Berkeley Machine Learning and AI Practical Assignment III

Bank Direct Marketing Classification

USING MACHINE LEARNING FOR BANK DIRECT MARKETING: AN APPLICATION OF THE CRISP-DM METHODOLOGY

Introduction

Overview:

In this third practical application assignment, we are tasked with comparing the performance of the classifiers (k-nearest neighbors, logistic regression, decision trees, and support vector machines).We were provided a dataset related to the marketing of bank products over the telephone.

The data is from a Portuguese banking institution and is a collection of the results of multiple marketing campaigns.

Deliverables:

After understanding, preparing, and modeling the data, I built a Jupyter Notebook that includes a clear statement demonstrating my understanding of the business problem, a correct and concise interpretation of descriptive and inferential statistics, my findings, and next steps and recommendations.

I compare the performance of the classifiers (k-nearest neighbors, logistic regression, decision trees, and support vector machines) .

Sources:

1) https://archive.ics.uci.edu/ml/datasets/bank+marketing
2) *Download*: Data Folder, Data Set Description
3) [Moro et al., 2014] S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014
4) https://core.ac.uk/display/55616194?utm_source=pdf&utm_medium=banner&utm_campaign=pdf-decoration-v1

Data Set Characteristics:	Multivariate	Number of Instances:	45211	Area:	Business
Attribute Characteristics:	Real	Number of Attributes:	17	Date Donated	2012-02-14
Associated Tasks:	Classification	Missing Values?	N/A	Number of Web Hits:	1809591

Analysis and Findings

Subscribe term deposit prediction with Bank data

Data Set Information:

The data is related to 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 predict if the product (bank term deposit) would be ('yes') or not ('no') subscribed.

The classification goal is to predict if the client will subscribe (yes/no) to a term deposit (variable y). Attribute Information:

Attribute Information:

Bank client data:

- Age (numeric)
- 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', 'unknown' ; note: 'divorced' means divorced or widowed)
- Education (categorical: 'basic.4y', 'basic.6y', 'basic.9y', 'high.school', 'illiterate', 'professional.course', 'university.degree', 'unknown')
- Default: has credit in default? (categorical: 'no', 'yes', 'unknown')
- Housing: has a housing loan? (categorical: 'no', 'yes', 'unknown')
- Loan: has a personal loan? (categorical: 'no', 'yes', 'unknown')

Related with the last contact of the current campaign:

- Contact: contact communication type (categorical: 'cellular','telephone') Month: last contact month of year (categorical: 'jan', 'feb', 'mar', ..., 'nov', 'dec')
- Day_of_week: last contact day of the week (categorical: 'mon','tue','wed','thu','fri')
- Duration: last contact duration, in seconds (numeric). Important note: this 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, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.

Other attributes:

- Campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)
- Pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
- Previous: number of contacts performed before this campaign and for this client (numeric)
- Poutcome: outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')

Social and economic context attributes

- Emp.var.rate: employment variation rate - quarterly indicator (numeric)
- Cons.price.idx: consumer price index - monthly indicator (numeric)
- Cons.conf.idx: consumer confidence index - monthly indicator (numeric)
- Euribor3m: euribor 3 month rate - daily indicator (numeric)
- Nr.employed: number of employees - quarterly indicator (numeric)

Output variable (desired target):

y - has the client subscribed a term deposit? (binary: 'yes', 'no') '''

CISSP-DM Model Process

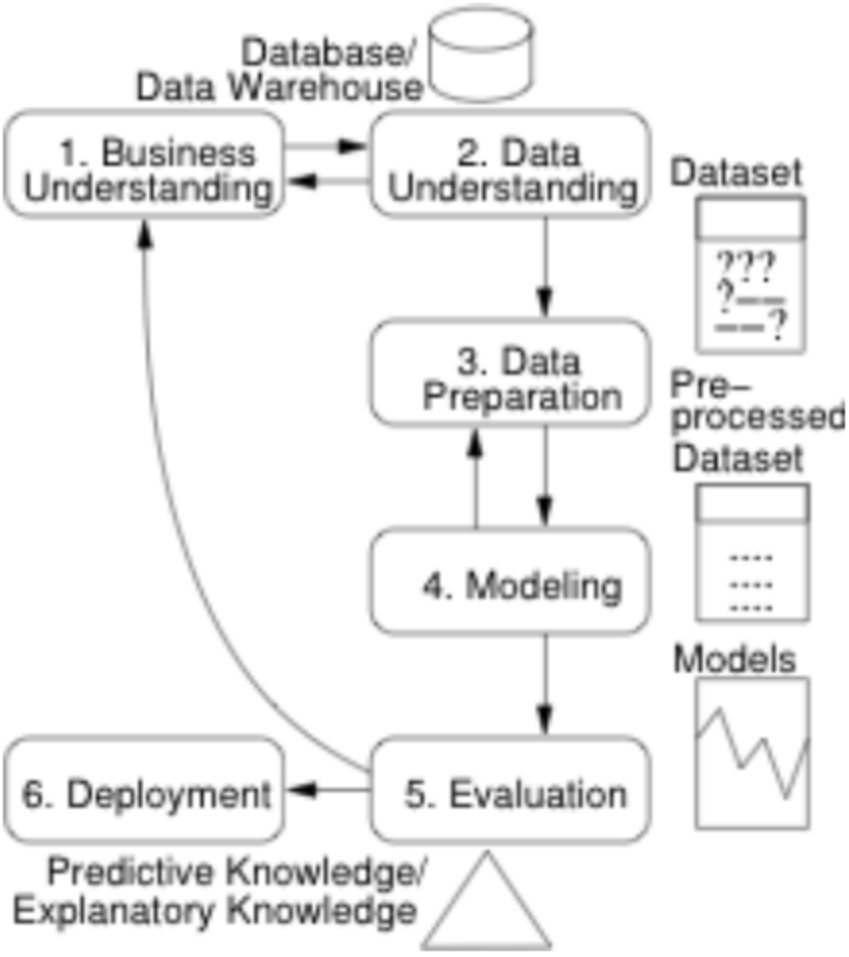


Figure 1 The CRISP-DM process model (adapted from Chapman et al., 2000)

Goal of the case study

In the above research paper, the researchers have mentioned that the best results achieved is of AUC- value 0.9, so our goal for this case study will be to achieve similar results as of the research paper. Though it is to be mentioned that after going through the research paper I realized that the data does not have all the features that are mentioned in the paper.

Type of Machine Learning Problem:

It is a binary classification problem where our objective is to predict whether a customer will subscribe to a term deposit or not given the data of the customer.

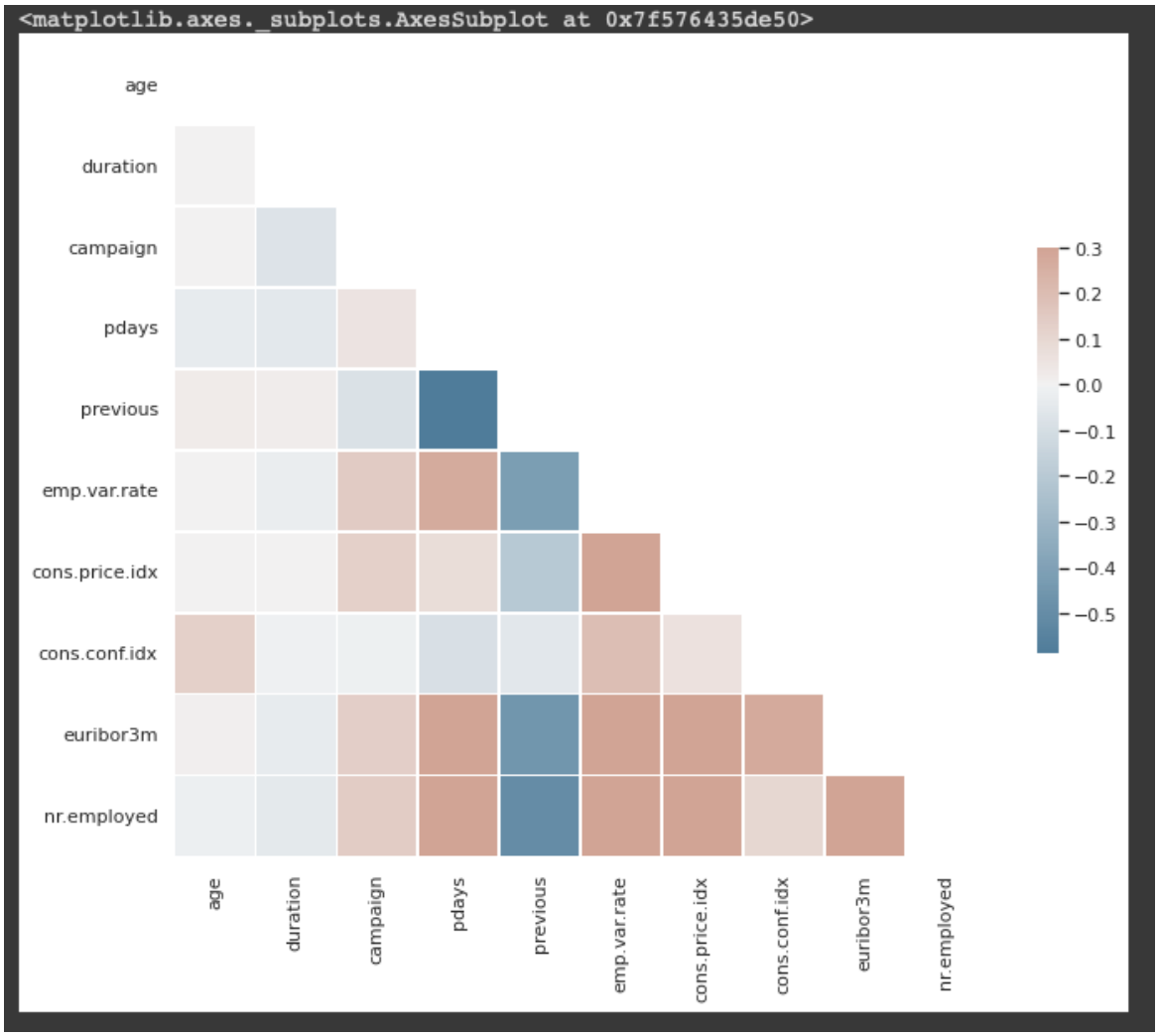
Analysis Report

The Analysis of the Logistic Regress, Decision Tree Classifier, KNeighborsClassifier and the Support Vector Machines was performed according to the following criteria
1. Imbalance Class Handling
2. Model Training Speed
3. Interpretable Results

Other criteria observed include
1. Accuracy
2. Precision
3. Recall
4. Specificity
5. Mean Squared Error

Steps I followed:

1) I performed exploratory data analysis (EDA) in the attached Notebook named "**Practical_Application_III_TB_1_Intro_and_EDA_Final.ipynb"**
    a) Performed analysis on the categorical feature
    b) Analyzed the numerical features



2) As a second step, I performed Feature Engineering
    a) Investigated missing values and other attributes of the data that needed to addressed
    b) Encoded the data
    c) Normalized and scaled the data to get ir ready for machine learning model building

3) Performed Classification and stated to build the baseline model in the file **“Practical Application III-TB-1-FeatureEngr-Modelling-Classification-Final.ipynb”**

```
Run time for baseline model :  0.09002113342285156
Accuracy: 0.8873239436619719
Precision: 0.0
Recall: 0.0
F1 Score: 0.0
Cohens Kappa Score: 0.0
Area Under Curve: 0.6216454124698279
Confusion Matrix:
 [[5481    0]
 [ 696    0]]
              precision    recall  f1-score   support

           0       0.89      1.00      0.94      5481
           1       0.00      0.00      0.00       696

    accuracy                           0.89      6177
   macro avg       0.44      0.50      0.47      6177
weighted avg       0.79      0.89      0.83      6177
```

4) I created a simple LogisticRegression without any hyper-parameter tuning. Here are the results:

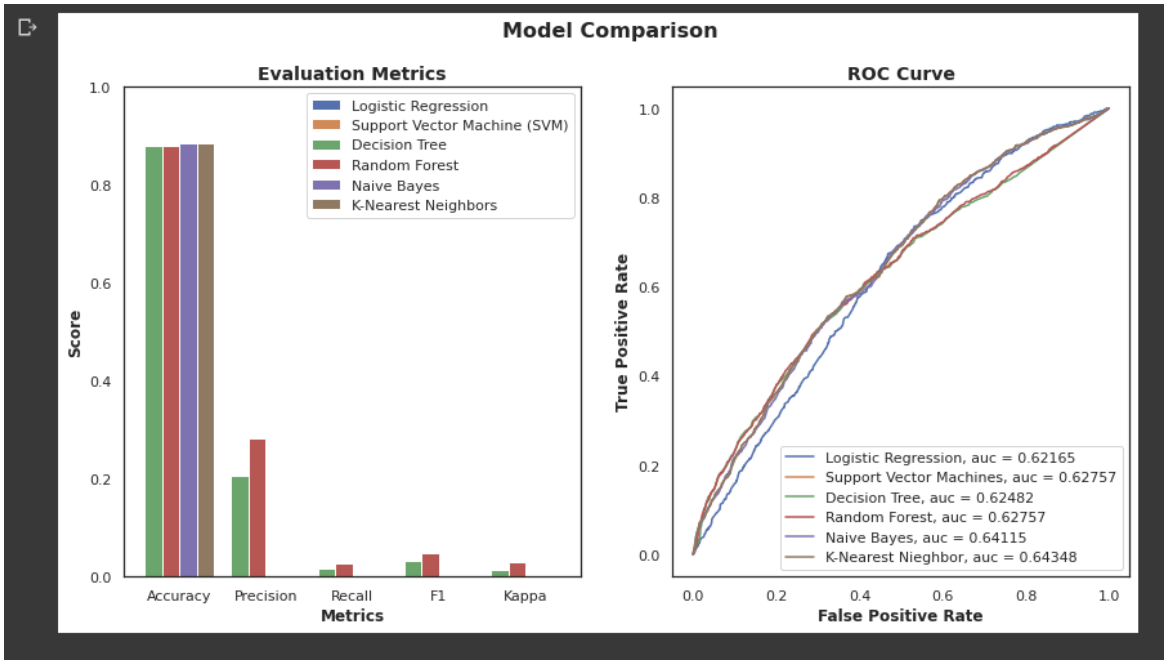
```
Run time for baseline model :  0.07568192481994629
Accuracy: 0.8873239436619719
Precision: 0.0
Recall: 0.0
F1 Score: 0.0
Cohens Kappa Score: 0.0
Area Under Curve: 0.6216454124698279
Confusion Matrix:
 [[5481    0]
 [ 696    0]]
              precision    recall  f1-score   support

           0       0.89      1.00      0.94      5481
           1       0.00      0.00      0.00       696

    accuracy                           0.89      6177
   macro avg       0.44      0.50      0.47      6177
weighted avg       0.79      0.89      0.83      6177
```

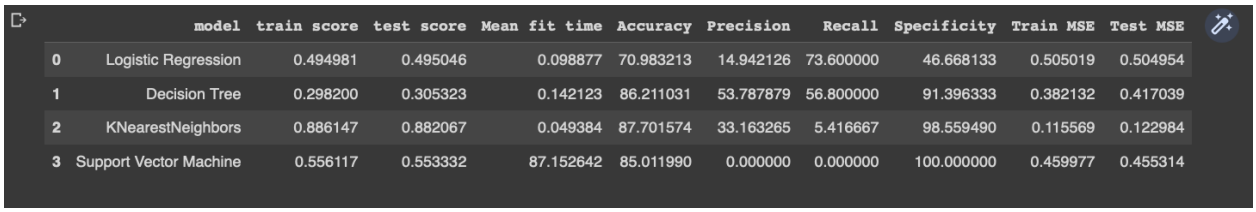
5) Finally, I created the models to compare to my baseline model of **Logistic Regression, Support Vector Machines, Decision Trees, Random Forest,**

Naive Bayes, and KNN using SKlearn. The graph below shows the comparison of all these algorithms.



6) Next, I worked on improving Logistic Regression, Sedition Tree, KNeighbours, and Support Vector Machine. All of our models had additional hyperparameters to tune

and explore. The table below shows the results of - Hyperparameter tuning and grid search.



	model	train score	test score	Mean fit time	Accuracy	Precision	Recall	Specificity	Train MSE	Test MSE
0	Logistic Regression	0.494981	0.495046	0.098877	70.983213	14.942126	73.600000	46.668133	0.505019	0.504954
1	Decision Tree	0.298200	0.305323	0.142123	86.211031	53.787879	56.800000	91.396333	0.382132	0.417039
2	KNearestNeighbors	0.886147	0.882067	0.049384	87.701574	33.163265	5.416667	98.559490	0.115569	0.122984
3	Support Vector Machine	0.556117	0.553332	87.152642	85.011990	0.000000	0.000000	100.000000	0.459977	0.455314

Summary and Conclusion

- Logistic Regression Classifier
    - Handles Imbalanced Classes
    - Speed of Training is moderately high at a Mean Fit Time of 0.09
    - Test Score was low at 50%, close to the same as the Train Score at 49%
    - Accuracy and Specificity are not too high at 71%.
    - Precision is low at 14.9
    - Recall is high at 73%
    - It appears not to overfit
- Decision Tree Classifier
    - Handles Imbalanced Classes
    - Speed of Training is moderately high at a Mean Fit Time of 0.14
    - Accuracy and Specificity are very high at 86% and 91%, respectively
    - The Test Score performs slightly better than the Train Score
    - Precision is above average at 53%
    - Recall is above average at 57%
    - The Decision Tree Classifier appears to overfit with Test MSE higher than Train MSE
- KNearest Neighbors Classifier (KNN)
    - KNN cannot handle Imbalanced classes
    - Speed of Training is very high at a Mean Fit Time of 0.04
    - Accuracy and Specificity are very high at 86% and 98%, respectively
    - The Train Score performs slightly better than the Test Score
    - Precision is low at 33%

    - Recall is very low at 5.4%
    - However, the KNNeighbors Classifier appears to overfit with Test MSE higher than Train MSE
- Support Vector Machine
    - SVM handles Imbalanced classes
    - Speed of Training was slow with a Mean Fit Time of 87.2
    - Accuracy and Specificity are very high at 85% and 100%, respectively
    - Precision is at 0
    - Recall is low at 0%
    - However, the KNNeighbors Classifier appears to not overfit with
    - Test MSE and Train MSE are close at 45%
    - The Test Score performs better than the Train Score

Model Selected: Logistic Regression

- The Logistic Regression Classifier is the one I would select based on the above table

Conclusion

For a simple model we can see that our model did decently on classifying the data. But there are still some weaknesses in our models, especially shown on the recall metric where we only get about 73%. This means that our models are only able to detect 73% of potential customers and miss the other 27%. The result is not that much different after optimizing the model using `GridSearchCV` which means that we hit our limit with this model. To improve our performance we can try to look into another algorithm such as `GradientBoostingClassifier.