COSC2670 @ RMIT University

Practical Data Science with Python

Semester 1, 2020

Assignment 2 – Data Modelling and Presentation

The Classification of new observations

I certify that this is all my own original work. If I took any parts from elsewhere, then they were non-essential parts of the assignment, and they are clearly attributed in my submission. I will show we I agree to this honour code by typing "Yes": **Yes**

Name: Duncan Do

Student Number: s3718718

Email Address: s3718718@student.rmit.edu.au

Date: 10/06/2020

Contents Page

Abstract

The aim of this report was to use a dataset about effects of numerous internal and external stimuli in the expression levels of various protein in mice to investigate if, based on those protein levels, we could accurately predict which combination of stimuli or "class" they belong to. All in the effort to identify the most effective classification technique to predict these classes. An experiment was conducted on mice with various combinations of these stimuli and the data on their "class" and protein levels were stored on an xls data file. The data was then processed and injected into a predictive 2 classification models to test which could yield the most accurate mice "class" predictions. Overall, the results indicate, with enough extensive data and testing, it is possible to accurately predict certain factors about the mice. The report concludes that the "K-Nearest Neighbour" classification model is the more accurate model for predicting a mouse's "class". It is recommended that classification tasks with high dimensional vectors use the "K-Nearest Neighbour" classification model to run their data.

Introduction

In machine learning and statistics, there always exists the problem of identifying of which set of categories (sub-populations) a new observation belongs to. Hence, in machine learning and statistics, many methods have been created to classify these new observations; dubbed "classifiers". These algorithms work under the basis of training a model with a set of data containing a set of observations (and their characteristics) whose category membership is known. Once trained, the model is then expected to use its prior knowledge of the test data to classify new observations as accurately as possible. This report will discuss the process of preparing data for classification, building the classifier, and using the classifier.

Methodology

There are several steps from acquiring the dataset of interest, to predicting observations using a classifier: all with their own subset of intermediate steps.

Data Preparation

As standard practice in Data Science, it is important to pre-process any data before subjecting it to thorough analysis and testing.

First step was to ensure the data types of all the fields in the dataset were of the correct type. The python command '<data frame>.dtypes' was used to identify all the field's current data types. Any corrections were done using the '<field>.astype(<desired data type>)' command, which would convert the field's data type specified parameter.

Next, to ensure none of the data entries came with manual spacing input errors, the command '<field>.str.strip()' was performed on all the string fields. Further human input errors such as typos were handled with the following replace command: '<field>replace(<incorrect value>, <correct value>, inplace = True)'.

Finally, whether by human error or design, datasets are bound to have missing data. Thus the command '<field>.fillna(<replacement for NaN>, inplace = True)' was used to replace them with 0. Other options would be to replace the missing data with the mean value of the field; however, in this investigation involving classification, using mean values to substitute an otherwise empty entry, can skew the results in undesirable ways; becoming an extraneous variable.

Data Exploration

The now clean data was subjected to thorough exploration to examine the data they hold and their relationships to its fellow fields. All singular field explorations were done by extracting an average value for all the values in said field. In this report, the expression levels of 10 mice proteins of the mice experimented on were individually extracted from the main dataset to perform a mean calculation. Once all 10 were calculated, they were recombined into a single data frame to examine the average levels of each protein (Table format).

Beyond exploring the data fields in isolation, it was important to explore the relationships between pairs of fields. In this report, 10 protein data fields were individually paired up with the class field. This was done to analyse the average protein level of select proteins in each

class of mice. Each record belonging to a class were grouped into their own data frame. Then like the singular field exploration, an average value for the expression levels of the protein of interest was obtained. Averages for all classes were then collated and plotted on a graph for visual comparison between classes. This process was repeated for the 10 selected proteins.

Data Modelling

The dataset was further manipulated to obtain the fields necessary to predict the class. Naturally, the "class" field was extracted into its own data frame, a singular column data frame dubbed the "target". Then, the features for which the model will use to predict the class had to be extracted into its own data frame dubbed the "data". (All 77 provided protein fields from the original dataset were used – further explained in the discussion).

To obtain convenient test data, the records from the data set were used to train and test the model; since for every set of protein level (mouse/observation), we know the category class they belong to. 2 sets of training data and 2 sets of testing data were extracted from the dataset, each half the size of the original dataset and randomised to obtain an out of order set of records.

The first model used to classify the data was the "K Nearest Neighbour Classifier". The parameters used were neighbours = 1, weight = uniform, metric = "minkowski", p = 2 (Explanations in the discussion). The model was fitted with the training data and then tested with the testing data to predict the categorical class. A confusion matrix was generated to view how often the classifier got the class right. Furthermore a decimal value was obtained, representing the percentage of accuracy displayed in the matrix. The same training and testing were done under a validation technique called "K-Folds Cross Validation". A strategy that divides the data into several partitions with a single partition being the test data. In various iterations of the experiment, the test data is implemented in a different injection point of the rest of the training data. The accuracy of these multiple experiments was extracted and averaged to obtain a mean value.

The process listed above was repeated for a secondary classification model to facilitate a discussion of effectiveness between multiple models. The second model chosen was "Decision Trees". The parameters being criterion = "gini", splitter = "best" and the rest left default (Explanations in the discussion). With both methodologies conducted, we are now able to discuss comparisons between them.

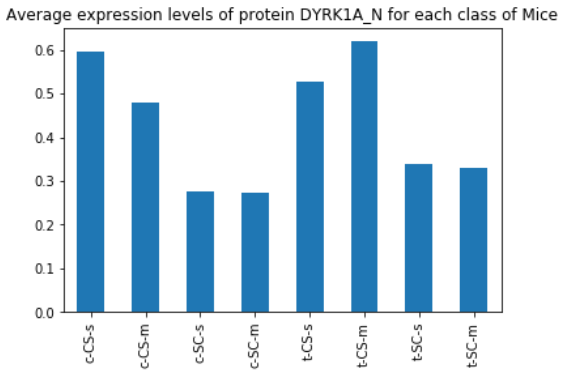
Results

Figure 1. Data Exploration. Singular field exploration (10 explorations represented in a single figure)

	DYRK1A_N mean	ITSN1_N mean	BDNF_N mean	NR1_N mean	NR2A_N mean	pAKT_N mean	pBRAF_N mean	pCAMKII_N mean	pCREB_N mean	pELK_N mean
0	0.42581	0.617102	0.319088	2.297269	3.843934	0.233168	0.181846	3.537109	0.212574	1.428682

Each field (10) in the figure represents the mean value from 1080 records. 1080 different expression levels for the given protein of the field compressed into a singular average.

Figure 2. Data Exploration. Relationships between fields (10 relationships were explored, figure 2 is one of them).



Each column represents the mean value for protein DYRK1A_N, of ever record where the mouse's categorical class is of the label. Mean values for all 8 classes were obtained and graphed to allow a side by side comparison with other classes. (Process **was** repeated for 9 other protein-class field relationships).

Figure 3. Data Modelling. Confusion matrix of the “K Nearest Neighbour” classification model (Accompanied by a decimal value representing the percentage of accuracy displayed in the matrix).

```
[[68  1  0  0  0  0  0  0]
 [ 6 65  0  0  0  1  0  0]
 [ 0  0 78  0  0  0  3  0]
 [ 0  0  0 56  0  0  0  0]
 [ 0  0  0  0 68  3  0  0]
 [ 3  1  0  0  0 53  0  0]
 [ 0  0  3  2  0  0 64  0]
 [ 0  0  1  0  0  0  0 64]]


[Train/test split] score: 0.95556
```

After training, the model yielded a 95.6% accuracy at predicting the correct categorical class of the test data. Under the parameters: neighbours = 1, weight = uniform, metric = “minkowski”, p = 2

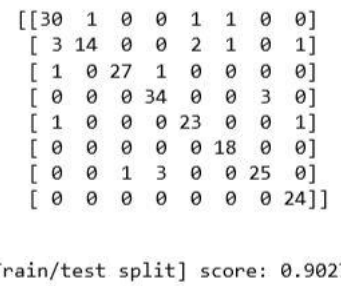
Figure 4. Data Modelling. Accuracy scores of the “K Nearest Neighbour” classification model over a “K-Folds Cross Validation” technique; 5 Folds (Accompanied by a decimal value representing the percentage of accuracy averaged from the accuracy scores for each fold).

```
[fold 0] score: 0.97222
[fold 1] score: 1.00000
[fold 2] score: 0.98148
[fold 3] score: 1.00000
[fold 4] score: 0.99537


KNN mean score [5 folds] = 0.989814
```

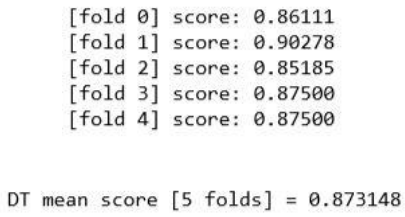
After 5 experiments under the K-Folds Cross Validation technique, the model yielded a 98.9% accuracy at predicting the correct categorical class of the test data.

Figure 5. Data Modelling. Confusion matrix of the “Decision Tree” classification model (Accompanied by a decimal value representing the percentage of accuracy displayed in the matrix).



After training, the model yielded a 90.3% accuracy at predicting the correct categorical class of the test data. Under the parameters: criterion = “gini”, splitter = “best” and the rest left default.

Figure 6. Data Modelling. Accuracy scores of the “Decision Tree” classification model over a “K-Folds Cross Validation” technique; 5 Folds (Accompanied by a decimal value representing the percentage of accuracy averaged from the accuracy scores for each fold).



After 5 experiments under the K-Folds Cross Validation technique, the model yielded an 87.3% accuracy at predicting the correct categorical class of the test data.

Discussion

Data Exploration

The results of the data exploration between various proteins and resulting class display the affect of said class' attributes on certain proteins. It was hypothesised that depending on what class a mouse belonged do (and their subsequent attributes) have drastic effects on certain protein levels. To investigate this, 10 proteins were selected: DYRK1A_N, ITSN1_N, BDNF_N, NR1_N, NR2A_N, pAKT_N, pBRAF_N, pCAMKII_N, pCREB_N, and pELK_N. Each were individually paired with the "class field to identify the differing average protein levels across all classes, example in figure 2. While many pairings resulted in similar protein levels across all classes. 2 classes had larger effects on most of the proteins. Those classes being: c-CS-s and t-CS-m. Each value in the class identification represents a specific trait of the mice in that class. For the former class, the mice are "control mice stimulated to learn, injected with saline" and the latter being "trisomy mice, stimulated to learn, injected with memantine". Both are mice stimulated to learn, but what is interesting is that the latter mice suffer from down syndrome yet yields similar results to the former. This could possibly suggest that the memantine that the latter class were injected with allowed fundamentally different mice to achieve similar resulting protein levels to control mice. Further specific investigation can be conducted on this.

Data Modelling

When choosing the appropriate features to be the basis on which our classification models will perform under; it was decided to use all 77 available protein fields. The string fields in the dataset provided no new data that was not included in the class field, which would become the target anyway. And to obtain the most accurate predictions on categorical classes possible, it was decided to utilise all the quantitative data available. If we can assess a mouses class with more protein expressions, we are more likely to accurately predict its class.

The first model chosen to classify the data was the "K Nearest Neighbour" model. The parameters chosen were: neighbours = 1, weight = uniform, metric = "minkowski", p = 2.

After multiple iterations of the model with varying neighbour variables, it became apparent that the higher the neighbour count went, the less accurate the model was. This may be attributed to the large number of dimensions in the dataset. With 77 axes of data to account for, slight changes in data values could have larger affects on its classification than if this were, for example, a 2-dimensional vector.

In saying that, with the neighbour count being only 1, it was unnecessary to put extra importance or "weight" on closer neighbours since there would only ever be 1 in consideration.

With the metric being set to minkowski and the p value being 2, it is equivalent to the standard Euclidean metric, a reliable distance metric for this model.

The Second model chosen to classify the data was the "Decision Tree" model. The parameters chosen were: criterion = "gini", splitter = "best", and the rest left default.

The default parameters have little room for improvement as they hold a solid basis for most classification tasks.

The gini criterion allows us to work with categorical target variables for which we have 8 of, allowing us to make use of gini method to create binary splits in the decision tree.

The "best" splitter parameter allows us to maintain relevance by splitting on the most relevant data.

After the implementation of both classification models, a basis for comparison can be made.

In relation to the results of this report, "K Nearest Neighbour" yields higher accuracy. However, accuracy does not always constitute "the best way". "Decision Trees" supports automatic feature interaction, whereas "K Nearest Neighbour" cannot. Furthermore, "Decision Trees" are faster due to "K Nearest Neighbour's" expensive real time execution. "K Nearest Neighbour" determines neighbourhood's, so there must be a distance metric involved. This implies that all the features must be numeric. Distance metrics may also be affected by varying scales between attributes and high-dimensional space. "Decision Trees", on the other hand, predicts a class for a given input vector. The attributes may be numeric or nominal.

Therefore both models provide stable and reliable classification strategies with their own enticing attributes.

Conclusion

It is clearly possible to predict the categorical class of new observations on the “mice proteins” dataset. The factors inclusive the “class’” of mice have noticeable impact on the expression levels of various protein. And different combinations of them can make up attributes they do not possess.

Both “K Nearest Neighbour” and “Decision Trees” yield high accuracy when classifying the protein expression data. However, the underlying qualities of each model bring out their own individuality. Despite the higher classification accuracy of “K Nearest Neighbour”, because of the flexibility and efficiency of “Decision Trees” and the reasons stated in the discussion; it is concluded that “Decision Trees” would be the optimal choice for a classification model (Especially in real-world, large corporate datasets).

References

https://archive.ics.uci.edu/ml/datasets/Mice+Protein+Expression# (Dataset)

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

https://www.geeksforgeeks.org/learning-model-building-scikit-learn-python-machine-learning-library/

https://www.youtube.com/watch?v=RlQuVL6-qe8&list=PL5-da3qGB5ICeMbQuqbbCOQWcS6OYBr5A&index=4

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

https://towardsdatascience.com/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222#:~:text=Decision%20tree%20vs%20KNN%20%3A,KNN's%20expensive%20real%20time%20execution.

https://datascience.stackexchange.com/questions/9228/decision-tree-vs-knn

https://towardsdatascience.com/how-to-tune-a-decision-tree-f03721801680