**Issue**: To predict whether a customer will make a term deposit based on the demographic data and responses to prior marketing campaign.

**Summary:** The K nearest neighbor (“KNN”) model with neighbors equals 8 was the most accurate model with a F1- score of 84% of predicting whether a customer would make deposit based on marketing campaign information. The F1-score is a balanced metric of precision and accuracy.

We tried the KNN, RandomForest(“RF”), and Naïve Bayes Bernoulli (“NBB”) models on the data set. The dataset had 21K observations, which we partitioned into 80% training data and 20% testing data. We used 5 – fold cross validation in our models.

We took a non-parametric approach, since there are over 10 variables that could influence whether an individual will make a term deposit. The attributes with the strongest correlations were (poutcome = true) and (contact = ‘cellular” ) with 30.7% and 13.6% correlation, respectively. These correlations are logical as most people use cell phones as their main contact method and if prior marketing campaigns were successful, the next marketing campaign is more likely to be successful.

**Data Source**: <https://archive.ics.uci.edu/ml/datasets/bank+marketing>

**Data Attributes**:

Input variables:

# bank client data:

1 - age

2 - job

3 -marital status

4 - education

5 - default: has credit in default?

6 - housing: has a housing loan?

7 - loan: has a personal loan?

# related with the last contact of the current campaign:

8 - contact: contact communication type (categorical: 'cellular','telephone')

9 - month: last contact month of year (categorical: 'jan', 'feb', 'mar', ..., 'nov', 'dec')

10 - day\_of\_week: last contact day of the week (categorical: 'mon','tue','wed','thu','fri')

# other attributes:

12 - campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)

13 - pdays: number of days that passed by after the client was last contacted from a previous campaign

14 - previous: number of contacts performed before this campaign and for this client (numeric)

15 - poutcome: outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')

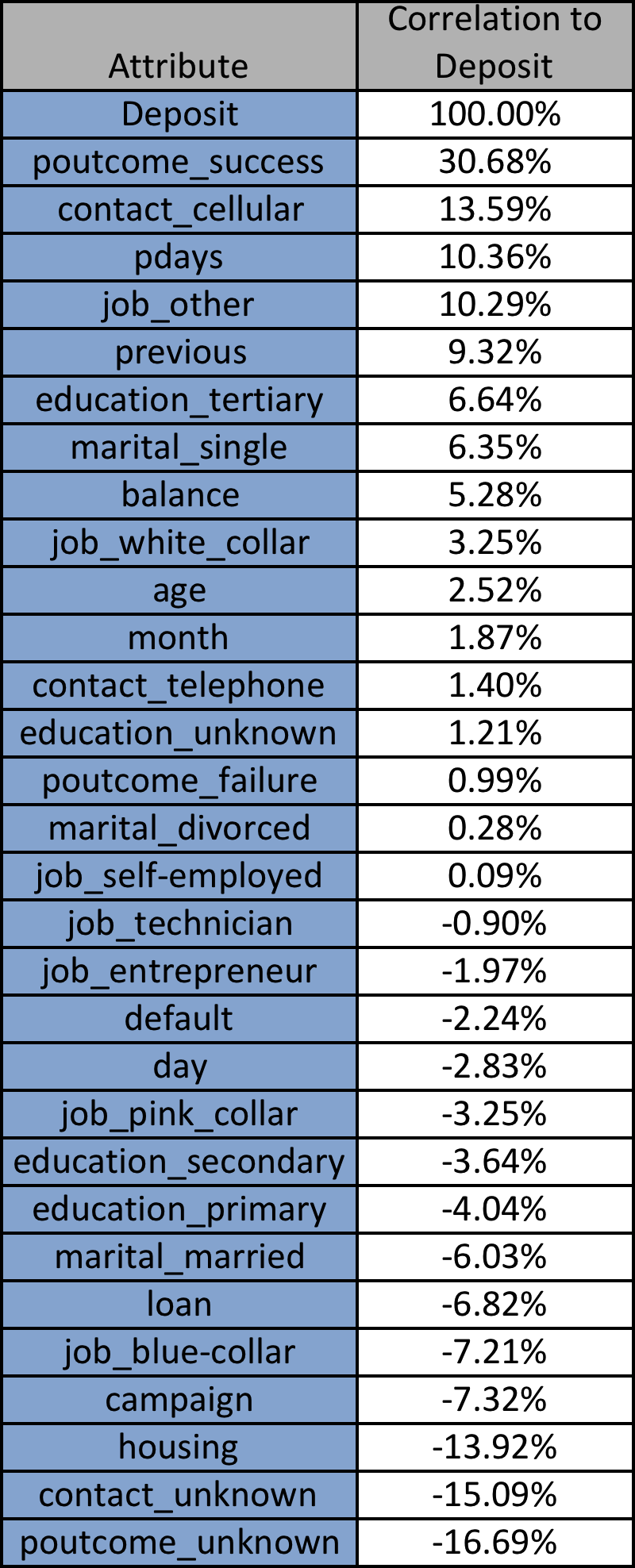
Output variable (desired target):

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

**Data cleaning steps**:

1. Reduce the number of categories for jobs to simply model ( e.g. adding management jobs to blue collar category)
2. Map categorical data to integer value for the following attributes (month, default, housing loan, personal loan, y(deposit) )
   1. Yes no binary mapping (1 = yes, 0 = no)
3. Dummifying remaining categorical attributes. Dummifying changes categories attributes into several columns with binary values for each category. For example, the job attribute has several categories (white-collar, blue-collar, service) and the “get dummies” function creates a columns for each category with a 1 and 0 as a result( 1 = True, 0 = False).
4. Create test train split for machine learning.

**Deposit Correlation**:



**Best Model**

KNN with neighbors equal 8 is the best model. Its accuracy metrics were similar to RandomForest. But it did not show signs of overfitting. Based on the graphs, the RF model shows signs of overfitting and NBB models had a gap in between training and test accuracies.

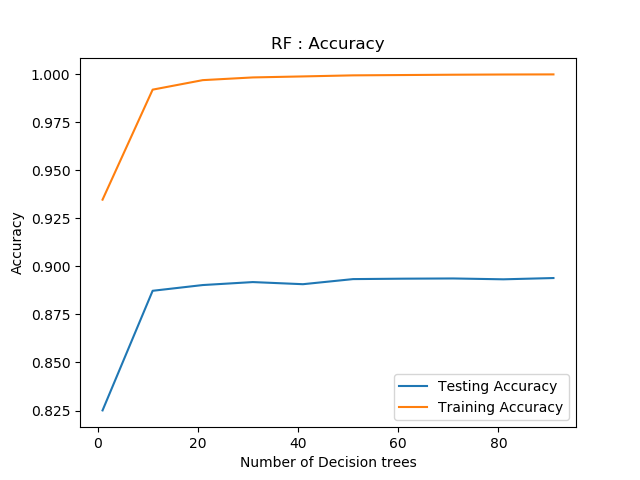
**Machine learning models and output**:

1. KNN



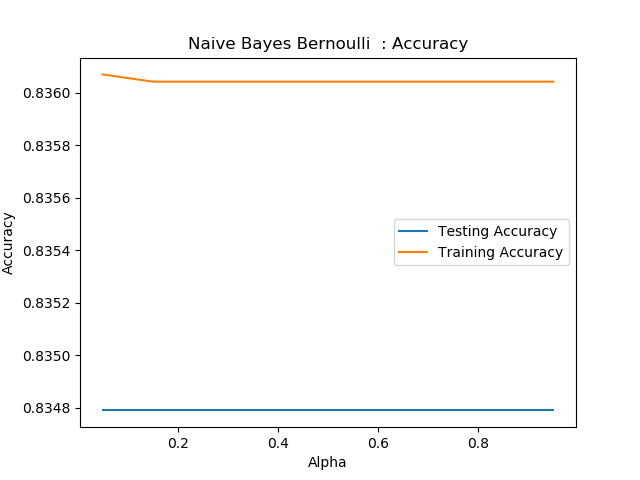
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Deposit | precision | recall | f1-score | support |
| No | 0.89 | 0.99 | 0.94 | 7986 |
| Yes | 0.47 | 0.06 | 0.11 | 1057 |
| avg/total | 0.84 | 0.88 | 0.84 | 9043 |

1. Random Forest



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Deposit | precision | recall | f1-score | support |
| No | 0.91 | 0.98 | 0.94 | 7986 |
| Yes | 0.61 | 0.23 | 0.33 | 1057 |
| avg/total | 0.87 | 0.89 | 0.87 | 9043 |

1. Naïve Bayes Bernoulli



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Deposit | precision | recall | f1-score | support |
| No | 0.91 | 0.9 | 0.91 | 7986 |
| Yes | 0.3 | 0.31 | 0.3 | 1057 |
| avg/total | 0.84 | 0.83 | 0.84 | 9043 |