



# IA como produto: arquitetando soluções reutilizáveis, governadas e escaláveis

Fefe Alves - Principal Consultant - Data & AI



# Fefe Alves

## Principal Consultant Data & AI

>> +14 anos trabalhando no ecossistema de dados: analista de BI, analista de dados, cientista de dados, head de dados, estrategista de dados...

>> Varejo, Mídia, Streaming, Bancos, Telecom

>> Data Mesh, Governança Computacional, Modernização de plataformas de dados e Scalable AI

Linkedin:

[www.linkedin.com/in/fefealves](https://www.linkedin.com/in/fefealves)



Email:

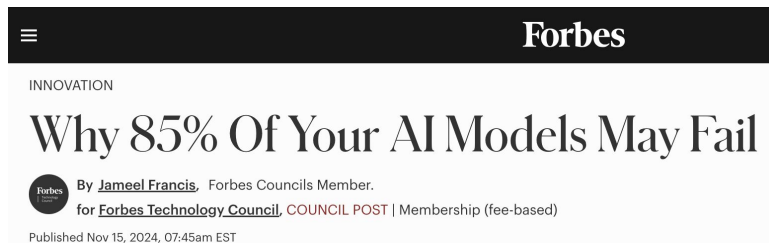
[fernanda.felix@thoughtworks.com](mailto:fernanda.felix@thoughtworks.com)

Repo da apresentação:

[https://github.com/ffalves/ia\\_conference\\_ai\\_products](https://github.com/ffalves/ia_conference_ai_products)

# 1. Por que projetos de IA falham nas empresas?

# Por que projetos de IA falham nas empresas?

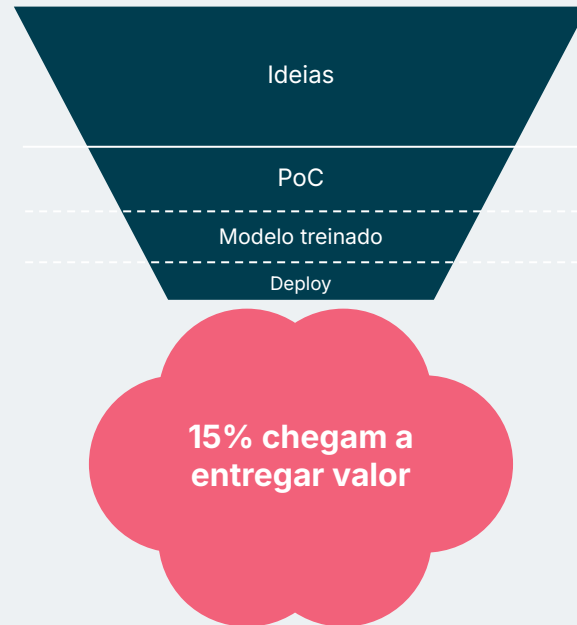


Gartner, 2021/2022

**VentureBeat**



VentureBeat, 2019



O desafio de trabalhar com IA em uma empresa não é apenas técnico mas de arquitetura, estratégia e cultura.

# Por que projetos de IA falham nas empresas?

Causas	Evidências
Iniciativas isoladas e descoladas do negócio	PoCs nascem em silos e não se integram à estratégia do negócio: > Não conseguem agregar valor real > Não se tornam viáveis
Falta de governança sobre dados e modelos	Dados indisponíveis, inconsistentes ou sem governança (rastreadabilidade/accountability) Modelos sem escalabilidade
Ausência de reuso e padronização	Cada projeto começa sempre do zero. Modelos sem reuso.
Infraestrutura e arquitetura mal preparada	Ambientes não suportam o ciclo de vida completo de um modelo e não são monitorados ou mantidos.
Desalinhamento entre times técnicos x negócios	Sem uma visão compartilhada sobre valor, riscos e retorno, um time busca acurácia e o outro impacto no negócio.

## 2. Produto de IA vs Projeto de IA

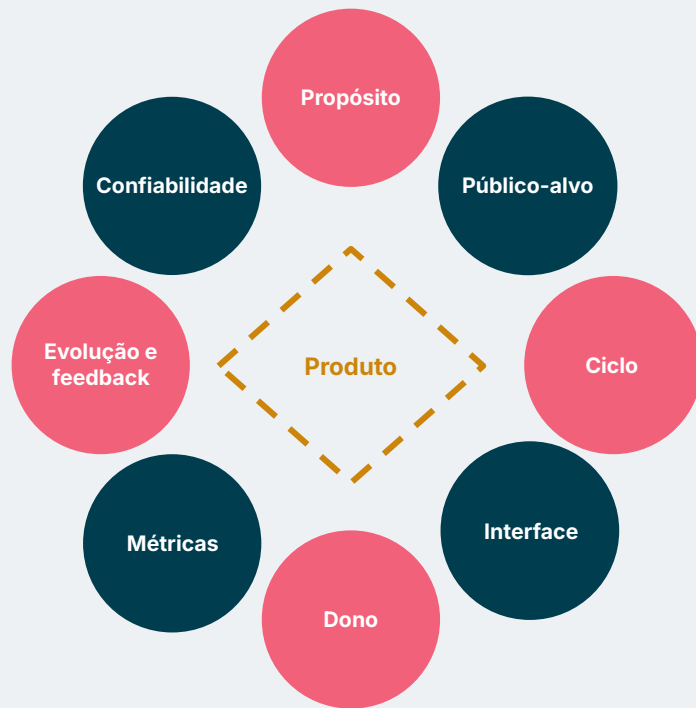


# O que é um Produto?

*"Produto é qualquer solução tangível ou intangível que resolve um problema ou atende a uma necessidade específica de um grupo de usuários ou clientes, entregando valor de forma contínua ao longo do tempo."*

*Inspirado em Melissa Perri, Marty Cagan e Roman Pichler*

Características comuns a um produto:



# O que é um Produto?

Ex: Cartão de crédito



Time de produtos financeiros - Cartão de Crédito



# O que é um Produto de Dados de IA (Data Mesh)?

*Data Mesh é uma abordagem sociotécnica baseada em quatro princípios:*

- 1) *Dados como produtos*
- 2) *Domínios são responsáveis pelos dados*
- 3) *Plataforma de autosserviço*
- 4) *Governança Federada Computacional*

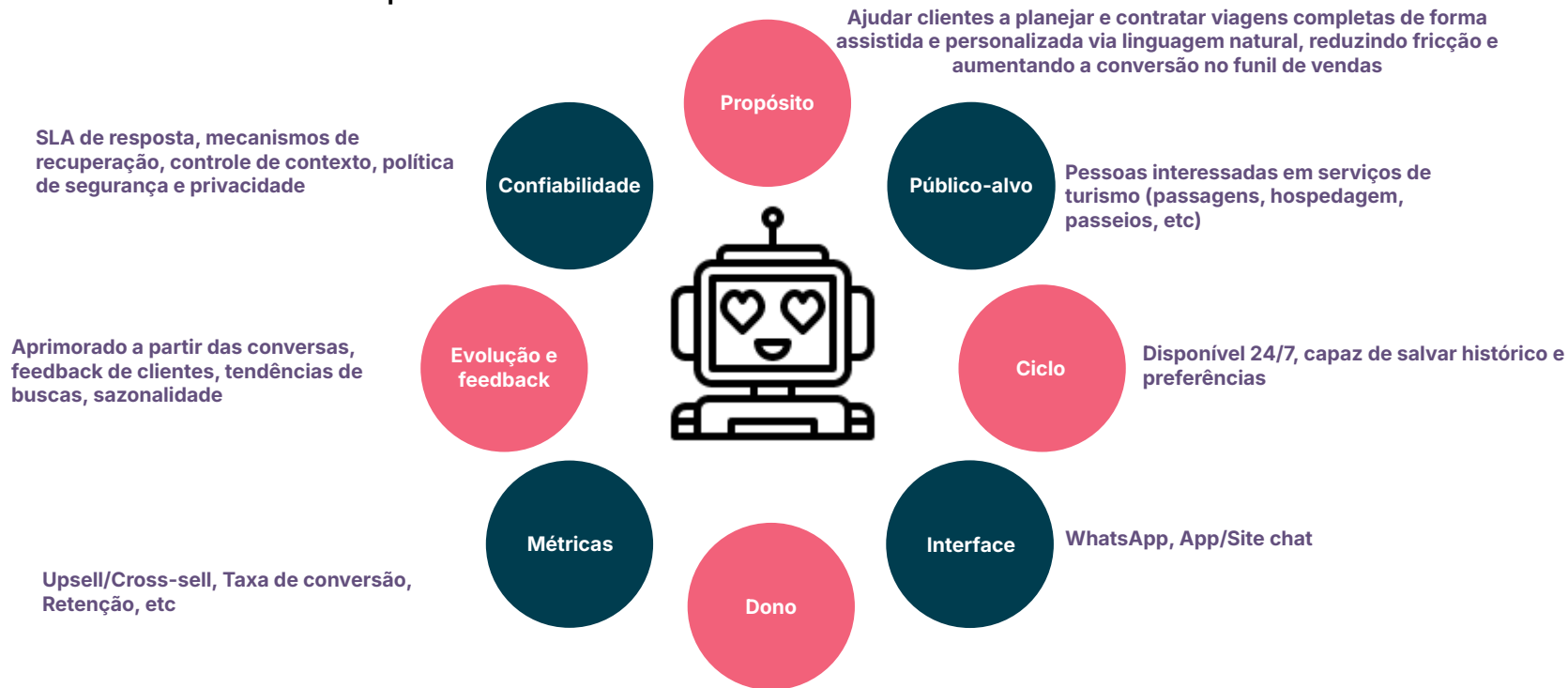


*Um produto de dados é um ativo com valor claro para consumidores, sob responsabilidade e propriedade de um domínio. O produto de dados é então empacotado com interfaces bem definidas, confiável, documentado, governado e acessível, tal como qualquer produto de software.*




# Produto de Dados de IA

Ex: Chatbot de uma empresa de turismo



# Produto de IA ≠ Projeto de IA

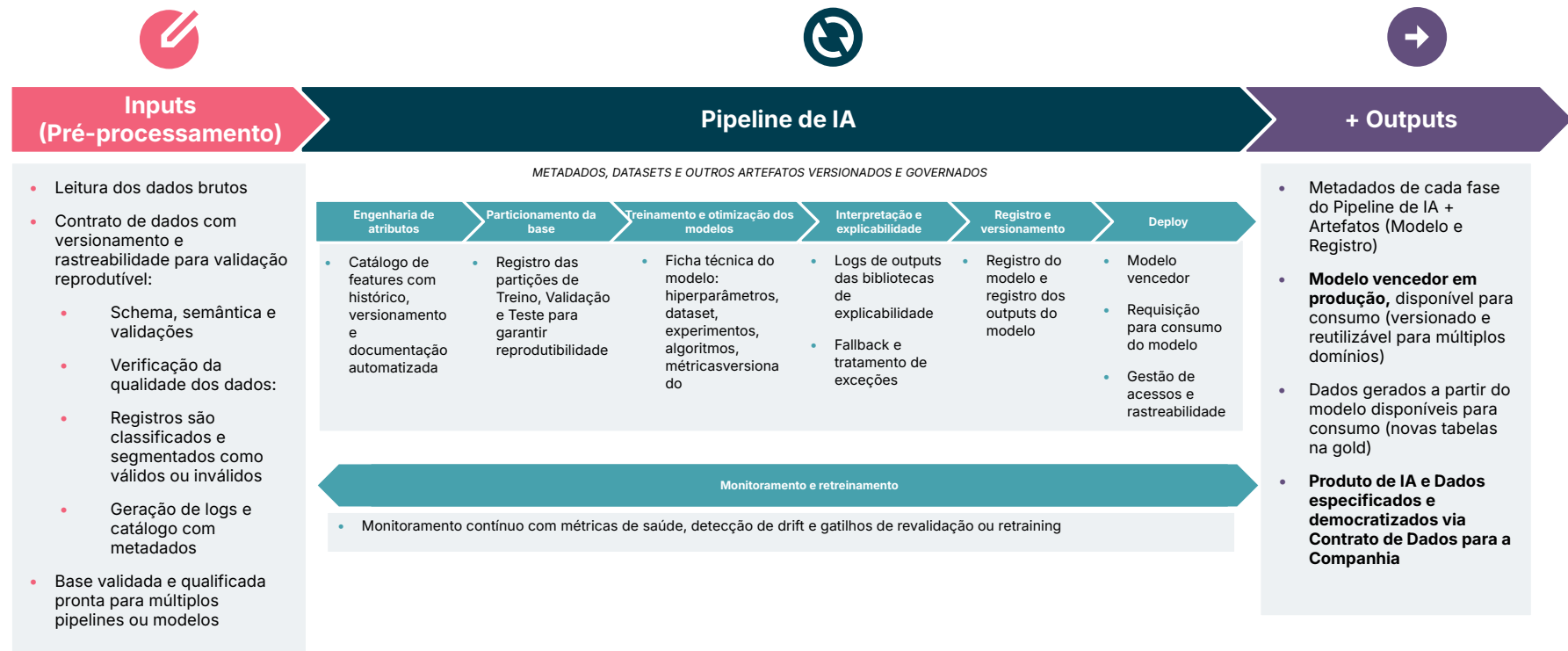
Mudança de mentalidade e arquitetura

	Projeto	Produto 
Objetivo	Treinar um modelo para obter uma classificação ou uma previsão	Entregar valor recorrente
Duração	Temporário/Pontual	Contínuo, com roadmap evolutivo
Formato	Notebook, script, relatório	API, documentação e versionamento
Integração	Isolado ou sob demanda	Integrado à plataforma para ser consumido por outros domínios
Métricas	Acurácia, entrega do modelo	Adoção, impacto no negócio, reuso
Governança	Ad hoc	Embutida: contrato, versionamento, rastreabilidade

# 3. Anatomia de um Produto de IA

# O que compõe um Produto de Dados de IA

Blocos fundamentais para um produto reutilizável e governado

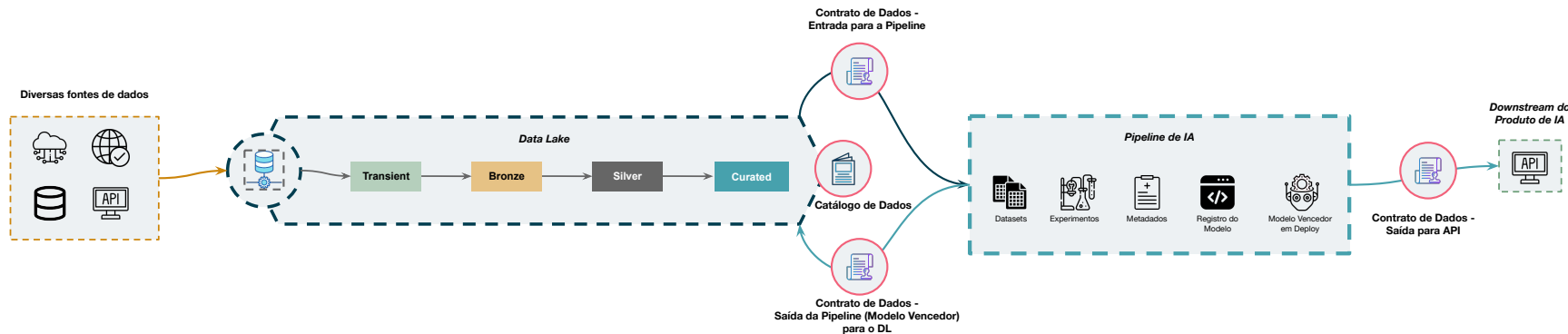


## 4. Materializando o Produto de IA (um pouco de código agora)

# Quais são os artefatos que materializam o

## Produto

Produto de IA que consiste em um Modelo de Propensão de Compra



# Artefatos fundamentais: Contrato de Dados

*Um contrato de dados é um acordo formal, geralmente automatizado, que especifica de forma clara e verificável as expectativas sobre a estrutura, o significado, a qualidade e a disponibilidade de um conjunto de dados, entre produtores e consumidores desses dados.*

Andrew Jones, Driving Data Quality with Data Contracts

Leituras:



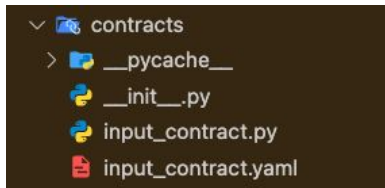
Data Contract Specification: <https://datacontract.com/>

## ARTEFATOS FUNDAMENTAIS



Contratos de Dados (input/output)

### Exemplo de Contrato de Dados



```
raw_layer > contracts > input_contract.yaml > {} models > {} users > {} fields > {} 1
1 dataContractSpecification: 1.0.0
2 id: input_contract
3 info:
4   title: Input Contract for Purchase Propensity Model
5   version: 1.0.1
6   description: Contrato de dados de entrada para modelo de propensão de compra
7   owner: CRM Team
8   status: active
9   contact:
10     name: Joao Maria (Data Product Owner)
11     email: joao.maria@crmtteam.com
12     created_at: 2023-08-01
13
14 terms:
15   usage:
16     - Este contrato define os dados de entrada necessários para o modelo de propensão de compra.
17     - Apenas registros que atendam a este contrato devem seguir para o modelo.
18     - Registros inválidos são segregados e logados.
19   limitations:
20     - Formato CSV; campos obrigatórios onde indicado.
21   policies:
22     - name: privacy_policy
23       url: https://company.com/privacy-policy
24
25 security:
26   classification: confidential
27   pii_handling:
28     mask_in_logs: ["name", "email", "cpf"]
29
30 observability:
31   acceptance_criteria:
32     batch_warn_if_invalid_pct_gt: 10
33     batch_fail_if_invalid_pct_gt: 30
34     required_fields_present:
35       email: 98 # % mínimo de preenchimento de email
36       per_field_max_null_pct: 5
37   metrics:
38     - name: invalid_records_pct
39     - name: null_ratio_per_field
40     - name: domain_violations_per_field
41
42 models:
43   users:
44     description: Um registro por usuário
```



```
dataContractSpecification: 1.0.0
id: input_contract
info:
  title: Input Contract for Purchase Propensity Model
  version: 1.0.0
  description: Contrato de dados de entrada para modelo de propensão de compra
  owner: CRM Team
  status: active
  contact:
    name: Joao Maria (Data Product Owner)
    email: joao maria@crmteam.com
    url: https://company.com/crm-team/contact
  created_at: 2025-08-01
```

A quem pertence o produto e a responsabilidade?

```
contracts
  __pycache__
  __init__.py
  input_contract.py
  input_contract.yaml
```

```
models:
  users:
    description: Um registro por usuário
    type: object
    primary_key: ["user_id"]
    unique_keys: ["user_id", "email"]

    fields:
      - name: user_id
        type: string
        required: true
        quality:
          - type: regex
            pattern: "^U\\d{3}$"
            description: U + 3 dígitos (U001..U999)
          - type: uniqueness
            level: dataset
      - name: name
        type: string
        required: true
        pii: true
      - name: email
        type: string
        required: true
        pii: true
        quality:
          - type: regex
            pattern: "^[^@\\s]+@[^@\\s]+\\.([^@\\s]+)$"
          - type: uniqueness
            level: dataset
```

Como devem estar os campos do produto?

```
terms:
  usage:
    - Este contrato define os dados de entrada necessários para o modelo de propensão de compra.
    - Apenas registros que atendam a este contrato devem seguir para o modelo.
    - Registros inválidos são segregados e logados.
  limitations:
    - Formato CSV; campos obrigatórios onde indicado.
  policies:
    - name: privacy_policy
      url: https://company.com/privacy-policy

  security:
    classification: confidential
    pii_handling:
      mask_in_logs: ["name", "email", "cpf"]
```

Quais são os termos e políticas?

```
observability:
  acceptance_criteria:
    batch_warn_if_invalid_pct_gt: 10
    batch_fail_if_invalid_pct_gt: 30
    required_fields_presence:
      email: 98 # % mínimo de preenchimento de email
    per_field_max_null_pct: 5
  metrics:
    - name: invalid_records_pct
    - name: null_ratio_per_field
    - name: domain_violations_per_field
```

Como vamos monitorar os registros?

Pesquise mais em: Data Contract Specification - <https://datacontract.com/>

## Validadores de schemas e checagem a partir do contrato

```

input_contract.py X
raw_layer > contracts > input_contract.py > ...

10 import pandas as pd
11 import pandera.pandas as pa
12 from pandera import Column, DataFrameSchema, Check
13 from pandera.errors import SchemaErrors
14
15 # -----
16 # Constantes / helpers
17 # -----
18 EMAIL_REGEX = r"^[a-zA-Z0-9]+@[a-zA-Z0-9]+\.[a-zA-Z]{2,}$"
19 CPF_REGEX = r"^\d{3}\.\d{3}\.\d{3}-\d{2}$"
20 USER_ID_REGEX = r"^[A-Z]{3}$"
21
22 def is_valid_cpf(cpf_str: str) -> bool:
23     if not isinstance(cpf_str, str):
24         return False
25     if not re.match(CPF_REGEX, cpf_str):
26         return False
27     nums = re.sub(r"\D", "", cpf_str)
28     if len(nums) != 11 or nums == nums[0] * 11:
29         return False
30
31     def calc_digito(n: str) -> str:
32         soma = sum((len(n) + 1 - i) * int(x) for i, x in enumerate(n))
33         resto = soma % 11
34         return "0" if resto < 2 else str(11 - resto)
35
36     d1 = calc_digito(nums[:9])
37     d2 = calc_digito(nums[:9] + d1)
38     return nums[-2:] == d1 + d2
39
40 def load_contract(path: str) -> Dict[str, Any]:
41     with open(path, "r") as f:
42         return yaml.safe_load(f)
43
44 def _collect_unique_keys(model: Dict[str, Any]) -> List[str]:
45     uniques: List[str] = []
46     if "unique_keys" in model and isinstance(model["unique_keys"], list):
47         uniques.extend(model["unique_keys"])
48     for fld in model.get("fields", []):
49         for q in (fld.get("quality") or []):
50             if str(q.get("type")).lower() == "uniqueness":
51                 uniques.append(fld["name"])
52     return list(dict.fromkeys(uniques))

```

```

# -----
# Schema a partir do contrato
# -----
107
108 def build_schema(contract: Dict[str, Any]) -> DataFrameSchema:
109     try:
110         model = contract["models"]["users"]
111         fields = model["fields"]
112     except KeyError:
113         model = {"fields": contract.get("fields", [])}
114         fields = model["fields"]
115
116     unique_keys = _collect_unique_keys(model)
117
118     type_map = {
119         "string": pa.String,
120         "text": pa.String,
121         "integer": pa.Int,
122         "float": pa.Float,
123         "boolean": pa.Bool,
124         "bool": pa.Bool,
125     }
126

```

```

contracts
├── __pycache__
├── __init__.py
└── input_contract.py
    └── input_contract.yaml

```

## Métricas a serem observadas

```

150 # Observabilidade / relatório
151 # -----
152
153 def _observability_report(
154     df: pd.DataFrame,
155     df_valid: pd.DataFrame,
156     df_invalid: pd.DataFrame,
157     contract: Dict[str, Any],
158     failure_cases_masked: pd.DataFrame,
159 ) -> Dict[str, Any]:
160     total = len(df)
161     invalid_pct = 0.0 if total == 0 else round(len(df_invalid) * 100.0 / total, 2)
162
163     obs = (contract.get("observability") or {})
164     acc = (obs.get("acceptance_criteria") or {})
165     warn_thr = acc.get("batch_warn_if_invalid_pct_gt")
166     fail_thr = acc.get("batch_fail_if_invalid_pct_gt")
167     per_field_max_null_pct = acc.get("per_field_max_null_pct")
168     required_fields_presence = (acc.get("required_fields_presence") or {})
169
170     status = "accepted"
171     if fail_thr is not None and invalid_pct > float(fail_thr):
172         status = "rejected"
173     elif warn_thr is not None and invalid_pct > float(warn_thr):
174         status = "accepted_with_warning"

```

## Metadados gerados a partir da aplicação do Contrato

```

# -----
# Metadados
# -----
def _write_metadata(
    status: str,
    count: int,
    source_file: Path,
    contract_version: str,
    out_root: Path,
    sample_output_file: Optional[Path] = None,
) -> Path:
    """
    Escreve um JSON de metadados para a partição 'valid' ou 'invalid'.
    """
    assert status in {"valid", "invalid"}
    ts = datetime.now().strftime("%Y%m%d_%H%M%S")
    base_dir = out_root / "catalog" / ("metadata_{}".format(status))
    base_dir.mkdir(parents=True, exist_ok=True)

    payload = {
        "status": status,
        "row_count": int(count),
        "source_file": str(source_file),
        "contract_version": contract_version,
        "timestamp": datetime.now().isoformat(),
        "sample_output_file": str(sample_output_file) if sample_output_file else None,
        "producer": "input_contract_enforcer",
    }

```

# Artefatos fundamentais: Metadados

## Rastreabilidade, qualidade e compliance

A geração intencional de metadados nos blocos de Pré-processamento dos dados para o Modelo e Pipeline de IA facilitam:

- identificação das bases utilizadas, seu nível de qualidade e versão dos artefatos utilizados
- identificação das informações do modelo, métricas, hiperparâmetros e URI dos artefatos, ambiente e políticas

Exemplos de metadados gerados durante o pré-processamento dos dados

```
metadata_valid_20250810_225404.json
raw_layer > data > catalog > metadata_valid > metadata_valid_20250810_225404.json > ...
1 {
2   "timestamp": "2025-08-11T14:32:00Z",
3   "source_file": "raw_data.csv",
4   "contract_version": "0.0.1",
5   "rows_total": 1000,
6   "rows_valid": 950,
7   "rows_invalid": 50,
8   "status": "accepted_with_warning",
9   "valid_file": "processed_raw_data/valid/valid_data_20250811.csv",
10  "invalid_file": "logs/invalid_records_20250811.csv"
11 }
```

Exemplos de metadados gerados durante a execução do Pipeline de IA

```
1 {
2   "schema_version": "1.0.0",
3   "created_at": "2025-08-11T21:10:00Z",
4   "model": {
5     "name": "purchase_propensity_winner",
6     "version": "2025.08.11-icmvp-v1",
7     "winner_tag": true,
8     "task": "binary_classification",
9     "framework": "xgboost",
10    "framework_version": "2.0.0",
11    "objective": "binary:logistic",
12    "random_seed": 42
13  },
14  "training": {
15    "start_time": "2025-08-11T20:05:00Z",
16    "end_time": "2025-08-11T20:35:00Z",
17  },
18  "hyperparameters": {
19    "n_estimators": 400,
20    "max_depth": 5,
21    "learning_rate": 0.05,
22    "subsample": 0.9,
23    "colsample_bytree": 0.8,
24    "min_child_weight": 1.0,
25    "gamma": 0.0,
26    "reg_alpha": 0.0,
27    "reg_lambda": 1.0,
28    "eval_metric": "auc"
29  },
30  "cross_validation": {
31    "n_folds": 5,
32    "stratified": true,
33    "metrics_mean": {
34      "auc": 0.812,
35      "average_precision": 0.702,
36      "log_loss": 0.528
37    },
38    "metrics_std": {
39      "auc": 0.014,
40      "average_precision": 0.021,
41      "log_loss": 0.017
42    }
43  },
44  "evaluation": {
45    "dataset": "holdout",
46    "metrics": {
47      "auc": 0.817,
48      "average_precision": 0.708,
49    }
50  }
51 }
```

### Governança

- **Rastreabilidade:**  
cada carga gera evidências verificáveis
- **Accountability:**  
Demonstra que as regras foram aplicadas e facilita auditorias

### Escala

- **Reuso rápido:**  
Facilita a geração de novos experimentos e novos modelos
- **Monitoramento:**  
Integração com ferramentas de dashboard/alertas

# Artefatos fundamentais: Catálogo de dados

Exemplos de informações que um Catálogo pode comportar

```
# catalog/ml/propensao_compra.yaml
catalogSpec: 1.0
domain: crm
product: propensao_compra
owner:
  name: CRM Team
  email: joao maria@crmteam.com

assets:
  - id: ds.feature_store_propensao_v1
    name: Feature Store Propensão de Compra (v1)
    type: table
    layer: gold
    physical:
      format: parquet
      path: s3://datalake/gold/crm/feature_store/propensao_compra/v1/
      partitioning: [dt]
    governance:
      classification: internal
      data_contract: raw_layer/contracts/input_contract.yaml
      retention: P660D
      pii_fields: [email, cpf] # todos tokenizados/hash na publicação
      access_policies:
        readers: [team.propensao, team.analytics]
        writers: [team.propensao]
    schema:
      primary_key: [user_id]
      fields:
        - {name: user_id, type: string, description: "Identificador do usuário"}
        - {name: idade, type: int, description: "Idade em anos"}
        - {name: renda_mensal, type: double, description: "Renda estimada BRL"}
        - {name: score_engajamento, type: double, description: "0..1"}
        - {name: freq_ult_12m, type: int, description: "Compras no período"}
        - {name: recencia_dias, type: int, description: "Dias desde última compra"}
        - {name: canal_preferido_onehot_app, type: int, description: "Feature engenharia"}
        - {name: canal_preferido_onehot_sitio, type: int, description: "Feature engenharia"}
        - {name: canal_preferido_onehot_whatsapp, type: int, description: "Feature engenharia"}
        - {name: canal_preferido_onehot_email, type: int, description: "Feature engenharia"}
        - {name: dt, type: date, description: "Partição de referência"}
    lineage:
      inputs:
        - id: raw.users.valid
          layer: raw
```



Centralizar informações sobre ativos de dados (nome, localização, schema, owner, classificação).



Governança: políticas de acesso, classificação de sensibilidade, retenção.



Qualidade: expectativas, métricas e relatórios associados.



Linhagem: conexão entre datasets de origem, transformação e destino.



Metadados de modelo (no caso de IA) ou de tabelas derivadas.

## 5. Porque essas práticas fortalecem o Produto de Dados/IA?

# Porque essas práticas fortalecem o Produto?

Contrato + Enforcer + Metadados + Catálogo = Produto de Dados confiável, reutilizável e governável, pronto para escalar

## Reuso

**Contrato de dados versionado** → garante que **outros times e produtos possam consumir** o dataset com segurança.

**Metadados ricos** → facilitam **entender rapidamente contexto, limitações e métricas** do dataset.

**Padrões consistentes de schema e qualidade** → **eliminam retrabalho** e transformações redundantes.

**Possibilidade de consumo cruzado** → outros domínios e produtos de dados podem **reaproveitar datasets, features e até modelos prontos, acelerando novos casos** de uso.

## Interoperabilidade

Formato aberto e documentado (YAML, CSV, JSON, .PY) → **leitura em múltiplas linguagens e plataformas.**

**Catálogo central** → local único para **descoberta de dados**, inclusive atributos criados por feature engineering.

## Governança computacional

**Registro automático de runs e versões** → cada execução do pipeline deixa **trilha completa para auditoria.**

**Linhagem explícita** → mostra a **origem de cada campo e como ele foi transformado.**

**Políticas declarativas** → regras de acesso, retenção e classificação aplicáveis de forma automatizada.

**Observabilidade integrada** → indicadores de qualidade e alertas no próprio fluxo do dado.

## 6. Impacto no negócio

# Quais as consequências para o negócio?

O que pode acontecer quando se muda a estratégia de Projeto para Produto de IA

## Eficiência Operacional

>> **Menos retrabalho** para times de dados e engenharia (menos tempo limpando dados e ajustando manualmente)

>> **Integração mais rápida entre domínios:** acelerando iniciativas entre áreas/departamentos.

>> **Time-to-market reduzido** para novos casos de uso a partir do reaproveitamento de features e modelos

## Escalabilidade

>> **Salto de PoC para produção sem refazer todo o ciclo:** modelos chegam à produção mais rápido pois pipelines padronizados e governados permitem esse salto.

>> **Diminuição de gargalos e respostas mais rápidas às mudanças no mercado** pois o processo de treino, atualização e versionamento foi facilitado.

>> **PoC/Modelo para Fábrica de Modelos**

## Ganho econômico

>> **Dados confiáveis geram decisões confiáveis**

>> **Aproveitamento máximo do investimento em IA:** modelos robustos têm maior taxa de sucesso e maior vida útil.

>> **Escalabilidade com menor custo incremental:** novas integrações e casos de uso são adicionados sem precisar recriar estruturas do zero.



# Quer saber mais sobre a cultura na Thoughtworks e nossas vagas abertas?



Siga as redes sociais da  
Thoughtworks



Explore as vagas  
da Thoughtworks

De IA como modelo → IA como produto → IA como plataforma

Pense como produto,  
modele como arquitetura  
e entregue com governança.

Muito obrigada!

**Fefe Alves**

Principal Consultant - Data & AI  
[fernanda.felix@thoughtworks.com](mailto:fernanda.felix@thoughtworks.com)

