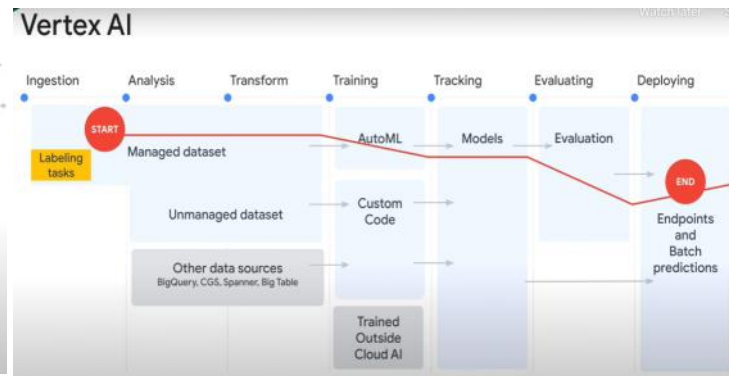
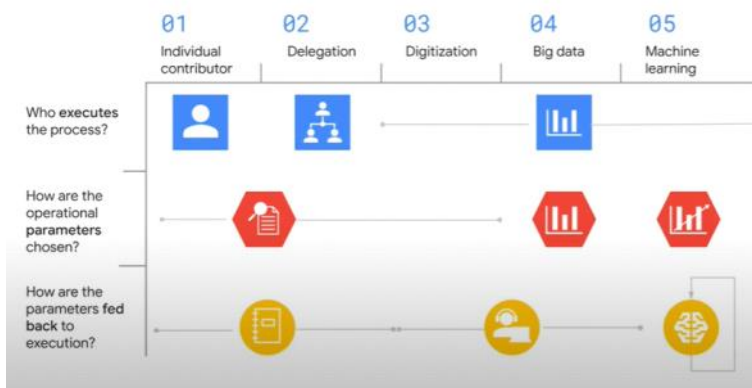


BASIC TUT

Thursday, October 27, 2022 6:58 PM



Framing the problem

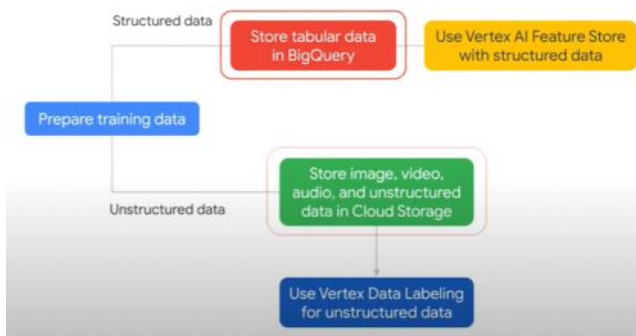
Prepare training data

Experimenting

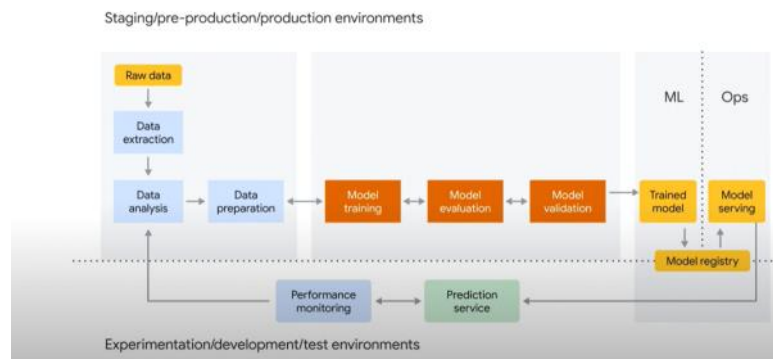
Evaluating the model

There is a tensorflow transform for preprocessing data

Avoid storing data in block storage



An ML pipeline recap

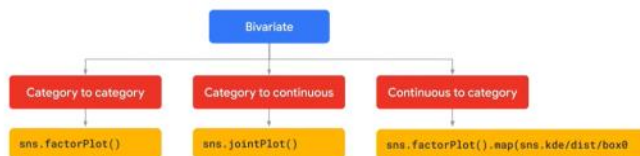
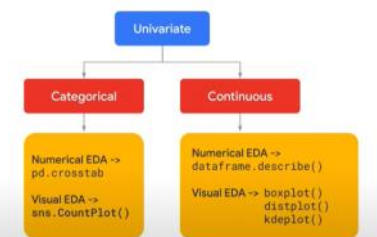


Data quality



Exploratory data analysis

Exploratory data analysis



TENSORFLOW

Wednesday, November 2, 2022 2:42 PM

A tensor is a N dimensional array of data

- Rank 0 tensor (scalar) (`tf.constant(3)`)
- Rank 1 tensor (vector) (`tf.constant([3,5,7])`)
- Rank 2 tensor (matrix)
- Rank 3 tensor ()
- Rank 4 tensor ()

Tensorflow lite allows us to put the model trained into our mobile device and run inference or predictions offline



`Tf.constant` produces constant tensors

`Tf.Variable` produces tensors that can be modified

`Tf.data.dataset` allows us to

- Preprocess data in parallel (and cache result of costly operations)
- Configure the way the data is fed into a model with a number of chaining methods
- Create data pipelines from in-memory or out of memory sharded data files

Discretization:

- Turns continuous numerical features into bucket data with discrete ranges

Normalization:

- Holds the mean and standard deviation of the features

TextVectorization:

- Holds a mapping between string tokens and integer indices

`Tf.keras.layers.Hashing`:

- Performs categorical feature hashing

`Tf.keras.layers.StringLookup`:

- Turns string categorical values into an encoded representation that can be read by an embedding layer or dense layer

`Tf.keras.layers.IntegerLookup`:

- Turns integer categorical values into an encoded rep that can be read by an embedding layer or dense layer

FEATURE STORE

Monday, November 7, 2022 4:33 PM

- Feature store is a top level container for features and their values
- Permitted users can add and share their features and also users can ingest values from various data sources
- An Entity type is a collection of semantically related features
- An entity is an instance of an entity type
Movie1, movie2 are entities of the movies entity type (entity should be a unique string)
- When we retrieve values from a feature store, the service returns an entity view that contains feature values that we requested

1) Batch serving

- High throughput, large volumes of data for offline processing

2) Online serving

- Low latency data retrieval of small batches of data for online real time processing

Pre requisites for feature store creation:

- No missing values
- Correct data type
- One hot encoding already done
- Include a column entity ID's, and the values to be string
- Source data must match the value types of the destination feature in the featurestore
- If providing feature generation timestamps choose TIMESTAMP type for bigquery, long, logical type for avro, rfc 3339 type for csv
- For array datatypes don't use csv and use an array instead of null value give an empty array

First Featurestore is created and entity type is created and features(fields in the dataset) are added to the entity type with its feature name type etc..

Csv , tfrecord format is used for batch serving outputs

Feature columns:

- Numerical
- Categorical
- Bucketized
- Crossed
- Embedding
- Hashed

Features should be known at prediction time

Features should be numeric

Features should have enough examples

Feature engineering:

- Representation transformation:
 - o Convert a numeric feature to a categorical feature(bucketization)
 - o Convert categorical features to a numeric representation (one hot encoding, feature embeddings etc..)
- Feature construction
 - o Create new features either by using typical techniques (polynomial expansion by using univariate mathematical functions or feature crossing)
 - o A feature cross is a synthetic feature formed by multiplying (crossing) two or more

features. Crossing combinations of features can provide predictive abilities beyond what those features can provide individually

`TRANSFORM(ML.FEATURE_CROSS(STRUCT(features)),ML.BUCKETIZE(feature,splitpoints(array)) etc..)`

Bucketnames has the name bin and it will be with the feature values

Machine Learning in the Enterprise

Tuesday, November 8, 2022 10:00 AM

Making training faster:

- Use tpu's
- Making an input pipeline faster
- Running parallelly in many devices in tpu's,gpu's etc..
- Distributed processing with multiple machines with multiple devices in parallel
- Use Data parallelism

Data parallelism:

- Use many workers and have their own data for training
- Update model parameters for the synchronization of the model (for combining results like)
- There will be a async parameter server for all the workers that syncs the parameters(choose if many low power or unreliable workers)
- There is also a sync allreduce parameter server that will synchronize the parameters form the parent worker to the child worker without the need of a central parameter server for all the workers(choose if multiple devices on one host)

Model parallelism:

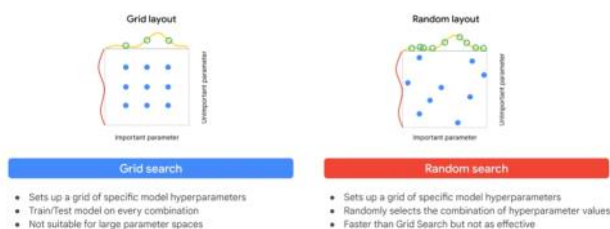
- Putting different devices on different layers of deeplearning

Maximum artifact size for custom training is like 5 tb

Vizier hyperparameter tuning:

- Grid search
- Random search
- Bayesian optimization(default)

Grid and Random Search



Bayesian optimization

Watch later

Advantages	Process
<ul style="list-style-type: none">• Past evaluations when choosing the hyperparameter are set• Typically requires less iterations to get to the optimal set of hyperparameter value• Limits the number of times a model needs to be trained for validation	<ul style="list-style-type: none">• Build a model• Select hyperparameters• Train and evaluate• Update the model• Repeat (or iterate) until max iterations are reached

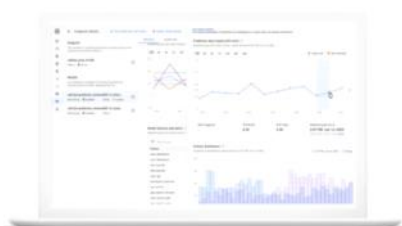
While creating a model custom training and give custom image and in the hyperparameters page click enable hyperparameter tuning and give the hyperparameters that matches with the command line args in your code

We can set a maximum number of trials and max no of parallel trials(lower than max trials)

Predictions:

- Batch prediction is asynchronous
- Online prediction is synchronous

Best practices: Model deployment and serving



Machine type Specify the number and type of machines you need.
Model inputs Plan inputs to the model.
Automatic scaling Turn on automatic scaling.
Specify performance requirements Define what is good and bad performance.

Best practices: Model deployment and serving

Host
Optimized for widely-used serial tasks

Accelerator
Optimized for many parallel distributed tasks

To deploy your model for prediction, choose hardware that is appropriate for your model, like different central processing unit (CPU) virtual machine (VM) types or graphics processing unit (GPU) types.

Best practices: Model deployment and serving

Model inputs Plan inputs to the model.
--

If you're using batch prediction you can fetch data from the data lake, or from [Vertex AI Feature Store batch serving API](#).

If you are using online prediction, send input instances to the service and it returns your predictions in the response.



Specify performance requirements
Define what is good and bad performance.

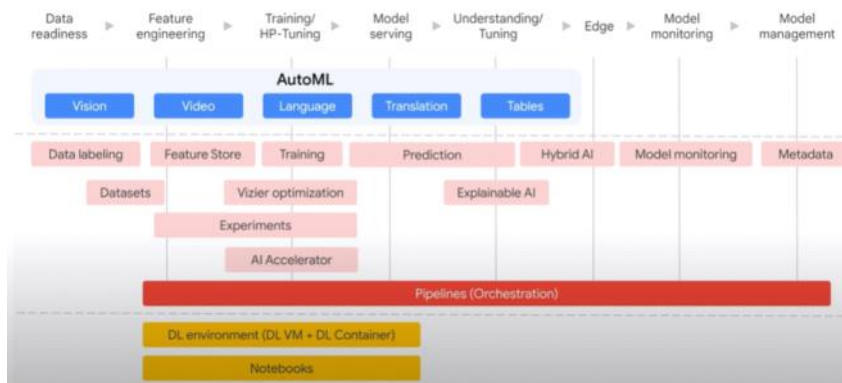
and serving

Model inputs
Plan inputs to the model.

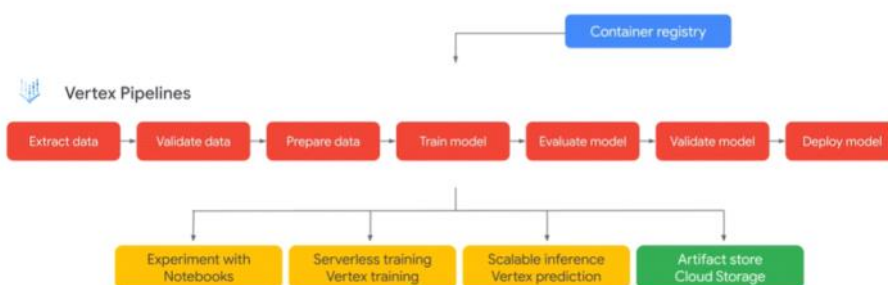
If you are using online prediction, send input instances to the service and it returns your predictions in the response.

in model monitoring use skew detection, drift detection
For seeing changes to production data in accordance with training

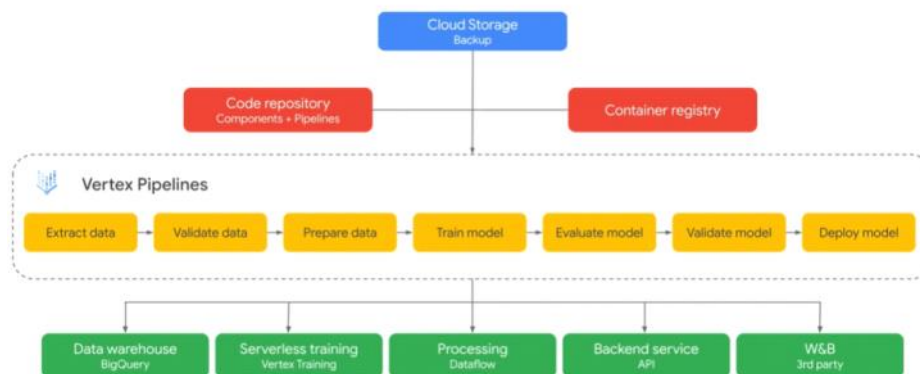
Use alert threshold



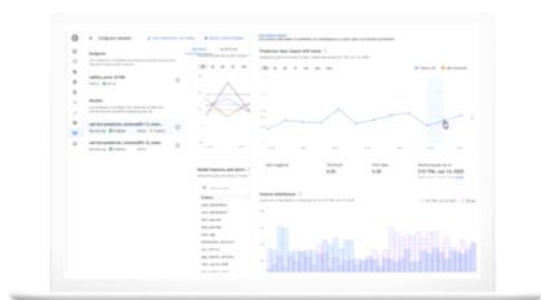
Pipelines automate the training and deployment of models



Supporting architecture



Best practices: Vertex AI Pipelines



Assess perfection

Why did a pipeline produce an especially accurate model?

Compare pipelines

Which pipeline run produced the most accurate model and parameters used?

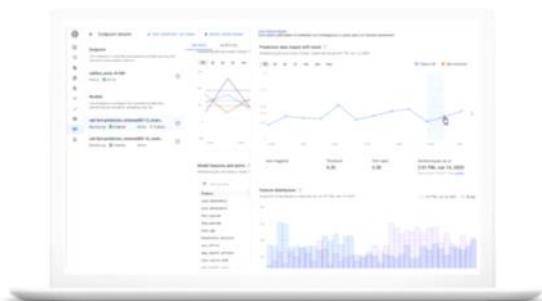
System governance

Which version of your model is in production at a given time?

Pipeline SDK

Kubeflow SDK

Best practices: Vertex AI Pipelines



Assess perfection

Why did a pipeline produce an especially accurate model?

Compare pipelines

Which pipeline run produced the most accurate model and parameters used?

System governance

Which version of your model is in production at a given time?

Pipeline SDK

Kubeflow SDK
TensorFlow Extended

Best practices: Artifact organization

Artifact lineage describes all the factors that resulted in an artifact.

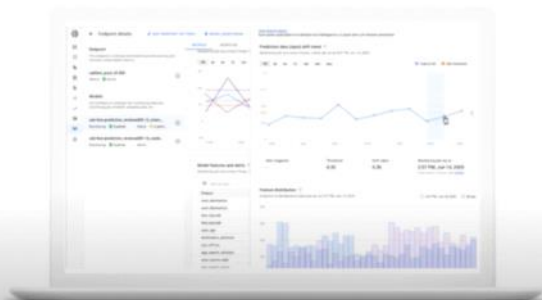
You can understand differences in performance or accuracy over several pipeline runs.

A model's lineage could include the following:

- The training, test, and evaluation data used to create the model.
- The hyperparameters used during model training.
- The code that was used to train the model.
- Metadata recorded from the training and evaluation process.
- Artifacts that descend from this model.

Best practices: Artifact organization

Watch later Share



Artifacts

Organize your ML model artifacts.

Git repo

Use a Git repo for pipeline definitions and training code.

Artifacts are outputs resulting from each step in the ML workflow.

Production Machine Learning System

Tuesday, November 8, 2022

8:54 PM

- For all non numeric columns , other than TIMESTAMP, bigquery ML performs a one hot encoding transformation

RECOMMENDATION SYSTEMS:

- Content based
- Collaborative filtering

Content based filtering uses item features to recommend new items that are similar to what the user has liked in the past

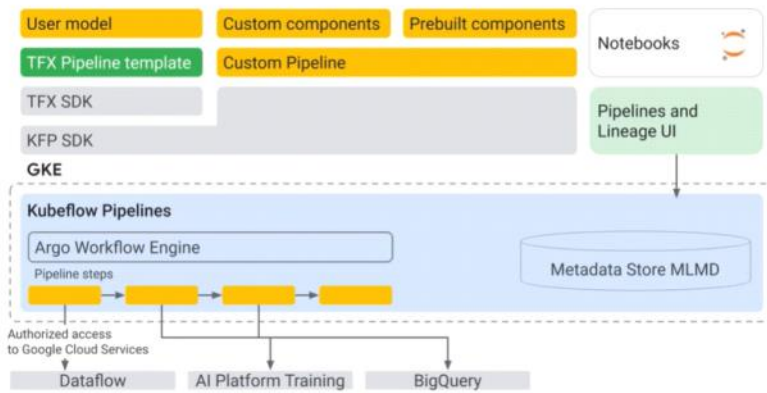
Current process for building an MLOps pipeline

- Set up a Google Kubernetes Engine (GKE) cluster.
- Create a Cloud Storage bucket for storing data.
- Install Kubeflow pipelines.
- Set up port forwarding.
- Create a process to share the pipeline with your team.

AI platform pipelines automates it

2. Easy authentication process
 - Fully secure and provides authenticated access to Pipelines UI
 - No need to set up port forwarding
 - Easy to share with team members
 - Easy to access through REST API service
 - Seamless performance of using Pipelines SDK from Notebooks
 - Define pipeline
 - Schedule run job

AI Platform Pipelines tech stack



AI Platform Pipelines implementation strategy

Kubeflow Pipelines

Through Kubeflow Pipelines SDK

- Lower-level ML framework-agnostic implementation
- Enables direct control of Kubernetes resource control
- Simple sharing of containerized components
- Use it for fully custom pipelines

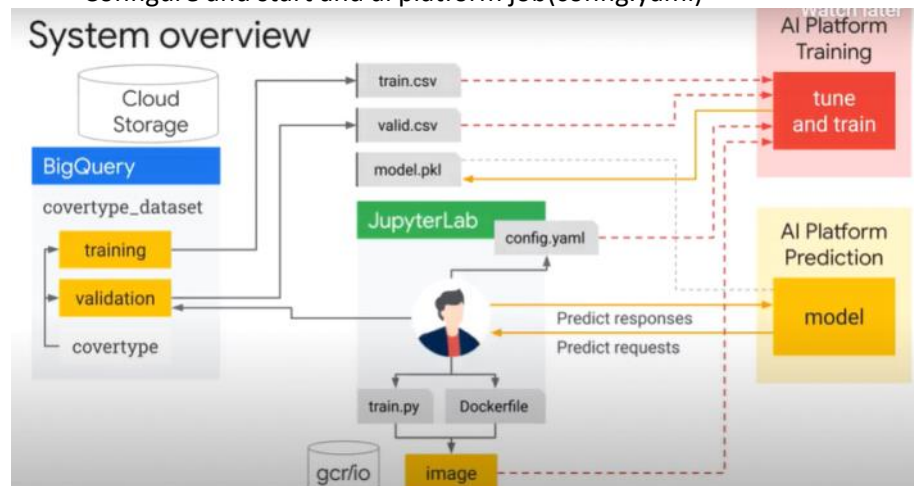
TensorFlow Extended (TFX)

Through TFX SDK

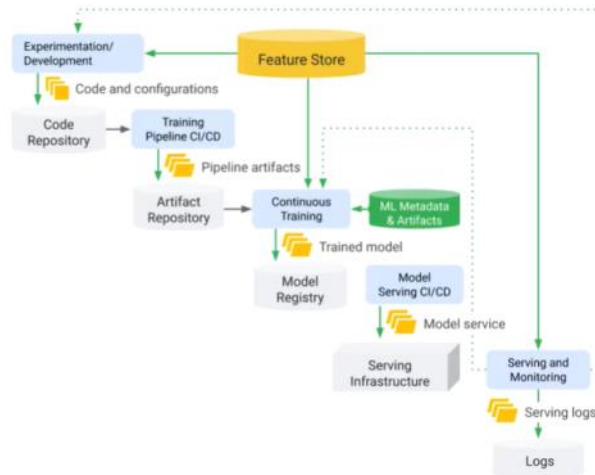
- Higher-level abstraction
- Prescriptive but customizable components with pre-defined ML types
- Brings Google best practices for robust/scalable ML workloads
- Use it for E2E TF-based pipeline with customizable data pre-processing and training code

Building and operationalizing models:

- Implement a tunable training application(train.py)
- Package your training application(Dockerfile)
- Configure and start an ai platform job(config.yaml)



Where we are going next



For splitting the dataset into training and test dataset we can use hashing and mod combined when querying the data like $\text{MOD}(\text{ABS}(\text{FARM_FINGERPRINT}(\text{FIELD})), 8) < 8$
 Or if we are not sure what to hash on we can use `TO_JSON_STRING(tablename)` inside the `farm_fingerprint()`

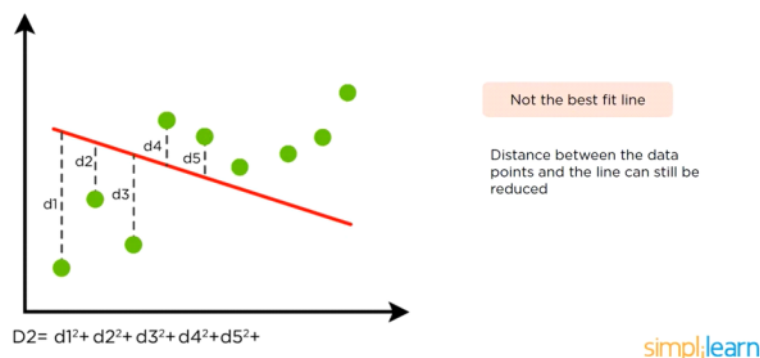
In code we can use fire python package for accepting cli args in function

```

Import fire
Def fucn(col1,col2):
  Pass
  Fire.Fire(func)
  (python train.py --col1=dd --col2=cc)
  
```

Linear Regression

Finding the best fit line: The best fit line can be found out by minimizing the distance between all the data points and the distance to the regression line. Ways to minimize this distance are sum of squared errors, sum of absolute errors etc.

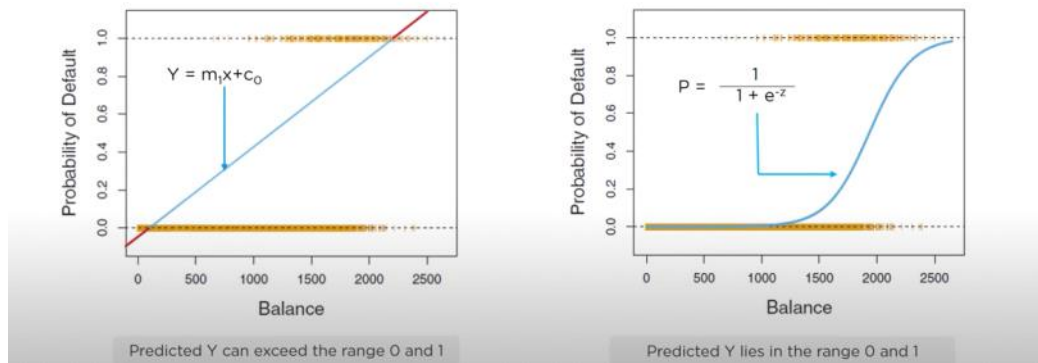


The regression line is drawn randomly and the distance between the line and the each data points summed together
 Should be less (sum of squared error)

Internally the code tried to get the line close to the data points that result to a less squared error
 Cost function says how far the line is from the data points

Logistic Regression

Plotting the Logistic Regression Curve: The Logistic Regression curve is known as the Sigmoid curve (S curve)



In logistic regression a sigmoid curve is used to design the model