***Dataset Information:***

My dataset is online\_shoppers\_intention.csv. It is from the UCI Machine Learning Repository. There are 12330 sessions. It consists of 10 numerical and 8 categorical attributes.

Classification models will be utilized to predict the target variable is Revenue, a Boolean.

***Data Wrangling:***

When exploring the dataset, there are two object datatype columns. To enable models running smoothly, they are transferred to integers. Then the dataset is checked for null values and all NA values are dropped.

The different ranges of dataset complicate the training process. To optimize the training speed and avoid possible problems, data is normalized before splitting into train and test sets.

84.5% of the data are negative class samples (False). This uneven distribution is shown in the histogram below. Due to this uneven distribution of True and False for the target variable, stratify=y is included when splitting data in training and testing sets.

Chart, bar chart

Description automatically generated

***Training Models:***

Baseline Accuracy = 0.8438862362466265

Six models, including generic bagging, random forest, adaboost classification, voting ensemble, decision tree and logistic regression, are trained for this dataset. For all models, random\_state is set at 42, n\_estimator is 100, and max\_depth is 10. For every model, accuracy and cross validation scores are calculated. These scores will be used to compared and select the most suitable model. To avoid over-fitting, pruning and regularization are utilized for decision tree and logistic regression.

**Ensemble methods**

Generic Bagging model Accuracy = 0.8974465434917999

[0.81030603 0.81182628 0.78110677 0.74245171 0.7493262 ]

0.78 accuracy with a standard deviation of 0.03

Classification Report:

precision recall f1-score support

0.0 0.92 0.96 0.94 4065

1.0 0.73 0.55 0.63 752

accuracy 0.90 4817

macro avg 0.82 0.76 0.78 4817

weighted avg 0.89 0.90 0.89 4817

Random Forest model Accuracy = 0.8968237492215072

[0.80649308 0.81213755 0.78492233 0.73808902 0.75045322]

0.78 accuracy with a standard deviation of 0.03

Classification Report:

precision recall f1-score support

0.0 0.92 0.96 0.94 4065

1.0 0.72 0.56 0.63 752

accuracy 0.90 4817

macro avg 0.82 0.76 0.78 4817

weighted avg 0.89 0.90 0.89 4817

AdaBoost Classification Accuracy = 0.8922565912393606

[0.79093833 0.78670757 0.74999431 0.7080819 0.72589734]

0.75 accuracy with a standard deviation of 0.03

Classification Report:

precision recall f1-score support

0.0 0.92 0.96 0.94 4065

1.0 0.71 0.53 0.61 752

accuracy 0.89 4817

macro avg 0.81 0.74 0.77 4817

weighted avg 0.88 0.89 0.89 4817

Voting Ensemble for Classification Accuracy = 0.8889350217977995

[0.79921743 0.81102649 0.79344052 0.75194419 0.75005975]

0.78 accuracy with a standard deviation of 0.03

Classification Report:

precision recall f1-score support

0.0 0.92 0.96 0.94 4065

1.0 0.73 0.55 0.63 752

accuracy 0.90 4817

macro avg 0.82 0.76 0.78 4817

weighted avg 0.89 0.90 0.89 4817

**Decision tree**

Decision Tree Accuracy = 0.8820842848245796

[0.78046899 0.81190918 0.72902594 0.74293292 0.71722611]

0.76 accuracy with a standard deviation of 0.04

Test Confusion matrix:

Chart, treemap chart

Description automatically generated

Decision tree:

Background pattern

Description automatically generated

Pruning

Decision Tree Accuracy = 0.8889350217977995

[0.78613425 0.7887965 0.68871552 0.67262767 0.65941851]

0.72 accuracy with a standard deviation of 0.06

Test Confusion matrix:

Chart, treemap chart

Description automatically generated

Decision tree (simplified to avoid overfitting):

Background pattern

Description automatically generated

**Logistic regression**

logistic regression Accuracy: 0.8717043803197011

[0.60481955 0.66130889 0.68091798 0.6745786 0.67927694]

0.66 accuracy with a standard deviation of 0.03

Regularization

After regularization using solver, liblinear, the accuracy increased a little.

logistic regression Accuracy: 0.8721195764998962

[0.60891887 0.6648213 0.67933337 0.67298354 0.67774283]

0.66 accuracy with a standard deviation of 0.03

**Conclusion**

Among all models, Generic Bagging model has the highest accuracy and high accuracy for positive cases. This result is a little surprising since generic bagging model might have overfitting problem and lower accuracy compared to other ensemble models. Maybe a very optimal model is chosen in this case with light overfitting and high accuracy.

***Problem of Multicollinearity***

Correlation matrix is drawn to check for high correlation between any two variables (param = 0.7). The high correlation might cause the problem of multicollinearity especially for linear regression models.

As shown in the matrix, the correlation between ProductRelated\_Duration and ProductRelated is 0.86 which are greater than 0.7. The correlation between ExitRates and BounceRates is 0.91 which are greater than 0.7.

Chart

Description automatically generated

To address the multicollinearity problem, the data is trained again using generic bagging model after removing ProductRelated\_Duration and BounceRates columns.

Baseline Accuracy = 0.8438970384721838

Generic Bagging model Accuracy = 0.8920564627733186

0.78 accuracy with a standard deviation of 0.03

Classification Report:

precision recall f1-score support

False 0.92 0.96 0.94 3049

True 0.70 0.55 0.61 564

accuracy 0.89 3613

macro avg 0.81 0.75 0.78 3613

weighted avg 0.88 0.89 0.89 3613

Before addressing the problem of multicollinearity:

Generic Bagging model Accuracy = 0.8974465434917999

[0.81030603 0.81182628 0.78110677 0.74245171 0.7493262 ]

0.78 accuracy with a standard deviation of 0.03

**Conclusion**

The resulting accuracy is slightly lower. Perhaps a set of optimal models are chosen in generic bagging, minimizing the problem of collinearity and overfitting. The original Generic Bagging model can be used to predict consumer behavior (purchase or not).

***Citation:***

Sakar, C.O., Polat, S.O., Katircioglu, M. et al. Neural Comput & Applic (2018). [[Web Link]](https://doi.org/10.1007/s00521-018-3523-0)