Assignment 2

Alexander Dunbar
Dept of Mathematics
University of New South Wales
Sydney, Australia
alex.b.dunbar@gmail.com

***Abstract*—Decision Tree analysis of churn data**
***Index Terms*—decision tree, machine learning, ensemble, random forest, classifier**

I. Introduction

This paper explores the machine learning techniques and analysis of the decision tree classification algorithm and associated ensemble methods.

II. Understanding the data set

a) Brief description: The data-set is from the IBM Sample Data Sets and consists of information regarding customers telecommunications (telco) services. Individual customers are represented by an instance of the data while the feature columns represent, for example, the services that they have or have not signed up for. Also included in the features are account information, demographic information and whether the customer has left the telco in the last month, titled 'Churn'. Churn are the labels that will be used in the classification. The context of the data is to "Predict behavior to retain customers". [1]

b) Labels: There are 21 features (attributes) in the raw data file including the labels. The labels consist of either 'yes' or 'no' with an unequal distribution of 1869 (yes) and 5163 (no). That represents a percentage churn of 26.6%. This means that a uniformly random selection will predict no churn 73.7% of the time.

c) Features: The 20 remaining features consist of 16 categorical and three numerical. The 3 numerical features ('tenure', 'MonthlyCharges' and 'TotalCharges') are transformed to categories by binning.

d) Subset: 'customerID' is an unique identifier for each customer (instance) and is dropped from the ongoing analysis.

e) Train Test Split: The data set (final sub set) was split into a training set and a test set. Of the 7031 instances, 20% were set aside for final testing. The test set was not seen in the training or validation.

training shape: (5625, 30), testing shape: (1407, 30)
training labels: (5625,), testing labels: (1407,)

III. Task-1: White Box Models

The so-called 'white box model' describes a model whereby "if a given situation is observable in a model, the explanation for the condition is easily explained by boolean logic". Compared to a 'black box model' where "results may be difficult to interpret". [2]

A. Preparing the data for computational analysis

a) Exploratory Data Analysis: Each feature was plotted in a pie chart. A comparison of two pie charts for each feature is shown in figure 1. The first pie chart shows the breakdown of each feature. The second pie chart is the breakdown according to 'Churn'=='yes'. One particularly interesting feature is 'SeniorCitizen'. Out of all the instances nearly 30% were identified as Senior Citizen but of all the instances that fell into the Churn=yes, over 72% of those were Senior Citizen. As it turned out, 'SeniorCitizen' became the first node to split. The following table describes the breakdown.

B. Methodology

a) Investigation: This paper investigates the use and effectiveness of Decision Tree classifiers in order to make predictions on customer likelihood to leave a particular telecommunication service. In part I we explore a 'white box' decision tree classifier and compare that to a 'black box' classification in part II.

b) Approach: The investigation will utilise the classification algorithms from the Scikit-Learn library. Scikit-Learn provides many classifiers and the ability to 'tune' the results by adjusting the built-in parameters. Scikit-Learn provides many tools to test and tune the results ultimately leading to a better classifier on previously unseen (untrained) data.

c) Evaluation: The evaluation of the model and ultimate determination of the 'best' model is based on comparison of metrics between the training data set and a test data set. In order to obtain a 'best' model certain trade-off factors (e.g. bias-variance trade-off, under-fitting vs over-fitting) will need to be incorporated to obtain the 'best' metrics from both training and test data sets. In Part I we will use the confusion matrix to evaluate the accuracy of the model.

d) Confusion matrix: The confusion matrix consists of rows of the true labels ('yes' or 'no') by columns of the predicted labels ('yes' or 'no').

C. Building and Evaluating Models

a) Building models: Models with no added parameter specifications are termed untuned, while models with parameters specified are termed tuned. In the 'white box model' we are using a Decision Tree Classifier. The untuned model consists of a RegressionTreeClassifier instance with no parameters set. The algorithm will continue splitting the data until a pure leaf is obtained. This will tend to produce over-fitted models

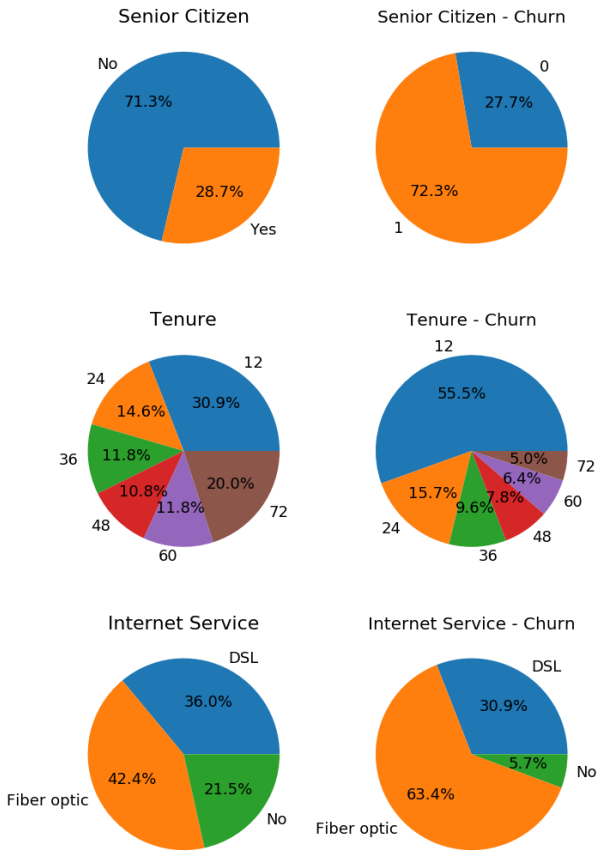

Fig. 1. The left column shows a particular attribute in a breakdown of all the data. The right column is the same attribute in a breakdown for when churn is true ('Churn'=='Yes'). Charts are ordered top to bottom in order of feature importance (the first three). Note. Tenure labels refer to months.

where the training data set has a very high accuracy while the test data set has a low accuracy. For the tuned model Scikit-Learn provides a library ('GridSearchCV') that can be used to search through combinations of parameter values that are set by the user 5

	UNTUNED		TUNED	
	Train	**Test**	**Train**	**Test**
Accuracy	0.979	0.854	0.854	0.860
Precision	0.609	0.758	0.758	0.760
Recall	0.596	0.640	0.640	0.668

TABLE I
ACCURACY OF UNTUNED AND TUNED DECISION TREE CLASSIFIERS

b) Evaluation of Untuned: The untuned training data displayed the evidence of over-fitting. The accuracy was high for the training set (0.979) and low for the test set (0.79). The precision and recall all lower for the untuned training set compared to the test set.

c) Evaluation of Tuned: The tuned training data displayed an improved training data error compared to the test data (0.854 vs 0.86). The prediction for 'Yes' has improved by 17 and the prediction for 'No' has improved by 54. The resulting precision $\frac{TP}{TP+FP}$ has improved $\frac{250}{250+79} = 0.760$ from 0.758 and recall $\frac{TP}{TP+FN}$ has improved $\frac{250}{250+124} = 0.668$ from 0.640.

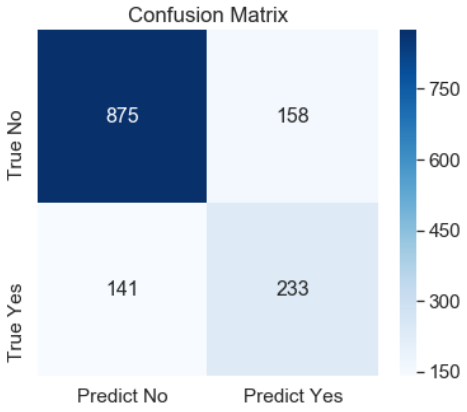

Fig. 2. Untuned (Test Data) Decision Tree Classifier

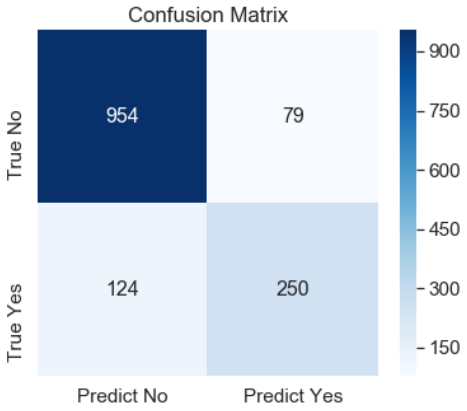

Fig. 3. Tuned (GridSearch) (Test Data) Decision Tree Classifier

d) Feature importance: The untuned model ultimately used all features whereas the tuned model only used six features (SeniorCitizen, tenure, InternetService, PaymentMethod, Contract, PaperlessBilling).

e) Derive Rules: The following rules are classed as interesting. It has a purity of >75% and contains at least 50 cases. See Table V.

D. Conclusion

A random selection of labels would produce a classification accuracy of 73%. The untuned raw model that was significantly over-fit produced an accuracy of 79% on the unseen test data and the tuned model produced an accuracy of 86% on unseen test data. In Part II we will investigate ensemble models to see if we can improve the accuracy further.

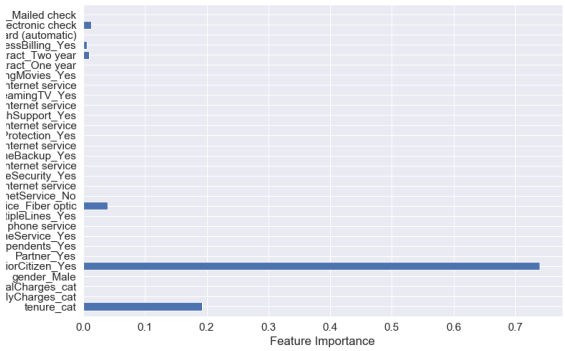

Fig. 4. Feature Importance for tuned Decision Tree Classifier

IV. Task-2: Black Box Models

A. Preparing the data for computational analysis

The data preparation from Part I was used without edit for Part II.

B. Methodology

a) Investigation: Investigate ensemble classifiers.

b) Approach: The approach will be the same as Part I. Utilise the Scikit-Learn library of classifiers and fine tune using the parameters.

c) Evaluation: Evaluate the models using cross validation.

C. Building and Evaluating Models

a) Building models: The following models were built and evaluated. Each model was run through cross-validation.

- Random Forest Classifier
- Grid Search CV Random Forest Classifier (RFC)
- Gradient Boosting
- Adaboost
- Bagging
- Voting Classifier (KNN, Logistic Regression, SVC, Gaus, RFC, Adaboost Classifier)
- Logistic Regression CV

Random Forest Classification		
Model	**Data set**	**Accuracy**
RFC()	Training	0.9671
RFC()	Test	0.8372
RFC(best params)	Training	0.8514
RFC(best params)	Test	0.8557
Gradient Boost	Training	0.8631
Gradient Boost	Test	0.8643
Adaboost	Training	0.8860
Adaboost	Test	0.8536
Bagging (RFC())	Training	0.8459
Bagging (RFC())	Test	0.8500
Bagging (RFC() best params)	Training	0.7342
Bagging (RFC() best params)	Test	0.7342

TABLE II
Testing Ensemble Classification Algorithms

b) Random Forest Classifier:

Voting Classifier	
Model	**Accuracy**
KNeighborsClassifier	0.828
LogisticRegression	0.860
SVC	0.8472
GaussianProcessClassifier	0.860
RandomForestClassifier	0.8564
AdaBoostClassifier	0.8557
VotingClassifier	0.8635

TABLE III
Voting Classifier

c) Voting Classifier:

Logistic Regression Classifier CV	
Data set	**Accuracy**
Training	0.9075
Test	0.9053

TABLE IV
Logistic Regression Classifier Cross-Validation

d) Logistic Regression Classifier CV:

D. Conclusion

The Logistic Regression Classifier CV was the best performing classifier with an accuracy on the test data of 90.53% IV. This was an improvement over the Decision Tree Classifier.

References

[1] https://community.ibm.com/community/user/businessanalytics/blogs/steven-macko/2019/07/11/telco-customer-churn-1113

[2] https://scikit-learn.org/stable/modules/tree.html/#tree

[3] https://scikit-learn.org/stable/#

APPENDIX

Node 1	Node 2	Node 3	Node 4	Label	Class 0	Class 1	Purity %
SeniorCitizen (no)	tenure (<1.5)	InternetService, Fiber optic (no)	PaperlessBilling (no)	Class 0	349	36	90.65
SeniorCitizen (no)	tenure (<1.5)	InternetService, Fiber optic (no)	PaperlessBilling (yes)	Class 0	236	68	77.63
SeniorCitizen (no)	tenure (>1.5)	PaymentMethod, Electronic check (no)		Class 0	2280	82	96.53
SeniorCitizen (no)	tenure (>1.5)	PaymentMethod, Electronic check (yes)	tenure (<2.5)	Class 0	479	62	88.54
SeniorCitizen (yes)	tenure (>5.5)	Contract, Two year (yes)		Class 0	102	21	82.93
SeniorCitizen (yes)	tenure (<5.5)	tenure (<1.5)		Class 1	68	532	88.67

TABLE V
INTERESTING FEATURES. THOSE HAVING A PURITY GREATER THAN 75% AND ITEMS GREATER THAN 50.

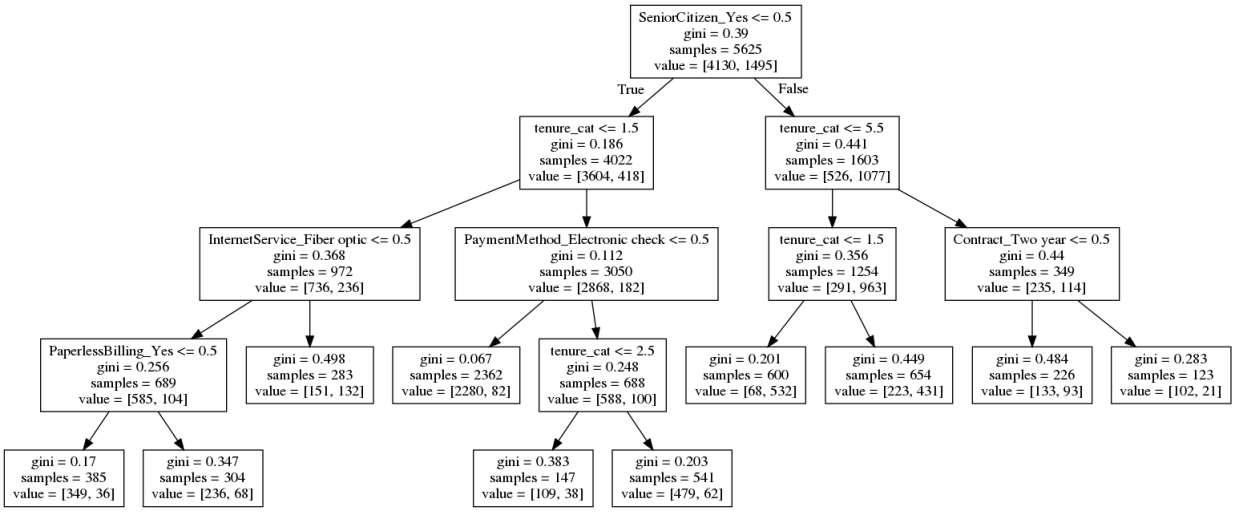

Fig. 5. Final model Decision Tree Classifier