PORTFOLIO

1. Crypto Price Forecasting

The core objective of this project was to accurately forecast the prices of Bitcoin by utilizing a variety of machine learning techniques. A significant emphasis was placed on accuracy, with a specific model showcasing exceptional performance. The project utilized historical Bitcoin cryptocurrency price data, including features such as opening, closing, high, and low prices, as well as trading volume, sourced from comprehensive cryptocurrency market datasets.

Project Components:

- **Exploratory Data Analysis (EDA)**: Initial data exploration to understand trends, patterns, and anomalies within the cryptocurrency market data.
- **Data Preprocessing**: Preparation of data for modeling, including cleaning, normalization, and feature engineering to enhance model performance.
- **Model Selection and Evaluation**: Comparative analysis of various machine learning models to select the most effective approach for price prediction.
- **Feature Importance Analysis**: Identification of key features that significantly impact the accuracy of price predictions.
- **Predictive Modeling**: Implementation of machine learning algorithms to develop a predictive model for cryptocurrency prices.

For visual representation of the data and findings, Python libraries such as Matplotlib and Seaborn were employed, providing clear and insightful visualizations of the cryptocurrency market's dynamics.

Features Used for Prediction:

The project utilized several features derived from historical price data to predict future cryptocurrency prices:

- **F1**: Open Price
- **F2**: Close Price
- **F3**: High Price
- **F4**: Low Price
- **F5**: Volume of Trades
- **F6**: Historical Volatility
- **F7**: Moving Averages
- **Additional Technical Indicators**

Key Features for Prediction:

The analysis highlighted the importance of specific features in predicting cryptocurrency prices effectively:

1. **F2**: Close Price
2. **F7**: Moving Averages

Machine Learning Algorithms Employed:

A variety of machine learning algorithms were explored to forecast cryptocurrency prices, leveraging the Scikit-learn library, among others:

- **Linear Regression**
- **Support Vector Regression (SVR)**
- **Random Forest Regressor**
- **Gradient Boosting Regressor**
- **K-Nearest Neighbors (KNN)**
- **Decision Tree Regressor**
- **Extra Trees Regressor**
- **Neural Networks (using frameworks like TensorFlow or PyTorch, if mentioned within the project)**

Tools Used: Python, pandas for data manipulation, Matplotlib and Seaborn for data visualization, and various machine learning models through Scikit-learn.
Project Link

2. Real Waste Image Classification for Efficient Waste Management

In this innovative AI project, I focused on enhancing waste management processes through the development of an advanced system for the classification of real waste images. This project was conceptualized to address the growing need for efficient waste sorting and management solutions, contributing to environmental sustainability efforts.
Project Overview:

- **Objective**: To develop a machine learning model capable of classifying images of waste into predefined categories, thereby facilitating automated sorting processes.
- **Dataset**: Utilized a comprehensive dataset comprising images of various waste types, sourced from public environmental datasets and waste management organizations.
- **Key Technologies**: PyTorch for model development, OpenCV for image processing, and custom algorithms for image classification.

Technical Approach:

- **Image Preprocessing**: Implemented advanced image processing techniques using OpenCV to prepare the dataset for model training, including resizing, normalization, and augmentation to enhance model robustness.
- **Model Development**: Leveraged PyTorch to construct and train a convolutional neural network (CNN) tailored for high-accuracy image classification tasks.
- **Classification Performance**: Focused on achieving high classification accuracy across multiple waste categories, employing techniques like transfer learning and fine-tuning to optimize model performance.

Project Deliverables:

- **Automated Waste Classification**: Successfully developed a model that accurately classifies waste images, supporting the automation of waste sorting processes.
- **Efficiency Improvement**: Demonstrated the potential for significant improvements in waste management efficiency, reducing manual labor requirements and enhancing sorting accuracy.
- **Environmental Impact**: Highlighted the project's contribution to environmental conservation efforts by enabling more effective recycling and waste management practices.

Key Outcomes:
This project successfully achieved its goal of developing a high-performance image classification system for waste management, showcasing the potential of AI and machine learning technologies in addressing environmental challenges. The model's high accuracy in classifying various types of waste materials paves the way for more efficient and automated waste sorting solutions, contributing to the broader goals of sustainability and environmental protection.
Tools Used: PyTorch for deep learning model development, OpenCV for image processing, and possibly other Python libraries for data manipulation and visualization.
Project Link

3. Deep Learning with PyTorch: Image Segmentation

This project aimed to enhance object detection and classification systems through deep learning-based image segmentation. Utilizing PyTorch, OpenCV, and CUDA for accelerated computing, I developed a model capable of segmenting objects with high precision across various datasets, including urban scenes and medical imaging.
Core Components:

- **Technologies**: PyTorch for model development, OpenCV for image processing, and CUDA for GPU acceleration.
- **Algorithms**: Employed CNNs, U-Net for specialized segmentation tasks, and explored different segmentation models for optimal performance.
- **Dataset**: Diverse datasets from public sources were used, focusing on a range of segmentation challenges.

Achievements:

- **Accuracy**: Achieved state-of-the-art segmentation accuracy, enabling precise object detection.
- **Efficiency**: Enhanced detection systems' efficiency by integrating advanced segmentation models.
- **Scalability**: Demonstrated model adaptability to various segmentation tasks, proving its potential for real-world applications.

This project showcases the power of deep learning in improving the accuracy and efficiency of image segmentation, paving the way for advancements in fields requiring precise object detection and classification.
Tools Used: PyTorch for deep learning, OpenCV for image processing, and CUDA for GPU acceleration.
Project Link

4. Face Counting with YOLO

This project developed a real-time system for accurately counting faces using the YOLO algorithm, optimized for speed and efficiency across diverse conditions.
Highlights:

- **Technology**: Integrated YOLO with Python and OpenCV, leveraging CUDA for enhanced processing speed.
- **Achievements**: Enabled real-time detection and counting with high accuracy, suitable for crowded environments and adaptable to various scenarios.

A practical solution demonstrating YOLO's effectiveness in face detection challenges, showcasing potential applications in surveillance and crowd management.
Tools Used: YOLO for object detection, Python, OpenCV for image processing, and CUDA for enhanced computational speed.
Project Link

5. Webcam Object Detection

This project focuses on deploying a real-time object detection system using webcam input. It leverages advanced deep learning models to identify and classify objects within the camera's field of view, aiming for high accuracy and low latency in diverse lighting and environmental conditions.
Highlights:

- **Technology**: Utilizes cutting-edge deep learning frameworks alongside Python and OpenCV for real-time image processing.
- **Outcome**: Achieves efficient and accurate object detection, demonstrating the system's robustness and practicality for real-time applications.

A streamlined approach to object detection, proving the viability of using webcams for dynamic, real-time surveillance and interactive systems.
Tools Used: Advanced deep learning frameworks (not specified, but likely TensorFlow or PyTorch), Python, OpenCV for real-time image processing.
Project Link

6. TensorFlow Object Detection

This project implements a sophisticated object detection system using TensorFlow. It showcases the application of TensorFlow's powerful object detection capabilities, fine-tuned for recognizing and tracking various objects in real-time video streams.
Highlights:

- **Technology**: Employs TensorFlow's object detection API, integrated with Python for comprehensive image analysis.
- **Performance**: Delivers real-time detection with high precision, adaptable to multiple object classes and environments.

A showcase of TensorFlow's versatility in object detection, this project illustrates the framework's potential for developing advanced real-time surveillance and monitoring solutions.
Tools Used: TensorFlow, especially its object detection API, and Python for image analysis and processing.
Project Link