Tasks performed

To earn this Microsoft Applied Skills credential, learners demonstrate the ability to create a natural language processing (NLP) solution by using Azure AI Language.

Candidates for this credential should have a solid understanding of creating and using various Azure NLP models through both Language Studio and in code, including custom models. They should also have experience programming in either Python or C#, be familiar with the Azure portal, and be comfortable provisioning Azure AI resources.

- Deploy a language resource, and use prebuilt models
- Create a custom text classification solution
- Create a custom named entity recognition (NER) solution

Create the Azure AI Language service

- In the Azure portal (portal.azure.com), search for Language, and click on "Language".
- In the Azure AI Services – Language service, click on "+ Create".
    - The Azure AI Language service combines the previously available Text Analytics, QnA Maker and LUIS (Language Understanding) Cognitive Services.
- You then have the choice to select:
    - Custom question answering – you will not need that for this course, and
    - Custom text classification, Custom named entity recognition, Custom summarization, Custom sentiment analysis & Custom Text Analytics for health. This is needed for this course, so click on Select.
- Then click on "Continue to create your resource".
- Select a Subscription, Resource group and region.
    - Not all features are available in all regions. However, all features are available in East US region.
    - Otherwise, choose one which is close to your location.
- Enter a name.
    - It can include upper and lower case letters, numbers and hyphens.
    - It cannot include a space.
- Select a pricing tier.
    - The free F0 tier allows for 5,000 text records to be processed free per month.
    - The Standard S0 tier allows for a greater number of text records to be processed.
- For the custom models, you will need a storage account.
    - Click on "New storage account".
    - Enter a storage account name and storage account type.
        - The "Standard LRS" is the cheapest version.
            - It costs 2.1 US cents per Gigabyte per month.

- Click on “Review + create” and then “Create”.
- After it has been created, go to the resource.

The Language Studio

- The Language Studio is a group of UI-based tools so you can use the Azure AI Language features in a user-friendly way.
- To go to the Language Studio, in the Overview tab click on “Get started with Language Studio”. This takes you to https://language.cognitive.azure.com/
- In the tabs, you can select from:
    - Featured,
    - Extract information,
    - Classify text,
    - Understand questions and conversational language,
    - Summarize text, and
    - Translate text (but this relies on an Azure Translator resource, not an Azure AI Language resource, and so is outside of this course).
- If you have more than one resource, you can switch between them by clicking on your profile (at the top-right corner of the screen), and then clicking “Select” next to “Current resource”.
    - This can be useful if you have resources in different geographical locations.

Classify text – Language detection

- Language detection detects the predominant language for more than 110 languages. It can also detect variants, dialects and some regional and cultural languages.
- For languages which can be written in multiple scripts, it can also detect which script is being used. This is current for more than 15 scripts and Romanized languages, including Bengali, Gujarati, Hindi, Tamil, Arabic, Serbian and Cyrillic.
- Cost: US$1 per 1,000 records
    - A record is up to 1,000 characters sent to one service.
        - If it goes over, it becomes a second record for charging purposes.
        - If it is sent to multiple services, it generally becomes multiple records for charging purposes.
    - The maximum length of a document is 5,120 characters.
        - However, this can be exceeded if you use an asynchronous way of calling the code, in which case the maximum length is 125,000 characters in up to 25 documents.
    - You can save 30%-65% if you prepay and commit to at least a million text records.
- Sample texts:

    The dishes were exquisitely prepared, and the generous hospitality was unmatched.

La característica Detección de idioma puede detectar el idioma en el que está escrito un document

こんにちは。私の名前はフィリップです。

- Kon'nichiwa. Watashi no namaeha Firippudesu.

C#

- Create a blank folder. In that folder, run the following command in the Terminal:

  ```
  dotnet new console -n Program
  ```

- Go to the appropriate "Program" subfolder.
- Install the TextAnalytics library in the Terminal:

  ```
  dotnet add package Azure.AI.TextAnalytics --version 5.3.0
  ```

- Add references to the Azure and Azure.AI.TextAnalytics namespaces.
- Add key and endpoint strings.
- Create the "client" using new TextAnalyticsClient, wrapping the endpoint in a new Uri, and the key in an (Azure.)AzureKeyCredential.
- Call the client using client. DetectLanguage, passing a phrase in ( ) s, and save it as "response"
  - If you find that you cannot see any autocompletion after entering "client.", then:
    - Open the C# Program folder as the root folder,
    - Press Ctrl+Shift+P to command the command palette,
    - Run the command "> .NET: Generate Assets for Build and Debug".
  - This might activate the autocompletion.
- Then you can use response.Name, response.Iso6391Name, and response.ConfidenceScore.

Python

- Create a blank folder.
- In the folder, install the library by running the following command in the Terminal:

  ```
  pip install azure-ai-textanalytics==5.3.0
  ```

- Add references to the AzureKeyCredential and TextAnalyticsClient classes.
- Create the "client" using new TextAnalyticsClient, wrapping the key in an AzureKeyCredential.
- Call the client using client.extract_key_phrases, passing a phrase as the document using a list (in hard brackets).
- As the response is a list, then add a [0] to access the first (and only) item in the response list.
- Then you can use response.primary_language.name, response.primary_language.iso6391_name, and response.primary_language.confidence_score.

Understand questions and conversational language – Question answering – pre-built

- Question answering finds answer from customer input in more than 50 languages.
- It is used to build chatbots, social media applications, and desktop applications.
- It can be used:
    - When you have static information,
    - When you want to provide the same answer to the same question,
- Cost: US$1-1.50 per 1,000 records
- There is also a custom version available, which allows you to create a project and customise the list of questions and answers.
    - This custom version is not needed for this Microsoft Applied Skill.
- The pre-built model is not part of the TextAnalyticsClient at the time of writing (although the custom version is), so we will not be create C# or Python code.

Summarize text – Document and conversation summarization

- Summarization allows you to produce a summary for a conversation or document.
- Some summarization features are available in very few regions and in 11 languages.
- Cost: US$2 per 1,000 records
- For the document version:
    - You can produce one or both of the following:
        - Extractive summarization.
            - This extracts important sentences from the document.
            - You can say the maximum number of sentences to be extracted.
        - Abstractive summarization.
            - This returns a summary of the document, not necessarily using the same words, but using the same idea.
- For the conversation version:
    - This is used for a conversation between an agent and customer, which includes issues and their resolution.
        - Example issue: My coffee machine doesn't work.
        - Example resolution: Power light is off. Rewired the plug.
    - It is currently available in English only.
    - It requires a dialog between exactly two people, and
        - Apart from a summary of the issue or resolution, each line must start by either:
            - The name of the speaker followed by a colon, or
            - "Customer" or "Agent" (in English), followed by a colon.

        - Each change of speaker must be on a new line.
    - You can produce any of the following:
        - Issue/resolution summarization
            - A summary of the issues and resolutions.
        - Chapter title summarization
            - Divides a conversation into chapters, and gives suggested chapter titles.
        - Recap
            - A short paragraph of the conversation.
        - Narrative summarization
            - Creates call or meeting notes or chat summarises.
        - Follow-up tasks
            - A list of follow-up tasks
- The code for C# and Python is in preview, and is only available in Sweden Central. As such, we will not be creating code for this in this course.

Extract information – Key phrase extraction

- Extracts words or phrases from sentences.
    - For example: "The dishes were exquisitely prepared, and the generous hospitality was unmatched" could extract "dishes" and "generous hospitality".
- Currently supported in 94 different languages.
    - Defaults to English.
    - The response may support multiple languages and emojis.
- Works better on larger amounts of text.
- Cost: US$1 per 1,000 records

C#

- After connecting to the TextAnalyticsClient (see "Language detection"), call the client using client.ExtractKeyPhrases.
- You can loop through each keyphrase in "response.Value".

Python

- After connecting to the TextAnalyticsClient (see "Language detection"), call the client using client.detect_language.
- You can loop through each key_phrase in "response.key_phrases".

Extract information – Named Entity Recognition (NER)- prebuilt

- Named Entity Recognition (NER) can identify and categorize entities in text.
- Generally available in over 70 languages and available in preview in over 90 languages.
- Cost: US$1 per 1,000 records.

- Entities include:
    - Person – Name of people,
    - PersonType – Job types or roles,
    - Location – Natural and human-made landmarks, structures, geographical features such as rivers, oceans and deserts, and geopolitical entities such as cities, countries, regions and states.
    - Organization – Companies, including medical companies and groups, political groups, musical bands, sport clubs and organizations, government bodies, public organizations, and stock exchange groups.
    - Events – Historical events, social events such as cultural events and holidays or sporting events, naturally occurring events.
    - Product – Physical objects, including computing products.
    - Skill – a capability, skill, or expertise.
    - Addresses, Phone Numbers, Emails, URLs, and network IP addresses.
    - Dates and times of days, including date and time ranges, and durations.
    - Quantities, including numbers, percentages, ordinal numbers, ages, currencies, dimensions and measurements, and temperatures.
- The resolution aims to provide a consistent output.
    - So “twenty” and “20” should resolve to the number 20, for instance.
- Categories for resolution include:
    - Age – unit can be Year, Month, Week or Day.
    - Currency – unit and ISO4217 would be the currency.
    - Date
        - where no year is given (for example “1 January”), the “timex” will indicate XXXX for the year, but “value” will indicate the current and next year.
        - where only a day is given, such as “Tuesday”, then “timex” will indicate “XXXX-WXX-” followed by the number of the day of the week (2, in this case).
    - Time
    - Datetime
    - Datetime ranges
    - Set – a reoccurring datetime period. “value” would be “not resolved”.
    - Duration – including:
        - Area,
        - storage (internally known as “InformationResolution”), such as Bit or Gigabyte,
        - Length – both metric and imperial,
        - Speed – both metric and imperial, per millisecond, second, minute, or hour

        - Volume – both metric and imperial,
        - Weight – both metric and imperial.
    - Number – "numberKind" could be Integer, Decimal, Fraction, Power and Percent.
    - Offset – such as first, last, or third.
    - Temperature – unit could be Celsius, Fahrenheit, Kelvin and Rankine.
- You can use it for:
    - Enhancing search capabilities and indexing.
        - You can analyse documents based on entities, and tag them.
    - Automate business processes.
        - Highlight name and location in a document, insurance claim, or support ticket.
    - Create analyses
        - Create an analysis based on popular information in reviews, emails and calls to extract relevant and repeated topics and trends.
- Sample text for programming: In the heart of the ancient city, on a bright morning of May 5th, 2024, 72-year-old Maria embarked on her journey. With her 50 euros, she was at the grand opening of the new 2,000 square meter museum, which would start at 10:30 AM, and last until 3:00 PM. Being the third person in line, she could hardly contain her excitement in the pleasant 23°C sunshine. She would come back every week.

C#

- After connecting to the TextAnalyticsClient (see "Key phrase extraction"), call the client using client.RecognizeEntities, passing a phrase in ( ) s, and save it as "response".
- You can loop through each entity in "response.Value", and retrieve Text, Category, SubCategory, ConfidenceScore, Length and Offset.

Python

- After connecting to the TextAnalyticsClient (see "Key phrase extraction"), call the client using client.recognize_entities.
- You can loop through each entity in "response.entities", and retrieve text, category, subcategory, confidence_score, Length and Offset

Extract information – Personally Identifiable Information (PII) detection

- This identifies, categorizes and redacts sensitive information.
- It is available in over 70 languages.
- Cost: US$1 per 1,000 records
- In addition to the standard entities, PII categories also include:
    - Financial account ID, including:
        - ABA routing number,
        - SWIFT code,
        - Bank account numbers,
        - Credit card information and
        - International Banking Account number (IBAN).

    - Government identification, including
        - Passport Numbers,
        - Tax IDs,
        - Driving licenses,
        - Medical account numbers,
        - Legal entity numbers,
        - ID cards,
        - Social Insurance numbers
- It can be used for:
    - Applying sensitivity labels.
        - If no PII entities are detected, this it might be labelled as public.
        - If US addresses and phone numbers are recognised, it might be labelled as confidential.
        - If bank numbers are recognised, it might be labelled as highly confidential.
    - Redacting personal information to give to others or from call center transcriptions.
        - For example, all PII excluding name.
    - Redacting personal information to reduce unconscious bias, or data cleaning or reducing unfairness from machine learning models.
- There is also “Conversation PII (preview)” available.
    - It is currently available in four languages: German, English, Spanish and French.
    - It requires a dialog between at least two people, and
    - Apart from a summary of the issue or resolution, each line must start by either:
        - The name of the speaker followed by a colon, or
        - “Customer” or “Agent” (in English), followed by a colon.
    - Each change of speaker must be on a new line.
- Sample Text for programming: “I, Alexander Defort of My Company Limited (company number 12345678), purchased yesterday a computer costing US$1,000. I used my VISA credit card, number 4523 4859 3953 1235, and expiry date 07/29. However, the computer has not yet arrived. Can you please call me on 555-0123 to resolve this issue.”

C#

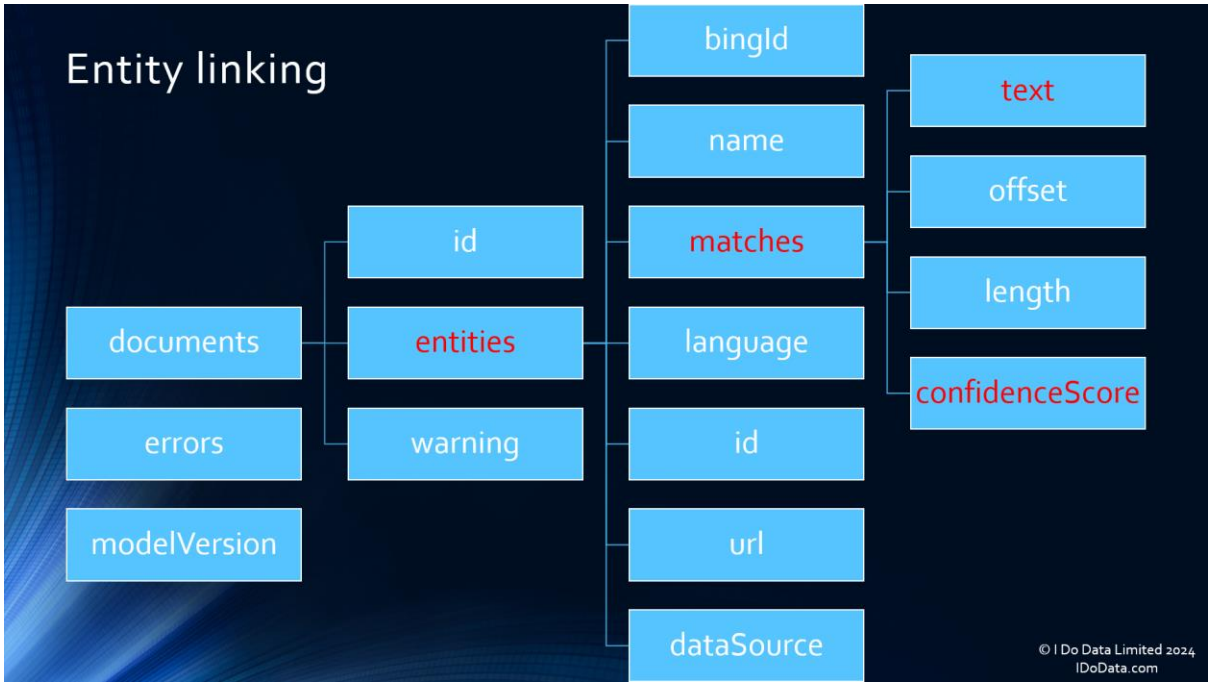
- After connecting to the TextAnalyticsClient (see “Key phrase extraction”), call the client using client. RecognizePiiEntities, passing a phrase in ( ) s, and save it as “response”.
- You can use response.Value.RedactedText to return the text with *s in place of PII.
- You can also loop through each entity in “response.Value”, and retrieve Text, Category, SubCategory, ConfidenceScore, Length and Offset.

Python

- After connecting to the TextAnalyticsClient (see “Key phrase extraction”), call the client using client.recognize_entities.
- You can use response.redacted_text to return the text with *s in place of PII.
- You can also loop through each entity in “response.entities”, and retrieve text, category, subcategory, confidence_score, Length and Offset.

Extract information – Entity linking

- This links entities to Wikipedia articles.
- It also disambiguates these entities – for example, “Mars” could refer to the planet or a chocolate bar.
  - “I saw Mars in the sky, next to the Moon.”
  - “I then ate some chocolate - I think it was Mars” or “Mars bar”.
- It is currently available in English and Spanish.
  - If you don’t specify a language, English will be used.
- Works better on smaller amounts of text.



C#

- After connecting to the TextAnalyticsClient (see “Key phrase extraction”), call the client using client.RecognizeLinkedEntities.
- Then you can loop through each entity to get its Name, URL and DataSource.
- You can also loop through each entity’s Match to get its Text and ConfidenceScore

Python

- After connecting to the TextAnalyticsClient (see “Key phrase extraction”), call the client using the client.recognize_linked_entities.
- Then you can loop through each entity to get its name, url and data_source.

- You can also loop through each entity's match to get its text and confidence_score.

Classify text – Sentiment analysis and opinion mining

- Sentiment analysis retrieves positive, neutral or negative sentiment in over 90 languages.
    - If there is both a negative sentence and a positive sentence, then it will be called "mixed".
- Opinion mining returns more about the opinions related to the text, in over 90 languages.
    - It retrieves a target (object) or multiple objects, and their assessment (opinion).
- Cost: US$1 per 1,000 records
    - There is no additional charge for opinion mining.

C#

- After connecting to the TextAnalyticsClient (see "Key phrase extraction"), call the client using client.AnalyzeSentimentBatch.
    - You can also use AnalyzeSentimentOptions to set IncludeOpinionMining to True.
- Then you can loop through each review to get its Sentiment and ConfidenceScores, Positive, Negative and Neutral.
- You can also loop through each review's sentences to get the same information.
- If IncludeOpinionMining is True, then you can also loop through each sentence.Opinion to get its Target's Text, Sentiment and ConfidenceScores, both Positive and Negative.
- You can also loop through each Opinion's Assessment to retrieve the same information.

Python

- After connecting to the TextAnalyticsClient (see "Key phrase extraction"), call the client using client.analyze_sentiment.
    - You can also use a second parameter to set show_opinion_mining to True.
- Then you can loop through each document to get its sentiment and confidence_score, positive, negative and neutral.
- You can also loop through each document's sentences to get the same information.
- If show_opinion_mining is True, then you can also loop through each sentence.mined_opinion to get its Target's text, sentiment and confidence_score, both Positive and Negative.
- You can also loop through each opinion's assessment to retrieve the same information.

Custom models – Custom text classification

- Allows you to classify documents. It can be used for:
    - Automatically responding to emails, or routing them to the correct place.
    - Applications which need to classify documents, such as document searches, retail product searches, or knowledge mining.
- Cost: US$5 per 1,000 records
- Needs text files in one of over 90 languages.

    - You can choose to enable a multilingual option, to train in one language and use it that or a different language.

- First of all, upload documents into your Blob storage.
    - In the Azure Portal (portal.azure.com), search for "Storage accounts".
    - Click on the Storage account that was automatically created when you created the Language resource.
    - Go to Settings – Configuration, and click on "Allow Blob anonymous access". Then click Save. It may take a few seconds for this to be enabled.
    - Go to Data storage – Containers, and click on "+ Container". Set the "Anonymous access level" to "Container".
    - Click on the container.
    - Click on "Upload" and upload your text documents.
- In the Language Studio, go to Classify text – Custom text classification.
- Click "Create new project".
    - Connect your storage account if necessary. This is needed once per language resource, and cannot be disconnected later.
    - Select a single or multi label classification.
        - Single label classification – one classification per document, or
        - Multi label classification – multiple classifications per document.
    - Enter a name, language, and an optional description.
    - If your dataset or if the data when you will use it contains documents of different languages, click the "enable multi-lingual dataset" option.
    - Select your dataset container from the connected storage.
    - Select "No, I need to label my documents as part of the project"
        - Unless you have already labelled your data, and have .json labels in the dataset, then you can select "Yes, my documents are already labelled and I have a correctly formatted JSON labels file".
- Label your documents.
    - Add the classifications you which to use. Then either:
        - Click on each document in turn and select either one label (for single label classification), or one or more labels (for multi label classification).
        - You can also use the "auto labelling" preview feature to label it with GPT.
            - You will need an Azure OpenAI resource and deployment for this, and so is outside of this course.
    - Select whether the document is part of the training or testing dataset.
        - You could also use automatic data splitting – see later.
- You can add data to the training or test set.
    - Click on Training jobs (on the left hand side) – Start a training job (at the top).

- Click on Train a new model (or overwrite an existing model)
- Select the data splitting method. You can choose:
  - Automatically splitting the testing set from training data.
  - The recommended split is 80% for training and 20% for testing.
    - The training set is used for training the model.
    - The testing set is used to evaluate the model, and see if the labels you have applied to the documents are added by the model.
- Click "Train".
- You can then see the Training progress, job status and other details.
  - It can take a few minutes to several hours.
  - If your files have 13,000,000 characters, the training time is around 6 hours.
  - You can only have one training job running at any time.

- Review the model's performance by clicking on "Model performance" on the left-hand side.
  - You can see:
    - Overview – F1 score, precision, recall, data and time, and number of training and testing documents.
      - The F1 score is based on precision and recall.
      - Precision is how accurate your model is – the ratio between the correctly identified true positive and all identified positives.
      - Recall is the ability to predict actual positive classes – the ratio between the predicted true positives and what was actually tagged.
      - There may also be guidance on how to improve the model.
    - Class type performance – how your model performed during testing.
    - Test set details – see any mismatches:
      - True positive – correct classifications,
      - False positive – incorrect classifications,
      - False negative – something was not classified, when it should have been.
    - Dataset distribution – how the entities were distributed across your training and testing sets.
    - Confusion matrix (for single classification projects only) – it shows what the classification should be against what the model assigned.
  - If the test set is too small, the good/bad scores may not be representative of the actual performance.
  - If your data only covers a few scenarios, it may perform badly on other scenarios.

    - If your training data is not representative of the actual data, model performance may be poor.
- You can then deploy your model.
    - Click on “Deploying a model” on the left-hand side.
    - Click “Add deployment”.
    - Click “Create new deployment” unless you want to “Overwrite an existing deployment”.
    - Optionally, you can click on the Regions tab if you want to be deployed into multiple regions.
    - Click Deploy.
- Your job will typically be deployed for up to 12 months after the training expires.
- You can then test the deployed model.
    - Click on “Testing deployments” on the left-hand side.
    - Select the deployment.
    - If it is a multilingual project, select the language for the text.
    - Enter the text or upload a .txt file.
    - Click “Run the test”.
    - You can then see the extracted entities, and the JSON response.
- Example texts:
    - Solo Travel: Solo travel is a growing trend where individuals explore destinations on their own terms, offering a unique opportunity for personal growth and self-discovery. Traveling alone allows for a flexible itinerary, encouraging travelers to step out of their comfort zones, meet new people, and immerse themselves in new cultures at their own pace. Solo adventurers often find that their journeys are not just about the places they visit but also about the introspective journey and the sense of independence and confidence gained along the way. Whether it's navigating the bustling streets of a foreign city, finding solace in the tranquility of a secluded beach, or joining a group tour to meet like-minded travelers, solo travel is a rewarding experience that caters to a wide range of interests and personal growth opportunities.
    - Azure IoT is a collection of cloud services that provides comprehensive solutions to support the Internet of Things. It offers tools and services to connect, monitor, and control billions of IoT assets. With Azure IoT, businesses can harness powerful analytics to process data from connected devices, integrate business systems to transform insight into action, and create robust IoT applications to adapt to their needs. Whether it's optimizing industrial operations, enhancing retail experiences, or creating smart city infrastructures, Azure IoT empowers organizations to build scalable and secure IoT solutions that can drive innovation and deliver real-time responses in a connected world.
    - Ethnic fusion cuisine is a culinary approach that combines elements of different culinary traditions, offering an innovative and contemporary dining experience. This trend reflects the globalized world we live in, where cultural boundaries are increasingly blurred. Chefs are experimenting with blending the flavors, techniques, and ingredients of various cuisines to create novel and exciting dishes. For instance, you might find a dish that combines Thai spice profiles with traditional Mexican ingredients, offering diners a chance to explore new flavor landscapes while celebrating the diversity of global culinary traditions.

C#

- You will need variables to contain projectname and deploymentname
- After connecting to the TextAnalyticsClient (see “Key phrase extraction”), call the client using:

```
var response = await client.SingleLabelClassifyAsync(WaitUntil.Completed,
textsToAnalyse, projectname, deploymentname);
```

- You can then loop through the response.Value.result.classification to get the Category and ConfidenceScore.
- Because this is an asynchronous operation, you need to use an await to call the client, and an await when retrieving the results.

Python

- After connecting to the TextAnalyticsClient (see "Key phrase extraction"), call the client using client.detect_language.
- You can then loop through the response.documents.classifications to get the Category and ConfidenceScore.
- Then you can use response.primary_language.name, response.primary_language.iso6391_name, and response.primary_language.confidence_score.

Custom models – Named Entity Recognition (NER)

- Custom NER is used to identify entities in text in over 90 languages.
- Because it is custom, you can define the entities.
- Cost: US$5 per 1,000 records
- Create a storage account and container as per the custom text classification model.
- In the Language Studio, create a new project.
    - go to Extract information – Custom named entity recognition.
    - Click "Create new project".
    - Connect your storage account, if you haven't already.
    - Enter the project name, description and language.
        - If you have text or will have text from multiple languages, select "enable multi-lingual dataset".
    - Select "No, I need to label my documents as part of the project"
        - Unless you have already labelled your data, and have .json labels in the dataset, then you can select "Yes, my documents are already labelled and I have a correctly formatted JSON labels file".
- Tag your documents.
    - Add the classifications you which to use. Then either:
        - Click on each document in turn.
        - Highlight the relevant word or words (click on the first character, then click on the last character).
        - Select your label.
            - It is advisable to label exactly, consistently and fully.
            - To remove labels, click on the entity and select "Remove label".

        - You can also “Auto-label” using GPT.
    - Select whether the document is part of the training or testing dataset.
        - You could also use automatic data splitting.
- Then train, deploy and test your model as per the Custom text classification.
    - It will take a lot to train your custom NER model compared with the custom text classification.

Summary of methods

Capability	Python	C#
Language detection	detect_language	DetectLanguage
Key phrase extraction	extract_key_phrases	ExtractKeyPhrases
Named Entity Recognition (NER) - prebuilt	recognize_entities	RecognizeEntities
Personally Identifiable Information (PII) detection	recognize_pii_entities	RecognizePiiEntities
Entity linking	recognize_linked_entities	RecognizeLinkedEntities
Sentiment analysis and opinion mining	analyze_sentiment	AnalyzeSentimentBatch
Custom Text Classification	begin_single_label_classify	SingleLabelClassifyAsync
Named Entity Recognition (NER) – custom	begin_recognize_custom_entities	RecognizeCustomEntitiesAsync
Conversational language understanding	analyze_conversation	AnalyzeConversation
Text Analytics for health	begin_analyze_healthcare_entities	StartAnalyzeHealthcareEntitiesAsync