7th June 2016

Machine Learning: First Assignment - Decision Trees

Machine Learning - First Assignment - Instructions:

This week's assignment involves decision trees, and more specifically, classification trees. Decision trees are predictive models that allow for a data driven exploration of nonlinear relationships and interactions among many explanatory variables in predicting a response or target variable. When the response variable is categorical (two levels), the model is a called a classification tree. Explanatory variables can be either quantitative, categorical or both. Decision trees create segmentations or subgroups in the data, by applying a series of simple rules or criteria over and over again which choose variable constellations that best predict the response (i.e. target) variable.

Run a Classification Tree.

You will need to perform a decision tree analysis to test nonlinear relationships among a series of explanatory variables and a binary, categorical response variable.

What to Submit:

Following completion of the steps described above, create a blog entry where you submit syntax used to run a Classification Tree (copied and pasted from your program) along with corresponding output and a few sentences of interpretation. Please note that your reviewers should NOT be required to download any files in order to complete the review.

Review Criteria

Your assessment will be based on the evidence you provide that you have completed all of the steps. When relevant, gradients in the scoring will be available to reward clarity (for example, you will get one point for submitting an inaccurate or incomplete description of your results, but two points if the description is accurate and complete). In all cases, consider that the peer assessing your work is likely not an expert in the field you are analyzing. You will be assessed equally on all parts of the assignment, and whether you post your program and output.

Your assessment will be based on the evidence you provide that you have completed all of the steps. In all cases, consider that the peer assessing your work is likely not an expert in the field you are analyzing.

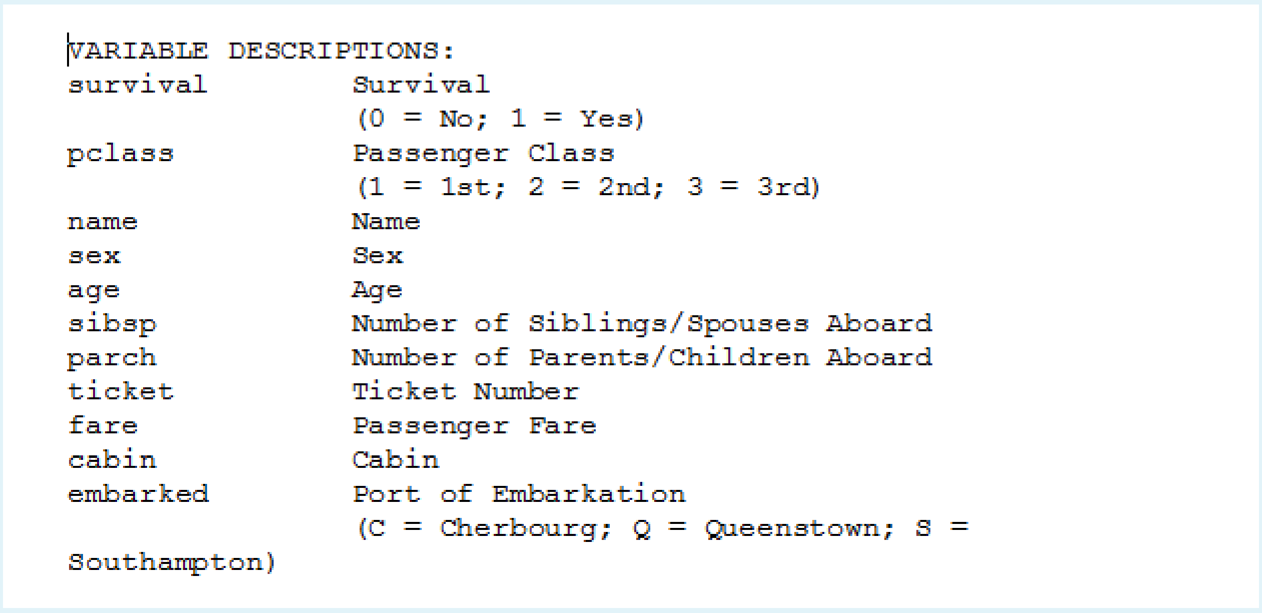
My Submission

[https://3.bp.blogspot.com/-L5_0-VaOpwk/V1ZZV3RhaCI/AAAAAAAABYE/5rLLcMLi1s0kikZTpStZdbhpBgEP1f4PgCLcB/s1600/800px-RMS_Titanic_3.jpg]

RMS Titanic was a British passenger liner that sank in the North Atlantic Ocean in the early morning of 15 April 1912, after colliding with an iceberg during her maiden voyage from Southampton to New York City. Of the 2,224 passengers and crew aboard, more than 1,500 died in the sinking, making it one of the deadliest commercial peacetime maritime disasters in modern history.

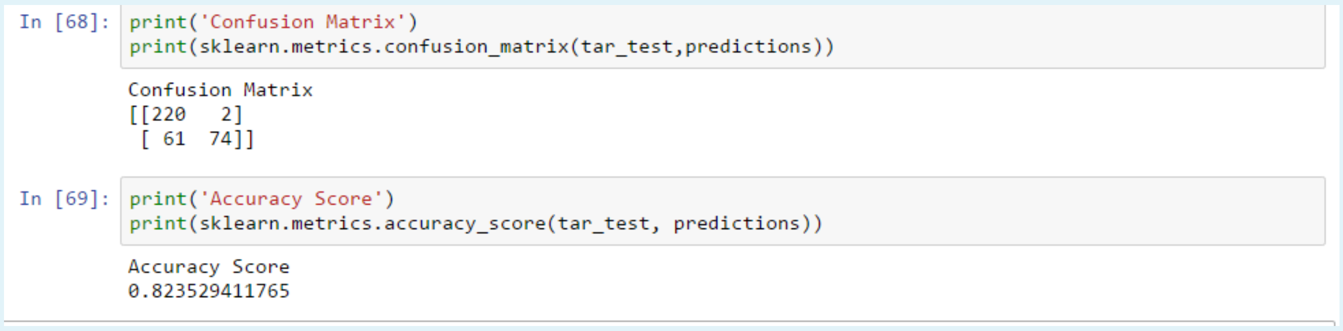
kaggle.com has a Titanic dataset in the 'Getting started' section along with some handy machine learning tutorials. I have been a member of kaggle for a year or so as I had to enter a competition as part of my first ever completed MOOC on Machine learning. Back then I had to learn R which seems very nice for machine learning but not much good for anything else. In the interim I have spent time learning some web development as part of a job requirement and Python because it seems a much better language for all purpose programming. I then enrolled in this course to learn about Machine Learning in Python and so now the circle closes as I get back to kaggle after a long break. My current kaggle ranking is 69,710th after hitting a high of 50,058th after my one and only competition. I intend using the next few lessons on this MOOC to begin pushing that higher.

The kaggle training set has 891 records with features as described below. For the purpose of this assignment I will see how a classification can predict survival using only Sex, Age and Passenger Class. I needed to create a categorical feature for sex which I called Gender (Female = 1, Male = 0) and I also split Age into three categories (Under 8 = 0, 8 - 16 = 1, over 16 = 2) simply to keep the picture of the tree nice and compact. Passenger class was naturally categorical with 1, 2 & 3 representing 1st, 2nd and 3rd class.



```
VARIABLE DESCRIPTIONS:
survival        Survival
                (0 = No; 1 = Yes)
pclass          Passenger Class
                (1 = 1st; 2 = 2nd; 3 = 3rd)
name            Name
sex             Sex
age             Age
sibsp           Number of Siblings/Spouses Aboard
parch           Number of Parents/Children Aboard
ticket          Ticket Number
fare            Passenger Fare
cabin           Cabin
embarked        Port of Embarkation
                (C = Cherbourg; Q = Queenstown; S =
Southampton)
```

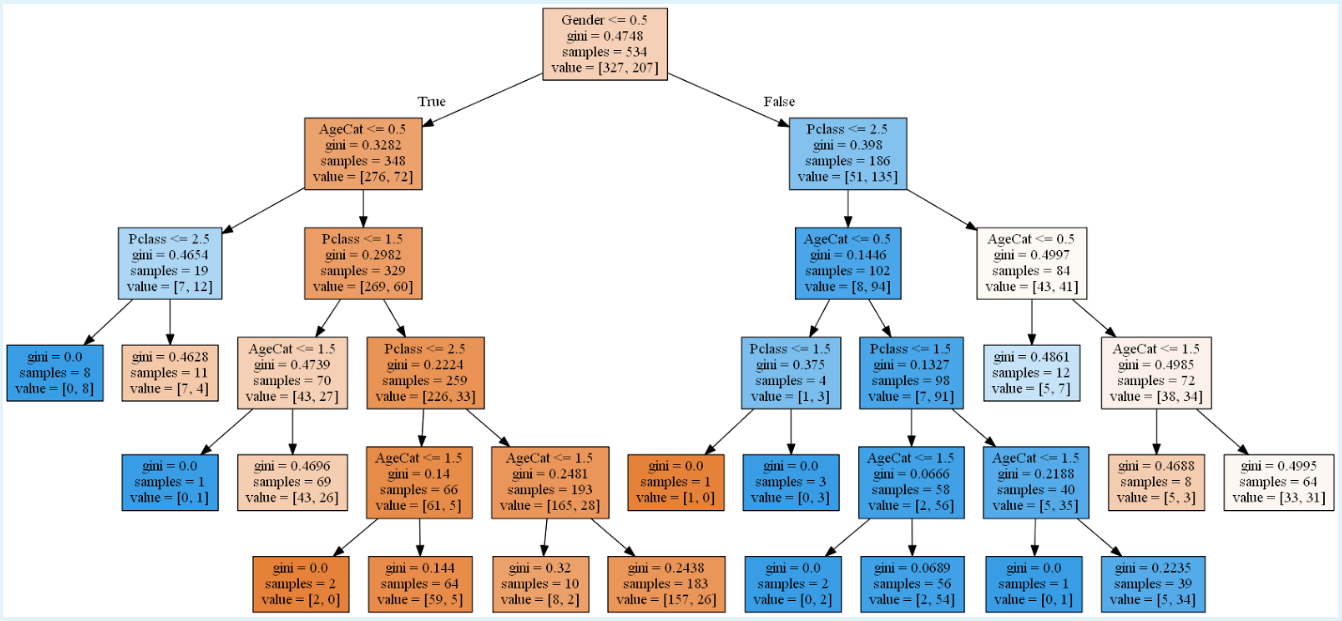
[https://4.bp.blogspot.com/-bg57jaQ3afY/V1ZZtn6mcLI/AAAAAAAABYM/TNzIKARYiU4aZNJr8Wj0ToDbe_qV0UZaQCLcB/s1600/DataDict.PNG]



```
In [68]: print('Confusion Matrix')
         print(sklearn.metrics.confusion_matrix(tar_test,predictions))

         Confusion Matrix
         [[220   2]
          [ 61  74]]

In [69]: print('Accuracy Score')
         print(sklearn.metrics.accuracy_score(tar_test, predictions))

         Accuracy Score
         0.823529411765
```

[https://2.bp.blogspot.com/-lIOWs6SG9qI/V1ZaSsbmwQI/AAAAAAAABYg/VTXktItVcVQfyfSEJNXH6iPYPmyTwz7ogCLcB/s1600/Conf%2Band%2BAccuracy.PNG]

The performance of even this simple model was quite reasonable with around 80% accuracy and correctly predicting 294 of the 357 test cases. One can see from the training set that of 534 passengers, 348 were male but only 72 of those survived while out of 207 female passengers 135 survived. At least 134 of the 207 survivors were 1st class passengers. Apparently the lifeboats were loaded on a women and children first basis but also they were often not filled leaving many passengers in the water around 2.5 hours after the accident. It is beleived that the high ratio of 1st class survivors is due to those cabins being on the upper decks of the ship.


Gender <= 0.5 gini = 0.4748 samples = 534 value = [327, 207]
True
False
AgeCat <= 0.5 gini = 0.3282 samples = 348 value = [276, 72]
Pclass <= 2.5 gini = 0.398 samples = 186 value = [51, 135]
Pclass <= 2.5 gini = 0.4654 samples = 19 value = [7, 12]
Pclass <= 1.5 gini = 0.2982 samples = 329 value = [269, 60]
AgeCat <= 0.5 gini = 0.1446 samples = 102 value = [8, 94]
AgeCat <= 0.5 gini = 0.4997 samples = 84 value = [43, 41]
gini = 0.0 samples = 8 value = [0, 8]
gini = 0.4628 samples = 11 value = [7, 4]
AgeCat <= 1.5 gini = 0.4739 samples = 70 value = [43, 27]
Pclass <= 2.5 gini = 0.2224 samples = 259 value = [226, 33]
Pclass <= 1.5 gini = 0.375 samples = 4 value = [1, 3]
Pclass <= 1.5 gini = 0.1327 samples = 98 value = [7, 91]
gini = 0.4861 samples = 12 value = [5, 7]
AgeCat <= 1.5 gini = 0.4985 samples = 72 value = [38, 34]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.4696 samples = 69 value = [43, 26]
AgeCat <= 1.5 gini = 0.14 samples = 66 value = [61, 5]
AgeCat <= 1.5 gini = 0.2481 samples = 193 value = [165, 28]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 3 value = [0, 3]
AgeCat <= 1.5 gini = 0.0666 samples = 58 value = [2, 56]
AgeCat <= 1.5 gini = 0.2188 samples = 40 value = [5, 35]
gini = 0.4688 samples = 8 value = [5, 3]
gini = 0.4995 samples = 64 value = [33, 31]
gini = 0.0 samples = 2 value = [2, 0]
gini = 0.144 samples = 64 value = [59, 5]
gini = 0.32 samples = 10 value = [8, 2]
gini = 0.2438 samples = 183 value = [157, 26]
gini = 0.0 samples = 2 value = [0, 2]
gini = 0.0689 samples = 56 value = [2, 54]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.2235 samples = 39 value = [5, 34]

[https://3.bp.blogspot.com/-OxOtG5SB3lQ/V1ZZ-zvf_dI/AAAAAAAABYY/Gmmdj0IzcewU0sfwX-o-KXUUdvHdQtU0wCLcB/s1600/NoiceTree.png]

Python code for logistic Regression

```
# coding: utf-8

'''
Decision tree to predict survival on the TtTanic
Written by Nick Kemp June 2016
'''

get_ipython().magic('matplotlib inline')
from pandas import Series, DataFrame
import pandas as pd
import numpy as np
import os
import matplotlib.pylab as plt
from sklearn.cross_validation import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report
import sklearn.metrics

titanic = pd.read_csv('train.csv')
```

```
print(titanic.describe())

print(titanic.head(5))

print(titanic.info())

# simplify Age

def catAge (row):
   if row['Age'] <= 8 :
      return 0
   elif row['Age'] <= 16 :
      return 1
   elif row['Age'] <= 60 :
      return 2
   else:
      return 2

# create categorical varioble for sex 1 = female, 0 = male
titanic['Gender'] = titanic['Sex'].map( {'female': 1, 'male': 0} ).astype(int)

# create age categories either 1 = Adult or 0 = child
titanic['AgeCat'] = titanic.apply(lambda row:catAge(row), axis = 1)

print(titanic.dtypes)

# get a clean subset
sub = titanic[['PassengerId', 'Gender','Pclass', 'AgeCat','SibSp', 'Survived']].dropna()
sub['Gender'] = sub['Gender'].astype('category')
sub['AgeCat'] = sub['AgeCat'].astype('category')
sub['Pclass'] = sub['Pclass'].astype('category')
print(sub.info())

# Split the data

predictors = sub[['Gender','Pclass', 'AgeCat','SibSp']]
targets = sub['Survived']

pred_train, pred_test, tar_train, tar_test  =   train_test_split(predictors, targets,
test_size=.4)

print(pred_train.shape, '\n',
pred_test.shape,'\n',
tar_train.shape,'\n',
tar_test.shape)

#Build model on training data
classifier=DecisionTreeClassifier()
classifier=classifier.fit(pred_train,tar_train)

predictions=classifier.predict(pred_test)

print('\nConfusion Matrix')
print(sklearn.metrics.confusion_matrix(tar_test,predictions))

print('Accuracy Score')
print(sklearn.metrics.accuracy_score(tar_test, predictions))

#Displaying the decision tree
from sklearn import tree
#from StringIO import StringIO
from io import StringIO
#from StringIO import StringIO
from IPython.display import Image

out = StringIO()
tree.export_graphviz(classifier, out_file=out, feature_names = ['Gender',
'Pclass','AgeCat','SibSp'],filled = True)

import pydotplus
graph=pydotplus.graph_from_dot_data(out.getvalue())
Image(graph.create_png())

# Split the data
predictors = sub[['Gender','Pclass', 'AgeCat']]
targets = sub['Survived']

pred_train, pred_test, tar_train, tar_test = train_test_split(predictors, targets,
test_size=.4)

print(pred_train.shape, '\n',
pred_test.shape,'\n',
tar_train.shape,'\n',
tar_test.shape)

#Build model on training data
classifier=DecisionTreeClassifier()
classifier=classifier.fit(pred_train,tar_train)

# Make prediction on the test data
predictions=classifier.predict(pred_test)

# Confusion Matrix
print('Confusion Matrix')
print(sklearn.metrics.confusion_matrix(tar_test,predictions))

#Accuracy
print('Accuracy Score')
print(sklearn.metrics.accuracy_score(tar_test, predictions))

# Show the tree
out = StringIO()
tree.export_graphviz(classifier, out_file=out, feature_names = ['Gender','Pclass',
'AgeCat'],filled = True)

# Save tree as png file
```

```
graph=pydotplus.graph_from_dot_data(out.getvalue())
Image(graph.create_png())

print(pred_train.describe())

# Observe various features of the traning data
print(titanic.Gender.sum())
print(titanic.Gender.count())
print(titanic.Gender.sum()/titanic.Gender.count())

# Table of survival Vs Gender
pd.crosstab(titanic.Sex,titanic.Survived)
```

Posted 7th June 2016 by Nick

0 Add a comment