This data provides a binary classification of Customer Churn for a particular telco. The aim is to produce an effective machine learning model to predict the churn rate from the data attributes.

Data Cleaning and Pre-processing

As the proportion of whitespaces in the dataset is very small and negligible, they are discarded to avoid introducing any data errors.

In the dataset, customers with “No Internet/Phone Service” are converted to “No” because they are essentially the same. This is substantiated by the parsimony principle which reinforces that the system in machine learning should be kept simple with only important independent variables. Binary columns are converted to 1s and 0s using the LabelEncoder() while multi-value columns are converted to dummies using the pd.get_dummies(). Numerical columns are standardised using MinMaxScaler(). Data pre-processing allows variables to be used for modelling, and normalizing numerical variables to a common scale ensures that equal weight is assigned to every feature. Doing so would produce a fair and accurate model.

Exploratory Data Analysis

To identify any relations between churn rate and variables, a series of graphs (bar graphs/boxplot/violin plot/distribution graph) are utilized. To measure the correlation between numerical variables, pairplot (scatterplot) and heatmap (correlation matrix) are used. To conduct multivariate analysis, a 3-dimensional scatter plane for numerical variables and a radar chart for binary features are plotted. To understand the distribution of numerical variables, a summary of their mean and standard deviation and percentiles are described.

Feature Selection

By the parsimony principle, statistically significant features are selected to maximise model performance. Incorporating all 19 features would overfit on the train dataset. In feature selection, chi2 is employed as a scoring function in evaluating the p-value of every feature. Gender and PhoneService are insignificant as their p-values are less than 0.05 at the 5% significance level, and are therefore removed. TotalCharges is also removed as it has high collinearity with tenure and monthly charges, and the addition of it does not improve the model significantly.

Machine Learning

Accuracy and sensitivity are performance metrics used to rank the models. By the accuracy paradox, accuracy alone can be a misleading measure.[1] Sensitivity and precision, however are less ambiguous. In the case of a telco, it is desirable to have high sensitivity (focus on the group of customers that have a higher propensity to churn) at the expense of precision since failing to recognize these customers leads to a greater loss for their company.

At the train_test_split, stratified sampling is done to reduce variability between the train and test set, thereby maintaining the data integrity of the original dataset which helps create a more representative model.[2]

As we are interested in the churned customers which represents the minority, oversampling by SMOTE on the minority is employed. SMOTE addresses the unbalanced training set and reduces information leakage by creating synthetic data points which are similar to the existing using nearest neighbors classification instead of duplicating existing data points.[3]

For cross-validation, a pipeline is created to ensure oversampling (SMOTE) is done at each iteration instead of before cross-validating. The validation set remains unseen with no similar entries as in the training set, hence reducing any data leakage and overfitting problems.[4]

Decision Tree, Ensemble Methods (Bagging, Random Forest, AdaBoost), Logistic Regression, KNN Classifier, Support Vector Machine (SVM) are models built to predict churn rate. Results of the models can be found in the appendix.[5]

Performance of learning models on data vary as they make different assumptions and have different rates of convergence. As such, multiple models have to be tested to find the appropriate one. Each model may require different constraints, weights or learning rates to understand data patterns. Default parameters are set for each model. The best 3 models will then have their hyperparameters fine-tuned to maximise their recall_score.

Hyperparameters Optimisation

Logistic Regression, Adaboost and SVM have relatively high accuracy and recall scores.[5] Using GridSearchCV,[6] the following hyperparameters are fine-tuned: *1) Logistic Regression: penalty and C 2) SVM: C, gamma and kernel 3) Adaboost: learning_rate and n_estimators*

A pipeline for GridSearchCV is created to ensure SMOTE is done at each iteration, reducing data leakage.

Both Adaboost and Logistic Regression with finetuned hyperparameters are evaluated as the best models given their high accuracy and recall scores.[7] The computational efficiency of each model can be then used to assess the better model.

Conclusion

There a few ways to improve the model. 1) Adding more significant information[8] such as customer's service rating, income bracket, wider age range and whether customer is exposed to advertisement from rival companies. 2) Feature Creation[8]. Deriving new variables from existing variables helps to reveal hidden relationship of a dataset. 3) Meta ensembling (stacking). It combines multiple predictive models to generate a new model. As it highlights good models and discredits bad ones, the resulting model outperforms the base models.[9] 4) Using neural networks which have proven to be highly accurate at classification.

Appendix

1. Precision vs Recall – Demystifying Accuracy Paradox in Machine Learning. Retrieved October 4, 2018, from https://www.newgenapps.com/blog/precision-vs-recall-accuracy-paradox-machine-learning
2. Why Stratify? Retrieved October 4, 2018, from http://wiki.landscapetoolbox.org/doku.php/general_design_topics:why_stratify
3. Deep learning unbalanced training data? Solve it like this. Retrieved October 4, 2018 from https://towardsdatascience.com/deep-learning-unbalanced-training-data-solve-it-like-this-6c528e9efea6
4. Dealing with Imbalanced Data: Undersampling, Oversampling and Proper Cross-Validation. Retrieved 4 October, 2018 from https://www.marcoaltini.com/blog/dealing-with-imbalanced-data-undersampling-oversampling-and-proper-cross-validation
5. Scores of every machine learning model. Retrieved October 4, 2018

Model	Accuracy_score	Recall_score	Precision	f1_score	ROC_AUC_Score
Decision Tree	0.7066	0.5365	0.4561	0.493	0.6524
Bagging	0.7237	0.5918	0.484	0.5325	0.6816
Random Forest	0.7782	0.59	0.5817	0.5858	0.7182
Adaboost	0.7735	0.7255	0.5568	0.63	0.7582
Logistic Regression	0.7583	0.7932	0.5304	0.6357	0.7694
KNN Classifier	0.7071	0.7451	0.4681	0.575	0.7192
SVC	0.7403	0.8164	0.5072	0.6257	0.7646 Export to plot.ly »

6. Understanding Hyperparameters and its Optimisation techniques. Retrieved October 4, 2018, from https://towardsdatascience.com/understanding-hyperparameters-and-its-optimisation-techniques-f0debba07568
7. Scores of models with/without GridsearchCV. Retrieved October 4, 2018

	Model	Accuracy_score	Recall_score	Precision	f1_score	ROC_AUC_Score
0	Adaboost	0.773460	0.725490	0.556772	0.630031	0.758161
1	Logistic Regression	0.758294	0.793226	0.530393	0.635714	0.769434
2	SVC	0.740284	0.816399	0.507198	0.625683	0.764559
3	Logistic Regression (GridSearchCV)	0.758768	0.793226	0.531026	0.636169	0.769757
4	SVC (GridSearchCV)	0.657820	0.894831	0.430901	0.581692	0.733406
5	Adaboost (GridSearchCV)	0.760664	0.798574	0.533333	0.639543	0.772754

8. 8 Proven Ways for improving the “Accuracy” of a Machine Learning Model. Retrieved October 4, 2018, from https://www.analyticsvidhya.com/blog/2015/12/improve-machine-learning-results/
9. A Kaggler’s Guide to Model Stacking in Practice. Retrieved October 4, 2018, from http://blog.kaggle.com/2016/12/27/a-kagglers-guide-to-model-stacking-in-practice/