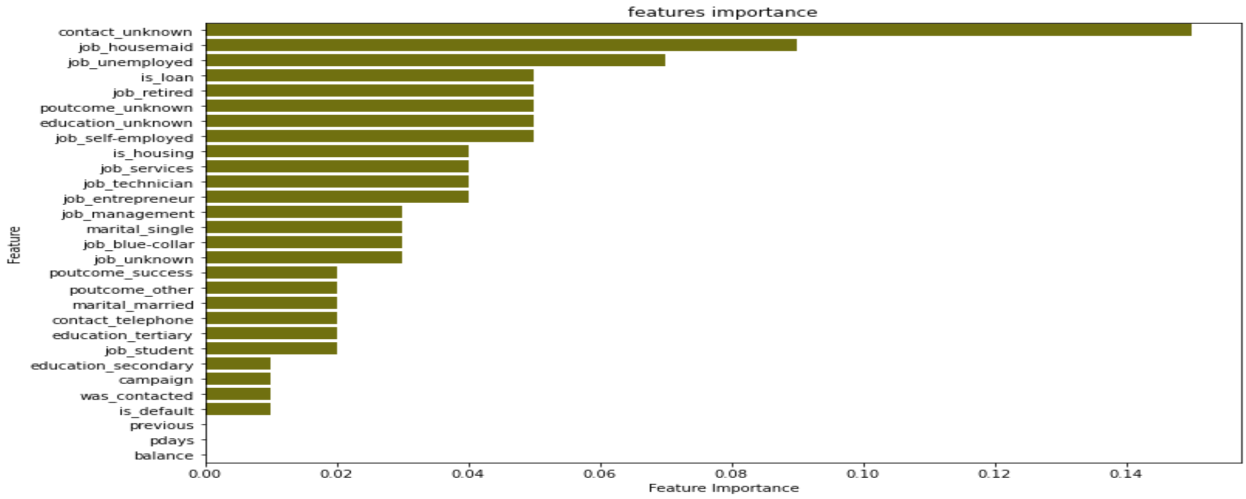
The criterion used for splitting is called information gain, which is based on a purity measure called entropy, a measure of disorder. The set with the highest impurity will have higher entropy whereas the set which has higher purity will have lower entropy. Information gain measures the change in entropy due to the amount of information added.

The higher the information gain, the more information that feature provides about the target variable. By default, the decision tree grows deep and complex until every leaf is pure and hence it is prone to overfitting.



XG BOOST Classifier:

XGBoost stands for "Extreme Gradient Boosting". XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements Machine Learning algorithms under the Gradient Boosting framework. It provides a parallel tree boosting to solve many data science problems in a fast and accurate way.

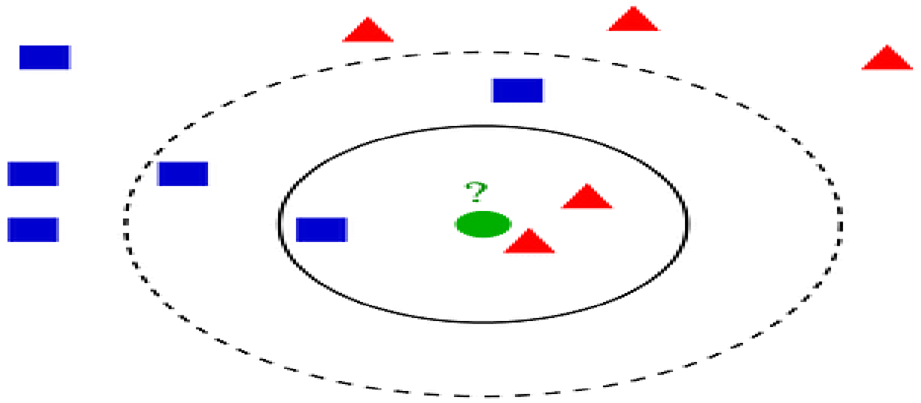


K-Nearest Neighbors (KNN):

K-Nearest Neighbors is a machine learning technique and algorithm that can be used for both regression and classification tasks. K-Nearest Neighbors examines the labels of a chosen number of data points surrounding a target data point, in order to make a prediction about the class that the data point falls into. K-Nearest Neighbors (KNN) is a conceptually simple yet very powerful algorithm, and for those reasons, it's one of the most popular machine learning algorithms. Let’s take a deep dive into the KNN algorithm and see exactly how it works. Having a good understanding of how KNN operates will let you appreciated the best and worst use cases for KNN.

KNN is a supervised learning algorithm, meaning that the examples in the dataset must have labels assigned to them/their classes must be known. There are two other important things to know about KNN. First, KNN is a non-parametric algorithm. This means that no assumptions about the dataset are made when the model is used. Rather, the model is constructed entirely from the provided data. Second, there is no splitting of the dataset into training and test sets when using KNN. KNN makes no generalizations between a training and testing set, so all the training data is also used when the model is asked to make predictions.

Overview of K-Nearest Neighbors (KNN):



Conclusion:

From this project, we learned how banks can improve their marketing campaigns by focusing their efforts on certain prime-grade clients and also how they can recognize market conditions which are favorable to increase client subscription for the fixed-term products they are offering. All of this was possible by implementing data science and machine learning methods in Python. Tools such as dataframes, arrays, for loops, etc. were all critical for the success of this project.

A large number of other tools and techniques from the Python for Data Science course were used and these were invaluable for making our analyses and predictions. This project demonstrated how powerful Python can be for data science applications.

Following points shows the conclusion on the basis of classifier.

- For age , most of the customers are in the age range of 30-40.
- For balance , above 1000$ is like to subscribe a term deposit..
- The model can help to classify the customers on the basis on which they deposit or not
- The model helps to target the right customer rather than wasting time on wrong customer
- Comparing to all algorithms XGboost algorithm has best accuracy score and ROC-AUC score . So it is concluded as optimal model.