WRITEUP: For Data Analysis Nanodegree January 2018 – January 2019

TERM 1

1. SQL

* **Covered:**
  + Select statements
  + Case statements: CASE WHEN total > 100 THEN ‘H’

WHEN total > 50 THEN ‘M’

ELSE ‘L’ END AS score

* + Inner queries: Select x from (select .. from .. where) alias\_name
  + With t1 as (select..from), t2 as (select..from t1) select \* from t2
  + LEFT(col, x)
  + RIGHT(col,x)
  + LENGTH(x)
  + UPPER(x)
  + LOWER(x)
  + POSITION(‘,’ IN col) OR STRPOS(col,’,’)
  + LEFT(UPPER(col)),LENGTH(col)-4
  + CONCAT(col1, ’,’ , col2, ’,’ , col3) AS NAME
  + CAST (travel\_date AS date) as col1 or travel\_date::date AS col1
  + COALESCE (amount,0) OR COALESCE (name,’unknown’) as cleansed\_col FROM T WHERE amount/name IS NULL
  + WINDOW FUNCTIONS (calculations over a set of rows)

**PROJECT**

Explore weather trends using SQL. This involved using SQL to select data from a database, then downloading to a csv for analysis. Excel was then used for analysis.

* I calculated a 7 year moving average (using AVERAGE(B1:B8) and copying down) to smooth out the data and make it easier to observe the long term trends.
* I created a line chart from the moving average and drew conclusions from the chart.

Exploring Weather Trends Project 1.pdf documents the findings.

**REVIEW FEEDBACK/REFLECTION**

Weather trends project feedback.doc. The only improvement would have been to extract data in a single pass rather than 2 sql queries – however this hadn’t been covered, and the project specification asked for 2 queries.

1. PYTHON

* The power of Python is in using containers.
  + LIST: mutable ordered sequence e.g. months = [‘January’,’February’]
  + LIST FUNCTIONS: min,max, sorted(l), sorted(l, reverse=TRUE), append()
  + TUPLE: immutable, ordered sequence used to store related info.
  + SET: unordered, unique elements e.g. towns = [‘London’,‘Manchester’,’Oxford’]
  + SET FUNCTIONS: add(), pop(), IN, NOT IN, UNION, INTERSECTION.
  + DICTIONARY: unordered key/value pairs or any datatype e.g. my\_dict = {‘Monday’:1, ‘Tuesday’:2}
  + DICTIONARY FUNCTIONS: get() looks up values and returns None if not found, is/is not
  + LOOPS: for key, value in my\_dict.items(): print “key: {} value{}”.format(key,value)
  + LIST COMPREHENSIONS: quickly create a list e.g.

lastnames = [name[name.find(‘ ‘): ].lower() for name in names]

* + FUNCTIONS
  + LAMBDA FUNCTIONS: one off anonymous functions e.g.

mean = lambda num\_list:sum(num\_list)/len(num\_list)

* + HIGH ORDER FUNCTIONS:
    - MAP(): takes a function and iterator as params and applies the function to each element e.g. averages= map(lambda x:sum(x)/len(x), numbers)
    - FILTER(): takes params as above and returns an iterator with elements that are TRUE e.g.
      * + Is\_short=lambda name:len(name) <10
        + Short\_cities = list(filter(is\_short,cities))
  + Opening files: with open(file, ‘r’) as f: file\_data = f.read()
  + Numpy: mathematical functions on large arrays.
  + Pandas: powerful library for data analysis (like powerful excel)
  + Matplotlip: visualisations.

**Data Analysis Process**

Data Gathering

* Read csv file: df = pd.read\_csv(f)
* Df.shape() rows & cols
* Df.dytpes()
* Df.info() summary info
* Df.nunique()
* Df.describe() useful stats

Data Cleansing (numpy)/EDA(pandas)

* Df(duplicated())
* Sum(df.duplicated())
* Df.drop\_duplicates(inplace=True)
* Df[‘timestamp’] = pd.to\_datetime(df[‘timestamp’])
* %matplotlib inline
* Df.hist()
* Df.hist(figsize=(8,8));
* Df[‘age’].hist()
* Df[‘education’].value\_counts().plot(kind=’bar’)
* Df.plotting.scatter-matrix(df,figsize=(15,15))
* Df.plot(x=’age’,y=’attendance’, kind=’scatter’)
* Df[‘col’].plot(kind=’box’)
* Using masks to select data: df\_m=df[‘diagnosis’] ==’m’
* Merging datasets: df = df1.append(df2)
* Df.groupby(‘col’).mean()
* Df\_m = df.query(‘gender == “M”’)
* Df\_h = df.query(‘value <10’)
* Import seaborn library for better visualisations.

**PROJECT**

US Bikeshare: Using Python (and Atom) to explore data related to bike share systems for Chicago, New York City, and Washington. Importing data and answering questions by computing descriptive statistics. Also writing a script that takes in raw input and creates an interactive experience to present the statistics.

Bikeshare.py

Readme.txt

**REVIEW FEEDBACK/REFLECTION**

I deliberately completed this project without using Numpy and Pandas in order to see the difference later (and also because Numpy/Pandas comes later in the Nanodegree). However, this project would have been easier if I had used Pandas, and the feedback also stated this.

1. DATA ANALYSIS PROCESS/INVESTIGATE A DATASET

* Advanced SQL

**PROJECT**

Analyse a dataset: I chose the medical appointments in Brazil data downloaded from <https://d17h27t6h515a5.cloudfront.net/topher/2017/October/59dd2e9a_noshowappointments-kagglev2-may-2016/noshowappointments-kagglev2-may-2016.csv>.

The aim was to follow the data analysis process (Question/Wrangle &EDA/ Explore/ Draw conclusions/communicate) and use Python libraries NumPy, pandas, and Matplotlib

The work was completed in Jupyter Notebook containing the code and also Markdown cells to document questions and conclusions. The final ipynb file was saved as an html file: P+Jasper+Investigate+Dataset-No+Show+Appointments.html

**REVIEW FEEDBACK/REFLECTION**

I had very good feedback. This was a tough project though.

1. STATISTICS

Probability (to make predictions) & Statistics (analyse the past)

* Probability (independent events)
  + Law of Probability P(A) = 1 – P(not A)
  + Truth Tables set out probability of each outcome
  + Binomial Distribution: calculation to determine probability of a string of independent events e.g. coin flips/fraudulent transactions. Determines the number of successes. Used in ML.
* Conditional Probability
  + Probability of A given B: P(A | B) e.g. P(pos test | disease) = P(pos ∩ disease)

P(disease)

* Bayes Rule: allows you to look at a hidden variable to give you better prediction

Prior probability: P(cancer) = 0.1 P(not cancer) = 0.9

P(pos | cancer) P(pos | not cancer)

P(neg | cancer) P(neg | not cancer)

Joint: P(pos | cancer) = P(cancer).P(pos|cancer) P(pos | not cancer)

P(neg | cancer) = P(cancer).P(neg|cancer) P(neg | not cancer)

Normalise: P(pos | cancer) / norm

P(pos |not cancer) / norm

P(neg | cancer) / norm

P(neg | cancer) / norm

Posterior Probability: P(cancer | pos) = P(cancer | pos)/norm

P(not cancer | pos) = P(not cancer | pos)/norm

P(cancer | neg)

P(not cancer | neg)

* Simulate random events e.g. coin flips in Python
  + np.random.randint(2) gives 0 or 1
  + np.random.randint(2, size=100) array of 100 flips
  + np.random.choice([0,1], size=100, p=0.8, 0.2) allows loaded outcomes
  + np.random.randint (2, size=(int(le6),2)) generates 1m flips of a fair coin

**Descriptive statistics**: describing data we have using measures of centre, spread, shape & outliers.

**Inferential**: Drawing conclusions for population based on sample i.e. drawing a conclusion of a parameter from a statistic is inference.

**Statistic:** summary from the sample we have.

**Parameter**: summary from the population (we need to estimate this).

**Sampling distribution**: distribution of a statistic i.e. we can plot the means of lots of samples. It will be centred on the original parameter.

**Law of large numbers**: the larger the sample the closer the statistic to the parameter.

**Central Limit Theorem**: With a large enough sample, the distribution will be normally distributed.

**Bootstrapping** (sampling with replacement): Used to create sampling distribution when the sample is not big enough.

**Confidence Interval**: Provides a range of values possible for a parameter.

**Confidence level**: The % that can be expected to include the true population parameter. 99% or 95%. Confidence intervals allow us to estimate a population mean, std, diff btw 2 population means, aggregate values, other numeric summary. They do not allow us to say something about an individual in a population. ML does.

**Hypothesis Testing**:

* H0 (null): what we assume to be true, e.g. H0: a-b = 0 there is no difference between samples
* H1 (alt): e.g. H1: a-b!= 0 there is a difference.
* Type 1 error: worst scenario, false positive, when null is true but we chose alt.
* Type 2 error: Alt is true but we chose null.

**Correlation Coefficient**: statistical measure of the degree to which changes to the value of one variable predict changes to another.

* Strength (weak 0.3/mod 0.7/strong > 0.7)
* Direction (pos/neg).

**Machine Learning**:

* Supervised: Input data, predict a label. E.g. predict fraudulent transactions based on features. Regression is a technique.
* Unsupervised: Group data based on common characteristics.

**Multiple Linear Regression**: Predicts the change in one variable for each unit increase of another variable, assuming all other variables remain constant.

* Using python to see a relationship between 2 variables
* Import statsmodels.api as sm
* Set the intercept: df[‘intercept’] = 1
* Least squares function: Lm = OLS(df[‘price’], df[[‘intercept’,’area’]]
* Fit the model: results = lm.fit()
* Show summary: lm.summary()
* Plot it: df.plot.scatter(‘price’, ‘area’)
* Interpret results:use the intercept and area to predict
* Use P values to see if there is statistical evidence that area is related to price.

**Logistic Regression**: Predicts 1 or 2 possible outcomes (0 or 1).

* Python
  + Log\_m = Logit(df[‘admit’]
* Then test accuracy of the LGM to predict a label e.g. fraud

**Confusion Matrices**: show how the model has performed.

* True pos: True and correctly classified.
* True neg: true but wrongly classified.
* False pos: False but classified as true
* False neg: False and correctly classified.
* Recall: Out of the true values, what proportion were correctly identified.
* Precision: Of all values labelled as true, what proportion are actually true.

**A/B Test**: Used to test whether a change is statistically significant or due to chance, which can determine whether to go ahead with a change.

* Calculate the difference between 2 groups (i.e. the get the 2 means and see the difference - - observed difference).
* Bootstrap the sample to simulate the sampling distribution.
* Plot the distribution.
* Simulate the distribution under the null.
* Compute the P value (the observed difference) for the statistic (the sample)
* If the P value is less than 1% we can reject H0 as it is unlikely our statistic came from it.

**PROJECT**

Working to understand the results of an A/B test run by an e-commerce website with the objective of working through the notebook to help the company understand if they should implement the new page, keep the old page, or perhaps run the experiment longer to make their decision.

The conclusion was that although it seemed initially that there was a difference between the conversion rates of receiving the old or new web page, there was actually no statistical evidence to reject the null hypothesis. I recommended that company keeps their old page and does not implement the new page.

**REVIEW FEEDBACK/REFLECTION**

Very difficult project!

The review contains some extra explanations.

TERM 2

1. TERM 1 RECAP– PROGRAMMING & STATS

* Recap of Python fundamentals – this was no problem.
* Recap of statistics – this was more time consuming as it is a weaker area for me.

**Probability**

* Number successes/total possibilities

**Conditional probability**

* Bayes Rule e.g. if I choose A, what is the probability it came from bag B?
* Probability of A given B – full details of Bayes Rule formula in notebook.

**Descriptive Statistics – measures of spread**

* Mean: Count/total
* Median: Middle value of sorted list
* Mode: Most frequently occurring
* Std: Variance from the mean
* Q1: Median of all values below the mean
* Q3: Median of all values above the mean
* IQR: Q3-Q1

**T-Test – statistical test to compare 2 means**

* We use the T test to look for significant differences in data – it is easy to see that there are differences but you can’t easily see if these are due to outliers in the data without the t test.
* T test gives us a critical value: the P value.
* We want to know if the result is due to random chance or not.
* If the P value is <0.05, there is <5% chance that this is random, so >=95% confidence that this is significant.
* If the P value was 0.52, it means in 52% of cases, it could be due to chance which is not good enough!
* T Tests are either
* 1 tailed (= or ≠) or 2 tailed (>= or <=). We usually use 2 tailed.
* Paired or non paired (paired if same population at different times)

**T Test by hand (very time consuming)**

* Doing the calculations (e.g. means, size, stds)
* Set up hypotheses
  + H0 - µ1- µ2 = 0 (no difference btw sample means)
  + H1 - µ1- µ2 ≠0 (2 tailed test. 1 tailed would test < or > 0. Usually best to do 2 tailed)
* Calculate the T Stat: Gives us the P value e.g. if >0.05 we would not reject H0.

**T Test in Excel**

* For each sample, calculate AVERAGE(range), COUNT(range), STDEV(range)
* P Value = TTEST(range1, range2, #tails, type)
* Type is paired(1), equal variance (2), unequal variance (3)
* T Test in Excel

**T Test in Python**

From statsmodels.stats.weightstats import ttest\_ind

Ttest = ttest\_ind(array1, array2, alternative = ‘two sided’, usevar=’pooled’)

Or

From scipy.stats import ttest\_ind

From scipy.stats import ttest\_rel (if paired samples)

Ttest = ttest\_ind(1,2,equal\_var = True) or Ttest\_paired = ttest\_rel(1,2)

This will give us the t-stat, p value and variance.

**PROJECT**

The project concerned taking 2 samples of people who had taken the Stroop test, and looking to see if the difference in the samples was significant or due to chance.

It was carried out in Python.

E:\My Documents\Paula\DATA ANALYST NANO DEGREE\TERM 2\Stroop

* Test a Perceptual Phenomenon-Paula Thur.html

**REVIEW FEEDBACK/REFLECTION**

Ttest assumptions.hrml and Stats project feedback.doc give extra help on understanding what the ttest is about and what we can deduce from it.

1. EXPLORATORY DATA ANALYSIS – R STUDIO

* EDA is often the 1st stage of data analysis
* Quick understanding of data through visualisations and statistical models.
  + Variable distributions – histograms.
  + Relationship between variables – scatter plots.
  + Understand the source of data.
* Be curious and sceptical. Decide what insights/questions you want to examine. Look for oddities.
* R is used for statistical programming and visualisations, exploration. R Studio is easy to install – from CRAN!

LEARNING – a process for analysing data with R

1. Read in the file: setwd(‘…’)/getwd()/mydata = read.csv(‘file.csv’)
2. Check the class: class(dataset)
3. View dimensions dim(dataset)
4. Look at the columns in the dataset – names(dataset)
5. Dig deeper and look at structure - str(dataset)
6. Load dplyr – allows use of select, filter, arrange, rename, mutate, summarize)
   1. Note install.packages("dplyr”) at command line.
7. Get stats on relevant variables - summary(data)
8. View the data – head/tail(dataset)
9. Visualise the data
   1. Hist(data$var)
   2. Qplot – histograms/scatter plots/line plots/frequency polygons
   3. Ggplot
10. View plots side by side – library(GridExtra)/grid.arrange(p1,p2,ncol)
11. Alter/restructure data with tidyr – make long (gather) or wide (spread)
12. Look at correlations cor\_test(x,y)

Example scatter plot which focuses on 13-90 year olds and adds a mean line:

ggplot(aes(x=age,y=friend\_count),data = pf) +

xlim(13,90) +

geom\_point(alpha=0.05,position=position\_jitter(h=0),color=’orange’) +

coord\_trans(y=’sqrt’) +

geom\_line(stat=’summary’,fun\_y=mean)

**PROJECT**

Apply EDA techniques using R to explore a selected data set (White Wine) for distributions, outliers, and anomalies.

RMD file contains analysis, final plots, summary and reflection.

HTML file knitted from RMD file (knitr)

E:/My Documents/Paula/DATA ANALYST NANO DEGREE/TERM 2/EDA/EDA project/

* whitewineEDA2.html
* wineQualityWhites.csv
* whitewineEDA2.rmd
* EDA project review.docx

**REVIEW FEEDBACK/REFLECTION**

* Easy to produce analysis on a dataset with R Studio.
* Might not be so easy to do a lot of data wrangling – might need to be cleansed in Python.
* Visualisations can be produced quickly and easily.
* Can produce presentations easily by knitting to html.
* The RMD file can be structured with code comment blocks so the file is a stream of consciousness. Also good practice to give each code block a label.

1. DATA WRANGLING – PYTHON

This generally must be done before analysis, visualising and building models. The process:

* Gather data
  + Download files from the internet programmatically using the requests and os libraries; requests.get(url) and read files into a dataframe.
  + Use APIs to scrape twitter, Instagram, wikipedia etc. Each API will have its own Python libraries, e.g. wptools for Wikipedia, tweepy for twitter. Most data from APIs is in JSON format which can be read into JSON objects which are like Dictionaries.
  + Use the Beautiful Soup library if there is no machine readable API.
  + There are libraries to allow you to connect to a database to get data.
* Assess data (ensure you follow formulaic process to make this simpler)
  + Quality (missing values and inconsistency = dirty data)
  + Tidiness (each variable in a column, each obs in a row, each obs unit in a table)
  + First assess visually – view in spreadsheet or table – and note the issues.
  + Then assess programmatically – types, counts, nulls, values, duplicates – for tidiness rules.
* Clean data
  + Programmatically – define in words, code and test each issue found.
  + Ensure you copy the dataframe before doing any cleansing.

**PROJECT**

Wrangle WeRateDogs Twitter data to create interesting analyses and visualizations using python. The project supports the three stages of the data wrangling process and

the work culminated with storing, analysing and visualising the cleansed data.

E:/My Documents/Paula/DATA ANALYST NANO DEGREE/TERM 2/ DataWrangling/Project

* wrangle\_report.pdf
* act\_report.pdf – the actual code and comments.
* Wrangling review.doc

**REVIEW FEEDBACK/REFLECTION**

This felt like it should have been a fairly simple and quick project but it was very intensive and the cleansing could have gone on for much longer. I think you need to be clear about how much cleaning is really necessary as there would always be more to do.

There are more succinct ways to cleanse – better use of functions and better use of Jupyter functionality.

Also important to note that actually gathering the data can be a big, time-consuming job!

1. DATA STORYTELLING – TABLEAU

This was the Explanatory phase of the data analysis process. It concerned presenting insights in a focused way, using the best visualisation techniques to engage the audience, using polished visualisations.

Visualisations should

* Have high data ink ratio – there should not be superfluous information that is not part of the main message and remove chart junk.
* Use simple colours – perhaps just one colour that is colour blind friendly (blue/orange) or just black. Only use a different colour if you want to make something stand out.
* Not necessary to use 3D visuals.

When presenting findings start with a question, then deeper questions. Use visuals to help.

Tableau allows you to first create visuals on a work sheet, drag worksheets to a dashboard, and add dashboards in order to create a datastory.

Visuals can be interactive – use hovers and interactive filters.

**PROJECT**

The task was to use Tableau to create an explanatory data visualisation from the Baseball dataset to communicate which factors influenced the performance of batters.

I firstly quickly visualised the variables (input and response variables) then looked at the response variables more closely and ended with a conclusion.

A big part of this project was sharing with others and getting feedback then providing new iterations of the presentation.

E:\My Documents\Paula\DATA ANALYST NANO DEGREE\TERM 2\Visualisation\Tableau\Project

* Data\_Story\_Writeup.pdf

**REVIEW FEEDBACK/REFLECTION**

* Tableau Project review.doc

I really liked Tableau as a tool and couldn’t believe how simple it was to produce something really effective for an audience.

I think Tableau would be used to create a final interactive product for users once preliminary data analysis had been performed – e.g. EDA, wrangling, etc.