
# BANK MARKETING CAMPAIGN
# VIRTUAL INTERNSHIP
# DATE : 06/10/2021

# Team member’s details as well as individual ones

- **Group Name:** The Greeks
- **Name:** Giorgos Moysiadis
- **Email:** giorgosmoysiadis@gmail.com
- **Country:** Greece
- **College/Company:** Data Glacier
- **Specialization:** Data Science
- **GitHub Repo Link:** https://github.com/gmoysiad/bank_marketing_campaign/tree/main/Week%2011%20Deliverables
- **Problem Description:** ABC Bank wants to sell its term deposit product to customers. Before launching the product, they want to develop a model which will help them understand whether a particular customer plans to buy their product or not (based on customer's past interaction with the bank or other Financial Institution).

# EDA presentation for business users

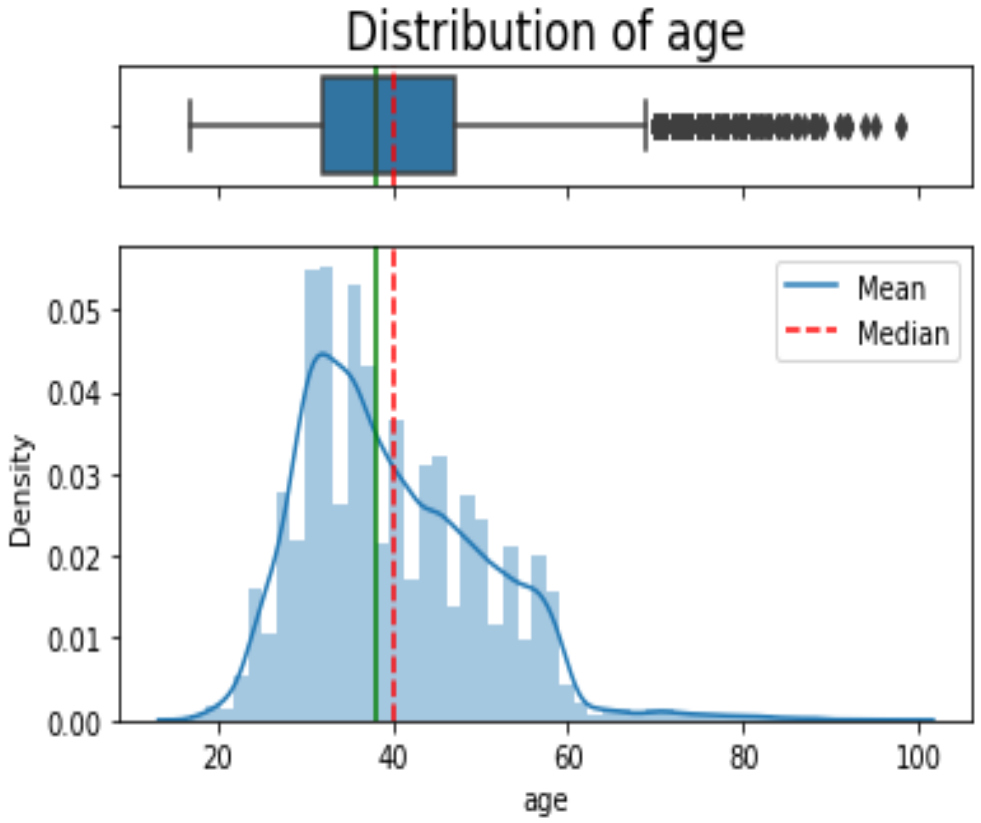
- To understand the steps involved in EDA, we will use Python as the programming language and Jupyter Notebooks because it's open-source, and not only it's an excellent IDE but also very good for visualization and presentation. First, we will import all the python libraries that are required for this, which include **NumPy** for numerical calculations and scientific computing, **Pandas** for handling data, **Matplotlib** and **Seaborn** for visualization. Furthermore, **Scikit-learn** for machine learning and **XGBoost,** which regularizes gradient boosting framework. Then, we will load the data into the **Pandas** data frame. For this analysis, we will use a dataset named "df", which has the following columns:
- Age, job, marital, education, default, housing, loan, contact, month, day of week, duration, campaign, pdays, previous, poutcome, emp.var.state, cons.price.idx, cons.conf.idx, euribor3m, nr.employed as well as the output variable y (whether the client subscribed a term deposit).

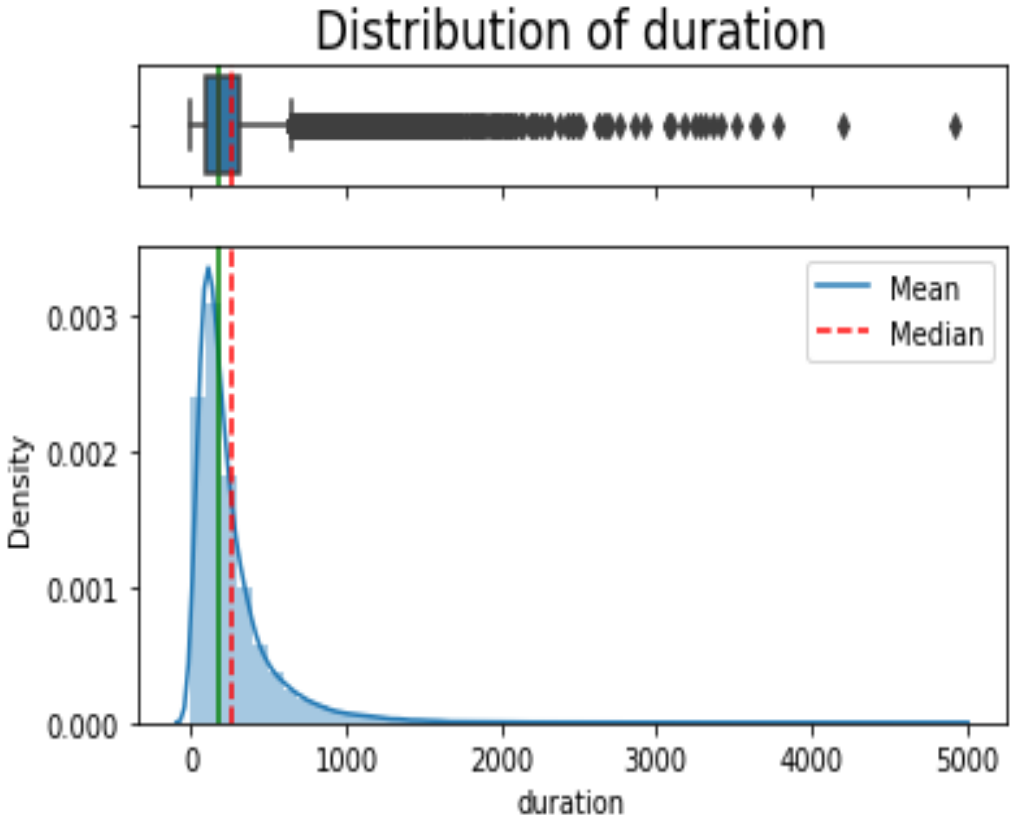
# EDA presentation for business users

- Those columns describe the extent to which, these variables contribute to a client subscribing a term deposit (or not). First and foremost, we handle the missing values. These missing values will be treated with by filling the missing values with a random label from the existing ones. On the whole, we avoid removing missing observations, (unless it is necessary) because it could result in a model with bias and loss of information. Alternatively, we can develop a model able to predict these missing values. Next, we visualize the distributions of the numerical variables to better understand the volume of the data that has outliers so that we may or may not, later remove from our dataset in the modeling phase. Afterwards, we create a function that will create both a basic distribution plot, and a boxplot, which is the proper way of displaying useful information about our data. That addresses their distribution, minimum/maximum, median values. Plus, it can also inform us whether our data is symmetrical, how tightly our data is grouped, as well as their skewness. By using describe() method, we get to know basic statistical characteristics of each numerical feature (int64 and float64 types): number of non-missing values, mean, standard deviation, range, median, 0.25, 0.50, 0.75 quartiles.

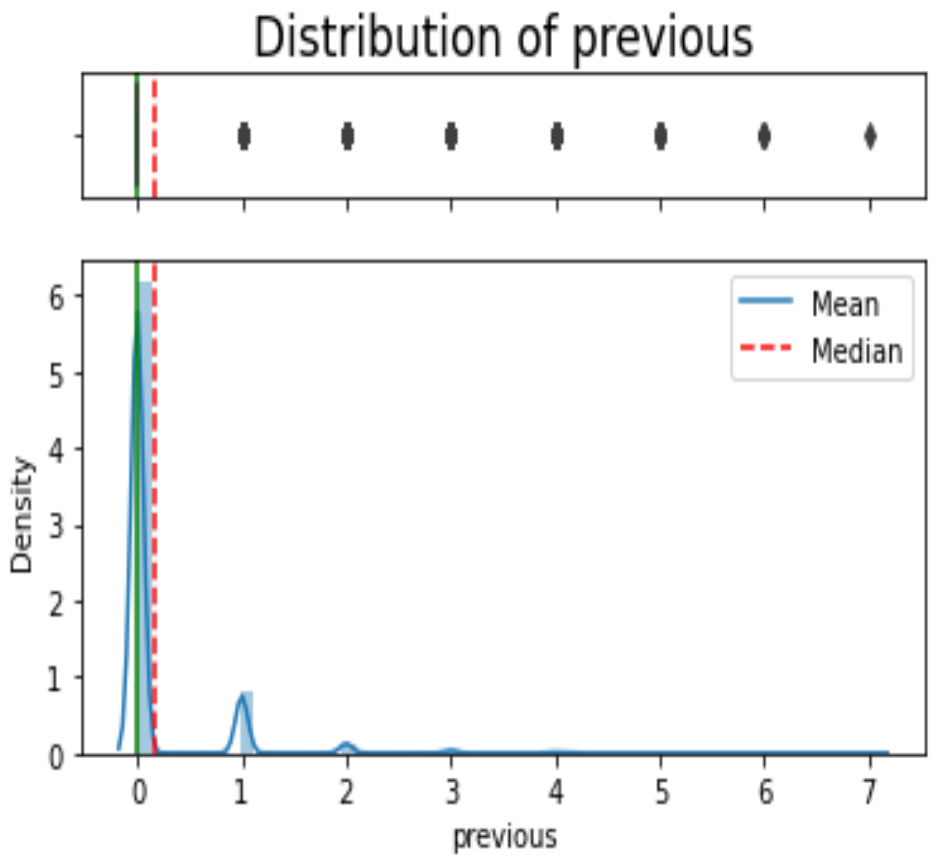
# Distributions and boxplots of age and duration variables

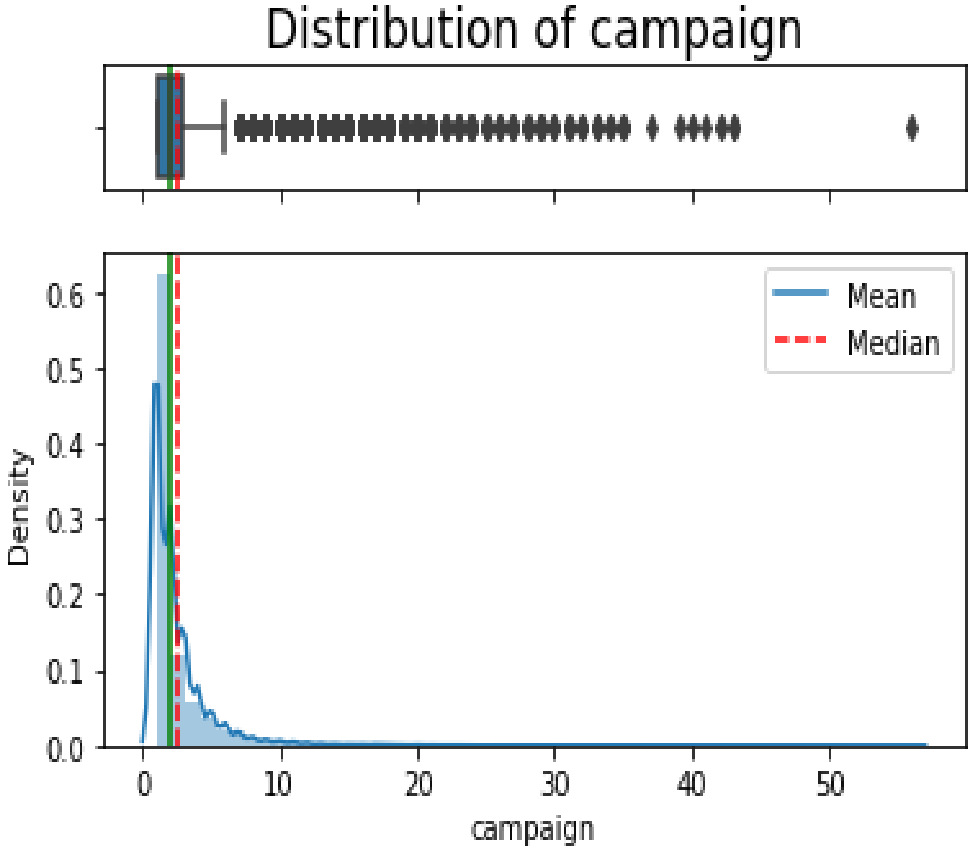
- The plots of the distributions of the numerical variables are as follows:



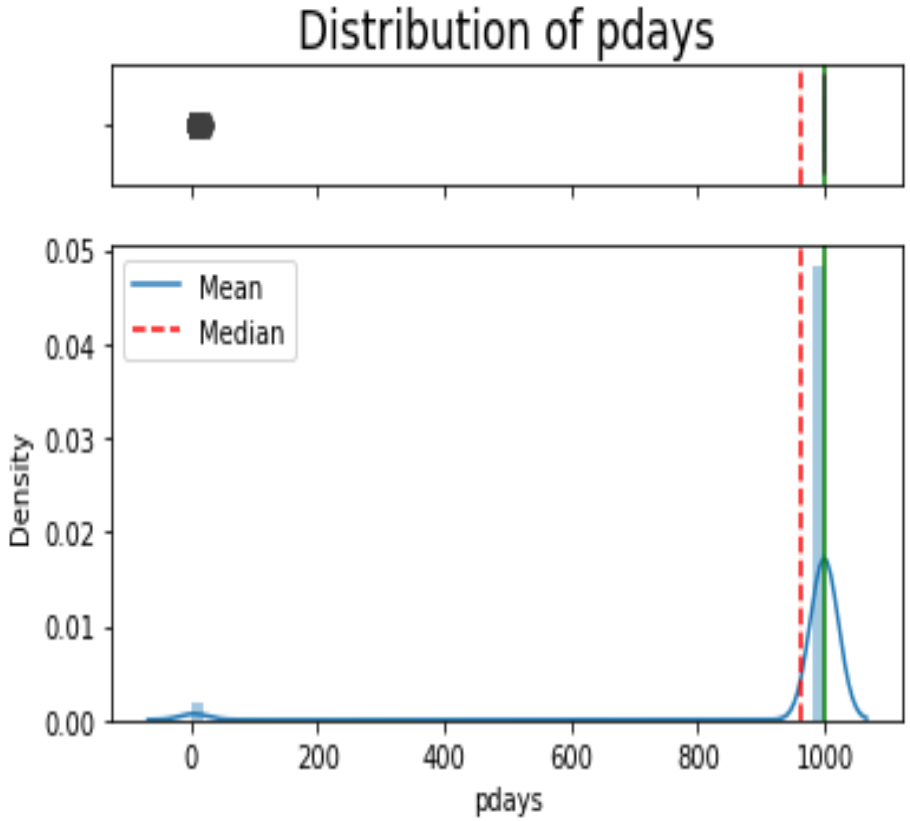



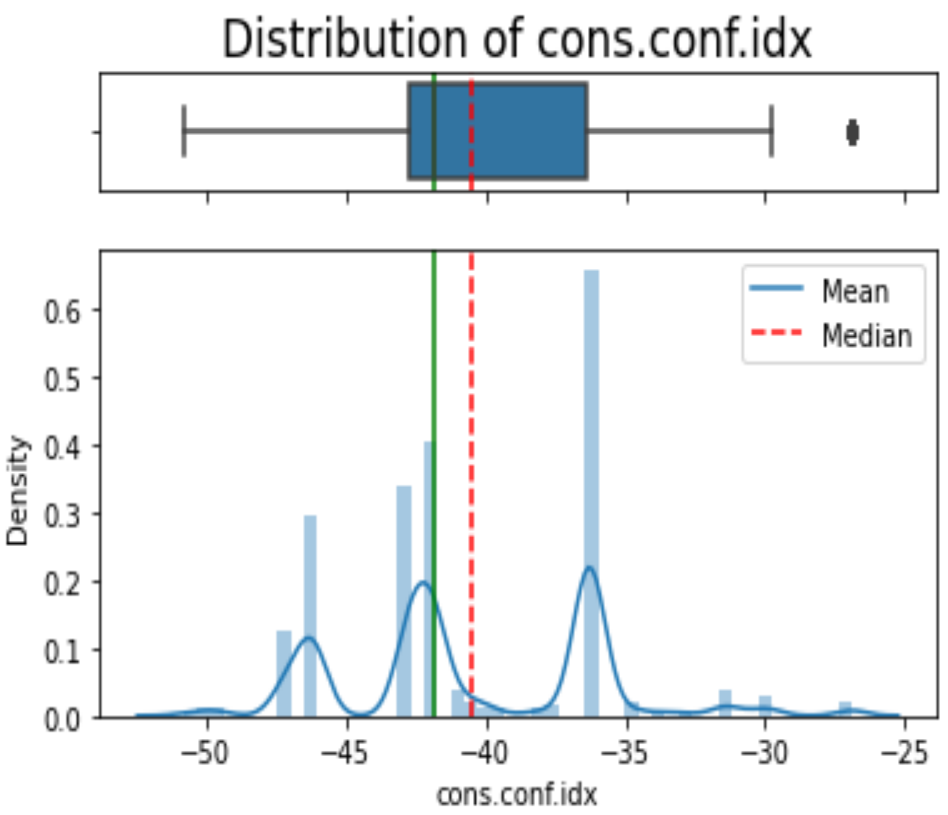
# Distributions and boxplots of previous and campaign variables



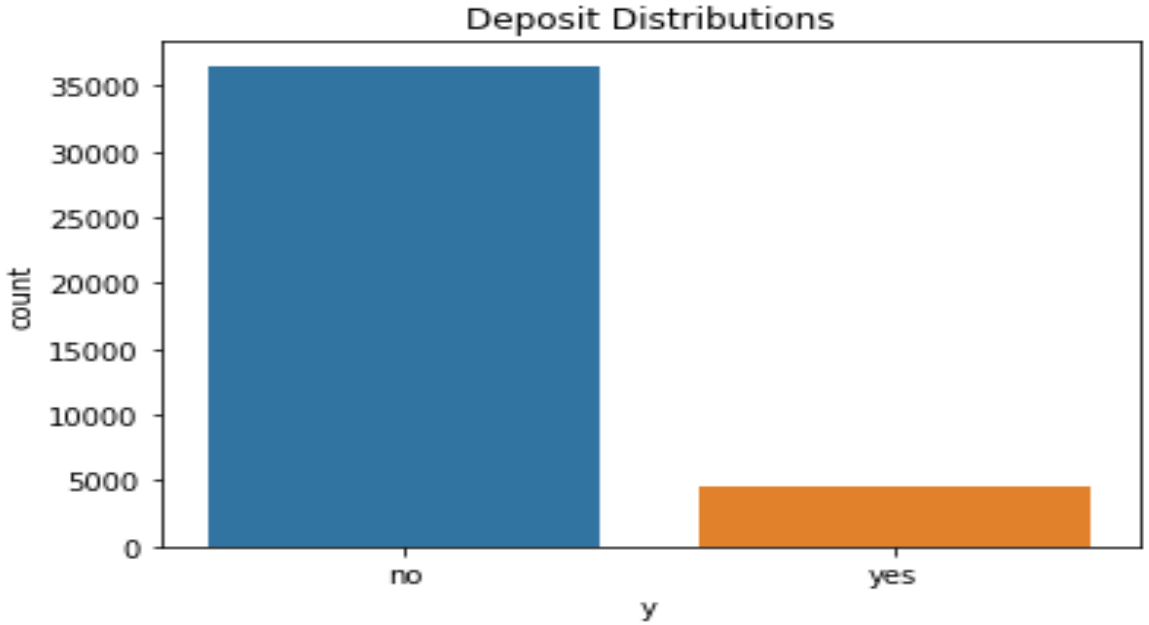
# Distributions and boxplots of pdays and cons.conf.idx variables



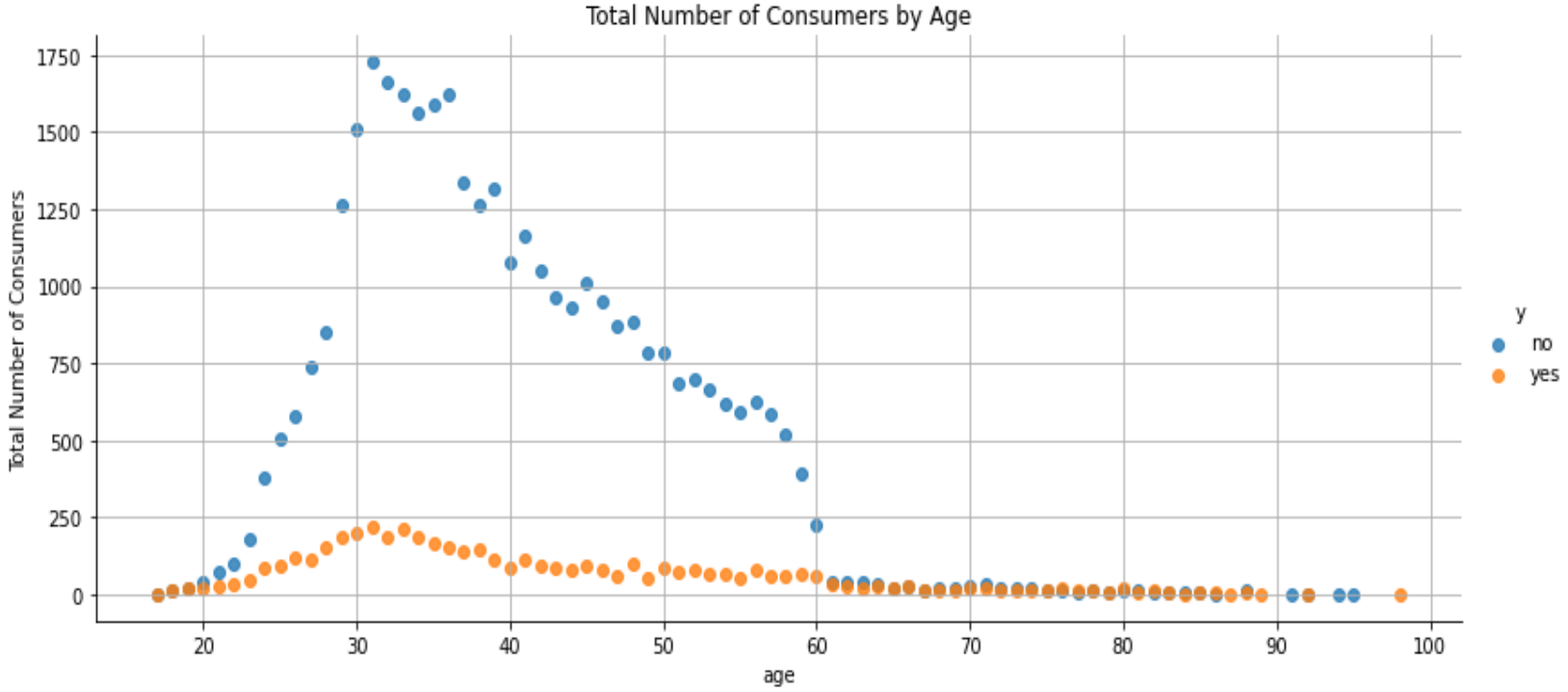
# Barplot of output variable y

- Moreover, we create a plot to check the symmetry of our output variable, where we can observe that the vast majority of the consumers did not subscribe a term deposit.




Finally, we proceed with the following assumptions to examine the relationship between the output variable and some of the rest of the features. For that reason, each of the assumptions provide a graph (scatterplot for the numerical variables and barplot for the categorical ones) to aid in clarifying their relationship:

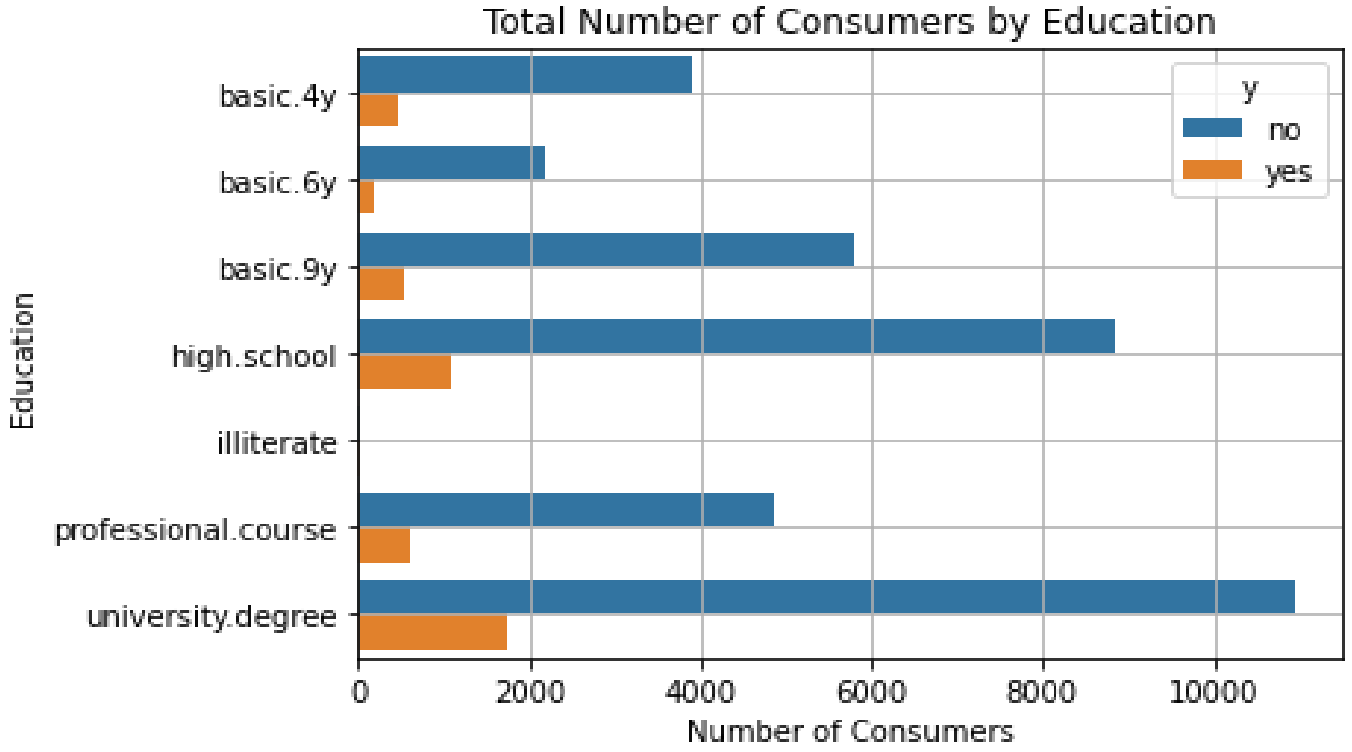
# Scatterplot of total number of consumers by age



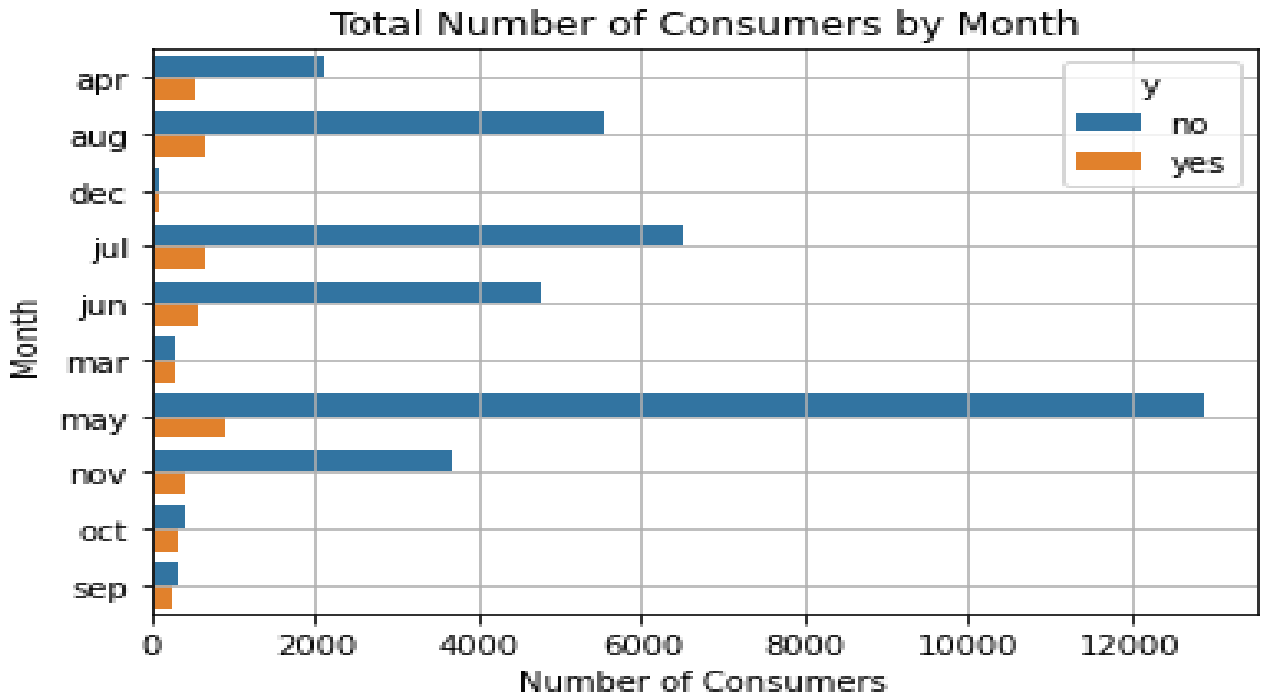

- In the graph above we can observe that from the ages 20 to 60 there is a significant difference between the ones who submitted and the others who did not. Note that at the age of 30, we observe the pick for both categories. Outliers are detected in the box plot of age.

# Barplot of total number of consumers by education

- In the graph below it's visible that those with university degree have the highest number of consumers for both of those who subscribed and for those who did not. The proportion of each category of education does not seem to have a significant difference.



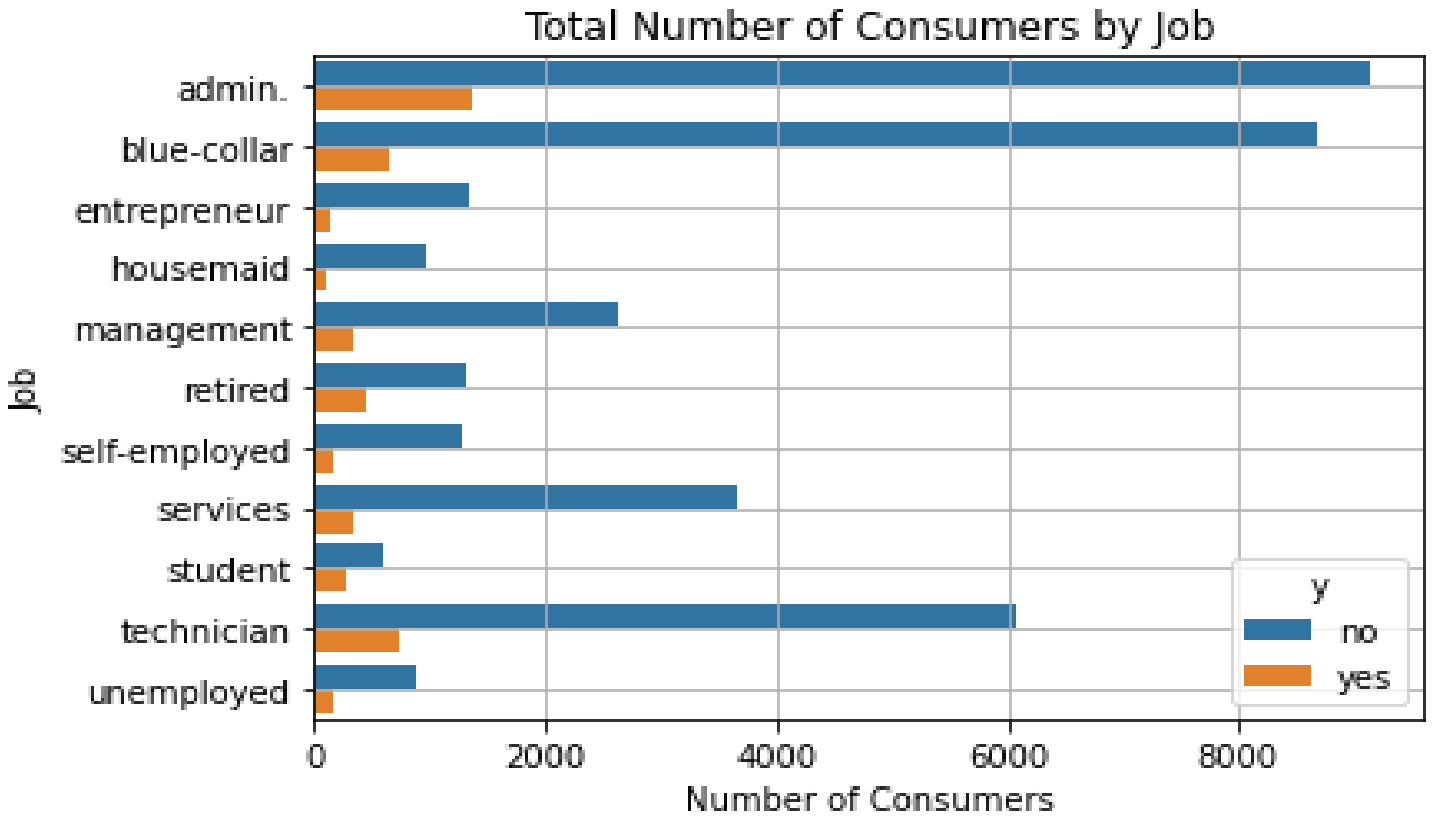
# Barplot of the total number of consumers by month



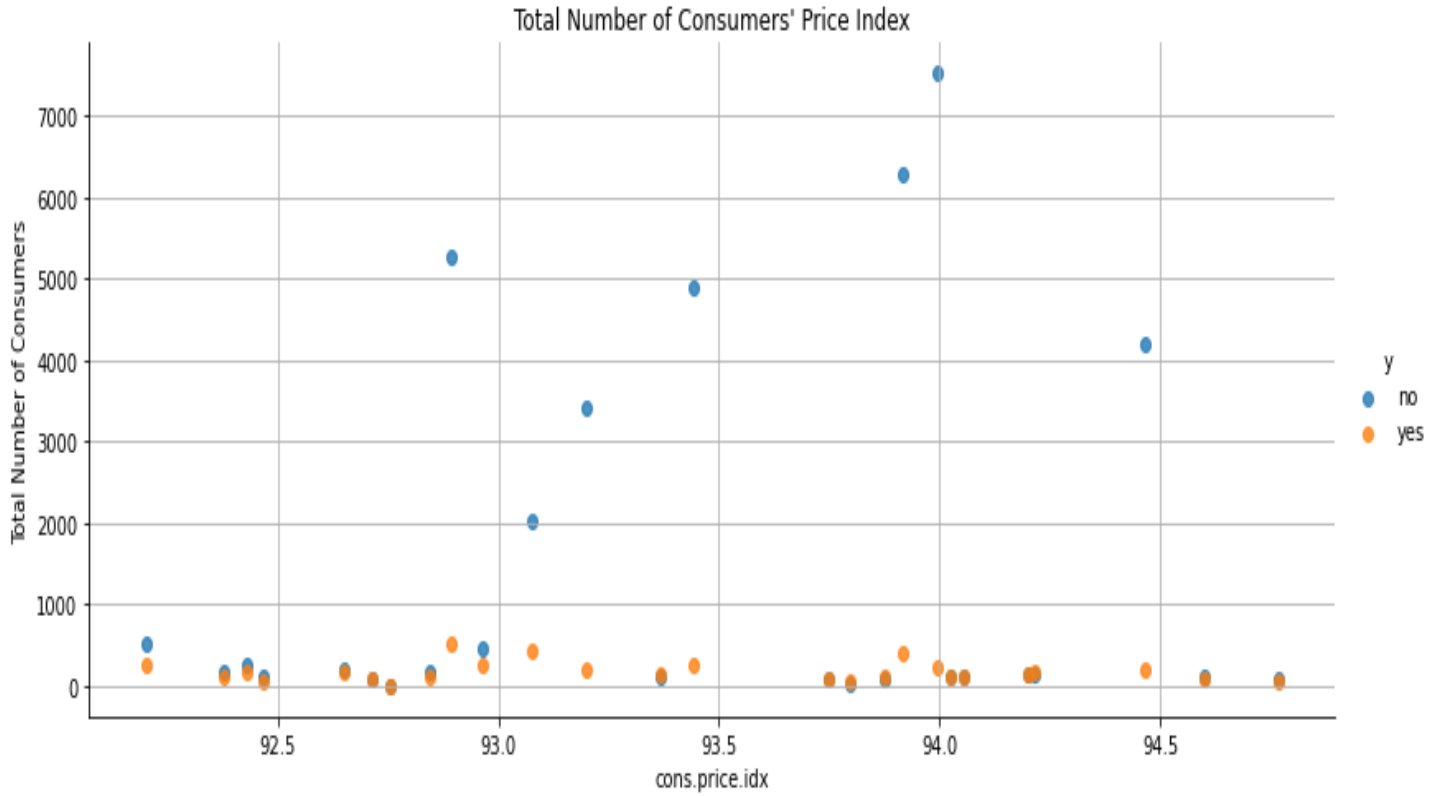

- In this graph we observe that in May we have by far the most last contracts in both cases. This result means that we have seasonality. However, in months such as December, March, October and November there are small differences between these two cases. Thus, these months may result in a consumer who might subscribe in a term deposit when we deploy our Machine Learning model.

# Barplot of the total number of consumers by job

- In the following graph we have admin and blue-collar with the highest number of consumers. However, in all categories of job we have the same proportion for both of those who subscribed and for those who did not.

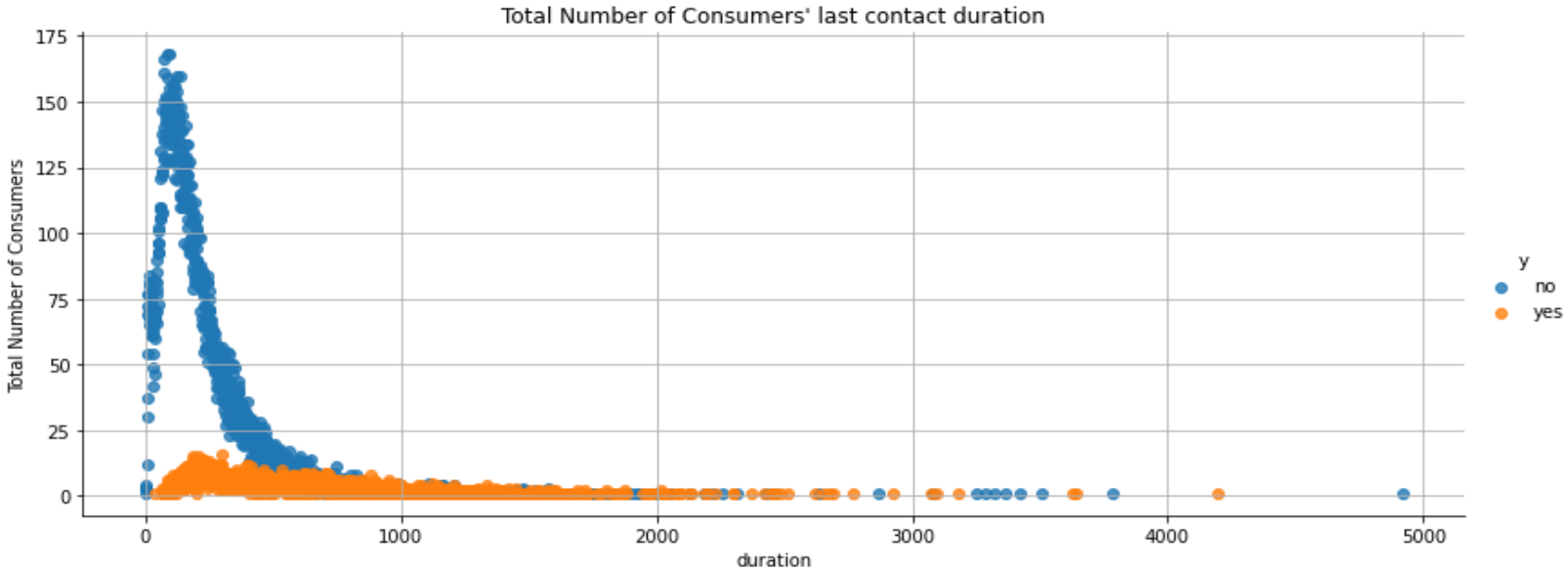


# Scatterplot of the total number of consumers by price index



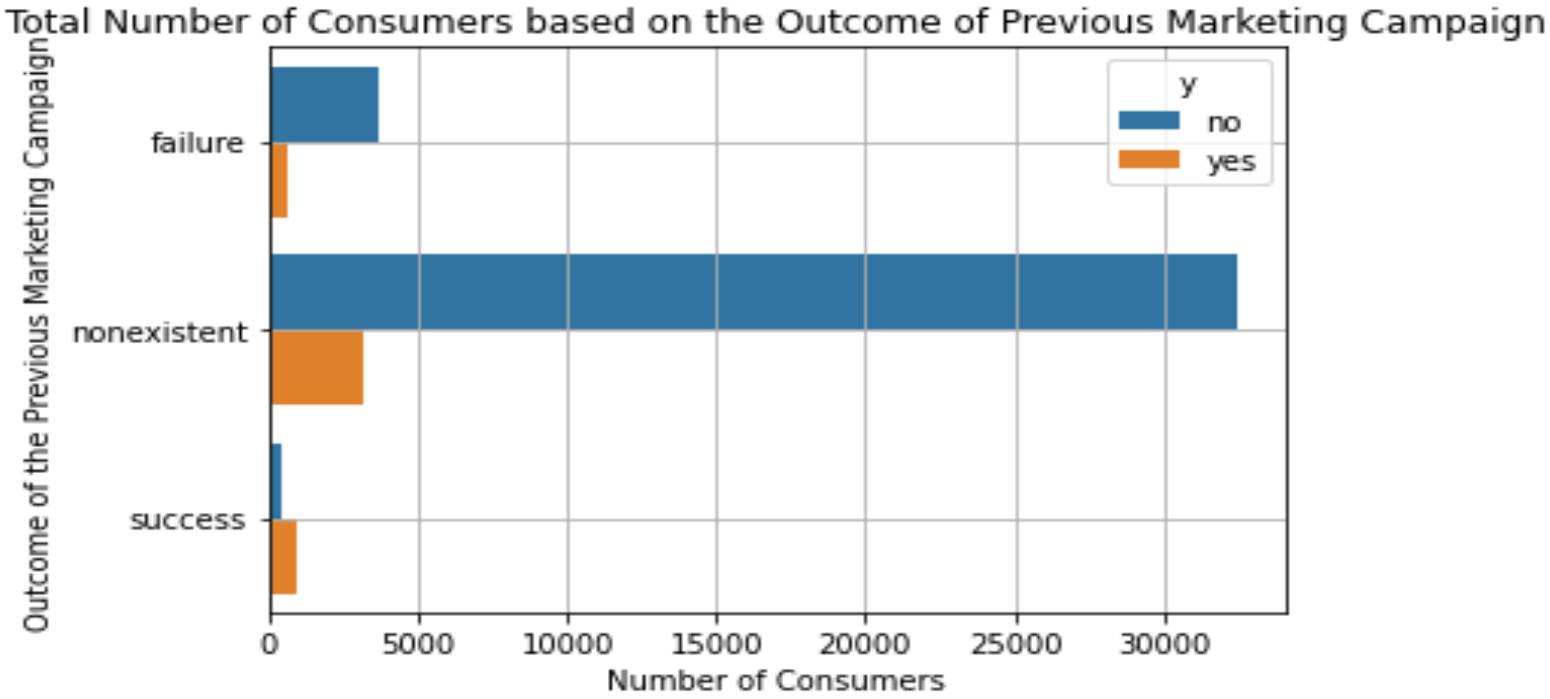

- This graph shows that in the case of those who subscribed we have no differences. However, in those who did not, it seems that in 93.5 and 94.0 price we have the highest number of consumers.

# Scatterplot of the total number of consumers last contact duration




- As regards the graph of the total number of consumers last contact duration, there are plenty of interesting observations. The most paramount one, is that in most cases of those who subscribed, the duration of the last contract is approximately zero, hence there was no last contract. This outcome is also confirmed by the fact that those who did not subscribe, had long duration in their last contract and decided not to subscribe after that, implying a clear dissatisfaction. As there are lots of zero inputs in this variable, we may exclude it in our final model. Outliers are detected in the box plot of duration.

# Scatterplot of the total number of consumers based on the outcome of the previous marketing campaign




- In our final graph, we detect a clear difference in the categories since there are over 30000 of nonexistent consumers who did not have previous marketing campaign. In addition, as we expected, those who had a previous successful campaign, finally decided to subscribe in a term deposit.

# Conclusions-Recommendations

- To sum up, variables such as duration, month, poutcome and age may have a significant role for the ABC bank to sell their term deposit to the customers.
- The recommended models as regards this dataset are as follows:
- Logistic Regression: model using logistic function to predict the result
- Decision Tree Classifier: model using series decision nodes to predict the result
- Random Forest Classifier: model using multiple decision tree classifier to predict the result
- Support Vector Classifier: model using vectors to predict the result
- Gradient Boosting Classifier: model using sequence of sub-models to sequentially correct predecessor's error and improve its performance
- KNearest Classifier: model using nearest datapoints to predict the result

# Results – Metrics

| | LogisticRegression | DecisionTreeClassifier | RandomForestClassifier | SVC | XGBClassifier | KNeighborsClassifier |
|---|---|---|---|---|---|---|
| **accuracy** | 0.91 | 0.89 | 0.91 | 0.9 | 0.91 | 0.89 |
| **f1_score** | 0.48 | 0.52 | 0.57 | 0.35 | 0.59 | 0.36 |
| **cross_val_score** | 0.91 | 0.89 | 0.91 | 0.9 | 0.91 | 0.89 |
| **recall** | 0.38 | 0.52 | 0.5 | 0.23 | 0.55 | 0.28 |
| **confusion_matrix** | [[7132, 178], [575, 353]] | [[6860, 450], [444, 484]] | [[7055, 255], [460, 468]] | [[7204, 106], [711, 217]] | [[7018, 292], [422, 506]] | [[7077, 233], [669, 259]] |

- With f1_score, we calculate a measure between precision and recall, where F1 score reaches its best value at 1 and worst score at 0.
- With cross_val_score, we evaluate the score by cross-validation.
- With recall, we calculate the true positive rate that was found by the model. The formula is: recall = tp/(tp+fn).

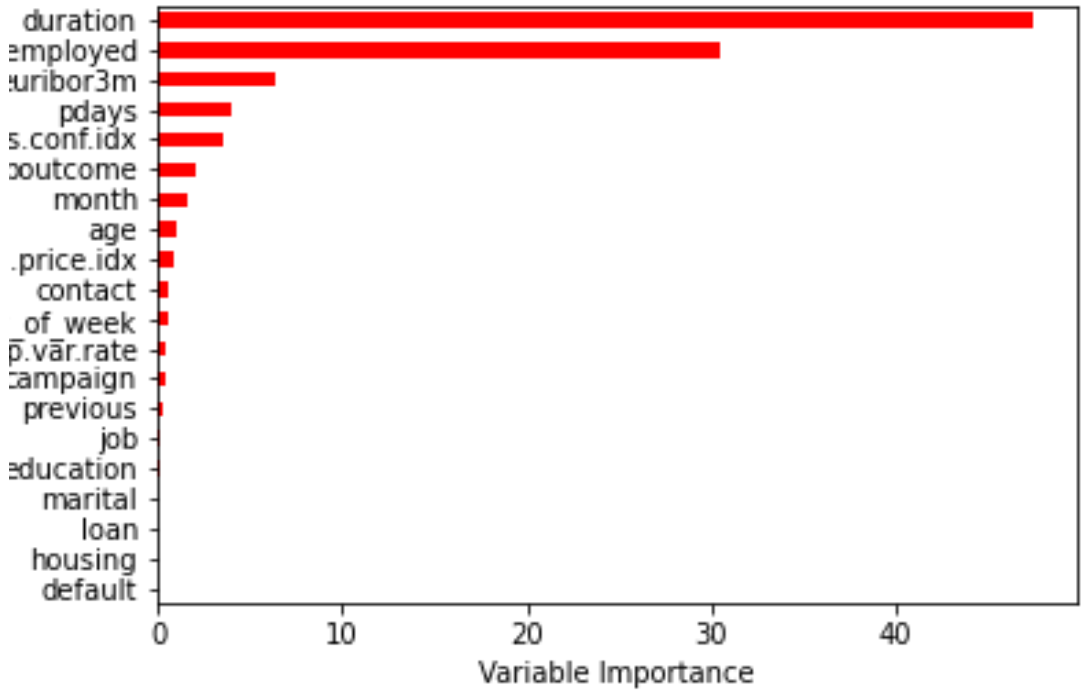
# Results – Metrics

- False Positive, means the client did **NOT SUBSCRIBE** to term deposit, but the model thinks he did, which is called a Type I Error.
- False Negative, means the client **SUBSCRIBED** to term deposit, but the model said he didn’t, which is called a Type II Error.

- Type II Error is the most harmful, because we think that we already have that client but we don’t and maybe we lost him in other future campaigns.
- Type I Error is not desirable as well, but we have that client and in the future we'll discover that in reality he's already our client.

# Results – Metrics

- Before we proceed into the final step of our project we use boosting in order to see which features should be utilized in our model.



# Results – Metrics

- As we can see, the most important features are duration, nr.employed, euribor3m, pdays and cons.conf.idx.
- Finally, we apply the important features into our XGboost model, since it provided the best performance in the previous assignment.

```
Accuracy: 0.92
F1_score: 0.62
Cross_val_score: 0.91
Recall: 0.57
Confusion Matrix:
 [[7057  262]
 [ 392  527]]
```

# Final Model Selection

- Based on the results we can suggest confidently that the XGBoost will fit the needs of the company.
- Higher precision score it indicates “How many predicted values are relevant?”
- Recall score that asks “How many relevant items are selected?”
- F1 score that is a measure using both precision and recall scores.
- In conclusion it has provided satisfactory results.
- It’s a reasonably good model for the bank to try and reach out to new customers hence we would definitely recommend using it.

# Thank You