PREDICTING MEDICAL COSTS 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 medical costs using medical claims data.

In Week 4, we discussed the importance of high-quality predictions of future medical costs based on information available in medical claims data. In this problem, you will predict future medical claims using part of the DE-SynPUF dataset (http://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/SynPUFs/DE_Syn_PUF.html), published by the United States Centers for Medicare and Medicaid Services (CMS). This dataset, available in reimbursement.csv (/c4x/MITx/15.071x/asset/reimbursement.csv), is structured to represent a sample of patients in the Medicare program, which provides health insurance to Americans aged 65 and older as well as some younger people with certain medical conditions. To protect the privacy of patients represented in this publicly available dataset, CMS performs a number of steps to anonymize the data, so we would need to re-train the models we develop in this problem on de-anonymized data if we wanted to apply our models in the real world.

The observations in the dataset represent a 1% random sample of Medicare beneficiaries in 2008, limited to those still alive at the end of 2008. The dependent variable, **reimbursement2009**, represents the total value of all Medicare reimbursements for a patient in 2009, which is the cost of the patient's care to the Medicare system. The following independent variables are available:

- **age**: The patient's age in years at the beginning of 2009
- **alzheimers:** Binary variable for whether the patient had diagnosis codes for Alzheimer's disease or a related disorder in 2008
- **arthritis**: Binary variable for whether the patient had diagnosis codes for rheumatoid arthritis or osteoarthritis in 2008
- **cancer**: Binary variable for whether the patient had diagnosis codes for cancer in 2008
- **copd**: Binary variable for whether the patient had diagnosis codes for Chronic Obstructive Pulmonary Disease (COPD) in 2008
- **depression**: Binary variable for whether the patient had diagnosis codes for depression in 2008
- **diabetes**: Binary variable for whether the patient had diagnosis codes for diabetes in 2008
- **heart.failure**: Binary variable for whether the patient had diagnosis codes for heart failure in 2008
- **ihd**: Binary variable for whether the patient had diagnosis codes for ischemic heart disease (IHD) in 2008
- **kidney**: Binary variable for whether the patient had diagnosis codes for chronic kidney disease in 2008
- **osteoporosis**: Binary variable for whether the patient had diagnosis codes for osteoporosis in 2008
- **stroke**: Binary variable for whether the patient had diagnosis codes for a stroke/transient ischemic attack (TIA) in 2008
- **reimbursement2008**: The total amount of Medicare reimbursements for this patient for 2008

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

Load reimbursement.csv into a data frame called claims.

How many Medicare beneficiaries are included in the dataset?

458005

458005

Answer: 458005

EXPLANATION

The dataset can be loaded with:

claims = read.csv("reimbursement.csv")

We can read the number of observations with nrow(claims) or from str(claims).

Hide Answer *You have used 1 of 3 submissions*

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

What proportion of patients have at least one of the chronic conditions described in the independent variables alzheimers, arthritis, cancer, copd, depression, diabetes, heart.failure, ihd, kidney, osteoporosis, and stroke?

0.6122793

0.6122793

Answer: 0.6122793

EXPLANATION

This can be obtained by checking if any of the condition variables are true:

table(claims$alzheimers == 1 | claims$arthritis == 1 | claims$cancer == 1 | claims$copd == 1 | claims$depression == 1 | claims$diabetes == 1 | claims$heart.failure == 1 | claims$ihd == 1 | claims$kidney == 1 | claims$osteoporosis == 1 | claims$stroke == 1)

280427 of 458005 patients had at least one condition, for a proportion of 280427/458005=0.6122793.

Another approach would have been to use subset and then nrow:

has.condition = subset(claims, alzheimers == 1 | arthritis == 1 | cancer == 1 | copd == 1 | depression == 1 | diabetes == 1 | heart.failure == 1 | ihd == 1 | kidney == 1 | osteoporosis == 1 | stroke == 1)

nrow(has.condition)

Hide Answer *You have used 2 of 3 submissions*

PROBLEM 1.3 - PREPARING THE DATASET (1/1 point)

What is the maximum correlation between independent variables in the dataset?

0.51461422

0.51461422

Answer: 0.51461422

EXPLANATION

From cor(claims), we see the largest correlation coefficient is 0.51461422, between ihd and diabetes.

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 1.4 - PREPARING THE DATASET (1/1 point)

Plot the histogram of the dependent variable. What is the shape of the distribution?

- (•) Skew right -- there are a large number of observations with a small value, but only a small number of observations with a large value. ✔
- ( ) Balanced -- there are roughly the same number of observations with an unusually large and unusually small value.
- ( ) Skew left -- there are a large number of observations with a large value, but only a small number of observations with a small value.

EXPLANATION

From hist(claims$reimbursement2009), we can see the the vast majority of reimbursements had a small value, but a small number of patients had very large reimbursement totals.

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 1.5 - PREPARING THE DATASET (1 point possible)

To address the shape of the data identified in the previous problem, we will log transform the two reimbursement variables with the following code:

claims$reimbursement2008 = log(claims$reimbursement2008+1)

claims$reimbursement2009 = log(claims$reimbursement2009+1)

Why did we take the log of the reimbursement value plus 1 instead of the log of the reimbursement value? Hint -- What happens when a patient has a reimbursement cost of $0?

- (•) Every patient in Medicare gets at least $1 in reimbursement ✖
- ( ) To avoid log-transformed values of negative infinity ✔
- ( ) To avoid log-transformed values of infinity
- ( ) There was no reason

EXPLANATION

From summary(claims$reimbursement2008) and summary(claims$reimbursement2009) before the transformation, we see that the minimum value of each variable is 0. log(0)=-infinity. By adding 1 before taking the log, we ensure the minimum transformed value is 0, because log(1)=0.

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 1.6 - PREPARING THE DATASET (1/1 point)

Plot the histogram of the log-transformed dependent variable. The distribution is reasonably balanced, other than a large number of people with variable value 0, corresponding to having had $0 in reimbursements in 2009. What proportion of beneficiaries had $0 in reimbursements in 2009?

0.1975917

0.1975917

Answer: 0.1975917

EXPLANATION

The histogram can be plotted with hist(claims$reimbursement2009). From table(claims$reimbursement2009 == 0), we see that 90284 beneficiaries had $0 in 2009 reimbursements, for a proportion of 90498/458005 = 0.1975917.

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 2.1 - INITIAL LINEAR REGRESSION MODEL (1/1 point)

In Week 3 when we learned about the sample.split function, we mentioned that you split data into a training and testing set a bit differently when there is a continuous outcome. Run the following commands to randomly select 70% of the data for the training set and 30% of the data for the testing set:

```
set.seed(144)

spl = sample(1:nrow(claims), size=0.7*nrow(claims))

train = claims[spl,]

test = claims[-spl,]
```

Use the train data frame to train a linear regression model (name it lm.claims) to predict reimbursement2009 using all the independent variables.

What is the training set Multiple R-squared value of lm.claims?

0.6924

0.6924

Answer: 0.6924

EXPLANATION

We can train the model with:

```
lm.claims = lm(reimbursement2009~., data=train)
```

From summary(lm.claims), we see that we achieved a Multiple R-squared value of 0.6924.

Hide Answer *You have used 1 of 3 submissions*

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

Obtain testing set predictions from lm.claims. What is the testing set RMSE of the model?

1.849212

1.849212

Answer: 1.849212

EXPLANATION

This can be obtained with:

```
pred.test = predict(lm.claims, newdata=test)

rmse.lm = sqrt(mean((pred.test - test$reimbursement2009)^2))
```

Alternately, we could compute the RMSE in two steps with:

```
sse.lm = sum((pred.test - test$reimbursement2009)^2)

rmse.lm = sqrt(sse.lm / nrow(test))
```

Hide Answer *You have used 1 of 3 submissions*

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

What is the "naive baseline model" that we would typically use to compute the R-squared value of lm.claims?

- Predict 0 for every observation
- Predict mean(train$reimbursement2008) for every observation
- Predict mean(test$reimbursement2008) for every observation
- Predict mean(train$reimbursement2009) for every observation ✔
- Predict mean(test$reimbursement2009) for every observation

EXPLANATION

The naive baseline predicts the average of the dependent variable (reimbursement2009) on the training set. Just like our models, the naive baseline is not allowed to learn from the testing set, so it's not allowed to predict mean(test$reimbursement2009).

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 2.4 - INITIAL LINEAR REGRESSION MODEL (1/1 point)

What is the testing set RMSE of the naive baseline model?

3.335486

3.335486

Answer: 3.335486

EXPLANATION

This can be obtained by first computing the naive baseline prediction, which is the mean of the dependent variable in the training set, and then computing the testing set RMSE of this prediction:

baseline.pred = mean(train$reimbursement2009)

sqrt(mean((baseline.pred - test$reimbursement2009)^2))

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 2.5 - INITIAL LINEAR REGRESSION MODEL (1/1 point)

In Week 4, we saw how D2Hawkeye used a "smart baseline model" that predicted that a patient's medical costs would be equal to their costs in the previous year. For our problem, this baseline would predict reimbursement2009 to be equal to reimbursement2008.

What is the testing set RMSE of this smart baseline model?

2.094668

2.094668

Answer: 2.094668

EXPLANATION

This RMSE can be computed with:

sqrt(mean((test$reimbursement2008 - test$reimbursement2009)^2))

As we can see, this smart baseline is much more competitive with our linear regression model than the naive baseline.

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 3.1 - CLUSTERING MEDICARE BENEFICIARIES (1/1 point)

In this section, we will cluster the Medicare beneficiaries. The first step in this process is to remove the dependent variable using the following commands:

train.limited = train

train.limited$reimbursement2009 = NULL

test.limited = test

test.limited$reimbursement2009 = 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 MEDICARE BENEFICIARIES (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(train.limited)

train.norm = predict(preproc, train.limited)

test.norm = predict(preproc, test.limited)

What is the mean of the arthritis variable in train.norm?

2.048714e-17

2.048714×10^{-17}

Answer: 2.048714e-17

What is the mean of the arthritis variable in test.norm?

-0.006124962

-0.006124962

Answer: -0.006124962

EXPLANATION

After running the provided normalization commands, we can read the means with mean(train.norm$arthritis) and mean(test.norm$arthritis).

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 3.3 - CLUSTERING MEDICARE BENEFICIARIES (1/1 point)

Why is the mean arthritis variable much closer to 0 in train.norm than in test.norm?

- Small rounding errors exist in the normalization procedure
- The distribution of the arthritis 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(train$arthritis) and mean(test$arthritis), we see that a slightly higher proportion of patients had arthritis in the training set than in the testing set. Since test.norm was constructed by subtracting by the mean arthritis value from the training set, this explains why the mean value of arthritis is slightly negative in test.norm.

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 3.4 - CLUSTERING MEDICARE BENEFICIARIES (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 train.norm, storing the result in an object called km.

The description "older-than-average beneficiaries with below average incidence of stroke and above-average 2008 reimbursements" uniquely describes which cluster center?

- Cluster 1
- Cluster 2
- Cluster 3 ✔

EXPLANATION

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

set.seed(144)

km = kmeans(train.norm, centers=3)

From km$centers, we can see that the center of cluster 3 has above-average age and reimbursement2008 value (positive normalized value), but below-average incidence of stroke (negative normalized value).

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 3.5 - CLUSTERING MEDICARE BENEFICIARIES (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, train.norm)

cluster.train = predict(km.kcca)

cluster.test = predict(km.kcca, newdata=test.norm)
```

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

62651

62651

Answer: 62650

EXPLANATION

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

Hide Answer *You have used 2 of 3 submissions*

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

Using the subset function, build data frames train1, train2, and train3, containing the elements in the train data frame assigned to clusters 1, 2, and 3, respectively (be careful to take subsets of train, not of train.norm). Similarly build test1, test2, and test3 from the test data frame.

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

- (•) train1 ✔
- ( ) train2
- ( ) train3

EXPLANATION

We can obtain the necessary subsets with:

```
train1 = subset(train, cluster.train == 1)

train2 = subset(train, cluster.train == 2)

train3 = subset(train, cluster.train == 3)

test1 = subset(test, cluster.test == 1)

test2 = subset(test, cluster.test == 2)

test3 = subset(test, cluster.test == 3)
```

From mean(train1$reimbursement2009), mean(train2$reimbursement2009), and mean(train3$reimbursement2009), we see that train1 has the patients with the highest average value of the dependent variable.

Hide Answer *You have used 1 of 1 submissions*

Help

PROBLEM 4.2 - CLUSTER-SPECIFIC PREDICTIONS (1/1 point)

Build linear regression models lm1, lm2, and lm3, which predict reimbursement2009 using all the variables. lm1 should be trained on train1, lm2 should be trained on train2, and lm3 should be trained on train3.

Which variables have a positive sign for the coefficient in at least one of lm1, lm2, and lm3 and a negative sign for the coefficient in at least one of lm1, lm2, and lm3?

- [x] age ✔
- [ ] alzheimers
- [ ] arthritis
- [ ] cancer
- [ ] copd
- [ ] depression
- [ ] diabetes
- [ ] heart.failure
- [ ] ihd
- [ ] kidney
- [ ] osteoporosis
- [ ] reimbursement2008

EXPLANATION

We can build the models with:

lm1 = lm(reimbursement2009~., data=train1)

lm2 = lm(reimbursement2009~., data=train2)

lm3 = lm(reimbursement2009~., data=train3)

From summary(lm1), summary(lm2), and summary(lm3), or e differs in sign between the models (it is positive in lm3 and negative in lm1 and lm2).

Hide Answer *You have used 1 of 3 submissions*

PROBLEM 4.3 - CLUSTER-SPECIFIC PREDICTIONS (1 point possible)

Using lm1, make test-set predictions called pred.test1 on data frame test1. Using lm2, make test-set predictions called pred.test2 on data frame test2. Using lm3, make test-set predictions called pred.test3 on data frame test3.

Which vector of test-set predictions has the smallest average predicted reimbursement amount?

- ( ) pred.test1 ✖
- ( ) pred.test2 ✔
- ( ) pred.test3

EXPLANATION

The predictions can be obtained with:

pred.test1 = predict(lm1, newdata=test1)

pred.test2 = predict(lm2, newdata=test2)

pred.test3 = predict(lm3, newdata=test3)

From mean(pred.test1), mean(pred.test2), and mean(pred.test3), we read that pred.test2 has the smallest average test-set predictions. This was expected, since train2 had the smallest average value of the dependent variable.

Hide Answer *You have used 1 of 1 submissions*

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

Obtain the test-set RMSE for each cluster. Which cluster has the largest test-set RMSE?

- Cluster 1
- Cluster 2 ✔
- Cluster 3

EXPLANATION

We can compute cluster-specific RMSE with:

sqrt(mean((pred.test1 - test1$reimbursement2009)^2))

sqrt(mean((pred.test2 - test2$reimbursement2009)^2))

sqrt(mean((pred.test3 - test3$reimbursement2009)^2))

Cluster 2 has the largest value.

Hide Answer *You have used 1 of 1 submissions*

PROBLEM 4.5 - CLUSTER-SPECIFIC PREDICTIONS (1/1 point)

To compute the overall test-set RMSE 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:

all.predictions = c(pred.test1, pred.test2, pred.test3)

all.outcomes = c(test1$reimbursement2009, test2$reimbursement2009, test3$reimbursement2009)

What is the test-set RMSE of the cluster-then-predict approach?

1.811335

1.811335

Answer: 1.811334

EXPLANATION

After combining the predictions and outcomes with the provided code, we compute the test-set RMSE with:

sqrt(mean((all.predictions - all.outcomes)^2))

We see a modest improvement over the original linear regression model, which is typical in situations where the observations do not cluster strongly into different "types" of observations. However, it is often a good idea to try the cluster-then-predict approach on datasets with a large number of observations to see if you can improve the accuracy of your 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

EdX is a non-profit created by founding partners Harvard and MIT whose mission is to bring the best of higher education to students of all ages anywhere in the world, wherever there is Internet access. EdX's free online MOOCs are interactive and subjects include computer science, public health, and artificial intelligence.