Task6 Report for Assignment 1

- A description of the crawling method and a brief summary the output for Task 1.

Crawling Method

In the crawling part of task1, I import some libraries for it, included 'requests', 'beautifulsoup' and 'urljoin'.

Firstly, specify the initial page to crawl, use 'requests.get()' to fetch document from that server and 'BeautifulSoup()' to parse the HTML documents fetched, then name the result as 'soup'.

Secondly, find all the <a> tags in 'soup' , because the <a> tags defines a hyperlink. Use 'urljoin()' to get all the new urls by using the 'href' part in those <a> tags, append them into a list called 'to_visit' for crawling later. And also create a dictionary called 'visited' to store all the url that has been crawled.

Thirdly, using a while loop to keep crawling the url in 'to_visit' list. In the while loop, it would pop out a link from 'to_visit' each time and crawl it, thus the while loop would not stop until 'to_visit' list is empty. For each link popped out, repeat those three processes above which are requesting data(fetch document) from it by 'requests.get()' , parsing the HTML documents by 'BeautifulSoup()' . Then find all the <a> tags in the parsed HTML documents and use 'urljoin()' to get new urls by the 'href' part in those <a> tags. If those new url has not been crawled and not in 'to_visit' list, append it into 'to_visit' for crawling later.

After finishing the processes above, the program would finish a web crawling and visit every page start from http://comp20008-jh.eng.unimelb.edu.au:9889/main/.

Summary of output

The output of the Task1 is a csv file called task1.csv, it contains 100 rows, 2 columns and has two columns headings 'url' and 'headline'. The first column of each row is the Url crawled and second column is the Headline of the article in this Url. Thus, this output contains totally 100 urls which are found by the crawler and the corresponding headlines of the article in it.

- A description of how you scraped data from each page, including any regular expressions used for Task 2 and a brief summary of the output.

Scrape data

In task2, what need to do is extracting the name of first player mentioned and the first complete score in the article of each page.

To extract the first name mentioned, firstly store all the player names from 'tennis.json' into a list called 'names', and then use for loop and 're.search()' to figure out if those names in the article or not. If a name is in the article, store the index represented where the name is into a list called 'index_list' and a dictionary called 'player_dic' would also record the index value as a key and that player name as its value. Secondly, by sorting 'index_list' in order to find the smallest index value, it is the position where the first player name mentioned. Thirdly, using that dictionary 'player_dic' to find the corresponding player name, which key is the smallest index value found. Thus, the player name found is the first mentioned one.

To extract the first complete score, firstly use regular expression and 're.findall()' to find all the possible valid score. The regular expression used is
r'(?:(?:[6-7]-[0-6]\ ?)|(?:[0-6]-[6-7]\ ?)|(?:[\(]\d{1,2}[-|\/]\d{1,2}[\)]\ ?)){2,10}(?:\d{1,2}-\d{1,2})?'.
(The sign (?:) is to create non-capturing group)

It could seperate into two parts to understand, the first part is (?:(?:[6-7]-[0-6]\ ?)|(?:[0-6]-[6-7]\ ?)|(?:[\(]\d{1,2}[-|\/]\d{1,2}[\)]\ ?)){2,10}. The minimum length of this part is 2, for example the score sets '6-1 6-1'. The maximum length is 10, for example the score set '7-6 (7-1) 6-7 (7-2) 7-6 (7-3) 6-7 (1-7) 7-6 (12-10)'. There are three possible situations separated by '|' in this part, (?:[6-7]-[0-6]\ ?) and (?:[0-6]-[6-7]\ ?) are to match the normal games score without brackets which aren't tie bracket, such as 6-5, 7-6, 3-6 and 6-7. The score of the wining side could only be 6 or 7 and the losing side could be between 0 and 6. The rest (?:[\(]\d{1,2}[-|\/]\d{1,2}[\)]\ ?) is for the tie break which must have brackets in it. There are no any restriction of the maximum value of the score in tie-break, but according to the real life tennis gaming, score of the longest tie break does not exceed 99 points. Thus, the scores in the tie-break are assumed to be either one digit number or two digits number.

And the second part is (?:\d{1,2}-\d{1,2})?. In some special tennis matches, they would not have tie-break for the final set, even if they tie at 6 games, and the tennis match wouldn't stop until one player win at least 2 games over the other player, such as the game with score '6-3 6-2 1-6 3-6 10-8' in http://comp20008-jh.eng.unimelb.edu.au:9889/main/ewittsur063.html. Thus, for a complete score set, it may have a score without brackets but the points in it could be either one digit point or two digits point at the end. This second part is to catch those special final set score.

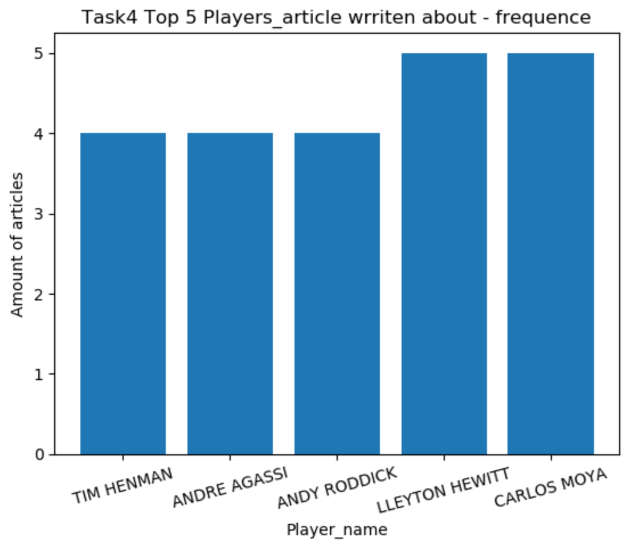
Due to the limitation of the regular expression, it could not detect some invalid score like '6-6' or '(12-9)', the program would further use a for loop and a function called 'check_valid_score()' to find the first complete score in all the match results found by regular expression. The 'check_valid_score' is used to double check the score found is complete valid.

're.findall()' would output a list of score sets satisfied the regular expression and they are arranged in sequential order. Therefore, in this list, the first one gets 'True' from 'check_valid_score()' is the first complete valid score in the article.
Once an article dose not contain a player name or/and a complete valid score, it would be discarded and not be recorded for task2.

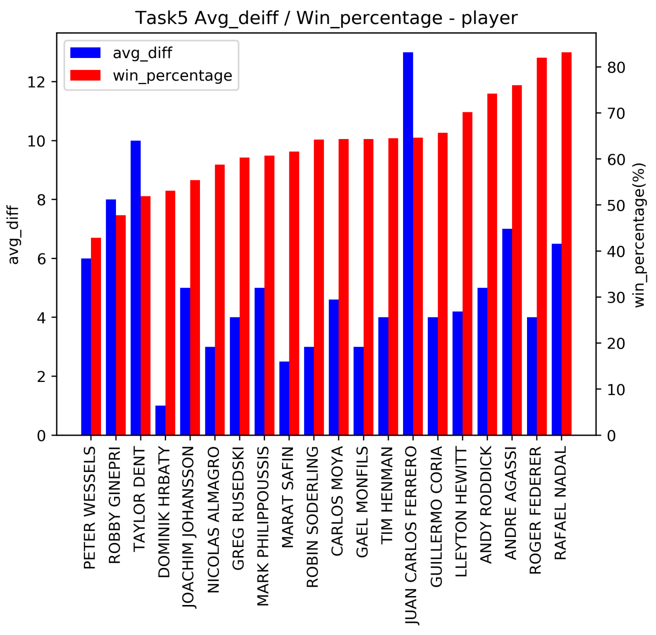
<u>Summary of output</u>

The output of the Task2 is a csv file called task2.csv, it only have 40 rows, 4 columns and has four column headings 'url', 'headline', 'player' and 'score'. The reason why there are 60 rows less than task1.csv is the program discards those articles which dose not have player name or/and a complete valid score. Thus, this output only contains 40 sets of data, each data includes url and headline, first player name mentioned, first complete score in the article of that url.

-**<u>An analysis of the information shown in the two plots produced for Tasks 4 & 5, including a brief summary of the data used. The plots are to be shown (included) along with your analysis.</u>**



The graph in the right is the plot produced for Tasks4, which is a bar plot.
Bar plot is really suitable to summarise data points over a categorical variable. In this case, the categorical variable is player names in x-axis, the frequencies of articles about those players are numeric quantity in y-axis.
The input data for this graph is 5 tennis players' name that articles are most frequently written about and the amount of the articles written on that player.
To get to frequencies, the program assumes that the article is written about the first named player in it. Using the data recorded for task2, the program store the total amount of articles for the players who at least has one article written about him into dictionary. The key is players' name, and the value is the amount of article written about that player. And then, get the top 5 highest frequencies and the corresponding players' name from that dictionary.
In this plot, the data points are arranged from the smallest frequency to the highest. Lleyton Hewitt and Carlos Moya have highest frequency which is 5. Tim Henman, Andre Agassi and Andy Roddick have totally 4 articles written about them. The largest difference of the frequency among those top 5 players is pretty small, which is just 1 article. It shows that the amounts of the articles written about them are very close, the differences are very small.



The graph in the right is the plot produced for Task5, it is a bar plot with 2 y-axis.
The reason why choose a two y-axis bar plot is bar plot could clearly show the data on both 'avg_diff' and 'win_percentage' for each player at one graph for this case.
The input data here is each player's average difference and his win_percentage. The x-axis is the player names in the task2 who has at least one complete valid score. There are two y-axis, one is the average different for each player in left and showed by blue bars, another is that player's win_percentage in right and showed by red bars.
To figure out the average game difference for each player, an assumption that the first complete valid score in an article relates to the first named player's is made. And then the average game difference of each player would equal to the sum of all the game differences of that player divided by the amount of games. On the other hand, the win_percentage for each player is from the 'tennis.json' provided.
In this bar plot, the data points are arranged according to the value of win_percentage of players, from smallest win_percentage to highest. All the win_percentages are above 40%, most of them even greater than 50%. The highest one is Rafael Nadal's which is over 80%, the lowest on is from Peter Wessels, which is about 45%. The difference of the highest and lowest is relatively large, about 40%. The highest average game difference is 13 and lowest one is 1. The difference of them is also large, which is 12 points. Most players average differences are not higher than 6.
In the graph, while the win_percentage is increasing, the average game difference does not show any obvious and regular increasing or decreasing(change). It shows the value of players' average game difference would not be affected by win_percentage.

Combining the information in these two graphes, all of the top 5 players that articles are most frequently written about, Lleyton Hewitt ,Carlos Moya, Tim Henman, Andre Agassi and Andy Roddick have relatively high winning percentage. All their win percentages are greater than 60%. It shows that if a player has higher win_percentage and performs well in the tennis matches, more articles would be written about him and reports the tennis matches he played.

- **<u>A discussion of the appropriateness of associating the first named player in the article with the first match score.</u>**

For some articles, it is appropriate to associate the first named player with the first complete valid score, while those articles only talk about one tennis match and the two players who played this tennis match, no other information about another tennis match or other tennis players. For example, the article of http://comp20008-jh.eng.unimelb.edu.au:9889/main/ohansson033.html is only about a tennis match at the Australian hardcourt championships in Adelaide between Joachim Johansson and Taylor Dent. The First match score is 7-5 6-3 and first name Joachim Johansson. For this kind of article, it is reasonable to say the first match score (7-5 6-3) is from the first name player(Joachim Johansson).

However, there are some article is not only about one tennis match but includes many news or information such as injury of a player or why a player withdraw a tennis match.
For example, the article in http://comp20008-jh.eng.unimelb.edu.au:9889/main/usedskif086.html is about a player withdrew from an Open tennis match due to injury. The first player name is Greg Rusedski, who is an injured player. The First match score is 6-4 7-5. However, this tennis score dose not relate to him but relates to his opponent of next tennis

match scheduled. Thus, to this kind of article, it is not appropriate to associating the first named player with the first match score. It may even cause some mistakes.

In summary, associating the first named player in the article with the first match score is reasonable and suitable for those articles only talk about one particular tennis match and those two players who played this tennis match. However, if the article is on some general information(news) about injury of a player, retire of a player or a player go through to next round and so on, it may be not absolute appropriate to associate these two, even cause some errors.

-**<u>At least one suggested method for how you could figure out from the contents of the article whether the first named player won or lost the match being reported on.</u>**

To figure out if the first named player won or lost, I suggest to use lemmatization toward each words of the first sentence in the article and then determine if any key words represent 'victory' or 'failure' in it. Because the first named player is always the subject in the first sentence and it always reveals if he(she) wins or loses. Lemmatization would be helpful to find key words.

Define two lists, one is winning_wordlist, it has 'win' 'victory', 'beat' and some other words show victory. The other one is losing_wordlist, it contains 'lose', 'defeat', 'fail', 'go down' and other words show failure. All the words in these two lists are in root form.

After doing the lemmatisation toward each word in the first sentence of the article, store all their root form words into a list called 'firstsen_rootwords'.

If a word in the winning_wordlist is also in this 'firstsen_rootwords' list, the first named player may win the tennis match. For example, in http://comp20008-jh.eng.unimelb.edu.au:9889/main/oderling085.html. The word 'win' is in the lemmatised first sentence. Thus, it could figure out that the first player who is ROBIN SODERLING in this article won the tennis match.

On the other hand, if any one word in losing_wordlist is in this list, the first named player may just lose the match reported on. For example, in http://comp20008-jh.eng.unimelb.edu.au:9889/main/ewittfal024.html, the word 'defeat' is in the first sentence of the article. So it figures out that the first named player Lleyton Hewitt lost the tennis match this article reported about.

-**<u>A discussion of what other information could be extracted from the articles to better understand player performance and a brief suggestion for how this could be done.</u>**

Assumption: The player performance represents his(her) overall performance in his(her) career.

To better understand player performance, it would be helpful to extract the information about the seeding of a player in the articles. The top seed is the player the tournament committee deems the strongest player in the field. And then follow by second seed, third seed and so on. Therefore, by knowing which seed the player is, could have a better indication about the player performance. For example, the article in <u>http://comp20008-jh.eng.unimelb.edu.au:9889/main/enmanhop095.html</u> shows that Tim Henman is the third seed. Therefore, we can know that he performed well and is one of the top players in the game, but still two other players are considered better than him.

I suggest to find the seed of player by checking those two words in the front of player's name. In tennis articles, the seeding of a player is always follow by the full name of that player. For example, in the article of <u>http://comp20008-jh.eng.unimelb.edu.au:9889/main/enmanhop095.html</u> writes 'Third seed Tim Henman', in the article of <u>http://comp20008-jh.eng.unimelb.edu.au:9889/main/gassiint096.html</u> writes 'Fourth seed Andre Agassi', 'top seed Roger Federer', in the article of <u>http://comp20008-jh.eng.unimelb.edu.au:9889/main/oyasuffe057.html</u> writes 'Fifth seed Carlos Moya'. Therefore, if the two word in the front of a full player name is combined by an ordinal numeral(or the word 'top') and 'seed', it represents the seeding of that player and show player's performance to some extent.