James Taylor

Assignment 2

University of Maryland University College

DATA650-9041 Spring 2021

Dr. Gortcheva

jtaylor380@student.umgc.edu

**Part 1 – Questions**

**Discuss the importance of data exploration and visualization prior to running the logistic regression method.**

Exploring and visualizing the data helps someone understand the data they’re working with. It will show errors in the data like outliers, missing data and just incorrect information. These may bias the regression if they are included for training. You also learn about the correlations and distributions of the variables, both dependent and independent. Some variables won’t be useful for including into a regression. An example of this is a unique identification number. This adds no real benefit to the model and only creates more noise in the data. Another concern is multicollinearity which makes the variable estimates less accurate. Variables may need to be transformed to better fit the assumptions of the model as well. A simple example of this would be to ensure all Titanic observations have the ‘survived’ value either a one or zero. Another potential transformation would be to convert a numerical variable into an ordinal one, or binning in other words. An example from the Titanic data is the ‘Class’ variable is stored as a sting and not a numerical value. Instead of first class having a numerical value of one it is stored as ‘1st’ which may cause problems if it isn’t transformed and used to train the model. Learning about the data through exploration would lead to these decisions.

**What do the Titanic data exploration results reveal about the relationships between the likelihood of survival and passenger data?**

Some observations are that 443 died and 313 lived. First class was the only ticket group to have greater than 50% survivorship, where 139 lived and 87 died. Third class had the worse survivorship, with 240 deaths and 78 surviving. Those who survived are on average slightly younger than those who did died (29.3 vs 31.1 years old respectively). Third class had the most deaths, totaling 240. Third class also has the worst survive/dead ratio. The most common age group was 20-30 year olds. The age distribution is somewhat skewed to the right. The age distribution of those who lived is not very different from the age distribution from all observations.

**Discuss the logistic regression method results, including the classification accuracy for training and test set.**

The model’s accuracy was 78% on the test data, meaning it correctly predicted if someone was going to survive or not by that amount. The precision was 69.8% on test data, or out of all survive predictions only 69.8% of them were correct. The recall was 77.2% on test data, or out of all actual survivors 77.2% of them were predicted to be one. These numbers would change depending on where the rounding threshold is set. It currently is set to round prediction values of .5 or more to one (survivor), but this could be increased or decreased for various reasons. Precision and recall have a trade-off between them. The F1 score was 73.3%, which is a score that looks at both precision and recall and corrects for an unbalanced target variable. It is similar to the accuracy measure. The results of the model on the training set was better in every measure but that’s to be expected and not of interest to us since the model has seen those data points.

**Is logistic regression suitable for this problem? Why or Why not?**

Yes, a binomial logistic regression is suitable. There is a binary dependent variable with a significant amount of observations on it. The input variables are numbers, or could be transformed into them so the model can use them. A model was able to achieve 78% on data it had not seen before. This proves there is information in the variables a logistic regression can extract to make predictions.

**What alternative machine learning methods could be suitable for this problem? Consider at least 2 alternative methods.**

Another potential machine learning model that could be used is a decision tree classifier (in Spark MLlib library). It would use the binary target variable and attempt to split branches to get the best purity in the leaves (all survived or all dead). Another is a Gradient-boosted tree classifier (also in Spark MLlib). These are both capable of handling binary classification dependent variables. They can also handle categorical independent variables but transformations would be needed.

**Part 2 – Low Birthweight Questions**

**Define the purpose of the study and the target variable. Which variables are used as predictors?**

The purpose of this analysis was to use a logistic regression to predict which mothers would have underweight births. The target variable was the dummy variable LOW, indicating if the birthweight was under 2500g or not. The variables used to make this prediction were mother’s age, race, smoking history, premature labor history, hypertension status, uterine irritability status and number of physician visits during first trimester.

**Interpret the data exploration and visualization results. What did you learn about the low birth weight data from data exploration, including possible relationships between predictors and the target variable?**

Firstly, the target variable is somewhat imbalanced. The dataset had 31% of its observations be underweight (59 underweight and 130 not). This wasn’t imbalanced enough to require remedies like weighting observations at the time. Many of the variables showed a relationship between itself and the target variable, however there were not many observations to make a definite conclusion on the existence of such a relationship. For example, mothers with no history of premature labor had a low birthweight-rate of 25%. For mothers with 1, 2 and 3 previous premature labor births, they had 66%, 40%, and 100% low birth-weight rate respectively. The evidence is strong for mothers with one previous premature birth with 24 observations but those with two or three have less, only having five and one observation respectively. Smokers have 40% chance of underweight birth compared to 25% non-smokers. Different races had variance on their low birthweight rates, with whites having 24%, blacks having 37% and other having 42%. Mothers with hypertension have a 58% chance of under-weight birth compared to 29% of non-hypertension mothers, however only 12 mothers have hypertension. Mothers with uterine irritability have a 50% chance of an underweight birth while normal mothers have only a 27% chance.

**Discuss the method results, including the classification accuracy for training set and test set and model evaluation metrics (precision, recall, ROC curve area).**

The model performed better on the training data than the test data in every recorded measure. This is because the model knows the correct answer on this data so it fits it better. The model was 70% accurate on the train data and 64% on the test. This demonstrates the model is able to distinguish between normal and low birthweight observations. Also, the test set is not significantly different to the training set.

The precision metric measures the proportion of positive predictions were actually correct. The model had a 56% precision metric on training data and 33% on the test data. The model produced many false positives on the test data. One way to alleviate this would be to make the prediction threshold higher than .5 which it currently is. That means the model will only give a True prediction on observations where it is very confident that it is true. The trade-off is the model will produce more false negatives as a result.

Recall is the proportion of positives that were actually identified and the model reported 33% and 25% on train and test data respectively. This shows there were many false negatives which lowered the recall metric value. A way to improve this would be to lower the threshold for rounding, reducing false negatives (the opposite of improving precision because there is a trade-off between them).

The ROC curve shows the tradeoff between the False Positive Rate and the True Positive Rate. Ideally, the model achieves 100% true positive rate and 0% false positive rate. With real world data this is almost never the case. The more a model’s cutoff is attempting to correctly detect True Positives, the more False Negatives will occur as well. The closer the line goes to the top left (closer to 0 FPR and 1 TPR) the better the model is at distinguishing the True’s from the False’s. The area under the curve is another way to make that measure. The AUC measure was .525 which mean the test does not perform well, nearly having no ability to accurately distinguish between classes.

The situation shown by these metrics of having generally good accuracy (64% on test) but AUC of .525 means the imbalanced target variable is causing problems. Also the precision and recall are not great is more evidence the imbalanced target is effecting the ability to distinguish accurately.

**Is the logistic regression method suitable for this study? Why or why not?**

Yes, the target variable is binary and the input variables can be numerical. A logistic regression requires the target variable to be between 0 and 1 while the input variables need to be numbers, either categorical or numerical. This data meets these conditions. Logistic regression may not be the best method, but it is a legitimate one.

**How would you improve the accuracy of your model?**

I would first handle the imbalanced target. This could be done a number of ways, like resampling the data so that an even number of True and False observations are in the training set, or duplicate True observations so there are more of them. I would look into weighting the True observations more heavily, making the model learn on these more.

Another approach I would use to improve accuracy is use various input variables. The model’s performance metrics don’t make it obvious that the model is overfitting the data (would have high train accuracy but low test accuracy if it did), but trying different input variables may help.

Another approach I would look into is more transformations on the input variables. Race needed to be converted to dummy variables so it could be used, but this could also be done for other variables. I could also bin variables like age into categorical from numerical.

**Discuss at least 2 alternative machine learning methods that could be suitable for this problem and explain why?**

Two other classification approaches could be gradient boosted trees and support vector machine (Both on spark’s MLlib). Both of them are capable of handling the given data (with some minor transformations). I would have to specify the models do a binary classification, which they are capable of handling if specified correctly (like a sigmoid activation function). I have had success with these two models.

**Part 3 – Research Questions**

**What is overfitting? What is the impact of overfitting on model performance? Discuss at least 2 approaches to avoid overfitting the model.**

Overfitting is when a model learns on a training set too closely which makes it inaccurate on test/unseen data later. This means a model performs well on the training data but poorly on other data. There are a variety of causes that can lead to this issue.

Using training data that is not similar to test data is one way. Imagine a mortgage lender has lots of data on mortgages over the past 30 years. This is what they will use to train a model that predicts which customers are risky to lend to. However, in recently there’s been new legislation which changed how the lending process works. The model is at risk for overfitting because it was trained on data that represents the old lending market, but it will be used on customers that live in the new lending market (or modern times). It may fit the training data well, but the data the model will be used on may not be similar to the old data because the world has changed in the intervening years. This is a type of sampling error, where the data is unrepresentative of the population. The model may fit the training data well, but that does not help the model on additional data because that data is different in nature. This fits the definition of overfitting as a model fits a sample of data too well to work well on other data. Another way training data may not be similar to test data is simple sampling error. This is when erroneous and random/chance relationships exist in the training data that don’t exist in the test data.

If a model uses too many input features, the model will pick up too much of the noise in the training data. In every data set there will be noise, the randomness that occurs in observations. For example, two people may both default on their mortgage but they are not the same person. They will not have the same job, be the same age, have the same income or same amount of debt. That is the random noise. A model is more likely to pick up noise when lots of features are included in its prediction. For example, a mortgage lender could include features that are unrelated to predicting a customer’s default risk, like person’s height in inches. The model will use it if it has some predictive power on the training set, but there’s no good reason to believe it actually relates to someone’s ability to pay back a loan. There may be some erroneous correlation between height and default risk in the training data. Including that variable would apply the same assumed (and incorrect) relationship to all additional data, adding noise which only makes detecting the signal in the data (making accurate predictions on unseen data) more difficult.

There are ways to alleviate this problem. One is find more data. This would reduce the potential sampling error and training on data that has random/erroneous relationships in the data. Another is to limit the number of features included in the model. This reduces the amount of noise the model picks up that doesn’t aide in making accurate predictions. Another is to use ensemble models. This helps handle noise and utilize the signal in the data (CFI, n.d.). Whenever a model performs well on training data but poorly on test data there is a potential overfitting is the culprit.

References

Corporate Finance Institute (CFI) (n.d.). Overfitting. Retrieved from https://corporatefinanceinstitute.com/resources/knowledge/other/overfitting/

**Discuss 5 (five) key differences between HDFS and Object Storage.**

Object storage has no hierarchical structure like the HDFS architecture, instead it is a flat across. HDFS uses the top-down path of the file to find where it exists in the database while object storage has no directory but a globally-unique identifier for the object.

Since it is flat address space oriented it is faster at scale because it doesn’t have to use long, complex directories when lots of data is stored in many folders and sub-folders. The metadata in object storage is more customizable and able to hold more information on the data it’s associated with than the metadata in HDFS.

The flexible metadata is a big advantage. Because it can hold lots of descriptors about the data it allows for better access to it. One can find things by location, date, user, source and any other imaginable information that might fall under metadata. HDFS has very little metadata in comparison.

Having a unique identifier makes it easier to retrieve data that is stored in a distributed storage system that is physically not near each other, like a two data centers on opposite sides of the country. The customizable metadata makes object storage more agile than file storage. Object storage can be accessed easily through API, which helps with low-latency web needs.

References

Alibaba Cloud (n.d.). Difference between object storage, file storage and block storage. Retrieved from: https://www.alibabacloud.com/knowledge/difference-between-object-storage-file-storage-block-storage

**We may use R, Python, Scala, and Java programming languages with Spark. Discuss the pros and cons of each language.**

Java is viewed as verbose, or too wordy to accomplish a task that other languages can do in less. Python is slow because it needs a Java Virtual Machine to use Spark. Python is simple and has libraries which make data science easier. Scala is a less common language and the syntax takes time to become comfortable with but it works fast with Spark. Spark is written in Scala so it works well with it. R is designed for statistics and visualizations, making data science work easy. R is not a general-purpose computer language however, so it cannot be used for easily for implementing into products.

References

Deshpande, S. (Oct. 2019). Scala Vs Python Vs R Vs Java - Which language is better for Spark & Why? Retrieved from: https://www.knowledgehut.com/blog/programming/scala-vs-python-vs-r-vs-java