Build a machine learning model to classify final diagnoses based on images, age, sex, and diagnostic text using Apache Spark. The final model and analysis will be accessible through a web interface. [**Ocular Disease Recognition Dataset**](https://www.kaggle.com/datasets/andrewmvd/ocular-disease-recognition-odir5k/data)

## **Project Timeline & Milestones**

### **Phase 1: Project Proposal & Data Collection (Deadline: April 29 )**

1. **Data Collection (April 29):**
   * Download and store the dataset on an EC2 instance.
   * Preprocess structured data and store in a structured format.
   * Test basic Spark functions for data manipulation and validation.
   * Document dataset statistics (record count, attributes, distribution).

#### **Deliverables:**

* Project proposal document.
* Data uploaded to EC2.
* Initial Spark code and screenshots.

[Cluster Mode Overview - Spark 3.5.5 Documentation](https://spark.apache.org/docs/3.5.5/cluster-overview.html)

[GitHub - amplab/spark-ec2: Scripts used to setup a Spark cluster on EC2](https://github.com/amplab/spark-ec2)

<https://docs.google.com/document/d/1-asX-Gn_jQ_FPAIIX8xtVH_RwHyVZhlWpB_nUczI8AY/edit?tab=t.0>

### **Phase 2.1: Data Processing (May 12) (5 analytical tasks needed)**

#### **Tasks:**

This phase we can use **transformations** in **pipelines** to handle processing the various data

[ML Pipelines - Spark 3.5.4 Documentation](https://spark.apache.org/docs/3.5.4/ml-pipeline.html#main-concepts-in-pipelines)

**Feature Engineering:**

[Extracting, transforming and selecting features - Spark 3.5.3 Documentation](https://spark.apache.org/docs/3.5.5/ml-features.html#feature-extractors)

* + Basic feature engineering
    - Convert numeric features into categorical features (age into age groups)
    - One hot encode sex (0/1)
    - Convert one hot encoded diagnosis to integer field (0 - # diagnoses)
  + Text Featurization:
    - Convert diagnostic text into numerical features using NLP techniques (TF-IDF, Word2Vec, BERT embeddings).
  + Extract image features using:
    - Data augmentation to increase training data
    - Pretrained CNN (ResNet/VGG) for embeddings.
    - OpenCV for manual features like color histograms

[pyspark.RDD.map](https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.RDD.map.html)

[Data sources - Image Sources - Spark 3.5.3 Documentation](https://spark.apache.org/docs/3.5.4/ml-datasource.html#image-data-source)

1. **Data Storage & Integration:**

[Spark SQL, DataFrames and Datasets Guide](https://spark.apache.org/docs/latest/sql-programming-guide.html)

* + Combine image features, text features, patient demographics and diagnoses.
  + Store analyzed data in MySQL/Cassandra for further use.

#### **Deliverables:**

* Processed augmented dataset with feature-engineered attributes and documentation

**MySQL Documentation**

[MySQL 8.4 Reference Manual :: 5 Tutorial](https://dev.mysql.com/doc/refman/8.4/en/tutorial.html)

[JDBC To Other Databases - Spark 3.5.4 Documentation](https://spark.apache.org/docs/3.5.4/sql-data-sources-jdbc.html)

### **Phase 2.2: Model Training (May 20)**

1. **Model Training:**
   * Train classification models (RandomForest, GBT, Logistic Regression) on structured data using the extracted features
   * Train on dataset with text features and without text features for interface flexibility

[Classification and regression - Spark 3.5.4 Documentation](https://spark.apache.org/docs/3.5.4/ml-classification-regression.html)

1. **Hyperparameter Tuning & Performance Evaluation:**  
   * Evaluate models using accuracy, precision, recall, and F1-score.
   * Optimize training with varying Spark worker configurations.

[ML Tuning - Spark 3.5.4 Documentation](https://spark.apache.org/docs/3.5.4/ml-tuning.html)

#### **Deliverables:**

* Trained models ( text diagnostics and no text diagnostics models)
* Performance comparison (models/features) report.
* Execution time graphs with different Spark worker configurations.

### **Phase 3: Web Interface Development (May 20 - June 3, 2025)**

[Flask](https://flask.palletsprojects.com/en/stable/)

1. **Design & Implement Web Interface:**
   * Implement search/filter functionalities for patient data as references for users
   * Display documentation/diagrams explaining our approach to processing/training
   * Deploy trained model for real-time diagnosis prediction.
   * Once diagnosis predicted we can return set of eyes similar to input
2. **Connect Web App to Database:**
   * Fetch analyzed data from MySQL/Cassandra.
   * Implement API endpoints for data retrieval and model predictions.

#### **Deliverables:**

* Display eye images based on filters for exploration
  + Ex: eyes of woman age 50+ with cataracts, eyes age 60+ with no ocular disease, diagnostic keyword filtering
* Return model results based on flexible input
  + User can input their demographics and eye images with or without imager diagnostics

NOTES

Feature importances

Model performance comparisons

Correlations between features / diagnosis