Bank Marketing

Neal Drakos

Dataset - Background & Context

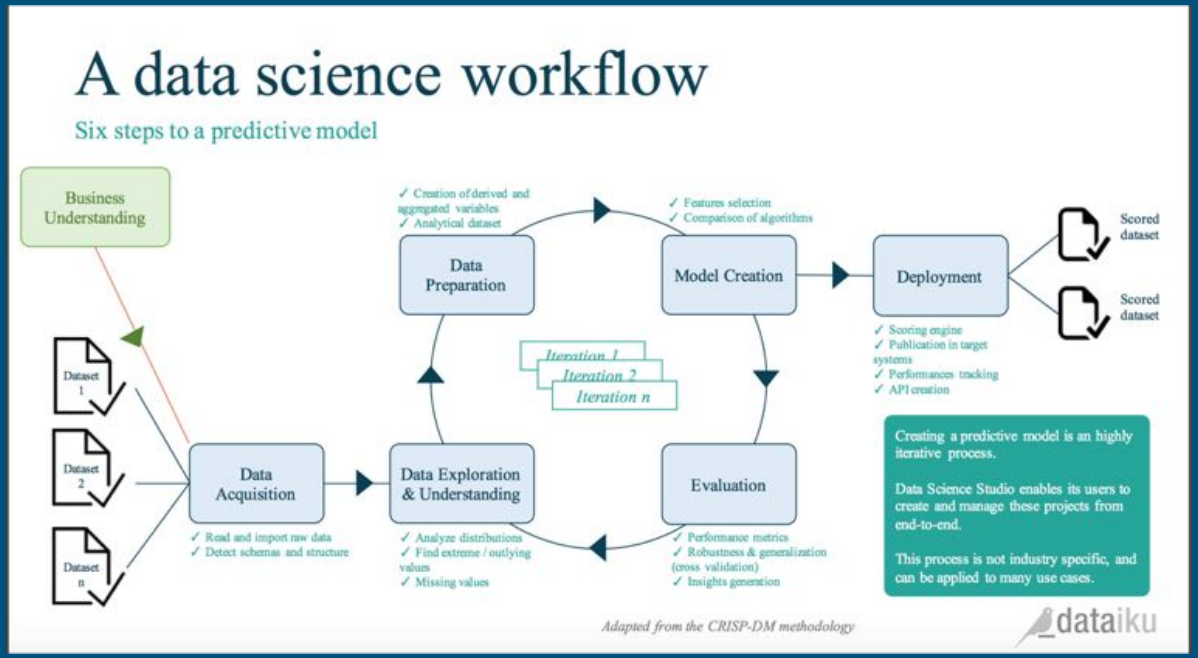
-The results of a direct marketing campaign of a Portuguese banking institution.

-The campaigns accessed whether the client would subscribe to the product (bank term deposits). Based on a ('yes') or ('no') response to a term deposit subscription with the bank.

- A Binary Classification problem.

Overview & Approach

- Business Understanding
- Data Acquisition
- Data Exploration
- Data Preparation
- Model Creation
- Evaluation



Business Understanding

Target Variable: Term Deposit

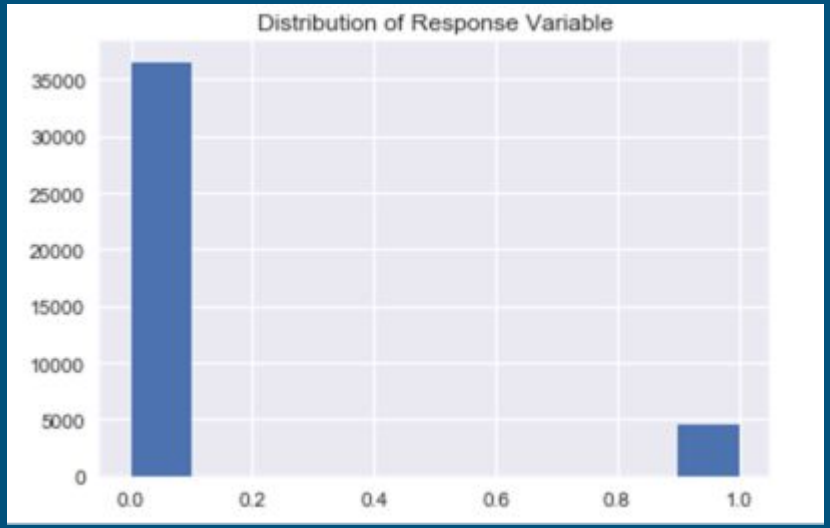
A term deposit (or CD) is a cash investment held at a financial institution. Your money is invested for an agreed rate of interest over a fixed amount of time, or (term)

Term deposits are considered an extremely safe investment and are therefore very appealing to conservative, low-risk investors

The marketing campaigns were based on phone calls.

Data Exploration

Inspecting the target variable:



NO	YES
36,548	4,640

Data Exploration

-Dimensions: 41,188 observations made up of 21 features.

#bank client data	#Last contact data	#Other attributes	#Social&Economic
age	type of contact	num_of_contacts	Employment var_rate
job	month	days_passed	consumer_price_indx
marital status	day_of_week	previous_campaigns	consumer_confidence_indx
education	duration	previous_outcomes	3m_euribor_rate
in credit default?			nr_employed
housing loan?			

Data Exploration

-Check distributions

-No Missingness?

-Imputing 'Unknown' missingness

```
# loan
#7 - loan: has personal loan? (categorical: 'no','yes','unknown')

print(data.loan.value_counts())
print("NAs for loan : " + str(data.loan.isnull().values.sum()))

data.loc[data["loan"] == "unknown", "loan"] = "no"
```

```
no         33950
yes         6248
unknown      990
Name: loan, dtype: int64
NAs for loan : 0
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41188 entries, 0 to 41187
Data columns (total 21 columns):
age               41188 non-null int64
job               41188 non-null object
marital           41188 non-null object
education         41188 non-null object
default           41188 non-null object
housing           41188 non-null object
loan              41188 non-null object
contact           41188 non-null object
month             41188 non-null object
day_of_week       41188 non-null object
duration          41188 non-null int64
campaign          41188 non-null int64
pdays             41188 non-null int64
previous          41188 non-null int64
poutcome          41188 non-null object
emp.var.rate      41188 non-null float64
cons.price.idx    41188 non-null float64
cons.conf.idx     41188 non-null float64
euribor3m         41188 non-null float64
nr.employed       41188 non-null float64
y                 41188 non-null object
dtypes: float64(5), int64(5), object(11)
memory usage: 6.6+ MB
```

Data Preparation

-Drop features

-One hot encoding for categorical features

-Standardize & scale with standard scalar

```
# default
#5 - default: has credit in default? (categorical: 'no','yes','unknown')

print(data.default.value_counts())
print("NAs for default : " + str(data.default.isnull().values.sum()))
```

```
no         32588
unknown     8597
yes            3
Name: default, dtype: int64
NAs for default : 0
```

```
data = data.drop(["default"], axis = 1) #not enough info on this, drop.
```

```
# Separate out the numerical features (minus the target) and categorical features
y = data.y
categorical_features = data.select_dtypes(include = ["object"]).columns
numerical_features = data.select_dtypes(exclude = ["object"]).columns
numerical_features = numerical_features.drop("y")
print("Numerical features : " + str(len(numerical_features)))
print("Categorical features : " + str(len(categorical_features)))
data_num = data[numerical_features]
data_cat = data[categorical_features]
```

```
Numerical features : 10
Categorical features : 9
```

Model Creation

-Type of problem: Binary Classification

- Models used:
    - Logistic Regression
    - Random Forest
    - Gradient Boosting

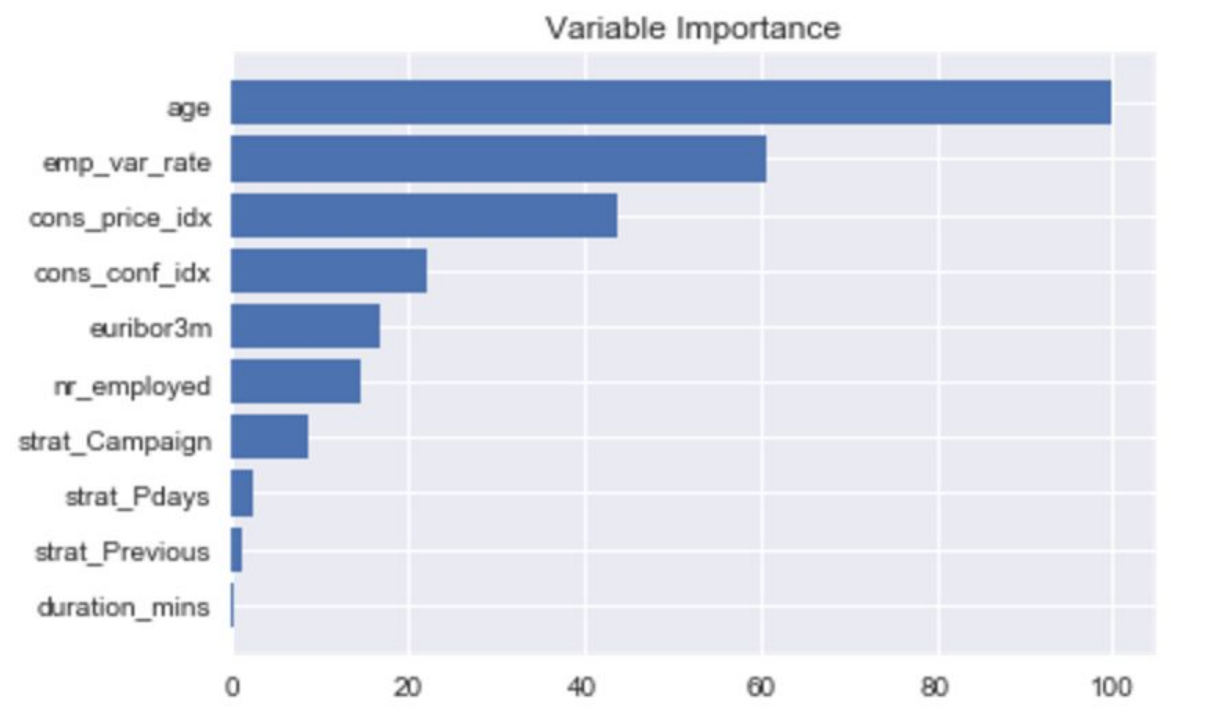
Model Creation

Methodology:

Assess training / test errors

Model Selection using grid search with best params

Model tuning with feature importance/selection



Evaluation - Logistic Regression

Training Classification Report:

```
print (classification_report(y_train, lr_pred_train, target_names = target_names))
```

	precision	recall	f1-score	support
class No	0.93	0.97	0.95	27409
class Yes	0.66	0.41	0.51	3482
avg / total	0.90	0.91	0.90	30891

Test Classification Report:

```
print (classification_report(y_test, lr_pred_Test, target_names = target_names))
```

	precision	recall	f1-score	support
class No	0.93	0.97	0.95	9139
class Yes	0.67	0.44	0.53	1158
avg / total	0.90	0.91	0.90	10297

Evaluation - Random Forest

Training Classification Report:

```
print (classification_report(y_train,rf_pred_train, target_names=target_names))
```

	precision	recall	f1-score	support
Class_No	0.94	0.96	0.95	27410
Class_Yes	0.61	0.50	0.55	3481
avg / total	0.90	0.91	0.90	30891

Test Classification Report:

```
print (classification_report(y_test,rf_pred_Test, target_names = target_names))
```

	precision	recall	f1-score	support
Class_No	0.93	0.96	0.95	9138
Class_Yes	0.58	0.47	0.52	1159
avg / total	0.89	0.90	0.90	10297

Evaluation- Gradient Boosting

Training Classification Report:

```
print (classification_report(y_train,gbt_pred_train, target_names = target_names))
```

	precision	recall	f1-score	support
class No	0.95	0.97	0.96	27369
class Yes	0.73	0.60	0.66	3522
avg / total	0.93	0.93	0.93	30891

Testing Classification Report:

```
print (classification_report(y_test,gbt_pred_Test, target_names = target_names))
```

	precision	recall	f1-score	support
class No	0.95	0.96	0.95	9179
class Yes	0.65	0.55	0.60	1118
avg / total	0.91	0.92	0.92	10297