**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **15-06-2020** | **Name:** | **Karthik J** |
| **Course:** | Introduction to Digital Marketing | **USN:** | **4AL16EC030** |
| **Topic:** |  | **Semester & Section:** | **8TH A** |
| **GitHub Repository:** | Karthik-J |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| Today’s era of Internet has opened a gate of vast variety of opportunities for businesses. Using social networks, one cannot only share a private picture of one’s birthday but also earn customers for one’s business and reach them conveniently. The speed and ease with which the digital media transmits information and help boost a business is amazing. Digital Marketing is the term used for the targeted, measurable, and interactive marketing of products or services using digital technologies to reach the viewers, turn them into customers, and retain them.  The traditional manner of marketing involved businesses to advertise their products or services on print media, radio and television commercials, business cards, bill boards, and in many other similar ways where Internet or social media websites were not employed for advertising. Traditional marketing policies had limited customer reachability and scope of driving customers’ buying behavior. Digital marketing achieves targets of marketing a business through different online channels.  According to [Salesforce](https://www.salesforce.com/form/marketingcloud/2015-state-of-b2b-marketing.jsp), 80% of senior-ranking marketers state that a cohesive customer journey is absolutely critical for success. These marketers understand how a customer or user journey map can streamline processes and provide customers with a consistent experience with a brand. Ideally, a customer journey map transforms the online customer journey into a visually accessible method for digital marketers to use to their advantage.  A customer journey map has proven to be so effective in companies that the [Gartner Group](http://www.mycustomer.com/experience/engagement/why-now-is-the-time-to-map-your-customer-journey) predicts that 60% of large organizations will contain in-house customer journey mapping abilities in 2018. If this percentage of large organizations understand the importance of a comprehensive customer journey map, its due time for other professional digital marketers to begin considering the creation of their own.  To streamline your customer experience and be consistently available to consumers throughout their online experience, read on to uncover how you can create a best-in-class customer journey map for your own brand.  New media marketing centers on promoting brands and selling products and services through established and emerging online channels, harnessing these elements of new media to engage potential and current customers. New media marketing encompasses many different mediums, including display advertising, content marketing and social media promotions. The objective of all new media marketing is to get consumers to interact with the brand, engaging them in a way that increases awareness and correlates to sales.  Brand building is the process of generating awareness and promotion of the services of a company through direct advertising campaigns or through sponsorship. Brand building strategies bring consumers closer to the brand and provide value for them so that they can know, feel and experience the brand. Digital marketers have a big responsibility on their shoulders – promotion of a business – and must take every step to convert that business into a brand so that leads will follow. They work closely with the designing, offline marketing and social media departments. If they collaborate and work simultaneously towards promoting the brand, the business is bound to be successful.  Facebook’s tools cater to the business that wants to form an authentic relationship with their audience. It allows marketers to create and distribute quality content that’s helpful for users. And it allows sales and customer services reps to connect with consumers interested in a brand. It’s not about being spammy, annoying, or deceiving. If you’re building a Facebook Page just to check one more thing off the branding to-do list, think again. True Facebook marketing requires a consistent, long-term commitment. But we promise, the awareness and demand will be worth it.  A major benefit of Facebook advertising is its ability to reach your exact audience. Facebook is the most targeted form of advertising. You can advertise to people by age, interests, behavior, and location. If you really know your customers, you can use Facebook advertising to engage them.  Ad campaigns are made up of ad sets. Ad sets are groups of ads that share settings for how, when and where to run. When you create an ad set, the choices you make at the ad set level automatically apply to all of the ads in the set. You’ll use ad set level options to choose audience characteristics such as location, gender and age. You can also create a budget, set a schedule and choose your placements.  A Facebook Custom Audience is a targeted advertising service that allows businesses to import user email addresses for retargeting on the social media platform. Custom Audiences are an effective way for online businesses to interact with relevant users across multiple channels.  Custom Audiences are essentially a matching system, pairing one mode of contact with that person's Facebook Profile. In most cases, businesses can expect 30 to 70 percent of their contacts to have matching profiles on the platform. Custom Audiences are one of the most highly-targeted forms of marketing, making them increasingly popular with ecommerce stores who want to keep their brand top-of-mind with interested consumers. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Date:** | | **15-06-2020** | **Name:** | **Karthik J** |  |
| **Course:** | | CNN for Computer Vision with Keras and TensorFlow in Python | **USN:** | **4AL16EC030** |  |
| **Topic:** | |  | **Semester & Section:** | **8th A** |  |
|  | **AFTERNOON SESSION DETAILS** | | | | |
|  | **My work:** <https://github.com/Karthikjsannakki/CNN>  **Image of session** | | | | |
|  | **Seaborn: Python's Statistical Data Visualization Library** One of the best but also more challenging ways to get your insights across is to visualize them: that way, you can more easily identify patterns, grasp difficult concepts or draw the attention to key elements. When you’re using Python for data science, you’ll most probably will have already used [Matplotlib](https://matplotlib.org/), a 2D plotting library that allows you to create publication-quality figures. Another complimentary package that is based on this data visualization library is [Seaborn](http://seaborn.pydata.org/), which provides a high-level interface to draw statistical graphics. NumPy NumPy is a Python package used for numerical computation. NumPy is one of the foundational packages for scientific computing with Python. NumPy's core data type is the array and NumPy functions operate on arrays. Installing NumPyInstalling NumPy Before NumPy's functions and methods can be used, NumPy must be installed. Depending on which distribution of Python you use, the installation method is slightly different. Install NumPy on Anaconda If you installed the Anaconda distribution of Python, NumPy comes pre-installed and no further installation steps are necessary.  If you use a version of Python from python.org or a version of Python that came with your operating system, the **Anaconda Prompt** and **conda** or **pip** can be used to install NumPy. Install NumPy with the Anaconda Prompt To install NumPy, open the **Anaconda Prompt** and type:  > conda install numpy  Type y for yes when prompted. Install NumPy with pip To install NumPy with **pip**, bring up a terminal window and type:  $ pip install numpy  This command installs NumPy in the current working Python environment. Verify NumPy installation To verify NumPy is installed, invoke NumPy's version using the Python REPL. Import NumPy and call the .\_\_version\_\_ attribute common to most Python packages.  In [1]:  import numpy as np  np.**version**  Out[1]:  '1.16.4'  A version number like '1.16.4' indicates a successful NumPy installation. Python Lists and NumPy Arrays NumPy is a Python package used for numerical calculations, working with arrays of homogeneous values, and scientific computing. This section introduces NumPy arrays then explains the difference between Python lists and NumPy arrays. Python Lists NumPy is used to construct homogeneous arrays and perform mathematical operations on arrays. A NumPy array is different from a Python list. The data types stored in a Python list can all be different.  python\_list = [ 1, -0.038, 'gear', True]  The Python list above contains four different data types: 1 is an integer, -0.038 is a float, 'gear' is a string, and 'True' is a boolean.  The code below prints the data type of each value store in python\_list.  In [1]:  python\_list = [1, -0.038, 'gear', True]  for item in python\_list:  print(type(item))  <class 'int'>  <class 'float'>  <class 'str'>  <class 'bool'> NumPy Arrays The values stored in a NumPy array must all share the same data type. Consider the NumPy array below:  np.array([1.0, 3.1, 5e-04, 0.007])  All four values stored in the NumPy array above share the same data type: 1.0, 3.1, 5e-04, and 0.007 are all floats.  The code below prints the data type of each value stored in the NumPy array above.  In [2]:  import numpy as np  for value in np.array([1.0, 3.1, 5e-04, 0.007]):  print(type(value))  <class 'numpy.float64'>  <class 'numpy.float64'>  <class 'numpy.float64'>  <class 'numpy.float64'>  If the same four elements stored in the previous Python list are stored in a NumPy array, NumPy forces all of the four items in the list to conform to the same data type.  In the next code section, all four items are converted to type '<U32', which is a string data type in NumPy (the U refers Unicode strings; all strings in Python are Unicode by default).  In [3]:  np.array([1, -0.038, 'gear', True])  Out[3]:  array(['1', '-0.038', 'gear', 'True'], dtype='<U32')  NumPy arrays can also be two-dimensional, three-dimensional, or up to n-dimensional. In practice, computer resources limit array size. Remember that regardless of size, all elements in a NumPy array must be the same type. NumPy arrays are useful because mathematical operations can be run on an entire array simultaneously. If numbers are stored in a regular Python list and the list is multiplied by a scalar, the list extends and repeats- instead of multiplying each number in the list by the scalar.  The code below demonstrates list repetition using the multiplication operator, \*.  In [4]:  lst = [1, 2, 3, 4]  lst\*2  Out[4]:  [1, 2, 3, 4, 1, 2, 3, 4]  To multiply each element in a Python list by the number 2, a loop can be used:  In [5]:  lst = [1, 2, 3, 4]  for i, item in enumerate(lst):  lst[i] = lst[i]\*2  lst  Out[5]:  [2, 4, 6, 8]  The method above is relatively cumbersome and is also quite computationally expensive. An operation that is computationally expensive is an operation that takes a lot of processing time or storage resources like RAM and CPU bandwidth.  Another way to complete the same operation in the loop above is to use a NumPy array. Array Multiplication An entire NumPy array can be multiplied by a scalar in one step. The scalar multiplication operation below produces an array with each element multiplied by the scalar 2.  In [6]:  nparray = np.array([1,2,3,4])  2\*nparray  Out[6]:  array([2, 4, 6, 8])  If we have a very long list of numbers, we can compare the amount of time it takes each of the two computation methods above, a list with a loop compared to array multiplication to complete the same operation. This comparison highlights an advantage of arrays compared to lists- speed. Timing Arrays Jupyter notebooks have a nice built-in method to time how long a line of code takes to execute. In a Jupyter notebook, when a line starts with %timeit followed by code, the kernel runs the line of code multiple times and outputs an average of the time spent to execute the line of code.  We can use %timit to compare a mathematical operation on a Python list using a for loop to the same mathematical operation on a NumPy array.  In [7]:  lst = list(range(10000))  %timeit for i, item in enumerate(lst): lst[i] = lst[i]\*2  3.21 ms ± 958 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)  In [8]:  nparray = np.arange(0,10000,1)  %timeit 2\*nparray  7.11 µs ± 200 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)  With 10,000 integers, the Python list and for loop takes an average of single milliseconds, while the NumPy array completes the same operation in tens of microseconds. This is a speed increase of over 100x by using the NumPy array (1 millisecond = 1000 microseconds).  For larger lists of numbers, the speed increase using NumPy is considerable. Array Slicing Multiple values stored within an array can be accessed simultaneously with array slicing. To pull out a section or slice of an array, the colon operator : is used when calling the index. The general form is:  <slice> = <array>[start:stop]  Where <slice> is the slice or section of the array object <array>. The index of the slice is specified in [start:stop]. Remember Python counting starts at 0 and ends at n-1. The index [0:2] pulls the first two values out of an array. The index [1:3] pulls the second and third values out of an array.  An example of slicing the first two elements out of an array is below.  In [1]:  import numpy as np  a = np.array([2, 4, 6])  b = a[0:2]  print(b)  [2 4] Array Indexing Elements in NumPy arrays can be accessed by indexing. Indexing is an operation that pulls out a select set of values from an array. The index of a value in an array is that value's location within the array. There is a difference between the value and where the value is stored in an array.  An array with 3 values is created in the code section below.  In [1]:  import numpy as np  a = np.array([2,4,6])  print(a)  [2 4 6]  The array above contains three values: 2, 4 and 6. Each of these values has a different index.  **Remember counting in Python starts at 0 and ends at n-1.**  The value 2 has an index of 0. We could also say 2 is in location 0 of the array. The value 4 has an index of 1 and the value 6 has an index of 2. The table below shows the index (or location) of each value in the array.   | **Index (or location)** | **Value** | | --- | --- | | 0 | 2 | | 1 | 4 | | 2 | 6 |   Individual values stored in an array can be accessed with indexing.  The general form to index a NumPy array is below:  <value> = <array>[index]  Where <value> is the value stored in the array, <array> is the array object name and [index] specifies the index or location of that value.  In the array above, the value 6 is stored at index 2.  In [2]:  import numpy as np  a = np.array([2,4,6])  print(a)  value = a[2]  print(value)  [2 4 6]  6  Python.org downloads page showing download for Windows button | | | | |