**Introduction**

The purpose of this assignment is to inform a marketing plan that will target future visitors to one of the Wisconsin Dells seven attractions referenced in the Harrington (2007) case study. This study will outline a specific set of recommendations to drive more visitors to “The Ducks” tour. A supervised learning model was developed using a 1,698 in-person survey conducted at various locations and attractions across the Wisconsin Dells area to predict those visitors who will take the Duck tour given a specific set of attributes.

The predictors and response variables are as follows (from Exhibit 1):

nnights = length of stay  
nadults = number of adults in party  
nchildren = number of children under 18 in the party  
planning = how far in advance the vacation was planned  
sex = sex of survey respondent  
age = survey respondent age category  
education = highest level of education completed  
income = level of total household income  
region = zip code of region  
rideducks = yes/no if a visitor would take the Duck tour (response)

An EDA is conducted to evaluate the predictive accuracy of the attributes of the variables to determine those that are appropriate for the model. A classification tree is then used to determine the attributes associated with visitors who are likely to take The Ducks tour.

**Results**

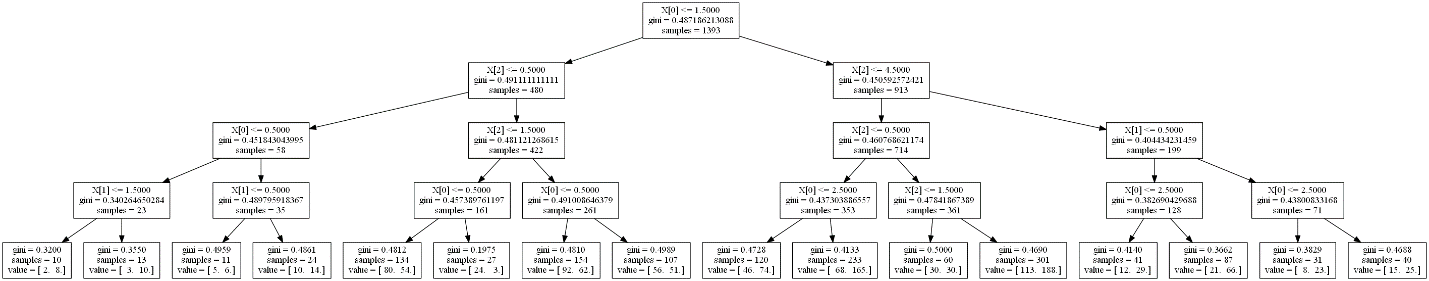
The first step of the study conducts an Exploratory Data Analysis (EDA) by calculating a distribution for each of the categorical variables to determine those variables that may be valid for predicting those visitors most likely to ride the duck tour.

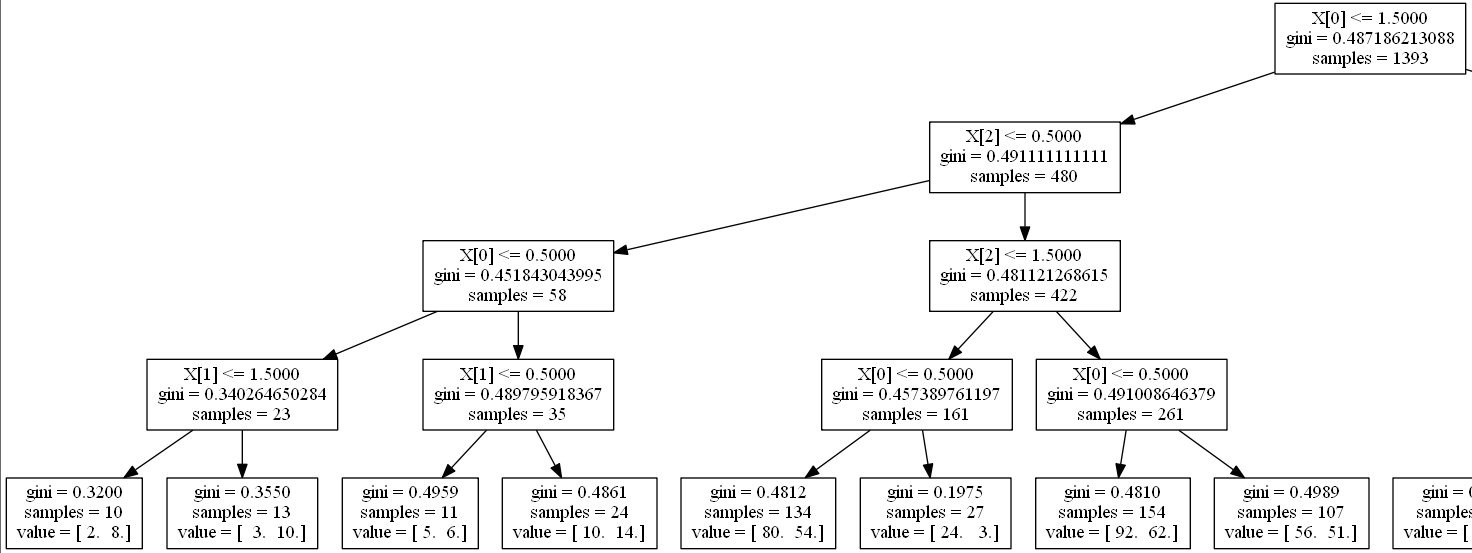
* age – the age of survey respondent had minimal variation on those visitors that took the Duck tour and therefore this variable will not be used in the model.
* sex – the sex of the survey respondent does not show a significant difference in those that did or did not take the Ducks tour so it will not be used in the model.
* education – there are imbalance showing correlations between age groups that are more likely to ride the ducks.
* income – the amount of household income had minimal variation across the categories on those groups that did and did not take the Ducks tour. This variable will not be used in the model
* region – there are imbalances showing correlations between certain regions that are more likely to ride the ducks. This variable will be used in the model.
* nchildren – there is variation across the distribution that the number of children influences those that took the Ducks tour so this variable will be used in the model.
* nadults – the number of the adults in the group that take the Ducks tour has minimal variation across the categories and therefore will not be used in the model.
* nnights – there is a strong correlation that visitors that are staying 2 or more nights are more likely to ride the ducks. This variable will be used in the model.
* planning – there is a correlation that the more days that the vacation is planned in advance the more likely visitors are to take the Ducks tour.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  | |

The model was initially run with selected variables above and the cross validation mean was 60.6%. Different permutations of all of the variables were modeled to determine if the prediction could be improved. The highest predictive accuracy was obtained using the three predictors of nnights, region and planning. The model accuracy was also slightly improved by setting the max tree height to 4 levels. A ten-folds cross-validation mean accuracy was 61.6%.

X[0] = nights, X[1] = planning, X[2] = region



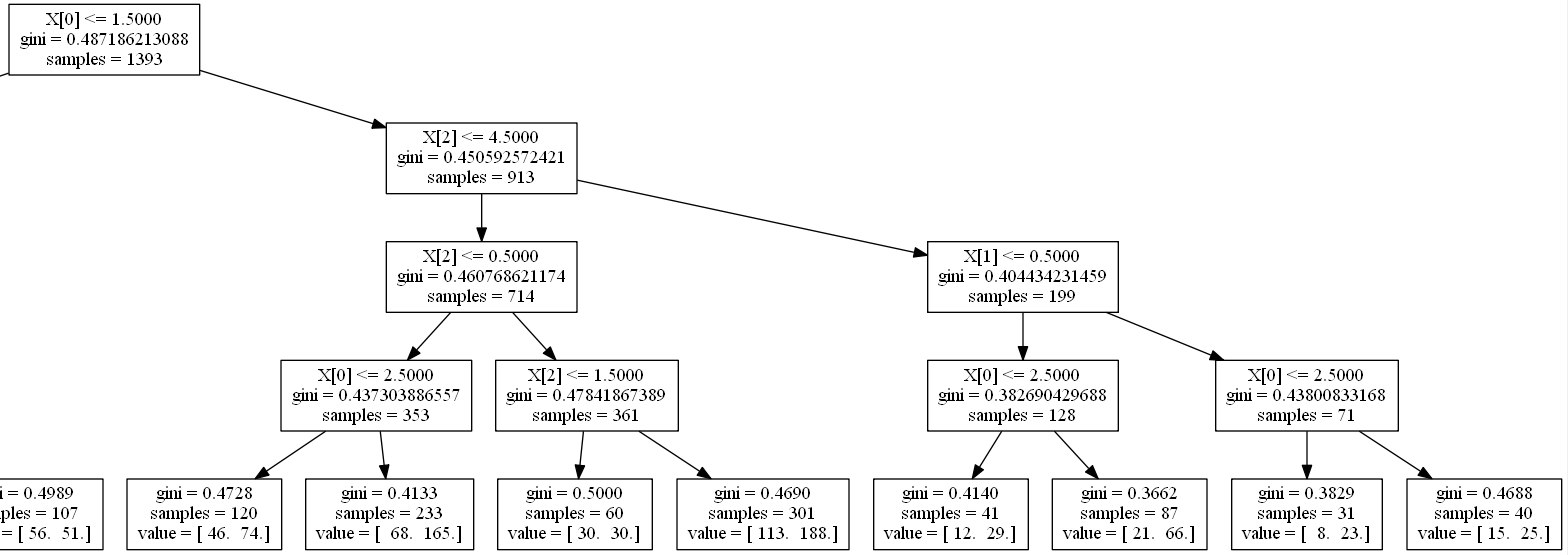


Planned this month or longer

0 nights

Chicago

1 or less nights



Chicago, Madison, Milwaukee, Minn., Wisconsin

Chicago

2 or less nights

Planned more than a month

Other regions

2 or more nights

3 or more nights

**Conclusions**

We can infer a few basic conclusions that will guide The Ducks marketing plan. The leaf nodes in the tree that are highlighted by the red boxes with the lower gini indices are combinations of factors that increase the probability that visitors will take the Duck tour. Recommendations for the marketing plan include:

* Heavily target visitors from the Chicago region.
* Make campaign offers for Chicago residents who will be making a day trip for date a month or more out.
* Target visitors from Chicago who will be staying 3 or more nights.
* Target other regional areas beyond Chicago, Madison, Milwaukee, Minnesota and Wisconsin for visitors who would stay for 2 or less nights.

**Code**

# Assignment 1: James Gray - graymatter@u.northwestern.edu - July 12, 2014

#

# This Python program will use the Wisconsin Dells case study data to target

# potential Wisconsin Dells visitors to "The Ducks" tour.

# here are a few of the packages we rely upon for work in predictive analytics

# import os # operating system module

# import pandas as pd # pandas for data frame operations

# import numpy as np # arrays an math functions

# import scipy as sp # statistics and more for science

# import sklearn as sk # machine learning tools

# import statsmodels.formula.api as smf # for working with R-style formulas

# import statsmodels.api as sm # for working with numpy arrays

# import matplotlib.pyplot as plt # 2D plotting

# import networkx as nx # software for network analysis

# these modules should prove especially useful in this first assignment

import pandas as pd # pandas for data frame operations

import numpy as np # arrays an math functions

import sklearn as sk # machine learning tools

import matplotlib.pyplot as plt # 2D plotting

# use scikit-learn as our toolset for machine learning

import sklearn as sk

pd.options.display.max\_rows = 20 # set max rows for Pandas output

# -----------------------------------------------------------------------------

# DATA PREPARATION NOTES

# -----------------------------------------------------------------------------

#

# Data from the Wisconsin Dells case study were collected as part of a

# survey study in the summer of 1995. The study and survey data are described in

# Harrington, J. C. (2007). Wisconsin Dells. Madison, Wisconsin:

# Research Publishers.

#

# the work below uses pandas... which provides the DataFrame data structure,

# a DataFrame is a 2-dimensional labeled data structure with

# columns of potentially different types... like an R data frame or SQL table

# -----------------------------------------------------------------------------

# READ IN DATA... Wisconsin Dells survey data

# -----------------------------------------------------------------------------

# read in Wisconsin Dells survey data after we ensure that

# the comma-delimited text file is in our working directory

# this creates a pandas DataFrame object

wi\_dells\_data = pd.read\_csv("wisconsin\_dells.csv")

print("\nContents of wi\_dells\_data object ---------------")

# examine the structure of this DataFrame object

print(wi\_dells\_data)

# -----------------------------------------------------------------------------

# Focus on the Wisconsin Dells visitors who "rideducks"

# -----------------------------------------------------------------------------

# to see the data we need to do a little work due to the large number of columns

# documentation for what we are doing here is available at

# http://pandas.pydata.org/pandas-docs/stable/basics.html?highlight=set\_option

pd.set\_option('display.max\_columns', 11) # to allow at most 11 columns of data output

# the Wisconsin Dells demographic data are provided in the first ten columns

# let's select those columns along with the column for rideducks,

# this will give us 11 columns of data for our analysis

# let's use df as a shorthand for our working data frame

# a Python list is used to specify the columns of interest

df = pd.DataFrame(wi\_dells\_data,

columns = ["id", "nnights", "nadults", "nchildren", "planning",

"sex", "age", "education", "income", "region", "rideducks"])

print("\nContents of the working data frame df---------------")

# examine the structure of this DataFrame object

print(df)

# print the first five rows of the DataFrame

print(pd.DataFrame.head(df)) # note NaN values for missing data

# -----------------------------------------------------------------------

# Data Cleansing and Transformation

# -----------------------------------------------------------------------

# Remove rows with missing data

df\_complete = df.dropna()

print(df\_complete.shape) # inspect how many rows were deleted

# Transform the binary target into numerical data using Dictionary

ducks\_mapper = {'NO': -1,'YES' : 1}

y = df\_complete['rideducks'].map (ducks\_mapper)

print("\nTarget variable array")

print(y[0:10])

# Transform explanatory nominal vars into numerical data using Dictionary

# Use Dictionary to store mapping

# Create new DF

sex\_mapper = {'Female': 0, 'Male': 1}

sex = df\_complete['sex'].map(sex\_mapper) #new DF to hold sex

print (sex[0:10]) # inspect sex transformation

age\_mapper = {'LT 25' : 1,

'25-34' : 2,

'35-44' : 3,

'45-54' : 4,

'55-64' : 5,

'65+' : 6}

age = df\_complete['age'].map(age\_mapper)

print (age[0:10])

education\_mapper = {'HS Grad or Less':0,

'Some College': 1,

'College Grad': 2,

'Post Grad':3}

education = df\_complete['education'].map(education\_mapper)

income\_mapper = {'Lower Income':0,

'Middle Income': 1,

'Upper Income': 2 }

income = df\_complete['income'].map(income\_mapper)

planning\_mapper = { 'One Month or More Ago': 0,

'This Month': 1,

'This Week': 2}

planning = df\_complete['planning'].map(planning\_mapper)

children\_mapper = {'No kids': 0,

'1': 1,

'2': 2,

'3': 3,

'4': 4,

'5+': 5}

children = df\_complete['nchildren'].map(children\_mapper)

nights\_mapper = { '0': 0,

'1': 1,

'2': 2,

'3': 3,

'4+': 4}

nights = df\_complete['nnights'].map(nights\_mapper)

adults\_mapper = { '1': 1,

'2': 2,

'3': 3,

'4': 4,

'5+': 5}

adults = df\_complete['nadults'].map(adults\_mapper)

region\_mapper = { 'Chicago': 0,

'Madison': 1,

'Milwaukee':2,

'Minneapolis/StPaul':3,

'Other Wisconsin': 4,

'Other': 5 }

region = df\_complete['region'].map(region\_mapper)

# ------------------------------------------------------------------------

# Perform EDA to evaluate predictive accuracy of each categorical variable

#-------------------------------------------------------------------------

ducksbyage = pd.crosstab(df\_complete.age, df\_complete.rideducks)

ducksbyagepct = ducksbyage.div(ducksbyage.sum(1).astype(float), axis =0)

ducksbyagepct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by respondent age ---------------")

print (ducksbyagepct)

ducksbysex = pd.crosstab(df\_complete.sex, df\_complete.rideducks)

ducksbysexpct = ducksbysex.div(ducksbysex.sum(1).astype(float), axis =0)

ducksbysexpct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by respondent sex---------------")

print (ducksbysexpct)

ducksbyeducation = pd.crosstab(df\_complete.education, df\_complete.rideducks)

ducksbyeducationpct = ducksbyeducation.div(ducksbyeducation.sum(1).astype(float), axis =0)

ducksbyeducationpct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by respondent education -------------")

print (ducksbyeducationpct)

ducksbyincome = pd.crosstab(df\_complete.income, df\_complete.rideducks)

ducksbyincomepct = ducksbyincome.div(ducksbyincome.sum(1).astype(float), axis =0)

ducksbyincomepct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by Household Income -------------")

print (ducksbyincomepct)

ducksbyregion = pd.crosstab(df\_complete.region, df\_complete.rideducks)

ducksbyregionpct = ducksbyregion.div(ducksbyregion.sum(1).astype(float), axis =0)

ducksbyregionpct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by Region -------------")

print (ducksbyregionpct)

ducksbychildren = pd.crosstab(df\_complete.nchildren, df\_complete.rideducks)

ducksbychildrenpct = ducksbychildren.div(ducksbychildren.sum(1).astype(float), axis =0)

ducksbychildrenpct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by Children in party -------------")

print (ducksbychildrenpct)

ducksbyadults = pd.crosstab(df\_complete.nadults, df\_complete.rideducks)

ducksbyadultspct = ducksbyadults.div(ducksbyadults.sum(1).astype(float), axis =0)

ducksbyadultspct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by Adults in party -------------")

print (ducksbyadultspct)

ducksbynights = pd.crosstab(df\_complete.nnights, df\_complete.rideducks)

ducksbynightspct = ducksbynights.div(ducksbynights.sum(1).astype(float), axis =0)

ducksbynightspct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by Nights in stay -------------")

print (ducksbynightspct)

ducksbyplanning = pd.crosstab(df\_complete.nnights, df\_complete.rideducks)

ducksbynightspct = ducksbynights.div(ducksbynights.sum(1).astype(float), axis =0)

ducksbynightspct.plot (kind='barh', stacked = True)

print("\nDistribution of Ducks by vacation planned in advance -------------")

print (ducksbynightspct)

# gather all explanatory variables into a numpy array

# here we use .T to obtain the transpose for the structure we want.

# The EDA showed that region, nights, planning had the most

# significant impact on taking the Ducks tour

x = np.array([np.array(nights),

np.array(planning), np.array(region)]).T

print(x.shape) # check the structure of the array of explanatory variables

print("\nInput variable array")

print(x[0:10,]) # examine the first 10 rows of data

# -----------------------------------------------------------------------------

# build a simple tree classifier for these data

#------------------------------------------------------------------------------

# define modeling method with random number seed for reproducibility

from sklearn import tree

model = tree.DecisionTreeClassifier(min\_samples\_split = 20, min\_samples\_leaf = 10,

max\_depth = 4, random\_state = 9999)

my\_tree = model.fit(x, y) # defines tree classifier object

# use tab completion to see methods available for this object

# my\_tree.<tab>

# use my\_tree? to obtain information about the object

# multi-fold cross-validation with ten folds to calculate predictive accuracy

from sklearn.cross\_validation import cross\_val\_score

cv\_results = cross\_val\_score(model, x, y, cv=10)

print(round(cv\_results.mean(),3)) # cross-validation average proportion correct

print (cv\_results)

# lets see what the fitted tree looks like

from sklearn.externals.six import StringIO

with open('ducks.dot', 'w') as f:

f = tree.export\_graphviz(my\_tree, out\_file=f)

**References**

Harrington, J. C. (2007). *Wisconsin Dells.* Madison, Wisconsin: Research Publishers.