

PRWP 2025: Innovation in the World Bank Group Portfolio

Research pipeline analyzing innovation patterns in World Bank Group project evaluations (ICRRs and PPARs). Combines rule-based keyword matching, LLM assistance, and expert validation to identify and classify innovation across 7,574 projects.

Overview

This package reproduces all quantitative exhibits in the paper. It processes the IEG ICRR-PPAR Lessons dataset through a 17-step pipeline to produce 28 tables, 4 figures, and a set of inline statistics. Six tables and one figure in the paper are qualitative or synthesis content and are not reproducible from code alone (see [List of Exhibits](#)).

Computational Requirements

Requirement	Details
Operating system	macOS, Linux, or Windows (developed and tested on macOS)
Python version	3.11 or higher (tested on 3.11, 3.12, 3.13, 3.14 via <code>tox</code>)
Dependencies	See <code>requirements.txt</code> (all packages pinned to exact versions)
Expected runtime	~15 seconds on a standard laptop
Memory	No special requirements; tested on a standard 16 GB laptop
Disk space	~200 MB total (dominated by the raw data file, ~15 MB, and output tables)

Instructions for Replicators

1. Install dependencies

```
python -m venv .venv
source .venv/bin/activate      # Windows: .venv\Scripts\activate
pip install -r requirements.txt
```

2. Run the analysis

```
python scripts/run_analysis.py --promote
```

This single command reproduces all tables, figures, and inline statistics. No path changes are required. Outputs are written to:

- Tables: [output/tables/generated/](#)
- Figures: [output/figures/](#)
- Inline statistics: [output/numbers_summary.xlsx](#)

3. Verify (optional)

```
pytest
```

Runs the full test suite, verifying that the pipeline output matches the committed baseline datasets in [data/processed/](#) and [data/intermediate/](#). All tests should pass. To test across all supported Python versions:

```
tox
```

Data Availability Statement

All data used in this analysis are publicly available and included in this package. Full details — including citations and license information — are in [DATA_AVAILABILITY.md](#).

Input datasets

Dataset	Filename	Included in package	Source
IEG ICRR-PPAR Project Lessons	data/raw/IEG_ICRR-PPAR_Lessons_2025-03-12.xlsx	Yes	IEG, World Bank Group
FCS Country List	data/reference/FCS/FCS.xlsx	Yes	World Bank Group
Innovation keyword dictionaries	data/reference/keywords/*.json	Yes	Constructed by authors

Reference data (author-constructed)

The JSON keyword dictionaries in [data/reference/keywords/](#) were constructed by the authors using a combination of LLM assistance and expert review (see [methodology/](#)). They are included in this repository and tracked with SHA256 checksums in [DATA_CHECKSUMS.sha256](#).

List of Exhibits

Figures

All figures are exported to [output/figures/](#).

Exhibit	Output file	Generated by	Reproducible
Figure 3.1	output/figures/Fig3_1.png	scripts/run_analysis.py	Yes
Figure 3.2 — Innovation Factors in IDA and IBRD Settings	Not generated	Manually constructed	No
Figure 3.3	output/figures/Fig3_3.svg	scripts/run_analysis.py	Yes
Figure 4.1	output/figures/Fig4_1.svg	scripts/run_analysis.py	Yes
Figure D.1	output/figures/FigD_1.svg	scripts/run_analysis.py	Yes

Tables

All pipeline-generated tables are exported to [output/tables/generated/](#) as Excel files. Output filenames match paper table numbers and titles (e.g., [Table_3.1._Evolution_of_Innovation_Intensity...xlsx](#) = Table 3.1 in the paper).

The following tables are **not** generated by the pipeline — they are qualitative or synthesis content and are not present in [output/tables/generated/](#):

Table	Reason
Table 3.4 — Taxonomy of Innovations in the World Bank Portfolio	Qualitative typology; not derived from data
Table 3.14 — Profile of Bank Group Projects with IFC Investment	Summary narrative; not pipeline output
Table 4.2 — Bank Group Enablers vs. Constraints on Innovation	Qualitative synthesis
Table 5.1 — Potential Additional Metrics for Capturing Information About Innovations	Forward-looking recommendations
Table E.6 — Citations of PPP Arrangements Across the Portfolio	LLM-generated illustrative content
Table E.11 — Citations of CDD Arrangements	LLM-generated illustrative content

Inline statistics

Output file	Contents
output/numbers_summary.xlsx	Inline statistics cited in the paper body, with paragraph references
output/numbers_manifest.xlsx	Full manifest of inline numbers with paragraph locations, for reviewer verification

Project Structure

```

innovation-patterns-in-wbg-portfolio/
|
|— data/
|   |— raw/                                # Input: IEG source file (not tracked; place
here before running)
|   |— intermediate/                       # Output: cleaned dataset (one row per
project, pre-keyword-matching)
|   |— processed/                           # Output: analysis-ready dataset (one row per
project, all tags)
|   |— reference/
|       |— keywords/                       # Innovation keyword dictionaries (JSON)
|       |— FCS/                           # FCS country list (Excel + JSON; PDF included
for provenance only)
|       |— CODEBOOK.md                     # Column definitions for
processed_dataset.xlsx
|
|— manuscript/                             # Working draft (.docx); used by
tests/table_extractor.py only
|
|— methodology/
|   |— prompts/                             # AI prompt records for non-automated pipeline
steps
|   |— human_review/                       # Human review protocols
|
|— notebooks/                             # Exploratory analyses (Steps 2 and 11; see
methodology/README.md)
|
|— output/
|   |— figures/                             # Output: figures (PNG/SVG)
|   |— tables/
|       |— generated/                       # Output: pipeline-generated tables (Excel)
|       |— auto_extracted/                 # Reference: tables extracted from manuscript
for regression tests
|   |— numbers_summary.xlsx                 # Output: inline statistics cited in the paper
|   |— numbers_manifest.xlsx               # Reference: manifest of inline numbers with
paragraph locations
|
|— scripts/
|   |— run_analysis.py                     # Main script – runs the full pipeline
|   |— checks/                             # Development verification scripts (not
required to reproduce results)
|
|— src/                                    # Python source modules called by
run_analysis.py
|   |— _vendor/keyword_matcher/            # Vendored keyword matching library (frozen at
analysis version)
|
|— tests/                                  # Pytest test suite verifying pipeline against
committed baselines
|
|— README.md                              # This file
|— DATA_AVAILABILITY.md                  # Data sources, download instructions,

```

```

citations, rights statement
├─ DATA_CHECKSUMS.sha256      # SHA256 checksums for all input files
├─ requirements.txt            # Python dependencies (pinned)
├─ pyproject.toml              # Pytest configuration
├─ tox.ini                     # Multi-Python version test runner
configuration
```

Analysis Pipeline

A 17-step methodology transforming raw evaluation text into structured innovation insights. Steps marked **Python** are fully automated by `scripts/run_analysis.py`; others use LLM assistance and/or expert human review (prompt records and review protocols in `methodology/`).

#	Description of the Step	Primary Tool(s) Used
1	Normalize the evaluation corpus: Load the IEG corpus of project evaluations (ICRRs and PPARs). Standardize GPs, regions, dates, and ratings (map outcomes to numeric values). Implement de-duplication rules (prefer PPAR over ICRR, or else keep the most recent ICRR). Fix text encoding artifacts in the Lessons Learned text, and drop projects with empty lessons. Recode FCS status to include any country on the World Bank Harmonized FCS country list at any point between FY15 and FY25. The result is a "one project, one record" table with the Lessons Learned text as the main unstructured field for mining. This approach yielded a clean, analysis-ready database of 7,574 projects from a total of 8,569 evaluations.	Python
2	Develop an innovation keyword dictionary: Ask two LLMs to generate lists of innovation-related terms, drawing on their knowledge of innovation practice and international development. Combine the two lists, remove duplicates, and review the resulting pool by the expert authors to remove irrelevant, ambiguous, or overly generic terms. Refine further using Python notebooks to inspect matched snippets in context and add morphological variants. The final innovation keyword dictionary of 165 terms is provided in Appendix B.	GenAI, expert human review, Python
3	Tag each project by innovation content: Scan each Lessons Learned text using the innovation keyword dictionary and count distinct hits per project (do not repeat the count of the same token within a document).	Python
4	Classify projects by innovation intensity and construct the Innovation Index: Use a threshold-based classification system: High innovation = 4 or more distinct keyword hits, Moderate innovation = 2–3 distinct keyword hits, Low innovation = 0 or 1 distinct keyword hit. Store both the raw hit count and the tier so sensitivity checks can vary thresholds ± 1 .	Python
5	Human validation: Draw a sample of High and Moderate projects (random, purposive, and targeted borderline cases); have two reviewers independently assess	Expert human review

#	Description of the Step	Primary Tool(s) Used
	innovation presence and whether the automated tier assignment is appropriate. Resolve disagreements through discussion.	
6	Build the "innovative subset": Classify projects as High + Moderate innovations to form the subset for taxonomy analyses and diffusion work. Use the full corpus for denominators or comparisons when needed (e.g., to assess technology premiums, PCM insights).	Python
7	Develop the innovation taxonomy: Use LLMs to generate tagging keywords for each of the four innovation categories (operational/institutional, technological, collaborative, and financial). Apply these keywords via Python to produce provisional category assignments across all projects, defaulting to operational/institutional where no category is matched and allowing multi-label assignments where supported. Apply a similar Python-assisted inspection procedure as in Step 2 to check for potential mismatches. Review a sample of assignments (random, purposive, and targeted borderline cases) against the Lessons Learned text; revise as needed.	GenAI + Python + expert human review
8	Conduct iterative human-model reasoning: Engage LLMs as interlocutors to explore pattern candidates (through dozens of prompts), propose alternative framings and cuts by time, GP, instrument, and FCV. Researchers steer prompts, filter noise, and anchor interpretations in Bank Group context and history.	GenAI + expert human input
9	Formulate hypotheses: Distill the co-interpreted ideas into testable hypotheses about innovation's distribution, correlates, and performance implications. Examples include "the technology premium is positive conditional on adaptive delivery"; "operational innovations dominate across decades"; and "IFC references are associated with higher innovation intensity". Record hypothesized mechanisms and observable proxies for further testing. Authors screen and retain only hypotheses that are internally coherent, observable in the dataset, and feasible to operationalize.	GenAI + expert human review
10	Operationalize each hypothesis: For each hypothesis, define measurable indicators and keyword expansions (e.g., pilot/scale lexicon, IFC/PCM lexicon, technology markers). These lexicons enable consistent text mining across hypotheses. The list of keyword expansions is available upon request.	GenAI
11	Apply feature tagging: Using the new lexicons, run targeted keyword matching scans to identify references to pilots, scale-up, technology, IFC, PCM, and so on. Attach boolean flags to each project.	Python
12	Analyze the portfolio: Produce a time series analysis (such as five-year cohorts); conduct regional/GP breakouts; compare FCS vs. non-FCS; and generate cross-tabulations by innovation tier to identify distributional patterns.	Python
13	Assess performance associations (technology or innovation premium): Calculate mean differences in ratings by innovation tier (the innovation premium) and by technology presence within tiers and by GP (the technology premium), across outcome, bank performance, quality at entry, and quality of supervision ratings.	Python

#	Description of the Step	Primary Tool(s) Used
14	Identify top innovation model candidates: Prompt LLMs to surface candidate innovation models prevalent across the portfolio (e.g., e-procurement, PPPs, CDD, digital government); authors review and refine the final list, retaining only models that are conceptually distinct, plausibly prevalent in the portfolio, and feasible to operationalize with keywords.	GenAI + expert human review
15	Generate model keyword dictionaries and count mentions: Use LLMs to generate keyword lists for each candidate model identified in Step 14. Apply Python binary keyword matching to flag each project for each model's presence, then count and compare model prevalence across the portfolio.	GenAI + expert human review + Python
16	Contextualize the narrative: After generating tables and figures, prompt LLMs to propose institutional interpretations (e.g., links to procurement reform, ESF, GP reorganization). Manually verify this against data and known milestones. Add sensitivity notes when inferences hinge on tagging breadth.	GenAI + expert editing
17	Synthesize and report results: Integrate quantitative outputs and curated exemplars into a coherent narrative. Ensure all figures carry sensitivity notes.	Expert synthesis

Keyword Matching Library

Keyword matching is handled by a vendored copy of the `keyword_matcher` library located at `src/_vendor/keyword_matcher/`. It is installed as an editable package via `pyproject.toml` and importable as `keyword_matcher`. The vendored copy is frozen at the version used for this analysis.

Python Implementation Reference

All Python-automated steps run through `scripts/run_analysis.py`, which calls modules in `src/`:

Steps	src/ module	Test
1	<code>clean_ieg_data.py</code>	<code>tests/test_cleaning.py</code>
3–4, 6	<code>create_kw_index.py</code>	<code>tests/test_innovation_keywords_replication.py</code>
7 (taxonomy tagging)	<code>create_kw_index.py</code>	<code>tests/test_taxonomy_replication.py</code>
11	<code>create_kw_index.py</code>	<code>tests/test_feature_tagging_replication.py</code>
12–13	<code>create_table_replication.py</code> , <code>create_figure_replication.py</code> , <code>create_numbers_replication.py</code>	<code>tests/test_tables_replication.py</code>
15	<code>create_kw_index.py</code> (binary flagging, shared with Step 11),	<code>tests/test_top_models_replication.py</code>

Steps	src/ module	Test
	<code>create_table_replication.py</code> (counting/tables)	

Documentation

- [Data Availability Statement](#) — dataset sources, download instructions, citations, rights statement
- [Data Codebook](#) — column definitions for `processed_dataset.xlsx`
- [Methodology](#) — AI prompt records and human review protocols for non-automated steps
- [Notebooks](#) — exploratory analyses supporting specific methodology steps:
 - `match_method_sensitivity.ipynb` — keyword matching method comparison (Step 2)
 - `ifc_keyword_sensitivity.ipynb` — IFC keyword sensitivity analysis (Step 11)

Contact

For questions about this analysis or to request access to keyword expansions and supplementary materials, please contact the research team through the World Bank Group.