Plan de Machine Learning

para el Hackathon

3 caminos prácticos para agregar ML al MVP (sin complicar).

Fecha: 10/02/2026

Objetivo: proponer 3 implementaciones de ML compatibles con lo que ya existe (backend + UI + datos) y con valor claro para el pitch.

- Mantener el MVP estable: ML como servicio opcional y trazable.
- Priorizar demostrabilidad: resultados visibles en la UI (Admin) y exportables (CSV).
- Permitir dividir el trabajo entre 5 integrantes de Data Science.

Repositorio: https://github.com/brodyandre/growth_equestre_hackathon_2026
Stack: Backend Node/Express · Postgres · UI Admin Streamlit · Docker Compose.

1) Lo que ya tenemos en el MVP

El proyecto ya funciona end-to-end: captura leads, calcula score y permite explorar socios/partners. La idea es acoplar ML sin romper el flujo actual.

UI Admin (Streamlit)	Backend (Node/Express)	Base de datos (Postgres)
Filtros · KPIs · Export CSV Acciones: score/handoff	Endpoints: /leads, /events, /partners, /handoff	Tablas: leads, events, partners, cnae_map

Salida esperada del ML: un nuevo endpoint (o ampliación del score existente) que devuelva predicciones/recomendaciones, y una sección visual en la UI para mostrarlo.

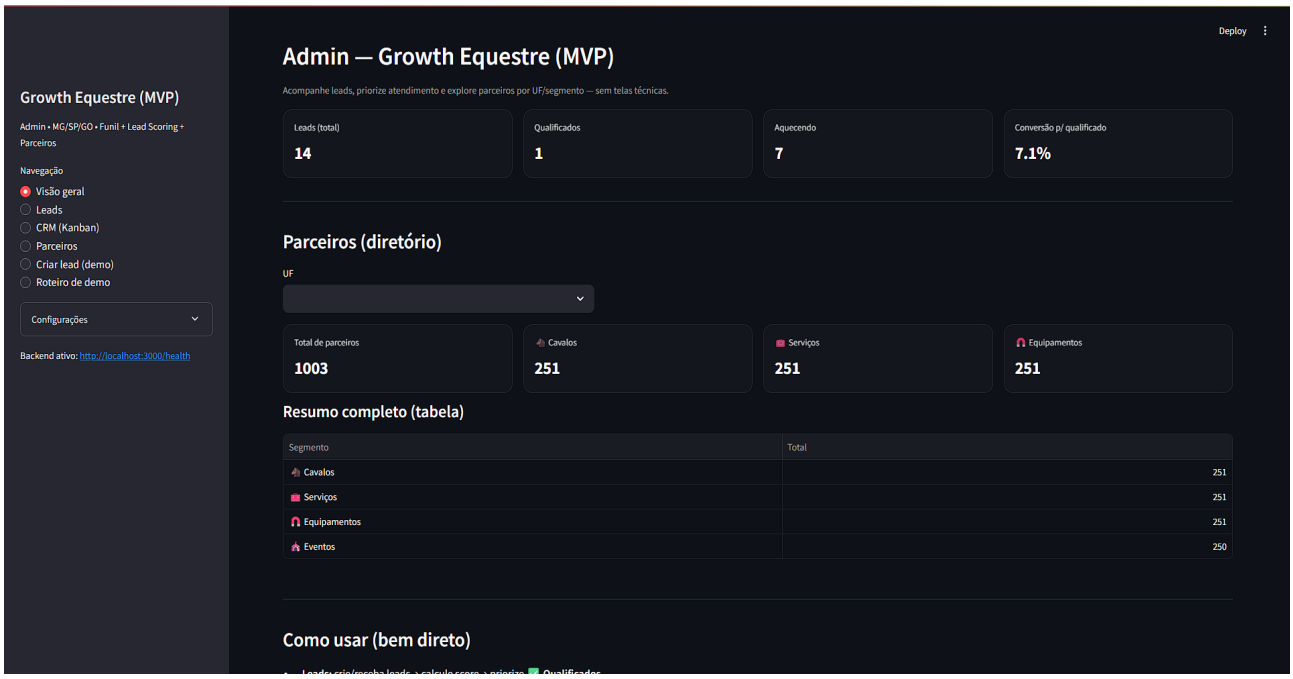

Figura: UI Admin (referencia).

2) Tres posibilidades de ML (vista rápida)

Las 3 propuestas usan datos que ya existen o que se pueden generar con poca instrumentación adicional.

Idea	Qué entrega	Datos mínimos	Dónde se integra
(A) Propensión de conversión de lead	Probabilidad de volverse “CALIFICADO” y sugerencia de acción	Leads + eventos del funnel + score actual	Backend: /leads/:id/score o /ml/lead-propensity UI: Leads / Visión general
(B) Recomendación de partners (matching)	Top-N partners por lead (ranking) + explicación simple	Lead (segmento/UF/ciudad) + partners (UF/segmento/CNAE)	Backend: /recommendations/partners?lead_id=... UI: Leads (detalle) / Partners
(C) Segmentación + anomalías	Clusters de leads/partners + alertas (outliers / duplicados)	Leads/partners con features numéricas/categóricas	Backend: /ml/segments y /ml/anomalies UI: KPIs, Kanban, listas

Recomendación: empezar por (A) o (B). Son fáciles de demostrar, generan números claros y se conectan bien con el flujo del producto.

3) (A) Propensión de conversión / calificación de lead

Objetivo: predecir, al momento de crear/actualizar un lead, la probabilidad de que se vuelva CALIFICADO y sugerir la próxima acción (priorizar vs nutrir).

Por qué encaja en el hackathon: convierte el scoring en “ML de verdad” y permite mostrar un KPI de mejora (ej.: mayor conversión).

Pasos (end-to-end)

1. Construir dataset: unir leads + events (conteos por tipo) + atributos (UF, segmento, presupuesto, plazo).
2. Definir etiqueta: CALIFICADO=1, resto=0 (o multi-clase por estado).
3. Entrenar baseline: Logistic Regression / RandomForest / XGBoost (si disponible).
4. Evaluar: AUC/ROC, precision@k (top leads), matriz de confusión.
5. Exportar modelo: joblib/pickle + versionado.
6. Servir predicción: endpoint nuevo o dentro del score existente.
7. UI: mostrar “Probabilidad” + “Sugerencia” + 2-3 razones (features importantes).

Archivos/pastas sugeridos para trabajar

- /scoring o /ml: notebooks/scripts de entrenamiento (pandas, scikit-learn).
- backend/routes: agregar endpoint (ej.: /ml/lead-propensity) o ampliar /leads/:id/score.
- ui_admin (Streamlit): sección en Leads → Detalle del lead.
- data/: dataset versionado (CSV/Parquet) + artefacto del modelo.

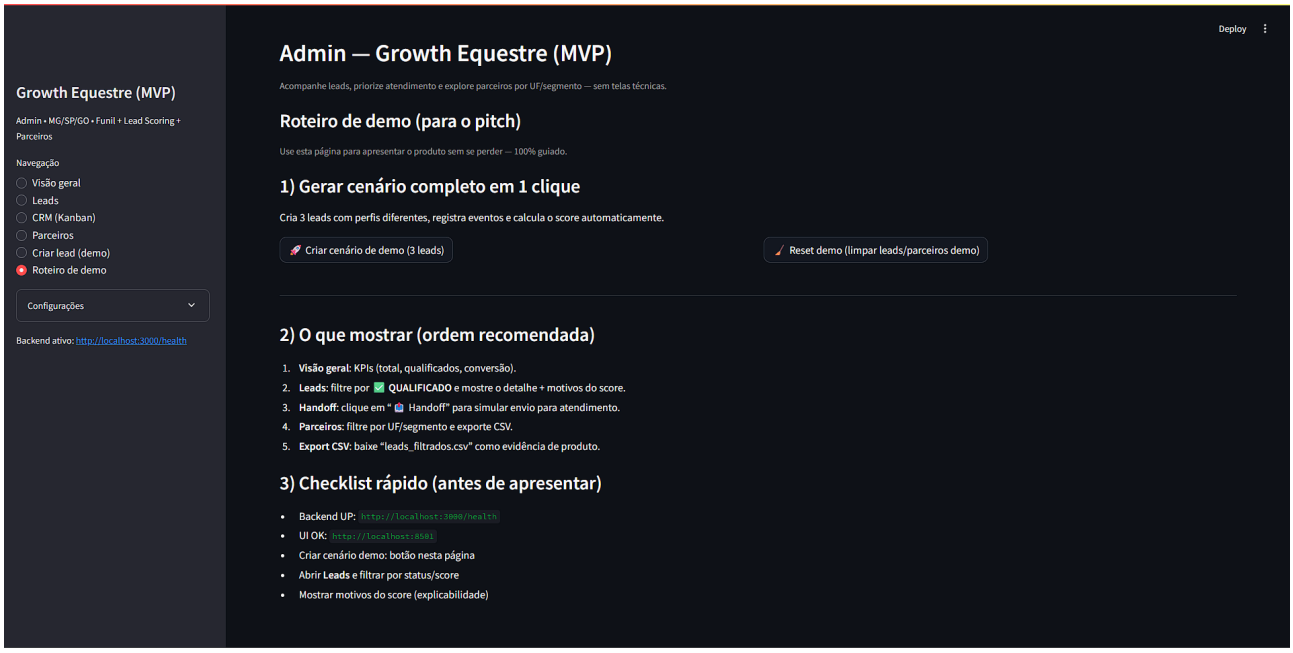

Figura: la UI ya contiene flujo de demo guiado; el modelo puede alimentar este guion.

4) (B) Recomendación de partners (matching lead ↔ partner)

Objetivo: dado un lead, devolver una lista de partners recomendados (Top-N) para facilitar el “handoff” y la prospección.

Cómo funciona (versión simple)

- Filtro duro: UF del lead (o UF vecina) + segmento compatible.
- Ranking: similitud de texto (TF-IDF) usando palabras-clave / CNAE / nombre (partner) + interés del lead.
- Explicación: mostrar 1-2 motivos (“mismo segmento”, “UF coincide”, “CNAE relacionado”).

Pasos (end-to-end)

1. Preparar base de partners: normalizar CNAE/segmento/UF y generar texto base por partner.
2. Entrenar o configurar: TF-IDF + cosine similarity (no necesita GPU).
3. Endpoint: /recommendations/partners?lead_id=... devuelve lista ordenada.
4. UI: en el detalle del lead, botón “Ver partners recomendados” + export CSV.
5. Validación rápida: comprobar si el Top-N tiene sentido para 10 leads de demo.

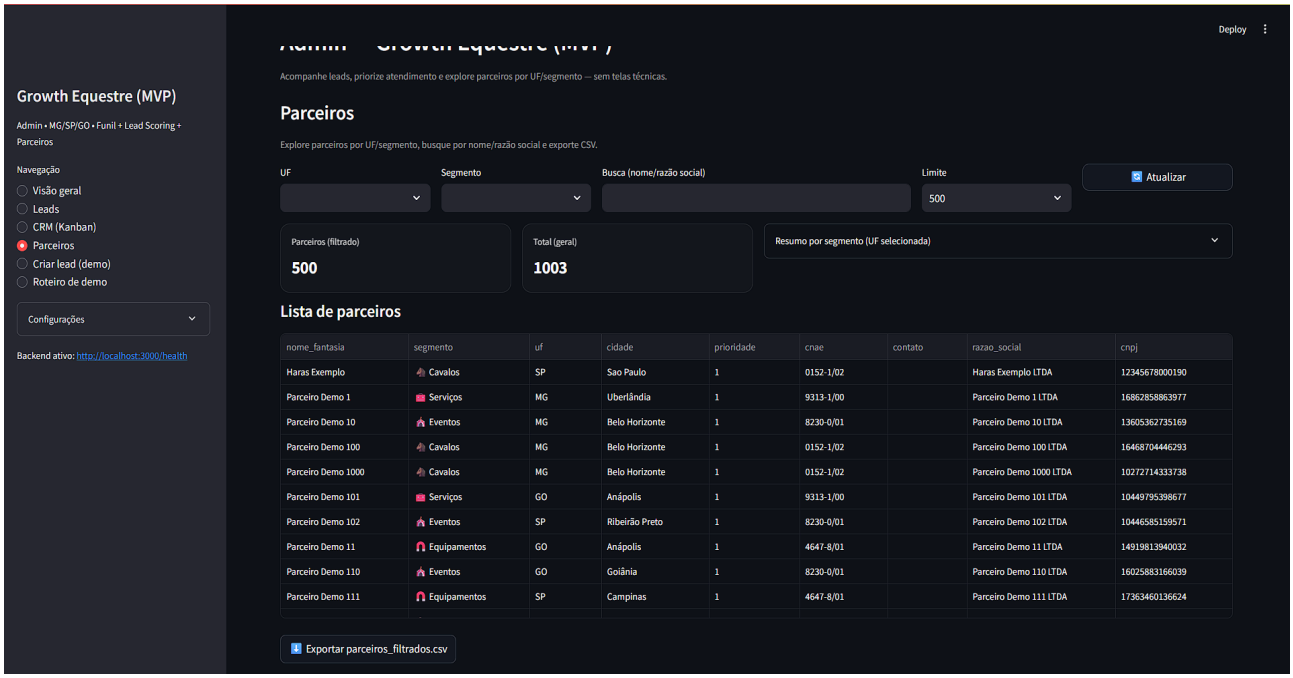

Figura: pantalla de Partners; es el lugar natural para mostrar recomendaciones/filtrado inteligente.

5) (C) Segmentación y detección de anomalías (para insights)

Objetivo: agrupar leads/partners en perfiles (clusters) y detectar casos fuera del patrón (outliers), mejorando priorización y calidad de datos.

Casos de uso típicos

- Segmentos de leads por intención: alto presupuesto + plazo corto vs curiosos.
- Partners “estrella” por densidad de demanda (UF/segmento) y CNAE.
- Alertas de datos: leads duplicados, valores incoherentes, partners con CNAE raro para el segmento.

Pasos (end-to-end)

1. Definir features: one-hot de segmento/UF + presupuesto/plazo (ordinal) + conteos de eventos.
2. Clustering: KMeans (3–6 clusters interpretables).
3. Anomalías: IsolationForest o LocalOutlierFactor para marcar top outliers.
4. Endpoint(s): /ml/segments y /ml/anomalies.
5. UI: mostrar “Cluster” en la tabla y una lista “Alertas” para revisión.

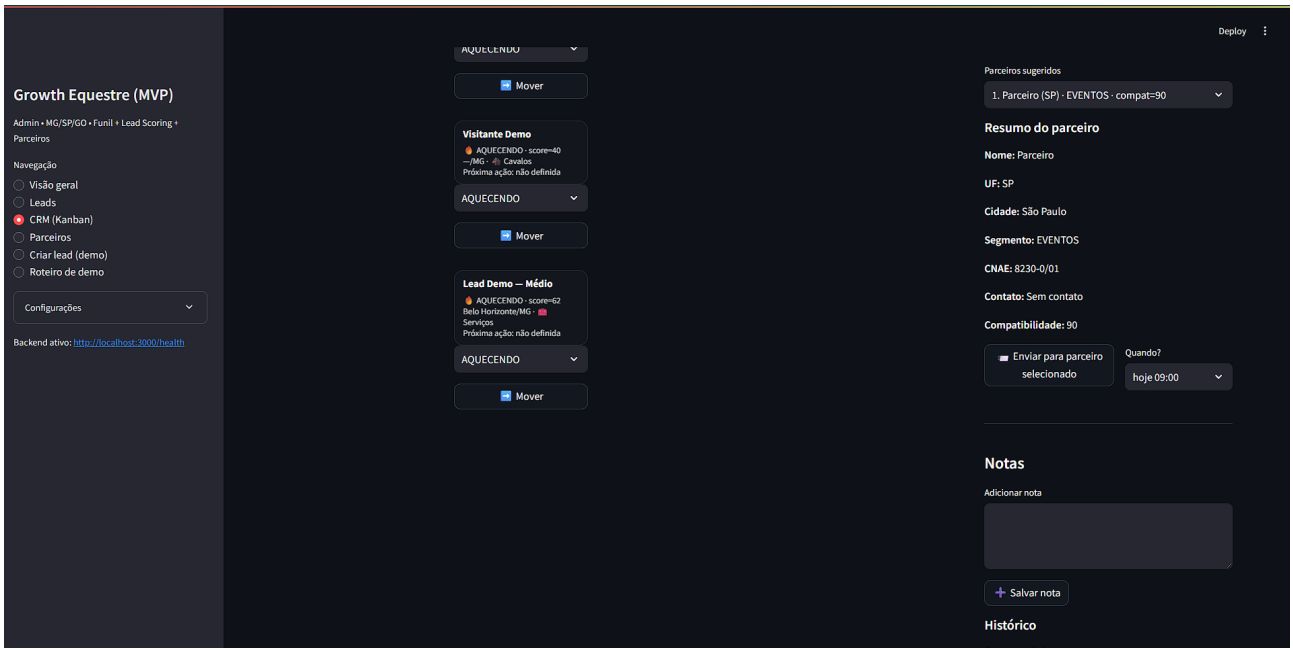

Figura: vista Kanban; la segmentación/anomalías puede alimentar badges y alertas.

6) Plan de trabajo para 5 integrantes de Data Science

Recomendación: elegir 1 opción como “principal” para el pitch y dejar otra como “extra” si hay tiempo.

Rol	Responsabilidad	Entregable
DS-1 (Data/ETL)	Dataset (leads+events+partners), limpieza, versionado	dataset_v1.csv + notebook ETL
DS-2 (Model A)	Propensión de lead (A): features, entrenamiento, métricas	model_lead.joblib + reporte
DS-3 (Model B)	Recomendación (B): TF-IDF, ranking, explicaciones	reco_partners.py + ejemplos
DS-4 (Integración)	Endpoints en backend + contratos JSON + tests básicos	routes /ml/* + docs
DS-5 (Producto/ Demo)	UX en UI + guion demo + KPIs + validación manual	pantallas actualizadas + script demo

Checklist final (antes de presentar)

- Backend activo y endpoints de ML respondiendo (health + /ml/*).
- Modelos/artefactos versionados y reproducibles (script de entrenamiento).
- UI mostrando predicciones/recomendaciones sin términos técnicos.
- Datos de demo listos (3 leads) y un ejemplo de “antes vs después”.