**1: Understanding the Data**

The data is related with direct marketing campaigns of a Portuguese banking institution. The bank used its own contact-center to do directed marketing campaigns. The telephone, with a human agent as the interlocutor, was the dominant marketing channel, although sometimes with an auxiliary use of the Internet online banking channel (e.g. by showing information to specific targeted client). Furthermore, each campaign was managed in an integrated fashion and the results for all channels were outputted together. The dataset collected is related to 17 campaigns that occurred between May 2008 and November 2010, corresponding to a total of 79354 contacts. During these phone campaigns, an attractive long-term deposit application, with good interest rates, was offered. For each contact, a large number of attributes were stored and if there was a success (the target variable). For the whole database considered, there were 6499 successes (8% success rate)

**2: Read in the Data**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**3: Understanding the Features**

Important note: 'duration': last contact duration, in seconds (numeric) attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, 'duration' input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model, therefore, has been taken out for regressions models

'euribor3m' = Euro Interbank Offered Rate

**4: Understanding the Task**

*Business Objective* of the task

Boost bank funds by offering clients competitive interest rates on long-term deposits and identify more effective marketing strategies to enhance success rates. Current marketing efforts yield only an 8% success rate, with 6,499 successful conversions, underscoring the need for optimized outreach and engagement channels

**5: Engineering Features**

‘feature\_importances’ used along with Random Forests for Feature Selection

A screenshot of a computer

Description automatically generated

**6: Train/Test Split**[**¶**](http://localhost:8889/notebooks/Downloads/module_17_starter/module_17_starter/prompt_III.ipynb#Problem-6:-Train/Test-Split)

Training set shape (X\_train): (32950, 20)

Testing set shape (X\_test): (8238, 20)

Training labels shape (y\_train): (32950,)

Testing labels shape (y\_test): (8238,)

**7: A Baseline Model**

Used of a naive model Majority class classifier.

Baseline accuracy (majority class classifier): 0.8873

**8: A Simple Model**

Used Logistic Regression to build a basic model on the data.

Logistic Regression Model Accuracy: 0.8555

**9: Score the Model**

A screenshot of a computer

Description automatically generated

**10: Model Comparisons**

A screenshot of a graph

Description automatically generated

The table shows a comparison of four machine learning models: Logistic Regression, K-Nearest Neighbors (KNN), Decision Tree, and Support Vector Machine (SVM), evaluated based on three metrics: Train Time (s), Train Accuracy, and Test Accuracy.

**1. Train Time (s)**

SVM has the longest training time by a large margin at 68.11 seconds, which could be due to its computational complexity, especially on larger datasets. Logistic Regression has a moderate training time at 2.96 seconds. KNN and Decision Tree are the fastest, with both taking around 0.21 seconds to train. This is expected as KNN and Decision Trees are generally faster to train compared to SVM.

**2. Train Accuracy**

Decision Tree has the highest training accuracy at 96.71%, indicating it fits the training data very well. However, this could mean it's overfitting. KNN also has high training accuracy at 89.77%, but not as high as the Decision Tree, which may imply it's somewhat overfitted though less than the Decision Tree. Logistic Regression and SVM have similar train accuracies around 71-72%, suggesting they might generalize better than the Decision Tree on the dataset.

**3. Test Accuracy**

Decision Tree has the highest test accuracy at 83.15%, indicating it performs well on unseen data. KNN has a test accuracy of 76.80%, lower than its training accuracy, indicating some overfitting but still decent performance. Logistic Regression and SVM have similar test accuracies around 72.98% and 72.78%, respectively. Both models are consistent between training and test accuracies, showing they might generalize better, but with slightly lower performance than Decision Tree on the dataset.

**Summary**

Decision Tree provides the highest test accuracy, but its high train accuracy suggests it may be overfitting. Logistic Regression and SVM show balanced train and test accuracies, suggesting good generalization but slightly lower performance.

KNN offers good accuracy but shows some overfitting as well

**11: Improving the Model**

* Question more feature engineering and exploration. For example, should we keep the gender feature? Why or why not?

Answer: As seen age is little correlation with target variable

**A graph with different colored bars

Description automatically generated**

* Question: Hyperparameter tuning and grid search. All of our models have additional hyperparameters to tune and explore. For example the number of neighbors in KNN or the maximum depth of a Decision Tree.

Answer:

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

Decision Tree could be fine-tuned (e.g., pruning or setting a maximum depth) to reduce overfitting.

KNN might benefit from hyperparameter tuning (such as optimizing the number of neighbors) to enhance generalization.

SVM could be computationally expensive, so it might not be ideal for larger datasets unless accuracy justifies its use.

Logistic Regression is a good choice for balanced performance and interpretability.

Interpretation: SVM generalizes well but has the lowest accuracy of the models, likely due to underfitting. It may benefit from kernel adjustments or tuning parameters like C and gamma.