MIE 1624 Introduction to Data Science and Analytics – Fall 2017

Assignment 2

Due Date: 6:00 pm, December 11, 2017

Submit via Blackboard

Background:

Classification is a supervised machine learning approach required by several data mining tasks to classify data into classes. It is supervised because labelled data is used to train models. Many types of machine learning models can be used for training classifiers, such as the kNN algorithm, Logistic Regression, SVM, XGBoost, Random Forest, and Decision Trees.

The purpose of this assignment is to train, validate, and tune multiple classifiers that can predict, given a set of observations about a person, which income bracket they fall into. The data for this assignment is census data from a previous year that contains information about people across the United States.

As seen in Assignment 1, data is often split into training and testing data. The training data is typically further divided to create validation sets, either by just splitting, if enough data exists, or by using cross-validation within the training set. The model can be iteratively improved by tuning the hyperparameters of the model, or through feature selection.

Produce a report in the form of an IPython Notebook detailing the analysis you performed to determine the best classifier for the given data set. Your analysis must include the following steps: data cleaning, exploratory data analysis, feature selection (or model preparation), model implementation, model validation, model tuning, and discussion. When writing the report, make sure to explain for each step, what it is doing, why it is important, and the pros and cons of that approach.

The training set and testing set for this assignment are in the files *income-training.csv* and *income-testing.csv*, respectively. Both datasets are labelled, and the *IncomeBracket* column is the target variable, which has one of three possible values (<50K, 50-100K, or >100K). The list of attributes contained in both datasets can be found in the appendix. The training set and testing set come from a modified version of the dataset found in the following UCI repository: https://archive.ics.uci.edu/ml/datasets/adult. More information can be found there, including papers that have used this data set.

Learning objectives:

1. Understand how to clean and prepare data for machine learning, including working with incomplete data, and categorical data.
2. Understand how to explore data to look for correlations between the features and the target variable.
3. Understand how to apply machine learning algorithms to the task of classification.
4. Improve on skills and competencies required to compare and contrast the performance of classification algorithms, including application of performance measurements, statistical hypothesis testing, and visualization of comparisons.
5. Understand how to improve the performance of your model.
6. Improve on skill and competencies required to collate and present domain specific, evidence-based insights.

To do:

1. **Data cleaning (20 marks):**
   While the data is made ready for analysis, several values are missing and majority of the features are categorical. For the data cleaning step, handle missing values however you see fit and justify your approach. Provide some insight on why you think the values are missing and how your approach might impact the overall analysis. Suggestions include filling the missing values with a certain value (e.g. mean for continuous data, mode for categorical data) and completely removing the features with missing values. Secondly, convert categorical data into numerical data by encoding and explain why this is important.
   - Create a function called 'clean' that takes in a raw testing dataframe, and returns a cleaned numerical dataframe

2. **Exploratory data analysis (20 marks):**
   a. Present 3 graphical figures that represent trends in the data. How could these trends be used to help with the task of classification of income bracket? All graphs should be readable and have all axes appropriately labelled.
   b. Visualize the order of feature importance using PCA, correlation plot, or a similar method. Given the data, which of the original attributes in the data are most related to an individual's income bracket?

3. **Feature selection (10 marks):**
   Create at least one additional feature that is not originally part of the dataset but is based on the original features of the dataset. Explain how feature engineering is a useful tool in machine learning. Then select the features to be used for analysis either manually or through some feature selection algorithm (e.g. stepwise regression, PCA). Not all features need to be used; features can be removed or added as desired although the same set of features must be used for all your machine learning models. Provide justification on why you selected the set of features.

4. **Model implementation (25 marks):**
   Implement 4 different classification algorithms of your choice on the training data using 10-fold cross-validation. How does your model accuracy compare across the folds? Which model performed best? For each algorithm, briefly talk about what it is, what its pros and cons are, and why you chose that algorithm.

5. **Model tuning (10 marks):**
   Improve the performance of the models from the previous step, and select a final, optimal model using grid search. Your optimal model must have an accuracy of at least 70% in cross-validation.
   - Please comment out any grid search technique, and save, and load your model using the *pickle* library

6. **Testing (5 marks):**
   Use your optimal model to make predictions on the test set. How does your model perform on the test set vs. the training set?

7. **Discussion (10 marks):**
   Finding the most suitable algorithm for a given application task (comparing multiple classifiers on a specific domain) requires selecting performance measures, such as accuracy, true positive rate (TPR), false positive rate (FPR), etc., according to which to judge the algorithms being considered. As part of your evaluation procedure, select an appropriate number of measures, review the results that your chosen methods obtained on the selected measures and try to explain these observations. In addition, explain what criteria you used to determine the "best" model.

Bonus:

We will give 10 bonus marks to 3 students who achieve the highest accuracy values on a different testing set then the one provided. The code for evaluating the code is at the end of the document. Please insert a cell at the bottom of your code, and paste the code provided. Please adhere to the variable naming conventions if you wish to be considered for the bonus marks.

Tools:

- **Software**
  - **Python Version 3.X** is required for this assignment. Your code should run on the Data Scientist Workbench (Kernel 3). All libraries are allowed but here is a list of the major libraries you might consider: Numpy, Scipy, Scikit, Matplotlib, Pandas.

- No other tool or software besides Python **and its component libraries** can be used to touch the data files. For instance, using Microsoft Excel to clean the data is not allowed.

- **Required data files**
  - **income-training.csv:** classified census data for several people across USA with their corresponding income bracket.
  - **income-testing.csv:** more classified data in the same structure as the income-training-set.csv.
  - The data files cannot be altered by any means. The IPython Notebooks will be run using local versions of these data files.

What to submit:

Submit via Blackboard an IPython notebook and model containing your implementation and motivation for all the steps of the analysis with the following naming convention:

lastname_studentnumber_assignment2.ipynb
lastname_studentnumber_assignment2_model.sav

Make sure that you comment your code appropriately and describe each step in sufficient detail. Respect the above convention when naming your file, making sure that all letters are lowercase and underscores are used as shown. **A program that cannot be evaluated because it varies from specifications will receive zero marks. Late submissions will not be accepted.**

Tips:

1. You have a lot of freedom with however you want to approach each step and with whatever library or function you want to use. As open-ended as the problem seems, the emphasis of the assignment is for you to be able to explain the reasoning behind every step.

2. While some suggestions have been made in certain steps to give you some direction, you are not required to follow them. Doing them, however, guarantees full marks if implemented and explained correctly.

3. The output of the classifier when evaluated on the training set must be the same as the output of the classifier when evaluated on the testing set, but you may clean and prepare the data as you see fit for the training set and the testing set.

4. When evaluating the performance of two algorithms, keep in mind that there can be an inherent tradeoff between the results on various performance measures. For example, the TPR and the FPR are quite different and often an algorithm with good results on one yields bad results on the other.

Appendix:

Below is the list of attributes/features contained in both the training set and testing set:

1. *Age:* Person's age (continuous)
2. *WorkClass:* Person's working status (Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked)
3. *FinalWeight*: Weight value generated by Current Population Survey (CPS) - people with similar demographic background should have similar weights within a given state (continuous)
4. *Education*: Highest education level completed by person (Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool)
5. *EducationLvl:* Numerical representation of the highest education level completed (discrete)
6. *MaritalStatus*: Person's marital status (Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse)
7. *Occupation*: Person's occupation (Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces)
8. *Relationship:* Relatives person has (Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried)
9. *Race:* Person's race (White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black)
10. *Sex:* Peron's sex (Female, Male)
11. *CapitalGain*: Person's yearly capital gain on investments (continuous)
12. *CapitalLoss*: Person's yearly capital loss on investments (continuous)
13. *HoursPerWeek:* The number of hours the person works per week (continuous)
14. *NativeCountry:* Person's native country (United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Colombia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinidad&Tobago, Peru, Hong, Holland-Netherlands)

Code for testing model against alternative test set:

Please create a new cell at the end of the workbook, and paste the following code. This code is what will be used to evaluate your model. To test the code, you make change 'income-testing-2.csv' to 'income-testing.csv'. Please adhere to the provided variable names.

```
## Code for testing with alternative test set
# clean(df): your function for cleaning and converting categorical data into numerical data
# model: your chosen model
# feature_sets: the list of features your model uses
```

```
# scaler: your scaler, if you are using one
# Change the following command to False if you are not using a scaler
use_scaler = True

from sklearn.metrics import accuracy_score

#model = pickle.load(open('model.sav', 'rb'))
trainDF = clean(pd.read_csv('income-training.csv'))
newTestDF = clean(pd.read_csv('income-testing-2.csv'))

X_train = trainDF[feature_set]
y_train = trainDF['IncomeBracket']

X_test = testDF[feature_set]
y_train = testDF['IncomeBracket']

if use_scaler:
  X_train = scaler.fit_transform(X_train)
  X_test = scaler.transform(X_test)

predictions = model.predict(X_test)
ACC = accuracy_score(y_test,predictions)*100

print ('Model performance on test set: {}'.format(round(ACC,2)))
```