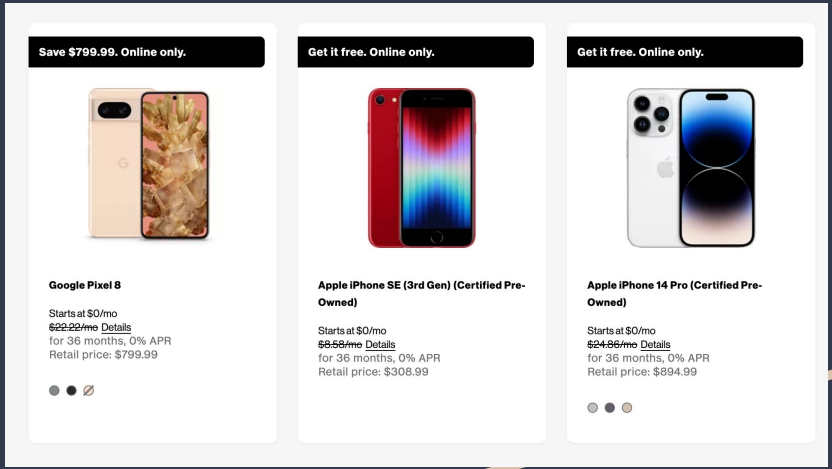
Mobile Phone Features and Prices

Group 4 Names: Baylor Dalsemer, Giovanni Rosa,
Sebastian Gonzalez Zurita, AJ Romaniello

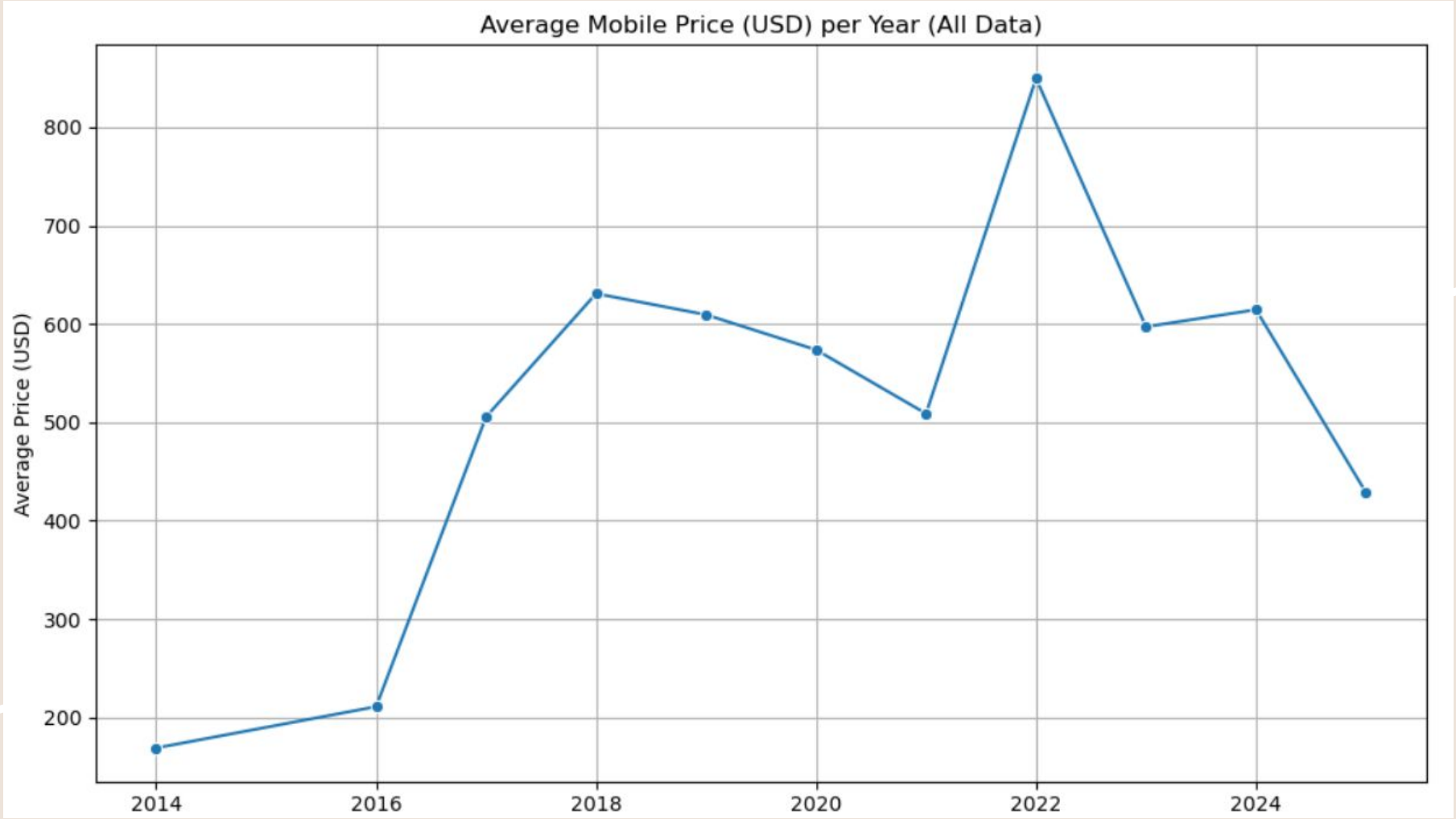
Group 4 Presentation

The Dataset



https://www.kaggle.com/datasets/abdulmalik1518/mobiles-dataset-2025/data

We used a dataset from kaggle to explore the relationship between phone pricing and phone hardware. This includes various data on RAM, camera specs, battery capacity, processor information, and screen size. Data for different countries currency and companies are provided as well, but we will be solely focusing on the prices in USD to avoid collinearity issues.


Average Mobile Price (USD) per Year (All Data)
Average Price (USD)
800
700
600
500
400
300
200
2014
2016
2018
2020
2022
2024

Project Goal

Predicting and identifying trends in mobile phone prices based on various hardware installations and software specifics.

Original Data

Company ...	Model Name	Mobile We...	RAM	Front Cam...	Back Cam...	Processor
Apple	iPhone 16 128GB	174g	6GB	12MP	48MP	A17 Bionic
Apple	iPhone 16 256GB	174g	6GB	12MP	48MP	A17 Bionic
Apple	iPhone 16 512GB	174g	6GB	12MP	48MP	A17 Bionic
Apple	iPhone 16 Plus 128GB	203g	6GB	12MP	48MP	A17 Bionic
Apple	iPhone 16 Plus 256GB	203g	6GB	12MP	48MP	A17 Bionic
Apple	iPhone 16 Plus 512GB	203g	6GB	12MP	48MP	A17 Bionic
Apple	iPhone 16 Pro 128GB	206g	6GB	12MP / 4K	50MP + 12MP	A17 Pro
Apple	iPhone 16 Pro 256GB	206g	8GB	12MP / 4K	50MP + 12MP	A17 Pro
Apple	iPhone 16 Pro 512GB	206g	8GB	12MP / 4K	50MP + 12MP	A17 Pro
Apple	iPhone 16 Pro Max 128GB	221g	6GB	12MP / 4K	48MP + 12MP	A17 Pro
Apple	iPhone 16 Pro Max 256GB	221g	8GB	12MP / 4K	48MP + 12MP	A17 Pro
Apple	iPhone 16 Pro	221g	8GB	12MP / 4K	48MP + 12MP	A17 Pro

Pre-Processing

- Cleaned all data to be numerical
- Removed null values
- Removed columns that couldn't be numerical
- Made 'Company Name' into dummy variable
- Dropped Columns
  - Prices except for USA
  - Processor Type (908 Unique values)
  - Model Name (217 Unique Values)

	Company	Weight (g)	RAM (GB)	Front Camera (MP)	Back Camera (MP)	Battery (mAh)	Screen Size (in)	Price USD	Launched Year
0	0	174.0	6.0	12.0	48.0	3600.0	6.1	799.0	2024
1	0	174.0	6.0	12.0	48.0	3600.0	6.1	849.0	2024
2	0	174.0	6.0	12.0	48.0	3600.0	6.1	899.0	2024
3	0	203.0	6.0	12.0	48.0	4200.0	6.7	899.0	2024
4	0	203.0	6.0	12.0	48.0	4200.0	6.7	949.0	2024
...	...	...	...	...	...	...	...	...	...
925	18	571.0	8.0	8.0	8.0	10000.0	12.1	280.0	2024
926	18	571.0	8.0	8.0	8.0	10000.0	12.1	300.0	2024
927	1	239.0	12.0	10.0	50.0	4400.0	7.6	1899.0	2024
928	1	239.0	12.0	10.0	50.0	4400.0	7.6	1719.0	2024
929	1	239.0	12.0	10.0	50.0	4400.0	7.6	2259.0	2024

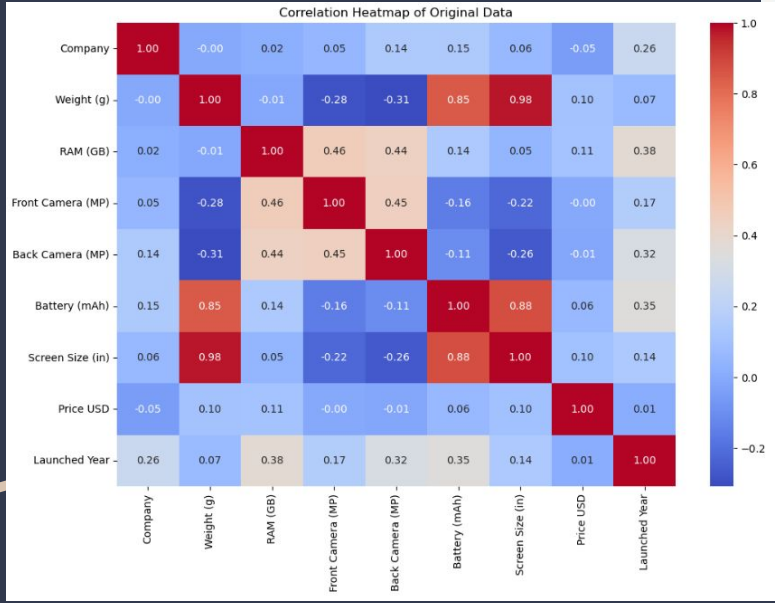
930 rows × 9 columns

```
Company Code Mapping:
0: Apple
1: Samsung
2: OnePlus
3: Vivo
4: iQOO
5: Oppo
6: Realme
7: Xiaomi
8: Lenovo
9: Motorola
10: Huawei
11: Nokia
12: Sony
13: Google
14: Tecno
15: Infinix
16: Honor
17: POCO
18: Poco
```

Descriptive Statistics

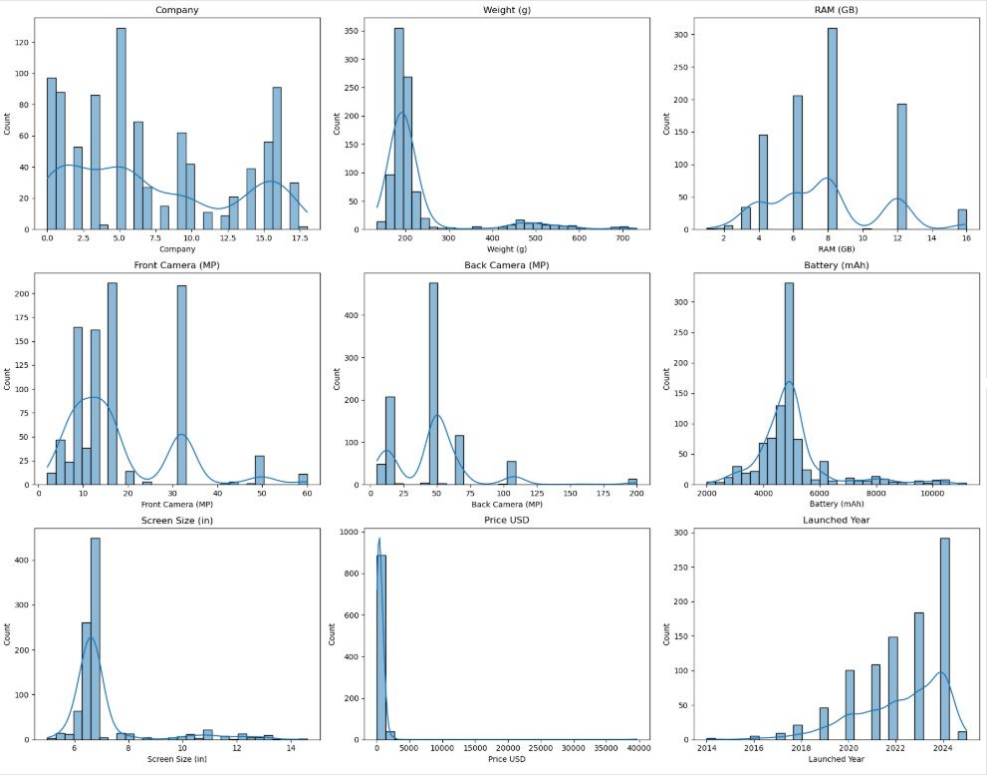
- Looked at the dataframe
  - Check data types and null values
- Look at descriptive statistics
- Heatmap for correlation

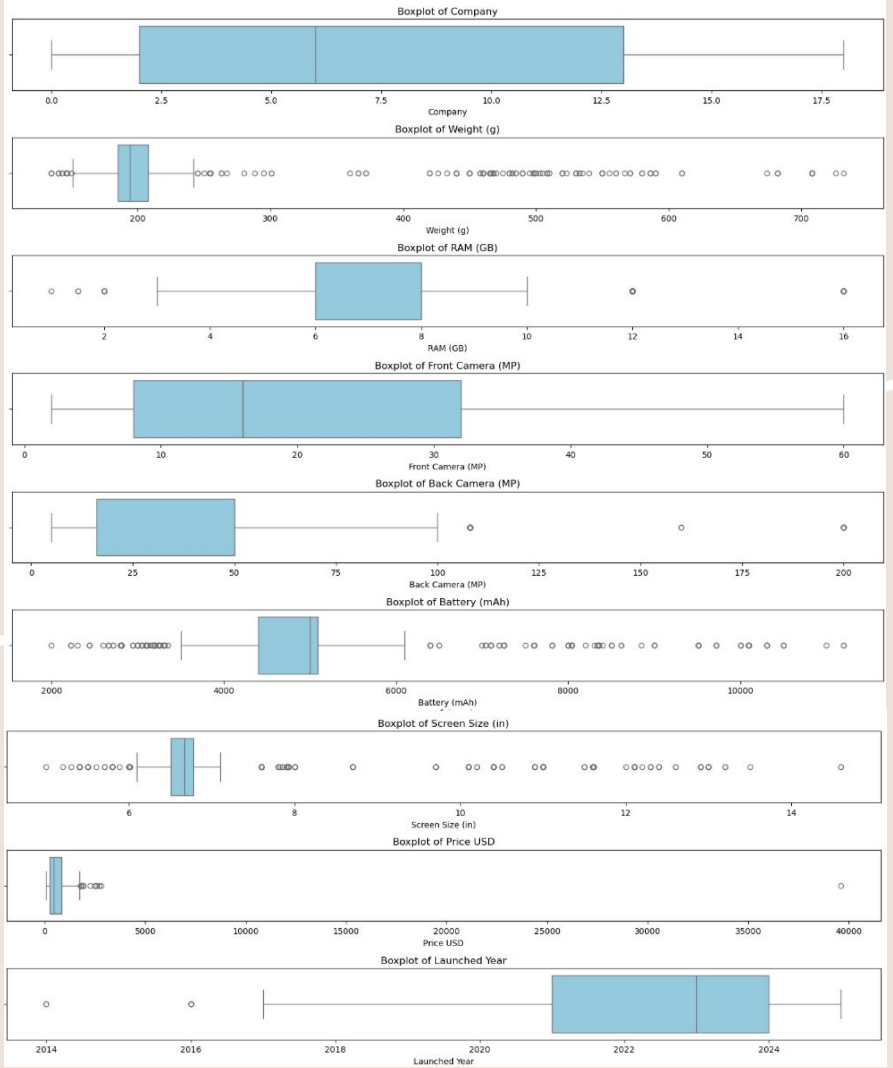
	Company	Weight (g)	RAM (GB)	Front Camera (MP)	Back Camera (MP)	Battery (mAh)	Screen Size (in)	Price USD	Launched Year
count	930.000000	930.000000	930.000000	930.000000	930.000000	930.000000	930.000000	930.000000	930.000000
mean	7.204301	228.267097	7.784946	18.163011	46.764301	5026.163441	7.083796	625.515763	2022.193548
std	5.596899	105.432503	3.179673	11.986228	31.069901	1355.548264	1.533690	1347.561211	1.862080
min	0.000000	135.000000	1.000000	2.000000	5.000000	2000.000000	5.000000	79.000000	2014.000000
25%	2.000000	185.000000	6.000000	8.000000	16.000000	4402.500000	6.500000	250.000000	2021.000000
50%	6.000000	194.000000	8.000000	16.000000	50.000000	5000.000000	6.670000	449.000000	2023.000000
75%	13.000000	208.000000	8.000000	32.000000	50.000000	5091.250000	6.780000	849.000000	2024.000000
max	18.000000	732.000000	16.000000	60.000000	200.000000	11200.000000	14.600000	39622.000000	2025.000000

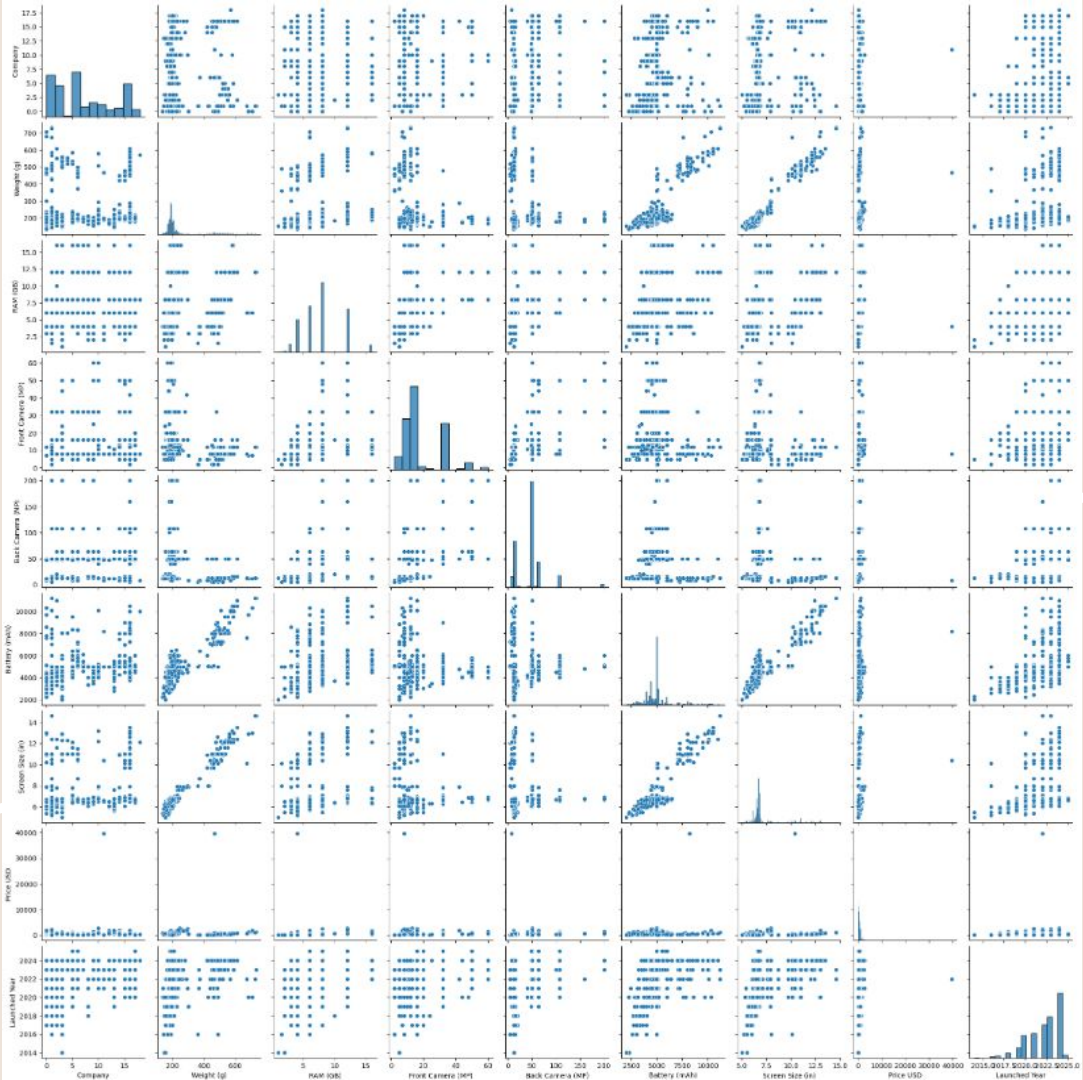



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 930 entries, 0 to 929
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Company             930 non-null    int64
 1   Weight (g)          930 non-null    float64
 2   RAM (GB)            930 non-null    float64
 3   Front Camera (MP)   930 non-null    float64
 4   Back Camera (MP)    930 non-null    float64
 5   Battery (mAh)       930 non-null    float64
 6   Screen Size (in)    930 non-null    float64
 7   Price USD           930 non-null    float64
 8   Launched Year       930 non-null    int64
dtypes: float64(7), int64(2)
memory usage: 65.5 KB
```

EDA


Company
Weight (g)
RAM (GB)
Front Camera (MP)
Back Camera (MP)
Battery (mAh)
Screen Size (in)
Price USD
Launched Year
Count


Boxplot of Company
Company
Boxplot of Weight (g)
Weight (g)
Boxplot of RAM (GB)
RAM (GB)
Boxplot of Front Camera (MP)
Front Camera (MP)
Boxplot of Back Camera (MP)
Back Camera (MP)
Boxplot of Battery (mAh)
Battery (mAh)
Boxplot of Screen Size (in)
Screen Size (in)
Boxplot of Price USD
Price USD
Boxplot of Launched Year
Launched Year


Company
Weight (g)
RAM (GB)
Front Camera (MP)
Back Camera (MP)
Battery (mAh)
Screen Size (in)
Price USD
Launched Year

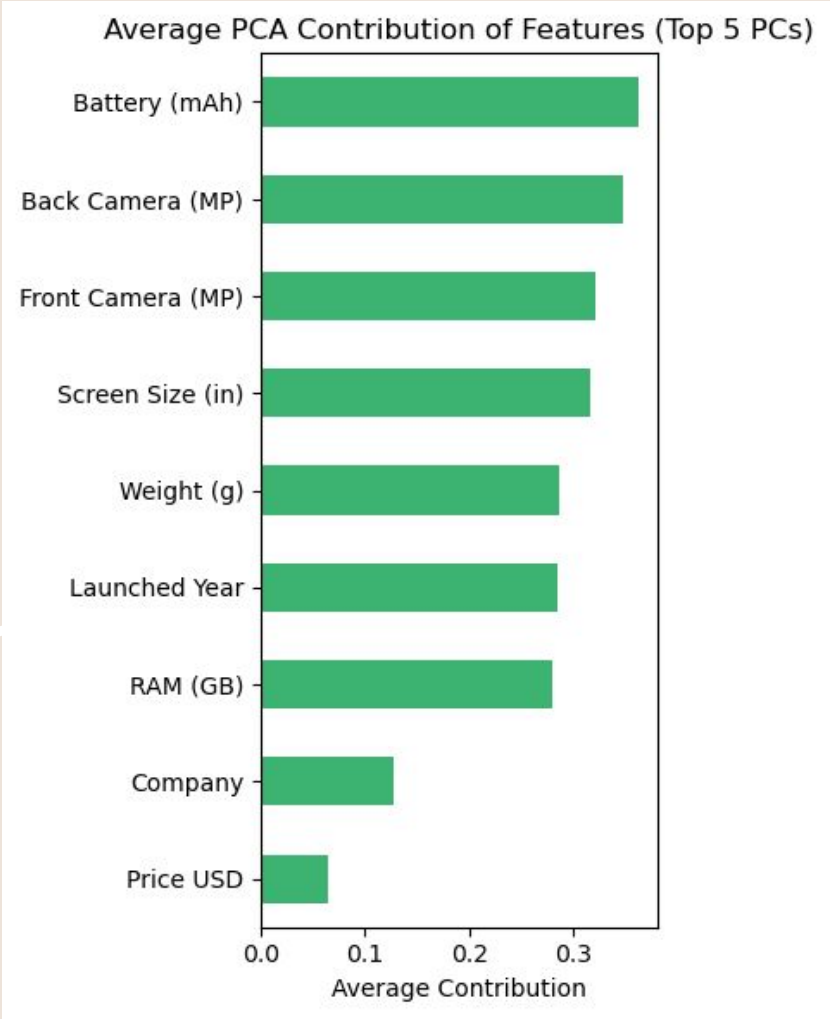
Principal Component Analysis (PCA)

Battery (mAh)	0.362566
Back Camera (MP)	0.346793
Front Camera (MP)	0.321721
Screen Size (in)	0.315650
Weight (g)	0.285786
Launched Year	0.284407
RAM (GB)	0.280318
Company	0.127565
Price USD	0.064163

dtype: float64

- Used to explore the structure of our data and look into dimension reduction.
- Top Variables contribute the most to the principal components when running PCA (Variance).
- Didn’t want to get rid of any predictor variables during this since PCA contribution doesn’t mean it’s a good predictor and vice versa.

PCA Visualized



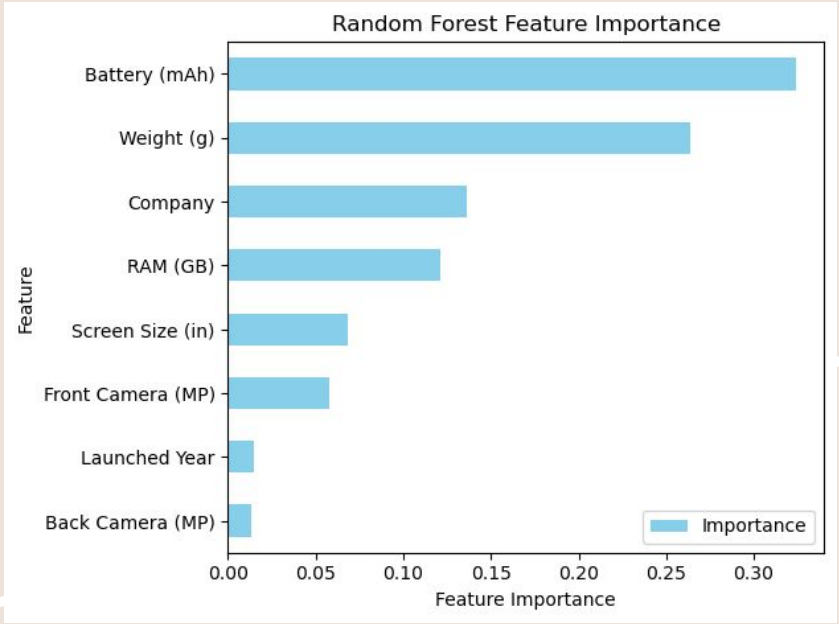
Random Forest

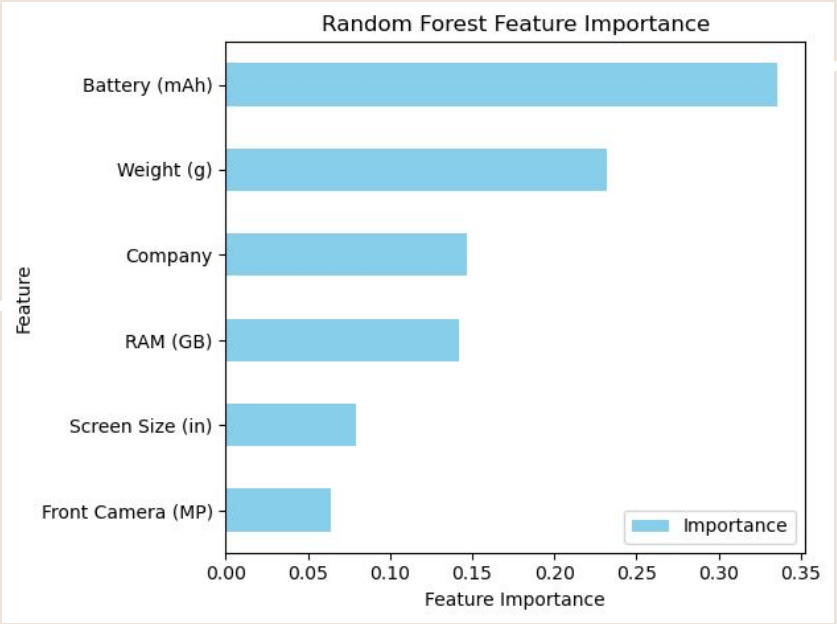
	Feature	Importance
5	Battery (mAh)	0.323707
1	Weight (g)	0.264075
0	Company	0.136600
2	RAM (GB)	0.121234
6	Screen Size (in)	0.068350
3	Front Camera (MP)	0.058138
7	Launched Year	0.014950
4	Back Camera (MP)	0.012945

	Feature	Importance
4	Battery (mAh)	0.335676
1	Weight (g)	0.231953
0	Company	0.147104
2	RAM (GB)	0.141974
5	Screen Size (in)	0.079510
3	Front Camera (MP)	0.063782

- We then ran a random forest regressor to find the feature importance (n_estimators = 100).
- Ended up dropping “Launched Year” and “Back Camera”
- Could say bottom two predictors aren’t relevant but don’t want to cut the amount of variables we have too much

Random Forest Visualized

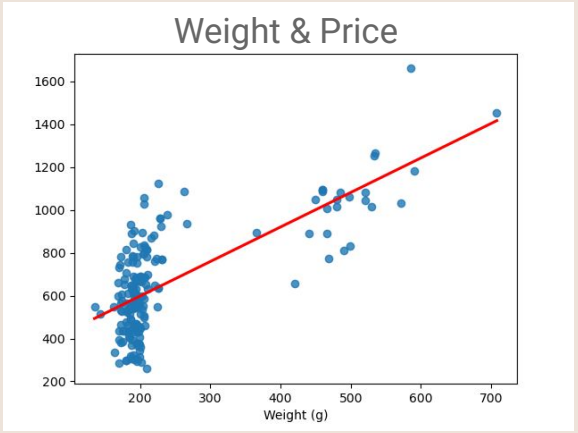


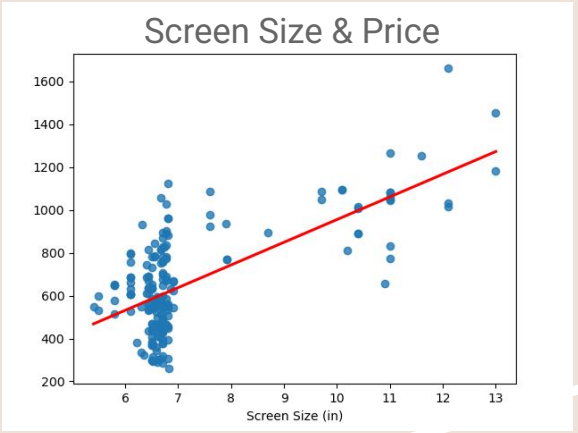


Simple Linear Regression Model

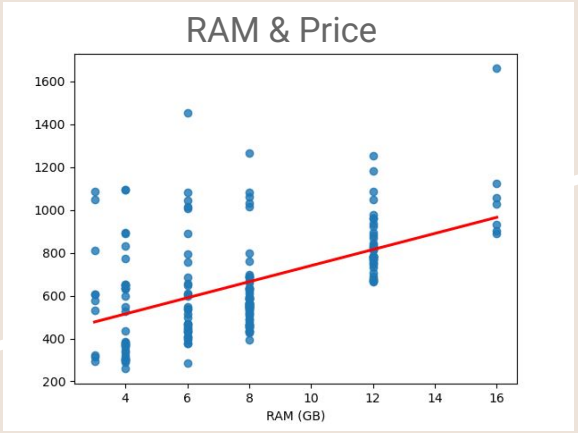
```
Mean Absolute Error: 248.59
Mean Squared Error: 97593.79
R-squared: 0.32
```

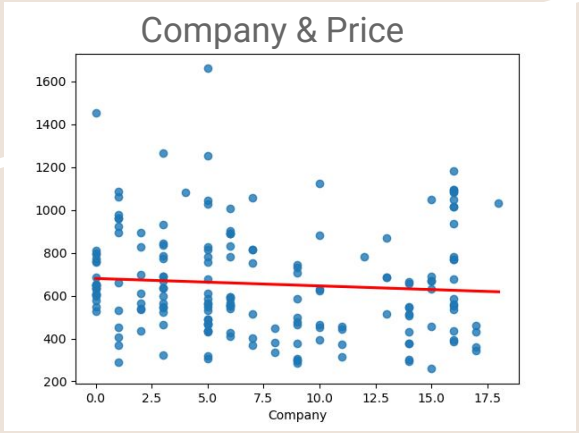
- We then ran Linear Regression in order to see the relationship Pricing had with the remaining variables.
- Most ended up having a positive relationship with Company and Front Camera being the only negative relationships.

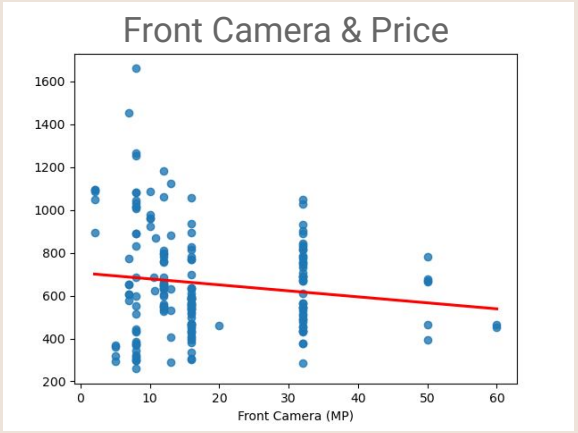


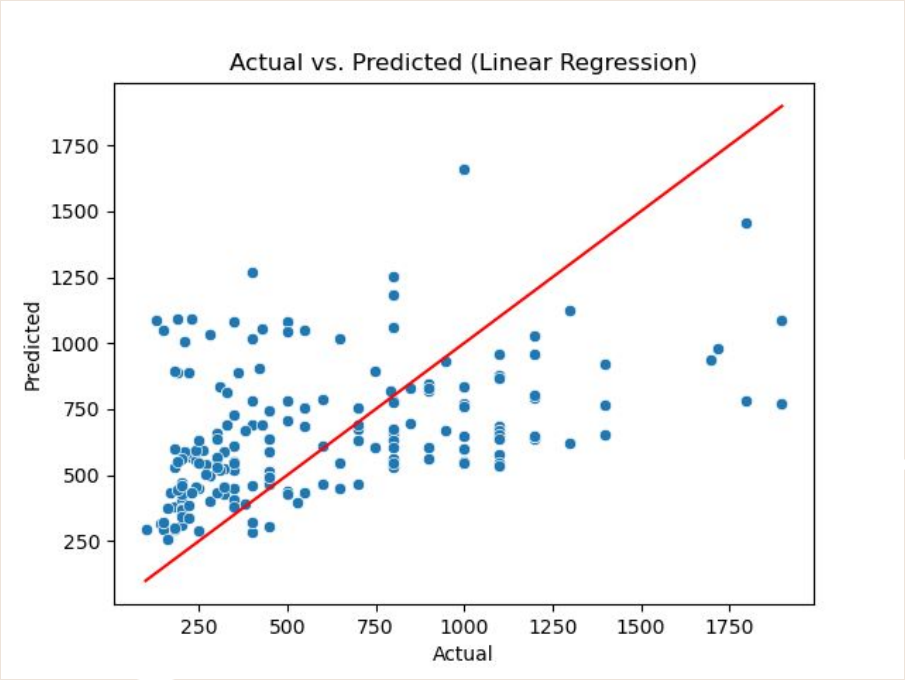


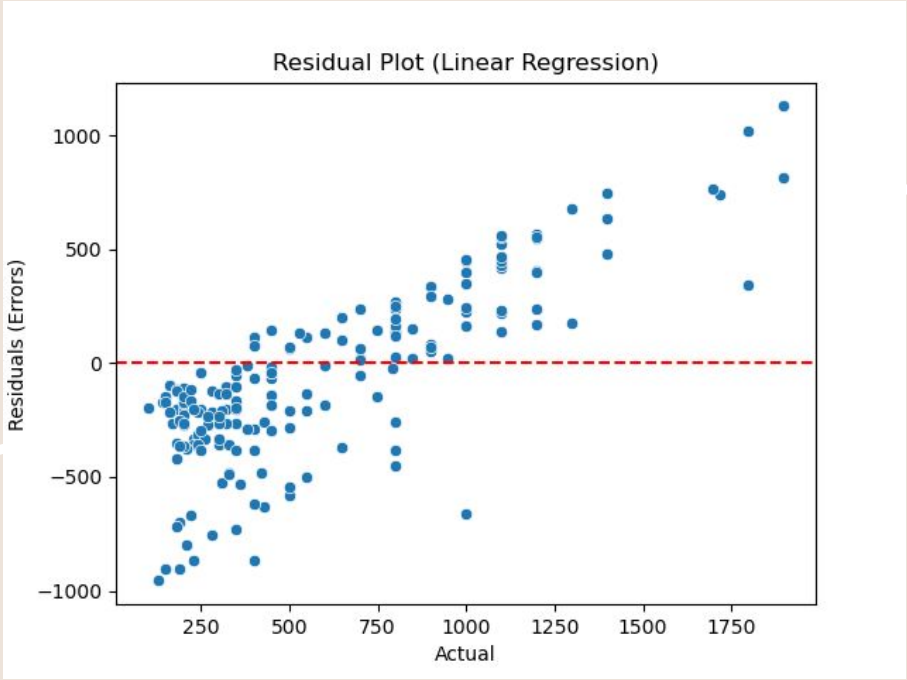
Linear Regression Visualized







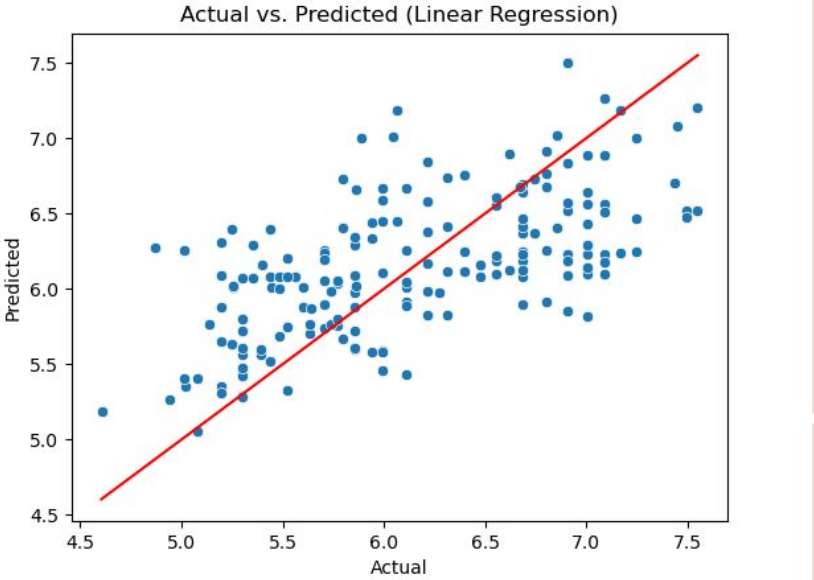

Actual vs. Predicted (Linear Regression)
Predicted
Actual
250
500
750
1000
1250
1500
1750

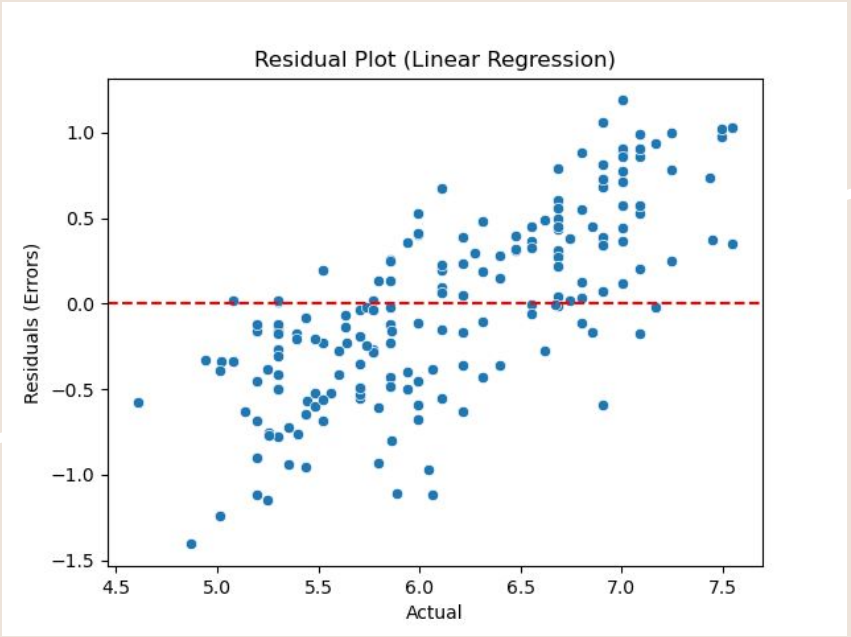

Residual Plot (Linear Regression)
Residuals (Errors)
Actual
1000
500
0
−500
−1000
250
500
750
1000
1250
1500
1750

2nd Linear Regression Model

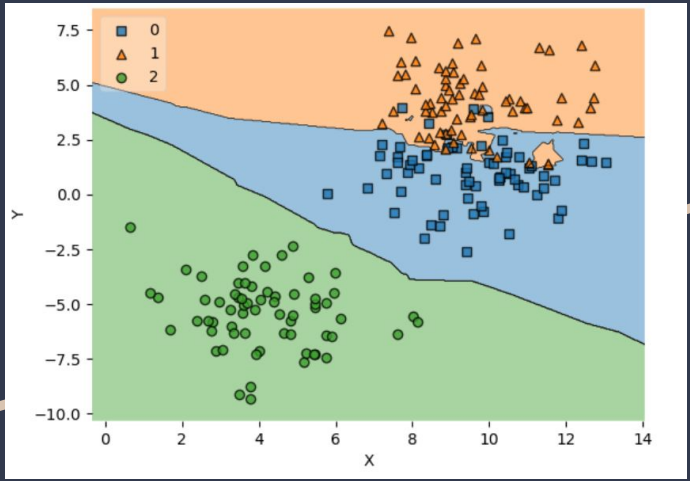
```
Mean Absolute Error: 0.40
Mean Squared Error: 0.23
R-squared: 0.49
```

- Model normalizes X values and uses a transformation on y


Actual vs. Predicted (Linear Regression)
Predicted
Actual
4.5
5.0
5.5
6.0
6.5
7.0
7.5


Residual Plot (Linear Regression)
Residuals (Errors)
Actual
1.0
0.5
0.0
−0.5
−1.0
−1.5
4.5
5.0
5.5
6.0
6.5
7.0
7.5

KNN Regression



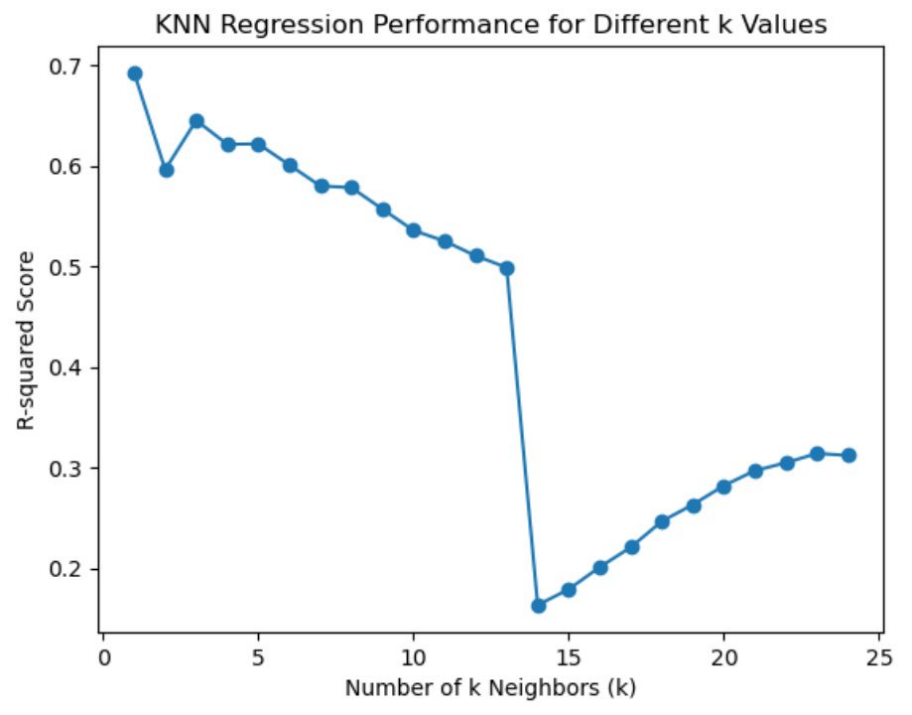
(ex. KNN classification)

- Similarity between points measured Euclidean distance.
- Evaluate regression models with R-squared.
- PCA removed features for KNN efficiency:

  Predict: Price USD

  With: Company, Weight (g), RAM (GB), Front Camera (MP), Battery (mAh), Screen Size (in).

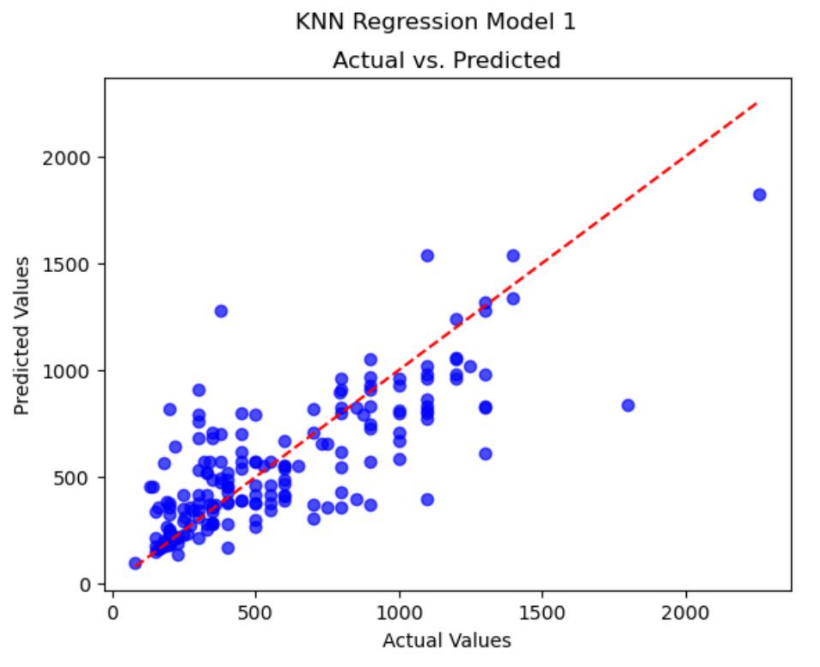
Testing different k values

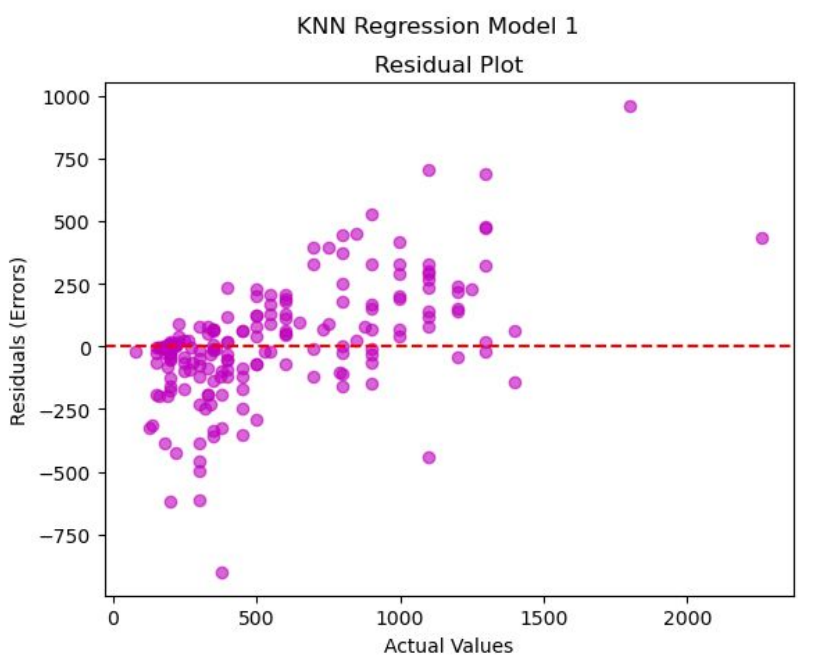


- k = 5

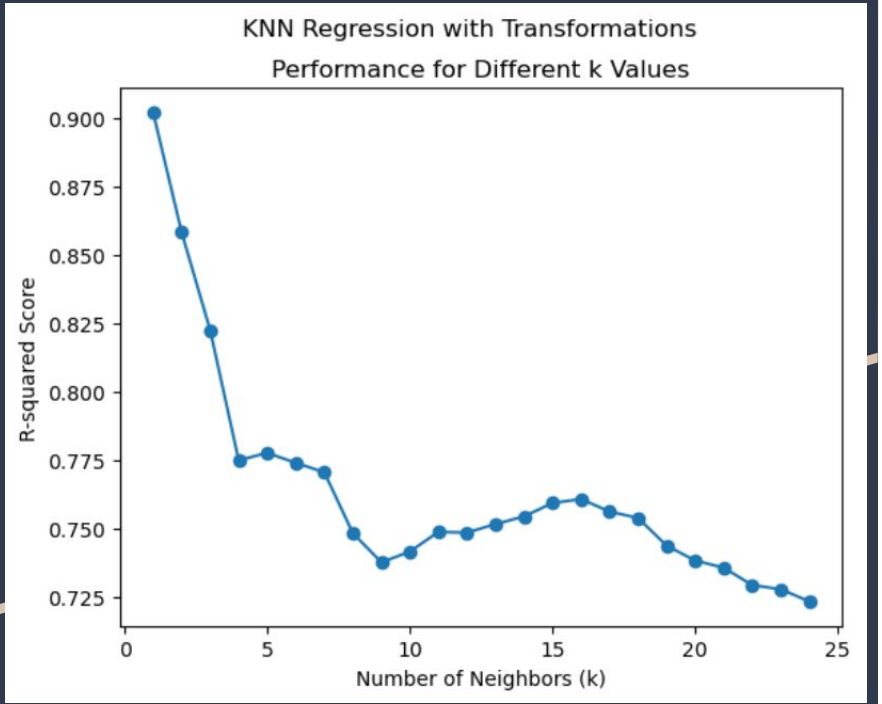
1st Simple KNN Model

```
Mean Absolute Error (MAE): 159.69
Mean Squared Error (MSE): 54228.27
R-squared Score: 0.62
```





Transformations & Scaling the Data



- KNN uses distance metrics.

- Standardize X:

  $z = (x - u) / s$

- Transform y: ln(y)

  stabilize variance

Tuning for the best Parameters

GridSearchCV

5-fold cross-validation

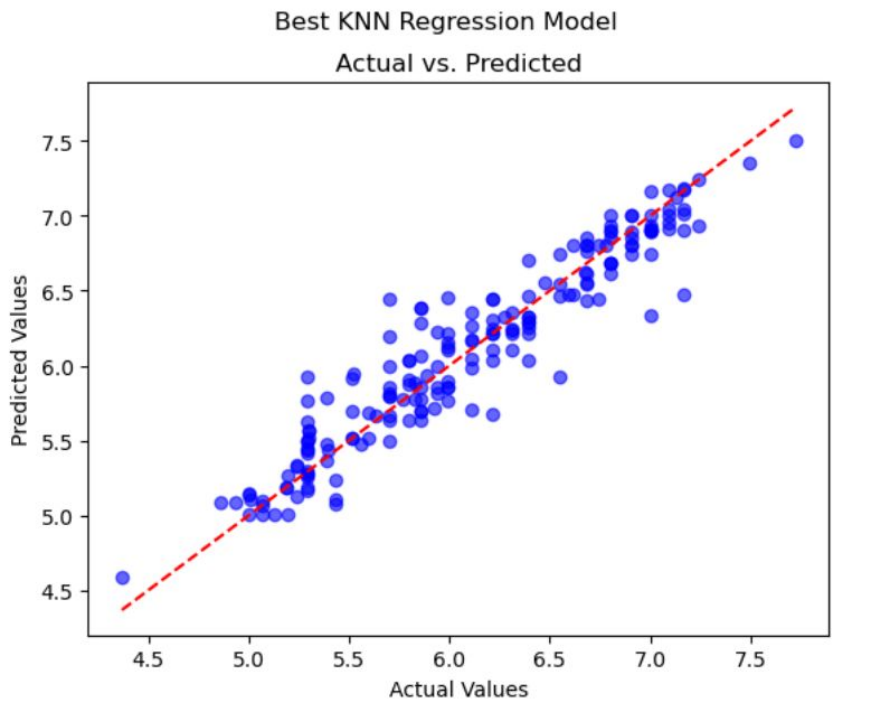
- n neighbors: (1,50)
- Weights: uniform or distance
- Metrics: manhattan or euclidean

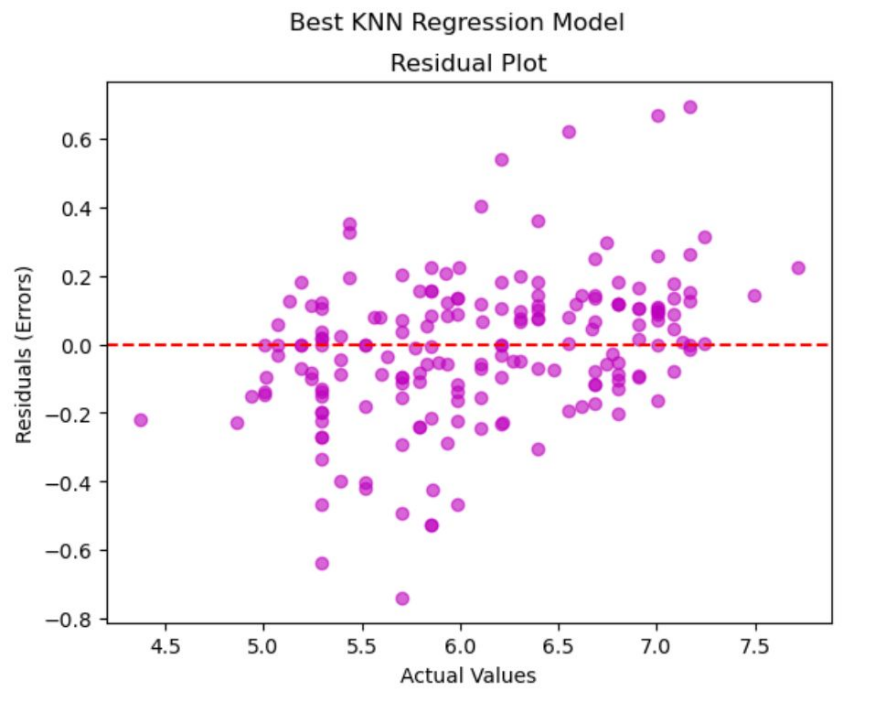
```
Best Parameters: {'metric': 'manhattan', 'n_neighbors': 8, 'weights': 'distance'}
```

Best KNN Model

```
Mean Absolute Error (MAE): 0.16
Mean Squared Error (MSE): 0.04
R-squared Score: 0.90
```

```
Best Parameters: {'metric': 'manhattan', 'n_neighbors': 8, 'weights': 'distance'}
```





Questions?