Toyota Used Car Price Prediction

DS512/513 Data Analytics
DS514/515 Data Science

68199160**268** ธนกร สอนเพ็ชร
68199160**255** จตุพร ตะเคียนทอง
68199160**310** อาภากร กาละ

14 December 2025

DATA PROJECT CANVAS

Designed by: Thanakorn (268), Jatuporn(255), Arphakon(310) / Date: 14-Dec-2025

Title: Toyota Used Car Price Prediction Project (SWU Motors)

1. Problem Statement/Background

- What do we know about?
- What problem are you trying to solve?
  - SWU Motors เป็นดีลเลอร์รถมือสองที่กำลังขยายตัวและจ้างพนักงานขายใหม่ (Junior Salespeople) จำนวนมาก
  - **ยอดขายลดลง 18% ในช่วงที่ผ่านมา**
  - **พนักงานขายใหม่ขาดประสบการณ์ในการตั้งราคา (Pricing) รถ Toyota มือสองที่รับเข้ามา ทำให้เกิดความผิดพลาดในการตั้งราคาขาย**
  - เรามี dataset (toyota.csv) ที่รวบรวมราคาขายจริงจากผู้ค้าปลีกรายอื่นในตลาด
- What is the business problem?
  - Revenue Loss: การตั้งราคาผิดพลาดทำให้เสียโอกาสในการขาย (ถ้าราคาสูงเกินไปรถขายไม่ออก) หรือเสียกำไร (ถ้าราคาต่ำเกินไป)
- Who are the stakeholders?
  - End-Users: พนักงานขายใหม่ (Junior Salespeople)
  - Management: ผู้จัดการฝ่ายขาย (ต้องการกู้ยอดขาย)
  - Data Science, Data Analytics

2. SMART Objectives/ Value Propositions

SMART Objectives:

- พัฒนาโมเดลทำนายราคาให้แล้ว**เสร็จภายใน 2 สัปดาห์ โดยมีค่าความคลาดเคลื่อน (RMSE) ต่ำกว่า 1,250 ปอนด์** เพื่อช่วยให้พนักงานใหม่**ตั้งราคาได้อย่างถูกต้องและกู้คืนยอดขายที่ลดลง 18%**

Value Proposition:

- สร้างเครื่องมือ **"Pricing Tool"** เพื่อช่วยให้พนักงานใหม่ตัดสินใจได้เร็วขึ้น แม่นยำขึ้น และกู้ยอดขายกลับคืนมา

3. Questions/Hypothesis

- Analytical Questions
  - ปัจจัยใดส่งผลต่อราคา Toyota มากที่สุด? (ปีจดทะเบียน?, เลขไมล์?, ขนาดเครื่องยนต์?)
  - ลักษณะการกระจายตัวของราคาในตลาดเป็นอย่างไร?
- Predictive Hypothesis
  - H1: เลขไมล์ (Mileage) น่าจะมีความสัมพันธ์เชิงลบกับราคา (ยิ่งวิ่งเยอะ ราคายิ่งตก)
  - H2: ขนาดเครื่องยนต์ (Engine Size) น่าจะมีความสัมพันธ์เชิงบวกกับราคา (เครื่องใหญ่ = ราคารถยิ่งสูง)

What can we predict?

- *ราคาขายต่อของรถยนต์ Toyota (£)*

4. Data Sources/Attributes

- Data sources & collection
  - ไฟล์ toyota.csv (รวบรวมข้อมูลราคาและคุณลักษณะรถจากผู้ค้าปลีกภายนอก)
- Data cleaning & preprocessing
  - ตรวจสอบ Missing Values และ Duplicate Rows
  - ตรวจสอบ Outliers (เช่น รถที่มีค่าภาษี tax เป็น 0 หรือค่า mpg ผิดปกติ
- Target variables & feature
  - **Target:** `price`.
  - **Features:** `model`, `year`, `transmission`, `mileage`, `fuelType`, `tax`, `mpg`, `engineSize`.
- Encoding & scaling strategies
  - **Encoding:** ใช้ One-Hot Encoding แปลงข้อมูลกลุ่ม (model, transmission, fuelType) เป็นตัวเลข
  - **Scaling:** ใช้ StandardScaler เพื่อปรับมาตรฐานข้อมูลตัวเลข (mileage, tax, mpg)

5. Analysis/Model Development

- Analytics Methodology
  - **Descriptive Analytics:** ใช้สถิติพื้นฐาน (Mean, Median, Skewness) และ Visualization เพื่อเข้าใจโครงสร้างตลาด
  - **Predictive Analytics**: ใช้ Supervised Learning (Regression) สร้างโมเดลทำนายราคาขาย
- Modeling Methodology
  - ***Pipeline:*** *ใช้ OneHotEncoder (แปลงกลุ่มข้อมูล) และ StandardScaler (ปรับสเกลตัวเลข) เพื่อเตรียมข้อมูลอัตโนมัติ*
  - ***Model Selection: เปรียบเทียบ 2 วิธีการ:***
  - ***Linear Models (Baseline)****: ทดสอบ Ridge, Lasso, Elastic Net (สมมติฐานเส้นตรง)*
  - ***Random Forest (Challenger)****: ใช้ Ensemble Learning เพื่อจับรูปแบบ Non-linear ที่ซับซ้อน*
  - ***Evaluation****: จูนพารามิเตอร์ด้วย GridSearchCV และวัดผลด้วย 5-Fold Cross-Validation เพื่อความแม่นยำสูงสุด*

6. Findings and Insights

- Business Insights

- Predictive Results

7. Recommendation/Action and Impact

Action:

Impact:

Business Problem & Objective

- **Problem (ปัญหา):** พนักงานขายใหม่ (SWU Motor) สาขาสหราชอาณาจักร ขาดประสบการณ์ในการตั้งราคา ทำให้ตั้งราคาผิดพลาด ส่งผลให้**ยอดขายลดลง 18%** และเสียโอกาสในการทำกำไร

- **Objective (วัตถุประสงค์):** สร้าง "เครื่องมือช่วยตั้งราคา (Pricing Assistant Tool)" ที่แม่นยำด้วย Machine Learning

- **Success Criteria (เกณฑ์ความสำเร็จ):** โมเดล**ต้องมีค่าความคลาดเคลื่อนเฉลี่ย (RMSE) ไม่เกิน £1,250**

Data Dictionary

Variable Name	Data Type	Description	Example
model	Nominal	รุ่นของรถยนต์ Toyota	GT86, Yaris, Aygo, Supra
year	Interval	ปีที่ผลิตรถยนต์	2016, 2019, 2020
price	Ratio (Continuous)	ราคาขายรถยนต์มือสอง (หน่วยปอนด์ £)	16000, 8500, 24990
transmission	Nominal	ระบบเกียร์	Manual, Automatic, Semi-Auto
mileage	Ratio (Continuous)	ระยะทางที่รถวิ่งมาแล้ว (หน่วยไมล์)	24089, 500, 100000
fuelType	Nominal	ประเภทเชื้อเพลิง	Petrol, Diesel, Hybrid, Other
tax	Ratio (Continuous)	ภาษีรถยนต์รายปี (Road Tax) (หน่วยปอนด์ £)	0, 145, 265
mpg	Ratio (Continuous)	อัตราการประหยัดน้ำมัน (Miles Per Gallon)	36.2, 55.4, 28.0
engineSize	Ratio (Continuous)	ขนาดเครื่องยนต์ (หน่วยลิตร)	1.0, 2.0, 3.0

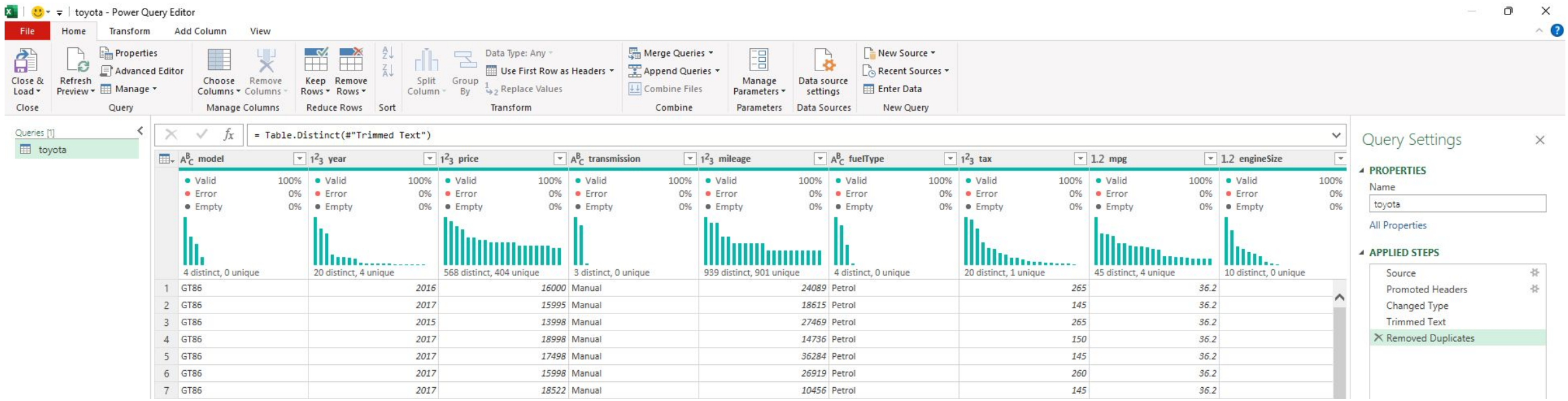
Target and Feature Variable

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	GT86	2016	16000	Manual	24089	Petrol	265	36.2	2.0
1	GT86	2017	15995	Manual	18615	Petrol	145	36.2	2.0
2	GT86	2015	13998	Manual	27469	Petrol	265	36.2	2.0
3	GT86	2017	18998	Manual	14736	Petrol	150	36.2	2.0
4	GT86	2017	17498	Manual	36284	Petrol	145	36.2	2.0
...	...	...	...	...	...	...	...	...	...
6733	IQ	2011	5500	Automatic	30000	Petrol	20	58.9	1.0
6734	Urban Cruiser	2011	4985	Manual	36154	Petrol	125	50.4	1.3
6735	Urban Cruiser	2012	4995	Manual	46000	Diesel	125	57.6	1.4
6736	Urban Cruiser	2011	3995	Manual	60700	Petrol	125	50.4	1.3
6737	Urban Cruiser	2011	4495	Manual	45128	Petrol	125	50.4	1.3

Feature

Target

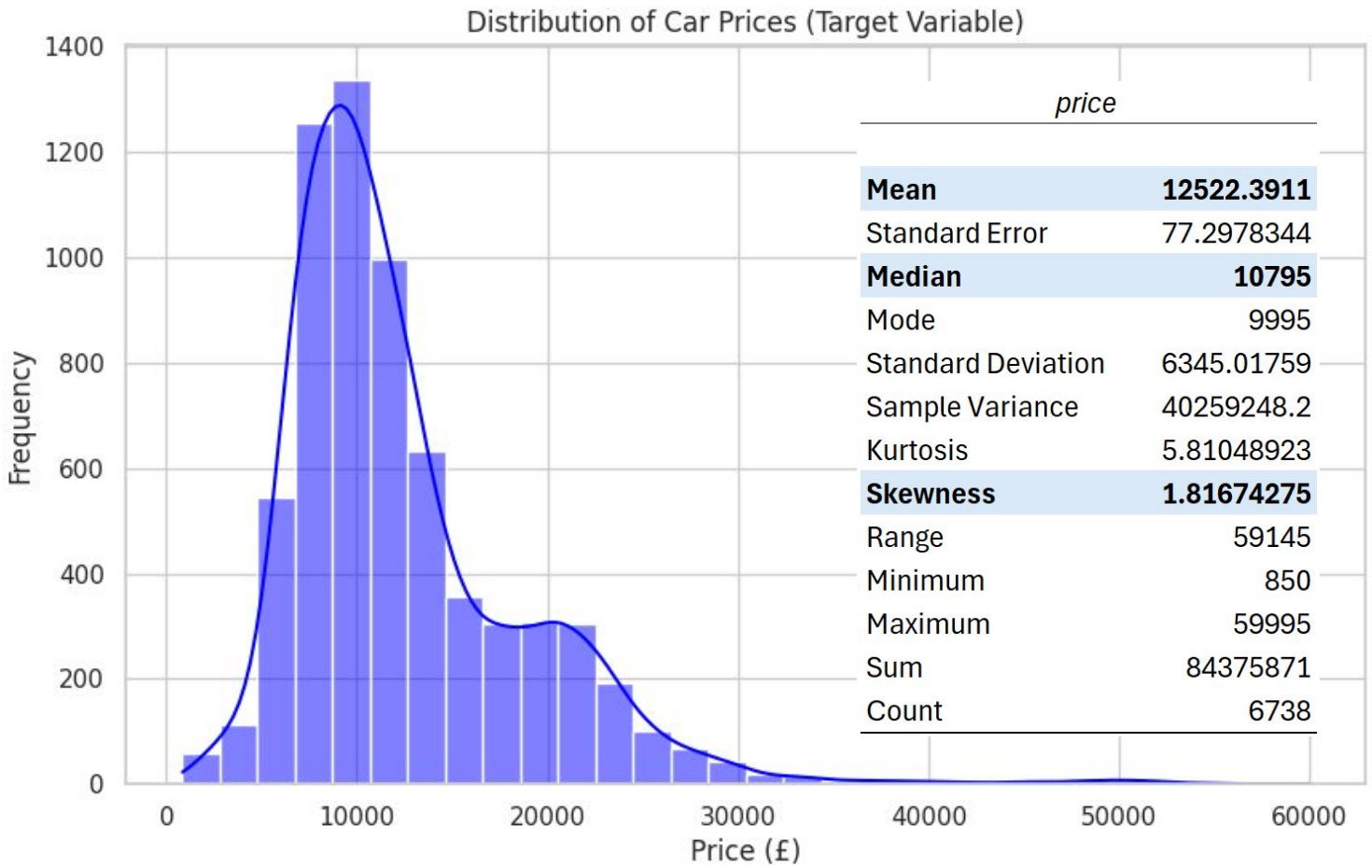
Data Preparation with Excel



Dataset: toyota.csv (from kaggle.com)
Tools Used: Excel (Power Query), Data Analysis Toolpak
Process:

1. **Cleaning:** ใช้ Power Query ตรวจสอบ Data Type, Trim ช่องว่าง, ลบข้อมูลซ้ำ ตรวจสอบความผิดปกติ (Data Profiling): Valid: ควรเป็น 100% ไม่มี Error หรือ Empty

Findings and Insights



price	
Mean	**12522.3911**
Standard Error	77.2978344
Median	**10795**
Mode	9995
Standard Deviation	6345.01759
Sample Variance	40259248.2
Kurtosis	5.81048923
Skewness	**1.81674275**
Range	59145
Minimum	850
Maximum	59995
Sum	84375871
Count	6738

Data: Price

สรุปพฤติกรรมราคารถ (Key Insights)

1. โครงสร้างตลาดแบบ "เบ้ขวา" (Right Skewed Market)

- ตลาดรถ Toyota ส่วนใหญ่เป็น "รถบ้านราคาประหยัด" (Budget Segment) แต่ค่าเฉลี่ยตลาดถูกดึงให้สูงเกินจริงจาก Premium Segment เพียงส่วนน้อย

2. ความเสี่ยงจากการกระจายตัวสูง (High Volatility)

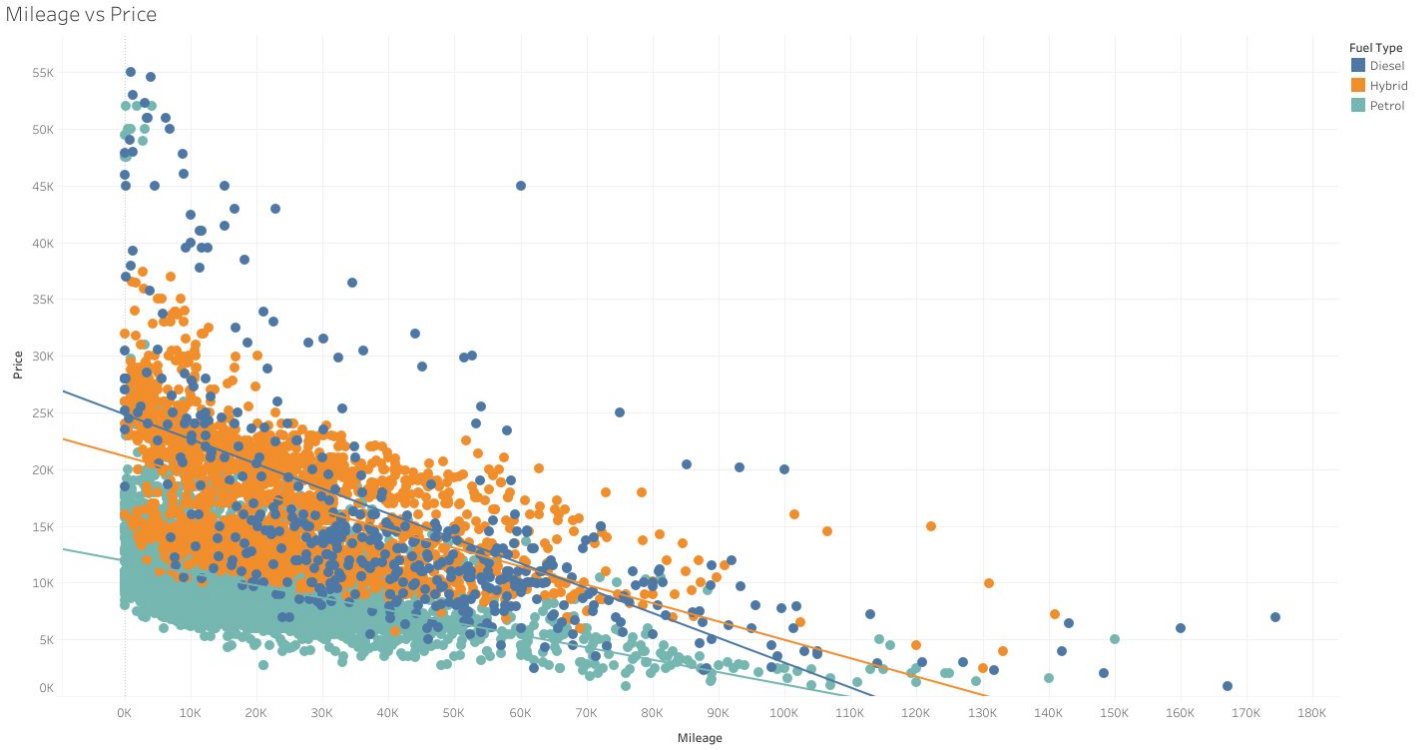
ค่าเบี่ยงเบนมาตรฐาน (SD) สูงถึง £6,345 และช่วงราคากว้างมาก (£850 - £60,000)

- ราคามีความหลากหลายสูงมาก ทำให้ "ยากต่อการประเมินด้วยสายตา" หรือประสบการณ์เพียงอย่างเดียว

Insights: ความซับซ้อนและช่วงราคาที่กว้างนี้ คือสาเหตุหลักที่ทำให้พนักงานขายตั้งราคาผิดพลาด และจำเป็นต้องนำ Machine Learning มาช่วยคำนวณเพื่อให้ได้ราคาที่แม่นยำ

Findings and Insights

Mileage vs Price
(Color by Fuel Type)



Data: Price, Mileage, Fuel Type

เจาะลึกรายกลุ่ม (Segment Analysis)

1. **Hybrid (ไฮบริด): "ราคาสูง & วิ่งน้อย"**
   ตำแหน่ง: จุดส่วนใหญ่อยู่ด้าน ซ้ายบน ของกราฟ
   - เป็นรถรุ่นใหม่ เทคโนโลยีสูง ทำให้ราคามือสองยังแข็งแรง

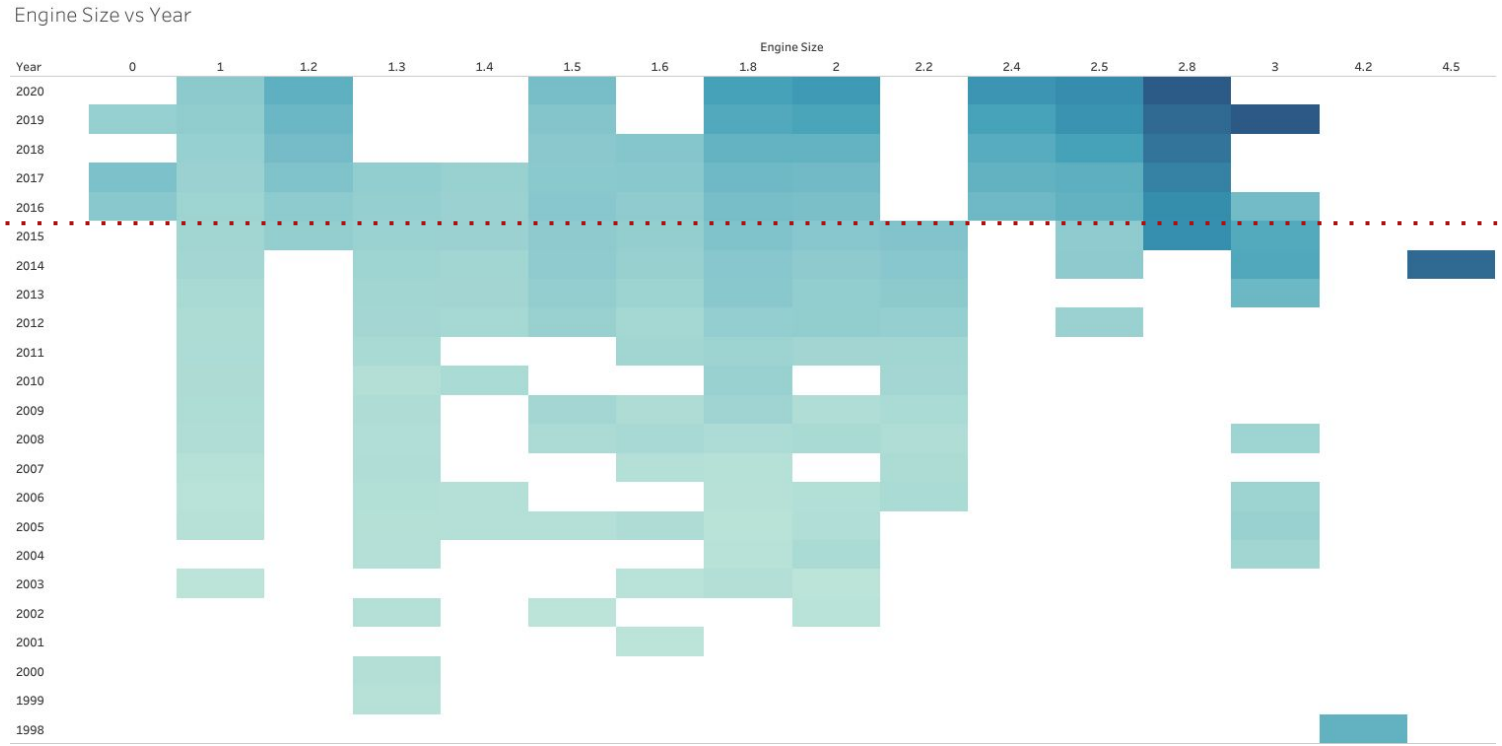
2. **Diesel (ดีเซล): "อึด ถึก ทน & วิ่งเยอะ"**
   ตำแหน่ง: กระจายตัวไปทาง ขวาสุด ของกราฟ
   - รถดีเซล ถูกออกแบบมาเพื่อการใช้งานหนักและวิ่งระยะไกล แม้เลขไมล์จะสูงแต่ราคาก็ไม่ได้ตกหมือนรถเบนซิน

3. **Petrol (เบนซิน): "รถตลาด & ราคาประหยัด"**
   ตำแหน่ง: กองอยู่ด้าน ล่างซ้าย
   - ราคาลดเร็วเมื่อ Mileage เพิ่ม เป็นรถตลาด การแข่งขันสูง

Insight: ราคาลดลง เมื่อ Mileage ขึ้น แต่ราคากระจายกว้างมาก ใน Mileage เดียวกัน Fuel Type มีผลชัดเจนต่อระดับราคา เป็นเหตุผลหลักที่ต้องใช้ ML ในการตั้งราคา

Findings and Insights

Engine Size vs Year
(Color by Price)



Data: Price, Engine_Size, Year

Key Observations

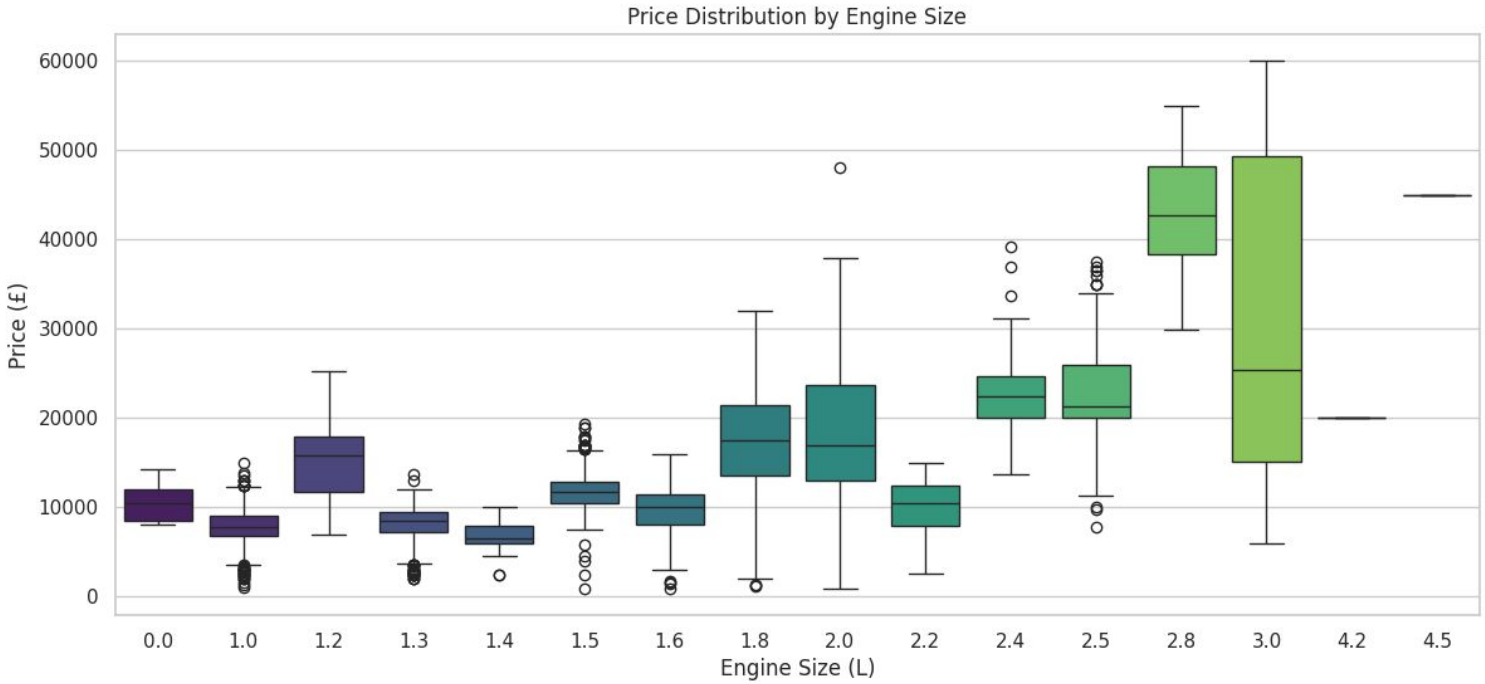
- รถ Toyota ส่วนใหญ่ใช้ เครื่องยนต์ขนาดเล็ก–กลาง (1.0–2.0 L) ซึ่งเป็นกลุ่มรถบ้านราคาประหยัด
- รถรุ่นใหม่ (หลังปี ~2016) มี ความหลากหลายของขนาดเครื่องยนต์มากขึ้น และเริ่มเห็นเครื่องยนต์ขนาดกลาง–ใหญ่ (1.8–3.0 L) ชัดเจน

Price Insight (จากความเข้มของสี)

- **สีเข้ม = ราคาสูง**
  พบมากในรถ ปีใหม่ + เครื่องยนต์ขนาดกลาง–ใหญ่
- **สีอ่อน = ราคาต่ำ**
  กระจุกตัวในรถ **ปีเก่า + เครื่องยนต์ขนาดเล็**

Insight: ราคาสูงขึ้นตามปีรถและขนาดเครื่องยนต์อย่างชัดเจน

Findings and Insights



Data: Price, Engine_Size

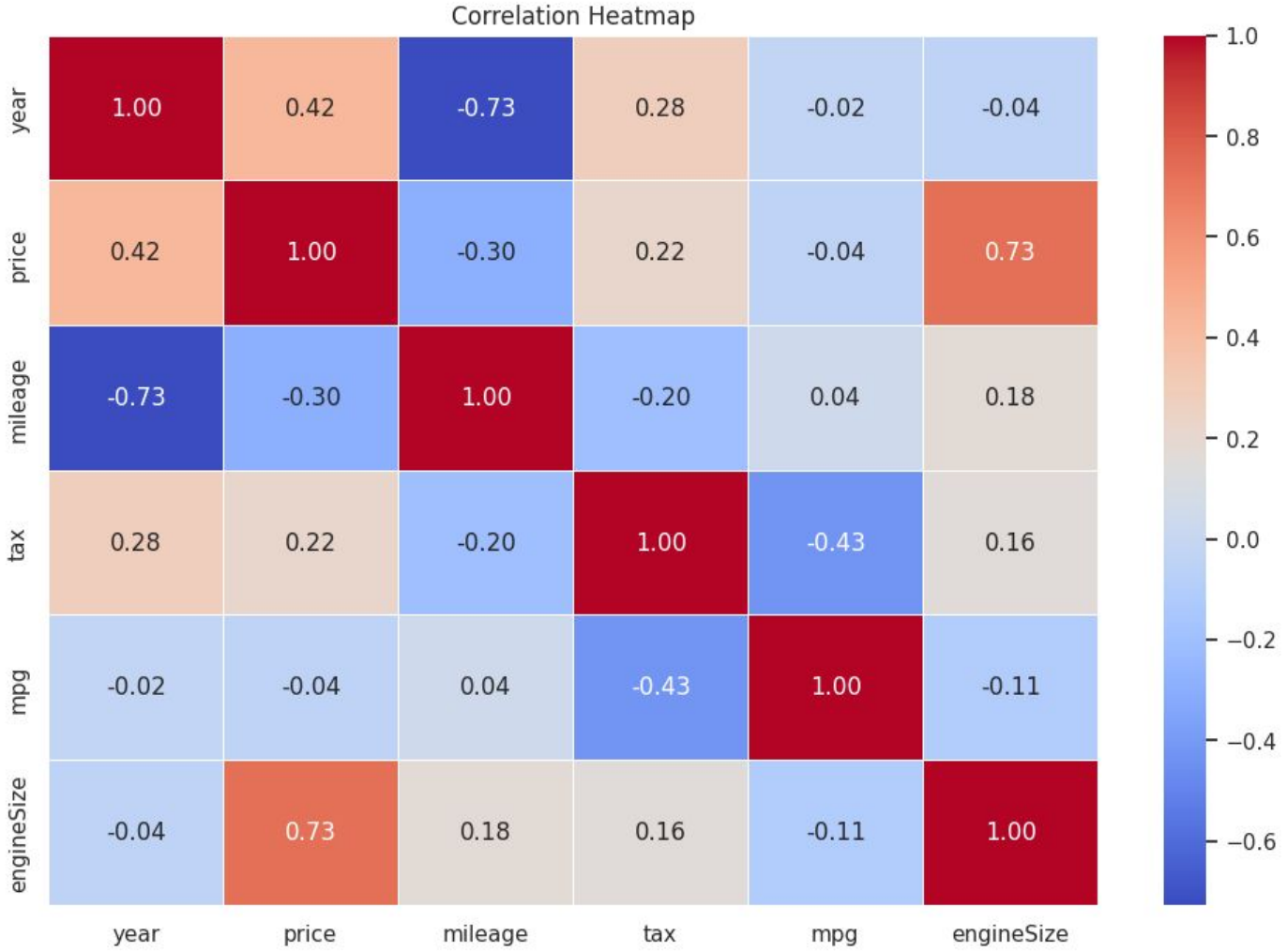
Key Observations:

- เครื่องใหญ่ ⟶ ราคาสูงขึ้นชัดเจน, ความผันผวนของราคาสูง
- เครื่องเล็ก ⟶ ราคาคงที่กว่า, ราคาค่อนข้างใกล้เคียงกัน

- **3.0 L: Median ไม่สูงที่สุด แต่ราคากระจายสุด มีทั้งรุ่นธรรมดาและรุ่นแพงมากปะปนกัน**

Insight: Engine Size เป็นปัจจัยหลัก ที่กำหนดระดับราคา แต่ขนาดเครื่องเพียงอย่างเดียวไม่เพียงพอในการตั้งราคา โดยเฉพาะในกลุ่มเครื่องยนต์ขนาดกลาง–ใหญ่ที่ราคาผันผวนสูง จึงจำเป็นต้องใช้ Machine Learning เพื่อประเมินราคาที่แม่นยำ

Findings and Insights



Data: Price, Engine_Size

ความสัมพันธ์กับราคา (Price)

- **Engine Size ↔ Price : +0.73 (สูงมาก)**
  ⟶ ขนาดเครื่องยนต์เป็นปัจจัยที่มีผลต่อราคามากที่สุด
- **Year ↔ Price : +0.42 (ปานกลาง)**
  ⟶ รถปีใหม่มีแนวโน้มราคาสูงกว่า
- **Mileage ↔ Price : −0.30 (เชิงลบ)**
  ⟶ รถที่วิ่งเยอะ ราคาลดลง
- **Tax ↔ Price : +0.22 (ต่ำ–ปานกลาง)**
  ⟶ ภาษีสูงสัมพันธ์กับรถสเปกสูง / ราคาแพง
- **MPG ↔ Price : −0.04 (แทบไม่มีผล)**
  ⟶ ความประหยัดน้ำมันแทบไม่ส่งผลต่อราคาโดยตรง

Insight จากการวิเคราะห์พบว่า ราคารถยนต์ได้รับอิทธิพลหลักจาก**ขนาดเครื่องยนต์และปีรถ ซึ่งมีความสัมพันธ์เชิงบวกกับราคาอย่างค่อนข้างชัดเจน**

ในขณะที่ ระยะทางการใช้งาน **(Mileage) มีความสัมพันธ์เชิงลบกับราคา** สะท้อนถึงผลของการเสื่อมมูลค่าจากการใช้งาน

ส่วน อัตราการประหยัดน้ำมัน (MPG) มีความสัมพันธ์กับราคาในระดับต่ำมาก จึงมีบทบาทค่อนข้างจำกัดในการอธิบายราคารถยนต์

Build Machine Learning Model: Import Dataset

```
import pandas as pd
gitURL = 'https://raw.githubusercontent.com/thanakornsonphet-swu/ds514-515-toyota-used-car-price-prediction/refs/heads/main/toyota.csv'
df = pd.read_csv(gitURL)
df
```

1.Import dataset: Import dataset จาก Github, ดูข้อมูลสรุปของ dataframe ที่ import มา

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	GT86	2016	16000	Manual	24089	Petrol	265	36.2	2.0
1	GT86	2017	15995	Manual	18615	Petrol	145	36.2	2.0
2	GT86	2015	13998	Manual	27469	Petrol	265	36.2	2.0
3	GT86	2017	18998	Manual	14736	Petrol	150	36.2	2.0
4	GT86	2017	17498	Manual	36284	Petrol	145	36.2	2.0
...	...	...	...	...	...	...	...	...	...
6733	IQ	2011	5500	Automatic	30000	Petrol	20	58.9	1.0
6734	Urban Cruiser	2011	4985	Manual	36154	Petrol	125	50.4	1.3
6735	Urban Cruiser	2012	4995	Manual	46000	Diesel	125	57.6	1.4
6736	Urban Cruiser	2011	3995	Manual	60700	Petrol	125	50.4	1.3
6737	Urban Cruiser	2011	4495	Manual	45128	Petrol	125	50.4	1.3

6738 rows × 9 columns

```
# Data Info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6738 entries, 0 to 6737
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   model         6738 non-null   object
 1   year          6738 non-null   int64
 2   price         6738 non-null   int64
 3   transmission  6738 non-null   object
 4   mileage       6738 non-null   int64
 5   fuelType      6738 non-null   object
 6   tax           6738 non-null   int64
 7   mpg           6738 non-null   float64
 8   engineSize    6738 non-null   float64
dtypes: float64(2), int64(4), object(3)
memory usage: 473.9+ KB
```

Build Machine Learning Model: Import Dataset

2. Preprocessing: เช็ค/ลบค่าซ้ำและจัดการ Missing Value, ดูข้อมูลทางสถิติเบื้องต้น

```python
# เช็คข้อมูลสูญหาย (Missing Values)
print("Missing Values:\n", df.isnull().sum())

# เช็คข้อมูลซ้ำ (Duplicate Rows)
duplicates = df.duplicated().sum()
print(f"\nNumber of duplicate rows: {duplicates}")

# ถ้ามีข้อมูลซ้ำ ให้ลบทิ้ง (Best Practice)
if duplicates > 0:
    df = df.drop_duplicates()
    print(f"Removed duplicates. New shape: {df.shape}")
```

```
Missing Values:
 model           0
year             0
price            0
transmission     0
mileage          0
fuelType         0
tax              0
mpg              0
engineSize       0
dtype: int64

Number of duplicate rows: 39
Removed duplicates. New shape: (6699, 9)
```

```python
# Descriptive Statistics
display(df.describe())
```

```
--- Descriptive Statistics ---
```

	year	price	mileage	tax	mpg	engineSize
count	6738.000000	6738.000000	6738.000000	6738.000000	6738.000000	6738.000000
mean	2016.748145	12522.391066	22857.413921	94.697240	63.042223	1.471297
std	2.204062	6345.017587	19125.464147	73.880776	15.836710	0.436159
min	1998.000000	850.000000	2.000000	0.000000	2.800000	0.000000
25%	2016.000000	8290.000000	9446.000000	0.000000	55.400000	1.000000
50%	2017.000000	10795.000000	18513.000000	135.000000	62.800000	1.500000
75%	2018.000000	14995.000000	31063.750000	145.000000	69.000000	1.800000
max	2020.000000	59995.000000	174419.000000	565.000000	235.000000	4.500000

Build Machine Learning Model: Data pre-processing

3. กำหนด Feature (X) และ Target (y) / แบ่งข้อมูล Train 80% / Test 20%

```
# 1. กำหนด Feature (X) และ Target (y)
target_col = 'price' # ตัวแปรที่เราต้องการทำนาย
X = df.drop(target_col, axis=1) # ข้อมูลทั้งหมด ยกเว้นราคา
y = df[target_col] # ราคาอย่างเดียว

# 2. แบ่งข้อมูล Train 80% / Test 20%
# random_state=42 เพื่อล็อคผลการสุ่มให้เหมือนเดิมทุกครั้ง
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print(f"Training set size: {X_train.shape}")
print(f"Testing set size: {X_test.shape}")
```

```
Training set size: (5359, 8)
Testing set size: (1340, 8)
```

4. Transformation: ใช้ OneHotEncoder (แปลงข้อความ) และ StandardScaler (ปรับสเกลตัวเลข)

```
# 1. แยกชื่อคอลัมน์ตามประเภท
# คอลัมน์ตัวเลข (Numerical)
numerical_features = ['year', 'mileage', 'tax', 'mpg', 'engineSize']

# คอลัมน์หมวดหมู่ (Categorical)
categorical_features = ['model', 'transmission', 'fuelType']

# 2. สร้าง Transformers (ตัวแปลงข้อมูล)
# สำหรับตัวเลข: ปรับสเกลให้เป็นมาตรฐาน (StandardScaler)
numerical_transformer = StandardScaler()

# สำหรับข้อความ: แปลงเป็นตัวเลข 0/1 (OneHotEncoder)
# handle_unknown='ignore': ถ้าเจอหมวดหมู่แปลกๆ ในอนาคต ให้ข้ามไป ไม่ต้อง Error
categorical_transformer = OneHotEncoder(handle_unknown='ignore')

# 3. รวมร่างเป็น ColumnTransformer (ตัวจัดการคอลัมน์)
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_features),
        ('cat', categorical_transformer, categorical_features)
    ])

# ลองแสดงหน้าตา Pipeline
print(preprocessor)
```

```
ColumnTransformer(transformers=[('num', StandardScaler(),
                                 ['year', 'mileage', 'tax', 'mpg',
                                  'engineSize']),
                                ('cat', OneHotEncoder(handle_unknown='ignore'),
                                 ['model', 'transmission', 'fuelType'])])
```

Build Machine Learning Model: Linear Models

5. Linear Models: Ridge, Lasso, Elastic Net (ทดสอบสมมติฐานเส้นตรง)

```python
# --- 1. กำหนด Parameter Grid (ช่วงค่า Alpha ที่จะให้ AI ลองสุ่ม) ---
# Alpha คือความแรงในการบีบโมเดล (ยิ่งเยอะ ยิ่งบีบมาก)
alphas = [0.01, 0.1, 1, 10, 100]

# --- 2. สร้าง GridSearch สำหรับแต่ละโมเดล ---

# 2.1 Ridge Regression
print("Training Ridge...")
ridge_pipe = Pipeline([('prep', preprocessor), ('algo', Ridge())])
grid_ridge = GridSearchCV(ridge_pipe, {'algo__alpha': alphas}, cv=5, scoring='r2', n_jobs=-1)
grid_ridge.fit(X_train, y_train)

# 2.2 Lasso Regression
print("Training Lasso...")
lasso_pipe = Pipeline([('prep', preprocessor), ('algo', Lasso(max_iter=10000))])
grid_lasso = GridSearchCV(lasso_pipe, {'algo__alpha': alphas}, cv=5, scoring='r2', n_jobs=-1)
grid_lasso.fit(X_train, y_train)

# 2.3 Elastic Net
print("Training Elastic Net...")
enet_pipe = Pipeline([('prep', preprocessor), ('algo', ElasticNet(max_iter=10000))])
# Elastic Net ต้องจูนทั้ง alpha และ l1_ratio (ผสม Lasso กี่ %)
grid_enet = GridSearchCV(enet_pipe,
                         {'algo__alpha': [0.01, 0.1, 1], 'algo__l1_ratio': [0.2, 0.5, 0.8]},
                         cv=5, scoring='r2', n_jobs=-1)
grid_enet.fit(X_train, y_train)

print("Linear Models Training Completed!")
```

	Model	R2 Score	RMSE (£)	Best Params
1	Lasso	0.931804	1680.243154	{'algo__alpha': 0.01}
0	Ridge	0.931800	1680.285084	{'algo__alpha': 0.1}
2	Elastic Net	0.919481	1825.746414	{'algo__alpha': 0.01, 'algo__l1_ratio': 0.8}

```
--------------------------------------------------
Best Model by RMSE: ** Lasso **
R2 Score: 0.9318
RMSE: £1680.24
Best Params: {'algo__alpha': 0.01}
--------------------------------------------------
```

Best Model: Lasso Regression

RMSE: £1,680.24

Analysis: โมเดลมีค่า R2 Score ค่อนข้างดี แต่ไม่สามารถจับรูปแบบความซับซ้อนของราคารถได้ (ยังไม่ผ่านเกณฑ์ Target £1,250)

Build Machine Learning Model: Random Forest

6. ทดลองเทรนกับ Model Random Forest

```python
# 1. สร้าง Pipeline
rf_pipe = Pipeline([
    ('prep', preprocessor),
    ('rf', RandomForestRegressor(random_state=42))
])

# 2. กำหนด Parameter Grid
rf_params = {
    'rf__n_estimators': [100, 200],
    'rf__max_depth': [10, 20, None],
    'rf__min_samples_leaf': [1, 2, 4]
}

# 3. เริ่ม GridSearch (เน้นหา RMSE ที่ต่ำที่สุดเป็นหลัก)
grid_rf = GridSearchCV(
    rf_pipe,
    rf_params,
    cv=5,
    scoring='neg_root_mean_squared_error', # Optimize หา RMSE ที่ต่ำที่สุด
    n_jobs=-1,
    verbose=1
)

grid_rf.fit(X_train, y_train)


# Best RMSE
best_rmse_cv = -grid_rf.best_score_

# Best R2
best_model = grid_rf.best_estimator_
cv_r2_scores = cross_val_score(best_model, X_train, y_train, cv=5, scoring='r2')
best_r2_cv = cv_r2_scores.mean()
```

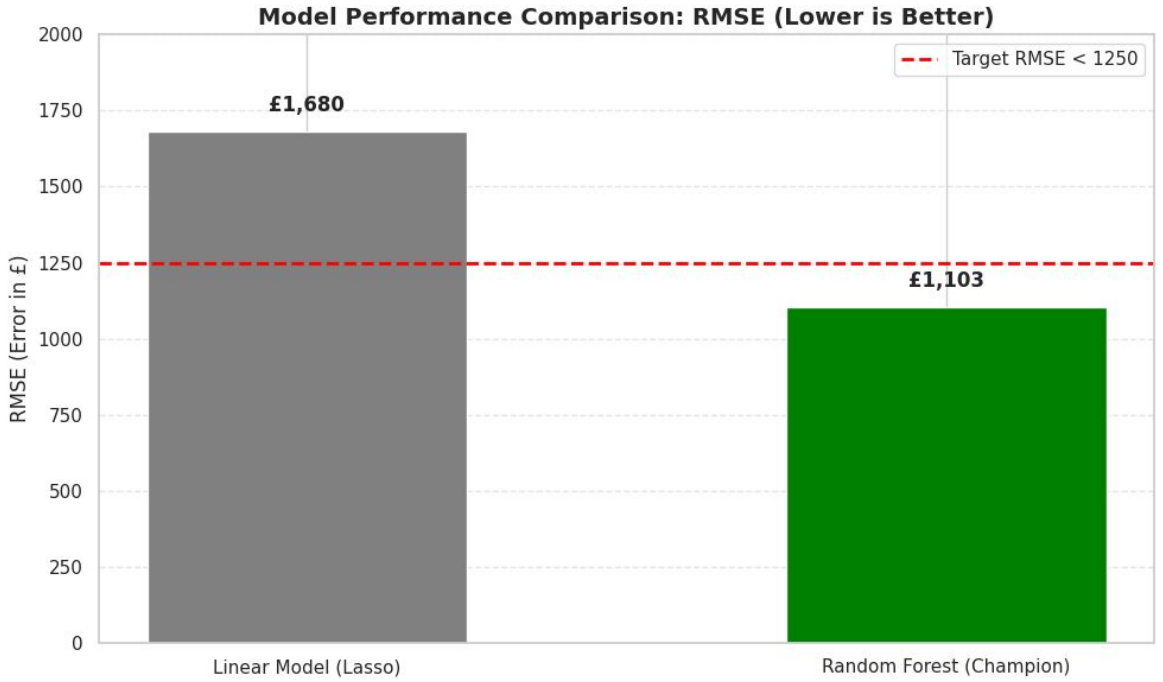
7.นำโมเดลที่ผ่านการเทรนและปรับพารามิเตอร์แล้วไปประเมินประสิทธิภาพบน Test set เพื่อวัดความสามารถของโมเดล

```python
# --- 5.3 Final Evaluation (เปรียบเทียบผลลัพธ์) ---

# 1. ดึงผลจาก Linear ที่ดีที่สุด
best_linear_model = grid_lasso.best_estimator_
y_pred_linear = best_linear_model.predict(X_test)

# 2. ดึงผลจาก Random Forest ที่ดีที่สุด
best_rf_model = grid_rf.best_estimator_
y_pred_rf = best_rf_model.predict(X_test)

# 3. คำนวณค่า Error จริง (Test Set)
def get_final_metrics(y_true, y_pred):
    rmse = np.sqrt(mean_squared_error(y_true, y_pred))
    r2 = r2_score(y_true, y_pred)
    mae = mean_absolute_error(y_true, y_pred)
    return rmse, r2, mae

rmse_lin, r2_lin, mae_lin = get_final_metrics(y_test, y_pred_linear)
rmse_rf, r2_rf, mae_rf = get_final_metrics(y_test, y_pred_rf)

# 4. สร้างตารางเปรียบเทียบ
comparison_df = pd.DataFrame({
    'Metric': ['RMSE (Error)', 'R2 Score (Accuracy)', 'MAE (Avg Error)'],
    'Linear Model (Lasso)': [rmse_lin, r2_lin, mae_lin],
    'Random Forest (Champion)': [rmse_rf, r2_rf, mae_rf]
})

# จัด Format ตัวเลข
pd.options.display.float_format = '{:,.2f}'.format
display(comparison_df)

# 5. ตัดสินผล
target = 1250
print("-" * 50)
print(f"(Target RMSE): ต่ำกว่า £{target}")
print(f"Random Forest (Test Set):   £{rmse_rf:.2f}")
```

Build Machine Learning Model: Summary

	Metric	Linear Model (Lasso)	Random Forest (Champion)
0	RMSE (Error)	1,680.24	1,102.96
1	R2 Score (Accuracy)	0.93	0.97
2	MAE (Avg Error)	1,102.26	749.55

(Target RMSE): ต่ำกว่า £1250
Random Forest (Test Set): £1102.96



สรุปผล

มีการทดสอบโมเดลทั้งหมด 2 แบบ ดังนี้:

- Linear Model (Lasso) **RMSE ~ £1,768**
  ไม่สามารถจับความสัมพันธ์ที่ซับซ้อนของราคารถได้ แต่ยังคงให้ค่า R2 ที่สูง

- **Random Forest RMSE = £1,102.96**
  เรียนรู้รูปแบบราคาได้ดี โดยเฉพาะความแตกต่างของแต่ละ Segment

ผลลัพธ์: จากการทดลอง พบว่าโมเดล
Random Forest ให้ผลดีที่สุด

- **Accuracy: 96% (R2)**
- **RMSE: 1,102.963**
- **ผ่านเกณฑ์ที่ตั้งไว้ (<1,250)**

Build Machine Learning Model: Summary

Summary

- โปรเจกต์ **Toyota Used Car Price Prediction** จัดทำขึ้นเพื่อแก้ปัญหาการตั้งราคารถมือสองผิดพลาด ซึ่งส่งผลให้ยอดขายลดลงประมาณ **18%**
- **เป้าหมาย:** สร้างโมเดลที่ทำนายราคาได้อย่างแม่นยำ โดยตั้งเกณฑ์ **RMSE 1,250**
- **ผลลัพธ์:** จากการทดลอง พบว่าโมเดล **Random Forest** ให้ผลดีที่สุด
    - **Accuracy:** 96% (R^2)
    - **RMSE:** 1,102.96
    - **สรุป: ผ่านเกณฑ์ที่ตั้งไว้**

Key Insights

ปัจจัย 3 อันดับแรกที่ส่งผลต่อราคามากที่สุด (Feature Importance):

1. **Engine Size (ขนาดเครื่องยนต์):** แยกแยะระหว่างรถราคาประหยัดและรถหรูได้อย่างชัดเจน
2. **Year (ปี):** รถปีใหม่ราคาสูงกว่า แต่ราคาไม่ได้ลดลงในลักษณะเชิงเส้น (Non-linear)
3. **Transmission (ระบบเกียร์):** รถเกียร์ออโต้มีราคาสูงกว่าเกียร์ธรรมดา

Business Recommendations

นำไปใช้: นำโมเดลไปใช้หน้างานทันทีเพื่อกู้วิกฤตยอดขาย

การปฏิบัติงาน: ให้ความสำคัญกับ ขนาดเครื่องยนต์ ในการประเมินราคา (โดยเฉพาะเครื่องยนต์ที่มีขนาดใหญ่)

การบำรุงรักษาโมเดล: อัปเดตข้อมูลเข้าโมเดลทุกเดือนเพื่อให้ทันต่อสภาพตลาดที่เปลี่ยนไป

DATA PROJECT CANVAS

Designed by: Thanakorn (268), Jatuporn(255), Arphakon(310) / Date: 14-Dec-2025

Title: Toyota Used Car Price Prediction Project (SWU Motors)

1. Problem Statement/Background

- What do we know about?
- What problem are you trying to solve?
  - SWU Motors เป็นดีลเลอร์รถมือสองที่กำลังขยายตัวและจ้างพนักงานขายใหม่ (Junior Salespeople) จำนวนมาก
  - **ยอดขายลดลง 18% ในช่วงที่ผ่านมา**
  - **พนักงานขายใหม่ขาดประสบการณ์ในการตั้งราคา (Pricing) รถ Toyota มือสองที่รับเข้ามา ทำให้เกิดความผิดพลาดในการตั้งราคาขาย**
  - เรามี dataset (toyota.csv) ที่รวบรวมราคาขายจริงจากผู้ค้าปลีกรายอื่นในตลาด
- What is the business problem?
  - Revenue Loss: การตั้งราคาผิดพลาดทำให้เสียโอกาสในการขาย (ถ้าราคาสูงเกินไปรถขายไม่ออก) หรือเสียกำไร (ถ้าราคาต่ำเกินไป)
- Who are the stakeholders?
  - End-Users: พนักงานขายใหม่ (Junior Salespeople)
  - Management: ผู้จัดการฝ่ายขาย (ต้องการกู้ยอดขาย)
  - Data Science, Data Analytics

2. SMART Objectives/ Value Propositions

SMART Objectives:

- พัฒนาโมเดลทำนายราคาให้แล้ว**เสร็จภายใน 2 สัปดาห์ โดยมีค่าความคลาดเคลื่อน (RMSE) ต่ำกว่า 1,250 ปอนด์** เพื่อช่วยให้พนักงานใหม่**ตั้งราคาได้อย่างถูกต้องและกู้คืนยอดขายที่ลดลง 18%**

Value Proposition:

- สร้างเครื่องมือ **"Pricing Tool"** เพื่อช่วยให้พนักงานใหม่ตัดสินใจได้เร็วขึ้น แม่นยำขึ้น และกู้ยอดขายกลับคืนมา

3. Questions/Hypothesis

- Analytical Questions
  - ปัจจัยใดส่งผลต่อราคา Toyota มากที่สุด? (ปีจดทะเบียน?, เลขไมล์?, ขนาดเครื่องยนต์?)
  - ลักษณะการกระจายตัวของราคาในตลาดเป็นอย่างไร?
- Predictive Hypothesis
  - H1: เลขไมล์ (Mileage) น่าจะมีความสัมพันธ์เชิงลบกับราคา (ยิ่งวิ่งเยอะ ราคายิ่งตก)
  - H2: ขนาดเครื่องยนต์ (Engine Size) น่าจะมีความสัมพันธ์เชิงบวกกับราคา (เครื่องใหญ่ = ราคารถยิ่งสูง)

What can we predict?

- *ราคาขายต่อของรถยนต์ Toyota (£)*

4. Data Sources/Attributes

- Data sources & collection
  - ไฟล์ toyota.csv (รวบรวมข้อมูลราคาและคุณลักษณะรถจากผู้ค้าปลีกภายนอก)
- Data cleaning & preprocessing
  - ตรวจสอบ Missing Values และ Duplicate Rows
  - ตรวจสอบ Outliers (เช่น รถที่มีค่าภาษี tax เป็น 0 หรือค่า mpg ผิดปกติ
- Target variables & feature
  - **Target:** `price`.
  - **Features:** `model`, `year`, `transmission`, `mileage`, `fuelType`, `tax`, `mpg`, `engineSize`.
- Encoding & scaling strategies
  - **Encoding:** ใช้ One-Hot Encoding แปลงข้อมูลกลุ่ม (model, transmission, fuelType) เป็นตัวเลข
  - **Scaling:** ใช้ StandardScaler เพื่อปรับมาตรฐานข้อมูลตัวเลข (mileage, tax, mpg)

5. Analysis/Model Development

- Analytics Methodology
  - **Descriptive Analytics:** ใช้สถิติพื้นฐาน (Mean, Median, Skewness) และ Visualization เพื่อเข้าใจโครงสร้างตลาด
  - **Predictive Analytics**: ใช้ Supervised Learning (Regression) สร้างโมเดลทำนายราคาขาย
- Modeling Methodology
  - ***Pipeline:*** *ใช้ OneHotEncoder (แปลงกลุ่มข้อมูล) และ StandardScaler (ปรับสเกลตัวเลข) เพื่อเตรียมข้อมูลอัตโนมัติ*
  - ***Model Selection: เปรียบเทียบ 2 วิธีการ:***
  - ***Linear Models (Baseline)****: ทดสอบ Ridge, Lasso, Elastic Net (สมมติฐานเส้นตรง)*
  - ***Random Forest (Challenger)****: ใช้ Ensemble Learning เพื่อจับรูปแบบ Non-linear ที่ซับซ้อน*
  - ***Evaluation****: จูนพารามิเตอร์ด้วย GridSearchCV และวัดผลด้วย 5-Fold Cross-Validation เพื่อความแม่นยำสูงสุด*

6. Findings and Insights

- Business Insights
  - Segmentation: "ขนาดเครื่องยนต์" คือปัจจัยหลักในการแบ่งเกรดและราคารถ (ชัดเจนกว่าปีผลิต)
  - Market Structure: ตลาดเป็นแบบ "เบ้ขวา" (Right Skewed) รถส่วนใหญ่ราคาประหยัด แต่มีรถหรูจำนวนน้อยดึงค่าเฉลี่ยขึ้น
  - Value Retention: รถเครื่องยนต์ดีเซลรักษามูลค่าได้ดีกว่าและเสื่อมราคาช้ากว่ารถเบนซินเมื่อใช้งานหนัก
- Predictive Results
  - โมเดล **Linear Regression (Lasso)** ให้ค่า RMSE เท่ากับ £1,680 ซึ่งไม่ผ่านเกณฑ์ที่กำหนด สาเหตุหลักมาจากข้อจำกัดของโมเดลเชิงเส้นในการอธิบายความสัมพันธ์ที่ซับซ้อนของข้อมูล
  - Performance of **Random Forest Model**
    โมเดล Random Forest แสดงผลลัพธ์ที่ดีกว่าอย่างชัดเจน โดยมีค่า RMSE เท่ากับ £1,102.96 ซึ่งต่ำกว่าเกณฑ์เป้าหมาย (£1,250)
  - โมเดล Random Forest มีค่าความแม่นยำสูง (R^2 = 97%) และมีความคลาดเคลื่อนเฉลี่ย (MAE) ประมาณ ± £750 ต่อคัน แสดงให้เห็นถึงศักยภาพในการนำไปใช้งานจริงในการตั้งราคารถยนต์

7. Recommendation/Action and Impact

Action:

- Deploy Tool: ติดตั้งเครื่องมือ "Pricing Tool" ให้ทีมขายใช้เป็น "ราคากลาง" หน้างานทันที

Impact:

- Sales Recovery: แก้ปัญหาการตั้งราคาผิดพลาด ช่วยกู้คืนยอดขายที่ตกลง 15% ให้กลับมาเติบโต
- Profit Optimization: ลดโอกาสการ (ขายถูกไป) และลดความเสี่ยงในการ (รับซื้อแพงไป)
- Standardization: สร้างมาตรฐานราคากลาง (Fair Price) ให้กับองค์กร ลดการพึ่งพาประสบการณ์ส่วนบุคคล (Gut Feeling) ของพนักงาน

Modified from Bill Schmarzo's Machine Learning Canvas and Jasmine Vasandani's Data Science Workflow Canvas for SWU MSDS