Title: Practical Data Science with Python – Assessment 2

Author: Heinzy Johan Bernt

Student number: 3923824

Contact Details: S3923824@student.rmit.edu.au

Date of report: 27th November 2021

Contents

Executive Summary

The dataset explores how successful direct marketing campaigns / phone calls are in getting customers to invest in a term deposit for a Portuguese bank.

The data is treated as a Classification Task. Two models used from sklearn are:

- K Nearest Neighbours (KNN)
- Decision Tree for Classification

For each type of model, the following training / testing splits are used:

- 50% for training and 50% for testing;
- 60% for training and 40% for testing;
- 80% for training and 20% for testing;

The investigation yielding strong prediction accuracies in both the KNN and Decision Tree models. However, the Decision Tree model yielding better prediciton accuracy overall. For a 80:20 train/test split and max depth of 4, the Decision Tree classifier gave a 91% accuracy result i.e. my model was able to predict which customers took up a new term deposit as a result of the marketing campaign with a 91% accuracy.

Introduction

Goal

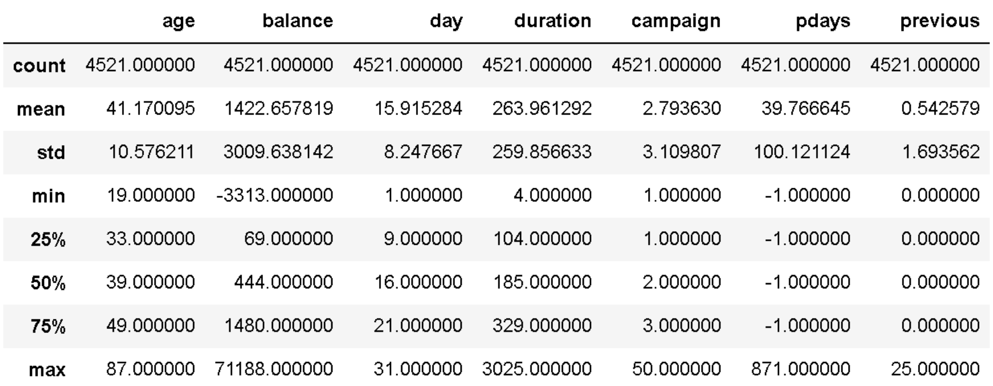
The goal is to predict if a bank's marketing campaigns are successful in getting a customer to take on a new banking product, in this case a term deposit.

Description of the data set

The dataset has 17 columns (variables) and 4521 rows (observations). I will explain the most important input variables for this investigation.

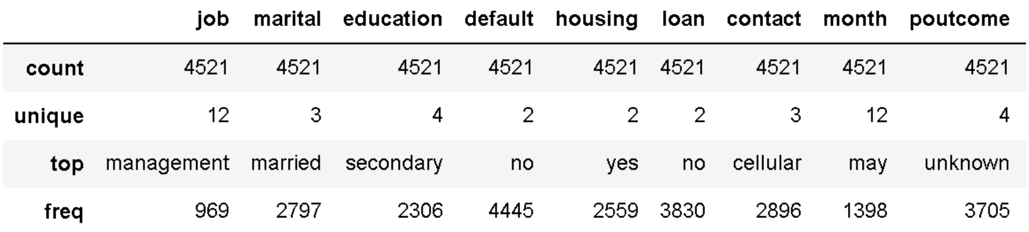
1. age: Age of the customer (numeric).
2. job: Type of job (categorical: 'admin', 'blue-collar', 'entrepreneur', 'housemaid', 'management', 'retired', 'self-employed', 'services', 'student', technician', 'unemployed', 'unknown')
3. marital: marital status (categorical: 'divorced','married','single','unknown'; note: 'divorced' means divorced or widowed)
4. education: refers to three levels of education (categorical: 'primary', 'secondary', 'tertiary', 'unknown')
5. default: has credit in default? (categorical: 'no','yes','unknown')
6. housing: has a housing loan? (categorical: 'no','yes','unknown')
7. loan: has a personal loan? (categorical: 'no','yes','unknown')
8. campaign:number of contacts performed during the campaign (numeric)
9. previous: number of contact calls performed in a previous campaign for the same customer (numeric)
10. poutcome: outcome of the previous telemarketing campaign (categorical: 'failure', 'nonexistent', 'success')
11. balance: this is the customer's bank account balance.

The below table shows the summary statistics of the numeric variables including, count, min and max, mean, standard deviation and interquartile range.



	age	balance	day	duration	campaign	pdays	previous
count	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000
mean	41.170095	1422.657819	15.915284	263.961292	2.793630	39.766645	0.542579
std	10.576211	3009.638142	8.247667	259.856633	3.109807	100.121124	1.693562
min	19.000000	-3313.000000	1.000000	4.000000	1.000000	-1.000000	0.000000
25%	33.000000	69.000000	9.000000	104.000000	1.000000	-1.000000	0.000000
50%	39.000000	444.000000	16.000000	185.000000	2.000000	-1.000000	0.000000
75%	49.000000	1480.000000	21.000000	329.000000	3.000000	-1.000000	0.000000
max	87.000000	71188.000000	31.000000	3025.000000	50.000000	871.000000	25.000000

The below table shows gives some statistics for the categorical variables in the data set, including count, number of unique values, the top value and frequency of the top value.

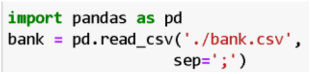


	job	marital	education	default	housing	loan	contact	month	poutcome
count	4521	4521	4521	4521	4521	4521	4521	4521	4521
unique	12	3	4	2	2	2	3	12	4
top	management	married	secondary	no	yes	no	cellular	may	unknown
freq	969	2797	2306	4445	2559	3830	2896	1398	3705

Methodology

Getting started

In order to read the data set I imported pandas and used the pd.read_csv() function, ensuring the data was separated by ';'.

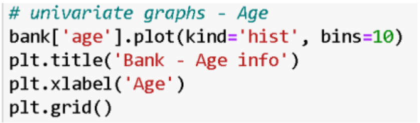


```
import pandas as pd
bank = pd.read_csv('./bank.csv',
                   sep=';')
```

In order to get an overall picture to decide which relationships I wanted to explore further I used a scatter matrix. (Due to space constraints please see .ipynb file.)

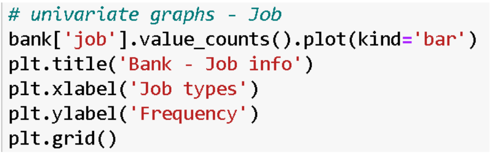
Exploring univariate relationships

For numerical values, I used a histogram, for example Age:



```
# univariate graphs - Age
bank['age'].plot(kind='hist', bins=10)
plt.title('Bank - Age info')
plt.xlabel('Age')
plt.grid()
```

For categorical values I used a bar graph. Note that it was necessary to include the value_counts() function in order to show the frequency of each category, for example Job type:

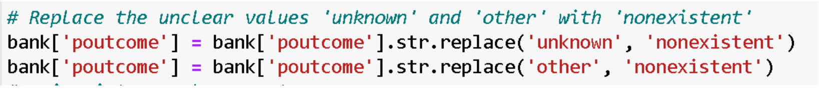


```
# univariate graphs - Job
bank['job'].value_counts().plot(kind='bar')
plt.title('Bank - Job info')
plt.xlabel('Job types')
plt.ylabel('Frequency')
plt.grid()
```

I also used the value_counts() function outside of the plot function to get the counts of each categorical and ensure there were no whitespace erros.

Noted that some variables contain an 'unknown' value. Since a portion of the dataset contains an unknown value, the creator of this data set elected to leave these values in (rather than removing or replacing with a mean or mode value) as these values are attributed to having new customers that might not have been contacted by a previous campaign.

Some data cleaning was required however, where the unclear values in poutcome had to have there values replaced with 'nonexistent', as per the dataset description provided by the author.

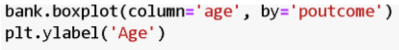


```
# Replace the unclear values 'unknown' and 'other' with 'nonexistent'
bank['poutcome'] = bank['poutcome'].str.replace('unknown', 'nonexistent')
bank['poutcome'] = bank['poutcome'].str.replace('other', 'nonexistent')
```

Lastly, outliers will be left in. Although outliers may increase the variability of the data, for this investigation they will be left to ensure all responses are captured unless it is an obvious data entry error.

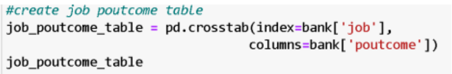
Exploring bivariate relationships

Visualising a bivariate relationship between a <u>numerical and categorical</u> variable can be easily done using a boxplot, for example Age v poutcome:

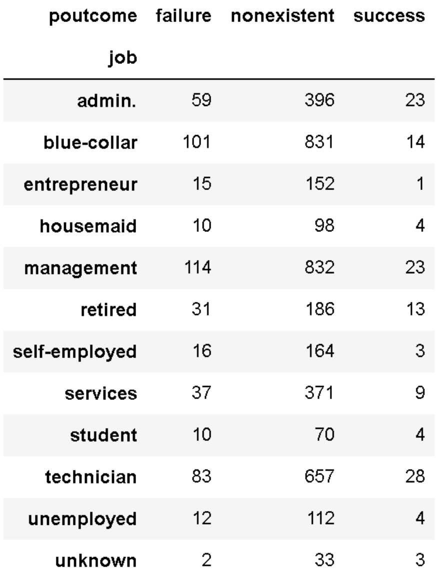


```
bank.boxplot(column='age', by='poutcome')
plt.ylabel('Age')
```

To visualise a bivariate relationship between two categorical variables, I created a crosstab table to find the count of each instance, for example the crosstab table for job type and previous outcome:

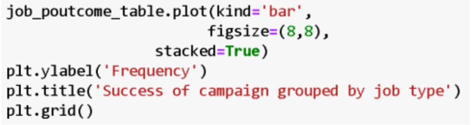


```
#create job poutcome table
job_poutcome_table = pd.crosstab(index=bank['job'],
                                 columns=bank['poutcome'])
job_poutcome_table
```



poutcome	failure	nonexistent	success
job			
admin.	59	396	23
blue-collar	101	831	14
entrepreneur	15	152	1
housemaid	10	98	4
management	114	832	23
retired	31	186	13
self-employed	16	164	3
services	37	371	9
student	10	70	4
technician	83	657	28
unemployed	12	112	4
unknown	2	33	3

Depending on which visualising looked easier to digest, I used either a stacked or unstacked bar graph to visualise bivariate relationships between <u>two categorical variables</u>, for example:



```
job_poutcome_table.plot(kind='bar',
                        figsize=(8,8),
                stacked=True)
plt.ylabel('Frequency')
plt.title('Success of campaign grouped by job type')
plt.grid()
```

However, since each the frequency of each job type differed, I also calculated the <u>rate of success</u> to more easily visualise how successful each job type was as a percentage and this made it easier to compare the length of each bar. This can be calculated by using the parameter normalize='index' within the crosstab() function.

K Nearest Neighbour Method

Training:

I trained the model using a 50/50%, 60/40%, and 80/20% train/test split:

```
x_train, x_test, y_train, y_test = train_test_split(X_data,
                                                    Y_data,
                                                    test_size=0.50,   #this becomes 0.4 and 0.2 for our investigation
                                                    random_state=0)
```

I set random_state=0 for these KNN models so that the randomstate object is randomly initialised.

I also tried different KNN classifier or 3 and 6 for each train/test split to compare results.

Fitting:

Fit the data using fit = clf.fit(x_train, y_train)

Predicting:

I predicted the data using: predicted = fit.predict(x_test)

Create confusion matrix

I created the confusion matrix using confusion_matrix(y_test, predicted)

Classification report

I created the classification report using:

```
report_knn = classification_report(y_test, predicted)
print(report_knn)
```

Lastly, I wanted to validate the accuracy of the results in the classification report. So I used the following formulas for validation:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, predicted)
f1_knn = f1_score(y_test, predicted, average='weighted')
```

Decision Tree Classifier Method

Again I trained the model using a 50/50%, 60/40%, and 80/20% train/test split:

I set random_state=0 for these models so that the randomstate object is randomly initialised.

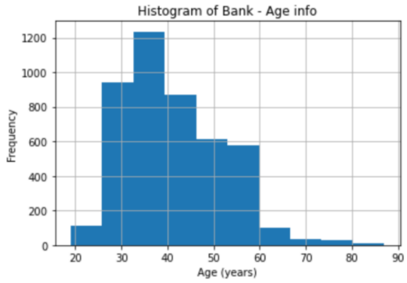
I allowed the Criterion to be the default gini index.

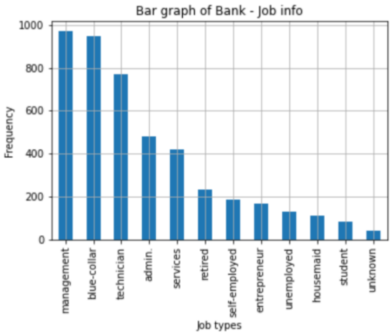
I test max_depth at 3, 4 and 5. More details in the results section that follows.

Results

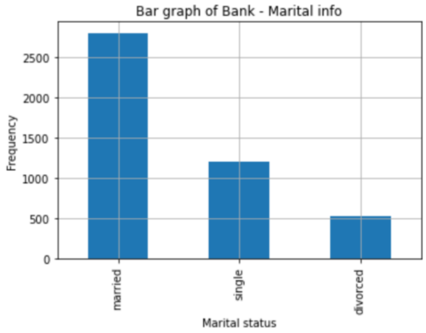
Univariate graphs

Here are the customers Ages shown in a histogram. As can be seen the data is skewed right and most customers fall between age 25 to 40 with the mean being 41 years.

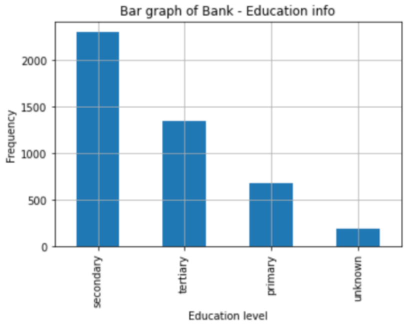




This bar graph shows the count of each job type. Out of 12 unique job types, management and blue-collar roles have the most amount of customers and household duties and students represent the fewest customers.

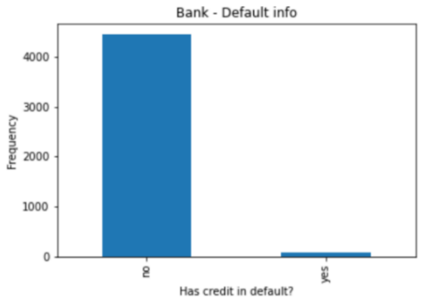


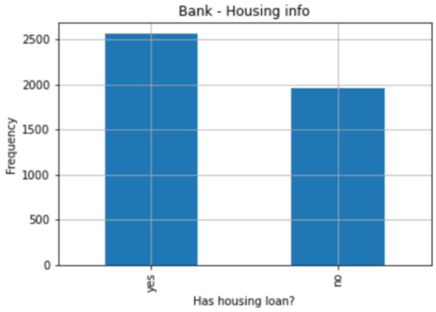
This bar graph shows the count of which customers are married, single or divorced. As can be seen, married customers occur more frequently and divorced customers represent the fewest customers.



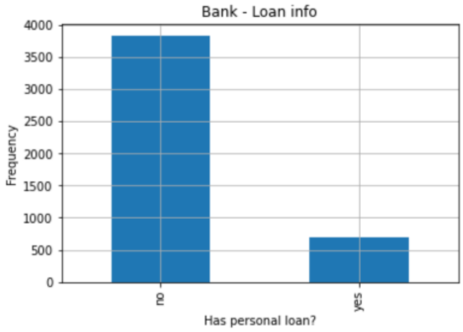
This bar graph shows the frequency of education types. As can be seen more customers have reach secondary education as the highest level of education, followed by tertiary then primary education.

This bar graph shows the proportion of customers that have and hadn't had a credit default. As can be seen only one percent of the customers have had a credit default.

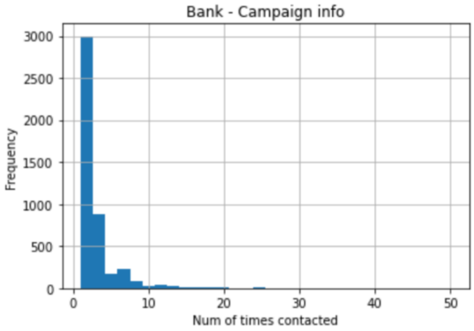




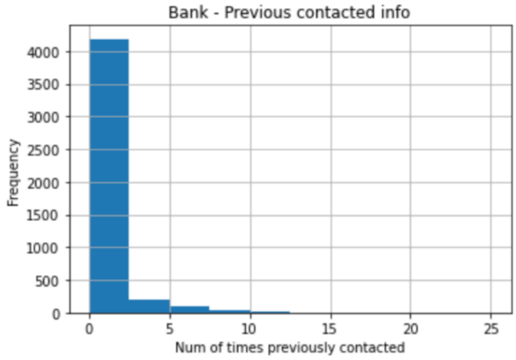
This bar graph shows how many customers have and don't have a housing loan. The split is closer than previous models above with 57% customers do have a housing loan.



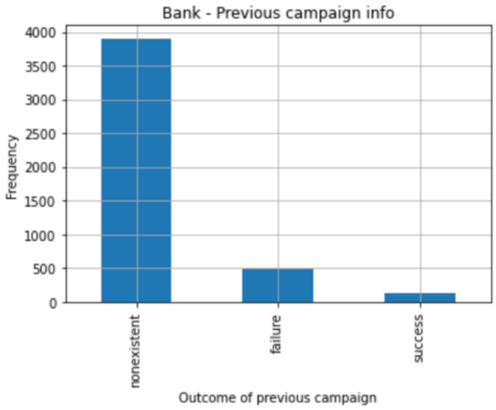
This bar graph shows how many of the customer base have and don't have a personal loan. As can be seen most of the customer base (3830 customers) do not have a personal loan.



This bar graph shows the amount of times a particular customer has been contacted for this current campaign. As can be seen most customers were contacted between 2 to 3 times with the average being 2.71. The max amount of times a customer was contacted was 50, skewing the data.



This bar graph shows the amount of times a particular customer has been contacted in a previous campaign. The results are a bit different this time as customers were contacted fewer times with the average being 0.54.



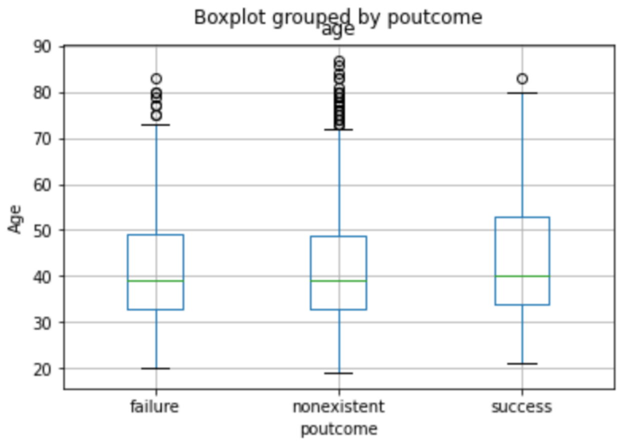
This bar graph shows the success of the previous campaign. As can be seen there is data that is non-existent, however, we can see that the instances of failure is significantly greater than the times the campaign had success.

Bivariate graphs

Hypothesis 1

Older customers are more likely to take up a new product (term deposit) in previous campaigns.

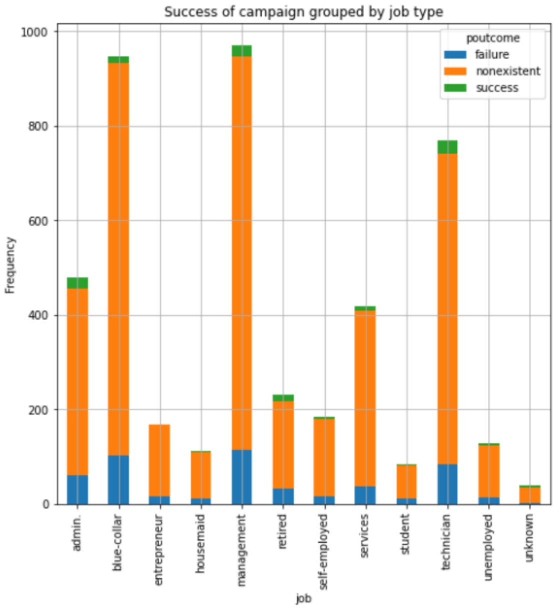
This boxplot shows the relationship between age and the outcome of the previous campaign. As can be seen the IQR, median and max range are higher for ‘success’ over ‘failure’, meaning that our hypothesis might be correct. However, the effect is small.

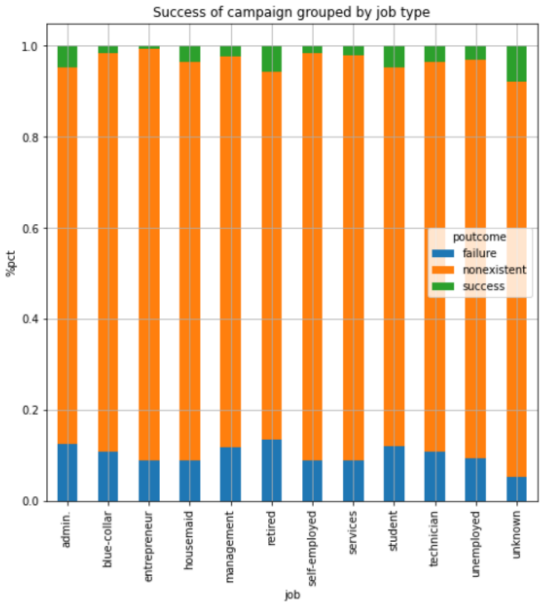


Hypothesis 2

Customers in management roles (possibly due to higher income and education) are the most likely to take on a term deposit, whereas the unemployed will be the least likely (possibly due to not having the financial means to invest in a term deposit).

The graphs here show the relationship between job types and the outcome of a previous campaign.



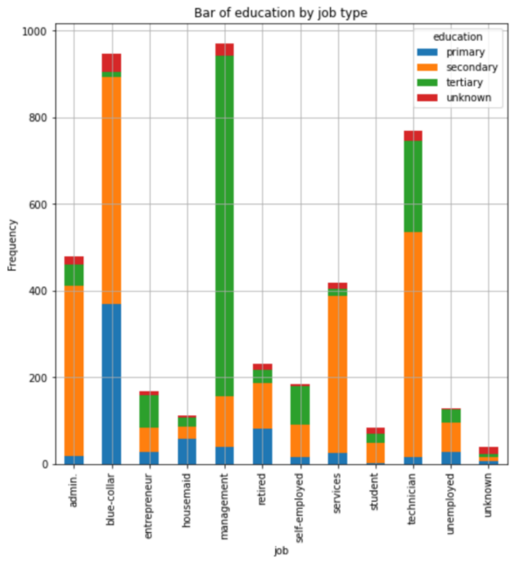


Since it is difficult to visually compare if management roles were the most likely to take up a term deposit I converted this to a rate in the second graph. Here we can see that the green (success) bar is actually greatest for retired customers whereas entrepreneurs took up a term deposit at a lower rate. This does make sense as retirees often require defensive assets (such as term deposits) in their later years. Furthermore entrepreneurs may not have the financial means to invest in a term deposit whilst they invest their efforts in their business.

Hypothesis 3

Customers in management roles have a higher education level than customers in blue-collar roles.

This bar graph shows the relationship between education and job type. As can be seen our hypothesis is correct as quite clearly the green (tertiary) bar is the most significant for

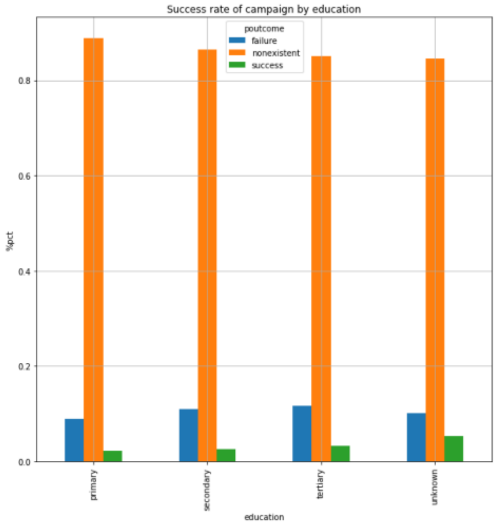


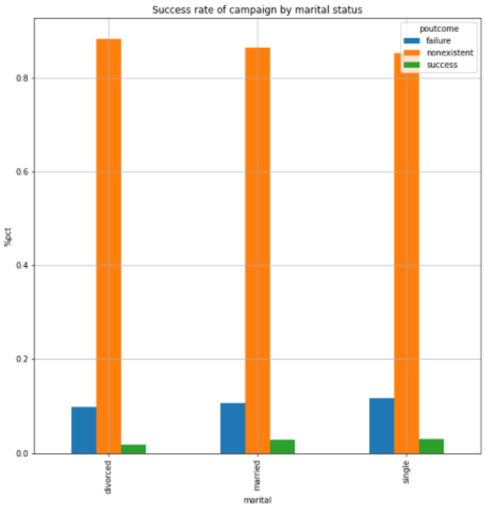
management roles whereas the orange (secondary) bar is most significant for blue-collar workers. Note this was easy to determine and we did not have to convert the data to a rate.

Hypothesis 4

Customers with a lower education are less likely to take up a new term deposit.

This bar graph shows the relationship between education and previous campaign outcome. As can be seen customers with primary education only had the lowest success rate.

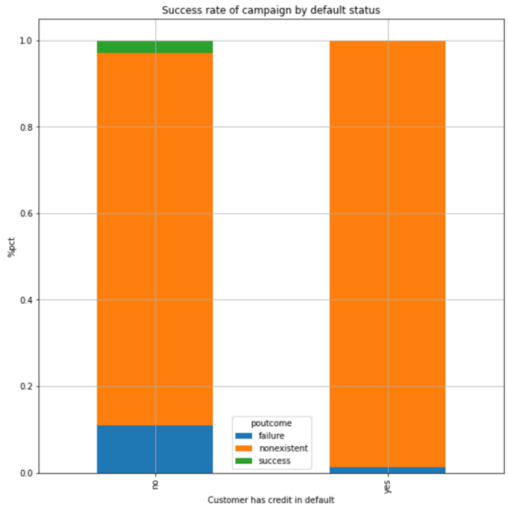




Hypothesis 5

Often a divorce can have financial strain on a person, so I hypothesise that divorcees will take up a new term deposit at the lowest rate.

This bar graph shows the relationship between martial status and previous campaign outcome. As can be seen, divorcees have the lowest take up of a term deposit whereas married and single customers have a similar success rate.



Hypothesis 6

Customers that have had a credit default are less likely to take up a term deposit.

These bar graphs show the relationship between credit default status and the success of the previous campaign.

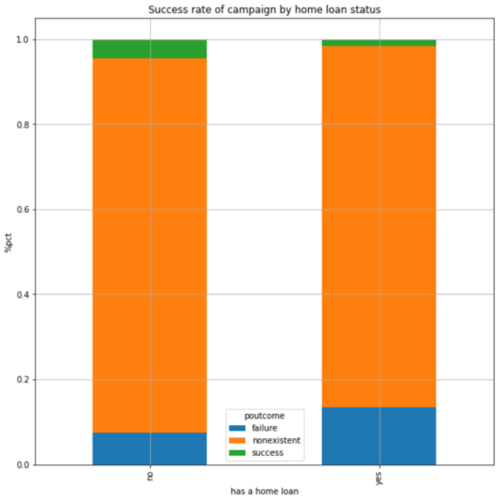
As can be seen those have defaulted were not taking up a term deposit.

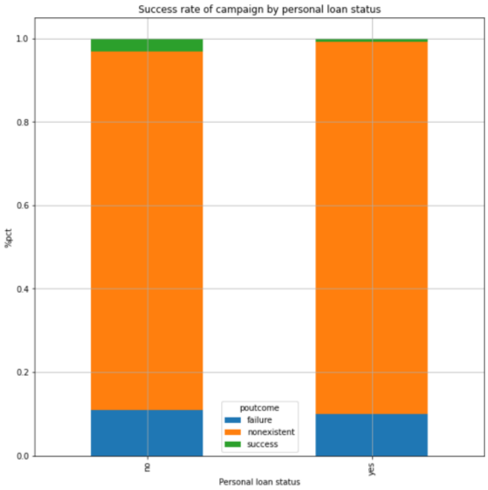
Hypothesis 7

Customers without a home loan are less financially encumbered and therefore have capacity to take on a term deposit.

This stacked bar graph shows the relationship between customers that have a home loan and whether the previous campaign was successful.

As can be seen by the green bar more customers took on a term deposit when they did not have a home loan.



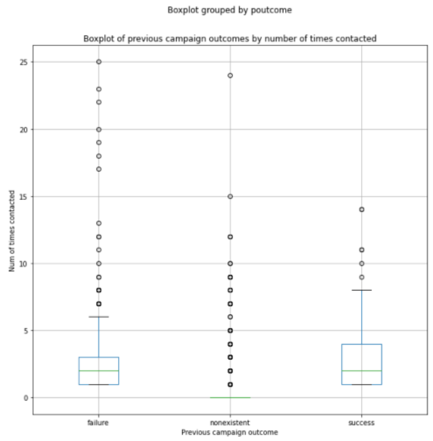


Hypothesis 8

Customers without a personal loan are more likely to take on a term deposit.

This stacked bar graph shows the relationship between customers that have a personal loan and whether the previous campaign was successful.

As can be seen by the green bar more customers took on a term deposit when they did not have a personal loan.



Hypothesis 9

Contacting a customer more times should lead to greater success of the campaign.

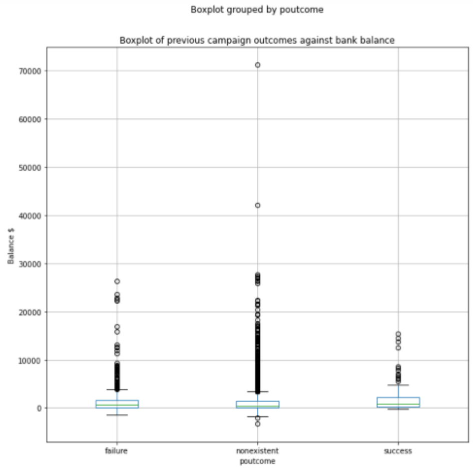
The boxplot here shows the relationship between the number of times a customer was contacted and the success rate of the campaign.

Whilst the success boxplot has a higher Q3 (IQR) median number of times contacted is similar between failure and success therefore the result is inconclusive.

Hypothesis 10

The success of taking up a term deposit will be greater for those customers with a higher bank balance.

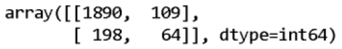
This boxplot shows the bivariate relationship between customers taking up a term deposit and customer’s bank account. As can be seen, the boxplots of failure and success are similar so there is no discernible difference between a customer’s bank balance and whether the previous campaign was a failure of success.



K Nearest Neighbour

50/50% train/test split and KNN=3

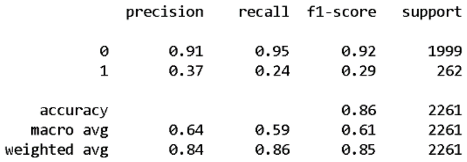
Confusion matrix



```
array([[1890,  109],
       [ 198,   64]], dtype=int64)
```

Classification error rate=0.14

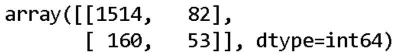
Classification Report



```
              precision    recall  f1-score   support

           0       0.91      0.95      0.92      1999
           1       0.37      0.24      0.29       262

    accuracy                           0.86      2261
   macro avg       0.64      0.59      0.61      2261
weighted avg       0.84      0.86      0.85      2261
```

60/40% train/test split and KNN=3

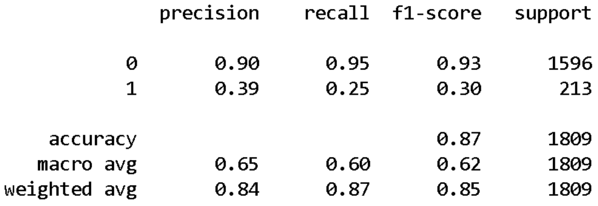
Confusion matrix



```
array([[1514,   82],
       [ 160,   53]], dtype=int64)
```

Classification error rate=0.13

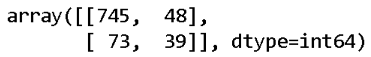
Classification report



```
              precision    recall  f1-score   support

           0       0.90      0.95      0.93      1596
           1       0.39      0.25      0.30       213

    accuracy                           0.87      1809
   macro avg       0.65      0.60      0.62      1809
weighted avg       0.84      0.87      0.85      1809
```

80/20% train/test split and KNN=3

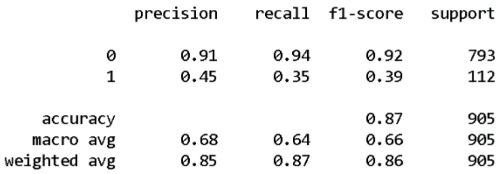
Confusion matrix



```
array([[745,  48],
       [ 73,  39]], dtype=int64)
```

Classification error rate=0.13

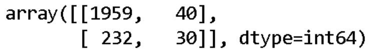
Classification report



```
              precision    recall  f1-score   support

           0       0.91      0.94      0.92       793
           1       0.45      0.35      0.39       112

    accuracy                           0.87       905
   macro avg       0.68      0.64      0.66       905
weighted avg       0.85      0.87      0.86       905
```

50/50% train/test split and KNN=6

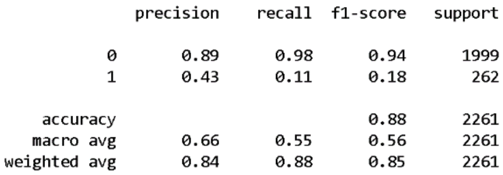
Confusion matrix



```
array([[1959,   40],
       [ 232,   30]], dtype=int64)
```

Classification error rate=0.12

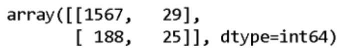
Classification report



```
              precision    recall  f1-score   support

           0       0.89      0.98      0.94      1999
           1       0.43      0.11      0.18       262

    accuracy                           0.88      2261
   macro avg       0.66      0.55      0.56      2261
weighted avg       0.84      0.88      0.85      2261
```

60/40% train/test split and KNN=6

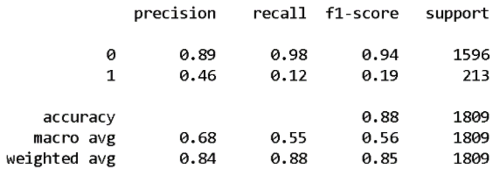
Confusion matrix



```
array([[1567,   29],
       [ 188,   25]], dtype=int64)
```

Classification error rate=0.12

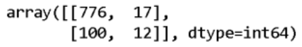
Classification report



```
              precision    recall  f1-score   support

           0       0.89      0.98      0.94      1596
           1       0.46      0.12      0.19       213

    accuracy                           0.88      1809
   macro avg       0.68      0.55      0.56      1809
weighted avg       0.84      0.88      0.85      1809
```

80/20% train/test split and KNN=6

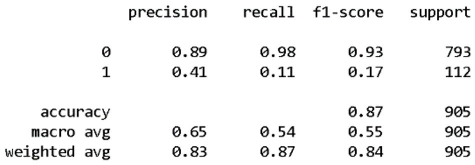
Confusion matrix



```
array([[776,  17],
       [100,  12]], dtype=int64)
```

Classification error rate=0.13

Classification report

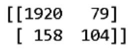


```
              precision    recall  f1-score   support

           0       0.89      0.98      0.93       793
           1       0.41      0.11      0.17       112

    accuracy                           0.87       905
   macro avg       0.65      0.54      0.55       905
weighted avg       0.83      0.87      0.84       905
```

Decision Tree Classifier

50/50% train/test split with max_depth=3

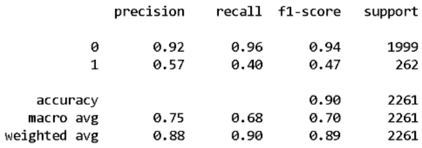
Confusion matrix



```
[[1920   79]
 [ 158  104]]
```

Classification error rate=0.10

Classification report



```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      1999
           1       0.57      0.40      0.47       262

    accuracy                           0.90      2261
   macro avg       0.75      0.68      0.70      2261
weighted avg       0.88      0.90      0.89      2261
```

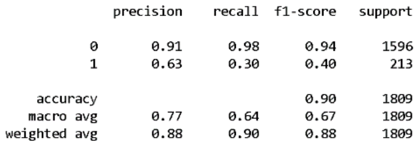
60/40% train/test split with max_depth=3

Confusion matrix

```
[[1559   37]
 [ 150   63]]
```

Classification error rate=0.10

Classification report



```
              precision    recall  f1-score   support

           0       0.91      0.98      0.94      1596
           1       0.63      0.30      0.40       213

    accuracy                           0.90      1809
   macro avg       0.77      0.64      0.67      1809
weighted avg       0.88      0.90      0.88      1809
```

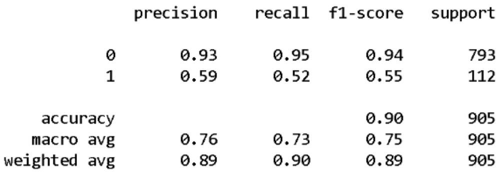
80/20% train/test split with max_depth=3

Confusion matrix

```
[[753  40]
 [ 54  58]]
```

Classification error rate=0.10

Classification report



```
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       793
           1       0.59      0.52      0.55       112

    accuracy                           0.90       905
   macro avg       0.76      0.73      0.75       905
weighted avg       0.89      0.90      0.89       905
```

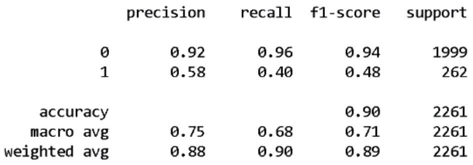
50/50% train/test split with max_depth=4

Confusion matrix

```
[[1921   78]
 [ 156  106]]
```

Classification error rate=0.10

Classification report



```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      1999
           1       0.58      0.40      0.48       262

    accuracy                           0.90      2261
   macro avg       0.75      0.68      0.71      2261
weighted avg       0.88      0.90      0.89      2261
```

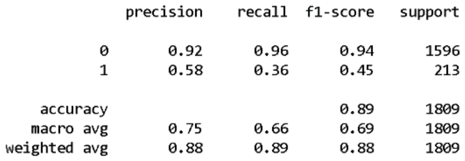
60/40% train/test split with max_depth=4

Confusion matrix

```
[[1540   56]
 [ 136   77]]
```

Classification error rate=0.11

Classification report



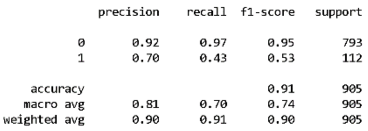
```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      1596
           1       0.58      0.36      0.45       213

    accuracy                           0.89      1809
   macro avg       0.75      0.66      0.69      1809
weighted avg       0.88      0.89      0.88      1809
```

80/20% train/test split with max_depth=4

Confusion matrix

```
[[772  21]
 [ 64  48]]
```

Classification report



```
              precision    recall  f1-score   support

           0       0.92      0.97      0.95       793
           1       0.70      0.43      0.53       112

    accuracy                           0.91       905
   macro avg       0.81      0.70      0.74       905
weighted avg       0.90      0.91      0.90       905
```

Classification error rate=0.09

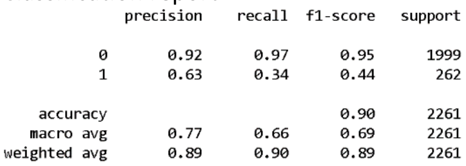
50/50% train/test split with max_depth=5

Confusion matrix

```
[[1946   53]
 [ 172   90]]
```

Classification error rate=0.10

Classification report



```
              precision    recall  f1-score   support

           0       0.92      0.97      0.95      1999
           1       0.63      0.34      0.44       262

    accuracy                           0.90      2261
   macro avg       0.77      0.66      0.69      2261
weighted avg       0.89      0.90      0.89      2261
```

Note: I could not continue with 60/40 and 80/20 test/train splits for max_depth=5 as found input variables with inconsistent numbers of samples.

Discussion

The confusion matrices above show that the True Positives plus True Negatives outweighed the False Negatives plus False Positives, indicating that our model was successful at predicting which customers would take up a new term deposit.

This lead to my classification report producing good results as shown in the table below:

Model	Split	Parameters	Acc.	Model	Split	Parameters	Acc.
KNN	50:50	n_neighbours =3	0.86	*Decision Tree*	50:50	max_depth=3	0.90
		n_neighbours =6	0.88			max_depth=4	0.90
	60:40	n_neighbours =3	0.87		60:40	max_depth=3	0.90
		n_neighbours =6	0.88			max_depth=4	0.89
	80:20	n_neighbours =3	0.87		80:20	max_depth=3	0.90
		n_neighbours =6	0.87			max_depth=4	0.91

As can be seen from the table above the KNN model yielding strong prediction results. Results where n_neighbours = 6 were stronger than n_neighbours=3, particularly when using the 50:50 or 60:40 split at 88% accuracy.

The Decision Tree model yielded better predicting results than the KNN model overall. Peak prediction outcomes where seen in the 80:20 train test split with a max_depth=4, this gave my model a 91% predication accuracy.

Note that I did not include the decision tree model with parameter max_depth=5, this returned an accuracy of 0.89. Since this is lower it is likely resulted from overfitting.

Conclusion

As can be seen from this investigation the Decision Tree Classifier model yielding better predicting results than the KNN model with an optimal predicting accuracy of 91%. This means that the Decision Tree Model with 80:20 train/test split and max depth of 4 had a 91% accuracy of predicting whether a customer will or won't take up a new term deposit in a marketing campaign. This is not surprising as Decision Tree Classifier models handle categorical variables better.

References

Source: Moro, S., Cortez, P., and Rita, P., 2014, A Data-Driven Approach to Predict the Success of Bank Telemarketing, UCI Machine Learning Repository, viewed 20 November 2021, http://archive.ics.uci.edu/ml/datasets/Bank+Marketing,