**Capstone Project Summary**

**Introduction**

This project aims to develop a machine learning model to predict whether data scientists are likely to seek job changes, aiding in talent retention and recruitment planning.

**Objectives**

1. Predict employees likely to leave using machine learning
2. Enable HR teams to optimize resource allocation.
3. Provides actionable insights for retention strategies

**Imported Necessary Libraries**

Imported necessary libraries like:

1. NumPy: We import the NumPy library in Python to efficiently perform numerical computations, especially when working with large datasets or multi-dimensional arrays.
2. Pandas: It excels at handling labelled, relational, or tabular data, making complex data tasks intuitive and efficient
3. Matplotlib: To create visualizations.
4. Seaborn: We import because it makes creating beautiful, statistically-informed visualizations in Python easier and more intuitive.

**Loading Data**

Loaded training dataset for model training.

Has 19158 Rows and 14 columns

**Visualization**

1. **Count of Target Variables**

Represents about 75.1% of the data are not looking for a job change

Represents about 24.9% of the data are looking for a job change

Approximately three-quarters of the population in this dataset are categorized as "Not Looking for a job change," while roughly one-quarter are "Looking for a job change."

1. **Distribution of Company Type**

The pie chart illustrates the distribution of company types within the dataset:

Pvt Ltd (Private Limited): 75.4% – by far the most common company type.

Funded Startup: 7.7%

Public Sector: 7.3%

Early-Stage Startup: 4.6%

NGO: 4.0%

Other: 0.9% – a very small fraction.

**Private Limited companies dominate**, comprising three-quarters of all entries. **Startups collectively make up around 12.3%** (funded + early-stage), showing moderate representation. **Public Sector and NGOs** together account for roughly 11.3%. **Other** is nearly negligible at under 1%.

1. **Distribution of Continues Variables**

**City Development Index**

Bimodal distribution: Most cities cluster around 0.6 and 0.9 on the development index.

The first peak (0.9) is much larger—over half of the data.

A smaller secondary peak occurs around 0.6.

Very few cities fall below 0.5 or in the 0.7–0.8 range.

Implication: The dataset includes primarily highly developed cities, with a smaller segment of moderately developed ones and few low-development ones.

**Training Hours**

Right-skewed distribution: Most individuals receive low to moderate training hours, with a long tail reaching higher values.

The majority (60–80%) complete 0–60 hours of training.

A gradual decline in frequency from 100 to 200+ hours, with very few exceeding 300 hours.

Implication: Training is concentrated among shorter programs, with occasional outliers undergoing extensive training.

**Summary**

The city development index shows two distinct clusters (moderate and high development), heavily leaning toward high development.

The training hours follow a typical skewed pattern: many individuals complete limited hours while a small number undergo extensive training.

1. **Distribution of Categorical Variables**

**Gender**

* Male: Vast majority, around 13,000.
* Female: Small minority (1,000).
* Other: Very rare (almost negligible).

**Relevant Experience**

* Has relevant experience: Roughly 13,000.
* No relevant experience: About 6,000.
* Takeaway: Twice as many respondents have relevant experience compared to those without.

**Enrolled University**

* No enrolment: Predominant 13,000.
* Full-time course: ~4,000.
* Part-time course: Very few.

**Education Level**

* Graduate: Largest group (~12,000).
* Master’s: ~5,000.
* High School: Few (~2,000).
* PhD and Primary School: Very small numbers.

**Major Discipline**

* STEM: Dominant (~14,000).
* Other categories (Humanities, Business, Arts, No Major): Minor representation.

**Work Experience (Years)**

* >20 years: Highest count (~3,000).
* 5 years: ~1,500.
* Others gradually taper down; <1 year is also notable.
* Experience levels like 10–15 years appear moderately represented.

**Overall Highlights**

* Gender imbalance: dataset is male-dominated.
* Strong experience: majority have relevant experience.
* Education skewed: most have STEM backgrounds and graduate/advanced education.
* University enrolment: many are no longer enrolled.
* Experienced cohort: wide spread in years of experience, with a notable number having over 20 years.

1. **Continues features vs Target Features**

**City Development Index by Job‑Change Status**

* Not Looking group shows two prominent peaks:
  + One around 0.90–0.95, indicating many people content with high-development cities.
  + A smaller bump near 0.65–0.70, suggesting some live in moderately developed cities without searching.
* Looking group shows:
  + One peak near 0.60–0.65, implying job-seekers often live in *mid-developed* cities.
  + Another around 0.90, but less pronounced than the “Not Looking” peak.

Insight: Those staying put are more likely in well-developed cities, whereas job-seekers are skewed toward less-developed areas.

**Training Hours by Job‑Change Status**

* Both groups have very similar training-hours distributions.
  + A primary cluster around 20–60 hours.
  + A long tail stretching to 200+ hours, though densities are low there.
* Looking group shows a slight bump around the mid-range, but the overlap is substantial.

Insight: Training hours don’t strongly differ by job‑search status—both groups invest similar amounts, with minor differences around medium effort levels.

**Overall Summary**

* City Development Index shows a clear difference: those not seeking jobs are more likely to be in highly developed cities, while job-seekers tend toward less-developed ones.
* Training Hours are largely similar, so training effort doesn’t clearly distinguish between the two groups.

1. **Categorical Features vs Target Variable**

**Gender**

* Male overwhelmingly dominates both groups, but still largely Not Looking.
* Female and Other genders follow similar proportions—higher counts for Not Looking.

**Relevant Experience**

* Majority have relevant experience.
* Job-seekers with no relevant experience form a substantial subgroup—higher share of “Looking” than in the “Has relevant experience” group.

**University Enrolment**

* Most users are not currently enrolled.
* Among those looking, a higher proportion are enrolled (full-time or part-time) compared to “Not Looking.”

**Education Level**

* Graduates make up the biggest chunk of both statuses.
* Masters, followed by High School, then PhD/Primary School in decreasing order.
* Job-seekers at every education level have noticeably fewer overall counts but similar relative distributions.

**Major Discipline**

* STEM majors dominate both categories—especially in “Not Looking.”
* Other majors (Business, Arts, Humanities, No Major, Other) collectively form smaller shares.

**Experience (Years)**

* Distribution spans many experience categories.
* “Not Looking” peaks at higher experience values; “Looking” sees consistent representation across nearly every experience bucket, especially mid-range (3–10 years).

**Company Size**

* Most currently employed in medium-to-large firms (50–99, 100–500 employees).
* A notable fraction in smaller (<10) or larger (10,000+) companies shows up more in “Looking” vs “Not Looking,” but overall “Looking” counts are smaller across all sizes.

**Company Type**

* Private Ltd is by far the most common type, especially for those Not Looking.
* Other company types (Startups, Public Sector, NGOs) have smaller yet consistent presence in both groups.

**Last New Job**

* Many users recently started a new job (coded ‘1’).
* Among job-seekers, there are still sizable counts across all “last new job” categories, notably “>4 years ago,” “never,” and “2.”

**Key Takeaways**

* Demographics (gender, education, major) look similar across both groups, with STEM and graduates dominating.
* Relevant experience and ongoing university enrolment appear linked to job-seeking activity.
* Company context (size and type) shows job-seekers are distributed across diverse workplaces, though always fewer overall than non-seekers.
* Experience histories—both in years and time since last job change—show job-seekers are spread across, with no single dominant pattern.

These features can provide insights into which factors may influence an individual’s likelihood of seeking a job change.

1. **Correlation Matrix:**

Key Correlations with Target (Looking for a New Job)

* City Development Index: Moderately negative correlation (–0.34), suggesting individuals in less-developed cities are more likely to be job-seeking.
* Relevant Experience: A small positive correlation (+0.13), indicating those with relevant experience have a slightly higher tendency to look for new jobs.
* Enrolled in University: Slight negative correlation (–0.15), meaning students tend to job-seek a bit less.
* All other features (gender, education\_level, major, experience years, company\_size/type, training\_hours, last\_new\_job) show very weak correlations (typically between –0.06 and +0.09), implying minimal direct linear impact on job-seeking status.

Interpretation

* City Development stands out as the most influential single factor: living in a better-developed city strongly associates with *not* looking for a job.
* Other variables contribute only marginally, suggesting job-seeking behavior isn’t strongly tied to most of these features on their own.
* Likely, interactions or non-linear relationships (e.g., city + experience, education + company type) could reveal additional patterns beyond what a simple linear correlation shows.

Summary

The most notable signals from the matrix:

* Living in less-developed cities increases the likelihood of job-seeking.
* Having relevant experience slightly raises job-seeking probability.
* University enrolment mildly reduces it.
* Other factors show negligible linear influence.

To better understand job-seeking behaviour, exploring complex relationships or conducting multivariate modelling would be the next valuable steps.

**Dropping Null Values**

Missing values were there only in categorical columns so these values were removed using **Ffill** and **Bfill** method.

And then applied DropNA.

**Dropping Columns**

All features in the dataset appears to be relevant except **Employee ID and City**. So, these two columns have been removed using **Drop** Function.

| **Column Name** | **Reason to Keep** |
| --- | --- |
| city\_development\_index | Numerical indicator of opportunity in the city |
| gender | May influence job switching behaviour (though use carefully due to bias concerns) |
| relevent\_experience | Directly indicates skill alignment |
| enrolled\_university | Reflects ongoing education or upskilling |
| education\_level | Education impacts job market readiness |
| major\_discipline | Academic background relevance |
| experience | Work experience is crucial |
| company\_size | May influence satisfaction and stability |
| company\_type | Different company types offer different prospects |
| last\_new\_job | Job hopping behaviour predictor |
| training\_hours | More hours might reflect re-skilling intent |

The dataset contains **job seekers' information** and whether they are looking for a job or not (target column: 1 = looking for a job, 0 = not looking).

**Target Variable**: target (binary classification)

| **Column** | **Reason not to keep** |
| --- | --- |
| enrollee\_id | Just a unique identifier – no predictive value. |
| city | High cardinality (e.g., 123+ unique values like city\_103, etc.). It leads to many dummy variables if one-hot encoded, which can cause overfitting or sparsity issues. Alternative city\_development\_index already captures location quality. |

**Label Encoding**

Applied Label Encoding on:

1. Gender
2. Relevant Experience
3. Enrolled University
4. Education Level
5. Major Discipline
6. Experience
7. Company Size
8. Company Type
9. Last New Job

**Correlation Matrix**

**City Index:** Lower city development index links to higher attrition.

**Training Hours:** Fewer training hours increase attrition risk.

**Demographics:** Younger employees show higher attrition rates.

**Variables somewhat correlated with target:**

* city\_development\_index: -0.34
  + Negative correlation – people from more developed cities are less likely to change jobs.
* relevent\_experience: +0.13
  + Having relevant experience slightly increases the likelihood of job change.
* enrolled\_university: -0.15
  + Those enrolled in university are slightly less likely to switch jobs.

**Correlation with Target**

target 1.000000

relevent\_experience 0.128540

major\_discipline 0.019928

company\_size 0.000322

experience -0.007955

company\_type -0.012414

gender -0.017311

last\_new\_job -0.018158

training\_hours -0.021489

education\_level -0.084458

enrolled\_university -0.151776

city\_development\_index -0.341771

**Handling Imbalance data by Oversampling**

Used SMOTE function so that Training data becomes balanced, helping the model learn both classes effectively.

**Predictive Modelling**

Divided features into X and Y variables

X:

* Identification: Employee ID, City
* Demographic: Gender, Education, Major Discipline
* Professional Info: Experience, Last New Job, Company Type, Company Size

Y: Job Seeking Indicator: Target (0: Not Looking for job change, 1: Looking for a job change)

**Feature Scaling:** Important for sensitive algorithms. In this StandardScalar has been used

**Train Test Split:**

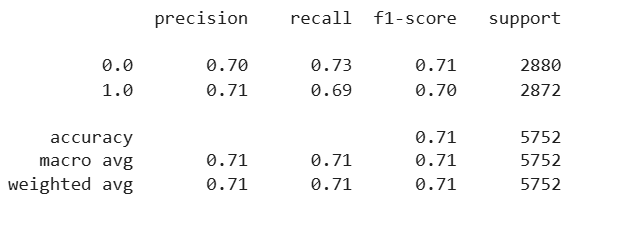
Data is divided into Training (80%) and Testing (20%) sets.

Setting a random state at 42 ensures consistent results.

**Model Selection**

* 1. **Logistic Regression:**

Commonly used for binary classification problems. It is preferred because it provides a simple and efficient way to model the relationship between the independent variables and the probability of a certain outcome.



**True Negatives (TN)**: 2095

**False Positives (FP)**: 785

**False Negatives (FN)**: 904

**True Positives (TP)**: 1968

**Interpretation:**

* The model delivers **balanced accuracy** (~71%) for both classes.
* Similar precision and recall values suggest **no strong bias** to either predicting "Looking" or "Not Looking."
* **F1-scores** (~0.71) indicate a consistent trade-off between precision and recall.
* **Macro-averages match weighted averages**, reflecting an evenly distributed dataset and stable per-class performance.

**Takeaways:**

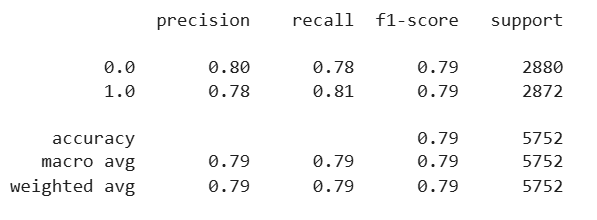
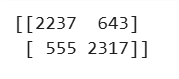
* Performance is **solid but improvable**, especially around misclassified instances (~25–30% errors).
* If reducing false positives or false negatives is critical, consider **adjusting decision thresholds**.
* Further gains may come from **feature engineering** or **model tuning** to better distinguish between classes.

**Summary:**

Classifier achieves about **71% accuracy**, with well-balanced performance across both classes as evidenced by similar precision, recall, and F1 values. It shows reliable but not exceptional predictive power.

* 1. **Random Forest:**

It is a robust supervised algorithm suitable for both regression and classification tasks.



**True Negatives**: 2237

**False Positives**: 643

**False Negatives**: 555

**True Positives**: 2317

**Key Insights**

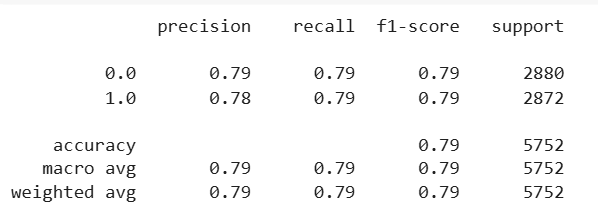
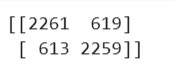
* **Solid overall performance**: With **79% accuracy**, the model correctly classifies about 4 out of 5 cases.
* **Balanced across classes**:
  + Precision (~0.79) and recall (~0.80) are closely aligned for both target classes.
  + The F1‑score (~0.79) indicates a harmonious balance between precision and recall.
* **Harmonic mean explained**:
  + F1‑score, a harmonic mean of precision and recall, is effective for balancing both metrics, especially important when either varies significantly.
* **A useful metric for balanced datasets**:
  + Since both classes have nearly equal support, the **macro and weighted averages** align — a sign of consistent performance across both groups.

**Overall Summary**

Classification model demonstrates **robust and balanced performance** with about **79% accuracy** and nearly identical precision, recall, and F1‑scores for both classes. The strong alignment across metrics suggests it's well-balanced and reliable for both predicting “Looking” and “Not Looking.”

* 1. **XGBoost:**

XGBoost (short for Extreme Gradient Boosting) is a powerful machine learning algorithm based on gradient boosting. It is widely used for structured (tabular) data in classification, regression, and ranking tasks.



**True Negatives (TN)**: 2237

**False Positives (FP)**: 643

**False Negatives (FN)**: 555

**True Positives (TP)**: 2317

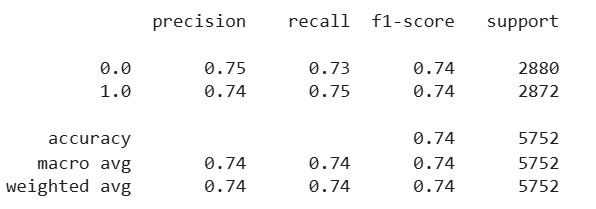
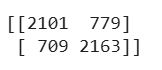
**Summary & Insight**

* **Balanced performance**: Precision, recall, and F1 are tightly clustered around 0.79 for both classes, signalling reliable and unbiased predictions.
* **Error distribution**:
  + About 21% of predictions are incorrect—405 false positives and 278 false negatives.
  + These are mirrored across the two classes (~18–22% misclassification rates).
* **F1‑score utility**: Ideal for your balanced dataset, giving a meaningful single metric reflecting both types of classification errors.

**Conclusion**: Model performs well with **79% accuracy and F1-score**, maintaining a strong balance across both classes, making it reliable for real-world prediction tasks.

* 1. **Decision Tree:**

Commonly used for classification because they are simple, computationally efficient, efficient, and effective in handling high- dimensional data. Works best for categorical independent columns.



**True Negatives (TN)**: 2,101

**False Positives (FP)**: 779

**False Negatives (FN)**: 709

**True Positives (TP)**: 2,163

**Key Takeaways**

* **Balanced yet moderate performance**:
  + Precision, recall, and F1 are fairly synchronized (~0.74), indicating no big performance gap between predicting “looking” or “not looking.”
* **Areas for improvement**:
  + With ~26% error rate, there’s room to reduce false positives (779) and false negatives (709).
* **Why F1‑Score matters**:
  + Accuracy alone can be misleading, especially with even class distribution. The F1‑score provides a single, robust metric for balanced error evaluation, penalizing cases where precision and recall aren’t aligned.

**In summary**: Classifier achieves **74% accuracy and F1**, with balanced precision and recall around 0.74—solid performance with clear scope for optimization.

**Comparison of Results**

**1. Cross-validation (CV) results for Logistic Regression model:**

Cross-Validation Performance

* Individual fold scores:
  + Fold 1: 65.0%
  + Fold 2: 68.8%
  + Fold 3: 72.8%
  + Fold 4: 72.1%
  + Fold 5: 72.7%

Mean CV accuracy: 70.3%

Insights

* Performance consistency: Accuracy steadily improves across folds, indicating stable model behavior.
* Modest low-end performance: The lowest fold scored 65%, suggesting the model underperforms mildly in certain data splits.
* Overall effectiveness: A mean accuracy above 70% signifies a reasonably predictive logistic regression, though there's room for enhancement.

**2. Cross-validation (CV) results for Random Forest:**

Cross-Validation Performance

* Fold scores:
  + Fold 1: 68.1%
  + Fold 2: 73.4%
  + Fold 3: 83.9%
  + Fold 4: 83.5%
  + Fold 5: 83.8%
* Mean CV accuracy: 78.5%

Insights

* Higher average accuracy: Random Forest (78.5%) outperforms Logistic Regression (70.3%), demonstrating stronger overall predictive power.
* Increased stability in later folds: After Fold 2, accuracy consistently stays above ~83%, indicating robust and consistent model performance.
* Variability across folds: The gap between early (~68%) and later (~84%) scores suggest dataset splits that could affect performance—still, variability is smaller than in Logistic Regression.

Interpretation

* Stronger model: Ensemble-based Random Forest delivers better and more stable accuracy, fitting well across multiple folds.
* Generalization confidence: Cross-validation confirms its reliability in real-world scenarios.
* Aligns with benchmarks: Studies find Random Forest often outperforming logistic regression in ~69% of cases.

**3. Cross-validation (CV) results for XGBoost:**

Cross‑Validation Performance

* Fold scores:
  + Fold 1: 64.8%
  + Fold 2: 73.8%
  + Fold 3: 83.9%
  + Fold 4: 83.9%
  + Fold 5: 83.4%
* Mean CV accuracy: 77.9%

Insights

* High overall accuracy: At 77.9%, XGBoost performs significantly better than Logistic Regression (~70%) and is comparable to Random Forest (~78.5%).
* Stable performance in later folds: Folds 3–5 consistently score above ~83%, demonstrating strong model reliability.
* Early fold volatility: Fold 1 (~65%) is lower, suggesting sensitivity to certain data splits—this is typical, and mean performance remains robust.

Interpretation

* Ensemble power: XGBoost leverages gradient boosting to focus on difficult-to-classify instances, improving overall metrics—especially for precision-critical tasks.
* Comparable to Random Forest: While RF marginally outperformed in your CV (78.5% vs. 77.9%), XGBoost often slightly leads in accuracy and handles class imbalance more effectively.
* Sensitivity to hyperparameters: XGBoost performance hinges on tuning parameters like tree depth, learning rate, and early stopping—cross-validation helps identify optimal settings.

**4. Cross-validation (CV) results for Decision Tree:**

Cross‑Validation Performance

* Fold scores:
  + Fold 1: 61.8%
  + Fold 2: 70.2%
  + Fold 3: 81.1%
  + Fold 4: 80.8%
  + Fold 5: 79.4%
* Mean CV accuracy: 74.7%

Insights

* Moderate performance: An overall accuracy of 74.7% places the Decision Tree between LogReg (~70%) and ensemble models (RF ~78.5%, XGBoost ~77.9%).
* Fold variability: Low performance on fold 1 (~62%) rising to ~81% in later folds suggest sensitivity to data splits—a known trait due to high variance.
* Evidence of overfitting risk: Decision Trees often overfit when unconstrained, leading to less generalizable performance compared to ensemble models.

Interpretation

* Easy to understand: Decision Trees offer interpretability—ideal for exploring feature importance and decision rules.
* Less robust: They tend to be unstable and prone to large performance swings with small data changes.
* Better with pruning/tuning: Regularization (depth, leaf size) can help reduce overfitting and variance.

**Model Performance Summary:**

| **Model** | **Training Accuracy** | **CV Accuracy** |
| --- | --- | --- |
| **Logistic Regression** | 70.24% | 70.27% |
| **Random Forest** | 79.26% | 78.54% |
| **XGBoost** | 79.12% | 77.95% |
| **Decision Tree** | 72.76% | 74.65% |

* Best performers:  
  Random Forest and XGBoost lead in both training and CV—significantly outperforming Logistic Regression. This aligns with the common trend where tree-based ensembles excel at capturing complex patterns in data.
* Consistency check:
  + Random Forest: Slight drop from 79.26% (training) to 78.54% (CV) suggests low overfitting.
  + XGBoost: Similar drop (~79.1% to ~77.9%) indicates stable generalization across folds.
* Interpretable but limited:  
  Decision Tree shows moderate accuracy (~74–75%) and a noticeable variance between training and CV, underscoring its susceptibility to overfitting compared to its ensemble versions.
* Simpler baseline:  
  Logistic Regression offers baseline performance (~70%), but is outstripped by ensemble methods on this dataset—consistent with typical findings.
* **Top accuracy**: Random Forest (~78.5%) & XGBoost (~77.9%)
* **Moderate consistency**: Both maintain strong CV performance with minimal overfitting
* **Simpler options**: Decision Tree (~74.7%) offers interpretability; Logistic Regression (~70%) serves as a solid baseline.

**Handling Testing Dataset**

**1. Loaded testing dataset** for model evaluation. Has 2129 Rows and 13 columns.

2. **Missing values** were there only in categorical columns so these values were removed

using **Ffill** and **Bfill** method and then applied DropNA.

3. Applied **Label Encoding** on:

Gender

Relevant Experience

Enrolled University

Education Level

Major Discipline

Experience

Company Size

Company Type

Last New Job

4. **Model Evaluation:**

* Extracts columns from test dataset.
* Ensures the model receives data in the precise format it was trained on.
* Uses trained **Random Forest model (model\_rf)** to predict labels on the test set. Outputs a predictions array—likely containing binary values (e.g., 0 or 1) that indicate the predicted target class.

**Summary:**

* Generating predictions for unseen (test) data using your trained Random Forest.
* These predictions are based on carefully curated input features, ensuring consistency with training.
* The predictions array is ready to evaluate performance or to be used in further pipelines (e.g., submission, analysis).

**Conclusion:**

* **Why Random Forest?**

Accuracy is 0.79 (Highest among all)

Precision (0.79):

Of predicted “job‑seeking” candidates, 79% are correctly identified.

Recall (0.81):

Of all actual job‑seekers, the model identifies 81%.

* **Which to Prioritize?**

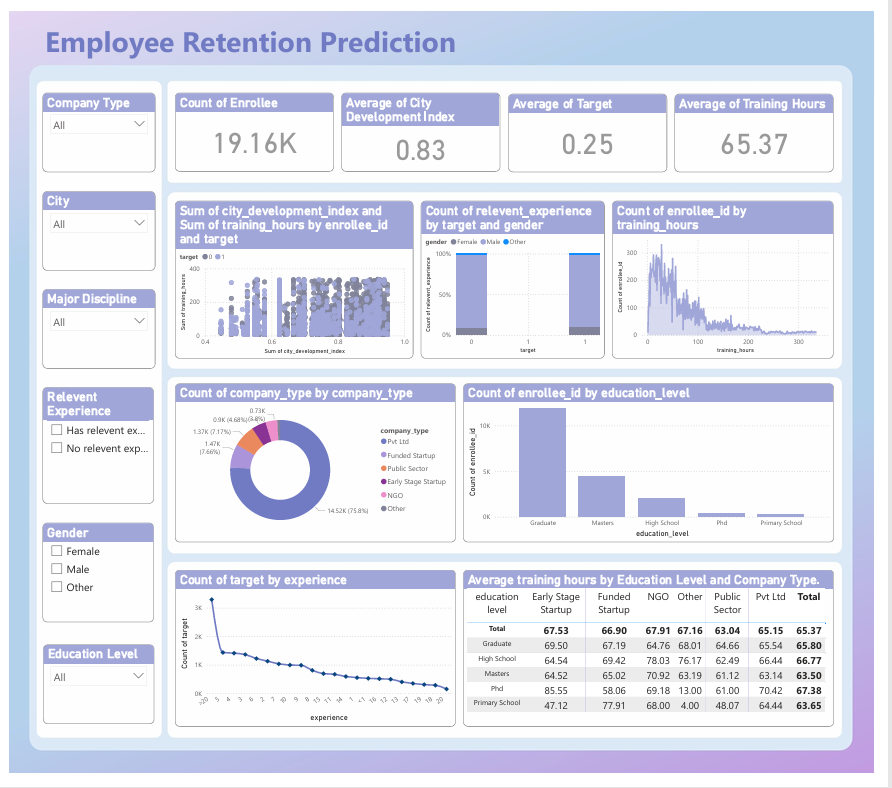
Recall Context: High‑stakes hiring → missing good candidates is costlier than some irrelevant outreach.

* **Key Findings**

Accuracy of all the models are almost same but Random Forest is a good choice due to non-linearity in data, Ability to handle missing and categorical variables (after preprocessing)

* Dataset is imbalanced, with only 25% likely to change jobs.
* Majority of candidates are male, graduates, and from STEM fields.
* Candidates with relevant experience are more likely to switch jobs.
* Education level and university enrollment show predictive significance.
* Those from less developed cities are more likely to be seeking a job change.
* **Limitations:**
* Significant missing values in key features like gender and company type.
* Imbalanced target variable may affect prediction accuracy.
* Lack of contextual factors like salary, role, and job satisfaction.
* **Future Research:**
* Collect additional features like salary, job role, and job satisfaction for deeper insights.
* Conduct fairness audits to ensure the model performs equally across demographics.
* Use time-series or longitudinal data to track employee behavior over time.
* Build a real-world feedback loop to continuously retrain and improve the model.
* Made Dashboard using Power BI.
* Deployed whole dataset using streamlit library.

**Dashboard**



**Deployment**