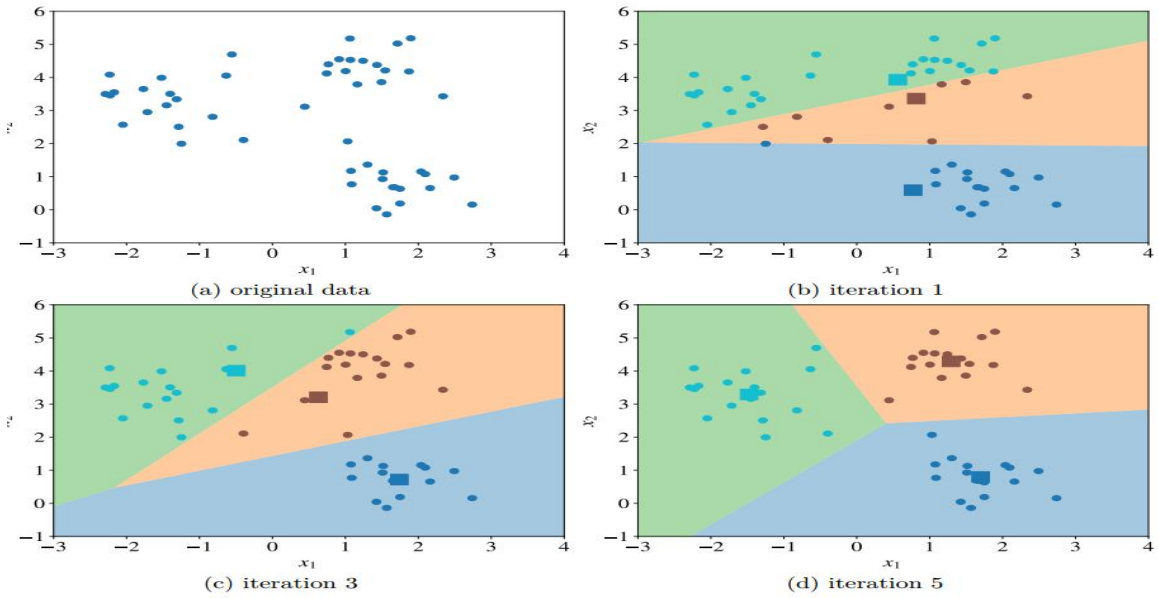
Introduction:

One of the main tasks of this report is to provide you with an objective analysis about the different Machine Learning Algorithms that can be integrated with your platform and add value to your final product. The report consists of two approaches that can be integrated and work at backend to provide with efficient ad recommendation. All the proposed algorithms have their advantages alongside disadvantages. Therefore, in order to reach to a conclusive stance, we first must understand their workings and then draw a comparison between all such algorithms.

Baseline:

However, before explaining all three algorithms, let me establish a baseline. Contextually, Baseline is a Machine Learning Algorithm called K-Mean. Baseline would be initiated at the very start of each algorithm irrespective of the choice. Before Explaining, why this Baseline is important, let me elaborate upon the working of the K-Mean Algorithm. The k-means clustering algorithm works as follows. First, the analyst must choose k—the number of classes (or clusters). Then we randomly put k-feature vectors, called centroids, to the feature space. We then compute the distance from each example x to each centroid using some metric, like the Euclidean distance.



Then we assign the closest centroid to each example (like if we labelled each example with a centroid id as the label). For each centroid, we calculate the average feature vector of the examples labelled with it. These average feature vectors become the new locations of the centroids. We recompute the distance from each example to each centroid, modify the assignment and repeat the procedure until the assignments don't change after the centroid locations were recomputed. The model is the list of assignments of centroids IDs to the examples.

Now, having understand how K-Means works, let's see why it is important and how it will be related to all two Algorithms (to be analysed). We will have user data for product and related features such as their geo-location data, social account data, streaming hours etc (all the explicit information being collected). We can use this explicit data to identify clusters of the users through K-Means. We will

give a higher weightage to geo-location features in the working of this algorithm, mainly it will have more influence in forming the clusters.

These clusters of users which will be mainly be decided by demographic features would enable to save computational power in all the two main algorithms. As now, before, starting our Main Algorithm, we restrict Main Algorithm working to one of the clusters of the user. For example, let's say USER-A belongs to cluster-one, then our main algorithm would only have to work on the cluster-one rather the whole dataset which save a lot of computational power. Even if we are training our algorithm on the whole dataset (don't worry-will explain later), we can restrict the core ad-recommendation to the relevant cluster, saving time. Moreover, this Baseline will also make our ads more relevant. As, we can optimise our Ad-Data Base according to these cluster. For example, Ads according to the locations, as clusters are formed mainly by demographic features but remember other features such as streaming hours also influence the cluster formation, but the influence will be less.

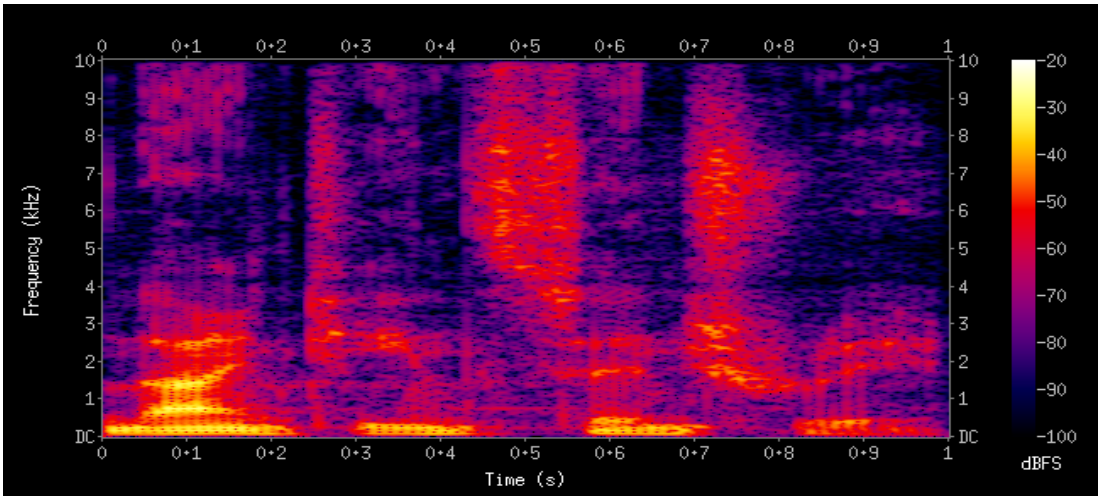
Conclusively, Baseline(K-Means) Algorithm is mainly working to save time and making ads more relevant by identifying the different clusters of the user and will be initiated before the Main Algorithm.

Main Proposed Algorithms:

1. Spectrogram Similarity Algorithm:

Now first let me explain the working and design of this algorithm. The Algorithm main dependency is on the Audio that user streams. Remember, this algorithm would only make ad-recommendation by optimizing the audio that user streams because other features have already been utilized by our Baseline Algorithm. So, at the start of this algorithm Baseline Algorithm would make clusters. **This mean Spectrogram Similarity Algorithm would restrict its working to the relevant cluster and would simply ignore other clusters making it faster.** However, we do not require all the volumes of audio that user listens to because it is not only inefficient but also infeasible. Therefore, for each user, we can optimise our Database to save 30mins of audio clips at random moment whenever user is streaming. For each user five 30 mins audio clips would be enough. What's important is to keep updating these samples in order to make the algorithm more relevant. Please note, I was not aware about the capacity of your Database therefore, I am only recommending saving five-30 mins audio clip for each user at specific instance, these number could be optimised according to your Database.

Now, we will have a Dataset, where for each user, there will be five audio Clips. The Next step is constructing Spectrogram which is perhaps the crux of this algorithm. In very simple terms, "A **spectrogram** is a visual representation of the spectrum of frequencies of a signal as it varies with time(Wikipedia)." Basically, for each audio clip saved, we would make a relevant spectrogram which visually looks like this:

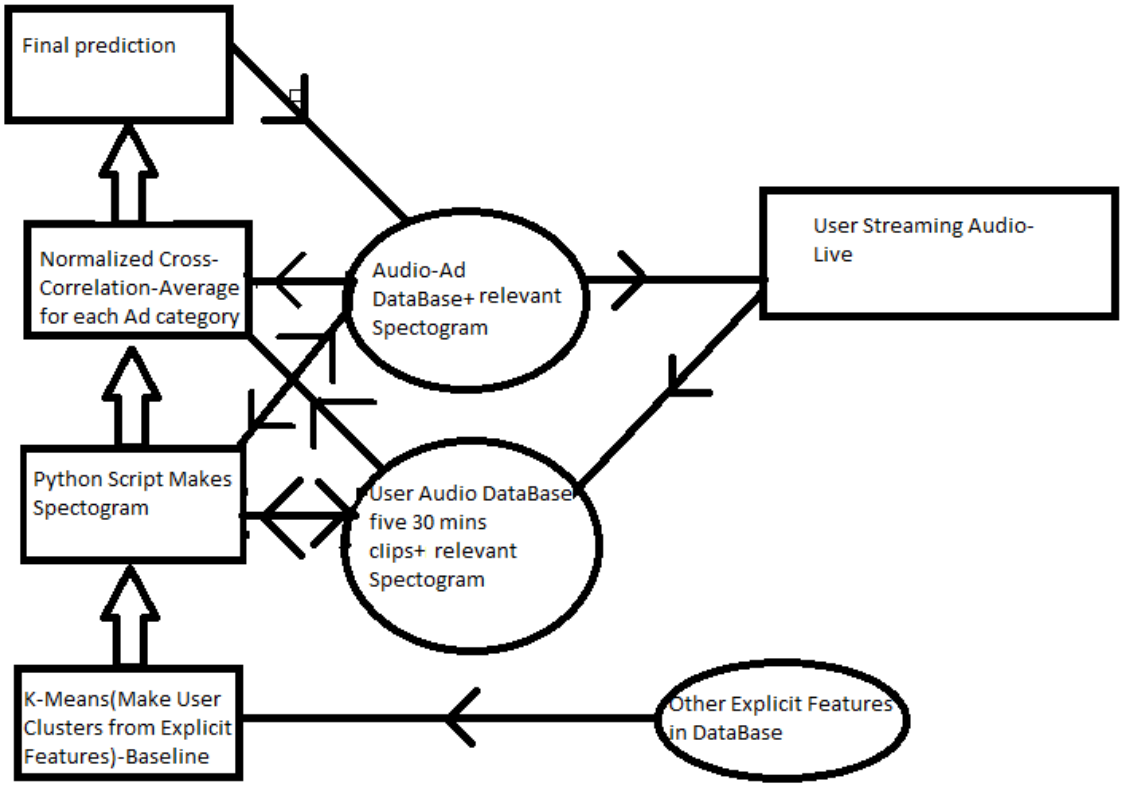


Instead of dwelling into mathematical details of Spectrogram, let me just assure you that it is one of the most efficient way to visually capture the audio pattern. These spectrograms can be constructed in a dynamic way by making use of python library called "**Librosa**". Now, for each user there will be five spectrogram and we will be reconstructing these as we take new samples and discard the old audio samples. Now, on the opposite end we will have a Ad-Database. This means that we will have an Audio Database of Ads with relevant categories. I am right now not restricting these categories as this depends very much on you. However, just for analysis, let say you have Ad-Database (with two categories-Sports and Lifestyle). We will construct Spectrograms for each of this Audio Ads. Now, the problem is that Ads are meant to be small like almost 1-2 mins long, but User audio is 30 mins long, so the comparison is not on equal basis. Therefore, what we can do is append Relevant Ad-Audio in the same category. For example, All Audio Ad in sports category can be appended into one audio file to make it almost 20-30 mins long. And Don't worry, if your database does not have enough Ads, we can crawl Audio Ads from internet for each category and make the desire Audio file long enough. We will save these spectrograms under the relevant Ad category

Now, the main part is final recommendation. The Algorithm would compare each of the spectrogram for user with the Ad-Spectrograms (for each category). This comparison would be done on something called," Normalized Cross Correlation". In simple terms, it measures the correlation (pattern similarity) between two images after normalizing the values and give a final value between 0 and 1. Higher the value, the more similar are the Images.

So, we will compare all five-user spectrogram with all the Ad-Audio Spectrograms and give a value between 0 and 1 for each comparison. The Ad-Category with highest score on average (average because remember there are five spectrograms for each user, and we will do comparison for each one of them with each Ad-Category and then calculate mean (where n=5) for each category). I know this sounds little confusing but in simple terms, we will have score of 0 and 1 for each category for each user. The category with highest score would be the relevant category and this category would be relevant for the user. Now, you can take a Ad-Audio (original file not appended or combined) from the selected category and make user listen to this ad.

The Visual below summaries the working of this algorithm:



Now Just let's just have a quick summary of pros and cons of this algorithm which itself are quite understandable. One of the main edges of this algorithm is that it tends to capture the context in real-time fashion. Elaboratively, this tends to continuously update users' preferences and in turn optimize the ad recommendation. Further, the algorithm is simple to understand, and the implementation is also not complex. Moreover, it avoids the complexity of training. As you will see that the next approach which involves of neural network, will require training before deployment but this approach does not work on this principle. Therefore, the deployment could be quick and fast.

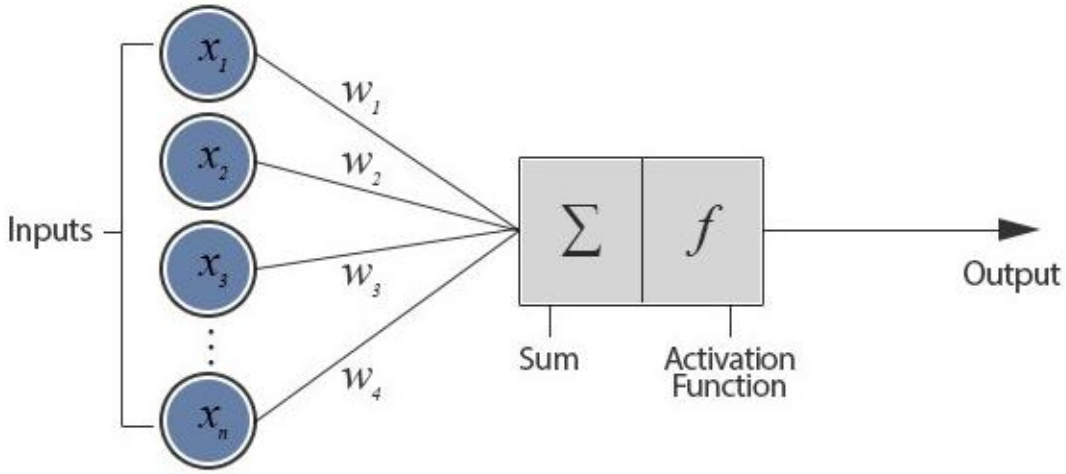
However, one of the main drawbacks of this approach is that it requires a lot of computational resources in terms of storage. Further, the large dataset for Audio for the user could possibly be a bottleneck for our algorithm if our database is not optimized. Similar is the case for the storage of spectrograms. Therefore, the optimization and efficiency of database and storage is a must for the working of this algorithm which could prove too expensive.

Testing/Validation:

One of the easiest methods to validate this approach is to directly ask feedback from user about the ad recommendation. If majority of the user are positive about their feedback, then our algorithm does not require optimization or else vice versa. Further, we can also create spectrogram of random audio related to specific content from internet and run through this algorithm, if the algorithm assign high similarity to the category which is similar to Audio content then we can be assured than our algorithm is working fine or else it requires further tuning.

2. Ad-Recommendation Neural Network:

The last four years have seen a large surge in use of neural network in every field. Therefore, it is no surprise that neural network would not prove effective in optimizing the Ad-Experience for our users. Before dwelling into the particulars of this approach, let me give you a very high understanding of Neural Network (without any mathematics).



Model of an artificial neuron

The figure depicts a neuron connected with *n* other neurons and thus receives *n* inputs (x1, x2, ….. xn). This configuration is called a **Perceptron.** The inputs (x1, x2, …. xn) and weights (w1, w2, …. wn) are real numbers and can be positive or negative. The perceptron consists of weights, summation processor and an activation function. All the inputs are individually weighted, added together and passed into the activation function. There are many different types of activation function but one of the simplest would be step function. A step function will typically output a 1 if the input is higher than a certain threshold, otherwise its output will be 0.

$$x1w1 + x2w2 = (0.6 \times 0.5) + (1 \times 0.8) = 1.1$$

Here, the total input is higher than the threshold and thus the neuron fires.

I wanted to give you this high understanding because without this, the algorithm is impossible to comprehend.

Now, the first step in this approach is similar i.e. run the baseline K-means algorithm to make clusters of users. However, this time the User Audio Data is not crucial because we are going to train our neural network on Ad Audio. Let me elaborate, we will first form n categories for our Ads. The categories could be as many as you want and pretty much depends on your requirement. These

categories would form the labels for our training Data. Clearly, the features for these labels would be the Ad Audio but here's the main optimization or trick of this algorithm. We would convert this Audio into Bag of words through speech recognition algorithm. Elaboratively, for each Ad, we could convert it into a list of key words.

Having form the bag of word for each Ad with its relevant category as feature. We can now make efficient use of Natural Language Processing to pre-process the data and feed it into neural network. Again, just a high-level understanding of NLP pre-processing, we could either make use of count vectorize or td-idf(see-below) to convert our features into numerical values.

- **Count Vectorizer**: The most straightforward one, it counts the number of times a token shows up in the document and uses this value as its weight.

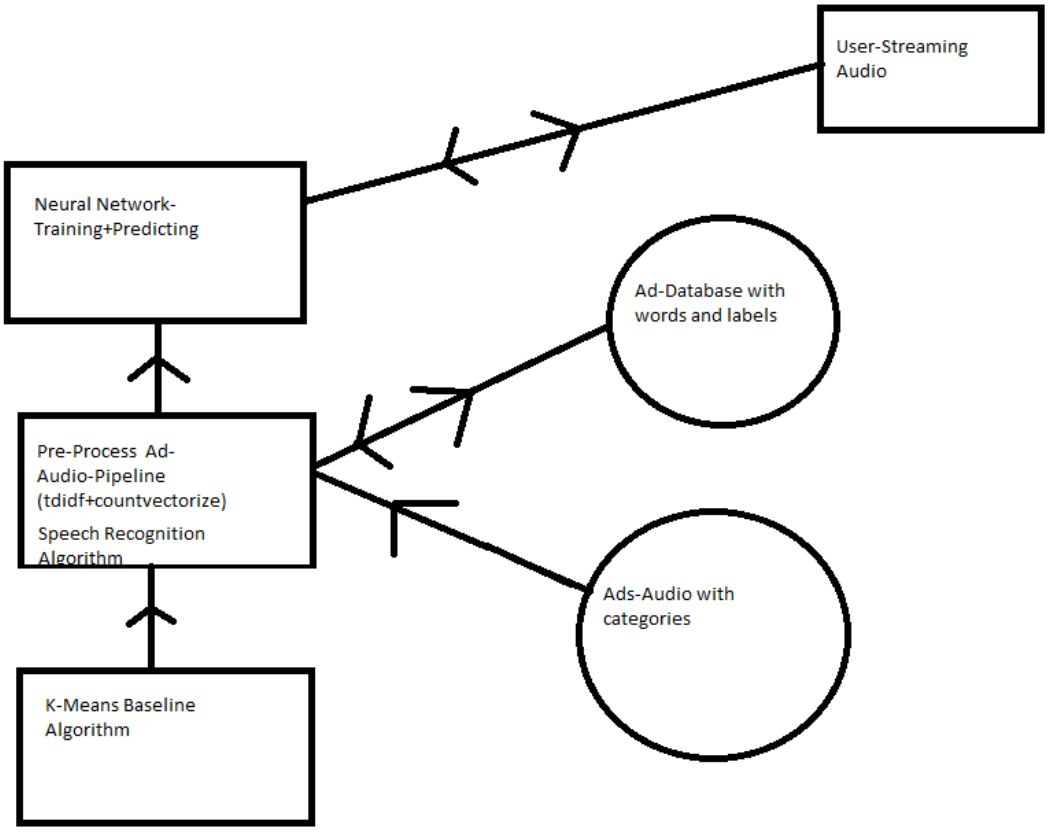
- **TF-IDF Vectorizer**: TF-IDF stands for "term frequency-inverse document frequency", meaning the weight assigned to each token not only depends on its frequency in a document but also how recurrent that term is in the entire corpora.

Now our features would be fixed array consisting of numerical values with the Ad category, as the relevant label. However, of the main demanding task of Neural Network is its requirement of large data. If we have a limited amount of Ad-Audio that is to be displayed to user. We could further train our neural network by taking Ads from internet and using it only for training our purposes.

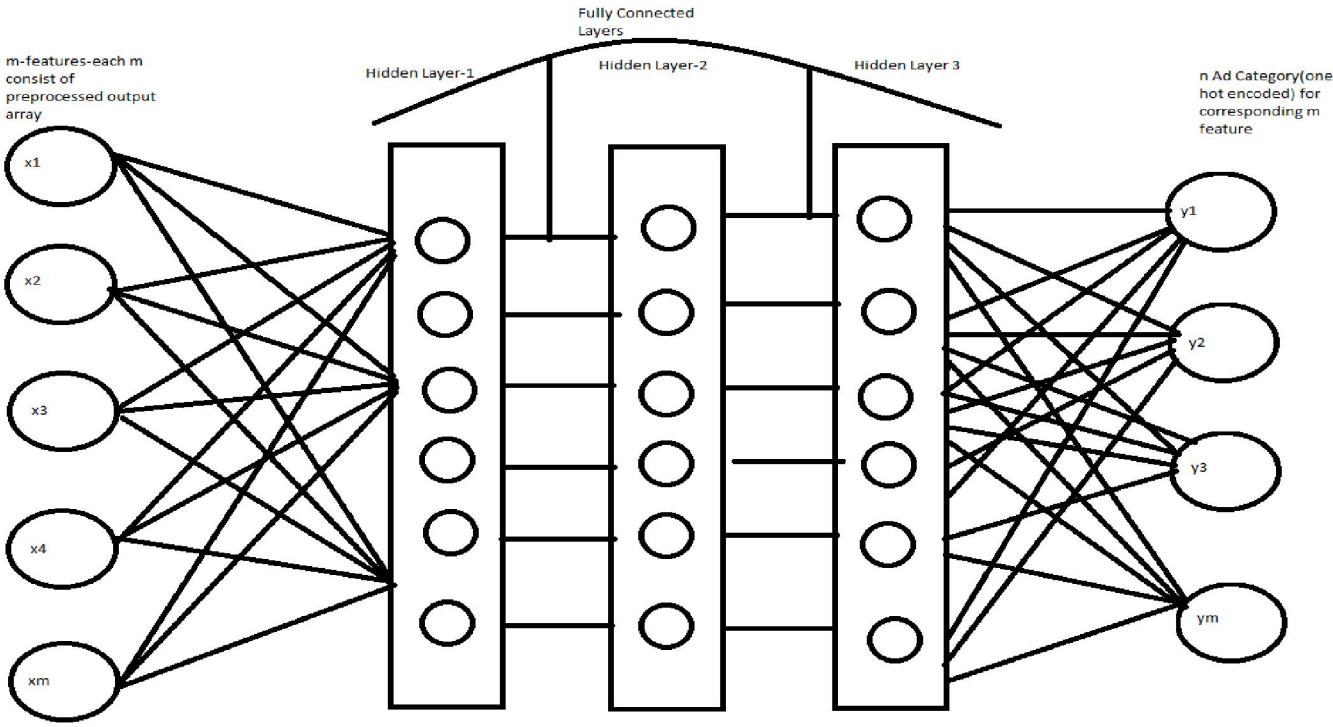
Our Neural Network would now be trained. The next step would be that we would take user audio, reprocessing it through the same pipeline and feed it into the trained neural network. The Neural Network would give probability to each category of Ad (Remember these are the labels we trained on) and the ad with highest probability would be recommend to User (Softmax function-don't worry it's a mathematical term). Now, the best part is that you do not need to store huge volume of User Audio Data. Once, the Neural Net is deployed, you could in real time pre-process User audio at specific interval and recommend Ads to user. Over time, you could store some Audio for user and make Neural Net learn from it but it is not necessary. However, to make this neural network work in real-time, data efficiency is a must. Further, at specific interval you could retrain your neural network over any new Ads you might have been given or collected or you could even add more categories (which could take some time but is easy to train).

Now, remember like spectrogram similarity algorithm we did make use of baseline algorithm remember our neural network is trained on entire Audio Ad Dataset (not for specific cluster), however, once our neural network predicts specific Ad Category for User, we could see to which cluster does this User belongs to and show to the User the same Ad for the same category but optimized for the cluster. For example, if Neural Net Predict Sport for two User but one user is in a cluster of Location A and other is cluster of Location B. We could both show them sport ad abut make them relevant to the demographic features identified by the baseline algorithm.

I know this analysis is not easy to comprehend so let me visually show the workflow of the algorithm:



As for the Main Neural Network Architecture, it will be as follow:



Now having understand the working of the algorithm lets discuss the pros and cons of this algorithm. Firstly, this algorithm is fast and more efficient with respect to the computational resources. As once neural network is trained, it can give ad-recommendation in real time by making using use of short-term memory rather than being reliant on long-term storage like the previous algorithm. Moreover, I prefer this algorithm because, it is cleaner. Elaboratively, this means, the algorithm does not require complex visualisation but simple bag of words, which are easier to understand and decode into numerical data. Further, Neural Net are more state of the architecture and therefore are more preferred over other approaches. Further, one of the, main reason why I believe this approach is better because it learns from the Ads which makes the recommendation more relevant, as the final product also must be the Ads. Even though this sounds confusing but in simple terms, we are learning from what we are going to deliver therefore, this approach sound more appropriate given the task.

However, like any other neural network, one of the major drawbacks of this neural network is its requirement of training. You see even though we might not need much of User Audio Clip, but we need to develop a huge dataset of Ad Audio with appropriate categories as label. This requires both time and computational resources. Moreover, periodic retraining will also has to be done which again is a constraint on time. Moreover, we also might need to crawl internet to capture Ad Audio for training purpose which is another task that requires time. Therefore, in summary one of the main disadvantages associated with neural network is amount of time require to train it and keep updating it at periodic interval.

Testing/Validation:

One of them main easy way to validate this neural net is to conduct a user survey. This mean why our neural network predict an ad to the user, we could ask for their feedback, as whether this ad was relevant to them. If majority of user are in affirmation than neural network is working effectively but if not, then it requires further optimization. Another way to validate is to create dummy user audio data. This mean you pick random content audio from internet and understand what type of content the audio type is about. You pick several examples like this and feed to the neural network. The Neural network will predict the ad category and if this category is in accordance with the content type, your network is optimized or else not. There are many other ways to determine the efficiency of neural like keep creating train/validate/test sets, A/B testing but I have mentioned which according to me will be most efficient given your product.

Conclusion:

Even though, final decision is at your discretion, but I would like to subjectively give my advice. Given both algorithms, I believe that both can easily be integrated with your product but given that most companies are transiting to deep learning rather than classical machine learning algorithm, I would suggest you implement Ad-recommendation neural network. Even though, as a Machine Learning Engineer, I cannot be completely assured which algorithm will perform better but, in my experience, neural network tends to perform better. However, in order to form an informed choice, further testing and implementing a protype is needed given the actual data.