Guía Práctica del Entregable 1: Análisis y Visualización

Proyecto: El Robo del Siglo Digital

Basada en el MVP Implementado - Versión para Estudiantes de Mentoría

---

Introducción

Esta guía está diseñada específicamente para estudiantes de la mentoría FAMAF 2025 y se basa en el código MVP (Minimum Viable Product) ya implementado. El objetivo es proporcionar una hoja de ruta clara y práctica para completar el Entregable 1, enfocándose únicamente en las funcionalidades que están desarrolladas y probadas en el notebook MVP.

El proyecto "El Robo del Siglo Digital" combina web scraping, análisis exploratorio y machine learning para estudiar el ecosistema web argentino y detectar sitios fraudulentos. En esta primera fase, nos concentraremos en el análisis y visualización de datos.

Objetivos del Entregable 1

Según el pipeline definido en el documento oficial, el Entregable 1 debe cumplir con los siguientes objetivos:

1. **Ingesta de Datos Inicial**: Trabajar con datasets de Argendir y Tranco-list.eu
2. **Preprocesamiento Estructural**: Normalizar y estandarizar los datos
3. **Análisis Exploratorio de Datos (EDA)**: Comprender las características principales
4. **Scraping Complementario Inicial**: Enriquecer registros con métricas externas
5. **Visualización**: Comunicar hallazgos visuales y patrones
6. **Data Insights Preliminares**: Identificar patrones sospechosos
7. **Documentación y Output**: Notebook documentado y dataset limpio

Estructura del Notebook Basada en el MVP

Sección 1: Preparación del Entorno

Esta sección corresponde al **Paso 1** del MVP implementado.

```
# Instalación de librerías necesarias
!pip install requests pandas fake_useragent tldextract python-whois Faker
```

Librerías principales utilizadas en el MVP:

- `pandas` : Manipulación y análisis de datos
- `requests` : Peticiones HTTP para web scraping
- `tldextract` : Extracción de información de dominios
- `whois` : Obtención de datos WHOIS
- `fake_useragent` : Simulación de navegadores reales
- `Faker` : Generación de datos sintéticos
- `tqdm` : Barras de progreso para procesos largos

Sección 2: Funciones Auxiliares y Configuración

Esta sección corresponde al **Paso 2** del MVP.

```
import pandas as pd
import requests
import tldextract
import whois
import random
import re
from fake_useragent import UserAgent
from faker import Faker
from datetime import datetime
from tqdm.notebook import tqdm

# Instancias globales
fake = Faker()
ua = UserAgent()
```

Puntos clave para estudiantes:

- Importar todas las librerías al inicio del notebook

- Crear instancias globales para evitar reinicializaciones
- Usar `tqdm` para mostrar progreso en operaciones largas

Sección 3: Extracción de Datos de Sitios Web Reales

Esta es la función principal implementada en el **Paso 3** del MVP.

3.1 Función Principal: `obtener_datos_sitio()`

La función `obtener_datos_sitio()` es el núcleo del análisis y extrae las siguientes características:

Información básica del dominio:
- `domain` : Nombre del dominio
- `tld` : Dominio de nivel superior (.com.ar, .gob.ar, etc.)
- `url_length` : Longitud total de la URL
- `is_https` : Indica si usa HTTPS

Características de la URL:
- `num_hyphens` : Número de guiones en la URL
- `num_digits` : Cantidad de números en la URL
- `num_special_chars` : Caracteres especiales (@, #, ?, %, etc.)
- `num_path_segments` : Número de segmentos en la ruta

Información de seguridad:
- `has_ssl` : Presencia de certificado SSL

Datos WHOIS:
- `whois_registrar` : Registrador del dominio
- `whois_country` : País del registrador
- `domain_age_days` : Antigüedad del dominio en días

Etiqueta objetivo:
- `is_phishing` : Etiqueta de clasificación (0 = legítimo, 1 = fraudulento)

3.2 Implementación Práctica

El MVP incluye dos opciones para obtener datos:

Opción 1: Lista fija de 25 sitios (recomendada para pruebas)

```
sitios_real = [
    "clarin.com.ar", "lanacion.com.ar", "infobae.com.ar",
    "afip.gob.ar", "anses.gob.ar", "bna.com.ar",
    "mercadolibre.com.ar", "santander.com.ar",
    # ... más sitios
]
```

Opción 2: Archivo CSV externo con 3200 sitios

Esta opción utiliza datos de https://tranco-list.eu/ filtrados por dominios .ar.

Sección 4: Procesamiento de Datos Reales

4.1 Extracción y Almacenamiento

```
# Crear lista para almacenar datos
datos_reales = []

# Procesar cada sitio
for sitio in tqdm(sitios_real, desc="Extrayendo datos"):
    try:
        datos_sitio = obtener_datos_sitio(sitio)
        datos_reales.append(datos_sitio)
        print(f"  Datos extraídos de: {sitio}")
    except Exception as e:
        print(f"   Error con {sitio}: {e}")

# Convertir a DataFrame
df_reales = pd.DataFrame(datos_reales)
```

4.2 Guardado de Resultados

```
# Guardar dataset enriquecido
df_reales.to_csv("dataset_sitios_reales.csv", index=False, encoding="utf-8")
```

Sección 5: Web Scraping de Argendir

Esta sección corresponde al **Paso 6** del MVP y implementa el scraping del directorio argentino.

5.1 Configuración del Scraping

```
import requests
from bs4 import BeautifulSoup
from urllib.parse import urljoin

# Categorías a scrapear (modificable según necesidades)
categorias_a_scrapear = ["Educacion", "Gobierno", "Salud", "Noticias"]

# URL base y headers
url_base = "https://www.argendir.com/"
headers = {
```

```
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36'
}
```

5.2 Proceso de Scraping

```
datos_totales = []

for categoria in categorias_a_scrapear:
    url_categoria = urljoin(url_base, f"{categoria}/")

    try:
        response = requests.get(url_categoria, headers=headers, timeout=10)
        response.raise_for_status()
        soup = BeautifulSoup(response.text, "html.parser")

        # Extraer sitios de la categoría
        sitios_encontrados = soup.select("a.link")

        for sitio in sitios_encontrados:
            nombre = sitio.text.strip()
            url_relativa = sitio["href"]
            url_absoluta = urljoin(url_base, url_relativa)

            # Buscar descripción
            descripcion_tag = sitio.find_next("p")
            descripcion = descripcion_tag.text.strip() if descripcion_tag else ""

            datos_totales.append([nombre, url_absoluta, categoria, descripcion])

    except requests.exceptions.RequestException as e:
        print(f"  Error al procesar '{categoria}': {e}")

# Crear DataFrame
df_argendir = pd.DataFrame(datos_totales, columns=["Nombre", "URL",
"Categoria", "Descripcion"])
```

Sección 6: Generación de Datos Sintéticos

Esta sección corresponde al **Paso 7** del MVP y es crucial para ampliar el dataset.

6.1 Función de Generación

```
def generar_datos_sinteticos(n=100):
  datos_sinteticos = []

  for _ in range(n):
    sitio = fake.domain_name()
    datos_sinteticos.append({
```

```
            "url": sitio,
            "longitud_url": len(sitio),
            "tiene_https": random.choice([True, False]),
            "cantidad_guiones": sitio.count("-"),
            "cantidad_digitos": sum(c.isdigit() for c in sitio),
            "ranking_alexa": random.randint(1, 500000),
            "longitud_titulo": random.randint(10, 70),
            "cantidad_palabras_clave": random.randint(0, 15),
            "tiempo_registro": random.randint(1, 20),
            "es_phishing": random.choice([True, False])
        })

    return pd.DataFrame(datos_sinteticos)

# Generar dataset sintético
df_sintetico = generar_datos_sinteticos(n=50000)
```

Sección 7: Análisis Exploratorio de Datos (EDA)

Nota importante: Esta sección debe ser desarrollada por los estudiantes basándose en los datos obtenidos. El MVP no incluye visualizaciones específicas, pero proporciona la base de datos necesaria.

7.1 Análisis Básico Recomendado

```
# Estadísticas descriptivas
print("Información general del dataset:")
print(df_reales.info())
print("\nEstadísticas descriptivas:")
print(df_reales.describe())

# Distribución de TLDs
print("\nDistribución de TLDs:")
print(df_reales['tld'].value_counts())

# Análisis de antigüedad de dominios
print("\nAntigüedad promedio de dominios:")
print(f"{df_reales['domain_age_days'].mean():.2f} días")
```

7.2 Visualizaciones Sugeridas

Basándose en el pipeline oficial, se recomienda crear:

1. **Distribución de sitios por categoría**
2. **Frecuencia de aparición por dominio y TLD**
3. **Nube de palabras de títulos y descripciones**
4. **Visualización de cobertura de categorías (mapa de calor o treemap)**

5. **Detección de outliers en nombres sospechosos**

Sección 8: Identificación de Patrones Sospechosos

8.1 Criterios de Detección

Basándose en las características extraídas, buscar:

- **Dominios recientes**: `domain_age_days` < 365
- **URLs sospechosas**: Alto número de caracteres especiales o guiones
- **TLDs inusuales**: Dominios que no usan .com.ar o .gob.ar
- **Falta de SSL**: `has_ssl` = 0
- **Patrones en nombres**: Uso de términos como "banco", "seguridad", "actualización"

8.2 Análisis de Texto Básico

```
# Buscar palabras clave sospechosas
palabras_sospechosas = ['banco', 'seguridad', 'actualización', 'verificación', 'urgente']

def analizar_texto_sospechoso(texto):
    if pd.isna(texto):
        return 0
    texto_lower = texto.lower()
    return sum(1 for palabra in palabras_sospechosas if palabra in texto_lower)

# Aplicar análisis
df_argendir['palabras_sospechosas'] = df_argendir['Nombre'].apply(analizar_texto_sospechoso)
```

Sección 9: Documentación y Resultados

9.1 Estructura del Notebook Final

El notebook debe incluir:

1. **Índice navegable** con enlaces a cada sección
2. **Introducción** explicando el contexto y objetivos
3. **Metodología** describiendo el proceso de extracción
4. **Resultados** con visualizaciones y análisis

5. **Conclusiones** con hallazgos principales
6. **Próximos pasos** para el Entregable 2

9.2 Datasets de Salida

Generar los siguientes archivos CSV:

- `dataset_sitios_reales.csv` : Datos extraídos de sitios reales
- `sitios_argentinos_multiples_categorias.csv` : Datos de Argendir
- `dataset_sintetico.csv` : Datos sintéticos generados
- `argendir_clean.csv` : Dataset maestro limpio y combinado

Sección 10: Buenas Prácticas y Recomendaciones

10.1 Estilo de Código

- Seguir PEP8 para el estilo de código Python
- Comentar cada función y sección importante
- Usar nombres de variables descriptivos
- Incluir docstrings en las funciones

10.2 Manejo de Errores

```
try:
    # Código que puede fallar
    datos_sitio = obtener_datos_sitio(sitio)
    datos_reales.append(datos_sitio)
except Exception as e:
    print(f"  Error con {sitio}: {e}")
    # Continuar con el siguiente sitio
    continue
```

10.3 Optimización de Rendimiento

- Usar `tqdm` para mostrar progreso en operaciones largas
- Implementar pausas entre requests para evitar bloqueos
- Guardar resultados intermedios frecuentemente

Conclusiones y Próximos Pasos

El MVP implementado proporciona una base sólida para el Entregable 1, enfocándose en:

1. **Extracción robusta de datos** de sitios web reales
2. **Web scraping estructurado** del directorio Argendir
3. **Generación de datos sintéticos** para ampliar el dataset
4. **Framework básico** para análisis exploratorio

Para el Entregable 2, los estudiantes deberán:
- Profundizar en el análisis de los datos recopilados
- Implementar técnicas avanzadas de limpieza de datos
- Desarrollar características adicionales (feature engineering)
- Preparar el dataset maestro para modelado

Elementos NO incluidos en el MVP (que deben evitarse en el entregable):
- Las secciones marcadas como "SECCION PRUEBAS"
- Código experimental con Scrapy
- Intentos fallidos de búsqueda en Google/DuckDuckGo
- Código de Common Crawl que genera errores

Esta guía proporciona un camino claro y probado para completar exitosamente el Entregable 1, basándose únicamente en código funcional y validado.