|  |
| --- |
| # File: word\_counter.py |
|  | # Name: Doug Marcum |
|  | # Date: 10/16/2019 |
|  | # Course: DSC 510 |
|  | # Assignment Number: 8.1 |
|  | # Description: Program opens a text file (i.e. gettysburg.txt), calculates the total words and |
|  | # outputs the number of occurrences of each word in the file |
|  |  |
|  |  |
|  | def main(): |
|  | """Main function opens the file, creates a dictionary, calls to the process\_line function |
|  | and the pretty\_prints function.""" |
|  | count = {} |
|  | gba\_file = open('gettysburg.txt', 'r') |
|  | for line in gba\_file: |
|  | process\_line(line, count) |
|  | print('Opening File....gettysburg.txt\nProcessing...Complete!') |
|  | pretty\_print(count) |
|  |  |
|  |  |
|  | def process\_line(line, count): |
|  | """This function imports the string library, formats the text file by stripping away punctuation, whitespaces, |
|  | makes all of the words the same case, and splits out the words. Makes a call to the add\_word function.""" |
|  | import string |
|  | line = line.rstrip().lower().translate(line.maketrans('', '', string.punctuation)) |
|  | words = line.split() |
|  | add\_word(words, count) |
|  |  |
|  |  |
|  | def add\_word(words, count): |
|  | """This function adds each word to the dictionary.""" |
|  | for word in words: |
|  | if word not in count: |
|  | count[word] = 1 |
|  | else: |
|  | count[word] += 1 |
|  |  |
|  |  |
|  | def pretty\_print(count): |
|  | """This function formats the printed results into a structured, easily readable format.""" |
|  | print('Length of the dictionary:', len(count), 'words') |
|  | print(f"{'Word':20}{'Count'}") |
|  | print(f"{'':-<26}") |
|  | for key, value in sorted(count.items(), key=lambda item: item[1], reverse=True): |
|  | print('{:20}{}'.format(key, value)) |
|  |  |
|  |  |
|  | main() |

<https://github.com/MarcumDoug/Python_Code_Projects/blob/master/word_counter.py>

print(f"{'word':18}{'count'}")

**Week 9 - What is multi-collinearity?**

First, I will define what collinearity which is when two or more predictor variables might be correlated with each other. While a Multi-collinearity is when the collinearity exists between three or more variables even if no pair of variables has a particularly high correlation. Meaning there is a redundance between the predictor variables.

Multicollinearity problems consist of different variables that have a similar predictive relationship with the outcome. This can be assessed for each predictor by computing the VIF value. VIF is known as the variance inflation factor which measures how much the variance of a regression coefficient is inflated due to multicollinearity in the model. As the smallest possible value of VIF is one which is the absence of multicollinearity. While a VIF value that exceeds 5 or 10 indicates a problematic amount of collinearity. In conclusion When faced to multicollinearity, the concerned variables should be removed, since the presence of multicollinearity implies that the information that this variable provides about the response is redundant in the presence of the other variables.

Reference: <http://www.sthda.com/english/articles/39-regression-model-diagnostics/160-multicollinearity-essentials-and-vif-in-r/#:~:text=There%20is%20an%20extreme%20situation,is%20redundancy%20between%20predictor%20variables>.

**Week 9 – startswith string method**

* Startswith() – This string method will return a true if the string starts with the specified prefix if it does not then it will return false. This search can be limited by the user specifying optional arguments such as start and end.
* Syntax - string.startswith(prefix,start,end)
  + Table

    Description automatically generated
* An example we can conduct is to check if the string starts with ‘Bob’
  + A picture containing text

    Description automatically generated
* Limit startswith() – When limiting a search to the substring, specific start and end parameters are needed.
  + We will check and see if the substring 9th to 18th character starts with CEO
  + Graphical user interface, text

    Description automatically generated with medium confidence
* Providing multiple prefixes to the method in the form of a tuple.
  + Graphical user interface, text

    Description automatically generated
  + As we see above if the string starts with any item of the tuple, the method returns True, if it did not it would return False

Reference: <https://www.learnbyexample.org/python-string-startswith-method/>

Thank you for the information on the absolute value for standardized residuals, as this makes sense as if it is greater than 3.29, we could raise some flags as manipulation of the data could be the reason for this outcome. As I researched further into standardized residuals, I found that A residual is the difference between an observed value and a predicted value in a regression model. As we see below the calculation for a residual:

**Residual = Observed value – Predicted value**

**While in R studios we can easily** standardized residuals for a fit model by following these easy steps:

**Create Data**

data <- data.frame(x=c(8, 12, 12, 13, 14, 16, 17, 22, 24, 26, 29, 30),

y=c(41, 42, 39, 37, 35, 39, 45, 46, 39, 49, 55, 57))

**Fit Model**

model <- lm(y ~ x, data=data)

**View Model Summary**

summary(model)

Text

Description automatically generated

**Calculate the Standardized Residuals**

We calculate this by using the built-in rstandard() function

**calculate the standardized residuals**

standard\_res <- rstandard(model)

**view the standardized residuals**

standard\_res

Table

Description automatically generated

Reference: <https://www.statology.org/standardized-residuals-in-r/>

The Adjusted predicted value is used to compare the goodness-of-fit for regression models that contain differing numbers of independent variables. As the adjusted predicted value adjusts for the number of terms in the model. Which tells us that the value increases only when the new term improves the model fit more than expected by chance alone. But when the term doesn’t improve the model fit by a sufficient amount the Adjusted predicted value will decrease. As seen in the example below we find that the adjusted predicted value will start to decrease meaning the term does not improve the model at 85.9.

Table

Description automatically generated with low confidence

Reference: <https://statisticsbyjim.com/regression/interpret-adjusted-r-squared-predicted-r-squared-regression/>

When testing the Homoscedasticity in R studios we can use the function ncvTest() which is a statistic that tests whether there is non-constant variance which is known as the Homoscedasticity that is the null states that there is constant variance. We can check this by using our ncvTest() function below:

ncvTest(Gobble.model.2)

Text

Description automatically generated

This tells us that the p value of 0.81 is greater than .05 meaning our data is homoscedastic.

Reference: <https://ademos.people.uic.edu/Chapter12.html#35_testing_the_homoscedasticity_assumption:_ncvtest()>

Thanks for sharing this neat walkthrough on splitting data in a regression model. As we can use split to give us the ability to divide the data from a data frame into groups. As the value that is returned from the split would be seen as a list of vectors containing the values for the groups. In previous assignments I have used split to manipulate my data frames to only have the variable I need to perform a specific calculation. This helps clean up data frames by making it easier for viewers to understand and get to the point on what the end user would want to see.

Reference: <https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/split>

As I researched further into R-squared I found that it is a goodness-of-fit measure for linear regression models. Which indicates the percentage of the variance in the dependent variable while we see the independent variables will explain the collectivity of the R-squared. This statistic measures the strength of the relationship between your model and the dependent variable on a convenient 0 – 100% scale.

Chart, scatter chart

Description automatically generated

As seen in the two plots above we see that the model on the left has an R-squared of 15% while on the right the R-Squared is 85%. This shows us that when a regression model accounts for more of the variance, the data points are closer to the regression line.

Reference: <https://statisticsbyjim.com/regression/interpret-r-squared-regression/>

These are great techniques to use for cross validation as I liked the use of the train test split which splits the data randomly in a range of 70:30-80:20 split. This seems like a great test, but I can also see how some bias can be seen if the data set was smaller. While another technique you can use is the Leave One Out Cross-Validation(LOOCV) which is a method that splits the dataset into 2 parts but it overcomes the drawbacks of the Validation set approach. As the advantages for this approach are that it is less bias model as the data points is used for training. The LOOCV approach can be seen as:

* Train the model on N-1 data points
* Testing the model against that one data points which was left in the previous step
* Calculate prediction error
* Repeat above 3 steps until the model is not trained and tested on all data points
* Generate overall prediction error by taking the average of prediction errors in every case

An example from R Studios can be seen below:

Text

Description automatically generated

With an output of:

Graphical user interface, text, application, email

Description automatically generated

Reference: <https://www.geeksforgeeks.org/cross-validation-in-r-programming/>

library(Rcmdr)

* united\_states\_covid19\_cases\_by\_state
  + <https://covid.cdc.gov/covid-data-tracker/#cases_totalcases>
* united\_states\_covid19\_testing\_by\_state
  + <https://covid.cdc.gov/covid-data-tracker/#cases_totaltests>
* united\_states\_covid19\_deaths\_by\_state
  + <https://covid.cdc.gov/covid-data-tracker/#cases_totaldeaths>

## Provisional\_COVID-19\_Death\_Counts\_by\_Sex\_\_Age\_\_and\_State data frame Covid-19 data.

Provisional\_COVID\_19\_Death\_Counts\_by\_Sex\_Age\_and\_State <- read\_csv("CDC Covid Data/Provisional\_COVID-19\_Death\_Counts\_by\_Sex\_\_Age\_\_and\_State.csv")

View(Provisional\_COVID\_19\_Death\_Counts\_by\_Sex\_Age\_and\_State)

## Covid-19 Deaths by race, sex, and age

AH\_Provisional\_COVID\_19\_Deaths\_by\_Educational\_Attainment\_Race\_Sex\_and\_Age <- read\_csv("CDC Covid Data/AH\_Provisional\_COVID-19\_Deaths\_by\_Educational\_Attainment\_\_Race\_\_Sex\_\_and\_Age.csv")

View(AH\_Provisional\_COVID\_19\_Deaths\_by\_Educational\_Attainment\_Race\_Sex\_and\_Age)

## Covid-19 Deaths by education and race

AH\_Provisional\_COVID\_19\_Deaths\_by\_Race\_and\_Educational\_Attainment <- read\_csv("CDC Covid Data/AH\_Provisional\_COVID-19\_Deaths\_by\_Race\_and\_Educational\_Attainment.csv")

View(AH\_Provisional\_COVID\_19\_Deaths\_by\_Race\_and\_Educational\_Attainment)

## Covid-19 Deaths by age in years

Provisional\_COVID\_19\_Deaths\_Counts\_by\_Age\_in\_Years <- read\_csv("CDC Covid Data/Provisional\_COVID-19\_Deaths\_Counts\_by\_Age\_in\_Years.csv")

View(Provisional\_COVID\_19\_Deaths\_Counts\_by\_Age\_in\_Years)

## Vaccine administered for Pfizer by state by week

## Vaccine administered for Moderna by state for the week of 05/10/2021

COVID\_19\_Vaccine\_Distribution\_Allocations\_by\_Jurisdiction\_Moderna <- read\_csv("CDC Covid Data/COVID-19\_Vaccine\_Distribution\_Allocations\_by\_Jurisdiction\_-\_Moderna.csv")

View(COVID\_19\_Vaccine\_Distribution\_Allocations\_by\_Jurisdiction\_Moderna)

## Vaccine administered for Johnson and Johnson by state for the week of 05/10/2021

COVID\_19\_Vaccine\_Distribution\_Allocations\_by\_Jurisdiction\_Janssen <- read\_csv("CDC Covid Data/COVID-19\_Vaccine\_Distribution\_Allocations\_by\_Jurisdiction\_-\_Janssen.csv")

View(COVID\_19\_Vaccine\_Distribution\_Allocations\_by\_Jurisdiction\_Janssen)