04_model_trainer

February 12, 2024

[1]:
```
import os
```

[2]:
```
%pwd
```

[2]:
```
'D:\\Desktop\\Deep Learning\\Lab 2\\MNSIT-MLPClassifer\\Research'
```

[3]:
```
os.chdir("../")
```

[4]:
```
%pwd
```

[4]:
```
'D:\\Desktop\\Deep Learning\\Lab 2\\MNSIT-MLPClassifer'
```

[6]:
```
import logging
import os
from dataclasses import dataclass
from pathlib import Path
import pandas as pd
from sklearn.neural_network import MLPClassifier
from joblib import dump

# Configure logging
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - ␣
 ↪%(message)s')

@dataclass(frozen=True)
class DataTransformationConfig:
    root_dir: Path
    X_train_file: Path
    y_train_file: Path
    mlp_mnist_model_file: Path

class ConfigurationManager:
    def __init__(self):
        self.root_dir = Path(os.getcwd())
        self.X_train_file = self.root_dir / "dataset/Modeltraining/X_train.csv"
        self.y_train_file = self.root_dir / "dataset/Modeltraining/y_train.csv"
        self.mlp_mnist_model_file = self.root_dir / "Model/mlp_mnist_model.pkl"
```

```
    def get_data_transformation_config(self) -> DataTransformationConfig:
        return DataTransformationConfig(
            root_dir=self.root_dir,
            X_train_file=self.X_train_file,
            y_train_file=self.y_train_file,
            mlp_mnist_model_file=self.mlp_mnist_model_file
        )

def save_model(model, file_path):
    dump(model, file_path)  # Corrected from joblib.dump to dump, assuming␣
 ↪joblib is imported via from joblib import dump

class DataTransformation:
    def __init__(self, config: DataTransformationConfig):
        self.config = config

    def train_model(self):
        # Load the training data
        X_train = pd.read_csv(self.config.X_train_file)
        y_train = pd.read_csv(self.config.y_train_file)

        logging.info("X_train and y_train loaded")

        # Train the MLP Classifier
        mlp = MLPClassifier(hidden_layer_sizes=(32,), max_iter=100)  # Adjusted␣
 ↪max_iter for better convergence, if needed
        mlp.fit(X_train, y_train.values.ravel())  # Using .values.ravel() to␣
 ↪ensure y is the correct shape

        logging.info("Model training completed")

        # Save the model
        save_model(mlp, self.config.mlp_mnist_model_file)
        logging.info("MLP MNIST model saved")

def main():
    try:
        config_manager = ConfigurationManager()
        data_transformation_config = config_manager.
 ↪get_data_transformation_config()
        data_transformation =␣
 ↪DataTransformation(config=data_transformation_config)

        # Train the model
        data_transformation.train_model()
    except Exception as e:
        logging.error(f"Error occurred: {e}")
```

```
        raise

if __name__ == "__main__":
    main()
```

```
2024-02-07 00:00:41,758 - INFO - X_train and y_train loaded
2024-02-07 00:03:10,116 - INFO - Model training completed
2024-02-07 00:03:10,131 - INFO - MLP MNIST model saved
```

[ ]: