

Memoria del Proyecto — Predicción de Rotación de Empleados

Fecha: 2025-08-29

1. Objetivo

Construir un flujo reproducible de **ETL** (ETL → ML → métricas → visualización) para estimar el **riesgo de rotación** de empleados y ofrecer un **dashboard** de negocio con KPIs y **drivers** explicativos.

2. Datos

- **Principal:** `WA_Fn-UseC_HR-Employee-Attrition.csv` (1.470 filas).
- **Encuesta complementaria:** `encuesta_clima.csv` (engagement, satisfacción, etc.).
- **Clave de unión:** `EmployeeNumber`.

> Tras ETL quedan **1470** filas. Casos positivos (attrition=Yes): **237** (**16.12%**).

3. Pipeline

1) **Calidad de datos** (Spark): nulos, duplicados y dominios válidos. 2) **ETL** (Spark): **join** por clave, **features** derivadas (e.g., `income_yearly`, `overtime_flag`, `tenure_ratio`) y export a **Parquet**. 3) **Preprocesado** (sklearn): `ColumnTransformer` (num ~ estándar, cat ~ One-Hot) → `transformer.joblib`. 4) **Entrenamiento**: Regresión logística (baseline & pipeline) y Random Forest. 5) **Persistencia**: métricas agregadas en JSON y predicciones en Postgres (`predictions`, `predictions_latest`). 6) **BI**: Dashboard en Power BI (páginas: `Resumen & KPIs`, `Drivers`, `Departamentos`, `Puestos`).

4. Métricas principales (validación hold-out)

LogReg (pipeline)- ROC-AUC: **0.817**
- PR-AUC: **0.563**
- F1* (ópt.): **0.529** a umbral **0.732**

Random Forest (pipeline)- ROC-AUC: **0.805**
- PR-AUC: **0.536**
- F1* (ópt.): **0.535** a umbral **0.22**

> En Postgres (`predictions_latest`) hay **1470** filas para LogReg y **1470** para RF.

5. Dashboard de negocio (Power BI)

- **Resumen & KPIs:** tasa global, nº empleados en riesgo (\geq umbral), **precision/recall/F1**, selector de **modelo**. - **Drivers:** tabla/visual de **top features** (desde `feature_effects.csv`), filtros por departamento/puesto. - **Departamentos:** barras por `Department` (n en riesgo, % en riesgo). - **Puestos:** barras por `JobRole` (n en riesgo, % en riesgo). - Buenas prácticas: colores consistentes, títulos claros y **tooltips** con definiciones (**F1**, **PR-AUC**, umbral*).

6. Reproducibilidad (mínima)

```
docker compose up -d
docker compose run --rm jupyter python /scripts/preprocess.py --input
/data/processed/employee_attrition.parquet --out /output/models/transformer.joblib
docker compose run --rm jupyter python /scripts/train_ml.py --input
/data/processed/employee_attrition.parquet --model logreg
```

`docker compose run --rm jupyter python /scripts/train_ml.py --input /data/processed/employee_attrition.parquet --model rf`
Opcional: persistir a Postgres y revisar ``predictions_latest``.

7. Estructura de entrega

- **PBIX:** ``bi/Employee_Attrition_Dashboard.pbix`` (tag `**v1.0-entrega**`).- **Notebooks (HTML):** `docs/01_EDA_Attrition.html`, `docs/02_Modelado_Baseline.html`, `docs/05_Dashboard_KPIs.html`.- **Modelos & métricas:** ``output/models/*.pkl``, ``output/metrics/model_compare.json``.- **Código:** ``scripts/``, ``notebooks/``, ``docker-compose.yml``.

8. Riesgos y mitigaciones

- **Desajuste de versión sklearn/joblib:** fijado `*pipeline*` y ``transformer.joblib``.- **Conectividad BI ↔ DB:** alternativa por CSV ``feature_effects.csv``.- **Clase minoritaria:** uso de PR ■ AUC y F1 al seleccionar umbral.

9. Próximos pasos

- Validación cruzada con `*grid search*`.- Calibración de probas (Platt/Isotónica) y `*threshold tuning*` por coste.- Monitoreo de deriva y `*retrain*` programado.- Segmentos específicos por antigüedad/área.

Repositorio: https://github.com/gitbeatriz123/Proyecto_Prediccion_Rotacion_Empleados_Beatriz.Velayos