CSE 635 - HW4

Miles Taylor

6/17/2022

```
knitr::opts_chunk$set(echo = TRUE)
library(tidyverse)
library(reticulate)
library(kableExtra)
library(broom)
```

Reading a Comma-Separated Values (.csv) Data

```
df <- read.csv("TitanicPassengers1.csv", skipNul= TRUE)
```

Converting to survived column as a numeric value for a Linear Regression Model

```
import pandas as pd
import numpy as np
nonulls = r.df.dropna().copy()
numeric_y = np.where(nonulls.Survived == 'Yes', 1, 0)

def transform_series_to_frame(series, actual, frame_name):
  make_frame = pd.DataFrame(series, columns = ['predicted'])
  make_frame['actuals'] = actual
  make_frame['name'] = frame_name

  return make_frame

def regression_autopsy(predicted, actual, sequence_name):
  sst = np.sum((actual - actual.mean())**2 )
  ssr = np.sum((predicted - actual.mean()) **2)
  sse = np.sum((predicted - actual)**2)
  r_squared = ssr/sst
  generate_payload = pd.DataFrame({'SST': [sst],
                                   'SSR': [ssr],
                                   'SSE': [sse],
                                   'R Squared': [r_squared] })

  generate_payload['diff'] = generate_payload.SST - generate_payload.SSR
  generate_payload['Errors Orthogonal?'] = np.where(generate_payload.diff == generate_payload.SSE,
                                                    'Yes', 'No')
```

```
  generate_payload.drop('diff', axis = 1, inplace= True)
  generate_payload['Name'] = sequence_name
  freeze_frame = generate_payload[['Name',  'SST', 'SSR',
                                   'SSE',
                                  'R Squared']].copy()


  error = predicted - actual
  error_mean = error.mean()
  error_std = np.round(error.std(),2)
  errors_frame = pd.DataFrame({'Name': [sequence_name],
                               'Error Mean': [error_mean],
                               'Error Std' : [error_std] })

  return freeze_frame, errors_frame
```

Running a Linear Regression Model using the best subset

```
py$nonulls['survived_numeric'] <- py$numeric_y

model1a <- lm(data = py$nonulls, survived_numeric ~ Age)
model1a_predicted <- predict(model1a)
model1b <- lm(data = py$nonulls, survived_numeric ~ Age + Pclass)
model1b_predicted <- predict(model1b)
model1c <- lm(data = py$nonulls, survived_numeric ~ Age + Pclass + Sex)
model1c_predicted <- predict(model1c)
model1d <- lm(data = py$nonulls, survived_numeric ~ Age + Pclass + Sex + Fare)
model1d_predicted <- predict(model1d)
model1e <- lm(data = py$nonulls, survived_numeric ~ Age + Pclass + Sex + Fare + Embarked)
model1e_predicted <- predict(model1e)
model1f <- lm(data = py$nonulls, survived_numeric ~ Age + Pclass + Sex + Embarked)
model1f_predicted <- predict(model1f)
```

```
model1_series = [r.model1a_predicted,
                 r.model1b_predicted,
                 r.model1c_predicted,
                 r.model1d_predicted,
                 r.model1e_predicted,
                 r.model1f_predicted]

model_names = ['Age', 'Age + Pclass', 'Age + Pclass + Sex', 'Age + Pclass + Sex + Fare',
               'Age + Pclass + Sex + Fare + Embarked', 'Age + Pclass + Sex +  Embarked']
all_the_models = []
all_the_errors = []
iterator = 0

for model in model1_series:
  model_as_frame = transform_series_to_frame(model, nonulls.survived_numeric.values, model_names[iterat
  fit, error = regression_autopsy(model_as_frame.iloc[:,0], model_as_frame.iloc[:,1], model_as_frame.il
  all_the_models.append(fit)
  all_the_errors.append(error)
```

```
  iterator += 1
models_out = pd.concat(all_the_models)
errors_out = pd.concat(all_the_errors)
```

All of the subsets of model 1 are below –

```
py$models_out %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

Name	SST	SSR	SSE	R Squared
Age	172.2129	1.026922	171.1860	0.0059631
Age + Pclass	172.2129	31.070306	141.1426	0.1804180
Age + Pclass + Sex	172.2129	67.191482	105.0214	0.3901652
Age + Pclass + Sex + Fare	172.2129	67.197692	105.0152	0.3902013
Age + Pclass + Sex + Fare + Embarked	172.2129	67.830973	104.3819	0.3938786
Age + Pclass + Sex + Embarked	172.2129	67.828595	104.3843	0.3938648

LM Best Model - Summary Output

```
predicted_values_model1 <- predict(model1f)
predicted_floor1 <- ifelse(predicted_values_model1 < 0.5, 0, 1)

tidy(summary(model1e)) %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

term	estimate	std.error	statistic	p.value
(Intercept)	1.3638344	0.0810256	16.8321363	0.0000000
Age	-0.0053092	0.0010906	-4.8682440	0.0000014
Pclass	-0.1925195	0.0228483	-8.4259713	0.0000000
Sexmale	-0.4775204	0.0308584	-15.4745763	0.0000000
Fare	-0.0000427	0.0003366	-0.1269217	0.8990385
EmbarkedQ	-0.1190345	0.0825212	-1.4424728	0.1496119
EmbarkedS	-0.0773979	0.0395962	-1.9546813	0.0510144

Creating a confusion matrix and accuracy for lm

```
confusion_matrix1 <- table(py$nonulls$survived_numeric, predicted_floor1)
confusion_matrix1 %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

The accuracy and probability of misclassifcation for model 1 is below –

	0	1
0	360	64
1	82	208

```
accuracy_confusion_matrix1 <- sum(diag(confusion_matrix1))/ sum(confusion_matrix1)
pmc_1 <- 1 - accuracy_confusion_matrix1
```

```
report_metrics1 = pd.DataFrame.from_dict({'Accuracy': [r.accuracy_confusion_matrix1],
                                          'Missclassification %' : [r.pmc_1]})
```

```
py$report_metrics1 %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),latex_options = "hold_position")
```

Accuracy	Missclassification %
0.7955182	0.2044818

Converting to survived column as a factor for a General Linear Regression Model

```
factor_survived <- as.factor(py$nonulls$Survived)
```

Running a Generalized Linear Regression Model with logit using the best subset

```
model2a <- glm(factor_survived ~ py$nonulls$Age,family="binomial"("logit"))
model2a_predicted <- model2a$fitted.values
model2b <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass,family="binomial"("logit"))
model2b_predicted <- model2b$fitted.values
model2c <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass + py$nonulls$Sex ,family="binomial"
model2c_predicted <- model2c$fitted.values
model2d <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass + py$nonulls$Sex + py$nonulls$Fare
model2d_predicted <- model2d$fitted.values
model2e <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass + py$nonulls$Sex + py$nonulls$Fare
model2e_predicted <- model2e$fitted.values
model2f <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass + py$nonulls$Sex  + py$nonulls$Emba
model2f_predicted <- model2e$fitted.values
```

```
model2_series = [r.model2a_predicted,
                 r.model2b_predicted,
                 r.model2c_predicted,
                 r.model2d_predicted,
                 r.model2e_predicted,
                 r.model2f_predicted]

model_names = ['Age', 'Age + Pclass', 'Age + Pclass + Sex', 'Age + Pclass + Sex + Fare',
               'Age + Pclass + Sex + Fare + Embarked',  'Age + Pclass + Sex +  Embarked']
all_the_models2 = []
```

```
all_the_errors2 = []
iterator2 = 0

for model in model2_series:
  model_as_frame = transform_series_to_frame(model, nonulls.survived_numeric.values, model_names[iterat
  fit2, error2 = regression_autopsy(model_as_frame.iloc[:,0], model_as_frame.iloc[:,1], model_as_frame.
  all_the_models2.append(fit2)
  all_the_errors2.append(error2)
  iterator2 += 1
models_out2 = pd.concat(all_the_models2)
errors_out2 = pd.concat(all_the_errors2)
```

All of the subsets of model 2 are below –

```
py$models_out2 %>%  kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

Name	SST	SSR	SSE	R Squared
Age	172.2129	1.027489	171.1610	0.0059664
Age + Pclass	172.2129	31.421933	140.5180	0.1824598
Age + Pclass + Sex	172.2129	68.221338	102.0785	0.3961454
Age + Pclass + Sex + Fare	172.2129	68.234997	102.0866	0.3962247
Age + Pclass + Sex + Fare + Embarked	172.2129	68.997991	101.6803	0.4006552
Age + Pclass + Sex + Embarked	172.2129	68.997991	101.6803	0.4006552

Logit Best Model - Summary Output

```
tidy(model2f) %>% kable()  %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

term	estimate	std.error	statistic	p.value
(Intercept)	5.2774227	0.5234359	10.082272	0.0000000
py$nonulls$Age	-0.0360344	0.0076707	-4.697659	0.0000026
py$nonulls$Pclass	-1.2166683	0.1432021	-8.496165	0.0000000
py$nonulls$Sexmale	-2.5284893	0.2087443	-12.112857	0.0000000
py$nonulls$EmbarkedQ	-0.8101985	0.5684097	-1.425378	0.1540480
py$nonulls$EmbarkedS	-0.4731823	0.2613466	-1.810554	0.0702099

```
predicted_values_model2 <- predict(model2f)
predicted_floor2 <- ifelse(predicted_values_model2 < 0.5, 0, 1)
```

Creating a confusion matrix and accuracy for logit

```
confusion_matrix2 <- table(py$nonulls$survived_numeric, predicted_floor2)
confusion_matrix2 %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

	0	1
0	390	34
1	112	178

The accuracy and probability of misclassifcation for model 2 is below –

```
accuracy_confusion_matrix2 <- sum(diag(confusion_matrix2))/ sum(confusion_matrix2)
pmc_2 <- 1 - accuracy_confusion_matrix2
```

```
report_metrics2 = pd.DataFrame.from_dict({'Accuracy': [r.accuracy_confusion_matrix2],
                                          'Missclassification %' : [r.pmc_2]})
```

```
py$report_metrics2 %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),latex_options = "hold_position")
```

Accuracy	Missclassification %
0.7955182	0.2044818

Running a Generalized Linear Regression Model with probit using the best subset

```
model3a <- glm(factor_survived ~ py$nonulls$Age,family="binomial"("probit"))
model3a_predicted <- model3a$fitted.values
model3b <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass,family="binomial"("probit"))
model3b_predicted <- model3b$fitted.values
model3c <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass + py$nonulls$Sex ,family="binomial"
model3c_predicted <- model3c$fitted.values
model3d <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass + py$nonulls$Sex + py$nonulls$Fare,
model3d_predicted <- model3d$fitted.values
model3e <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass + py$nonulls$Sex
             + py$nonulls$Fare + py$nonulls$Embarked,family="binomial"("probit"))
model3e_predicted <- model3e$fitted.values
model3f <- glm(factor_survived ~ py$nonulls$Age + py$nonulls$Pclass + py$nonulls$Sex
             +  py$nonulls$Embarked,family="binomial"("probit"))
model3f_predicted <- model3e$fitted.values
```

```
model3_series = [r.model3a_predicted,
                 r.model3b_predicted,
                 r.model3c_predicted,
                 r.model3d_predicted,
                 r.model3e_predicted,
                 r.model3f_predicted]
```

```
model_names = ['Age', 'Age + Pclass', 'Age + Pclass + Sex', 'Age + Pclass + Sex + Fare',
               'Age + Pclass + Sex + Fare + Embarked', 'Age + Pclass + Sex + Embarked']
all_the_models3 = []
all_the_errors3 = []
iterator3 = 0

for model in model3_series:
  model_as_frame = transform_series_to_frame(model, nonulls.survived_numeric.values, model_names[iterat
  fit, error = regression_autopsy(model_as_frame.iloc[:,0], model_as_frame.iloc[:,1], model_as_frame.il
  all_the_models3.append(fit)
  all_the_errors3.append(error)
  iterator3 += 1
models_out3 = pd.concat(all_the_models3)
errors_out3 = pd.concat(all_the_errors3)
```

All of the subsets of model 3 are below –

```
py$models_out3 %>%  kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

Name	SST	SSR	SSE	R Squared
Age	172.2129	1.019096	171.1666	0.0059177
Age + Pclass	172.2129	31.100556	140.5742	0.1805937
Age + Pclass + Sex	172.2129	67.249081	102.6777	0.3904997
Age + Pclass + Sex + Fare	172.2129	67.253370	102.6805	0.3905246
Age + Pclass + Sex + Fare + Embarked	172.2129	68.165222	102.1933	0.3958195
Age + Pclass + Sex + Embarked	172.2129	68.165222	102.1933	0.3958195

Probit Best Model - Summary Output

```
tidy(model3f) %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

term	estimate	std.error	statistic	p.value
(Intercept)	3.0094743	0.2846870	10.571169	0.0000000
py$nonulls$Age	-0.0197652	0.0043460	-4.547866	0.0000054
py$nonulls$Pclass	-0.6787842	0.0792268	-8.567611	0.0000000
py$nonulls$Sexmale	-1.4888001	0.1166359	-12.764515	0.0000000
py$nonulls$EmbarkedQ	-0.4944161	0.3279995	-1.507368	0.1317163
py$nonulls$EmbarkedS	-0.2908313	0.1499303	-1.939776	0.0524069

```
predicted_values_model3 <- predict(model3f)
predicted_floor3 <- ifelse(predicted_values_model3 < 0.5, 0, 1)
```

Creating a confusion matrix and accuracy for probit

```
confusion_matrix3 <- table(py$nonulls$survived_numeric, predicted_floor3)
confusion_matrix3 %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),
                latex_options = "hold_position")
```

	0	1
0	404	20
1	131	159

```
accuracy_confusion_matrix3 <- sum(diag(confusion_matrix3))/ sum(confusion_matrix3)
pmc_3 <- 1 - accuracy_confusion_matrix3
```

```
report_metrics3 = pd.DataFrame.from_dict({'Accuracy': [r.accuracy_confusion_matrix3],
                                          'Missclassification %' : [r.pmc_3]})
```

```
py$report_metrics3 %>% kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover"),latex_options = "hold_position")
```

Accuracy	Missclassification %
0.7885154	0.2114846

Finding the best subset

Please see the tables above for comparisons of the subset with SSE and R-squared for each model.

The best model specification is Age, Pclass, Sex, and Embarked as independent variables for two reasons. The first reason is through an analysis of the SSE for each model. Looking at the tabular outputs from each model, Age, Pclass, Sex, and Embarked and Age, Pclass, Sex, Fare, and Embarked have the lowest SSEs and narrows down the candidates to two possible models. The winner can be determined by looking at the p-values from each regression model. In this case, Fare was found to be non significant at the 5% level so it was removed from the model. It is important to look at the p-values for verification because the coefficient of determination (R^2) is not adjusted, so the SSR will always increase when new variables are added because there is more variation in the system.