**CareerCompass: Machine Learning-Based Career Recommendation System Documentation**

This document provides a comprehensive overview of the **CareerCompass** system, a machine learning-based career recommendation solution designed to address the challenges students face in career planning. It combines a robust machine learning model with a user-friendly API to deliver personalized career recommendations. The document covers data collection, exploratory data analysis (EDA), model architecture selection, design, algorithm selection, evaluation, and recommendations, with justifications for each component in line with industry standards.

**1. Introduction**

**CareerCompass** is a system designed to tackle the limitations of traditional career counseling by providing personalized, data-driven career recommendations. It leverages machine learning to analyze a student’s academic performance, skills, interests, personality traits, and experiences, recommending career paths that align with their unique profile. The system integrates multiple datasets, employs natural language processing (NLP) for feature extraction, and uses a FastAPI-based interface to deliver actionable insights to users.

**Problem Statement**

Students often face challenges in career planning due to:

* Overemphasis on academic performance (e.g., recommending engineering for high math scores, ignoring interests in art or social work).
* Generic, outdated, or contradictory online career advice.
* Simplistic questionnaires in existing tools that fail to account for unique skills, personality, or experiences.
* Lack of relatable role models for marginalized communities, making fields like technology seem unattainable.

**CareerCompass** addresses these issues by providing a personalized, practical, and inspiring solution using machine learning and a user-friendly API.

**Objectives**

* Deliver top-3 career recommendations with confidence scores and clear explanations.
* Incorporate diverse data sources to capture academic, skill-based, and personality-based features.
* Ensure scalability and real-time inference via a robust API.
* Handle imbalanced data to ensure fair recommendations across underrepresented fields.

**2. Data Collection**

**Data Sources**

The system integrates multiple datasets to capture a wide range of features relevant to career planning:

1. **career\_guidance\_dataset.csv**: Contains 1,000 rows with features like GPA, field of study, career interests, and entrepreneurial aspirations.
2. **career\_path\_in\_all\_field.csv**: Includes 9,000 rows with data on GPA, extracurricular activities, internships, and various skills (e.g., coding, communication).
3. **career\_pred.csv**: Provides 7,525 rows with academic performance (e.g., CGPA), skills (e.g., programming, problem-solving), and role information.
4. **Data\_final.csv**: A smaller dataset (105 rows) with personality traits (OCEAN model) and aptitude scores.
5. **google\_form\_responses.csv**: Contains 1,028 responses with qualitative data on interests, skills, and career satisfaction.
6. **Student\_performance\_data.csv**: Includes 2,392 rows with academic performance, extracurricular activities, and GPA.
7. **Dataset Project 404.xlsx**: Provides 3,600 rows with multiple intelligence scores (e.g., linguistic, musical) and job professions.
8. **Augmented Dataset (Wikipedia Scraping)**: Generated to address class imbalance, adding ~700 samples across minority fields using Wikipedia content.

**Justification**:

* **Diversity of Data**: Combining datasets ensures a holistic view of a student’s profile, covering academic performance, skills, personality, and interests. This aligns with the problem statement’s emphasis on personalized recommendations beyond academic metrics.
* **Wikipedia Augmentation**: Scraping Wikipedia pages for underrepresented fields (e.g., Architecture, Legal) addresses class imbalance, ensuring the model learns from sufficient samples for minority classes. This is critical for fair predictions across diverse fields.
* **Real-World Relevance**: Datasets like google\_form\_responses provide qualitative insights from real users, enhancing the system’s ability to capture nuanced preferences.

**Data Collection Process**

* **Public Datasets**: Sourced from platforms like Kaggle and other open repositories, ensuring accessibility and relevance to career planning.
* **Wikipedia Scraping**: A script using the wikipediaapi library was developed to scrape content for 14 minority fields (e.g., Technology, Healthcare). For each field, up to 50 samples were collected from predefined and dynamically searched Wikipedia pages, tokenized, and processed to extract features.
* **Feature Extraction**: Text data from datasets and Wikipedia was processed using NLP (tokenization, keyword matching) to align with a unified schema.

**Justification**:

* **Scalability**: Using public datasets and Wikipedia ensures a scalable data collection pipeline that can be extended with additional sources.
* **Relevance**: Wikipedia pages provide detailed, field-specific content, enriching the dataset with contextual information about careers.
* **Ethical Considerations**: Public datasets and Wikipedia are openly accessible, ensuring ethical data usage with proper attribution (user agent specified in the script).

**3. Exploratory Data Analysis (EDA)**

**Overview**

EDA was conducted on all datasets to understand their structure, distributions, correlations, and quality. Key findings informed preprocessing and model design.

**Key EDA Insights**

1. **career\_guidance\_dataset**:
   * **Rows/Columns**: 1,000 rows, 22 columns (e.g., GPA, Career\_Interests, Entrepreneurship\_Suitability\_Score).
   * **Data Types**: Mix of numerical (e.g., GPA: float64) and categorical (e.g., Field\_of\_Study: object).
   * **Missing Values**: None, indicating good data quality.
   * **Stats**: GPA mean = 3.0, User\_Satisfaction mean = 5.46 (scale 1-10).
   * **Correlation**: Weak correlations between features (e.g., GPA and User\_Satisfaction: 0.061), suggesting diverse factors influence outcomes.
   * **Categorical Distribution**: Finance is the top recommended industry (230 instances).
2. **career\_path\_in\_all\_field**:
   * **Rows/Columns**: 9,000 rows, 17 columns (e.g., GPA, Coding\_Skills, Communication\_Skills).
   * **Data Types**: Numerical (e.g., GPA: float64) and categorical (e.g., Field: object).
   * **Missing Values**: None.
   * **Stats**: GPA mean = 3.74, Coding\_Skills mean = 1.99 (scale 0-4).
   * **Correlation**: Low correlations (e.g., GPA and Coding\_Skills: -0.01), indicating diverse skill contributions.
   * **Categorical Distribution**: Education is the most common field (667 instances).
3. **career\_pred**:
   * **Rows/Columns**: 7,525 rows, 26 columns (e.g., CGPA, programming\_skill, ROLE).
   * **Missing Values**: Significant missing values in ROLE (7,476 missing), requiring careful handling.
   * **Stats**: CGPA mean = 2.14 (scale 1-4), problem\_solving\_skill mean = 3.07.
   * **Correlation**: Moderate correlations (e.g., sslc and hsc: 0.59), suggesting academic consistency.
   * **Categorical Distribution**: Software Developer is the top role (12 instances).
4. **Data\_final**:
   * **Rows/Columns**: 105 rows, 11 columns (e.g., O\_score, Numerical\_Aptitude, Career).
   * **Data Types**: Mostly numerical (e.g., O\_score: float64), with Career as categorical.
   * **Missing Values**: None.
   * **Stats**: O\_score (Openness) mean = 7.29, Verbal\_Reasoning mean = 6.79.
   * **Correlation**: Strong negative correlation between Abstract\_Reasoning and A\_score (Agreeableness: -0.68), indicating personality traits influence aptitude.
   * **Categorical Distribution**: Environmental Scientist is the most common career (2 instances).
5. **google\_form\_responses**:
   * **Rows/Columns**: 1,028 rows, 29 columns (mostly qualitative, e.g., favorite subjects, skills).
   * **Missing Values**: Minimal (e.g., 5 missing in advice column).
   * **Categorical Insights**: Top skills include problem-solving (396 mentions), creativity (383). Top subjects include science (262) and art (234).
   * **Text Analysis**: Top words in skills (e.g., “problem,” “solving”) and interests (e.g., “learning,” “creating”) informed feature extraction.
6. **Student\_performance\_data**:
   * **Rows/Columns**: 2,392 rows, 15 columns (e.g., GPA, Extracurricular, Volunteering).
   * **Data Types**: Numerical (e.g., GPA: float64) and binary (e.g., Tutoring: int64).
   * **Missing Values**: None.
   * **Stats**: GPA mean = 1.91, Absences mean = 14.54.
   * **Correlation**: Strong negative correlation between GPA and Absences (-0.92), highlighting attendance’s impact on performance.
7. **Dataset Project 404**:
   * **Rows/Columns**: 3,600 rows, 21 columns (e.g., Linguistic, Musical, Job profession).
   * **Missing Values**: Significant missing values in Sr.No. (3,528) and Course (3,600), but key features are complete.
   * **Stats**: Interpersonal mean = 15.55, Naturalist mean = 11.04.
   * **Correlation**: Weak correlations (e.g., Linguistic and Interpersonal: 0.13), suggesting diverse intelligences.
   * **Categorical Distribution**: Astronomer is the most common profession (50 instances).
8. **Augmented Dataset**:
   * **Rows**: ~700 (targeting 50 samples per minority field).
   * **Features**: Aligned with schema (e.g., GPA, Creative\_Skills, Field).
   * **Purpose**: Addressed class imbalance for fields like Legal and Architecture.

**Justification**:

* **Comprehensive Analysis**: EDA revealed data quality, distributions, and correlations, guiding preprocessing (e.g., handling missing ROLE values in career\_pred).
* **Feature Selection**: Low correlations in most datasets justified including diverse features (e.g., GPA, skills, personality) to capture multifaceted profiles.
* **Class Imbalance**: High representation of Creative (2,403 samples) versus Legal (9 samples) necessitated augmentation, validated by EDA.

**4. Choosing Architecture**

**Architecture Overview**

The system uses a **pipeline-based machine learning architecture** with the following components:

1. **Preprocessing Pipeline**: Handles numerical feature scaling and imputation.
2. **Feature Selection Pipeline**: Applies VarianceThreshold and SelectKBest to reduce dimensionality.
3. **Classifier**: LightGBM (LGBMClassifier) for multi-class classification of career fields.
4. **API**: FastAPI for real-time inference and user interaction.

**Justification for Architecture**

* **Preprocessing Pipeline**:
  + **Reason**: Datasets contain numerical features (e.g., GPA, skills) with varying scales and occasional missing values. A ColumnTransformer with SimpleImputer (median strategy) and StandardScaler ensures robust handling of missing data and standardization for model stability.
  + **Industry Standard**: Pipelines ensure reproducibility and scalability, common in production ML systems.
* **Feature Selection Pipeline**:
  + **Reason**: With 27 features (e.g., GPA, Creative\_Skills, Openness), dimensionality reduction is critical to prevent overfitting and reduce computational cost. VarianceThreshold (threshold=0.01) removes low-variance features, and SelectKBest (k=20) selects the most predictive features using f\_classif.
  + **Industry Standard**: Feature selection is standard for high-dimensional datasets to improve model interpretability and performance.
* **LightGBM Classifier**:
  + **Reason**: LightGBM is a gradient boosting framework optimized for speed, memory efficiency, and handling imbalanced datasets. Its ability to manage multi-class classification (16 fields) and categorical features makes it ideal for this task.
  + **Comparison**: Alternatives like Random Forest or XGBoost were considered, but LightGBM’s faster training (histogram-based splits) and better handling of class imbalance (via class\_weight='balanced') made it superior.
  + **Industry Standard**: LightGBM is widely used in production for tasks requiring high accuracy and efficiency (e.g., Kaggle competitions, industry applications).
* **FastAPI**:
  + **Reason**: FastAPI is lightweight, asynchronous, and supports rapid development of RESTful APIs. It enables real-time career predictions, aligning with the system’s goal of user accessibility.
  + **Industry Standard**: FastAPI is popular for ML deployment due to its performance and ease of integration with Python-based ML models.

**Justification for Including Wikipedia Augmentation**:

* **Class Imbalance Handling**: The augmentation script increased samples for minority fields (e.g., Legal, Architecture), improving model fairness. This is critical as the original dataset had 2,403 Creative samples but only 7-11 samples for fields like Human Resources and Social Services.
* **Industry Relevance**: Data augmentation via external sources (e.g., Wikipedia) is a standard practice in NLP and ML when training data is limited, ensuring robust model generalization.

**5. Design**

**System Design**

The system comprises:

1. **Data Ingestion**: Loads and merges multiple datasets into a unified schema.
2. **Feature Extraction**: Uses NLP (tokenization, keyword matching) to extract features from text inputs.
3. **Preprocessing**: Imputes missing values, scales numerical features, and encodes categorical targets.
4. **Feature Selection**: Reduces dimensionality to improve model performance.
5. **Model Training**: Trains a LightGBM classifier with hyperparameter tuning and early stopping.
6. **API Deployment**: FastAPI serves predictions, returning top-3 careers with confidence scores and explanations.
7. **Output**: JSON response with career, field, confidence, and explanation, plus downloadable PDF reports (planned feature).

**Design Components**

* **Unified Schema**: 27 features (e.g., GPA, Creative\_Skills, Openness, Field) to standardize diverse datasets.
* **NLP Feature Extraction**: Tokenizes text inputs and matches against top\_words\_dict and field\_keywords to quantify skills and interests.
* **Feature Selection**: VarianceThreshold and SelectKBest ensure only relevant features are used.
* **Model Pipeline**: Combines preprocessing, feature selection, and classification for end-to-end training.
* **API**: Handles user inputs, processes them through the pipeline, and returns structured predictions.

**Justification**:

* **Unified Schema**: Standardizing features across datasets ensures consistency and enables seamless merging, critical for training a single model.
* **NLP Feature Extraction**: Captures qualitative data (e.g., interests, skills) from text, addressing the problem of simplistic questionnaires.
* **Feature Selection**: Reduces noise and computational cost, improving model generalization and interpretability.
* **Pipeline Design**: Ensures modularity and reproducibility, aligning with MLOps best practices.
* **API**: Provides a user-friendly interface, supporting real-time interaction and scalability for production use.

**6. Algorithm Selection**

**Algorithm: LightGBM Classifier**

* **Description**: LightGBM is a gradient boosting framework that uses tree-based learning with histogram-based splits for efficiency.
* **Hyperparameters**:
  + Tuned via RandomizedSearchCV: n\_estimators, learning\_rate, max\_depth, num\_leaves, min\_child\_samples, subsample, colsample\_bytree.
  + Early stopping with 20 rounds to prevent overfitting.
  + class\_weight='balanced' to address class imbalance.
* **Feature Selection**: VarianceThreshold (threshold=0.01) and SelectKBest (k=20) to select top features.

**Justification**

* **Performance**: LightGBM’s histogram-based approach reduces memory usage and training time, critical for large datasets (15,000+ rows after merging).
* **Class Imbalance Handling**: The class\_weight='balanced' parameter and SMOTE oversampling ensure fair predictions across minority classes (e.g., Legal, Human Resources).
* **Multi-Class Support**: LightGBM handles the 16-class problem (fields like Technology, Healthcare) effectively.
* **Scalability**: Faster training and inference compared to XGBoost or Random Forest, making it suitable for real-time API predictions.
* **Interpretability**: Feature importances (e.g., Interests: 11,984, Analytical\_Skills: 9,799) provide insights into prediction drivers, enhancing explainability.

**Alternatives Considered**:

* **Random Forest**: Slower training and less effective for imbalanced data.
* **XGBoost**: Similar performance but higher computational cost and less optimized for large datasets.
* **SVM**: Inefficient for multi-class problems and large datasets.
* **Neural Networks**: Require more data and computational resources, less interpretable.

**SMOTE and Wikipedia Augmentation**:

* **SMOTE**: Applied to oversample minority classes with dynamic k\_neighbors based on class size, ensuring robust training.
* **Wikipedia Augmentation**: Added ~700 samples for minority fields, improving class balance and model fairness.

**Justification for Including SMOTE and Augmentation**:

* **Class Imbalance**: SMOTE and Wikipedia augmentation address the skewed distribution (e.g., 2,403 Creative vs. 7 Social Services), ensuring the model learns from underrepresented fields.
* **Industry Practice**: SMOTE is a standard technique for imbalanced datasets, and external data augmentation (e.g., Wikipedia) is common in NLP tasks.

**7. Evaluation**

**Metrics**

* **Accuracy**: 96.86%, indicating high overall correctness.
* **Hit Rate@3**: 98.82%, showing the model’s ability to include the true field in the top-3 predictions.
* **F1-Score (Macro)**: 0.46, reflecting challenges with minority classes due to imbalance.
* **Classification Report**:
  + High precision/recall for Creative (1.00/1.00) and Other (1.00/1.00) due to sufficient samples.
  + Lower performance for minority classes (e.g., Legal: 0.12 precision, Social Services: 0.25 precision), highlighting imbalance issues.
* **Confusion Matrix**: Visualized as a heatmap, showing correct predictions for Creative and Other, with some misclassifications for minority classes.

**Feature Importances**

* Top features: Interests (11,984), Analytical\_Skills (9,799), Communication\_Skills (8,464).
* **Justification**: High importance of Interests aligns with the problem statement’s emphasis on personalized recommendations based on user preferences. Analytical and Communication Skills are critical across fields, supporting their high importance.

**Evaluation Insights**

* **Strengths**:
  + High accuracy and Hit Rate@3 demonstrate robust performance for well-represented classes.
  + Feature importances align with domain knowledge (e.g., Interests driving career fit).
* **Weaknesses**:
  + Low F1-scores for minority classes (e.g., Legal, Social Services) indicate imbalance issues, despite SMOTE and augmentation.
  + Repeated entries in the classification report (e.g., Healthcare, Legal) suggest a bug in logging or data processing, requiring investigation.

**Justification**:

* **Metrics Choice**: Accuracy and Hit Rate@3 are standard for multi-class problems, while F1-macro highlights performance on minority classes, critical for fairness.
* **Feature Importance Analysis**: Guides explanation generation, ensuring transparency and user trust, aligning with industry standards for interpretable ML.

**8. Recommendations**

**Model Improvements**

1. **Address Class Imbalance Further**:
   * **Recommendation**: Collect more real-world data for minority classes (e.g., Legal, Social Services) via surveys or partnerships with educational institutions.
   * **Justification**: Despite SMOTE and Wikipedia augmentation, low F1-scores for minority classes indicate insufficient representation. Real-world data would improve authenticity and model fairness.
2. **Fix Classification Report Bug**:
   * **Recommendation**: Investigate and correct repeated entries in the classification report (e.g., multiple Healthcare, Legal entries).
   * **Justification**: Ensures accurate evaluation and reporting, critical for production systems.
3. **Enhance Feature Extraction**:
   * **Recommendation**: Incorporate advanced NLP techniques (e.g., BERT embeddings) for text feature extraction.
   * **Justification**: Current keyword-based extraction may miss nuanced user inputs. BERT can capture contextual relationships, improving feature quality.

**API Enhancements**

1. **User Feedback Loop**:
   * **Recommendation**: Add an endpoint for users to provide feedback on predictions, enabling continuous model retraining.
   * **Justification**: Feedback loops are standard in production ML systems to improve model accuracy and user satisfaction over time.
2. **PDF Report Generation**:
   * **Recommendation**: Implement PDF report generation for career plans, as mentioned in the problem statement.
   * **Justification**: Enhances user experience by providing tangible outputs for sharing with advisors, aligning with the system’s goal of practical guidance.

**Deployment Considerations**

1. **Scalability**:
   * **Recommendation**: Deploy the API on a cloud platform (e.g., AWS, GCP) with auto-scaling to handle varying user loads.
   * **Justification**: Ensures reliability and performance for a growing user base, a standard practice for production APIs.
2. **Monitoring**:
   * **Recommendation**: Implement logging and monitoring (e.g., Prometheus, Grafana) for API performance and prediction accuracy.
   * **Justification**: Enables proactive issue detection and model drift monitoring, critical for maintaining system reliability.

**User Experience**

1. **Interactive Quizzes**:
   * **Recommendation**: Integrate interactive quizzes for skill assessment, as outlined in the problem statement.
   * **Justification**: Enhances user engagement and provides additional data for feature extraction, improving personalization.
2. **Community Forum**:
   * **Recommendation**: Develop a forum for users to connect with peers and mentors, as proposed.
   * **Justification**: Fosters inspiration and guidance, especially for marginalized communities, addressing the lack of relatable role models.

**Justification for Recommendations**:

* **Model Improvements**: Address identified weaknesses (e.g., minority class performance, logging issues) to enhance fairness and reliability, aligning with industry standards for robust ML systems.
* **API Enhancements**: Feedback loops and PDF reports improve user interaction and practical utility, supporting the system’s goal of actionable guidance.
* **Deployment and UX**: Scalability, monitoring, and interactive features ensure a production-ready system that meets user needs, following MLOps and user-centric design principles.

**9. Conclusion**

**CareerCompass** is a robust, industry-standard machine learning system that addresses the challenges of career planning with personalized, data-driven recommendations. By integrating diverse datasets, employing LightGBM for efficient classification, and deploying via FastAPI, it delivers accurate and user-friendly career guidance. The use of SMOTE and Wikipedia augmentation ensures fairness across underrepresented fields, while high accuracy (96.86%) and Hit Rate@3 (98.82%) demonstrate strong performance. Recommendations for further data collection, advanced NLP, and user feedback loops will enhance the system’s effectiveness, making it a valuable tool for students seeking informed career paths.

**Note**: The Wikipedia augmentation script is included in the documentation as it directly addresses class imbalance, a critical challenge identified in EDA. Its inclusion aligns with industry practices for handling imbalanced datasets and ensures transparency in data collection methods.