Chapter 12
Leveraging Prebuilt Models as a Service

Google Cloud Professional Data Engineer Exam objectives covered in this chapter include the following:

- **3. Operationalizing machine learning models**
- **✓ 3.1 Leveraging pre-built ML models as a service. Considerations include:**
- ML APIs (e.g., Vision AI, Speech AI)
- Customizing ML APIs (e.g., Auto ML Vision, Auto ML text)
- Conversational experiences (e.g., Dialogflow)

Google Cloud Platform provides several services that use pretrained machine learning models to help developers build and deploy intelligent services more quickly. The services are broadly grouped into the following categories:

- Sight
- Conversation
- Language
- Structured data

These services are available through APIs or Cloud AutoML services. Cloud AutoML uses the APIs to provide easier-to-use services such as AutoML Vision. Those services are described in Chapter 9, "Deploying Machine Learning Pipelines." This chapter focuses on the structure and use of the intelligence APIs.

Sight

GCP has two APIs for working with sight-related intelligence: Vision AI and Video AI. There is some similarity between the services. For example, both services provide functions to identify objects and filter content. In addition, the Video Intelligence AI has video-specific features, such as tracking objects across frames and transcribing audio.

VISION AI

The Vision AI API is designed to analyze images and identify text using OCR, to enable the search of images, and to filter explicit images. Images are sent to Vision AI by specifying a URI path to an image or by sending the image data as Base64-encoded text.

There are three options for calling the Vision AI API: Google-supported client libraries, REST, and gRPC. Google recommends using the client libraries, which are available for C#, Go, Java, Node.js, Python, PHP, and Ruby.

For each image sent to the API, the following operations can be performed:

- Detecting text in images
- Detecting handwriting in images
- Detecting text in PDF, TIFF, and other types of files
- Detecting faces
- Detecting hints for suggested vertices for a cropped region of an image

- Detecting image properties
- Detecting landmarks
- Detecting logos
- Detecting multiple objects
- Detecting explicit content (Safe Search)
- Detecting web entities and pages

The following code from the Google documentation shows an example of detecting labels in a JPEG file using Python and the Vision AI API. The script starts with importing io, os, and vision libraries, and then it instantiates a client. A variable is created to hold the filename (`file_name`), the file is opened, and the content of the file is read into a variable (`content`). The content is mapped to a Vision AI API image data structure and then passed to the label detection service, which returns a list of labels. The script ends with printing the list of labels.

```
import io
    import os
    # Imports the Google Cloud client library
    from google.cloud import vision
    from google.cloud.vision import types

    # Instantiates a client
    client = vision.ImageAnnotatorClient()

    # The name of the image file to annotate
    file_name = os.path.abspath('resources/wakeupcat.jpg')

    # Loads the image into memory
    with io.open(file_name, 'rb') as image_file:
        content = image_file.read()
    image = types.Image(content=content)

    # Performs label detection on the image file
    response = client.label_detection(image=image)
    labels = response.label_annotations
    print('Labels:')
    for label in labels:
        print(label.description)
```

Source: https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/vision/cloud-client/quickstart/quickstart.py

Other functions in the Vision AI API function in similar ways. For example, to detect landmarks in an image, you could replace the two lines of label detection from the previous snippet with the following:

```
response = client.landmark_detection(image=image)
Landmarks = response.landmark_annotations
```

Vision AI also provides for batch processing. This is done by using the Vision AI API to submit an offline, or asynchronous, request. That process starts a long-running job that does not immediately return a response. When the job completes, annotations are stored in a file in a Cloud Storage bucket. Up to 2,000 images can be submitted with a single API call. Multiple features, such as labels, landmarks, and logos, can be detected with a single API call.

VIDEO AI

Video AI provides models that can extract metadata; identify key persons, places, and things; and annotate video content. This service has pretrained models that automatically recognize objects in videos. Specifically, this API can be used to perform the following:

- Identifying objects, locations, activities, animal species, products, and so on
- Detecting shot changes
- Detecting explicit content
- Tracking objects
- Detecting text
- Transcribing videos

Videos are sent to the Video AI API by specifying a URI path to a video or by encoding the image data as a Base64 text and passing it into the content field of the request when using the REST API. The gRPC client can accept binary data directly. Google recommends embedding Base64-encoded files into a variable in code and passing that variable into API function calls.

The following sample code from the Google documentation shows an example of invoking the Video AI API for label detection. The script imports libraries and creates a video client instance, and it specifies the label detection operation and a URI to the video file to process. The video-processing operation is invoked with a timeout. When the operation finishes, a set of segment labels is returned. For each segment label, the script prints entity descriptions and categories. The script also displays the time range in the video of each segment, along with confidence scores for the predictions.

```
from google.cloud import videointelligence
    video_client = videointelligence.VideoIntelligenceServiceClient()
    features = [videointelligence.enums.Feature.LABEL_DETECTION]
    operation = video_client.annotate_video(
```

```
        'gs://cloud-samples-data/video/cat.mp4', features=features)
    print('\nProcessing video for label annotations:')

    result = operation.result(timeout=120)
    print('\nFinished processing.')

    # first result is retrieved because a single video was processed
    segment_labels = result.annotation_results[0].segment_label_annotat
    for i, segment_label in enumerate(segment_labels):
        print('Video label description: {}'.format(
            segment_label.entity.description))
        for category_entity in segment_label.category_entities:
            print('\tLabel category description: {}'.format(
                category_entity.description))

        for i, segment in enumerate(segment_label.segments):
            start_time = (segment.segment.start_time_offset.seconds +
                          segment.segment.start_time_offset.nanos / 1e9
            end_time = (segment.segment.end_time_offset.seconds +
                        segment.segment.end_time_offset.nanos / 1e9)
            positions = '{}s to {}s'.format(start_time, end_time)
            confidence = segment.confidence
            print('\tSegment {}: {}'.format(i, positions))
            print('\tConfidence: {}'.format(confidence))
        print('\n')
```

Source: https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/video/cloud-client/quickstart/quickstart.py

The Video AI API also includes a service for transcribing audio tracks in a video. That service supports the following:

- Ability to include up to 30 alternative translations for a word, listed in descending order of confidence
- Profanity filtering
- Transcription hints, which are unusual phrases used in the audio that may be otherwise difficult to transcribe
- Audio track detection
- Support for identifying multiple speakers in a video
- Automatically adding punctuation

This API supports several video formats, including MOV, MPEG4, MP4, AVI, and formats that can be decoded from FFmpeg, an open source suite of libraries supporting cross-platform use of video, audio, and multimedia files.

Conversation

Three APIs support conversation services:

- Dialogflow
- Cloud Text-to-Speech API
- Cloud Speech-to-Text API

Dialogflow is used for creating conversational interfaces, whereas the other services can be used for interactive or batch processing that transforms speech to text and text to speech.

DIALOGFLOW

Dialogflow is used for chatbots, interactive voice response (IVR), and other dialogue-based interactions with human speech. This service is based on natural language–understanding technology that is used to identify entities in a conversation and extract numbers, dates, and time, as well as custom entities that can be trained using examples. Dialogflow also provides prebuilt agents that can be used as templates.

A *Dialogflow Agent* is a virtual agent that participates in a conversation. Conversations can be audio or text based. Agents can be configured with the following:

- Agent settings for language options and behavior control
- Intents for categorizing a speaker's, or end user's, intentions
- Entities to identify and extract data from interactions
- Knowledge to parse documents, such as FAQs
- Integrations for applications that process end-user interactions
- Fulfillment to connect the service to integrations

A key component of Dialogflow is intents. *Intents* categorize a speaker's intention for a single statement in a conversation. The speaker's statement is assigned an intent classification. For example, a question about the weather may be mapped to a forecast intent, which would then be configured to process that statement to extract information, such as the time and location of the desired weather forecast. Intents are structures that include the following:

- Training phrases
- Actions
- Parameters to support extraction
- Responses to provide the speaker with answers

Dialogflow also supports contexts to model natural language contexts. *Contexts* help match intent. Contexts are used to analyze anaphora, which

are words that refer to other entities. For example, in the phrase "they went to the store," *they* refers to some group of people presumably mentioned earlier in the conversation. Context is used to resolve that reference to the entities mentioned earlier.

Dialogflow is accessible through REST, gRPC, and client libraries. Client libraries are available for C#, Go, Java, Node.js, PHP, Python, and Ruby.

CLOUD TEXT-TO-SPEECH API

GCP's Cloud Text-to-Speech API maps natural language texts to humanlike speech. The API works with more than 30 languages and has more than 180 humanlike voices. The service is based on speech synthesis technology called WaveNet, which is a deep generative model developed by DeepMind.

The API works with plain text or Speech Synthesis Markup Language (SSML) and audio files, including MP3 and WAV files. To generate speech, you call a synthesize function of the API. That function returns a Base64-encoded string that has to be decoded into an audio file. Linux users can use the Base64 command-line utility to perform that conversion.

One of the parameters needed for synthesis is a voice specification. The voices vary by language, gender, and, in some languages, dialects, such as French as spoken in France versus French as spoken in Canada. Supported languages include Arabic, Czech, Danish, Dutch, English, French, Greek, Hindi, Hungarian, Indonesian Italian, Korean, Mandarin Chinese, Norwegian, Polish, Portuguese, Swedish, Turkish, and Vietnamese. These are available in standard voice or WaveNet voice, which is higher quality. WaveNet synthesis costs more than standard.

Another synthesis parameter is the device specification. Cloud Text-to-Speech can optimize the generated speech for particular devices, such as wearable devices, headphones, small Bluetooth speakers, and large home entertainment devices.

CLOUD SPEECH-TO-TEXT API

The Cloud Speech-to-Text API is used to convert audio to text. The service is based on deep learning technology and supports 120 languages and variants. The service can be used for transcribing audio files as well as for supporting voice-activated interfaces. Cloud Speech-to-Text automatically detects the language being spoken. This feature is in beta, but it is already available for a large number of languages. Generated text can be returned as a stream of text or in batches as a text file.

The service has several pretrained models that are optimized for particular kinds of speech, including the following:

- Voice commands
- Phone calls
- Video
- Default voice for other domains

Other features of the service include noise handling; automatic punctuation; speaker diarization, which identifies the speaker of each utterance; and the ability to handle streaming audio.

Google has several recommendations for best results. Audio should be captured at a sampling rate of 16,000 Hz or higher. Use a lossless codec, such as FLAC or LINEAR16, for recording and transmitting audio. When recording multiple individuals, use a separate channel for each individual. If the speakers are using a specialized vocabulary, use word and phrase hints to improve accuracy. Finally, for short queries or commands, use the `StreamingRecognize` function with `single_utterance` set to `true`.

The Cloud Speech-to-Text API can be called using client libraries. You can also use the `gcloud ml speech` command from the command line.

Language

GCP has two APIs to support language processing: a translation and an analysis service.

TRANSLATION

Google's translation technology is available for use through the Cloud Translation API. The basic version of this service, Translation API Basic, enables the translation of texts between more than 100 languages. An advanced API, Translation API Advanced, is also available that supports customization for domain-specific and context-specific terms and phrases.

Translation API Basic can translate text and HTML content between languages. It can automatically detect languages, so users do not need to specify the source language. The basic API supports REST but not gRPC. Translation API Basic has several client APIs, including ones for C#, Go, Java, Python, Node.js, PHP, and Ruby.

Translation API Advanced has most of the features of Translation API Basic, plus support for glossaries, batch translations, custom models, and a gRPC API.

The following code example shows a Python function for translating a text. This function begins by importing libraries and creating an instance

of a translation client. Next, the text is decoded to UTF-8 if needed. The call to the function `translate_client.translate` does the actual translation to the target languages, which are passed as parameters. The function finishes with printing results of the translation and related data.

```
def translate_text(target, text):
    """Translates text into the target language.

    Target must be an ISO 639-1 language code.
    See https://g.co/cloud/translate/v2/translate-reference#supported_l

    """
    from google.cloud import translate_v2 as translate
    translate_client = translate.Client()

    if isinstance(text, six.binary_type):
        text = text.decode('utf-8')

    # Text can also be a sequence of strings, in which case this method
    # will return a sequence of results for each text.
    result = translate_client.translate(
        text, target_language=target)

    print(u'Text: {}'.format(result['input']))
    print(u'Translation: {}'.format(result['translatedText']))
    print(u'Detected source language: {}'.format(
        result['detectedSourceLanguage']))
```

Source: https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/translate/cloud-client/snippets.py

Batch translations are also supported by Translation API Advanced for HTML and plain-text files. Source files are located in a Cloud Storage bucket, and output files are written to a different Cloud Storage bucket.

In addition to translating text, Google Cloud Language provides an API for text analysis, referred to as Natural Language.

NATURAL LANGUAGE

The Natural Language API uses machine learning–derived models to analyze texts. With this API, developers can extract information about people, places, events, addresses, and numbers, as well as other types of entities. The service can be used to find and label fields within semi-structured documents, such as emails. It also supports sentiment analysis.

The Natural Language API has a set of more than 700 general categories, such as sports and entertainment, for document classification. It can also be combined with other machine learning services, like Speech-to-Text API, to extract information from audio content.

For more advanced users, the service performs syntactic analysis that provides parts of speech labels and creates parse trees for each sentence. Users of the API can specify domain-specific keywords and phrases for entity extraction and custom labels for content classification. It is also possible to use spatial structure understanding to improve the quality of entity extraction. For example, you may be able to take advantage of the layout of text to improve custom entity extraction.

The API can support working with up to 5,000 classification labels and 1 million documents. Documents may be up to 10 MB in size.

The Natural Language API includes functions to perform a variety of text analysis operations, including the following:

- Identifying entities
- Analyzing sentiment associated with each entity
- Analyzing sentiment of the overall text
- Generating syntactic analysis, including parts-of-speech tags and syntax trees
- Classifying documents into categories

  Here are some example high-level content categories:

- Arts and Entertainment
- Autos and Vehicles
- Business and Industrial
- Computer and Electronics
- Food and Drink
- Games
- Health
- People and Society
- Law and Government
- News

  Each of these high-level categories have finer-grained categories as well; here is a small set of examples:

- /Business & Industrial/Chemicals Industry/Plastics & Polymer
- /Computers & Electronics/Computer Hardware/Computer Drives & Storage
- /Food & Drink/Food/Grains & Pasta

- /Games/Table Games/Billiards
- /Health/Medical Facilities & Services/Medical Procedures
- /People & Society/Kids & Teens/Children's Interests
- /Law & Government/Public Safety
- /News/Business News/Financial Markets News

For applications that could benefit from text classification and information extraction, the Natural Language API can provide general functionality and pretrained models that can be customized for domain-specific texts.

Structured Data

GCP has three machine learning services for structured data: AutoML Tables, which was described earlier in Chapter 9, and the Recommendations AI API and Cloud Inference API.

RECOMMENDATIONS AI API

The *Recommendations AI API* is a service for suggesting products to customers based on their behavior on the user's website and the product catalog of that website. The service builds a recommendation model specific to the site. Recommendations AI is currently in beta.

The product catalog contains information on products that are sold to customers, such as names of products, prices, and availability. End-user behavior is captured in logged events, such as information about what customers search for, which products they view, and which products they have purchased.

The two primary functions of the Recommendations AI API are ingesting data and making predictions. Data is ingested using either the `catalogItems.create` function for individual items or the `catalogItems.import` method for bulk loading. Google recommends providing as much detail as possible and updating the product catalog information as needed to keep the model up to date with the actual product catalog.

Customer activity records also need to be ingested. Users events that are useful for the recommendation service include clicking a product, adding an item to a shopping cart, and purchasing a product. Customer events can be recorded using a JavaScript pixel placed on the website to record actions, using the Google Tag Manager to tag and record events, or sending events directly to the Recommendations AI API using the `userEvents.write` method.

In addition to loading data, users of the Recommendations AI API will need to specify some configuration parameters to build a model, including recommendation types.

Recommendation types are one of the following:

- **Others you may like:** These are additional items that the customer is most likely to purchase.
- **Frequently bought together:** These are items often purchased during the same session.
- **Recommended for you:** This predicts the next product with which the customer is likely to engage.
- **Recently viewed:** This is simply a set of catalog IDs of products with which the customer has recently interacted.

You will also need to specify an optimization objective. There are three such objectives:

- **Click-through rate (CTR):** This is the default optimization, and it maximizes the likelihood that the user engages the recommendation.
- **Revenue per order:** This is the default objective for the frequently bought together recommendation type, and it cannot be used with other types.
- **Conversion rate:** This rate maximizes the likelihood that the user purchases the recommended product.

The Recommendations AI API tracks metrics to help you evaluate the performance of recommendations made. The metrics include the following:

- **Total revenue from all recorded purchase events:** This is the sum of all revenue from all purchases.
- **Recommender-engaged revenue:** This is the revenue from purchase events that include at least one recommended item.
- **Recommendation revenue:** This is the revenue from recommended items.
- **Average order value (AOV):** This is the average of orders from all purchase events.
- **Recommender-engaged AOV:** This is the average value of orders that include at least one recommended item.
- **Click-through rate:** This is the number of product views from a recommendation.
- **Conversion rate:** This is the number of times that an item was added to a cart divided by the total number of recommendations.

- **Revenue from recommendations:** This is the total revenue from all recommendations.

The Recommendations AI API is tailored for interacting with customers on an e-commerce site. The Cloud Inference API is a machine learning service designed to help analyze time-series data.

The Recommendations AI API is in beta release as of this writing.

CLOUD INFERENCE API

Many activities and operations of interest to business can be captured in time-series data. This can include tracking the number of customers visiting a website or physical store in some time period, collecting sensor measurements from manufacturing machinery, and collecting telemetry data from fleet vehicles. The Cloud Inference API provides real-time analysis of time-series data. The Cloud Inference API is currently in alpha.

The Cloud Inference API provides for processing time-series datasets, including ingesting from JSON formats, removing data, and listing active datasets. It also supports inference queries over datasets, including correlation queries, variation in frequency over time, and probability of events given evidence of those events in the dataset.

The Cloud Inference API is in alpha release as of this writing.

Exam Essentials

Understand the functionality of the Vision AI API. The Vision AI API is designed to analyze images and identify text, enable the search of images, and filter explicit images. Images are sent to the Vision AI API by specifying a URI path to an image or by sending the image data as Base64-encoded text. There are three options for calling the Vision AI API: Google-supported client libraries, REST, and gRPC.

Understand the functionality of the Video Intelligence API. The Video Intelligence API provides models that can extract metadata; identify key persons, places, and things; and annotate video content. This service has pretrained models that automatically recognize objects in videos. Specifically, this API can be used to identify objects, locations, activities, animal species, products, and so on, and detect shot changes, detect explicit content, track objects, detect text, and transcribe videos.

Understand the functionality of Dialogflow. Dialogflow is used for chatbots, interactive voice response (IVR), and other dialogue-based interactions with human speech. The service is based on natural language–understanding technology that is used to identify entities in a conversation and extract numbers, dates, and time, as well as custom entities that can be trained using examples. Dialogflow also provides prebuilt agents that can be used as templates.

Understand the functionality of the Cloud Text-to-Speech API. GCP's Cloud Text-to-Speech API maps natural language texts to human-like speech. The API works with more than 30 languages and has more than 180 humanlike voices. The API works with plain-text or Speech Synthesis Markup Language (SSML) and audio files, including MP3 and WAV files. To generate speech, you call a synthesize function of the API.

Understand the functionality of the Cloud Speech-to-Text API. The Cloud Speech-to-Text API is used to convert audio to text. This service is based on deep learning technology and supports 120 languages and variants. The service can be used for transcribing audios as well as for supporting voice-activated interfaces. Cloud Speech-to-Text automatically detects the language being spoken. Generated text can be returned as a stream of text or in batches as a text file.

Understand the functionality of the Cloud Translation API. Google's translation technology is available for use through the Cloud Translation API. The basic version of this service, Translation API Basic, enables the translation of texts between more than 100 languages. There is also an advanced API, Translation API Advanced, which supports customization for domain-specific and context-specific terms and phrases.

Understand the functionality of the Natural Language API. The Natural Language API uses machine learning–derived models to analyze texts. With this API, developers can extract information about people, places, events, addresses, and numbers, as well as other types of entities. The service can be used to find and label fields within semi-structured documents, such as emails. It also supports sentiment analysis. The Natural Language API has a set of more than 700 general categories, such as sports and entertainment, for document classification. For more advanced users, the service performs syntactic analysis that provides parts of speech labels and creates parse trees for each sentence. Users of the API can specify domain-specific keywords and phrases for entity extraction and custom labels for content classification.

Understand the functionality of the Recommendations AI API. The Recommendations AI API is a service for suggesting products to customers based on their behavior on the user's website and the product catalog of that website. The service builds a recommendation model specific to the site. The product catalog contains information on products that are sold to customers, such as names of products, prices, and availability. End-user behavior is captured in logged events, such as information about what customers search for, which products they view, and which products they have purchased. There are two primary functions the Recommendations AI API: ingesting data and making predictions.

Understand the functionality of the Cloud Inference API. The Cloud Inference API provides real-time analysis of time-series data. The Cloud Inference API provides for processing time-series datasets, including ingesting from JSON formats, removing data, and listing active datasets. It also supports inference queries over datasets, including correlation queries, variation in frequency over time, and probability of events given evidence of those events in the dataset.

Review Questions

You can find the answers in the appendix.

1. You are building a machine learning model to analyze unusual events in traffic through urban areas. Your model needs to distinguish cars, bikes, pedestrians, and buildings. It is especially important that the model be able to identify and track moving vehicles. Video will be streamed to your service from cameras mounted on traffic lights. What GCP service would you use for the object analysis and tracking?

1. Cloud Video Intelligence API
2. Cloud Vision API
3. Cloud Inference API
4. Cloud Dataflow

2. A startup is building an educational support platform for students from ages 5–18. The platform will allow teachers to post assignments and conduct assessments. Students will be able to upload content, including text and images. The founder of the startup wants to make sure that explicit images are not uploaded. What GCP service would you use?

1. Cloud Video Intelligence API
2. Cloud Vision API
3. Cloud Inference API
4. Cloud Dataprep

3. You are using the Cloud Vision API to detect landmarks in images. You are using the batch processing with asynchronous requests. The source images for each batch is in a separate Cloud Storage bucket. There are between 1 and 5,000 images in each bucket. Each batch request processes one bucket. All buckets have the same access controls. Sometimes, the operations succeed and sometimes they fail. What could be the cause of the errors?

1. Cloud Video Intelligence API
2. Some buckets have more than 2,000 images.
3. There is an issue with IAM settings.
4. Images have to be uploaded directly from a device, not a Cloud Storage bucket.

4. Your team is building a chatbot to support customer support. Domain experts from the customer support team have identified several kinds of questions that the system should support, including questions about returning products, getting technical help, and asking for product recommendations. You will use Dialogflow to implement the chatbot. What component of Dialogflow will you configure to support the three question types?

1. Entities
2. Fulfillments
3. Integrations
4. Intents

5. A developer asks for your help tuning a text-to-speech service that is used with a health and wellness app. The app is designed to run on watches and other personal devices. The sound quality is not as good as the developer would like. What would you suggest trying to improve the quality of sound?

1. Change the device specification to optimize for a wearable device
2. Change from standard to WaveNet-quality voice
3. Encode the text in Base64
4. Options A and B

5. Options A, B, and C

6. A developer asks for your help tuning a speech-to-text service that is used to transcribe text recorded on a mobile device. The quality of the transcription is not as good as expected. The app uses LINEAR16 encoding and a sampling rate of 12,000 Hz. What would you suggest to try to improve the quality?

1. Use WaveNet option
2. Increase the sampling rate to at least 16,000 Hz
3. Use Speech Synthesis Markup Language to configure conversion parameters
4. Options A and B

7. You have developed a mobile app that helps travelers quickly find sites of interest. The app uses the GCP Translation service. The initial release of the app used the REST API, but adoption has grown so much that you need higher performance from the API and plan to use gRCP instead. What changes do you need to make to the way that you use the Translation service?

1. Use the WaveNet option
2. Use Translation API Basic
3. Use Translation API Advanced
4. Option A or B

8. You are experimenting with the GCP Translation API. You have created a Jupyter Notebook and plan to use Python 3 to build a proof-of-concept system. What are the first two operations that you would execute in your notebook to start using the Translation API?

1. Import Translation libraries and create a translation client
2. Create a translation client and encode text in UTF-8
3. Create a translation client, and set a variable to `TRANSLATE` to pass in as a parameter to the API function call
4. Import Translation libraries, and set a variable to `TRANSLATE` to pass in as a parameter to the API function call

9. You have been hired by a law firm to help analyze a large volume of documents related to a legal case. There are approximately 10,000 documents ranging from 1 to 15 pages in length. They are all written in English. The lawyers hiring you want to understand who is mentioned in each document so that they can understand how those individuals worked together. What functionality of the Natural Language API would you use?

1. Identifying entities
2. Analyzing sentiment associated with each entity
3. Analyzing sentiment of the overall text
4. Generating syntactic analysis

10. As a founder of an e-commerce startup, you are particularly interested in engaging with your customers. You decide to use the GCP Recommendations AI API using the “others you may like” recommendation type. You want to maximize the likelihood that users will engage with your recommendations. What optimization objective would you choose?

1. Click-through rate (CTR)
2. Revenue per order
3. Conversation rate
4. Total revenue

11. Your e-commerce startup has been growing rapidly since its launch six months ago. You are starting to notice that the rate of revenue growth is slowing down. Your board of directors is asking you to develop a strategy to increase revenue. You decide to personalize each customer’s experience. One of the ways in which you plan to implement your strategy is by showing customers products that they are likely to interact with next. What recommendation type would you use?

1. Others you may like
2. Frequently bought together
3. Recommended for you
4. Recently viewed

12. You work for an enterprise with a large fleet of vehicles. The vehicles are equipped with several sensors that transmit data about fuel utilization, speed, and other equipment operating characteristics. The chief of operations has asked you to investigate the feasibility of building a predictive maintenance application that can help identify breakdowns before they occur. You decide to prototype an anomaly detection model as a first step.

You want to build this as quickly as possible, so you decide to use a machine learning service. Which GCP service would you use?

1. Cloud Inference API
2. AutoML Tables
3. AutoML Vision
4. Cloud Anomaly Detection API

Support / Sign Out

PREV
Chapter 11 Measuring, Monitoring, and Troubleshooting Ma…

NEXT
Appendix Answers to Review Questions