Tutorial 2

In this tutorial you will learn how to train and test a decision tree classifier. By the end of this tutorial you will be able to perform a classification task using rpart decision tree classification package and analyze your classification results.

1- **[Pre-lab]** Load the epinions training and test datasets
2- Preprocess the training and test datasets
3- Load package with decision tree classifier (rpart)
4- Train a decision tree classifier over the same training data
5- Make predictions on test samples
6- Calculate the confusion matrix

Hand in: Report the results based on different splitting functions ("anova", "poisson", "class" or "exp" methods) and briefly provide your insight on the results.

Tutorial 3

In this tutorial you will learn how to use regression models to predict different attributes. By the end of this tutorial you will be able to train and test different regression models using "lm" package in R.

I) Linear Regression with one variable:

1. [Pre-lab] Load the height-age prediction training and test datasets
2. Plot the datasets
3. Learn a model using "lm" package in R
4. Print summary of the model
5. Plot the fitting function over the training data
6. Make predictions on test samples
7. Print the gold label results and predicted results

II) Linear Regression with multiple variables:

1. Load the housing prices training and test dataset
2. Plot the data set for each variable
3. Learn a model using "lm" package in R without feature scaling
4. Print summary of the model
5. Plot the fitting function over the training data
6. Make predictions on test samples using no feature scaling
7. Print the gold label results and predicted results
8. Repeat steps 3 to 6 with feature scaling
9. Compare both models

Tutorial 4

In this tutorial you will learn how to use logistic regression for binary classification using the "glm" package in R.

III) Logistic Regression:

1. Load the graduate school admission prediction datasets (training and test)
2. Print the summary of training data
3. Print a two-way contingency table of rank and other predicting features
4. Convert numerical features to categorical ones for the rank feature
5. Learn a logistic regression model to predict the admission using "glm" package in R with only gre and gpa as features
6. Print summary of the model
7. Make predictions on test samples
8. Learn a logistic regression model to predict the admission using "glm" package in R with all features
9. Print summary of the model and compare it with the previous model
10. Make predictions on test samples using the new model
11. Compare both models

Hand in: Provide the summary of both models and briefly explain their differences.

Assignment 2

In this assignment you will be given a small dataset (train and test) similar with the one in Tutorial 4, where some feature values are missing. You will be also given the missing values to conduct a comparative analysis. Perform the following tasks and report the results accompanied by the codes and created files.

1- Classification with missing value samples removed: remove the samples with missing features, train a logistic regression model using all features and test your model.
2- Missing value prediction: predict the missing values in the dataset using a regression model. You can train multiple regression models (7 models) to estimate the missing values.
3- Classification with predicted missing values: since you have the missing values obtained from your regression models, retrain a logistic regression model with all data and test your results. You can train and test different logistic regression models based on different missing values you've predicted (7 models).
4- Classification with all true values: retrain a logistic regression model with the actual value of missing features and test your results.
5- Provide the analysis and final results for:
    a. Your regression model[s] and predicted missing values
    b. Quantitative and qualitative analysis over your regression model and prediction of missing values
    c. Your logistic regression model[s] with/out missing values

Tutorial 5

In the first part of this tutorial you will learn how to train a neural network classifier for classifying 3 wine types based on their chemical ingredients. The R library 'neuralnet' will be used to train and build the neural network.

1- [Pre-lab] Install and set the 'neuralnet' library
2- [Pre-lab] Load the given wine dataset
3- Observe the dataset using 'head' command
4- Extract the first 100 samples as a training set and the rest as a test set
5- Train a neural network model based on all the given features
6- Classify the test set using the trained model
7- Look at the predicting results
8- Calculate the accuracy of your model

In the second part you will be able to transform the features using PCA and analyze how the results will differ in terms of effectiveness and efficiency.

1- Load the 'prcomp' library
2- Run PCA over all the features of the previous dataset after scaling
3- Observe the dataset using 'head' command
4- Compare the standard deviation of the modified dataset and the original one
5- Decide how many principal components should be retained using the plot function
6- Follow steps 4 to 7 of the first part using the PCAs you obtained from the previous step

Hands in: Compare the results in terms of accuracy and efficiency of the models before and after PCA.

Tutorial 6

In this tutorial you will learn how to use the "e1071" package for classification using naïve bayes and SVM algorithms. By the end of this tutorial you will be able to train various naïve bayes and svm classification models and use the trained models for prediction and evaluate the results.

I) Naïve Bayes Classification: classify the products based on their reviews

1- Load the epinions dataset
2- Preprocess the training and test datasets
3- Load package with naive bayes function (e1071)
4- Train a naïve bayes model over the training data
5- Make predictions on test samples
6- Calculate the confusion matrix

II) SVM Classification

1- Use SVM package in e1071 to train your SVM model
2- Train a default SVM classifier over the same training data
3- Test the model over the test dataset
4- Calculate the confusion matrix
5- Test different parameters and analyze the results

Hand in: Report the accuracy results based on different classifiers you used in the lab and provide a brief analysis about the best performing model.

Assignment 3

In this assignment you will be given a dataset (train and test) of documents containing 20 directories, where each document contains the text belonging to one newsgroup. You can download the dataset from this link:
http://qwone.com/~jason/20Newsgroups/20news-bydate.tar.gz
Perform the following tasks, keeping the same training and test data, and report the results accompanied by the codes and created files.

Part-1:

1- Load the dataset in R.
2- Preprocess the dataset using following steps (hint: you can use the tm package, tmMap(Corpus,Function):
    a. Remove XML from the document
    b. Remove the author of the message
    c. Remove stop words
    d. Remove extra spaces
    e. Transform all upper cases to lower case
    f. Remove punctuations
    g. Remove numbers
3- Create document-term matrix using TFIDF and stemming options in DocumentTermMatrix() function in the tm package
4- Convert the DT matrix to the R data frame
5- Train a SVM classifier using training data and report the results on the test set

Part-2:

1- Combine the training and test datasets (the preprocessed version)
2- Implement a 10-fold Cross Validation (CV) algorithm and perform the classification task using a SVM classifier
3- Report the results over each fold and final CV results
4- Compare the CV results with part

Bonus question: find the best model using different preprocessing and SVM options and report your best results using 10-fold cross validation.

Tutorial 7

In this tutorial you will learn how to take advantage of ensemble learning in order to improve the performance of classification tasks. By the end of this tutorial you will be able to train various classification models and use them in an ensemble model for prediction and evaluate the results.

1- [Prelab] Load the iris dataset and randomForest library
2- Randomize the dataset
3- Divide the dataset set to train (first 100 example) and test (the rest)
4- Train a randomForest model over the training data using 100 trees
5- Calculate the error rate for the randomForest classifier over the test set
6- Train NNET and SVM over the training set and predict over the test set, then calculate the error rate for each classifier
7- Create an ensemble model by combining the results of randomForest and NNET with equal weights.
8- Compare the error rate of the ensemble model with NNET and randomForest
9- Create another ensemble model by combining the results of randomForest, NNET and SVM with equal weights.
10- Compare the error rate of the ensemble model with other models

Hand in: How the weight of each individual classifier impacts the results? Justify your answer with few examples.

Tutorial 8

In this tutorial you will learn how to take advantage of Hidden Markov Models for a scenario. By the end of this tutorial you will be able to initiate a HMM model and calculate the Viterbi, forward, backward and posterior probabilities of a given sequence.

Given the information about the weather and possible activities, perform the following tasks:

Weather = ('Rainy', 'Sunny')
Activities = ('walk', 'shop', 'clean')
Initial probabilities = {'Rainy': 0.6, 'Sunny': 0.4}
Transition probabilities = { 'Rainy' : {'Rainy': 0.7, 'Sunny': 0.3}, 'Sunny' : {'Rainy': 0.4, 'Sunny': 0.6}, }
Emission probabilities = { 'Rainy' : {'walk': 0.1, 'shop': 0.4, 'clean': 0.5}, 'Sunny' : {'walk': 0.6, 'shop': 0.3, 'clean': 0.1}, }

1- Install and load the HMM package in R.
2- Initiate a HMM model based on the given data.
3- Return a path of states and associated observations with length 10.
4- Given the activity path {'walk', 'shop', 'walk', 'walk', 'clean', 'shop'}, return the Viterbi path of weather states.
5- Compute the forward, backward and posterior probability of the given activity path.
6- Produce a sample observation dataset randomly and infer optimal parameters to the HMM using Viterbi training.

Another possiblity: fair/unfair dices