OWNML MACHINE LEARNING CANVAS Designed for: Designed by: Date: Iteration:

PREDICTION TASK

- **Type of task**: Supervised Regression
- **Entity**: Individual student or job-seeker profile
- **Predicted outcome**: Annual starting salary in USD
- **When observed**: After the individual secures a job and receives a salary offer

DECISIONS

Actionable output: Predicted salary is used to:

- Set realistic expectations for students
- Help career counselors offer tailored advice
- Guide students toward skill-building decisions (e.g., gaining internships or pursuing further education)

How: Via an interactive web app where users enter profile details (age, education, job title, experience) and instantly receive a salary prediction

Parameters: Real-time input, instant response, and suggested profile improvements based on salary gaps

VALUE PROPOSITION

End beneficiary: Fresh graduates, jobseekers, university career services, training institutes

Pain points addressed:

- Unclear salary benchmarks
- Mismatched expectations in career planning
- Lack of evidence-based career guidance

Integration & UI:

- Streamlit-based web interface for individual use
- Future integration with university career portals or job prep platforms
- Potential chatbot or API for guided Q&A salary predictions

DATA COLLECTION

Initial sourcing:

- Kaggle's publicly available structured Salary Dataset
- Real-world job and education data, manually curated and cleaned

Update strategy:

- Periodic scraping or API integration with job boards (e.g., LinkedIn, Indeed) in future iterations
- Consider cost-effective crowdsourced validation or surveys for fresh graduate outcomes

DATA SOURCES

External sources:

- Kaggle Salary Dataset (structured CSV)
- Future potential: APIs from LinkedIn, Glassdoor, PayScale, or government wage data

Internal sources (future):

- Student databases from academic institutions
- Career services intake forms

IMPACT SIMULATION

Cost/gain of correct predictions: Accurate predictions improve student trust, better job targeting, and resource alignment

Cost of incorrect predictions:

- Underestimation could demotivate students
- Overestimation might lead to unrealistic goals or dissatisfaction

Pre-deployment simulation:

- Validated model using holdout test sets (20%) and cross-validation

Deployment criteria:

- Minimum $R^2 > 0.95$
- MAE under $5,000

Fairness constraints:

- Avoid bias based on gender, race, or age (planned in future iterations)
- Audit feature contributions using SHAP to detect unfair influence

MAKING PREDICTIONS

Mode: On-demand (real-time) via Streamlit UI

Frequency: As often as user submits input

Latency: Few milliseconds (lightweight Random Forest inference)

Resources: Local CPU for Streamlit app, scalable to cloud GPU/VM instance if hosted

BUILDING MODELS

Number of models: One primary regression model (Random Forest Regressor)

Update frequency:

- Retrain quarterly or when new data is acquired
- Reassess feature relevance every 6 months

Time constraints: Short training time (~minutes), retrain offline

Resources: Local machine or cloud-based training using scikit-learn or XGBoost pipelines

FEATURES

Number of models: One primary regression model (Random Forest Regressor)

Update frequency:

- Retrain quarterly or when new data is acquired
- Reassess feature relevance every 6 months

Time constraints: Short training time (~minutes), retrain offline

Resources: Local machine or cloud-based training using scikit-learn or XGBoost pipelines

MONITORING

KPIs (Model-centric):

- R^2 score > 0.95
- MAE < $3,000
- Low variance across demographic groups

KPIs (Business-centric):

- % of users reporting improved career planning
- % of students following advice and reporting higher satisfaction
- Number of accurate predictions verified by actual salaries (future phase)

Review cadence:

- Monthly model performance reports
- Quarterly feedback loops from users and institutions