AML Final

Chapter 5

Explain the file format of a dataset that ingest in the model for training

- **.csv**

In general, most common file format used is CSV.

Many Amazon SageMaker algorithms support training with data in CSV format.

For use with Amazon SageMaker, CSV files can't have a header record, and the target variable must be in the first column.

- **.rec**
- **Text**
- **JSON lines**
- **Protobuf recordIO**

Amazon SageMaker algorithms work best when you use the optimized protobuf recordIO format for the training data.

In the protobuf recordIO format, Amazon SageMaker converts each instance in the dataset into a binary representation as a set of 4-byte floats, then loads it in the protobuf values field.

This format enables you to take advantage of Pipe mode when you train the algorithms that support it.

In Pipe mode, your training job streams data directly from Amazon S3. In contrast, File mode loads all your data from Amazon S3 to the training instance volumes. Streaming with Pipe mode can provide faster start times for training jobs and better throughput.

You can also reduce the size of the Amazon Elastic Block Store (Amazon EBS) volumes for your training instances because pipe mode needs only enough disk space to store your final model artifacts.

In contrast, File mode needs disk space to store both your final model artifacts and your full training dataset.

Why splitting and holding out data is required? Explain with an example.

- Evaluating a model with the same data that it trained on can lead to overfitting.
- Overfitting is where your model learns the particulars of a dataset too well.
- It essentially memorizes the training data, instead of learning the relationships between features and labels.

Thus, the model can't use those relationships and patterns to apply to new data in the future.

How K-fold cross-validation can help you? Explain with an example.

- For a small dataset, you can use k-fold cross-validation to use as much data as possible.

- This technique provides good metrics to choose which model is better.
- K-fold cross-validation randomly partitions the data into K different segments.
- You apply different models to different pieces of the validation dataset to have some idea about how well the models perform.

As a result, you use all the data for training. The cross-validation results are averaged to give you an idea about the performance of the model.

Explain the deployment options in AWS Sagemaker.

- You can deploy your model in two ways.
- For single predictions, deploy your model with Amazon SageMaker hosting services.
- Applications can call the API at the endpoint to make predictions.
- With this method, you can scale the number of compute instances up or down, based on demand. (Amazon SageMaker deploys multiple compute instances that run your model behind a load-balanced endpoint.)
- To get predictions for an entire dataset, use Amazon SageMaker batch transform.
- Instead of deploying and maintaining a permanent endpoint, Amazon SageMaker spins up your model and performs the predictions for the entire dataset that you provide.
- Performing batch predictions when you test the model is useful because you can quickly run your entire validation set against the model.
- (It stores the results in Amazon S3 before it shuts down and terminates the compute instances.)

Chapter 6

1.Explain the "Evaluate Model" phase of the ML pipeline with an example.

Model evaluation is the process that uses some metrics which help us to analyze the performance of the model. As we all know that model development is a multi-step process and a check should be kept on how well the model generalizes future predictions. Therefore evaluating a model plays a vital role so that we can judge the performance of our model. The evaluation also helps to analyze a model's key weaknesses. There are many metrics like Accuracy, Precision, Recall, F1 score, Area under Curve, Confusion Matrix, and Mean Square Error. Cross Validation is one technique that is followed during the training phase and it is a model evaluation technique as well.

Cross Validation is a method in which we do not use the whole dataset for training. In this technique, some part of the dataset is reserved for testing the model. There are many types of Cross-Validation out of which K Fold Cross Validation is mostly used. In K Fold Cross Validation the original dataset is divided into k subsets. The subsets are known as folds. This is repeated k times where 1 fold is used for testing purposes. Rest k-1 folds are used for training the model. So each data point acts as a test subject for the model as well as acts as the training subject. It is seen that this technique generalizes the model well and reduces the error rate

Holdout is the simplest approach. It is used in neural networks as well as in many classifiers. In this technique, the dataset is divided into train and test datasets. The dataset is usually divided into ratios like 70:30 or 80:20. Normally a large percentage of data is used for training the model and a small portion of the dataset is used for testing the model.

Accuracy is defined as the ratio of the number of correct predictions to the total number of predictions. This is the most fundamental metric used to evaluate the model.

However, Accuracy has a drawback. It cannot perform well on an imbalanced dataset. Suppose a model classifies that the majority of the data belongs to the major class label. It yields higher accuracy. But in general, the model cannot classify on minor class labels and has poor performance.

Precision is the ratio of true positives to the summation of true positives and false positives. It basically analyses the positive predictions.

The drawback of Precision is that it does not consider the True Negatives and False Negatives.

Recall is the ratio of true positives to the summation of true positives and false negatives. It basically analyses the number of correct positive samples.

The drawback of Recall is that often it leads to a higher false positive rate.

The F1 score is the harmonic mean of precision and recall. It is seen that during the precision-recall trade-off if we increase the precision, recall decreases and vice versa. The goal of the F1 score is to combine precision and recall.

2.Explain the “Confusion matrix” with an example.

•Suppose that you have a simple image recognition model that labels data as either cat or not cat.

•After the model is trained, you can use the test dataset that you held back to perform predictions.

•To help examine the performance of the model, you can compare the predicted values with the actual values.

True positive

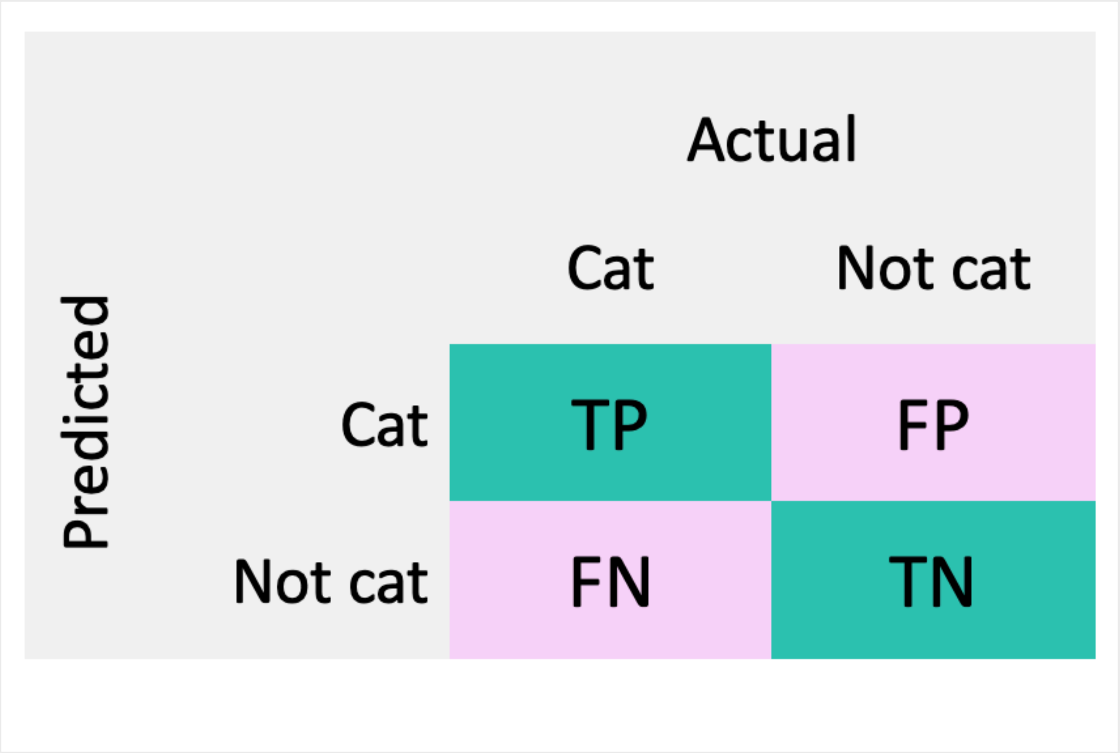
True negative

False Positive

False negative

Sensitivity: The first metric is sensitivity, which is sometimes referred to as hit rate or true positive rate. This metric is the percentage of positive

identifications TP/TP+FN
Specificity: sometimes referred to as selectivity or true negative rate, is the percentage of negatives that were correctly identified TN/TN+FP



		Actual	
		Cat	Not cat
Predicted	Cat	TP	FP
	Not cat	FN	TN

3.How the "Sensitivity" and "Specificity" can help with model tuning? with an example.

Sensitivity: The first metric is sensitivity, which is sometimes referred to as hit rate or true positive rate. This metric is the percentage of positive identifications TP/TP+FN
Specificity: sometimes referred to as selectivity or true negative rate, is the percentage of negatives that were correctly identified TN/TN+FP
Let's consider an example of a medical test for a disease:

- **Positive Cases (Disease Present)**: Patients who have the disease.
- **Negative Cases (Disease Absent)**: Patients who do not have the disease.

Suppose we're building a machine learning model to predict the presence or absence of this disease using a set of features.
Example:

  - We have a dataset of 1000 patients:
  - 100 patients have the disease (positive cases).
  - 900 patients do not have the disease (negative cases).

Now, let's say our model predicts disease presence for 150 patients, out of which:

- 120 predictions are correct (True Positives).
- 30 predictions are incorrect (False Positives).

Additionally, for the disease-absent cases:

- The model correctly predicts 850 cases as disease-absent (True

Negatives).
- 50 cases are predicted as disease-present but are actually disease-absent (False Negatives).

Calculating Sensitivity and Specificity:

- Sensitivity = True Positives / (True Positives + False Negatives) = 120 / (120 + 50) = 0.706 (or 70.6%)
- Specificity = True Negatives / (True Negatives + False Positives) = 850 / (850 + 30) = 0.966 (or 96.6%)
- High sensitivity is desired when correctly identifying positive cases is crucial (e.g., disease detection). Tuning the model to increase sensitivity reduces false negatives, ensuring that fewer cases of the disease are missed.
- High specificity is crucial when correctly identifying negative cases is critical (e.g., fraud detection). Tuning the model to increase specificity reduces false positives, ensuring that fewer non-disease cases are incorrectly flagged as positive.

When tuning a model, you might adjust thresholds, feature selection, or algorithms to optimize either sensitivity or specificity based on the specific requirements of the problem domain. For instance, in the medical example, a model tuned for higher sensitivity might be preferred to ensure fewer cases of the disease are missed, even at the expense of higher false positives. Conversely, for fraud detection, higher specificity might be prioritized to reduce false alarms, even if it means missing some fraudulent cases.

4.How “Area under the curve-receiver operator curve (AUC-ROC)” can help to find out which model is better? with an example.

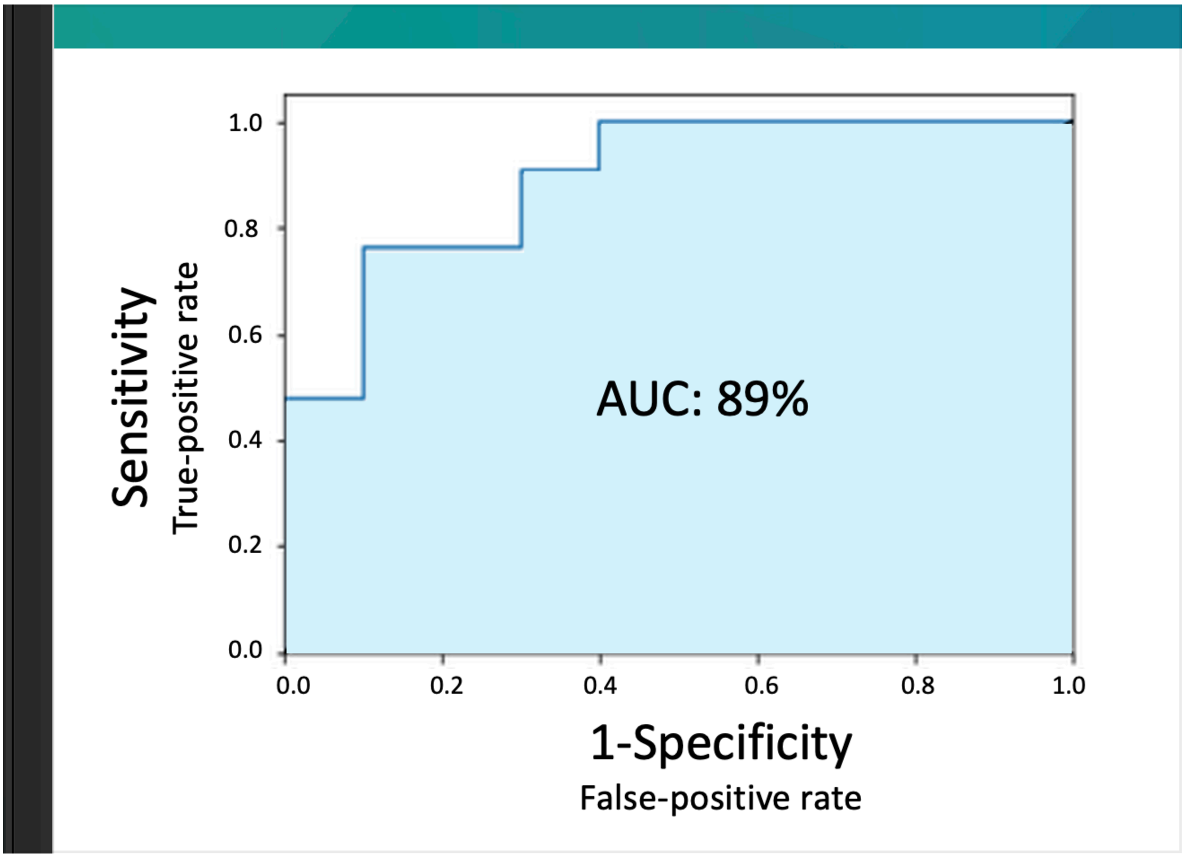
The Area Under the Curve - Receiver Operating Characteristic (AUC-ROC) curve is a popular evaluation metric used to assess the performance of machine learning classifiers, particularly in binary classification problems. It measures the ability of a model to distinguish between classes by plotting the true positive rate (Sensitivity) against the false positive rate (1 - Specificity) at various threshold settings.

The AUC-ROC score ranges from 0 to 1, where a score closer to 1 indicates better performance. An AUC-ROC of 0.5 suggests that the model performs no better than random chance (like tossing a coin), while an AUC-ROC of 1 signifies perfect performance.

Using AUC-ROC to Choose the Better Model:

- In this example, Model A with the higher AUC-ROC score of 0.90 is considered better at discriminating between positive and negative cases compared to Model B with an AUC-ROC score of 0.85.
- When comparing models, the one with the higher AUC-ROC score is generally preferred as it demonstrates better overall performance in binary classification tasks. However, it's essential to consider other factors and domain-specific requirements before finalizing a model

choice.



5.What are the Precision and F1 Score ? with an example.

Precision and F1 Score are evaluation metrics used to assess the performance of machine learning models, especially in binary classification problems where there are two classes: positive and negative.

Precision measures the accuracy of the positive predictions made by the model. It calculates the ratio of correctly predicted positive instances (True Positives) to the total predicted positive instances (True Positives + False Positives). TP/TP+FP

F1 Score is the harmonic mean of precision and recall (sensitivity). It provides a balance between precision and recall, especially when there's an uneven class distribution. F1 Score combines both metrics into a single value, where higher values indicate better model performance. F1 Score = 2 * (Precision * Recall)/(Precision + Recall)

6.Explain the "Tune Model" phase of the ML pipeline with an example.

The "Tune Model" phase in the machine learning pipeline involves optimizing the model's hyperparameters and fine-tuning its configuration to improve its performance on a given dataset. Hyperparameters are settings or configurations that are set before the training process and cannot be learned from the data, such as the learning rate in neural networks, the depth of a decision tree, or the regularization parameter in regression models.

Here's an example to illustrate the "Tune Model" phase using a hypothetical

scenario of tuning hyperparameters for a classification model:
Scenario:
Suppose you're working on a dataset containing information about customers, and your task is to predict whether a customer will subscribe to a service based on various features like age, income, and browsing behavior.

Model:
Let's consider using a Gradient Boosting Classifier as the model for this task.
Tune Model Phase Steps:

- **Select Initial Hyperparameters**: Start with default or common hyperparameters for the Gradient Boosting Classifier.
- **Split Data**: Split the dataset into training and validation sets. This allows you to train the model on one portion of the data and validate its performance on another.
- **Hyperparameter Grid Search**:
    - Define a grid of hyperparameters to explore. For example, consider tuning parameters like learning rate, maximum depth of trees, number of estimators, etc.
    - Use techniques like grid search or random search to systematically explore combinations of these hyperparameters.
- **Cross-Validation**:
    - Perform cross-validation on the training set using different hyperparameter combinations. This helps assess how well each set of hyperparameters generalizes to unseen data.
- **Evaluate Performance**:
    - For each set of hyperparameters, train the model on the training set and evaluate its performance on the validation set.
    - Metrics such as accuracy, precision, recall, F1 score, or area under the ROC curve can be used to evaluate model performance.
- **Select Best Hyperparameters**:
    - Identify the set of hyperparameters that result in the best performance on the validation set based on the chosen evaluation metric(s).
- **Final Model Training**:
    - Retrain the model using the entire training dataset with the selected optimal hyperparameters.

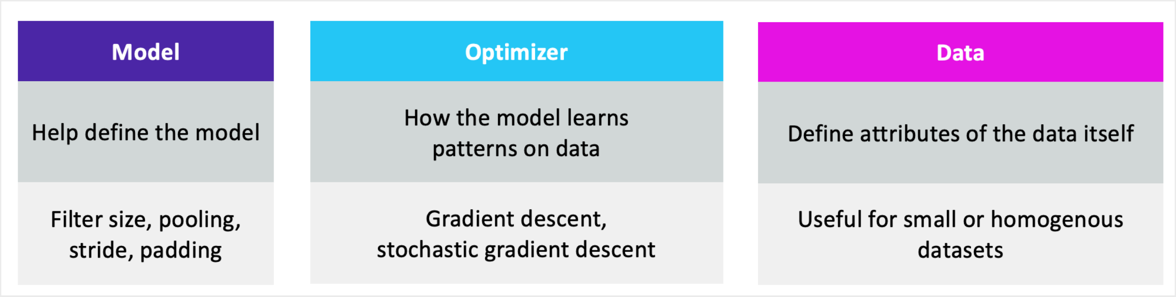
7.Describe categories of Hyperparameters.

•Hyperparameters have a few different categories. The first kind is **model hyperparameters**.
•For example, some neural-network-based models require you to define an architecture before you can start training them. The architecture includes a specific number of layers in the neural network and the activation functions that are used within.

•Suppose, it is a computer vision problem, in a neural network, additional attributes of the architecture must be defined. Such attributes include filter size, pooling, and stride or padding.

•The second kind is **optimizer hyperparameters**. These hyperparameters are related to how the model learns the patterns that are based on data. These types of hyperparameters include optimizers like gradient descent and stochastic gradient descent.
•They can also include optimizers that use momentum like Adam, or initialize the parameter weights by using methods such as Xavier initialization or He initialization.

•The third kind is data hyperparameters, which relate to the attributes of the data itself.
•They include attributes that define different data augmentation techniques, such as cropping or resizing for image-related problems.
•They are often used when you don't have enough data or enough variation in your data.



Model	Optimizer	Data
Help define the model	How the model learns patterns on data	Define attributes of the data itself
Filter size, pooling, stride, padding	Gradient descent, stochastic gradient descent	Useful for small or homogenous datasets

Chapter 7

1.What is Forecasting? Describe related use cases.
•Forecasting is an important area of machine learning.
•Predict future values that are based on historical data.
•Many of these opportunities involve a time component
•You can think of time series data as falling into two broad categories.
•The first type is univariate, which means that it has only one variable.
•The second type is multivariate, which means that it has more than one variable.
•In addition to these two categories, most time series datasets also follow one of the following patterns:
•Trends: Patterns that increase, decrease, or are stagnant
•Seasonal: Pattern that is based on seasons

- •Cyclical: Other repeating patterns
- •Irregular: Patterns that might appear to be random

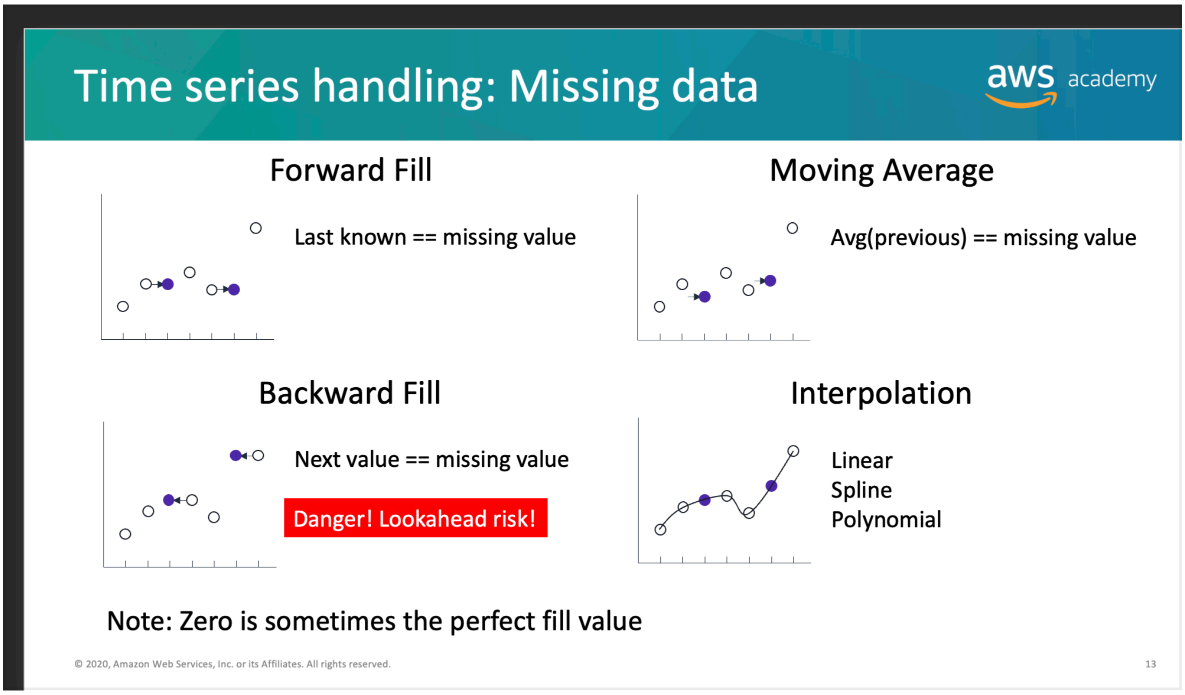
USE CASES

•Marketing applications, such as sales forecasting or demand projections.
•Inventory management systems to anticipate required inventory levels. Often, this type of forecast includes information about delivery times.
•Energy consumption to determine when and where energy is needed.
•Weather forecasting systems for governments, and commercial applications such as agriculture.

2.How "Time series handling for Missing data" can be done in real-world forecasting problems?

•A common occurrence in real-world forecasting problems is missing values in the raw data. Missing values make it harder for a model to generate a forecast.
•The primary example in retail is an out-of-stock situation in demand forecasting. If an item goes out of stock, the sales for the day will be zero. If the forecast is generated based on those zero sales values, the forecast will be incorrect.
•Missing values can occur because of no transaction, or possibly because of measurement errors. Maybe a service/ device that monitored certain data was not working correctly, or the measurement could not occur correctly.

•The missing data can be calculated in several ways:
•Forward fill – Uses the last known value for the missing value.
•Moving average – Uses the average of the last known values to calculate the missing value.
•Backward fill – Uses the next known value after the missing value. Be aware that it is a potential danger to use the future to calculate the past, which is bad for forecasting. This practice is known as lookahead, and it should be avoided.
•Interpolation – Essentially uses an equation to calculate the missing value.



Time series handling: Missing data

Forward Fill

Last known == missing value

Moving Average

Avg(previous) == missing value

Backward Fill

Next value == missing value

Danger! Lookahead risk!

Interpolation

Linear
Spline
Polynomial

Note: Zero is sometimes the perfect fill value

3.Describe features of any one forecasting algorithm of your choice.
(you can prepare an answer from the link https://docs.aws.amazon.com/forecast/latest/dg/aws-forecast-choosing-recipes.html

CNN-QR is a sequence-to-sequence (Seq2Seq) model for probabilistic forecasting that tests how well a prediction reconstructs the decoding sequence, conditioned on the encoding sequence.
The algorithm allows for different features in the encoding and the decoding sequences, so you can use a related time series in the encoder, and omit it from the decoder (and vice versa). By default, related time series with data points in the forecast horizon will be included in both the encoder and decoder. Related time series without data points in the forecast horizon will only be included in the encoder.
CNN-QR performs quantile regression with a hierarchical causal CNN serving as a learnable feature extractor.
To facilitate learning time-dependent patterns, such as spikes during weekends, CNN-QR automatically creates feature time series based on time-series granularity. For example, CNN-QR creates two feature time series (day-of-month and day-of-year) at a weekly time-series frequency. The algorithm uses these derived feature time series along with the custom feature time series provided during training and inference.

4.Write on any one “Amazon Forecast evaluation metric” with an example.
One essential evaluation metric used in Amazon Forecast is the Weighted Quantile Loss (WQL). It's a crucial metric for assessing the accuracy and reliability of quantile forecasts produced by the Amazon Forecast service.
Weighted Quantile Loss (WQL):
WQL measures the discrepancy between the predicted quantiles and the actual

values in a time series. It's particularly useful when dealing with probabilistic forecasting, where predictions are made at various quantile levels, providing a range of possible outcomes and associated probabilities.

5.Describe the Amazon forecasting phases, using the diagram.

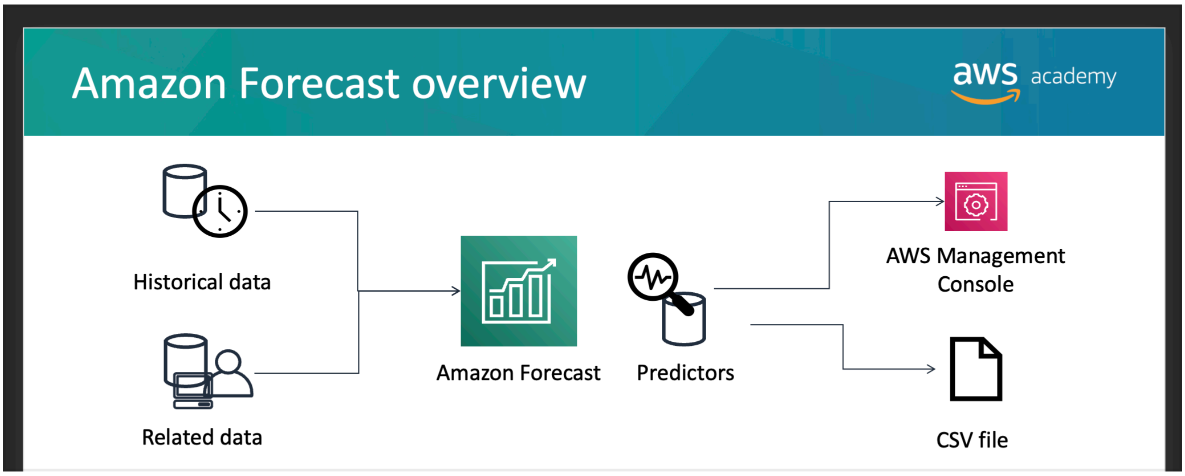
•When you generate forecasts, you can apply the ML development pipeline that you use throughout this course.

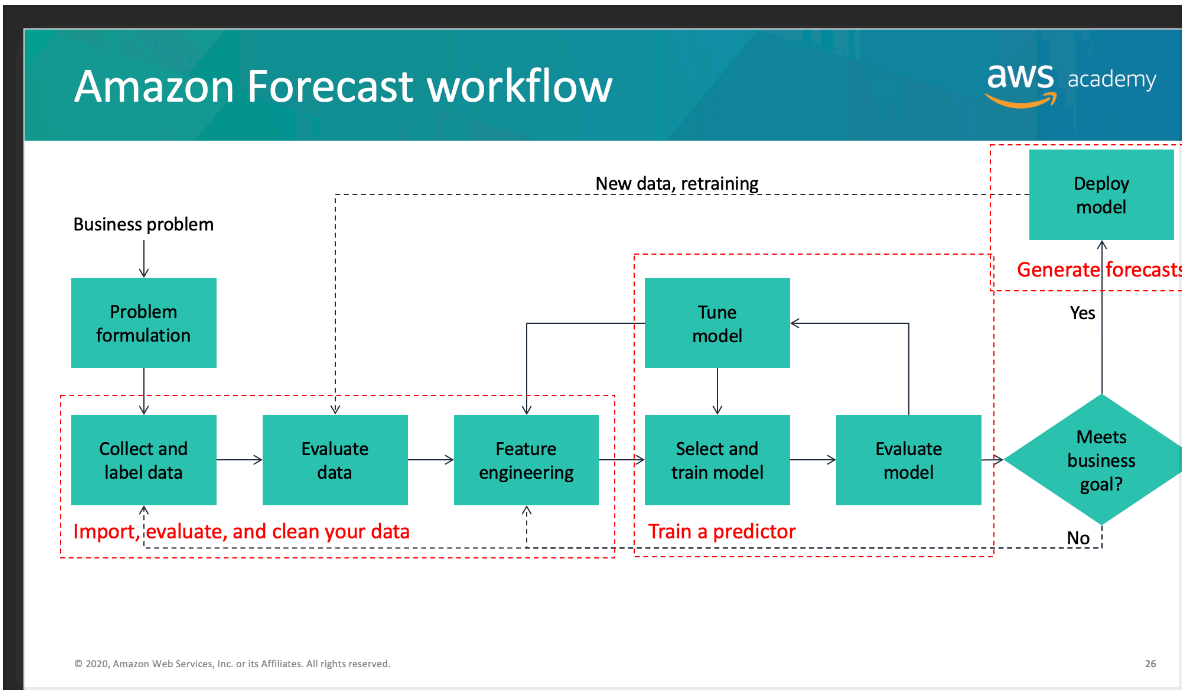
•Import your data – You must import as much data as you have—both historical data and related data. You should do some basic evaluation and feature engineering before you use the data to train a model.

•Train a predictor – To train a predictor, you must choose an algorithm. If you are not sure which algorithm is best for your data, you can let Amazon Forecast choose by selecting AutoML as your algorithm. You also must select a domain for your data, but if you're not sure which domain fits best, you can select a custom domain. Domains have specific types of data that they require.

•Generate forecasts – As soon as you have a trained model, you can use the model to make a forecast by using an input dataset group. After you generate a forecast, you can query the forecast, or you can export it to a Amazon S3 bucket. You also have the option to encrypt the data in the forecast before you export it.

The overall process for working with Amazon Forecast is to **1)** import historical and related data. **2)** Amazon Forecast inspects the data, identifies key data, and selects an appropriate algorithm. **3**) It uses the algorithm to train and optimize a custom model and produce a predictor. **4)** You create forecasts by applying the predictor to your dataset. **5)** Then, you can either retrieve these forecasts in the AWS console, or export the forecasts as comma-delimited files.





Amazon Forecast workflow

aws academy

New data, retraining

Business problem

Problem formulation

Collect and label data

Evaluate data

Feature engineering

Import, evaluate, and clean your data

Tune model

Select and train model

Evaluate model

Train a predictor

Meets business goal?

Yes

No

Deploy model

Generate forecasts

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

26

Chapter 8

1.What is NLP? Describe any two use cases of NLP.

- NLP is a broad term for a general set of business or computational problems that you can solve with machine learning.
- NLP systems predate ML.
- Two examples are speech-to-text on your old cell phone and screen readers.
- Many NLP systems now use some form of machine learning.
- NLP considers the hierarchical structure of language. Words are at the lowest layer of the hierarchy. A group of words makes a phrase. The next level up consists of phrases, which make a sentence, and ultimately, sentences convey ideas.

Some of the more common applications include:

- Search applications (such as Google and Bing)
- Human-machine interfaces (such as Alexa)
- Sentiment analysis for marketing or political campaigns
- Social research that is based on media analysis
- Chatbots to mimic human speech in applications

2.Describe the main challenges of NLP.

- Language is not precise.
- Words can have different meanings, which are based on the other words that

surround them (context). Often, the same words or phrases can have multiple meanings.
•For example, consider the term weather. You might be under the weather, which means that you are sick. However, there is wonderful weather outside means that the weather conditions outside are good.
•In another example, The phrase Oh, really? might convey surprise, disagreement, or many other meanings, depending on context and inflection.

•Some of the main challenges for NLP include:
•Discovering the structure of the text – One of the first tasks of any NLP application is to break the text into meaningful units, such as words, phrases, and sentences.
•Labeling data – After the system converts the text to data, the next challenge is to apply labels that represent the various parts of speech. Every language requires a different labeling scheme to match the language's grammar.
•Representing context – Because word meaning depends on context, any NLP system needs a way to represent the context. Because of the large number of contexts, converting context into a form that computers can understand is difficult.
•Applying grammar – Although grammar defines a structure for language, the application of grammar is nearly infinite. Dealing with the variation in how humans use language is a major challenge for NLP systems.

3.Explain the NLP model "Bag of words" with an example scenario.
•*Bag of words* is a vector model. Vector models convert each sentence or phrase into a vector, which is a mathematical object that records both directionality and magnitude.
•In the example, a simple sentence is converted into a vector where each word is recorded in terms of frequency. The word "is" has a value of 2 because it appears twice in the sentence.
•It is often used to classify documents into different categories. It is also used to derive attributes that feed into NLP applications, such as sentiment analysis.

4.Describe any one AWS NLP managed services.
•Amazon Polly is a managed service that converts text into lifelike speech. Amazon Polly supports multiple languages and includes various lifelike voices.
•Generate voice from plain text or Speech Synthesis Markup Language (SSML) format.
•SSML is a markup language that you can use to provide special instructions for how speech should sound. For example, you might want to introduce a pause in the flow of speech. You can add an SSML tag to instruct Amazon Polly to pause between two words.
•Create output in multiple audio formats.
•You can also output speech from Amazon Polly to MP3, Vorbis, and pulse-code modulation (PCM) audio stream formats.

•Amazon Polly is eligible for use with regulated workloads for the U.S. Health Insurance Portability and Accountability Act of 1996 (HIPAA), and the Payment Card Industry Data Security Standard (PCI DSS).

News service production – Major news companies use Amazon Polly to generate vocal content directly from their written stories.

Language training systems – Language training companies use Amazon Polly to create systems for learning a new language.

Navigation systems – Amazon Polly is embedded in mapping application programming interfaces (APIs) so that developers can add voice to their geo-based applications.

Animation production – Animators use Amazon Polly to add voices to their characters.