CPS3235 Data Science: From Data to Knowledge

Study-Unit Assignment

Marc Ferriggi

Supervised by Dr Jean Paul Ebejer

Department of Computer Science

Faculty of ICT

University of Malta

January, 2019

A study-unit assignment submitted in partial fulfilment of the requirements for the degree of B.Sc. (hons.) Conputer Science AND Statistics and Operations Research.

Statement of Originality

I, the undersigned, declare that this is my own work unless where otherwise acknowledged and referenced.

Candidate	Marc Ferriggi
Signed	____________________
Date	January 20, 2019

Contents

List of Figures

1

Data Storage - Computer Science Papers Dataset

1.1 The Data Model

The data used for this task was taken from the `dblp` computer science bibliography dataset[1]. This data came in the form of an XML file. The interesting features from this file were extracted and stored as a graph database (using Neo4J) in order to perform the required analysis. The chosen model was quite a simple one and can be seen in Figure 1.1 where there are two types of nodes (`Article` and `Author`) and 1 relationship (`PUBLISHED`).

The `Author` node simply contains the name of the author. This field was also set as the key since according to the documentation of `dblp` the name is stored in a way which is guaranteed to be unique. The `Article` Node, despite its misleading name, stores data on any of the publications listed in the `dblp` dataset. This includes articles,articles in proceedings, proceedings, books, articles in collections, PhD thesis, Masters thesis and publications from web sources. These nodes have 2 fields; a key field which contains the

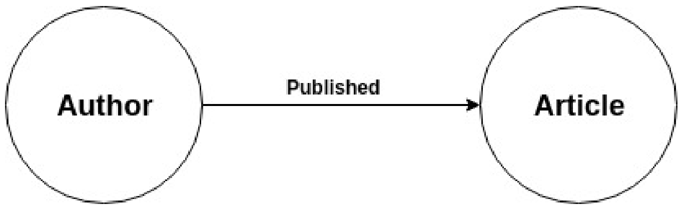

Figure 1.1: Graph Model Used to Store the `dblp` Data in Neo4J

[1]https://dblp.uni-trier.de/xml/

unique key identifier assigned to them in the `dblp` database and the title field containing the name of the publication. Finally, the relationship `PUBLISHED` contains no fields as it wasn't necessary.

In order to import the data into Neo4J, the following steps were done:

- Firstly, Neo4J was installed on the local machine. The command `sudo service neo4j status` was used to check the status of the server and since it was inactive `sudo service neo4j start` was used to start the server.
- The Jupyter Notebook code in '01 - Data Extraction.ipynb' was run to create the data files.
- The generated data files were then copied to the Neo4J import directory by running `sudo cp [file.csv] /var/lib/neo4j/import/[file.csv]` .
- Finally, some quotes found in titles of the publications were removed by running `sudo sed -i ‘s/\"//g’ articles.csv`

In particular, in the python file, `lxml` was first used to extract the required features from the xml file into three csv files (one for each node and one for the relationship). This was slightly tricky to handle due to the size of the xml file and the amount of memory available on the laptop used to run this program. Thus, a generator function was used that would access the xml file element by element rather than storing the whole file in memory. Similarly, the csv files were accessed only when needed in order to ensure that not too much memory we being used. This was a necessary step that traded speed for memory. `py2neo` was then used to access the database, set the uniqueness constraints on the desired fields in the nodes and load the csv data.

1.2 Some Cypher Queries

After loading everything in Neo4J, some queries were run in Cypher using `py2neo` to explore the dataset. In particular; the amount of `Article` nodes were counted and it was found that there are a total of 6,661,235 nodes, the amount of actual articles were counted and it was found that there are a total of 1,956,852, and the author that published the most articles was found to be H. Vincent Poor, who published 1,568 articles! The following queries were used to obtain this information:

Count the number of `Article` nodes:

```
MATCH (n:Article) RETURN count(*)
```

Count the number of articles:

```
MATCH (n:Article) WHERE n.key STARTS WITH ‘journals’
RETURN  count(*)
```

Get the top 10 authors that published the most amount of articles:

```
MATCH (a:Author)-[p:PUBLISHED]-(:Article)
RETURN a.name, count(p) as rel_count
ORDER BY rel_count desc LIMIT 10
```

1.3 Authorship Sub-Graph

Figure 1.2 was then extracted to show the authorship subgraph for members of the Computer Science department at the University of Malta. The list of members was taken from the department website[2]. The following Cypher query was then used to extract the nodes as required:

```
MATCH (a:Author)-[p:PUBLISHED]->(b:Article)
WHERE a.name in [‘Kevin Vella’,‘Keith Bugeja’,‘Christian Colombo’,
    ‘Joshua Ellul’,‘Adrian Francalanza’,‘Mark Micallef’,
    ‘Gordon J. Pace’,‘Sandro Spina’,‘Mark Vella’,
```

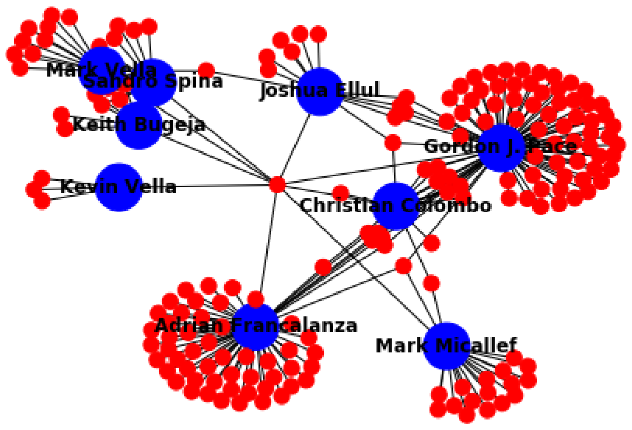

Figure 1.2: Authorship Sub-Graph of the CS Department at UoM

[2]https://www.um.edu.mt/ict/cs/about/staff

```
    'Kevin Cortis']
RETURN a,p,b
```

The extracted table was then converted into a `pandas` data-frame and `networkx` was used to display the extracted sub-graph.

1.4 Shortest Paths

The shortest path function in Cypher was then used to find the shortest path between two different authors. The first query was run to show one of the shortest paths that exist from Dr Jean-Paul Ebejer to Prof. Gordon Pace from the University of Malta in order to test out the function between two authors that are known to be closely connected.

```
MATCH (p1:Author {name:'Jean-Paul Ebejer'}),
    (p2:Author {name:'Gordon J. Pace'}),
    path = shortestpath((p1)-[:PUBLISHED*]-(p2))
RETURN path
```

The second query was then run to find the shortest path between Jean-Paul Ebejer and Donald E. Knuth.

```
MATCH (p1:Author {name:'Jean-Paul Ebejer'}),
    (p2:Author {name:'Donald E. Knuth'}),
    path = shortestpath((p1)-[:PUBLISHED*]-(p2))
RETURN path
```

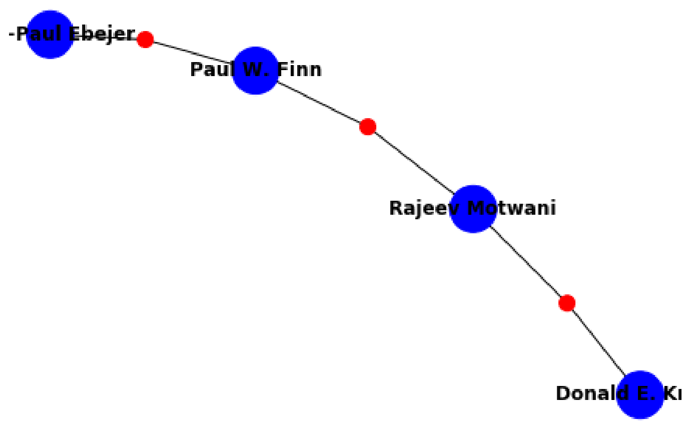

Figure 1.3: Path from Jean-Paul Ebejer to Donald E. Knuth

It was found that a connection actually does exist between these two authors and the path returned can be seen in figure 1.3.

1.5 Erdős Numbers

A function was then written to extract the Erdős Number of a given author. The function made use of the following query (where `NAME` refers to the author name passed to the function):

```
MATCH (p1:Author {name: NAME}),
    (p2:Author {name:‘Paul Erd&#246;s’}),
    path = shortestpath((p1)-[:PUBLISHED*]-(p2))
RETURN length(path)
```

The Erdős Number of Dr. Jean-Paul Ebejer was then found to be 4. Figure 1.4 was then plotted to show the Erdős Numbers of the members of the Faculty of ICT at UoM. Note that the members of the faculty were taken from each department's website. A histogram was also plotted for the Erdős numbers to further show the distribution of these values and can be seen in figure 1.5.

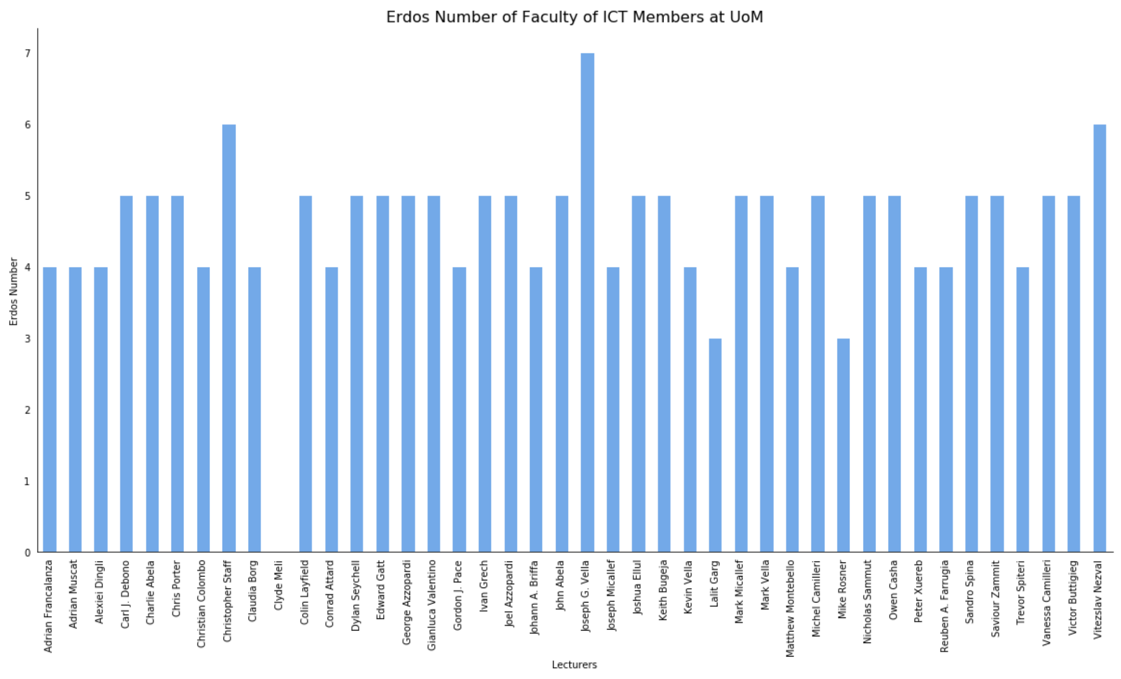

Figure 1.4: Erdős Numbers for members of the Faculty of ICT at UoM

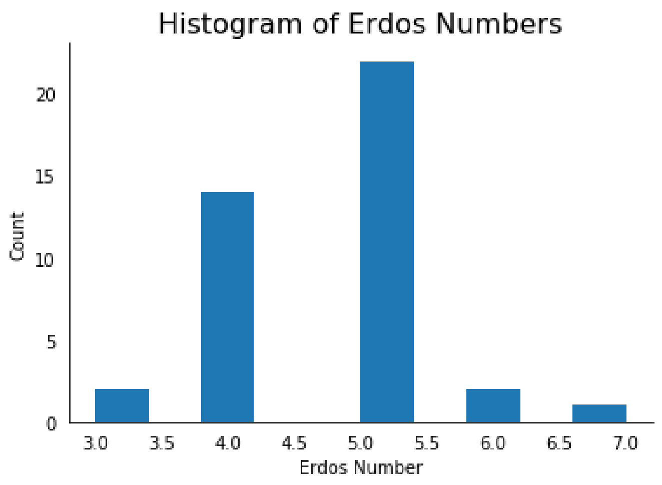

Figure 1.5: Histogram of Erdős Numbers for members of the Faculty of ICT at UoM

1.6 Six Degrees of Separation

The six degrees of separation concept states that all living things are six or fewer steps away from each other. In order to apply this to the `dblp` dataset and check if this statement holds up in this case too, the Erdős Number Function was used in order to check if all authors have an Erdős number of six or less. A sample of 1,000 authors was taken by randomly sampling the list of author IDs. Each author's Erdős number was then calculated and figure 1.6 was plotted.

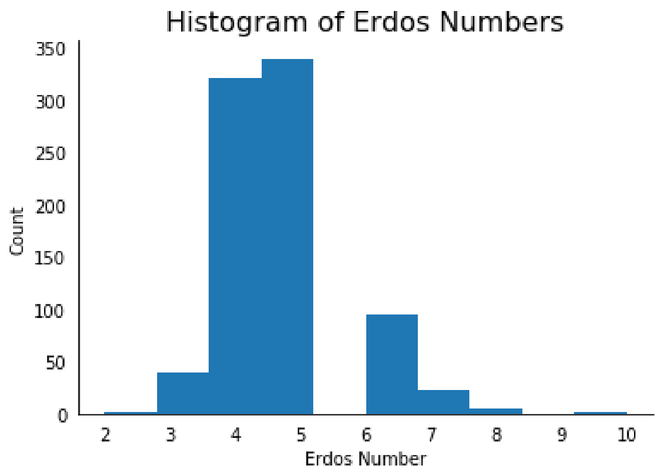

Figure 1.6: Histogram of Erdős Numbers for the Randomly Selected Sample of 1,000 Authors

As can be seen in figure 1.6, not all authors are connected to Paul Erdős in six steps or less, however the vast majority are. In fact, for this particular sample, the average Erdős number was found to be 4.7. Thus, although not all authors are connected to Paul Erdős in six steps or less, the six degrees of separation concept hods up on average and it can be concluded that the majority of authors have an Erdős number of 6 or less.

1.7 Relational DBMS

In order to store this data in a relational DBMS using a similar model, 3 tables would be needed; an author table, an article table and a published table. This would work in a similar way to the nodes and relationships of the graph database where the author table would store all the details of the authors, the article table would store all the details or the articles and the published table would link the primary keys of the other two tables to form the many-to-many relationship required.

An advantage of using a relational database over a graph database is that relational databases have been the go-to method of storing data for a while, thus there are many research papers written on these and one can find information on running queries and ways to store data properly very easily. Also, anyone with basic knowledge in computing will be able access and use a relational database as its easily accessible with software packages such as Microsoft Access.

Furthermore, relational databases use the ACID model which ensures:

- Atomicity: where everything in a transaction must succeed in order to be implemented.
- Consistency: where a transaction will only leave the database if it is consistent with the data.
- Isolation: which ensures proper concurrent transaction handling.
- Durability: where transactions that are completed are persistent and will be there if the server needs restarting.

When using relational databases however, due to the fact that a *JOIN* operation must match the primary and foreign keys of all the entities in a table, these are computationally and memory intensive and thus have exponential cost. This is especially bad for many-to-many relationships such as the *published* relationship being used in this model as the published table would increase further the cost of the *JOIN* operation. Graph databases were designed with this disadvantage in mind, ensuring that the connections are clear, where each node stores a list of all the connections in it, thus the *JOIN* operations are not needed. This however comes at the cost of memory as a node with many connections will take up more memory than one with no connections at all.

2

Data Extraction and Visualization - EU Stats

2.1 Introduction and the Datasets Chosen

The datasets used for this task were both taken from the Eurostat portal. Links to download the data can be found in the 'README' text file found in the data folder provided for this task. The dataset chosen where Malta ranks surprisingly well is that of the number of healthy life years. According to the eurostat website, healthy life years measures the number of years that a person at birth is still expected to live in a healthy condition. The dataset chosen were Malta ranks surprisingly badly is the employment rate data, specifically the female employment rate.

2.2 Healthy Life Years

Since 2006, the number of healthy life years has increased from 69.5 years to 72.4 years for females and from 68.3 years to 71.1 years for Males, however this increase has not been linear. In fact, when looking at figure 2.1 we can note that the number of healthy life years in Malta peaked for both males and females in 2015. Since 2015, the healthy life years for females has dropped by by 2.2 years while that of males has dropped by 1.5 years. An interesting study would be to determine the source of this decrease and to answer the question *"Will this value continue to decrease in the future?"*

Surprisingly, when comparing the Maltese Data to the rest of the EU, Malta ranked the second best in 2016 when it comes to healthy life years, being only beaten by Sweden with an average healthy life years value of 73.15 years (where Malta's average healthy

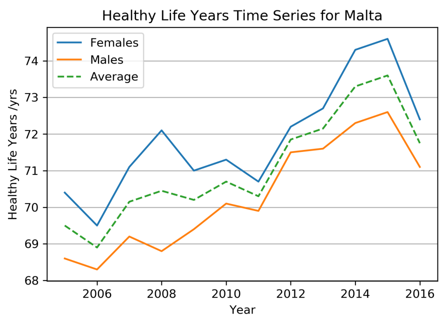

Figure 2.1: Time Series Plot showing Malta's Expected Healthy Life Years

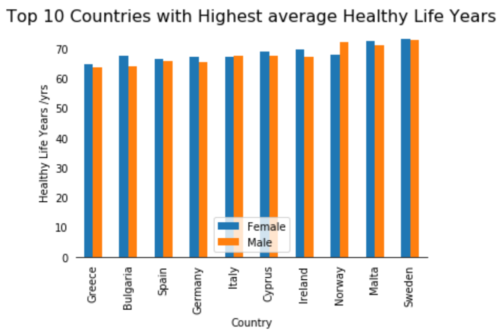

Figure 2.2: Top 10 EU States with the Highest Expected Healthy Life Years

life years value is 71.75). Figure 2.2 shows the top 10 countries in the EU with the highest healthy life years expected value.

2.3 Employment Rate

Sine 2005, the employment rate in Malta has been on a nearly constant increase, having only dipped slightly in 2009. In particular when looking at the time series plot for the employment rate in Malta (figure 2.3) we can note that since 2009, female employment rate has increased significantly. The rate of this change has also increased but although this is a positive step in the right direction, these numbers are not too good when compared to other EU States. In fact, as can be seen in figure 2.4, Malta ranks in the top 10 worst countries for female employment rate!

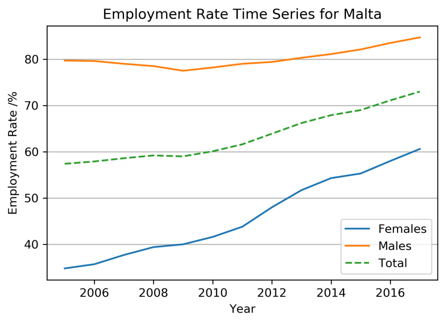

Figure 2.3: Time Series Plot showing Malta's Employment Rate

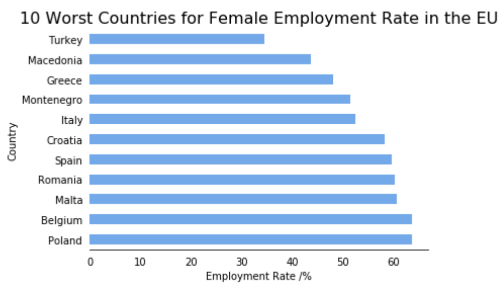

Figure 2.4: Top 10 worst EU States for Employment Rates

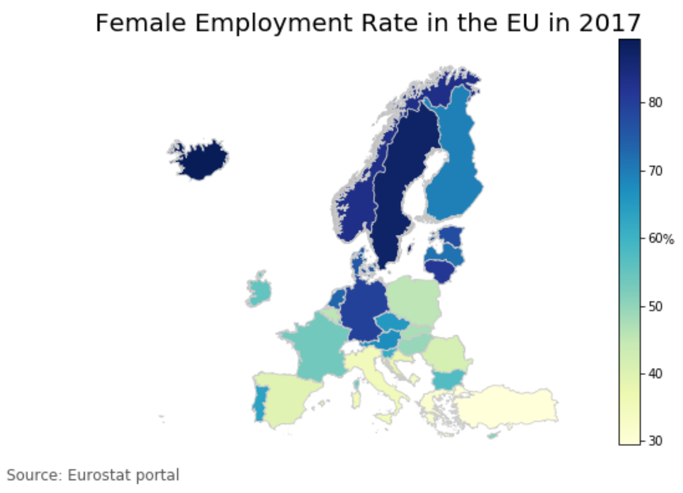

Figure 2.5: Female Employment Rate in the EU States in 2017

The geovisualization plot seen in figure 2.5 shows the female employment rates in the EU states for 2017.

2.4 Criticising Local Media

The bad visualization from local media was taken from a *Malta Independent* article published on Sunday 2nd December 2018 titled "Determinants: What is causing the rise

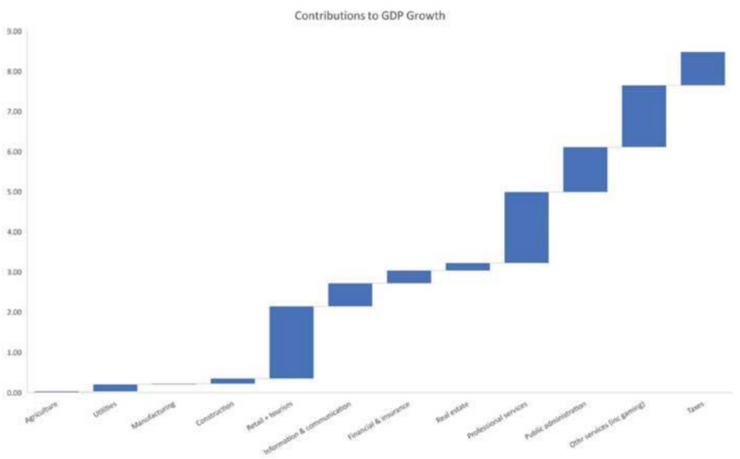

Figure 2.6: Bad Visualization taken from Malta Independent

in property prices?" and can be seen in figure 2.6. The most glaring issue with this plot is the fact that it is represented as a cumulative bar chart where each category's bar starts from the top of the prior category's bar. This is not an ideal visualisation as it makes it very difficult to compare the categories' contribution to GDP growth. Furthermore, it gives the impression that the order of the categories is important and in this case, the order is simply random. The graph even does a bad job at showing that all these categories add up to 8.5% of the GDP growth as the y-axis has no label and there are no grid lines to help in reading the values presented.

The improved plot can be seen in figure 2.7. All the issues addressed in the prior paragraph have been sorted out.

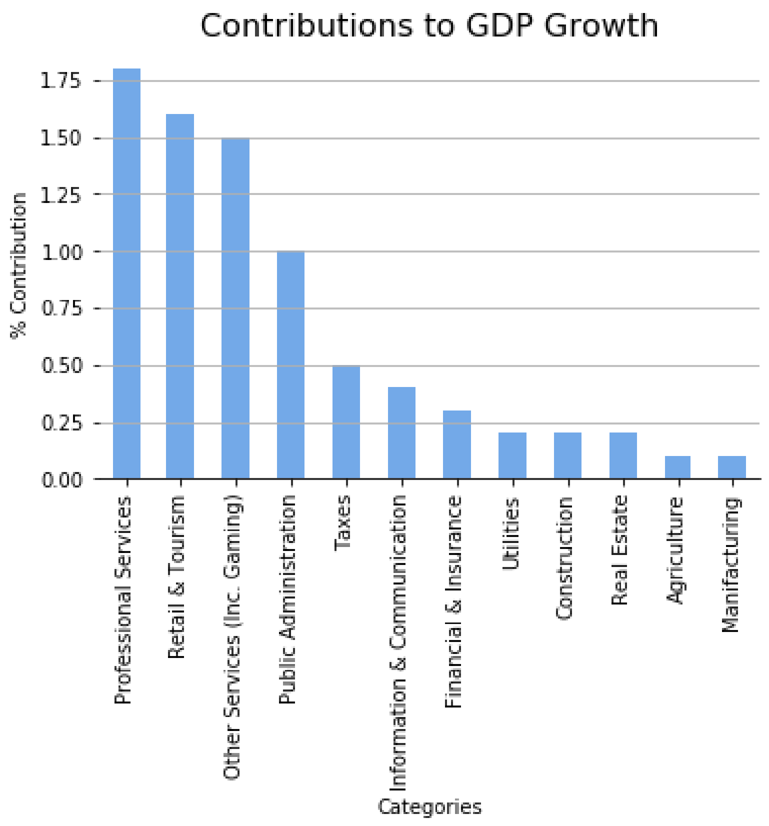

Figure 2.7: Contributions to GDP Growth

3

Data Science Project - Dataset Analysis

3.1 The Data Supplied

The data supplied was taken directly from *Times of Malta* classified listing page on some of the days when the listings would go live from the 23rd of April 2015 to the 24th of October 2018. For mostly every week in that time frame, the data was either collected on a Monday or on a Wednesday or on both those days of the week. Each file contains the property listings for a whole week, thus there are duplicate entries in the dataset. Note that due to the differences in the time frame between each data point, if this data is going to be used for time series analysis then a sample needs to be taken with even time intervals to ensure accuracy of the tests.

The data collected is simply the *html* code used to display the web page. This is available to anyone that has access to the website, thus anyone can have access to this data given the page was accessed on the specified date. Since anyone can book a classified advert on *Times of Malta*, there is no guarantee that the information is correct or accurate, thus we cannot be certain that the prices of the properties listed on the classified page reflect the true market value of property in Malta set by experts in the field. However, this could still give us a good indication of the trend.

With regards to data quality, the data provided is not complete as not all listings have information on the area of the house, whether or not the house has a garage or information on the type of property. However, from glancing at the raw data, it would seem that the majority of entries have data on the location and price of the property. Since the data is extracted from the same system it must be consistent. Finally, when doing a study on the current property situation in Malta, data from 2015, 2016 or even 2017 will not be relevant or timely as the property marked has changed drastically in these past years in Malta.

3.2 Features of Interest

The features of interest from the raw data were extracted with the task in mind of building a predictive model that would predict the expected price of the property given several features. Thus, a sample was chosen from the provided dataset. In particular, datasets from the beginning of August, September and October were chosen since these would reflect most accurately the current prices in the property market.

- Property ID: This will be extracted from the *name* variable in the *html* code in order to have a unique ID referencing each entry.
- Location: This categorical variable will store the location of the property. Since the price of property depends on the location, this would make a good predictor for the model.
- Property Type: Another categorical variable which lists the type of property for sale. The categories are the following: house, penthouse, maisonette, apartment, farmhouse, villa, house of character, block or unknown.
- Plot Area: This continuous quantitative variable stores the area of the land in square meters (thus its measured in a ratio scale). The plot area of the land should have a direct effect on the final price of the property, thus it should make a good predictor.
- Has Pool: This dichotomous variable takes a value of 1 if the property listed has a pool and 0 otherwise.
- Has Garage: This categorical variable has 3 categories; yes (1), no (0) and optional (2).

3.3 Feature Extraction

The data was extracted from the provided *html* files by using the *Python* package *Beautiful Soup*. Once the property listings were found, the features were extracted using regular expressions and python's string manipulation libraries. The data in the location category was then arranged to ensure there's only 1 category per location. Finally, the duplicated entries were dropped, leaving a total of 1422 unique entries.

3.4 Some Interesting Results

Following some analysis on the extracted data, some interesting statements were initially made. Note that at this stage no statistical tests were done as this would be done later on in the analysis.

The first interesting conclusion was made on data to do with properties having pools. Firstly, the average value of properties with pools seems to be significantly larger than those without pools (as expected). In fact the average value of a property with a pool (regardless of location or size) was found to be €818,616.07 while the average value of a property without a pool was found to be €342,582. Furthermore, as can be seen in figure 3.1, at the time of the study, the locality selling the most properties with pools was Zejtun.

Another interesting result found during the data exploration stage was that currently on the market, Sliema is the locality that has the larges number of properties for sale. The top 10 localities selling properties can be seen in figure 3.2.

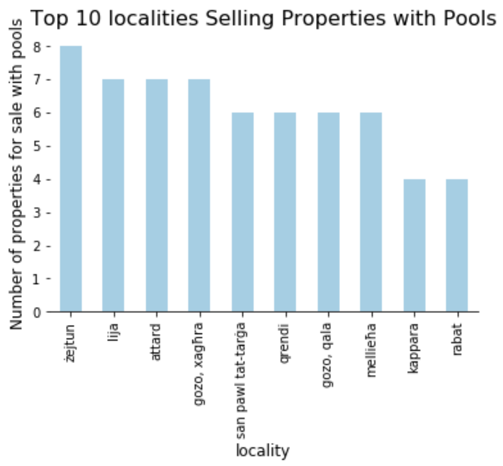

Figure 3.1: Localities Selling the most Properties with Pools

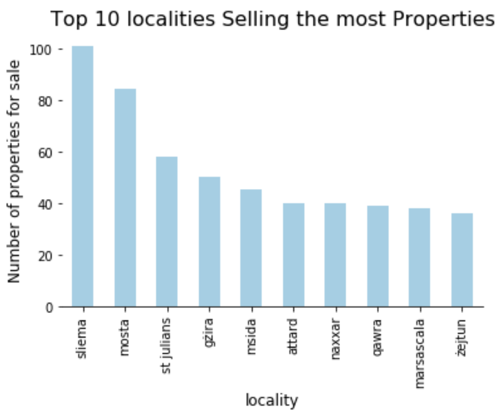

Figure 3.2: Localities Selling the most Properties

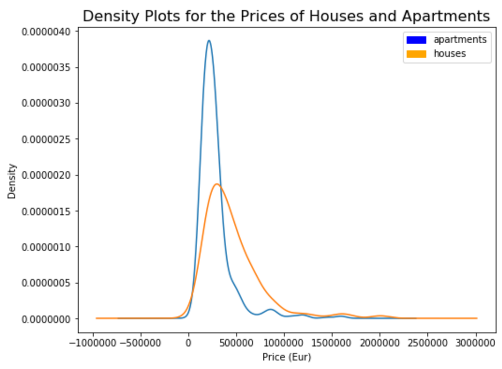

Figure 3.3: Density plots for the Prices of Houses and Apartments

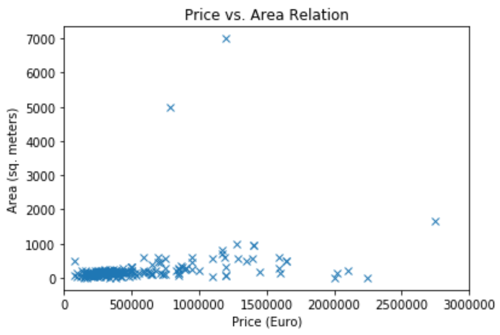

Figure 3.4: Scatter Plot of Price vs. Area

One of the questions that would be interesting to look at was *"which locality has the most expensive land?"*. In order to answer this, the price per square meter of land for each property on the market was worked out by dividing the price by the area. The average was then worked out for each locality using the *groupby* function and it was concluded that the most expensive land on the market right now is in Pender Gardens with an average of €21,818.18 per square meter! Note that at this stage, properties with less than 35 square meters of land were removed as they were considered to be outliers.

The last interesting conclusion was made when analysing the property types. In general, as expected, the most expensive property type for sale was found to be the block of apartments which have an average price of €1,414,214. However it was interesting to note from looking at the density plots for the prices of apartments and houses (figure 3.3) that prices for houses have greater variability than those of apartments.

3.5 Identifying a Correlation

The two variables that would seem to have an obvious correlation would be *price* and *area*. As the area of the land increases we would expect the price to increase too. In fact, having a quick look at the scatter plot in figure 3.4 (after limiting the x-axis to remove some outliers) seems to confirm this. The outliers are points that are affected by the locality, it's obviously important not to remove these when building the model as the locality will be a key feature in the model.

A Pearson correlation test was then run and the data was found to have a correlation coefficient of 0.4861. The p-value of the test was found to be 1.84252e-15 which is less than 0.01, thus we are 99% confident that the value is significant.

3.6 Statistical Analysis

Initially, tests for normality were done on the continuous variables, these tests are important as they would decide weather to use parametric tests or non-parametric tests in the future, as well as decide what kind of model to build to predict the prices of properties. In order to ensure the accuracy of the normality tests, two different tests were run; the Shapiro-Wilk normality test and the D'Agostino-Pearson omnibus test. Both these tests test the null hypothesis that the data is sampled from a normal distribution. Both these tests on both the continuous variables gave us the result that our data does not in fact come from an underlying normal distribution.

After running the normality tests, a test to see if there's a significant difference in the area of the property if it has a garage was run. It was decided to use the Mann Whitney-U Test with continuity correction as this is a non-parametric alternative to the independent sample t-test which tests for a significant difference in the medians of two independent groups. The test results showed with 99% confidence that as expected, the average area of properties with a garage was significantly larger to those without a garage. Note that more details on the test can be seen in the markdown cells in the JuPyter notebook file '03 - Statistical Analysis'.

3.7 The ANCOVA Regression Model

The ANCOVA (Analyss of Covariance) regression model is similar to the ANOVA and OLS regression models, however it caters for categorical variables as predictors as well as covariates. The equation of this model is similar to that of the ANOVA model and can be seen in equation 3.1.

$$\boldsymbol{y} = \mathbb{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{3.1}$$

Where $\boldsymbol{y}$ is an n-vector of independent normally distributed responses, $\boldsymbol{\beta}$ is a p-vector of unknown regression parameters, $\boldsymbol{\epsilon}$ is an n-vector of error terms whose elements are assumed to follow a standard normal distribution and $\mathbb{X}$ is an $(n\text{x}p)$ matrix whose values are both real values and dummy 0-1 indicators.

Similar to ANOVA regression, the equation for estimating the vector of parameters can be seen in equation 3.2.

$$\hat{\boldsymbol{\beta}} = (\mathbb{X}'\mathbb{X})^{-1}\mathbb{X}'\boldsymbol{y} \tag{3.2}$$

$\mathbb{X}'\mathbb{X}$ has no inverse for the columns of each of the dummy variables since these are linearly dependent, thus a column from each of the dummy categorical variables must be dropped before performing the analysis, this is known as intrinsic aliasing.

The following assumptions were made when using the ANCOVA regression model:

- The response variable must have a normal distribution.
- The response variable must be highly correlated with all quantitative predictors.
- There shouldn't be any multicollinearity between the quantitative predictors.

Note that in this case, the response variable is not normally distributed, however it was decided to use this model anyway, keeping in mind that some degree of accuracy is lost due to this. A work-around solutions to this would be to use a different GLM (General Linear Model) that transforms the response variable using a link function. It was decided not to do this however, as this would greatly increase the complexity of the model which seems beyond the scope of this task.

The Model was first run to include *location* as a predictor, however the majority of the coefficients turned out to not be significant, thus *location* was removed and the model was built using *area*, *garage*, *property type* and *pool* as predictors. The final output shows us (from the adjusted $\boldsymbol{R}^2$) value, that this model explains 62.5% of the total variation of the dependent variable, which is what is to be expected when building models of this type.

The testing set was then used to test the accuracy of the implemented model and it was found that the model was quite inaccurate overall, having an average error of €299,444.35, rendering the model rather useless. Some solutions to building a better model of this type are to have a larger training set and to include interaction effects to the model with the hopes of building a more robust parsimonious model.