Machine Learning Case Study

Ummugulsum Arslan

In [52]:
```
# Read the data and convert data to matrix
import csv
import numpy as np
from sklearn.feature_extraction import DictVectorizer
from sklearn.model_selection import train_test_split
import numpy as np
np.random.seed(42)
import random
random.seed(42)

X_dicts = []
y = []
myfile = open('churn.csv')
iCSV = csv.reader(myfile, delimiter = ',')
header = next(iCSV)

for row in iCSV:
    y.append(int(row[-1]))
    new_dict = {}
    new_dict[header[3]] = float(row[3])
    new_dict[header[4]] = row[4]
    new_dict[header[5]] = row[5]
    for h, i in zip(header[6:-1], row[6:-1]):
        new_dict[h] = float(i)
    X_dicts.append(new_dict)
myfile.close()
print(len(y), len(X_dicts))

vec = DictVectorizer(sparse=False)

X = vec.fit_transform(X_dicts)
y=np.array(y)
#print(X.shape)
#print(vec.feature_names_)

# Explore Data

stats = X.mean(axis=0)

for f,x in zip(vec.feature_names_, stats):
    print(f,x)
print()
print(y.mean())
```

```
10000 10000
Age 38.9218
Balance 76485.88928799961
CreditScore 650.5288
EstimatedSalary 100090.2398809998
Gender=Female 0.4543
Gender=Male 0.5457
Geography=France 0.5014
Geography=Germany 0.2509
Geography=Spain 0.2477
HasCrCard 0.7055
IsActiveMember 0.5151
NumOfProducts 1.5302
Tenure 5.0128

0.2037
```

In [53]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.2, rando
print(X_train.shape, X_test.shape)
```

```
(8000, 13) (2000, 13)
```

In [54]:
```
# Train model on Data
from sklearn.svm import LinearSVC
from sklearn.metrics import f1_score
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier

params = {'C': [0.001,0.01,0.1,1.]}

svc = LinearSVC()

clf_svm = GridSearchCV(svc, params, scoring='f1', cv=2)

clf_svm.fit(X_train, y_train)
```

```
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
/Users/gulsum/opt/anaconda3/lib/python3.9/site-packages/sklearn/svm/_base.py:1
225: ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
```

Out[54]:

```
▸ GridSearchCV
▸ estimator: LinearSVC
    ▸ LinearSVC
```

In [55]:

```
clf_svm.best_score_
```

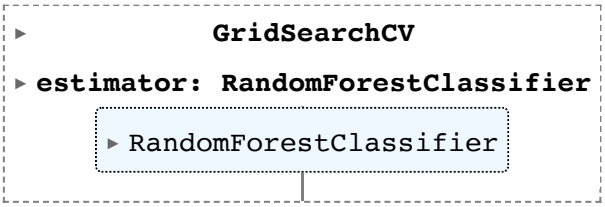
Out[55]:

```
0.35301370643722546
```

In [56]:

```
# lets try another model

params = {'n_estimators': [10,100,200,300,400]}

svc = RandomForestClassifier()

rf_clf = GridSearchCV(svc, params, scoring='f1', cv=2)

rf_clf.fit(X_train, y_train)
```

Out[56]:



In [57]:
```
rf_clf.best_score_
```

Out[57]:
```
0.5701264802131772
```

In [58]:
```
rf_clf.best_params_
```

Out[58]:
```
{'n_estimators': 400}
```

In [59]:
```
# Evaluate model
```

In [60]:
```
rf_preds = rf_clf.predict(X_test)
svm_preds = clf_svm.predict(X_test)

rf_f1 = f1_score(y_test,rf_preds)
svm_f1 = f1_score(y_test, svm_preds)
print("RF: {:.4f}".format(rf_f1))
print("RF: {:.4f}".format(svm_f1))
```

```
RF: 0.5853
RF: 0.3515
```

In [ ]: