Decision tree and Ensembling models to predict customer churn - Kenny Cai

Introduction:

This report discusses the processes in developing models to predict customer retention.

Part A focuses on the use of a single Decision Tree model to find an interpretable 'white box' model. It is designed to illustrate a 'best' single Decision Tree by exploring a variety of aspects including feature selection and parameter exploration.

Part B will focus on building the most accurate model that is not limited to a single Decision Tree, but instead, using Ensembling methods. The best model will then be used to predict a possible churn value (customer retention) for a customer.

Part A – Interpretable model using a Decision Tree

1. Understanding the Data set:

The data set given has a total of 18 features and 1 target (customer churn) from the previous month. It contains details including demographic information, customer account information, and services that customers had signed up for.

A few interesting features:

SeniorCitizen – the data set had 29% senior citizens with a churn of 67%, as compared with non-senior citizens with a churn of only 10%. This shows that a higher proportion of senior citizens decided to leave within the last month.

TechSupport – this feature had a distribution of 58% with tech support and 42% without. This could be an interesting feature since having weak customer support could potentially be a predictor for customers leaving.

Contract – the distribution of contracts is 55% month-to-month, 21% one-year contracts and 24% two-year contracts. This feature could also be interesting as customers on month-to-month contracts could be more likely to leave as they are not tied down to a long-term contract.

2. Data Pre-processing:

In preparation for fitting the Decision Tree model, the data needed to be pre-processed into numerical data for sklearn. Table 1 shows a breakdown of the data types.

Table 1: Features in the dataset and their encoded types

Features	**Quantity**	**Type**	**Encoded Type**
'tenure', 'MonthlyCharges', 'TotalCharges'	3	Numerical	*Unchanged
'gender','Partner', 'Dependents', 'PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling', 'PaymentMethod'	15	Categorical nominal	One-hot-encoded
Target		**Type**	
'Churn'	1	Categorical nominal	Binary label encoded (0 - No,1 - yes)

Additional points:

- Numerical data – Remained unchanged. Normalisation or Standardisation is not strictly needed for a single decision tree.
- Categorical nominal (Features) – One-hot-encoded. As part of the model building, these feature
- Categorical nominal (Target) – binary numerical encoded (1 for yes/ 0 for no), standard inputs for sklearn.

3. Train test split:

70% of the data was used to train the model and 30% of the data was used as the validation set.

4. Searching for best model

4.1 Parameter searching

Sklearn's GridSearchCV allows us to search through every combination of the parameters using cross validation and taking the best mean score identified by a specific score metric (accuracy, auc_roc etc.).

Table 2 shows the results of some of the parameters searched.

Table 2: Results for a Decision Tree Model

Parameters searched	**Model Parameters**	**Score metric**	**Train Data results**	**Test data results**	**10 fold CV hold out data set results**
None	Max_depth = 20 Random state = 42	N/A	Score = 0.995 Precision = 0.996 Recall = 0.986	Score = 0.816 Precision = 0.650 Recall = 0.626	Score = 0.806 Precision = 0.623 Recall = 0.634
Max_depth = 1-10 **10 fold CV**	**Max_depth = 5** **Random_state = 42**	**roc_auc**	**Score = 0.858** **Precision = 0.791** **Recall = 0.639**	**Score = 0.861** **Precision = 0.757** **Recall = 0.630**	**Score = 0.857** **Precision = 0.757** **Recall = 0.703**
Max_depth = 1-10 Min_samples_split = [1,2,3,4,5] Min_samples_leaf = [75,80,85,90,95,100] 10 fold CV	Max_depth = 6 Min_samples_leaf = 75 Min_samples_split = 2 Random_state = 42	roc_auc	Score = 0.860 Precision = 0.773 Recall = 0.669	Score = 0.866 Precision = 0.788 Recall = 0.695	Score = 0.864 Precision = 0.777 Recall = 0.695

Observations:

- When max_depth was high (e.g. 20) the train data results were very high, but test data results were low.
- Applying GridsearchCV with the following parameter search (max_depth, min_samples_split, min_samples_leaf) had minimal differences.
- Changing the k-fold CV parameter also had minimal differences.

- Changing the score metric when parameter searching from ‘accuracy’ to ‘roc_auc’ also had minimal differences. Although recall and precision were on average slightly higher on the roc_auc metric.
- Variances in each model’s precision and recall were considerably high, even after cross - validation predictions.

4.2 Feature pruning:

In another attempt to build a better model, some of the features were examined and removed from the feature set. The features that were removed were:

PhoneService – Whether the answer was Yes or No the customer churn percentage was 25% and 26% respectively, and thus was removed because of no correlation with the churn value.

MultipeLines – Same reason as above, it didn’t matter if that answer was Yes or No, the percentage of churn was very similar.

OnlineBackup and Device Protection – These two features had very similar churn percentages as OnlineSecurity. Also, both features could be a sub-feature of OnlineSecurity and thus not needed.

StreamingTV and StreamingMovies – These two features had very similar churn rates on either answer (Yes/No), thus was also removed.

Table 3: Results for model with reduced features

Parameters searched	Model Parameters	Score metric	Train Data results	Test data results	10 fold CV hold out data set results
Max_depth = 1-10	Max_depth = 5	roc_auc	Score = 0.869 Precision = 0.745 Recall = 0.693	Score = 0.851 Precision = 0.736 Recall = 0.685	Score = 0.848 Precision = 0.742 Recall = 0.683

5. Best Model

The best scoring model (accuracy, recall, precision) combined with least complexity chosen was the model with max_depth = 5 on the *entire feature set*. When applying GridSearchCV, this led to scores being either lower or had little difference. Additionally, the above model with reduced features also showed little improvement over the model with no feature reduction at all.

Thus the *best* model for DecisionTreeClassifier had the parameters:

Max_depth = 5, and default for the rest.

6. Model Evaluation

Table 4: Best model results

Score = 0.851	Precision = 0.757	Recall = 0.703

Confusion matrix

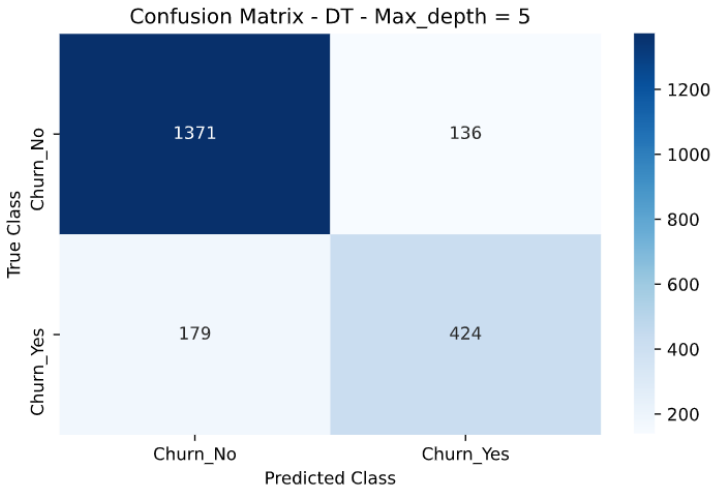

Figure 1: Confusion Matrix - Best Model

The following confusion matrix shows the results of the predictions of the test set with 2110 rows of data using 10-fold cross-validation. The accuracy was quite high at 85.7%. The confusion matrix had falsely predicted 179 customers staying when in fact they left (false negatives), and falsely predicted 136 customers leaving when in fact they stayed (false positive).

Given that we want to prevent customers from leaving, it would be wise to optimise recall, and reduce the number of false negatives if possible. If we could predict which customers were likely to leave, then directed marketing or other strategies could be implemented to prevent this from happening.

7. Example rules for predicting customer Churn (See appendix 1 for more details)

For Churn 1 (Customer leaves)

1. LEAVE – If customer is a senior citizen -> not on a month-to-month contract -> has a tenure of less than 15.5 -> has a tenure of less than 3.5 -> has streaming TV.
2. LEAVE – If customer is a senior citizen -> not on a month-to-month contract -> has a tenure of less than 15.5 -> has a tenure of less than 3.5 -> does not have streaming TV.
3. LEAVE – If customer is a senior citizen -> not on a month-to-month contract -> has a tenure of less than 15.5 -> does not have a tenure of less than 3.5 -> monthly charges are not over 83.4.

For Churn 0 (Retain customer)

1. STAY – Not a senior citizen -> has tenure of less than or equal to 13.5 -> does not have fibre optic -> does not have a tenure of 3.5 -> does not have a payment of mailed check.
2. STAY – Not a senior citizen -> has tenure of less than or equal to 13.5 -> does not have fibre optic -> does not have a tenure of 3.5 -> has the payment method of mailed check.

3. STAY – Not a senior citizen -> has tenure greater than 13.5 -> is not on a month-to-month contract -> has total charges less than 8678.63 -> has monthly charges less than 100.58.

See appendix 2 for decision tree diagram.

Part B: Ensemble Model

1. Model Requirements:

For this part, the requirements imposed were 'precision at least 80% and recall at least 65%'.

2. Data processing

Same as Part A 2 from above.

3. Train test split:

70% of the data was used to train the model and 30% of the data was used as the validation set.

4. Searching for the best model

Three different models were applied to try and find the best model using ensembling methods. (1) Random Forest, (2) AdaBoost Classifier, (3) A voting classifier consisting of (1) and (2).

Due to the number of parameters that needed to be searched RandomSearchCV was used for all three models, then GridSearchCV was used to try and narrow the search for optimal parameters.

Table 5: Ensembling model results

Classifier	Parameters searched	Parameters used	Train data results	Test data results	10-fold CV hold out set
Random Forest	**n_estimators = 50 to 400** **max_features = auto, sqrt** **max_depth = 4 to 9** **min_samples_leaf = 10 to 30**	**Max_depth = 8** **Max_features = sqrt** **Min_samples_leaf= 18** **N_estimators = 215**	**Score = 0.866** **Precision = 0.788** **Recall = 0.671**	**Score = 0.875** **Precision = 0.805** **Recall = 0.709**	**Score = 0.869** **Precision = 0.801** **Recall = 0.684**
AdaBoost (Decision tree)	Max_depth = 1-7 Min_samples leaf = [50, 55, 60, 65 ,70] N_estimators = 100, 110, 120, 125, 130 Learning_rate = 0.1 - 1	N_estimators = 120 Max_depth = 1 Min_samples_leaf = 60 Learning_rate = 0.3	Score = 0.862 Precision = 0.769 Recall = 0.688	Score = 0.866 Precision = 0.777 Recall = 0.707	Score = 0.859 Precision = 0.769 Recall = 0.671
Voting Classifier (Random forest (rf) + AdaBoost (abc))	'abc__base_estimator__max_depth': [1,3,5,7,9], 'abc__base_estimator__min_samples_leaf': [30,60,120,150,180], 'abc__n_estimators': [40, 80,120,160,200, 240], 'abc__learning_rate': [0.05, 0.1, 0.4, 0.6, 1.0], 'rf__max_depth': [2,4,6,8,10], 'rf__max_features':['auto', 'sqrt'], 'rf__n_estimators': [50,100,150,200], 'rf__min_samples_leaf': [15,18,25,30]}	{'rf__n_estimators': 200, 'rf__min_samples_leaf': 25, 'rf__max_features': 'auto', 'rf__max_depth': 6, 'abc__n_estimators': 80, 'abc__learning_rate': 0.05, 'abc__base_estimator__min_samples_leaf': 180, 'abc__base_estimator__max_depth': 5}	Score = 0.863 Precision = 0.787 Recall = 0.656	Score = 0.862 Precision = 0.809 Recall = 0.654	Score = 0.860 Precision = 0.788 Recall = 0.671

Observations:

- All 3 models with obtained similar accuracy scores however, precision and recall varied slightly.
- Within each model, due to randomness the precision and recall scores varied considerably and even after using cross-validation to evaluate the model, the scored would continue to be quite varied.
- Due to limited processing power and time constraints, the absolute optimal parameters were not found but instead, approximates to optimal parameters were found using RandomSearchCV.

5. Best Model

The model chosen as the best model here is the Random Forest model. It was simple in its implementation, parameter searching took the least amount of time and cross-validation scores on the hold-out set were the highest when it came to precision and recall.

6. Model Evaluation

Table 6: Best model- Random forest - scores

Accuracy = 0.869	Recall = 0.684	Precision = 0.801

Confusion Matrix

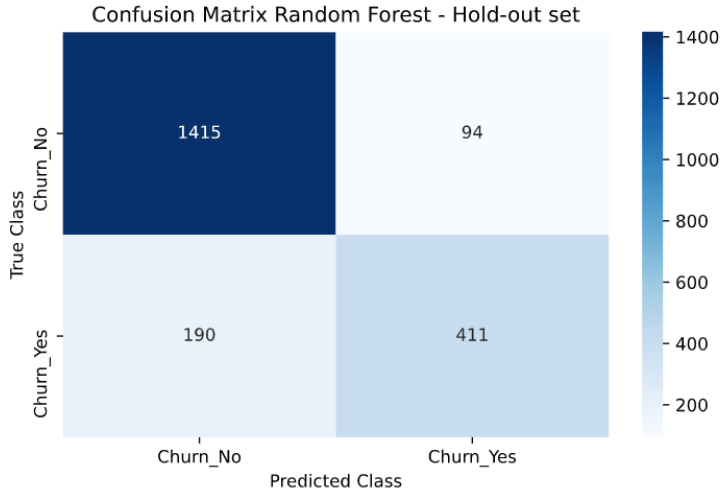

Figure 2: Confusion matrix - Random Forest

The model had a prediction accuracy of 89.6%. From a total of 2110 observations there were 190 false negatives (falsely predicting a customer would stay when in fact they left) and 94 false positives (falsely predicting a customer would leave when in fact they stayed). Out of the three models above, this model had the highest recall score of the three models.

7. Conclusion and further recommendations

Part A showed that a single decision tree could make for a decent interpretable model with a recall of 70.3% and a precision of 75.7%. The issue with a single decision tree was that the variance of the recall and precisions were very high, which agrees with the literature about decision trees begin low bias and high variance models. Part B with an Ensembling model was able to reduce some of this variance and came out with a random forest model with a recall of 68.4% and precision of 80.1%. This show the precision recall trade-off and does indeed meet the set requirements for the model of 80% precision and 65% accuracy.

Furthermore, as stated in Part A, it would be wise to optimise for recall (lowering the false negative predictions) as this would help predict potential customers who would leave. This would then allow marketing and strategies to be directed towards these customers to prevent them from leaving.

For improving the models, further research and work into feature selection and feature engineering would greatly benefit the model building process as hyperparameter searching can only get you so far!

Appendices:

Appendix 1: Decoding features for the rules in part 1.

Rules:

1. X[3] <=0.5 FALSE ---> X[34] <=0.5 FALSE ---> X[43]<=15.5 TRUE ---> X[43]<=3.5 TRUE ---> X[28]<=0.5 TRUE LEAVE
2. X[3] <=0.5 FALSE ---> X[34] <=0.5 FALSE ---> X[43]<=15.5 TRUE ---> X[43]<=3.5 TRUE ---> X[28] FALSE LEAVE
3. X[3] <=0.5 FALSE ---> X[34] <=0.5 FALSE ---> X[43]<=15.5 TRUE ---> X[43]<=3.5 FALSE ---> X[44]<=83.4 FALSE LEAVE
4. X[3] <=0.5 TRUE ----> X[43] <=13.5 TRUE ---> X[14] <0.5 TRUE ---> X[43] <= 3.5 FALSE ---> X[42] <= 0.5 FALSE STAY
5. X[3] <=0.5 TRUE ----> X[43] <=13.5 TRUE ---> X[14] <0.5 TRUE ---> X[43] <= 3.5 FALSE ---> X[42] <= 0.5 TRUE STAY
6. X[3] <=0.5 TRUE ----> X[43] <=13.5 FALSE ---> X[34]<=0.5 TRUE ---> X[45] <=8678.63 TRUE ---> X[44] <=100.58 TRUE STAY

Feature codes:

X[3] - senior > 0.5, not senior <= 0.5

X[34] – on a month-to-month contract >0.5, no if <=0.5

X[14] – has fiber optic >0.5, no if <=0.5

X[43] – tenure

X[28] – No streamingTV >0.5, streaming TV <=0.5

X[44] – MonthlyCharges

X[45] – TotalCharges

X[42] – mailed check > 0.5, not mailed check <=0.5

Appendix 2: Decision Tree for Part A (see page below)