Ciara Brady (Ciara.brady.1@ucdconnect.ie)

A picture containing clipart

Description automatically generated

UCD Professional Academy – Review of Dublin Bikes availability – Q3 2020

Project Report

# GitHub URL:

https://github.com/cbrady237/UCDPA\_C\_Brady

# Abstract

This project reviews the bikes available at Dublin Bike stations in Q3 2020 with specific analysis performed on July 1 2020. Features include summarising the total number of bike stands available per station, the percentage of availability and a comparison of different stations.

# Introduction

This project usecase relates to a service I use frequently while living in Dublin which is Dublin Bikes. Dublin Bikes are a public bike rental scheme which operates all across the city station with the use of docking bike stations. The usecase is freely available to all and provides for interesting review. This project explores the very basics of what can be done with the data provided for Dublin Bikes based on the modules studied as part of the UCD Professional Academy Professional Certificate in Data Analytics.

# Dataset

<<Provide a description of your dataset and source. Also justify why you chose this source>>

This real world datasets used in this project all relate to Dublin Bikes and are provided by SMART DUBLIN at the following location:

<https://data.smartdublin.ie/dataset/dublinbikes-api>

The following datasets were used as part of this process:

1. Dublin Bikes API From JCDecaux (<https://data.smartdublin.ie/dataset/dublinbikes-api/resource/9a0b87ea-eb4e-489b-9dab-05411ffaf6c1>
2. Dublin Bike Stations GPS Co-ords (<https://data.smartdublin.ie/dataset/33ec9fe2-4957-4e9a-ab55-c5e917c7a9ab/resource/2dec86ed-76ed-47a3-ae28-646db5c5b965/download/dublin.csv>)
3. DublinBikes 2020 Q3 usage data (<https://data.smartdublin.ie/dataset/33ec9fe2-4957-4e9a-ab55-c5e917c7a9ab/resource/99a35442-6878-4c2d-8dff-ec43e91d21d7/download/dublinbikes_20200701_20201001.csv>)

As referenced above, I chose this project usecase as it relates to a service I use frequently while living in Dublin which is Dublin Bikes. It also recently got a new sponsor in early 2021 which brought the availability of the API to my attention. The usecase is freely available to all and provides for interesting review.

# Implementation Process

### Step one – Import pacakges

The following packages were imported into pycharm for use in this project:

* Requests
* Json
* Pandas
* Numpy
* Matplotlib
* Seaborn

This was completed in advance to ensure all commands could be processed thoughout.

### Step two – import files

#### API

The Dublin Bikes API file was imported in the following way:

* Assign the URL to a variable: url which includes the contract name and API key.

url = 'https://api.jcdecaux.com/vls/v1/stations?contract=dublin&apiKey=4aed48738d7bb38f74643c9e926692a1c6501a45'

The request was then packaged, the request sent and the response was caught as ‘r’ as per the code in Pycharm / GitHub.

The text of the response was then printed to ensure correctly imported.

1. Import a CSV file into a Pandas DataFrame

The two CSV files referenced above were imported using the following code:

= pd.read\_csv(file)

These were then printed to ensure they were imported correctly.

The datasets required the following additional steps before they could be analysed:

1. Convert the time column to datetime type
2. Add weekday column using the time column

### Step 3: Analysing Data

#### Indexing [sorting, indexing, grouping]

As part of the import process, the Dublin Bike Stations GPS coordinates file (‘dublin bikes’) were indexed by name to allow for further analysis later in the project.

Later in the project, index is used a number of times to allow for functions such as merging datasets and preparing the data for the charts.

Groupby was also used in the project by Name to illustrate how it can be used.

Indexing was chosen as the preferred option for this criteria over Sorting and Groupby as it was required to be used multiple times as part of this project.

#### Dropping Duplicates [Replace missing values or dropping duplicates]

Dropping duplicates was used in this report when aiming to provide a summary of the total number of bike stands available per location. As the file itself contains multiple times per bike station, a summarised file was creating by dropping duplicates to show this. The following code was used to illustrate this.

bike\_stands = df.drop\_duplicates(subset=["NAME", "BIKE STANDS"])

This file was saved as a csv file for reference.

Dropping duplicates was illustrated over missing values as there were no missing values required to be filled in.

#### Loc (Slicing, loc or iloc)

Loc was used to extract all journeys that occurred on a weekday from the complete dataset.

A list was used to define what a weekday was and then loc was used to extract these using that list:

df\_weekday = df.set\_index("weekday")

weekdays = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday"]

print(df\_weekday.loc[weekdays])

Loc was used over iloc or slicing as the column headings were known and this appears to be the most efficient way of extracting the required dataset.

Loc was also used again to prepare data for the charts.

#### iterrows [Looping, iterrows]

iterrows was used to illustrate the data in the ‘bike stands’ dataset that was created earlier when dropping duplicates. The following code was used:

for key, row in bike\_stands\_ind.head().iterrows():

print(key)

print(row)

iterrows was used over looping because I wanted to loop through all rows in the project rather than a specific row.

#### Merge dataframes

The Dublin Bikes geo location dataset was merged with the Bike Stands dataset to illustrate a summarised file of bike locations with just the number of bike stands available.

The following code was used:

dublin\_bikes\_summary = dublin\_bikes\_ind.merge(bike\_stands[["NAME", "BIKE STANDS"]], on = "NAME", how = 'left')

This file was then saved as csv file.

## Python

### Use functions to create reusable code

A code was used to define importing csv files, and printing the shape and the first five rows of the csv file as follows:

def readfile(filename) :

data = pd.read\_csv(filename)

print(data.shape, data.head())

This code can then be reused when importing any csv file as shown in the example in the project.

### Numpy

The availability of bikes on July 1 2020 was indexed by time and extracted using loc. This file was then converted to numpy using the following code:

July0120 = df\_July1.to\_numpy()

np\_July = np.array(July0120)

Numpy arrays were created from this numpy array to show the total number of bikes and the number of available bikes.

np\_total\_bikes = np\_July[:,3]

print(np\_total\_bikes)

np\_available\_bikes = np\_July[:,5]

print(np\_available\_bikes)

These arrays were then used to calculated the percentage of available bikes:

np\_percentage\_available\_bikes = np\_available\_bikes / np\_total\_bikes

print(np\_percentage\_available\_bikes)

### Dictionary or Lists

Lists were chosen to illustrate in this project over dictionaries as lists were relevant to a number of the other functions required.

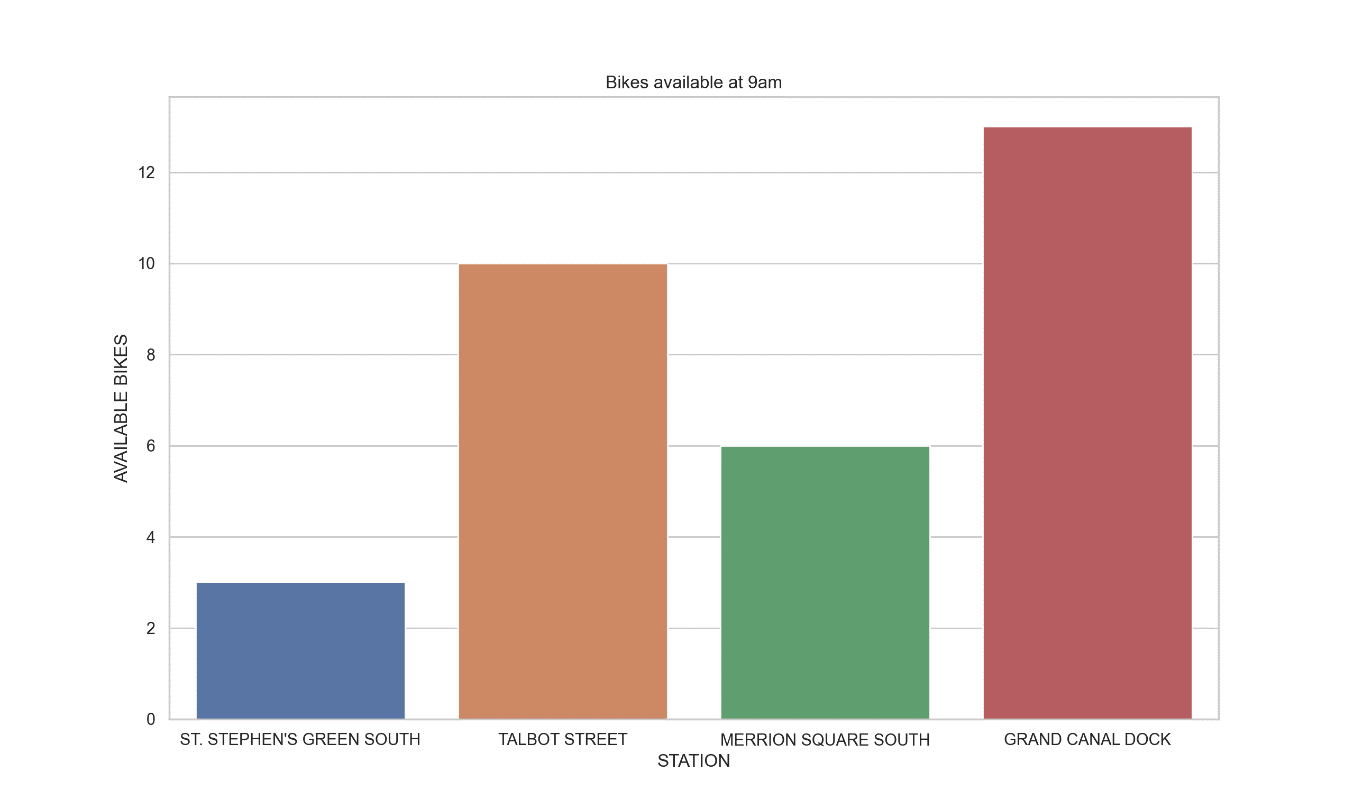
Lists were used as part of the loc process described above when a list of weekdays was required to proceed with the loc function. Lists were also required when gathering information for use in the graphs created for the bike stations chosen for review below.

weekdays = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday"]

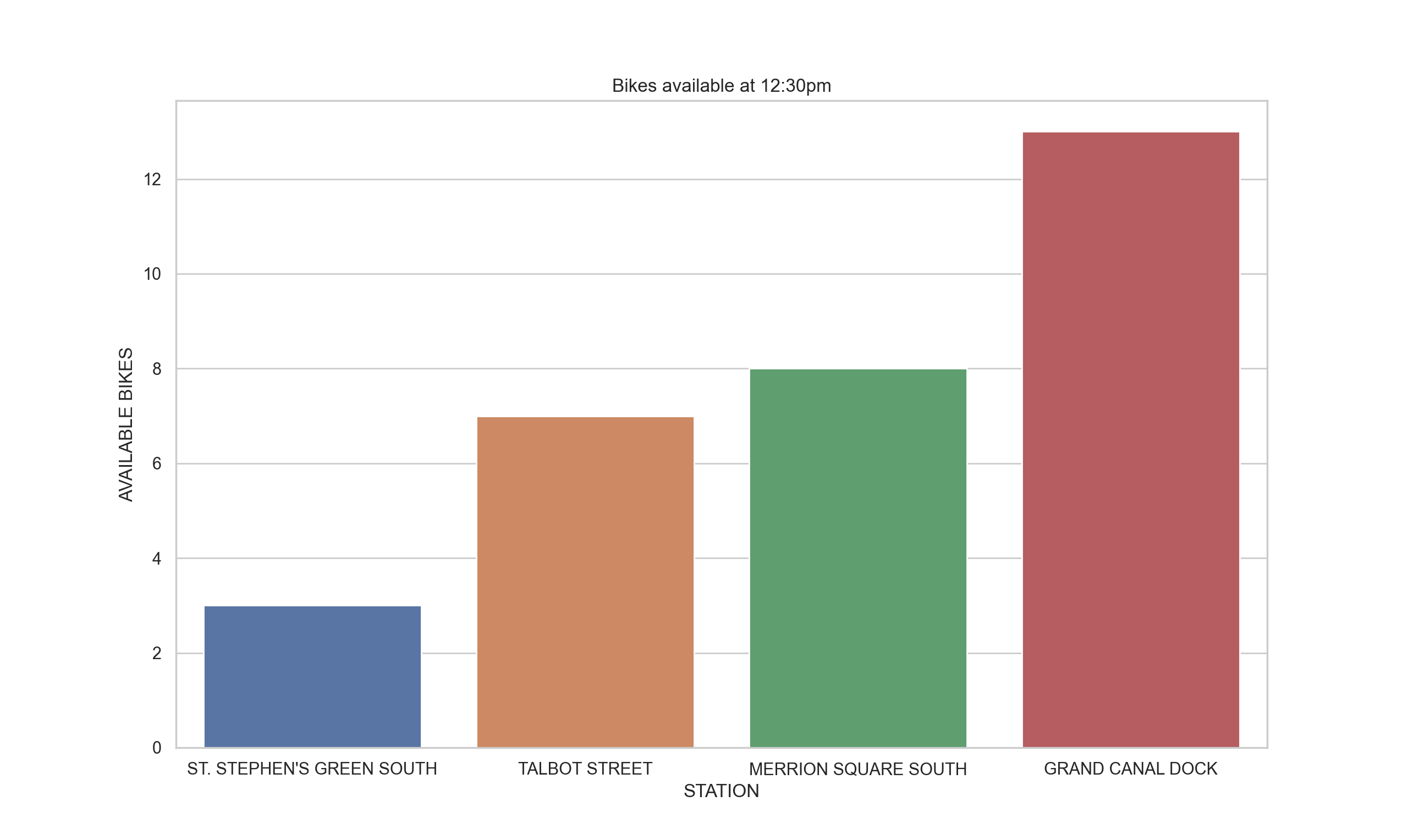
# Results

The number of bikes available at four stations in Dublin City Centre at different times in the day were analysed in a bar chart to assess the level of use of bikes for one select day (Wednesday July 1 2020) at morning peak time (9am), lunch time (12:30PM), and evening peak time (5:30PM). These graphs are detailed were created with Seaborn and are shown below in Figures 1 - 3:

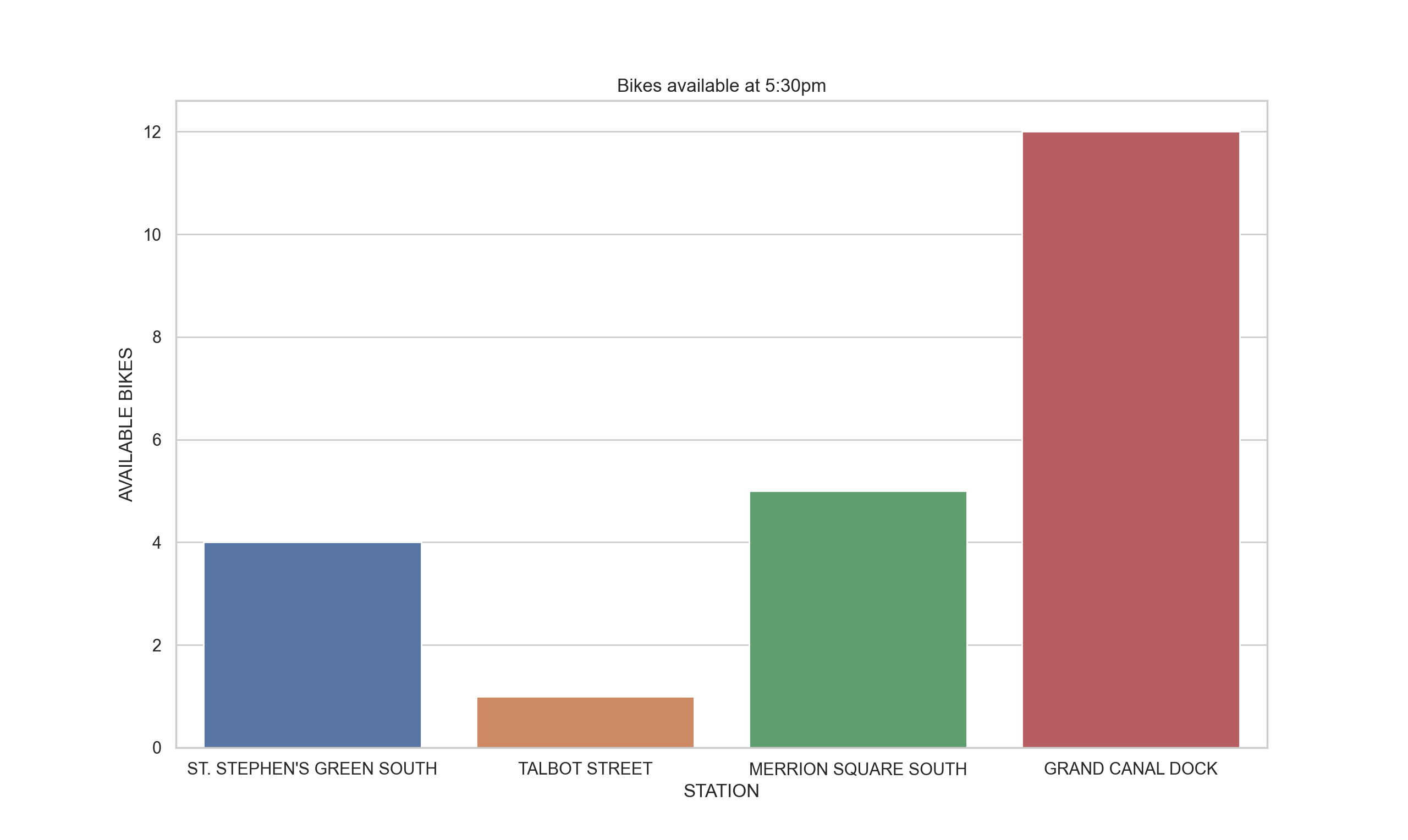
#### Figure 1 – Bike availability - 9:30 AM – July 1 2020



#### Figure 2 – Bike availability - 12:30 PM – July 1 2020

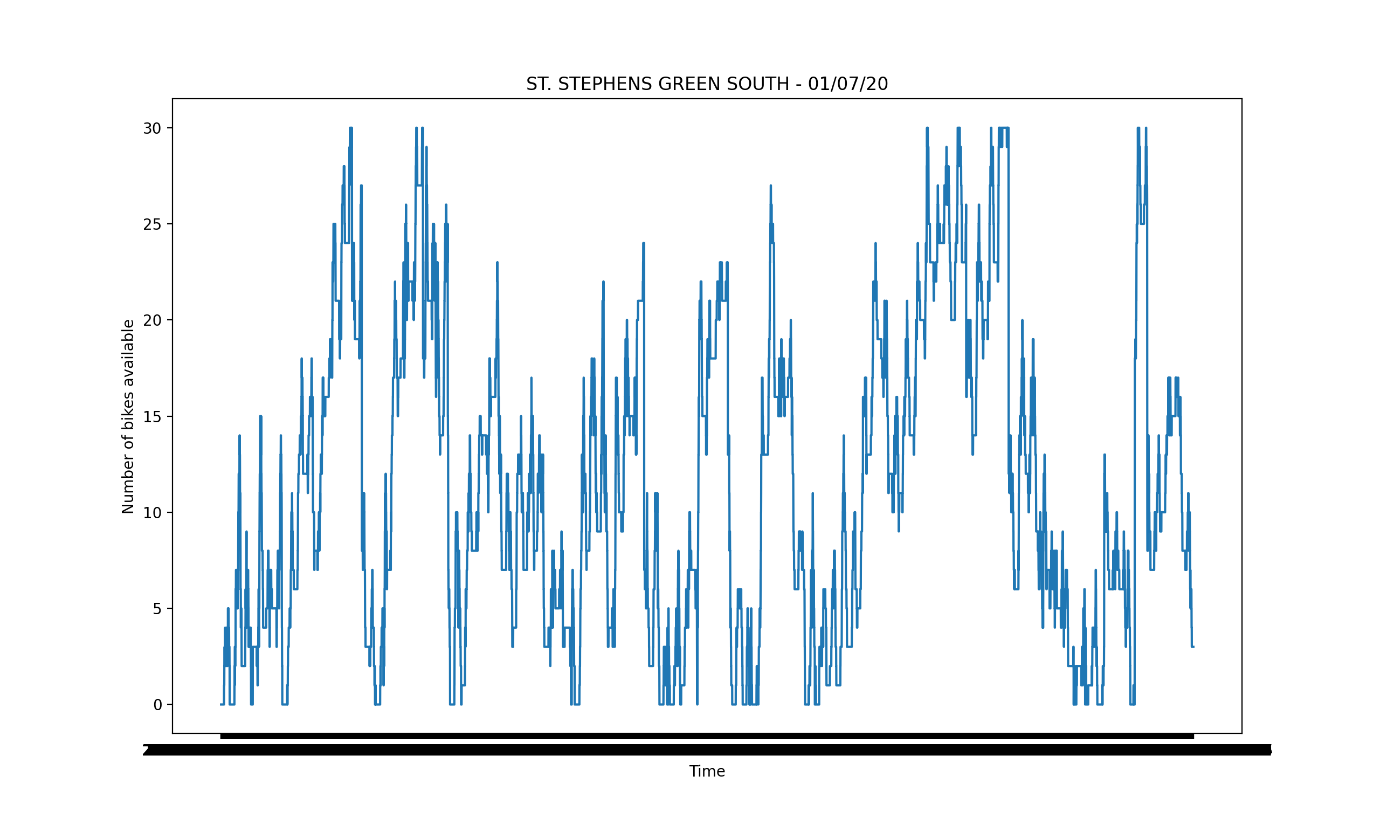


#### Figure 3 – Bike availability- 5:30 PM – July 1 2020



Separately, for the St. Stephens Green South bike station in Dublin City Centre, a chart showing the movement of available bikes throughout a 24 hour period on Wednesday July 1 2020 was plotted using matplotlib. This is shown in Figure 4 below:

#### Figure 4 – St. Stephens Green South bike availability – July 1 2020



# Insights

* At the Grand Canal Dock station, which would predominately be used by workers commuting to Grand Canal Docks, the level of bikes available at the three times analysed remained constant at 13 for 9:30AM and 12:30PM. At 5:30PM, a bike was used, reducing availability. This could indicate that the workers commuting to Grand Canal Dock had arrived by 9AM and are beginning to commute home from work from 5:30PM onwards. This could indicate that this station is a bigger station and have capacity to hold more bikes or also managed well by the operator.
* Talbot Street, which is right in the city centre experienced more change in movement at the times analysed ranging from 10 available at 9AM, 7 at 12:30PM and 1 available at 5:30PM. This could indicate that Talbot Street is used as a commuting station as people cycle into the city centre in the morning and lower availability as people leave the city in the evening.
* Stephens Green South bike station usually has very low availability which is to be expected given it’s location to the Luas and buses, shopping areas, tourist attractions and offices.
* For the chosen time periods, no station was ever empty indicating that they are adequately managed by the operator.
* The number of bikes available may indicate that a station with a consistent level of bikes available is a larger bike station compared to a station that has lower availability constantly.
* While looking at the Stephens Green South data for the 24 hour period, it indicates that bikes are resupplied pre peak hours to allow for use at busier periods throughout the day.

# References

Dublin Bikes files: <https://data.smartdublin.ie/dataset/dublinbikes-api>