**Project Overview:**

Arvato-Bertelsmann financial solutions company is headquartered in Germany; it provides customer support to various companies in different domain. Arvato-Bertelsmann uses latest technology to help their customers in the best way possible. In this project, I will be using data provided by Arvato to determine segment of population who are more likely to become customers at a German mail-order company.

***Note: The python notebook is in two parts. ‘Final Model.ipynb’ contains most of the work including data preprocessing, visualtization, unsupervised learning, and some of supervised learning. The second notebook ‘Supervised\_Feature\_Selection.ipynb’ includes supervised learning with feature selection/scale on the data. This was divided to give due attention to details for supervised learning.***

**Data Overview:**

There are four differest datasets provided and two excel sheets to provide additional information about the data.

The four separate datasets are listed below with brief descriptions:

* [**Udacity\_AZDIAS\_052018.csv**](http://localhost:8888/edit/data/Udacity_AZDIAS_052018.csv)**:** contains demographics data for the general population of Germany. Each row represents an individual and the columns represent features [891,211 rows and 366 features]
* [**Udacity\_CUSTOMERS\_052018.csv**](http://localhost:8888/edit/data/Udacity_CUSTOMERS_052018.csv): contains demographics data for the existing customers of mail-order Company [191,652 rows (individuals) with 369 columns (features)].
* [**Udacity\_MAILOUT\_052018\_TEST.csv**](http://localhost:8888/edit/data/Udacity_MAILOUT_052018_TEST.csv): Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns).
* [**Udacity\_MAILOUT\_052018\_TRAIN.csv**](http://localhost:8888/edit/data/Udacity_MAILOUT_052018_TRAIN.csv): Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns).

There are two other excel sheets which provide additional attributes to the datasets.

* [**DIAS Information Levels - Attributes 2017.xlsx**](http://localhost:8888/edit/data/DIAS%20Information%20Levels%20-%20Attributes%202017.xlsx): top-level list of attributes and descriptions, organized by informational category.
* [**DIAS Attributes - Values 2017.xlsx**](http://localhost:8888/edit/data/DIAS%20Attributes%20-%20Values%202017.xlsx): detailed mapping of data values for each feature in alphabetical order.

**Libraries:**

Missingno and seaborn for visualizing the data; matplotlib.pyplot, pandas, numpy, sklearn for algorithms and hyperparameter tuning.

**Data Preprocessing:**

Clean data function is created to map different abnormalities in columns and transform them accordingly. Non-numeric data is one hot encoded, missing data is filled with mean values and some columns with insignificant are dropped.

**Unsupervised Learning:**

Clustering is applied with and without feature selection, and scaling. Elbow graph is used to determine the optimum number of clusters. Performance metric are to evaluate each of the algorithms: accuracy, f1, precision, recall and roc\_auc. Hyperparameter tuning is then applied using GridSearchCV. GridSearchCV is applied on three of the best supervised learning algorithms; Logistic Regression, Random Forest Classifier and XGBoost. The results are saved as: logisticresults.csv, rfcresults.csv, and XGBresults.csv.

**Supervised Learning:**

Logistic Regression, Decision Tree Classifier, Random Forest Classifier, XGBoost, and KNeighbors Classifier are all experimented with.

Hyperparameter tuning is then applied using GridSearchCV. GridSearchCV is applied on three of the best supervised learning algorithms; Logistic Regression, Random Forest Classifier and XGBoost. The results are saved as: logisticresults.csv, rfcresults.csv, and XGBresults.csv.

Feature selection, PCA and scaling are also applied to improve the results and submitted to the Kaggle competition.

**Results and Analysis:**

The table below shows models with their parameters and scores on Kaggle competition. I chose to look further into Logistic regression, random forest classifier and XGBoost classifier and test how well they perform on the Kaggle competition.

XG Boost Classifier performed the best on Kaggle competition, with a score of about 72%. The parameters used are the following: n\_estimators = 100, max\_depth=1, verbosity=1, random\_state=42, n\_jobs=-1). These parameters were chosen by default, there were no feature selection, or scaling done on the data to arrive at this result. The parameters were further tuned to achieve higher score (as shown in the table below); with feature scaling/selection and GridSearchCV. However, that decreased the scores on the Kaggle Competition. XGBoost Classifier is a tree-based model, in such models feature scaling does not make much of a difference. Ensembles of decision tree methods [XG Boost] automatically determines feature importance, and parameters based on the given data. Therefore, further feature selection, scaling, and hyperparameter tuning only lowered the performance for XG Boost. XG Boost was performing at its optimum on default settings.

The second-best performance was by Random Forest Classifier, which is also a method based on set of decision trees. The parameters used were the following: (class\_weight='balanced’, criterion='entropy', max\_depth=4, n\_estimators=200, random\_state=42, max\_features='auto').

Random Forest performed at its optimum after hyperparameter tuning based on GridSearchCV. However, using features selection/scaling along with GridSearchCV decreased the score on the Kaggle competition.

Logistic Regression performed the worst overall, especially at its default (as shown in table below). Logistic Regression performance improved after feature selection/scaling and hyperparameter tuning. Unlike Random Forest Classifier and XG Boost Classifier, Logistic regression is not a tree-based model. Instead, it is linear based and creates one separable line to classify the data.

**Conclusion:**

In this project, a large dataset was given to determine a segment of population that are most likely to become customers to the mail-order company. The two most interesting and challenging parts of the project was cleaning the data to decrease noise and determining which model works best for the problem.

I learned that Data cleaning is very crucial step, and it requires a lot of thinking to determine what data is worth keeping. It is important to get rid of the noise, but it is also very important to retain valuable. That was a bit of struggle for me, keeping the balance is the key.

The second challenging part was analyzing each model and determining which one performs the best and why. There are large number of models to choose from, but its counterproductive to try using all of them. It is not only important to choose the optimum model, but to also choose the optimum parameters for the models. Each model has its own set of advantages and disadvantages, there is no one correct answer, but decisions must be based on proper results and analysis.

There are lots of ways to improve the machine learning models in this project. I focused on three models, but there are many more models to experiment with, and endless numbers of parameters that can be tuned further to improve the data. I applied a lot of hyperparameter tuning and feature selections myself, but there are so many ways they can be refined and changed to improve model performance. However, it is important to note that constant attempts at hyperparameter tuning will not always improve the performance of the model.