COSC2670

Practical Data Science with Python

Semester 1, 2020

Assignment 1 – Part 2 – Building a Classifier

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 I agree to this honour code by typing "Yes": ***Yes***.

Name: Duncan Do

Student Number: s3718718

Contents Page

Cover Page	1
Contents Page	2
Feature Engineering and Model Selection	3
Training the Model	4
Model Validation and Selection	5
Applying the trained model to unseen future data	6

Feature Engineering and Model Selection

Features

To build a good classifier, we must choose good features for the classifier to use. Firstly, if we are trying to classify users based on their opinions on Star Wars movies, we must train our model with data of people who have seen Star Wars. Thus their opinion is of value. To do this, records of users where they answered "No" to seeing Star Wars were voided. In addition to this, an executive decision was made to do the same voiding to records where they answered "No", to be a Star Wars fan. Reason being; if they are not a fan of Star Wars, their opinions may be unfairly negatively skewed. Further voiding of data from the feature candidates was done to ensure only relevant fields are used. Fields such as "Are you are a Star Trek fan?" were voided due to irrelevance.

In the end, the fields used for features are:

- Have you seen X Star Wars movie?
- Rank X Star Wars Movie
- View on X Character
- Who shot first?
- Know the Expanded Universe?
- Fan of the Expanded Universe?

After the records are omitted where the answers to "Seen Star Wars?" and "Fan of Star Wars?" are "No".

Furthermore, all qualitative data (data type: integer) must be converted to quantitative numerical data. This is because numerical data is easier to compare than string data. (E.g. the view on Star Wars characters will be put on a numerical Likert scale).

(All further discussion about the model can refer to any classifier for any of the demographics from the data set: age, gender, household income etc. Simply supply the desired demographic as the response category)

Model

The K-Nearest Neighbour model was selected to classify the data. This model was selected because it's a simple algorithm that can use data (Star Wars preferences) to predict fairly unrelated categories (peoples age, gender, household income etc.) by simply using the frequency of each of particular relations between Star Wars opinions and their resulting response value. In other words, it can use the trends in Star Wars Fans to predict peoples (age/gender/household income etc.) category. And the accuracy of the model's predictions grows exponentially based on the size of the data (neighbours) provided.

Pros of K-Nearest Neighbour	**Cons of K-Nearest Neighbour**
No Assumptions about the data	Expensive computation wise (Stores all training data)
Versatile	High memory requirement (Stores all training data)
Highly Accurate	Predictions grow slower based on the number of neighbours stipulated
Simple	Sensitive to irrelevant features and the scale of the data

Training the Model

To train the model, it needs to be fed data which it can learn from. This is done so the model and learn the relationship between the feature data and the response category (Learn through the repeated association between certain features and certain categories.

Once loaded with data; the model should be able to predict, to a degree, the category based on its feature inputs. To test this, we can supply the model with a set of features (parameters) that is expected to yield a specific category as output (Perhaps the feature inputs of an existing record). Then run the "predict" command from K-Nearest Neighbour Library import to run the model and view the outcome.

```
<K-Nearest Neighbour object>.predict([<list of features>])
```

If we used hard-coded features of an existing record, we know what category it should be placed in. Thus, we can gauge the accuracy of initial model. The output will be an artificial number representing a particular response category, we must relate this output to the possible response categories to identify which category was outputted.

This training method can be looped with a large test sample size to run multiple tests simultaneously and gauge a more comprehensive analysis of the model's accuracy. Furthermore this test can be extended by implementing the K-Nearest Neighbour algorithm with differing K-Values, to identify how varying amounts of data/neighbours affects the model's accuracy. (Accuracy tends to increase with larger amounts of neighbours taken in the computation).

Model Validation and Selection

To validate the effectiveness of this model, it must be measured on two specific parameters

- Its “good predictive power”
- Its ability to “well generalise” data it has not seen

First step is to define the error measure of the model. The prominent options are:

- The Classification Error Rate
- Mean Squared Error

Overall, the Mean Squared Error method is more in-depth, considering the degree of error the model makes, for this model and dataset, the Classification Error Rate is the better error measure. Reason being, depending on the demographic being analysed in the model (Age, Gender etc.), the model can only be wrong by so much, since the options are far more limited than other datasets. For example, if we are building a classifier for the gender category, and the model predicts wrong, the degree to which it was wrong is the same, since there are only 2 options. This is the case for all the possible response categories since the amount of options for any given survey question from the dataset is 5.

Because of the high number of records available for testing/training in the dataset, the “K-Folds Cross Validation” strategy was selected. Placing the test records in differing places between the training records allows us to view the progression of the model’s learning and accuracy. The results of those testing records will validate the model as a suitable classifier.

Figure 1

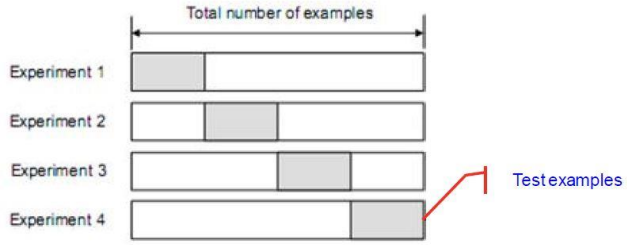

Figure 2

Figure 1: Y. Ren, “Practical Data Science: Classification”[SLIDE 20], presented to COSC2670. To Practical Data Science with Python, RMIT, VIC, Australia. DATE N/A. [*PowerPointSlides*]. Available: https://rmit.instructure.com/courses/67430/files/11273015?module_item_id=2268982, Accessed on: April. 30, 2020

Figure 2: Y. Ren, “Practical Data Science: Classification”[SLIDE 21], presented to COSC2670. To Practical Data Science with Python, RMIT, VIC, Australia. DATE N/A. [*PowerPointSlides*]. Available: https://rmit.instructure.com/courses/67430/files/11273015?module_item_id=2268982, Accessed on: April. 30, 2020

Applying the trained model to unseen future data

With a fully trained model, now is time to present the model to new data sets where the outcome is unknown.

The process for this, model scoring is like the training and testing/validation method. This involves:

- Preparing a dataset that has features in the defined style of the model. However, since this dataset is new, the you do not know the correct outcome. The trained model should be developed enough to make the correct prediction
- This dataset is then run by the model's prediction scheme like the training and validation step, resulting in a prediction based on its previous training