Here's the breakdown of all files in your SageMaker Calculator project and their specific purposes:

**Core Source Files (88src/)**

[**calculator\_model.py**](file:///c:\MyProjects\MLProjects\SageMakerCalculator\src\calculator_model.py)

* Contains the MathCalculator class with core business logic
* Implements 10 mathematical operations (add, subtract, multiply, divide, power, sqrt, sin, cos, tan, log)
* Handles error cases (division by zero, negative square roots, etc.)
* Pure Python math logic, no SageMaker dependencies

[**inference.py**](file:///c:\MyProjects\MLProjects\SageMakerCalculator\src\inference.py)

* SageMaker-specific inference handler
* Implements required SageMaker functions: model\_fn, input\_fn, predict\_fn, output\_fn
* Handles JSON request/response formatting
* Bridges between SageMaker runtime and calculator model

**Notebooks (notebooks/)**

[**calculator\_development.ipynb**](file:///c:\MyProjects\MLProjects\SageMakerCalculator\notebooks\calculator_development.ipynb) (currently active)

* Interactive development and testing environment
* Tests calculator model locally before deployment
* Validates inference functions
* Sets up SageMaker session and role

**Deployment (deployment/)**

[**deploy\_model.py**](file:///c:\MyProjects\MLProjects\SageMakerCalculator\deployment\deploy_model.py)

* Automated deployment script
* Creates model.tar.gz package
* Uploads to S3 and deploys SageMaker endpoint
* Command-line deployment option

[**sagemaker\_deployment.ipynb**](file:///c:\MyProjects\MLProjects\SageMakerCalculator\deployment\sagemaker_deployment.ipynb)

* Interactive deployment workflow
* Step-by-step deployment process in notebook format
* Includes testing of deployed endpoint
* Visual deployment alternative to script

**Testing (tests/)**

[**test\_api.py**](file:///c:\MyProjects\MLProjects\SageMakerCalculator\tests\test_api.py)

* End-to-end API testing
* Tests deployed SageMaker endpoint via boto3
* Validates multiple operations and response formats
* Production readiness verification

**Configuration**

[**requirements.txt**](file:///c:\MyProjects\MLProjects\SageMakerCalculator\requirements.txt)

* Project dependencies (boto3, sagemaker, numpy, scikit-learn)

**Key Differences:**

1. **Development vs Deployment**: Development notebook tests locally, deployment files handle SageMaker
2. **Script vs Notebook**:
   1. deploy\_model.py for automation,
   2. sagemaker\_deployment.ipynb , for interactive deployment
3. **Local vs Remote Testing**: Development notebook tests locally, test\_api.py tests deployed endpoint
4. **Core vs Interface**:
   1. calculator\_model.py has business logic.
   2. inference.py handles SageMaker integration

Each file serves a distinct role in the ML lifecycle from development to production deployment.

**How much time it takes to create a sagemaker domain**

Creating a SageMaker domain typically takes **5-15 minutes**, depending on several factors:

**Time Breakdown:**

**Quick Setup (5-8 minutes):**

* Using default VPC and subnets
* Standard domain configuration
* No custom networking

**Standard Setup (8-12 minutes):**

* Custom VPC configuration
* Multiple user profiles
* Basic security settings

**Complex Setup (12-15+ minutes):**

* Custom networking with private subnets
* Advanced security configurations
* Multiple execution roles
* Custom lifecycle configurations

**Factors Affecting Creation Time:**

1. **Network Configuration**
   * Default VPC: Faster
   * Custom VPC: Additional 3-5 minutes
2. **Security Setup**
   * Basic IAM roles: Standard time
   * Custom policies: +2-3 minutes
3. **User Profiles**
   * Single user: Faster
   * Multiple users: +1-2 minutes per user
4. **Region Load**
   * Peak hours may add 2-3 minutes

**Quick Setup Steps:**

1. **AWS Console → SageMaker → Domains**
2. **Create Domain → Quick Setup**
3. **Choose default settings**
4. **Wait 5-8 minutes**

**What Happens During Creation:**

* VPC and security group setup
* IAM role configuration
* EFS file system creation
* Studio environment initialization
* User profile provisioning

Once created, launching Studio takes an additional **2-3 minutes** for the first time, then **30-60 seconds** for subsequent launches.