Know Your Data, Classification, K-fold Cross Validation & Learning Curves

Mini Project Report

for

CSC 869 – Data Mining

Submitted to

Prof. Hui Yang

by

Rohan Samir Patel

SFSU ID: 917583698

Spring 2018

1. Overview

In this project we apply Naïve Bayesian and Decision Tree classifiers to a Census Income dataset and then use popular evaluation techniques like k-fold cross validation and learning curves to evaluate our machine learning model and diagnose whether these two classifiers suffer from any underfitting or overfitting symptoms. The main goal of this project is to follow a typical machine learning approach to solving any classification problem by first understanding our data through analysis and visualization, then applying the classification technique and then finally using a sound evaluation strategy to further improve our results.

The dataset used for this project is the Census Income or Adult dataset, which is available at http://archive.ics.uci.edu/ml/datasets/Adult. The dataset contains 48842 instances (train=32561, test=16281) that contain a mix of both continuous and discrete features. The features themselves are various demographic characteristics such as age, education, marital-status, occupation, race, sex, etc., and the class attribute is the income status (i.e. whether income is >50K or <=50K).

2. Methodology / Main Steps

The following section gives a brief description of the methodology and main steps that have been adopted to accomplish this task.

2.1 Know Your Data

The first step is to perform exploratory data analysis on our dataset with the aim of analysing and understanding the data better. We use Tableau to visualize and explore the relationship between each attribute and the class label, aiming to gain an initial understanding of the dataset. Also, the relationships between each pair of continuous attributes are visualized to see whether the dataset contains highly correlated attributes. Observations and insights from this step are covered in the final 'Evaluation and Analysis' section of this document.

2.2 Classification

Once we have a good understanding of the data, we apply Weka's implementation of the Naïve Bayes and Decision Tree classifiers (weka.classifiers.bayes.NaiveBayes and weka.classifiers.trees.J48 respectively) to perform the classification task. To do this, I use Python's wrapper for Weka (https://github.com/fracpete/python-weka-wrapper3) to call Weka's APIs to these two classifiers.

In this step, I load the training data from the 'adult.csv' file, build the two classifiers on the entire training set, and then use the classifiers to label the income status of each instance. Then I count the number of instances that have been classified into each class to simply get an initial idea of how the classifiers are performing.

2.3 K-fold Cross Validation

The next step is to implement a k-fold cross validation strategy to evaluate the performance of the Naïve Bayes and Decision Tree Classifiers on the Adult dataset with k=5 and 10, respectively. We first randomize the data, then we generate k different training folds and validation folds respectively and build our classifier on each training fold and test it on its subsequent validation fold. We then check the average error rate generated in the cross-validation to get an estimate of how well our classifier would predict in real practice.

2.4 Learning Curve

The next evaluation strategy we apply is the learning curve. Learning curves are used to diagnose bias and variance and thereby diagnose whether our classifier shows any underfitting or overfitting symptoms. The learning curve is generated by plotting the error vs. the training set size (i.e., how better does the model get at predicting the target as you the increase number of instances used to train it). It is important to note that in the learning curve, there are two error scores that are monitored: one for the validation set, and one for the training sets. We plot the evolution of the two error scores as training sets change and end up with two curves. The evolution of the two curves over the training set size gives us a good indication of bias and variance in our model.

2.5 Applying Classifier to Test Data

In the final step we simply apply the two learned classifiers (Naïve Bayes and Decision Tree) to the 'adult_test.csv' and report their precision, recall, F1-measure, and accuracy.

3. Instructions for Compiling

The project folder contains a total of four python scripts namely, classification.py, kfold_crossvalidation.py, learning_curve.py, and test.py that must be compiled to produce the final results. There are also two csv files, adult.csv and adult_test.csv that contain the training and test data respectively, and a Tableau workbook titled Book1.twb that contains visualization performed for '2.1 - Know Your Data' step. The step-wise instructions to compile and run the program are given below.

a) For this project, I have used *python-weka-wrapper3* that allows you to use Weka from within Python3. To compile and run all the scripts in this project you will first need to install *python-weka-wrapper3* on your system. The library has the following requirements:

- Python3 (does not work with Python 2)
- javabridge (>=1.0.14)
- Oracle JDK 1.8+

I have used the linux bash shell within windows for this project. If you are using Ubuntu or the linux bash shell in windows to execute this project, please follow the installation instructions given below to install *python-weka-wrapper3,* javabridge and Oracle JDK.

First, you need to be able to compile C/C++ code and Python modules. Use the following command:

```
$ sudo apt-get install build-essential python3-dev
```

Now, you can install the various packages that we require for installing python-weka-wrapper3:

```
$ sudo apt-get install python3-pip python3-numpy
```

Install OpenJDK as well, in order to get all the header files that javabridge compiles against (but don't use it for starting up JVMs):

```
$ sudo apt-get install default-jdk
```

Finally, you can use pip3 to install the Python packages that are not available in the repositories:

```
$ sudo pip3 install javabridge
```

```
$ sudo pip3 install python-weka-wrapper3
```

NOTE: If you are using Mac or any other OS for this project, please follow the installation instructions on http://fracpete.github.io/python-weka-wrapper3/install.html

Please also note, if you're using Mac, you might get an error "Python was unable to find JVM". In this case, please execute the following command in your terminal:

```
$ export JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_161.jdk/Contents/Home
```

b) Once you have installed *python-weka-wrapper3*, you can open your terminal and navigate to the project directory. The first script to execute is classification.py to perform the initial classification task. Use the following command:

```
$ python3 classification.py
```

c) Next you can execute the k-fold cross validation module which can be found in the kfold_crossvalidation.py script:

```
$ python3 kfold_crossvalidation.py k
```

where k = 5 or 10

d) Next you can execute the learning curve module which is present in the learning_curve.py script:

```
$ python3 learning_curve.py k
```

where k = 5 or 10

NOTE: In order to plot the learning curve, you will need python's matplotlib library installed. To install this, execute the command:

```
$ pip install matplotlib
```

e) The final step is to apply the two learned classifiers to the test dataset found in 'adult_test.csv'. Use the following command:

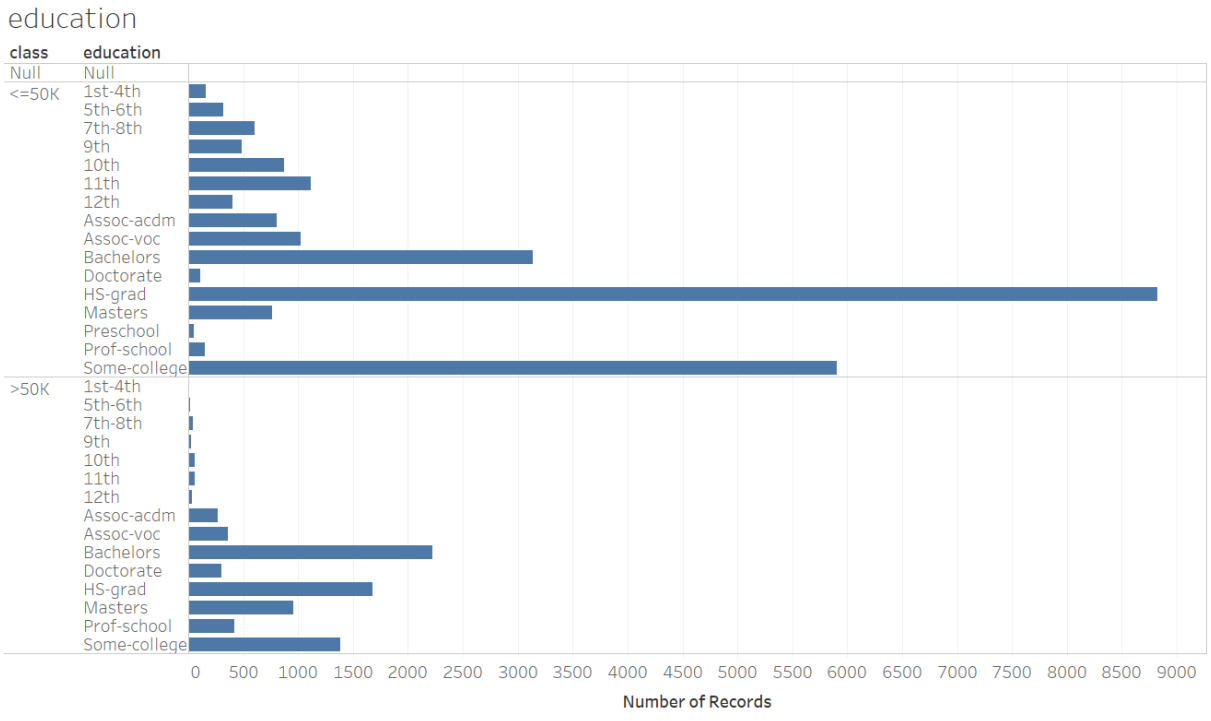
```
$ python test.py
```

4. Evaluation and Analysis

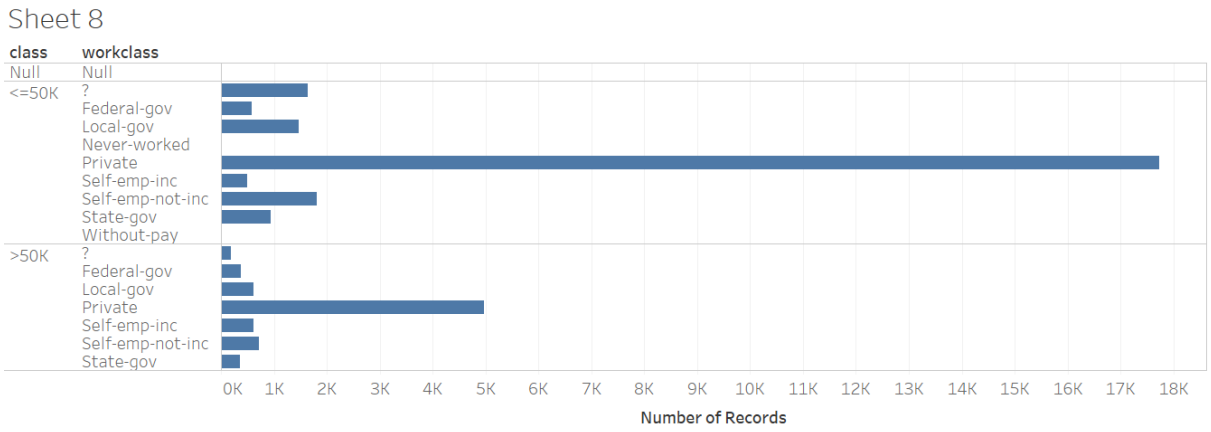
The following section covers a detailed discussion of the results produced.

4.1 Know Your Data

In the initial 'Know Your Data' step, we use Tableau to visualize and explore the relationship between various attributes in our dataset. Some of the key observations and insights are discussed below.

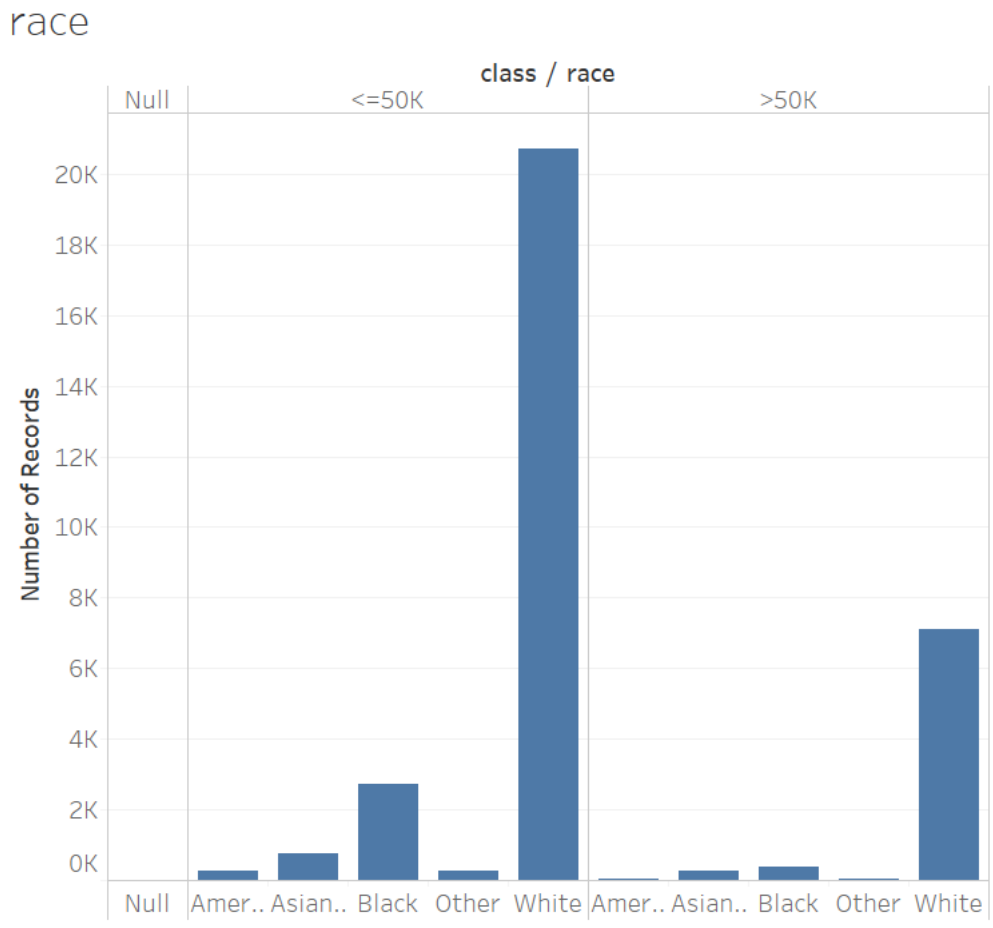


Sum of Number of Records for each education broken down by class.



Sum of Number of Records for each workclass broken down by class.

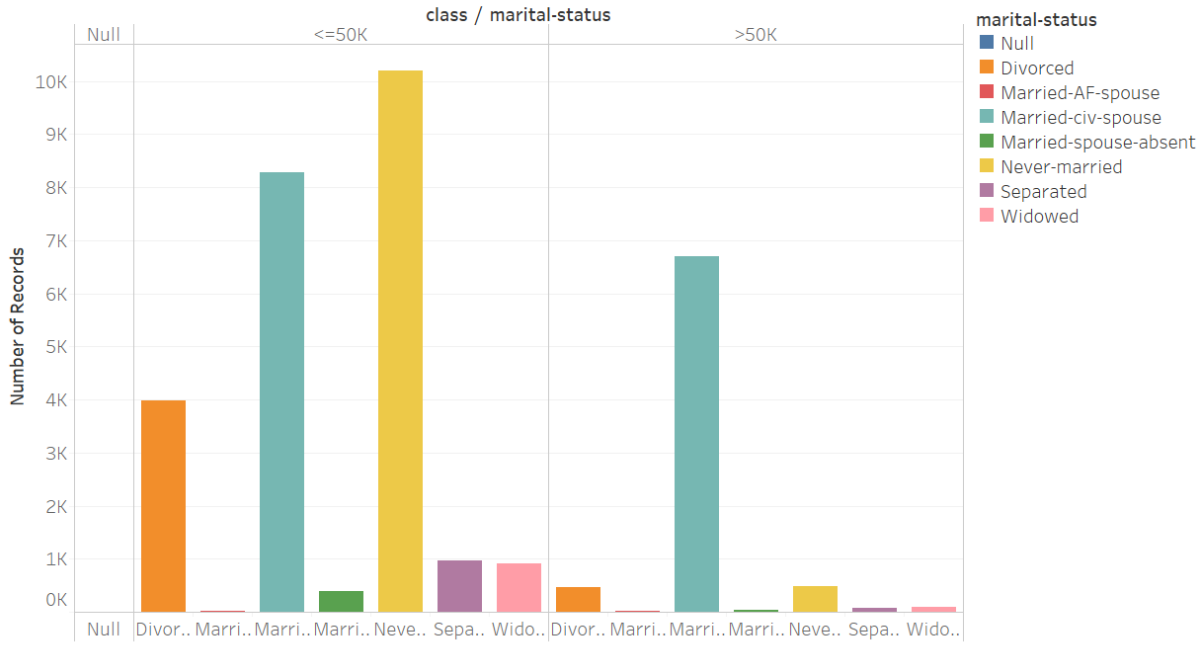
Interesting to note that a lot of people with higher than 50K salary come from the private sector.



Sum of Number of Records for each race broken down by class.

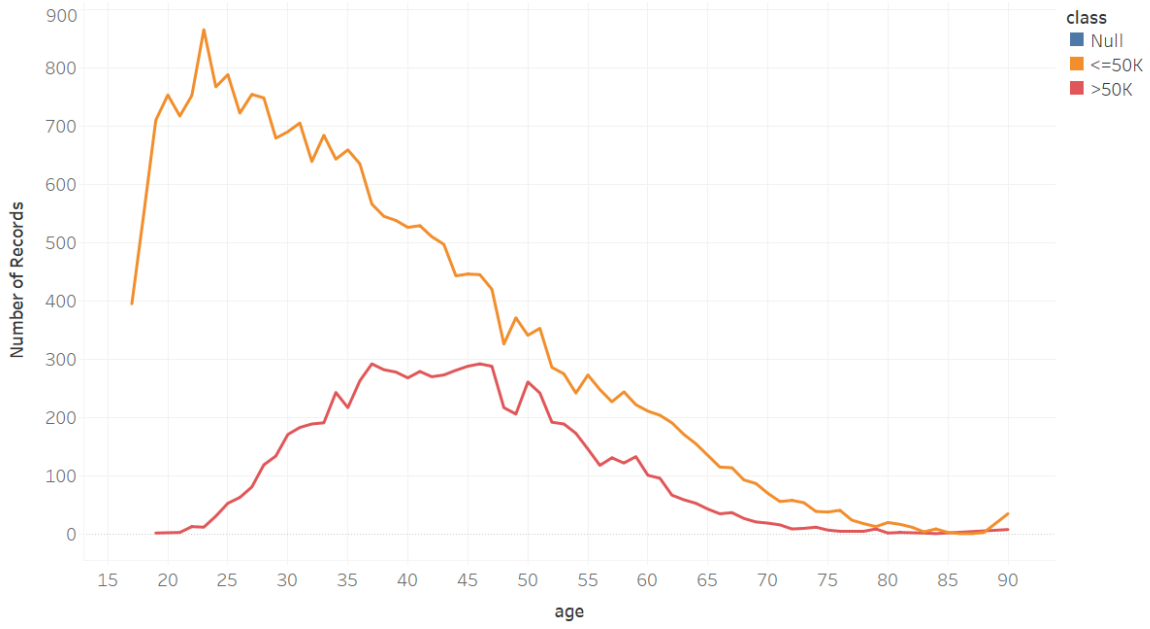
Tough to say anything conclusive from this graph, as it seems most of the records are of white people.

marital-status



Sum of Number of Records for each marital-status broken down by class. Color shows details about marital-status.

age

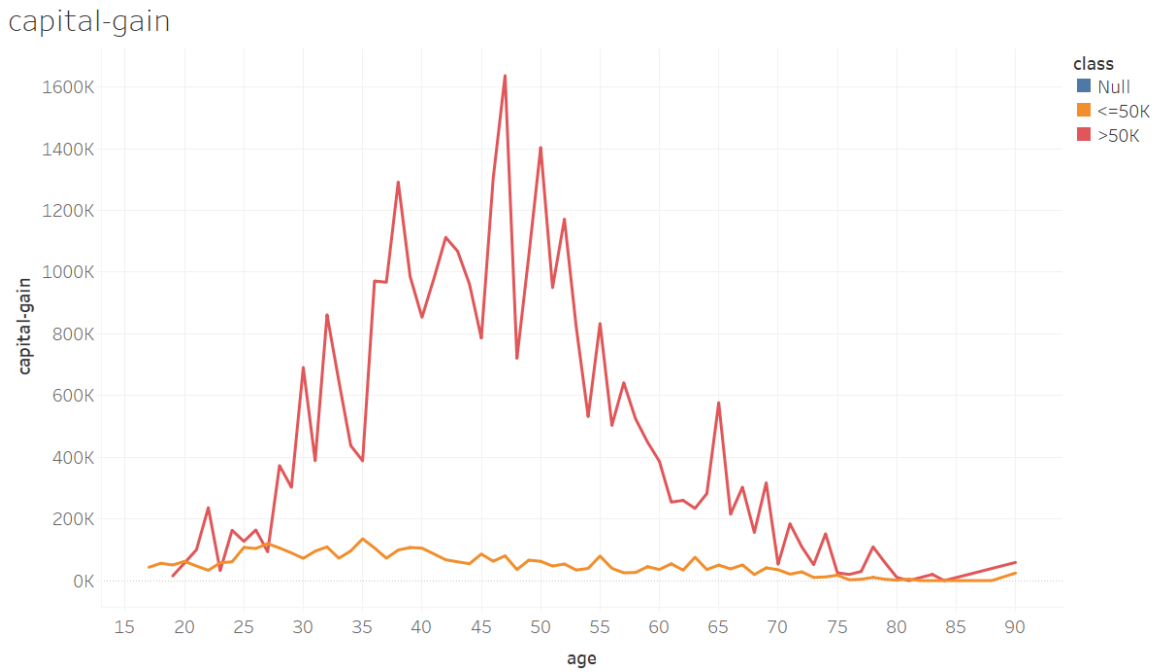


The trend of sum of Number of Records for age. Color shows details about class.

Above graph gives an indication of the distribution of age for each class in the dataset. As we would expect, most people with higher than 50K salary seem in the age range of 35 to 55. On the other hand, less than 50k salary class seems to be peaking between the age of 20 to 35.

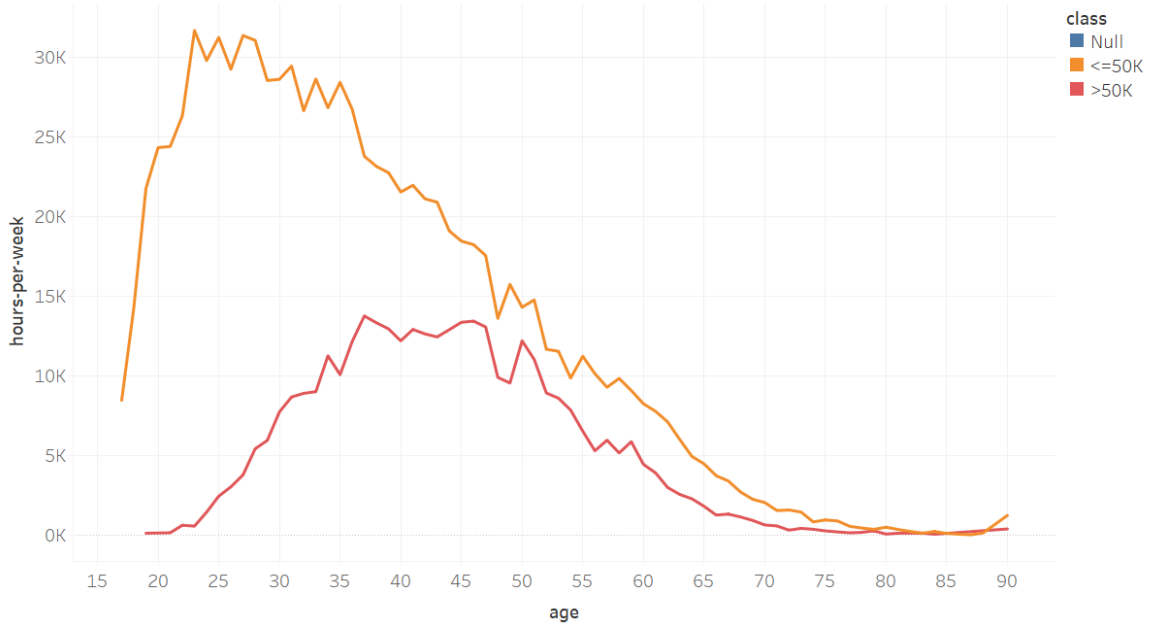
Hours-per-week, age, capital-gain, capital-loss, fnlwgt and education-num vs. hours-per-week, age, capital-gain, capital-loss, fnlwgt and education-num.

Looking at the above scatter plots between different continuous attributes, it seems there is no concrete correlation between any of them.



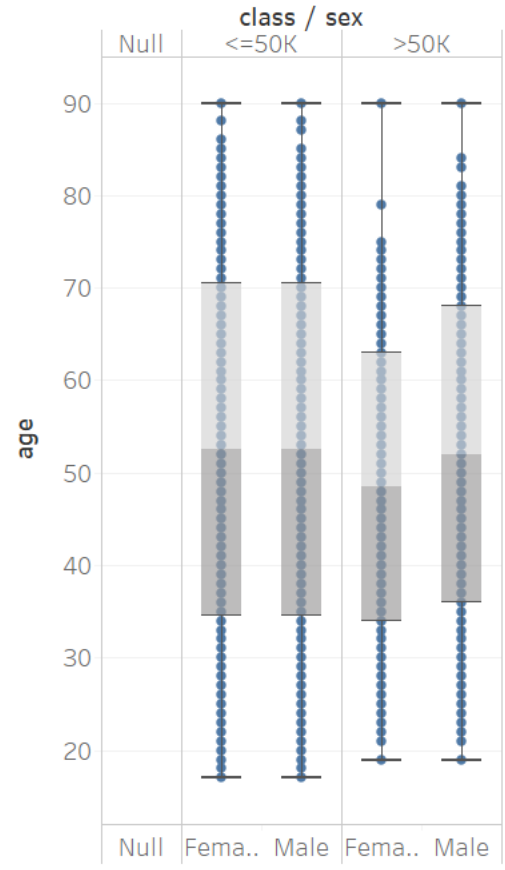
The trend of sum of capital-gain for age. Color shows details about class.

hours-per-week

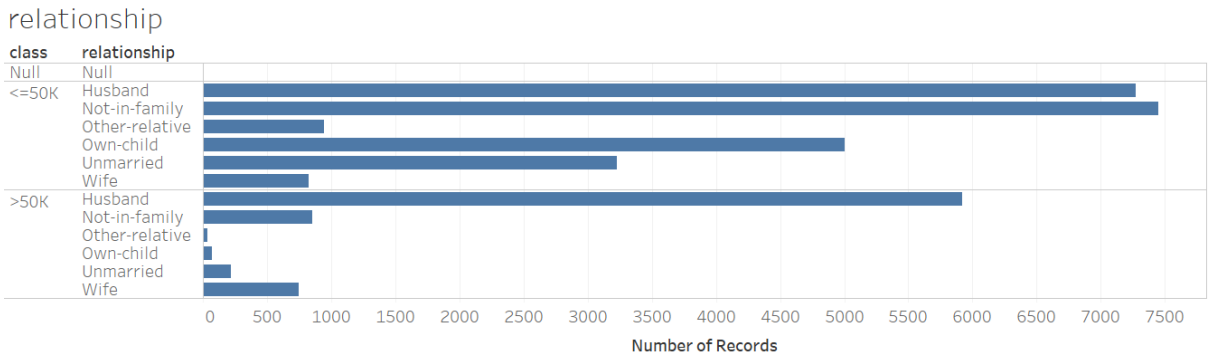


The trend of sum of hours-per-week for age. Color shows details about class.

age_sex



Age for each sex broken down by class.



Sum of Number of Records for each relationship broken down by class.

Looking at the above bar plot, it seems a lot of people with > 50K salary are husbands.

4.2 K-fold Cross Validation

K = 5

```
=== Naive Bayes ===
Classifier: weka.classifiers.bayes.NaiveBayes
Dataset: adult
Folds: 5
Seed: 1

=== 5-fold Cross-Validation ===
Correctly Classified Instances       27177               83.4649 %
Incorrectly Classified Instances      5384               16.5351 %
Kappa statistic                          0.502
Mean absolute error                      0.1732
Root mean squared error                  0.3717
Relative absolute error                 47.3571 %
Root relative squared error             86.9212 %
Total Number of Instances            32561


=== Decision Tree ===
Classifier: weka.classifiers.trees.J48 -C 0.25 -M 2
Dataset: adult
Folds: 5
Seed: 1

=== 5-fold Cross-Validation ===
Correctly Classified Instances       28034               86.0969 %
Incorrectly Classified Instances      4527               13.9031 %
Kappa statistic                          0.5959
Mean absolute error                      0.1943
Root mean squared error                  0.3221
Relative absolute error                 53.1506 %
Root relative squared error             75.321  %
Total Number of Instances            32561
```

The screenshot above shows the evaluation summary for k-fold cross validation when k = 5 for both Naïve Bayesian and Decision Tree classifiers. We get a total of 27177 (= 83.4649%) correctly classified instances for Naïve Bayes and a total of 28034 (= 86.0969%) correctly classified instances for Decision Tree. Clearly, Decision Tree is the better performing classifier although the difference is barely 800 instances (approx. 3%). It is also interesting to note that we are getting a pretty good MAE and RMSE for both classifiers.

K = 10

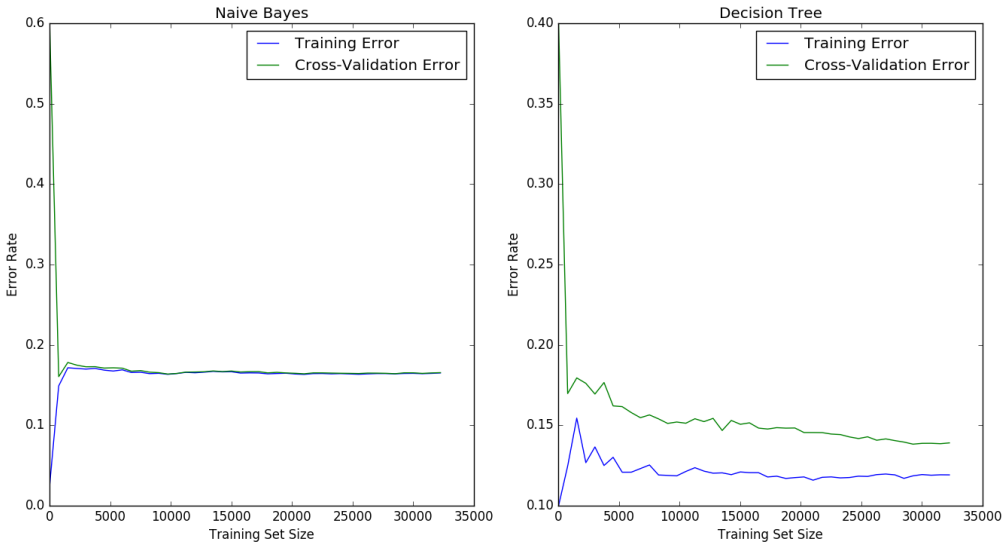
```
=== Naive Bayes ===
Classifier: weka.classifiers.bayes.NaiveBayes
Dataset: adult
Folds: 10
Seed: 1

=== 10-fold Cross-Validation ===
Correctly Classified Instances       27181               83.4772 %
Incorrectly Classified Instances      5380               16.5228 %
Kappa statistic                          0.5019
Mean absolute error                      0.173
Root mean squared error                  0.3715
Relative absolute error                 47.3193 %
Root relative squared error             86.884  %
Total Number of Instances            32561


=== Decision Tree ===
Classifier: weka.classifiers.trees.J48 -C 0.25 -M 2
Dataset: adult
Folds: 10
Seed: 1

=== 10-fold Cross-Validation ===
Correctly Classified Instances       28068               86.2013 %
Incorrectly Classified Instances      4493               13.7987 %
Kappa statistic                          0.5997
Mean absolute error                      0.1927
Root mean squared error                  0.321
Relative absolute error                 52.6965 %
Root relative squared error             75.0703 %
Total Number of Instances            32561
```

The screenshot above shows the evaluation summary for k-fold cross validation when k = 10 for both Naïve Bayesian and Decision Tree classifiers. As you would expect, as we increase k, our classifier performs better. Although it is interesting that in the case of Naïve Bayes, our results only improve by 4 instances (27181 correctly classified instances at 83.4772% accuracy). Decision Tree on the other hand shows a greater improvement of 34 instances (28068 correctly classified instances at 86.2013% accuracy).

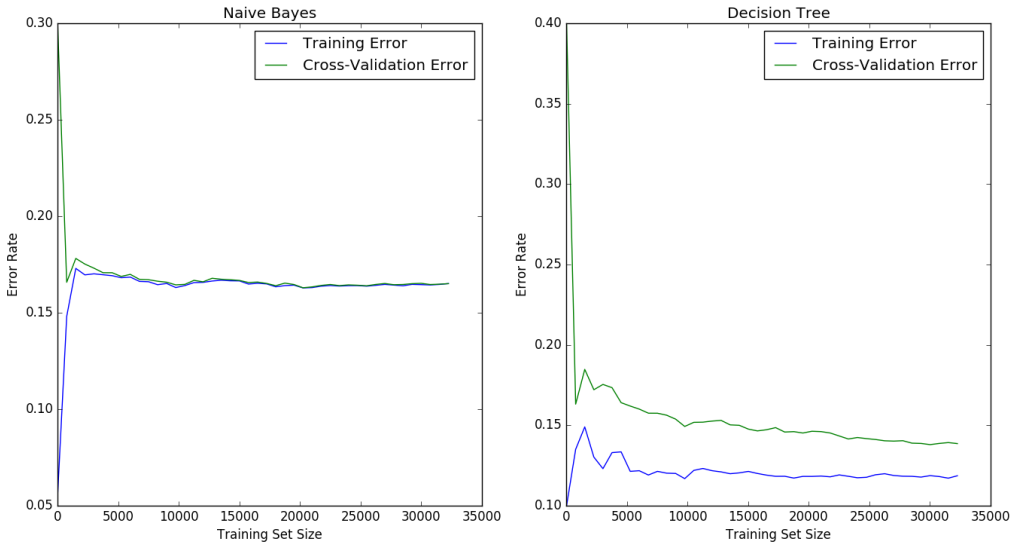
Again, Decision Tree is the better performing classifier as was the case with k = 5, and we get pretty good MAE and RMSE scores.

Few comments – The main goal of cross validation is to estimate how accurately a predictive model will perform in real practice. Looking at our results from 5-fold and 10-fold cross-validation, it is safe to estimate that when we finally apply our learned classifiers to the test set, our accuracy should be somewhere around 80%. If its less than that, then we probably need to reconsider our machine learning model.

4.3 Learning Curve



Learning Curve for K = 5



Learning Curve for K = 10

The two images above show the learning curves generated for both Naïve Bayes and Decision Tree classifiers for k = 5 and 10 respectively.

The first observation that can be made is that when the training set size is very small, the training error in all cases is very small. This is expected because when training set is very small, it is easy to fit our machine learning model on almost every training example. But as the training set gets bigger and bigger, it gets harder to fit our model on every training data point. Similarly, since the cross-validation error comes from data that is previously unseen,

when the training set is very small, you're not going to generalize well and hence error is initially large. But as the data set size becomes bigger, we will be able to fit the data better and hence the cross-validation error will decrease.

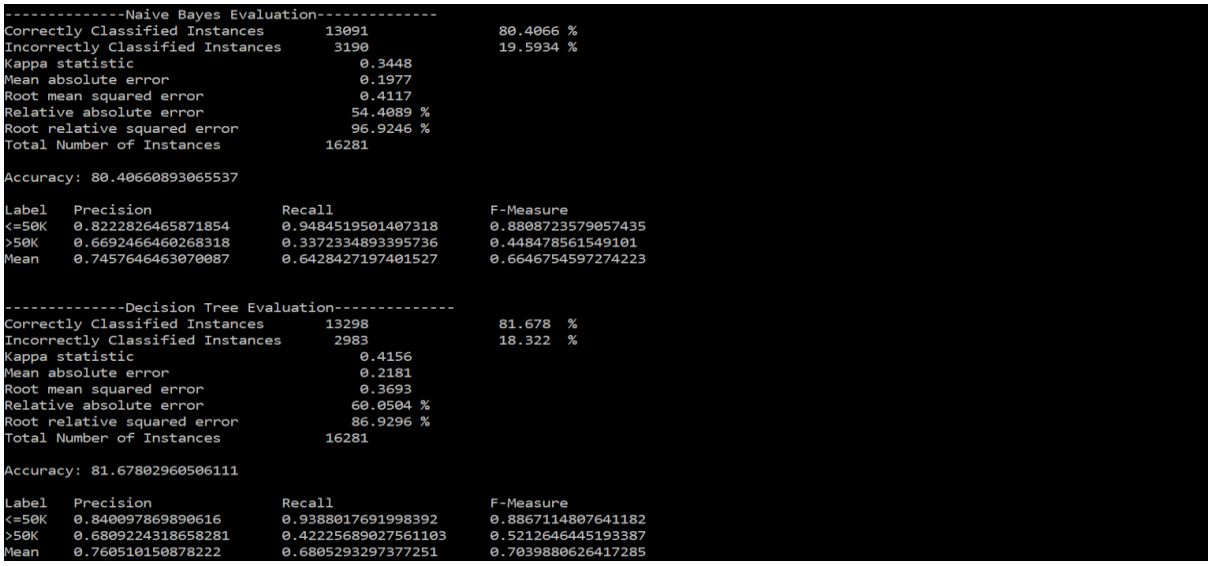
The next observation that can be made is regarding the Naïve Bayes classifier. In both cases (i.e. for k= 5 and 10), the learning curve has a very peculiar evolution pattern. Initially, the training error increases and the cv error decreases, but after a certain point, they both plateau out and there is no change in either. Also, after a certain point, both errors become very close to each other. Such behaviour is generally seen in machine learning models with high bias. Looking at the learning curve of our Naïve Bayes classifier, it is very clear that it has high bias and hence suffering from underfitting.

Regarding the Decision Tree classifier, it again has its own evolution pattern. The training error for both k = 5 and 10 remains low when training set size is small. As the training set size increases the training error expectedly increases. However, compared to Naïve Bayes, the training error despite increasing, still always stays very low. This is classic behaviour for an overfitting machine learning model where despite a lot of data, the model is almost fitting all data points and hence the training error always stays low. On the other hand, the cross-validation error continues to decrease as training set size increases instead of plateauing out. Another indication of high variance and overfitting in our Decision Tree classifier is the fact that there is a substantial gap between the training error and cv error curves.

4.4 Applying Classifier to Test Data

The figure below shows the final result that is generated when we apply our two learned classifiers to the 'adult_test.csv' test set. We get an accuracy of 80.4% for our Naïve Bayes classifier and an accuracy of 81.68% for our Decision Tree classifier which is pretty good. Also, our precision, recall and F1-Measure scores are mostly ranging from 0.65 to 0.75, which is decent but there's definitely room for improvement.

It is also worth mentioning that looking at these results, we could say that our cross-validation strategy did a pretty good job in estimating the final accuracy of our classifiers.



```
--------------Naive Bayes Evaluation--------------
Correctly Classified Instances       13091               80.4066 %
Incorrectly Classified Instances      3190               19.5934 %
Kappa statistic                          0.3448
Mean absolute error                      0.1977
Root mean squared error                  0.4117
Relative absolute error                 54.4089 %
Root relative squared error             96.9246 %
Total Number of Instances            16281

Accuracy: 80.40660893065537

Label   Precision               Recall                  F-Measure
<=50K   0.8222826465871854      0.9484519501407318      0.8808723579057435
>50K    0.6692466460268318      0.3372334893395736      0.448478561549101
Mean    0.7457646463070087      0.6428427197401527      0.6646754597274223


--------------Decision Tree Evaluation-------------
Correctly Classified Instances       13298               81.678  %
Incorrectly Classified Instances      2983               18.322  %
Kappa statistic                          0.4156
Mean absolute error                      0.2181
Root mean squared error                  0.3693
Relative absolute error                 60.0504 %
Root relative squared error             86.9296 %
Total Number of Instances            16281

Accuracy: 81.67802960506111

Label   Precision               Recall                  F-Measure
<=50K   0.840097869890616       0.9388017691998392      0.8867114807641182
>50K    0.6809224318658281      0.4222568927561103      0.5212646445193387
Mean    0.760510150878222       0.6805293297377251      0.7039880626417285
```