Udacity Machine learning Engineer NanoDegree (Bertelsmann/Arvato project report)

Project overview

This project is Udacity Machine Learning Nanodegree capstone program, Create a Customer SegmentationReport for Arvato Financial Services.
A mail-order sales company in Germany, Arvato Financial Services, is interested in identifying the data to acquire a new client base for their email marketing. One of the challenges for email marketing is that emails are easily ignored by recipients due to a lack of customer base who is potentially interested in their service (Mari, 2019). In order to create a customer base, this project will analyse demographic data for both the general population at large as well as for prior customers for targets of a mail-out marketing campaign.
This project will explore 4 datasets, provided by the Arvato in the context of Udacity Machine Learning Engineer Nanodegree course.

- Udacity_AZDIAS_052018.csv: Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns).
- Udacity_CUSTOMERS_052018.csv: Demographics data for customers of a mail-order company; 191 652 persons (rows) x 369 features (columns).
- Udacity_MAILOUT_052018_TRAIN.csv: Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns).
- Udacity_MAILOUT_052018_TEST.csv: Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns).

Also, following datasets are provided to help better mapping of attributes data.

- DIAS Information Levels - Attributes 2017.xlsx: A top-level list of attributes and descriptions, organised by informational category.
- DIAS Attributes - Values 2017.xlsx: A detailed mapping of data values for each feature in alphabetical order.

Problem and solution statement

The problem to solve is “How might a mail-order sales company in Germany be able to acquire new clients in an efficient manner?”
This project will firstly use unsupervised learning approach to segment customer base, and then use supervised learning approach to predict potential targets.
For unsupervised learning, feature scaling, PCA, and KMeans clustering are considered to be used for a solution. For supervised learning, GridSearchCV is used to identify targets.

Evaluation Metrics

This project will use metrics such as accuracy, precision, and recall as evaluation metrics for supervised learning. It is also suggested that ROC-AUC is an option sinde this project is a binary classification problem with class imbalances.

Data Exploration and data processing

The findings based on my data exploration are as follows:

1. Mixed data type in a column “CAMEO_DEUG_2015”, “CAMEO_INTL_2015” with missing values as shown as “X” or “XX”. The data type will be converted to float, and values of “X” and “XX” will be replaced with np.nan. There’s also an unnecessary column with the “Unnamed: 0” label, which will be dropped.

```
print(azdias.iloc[:,18:20].columns)
print(customers.iloc[:,18:20].columns)

def data_cleanup(df):
    cols = ['CAMEO_DEUG_2015', 'CAMEO_INTL_2015']
    df[cols] = df[cols].replace({'X': np.nan, 'XX': np.nan})
    df[cols] = df[cols].astype(float)

    return df

def dias_cleanup(df):
    df.drop(columns=['Unnamed: 0'], inplace=True)
    return df
```

```
Index(['CAMEO_DEUG_2015', 'CAMEO_INTL_2015'], dtype='object')
Index(['CAMEO_DEUG_2015', 'CAMEO_INTL_2015'], dtype='object')
```

2. There is “unknown” meaning in attribute data, and these attribute data will be dropped in customer data.

```
unknown = dias_xls_cleaned.loc[np.where(dias_xls_cleaned['Meaning'].str.contains('unknown'))]['Meaning'].unique()
unknown_meaning = dias_xls_cleaned[dias_xls_cleaned['Meaning'].isin(unknown)]

unknown_meaning
```

	Attribute	Description	Value	Meaning
0	AGER_TYP	best-ager typology	-1	unknown
5	ALTERSKATEGORIE_GROB	age classification through prename analysis	-1, 0	unknown
11	ALTER_HH	main age within the household	0	unknown / no main age detectable
33	ANREDE_KZ	gender	-1, 0	unknown
40	BALLRAUM	distance to next urban centre	-1	unknown
...	...	...	...	...
2219	WOHNDAUER_2008	length of residence	-1, 0	unknown
2229	WOHNLAGE	residential-area	-1	unknown
2238	WACHSTUMSGEBIET_NB	growing area (population growth in the last 5 ...	-1, 0	unknown
2244	W_KEIT_KIND_HH	likelihood of a child present in this household	-1, 0	unknown
2251	ZABEOTYP	typification of energy consumers	-1, 9	unknown

244 rows × 4 columns

3. With the data exploration process, there are attributes that have more than 30% null value and will be dropped in customer dataset.

```
# Find attributes with more than 30% null data
missing_val_azdias = pd.DataFrame(data = azdias_copy.isnull().mean(), columns = ['null_percentage'])
missing_val_customers = pd.DataFrame(data = customers_copy.isnull().mean(), columns = ['null_percentage'])

azdias_over30 = missing_val_azdias[missing_val_azdias['null_percentage'] > 0.3]
customers_over30 = missing_val_customers[missing_val_customers['null_percentage'] > 0.3]
```

```
# Find unique attributes (with more than 30% null data) between customers and azdias
np.setdiff1d(customers_over30.index, azdias_over30.index)
np.setdiff1d(azdias_over30.index, customers_over30.index)

# Find common attributes between customers and azdias in order to drop them
common_attr = np.intersect1d(customers_over30.index, azdias_over30.index)
common_attr
```

```
array(['LP_FAMILIE_GROB'], dtype=object)
```

```
array([], dtype=object)
```

```
array(['AGER_TYP', 'ALTER_HH', 'ALTER_KIND1', 'ALTER_KIND2',
       'ALTER_KIND3', 'ALTER_KIND4', 'EXTSEL992', 'KK_KUNDENTYP',
       'LP_STATUS_GROB'], dtype=object)
```

Algorithms and Techniques

Unsupervised learning

For unsupervised learning (customer segmentation), following techniques will be used.

Principal component analysis (PCA)

PCA transforms complex data into the most relevant components in order to reduce the complexity of analysis. Since the data set has around 370 features, this technique is useful to find most relevant components and drop irrelevant ones.

K-means clustering

K-means clustering is a method of partitioning k clusters with the nearest mean in order to identify k number of centroids (cluster centers). This method helps to identify groups with similar features and characteristics.

Supervised learning

For supervised learning, following techniques will be tested as a prediction model.

Random Forest Classifier

This model creates various decision trees by selecting a partial training data randomly, and seek important feature to contribute to the prediction

Ada Boost Classifier

This model (Adaptive boosting classifier) converts a set of weak classifiers into a single strong classifier by assigning weight to each classifier and iterating till certain point (either all the data points have been correctly classified, or reached to maximum iteration level)

Gradient Boosting Classifier

This method uses weak classifiers and combines with weighted minimization aiming to minimise the loss between the trained value and predicted value.

Logistic Regression

This model creates segmentation with indicators (0 or 1, in this project, if a customer responded to the email campaign) using a logistic function

GridSearchCV

This method helps to find the optimal hyperparameter values for a given model by automating the hyperparameter tuning.

Benchmark Model

This project will use Logistic Regression as a benchmark model for core customer base since this method is the simplest model.

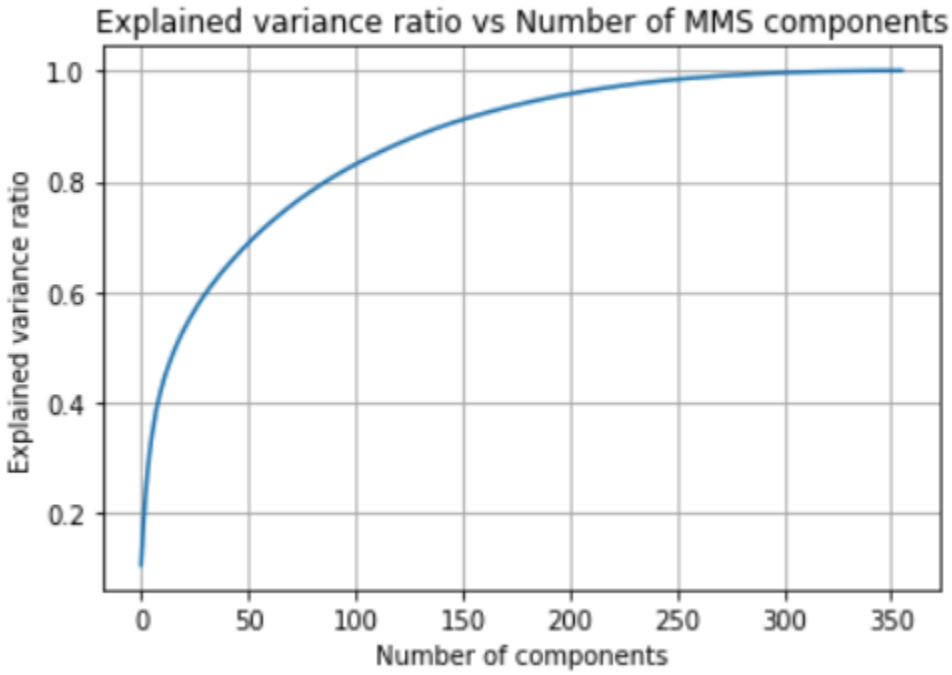
	Model	Score
0	Logistic Regression	0.674862
1	Random Forest Classifier	0.605910
2	Ada Boost Classifier	0.727178
3	Gradient Boosting Classifier	0.758261

Implementation

Unsupervised learning model

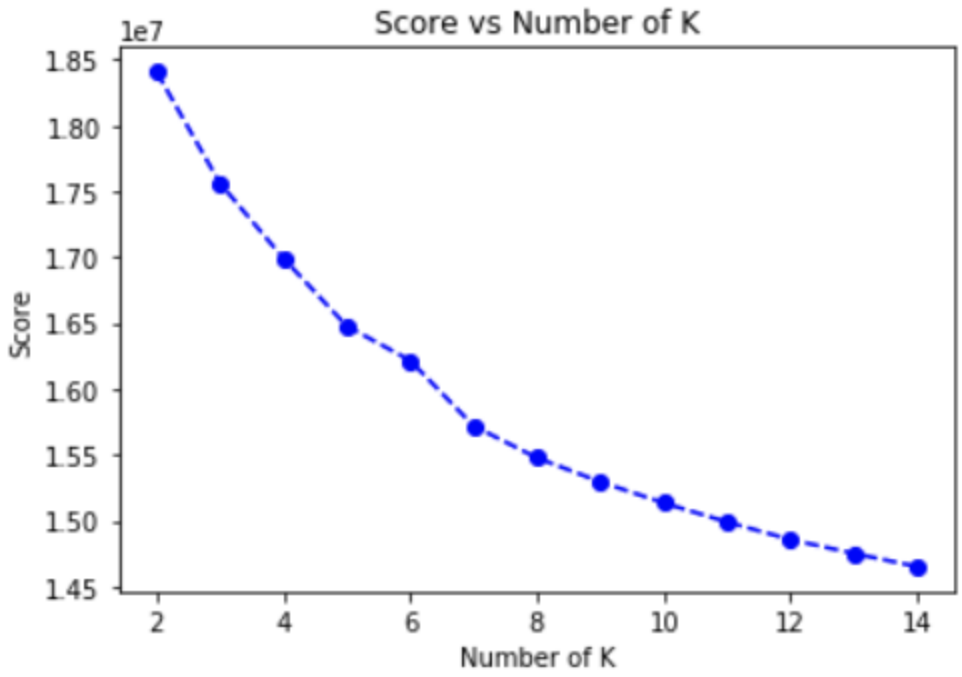
Dimensionality Reduction

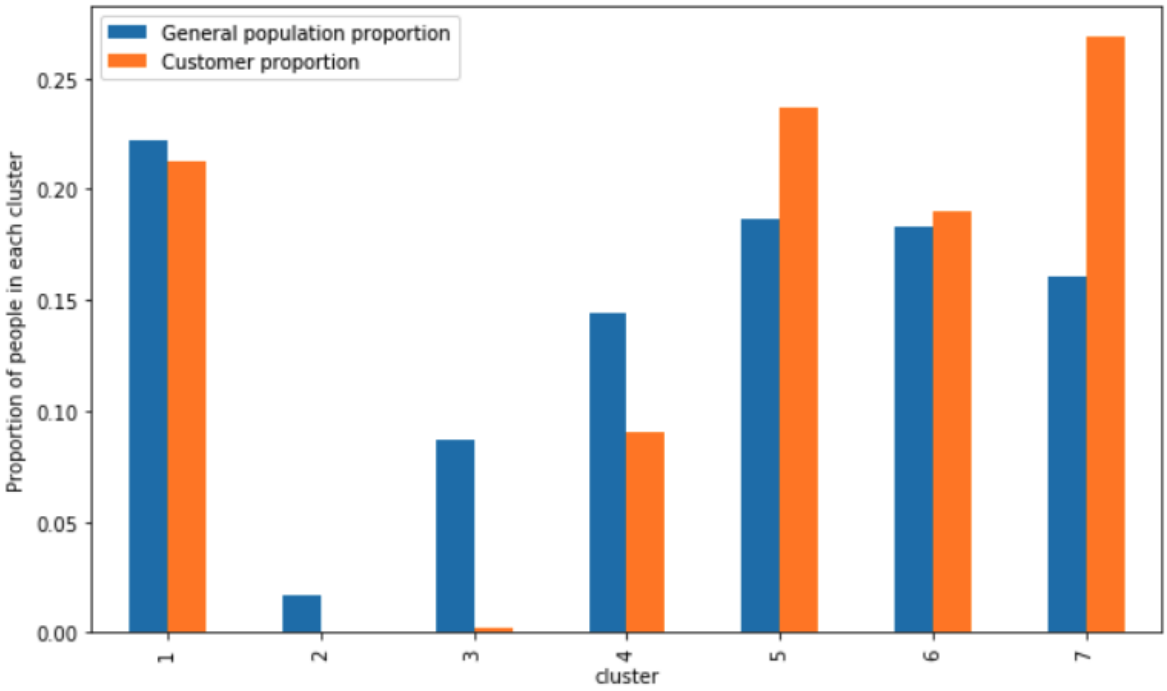
Dimensionality reduction creates less features with model creation, and this makes it less complex and prevents overfitting. This project used MinMaxScaler.
After scaling the data, principal component analysis (PCA) is used to extract features that are relevant to the problem.
The scree plots below (line plot of PCA analysis) show that 200 components representing 90% of original variance, and this project will use the 200 components for further analysis.



K means clustering

K-means clustering is to find k clusters with the nearest mean, and seek the optimal k by looking at the number of k and its score (sum of squares of the distance between data points and respective cluster centroids). The figure below shows the relationship between k and its score, and k=7 looks to be an optimal number.
Also, the figure about proportion of people in each cluster depicts the cluster 1, 5, 6, and 7 representing the core customer base.



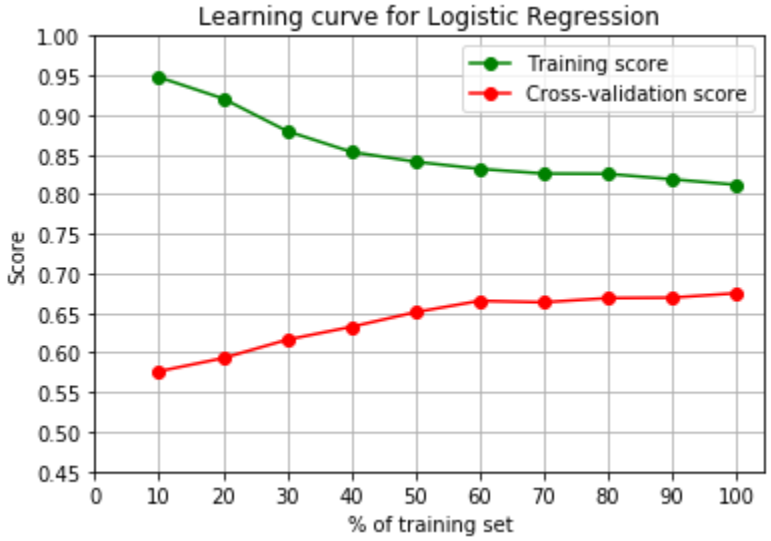


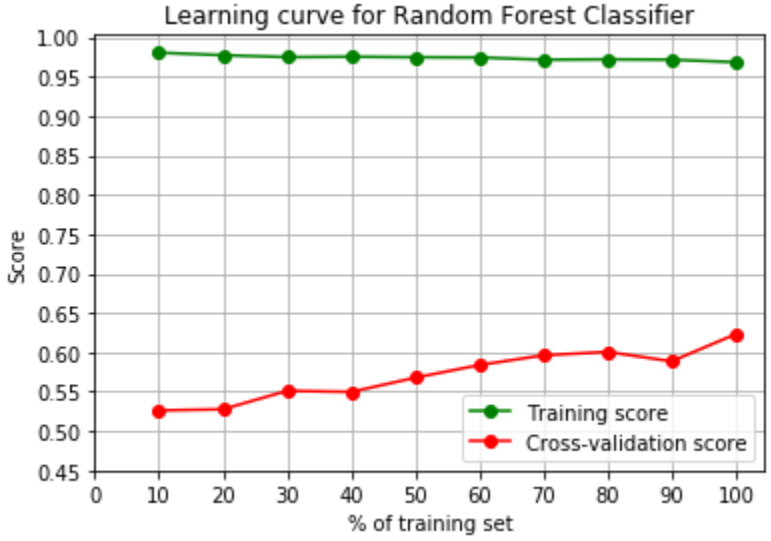
Supervised learning model

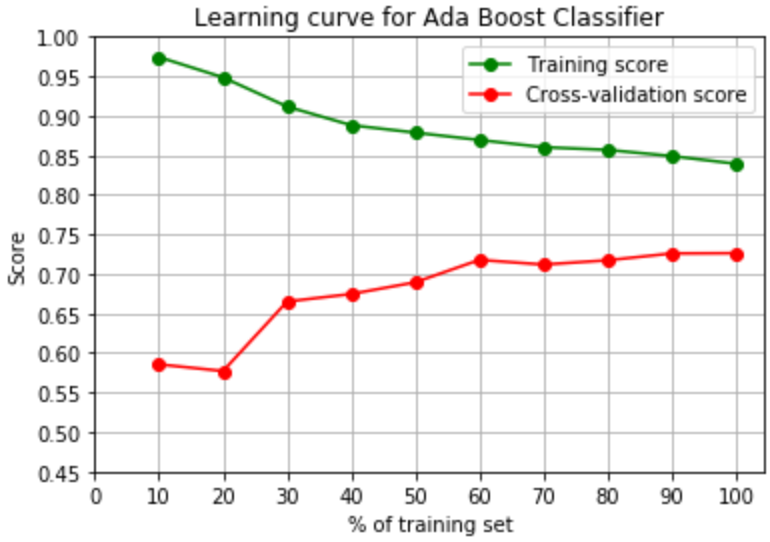
This project will test the following classifiers, and each models' scores are as follows. Based on the results, this project will use Gradient boosting classifier for further training and tuning.

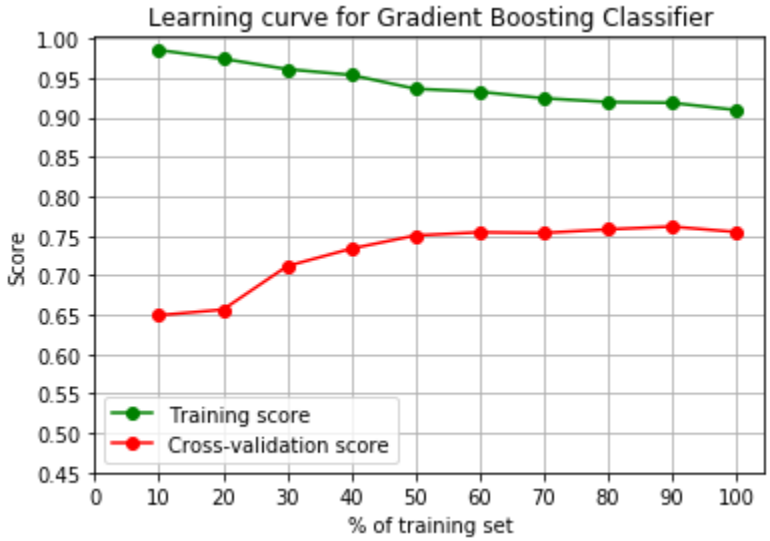
- Logistic regression (roc_auc train score: 0.81, roc_auc validation score: 0.67)

- Random forest classifier (roc_auc train score: 0.97, roc_auc validation score: 0.62)
- Ada boost classifier (roc_auc train score: 0.84, roc_auc validation score: 0.73)
- Gradient boosting classifier (roc_auc train score: 0.91, roc_auc validation score: 0.75)









Refinement (Hyperparameter tuning)

In order to find the better hyperparameter for the model, this project will use GridSearchCV technique, and the roc_auc score was
Initial score: 0.758261
Final score: 0.760382

```
X_train, X_test, y_train, y_test = train_test_split(processed_mailout_train, target, test_size=
GradientBoosting = GradientBoostingClassifier()
parameters = {'n_estimators': [100, 200, 500],
              "max_depth":[3,5]
             }
gridSearch_model = GridSearchCV(estimator=GradientBoosting, param_grid=parameters, cv=5, scorin
gridSearch_model.fit(X_train, y_train)
gridSearch_model.best_params_, gridSearch_model.best_score_
```

```
GridSearchCV(cv=5, error_score='raise-deprecating',
             estimator=GradientBoostingClassifier(criterion='friedman_mse',
                                                  init=None, learning_rate=0.1,
                                                  loss='deviance', max_depth=3,
                                                  max_features=None,
                                                  max_leaf_nodes=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=100,
                                                  n_iter_no_change=None,
                                                  presort='auto',
                                                  random_state=None,
                                                  subsample=1.0, tol=0.0001,
                                                  validation_fraction=0.1,
                                                  verbose=0, warm_start=False),
             iid='warn', n_jobs=None,
             param_grid={'max_depth': [3, 5], 'n_estimators': [100, 200, 500]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring='roc_auc', verbose=0)

({'max_depth': 3, 'n_estimators': 100}, 0.7564029222808666)
```

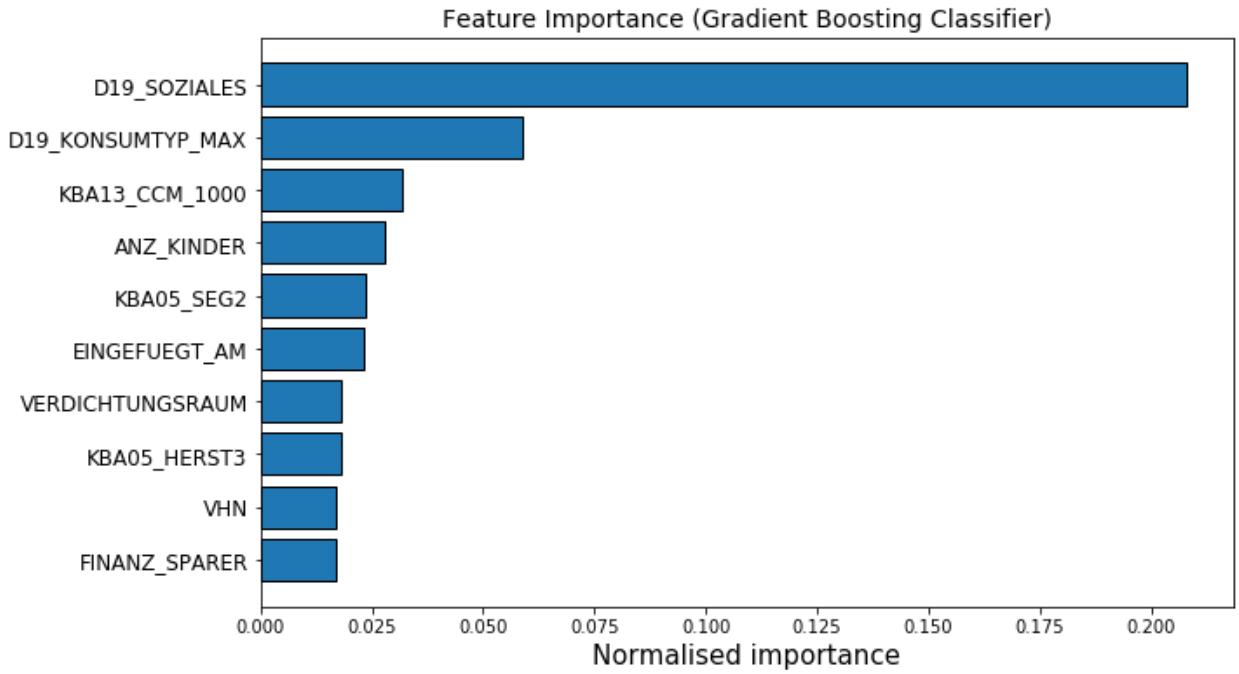
```
parameters = {"learning_rate": [0.05, 0.1, 0.15],
              "min_samples_split": [2, 3, 4],
              "min_samples_leaf": [1, 2, 3]
             }
gridSearch_model = GridSearchCV(estimator=GradientBoostingClassifier(max_depth=3,n_estimators=1
                                param_grid=parameters,
                                cv=5,
                                scoring='roc_auc')
gridSearch_model.fit(X_train, y_train)
gridSearch_model.best_params_, gridSearch_model.best_score_
```

```
GridSearchCV(cv=5, error_score='raise-deprecating',
             estimator=GradientBoostingClassifier(criterion='friedman_mse',
                                                  init=None, learning_rate=0.1,
                                                  loss='deviance', max_depth=3,
                                                  max_features=None,
                                                  max_leaf_nodes=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=100,
                                                  n_iter_no_change=None,
                                                  presort='auto',
                                                  random_state=None,
                                                  subsample=1.0, tol=0.0001,
                                                  validation_fraction=0.1,
                                                  verbose=0, warm_start=False),
             iid='warn', n_jobs=None,
             param_grid={'learning_rate': [0.05, 0.1, 0.15],
                         'min_samples_leaf': [1, 2, 3],
                         'min_samples_split': [2, 3, 4]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring='roc_auc', verbose=0)
```

```
({'learning_rate': 0.05, 'min_samples_leaf': 2, 'min_samples_split': 2},
 0.7603820448463853)
```

Results

With comparison to benchmark (Logistic regression) being 0.67 and the final model being 0.76, the final model gives a better prediction to identify possible customer to likely respond to Arvato Financial Services's email campaign.
Also, plotting feature importance from the tuned model with hyperparameters, "D19_SOZIALES" feature has the most significant impact in the model.



Citation

The Rise of Machine Learning in Marketing: Goal, Process, and Benefit of AI-Driven Marketing (Alex Mark, 2019)