

# DimensionalityReduction

September 11, 2018

## 1 Solutions to Dimensionality Reduction

## 2 Dimensionality Reduction Task

- Use PCA from [MultivariateStats.jl](#), to reduce 100 dimensional word embedding down to 3,2 and 1 dimentionions.
- Plot these using [Plots.jl](#), coloring according to class

### 2.1 Tips:

- plotly is a good backend for 3D Plotting.
- The command `scatter(xs[1,:], xs[2,:], xs[3,:]; hover=all_words, zcolor=classes)`
- will plot a 3D scatter plot
- coloring each point according to the numerical array classes
- and putting a tooltip on each point, according to the string array all\_words

## 3 First we loadup some data

For the the example presented here, we will use a subset of some pretrained word2vec word embedding, using the [Embeddings.jl](#) package. These are 300 dimensional vectors, which encode syntactic and semantic information about words.

Example code for the loading, together with the words sorted into their original classes is below.

In [48]: `using Embeddings`

```
countries = ["Afghanistan", "Algeria", "Angola", "Arabia", "Argentina", "Australia", "B  
usa_cities = ["Albuquerque", "Atlanta", "Austin", "Baltimore", "Boston", "Charlotte", "B  
world_capitals = ["Accra", "Algiers", "Amman", "Ankara", "Antananarivo", "Athens", "B  
animals = ["alpaca", "camel", "cattle", "dog", "dove", "duck", "ferret", "goldfish", "goose", "B  
sports = ["archery", "badminton", "basketball", "boxing", "cycling", "diving", "equestrian"]  
  
words_by_class = [countries, usa_cities, world_capitals, animals, sports]  
all_words = reduce(vcat, words_by_class)  
embedding_table = load_embeddings(Word2Vec; keep_words = all_words)  
@assert Set(all_words) == Set(embedding_table.vocab)
```

```

embeddings = embedding_table.embeddings
all_words = embedding_table.vocab
classes = map(all_words) do word
    findfirst(col -> word col, [countries, usa_cities, world_capitals, animals, sports])
end;

```

## 4 Extension: T-SNE

- Use [TSne.jl](#), to perform similar dimensionality reduction, and to produce plots.

T-SNE is another popular DR method.

Be warned: it is sideways -- it is row major, so transpose the inputs and outputs

You may have to play with the perplexity to get it to work well.

If you look at the resulting plots, you may note that countries are often paired up with their capital city.