Final Class Project Lending Club Dataset

Jiannliang Tsay
NYU-Cybersecurity
CS-GY6923
jlt245@nyu.edu

Outline

- What is Deep Learning
- Deep Learning in R package H2O
- What is Ensemble Learning
- Lending Club Loan Datasets
- Load and Inspect Datasets.
- Feature Selection and Data Conversion
- Perform Deep Learning in H2O
- H2O Ensemble Methods
- Conclusion
- References

What is Deep Learning

- Deep Learning is based on a set of algorithms in machine learning that attempt to model high level abstractions in data by using architectures composed of multiple linear and non-linear transformations. - Wikipedia

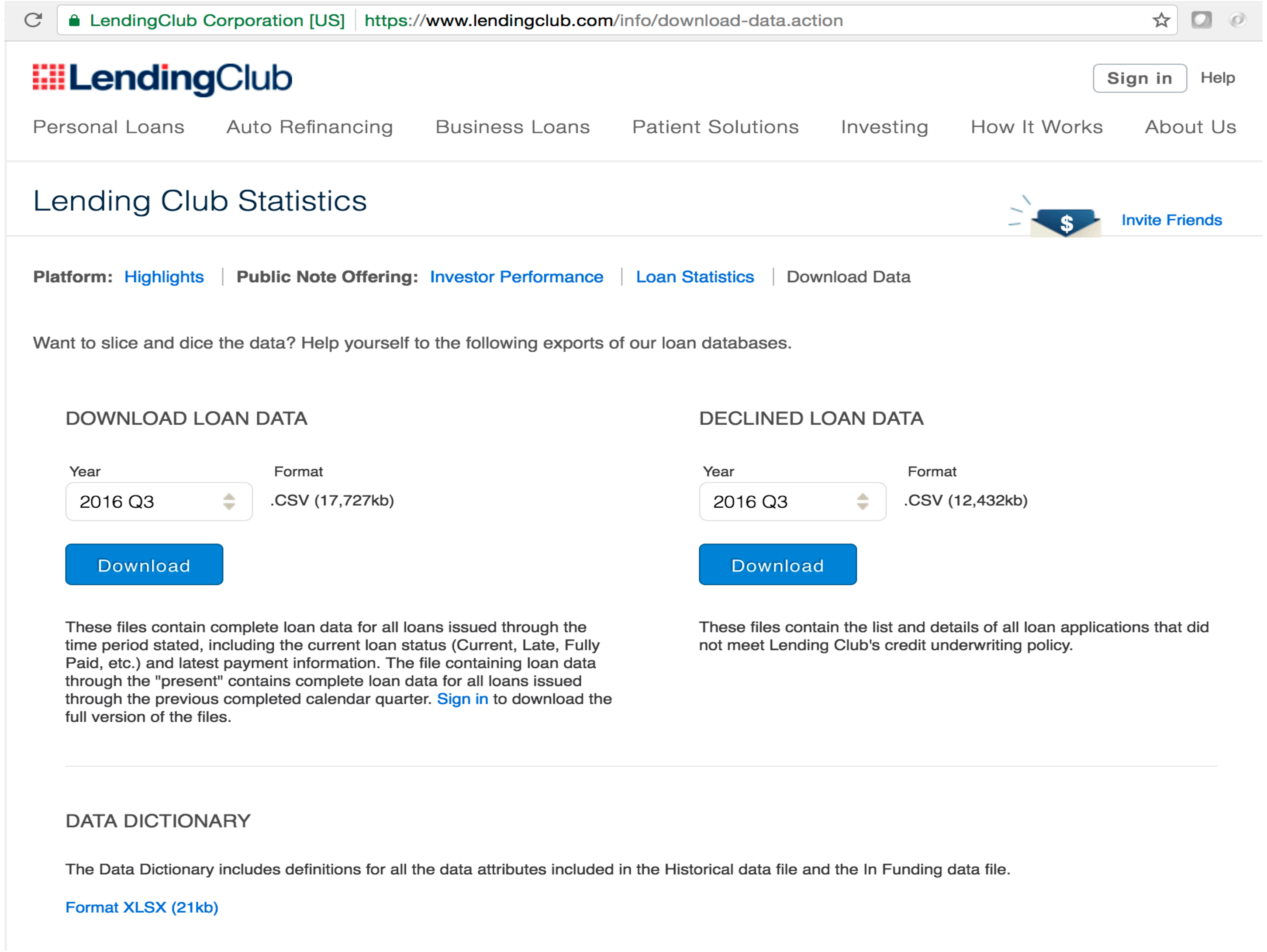
Deep Learning in R package H2O

- R scripting functionality for H2O, the open source math engine for big data that computes parallel distributed machine learning algorithms such as generalized linear models, gradient boosting machines, random forests, and neural networks (deep learning) within various cluster environments - https://cran.r-project.org/web/packages/h2o/h2o.pdf

What is Ensemble Learning

- Ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone. – Wikipedia
- H2O Ensemble implements the Super Learner ensemble (stacking) algorithm using the H2O R interface to provide base learning algorithms. http://www.stat.berkeley.edu/~ledell/R/h2oEnsemble.pdf

Lending Club Loan Datasets


LendingClub Corporation [US] https://www.lendingclub.com/info/download-data.action

LendingClub

Sign in Help

Personal Loans | Auto Refinancing | Business Loans | Patient Solutions | Investing | How It Works | About Us

Lending Club Statistics

$ Invite Friends

Platform: Highlights | **Public Note Offering:** Investor Performance | Loan Statistics | Download Data

Want to slice and dice the data? Help yourself to the following exports of our loan databases.

DOWNLOAD LOAN DATA

Year: 2016 Q3

Format: .CSV (17,727kb)

Download

These files contain complete loan data for all loans issued through the time period stated, including the current loan status (Current, Late, Fully Paid, etc.) and latest payment information. The file containing loan data through the "present" contains complete loan data for all loans issued through the previous completed calendar quarter. Sign in to download the full version of the files.

DECLINED LOAN DATA

Year: 2016 Q3

Format: .CSV (12,432kb)

Download

These files contain the list and details of all loan applications that did not meet Lending Club's credit underwriting policy.

DATA DICTIONARY

The Data Dictionary includes definitions for all the data attributes included in the Historical data file and the In Funding data file.

Format XLSX (21kb)

Load and Inspect Datasets

- > filenames <- c("LoanStats_2016Q1.csv", "LoanStats_2016Q2.csv", "LoanStats_2016Q3.csv")
- > data_list <- lapply(filenames, function (x) read_csv(file=x, skip=1))
- > data_frame <- do.call(rbind, data_list)
- > dim(data_frame)
- [1] 330867   111
- The size of the merged datasets is 111 features with 330k records. It is obviously too many to analyze and with a lot of redundancy.

Feature selection and Data conversion

- > loan_data <- data %>%+
- mutate(bad_loan = ifelse(loan_status=="Charged Off", 1, 0),+
- issue_d = mdy(issue_d),+
- earliest_cr_line = mdy(earliest_cr_line),+
- time_history = as.numeric(issue_d - earliest_cr_line),+
- revol_util = as.numeric(sub("%", "", revol_util)),+
- emp_listed = as.numeric(!is.na(emp_title) * 1),+
- empty_desc = as.numeric(is.na(desc)),+
- emp_na = ifelse(emp_length == "n/a", 1, 0),+
- emp_length = ifelse(emp_length == "< 1 year" | emp_length == "n/a", 0, emp_length),+
- emp_length = as.numeric(gsub("\\D", "", emp_length)),+
- delinq_ever = as.numeric(!is.na(mths_since_last_delinq)),+
- home_ownership = ifelse(home_ownership == "NONE", "OTHER", home_ownership)) %>%+
- select(bad_loan, loan_amnt, empty_desc, emp_listed, emp_na, emp_length, verification_status, home_ownership,+ annual_inc, purpose, time_history, inq_last_6mths, open_acc, pub_rec, revol_util, dti, total_acc,+ delinq_2yrs, delinq_ever, int_rate)
- > (ldd <- dim(loan_data))
- [1] 330867 20
- > colnames(loan_data) **# reduce the 111 columns into 20 meaningful features.**
- [1] "bad_loan" "loan_amnt" "empty_desc" "emp_listed" "emp_na"
- [6] "emp_length" "verification_status" "home_ownership" "annual_inc" "purpose"
- [11] "time_history" "inq_last_6mths" "open_acc" "pub_rec" "revol_util"
- [16] "dti" "total_acc" "delinq_2yrs" "delinq_ever" "int_rate”

Perform Deep Learning

Rscript jlt245_class_project_bad_loan_predict_deep_learning.R

be aware that the run time may take a few hours to complete the above script.

```r
# perform a 5-fold cross-validation deep learning model and validate on a test set
model <- h2o.deeplearning (
  x = x,
  y = y,
  training_frame = train,
  validation_frame = test,
  distribution = "multinomial",
  activation = "RectifierWithDropout",
  hidden = c(64, 128, 64),
  input_dropout_ratio = 0.2,
  l1 = 1e-5,
  epochs = 10,
  nfolds = 5
)

predictions <- predict(object = model, newdata = test)
perf <- h2o.performance(model, test)
perf
```

Comparisons of performance

```
Comparisons of performances:

[1] 5-fold CV, hidden = 64, 128, 64 nodes

  H2OBinomialMetrics: deeplearning

  MSE:  0.003277576
  RMSE:  0.05725012
  LogLoss:  0.02733859
  Mean Per-Class Error:  0.4864729
  AUC:  0.7231921
  Gini:  0.4463842

[2] 10-fold CV, hidden = 64, 128, 64 nodes

  H2OBinomialMetrics: deeplearning

  MSE:  0.003278861
  RMSE:  0.05726134
  LogLoss:  0.03098881
  Mean Per-Class Error:  0.4469392
  AUC:  0.7248728
  Gini:  0.4497456

 * 10-fold CV seems to improve a bit, but not that much in this case: AUC from 0.7232 to 0.7249 *


[3] 5-fold CV, hidden = 64, 128, 2, 128, 64 nodes

  H2OBinomialMetrics: deeplearning

  MSE:  0.003276932
  RMSE:  0.0572445
  LogLoss:  0.02568059
  Mean Per-Class Error:  0.5
  AUC:  0.5
  Gini:  0

 * Not good for 5 hidden layers in this case *
```

H2O Ensemble Methods

```
library(h2oEnsemble)
# Specify the base learner library & the metalearner
learner <- c("h2o.glm.wrapper", "h2o.randomForest.wrapper",
             "h2o.gbm.wrapper", "h2o.deeplearning.wrapper")

metalearner <- "h2o.glm.wrapper"
family <- "binomial"


# Train the ensemble using 5-fold CV to generate level-one data
# More CV folds will take longer to train, but should increase performance
fit <- h2o.ensemble(x = x, y = y,
                    training_frame = train,
                    family = family,
                    learner = learner,
                    metalearner = metalearner,
                    cvControl = list(V = 5, shuffle = TRUE))


# Evaluate performance on a test set by h2o.ensemble_performance
perf <- h2o.ensemble_performance(fit, newdata = test)
perf
```

Ensemble Methods Performance

```
 Rscript jlt245_class_project_bad_loan_predict_ensemble.R


The results are:

  Base learner performance, sorted by specified metric:
                     learner       AUC
  4 h2o.deeplearning.wrapper 0.5256778
  2 h2o.randomForest.wrapper 0.5853513
  1          h2o.glm.wrapper 0.6195399
  3          h2o.gbm.wrapper 0.6588160

  H2O Ensemble Performance on <newdata>:
  ----------------
  Family: binomial

  Ensemble performance (AUC): 0.635881662244555

and further algorithms case, the results are:

  Base learner performance, sorted by specified metric:
                learner       AUC
  14 h2o.deeplearning.2 0.5263948
  13 h2o.deeplearning.1 0.5401991
  15 h2o.deeplearning.3 0.5490740
  9           h2o.gbm.3 0.6138080
  1           h2o.glm.1 0.6183734
  2           h2o.glm.2 0.6195399
  3           h2o.glm.3 0.6208864
  4  h2o.randomForest.1 0.6322672
  6  h2o.randomForest.3 0.6417271
  5  h2o.randomForest.2 0.6447206
  8           h2o.gbm.2 0.6488665
  7           h2o.gbm.1 0.6501414
  10          h2o.gbm.4 0.6532700
  11          h2o.gbm.5 0.6532700
  12          h2o.gbm.6 0.6552727


  H2O Ensemble Performance on <newdata>:
  ----------------
  Family: binomial

  Ensemble performance (AUC): 0.644141126382818

 * as can be seen that Ensemble performance is above the average of the performance from all base algorithms *
```

Conclusion

- A few key points of achieving better performance are:
  - 1. feature selection - I did try some additional fields or narrow down to less fields to come up different results.
  - 2. model selection - deep learning comes with some alternatives such as distribution, hidden layers etc. that does impact the performance. In addition, different algorithms run against different datasets may achieve different performance.
  - 3. Ensemble method - seems to work out an average performance among the chosen base algorithms. Although I question that as it is supposed to get better results from individual algorithms.

References

- [1] https://h2o-release.s3.amazonaws.com/h2o/rel-turan/4/docs-website/h2o-docs/booklets/R_Vignette.pdf
- [2] https://h2o-release.s3.amazonaws.com/h2o/rel-tibshirani/8/docs-website/h2o-docs/booklets/DeepLearning_Vignette.pdf
- [3] http://docs.h2o.ai/h2o/latest-stable/h2o-docs/index.html
- [4] http://www.dataversity.net/efficient-machine-learning-h2o-r-python-part-1/
- [5] Ensemble Methods (Foundations and Algorithms) by Zhi-Hua Zhou
- [6] http://www.stat.berkeley.edu/~ledell/R/h2oEnsemble.pdf
- [7] https://rdrr.io/cran/h2o/man/h2o.prcomp.html
- [8] https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principal-component-analysis-python/
- [9] Deep Learning by Ian Goodfellow, Yoshua Bengio, Aaron Courville