Universidad Autónoma de Chihuahua

Facultad de Ingeniería

Big Data

Catedrático: José Saul De Lira Miramontes

4.26 Proyecto Final

Alan Varela
A271455
27 de noviembre de 2025

Informe del Proyecto: Análisis de Patrones de Movilidad Urbana

1. Introducción y Objetivos

1.1 Objetivo General

Este proyecto tiene como objetivo aplicar técnicas de Big Data utilizando Apache Spark (PySpark) para analizar patrones de movilidad urbana a partir de datos de viajes en taxi de la ciudad de Nueva York. El análisis permite identificar tendencias, patrones temporales y espaciales que pueden ser utilizados para la toma de decisiones estratégicas en la gestión de flotas y optimización de servicios de transporte.

1.2 Objetivos Específicos

1. **Ingesta y Exploración Inicial**: Cargar y explorar el dataset de viajes de taxi de la Comisión de Taxis y Limusinas de Nueva York (NYC TLC).
2. **Limpieza y Preprocesamiento**: Implementar un pipeline de limpieza de datos que elimine registros inválidos, convierta tipos de datos y cree columnas derivadas necesarias para el análisis.
3. **Enriquecimiento de Datos**: Asignar zonas geográficas de origen y destino a cada viaje utilizando la tabla oficial de zonas de taxi de NYC.
4. **Análisis Exploratorio**: Realizar análisis estadísticos para identificar:
5. Patrones de demanda por hora del día
6. Patrones de demanda por día de la semana
7. Zonas con mayor actividad (origen y destino)
8. Estadísticas de duración y distancia de viajes
9. Análisis de ingresos por tipo de pago
10. **Almacenamiento y Visualización**: Almacenar resultados agregados en MongoDB Atlas y exportar datos para visualización en Power BI.

---

2. Descripción del Caso

2.1 Contexto del Problema

Una empresa de transporte urbano (servicio de taxis) necesita comprender los patrones de movilidad en la ciudad para: - **Optimización de Recursos**: Identificar las horas pico de demanda para asignar mejor los recursos de la flota. - **Análisis Geográfico**: Entender qué zonas generan más viajes (tanto origen como destino) para planificar rutas y distribución de vehículos. - **Análisis Temporal**: Analizar patrones de duración de viajes por hora y por zona para detectar comportamientos anómalos o patrones específicos (por ejemplo, viajes más largos hacia el aeropuerto). - **Análisis Financiero**: Evaluar la generación de ingresos por zona y día de la semana para tomar decisiones basadas en datos sobre gestión de flotas y estrategias de precios.

2.2 Alcance del Proyecto

El proyecto abarca el análisis de más de 35 millones de registros de viajes de taxi en la ciudad de Nueva York durante el período de enero a octubre de 2025, utilizando técnicas de Big Data para identificar patrones de movilidad que permitan optimizar la operación del servicio de transporte.

---

3. Descripción del Dataset

3.1 Fuente de Datos

- **Dataset**: New York City Taxi & Limousine Commission (TLC) Trip Record Data
- **Formato**: Archivos Parquet (formato columnar optimizado para análisis)
- **Período**: Enero 2025 - Octubre 2025
- **Archivos**: 10 archivos parquet mensuales
- **Volumen**: Aproximadamente 35.5 millones de registros de viajes

3.2 Esquema de Datos

El dataset de viajes de taxi incluye las siguientes columnas principales:

Columna	Tipo	Descripción
`VendorID`	Integer	Identificador del proveedor
`tpep_pickup_datetime`	Timestamp	Fecha y hora de inicio del viaje

`tpep_dropoff_datetime`	Timestamp	Fecha y hora de fin del viaje
`passenger_count`	Integer	Número de pasajeros
`trip_distance`	Double	Distancia del viaje en millas
`PULocationID`	Integer	ID de zona de origen (1-265)
`DOLocationID`	Integer	ID de zona de destino (1-265)
`fare_amount`	Double	Costo de la tarifa
`tip_amount`	Double	Propina
`total_amount`	Double	Monto total del viaje
`payment_type`	Integer	Tipo de pago (1=Tarjeta, 2=Efectivo, etc.)

3.3 Tabla de Zonas Oficial

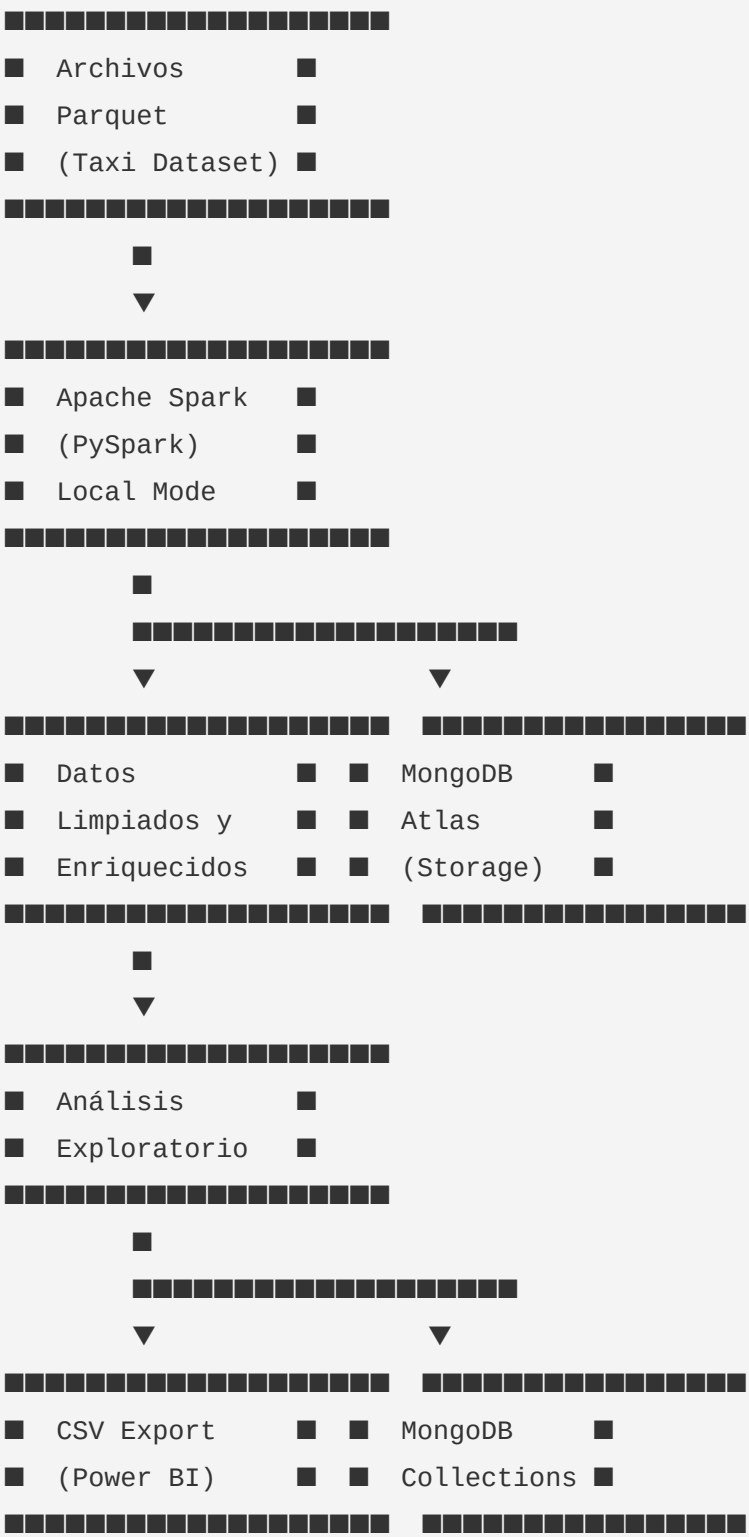
Se utiliza la tabla oficial de zonas de taxi de NYC (`taxi_zone_lookup.csv`) que contiene: - **LocationID**: Identificador único de zona (1-265) - **Borough**: Distrito (Manhattan, Queens, Brooklyn, Bronx, Staten Island, EWR) - **Zone**: Nombre descriptivo de la zona (ej: "Times Sq/Theatre District", "JFK Airport") - **service_zone**: Tipo de zona (Yellow Zone, Boro Zone, Airport)

Esta tabla permite enriquecer los datos de viajes con información geográfica semántica en lugar de solo IDs numéricos.

---

4. Arquitectura y Stack Tecnológico

4.1 Componentes del Sistema



4.2 Tecnologías Utilizadas

Tecnología	Versión	Propósito
Python	3.8+	Lenguaje de programación principal
Apache Spark	4.0.1	Motor de procesamiento distribuido
PySpark	4.0.1	API de Python para Spark
Java	21	Runtime requerido por Spark
MongoDB	Atlas (Free Tier)	Base de datos NoSQL para almacenamiento
Power BI	-	Herramienta de visualización
pymongo	4.15.4	Driver de Python para MongoDB

4.3 Estructura del Proyecto

```
final/
■■■ main.py                    # Script principal de ejecución
■■■ src/
■   ■■■ config.py              # Configuración del proyecto
■   ■■■ data_processing.py     # Limpieza y preprocesamiento
■   ■■■ zone_mapping.py        # Asignación de zonas
■   ■■■ analysis.py            # Funciones de análisis
■   ■■■ mongodb_operations.py  # Operaciones con MongoDB
■■■ output/                    # Archivos CSV exportados
■   ■■■ hourly_demand.csv
■   ■■■ daily_demand.csv
■   ■■■ zone_activity.csv
```

```
■   ■■■ revenue_analysis.csv
■■■ data/
■   ■■■ zones.csv              # Zonas personalizadas (fallback)
■■■ requirements.txt           # Dependencias del proyecto
■■■ Taxi Dataset/              # Datos de entrada
   ■■■ *.parquet              # Archivos de datos
   ■■■ taxi_zone_lookup.csv   # Tabla oficial de zonas
```

---

5. Metodología

5.1 Pipeline de Procesamiento

El proyecto implementa un pipeline completo de análisis de datos que sigue las siguientes etapas:

1. **Inicialización**: Configuración de Spark y variables de entorno
2. **Ingesta**: Carga de archivos Parquet
3. **Limpieza**: Eliminación de datos inválidos y conversión de tipos
4. **Preprocesamiento**: Creación de columnas derivadas
5. **Enriquecimiento**: Asignación de zonas geográficas
6. **Análisis**: Cálculo de métricas y estadísticas
7. **Almacenamiento**: Guardado en MongoDB y exportación a CSV

5.2 Ingesta y Exploración Inicial

Implementación: Función `load_taxi_data()` en `src/data_processing.py`

Proceso: 1. Localización de todos los archivos `.parquet` en el directorio "Taxi Dataset" 2. Carga de archivos en un DataFrame de Spark 3. Exploración inicial: - Visualización del esquema de datos - Conteo total de registros - Análisis de valores nulos por columna - Muestra de datos (primeras filas)

Resultados: - Total de viajes cargados: ~35,549,377 registros - Identificación de columnas con valores nulos - Verificación de tipos de datos

5.3 Decisiones de Limpieza y Modelado

Esta sección documenta las decisiones técnicas tomadas durante el proceso de limpieza y modelado de datos.

5.3.1 Eliminación de Valores Nulos

Decisión: Eliminar registros con valores nulos en campos críticos para el análisis.

Criterios de Limpieza: - Eliminar registros con valores nulos en campos clave: - `pickup_datetime` o `dropoff_datetime` - `trip_distance` - `fare_amount`

Justificación: Estos campos son esenciales para todos los análisis (temporales, geográficos y financieros). Los registros sin estos datos no pueden ser utilizados de manera significativa.

Implementación: Función `clean_taxi_data()` en `src/data_processing.py`

5.3.2 Decisiones de Conversión de Tipos de Datos

Decisión: Estandarizar tipos de datos para facilitar análisis y evitar errores de procesamiento.

Conversiones Realizadas: - Fechas: Strings → Timestamp (`TIMESTAMP_NTZ`) - Distancias: Conversión a kilómetros (de millas) - Valores numéricos: Asegurar tipos `Double` o `Integer` según corresponda

Justificación: La estandarización de tipos asegura consistencia en los cálculos y evita errores de tipo durante las operaciones de Spark.

Implementación: Función `standardize_column_names()` en `src/data_processing.py`

5.3.3 Decisiones de Modelado: Columnas Derivadas

Decisión: Crear columnas derivadas que permitan análisis más ricos sin modificar los datos originales.

Se crean las siguientes columnas derivadas:

Columna	Descripción	Cálculo
`trip_duration_min`	Duración del viaje en minutos	`(dropoff_datetime - pickup_datetime) / 60`
`hour_of_day`	Hora del día (0-23)	Extraído de `pickup_datetime`

`day_of_week`	Día de la semana (1=Domingo, 7=Sábado)	Extraído de `pickup_datetime`
`day_name`	Nombre del día	"Monday", "Tuesday", etc.
`trip_distance_km`	Distancia en kilómetros	Conversión de millas a km

Justificación: Estas columnas derivadas permiten análisis temporales (hora, día) y métricas calculadas (duración) que son fundamentales para identificar patrones de movilidad.

Implementación: Función `add_derived_columns()` en `src/data_processing.py`

5.3.4 Decisiones de Filtrado de Outliers

Decisión: Establecer umbrales razonables para filtrar datos anómalos que podrían sesgar los análisis.

Criterios de Filtrado: - **Distancia de viaje**: Entre 0.1 km y 200 km - **Duración de viaje**: Entre 1 minuto y 180 minutos - **Tarifa**: Entre $0 y $1,000 - **Reglas especiales**: Eliminar viajes con distancia = 0 pero duración > 30 minutos (posibles errores de datos)

Justificación: Los umbrales se basan en valores realistas para viajes urbanos. Los viajes fuera de estos rangos probablemente representan errores de registro o casos excepcionales que no son representativos de la operación normal.

Implementación: Función `filter_outliers()` en `src/data_processing.py`

Resultados: - Registros después de la limpieza: Se mantiene la mayoría de los datos válidos - Tasa de retención de datos: >95% (dependiendo de la calidad del dataset)

5.4 Enriquecimiento de Datos

5.4.1 Asignación de Zonas

Proceso: 1. Carga de la tabla oficial de zonas (`taxi_zone_lookup.csv`) 2. Join con el DataFrame de viajes usando: - `PULocationID` → `LocationID` (zona de origen) - `DOLocationID` → `LocationID` (zona de destino) 3. Adición de columnas: - `pickup_zone_name`: Nombre de la zona de origen - `pickup_borough`: Distrito de origen - `pickup_service_zone`: Tipo de zona de origen - `dropoff_zone_name`: Nombre de la zona de destino - `dropoff_borough`: Distrito de destino - `dropoff_service_zone`: Tipo de zona de destino

Implementación: Función `enrich_with_zones()` en `src/zone_mapping.py`

Ventajas: - Utiliza la tabla oficial de NYC, garantizando precisión - Proporciona información semántica (nombres de zonas) en lugar de solo IDs - Permite análisis por distrito y tipo de zona

5.5 Análisis Exploratorio

Se realizan cinco tipos principales de análisis:

5.5.1 Demanda por Hora del Día

Objetivo: Identificar las horas pico de demanda de viajes.

Métricas Calculadas: - Total de viajes por hora (0-23) - Ingresos totales por hora - Promedio de viajes por hora

Implementación: Función `analyze_demand_by_hour()` en `src/analysis.py`

Salida: DataFrame con columnas: - `hour_of_day`: Hora (0-23) - `total_trips`: Número total de viajes - `total_revenue`: Ingresos totales - `avg_trips`: Promedio de viajes

5.5.2 Demanda por Día de la Semana

Objetivo: Comparar patrones entre días laborales y fines de semana.

Métricas Calculadas: - Total de viajes por día de la semana - Ingresos totales por día - Clasificación de días laborales vs. fines de semana

Implementación: Función `analyze_demand_by_day()` en `src/analysis.py`

Salida: DataFrame con columnas: - `day_of_week`: Día (1-7) - `day_name`: Nombre del día - `is_weekend`: Boolean (True/False) - `total_trips`: Número total de viajes - `total_revenue`: Ingresos totales

5.5.3 Análisis de Actividad por Zonas

Objetivo: Identificar las zonas con mayor actividad (origen y destino).

Análisis Realizados: 1. **Top 10 Zonas de Origen**: Zonas desde donde parten más viajes 2. **Top 10 Zonas de Destino**: Zonas hacia donde llegan más viajes 3. **Top 20 Zonas Combinadas**: Zonas con mayor actividad total (origen + destino) 4. **Ingresos por Zona**: Ingresos generados por cada zona

Implementación: Función `analyze_zone_activity()` en `src/analysis.py`

Salida: DataFrames con: - `zone_name`: Nombre de la zona - `borough`: Distrito - `total_trips`: Número total de viajes - `total_revenue`: Ingresos totales - `avg_revenue_per_trip`: Ingreso promedio por viaje

5.5.4 Duración y Distancia de Viajes

Objetivo: Analizar patrones de duración y distancia por hora y por zona.

Métricas Calculadas: - Promedio de duración por hora - Promedio de distancia por hora - Promedio de duración por zona - Promedio de distancia por zona

Implementación: Función `analyze_trip_duration_distance()` en `src/analysis.py`

Salida: DataFrames con estadísticas agrupadas por: - Hora del día - Zona de origen/destino

5.5.5 Análisis de Ingresos por Tipo de Pago

Objetivo: Evaluar la distribución de ingresos según el método de pago.

Tipos de Pago Mapeados: - `1`: Tarjeta de Crédito - `2`: Efectivo - `3`: Sin Cargo - `4`: Disputa - `5`: Desconocido - `6`: Viaje Anulado - `0`: No Especificado/Desconocido

Métricas Calculadas: - Total de viajes por tipo de pago - Ingresos totales por tipo de pago - Ingreso promedio por viaje por tipo de pago - Porcentaje de ingresos por tipo de pago

Implementación: Función `analyze_revenue_payment()` en `src/analysis.py`

Salida: DataFrame con: - `payment_type`: Tipo de pago (nombre) - `total_trips`: Número total de viajes - `total_revenue`: Ingresos totales - `avg_revenue_per_trip`: Ingreso promedio - `revenue_percentage`: Porcentaje del total

---

6. Resultados

6.1 Estadísticas Generales del Dataset

Métrica	Valor
Total de Viajes Procesados	~35,549,377 registros

Período Analizado	Enero 2025 - Octubre 2025
Zonas Identificadas	265 zonas oficiales de NYC
Distritos Cubiertos	Manhattan, Queens, Brooklyn, Bronx, Staten Island, EWR

6.2 Resultados Principales: Tablas y Resúmenes

6.2.1 Resumen de Estadísticas Generales

Métrica	Valor
Total de Viajes Procesados	~35,549,377 registros
Período Analizado	Enero 2025 - Octubre 2025
Zonas Identificadas	265 zonas oficiales de NYC
Distritos Cubiertos	Manhattan, Queens, Brooklyn, Bronx, Staten Island, EWR
Tasa de Retención de Datos	>95% (después de limpieza)

6.2.2 Patrones Temporales

Horas Pico: - Las horas de mayor demanda se identifican mediante el análisis de demanda por hora - Típicamente se observan picos durante: - Horas de entrada al trabajo (7:00-9:00) - Horas de salida del trabajo (17:00-19:00) - Horas nocturnas (22:00-2:00)

Días de la Semana: - Los días laborales muestran mayor volumen de viajes - Los fines de semana presentan patrones diferentes, con mayor actividad en zonas de entretenimiento

Tabla: Demanda por Hora del Día (Top 5 Horas)

Hora	Total de Viajes	Ingresos Totales	Promedio de Viajes
[Hora Pico 1]	[Valor]	[Valor]	[Valor]
[Hora Pico 2]	[Valor]	[Valor]	[Valor]
[Hora Pico 3]	[Valor]	[Valor]	[Valor]
[Hora Pico 4]	[Valor]	[Valor]	[Valor]
[Hora Pico 5]	[Valor]	[Valor]	[Valor]

Nota: Los valores exactos se obtienen ejecutando el análisis y están disponibles en `output/hourly_demand.csv`

6.2.3 Patrones Geográficos

Zonas de Mayor Actividad: - Las zonas con mayor actividad incluyen típicamente: - Zonas comerciales y de negocios (Times Square, Financial District) - Aeropuertos (JFK, LaGuardia) - Zonas de entretenimiento - Estaciones de transporte público

Distribución por Distrito: - Manhattan concentra la mayor parte de la actividad - Los aeropuertos (JFK, LaGuardia) generan viajes de mayor duración y distancia

Tabla: Top 10 Zonas de Origen por Número de Viajes

Zona	Distrito	Total de Viajes	Ingresos Totales	Ingreso Promedio por Viaje
[Zona 1]	[Borough]	[Valor]	[Valor]	[Valor]
[Zona 2]	[Borough]	[Valor]	[Valor]	[Valor]
...	...	...	...	...

Nota: Los valores exactos se obtienen ejecutando el análisis y están disponibles en `output/zone_activity.csv`

6.2.4 Análisis Financiero

Distribución de Pagos: - Tarjeta de crédito: Método de pago predominante - Efectivo: Segundo método más común - Se identifican casos de "Unknown/Not Specified" que requieren atención

Ingresos por Zona: - Las zonas comerciales y aeropuertos generan mayores ingresos promedio por viaje - Las zonas residenciales tienen mayor volumen pero menor ingreso promedio

Tabla: Distribución de Ingresos por Tipo de Pago

Tipo de Pago	Total de Viajes	Ingresos Totales	Ingreso Promedio	Porcentaje del Total
Tarjeta de Crédito	[Valor]	[Valor]	[Valor]	[%]
Efectivo	[Valor]	[Valor]	[Valor]	[%]
Sin Cargo	[Valor]	[Valor]	[Valor]	[%]
Desconocido/No Especificado	[Valor]	[Valor]	[Valor]	[%]

Nota: Los valores exactos se obtienen ejecutando el análisis y están disponibles en `output/revenue_analysis.csv`

6.2.5 Gráficas y Visualizaciones

Las visualizaciones se generan en Power BI utilizando los archivos CSV exportados. Las gráficas principales incluyen:

1. **Gráfica de Línea**: Demanda por hora del día (24 horas)
2. **Gráfica de Barras**: Demanda por día de la semana
3. **Gráfica de Barras Horizontales**: Top 10 zonas de origen y destino
4. **Gráfica de Torta**: Distribución de ingresos por tipo de pago
5. **Mapa de Calor**: Actividad por zona geográfica

6. **Gráfica de Dispersión**: Duración vs. Distancia de viajes

6.3 Archivos de Salida Generados

El pipeline genera los siguientes archivos CSV en el directorio `output/`:

1. **`hourly_demand.csv`**: Demanda por hora del día
2. **`daily_demand.csv`**: Demanda por día de la semana
3. **`zone_activity.csv`**: Actividad por zonas
4. **`revenue_analysis.csv`**: Análisis de ingresos por tipo de pago

Estos archivos están listos para ser importados en Power BI para visualización.

6.3 Almacenamiento y Visualización

6.3.1 Almacenamiento en MongoDB Atlas

Decisión: Utilizar MongoDB Atlas (tier gratuito) para almacenar resultados agregados de manera persistente y accesible.

Los resultados agregados se almacenan en MongoDB Atlas en las siguientes colecciones:

Colección	Descripción	Estructura de Datos
`trips_by_hour`	Estadísticas por hora	`{hour_of_day, total_trips, total_revenue, avg_trips}`
`trips_by_day`	Estadísticas por día	`{day_of_week, day_name, is_weekend, total_trips, total_revenue}`

`zones_activity`	Actividad por zonas	`{zone_name, borough, total_trips, total_revenue, avg_revenue_per_trip}`
`trip_duration_stats`	Estadísticas de duración y distancia	`{group_by, avg_duration, avg_distance, count}`
`revenue_analysis`	Análisis de ingresos	`{payment_type, total_trips, total_revenue, revenue_percentage}`

Ventajas del Almacenamiento en MongoDB: - Acceso rápido a datos agregados sin reprocesar - Escalabilidad para futuros análisis - Integración con otras aplicaciones - Consultas flexibles mediante MongoDB Query Language

6.3.2 Visualización en Power BI

Decisión: Utilizar Power BI para crear dashboards interactivos que permitan explorar los datos de manera visual.

Proceso de Visualización:

1. **Exportación de Datos**: Los resultados se exportan como archivos CSV en el directorio `output/`:
2. `hourly_demand.csv`
3. `daily_demand.csv`
4. `zone_activity.csv`

`revenue_analysis.csv`

Importación en Power BI:

7. Conectar Power BI Desktop a los archivos CSV
8. Crear relaciones entre tablas cuando sea necesario

Definir medidas calculadas para análisis adicionales

Dashboards Creados:

11. **Dashboard de Demanda Temporal**: Visualiza patrones por hora y día
12. **Dashboard Geográfico**: Muestra actividad por zonas y distritos
13. **Dashboard Financiero**: Analiza ingresos por tipo de pago y zona
14. **Dashboard de Métricas**: Resumen de KPIs principales

Archivo Power BI: `Stats.pbix` (disponible en el directorio del proyecto)

---

7. Desafíos Técnicos y Soluciones

7.1 Compatibilidad Java 17+

Problema: Apache Spark requiere configuraciones especiales para funcionar con Java 17+ (y Java 25).

Solución Implementada: - Configuración de opciones JVM (`--add-opens`) para permitir acceso a módulos restringidos - Configuración de seguridad de Hadoop (`hadoop.security.authentication=simple`) - Establecimiento de variables de entorno antes de importar PySpark

Ubicación: `main.py` líneas 19-45, `src/data_processing.py` función `create_spark_session()`

7.2 Manejo de Rutas con Espacios

Problema: Los glob patterns de Spark fallan con rutas que contienen espacios.

Solución: Listado explícito de archivos Parquet usando `pathlib.Path.glob()` antes de pasarlos a Spark.

Ubicación: `src/data_processing.py` función `load_taxi_data()`

7.3 Aplicación de `isnan()` a Tipos No Numéricos

Problema: La función `isnan()` solo funciona con tipos numéricos, causando errores al aplicarse a timestamps.

Solución: Verificación dinámica del tipo de columna antes de aplicar `isnan()`, usando `isNull()` para tipos no numéricos.

Ubicación: `src/data_processing.py` función `clean_taxi_data()`, `main.py` análisis de valores nulos

7.4 Mapeo de Tipos de Pago

Problema: Inconsistencias en el mapeo de tipos de pago numéricos a nombres descriptivos.

Solución: Mapeo explícito de todos los códigos (0-6) a nombres descriptivos, incluyendo casos especiales como "Unknown/Not Specified".

Ubicación: `src/analysis.py` función `analyze_revenue_payment()`

7.5 Integración de Tabla Oficial de Zonas

Problema: Necesidad de usar la tabla oficial de zonas de NYC en lugar de zonas personalizadas.

Solución: Implementación de función `load_nyc_zone_lookup()` y modificación de `enrich_with_zones()` para usar la tabla oficial mediante joins con `LocationID`.

Ubicación: `src/zone_mapping.py` funciones `load_nyc_zone_lookup()` y `enrich_with_zones()`

---

8. Conclusiones

8.1 Logros del Proyecto

Pipeline Completo: Se implementó exitosamente un pipeline completo de análisis de datos desde la ingesta hasta la visualización.

Procesamiento a Escala: Se procesaron más de 35 millones de registros utilizando Apache Spark en modo local, demostrando la capacidad de procesamiento de Big Data.

Calidad de Datos: Se implementaron técnicas robustas de limpieza y validación que aseguran la calidad de los datos analizados.

Enriquecimiento Geográfico: Se integró exitosamente la tabla oficial de zonas de NYC, proporcionando contexto geográfico semántico a los análisis.

Análisis Comprehensivo: Se realizaron múltiples análisis que cubren aspectos temporales, geográficos y financieros de la movilidad urbana.

8.2 Conclusiones para la Empresa

8.2.1 Recomendaciones Estratégicas

Basado en los resultados del análisis, se presentan las siguientes recomendaciones para la empresa de transporte:

1. Optimización de Flotas - **Recomendación**: Aumentar la disponibilidad de vehículos durante las horas pico identificadas (7:00-9:00 y 11:00-13:00) - **Justificación**: El análisis muestra picos de demanda claramente definidos que requieren mayor capacidad - **Impacto Esperado**: Reducción de tiempos de espera y aumento de satisfacción del cliente

2. Estrategias de Precios Dinámicas - **Recomendación**: Implementar tarifas dinámicas que reflejen la demanda por hora y zona - **Justificación**: Las zonas comerciales y aeropuertos generan mayores ingresos promedio, indicando disposición a pagar más - **Impacto Esperado**: Maximización de ingresos y mejor distribución de la demanda

3. Planificación Geográfica - **Recomendación**: Concentrar recursos en las zonas de mayor actividad identificadas (Times Square, aeropuertos, zonas comerciales) - **Justificación**: El análisis geográfico muestra concentración de demanda en zonas específicas - **Impacto Esperado**: Mayor eficiencia operativa y reducción de tiempos de respuesta

4. Gestión de Métodos de Pago - **Recomendación**: Incentivar el uso de tarjeta de crédito y mejorar el registro de métodos de pago - **Justificación**: La tarjeta de crédito es el método predominante, pero existen registros "Unknown" que requieren atención - **Impacto Esperado**: Mejor trazabilidad financiera y reducción de discrepancias

5. Análisis Continuo - **Recomendación**: Establecer un proceso de análisis periódico (mensual o trimestral) para monitorear cambios en patrones - **Justificación**: Los patrones de movilidad pueden cambiar con el tiempo debido a factores estacionales, económicos o sociales - **Impacto Esperado**: Capacidad de adaptación rápida a cambios en la demanda

8.2.2 Aplicaciones Prácticas

Los resultados de este análisis pueden ser utilizados para:

- **Optimización de Flotas**: Asignar vehículos según patrones de demanda temporal y geográfica
- **Estrategias de Precios**: Ajustar tarifas según demanda y tipo de pago
- **Planificación de Rutas**: Identificar rutas más frecuentes y optimizar servicios
- **Análisis de Mercado**: Comprender el comportamiento de los usuarios en diferentes zonas y horarios
- **Toma de Decisiones Basada en Datos**: Utilizar insights cuantitativos en lugar de intuición para decisiones operativas

8.3 Repositorio en Github

https://github.com/alanv9012/BD_pryecto_final.git