November 2019

Embedding Explorer

Anonymous Anonymous

Embedding Explorer

Abstract

The present disclosure describes an embedding explorer that allows a user to interactively explore properties of an embedding space and how the embedding space relates to features of entities being embedded. The embedding space is a low-dimensional space onto which the embedding explorer translates high-dimensional vectors to low-dimensional vectors. In particular, the embedding explorer allows the user to load a table of embeddings and feature providers that are required by the user during the exploration. The table includes at least one column with an entity ID and another column with an array of floats representing the embedding for the entity. The user may further add a new embedding to the embedding space. To add the new embedding to the embedding space, the user provides hive tables or CSV files as input to a preprocessing workflow of the embedding explorer. The preprocessing workflow utilizes a t-distributed stochastic neighbor embedding (t-SNE) to scale down tens of millions of points representing the entities. After the preprocessing workflow is completed, it is required to manually register the embedding by adding an element to an “EmbeddingExplorationSources” function of the embedding explorer. An “allDataRootFilepath” field of the “EmbeddingExplorationSources” function accepts an output folder returned by the preprocessing workflow.

Problem statement

For exploring high-dimensional vectors describing many entities, dimensionality reduction needs to be performed to translate the high-dimensional vectors onto the low-dimensional space. In prior art, a principal component analysis (PCA) technique has been used to perform the dimensionality reduction for translating the high-dimensional vectors. However, the PCA fails to capture non-linear structures in the features when data has a multitude of dimensions. Thus, a dimension-reduced embedding is not a faithful representation of its local neighborhood in a high-dimensional space. Consequently, the PCA fails to preserve local distances among the high-dimensional vectors, while mapping to the low-dimensional vectors.

The present disclosure proposes a novel solution to overcome the stated problem.

System and working

The present disclosure provides an embedding explorer, which further includes a user interface (UI) allowing a user to explore embeddings. Specifically, the UI allows a user to interactively explore properties of an embedding space and how the embedding space relates to features of entities being embedded. The embedding space is a low-dimensional space onto which the embedding explorer translates high-dimensional vectors to low-dimensional vectors. Specifically, the embedding explorer allows the user to load a table of embeddings and feature providers that are required by the user during the exploration. The feature providers include, but not limited to, laser tiers, MySQL tables. The table includes at least one column with an entity ID and another column with an array of floats representing the embedding for the entity. The embedding enables the abstraction of specifics about the entity that it represents. Also, the embeddings may refer to multiple entity types. If the embeddings refer to the multiple entity types, the table includes a column with the entity type for each row. The user may check out a feature map in a configerator to see the entity types for which the features are registered. Thus, if the embeddings in the table refer to the standard entities, such as user, pages, group etc., the user should use right naming conventions for the entity type in accordance with the feature map so that the user accesses the features for the entities directly. To achieve this, the user may map names of the entity types in the table to standard entity type names through a flow configuration. The user is required to provide an entity type mapping parameter that maps the names of the entity types in the table to the standard entity type names.

The user may further add a new embedding to the embedding space. Steps to add a new embedding and to make changes therein are explained in subsequent sections:

Preprocessing

A preprocessing workflow of the embedding explorer accepts both hive tables and CSV files as input. The hive tables should have a column with the entity ID, and another column with the embeddings stored as an array. The CSV files should have the first column with the entity ID, and the remaining columns to include values of the embedding for different dimensions. Further, the CSV files should not have any header. The preprocessing workflow requires entity IDs to be numerical. If the entity IDs are not numerical (e.g. string IDs), the user sets ‘Map IDs’ to true in the preprocessing workflow to create a dictionary that maps the string IDs to a data type called long. The user sees both the original entity ID and the mapped

long in the UI as shown in Figure 1. The user also provides a value for the entity type mapping parameter along with the input.

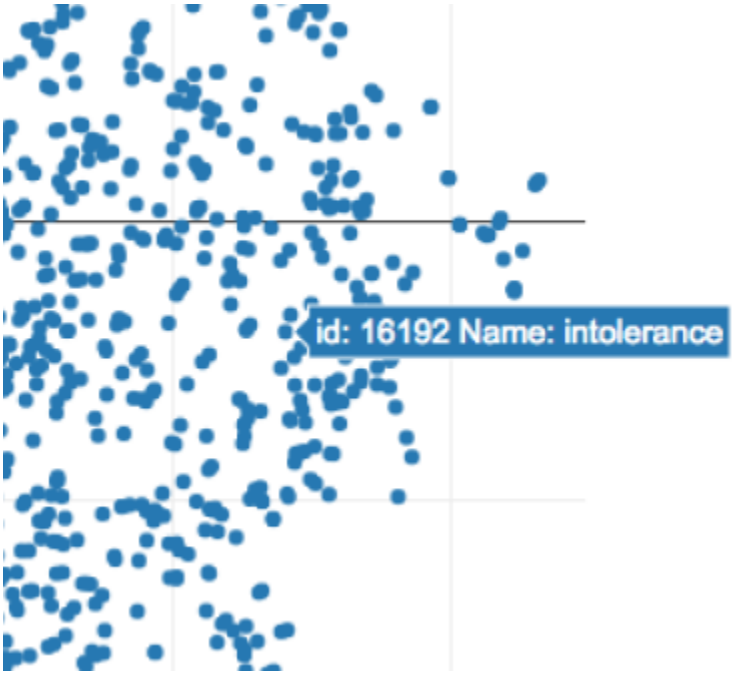


Figure 1: mapping the strings IDs to long

Once the input is ready, the user runs the preprocessing workflow that creates required data structures for the embedding explorer to operate on. The user may change experiment name and dataset fields to match his/her use case in the preprocessing workflow. The experiment name is used to construct an output folder where results of the preprocessing workflow are written to.

In order to scale down tens of millions of points representing the entities, an inductive approximate to t-distributed stochastic neighbor embedding (t-SNE) is used. The t-SNE is a nonlinear dimensionality reduction technique for embedding the high-dimensional vectors for visualization in the low-dimensional space of two or three dimensions. Due to demanding memory costs, it is impossible to run the t-SNE on large datasets. To overcome this, the t-SNE is first run on a reasonably small subset of the embeddings, S (of size 'num_training_samples'). Then, for each vector v not in S, a distance weighted average of k-nearest neighbors of v in S from 2-dimensional vectors resulting from the initial run of the t-SNE is taken. Resulting vector includes distance weighted averages calculated for all vectors not in S. It is recommended to select the subset of embeddings of size up to 1 million. In order to finish the preprocessing workflow quickly, the 'num_training_samples' equal to 100k usually suffices, but projection quality increases with a higher value of the 'num_training_samples'.

Registering the embedding

After the preprocessing workflow is completed, it is required to manually register the embedding in the configerator.

In one example, the user has a new video embedding. In order to register the new video embedding, the user adds an element to "EmbeddingExplorationSources" for the "VIDEO" entity (as shown below for "My New Test Embedding").

```
u 'VIDEO' : EmbeddingExplorationSources(
 embeddings = [
      EmbeddingCollection(
         name = u 'NewsFeed_5M_test', //the name field is used to refer to this embedding in the UI
         dataProvider = None,
         allDataRootFilepath = u '…', //fill in the allDataRootFilepath field with the output folder returned by the preprocessing workflow
         servedPartitions =None),
      EmbeddingCollection(
         name = u 'My New Test Embedding', //the name field is used to refer to this embedding in the UI
         dataProvider = None,
         allDataRootFilepath = u '…', //fill in the allDataRootFilepath field with the output folder returned by the preprocessing workflow
         servedPartitions = None)
    ],
 features = [FeatureCollection(
     . . . .
```

This way, the user successfully registers the embedding in the configerator. The user may further explore following options with the registered embedding:

Adding new features for an entity type

The user adds new features for an entity type by registering a new feature provider for that entity type in the configerator. The user specifies how to access the new feature provider and some metadata about the features.

Testing changes

A config file is processed by a backend of the embedding explorer. A backend python server includes codes to handle API calls such as, retrieving points, or getting feature values and names. The config file is centered around the notion of the entities that have associated embeddings and features, and the entities are used as an abstraction to bring both the embeddings and features together. The config file includes parameters and settings representing the changes done by the user. The safest way to test his/her

changes for the user is to have a local version of the backend running on a sandbox. The sandbox is a testing environment that isolates untested code changes and outright experimentation from a production environment or repository. The user sends the config file to the sandbox from a configerator UI and then point the embedding explorer to the local version of the backend. Alternatively, if the user is confident of what he/she is doing, he/she may directly test the config file in an embedding explorer tier, which will affect the production environment.

Making and testing local changes

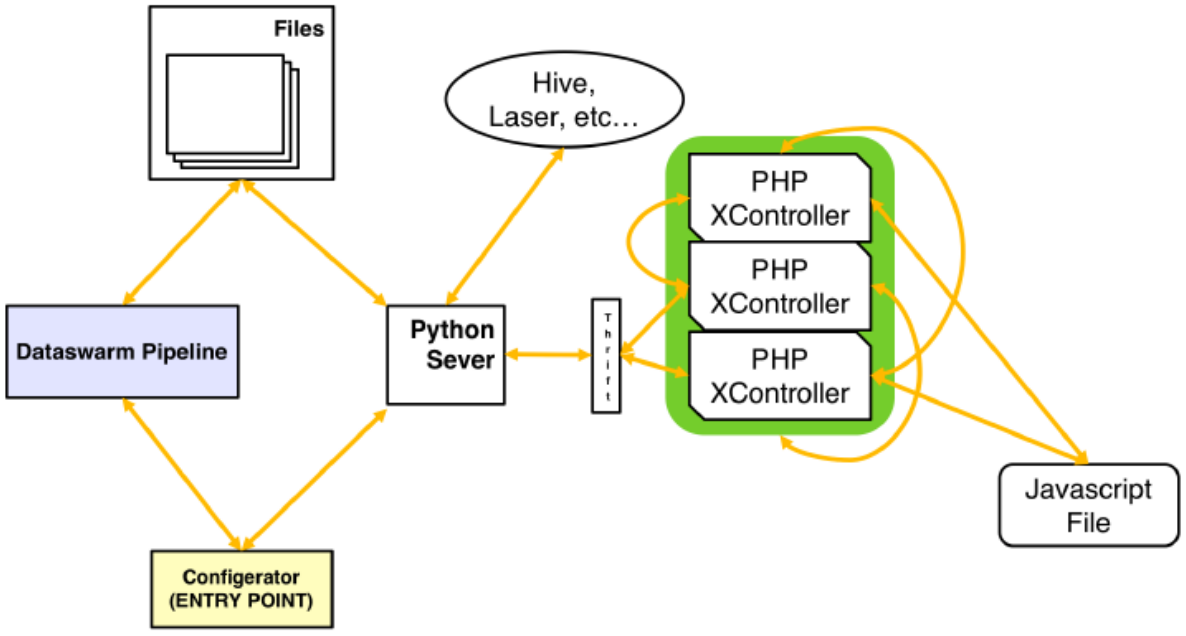


Figure 2: System architecture of the embedding explorer

Figure 2 illustrates a system architecture of the embedding explorer. A frontend of the embedding explorer includes various PHP XControllers and React code for UI interactions. The user edits python files and thrift files according to his/her needs, and registers changes in the frontend. The frontend utilizes a plotly library for interactive plotting. The plotly library resides in a third-party tool.

Additionally, the user may utilize the UI of the embedding explorer to interactively explore functions available in the embedding explorer. These functions are described in following section “Exploring the operating environment”. The user may explore some or all the functions in any order as desired, not necessarily in the order as described here.

Exploring the operating environment

As a first step, the user selects the entity corresponding to the embedding using a leftmost selector box (shown in Figure 3) underneath a visualization plot. Once the user selects the entity, the user selects the embedding using a middle box (shown in the Figure 3). Finally, the user presses an explore button to load the data.

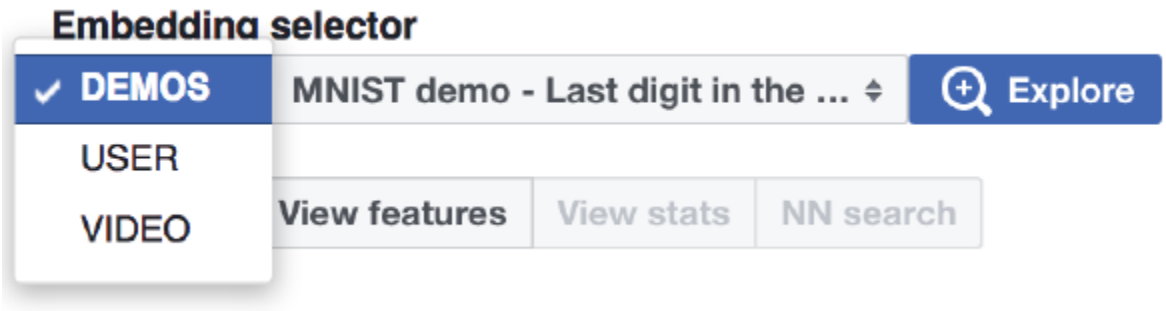


Figure 3: selecting the embedding

Once the embedding is selected, the user may explore following functions available in the embedding explorer:

Adaptive zoom

The embedding explorer is designed to work with the embeddings of a large number of entities (tens of millions). So, it is practically impossible to plot all these points on an explorer window at the same time. Therefore, an adaptive zoom is utilized to show a reasonable number of points (a few thousands) at a point of time. As the user moves the embedding around or zoom in and out, the backend sends a new list of points to fill in the explorer window. In order to create a seamless and low latency experience for the user, while interacting with the embedding explorer, a Faiss index is used to store the points at every zoom level. The Faiss indexes encode the vectors into codes of a fixed size and store them in an array.

View features

The user selects a region of the embedding space in the visualization plot using one of tools (square, free-form) as shown in Figure 4. The user then clicks a “View Features” button to display a table (shown in Figure 5) with values of the available features for points in a selected region.

Figure 4: Selecting a region of the embedding space to view the features

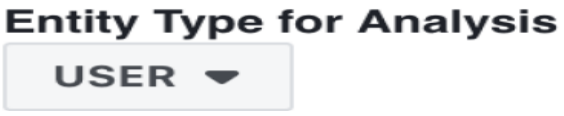
Feature table

FBID	country	locale	age	lifestage_prediction	gender	friend_count	facebookage	is_employee	l1	l7	l28	web_l1	mobile_l1	web_l7	mobile_l7	web_l28	mobile_l28	lastactiontime	region_from_ip	country_from_ip	data
595707187	TR	tr_TR	31	late_work	1	97	3199	1	1	7	28	0	1	0	7	0	28	2018-01-09	-	-	-
605994136	TR	tr_TR	49	late_work	2	475	3423	1	1	6	27	0	1	0	6	0	27	2018-01-08	-	-	-
618948253	TR	tr_TR	26	early_work	2	122	3592	1	1	7	28	0	1	0	7	0	28	2018-01-09	-	-	-
622069708	TR	tr_TR	44	late_work	2	463	3421	1	1	5	18	0	1	0	5	0	18	2018-01-09	-	-	-
665299516	TR	tr_TR	37	late_work	1	704	3711	1	1	7	26	1	1	4	7	17	26	2018-01-09	-	-	-
667044729	TR	tr_TR	42	late_work	2	289	3302	1	1	7	28	0	1	0	7	0	28	2018-01-09	-	-	-
710882114	TR	tr_TR	33	late_work	1	514	3722	1	1	7	28	0	1	0	7	0	28	2018-01-09	-	-	-
1190255031	TR	tr_TR	48	late_work	2	973	3291	1	1	7	28	0	1	0	7	0	28	2018-01-09	-	-	-
1660384673	TR	tr_TR	36	late_work	2	305	3252	1	1	6	24	0	1	0	6	0	24	2018-01-09	-	-	-

Figure 5: The table with values of the available features for the selected points

Histogram comparison

The embedding explorer further has capabilities to display histograms of feature values for selected regions of points. The user first selects the entity type that he/she wishes to select in an “Entity Type for Analysis” dropdown shown below.



The user then selects a “Histogram Comparison” mode (shown in Figure 6) and selects a group of points in the same way as explained earlier. In one example, the user selects a Group 0 and a Group 1 from the embedding space. The user further clicks on an “Overlay” tab within the “Histogram Comparison” mode. Upon clicking the overlay tab, an overlay plot is generated that displays the histograms for both the Group 0 and the Group 1.

Figure 6: Histogram comparison

However, if the user selects multiple groups in the “Histogram Comparison” mode, a feature salience ranking is computed and displayed in a “Salience” tab, as shown in Figure 7.

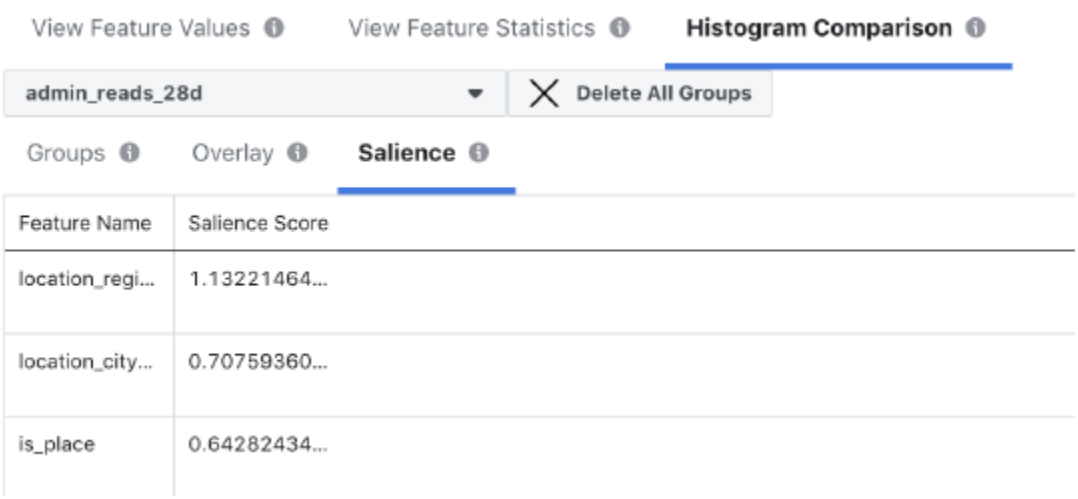


Figure 7: The feature salience ranking

Hover

Whenever the user hovers the mouse over a point in the visualization plot, a small window describing the entity that the point represents pops up. The user can select the features that he/she wants to be included in the pop-up window using a "Feature Hover" multiple selection drop-box. Figure 8 shows the pop-window including the features selected by the user.

Figure 8: Feature hover

Feature Providers

By unchecking a feature provider from a “Feature providers” checkbox (shown beneath the visualization plot in the Figure 8), the features from that feature provider are ignored. This reduces latency of feature retrieval operations. Also, the feature table is in a better readable form.

Filtering

In order to perform filtering, the features are exposed inside a filtering engine as a map called feats. In one example, the user wants to filter out users younger than 20 years old, he/she writes a command in an input box: *int (feats[‘age’]) > 20*

It is possible to use logical operators and string functions as well:

int (feats [‘age’]) >20 and feats[‘country’] == ‘US’

feats[‘locale’].startswith(‘en_’)

The user clears up the filter by deleting the text in the input box and then by clicking apply.

Entity Lookups

An entity lookup section allows the user to find the entities by the numerical entity ID or by the string ID (if the user maps the string IDs to the numerical entity IDs in the preprocessing workflow). The user gets the numerical entity ID of the point currently in the visualization plot by clicking on that point, and its numerical entity ID will be copied to clipboard. The entities need not be visible in the current visualization plot, the lookups happen in full data within the embedding explorer. The lookups are cumulative, which means that they are added to an internal list of the selected points and are displayed simultaneously in the current visualization plot. The user presses a “Clear extra points” button to clear up all the current lookups. The user further sees the available features for the entities returned by the latest lookup by accessing the “View features” mode.

Go to entity

To draw a marker on a specific entity, the user needs to type its entity ID and click on a “Go to” button, as shown in Figure 9.

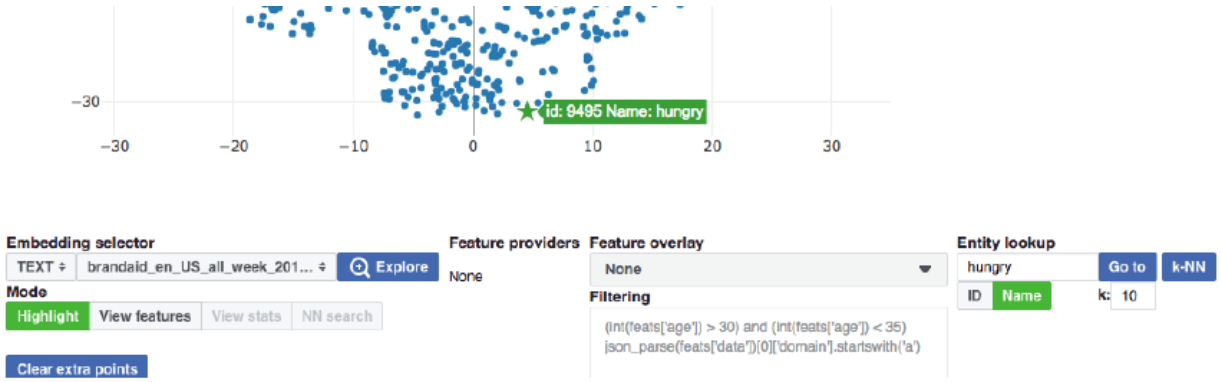


Figure 9: “Go to entity” mode

Nearest neighbors

The embedding explorer finds nearest neighbors of a point in the original embedding space. The user enters the entity ID/name of a seed point in an NN lookup box. The user then selects the number of neighbors and click on a “k-NN” button. The nearest neighbors are showed in a list and also overlaid on the visualization plot using red star markers as shown in Figure 10. The user may further get the features for the nearest neighbors by clicking on the “View Features” button.

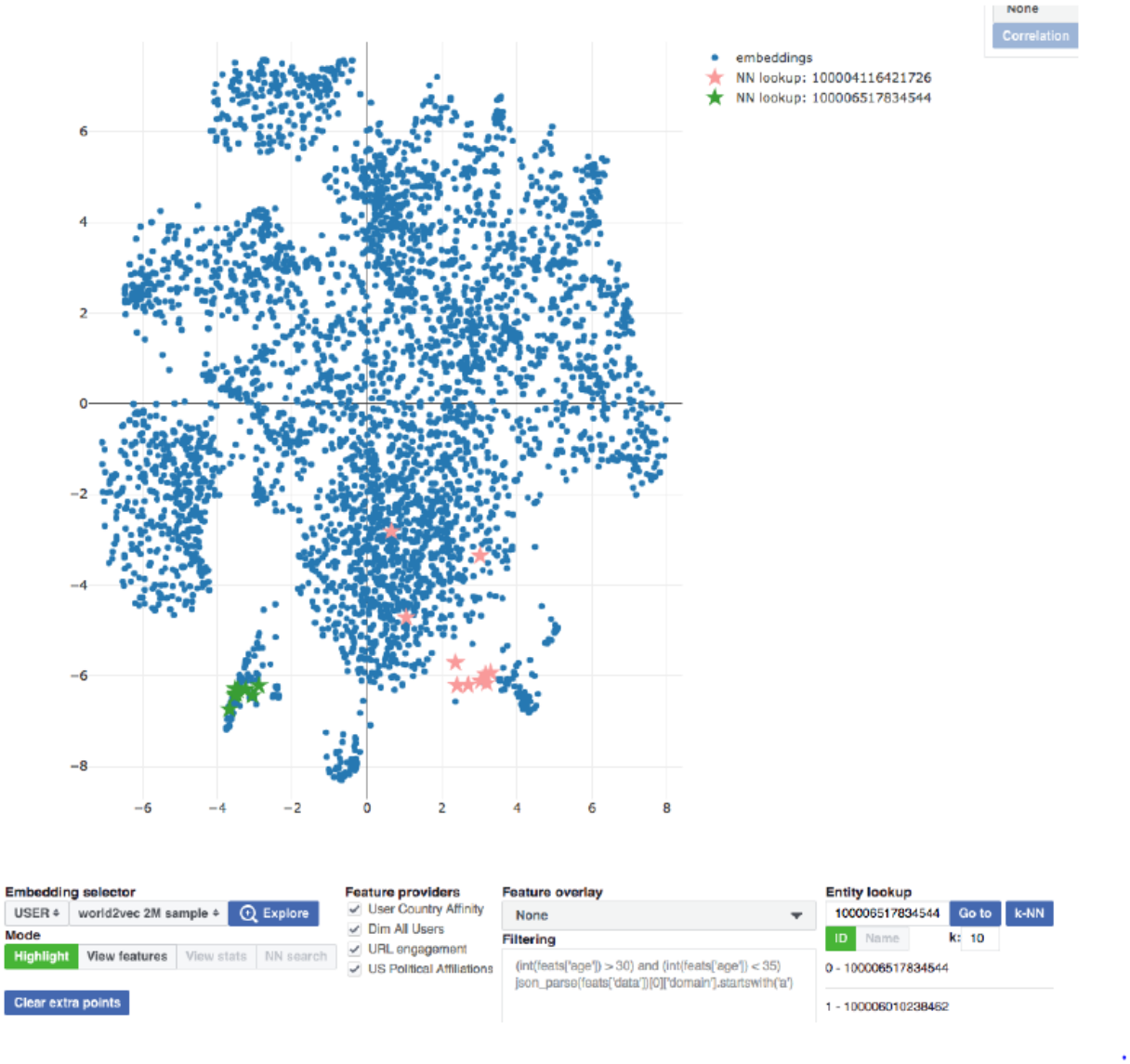


Figure 10: Finding nearest neighbors

Embedding-feature information measure

Embedding-feature information measure is utilized to measure the amount of information about a given feature that is included in the embedding. The present disclosure supports “correlation” as the embedding-feature information measure.

Correlation: The user selects a cluster of points and the embedding explorer shows the features to the user that are correlated with the cluster of points selected by the user. The embedding also shows the features that take on significantly different values for the points inside and outside the cluster. For discrete features, an F-statistic of an analysis of variance (ANOVA) test is calculated. The ANOVA test is a statistical

technique that is used to check if means of two or more clusters are significantly different from each other. The ANOVA test checks impact of one or more features on the varying means of the clusters by comparing means of different samples of the clusters. For continuous features, a Pearson's correlation coefficient between the feature values and individual dimensions of the embedding space is calculated. The Pearson correlation coefficient is a measure of linear correlation between two continuous variables.

Conclusion

Embedding is a highly applicable concept that can significantly improve machine learning on structured data. With advancement in machine learning, researchers have been able to do tremendous tasks ranging from image recognition, natural language processing, medical diagnosis and more. With the widespread adoption of ML systems, it is becoming extremely important for the researchers to be able to explore how the data is being interpreted by the models. However, the primary challenge in exploring this data is that it has a multitude of dimensions. Thus, it is a cumbersome job to analyze and explore the high-dimensional data, and it requires sophisticated tools to investigate the space. The present disclosure provides an embedding explorer for interactive visualization and analysis of the high-dimensional data.