PowerShell + DevOps
Global Summit

Mastering Data Insights: Zero-Shot Classification with OpenAI and PowerShell

Frank Lesniak

X: @FrankLesniak

Danny Stutz

X: @danny_stutz

While you wait: PLEASE take our survey* 👉
(the results will be used during this talk!)

April 8-11, 2024

* = Disclaimer: Any information submitted via the form linked above will become public information with no expectation of privacy, however it is an anonymous form in the sense that your name will not be collected.

Command Prompt

```
C:\Users\flesniak>whoami
Frank Lesniak (@franklesniak)
Senior Enterprise Technology Architect at West Monroe

Experience:
Almost 20 years in consulting. Microsoft 365 (modern work) consulting team lead.

Credentials:
MCSEs. PowerShellin' since 2006.
```

```
**********************************************************************************************
Ask Me About:

- PowerShell automation (duh!)
- Executing corporate divestitures and integrations - where I spent most of my consulting client time
- Retro video game software and FPGA emulation
- Building a passive aircraft radar
- My upcoming home construction project

**********************************************************************************************
Contact:

- x.com/franklesniak
- linkedin.com/in/flesniak
- github.com/franklesniak
- bsky.app/profile/franklesniak.com or infosec.exchange/@franklesniak
- flesniak ATSIGN westmonroe.com


C:\Users\flesniak>
```

Command Prompt

```
C:\Users\dstutz>whoami
Danny Stutz (@danny_stutz)
Cloud and Infrastructure Architect at West Monroe

Experience:
Almost 5 years working as a Microsoft 365/Entra/Azure technical lead

Credentials:
West Monroe PowerShell Interest Group lead.

**********************************************************************************************
Ask Me About:

- My mechanical keyboard! (Keychron K2, Boba U4's)
- Migration tool automation with PowerShell (ShareGate, MigrationWiz, azcopy, etc...)
- Threat hunting, M365/Entra/Azure security
- Carve-out and Divestiture migration work (where I also spend the majority of my consulting time)
- My music taste (send me your favorite tunes on Slack!)

**********************************************************************************************
Contact:

- x.com/danny_stutz
- linkedin.com/in/daniel-stutz
- github.com/danstutz
- dstutz ATSIGN westmonroe.com


C:\Users\dstutz>
```

THANK YOU TO OUR SPONSORS

Agenda

- Survey
- Introduction and Context
- Data Scrubbing & Anonymization
- Embeddings
- K-means Clustering
- The Centroid
- Getting the Topic (or Category/Theme)
- Code Review
- Demo
- Q&A

* = Disclaimer: Any information submitted via the form linked above will become public information with no expectation of privacy, however it is an anonymous form in the sense that your name will not be collected.

AHEM!: PLEASE take our survey* 👉
(the results will be used during this talk!)

Survey Says

- We're doing data analysis, so guess what? We need data! 🤯
- If you haven't already, please scan the QR code on the right and fill out the short survey.
- We'll use your responses as sample data to demonstrate how our process works.

Disclaimer: Any information submitted via the form linked above will become public information with no expectation of privacy, however it is an anonymous form in the sense that your name will not be collected.

Introduction and Context

- Why is it difficult to analyze free response/text data?

Introduction and Context

- Why is it difficult to analyze free response/text data?
  - Inherently unstructured
  - Typos/misspellings/l33t
  - Inconsistent from person to person
  - Difficult to categorize/organize
  - Differences in language/dialect
  - Data privacy
  - Context/ambiguity

Introduction and Context

- Why is it difficult to analyze free response/text data?
- Sure, it's difficult – but we still need to do it!
  - Survey responses
  - Service desk ticket analysis
  - Restaurant/product reviews
  - Any other use cases?

Introduction and Context

- Why is it difficult to analyze free response/text data?
- Sure, it's difficult – but we still need to do it!
- So, how might we do this **manually**? 🥴

	G	I	J
1	Question	Response-Scrubbed	Notes
2	Without mentioning the Client name or specific people, what made your most challenging project challenging?	Lack of client buy-in and being undercut on cost - preventing us from winning build work	Cost cutting led to Acme Company only doing design work
3	Without mentioning the Client name or specific people, what did you like the most about your favorite project?	A team of competent, capable professionals and a laid-back but responsive client	A great Acme Company team and the client being easy to work with
4	Imagine the worst project possible. What is it about the project that would make it the worst?	A bad Acme Company team	A bad Acme Company team

Scenario We Solve For

- Comments on an employee survey – need to categorize them
- Product reviews
- Any free response/text data that needs to be categorize

Building Context

There's a big difference between:

- "Twice a week" (lack of any context)
- "Question: How often do you exercise?
  Answer: Twice a week"
- "Question: How often do you want to quit your job?
  Answer: Twice a week"

Sometimes, including context in our analysis is important!

Data Scrubbing + Anonymization

- Spoiler alert: we will be calling some large language model (LLM) APIs to help with our analysis

Azure OpenAI

- Dedicated instance – no privacy concerns
- Model availability sometimes lags

Public OpenAI

- Data submitted used for training (becomes public!)
- Models so fresh

Data Scrubbing + Anonymization

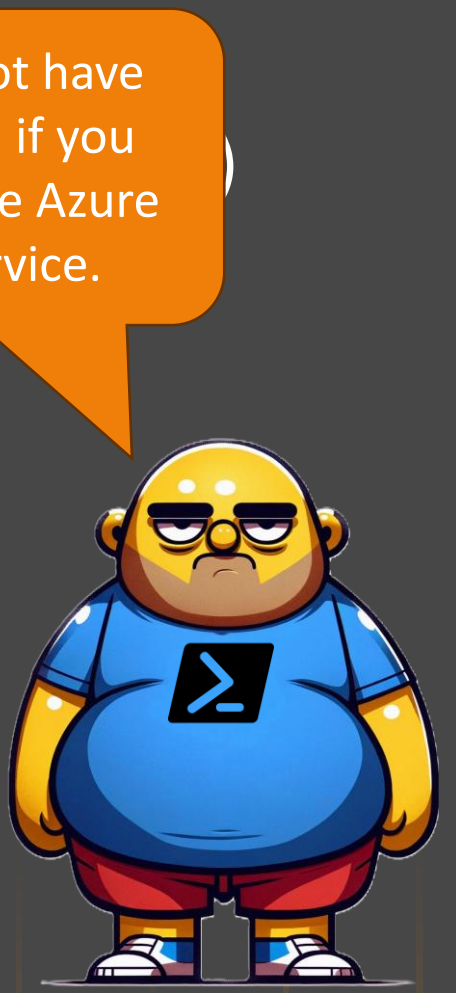
- Spoiler alert: we will be calling some large language model (LLM) APIs to help with our analysis
- We'll be using public OpenAI APIs in this talk

So, we need a way to "scrub" the data before making it effectively public domain

Data Scrubbing + Anonymization

- Spoiler alert: we will be calling some large lar APIs to help with our analysis
- We'll be using public OpenAI APIs in this talk

So, we need a way to "scrub" the data before making it effectively public domain



Data Scrubbing + Anonymization

- Spoiler alert: we will be calling some large language model (LLM) APIs to help with our analysis
- We'll be using public OpenAI APIs in this talk
- Depending on what you are doing, jargon can also be a problem

"Everyone I worked with in ET is great! Especially C-SC and PAMs"

Disclaimer: we are not data scientists

But we try our best ☺

Embeddings

- Embeddings are like GPS coordinates
  - But instead of three numbers representing an X-Y position on the Earth and an altitude, we have numbers that represent a piece of text
  - …but instead of three dimensions, the text-embedding-3-large model has 3072!

Embeddings

- Embeddings are like GPS coordinates
  - But instead of three numbers representing an X-Y position on the Earth and an altitude, we have numbers that represent a piece of text
  - …but instead of three dimensions, the text-embedding-3-large model has 3072!
  - In our case, we are “embedding” text – but you can also embed images

Embeddings

- Embeddings are like GPS coordinates
- Similar text (words, ideas, themes, etc.) have similar embeddings
  - The dimensions are not public information; they mean something to the robots but are meaningless to flesh bags like us
  - Nevertheless, we know thanks to the data scientists that similar text will have similar “coordinates”

Embeddings

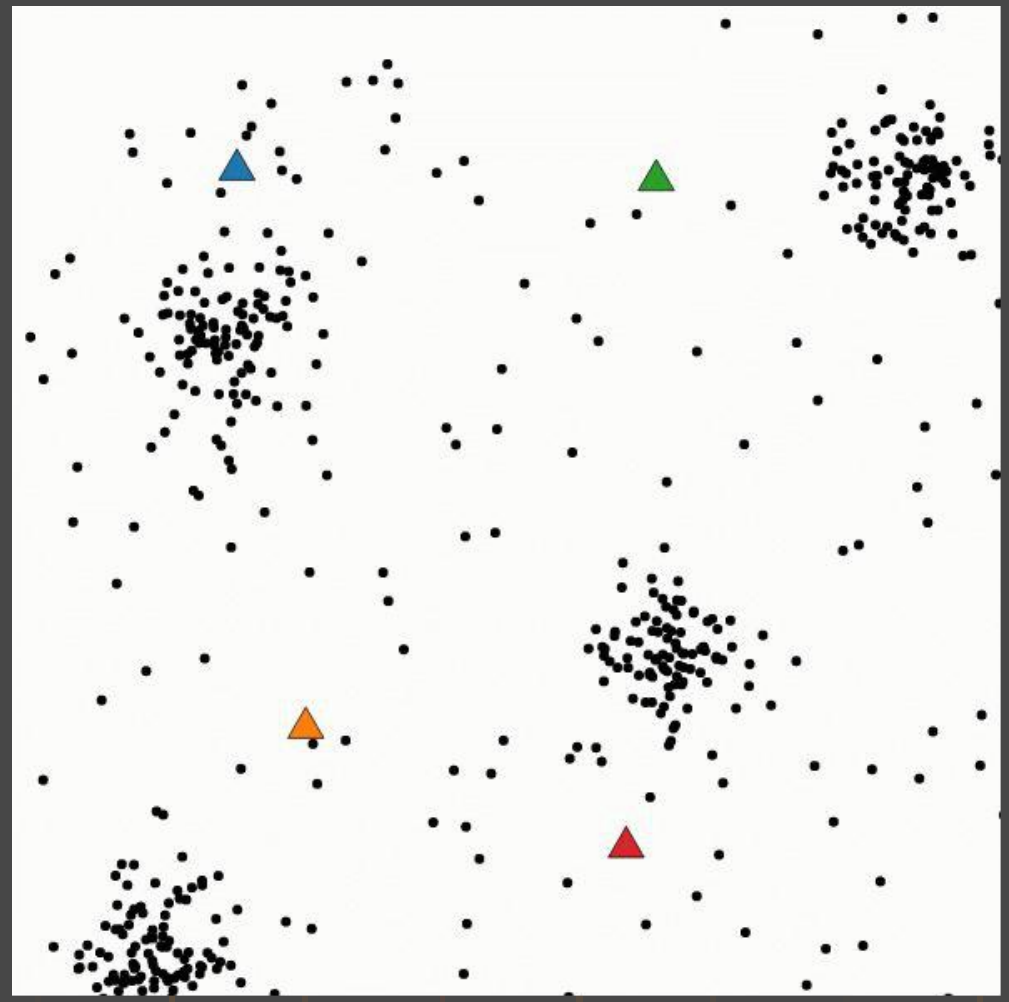
- Embeddings are like GPS coordinates
- Similar text (words, ideas, themes, etc.) have similar embeddings
- Embeddings cost money and take time to generate
  - The text-embedding-3-large model costs $0.13/1M tokens
  - Each word is typically 2-4 tokens
  - For small pieces of text, storing 3072 floating-point numbers (coordinates) may take up more memory than storing the text itself
  - The size difference is very pronounced if we write the floating-point numbers to CSV

Embeddings

- Embeddings are like GPS coordinates
- Similar text (words, ideas, themes, etc.) have similar embeddings
- Embeddings cost money and take time to generate
- OpenAI is not the only game in town:
  - Google's BERT and its variants
  - Facebook's FastText
  - GloVe (Global Vectors for Word Representation)
  - ELMo (Embeddings from Language Models)

K-means Clustering

- K-means Clustering is an algorithm that partitions data points (embeddings) into K distinct, non-overlapping clusters (think of clusters as categories).



K-means Clustering

- There are many alternative algorithms to K-means Clustering
  - DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
  - Gaussian Mixture Models (GMM)
  - Hierarchical Clustering
  - Spectral Clustering
  - OPTICS (Ordering Points To Identify the Clustering Structure)

K-means Clustering

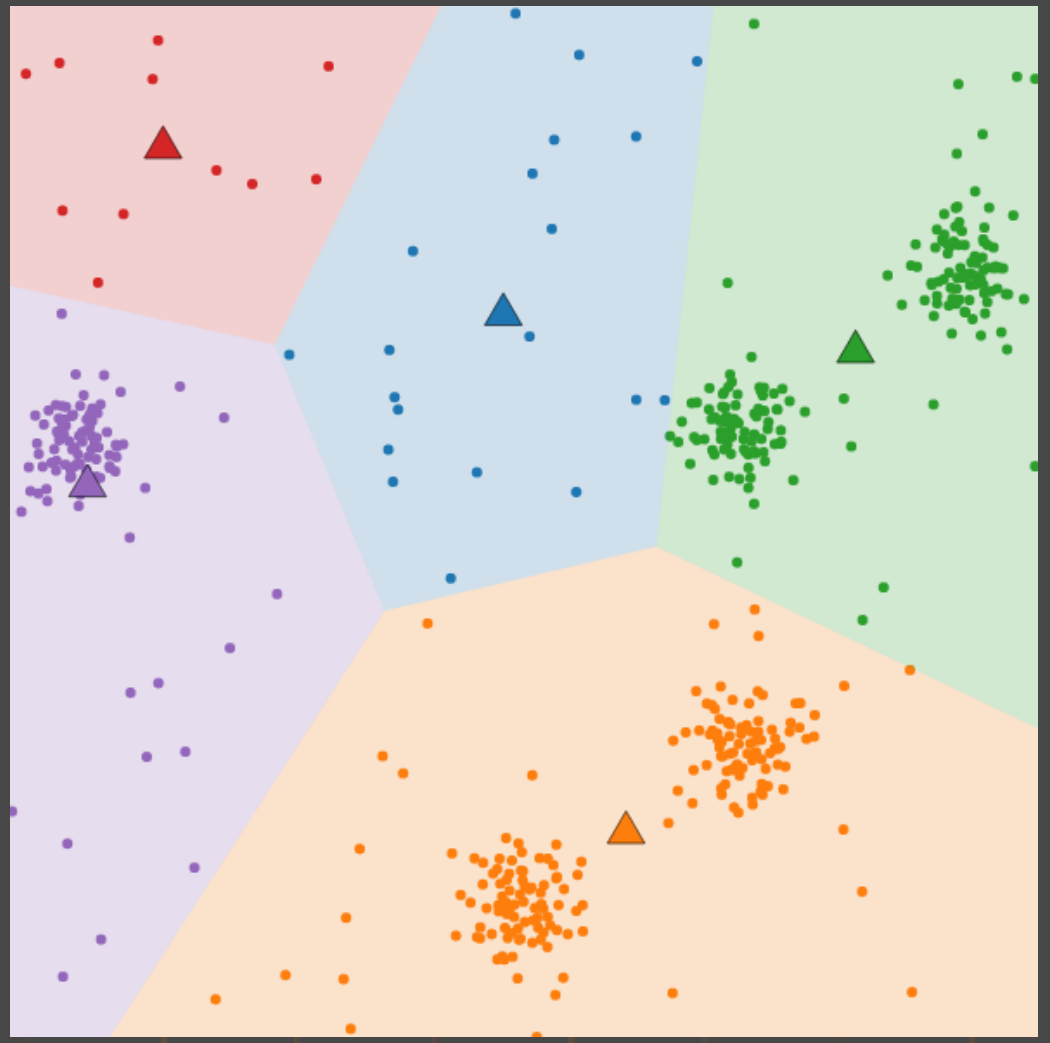
- There are many alternative algorithms to K-means Clustering
  - DBSCAN (Density-Based Spatial Clustering of Applicatio
  - Gaussian Mixture Models (GMM)
  - Hierarchical Clustering
  - Spectral Clustering
  - OPTICS (Ordering Points To Identify the Clustering Structure)

K-means Clustering

- There are many alternative algorithms to K-means Clustering
  - DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
  - Gaussian Mixture Models (GMM)
  - Hierarchical Clustering
  - Spectral Clustering
  - OPTICS (Ordering Points To Identify the Clustering Structure)
- We use K-means Clustering in our process mainly because K-means Clustering is the most available clustering method available in PowerShell (okay, .NET). While other clustering methods have their advantages and disadvantages, for simplicity and ease of integration, we decided to roll with K-means Clustering

The Centroid

- Once we've created our clusters, all we've done is grouped similar ideas together
  - Essentially, we've categorized the comments without knowing what the categories are
- Each cluster has a specific coordinate known as the "centroid" (▲) , which is reported to us as part of the algorithm



The Centroid

- Since we have the embeddings (coordinates) for each item in the cluster and the centroid's "coordinates," we can find the item closest to the centroid.
- This closest item *should* be most representative of the cluster

The Centroid

- Since we have the embeddings (coordinates) for each item in the cluster and the centroid's "coordinates," we can find the item closest to the centroid.
- This closest item *should* be most representative of the cluster

The Centroid

- Since we have the embeddings (coordinates) for each item in the cluster and the centroid's "coordinates," we can find the item closest to the centroid.
- This closest item *should* be most representative of the cluster
- We can also select the *n*-most central items in the cluster and treat those as a set

Getting the Topic

- We have our data grouped and know the “most representative” data point(s) in each group, but we still haven’t surfaced the topic or summary of each cluster!
- To do this programmatically, we can take the most representative comment(s) and ask ChatGPT for help.

End to End Process

1. Add context to the data, if necessary
2. Remove company-specific jargon and identifiable information
3. Pull down the embeddings for the dataset from OpenAI
4. Then, using the local embeddings file, invoke K-means Clustering on the dataset to cluster data points together
5. Identify the “most representative” data point(s) for each cluster
6. Submit the “most representative” data point(s) to ChatGPT to define the categories for each cluster
7. Output the categories and their respective comments for a full view of the insights!

PowerShell + DevOps
Global Summit

Code Review

April 8-11, 2024

Follow along in our repo on GitHub: https://github.com/frankl esniak/AutoCategorizerPS

Add-ContextToDataset.ps1

Parameterized string concatenation – nothing too fancy!

```
.\Add-ContextToDataset.ps1
   -InputCSVPath 'C:\Users\jdoe\Documents\Contoso Employee Survey
Comments Aug 2021.csv'
   -TextBeforeFieldName1 'On an employee engagement survey, a question
was asked: '
   -FieldName1 'Question'
   -TextBeforeFieldName2 ' #### In response, the employee wrote the
following comment: '
   -FieldName2 'Comment'
   -AdditionalContextDataFieldName 'AdditionalContext'
   -OutputCSVPath 'C:\Users\jdoe\Documents\Contoso Employee Survey
Comments Aug 2021 - With Additional Context.csv'
```

Add-ContextToDataset.ps1

Parameterized string concat

```
.\Add-ContextToDataset.ps1
    -InputCSVPath 'C:\Users\jdo
Comments Aug 2021.csv'
    -TextBeforeFieldName1 'On a
was asked: '
    -FieldName1 'Question'
    -TextBeforeFieldName2 ' ##
following comment: '
    -FieldName2 'Comment'
    -AdditionalContextDataFieldName 'AdditionalContext'
    -OutputCSVPath 'C:\Users\jdoe\Documents\Contoso Employee Survey
Comments Aug 2021 - With Additional Context.csv'
```

We use PSObject Properties to dynamically access CSV column names. For example:

```
($RowInCSV.PSObject.Properties |
Where-Object {$_.MemberType -eq
'NoteProperty' -and $_.Name -eq
$FieldName1 }).Value
```

Add-ContextToDataset.ps1

Parameterized string concate

```
.\Add-ContextToDataset.ps1
    -InputCSVPath 'C:\Users\jdo
Comments Aug 2021.csv'
    -TextBeforeFieldName1 'On a
was asked: '
    -FieldName1 'Question'
    -TextBeforeFieldName2 ' ###
following comment: '
    -FieldName2 'Comment'
    -AdditionalContextDataField
    -OutputCSVPath 'C:\Users\jd
Comments Aug 2021 - With Additional Context.csv'
```

In this specific example, the contents of the "Question" field and the "Comment" field are concatenated into a new column ("AdditionalContext"). Based on the supplied parameters, the resulting field is structured like:

```
On an employee engagement survey, a
question was asked: <Question> ### In
response, the employee wrote the following
comment: <Comment>
```

Add-ContextToDataset.ps1

BUT WAIT, THERE'S MORE!:

Things we wish we could talk about, but don't have time to:

- Primitive vs. complex object types
- Shallow-cloning vs. deep cloning
- Serialization techniques; imperfections
- PowerShell forward and backward compatibility
- Security concerns with serialization
- Serialization performance
- Copy-Object

(if these are interesting, come find us later, or tell everyone how great this talk was so that we get invited back next year 😁)

Convert-DataToAnonymizeAndRemoveJargon.ps1

- Simple “find and replace” operation, with find and replace text supplied via CSVs:
  - One for case-sensitive replacement
  - One for case-insensitive replacement
- “West Monroe” becomes “Acme Company”
- “SS team” becomes “Shared Services divisions”
  (but “compass team” does **not** become “compaShared Services divisions”!)

Follow along in our repo on GitHub: https://github.com/franklesniak/AutoCategorizerPS

Convert-DataToAnonymizeAndRemoveJargon.ps1

case-sensitivewords.csv

```
"Find","Replace"
"SC-SPAMs","Senior Consultants, Managers, and Senior Managers"
"The ICT team","the Information and Communications Technology division"
"SOW","statement of work"
"SC","Senior Consultant"
"PAMs","Managers"
"The TTS","The Technology Transaction Services specialty in the Technology Consulting practice"
"the West Monroe Partners","the Acme Company"
"SPAMs","Senior Managers"
"the ICT team","the Information and Communications Technology division"
"PTO","flexible time-off"
"The DEA","The Data Engineering and Analytics specialty in the Technology Consulting practice"
"PTs","Pricing Tools"
"SC-PAM","Senior Consultants and Managers"
"SPAM+","Senior Managers, Directors, and Partners"
"SC+","Senior Consultants, Managers, Senior Managers, Directors, and Partners"
```

Convert-DataToAnonymizeAndRemoveJargon.ps1

case-insensitivewords.csv

```
"Find","Replace"
"PE firm","private equity firm"
"West Monroe Partners","Acme Company"
"the mega-tech,"the Technology Consulting practice"
"the megatech","the Technology Consulting practice"
" ERG "," employee resource group "
"PXEL","the Digital Product practice"
"the MSD acquisition","investment by Our Investors"
" the line ET."," the Enterprise Technology specialty in the Technology Consulting practice."
"the MSD affiliation","the affiliation between Acme Company and Our Investors"
" ERGs."," employee resource groups."
"the PXEL","the Digital Product practice"
"the line ET team","the Enterprise Technology specialty in the Technology Consulting practice"
"the tech practice","the Technology Consulting practice"
" the ET "," the Enterprise Technology specialty in the Technology Consulting practice "
"technology practice","the Technology Consulting practice"
```

Get-TextEmbeddingsUsingOpenAI.ps1

- A simple API call…

```
$hashtableOpenAIHeaders = @{
    'Content-Type' = 'application/json'
    'Authorization' = ('Bearer ' + ($refStrGPTAPIKey.Value))
}

$strJSONRequestBody = @{
    input = $refStrTextToEmbed.Value
    model = $refStrGPTModel.Value
    max_tokens = $intGPTMaxTokens
    temperature = $doubleTemperature
} | ConvertTo-Json
```

```
Invoke-RestMethod -Uri 'https://api.openai.com/v1/embeddings' -Headers $hashtableOpenAIHeaders -Meth
```

Get-TextEmbeddingsUsingOpenAI.ps1

- A simple API call…

```
$hashtableOpenAIHeaders = @{
    'Content-Type' = 'application/j
    'Authorization' = ('Bearer ' +
}

$strJSONRequestBody = @{
    input = $refStrTextToEmbed.Value
    model = $refStrGPTModel.Value
    max_tokens = $intGPTMaxTokens
    temperature = $doubleTemperature
} | ConvertTo-Json
```

While the OpenAI embeddings model is a parameter here, we have a separate wrapper function that specifies the 'text-embedding-3-large' model

```
Invoke-RestMethod -Uri 'https://api.openai.com/v1/embeddings' -Headers $hashtableOpenAIHeaders -Meth
```

Get-TextEmbeddingsUsingOpenAI.ps1

- A simple API call…

```
$hashtableOpenAIHeaders = @{
    'Content-Type' = 'application/json'
    'Authorization' = ('Bearer ' + ($refStrGPTAPIK
}

$strJSONRequestBody = @{
    input = $refStrTextToEmbed.Value
    model = $refStrGPTModel.Value
    max_tokens = $intGPTMaxTokens
    temperature = $doubleTemperature
} | ConvertTo-Json
```

```
Invoke-RestMethod -Uri 'https://api.openai.com/v1/embeddings' -Headers $hashtableO  H  s -Meth
```

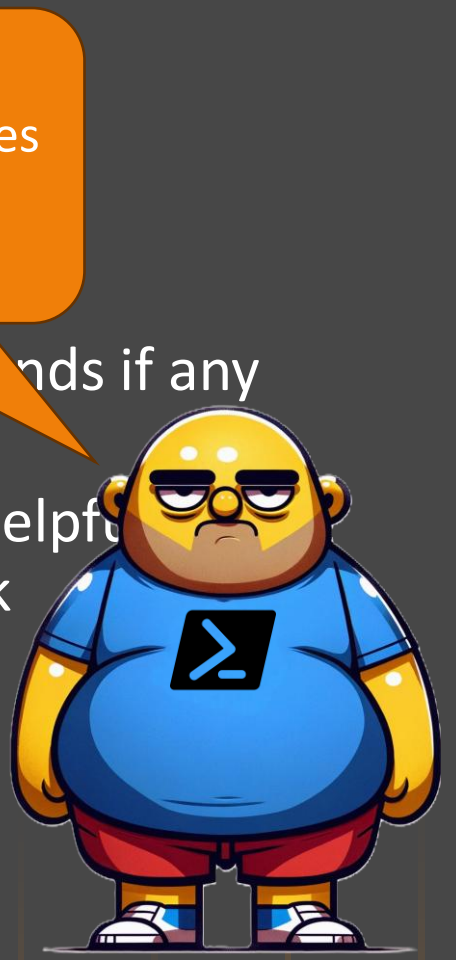
Pssh. You amateurs are calling an Internet API from a conference, during a live demo? Good luck!

Get-TextEmbeddingsUsingOpenAI.ps1

- A simple API call… with lots of goodies!
  - Automatic retries via error handling, automatic recursion, and exponential back-off timer
  - Checks for PowerShell modules; provides helpful install commands if any are missing
  - Checks to see if PowerShell modules are up to date; provides helpful warning message if modules are out of date but does not block execution (check can be suppressed)

Get-TextEmbeddingsUsingOpenAI.ps1

- A simple API call… with lots of goodies!
  - Automatic retries via error handling, autor… exponential back-off timer
  - Checks for PowerShell modules; provides helpful install com…nds if any are missing
  - Checks to see if PowerShell modules are up to date; provides helpf… warning message if modules are out of date but does not block execution (check can be suppressed)



Get-TextEmbeddingsUsingOpenAI.ps1

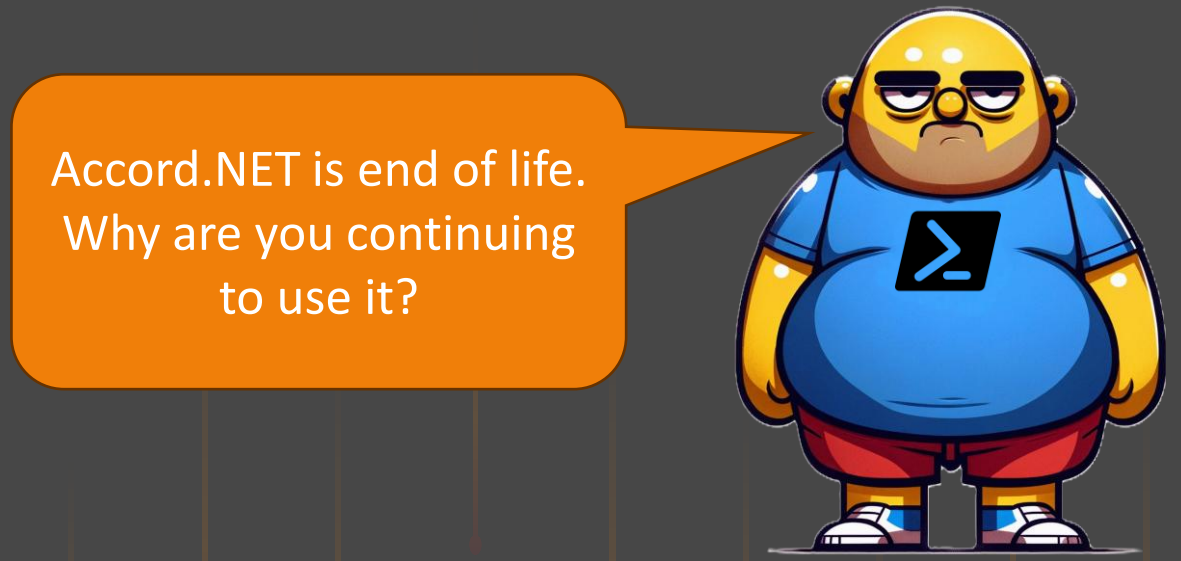
- A simple API call… with lots of goodies!
  - Automatic retries via error handling, automatic recursion, and exponential back-off timer
  - Checks for PowerShell modules; provides helpful install commands if any are missing
  - Checks to see if PowerShell modules are up to date; provides helpful warning message if modules are out of date but does not block execution (check can be suppressed)
  - We use the SecretManagement module to securely store and retrieve the OpenAI API key from an Azure Key Vault

Invoke-KMeansClustering.ps1

- Uses the Accord.NET NuGet package to perform K-means Clustering

Invoke-KMeansClustering.ps1

- Uses the Accord.NET NuGet package to perform K-means Clustering



Invoke-KMeansClustering.ps1

- Uses the Accord.NET NuGet package to perform K-means Clustering
  - We use Accord.NET because it's compatible with "full diesel" .NET Framework 4.x as well as .NET Standard 2.0
  - This means we can run it on Windows PowerShell 5.1 as well as newer PowerShell 7.x without any trouble

Invoke-KMeansClustering.ps1

- Uses the Accord.NET NuGet package to perform K-means Clustering
- Lots of goodies in this script!
  - Split strings without RegEx!

Invoke-KMeansClustering.ps1

- Uses the Accord.NET NuGet package to perform K-means Clustering
- Lots of goodies in this script!
  - Split strings without RegEx!

Invoke-KMeansClustering.ps1

- Uses the Accord.NET NuGet package to perform K-means Clustering
- Lots of goodies in this script!
  - Split strings without RegEx!
  - Checks for registration of nuget.org as a package provider; helpful warning message if not registered
  - Checks to see if required NuGet packages are “installed”; provides a helpful warning message if not
  - Dynamically locates the “installed” NuGet package DLL file(s) and loads them into memory

Invoke-KMeansClustering.ps1

- Uses the Accord.NET NuGet package to perform K-means

```
# Load the .dll
Write-Debug ('Loading .dll: "' + $strDLLPath + '"')
try {
    Add-Type -Path $strDLLPath
} catch {
    $strMessage = 'Error loading .dll: "' + $strDLLPath + '"; the LoaderException(s) are: '
    $_.Exception.LoaderExceptions | ForEach-Object { $strMessage += $_.Message + '; ' }
    Write-Warning $strMessage
    return
}
```

- Dynamically locates the “installed” NuGet package DLL file(s) and loads them into memory

Invoke-KMeansClustering.ps1

- How many clusters is the right number?
  - The square root of the number of items is a good guess

```
# TODO: Dynamically set the number of clusters
if (($null -eq $NumberOfClusters) -or ($NumberOfClusters -le 0)) {
    $intNumberOfClusters = [int]([Math]::Ceiling([Math]::Sqrt($arrInputCSV.Count)))
} else {
    $intNumberOfClusters = $NumberOfClusters
}
```

Invoke-KMeansClustering.ps1

- How many clusters is the right number?
  - The square root of the number of items is a good guess
  - There are algorithms (e.g., “elbow method” - not yet implemented) that can programmatically determine the optimal number of clusters

Invoke-KMeansClustering.ps1

- Because Accord.NET is loaded into memory, and because it's just .NET, we can access it like any other object:

```
$kmeans = New-Object -TypeName 'Accord.MachineLearning.KMeans' -ArgumentList @($intNumberOfClusters)
[void]($kmeans.Learn($arrEmbeddings))
$arrClusterNumberAssignmentsForEachItem = $kmeans.Clusters.Decide($arrEmbeddings)
```

(although, the syntax is a bit strange if you're not a data scientist)

Invoke-KMeansClustering.ps1

- For each item in each cluster, we calculate its “Euclidian distance” to the centroid of the cluster
  - Think of this as the distance between two points on a map

```
function Measure-EuclideanDistance($Point1, $Point2) {
    $doubleSum = [double]0
    for ($i = 0; $i -lt $Point1.Length; $i++) {
        $doubleSum += [Math]::Pow($Point1[$i] - $Point2[$i], 2)
    }
    return [Math]::Sqrt($doubleSum)
}
```

Invoke-KMeansClustering.ps1

- Finally, we sort the clustered items by their distance, shortest to longest, and generate our output!

Get-TopicForEachCluster.ps1

- Again, a simple API call… this time against OpenAI's chat API

Get-TopicForEachCluster.ps1

```
$hashtableOpenAIHeaders = @{
    'Content-Type' = 'application/json'
    'Authorization' = ('Bearer ' + ($refStrGPTAPIKey.Value))
}

$arrMessages = @(
    @{
        'role' = 'system'
        'content' = 'You are ChatGPT, a large language model trained by OpenAI.'
    },
    @{
        'role' = 'user'
        'content' = ($refStrTextToSend.Value)
    }
)

$strJSONRequestBody = @{
    model = $refStrGPTModel.Value
    messages = $arrMessages
    temperature = $doubleTemperature
} | ConvertTo-Json
```

Get-TopicForEachCluster.ps1

- Again, a simple API call… this time against OpenAI's chat API

```
Invoke-RestMethod -Uri 'https://api.openai.com/v1/chat/completions' -Headers $hashtableOpenAIHeaders
```

Get-TopicForEachCluster.ps1

- Again, a simple API call… this time against OpenAI's chat API
- We don't need a "conversation" – a single question/answer exchange will do

Get-TopicForEachCluster.ps1

- Again, a simple API call… this time against OpenAI's chat API
- We don't need a "conversation" – a single question/answer exchange will do
  - Here's an example prompt that the script uses:

```
In as few words as possible (certainly no more than 1-3 words),
describe the topic, main idea, or theme of the following five texts,
treated as a set. Each text is separated by three forward slashes
(///): <top five representative texts>
```

Get-TopicForEachCluster.ps1

- Again, a simple API call… this time against OpenAI's chat API
- We don't need a "conversation" – a single question/answer exchange will do
- Again, we have automatic retries via error handling, automatic recursion, and exponential back-off timer – to ensure Internet "blips" don't screw us up

Demo

Q+A

THANK YOU

Please review this session

Session Review

April 8-11, 2024