Predictive Models

This repository uses the insights we captured from the data to make a predictive model. We use several models and compare the results obtained in each case. The first one displayed below is the **Random Forest** model. Then **SVM** approach is used to compare results

1. Random Forest

This cell contains all the dependencies needed

In [25]:

```
import numpy as np #Matrix-Maths
import pandas as pd #DataFrame
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.svm import SVC
```

In [2]:

```
data = pd.read_csv('Data/bank-additional-full.csv', delimiter=';')
```

In [3]:

```
data.head()
```

Out[3]:

	age	job	marital	education	default	housing	loan	contact	month	day_of_w
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	n
1	57	services	married	high.school	unknown	no	no	telephone	may	n
2	37	services	married	high.school	no	yes	no	telephone	may	n
3	40	admin.	married	basic.6y	no	no	no	telephone	may	n
4	56	services	married	high.school	no	no	yes	telephone	may	n

5 rows × 21 columns

In [4]:

```
data.shape
```

Out[4]:

```
(41188, 21)
```

In [5]:

```
for col in data.columns:
    if(data[col].dtype == object):
        data[col] = data[col].astype('category')
```

In [6]:

```
data['education'] = data['education'].cat.codes
data['job'] = data['job'].cat.codes
data['default'] = data['default'].cat.codes
data['loan'] = data['loan'].cat.codes
data['day_of_week'] = data['day_of_week'].cat.codes
data['y'] = data['y'].cat.codes
data['poutcome'] = data['poutcome'].cat.codes
data['contact'] = data['contact'].cat.codes
data['marital'] = data['marital'].cat.codes
data['month'] = data['month'].cat.codes
```

In [7]:

```
data.shape
```

Out[7]:

```
(41188, 21)
```

In [8]:

```
data.head()
```

Out[8]:

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	ca
0	56	3	1	0	0	no	0	1	6	1	...	
1	57	7	1	3	1	no	0	1	6	1	...	
2	37	7	1	3	0	yes	0	1	6	1	...	
3	40	0	1	1	0	no	0	1	6	1	...	
4	56	7	1	3	0	no	2	1	6	1	...	

5 rows × 21 columns

In [11]:

```
cols = list(data.columns)
```

In [15]:

```
features = ['job',
             'education',
            'default',
            'loan',
            'month',
            'day_of_week',
            'pdays',
            'previous',
            'emp.var.rate',
            'cons.price.idx',
            'cons.conf.idx',
            'euribor3m',
             'poutcome',
             'contact',
             'marital',
            'y']
```

In [16]:

```
data = data[features]
```

In [17]:

```
data.shape
```

Out[17]:

```
(41188, 16)
```

In [19]:

```
X = data.values[:, :15]
Y = data.values[:, 15]
```

In [21]:

```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3)
```

In [22]:

```
ran_for = RandomForestClassifier()
```

In [23]:

```
model = ran_for.fit(X_train, y_train)
```

In [24]:

```
model.score(X_test, y_test)
```

Out[24]:

```
0.8863801893663511
```

2. Support Vector Machine

In [30]:

```
clf_svm = SVC(kernel='rbf', C=100)
```

In [31]:

```
clf_svm.fit(X_train, y_train)
```

Out[31]:

```
SVC(C=100, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape=None, degree=3, gamma='auto', kernel='rbf'
,
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

In [32]:

```
clf_svm.score(X_test, y_test)
```

Out[32]:

```
0.8896172210083354
```

In [ ]:

```

```

Predictive Models

This repository uses the insights we captured from the data to make a predictive model. We use several models and compare the results obtained in each case. The first one displayed below is the **Random Forest** model. Then **SVM** approach is used to compare results

1. Random Forest

This cell contains all the dependencies needed

In [25]:

In [2]:

In [3]:

Out[3]:

	age	job	marital	education	default	housing	loan	contact	month	day_of_w
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	n
1	57	services	married	high.school	unknown	no	no	telephone	may	n
2	37	services	married	high.school	no	yes	no	telephone	may	n
3	40	admin.	married	basic.6y	no	no	no	telephone	may	n
4	56	services	married	high.school	no	no	yes	telephone	may	n

5 rows × 21 columns

In [4]:

Out[4]:

```
(41188, 21)
```

In [5]:

In [6]:

In [7]:

Out[7]:

```
(41188, 21)
```

In [8]:

Out[8]:

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	ca
0	56	3	1	0	0	no	0	1	6	1	...	
1	57	7	1	3	1	no	0	1	6	1	...	
2	37	7	1	3	0	yes	0	1	6	1	...	
3	40	0	1	1	0	no	0	1	6	1	...	
4	56	7	1	3	0	no	2	1	6	1	...	

5 rows × 21 columns

In [11]:

In [15]:

In [16]:

In [17]:

Out[17]:

```
(41188, 16)
```

In [19]:

In [21]:

In [22]:

In [23]:

In [24]:

Out[24]:

```
0.8863801893663511
```

2. Support Vector Machine

In [30]:

In [31]:

Out[31]:

```
SVC(C=100, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape=None, degree=3, gamma='auto', kernel='rbf'
,
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

In [32]:

Out[32]:

0.8896172210083354

In [ ]: