DAEN 500- DL2 – Data Analytics Fundamentals

Fall 2020 Final Examination Exercise Package

10/7 – 10/13/2020

Final Submission Deadline: NLT 11:59PM (EST). Tuesday, 13 Oct 2020

*Failure to submit will result in DAEN COURSE FAIL/NoCredit*

Name:\_\_Esteban A. Echandi Maroto\_\_\_\_\_ GMU G#\_\_\_\_ G01253324 \_\_\_\_\_\_\_\_\_\_

Student Signature (Honor Certification): \_Esteban Echandi\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

This exam is **OPEN BOOK/OPEN NOTES**. You may consult any of the course texts, and the various reference materials recommended in the syllabus. ***The exam of course IS NOT “Open Web”,*** especially in that you may NOT utilize expert “help” sites such as Stack Overflow, or other programming help or collaboration sites.

HONOR CODE CERTIFICATION

**Your signature above declares that you have followed the conditions of this exam, and that the work is yours alone**. **Specifically:**

This must be your own work, authored and completed by you. As stated earlier, this is an “open source exam” – allowing books, notes or courseware, as well as *general* expert advice gained PRIOR to exam. YOU MAY NOT, HOWEVER, SEED OR USE ANY ADVICE ON HOW TO SOLVE THE QUESTION OR ANY CODE WRITTEN BY ANY OTHER INDIVIDUAL. *Any violation will result in an immediate failure in the exam and for the course, as well as referral to the GMU Honor Committee for determination of any other appropriate disciplinary consequences.*

*NOTE: Your* ***submission*** *of any responses, files, programs, etc. in response to the DAEN500 final exam instructions, will also be your personal certification of your full compliance with the spirit and letter of the* ***GMU Honor Code*** *standards for take home and/or in-class exams.*

Additionally, you are restricted from discussing the substance of the questions on this exam with any other individual, until after you have submitted your final response for grading. The completed exam -- with your answers embedded in this docx document (add extra pages as necessary) should be submitted following instructions contained in the Final Exam Instructions BB site. If you have any trouble submitting and have extra parts of the answers you have trouble appending to this document, you may simply submit additional pages separately (***the exam submission site is set for multiple submissions, just in case***). Make certain all are submitted PRIOR TO THE DEADLINE!

 FINAL EXAM PROBLEMS

COMPLETE ALL & INSERT ANSWERS BELOW QUESTIONS

# Problem 1: Python Programming Problem (15 Points Total)

* **Design and implement a Python program that is based on the following requirements: a) program will *find* *all numbers -- within a specified range -- which are divisible by 7 but are not a multiple of 5*; and b) demonstrate the program works by running the program for the range: *numbers between 2000 and 3200*.**
* **INSERT (cut&paste) your Python code in space below and *then insert a screen shot in space below, showing your successful run and output.***

NOTE of alternative for help: To help test your code, you also may use a Python “programming window” found in the. **Zybooks Section 35 Additional Material**.

#Enter range input

ent\_range1 = int(input("Enter first # in the range: "))

ent\_range2 = int(input("Enter last # in the range: "))

#Answer line

print('Number(s) divisible by 7 and not multiple of 5 are: ')

for x in range(ent\_range1, ent\_range2 + 1, 1):

if x % 7 **==** 0 :

if x % 5 **!=** 0:

print(x)

else:

print(end='')

Enter first # in the range: 2000

Enter last # in the range: 3200

Number(s) divisible by 7 and not multiple of 5 are:

2002

2009

2016

2023

2037

2044

2051

2058

2072

2079

2086

2093

2107

2114

2121

2128

2142

2149

2156

2163

2177

2184

2191

2198

2212

2219

2226

2233

2247

2254

2261

2268

2282

2289

2296

2303

2317

2324

2331

2338

2352

2359

2366

2373

2387

2394

2401

2408

2422

2429

2436

2443

2457

2464

2471

2478

2492

2499

2506

2513

2527

2534

2541

2548

2562

2569

2576

2583

2597

2604

2611

2618

2632

2639

2646

2653

2667

2674

2681

2688

2702

2709

2716

2723

2737

2744

2751

2758

2772

2779

2786

2793

2807

2814

2821

2828

2842

2849

2856

2863

2877

2884

2891

2898

2912

2919

2926

2933

2947

2954

2961

2968

2982

2989

2996

3003

3017

3024

3031

3038

3052

3059

3066

3073

3087

3094

3101

3108

3122

3129

3136

3143

3157

3164

3171

3178

3192

3199

​

​

# Problem 2: Python Programming Problem

# (15 Points Total)

* **Design and implement a Python program that is based on the following requirements:**

**a) define a class which has *at least two* methods**

* + **Method 1 – getString: to get a string from console input; and,**
  + **Method 2 - printString: to print the string in upper case.**

**b) *demonstrate code works using three different test input strings***

* ***INSERT* *code below* and *INSERT* a screen shot of the program and successfully run output that *includes test input for input strings (test strings must include (a) all upper case, (b) all lower case, and (c) mix of upper and lower case).***

class get\_print\_string:

def \_\_init\_\_(self):

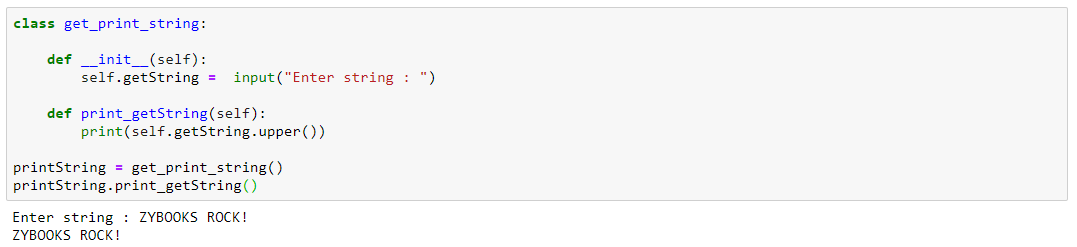
self.getString **=** input("Enter string : ")

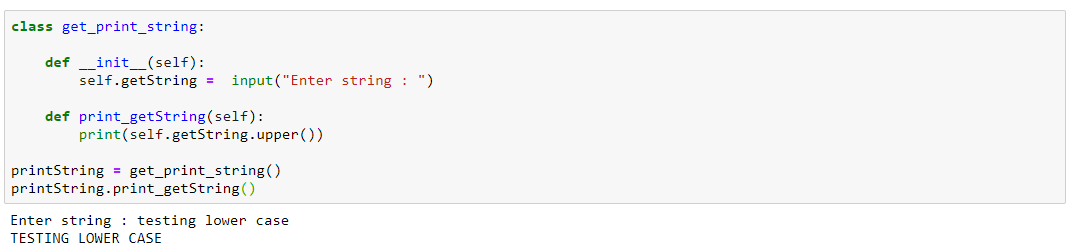
def print\_getString(self):

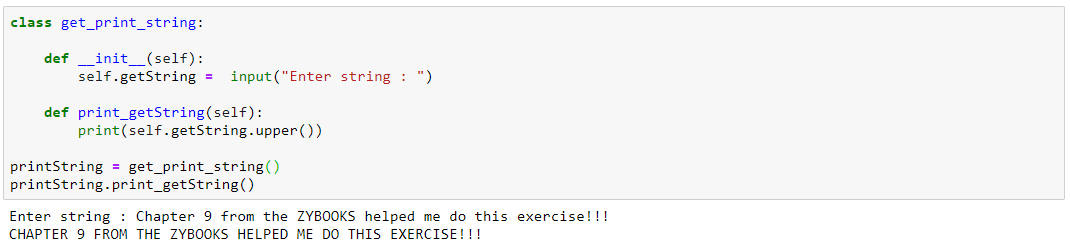
print(self.getString.upper())

printString **=** get\_print\_string()

printString.print\_getString()









# Problem 3: R Programming Problem

# (20 Points Total)

* **Perform the following problems using R:**
  + Create a vector of courses (e.g., MATH 101) you have taken previously. Make sure you have at least 8 courses. Name the vector myCourses
  + Get the length of the vector myCourses
  + Get the first two courses from myCourses
  + Get the 3rd and 4th courses from myCourses
  + Sort myCourses using a method
  + Sort myCourse in the reverse direction
* *INSERT* *code below* and *INSERT* a screen shot of the program and successfully run output.

install.packages("stringr")

library(stringr)

#1

myCourses <- c("DAEN 500","AIT 580","STAT 515","OR 531","CS 504","DAEN 690","SYST 568","AIT 664")

#2

length(myCourses)

#3

myCourses[c(1,2)]

#4

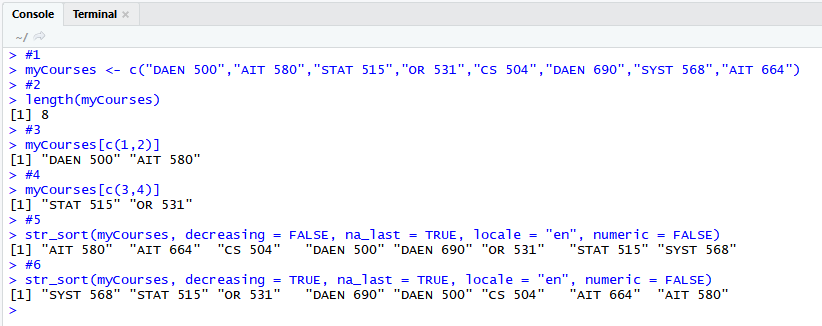
myCourses[c(3,4)]

#5

str\_sort(myCourses, decreasing = FALSE, na\_last = TRUE, locale = "en", numeric = FALSE)

#6

str\_sort(myCourses, decreasing = TRUE, na\_last = TRUE, locale = "en", numeric = FALSE)





# Problem 4: Principal Component Analysis

# (25 points)

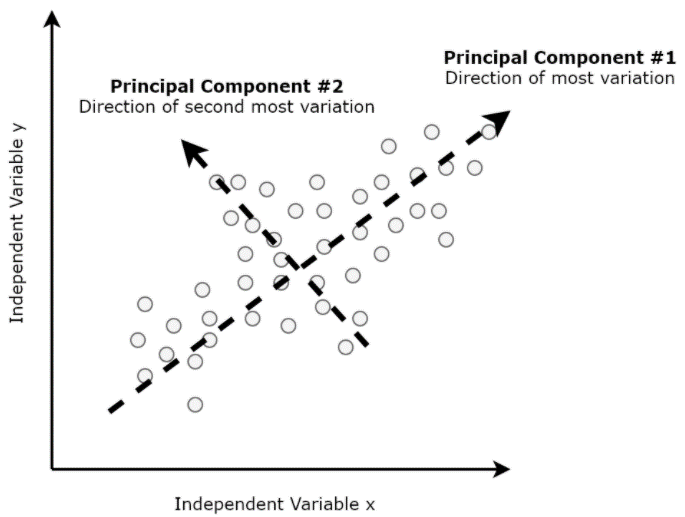
**Provide a description of the following:**

1. What is a component – Provide a description (5 points)
2. Principal Component Analysis – Provide a description.(5 points)
3. **Provide and explain an specific example of a Principal Component Analysis (15 points)**

A component is a reduction of different variables into a new one. Its represented by the first component that will be determined by the direction where projections have the longest variances and the second component which will maximize variance in all directions orthogonal to the first component.

For example, enclosed an image to illustrate.

**Graph 1. Principal component visualization for 1st and 2nd components.**

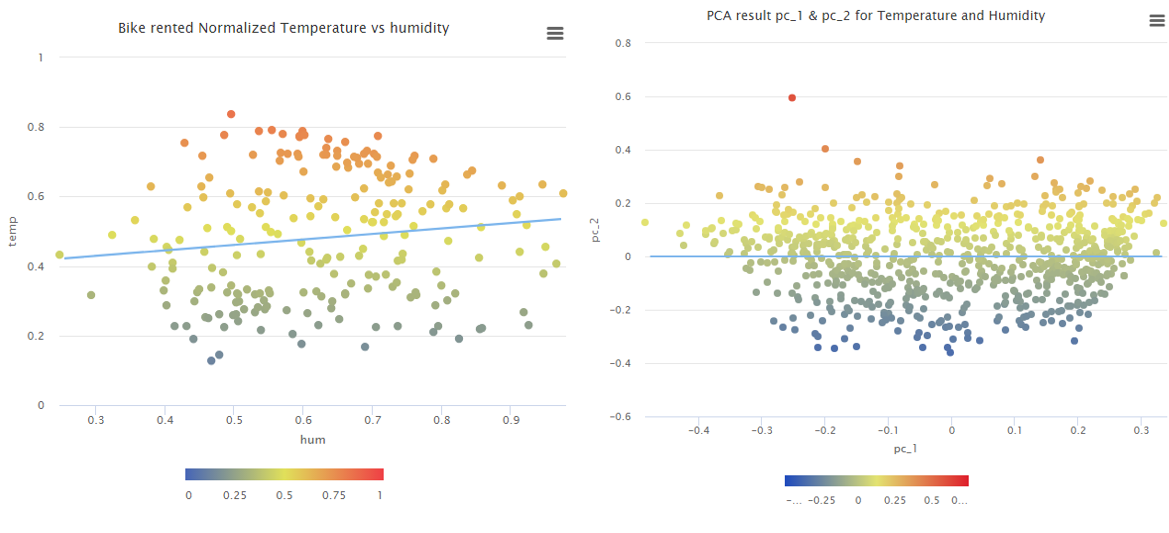


Source: Principal Component Analysis — Unsupervised Learning Model Originally published by Packt\_Pub on January 31st 2019 [link](https://hackernoon.com/principal-component-analysis-unsupervised-learning-model-8f18c7683262)

Principal Component Analysis is a mathematical method that helps use decrease the number of dimensions that especially large data sets could have. So basically, is a reduction of dimensionality to make analysis more agile and easy to interpret specially when working with several variables. They will usually point toward the direction of where the data has more variance of where it has more spread.

For example, by using RapidMiner we can do a PCA to a set of 2 variables used form the Bike rent data. For this specific example we used temperature and humidity since they do not have a clear correlation in our data, and they will be considered as the components. Reduction on dimensionality can be seen on graph b. where all the points have departed from the x-axis.

**Graphs 2. a. On the left data with no PCA applied for Temperature and humidity data. b. On the right those 2 same variables with PCA applied.**



# Problem 5: Multiple vs. Logistic

# (30 points)

# Describe: What is difference between Multiple Regression and Logistic Regression? What circumstances might determine which to use? (10 points)

# Demonstrate: Using any data, and any tool set you’ve learned about, show differences (20 points)

# SUGGESTION: may be solved using RapidMiner, or other toolsets, BOTH TO ANALYZE AND TO VISUALIZE REGRESSION DIFFERENCES..

Step 1: Perform a quick search of the [UCIS public data archive](https://archive.ics.uci.edu/), a well-curated site which you already have seen as part of your introductory RapidMiner training.

Step 2: Pick a dataset you find interesting, input dataset into regression tools you’ve chosen.

Step 3: Run the dataset (*may be a significant subset, if the dataset is very large*) first. a Multiple Regression and then a Logistic Regression, .and use visualizations to demonstrate the conceptual answers you provided for 5.(a).

5. a.1. Describe - Multiple Regression vs Logistic Regression.

Multiple and Logistic regression are two different concepts and have clearly some important differences that are key to understand which model is best for analysis. One of the first important differences that we can mention is the type of variable that they use. For instance, multiple regression uses a continuous variable for its dependent variable, meaning that it is a variable that can change to any value for its independent variable. While in the other hand for Logistic regression, the dependent variable is based on a discrete value that is considered binary between (0,1).

Another important aspect to mention that distinguishes them is how this regressions are associated with different types of distributions, for example linear regression models are more associated with a normal distribution (bell curve) for its dependent variable and logistic regression to a binomial distribution for its dependent variable since it’s based on a Boolean value outcome.

So, any circumstance where you have a dependent value that is within a true/false, yes/no outcome if will be identified as a logistic or binomial regression.

Some additional differences to add is also how they adjust to a linear line or curve models. The key of multiple or linear regression is to associate data points with a common line within them while the logistic regression since values oscillate between (0,1) the line will not adjust to the probability so a curve S share line will be utilized to show the probabilities.

5. b.1. Demonstrate - Multiple Regression – Bikeshare app data.

Dataset: <https://archive.ics.uci.edu/ml/datasets/bike+sharing+dataset>

Summary:

Bike share data set which measures the number of bikes rented by day with different indicators like humidity, windspeed and temperature.

Variables:

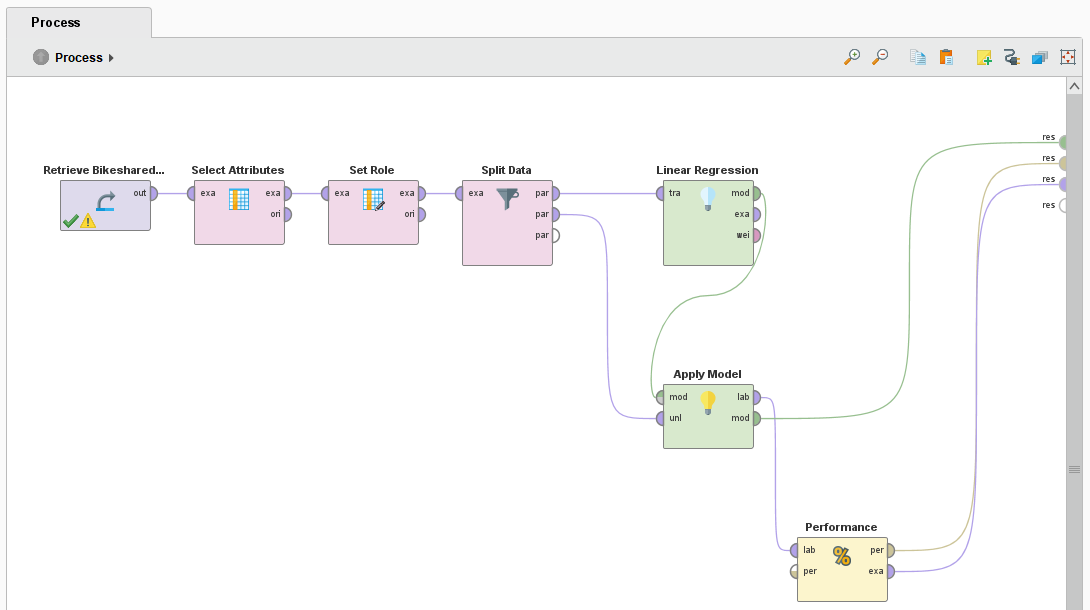
* temp: Normalized temperature in Celsius. The values are divided to 41 (max).
* hum: Normalized humidity. The values are divided to 100 (max).
* windspeed: Normalized wind speed. The values are divided to 67 (max).
* cnt: count of total rental bikes including both casual and registered.

Variable identification and Model selection

Our dependent variable will be cnt since what will be evaluated impacts the independent variables on the number of bikes rented. Since from the raw data that our independent variables are continuous, we determine that the multiple regression model is our best fit, since we do not have any Boolean data to use with our data. On the other hand, if the raw data contained Boolean information like sex type (M/F) or holiday (yes/no) and we wanted to determine a probability of how likely is for a Male or Female to rent a bike we could use a Logistic regression model.

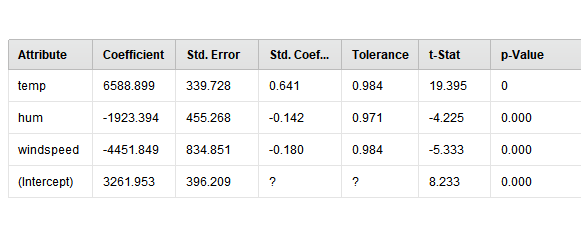
In conclusion the model selected for this data was the regression model since the model is going to evaluate how windspeed, humidity and temperature impact the number of rented bikes.

Fig. 1. Multiple regression RapidMiner design process



Data was reduced by selecting only the variables mentioned on the analysis that will be cnt, temp, hum, and windspeed. Cnt variable was selected as the dependent variable.

Table. 1. Multiple regression attribute detail.



Something important to mention from this table is that the p-Values are under 0.05 which give the model confidence. The equation of the line is determined by the coefficient values where 3261 is the Y-intercept and 1 independent variable shows a positive slope while the others show a negative slope.

Another important trait about the multiple regression is that it provides a line equation in which Y will be determined by cnt or the number of bikes rented based on the independent variable values show in the equation.

Fig. 2. Multiple regression - Equation

Fig. 3. Multiple regression dependent variable data represented as a bell curve

# 

# Since it’s a multiple regression the number of bikes rented can be determined by a bell curve or a normal distribution. On a logistic regression this is not the case since we will probably see data either TRUE/FALSE or 0/1 format like seen on Fig. 4.

# Fig. 4. Logistic regression dependent variable represented by a Boolean

# 

# Graph 1. Linear regression from Multiple regression analysis Bikes rented vs Temperature.

# 

5.b.2 Logistic Regression – Absenteeism at work data.

Dataset: <https://archive.ics.uci.edu/ml/datasets/Absenteeism+at+work>

Summary:

Absenteeism at work data measures from a courier company in Brazil from July 2007 to July 2010.

Variables:

Social drinker: identified by yes or no. (yes=1; no=0).  
Social smoker: identified by yes or no. (yes=1; no=0).  
Absenteeism hrs.: amount of time absent at work.

Distance from residence to work measured in km.

Variable identification and Model selection

# The key to select a variable when using Logistic regression is to determine if you want to measure 2 possible outcomes that might happen. Either it could be whether an event happens, or it does not, for example with our data the social drinker variable. There are only 2 possible answers for that question, either you are social drinker, or you are not. By having this knowledge, we proceeded to select as our label in RapidMiner “Social Drinker” variable, making sure that it was set up as a Boolean for the Logistic Regression to work.

Fig. 5. Logistic regression RapidMiner design process

# 

# Table. 2. Logistic regression confidence and prediction data.

# 

# When the model was finalized a summary of our data the report will show the following table were we can see the 2 yellow columns that will indicate the probability of the event happening which will be identified by confidence (1) or its inverse that will be confidence (0).

# Table. 3. Logistic regression overall accuracy probability.

# 

# On this last table we can see how accurate the model is on predicting a social drinker by 89.77% on the two inputs given by the data either been a social drinker or not.

# And finally, on the following graph above we can see the two Booleans on the y-axis against the identified whether you are a social drinker or not compared with distance from residence to work that will be a quantitative variable. Logistic regression is characterized for following a “S” curve type of patter, like seen below.

# Graph 2. Logistic regression prediction (Social drinker) vs Distance from Residence to Work

# 