MITx: 15.071x The Analytics Edge

Courseware Course Info Discussion Progress Syllabus Schedule Files Wiki

PREDICTING STOCK RETURNS WITH CLUSTER-THEN-PREDICT

In the second lecture sequence this week, we heard about cluster-then-predict, a methodology in which you first cluster observations and then build cluster-specific prediction models. In the lecture sequence, we saw how this methodology helped improve the prediction of heart attack risk. In this assignment, we'll use cluster-then-predict to predict future stock prices using historical stock data.

When selecting which stocks to invest in, investors seek to obtain good future returns. In this problem, we will first use clustering to identify clusters of stocks that have similar returns over time. Then, we'll use logistic regression to predict whether or not the stocks will have positive future returns.

For this problem, we'll use StocksCluster.csv, which contains monthly stock returns from the NASDAQ stock exchange. The NASDAQ is the second-largest stock exchange in the world, and it lists many technology companies. The stock price data used in this problem was obtained from infochimps, a website providing access to many datasets.

Each observation in the dataset is the monthly returns of a particular company in a particular year. The years included are 2000-2009. The companies are limited to tickers that were listed on the exchange for the entire period 2000-2009, and whose stock price never fell below $1. So, for example, one observation is for Yahoo in 2000, and another observation is for Yahoo in 2001. Our goal will be to predict whether or not the stock return in December will be positive, using the stock returns for the first 11 months of the year.

This dataset contains the following variables:

- **ReturnJan** = the return for the company's stock during January (in the year of the observation).
- **ReturnFeb** = the return for the company's stock during February (in the year of the observation).
- **ReturnMar** = the return for the company's stock during March (in the year of the observation).
- **ReturnApr** = the return for the company's stock during April (in the year of the observation).
- **ReturnMay** = the return for the company's stock during May (in the year of the observation).
- **ReturnJune** = the return for the company's stock during June (in the year of the observation).
- **ReturnJuly** = the return for the company's stock during July (in the year of the observation).
- **ReturnAug** = the return for the company's stock during August (in the year of the observation).
- **ReturnSep** = the return for the company's stock during September (in the year of the observation).
- **ReturnOct** = the return for the company's stock during October (in the year of the observation).
- **ReturnNov** = the return for the company's stock during November (in the year of the observation).
- **PositiveDec** = whether or not the company's stock had a positive return in December (in the year of the observation). This variable takes value 1 if the return was positive, and value 0 if the return was not positive.

For the first 11 variables, the value stored is a proportional change in stock value during that month. For instance, a value of 0.05 means the stock increased in value 5% during the month, while a value of -0.02 means the stock decreased in value 2% during the month.

PROBLEM 1.1 - EXPLORING THE DATASET (1/1 point)

Load StocksCluster.csv into a data frame called "stocks". How many observations are in the dataset?

11580

Answer: 11580

EXPLANATION

You can load the dataset with the read.csv function:

stocks = read.csv("StocksCluster.csv")

and see how many observations are included with either the str or nrow function:

str(stocks)

nrow(stocks)

Both tell us that there are 11580 observations in this dataset.

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 1.2 - EXPLORING THE DATASET (1/1 point)

What proportion of the observations have positive returns in December?

0.546114

Answer: 0.546114

EXPLANATION

You can compute the proportion of observations with positive returns by using the table function:

table(stocks$PositiveDec)

It tells us that 6324 observations have PositiveDec = 1, so 6324/11580 = 0.546 of the observations have positive returns in December.

Alternatively, you could use the mean function to compute the proportion:

mean(stocks$PositiveDec)

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 1.3 - EXPLORING THE DATASET (1 point possible)

What is the maximum correlation between any two return variables in the dataset? You should look at the pairwise correlations between ReturnJan, ReturnFeb, ReturnMar, ReturnApr, ReturnMay, ReturnJune, ReturnJuly, ReturnAug, ReturnSep, ReturnOct, and ReturnNov.

1

Answer: 0.19167279

EXPLANATION

From cor(stocks), we see the largest correlation coefficient is 0.19167279, between ReturnOct and ReturnNov.

Hide Answer *You have used 3 of 3 submissions*

PROBLEM 1.4 - EXPLORING THE DATASET (2/2 points)

Which month (from January through November) has the largest mean return across all observations in the dataset?

April ▾ April

Which month (from January through November) has the smallest mean return across all observations in the dataset?

September ▾ September

EXPLANATION

These can be determined using the summary function:

summary(stocks)

If you look at the mean value for each variable, you can see that April has the largest mean value (0.026308), and September has the smallest mean value (-0.014721).

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 2.1 - INITIAL LOGISTIC REGRESSION MODEL (2/2 points)

Run the following commands to split the data into a training set and testing set, putting 70% of the data in the training set and 30% of the data in the testing set:

set.seed(144)

spl = sample.split(stocks$PositiveDec, SplitRatio = 0.7)

stocksTrain = subset(stocks, spl == TRUE)

stocksTest = subset(stocks, spl == FALSE)

Then, use the stocksTrain data frame to train a logistic regression model (name it StocksModel) to predict PositiveDec using all the other variables as independent variables. Don't forget to add the argument family=binomial to your glm command.

What is the overall accuracy on the training set, using a threshold of 0.5?

0.5711818 **Answer:** 0.5711818

EXPLANATION

We can train the model with:

```
StocksModel = glm(PositiveDec ~ ., data=stocksTrain, family=binomial)
```

Then, we can compute our predictions on the training set with:

```
PredictTrain = predict(StocksModel, type="response")
```

And construct a classification matrix with the table function:

```
table(stocksTrain$PositiveDec, PredictTrain > 0.5)
```

The overall accuracy of the model is (990 + 3640)/(990 + 2689 + 787 + 3640) = 0.571.

Hide Answer *You have used 1 of 5 submissions*

PROBLEM 2.2 - INITIAL LOGISTIC REGRESSION MODEL (1/1 point)

Now obtain test set predictions from StocksModel. What is the overall accuracy of the model on the test, again using a threshold of 0.5?

0.5670697

Answer: 0.5670697

EXPLANATION

You can compute predictions on the test set using the predict function:

```
PredictTest = predict(StocksModel, newdata=stocksTest, type="response")
```

Then, you can compute the classification matrix on the test set with the table function:

```
table(stocksTest$PositiveDec, PredictTest > 0.5)
```

The overall accuracy of the model on the test set is (417 + 1553)/(417 + 1160 + 344 + 1553) = 0.567

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 2.3 - INITIAL LOGISTIC REGRESSION MODEL (1/1 point)

What is the accuracy on the test set of a baseline model that always predicts the most common outcome (PositiveDec = 1)?

0.5460564

Answer: 0.5460564

EXPLANATION

This can be computed by making a table of the outcome variable in the test set:

```
table(stocksTest$PositiveDec)
```

The baseline model would get all of the PositiveDec = 1 cases correct, and all of the PositiveDec = 0 cases wrong, for an accuracy of 1897/(1577 + 1897) = 0.5460564.

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 3.1 - CLUSTERING STOCKS (1/1 point)

Now, let's cluster the stocks. The first step in this process is to remove the dependent variable using the following commands:

```
limitedTrain = stocksTrain

limitedTrain$PositiveDec = NULL

limitedTest = stocksTest

limitedTest$PositiveDec = NULL
```

Why do we need to remove the dependent variable in the clustering phase of the cluster-then-predict methodology?

- ○ Leaving in the dependent variable might lead to unbalanced clusters
- ○ Removing the dependent variable decreases the computational effort needed to cluster
- ● Needing to know the dependent variable value to assign an observation to a cluster defeats the purpose of the methodology ✓

EXPLANATION

In cluster-then-predict, our final goal is to predict the dependent variable, which is unknown to us at the time of prediction. Therefore, if we need to know the outcome value to perform the clustering, the methodology is no longer useful for prediction of an unknown outcome value.

This is an important point that is sometimes mistakenly overlooked. If you use the outcome value to cluster, you might conclude your method strongly outperforms a non-clustering alternative. However, this is because it is using the outcome to determine the clusters, which is not valid.

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 3.2 - CLUSTERING STOCKS (2/2 points)

In the market segmentation assignment in this week's homework, you were introduced to the preProcess command from the caret package, which normalizes variables by subtracting by the mean and dividing by the standard deviation.

In cases where we have a training and testing set, we'll want to normalize by the mean and standard deviation of the variables in the training set. We can do this by passing just the training set to the preProcess function:

```
library(caret)

preproc = preProcess(limitedTrain)
```

normTrain = predict(preproc, limitedTrain)

normTest = predict(preproc, limitedTest)

What is the mean of the ReturnJan variable in normTrain?

2.100586e-17 **Answer:** 2.100586e-17

What is the mean of the ReturnJan variable in normTest?

-0.0004185886 **Answer:** -0.0004185886

EXPLANATION

After running the provided normalization commands, we can read the means with mean(normTrain$ReturnJan) and mean(normTest$ReturnJan).

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 3.3 - CLUSTERING STOCKS (1/1 point)

Why is the mean ReturnJan variable much closer to 0 in normTrain than in normTest?

○ Small rounding errors exist in the normalization procedure
● The distribution of the ReturnJan variable is different in the training and testing set ✓
○ The distribution of the dependent variable is different in the training and testing set

EXPLANATION

From mean(stocksTrain$ReturnJan) and mean(stocksTest$ReturnJan), we see that the average return in January is slightly higher in the training set than in the testing set. Since normTest was constructed by subtracting by the mean ReturnJan value from the training set, this explains why the mean value of ReturnJan is slightly negative in normTest.

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 3.4 - CLUSTERING STOCKS (1/1 point)

Set the random seed to 144 (it is important to do this again, even though we did it earlier). Run k-means clustering with 3 clusters on normTrain, storing the result in an object called km.

Which cluster has the largest number of observations?

○ Cluster 1
● Cluster 2 ✓
○ Cluster 3

EXPLANATION

We can set the seed and run the k-means algorithm with:

```
set.seed(144)

km = kmeans(normTrain, centers = 3)
```

From table(km$cluster), we can see that cluster 2 has the largest number of observations. Alternatively, you can see the number of observations in each cluster by typing km$size in your console.

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 3.5 - CLUSTERING STOCKS (1/1 point)

Recall from the recitation that we can use the flexclust package to obtain training set and testing set cluster assignments for our observations (note that the call to as.kcca may take a while to complete):

```
library(flexclust)

km.kcca = as.kcca(km, normTrain)

clusterTrain = predict(km.kcca)

clusterTest = predict(km.kcca, newdata=normTest)
```

How many test-set observations were assigned to Cluster 2?

2080 **Answer:** 2080

EXPLANATION

After running the provided commands, we can obtain the breakdown of the testing set clusters with table(clusterTest).

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 4.1 - CLUSTER-SPECIFIC PREDICTIONS (1 point possible)

Using the subset function, build data frames stocksTrain1, stocksTrain2, and stocksTrain3, containing the elements in the stocksTrain data frame assigned to clusters 1, 2, and 3, respectively (be careful to take subsets of stocksTrain, not of normTrain). Similarly build stocksTest1, stocksTest2, and stocksTest3 from the stocksTest data frame.

Which training set data frame has the highest average value of the dependent variable?

○ stocksTrain1 ✓
○ stocksTrain2
○ stocksTrain3

EXPLANATION

We can obtain the necessary subsets with:

stocksTrain1 = subset(stocksTrain, clusterTrain == 1)

stocksTrain2 = subset(stocksTrain, clusterTrain == 2)

stocksTrain3 = subset(stocksTrain, clusterTrain == 3)

stocksTest1 = subset(stocksTest, clusterTest == 1)

stocksTest2 = subset(stocksTest, clusterTest == 2)

stocksTest3 = subset(stocksTest, clusterTest == 3)

From mean(stocksTrain1$PositiveDec), mean(stocksTrain2$PositiveDec), and mean(stocksTrain3$PositiveDec), we see that stocksTrain1 has the observations with the highest average value of the dependent variable.

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 4.2 - CLUSTER-SPECIFIC PREDICTIONS (2 points possible)

Build logistic regression models StocksModel1, StocksModel2, and StocksModel3, which predict PositiveDec using all the other variables as independent variables. StocksModel1 should be trained on stocksTrain1, StocksModel2 should be trained on stocksTrain2, and StocksModel3 should be trained on stocksTrain3.

Which variables have a positive sign for the coefficient in at least one of StocksModel1, StocksModel2, and StocksModel3 and a negative sign for the coefficient in at least one of StocksModel1, StocksModel2, and StocksModel3? Select all that apply.

- ☐ ReturnJan ✓
- ☐ ReturnFeb ✓
- ☐ ReturnMar ✓
- ☐ ReturnApr
- ☐ ReturnMay
- ☐ ReturnJune ✓
- ☐ ReturnJuly
- ☐ ReturnAug ✓
- ☐ ReturnSep
- ☐ ReturnOct ✓
- ☐ ReturnNov

EXPLANATION

We can build the models with:

StocksModel1 = glm(PositiveDec ~ ., data=stocksTrain1, family=binomial)

StocksModel2 = glm(PositiveDec ~ ., data=stocksTrain2, family=binomial)

StocksModel3 = glm(PositiveDec ~ ., data=stocksTrain3, family=binomial)

From summary(StocksModel1), summary(StocksModel2), and summary(StocksModel3), ReturnJan, ReturnFeb, ReturnMar, ReturnJune, ReturnAug, and ReturnOct differ in sign between the models.

Hide Answer *You have used 3 of 3 submissions*

PROBLEM 4.3 - CLUSTER-SPECIFIC PREDICTIONS (6/6 points)

Using StocksModel1, make test-set predictions called PredictTest1 on the data frame stocksTest1. Using StocksModel2, make test-set predictions called PredictTest2 on the data frame stocksTest2. Using StocksModel3, make test-set predictions called PredictTest3 on the data frame stocksTest3.

What is the overall accuracy of StocksModel1 on the test set stocksTest1, using a threshold of 0.5?

0.6194145 **Answer:** 0.6194145

What is the overall accuracy of StocksModel2 on the test set stocksTest2, using a threshold of 0.5?

0.5504808 **Answer:** 0.5504808

What is the overall accuracy of StocksModel3 on the test set stocksTest3, using a threshold of 0.5?

0.6458333 **Answer:** 0.6458333

EXPLANATION

The predictions can be obtained with:

```
PredictTest1 = predict(StocksModel1, newdata = stocksTest1, type="response")

PredictTest2 = predict(StocksModel2, newdata = stocksTest2, type="response")

PredictTest3 = predict(StocksModel3, newdata = stocksTest3, type="response")
```

And the classification matrices can be computed with:

```
table(stocksTest1$PositiveDec, PredictTest1 > 0.5)

table(stocksTest2$PositiveDec, PredictTest2 > 0.5)

table(stocksTest3$PositiveDec, PredictTest3 > 0.5)
```

The overall accuracy of StocksModel1 is (30 + 774)/(30 + 471 + 23 + 774) = 0.6194145, the overall accuracy of StocksModel2 is (388 + 757)/(388 + 626 + 309 + 757) = 0.5504808, and the overall accuracy of StocksModel3 is (49 + 13)/(49 + 13 + 21 + 13) = 0.6458333.

Hide Answer *You have used 1 of 5 submissions*

PROBLEM 4.4 - CLUSTER-SPECIFIC PREDICTIONS (1/1 point)

To compute the overall test-set accuracy of the cluster-then-predict approach, we can combine all the test-set predictions into a single vector and all the true outcomes into a single vector:

AllPredictions = c(PredictTest1, PredictTest2, PredictTest3)

AllOutcomes = c(stocksTest1$PositiveDec, stocksTest2$PositiveDec, stocksTest3$PositiveDec)

What is the overall test-set accuracy of the cluster-then-predict approach, again using a threshold of 0.5?

0.5788716

Answer: 0.5788716

EXPLANATION

After combining the predictions and outcomes with the provided code, we can compute the overall test-set accuracy by creating a classification matrix:

table(AllOutcomes, AllPredictions > 0.5)

Which tells us that the overall accuracy is (467 + 1544)/(467 + 1110 + 353 + 1544) = 0.5788716.

We see a modest improvement over the original logistic regression model. Since predicting stock returns is a notoriously hard problem, this is a good increase in accuracy. By investing in stocks for which we are more confident that they will have positive returns (by selecting the ones with higher predicted probabilities), this cluster-then-predict model can give us an edge over the original logistic regression model.

Hide Answer *You have used 1 of 3 submissions*

Please remember not to ask for or post complete answers to homework questions in this discussion forum.

Show Discussion New Post

‹ ›

math, medicine, music, philosophy, physics, science, statistics and more. EdX is a non-profit online initiative created by founding partners Harvard and MIT.

Terms of Service and Honor Code

Privacy Policy (Revised 10/22/2014)

edX Blog

Donate to edX

Jobs at edX

Google+

Tumblr

Meetup

Reddit

Youtube