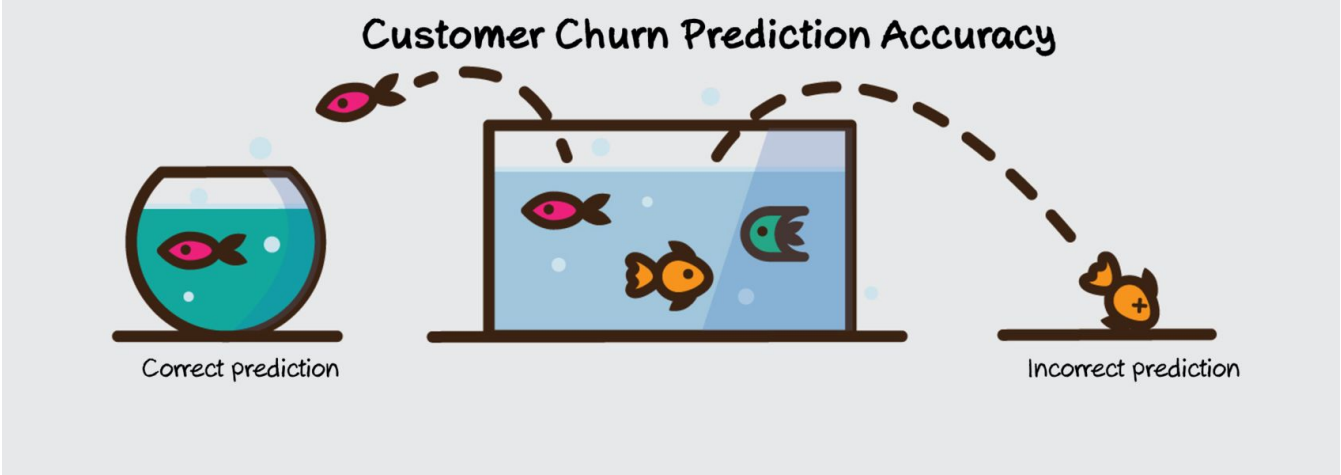
DMW Mini Project

Bank Customer Analysis using Churn Modelling

Yash Nadkarni (BC128)

What is Churn Modelling?

Churn modelling is a mathematical representation of how churn impacts a business. A predictive churn model extrapolates on this data to show future potential churn rates.



Our Approach

Data Collection

Imported python libraries like scikit-learn, numpy, pandas, matplotlib and used the dataset - Churn_modelling.csv from Kaggle

Data Pre-processing

Removed the outliers, scaled the data using MinMaxScaler function and implemented Label Encoding

Model Training

Trained 4 different classifiers - Naive Bayes, Support Vector Machines, KNN and Random Forest

Evaluation

Evaluated each classifier by its confusion matrix and accuracy score

Churn Modelling Dataset

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	619	France	Female	42	2	0.00	1	1	1	101348.88	1
1	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0
...	...	...	...	...	...	...	...	...	...	...	...
9995	771	France	Male	39	5	0.00	2	1	0	96270.64	0
9996	516	France	Male	35	10	57369.61	1	1	1	101699.77	0
9997	709	France	Female	36	7	0.00	1	0	1	42085.58	1
9998	772	Germany	Male	42	3	75075.31	2	1	0	92888.52	1
9999	792	France	Female	28	4	130142.79	1	1	0	38190.78	0

10000 rows × 11 columns

Implementation of Naive Bayes

```
[21] from sklearn.naive_bayes import GaussianNB
     classifier = GaussianNB()
     classifier.fit(X_train, y_train)
```

```
GaussianNB(priors=None, var_smoothing=1e-09)
```

```
y_pred = classifier.predict(X_test)
cm_naive_bayes = confusion_matrix(y_test, y_pred)
acc_score_naive_bayes=accuracy_score(y_test,y_pred)
print("The confusion matrix :\n",cm_naive_bayes)
print("The accuracy score is : ",acc_score_naive_bayes)
```

```
The confusion matrix :
 [[1799  192]
 [ 293  216]]
The accuracy score is :  0.806
```

Implementation of Support Vector Machines

```
[8]  from sklearn.svm import SVC
     classifier = SVC(kernel = 'linear', random_state = 0)
     classifier.fit(X_train, y_train)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
    max_iter=-1, probability=False, random_state=0, shrinking=True, tol=0.001,
    verbose=False)
```

```
[9]  y_pred = classifier.predict(X_test)
     cm_svm = confusion_matrix(y_test, y_pred)
     acc_score_svm=accuracy_score(y_test,y_pred)
     print("The confusion matrix :\n",cm_svm)
     print("The accuracy score is : ",acc_score_svm)
```

```
The confusion matrix :
 [[1991    0]
 [ 509    0]]
The accuracy score is :  0.7964
```

Implementation of K Nearest Neighbours

```
[10] from sklearn.neighbors import KNeighborsClassifier
     classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
     classifier.fit(X_train, y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

```
[11] y_pred = classifier.predict(X_test)
     cm_knn = confusion_matrix(y_test, y_pred)
     acc_score_knn=accuracy_score(y_test,y_pred)
     print("The confusion matrix :\n",cm_knn)
     print("The accuracy score is : ",acc_score_knn)
```

```
The confusion matrix :
 [[1859  132]
 [ 307  202]]
The accuracy score is :  0.8244
```

Implementation of Random Forest

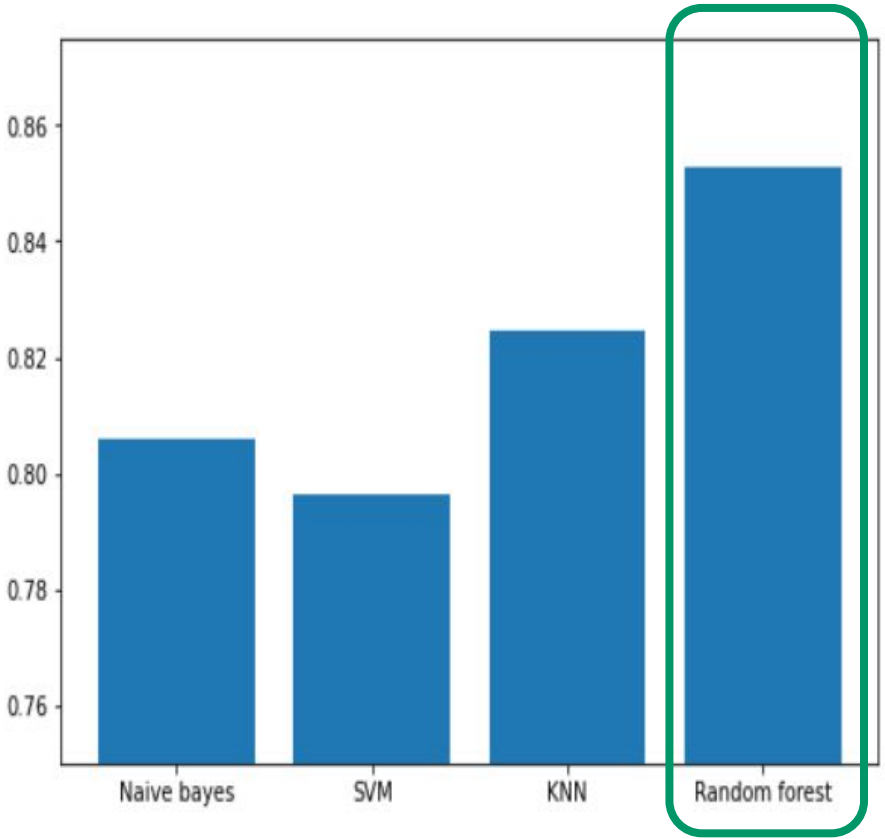
```
[12] from sklearn.ensemble import RandomForestClassifier
     classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
     classifier.fit(X_train, y_train)
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='entropy', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=10,
                       n_jobs=None, oob_score=False, random_state=0, verbose=0,
                       warm_start=False)
```

```
[13] y_pred = classifier.predict(X_test)
     cm_random_forest = confusion_matrix(y_test, y_pred)
     acc_score_random_forest=accuracy_score(y_test,y_pred)
     print("The confusion matrix :\n",cm_random_forest)
     print("The accuracy score is : ",acc_score_random_forest)
```

```
The confusion matrix :
 [[1903   88]
 [ 280  229]]
The accuracy score is :  0.8528
```

Evaluation of Classifiers



Thank You