Modular Coding

1. Data Ingestion
2. Data Validation
3. Data Transformation
4. Model Trainer
5. Model Evaluation
6. Model pusher
7. **Data Ingestion**

**Workflow**

1. Us-visa 🡪 constants 🡪 copy the code

For all the components you need to assigned in the constant folder , so that instead of changing it in each file , we can change it from the single file

File contain important variables

1. In us-visa 🡪 configuration 🡪 create a mongo db connection.py file

All the folder and file path should be managed in the config

1. Us visa 🡪 create folder data\_access 🡪 create \_\_init\_\_.py and usvisa\_data.py 🡪
2. Us visa 🡪 entity🡪 config entity🡪
3. Us visa 🡪 entity 🡪 artifact (return type of the component . Here : data ingestion)🡪

Output from the components

Here train.py and test.py

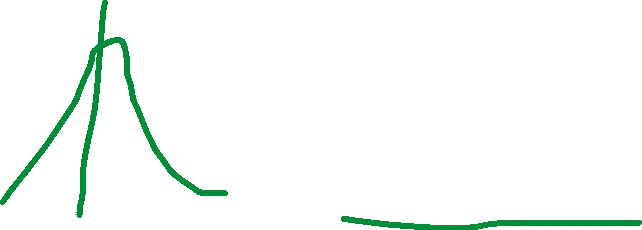
1. Us visa 🡪 Data ingestion.py 🡪
2. Us visa 🡪 components 🡪 Pipeline🡪Training\_pipeline
3. Before that set the MONGODB\_URL in env variable or set MONGODB\_URL=”-------------"
4. End point

**Data Drift**

Data drift, also known as concept drift or dataset shift, refers to the gradual or abrupt change in the statistical properties of the data used to train a machine learning model. These statistical properties can include changes in the distribution of features, target labels, or the relationships between them.



No drift occurs



Drift occurs

Training and testing data should be follows the same distribution . To detect the data drift we can use the MLOPs tool Evidently AI

Train data:Old data Test data : new data

1. **Data Validation**

* In schema.yaml file , we need to mention the data schema like the column name , data types (meta data )
* After that drift prediction occurs and the status is stored as a yaml file
* In schema.yaml file ,

Mention all the features like columns, numercal columns , categorical columns , drop\_columns , numerical features , ordinal encoding , onehot encoding , transform columns

* Yaml file is used to manage the configuration

1. Us – visa 🡪 constants
2. Us visa 🡪 entity 🡪 config entity.py
3. Us Visa 🡪 entity 🡪 artifact entity

* It returns the Boolean value ie., True or False
* It also returns the path of the report file

1. Us visa 🡪 components 🡪 data validation 🡪
2. Us visa 🡪 components 🡪 Pipeline🡪Training\_pipeline
3. Update the main\_utils .py file
4. And schema file before that
5. End point
6. **Data Transformation**
7. Us visa 🡪 Constants
8. Us visa 🡪 entity 🡪 config entity.py
9. Us visa 🡪 entity 🡪 artifact entity.py

It returns 3 files

* train.npy
* test.npy
* preprocessor.pkl

Now we need to convert the target variable (here category ) into number

1. Us visa 🡪 entity 🡪 create a file ‘estimator.py’
2. Us visa 🡪 components 🡪 data transformation

* Load the DataTransformation component
* “ “ DataTransformationconfig
* “ “ DataTransformationArtifact
* Initialize it in the class variable
* Define start\_data\_transformation
* Call data\_transformation inside the run pipeline

1. Us visa 🡪 pipeline
2. End point
3. **Model Trainer**

* All the parameters for the hyperparameter tuning
* And the best models are stored in the model.yaml file
* Update Yaml file ; you can mention multiple model and corresponding parameters

1. Us visa 🡪 Constants
2. Us visa 🡪 entity 🡪 config enitity.py
3. Us visa 🡪 entity 🡪artifact entity.py

* ClassificationMetricArtifact and then ModelTrainerArtifact

1. Us visa 🡪 entity 🡪 update estimator.py {add UsVisaModel}
2. Us visa 🡪 components 🡪 Model Trainer

* Load the ModelTrainer component
* “ “ ModelTrainerConfig
* “ “ ModelTrainer Artifact
* Initialize it in the class variable
* Define start\_Model\_Trainer
* Call model\_trainer inside the run pipeline

1. Us visa🡪 pipeline
2. End point