Bank Marketing Analysis

By: Andrew Stephens

Overview

- The likelihood that someone will take a loan in their lifetime is almost a guarantee in today’s society.

- What are the chances, that a customer will subscribe to a loan ?

Background

- Obrigado Bank is a Portuguese banking institution, with locations all over Portugal and Europe.

- Term deposits allow banks to hold onto a deposit for a specific amount of time, banks can invest in higher gain financial products to make a profit.

- Banks hold better chances to persuade term deposit clients into buying other products to increase their revenues.

Goals

- Create and implement a machine learning model that will classify customers that are more likely to subscribe to a loan.
- The model will have an AUC score higher than 90% and an FP score less than 10%.
- Find features any feature that may be significant with regards to the outcome of the model and the campaign.

Dataset

- Data comes from UCI Machine Learning Repository
- Instances : 45211
- Attributes/Features: 17

Dataset explored

Attribute Information:

Input variables:
bank client data:
1 - age (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 (categorical: 'basic.4y','basic.6y','basic.9y','high.school','illiterate','professional.course','university.degree','unknown')
5 - default: has credit in default? (categorical: 'no','yes','unknown')
6 - housing: has housing loan? (categorical: 'no','yes','unknown')
7 - loan: has personal loan? (categorical: 'no','yes','unknown')
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')
11 - 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:
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 (numeric; 999 means client was not previously contacted)
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')
social and economic context attributes
16 - emp.var.rate: employment variation rate - quarterly indicator (numeric)
17 - cons.price.idx: consumer price index - monthly indicator (numeric)
18 - cons.conf.idx: consumer confidence index - monthly indicator (numeric)
19 - euribor3m: euribor 3 month rate - daily indicator (numeric)
20 - nr.employed: number of employees - quarterly indicator (numeric)

Output variable (desired target):
21 - y - has the client subscribed a term deposit? (binary: 'yes','no')

Mostly categorical attributes/features with about 6 or 7 numerical features.

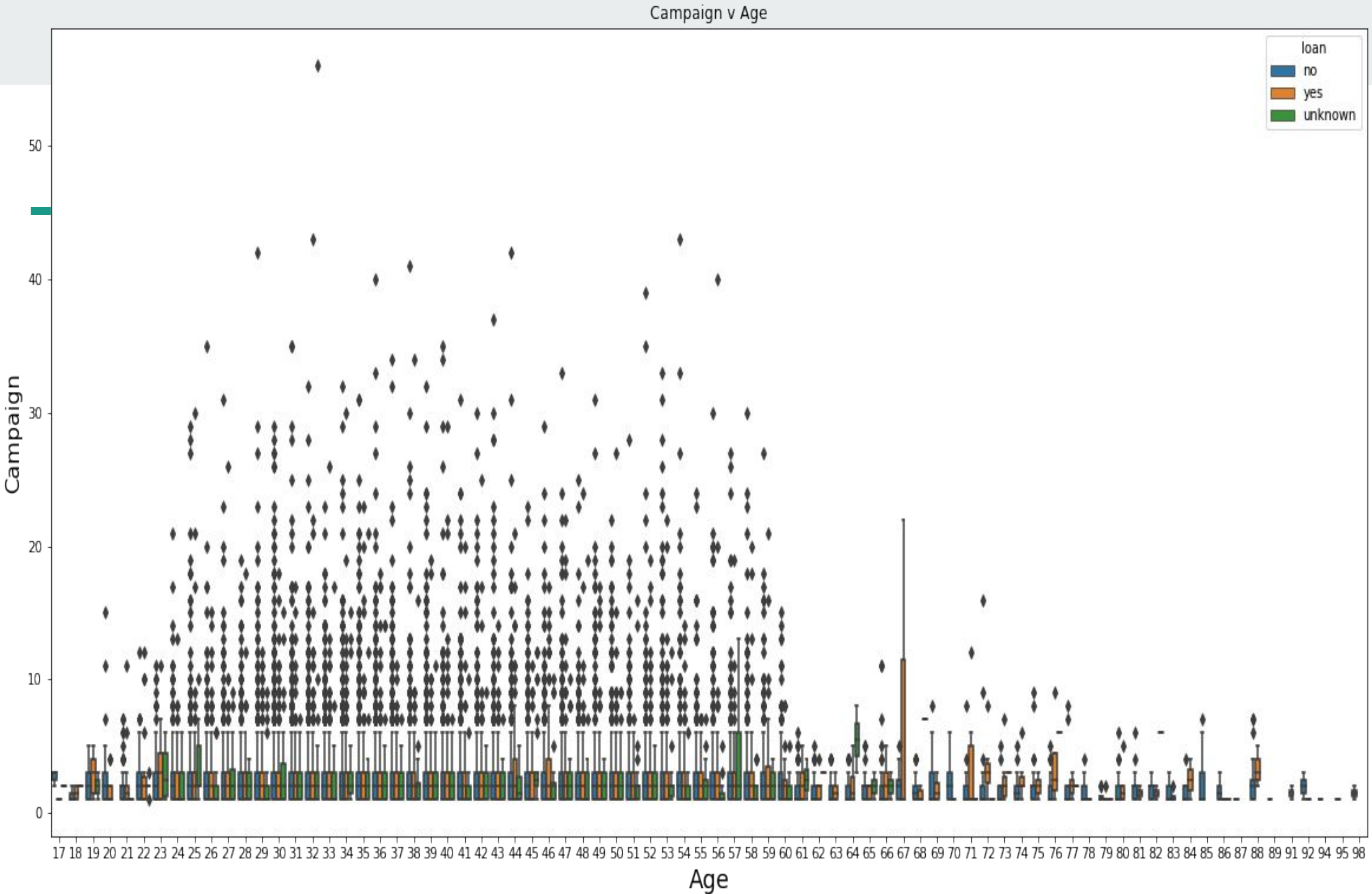
Data Wrangling

- The data came clean so there was no data wrangling needed.

Exploratory Data Analysis

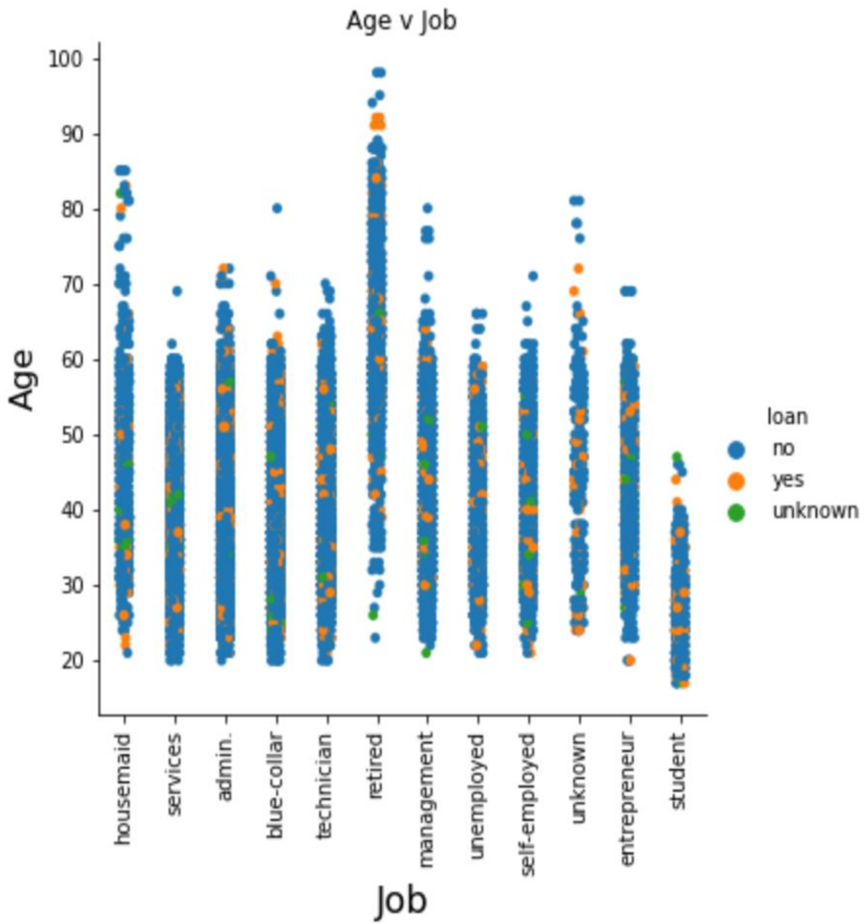
- Does being married play a factor in whether a person has a loan or not ?
- Does having a specific profession or career factor in whether someone has a loan?
- Does age play a big factor in who will take a loan ?

Age



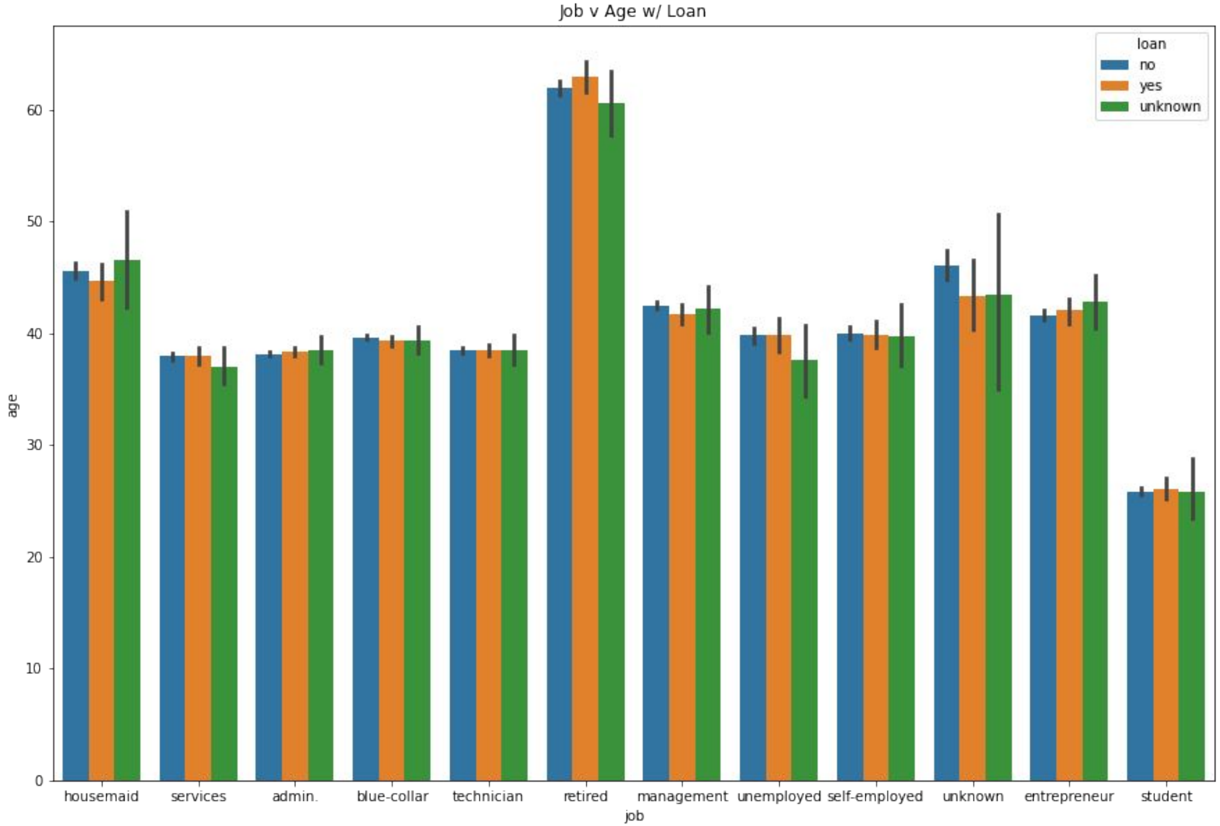
Campaign v Age
loan
no
yes
unknown
Campaign
0
10
20
30
40
50
Age
17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 91 92 94 95 98

Continued

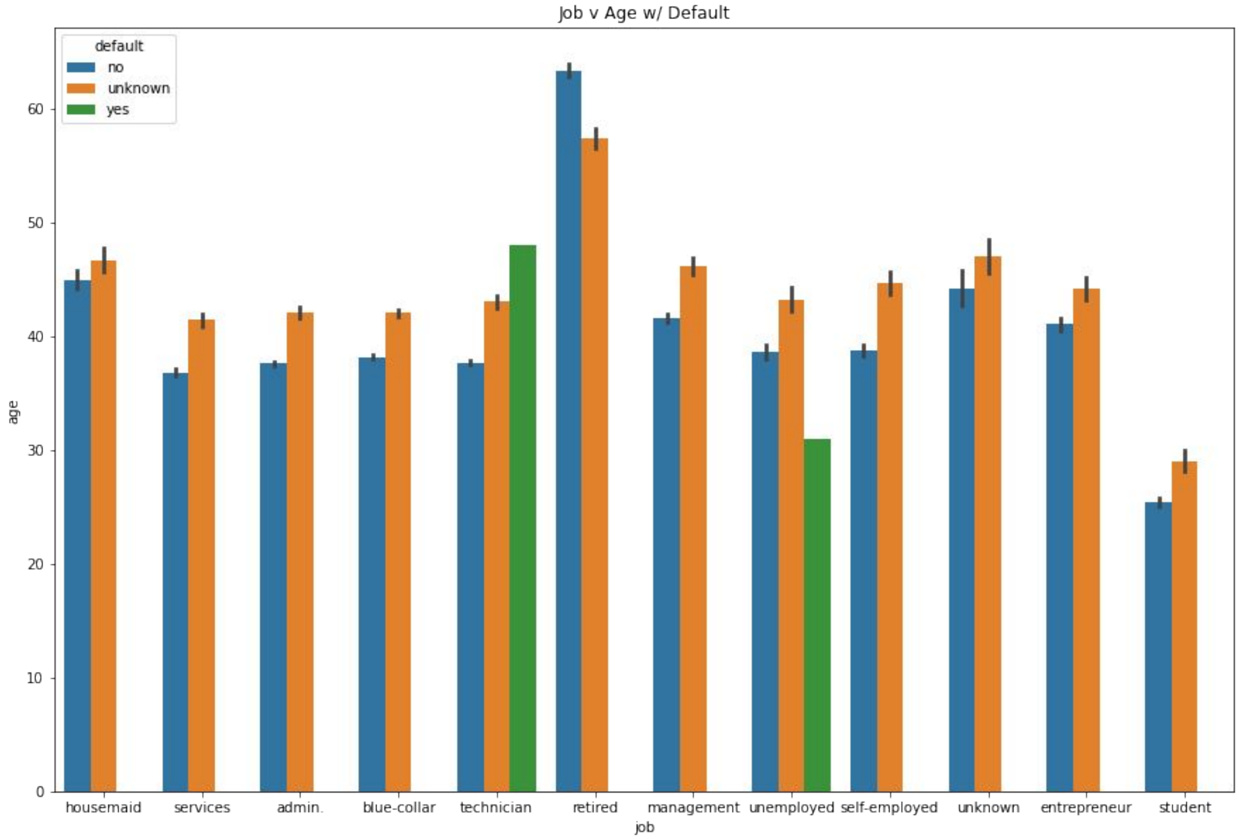


Age v Job
Age
100
90
80
70
60
50
40
30
20
Job
housemaid
services
admin.
blue-collar
technician
retired
management
unemployed
self-employed
unknown
entrepreneur
student
loan
no
yes
unknown

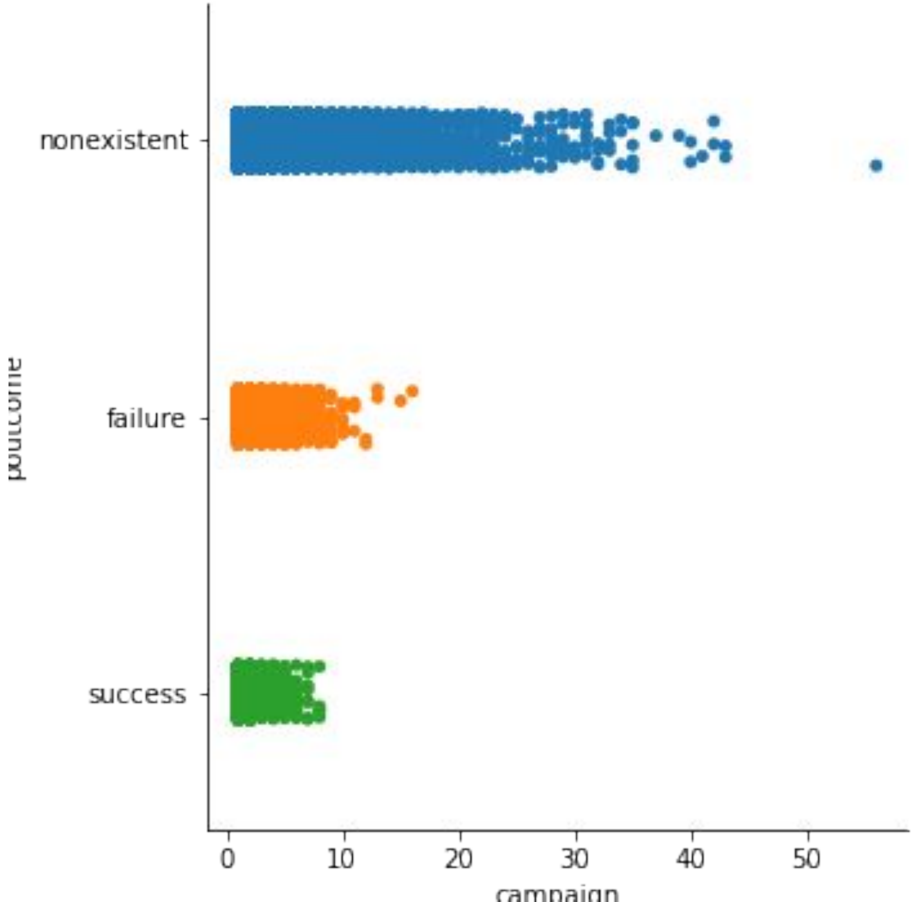
Occupation


Job v Age w/ Loan
loan
no
yes
unknown
age
0
10
20
30
40
50
60
housemaid
services
admin.
blue-collar
technician
retired
management
unemployed
self-employed
unknown
entrepreneur
student
job

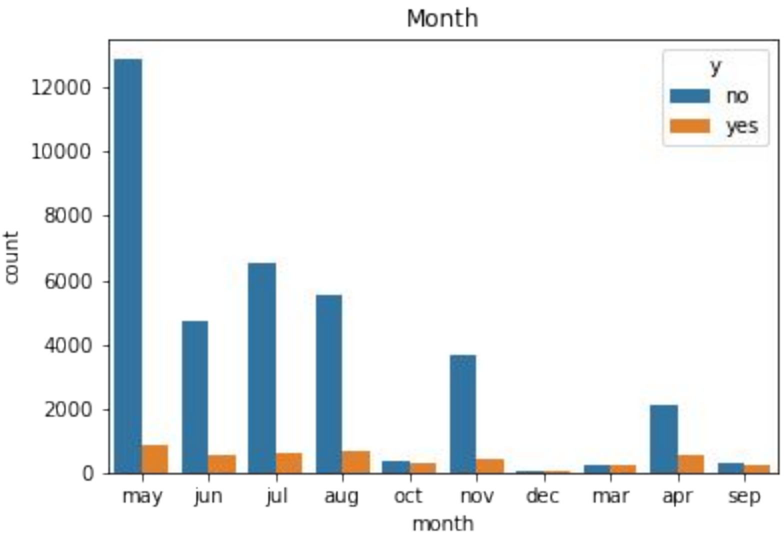
Continued


Job v Age w/ Default
default
no
unknown
yes
age
0
10
20
30
40
50
60
housemaid
services
admin.
blue-collar
technician
retired
management
unemployed
self-employed
unknown
entrepreneur
student
job

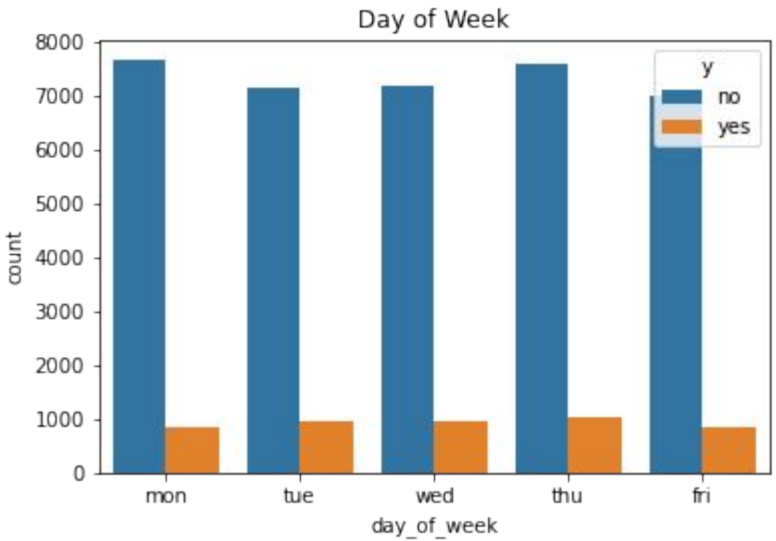
Campaign Performance



Continued

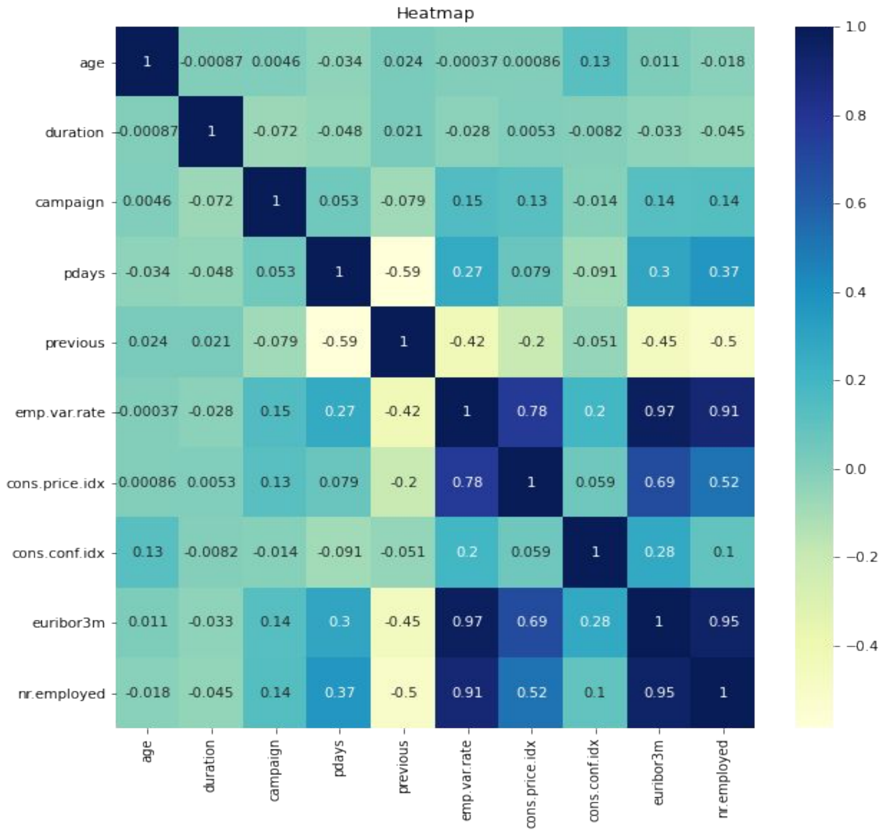


Month
y
no
yes
12000
10000
8000
6000
4000
2000
0
count
may
jun
jul
aug
oct
nov
dec
mar
apr
sep
month



Day of Week
y
no
yes
8000
7000
6000
5000
4000
3000
2000
1000
0
count
mon
tue
wed
thu
fri
day_of_week

Heatmap


Heatmap
age
duration
campaign
pdays
previous
emp.var.rate
cons.price.idx
cons.conf.idx
euribor3m
nr.employed
1 -0.00087 0.0046 -0.034 0.024 -0.00037 0.00086 0.13 0.011 -0.018
-0.00087 1 -0.072 -0.048 0.021 -0.028 0.0053 -0.0082 -0.033 -0.045
0.0046 -0.072 1 0.053 -0.079 0.15 0.13 -0.014 0.14 0.14
-0.034 -0.048 0.053 1 -0.59 0.27 0.079 -0.091 0.3 0.37
0.024 0.021 -0.079 -0.59 1 -0.42 -0.2 -0.051 -0.45 -0.5
-0.00037 -0.028 0.15 0.27 -0.42 1 0.78 0.2 0.97 0.91
0.00086 0.0053 0.13 0.079 -0.2 0.78 1 0.059 0.69 0.52
0.13 -0.0082 -0.014 -0.091 -0.051 0.2 0.059 1 0.28 0.1
0.011 -0.033 0.14 0.3 -0.45 0.97 0.69 0.28 1 0.95
-0.018 -0.045 0.14 0.37 -0.5 0.91 0.52 0.1 0.95 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

Feature Engineering

- First, we removed any features with correlation > 80%

- Next we transformed our categorical data using Onehotencoding

Handling Categorical Data

Because most of the variables we are working with are categorical we will be using 1 hot encoding to satisfy the mathmatical equations that our model will be solving for

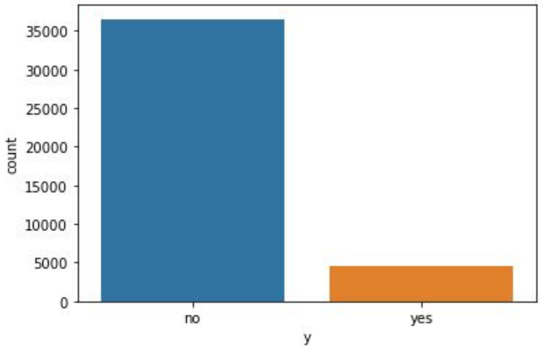
In [21]:
```python
columns=df.select_dtypes(include=[object]).columns
df=pd.concat([df,pd.get_dummies(df[columns])],axis=1)
df=df.drop(['job','marital','education','default','housing','loan','contact','month','day_of_week','poutcome'],axis=
df.info()
df.head()
```

```
Int64Index: 41188 entries, 0 to 41187
Data columns (total 63 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   age                 41188 non-null  int64
 1   campaign            41188 non-null  int64
 2   pdays               41188 non-null  int64
 3   previous            41188 non-null  int64
 4   cons.price.idx      41188 non-null  float64
 5   cons.conf.idx       41188 non-null  float64
 6   nr.employed         41188 non-null  float64
 7   y                   41188 non-null  object
 8   job_admin.          41188 non-null  uint8
 9   job_blue-collar     41188 non-null  uint8
 10  job_entrepreneur    41188 non-null  uint8
 11  job_housemaid       41188 non-null  uint8
 12  job_management      41188 non-null  uint8
 13  job_retired         41188 non-null  uint8
```

Continued

Taking care of imbalanced data

Before



After

```
Before oversampling:  Counter({'no': 36548, 'yes': 4640})
After oversampling:  Counter({'no': 36548, 'yes': 36548})
```

Machine Learning

Models we are using are :

- KNN
- SVM
- Logistic Regression
- Random Forest
- XGBoost

KNN

Code snippet

```
In [15]: # grid searh to choose the best (combination of) hyperparameters
         gs_knn=GridSearchCV(estimator= pipe_knn,
                             param_grid={'kneighborsclassifier__n_neighbors':[5,6,7]},
                             scoring='accuracy',
                             cv=5)
```

```
In [16]: # nested cross validation combining grid search (inner loop) and k-fold cv (outter loop)
         gs_knn_scores = cross_val_score(gs_knn, X=X_train, y=y_train, cv=5,scoring='accuracy', n_jobs=-1)
```

SVM

Code snippet

```
# grid searh to choose the best (combination of) hyperparameters
r=[0.1,1]
pg_svm=[ {'svc__C':r, 'svc__gamma':r}]

gs_svm=GridSearchCV(estimator= pipe_svm,
                    param_grid= pg_svm,
                    scoring='accuracy',
                    cv=3)

# nested cross validation combining grid search (inner loop) and k-fold cv (outter loop)
gs_svm_scores = cross_val_score(gs_svm, X=X_train, y=y_train, cv=3,scoring='accuracy', n_jobs=-1)

# fit, and fit with best estimator
gs_svm.fit(X_train, y_train)
gs_svm_best=gs_svm.best_estimator_
gs_svm_best.fit(X_train, y_train)
```

Logistic Regression

Code snippet

```
In [23]: lr = LogisticRegressionCV(random_state=1, max_iter =1000)
         lr.fit(X_train, y_train)
         preds = lr.predict(X_test)

In [24]: test_accuracy = accuracy_score(y_test,preds)
         print(test_accuracy)

         0.9379616963064296
```

Random Forest

Code snippet

```
rf= RandomForestClassifier(random_state=1)

# grid searh to choose the best (combination of) hyperparameters
pg_rf={'n_estimators': [100,250,400],'max_depth': [10,20,40,50,60]}

gs_rf=GridSearchCV(estimator= rf,
               param_grid= pg_rf,
               scoring='accuracy',
               cv=5)

# nested cross validation combining grid search (inner loop) and k-fold cv (outter loop)
gs_rf_scores = cross_val_score(gs_rf, X=X_train, y=y_train, cv=5,scoring='accuracy', n_jobs=-1)
```

XGboost

Code snippet

```
# estimator
xb= xgb.XGBClassifier(random_state=1)

# grid searh to choose the best (combination of) hyperparameters
pg_xb={'n_estimators':[150,200,400], 'max_depth':[5,10,20,30],'min_child_weight':[.25,.5,.75]}

gs_xb=GridSearchCV(estimator= xb,
                   param_grid= pg_xb,
                   scoring='accuracy',
                   cv=5)

# nested cross validation combining grid search (inner loop) and k-fold cv (outter loop)
gs_xb_scores = cross_val_score(gs_xb, X=X_train, y=y_train, cv=5,scoring='accuracy', n_jobs=-1)

# fit, and fit with best estimator
gs_xb.fit(X_train, y_train)
gs_xb_best=gs_xb.best_estimator_
gs_xb_best.fit(X_train, y_train)
```

Oversampling vs Undersampling Comparison

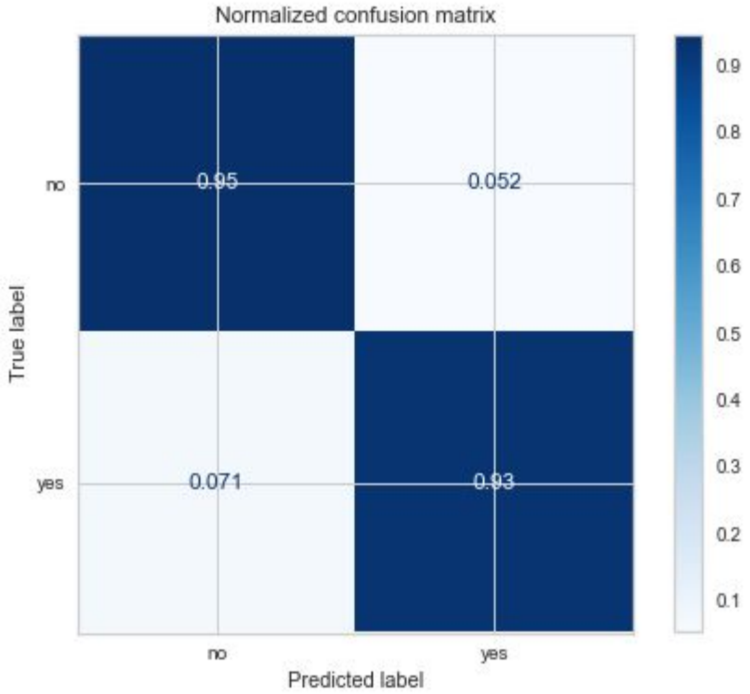
Oversampling

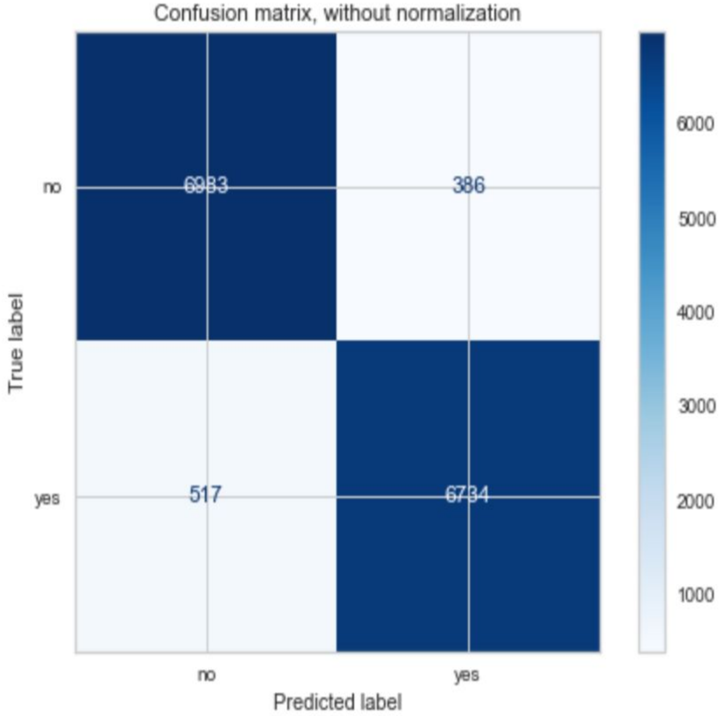
Model	FP%	FN%	AUC%
KNN	7.9	7.3	96
LR	11	1.9	97
SVM	11	2.3	97
RF	7.1	5.2	98
XGBoost	9.2	2.9	97

Undersampling

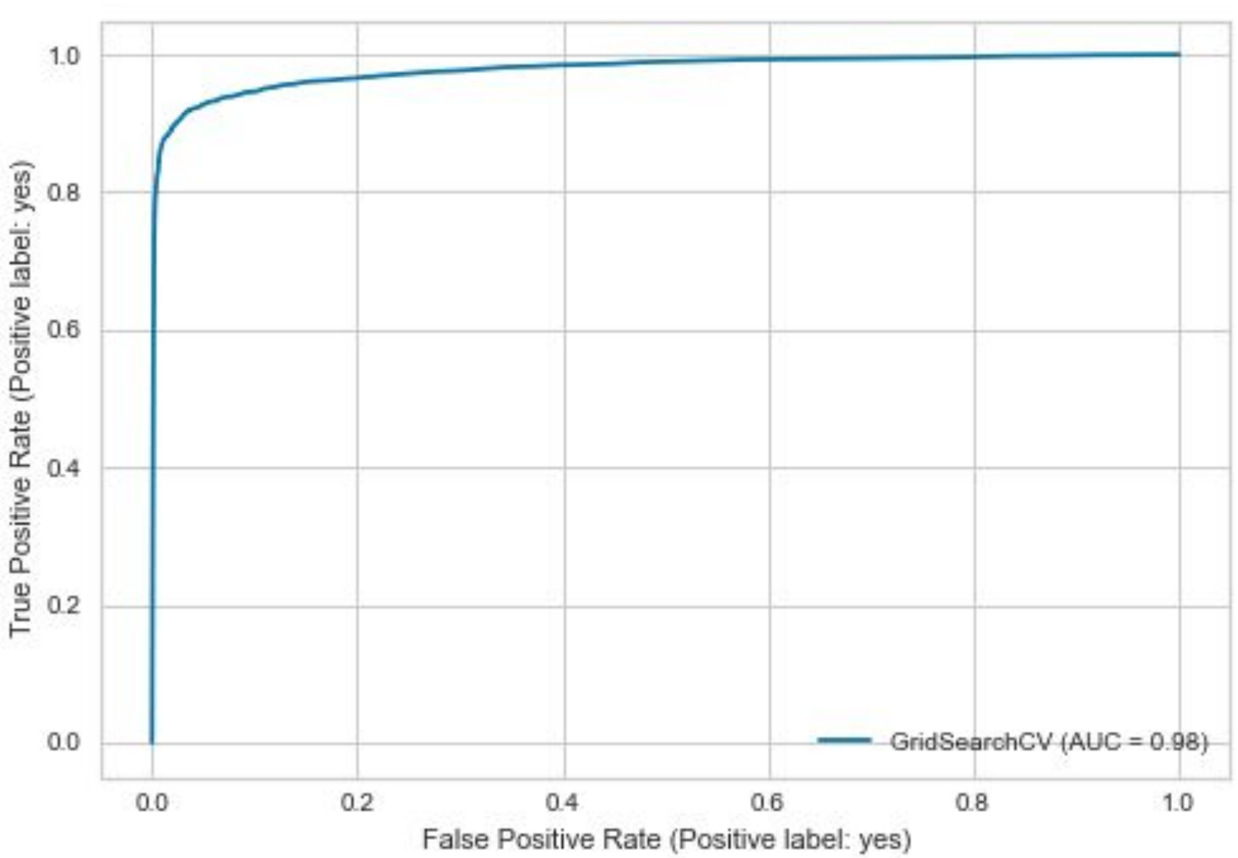
Model	FP%	FN%	AUC%
KNN	37	21	75
LR	38	14	80
SVM	29	32	74
RF	37	13	80
XGBoost	36	18	78

Random Forest Confusion Matrix



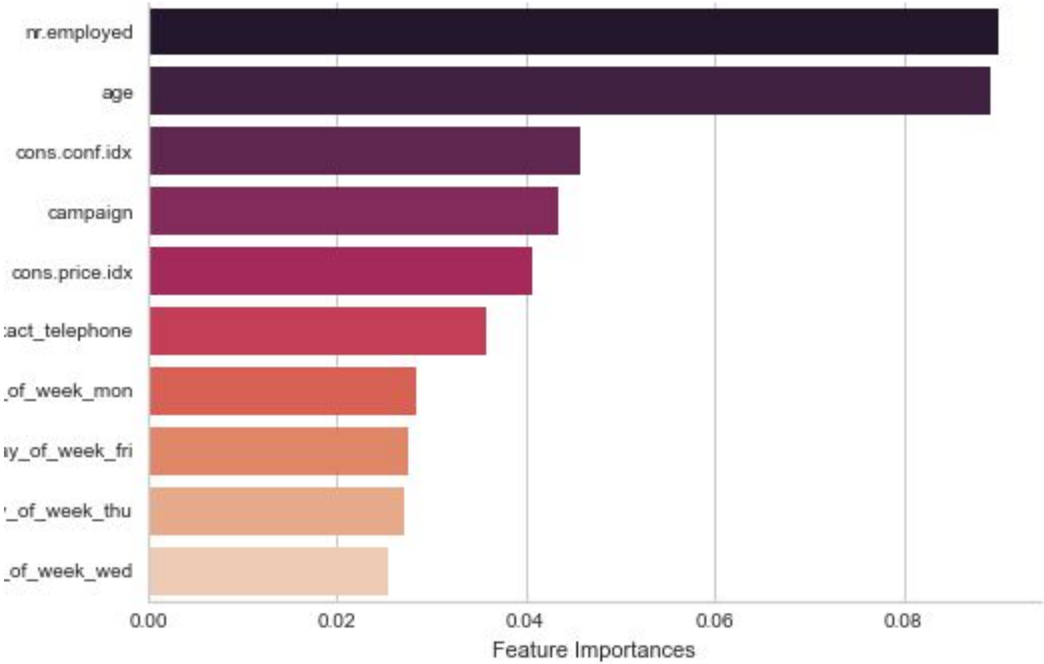


LR ROC



1.0
0.8
0.6
0.4
0.2
0.0
True Positive Rate (Positive label: yes)
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate (Positive label: yes)
GridSearchCV (AUC = 0.98)

Feature Importance



Recommendations

- Focus on calling customers Tues,Wed,Thur

- Drop any customer that hasn't subscribed in at least 15 days or 8 call attempts.

- Keep employees working and happy, any drop in workforce affects the campaign.

- Segment customers by age and incorporate lifestyle segmentation

- Focus on targeting entrepreneurs.Target the management occupation to help drive volume.

Takeaways

- April-August produces the highest results for the campaign.
- Age is a factor as we can see from the feature importances chart and heatmap
- Any chance of success will happen within 10 days of direct marketing to a customer; after 10 days there were no successful attempts.
- Employees carry high significance, with regards to the feature importances chart, with how many employees are there to make the calls

Future work

- Include time
- Include more personal data such as loan balance, account balance, debt to income ratio
- Include geographic location

Thank You