# Linear Regression

## Importing the relevant libraries

In [2]:
```
# we will need the following libraries and modules for this project
import numpy as np
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
import seaborn as sns
sns.set()
```

## Loading the raw data

In [3]:
```
# Importing the dataset from a CSV file.
# This dataset contains vehicle data such as brand, price, mileage, and engine type.
# I ensured the file path is correct and the dataset loads properly.

# Loading the dataset into a pandas DataFrame.
raw_data = pd.read_csv('1.04. Real-life example.csv')

# Displaying the first five rows of the dataset to understand its structure and previe
# This step is essential for identifying the columns, data types, and potential incons
raw_data.head()
```

Out[3]:

| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Year | Model |
|---|---|---|---|---|---|---|---|---|---|
| **0** | BMW | 4200.0 | sedan | 277 | 2.0 | Petrol | yes | 1991 | 320 |
| **1** | Mercedes-Benz | 7900.0 | van | 427 | 2.9 | Diesel | yes | 1999 | Sprinter 212 |
| **2** | Mercedes-Benz | 13300.0 | sedan | 358 | 5.0 | Gas | yes | 2003 | S 500 |
| **3** | Audi | 23000.0 | crossover | 240 | 4.2 | Petrol | yes | 2007 | Q7 |
| **4** | Toyota | 18300.0 | crossover | 120 | 2.0 | Petrol | yes | 2011 | Rav 4 |

# Preprocessing

## Exploring the descriptive statistics of the variables

In [4]:
```
# Generating descriptive statistics for the dataset.
# This step helps in understanding the distribution and summary of both numerical and

# By default, the describe() method provides statistics for numerical columns.
# Using the argument `include='all'` ensures that categorical variables are also inclu
```

```
# Observations:
# - Numerical columns provide metrics like mean, min, max, and standard deviation.
# - Categorical columns provide metrics like the most frequent category and unique cou

# Exploring statistics for all variables in the dataset.
raw_data.describe(include='all')

# Note:
# Based on this step, I identified columns with missing values or inconsistent data fo
```

Out[4]:

| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Yea |
|---|---|---|---|---|---|---|---|---|
| **count** | 4345 | 4173.000000 | 4345 | 4345.000000 | 4195.000000 | 4345 | 4345 | 4345.00000 |
| **unique** | 7 | NaN | 6 | NaN | NaN | 4 | 2 | Na |
| **top** | Volkswagen | NaN | sedan | NaN | NaN | Diesel | yes | Na |
| **freq** | 936 | NaN | 1649 | NaN | NaN | 2019 | 3947 | Na |
| **mean** | NaN | 19418.746935 | NaN | 161.237284 | 2.790734 | NaN | NaN | 2006.55005 |
| **std** | NaN | 25584.242620 | NaN | 105.705797 | 5.066437 | NaN | NaN | 6.71909 |
| **min** | NaN | 600.000000 | NaN | 0.000000 | 0.600000 | NaN | NaN | 1969.00000 |
| **25%** | NaN | 6999.000000 | NaN | 86.000000 | 1.800000 | NaN | NaN | 2003.00000 |
| **50%** | NaN | 11500.000000 | NaN | 155.000000 | 2.200000 | NaN | NaN | 2008.00000 |
| **75%** | NaN | 21700.000000 | NaN | 230.000000 | 3.000000 | NaN | NaN | 2012.00000 |
| **max** | NaN | 300000.000000 | NaN | 980.000000 | 99.990000 | NaN | NaN | 2016.00000 |

## Determining the variables of interest

In [5]:

```
# Dropping the 'Model' column as it is not relevant for this analysis
data = raw_data.drop(['Model'], axis=1)

# Reevaluating the dataset after dropping the column
data.describe(include='all')
```

Out[5]:

| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Yea |
|---|---|---|---|---|---|---|---|---|
| **count** | 4345 | 4173.000000 | 4345 | 4345.000000 | 4195.000000 | 4345 | 4345 | 4345.00000 |
| **unique** | 7 | NaN | 6 | NaN | NaN | 4 | 2 | Na |
| **top** | Volkswagen | NaN | sedan | NaN | NaN | Diesel | yes | Na |
| **freq** | 936 | NaN | 1649 | NaN | NaN | 2019 | 3947 | Na |
| **mean** | NaN | 19418.746935 | NaN | 161.237284 | 2.790734 | NaN | NaN | 2006.55005 |
| **std** | NaN | 25584.242620 | NaN | 105.705797 | 5.066437 | NaN | NaN | 6.71909 |
| **min** | NaN | 600.000000 | NaN | 0.000000 | 0.600000 | NaN | NaN | 1969.00000 |
| **25%** | NaN | 6999.000000 | NaN | 86.000000 | 1.800000 | NaN | NaN | 2003.00000 |
| **50%** | NaN | 11500.000000 | NaN | 155.000000 | 2.200000 | NaN | NaN | 2008.00000 |
| **75%** | NaN | 21700.000000 | NaN | 230.000000 | 3.000000 | NaN | NaN | 2012.00000 |
| **max** | NaN | 300000.000000 | NaN | 980.000000 | 99.990000 | NaN | NaN | 2016.00000 |

## Dealing with missing values

In [6]:

```
# data.isnull() # shows a df with the information whether a data point is null
# Since True = the data point is missing, while False = the data point is not missing,
# This will give us the total number of missing values feature-wise
data.isnull().sum()
```

Out[6]:

```
Brand             0
Price           172
Body              0
Mileage           0
EngineV         150
Engine Type       0
Registration      0
Year              0
dtype: int64
```

In [7]:

```
# Let's simply drop all missing values
# This is not always recommended, however, when we remove less than 5% of the data, it
data_no_mv = data.dropna(axis=0)
```

In [8]:

```
# Let's check the descriptives without the missing values
data_no_mv.describe(include='all')
```

Out[8]:

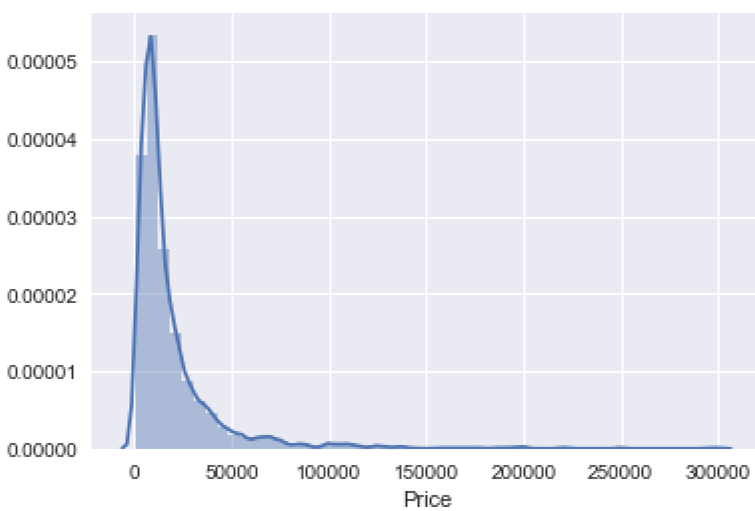
| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Yea |
|---|---|---|---|---|---|---|---|---|
| count | 4025 | 4025.000000 | 4025 | 4025.000000 | 4025.000000 | 4025 | 4025 | 4025.00000 |
| unique | 7 | NaN | 6 | NaN | NaN | 4 | 2 | NaN |
| top | Volkswagen | NaN | sedan | NaN | NaN | Diesel | yes | NaN |
| freq | 880 | NaN | 1534 | NaN | NaN | 1861 | 3654 | NaN |
| mean | NaN | 19552.308065 | NaN | 163.572174 | 2.764586 | NaN | NaN | 2006.37962 |
| std | NaN | 25815.734988 | NaN | 103.394703 | 4.935941 | NaN | NaN | 6.69559 |
| min | NaN | 600.000000 | NaN | 0.000000 | 0.600000 | NaN | NaN | 1969.00000 |
| 25% | NaN | 6999.000000 | NaN | 90.000000 | 1.800000 | NaN | NaN | 2003.00000 |
| 50% | NaN | 11500.000000 | NaN | 158.000000 | 2.200000 | NaN | NaN | 2007.00000 |
| 75% | NaN | 21900.000000 | NaN | 230.000000 | 3.000000 | NaN | NaN | 2012.00000 |
| max | NaN | 300000.000000 | NaN | 980.000000 | 99.990000 | NaN | NaN | 2016.00000 |

## Exploring the PDFs

In [9]:

```
# A great step in the data exploration is to display the probability distribution func
# The PDF will show us how that variable is distributed
# This makes it very easy to spot anomalies, such as outliers
# The PDF is often the basis on which we decide whether we want to transform a feature
sns.distplot(data_no_mv['Price'])
```

Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0x2b897ab4940>




## Dealing with outliers

In [10]:

```
# Identifying and handling outliers in the 'Price' column
# Outliers in this dataset are primarily present in the higher price range, as observe
# Since this dataset represents used cars, excessively high prices (e.g., $300,000) ar
```

```
# Setting a threshold to remove the top 1% of 'Price' values to mitigate the impact of
q = data_no_mv['Price'].quantile(0.99)

# Creating a filtered dataset that includes only rows where 'Price' is below the 99th
data_1 = data_no_mv[data_no_mv['Price'] < q]

# Generating descriptive statistics to validate the changes after removing the top 1%
data_1.describe(include='all')
```

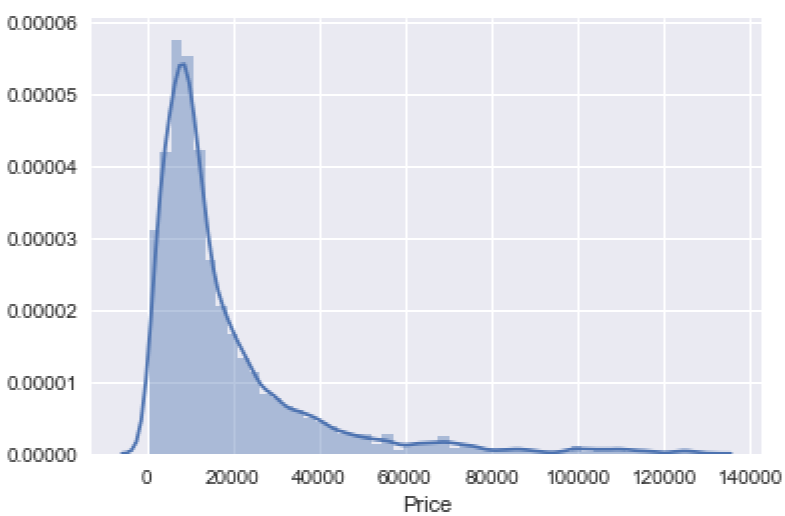
Out[10]:

| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Yea |
|---|---|---|---|---|---|---|---|---|
| **count** | 3984 | 3984.000000 | 3984 | 3984.000000 | 3984.000000 | 3984 | 3984 | 3984.00000 |
| **unique** | 7 | NaN | 6 | NaN | NaN | 4 | 2 | Na |
| **top** | Volkswagen | NaN | sedan | NaN | NaN | Diesel | yes | Na |
| **freq** | 880 | NaN | 1528 | NaN | NaN | 1853 | 3613 | Na |
| **mean** | NaN | 17837.117460 | NaN | 165.116466 | 2.743770 | NaN | NaN | 2006.29292 |
| **std** | NaN | 18976.268315 | NaN | 102.766126 | 4.956057 | NaN | NaN | 6.67274 |
| **min** | NaN | 600.000000 | NaN | 0.000000 | 0.600000 | NaN | NaN | 1969.00000 |
| **25%** | NaN | 6980.000000 | NaN | 93.000000 | 1.800000 | NaN | NaN | 2002.75000 |
| **50%** | NaN | 11400.000000 | NaN | 160.000000 | 2.200000 | NaN | NaN | 2007.00000 |
| **75%** | NaN | 21000.000000 | NaN | 230.000000 | 3.000000 | NaN | NaN | 2011.00000 |
| **max** | NaN | 129222.000000 | NaN | 980.000000 | 99.990000 | NaN | NaN | 2016.00000 |

In [11]:

```
# We can check the PDF once again to ensure that the result is still distributed in th
# however, there are much fewer outliers
sns.distplot(data_1['Price'])
```

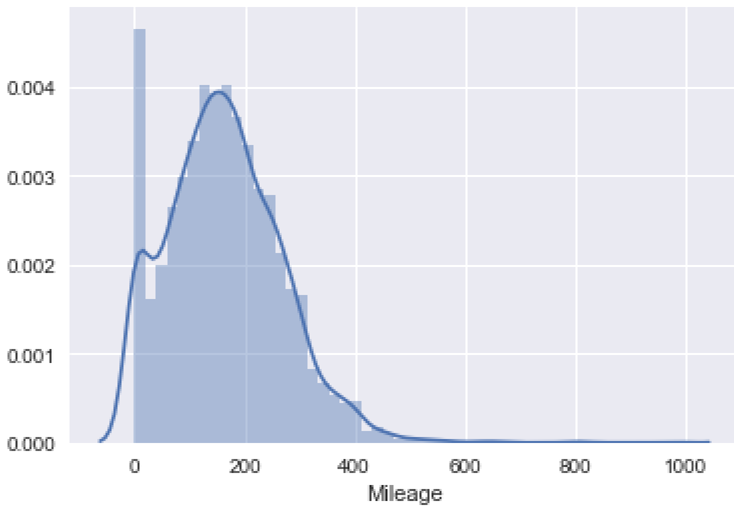
Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x2b897eacfd0>




In [12]:

```
# We can treat the other numerical variables in a similar way
sns.distplot(data_no_mv['Mileage'])
```

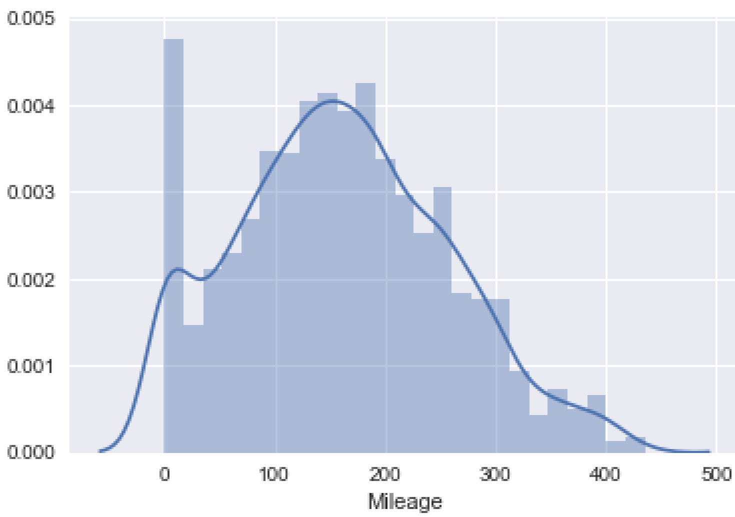
Out[12]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b897e36e80>
```




In [13]:
```
q = data_1['Mileage'].quantile(0.99)
data_2 = data_1[data_1['Mileage']<q]
```

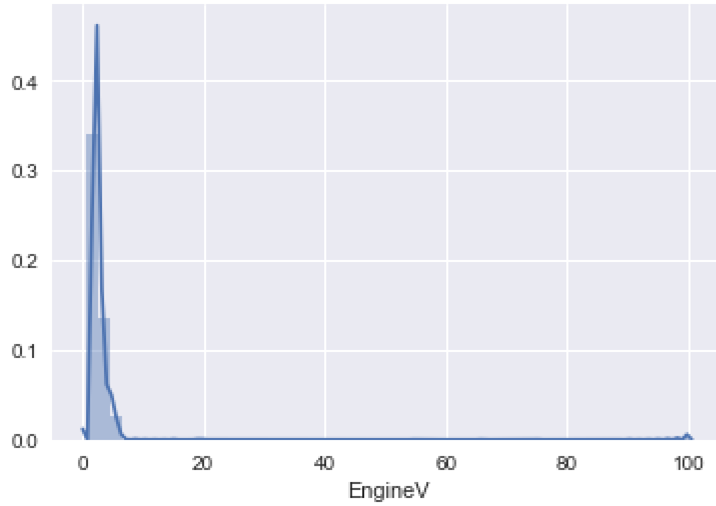
In [14]:
```
# The curve appears relatively normal, which indicates that the data is well-behaved.
# This visualization helps identify any significant deviations or anomalies in the mil
sns.distplot(data_2['Mileage'])
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b8980359b0>
```




In [15]:
```
# Analyzing the distribution of the 'EngineV' (Engine Volume) variable
# The histogram indicates some irregularities, with excessively high values.
# These values might not be realistic for engine volumes, such as entries with values
# This insight demonstrates the need for data cleaning and domain knowledge to spot er
sns.distplot(data_no_mv['EngineV'])
```

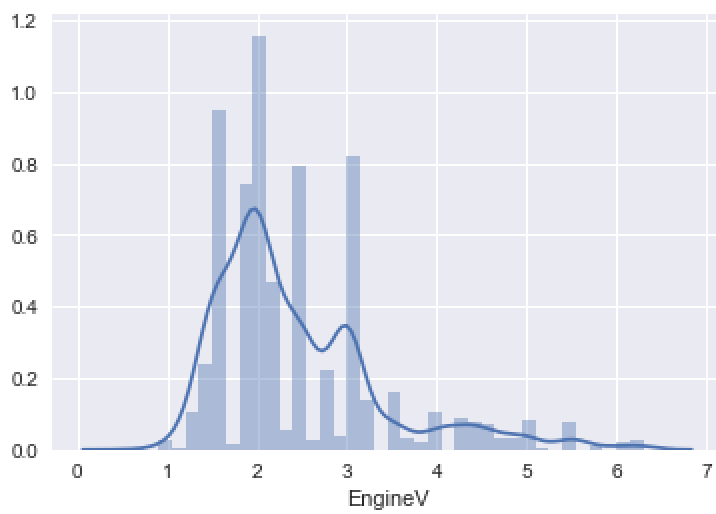
Out[15]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b897f681d0>
```




In [16]:
```
# Filtering out unrealistic 'EngineV' values
# Through domain knowledge, we know that typical car engine volumes are below 6.5 lite
# By removing these unrealistic values, we clean up the data for accurate analysis.
data_3 = data_2[data_2['EngineV'] < 6.5]
```

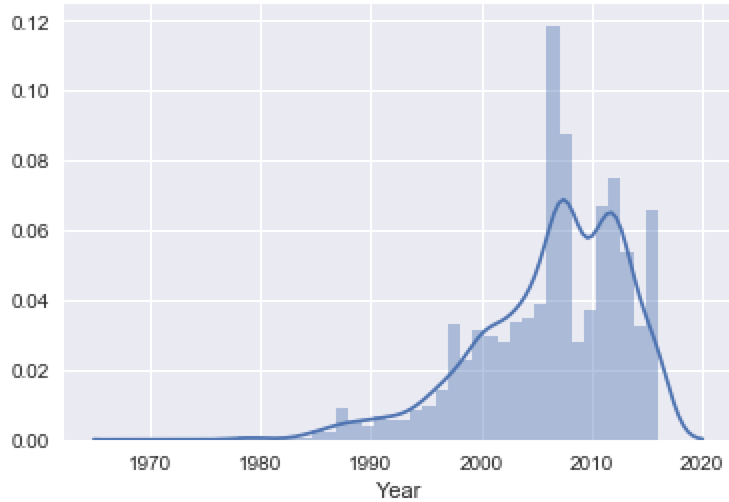
In [17]:
```
# Visualizing the filtered 'EngineV' variable
# This new distribution shows realistic values for engine volumes, with anomalies remo
# The cleaned data is now ready for further analysis or modeling.
sns.distplot(data_3['EngineV'])
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x2b8981a0b00>




In [18]:
```
# Finally, the situation with 'Year' is similar to 'Price' and 'Mileage'
# However, the outliers are on the low end
sns.distplot(data_no_mv['Year'])
```

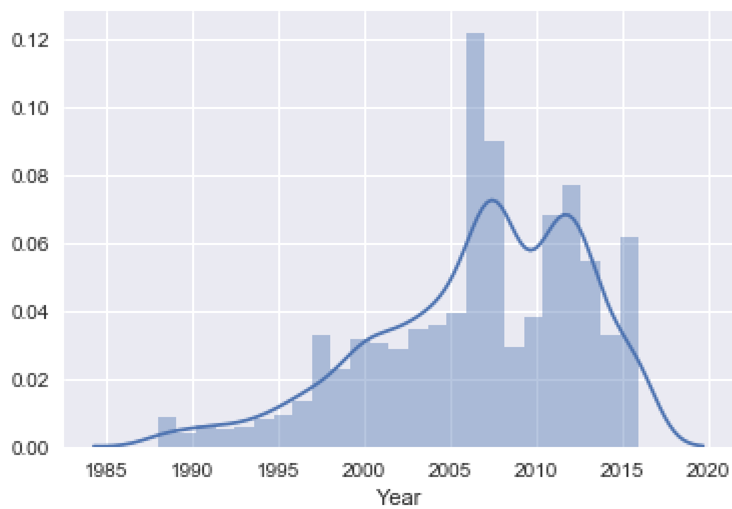
Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x2b89825e6a0>




In [19]:
```
# Removing the oldest 1% of the data based on the 'Year' variable
# The rationale is that vehicles from very early years might not be relevant for analy
# By focusing on the 99th percentile and above, we ensure a more balanced dataset.
q = data_3['Year'].quantile(0.01)
data_4 = data_3[data_3['Year'] > q]
```

In [20]:
```
# Here's the new result
sns.distplot(data_4['Year'])
```

Out[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b898296d30>
```




In [21]:
```
# Resetting the index of the cleaned dataset
# Explanation: After removing some rows during cleaning, the original index numbers ar
# For instance, if rows with indices 2 and 3 are removed, the index may skip numbers,
# By resetting the index, we ensure it starts from 0 and is sequential.
# The 'drop=True' argument ensures the old index is not retained as a new column.
data_cleaned = data_4.reset_index(drop=True)
```

In [22]:
```
# Let's see what's left
data_cleaned.describe(include='all')
```

Out[22]:

| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Yea |
|---|---|---|---|---|---|---|---|---|
| **count** | 3867 | 3867.000000 | 3867 | 3867.000000 | 3867.000000 | 3867 | 3867 | 3867.00000 |
| **unique** | 7 | NaN | 6 | NaN | NaN | 4 | 2 | Na |
| **top** | Volkswagen | NaN | sedan | NaN | NaN | Diesel | yes | Na |
| **freq** | 848 | NaN | 1467 | NaN | NaN | 1807 | 3505 | Na |
| **mean** | NaN | 18194.455679 | NaN | 160.542539 | 2.450440 | NaN | NaN | 2006.70985 |
| **std** | NaN | 19085.855165 | NaN | 95.633291 | 0.949366 | NaN | NaN | 6.10387 |
| **min** | NaN | 800.000000 | NaN | 0.000000 | 0.600000 | NaN | NaN | 1988.00000 |
| **25%** | NaN | 7200.000000 | NaN | 91.000000 | 1.800000 | NaN | NaN | 2003.00000 |
| **50%** | NaN | 11700.000000 | NaN | 157.000000 | 2.200000 | NaN | NaN | 2008.00000 |
| **75%** | NaN | 21700.000000 | NaN | 225.000000 | 3.000000 | NaN | NaN | 2012.00000 |
| **max** | NaN | 129222.000000 | NaN | 435.000000 | 6.300000 | NaN | NaN | 2016.00000 |

# Checking the OLS assumptions

In linear regression, the following assumptions must be met for the model to provide reliable and unbiased estimates:

Linearity:

The relationship between the dependent variable (Price) and independent variables (Year, Mileage, EngineV) must be linear.

Independence:

Observations must be independent of each other (no autocorrelation or dependency between data points). Homoscedasticity:

The variance of residuals (errors) must remain constant across all levels of the independent variables.

Normality:

The residuals should be approximately normally distributed.

No Multicollinearity:

Independent variables should not be highly correlated with each other.

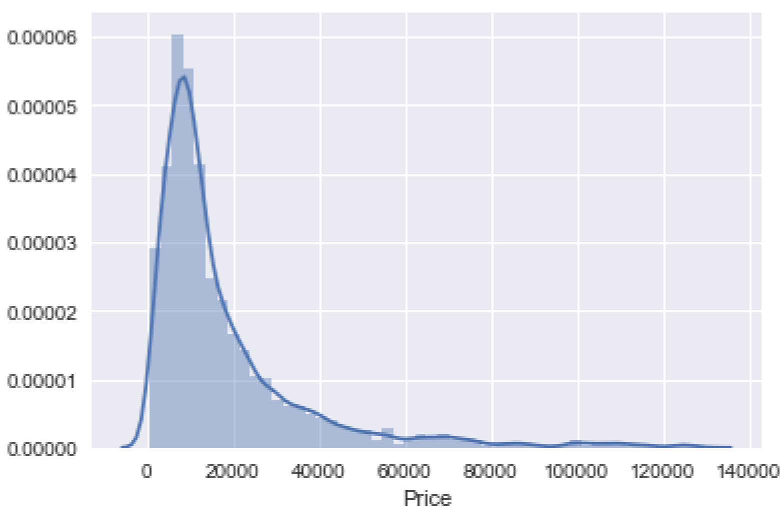
No Outliers:

Extreme values that can distort the relationship between variables should be removed or mitigated.

In [23]:
```
# Using Scatter Plot to analyze the relationship between 'Price' and other features
# This step helps identify any patterns, trends, or potential issues with the data
# Subplots are used to visualize multiple relationships side by side for comparison
f, (ax1, ax2, ax3) = plt.subplots(1, 3, sharey=True, figsize=(15, 3))  # sharey=True e
ax1.scatter(data_cleaned['Year'], data_cleaned['Price'])
ax1.set_title('Price and Year')  # Title indicates the relationship being analyzed
ax2.scatter(data_cleaned['EngineV'], data_cleaned['Price'])
ax2.set_title('Price and EngineV')  # Focuses on the engine volume feature
ax3.scatter(data_cleaned['Mileage'], data_cleaned['Price'])
ax3.set_title('Price and Mileage')  # Highlights the mileage feature
plt.show()
```


In [24]:
```
# From the subplots and the PDF of price, we can easily determine that 'Price' is expo
# A good transformation in that case is a log transformation
sns.distplot(data_cleaned['Price'])
```

Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x2b8994ccdd8>




## Relaxing the assumptions

In [25]:
```
# Let's transform 'Price' with a log transformation
log_price = np.log(data_cleaned['Price'])

# Then we add it to our data frame
data_cleaned['log_price'] = log_price
data_cleaned
```

Out[25]:

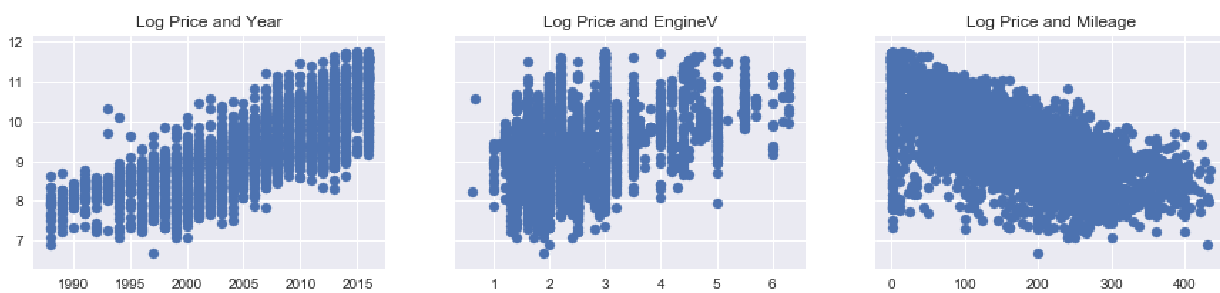
| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Year | log_price |
|---|---|---|---|---|---|---|---|---|---|
| **0** | BMW | 4200.0 | sedan | 277 | 2.00 | Petrol | yes | 1991 | 8.342840 |
| **1** | Mercedes-Benz | 7900.0 | van | 427 | 2.90 | Diesel | yes | 1999 | 8.974618 |
| **2** | Mercedes-Benz | 13300.0 | sedan | 358 | 5.00 | Gas | yes | 2003 | 9.495519 |
| **3** | Audi | 23000.0 | crossover | 240 | 4.20 | Petrol | yes | 2007 | 10.043249 |
| **4** | Toyota | 18300.0 | crossover | 120 | 2.00 | Petrol | yes | 2011 | 9.814656 |
| **5** | Audi | 14200.0 | vagon | 200 | 2.70 | Diesel | yes | 2006 | 9.560997 |
| **6** | Renault | 10799.0 | vagon | 193 | 1.50 | Diesel | yes | 2012 | 9.287209 |
| **7** | Volkswagen | 1400.0 | other | 212 | 1.80 | Gas | no | 1999 | 7.244228 |
| **8** | Renault | 11950.0 | vagon | 177 | 1.50 | Diesel | yes | 2011 | 9.388487 |
| **9** | Renault | 2500.0 | sedan | 260 | 1.79 | Petrol | yes | 1994 | 7.824046 |
| **10** | Audi | 9500.0 | vagon | 165 | 2.70 | Gas | yes | 2003 | 9.159047 |
| **11** | Volkswagen | 10500.0 | sedan | 100 | 1.80 | Petrol | yes | 2008 | 9.259131 |
| **12** | Toyota | 16000.0 | crossover | 250 | 4.70 | Gas | yes | 2001 | 9.680344 |
| **13** | Renault | 8600.0 | hatch | 84 | 1.50 | Diesel | yes | 2012 | 9.059517 |
| **14** | BMW | 2990.0 | other | 203 | 2.00 | Petrol | no | 2001 | 8.003029 |
| **15** | Toyota | 26500.0 | crossover | 21 | 2.00 | Petrol | yes | 2013 | 10.184900 |
| **16** | Audi | 3500.0 | vagon | 250 | 2.50 | Diesel | no | 1998 | 8.160518 |
| **17** | Toyota | 38233.0 | other | 0 | 2.40 | Diesel | yes | 2016 | 10.551454 |
| **18** | Volkswagen | 7500.0 | hatch | 132 | 1.40 | Diesel | yes | 2006 | 8.922658 |
| **19** | Audi | 6800.0 | sedan | 225 | 2.40 | Gas | yes | 1998 | 8.824678 |
| **20** | Mitsubishi | 10500.0 | crossover | 130 | 2.40 | Gas | yes | 2006 | 9.259131 |
| **21** | Audi | 24900.0 | sedan | 163 | 4.20 | Diesel | yes | 2008 | 10.122623 |
| **22** | Volkswagen | 20800.0 | crossover | 151 | 3.00 | Diesel | yes | 2008 | 9.942708 |
| **23** | Audi | 6500.0 | sedan | 330 | 2.40 | Petrol | yes | 1999 | 8.779557 |
| **24** | Mercedes-Benz | 13566.0 | other | 171 | 2.20 | Other | no | 2011 | 9.515322 |
| **25** | Mitsubishi | 8500.0 | hatch | 65 | 1.30 | Petrol | yes | 2010 | 9.047821 |
| **26** | Audi | 2900.0 | sedan | 1 | 2.30 | Gas | yes | 1989 | 7.972466 |
| **27** | BMW | 21500.0 | other | 72 | 3.00 | Petrol | yes | 2007 | 9.975808 |
| **28** | Mitsubishi | 17900.0 | crossover | 87 | 3.80 | Gas | yes | 2008 | 9.792556 |
| **29** | BMW | 28500.0 | crossover | 160 | 4.80 | Gas | yes | 2008 | 10.257659 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |

| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Year | log_price |
|---|---|---|---|---|---|---|---|---|---|
| **3837** | Mercedes-Benz | 11500.0 | van | 180 | 2.20 | Diesel | yes | 2011 | 9.350102 |
| **3838** | Mitsubishi | 9700.0 | van | 247 | 2.40 | Gas | yes | 2008 | 9.179881 |
| **3839** | Renault | 10500.0 | vagon | 185 | 1.50 | Diesel | yes | 2011 | 9.259131 |
| **3840** | Renault | 10900.0 | vagon | 180 | 1.50 | Diesel | yes | 2012 | 9.296518 |
| **3841** | Mercedes-Benz | 25500.0 | crossover | 77 | 3.50 | Petrol | yes | 2008 | 10.146434 |
| **3842** | Volkswagen | 15500.0 | sedan | 80 | 1.40 | Petrol | yes | 2013 | 9.648595 |
| **3843** | Volkswagen | 9750.0 | van | 159 | 1.90 | Diesel | yes | 2009 | 9.185023 |
| **3844** | BMW | 16100.0 | crossover | 194 | 3.00 | Diesel | yes | 2004 | 9.686575 |
| **3845** | Volkswagen | 9200.0 | vagon | 171 | 1.60 | Petrol | yes | 2008 | 9.126959 |
| **3846** | Volkswagen | 5150.0 | van | 240 | 2.00 | Diesel | yes | 2005 | 8.546752 |
| **3847** | Toyota | 100000.0 | crossover | 0 | 4.50 | Diesel | yes | 2016 | 11.512925 |
| **3848** | Renault | 8999.0 | other | 126 | 2.00 | Diesel | yes | 2012 | 9.104869 |
| **3849** | Mercedes-Benz | 16800.0 | sedan | 125 | 1.80 | Petrol | yes | 2008 | 9.729134 |
| **3850** | Mercedes-Benz | 8200.0 | sedan | 280 | 2.40 | Gas | yes | 1997 | 9.011889 |
| **3851** | Mercedes-Benz | 24950.0 | other | 60 | 1.80 | Petrol | yes | 2013 | 10.124629 |
| **3852** | Audi | 80999.0 | crossover | 0 | 3.00 | Diesel | yes | 2016 | 11.302192 |
| **3853** | Mercedes-Benz | 7300.0 | van | 207 | 2.20 | Diesel | yes | 2003 | 8.895630 |
| **3854** | BMW | 21335.0 | other | 105 | 3.00 | Petrol | yes | 2008 | 9.968104 |
| **3855** | BMW | 45000.0 | crossover | 80 | 3.00 | Petrol | yes | 2011 | 10.714418 |
| **3856** | Renault | 6750.0 | van | 155 | 1.50 | Diesel | yes | 2012 | 8.817298 |
| **3857** | Renault | 7000.0 | van | 210 | 1.50 | Diesel | yes | 2005 | 8.853665 |
| **3858** | BMW | 12090.0 | hatch | 145 | 1.60 | Petrol | yes | 2010 | 9.400134 |
| **3859** | BMW | 27900.0 | sedan | 38 | 2.00 | Petrol | yes | 2013 | 10.236382 |
| **3860** | Renault | 2100.0 | vagon | 237 | 1.90 | Diesel | no | 2001 | 7.649693 |
| **3861** | Renault | 6800.0 | sedan | 152 | 1.60 | Petrol | yes | 2007 | 8.824678 |
| **3862** | Volkswagen | 11500.0 | van | 163 | 2.50 | Diesel | yes | 2008 | 9.350102 |
| **3863** | Toyota | 17900.0 | sedan | 35 | 1.60 | Petrol | yes | 2014 | 9.792556 |
| **3864** | Mercedes-Benz | 125000.0 | sedan | 9 | 3.00 | Diesel | yes | 2014 | 11.736069 |
| **3865** | BMW | 6500.0 | sedan | 1 | 3.50 | Petrol | yes | 1999 | 8.779557 |

| | Brand | Price | Body | Mileage | EngineV | Engine Type | Registration | Year | log_price |
|---|---|---|---|---|---|---|---|---|---|
| **3866** | Volkswagen | 13500.0 | van | 124 | 2.00 | Diesel | yes | 2013 | 9.510445 |

3867 rows × 9 columns

In [26]:
```
# Let's check the three scatters once again
f, (ax1, ax2, ax3) = plt.subplots(1, 3, sharey=True, figsize =(15,3))
ax1.scatter(data_cleaned['Year'],data_cleaned['log_price'])
ax1.set_title('Log Price and Year')
ax2.scatter(data_cleaned['EngineV'],data_cleaned['log_price'])
ax2.set_title('Log Price and EngineV')
ax3.scatter(data_cleaned['Mileage'],data_cleaned['log_price'])
ax3.set_title('Log Price and Mileage')


plt.show()

# The relationships show a clear linear relationship
# This is some good linear regression material

# Alternatively we could have transformed each of the independent variables
```




In [27]:
```
# Since we will be using the log price variable, we can drop the old 'Price' one
data_cleaned = data_cleaned.drop(['Price'],axis=1)
```

In [28]:
```
# Let's quickly see the columns of our data frame
data_cleaned.columns.values
```

Out[28]:
```
array(['Brand', 'Body', 'Mileage', 'EngineV', 'Engine Type',
       'Registration', 'Year', 'log_price'], dtype=object)
```

# Checking Multicollinearity Using Variance Inflation Factor (VIF)

Multicollinearity occurs when independent variables in a regression model are highly correlated, leading to unstable estimates of coefficients. To ensure the reliability of our model, we check for multicollinearity using the Variance Inflation Factor (VIF).

In [29]:
```
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Selecting only the numerical independent variables for the VIF calculation.
# Categorical variables are excluded as they are not suitable for VIF analysis.
variables = data_cleaned[['Mileage', 'Year', 'EngineV']]
```

```
# Creating a DataFrame to store the VIF values.
# Each feature will have a corresponding VIF value, which indicates how much multicoll
vif = pd.DataFrame()

# Calculating the VIF values for each variable.
# The variance_inflation_factor function computes VIF for each feature.
# It loops through all features and calculates their respective VIF values.
vif["VIF"] = [variance_inflation_factor(variables.values, i) for i in range(variables.

# Adding the feature names to the DataFrame for better readability of the results.
# This ensures the VIF values are linked to their respective variables.
vif["Features"] = variables.columns
```

In [30]:

```
# Let's explore the result
vif
```

Out[30]:

| | VIF | Features |
|---|---|---|
| **0** | 3.791584 | Mileage |
| **1** | 10.354854 | Year |
| **2** | 7.662068 | EngineV |

In [31]:

```
# Since Year has the highest VIF, I will remove it from the model
# This will drive the VIF of other variables down!!!
# So even if EngineV seems with a high VIF, too, once 'Year' is gone that will no long
data_no_multicollinearity = data_cleaned.drop(['Year'],axis=1)
```

# Create dummy variables

In [32]:

```
# To include the categorical data in the regression, let's create dummies
# There is a very convenient method called: 'get_dummies' which does that seemlessly
# It is extremely important that we drop one of the dummies, alternatively we will int
data_with_dummies = pd.get_dummies(data_no_multicollinearity, drop_first=True)
```

In [33]:

```
# Here's the result
data_with_dummies.head()
```

Out[33]:

| | Mileage | EngineV | log_price | Brand_BMW | Brand_Mercedes-Benz | Brand_Mitsubishi | Brand_Renault | Bra |
|---|---|---|---|---|---|---|---|---|
| **0** | 277 | 2.0 | 8.342840 | 1 | 0 | 0 | 0 | |
| **1** | 427 | 2.9 | 8.974618 | 0 | 1 | 0 | 0 | |
| **2** | 358 | 5.0 | 9.495519 | 0 | 1 | 0 | 0 | |
| **3** | 240 | 4.2 | 10.043249 | 0 | 0 | 0 | 0 | |
| **4** | 120 | 2.0 | 9.814656 | 0 | 0 | 0 | 0 | |

# Rearrange a bit

In [34]:
```
# To make our data frame more organized, we prefer to place the dependent variable in
# Since each problem is different, that must be done manually
# We can display all possible features and then choose the desired order
data_with_dummies.columns.values
```

Out[34]:
```
array(['Mileage', 'EngineV', 'log_price', 'Brand_BMW',
       'Brand_Mercedes-Benz', 'Brand_Mitsubishi', 'Brand_Renault',
       'Brand_Toyota', 'Brand_Volkswagen', 'Body_hatch', 'Body_other',
       'Body_sedan', 'Body_vagon', 'Body_van', 'Engine Type_Gas',
       'Engine Type_Other', 'Engine Type_Petrol', 'Registration_yes'],
      dtype=object)
```

In [35]:
```
# To make the code a bit more parametrized, let's declare a new variable that will con
# Conventionally, the most intuitive order is: dependent variable, indepedendent numer
cols = ['log_price', 'Mileage', 'EngineV', 'Brand_BMW',
       'Brand_Mercedes-Benz', 'Brand_Mitsubishi', 'Brand_Renault',
       'Brand_Toyota', 'Brand_Volkswagen', 'Body_hatch', 'Body_other',
       'Body_sedan', 'Body_vagon', 'Body_van', 'Engine Type_Gas',
       'Engine Type_Other', 'Engine Type_Petrol', 'Registration_yes']
```

In [36]:
```
# To implement the reordering, we will create a new df, which is equal to the old one
data_preprocessed = data_with_dummies[cols]
data_preprocessed.head()
```

Out[36]:

| | log_price | Mileage | EngineV | Brand_BMW | Brand_Mercedes-Benz | Brand_Mitsubishi | Brand_Renault | Bra |
|---|---|---|---|---|---|---|---|---|
| **0** | 8.342840 | 277 | 2.0 | 1 | 0 | 0 | 0 | |
| **1** | 8.974618 | 427 | 2.9 | 0 | 1 | 0 | 0 | |
| **2** | 9.495519 | 358 | 5.0 | 0 | 1 | 0 | 0 | |
| **3** | 10.043249 | 240 | 4.2 | 0 | 0 | 0 | 0 | |
| **4** | 9.814656 | 120 | 2.0 | 0 | 0 | 0 | 0 | |

# Linear regression model

## Declaring the inputs and the targets

In [37]:
```
# The target(s) (dependent variable) is 'log price'
targets = data_preprocessed['log_price']

# The inputs are everything BUT the dependent variable, so we can simply drop it
inputs = data_preprocessed.drop(['log_price'],axis=1)
```

## Scaling the data

In [38]:
```
# Importing the scaling module
from sklearn.preprocessing import StandardScaler

# Creating a scaler object
scaler = StandardScaler()
```

```
# Fit the inputs (calculate the mean and standard deviation feature-wise)
scaler.fit(inputs)
```

Out[38]:

```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

In [39]:

```
# Scaled the features and storing them in a new variable (the actual scaling procedure
inputs_scaled = scaler.transform(inputs)
```

## Train Test Split

In [40]:

```
# Importing the module for the split
from sklearn.model_selection import train_test_split

# Spliting the variables with an 80-20 split and some random state
x_train, x_test, y_train, y_test = train_test_split(inputs_scaled, targets, test_size=
```

## Create the regression

In [41]:

```
# Creating a linear regression object
reg = LinearRegression()
# Fiting the regression with the scaled TRAIN inputs and targets
reg.fit(x_train,y_train)
```

Out[41]:

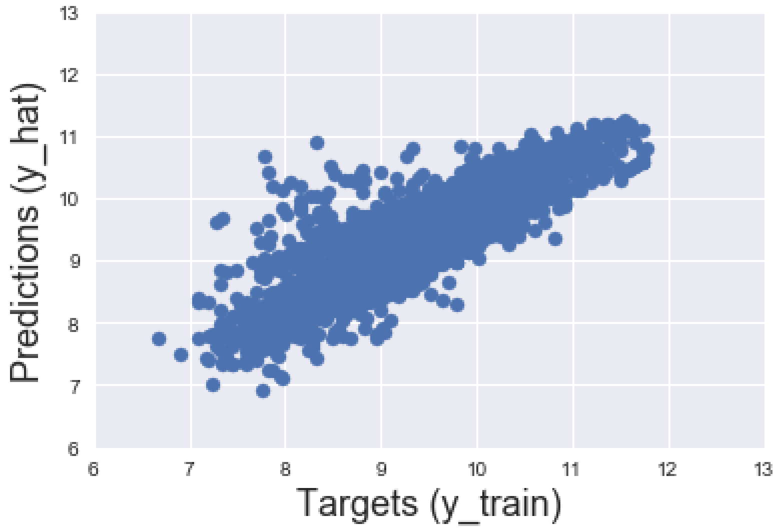
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [42]:

```
# Let's check the outputs of the regression
# I'll store them in y_hat
y_hat = reg.predict(x_train)
```

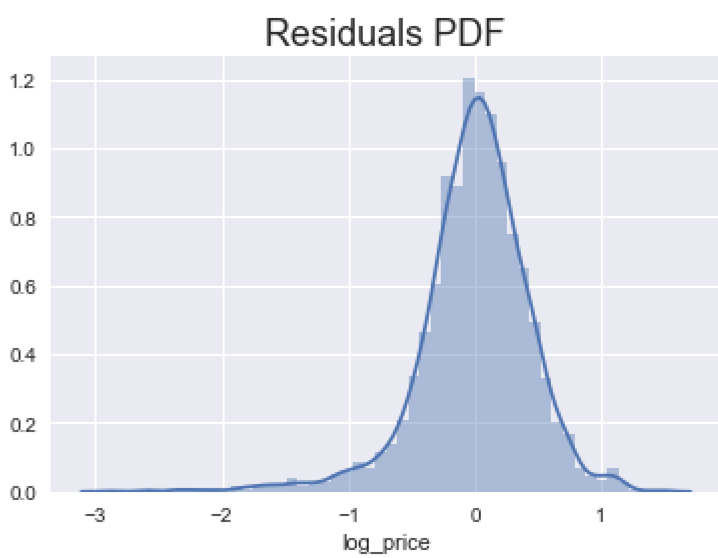
In [43]:

```
# The simplest way to compare the targets (y_train) and the predictions (y_hat) is to
# The closer the points to the 45-degree line, the better the prediction
plt.scatter(y_train, y_hat)
# Let's also name the axis
plt.xlabel('Targets (y_train)',size=18)
plt.ylabel('Predictions (y_hat)',size=18)
# Sometimes the plot will have different scales of the x-axis and the y-axis
# This is an issue as we won't be able to interpret the '45-degree line'
# We want the x-axis and the y-axis to be the same
plt.xlim(6,13)
plt.ylim(6,13)
plt.show()
```




In [44]:
```
# Another useful check of our model is a residual plot
# We can plot the PDF of the residuals and check for anomalies
sns.distplot(y_train - y_hat)

# Including a title
plt.title("Residuals PDF", size=18)

# In the best case scenario this plot should be normally distributed
# In our case we notice that there are many negative residuals (far away from the mean
# Given the definition of the residuals (y_train - y_hat), negative values imply that
```

Out[44]: Text(0.5,1,'Residuals PDF')




In [45]:
```
# Finding the R-squared of the model
reg.score(x_train,y_train)
```

Out[45]: 0.744996578792662

## Finding the weights and bias

In [46]:
```
# Obtain the bias (intercept) of the regression
reg.intercept_
```

Out[46]:
```
9.415239458021299
```

In [47]:
```python
# Obtain the weights (coefficients) of the regression
reg.coef_
```

Out[47]:
```
array([-0.44871341,  0.20903483,  0.0142496 ,  0.01288174, -0.14055166,
       -0.17990912, -0.06054988, -0.08992433, -0.1454692 , -0.10144383,
       -0.20062984, -0.12988747, -0.16859669, -0.12149035, -0.03336798,
       -0.14690868,  0.32047333])
```

In [48]:
```python
# Creating a regression summary where we can compare them with one-another
reg_summary = pd.DataFrame(inputs.columns.values, columns=['Features'])
reg_summary['Weights'] = reg.coef_
reg_summary
```

Out[48]:

| | Features | Weights |
|---|---|---|
| **0** | Mileage | -0.448713 |
| **1** | EngineV | 0.209035 |
| **2** | Brand_BMW | 0.014250 |
| **3** | Brand_Mercedes-Benz | 0.012882 |
| **4** | Brand_Mitsubishi | -0.140552 |
| **5** | Brand_Renault | -0.179909 |
| **6** | Brand_Toyota | -0.060550 |
| **7** | Brand_Volkswagen | -0.089924 |
| **8** | Body_hatch | -0.145469 |
| **9** | Body_other | -0.101444 |
| **10** | Body_sedan | -0.200630 |
| **11** | Body_vagon | -0.129887 |
| **12** | Body_van | -0.168597 |
| **13** | Engine Type_Gas | -0.121490 |
| **14** | Engine Type_Other | -0.033368 |
| **15** | Engine Type_Petrol | -0.146909 |
| **16** | Registration_yes | 0.320473 |

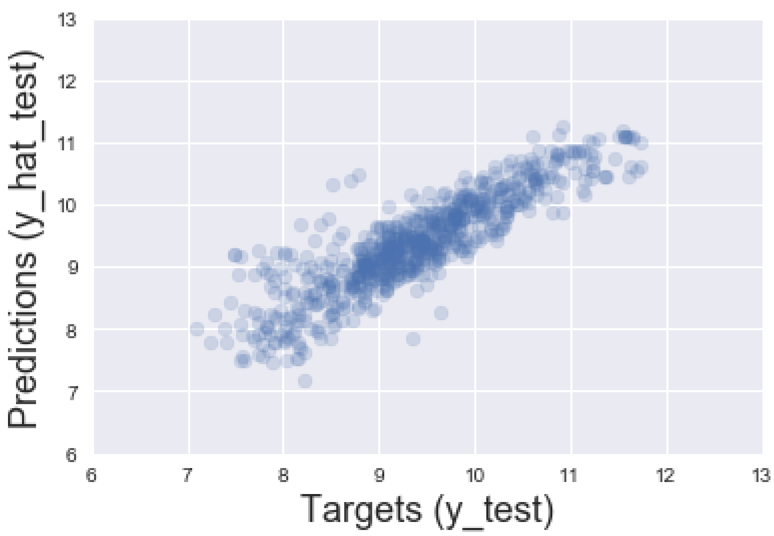
In [49]:
```python
# Checking the different categories in the 'Brand' variable
data_cleaned['Brand'].unique()

# In this way we can see which 'Brand' is actually the benchmark
```

Out[49]:
```
array(['BMW', 'Mercedes-Benz', 'Audi', 'Toyota', 'Renault', 'Volkswagen',
       'Mitsubishi'], dtype=object)
```

# Testing

In [50]:
```
# Once we have trained and fine-tuned our model, we can proceed to testing it
# Testing is done on a dataset that the algorithm has never seen
# Luckily we have prepared such a dataset
# Our test inputs are 'x_test', while the outputs: 'y_test'
# If the predictions are far off, we will know that our model overfitted
y_hat_test = reg.predict(x_test)
```

In [51]:
```
# Creating a scatter plot with the test targets and the test predictions
plt.scatter(y_test, y_hat_test, alpha=0.2)
plt.xlabel('Targets (y_test)',size=18)
plt.ylabel('Predictions (y_hat_test)',size=18)
plt.xlim(6,13)
plt.ylim(6,13)
plt.show()
```




In [52]:
```
# Finally, let's manually check these predictions
# To obtain the actual prices, we take the exponential of the log_price
df_pf = pd.DataFrame(np.exp(y_hat_test), columns=['Prediction'])
df_pf.head()
```

Out[52]:

| | Prediction |
|---|---|
| 0 | 10685.501696 |
| 1 | 3499.255242 |
| 2 | 7553.285218 |
| 3 | 7463.963017 |
| 4 | 11353.490075 |

In [53]:
```
# We can also include the test targets in that data frame (so we can manually compare
df_pf['Target'] = np.exp(y_test)
df_pf

# Note that we have a lot of missing values
# There is no reason to have ANY missing values, though
# This suggests that something is wrong with the data frame / indexing
```

Out[53]:

| | Prediction | Target |
|---|---|---|
| **0** | 10685.501696 | NaN |
| **1** | 3499.255242 | 7900.0 |
| **2** | 7553.285218 | NaN |
| **3** | 7463.963017 | NaN |
| **4** | 11353.490075 | NaN |
| **5** | 21289.799394 | 14200.0 |
| **6** | 20159.189144 | NaN |
| **7** | 20349.617702 | NaN |
| **8** | 11581.537864 | 11950.0 |
| **9** | 33614.617349 | NaN |
| **10** | 7241.068243 | NaN |
| **11** | 5175.769541 | 10500.0 |
| **12** | 5484.015362 | NaN |
| **13** | 13292.711243 | NaN |
| **14** | 8248.666686 | NaN |
| **15** | 10621.836767 | NaN |
| **16** | 23721.581637 | 3500.0 |
| **17** | 11770.636010 | NaN |
| **18** | 37600.146722 | 7500.0 |
| **19** | 16178.143307 | 6800.0 |
| **20** | 11876.820988 | NaN |
| **21** | 31557.804999 | NaN |
| **22** | 6102.358118 | NaN |
| **23** | 13111.914144 | NaN |
| **24** | 23650.150725 | NaN |
| **25** | 45272.248411 | NaN |
| **26** | 2178.941672 | NaN |
| **27** | 2555.022542 | NaN |
| **28** | 35991.510539 | NaN |
| **29** | 26062.229419 | NaN |
| **...** | ... | ... |
| **744** | 2379.583414 | NaN |
| **745** | 6421.180201 | 7777.0 |
| **746** | 13355.106770 | NaN |

| | Prediction | Target |
|---|---|---|
| **747** | 8453.281424 | 10500.0 |
| **748** | 48699.979367 | NaN |
| **749** | 6082.849234 | 4100.0 |
| **750** | 10381.621436 | NaN |
| **751** | 8493.042746 | NaN |
| **752** | 8591.658845 | 13999.0 |
| **753** | 6358.547301 | NaN |
| **754** | 17028.451182 | NaN |
| **755** | 15885.658673 | NaN |
| **756** | 3752.540952 | NaN |
| **757** | 12028.905190 | NaN |
| **758** | 9380.459827 | 16999.0 |
| **759** | 10125.265176 | NaN |
| **760** | 13443.324968 | NaN |
| **761** | 9097.127448 | NaN |
| **762** | 12201.288474 | 4700.0 |
| **763** | 12383.352887 | NaN |
| **764** | 14049.760996 | NaN |
| **765** | 11034.660068 | 3750.0 |
| **766** | 18982.148845 | NaN |
| **767** | 24323.483753 | NaN |
| **768** | 38260.361723 | NaN |
| **769** | 29651.726363 | 6950.0 |
| **770** | 10732.071179 | NaN |
| **771** | 13922.446953 | NaN |
| **772** | 27487.751303 | NaN |
| **773** | 13491.163043 | NaN |

774 rows × 2 columns

In [54]:
```
# After displaying y_test, we find what the issue is
# The old indexes are preserved

# Therefore, to get a proper result, we must reset the index and drop the old indexing
y_test = y_test.reset_index(drop=True)

# Check the result
y_test.head()
```

Out[54]:
```
0    7.740664
1    7.937375
2    7.824046
3    8.764053
4    9.121509
Name: log_price, dtype: float64
```

In [55]:
```
# Let's overwrite the 'Target' column with the appropriate values
# Again, we need the exponential of the test log price
df_pf['Target'] = np.exp(y_test)
df_pf
```

Out[55]:

| | Prediction | Target |
|---|---|---|
| 0 | 10685.501696 | 2300.0 |
| 1 | 3499.255242 | 2800.0 |
| 2 | 7553.285218 | 2500.0 |
| 3 | 7463.963017 | 6400.0 |
| 4 | 11353.490075 | 9150.0 |
| 5 | 21289.799394 | 20000.0 |
| 6 | 20159.189144 | 38888.0 |
| 7 | 20349.617702 | 16999.0 |
| 8 | 11581.537864 | 12500.0 |
| 9 | 33614.617349 | 41000.0 |
| 10 | 7241.068243 | 12800.0 |
| 11 | 5175.769541 | 5000.0 |
| 12 | 5484.015362 | 7900.0 |
| 13 | 13292.711243 | 16999.0 |
| 14 | 8248.666686 | 9200.0 |
| 15 | 10621.836767 | 11999.0 |
| 16 | 23721.581637 | 20500.0 |
| 17 | 11770.636010 | 9700.0 |
| 18 | 37600.146722 | 39900.0 |
| 19 | 16178.143307 | 16400.0 |
| 20 | 11876.820988 | 15200.0 |
| 21 | 31557.804999 | 24500.0 |
| 22 | 6102.358118 | 5650.0 |
| 23 | 13111.914144 | 12900.0 |
| 24 | 23650.150725 | 20900.0 |
| 25 | 45272.248411 | 31990.0 |
| 26 | 2178.941672 | 3600.0 |
| 27 | 2555.022542 | 11600.0 |
| 28 | 35991.510539 | 43999.0 |
| 29 | 26062.229419 | 42500.0 |
| ... | ... | ... |
| 744 | 2379.583414 | 3000.0 |
| 745 | 6421.180201 | 4400.0 |
| 746 | 13355.106770 | 7500.0 |

| | Prediction | Target |
|---|---|---|
| **747** | 8453.281424 | 10900.0 |
| **748** | 48699.979367 | 77500.0 |
| **749** | 6082.849234 | 7450.0 |
| **750** | 10381.621436 | 3000.0 |
| **751** | 8493.042746 | 12800.0 |
| **752** | 8591.658845 | 12000.0 |
| **753** | 6358.547301 | 4850.0 |
| **754** | 17028.451182 | 18700.0 |
| **755** | 15885.658673 | 17300.0 |
| **756** | 3752.540952 | 2600.0 |
| **757** | 12028.905190 | 10500.0 |
| **758** | 9380.459827 | 7950.0 |
| **759** | 10125.265176 | 6700.0 |
| **760** | 13443.324968 | 9000.0 |
| **761** | 9097.127448 | 8000.0 |
| **762** | 12201.288474 | 12999.0 |
| **763** | 12383.352887 | 10800.0 |
| **764** | 14049.760996 | 10700.0 |
| **765** | 11034.660068 | 9800.0 |
| **766** | 18982.148845 | 17900.0 |
| **767** | 24323.483753 | 18800.0 |
| **768** | 38260.361723 | 75555.0 |
| **769** | 29651.726363 | 29500.0 |
| **770** | 10732.071179 | 9600.0 |
| **771** | 13922.446953 | 18300.0 |
| **772** | 27487.751303 | 68500.0 |
| **773** | 13491.163043 | 10800.0 |

774 rows × 2 columns

In [56]:

```
# Additionally, we can calculate the difference between the targets and the prediction
df_pf['Residual'] = df_pf['Target'] - df_pf['Prediction']

# Since OLS is basically an algorithm which minimizes the total sum of squared errors
# this comparison makes a lot of sense
```

In [57]:

```
# Finally, it makes sense to see how far off we are from the result percentage-wise
# Here, we take the absolute difference in %, so we can easily order the data frame
```

```
df_pf['Difference%'] = np.absolute(df_pf['Residual']/df_pf['Target']*100)
df_pf
```

Out[57]:

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| 0 | 10685.501696 | 2300.0 | -8385.501696 | 364.587030 |
| 1 | 3499.255242 | 2800.0 | -699.255242 | 24.973402 |
| 2 | 7553.285218 | 2500.0 | -5053.285218 | 202.131409 |
| 3 | 7463.963017 | 6400.0 | -1063.963017 | 16.624422 |
| 4 | 11353.490075 | 9150.0 | -2203.490075 | 24.081859 |
| 5 | 21289.799394 | 20000.0 | -1289.799394 | 6.448997 |
| 6 | 20159.189144 | 38888.0 | 18728.810856 | 48.160900 |
| 7 | 20349.617702 | 16999.0 | -3350.617702 | 19.710675 |
| 8 | 11581.537864 | 12500.0 | 918.462136 | 7.347697 |
| 9 | 33614.617349 | 41000.0 | 7385.382651 | 18.013128 |
| 10 | 7241.068243 | 12800.0 | 5558.931757 | 43.429154 |
| 11 | 5175.769541 | 5000.0 | -175.769541 | 3.515391 |
| 12 | 5484.015362 | 7900.0 | 2415.984638 | 30.582084 |
| 13 | 13292.711243 | 16999.0 | 3706.288757 | 21.802981 |
| 14 | 8248.666686 | 9200.0 | 951.333314 | 10.340580 |
| 15 | 10621.836767 | 11999.0 | 1377.163233 | 11.477317 |
| 16 | 23721.581637 | 20500.0 | -3221.581637 | 15.715032 |
| 17 | 11770.636010 | 9700.0 | -2070.636010 | 21.346763 |
| 18 | 37600.146722 | 39900.0 | 2299.853278 | 5.764043 |
| 19 | 16178.143307 | 16400.0 | 221.856693 | 1.352785 |
| 20 | 11876.820988 | 15200.0 | 3323.179012 | 21.863020 |
| 21 | 31557.804999 | 24500.0 | -7057.804999 | 28.807367 |
| 22 | 6102.358118 | 5650.0 | -452.358118 | 8.006338 |
| 23 | 13111.914144 | 12900.0 | -211.914144 | 1.642745 |
| 24 | 23650.150725 | 20900.0 | -2750.150725 | 13.158616 |
| 25 | 45272.248411 | 31990.0 | -13282.248411 | 41.520001 |
| 26 | 2178.941672 | 3600.0 | 1421.058328 | 39.473842 |
| 27 | 2555.022542 | 11600.0 | 9044.977458 | 77.973944 |
| 28 | 35991.510539 | 43999.0 | 8007.489461 | 18.199253 |
| 29 | 26062.229419 | 42500.0 | 16437.770581 | 38.677107 |
| ... | ... | ... | ... | ... |
| 744 | 2379.583414 | 3000.0 | 620.416586 | 20.680553 |
| 745 | 6421.180201 | 4400.0 | -2021.180201 | 45.935914 |
| 746 | 13355.106770 | 7500.0 | -5855.106770 | 78.068090 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **747** | 8453.281424 | 10900.0 | 2446.718576 | 22.446959 |
| **748** | 48699.979367 | 77500.0 | 28800.020633 | 37.161317 |
| **749** | 6082.849234 | 7450.0 | 1367.150766 | 18.351017 |
| **750** | 10381.621436 | 3000.0 | -7381.621436 | 246.054048 |
| **751** | 8493.042746 | 12800.0 | 4306.957254 | 33.648104 |
| **752** | 8591.658845 | 12000.0 | 3408.341155 | 28.402843 |
| **753** | 6358.547301 | 4850.0 | -1508.547301 | 31.104068 |
| **754** | 17028.451182 | 18700.0 | 1671.548818 | 8.938764 |
| **755** | 15885.658673 | 17300.0 | 1414.341327 | 8.175383 |
| **756** | 3752.540952 | 2600.0 | -1152.540952 | 44.328498 |
| **757** | 12028.905190 | 10500.0 | -1528.905190 | 14.561002 |
| **758** | 9380.459827 | 7950.0 | -1430.459827 | 17.993205 |
| **759** | 10125.265176 | 6700.0 | -3425.265176 | 51.123361 |
| **760** | 13443.324968 | 9000.0 | -4443.324968 | 49.370277 |
| **761** | 9097.127448 | 8000.0 | -1097.127448 | 13.714093 |
| **762** | 12201.288474 | 12999.0 | 797.711526 | 6.136715 |
| **763** | 12383.352887 | 10800.0 | -1583.352887 | 14.660675 |
| **764** | 14049.760996 | 10700.0 | -3349.760996 | 31.306178 |
| **765** | 11034.660068 | 9800.0 | -1234.660068 | 12.598572 |
| **766** | 18982.148845 | 17900.0 | -1082.148845 | 6.045524 |
| **767** | 24323.483753 | 18800.0 | -5523.483753 | 29.380233 |
| **768** | 38260.361723 | 75555.0 | 37294.638277 | 49.360914 |
| **769** | 29651.726363 | 29500.0 | -151.726363 | 0.514327 |
| **770** | 10732.071179 | 9600.0 | -1132.071179 | 11.792408 |
| **771** | 13922.446953 | 18300.0 | 4377.553047 | 23.921055 |
| **772** | 27487.751303 | 68500.0 | 41012.248697 | 59.871896 |
| **773** | 13491.163043 | 10800.0 | -2691.163043 | 24.918176 |

774 rows × 4 columns

In [58]:
```
# Exploring the descriptives here gives us additional insights
df_pf.describe()
```

Out[58]:

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **count** | 774.000000 | 774.000000 | 774.000000 | 774.000000 |
| **mean** | 15946.760167 | 18165.817106 | 2219.056939 | 36.256693 |
| **std** | 13133.197604 | 19967.858908 | 10871.218143 | 55.066507 |
| **min** | 1320.562768 | 1200.000000 | -29456.498331 | 0.062794 |
| **25%** | 7413.644234 | 6900.000000 | -2044.191251 | 12.108022 |
| **50%** | 11568.168859 | 11600.000000 | 142.518577 | 23.467728 |
| **75%** | 20162.408805 | 20500.000000 | 3147.343497 | 39.563570 |
| **max** | 77403.055224 | 126000.000000 | 85106.162329 | 512.688080 |

In [59]:

```
# To see all rows, we use the relevant pandas syntax
pd.options.display.max_rows = 999
# Moreover, to make the dataset clear, we can display the result with only 2 digits af
pd.set_option('display.float_format', lambda x: '%.2f' % x)
# Finally, we sort by difference in % and manually check the model
df_pf.sort_values(by=['Difference%'])
```

Out[59]:

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **698** | 30480.85 | 30500.00 | 19.15 | 0.06 |
| **742** | 16960.31 | 16999.00 | 38.69 | 0.23 |
| **60** | 12469.21 | 12500.00 | 30.79 | 0.25 |
| **110** | 25614.14 | 25500.00 | -114.14 | 0.45 |
| **367** | 42703.68 | 42500.00 | -203.68 | 0.48 |
| **369** | 3084.69 | 3100.00 | 15.31 | 0.49 |
| **769** | 29651.73 | 29500.00 | -151.73 | 0.51 |
| **272** | 9749.53 | 9800.00 | 50.47 | 0.52 |
| **714** | 23118.07 | 22999.00 | -119.07 | 0.52 |
| **630** | 8734.58 | 8800.00 | 65.42 | 0.74 |
| **380** | 3473.79 | 3500.00 | 26.21 | 0.75 |
| **648** | 21174.10 | 21335.00 | 160.90 | 0.75 |
| **308** | 8967.74 | 8900.00 | -67.74 | 0.76 |
| **665** | 17858.02 | 18000.00 | 141.98 | 0.79 |
| **379** | 17654.84 | 17800.00 | 145.16 | 0.82 |
| **719** | 11391.95 | 11500.00 | 108.05 | 0.94 |
| **102** | 28625.56 | 28900.00 | 274.44 | 0.95 |
| **94** | 7724.17 | 7800.00 | 75.83 | 0.97 |
| **561** | 6429.03 | 6500.00 | 70.97 | 1.09 |
| **242** | 7597.39 | 7500.00 | -97.39 | 1.30 |
| **528** | 18555.09 | 18800.00 | 244.91 | 1.30 |
| **61** | 7396.87 | 7300.00 | -96.87 | 1.33 |
| **19** | 16178.14 | 16400.00 | 221.86 | 1.35 |
| **280** | 12327.10 | 12499.00 | 171.90 | 1.38 |
| **311** | 51287.19 | 52055.25 | 768.06 | 1.48 |
| **723** | 6009.63 | 6100.00 | 90.37 | 1.48 |
| **49** | 4973.17 | 4900.00 | -73.17 | 1.49 |
| **114** | 27716.14 | 27300.00 | -416.14 | 1.52 |
| **636** | 28498.91 | 28950.00 | 451.09 | 1.56 |
| **612** | 2953.17 | 3000.00 | 46.83 | 1.56 |
| **47** | 26425.14 | 25999.00 | -426.14 | 1.64 |
| **23** | 13111.91 | 12900.00 | -211.91 | 1.64 |
| **31** | 12858.08 | 12650.00 | -208.08 | 1.64 |
| **91** | 13421.16 | 13200.00 | -221.16 | 1.68 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **329** | 7327.18 | 7200.00 | -127.18 | 1.77 |
| **549** | 3816.33 | 3750.00 | -66.33 | 1.77 |
| **252** | 9721.50 | 9900.00 | 178.50 | 1.80 |
| **387** | 44173.72 | 44999.00 | 825.28 | 1.83 |
| **267** | 40753.58 | 40000.00 | -753.58 | 1.88 |
| **467** | 22262.80 | 22711.65 | 448.85 | 1.98 |
| **556** | 18231.44 | 18600.00 | 368.56 | 1.98 |
| **165** | 9596.94 | 9400.00 | -196.94 | 2.10 |
| **259** | 6067.79 | 6200.00 | 132.21 | 2.13 |
| **601** | 35371.16 | 34600.00 | -771.16 | 2.23 |
| **708** | 11967.39 | 11700.00 | -267.39 | 2.29 |
| **593** | 17908.00 | 17500.00 | -408.00 | 2.33 |
| **398** | 8707.13 | 8500.00 | -207.13 | 2.44 |
| **526** | 29049.27 | 28350.00 | -699.27 | 2.47 |
| **603** | 14513.46 | 14900.00 | 386.54 | 2.59 |
| **53** | 20453.89 | 21000.00 | 546.11 | 2.60 |
| **632** | 15383.35 | 14990.00 | -393.35 | 2.62 |
| **533** | 24642.50 | 24000.00 | -642.50 | 2.68 |
| **497** | 50099.92 | 51500.00 | 1400.08 | 2.72 |
| **212** | 16133.86 | 15700.00 | -433.86 | 2.76 |
| **130** | 17489.92 | 18000.00 | 510.08 | 2.83 |
| **290** | 1894.40 | 1950.00 | 55.60 | 2.85 |
| **78** | 30810.25 | 29900.00 | -910.25 | 3.04 |
| **642** | 8721.97 | 8999.00 | 277.03 | 3.08 |
| **437** | 18866.50 | 18300.00 | -566.50 | 3.10 |
| **101** | 5958.63 | 6150.00 | 191.37 | 3.11 |
| **314** | 5811.74 | 6000.00 | 188.26 | 3.14 |
| **150** | 9800.43 | 9500.00 | -300.43 | 3.16 |
| **565** | 7324.63 | 7100.00 | -224.63 | 3.16 |
| **574** | 12583.52 | 13000.00 | 416.48 | 3.20 |
| **591** | 10115.13 | 9800.00 | -315.13 | 3.22 |
| **172** | 11156.38 | 10800.00 | -356.38 | 3.30 |
| **133** | 9279.28 | 9600.00 | 320.72 | 3.34 |
| **480** | 31369.37 | 32500.00 | 1130.63 | 3.48 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **87** | 2315.71 | 2400.00 | 84.29 | 3.51 |
| **11** | 5175.77 | 5000.00 | -175.77 | 3.52 |
| **43** | 21611.83 | 22400.00 | 788.17 | 3.52 |
| **96** | 7976.26 | 7700.00 | -276.26 | 3.59 |
| **406** | 24874.86 | 23999.00 | -875.86 | 3.65 |
| **173** | 36516.35 | 37900.00 | 1383.65 | 3.65 |
| **540** | 4666.05 | 4500.00 | -166.05 | 3.69 |
| **40** | 18672.68 | 18000.00 | -672.68 | 3.74 |
| **340** | 14815.83 | 15400.00 | 584.17 | 3.79 |
| **239** | 10581.62 | 10999.00 | 417.38 | 3.79 |
| **109** | 12663.54 | 12200.00 | -463.54 | 3.80 |
| **256** | 1825.44 | 1900.00 | 74.56 | 3.92 |
| **317** | 12247.90 | 12750.00 | 502.10 | 3.94 |
| **77** | 5930.73 | 5700.00 | -230.73 | 4.05 |
| **301** | 9782.47 | 10200.00 | 417.53 | 4.09 |
| **333** | 12452.22 | 11960.00 | -492.22 | 4.12 |
| **570** | 23163.87 | 24171.42 | 1007.55 | 4.17 |
| **581** | 3246.94 | 3390.00 | 143.06 | 4.22 |
| **693** | 12354.16 | 12900.00 | 545.84 | 4.23 |
| **381** | 33499.44 | 35000.00 | 1500.56 | 4.29 |
| **438** | 16257.03 | 16999.00 | 741.97 | 4.36 |
| **368** | 8415.81 | 8800.00 | 384.19 | 4.37 |
| **273** | 12318.39 | 12900.00 | 581.61 | 4.51 |
| **235** | 2765.14 | 2900.00 | 134.86 | 4.65 |
| **168** | 11420.95 | 11999.00 | 578.05 | 4.82 |
| **707** | 2725.40 | 2600.00 | -125.40 | 4.82 |
| **72** | 23624.69 | 22500.00 | -1124.69 | 5.00 |
| **559** | 11377.84 | 11999.00 | 621.16 | 5.18 |
| **134** | 12096.18 | 11500.00 | -596.18 | 5.18 |
| **446** | 9363.27 | 8900.00 | -463.27 | 5.21 |
| **127** | 12311.47 | 11700.00 | -611.47 | 5.23 |
| **450** | 14218.94 | 13500.00 | -718.94 | 5.33 |
| **293** | 8431.89 | 7999.00 | -432.89 | 5.41 |
| **18** | 37600.15 | 39900.00 | 2299.85 | 5.76 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **452** | 39882.67 | 37700.00 | -2182.67 | 5.79 |
| **195** | 20588.57 | 21900.00 | 1311.43 | 5.99 |
| **676** | 6861.84 | 7300.00 | 438.16 | 6.00 |
| **766** | 18982.15 | 17900.00 | -1082.15 | 6.05 |
| **328** | 15067.85 | 14200.00 | -867.85 | 6.11 |
| **762** | 12201.29 | 12999.00 | 797.71 | 6.14 |
| **649** | 11147.91 | 10500.00 | -647.91 | 6.17 |
| **299** | 20183.30 | 18999.00 | -1184.30 | 6.23 |
| **641** | 17334.11 | 18500.00 | 1165.89 | 6.30 |
| **5** | 21289.80 | 20000.00 | -1289.80 | 6.45 |
| **545** | 6826.78 | 7300.00 | 473.22 | 6.48 |
| **622** | 4472.47 | 4200.00 | -272.47 | 6.49 |
| **422** | 53294.61 | 57000.00 | 3705.39 | 6.50 |
| **740** | 6658.73 | 6250.00 | -408.73 | 6.54 |
| **85** | 19001.29 | 17800.00 | -1201.29 | 6.75 |
| **315** | 4590.49 | 4300.00 | -290.49 | 6.76 |
| **462** | 43236.92 | 40500.00 | -2736.92 | 6.76 |
| **394** | 9076.42 | 8500.00 | -576.42 | 6.78 |
| **188** | 10159.18 | 10900.00 | 740.82 | 6.80 |
| **731** | 16027.02 | 15000.00 | -1027.02 | 6.85 |
| **448** | 13502.35 | 14500.00 | 997.65 | 6.88 |
| **254** | 6413.26 | 5999.00 | -414.26 | 6.91 |
| **667** | 49221.63 | 52999.00 | 3777.37 | 7.13 |
| **600** | 21816.43 | 23500.00 | 1683.57 | 7.16 |
| **144** | 11969.66 | 12900.00 | 930.34 | 7.21 |
| **270** | 8263.95 | 7700.00 | -563.95 | 7.32 |
| **8** | 11581.54 | 12500.00 | 918.46 | 7.35 |
| **433** | 6977.90 | 6500.00 | -477.90 | 7.35 |
| **251** | 6978.91 | 6500.00 | -478.91 | 7.37 |
| **553** | 16392.22 | 17700.00 | 1307.78 | 7.39 |
| **725** | 15742.41 | 16999.00 | 1256.59 | 7.39 |
| **615** | 7869.47 | 8500.00 | 630.53 | 7.42 |
| **518** | 21297.65 | 19800.00 | -1497.65 | 7.56 |
| **268** | 6893.62 | 6400.00 | -493.62 | 7.71 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **71** | 15627.03 | 14500.00 | -1127.03 | 7.77 |
| **579** | 3411.53 | 3700.00 | 288.47 | 7.80 |
| **330** | 11336.71 | 12300.00 | 963.29 | 7.83 |
| **59** | 8195.34 | 7600.00 | -595.34 | 7.83 |
| **354** | 16717.08 | 15500.00 | -1217.08 | 7.85 |
| **147** | 7772.57 | 7200.00 | -572.57 | 7.95 |
| **508** | 16197.42 | 15000.00 | -1197.42 | 7.98 |
| **22** | 6102.36 | 5650.00 | -452.36 | 8.01 |
| **103** | 4488.43 | 4150.00 | -338.43 | 8.16 |
| **755** | 15885.66 | 17300.00 | 1414.34 | 8.18 |
| **198** | 17983.10 | 19600.00 | 1616.90 | 8.25 |
| **470** | 7045.72 | 6500.00 | -545.72 | 8.40 |
| **710** | 3795.06 | 3500.00 | -295.06 | 8.43 |
| **185** | 6130.96 | 6700.00 | 569.04 | 8.49 |
| **542** | 5947.51 | 6500.00 | 552.49 | 8.50 |
| **412** | 7609.28 | 7000.00 | -609.28 | 8.70 |
| **415** | 8117.44 | 8900.00 | 782.56 | 8.79 |
| **200** | 11098.18 | 10200.00 | -898.18 | 8.81 |
| **754** | 17028.45 | 18700.00 | 1671.55 | 8.94 |
| **65** | 19847.25 | 18200.00 | -1647.25 | 9.05 |
| **682** | 5454.34 | 5000.00 | -454.34 | 9.09 |
| **222** | 20662.46 | 18900.00 | -1762.46 | 9.33 |
| **89** | 11696.25 | 12900.00 | 1203.75 | 9.33 |
| **52** | 10716.84 | 9800.00 | -916.84 | 9.36 |
| **126** | 13459.92 | 12300.00 | -1159.92 | 9.43 |
| **598** | 8767.54 | 9700.00 | 932.46 | 9.61 |
| **504** | 23272.83 | 25749.75 | 2476.92 | 9.62 |
| **493** | 5484.97 | 4999.00 | -485.97 | 9.72 |
| **449** | 9495.62 | 8650.00 | -845.62 | 9.78 |
| **516** | 11858.48 | 10800.00 | -1058.48 | 9.80 |
| **261** | 53312.81 | 48535.50 | -4777.31 | 9.84 |
| **531** | 47047.70 | 52300.00 | 5252.30 | 10.04 |
| **68** | 6717.17 | 6100.00 | -617.17 | 10.12 |
| **376** | 5391.77 | 6000.00 | 608.23 | 10.14 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **538** | 17591.09 | 19600.00 | 2008.91 | 10.25 |
| **419** | 17591.09 | 19600.00 | 2008.91 | 10.25 |
| **595** | 25576.58 | 28500.00 | 2923.42 | 10.26 |
| **472** | 20367.25 | 22700.00 | 2332.75 | 10.28 |
| **386** | 7267.36 | 8100.00 | 832.64 | 10.28 |
| **473** | 1792.36 | 1999.00 | 206.64 | 10.34 |
| **14** | 8248.67 | 9200.00 | 951.33 | 10.34 |
| **123** | 5643.45 | 6300.00 | 656.55 | 10.42 |
| **142** | 6839.57 | 7650.00 | 810.43 | 10.59 |
| **257** | 41503.90 | 37500.00 | -4003.90 | 10.68 |
| **194** | 9310.69 | 8400.00 | -910.69 | 10.84 |
| **637** | 35662.26 | 40000.00 | 4337.74 | 10.84 |
| **410** | 16407.29 | 14800.00 | -1607.29 | 10.86 |
| **395** | 21303.57 | 23900.00 | 2596.43 | 10.86 |
| **213** | 15872.91 | 14300.00 | -1572.91 | 11.00 |
| **653** | 10332.54 | 9300.00 | -1032.54 | 11.10 |
| **54** | 18575.43 | 20900.00 | 2324.57 | 11.12 |
| **105** | 8882.65 | 10000.00 | 1117.35 | 11.17 |
| **597** | 7797.18 | 6999.00 | -798.18 | 11.40 |
| **15** | 10621.84 | 11999.00 | 1377.16 | 11.48 |
| **770** | 10732.07 | 9600.00 | -1132.07 | 11.79 |
| **674** | 2381.01 | 2700.00 | 318.99 | 11.81 |
| **132** | 11200.20 | 9999.00 | -1201.20 | 12.01 |
| **525** | 2373.66 | 2700.00 | 326.34 | 12.09 |
| **730** | 10696.56 | 12179.00 | 1482.44 | 12.17 |
| **506** | 3590.97 | 3200.00 | -390.97 | 12.22 |
| **278** | 53063.22 | 60500.00 | 7436.78 | 12.29 |
| **108** | 7300.05 | 6500.00 | -800.05 | 12.31 |
| **503** | 6392.70 | 7300.00 | 907.30 | 12.43 |
| **536** | 6746.43 | 6000.00 | -746.43 | 12.44 |
| **156** | 9192.06 | 10500.00 | 1307.94 | 12.46 |
| **765** | 11034.66 | 9800.00 | -1234.66 | 12.60 |
| **534** | 16939.86 | 19400.00 | 2460.14 | 12.68 |
| **50** | 2479.43 | 2200.00 | -279.43 | 12.70 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **562** | 9593.52 | 11000.00 | 1406.48 | 12.79 |
| **131** | 23967.81 | 27500.00 | 3532.19 | 12.84 |
| **157** | 7491.01 | 8600.00 | 1108.99 | 12.90 |
| **638** | 8242.62 | 7300.00 | -942.62 | 12.91 |
| **543** | 9489.56 | 10900.00 | 1410.44 | 12.94 |
| **724** | 44312.98 | 50900.00 | 6587.02 | 12.94 |
| **527** | 9829.53 | 8700.00 | -1129.53 | 12.98 |
| **610** | 9269.68 | 8200.00 | -1069.68 | 13.04 |
| **359** | 4524.77 | 3999.00 | -525.77 | 13.15 |
| **24** | 23650.15 | 20900.00 | -2750.15 | 13.16 |
| **402** | 8850.21 | 10200.00 | 1349.79 | 13.23 |
| **360** | 4770.52 | 5500.00 | 729.48 | 13.26 |
| **184** | 29479.97 | 34000.00 | 4520.03 | 13.29 |
| **324** | 8316.92 | 9600.00 | 1283.08 | 13.37 |
| **307** | 24974.57 | 28900.00 | 3925.43 | 13.58 |
| **118** | 14943.75 | 17300.00 | 2356.25 | 13.62 |
| **607** | 28498.91 | 33000.00 | 4501.09 | 13.64 |
| **332** | 14591.35 | 16900.00 | 2308.65 | 13.66 |
| **140** | 10358.58 | 12000.00 | 1641.42 | 13.68 |
| **627** | 13074.74 | 11500.00 | -1574.74 | 13.69 |
| **761** | 9097.13 | 8000.00 | -1097.13 | 13.71 |
| **207** | 15010.51 | 13200.00 | -1810.51 | 13.72 |
| **685** | 12593.00 | 14600.00 | 2007.00 | 13.75 |
| **350** | 20246.04 | 23500.00 | 3253.96 | 13.85 |
| **143** | 8768.29 | 7700.00 | -1068.29 | 13.87 |
| **318** | 39744.80 | 34900.00 | -4844.80 | 13.88 |
| **377** | 9909.62 | 8700.00 | -1209.62 | 13.90 |
| **152** | 44680.77 | 51900.00 | 7219.23 | 13.91 |
| **57** | 21942.05 | 25500.00 | 3557.95 | 13.95 |
| **277** | 12900.31 | 15000.00 | 2099.69 | 14.00 |
| **253** | 18128.32 | 15900.00 | -2228.32 | 14.01 |
| **384** | 4296.77 | 5000.00 | 703.23 | 14.06 |
| **113** | 6006.27 | 6999.00 | 992.73 | 14.18 |
| **287** | 1972.28 | 2300.00 | 327.72 | 14.25 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **650** | 8236.36 | 7200.00 | -1036.36 | 14.39 |
| **319** | 10612.51 | 12400.00 | 1787.49 | 14.42 |
| **430** | 14197.26 | 12400.00 | -1797.26 | 14.49 |
| **95** | 47013.28 | 55000.00 | 7986.72 | 14.52 |
| **757** | 12028.91 | 10500.00 | -1528.91 | 14.56 |
| **153** | 3666.31 | 3200.00 | -466.31 | 14.57 |
| **720** | 2860.97 | 3350.00 | 489.03 | 14.60 |
| **631** | 7256.86 | 8500.00 | 1243.14 | 14.63 |
| **36** | 10499.60 | 12300.00 | 1800.40 | 14.64 |
| **763** | 12383.35 | 10800.00 | -1583.35 | 14.66 |
| **344** | 11937.31 | 13995.00 | 2057.69 | 14.70 |
| **389** | 2810.71 | 2450.00 | -360.71 | 14.72 |
| **204** | 7664.32 | 9000.00 | 1335.68 | 14.84 |
| **718** | 8041.62 | 6999.00 | -1042.62 | 14.90 |
| **484** | 13334.19 | 11600.00 | -1734.19 | 14.95 |
| **258** | 13680.48 | 11900.00 | -1780.48 | 14.96 |
| **616** | 13500.69 | 15900.00 | 2399.31 | 15.09 |
| **281** | 11638.89 | 13708.50 | 2069.61 | 15.10 |
| **476** | 15700.53 | 18500.00 | 2799.47 | 15.13 |
| **716** | 11425.01 | 13500.00 | 2074.99 | 15.37 |
| **529** | 53312.81 | 63000.00 | 9687.19 | 15.38 |
| **481** | 6599.62 | 7800.00 | 1200.38 | 15.39 |
| **706** | 61639.73 | 72900.00 | 11260.27 | 15.45 |
| **558** | 7098.53 | 8400.00 | 1301.47 | 15.49 |
| **151** | 10889.39 | 12900.00 | 2010.61 | 15.59 |
| **663** | 2613.97 | 3099.00 | 485.03 | 15.65 |
| **16** | 23721.58 | 20500.00 | -3221.58 | 15.72 |
| **356** | 10133.68 | 8750.00 | -1383.68 | 15.81 |
| **297** | 22261.50 | 19200.00 | -3061.50 | 15.95 |
| **119** | 10448.50 | 8999.00 | -1449.50 | 16.11 |
| **390** | 24271.90 | 20900.00 | -3371.90 | 16.13 |
| **439** | 14926.69 | 17800.00 | 2873.31 | 16.14 |
| **441** | 11619.16 | 10000.00 | -1619.16 | 16.19 |
| **548** | 46465.02 | 39900.00 | -6565.02 | 16.45 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **460** | 50954.68 | 61000.00 | 10045.32 | 16.47 |
| **211** | 6848.88 | 8200.00 | 1351.12 | 16.48 |
| **106** | 16547.03 | 14200.00 | -2347.03 | 16.53 |
| **599** | 4590.16 | 5500.00 | 909.84 | 16.54 |
| **220** | 13348.91 | 15999.00 | 2650.09 | 16.56 |
| **226** | 11191.05 | 9600.00 | -1591.05 | 16.57 |
| **197** | 2126.49 | 2550.00 | 423.51 | 16.61 |
| **3** | 7463.96 | 6400.00 | -1063.96 | 16.62 |
| **378** | 2749.08 | 3300.00 | 550.92 | 16.69 |
| **507** | 18282.75 | 21950.00 | 3667.25 | 16.71 |
| **186** | 19071.43 | 22900.00 | 3828.57 | 16.72 |
| **100** | 32886.86 | 39500.00 | 6613.14 | 16.74 |
| **620** | 25752.19 | 30990.00 | 5237.81 | 16.90 |
| **229** | 7719.69 | 9300.00 | 1580.31 | 16.99 |
| **400** | 25986.31 | 31310.00 | 5323.69 | 17.00 |
| **336** | 8250.47 | 9950.00 | 1699.53 | 17.08 |
| **483** | 28937.50 | 34900.00 | 5962.50 | 17.08 |
| **171** | 8392.42 | 7150.00 | -1242.42 | 17.38 |
| **421** | 8033.61 | 9750.00 | 1716.39 | 17.60 |
| **643** | 8156.11 | 9900.00 | 1743.89 | 17.62 |
| **696** | 2470.61 | 3000.00 | 529.39 | 17.65 |
| **286** | 19659.97 | 16700.00 | -2959.97 | 17.72 |
| **496** | 17540.72 | 14899.00 | -2641.72 | 17.73 |
| **655** | 7814.16 | 9500.00 | 1685.84 | 17.75 |
| **407** | 21097.66 | 17900.00 | -3197.66 | 17.86 |
| **464** | 4266.27 | 5200.00 | 933.73 | 17.96 |
| **758** | 9380.46 | 7950.00 | -1430.46 | 17.99 |
| **670** | 33614.62 | 41000.00 | 7385.38 | 18.01 |
| **9** | 33614.62 | 41000.00 | 7385.38 | 18.01 |
| **263** | 9089.35 | 7700.00 | -1389.35 | 18.04 |
| **521** | 27740.81 | 23500.00 | -4240.81 | 18.05 |
| **46** | 3542.72 | 3000.00 | -542.72 | 18.09 |
| **237** | 25037.87 | 21200.00 | -3837.87 | 18.10 |
| **28** | 35991.51 | 43999.00 | 8007.49 | 18.20 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **125** | 13733.33 | 16800.00 | 3066.67 | 18.25 |
| **721** | 11431.35 | 13999.00 | 2567.65 | 18.34 |
| **749** | 6082.85 | 7450.00 | 1367.15 | 18.35 |
| **55** | 12674.57 | 10700.00 | -1974.57 | 18.45 |
| **466** | 25975.12 | 21900.00 | -4075.12 | 18.61 |
| **651** | 3026.25 | 2550.00 | -476.25 | 18.68 |
| **560** | 33172.20 | 40800.00 | 7627.80 | 18.70 |
| **247** | 13159.16 | 16200.00 | 3040.84 | 18.77 |
| **738** | 20216.67 | 24900.00 | 4683.33 | 18.81 |
| **715** | 10220.74 | 8600.00 | -1620.74 | 18.85 |
| **608** | 29127.72 | 24500.00 | -4627.72 | 18.89 |
| **509** | 5839.08 | 7200.00 | 1360.92 | 18.90 |
| **435** | 53063.22 | 65500.00 | 12436.78 | 18.99 |
| **187** | 28256.79 | 34900.00 | 6643.21 | 19.03 |
| **689** | 4524.77 | 3800.00 | -724.77 | 19.07 |
| **51** | 2543.57 | 3149.25 | 605.68 | 19.23 |
| **501** | 60493.51 | 75000.00 | 14506.49 | 19.34 |
| **292** | 8962.29 | 7500.00 | -1462.29 | 19.50 |
| **193** | 14832.32 | 12400.00 | -2432.32 | 19.62 |
| **711** | 19676.51 | 24500.00 | 4823.49 | 19.69 |
| **7** | 20349.62 | 16999.00 | -3350.62 | 19.71 |
| **569** | 1926.60 | 2400.00 | 473.40 | 19.72 |
| **453** | 8381.60 | 6999.00 | -1382.60 | 19.75 |
| **444** | 38334.67 | 32000.00 | -6334.67 | 19.80 |
| **454** | 20972.54 | 17500.00 | -3472.54 | 19.84 |
| **176** | 6732.97 | 8400.00 | 1667.03 | 19.85 |
| **240** | 10419.20 | 13000.00 | 2580.80 | 19.85 |
| **688** | 24572.28 | 20500.00 | -4072.28 | 19.86 |
| **313** | 10014.71 | 12500.00 | 2485.29 | 19.88 |
| **471** | 4209.17 | 3500.00 | -709.17 | 20.26 |
| **535** | 34678.95 | 43500.00 | 8821.05 | 20.28 |
| **79** | 18893.76 | 15700.00 | -3193.76 | 20.34 |
| **409** | 8121.22 | 10200.00 | 2078.78 | 20.38 |
| **154** | 11443.85 | 9499.00 | -1944.85 | 20.47 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **494** | 33392.40 | 42000.00 | 8607.60 | 20.49 |
| **98** | 6746.48 | 8500.00 | 1753.52 | 20.63 |
| **227** | 17251.60 | 14300.00 | -2951.60 | 20.64 |
| **115** | 6981.34 | 8800.00 | 1818.66 | 20.67 |
| **744** | 2379.58 | 3000.00 | 620.42 | 20.68 |
| **474** | 10932.17 | 13800.00 | 2867.83 | 20.78 |
| **459** | 2779.04 | 2300.00 | -479.04 | 20.83 |
| **331** | 11002.01 | 13900.00 | 2897.99 | 20.85 |
| **76** | 5144.02 | 6500.00 | 1355.98 | 20.86 |
| **244** | 7110.32 | 8990.00 | 1879.68 | 20.91 |
| **644** | 53312.81 | 67431.00 | 14118.19 | 20.94 |
| **174** | 7745.02 | 6400.00 | -1345.02 | 21.02 |
| **358** | 63921.33 | 80999.00 | 17077.67 | 21.08 |
| **425** | 3940.49 | 4999.00 | 1058.51 | 21.17 |
| **321** | 23630.85 | 19500.00 | -4130.85 | 21.18 |
| **312** | 14792.29 | 18800.00 | 4007.71 | 21.32 |
| **17** | 11770.64 | 9700.00 | -2070.64 | 21.35 |
| **517** | 9424.34 | 12000.00 | 2575.66 | 21.46 |
| **392** | 8139.48 | 6700.00 | -1439.48 | 21.48 |
| **416** | 22367.52 | 28500.00 | 6132.48 | 21.52 |
| **335** | 13004.34 | 10700.00 | -2304.34 | 21.54 |
| **45** | 8512.03 | 7000.00 | -1512.03 | 21.60 |
| **515** | 32121.20 | 41000.00 | 8878.80 | 21.66 |
| **557** | 12523.73 | 16000.00 | 3476.27 | 21.73 |
| **41** | 19845.36 | 16300.00 | -3545.36 | 21.75 |
| **13** | 13292.71 | 16999.00 | 3706.29 | 21.80 |
| **691** | 8467.17 | 6950.00 | -1517.17 | 21.83 |
| **20** | 11876.82 | 15200.00 | 3323.18 | 21.86 |
| **275** | 22547.55 | 18500.00 | -4047.55 | 21.88 |
| **120** | 50728.89 | 65000.00 | 14271.11 | 21.96 |
| **647** | 2625.10 | 2150.00 | -475.10 | 22.10 |
| **524** | 5863.41 | 4800.00 | -1063.41 | 22.15 |
| **547** | 9175.06 | 7499.00 | -1676.06 | 22.35 |
| **747** | 8453.28 | 10900.00 | 2446.72 | 22.45 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **295** | 5350.91 | 6900.00 | 1549.09 | 22.45 |
| **80** | 6819.44 | 8800.00 | 1980.56 | 22.51 |
| **661** | 10303.21 | 13300.00 | 2996.79 | 22.53 |
| **537** | 9139.04 | 11800.00 | 2660.96 | 22.55 |
| **626** | 18621.12 | 24100.00 | 5478.88 | 22.73 |
| **576** | 15954.72 | 12999.00 | -2955.72 | 22.74 |
| **499** | 12270.01 | 15900.00 | 3629.99 | 22.83 |
| **218** | 11490.44 | 14900.00 | 3409.56 | 22.88 |
| **458** | 14136.01 | 11500.00 | -2636.01 | 22.92 |
| **138** | 11095.57 | 8999.00 | -2096.57 | 23.30 |
| **320** | 10350.25 | 13500.00 | 3149.75 | 23.33 |
| **606** | 20680.40 | 27000.00 | 6319.60 | 23.41 |
| **709** | 34446.96 | 27900.00 | -6546.96 | 23.47 |
| **75** | 4209.17 | 5500.00 | 1290.83 | 23.47 |
| **155** | 3086.38 | 2499.00 | -587.38 | 23.50 |
| **602** | 16427.12 | 13300.00 | -3127.12 | 23.51 |
| **736** | 53312.81 | 43163.25 | -10149.56 | 23.51 |
| **495** | 3747.21 | 4900.00 | 1152.79 | 23.53 |
| **399** | 14410.15 | 18932.55 | 4522.40 | 23.89 |
| **771** | 13922.45 | 18300.00 | 4377.55 | 23.92 |
| **662** | 9232.79 | 7450.00 | -1782.79 | 23.93 |
| **634** | 18165.16 | 23900.00 | 5734.84 | 24.00 |
| **4** | 11353.49 | 9150.00 | -2203.49 | 24.08 |
| **580** | 6205.48 | 5000.00 | -1205.48 | 24.11 |
| **687** | 11004.05 | 14500.00 | 3495.95 | 24.11 |
| **305** | 10505.98 | 13900.00 | 3394.02 | 24.42 |
| **206** | 19920.63 | 16000.00 | -3920.63 | 24.50 |
| **659** | 3621.08 | 4799.00 | 1177.92 | 24.55 |
| **178** | 17316.59 | 13900.00 | -3416.59 | 24.58 |
| **692** | 8471.55 | 6799.00 | -1672.55 | 24.60 |
| **479** | 36883.77 | 49000.00 | 12116.23 | 24.73 |
| **209** | 8242.62 | 6600.00 | -1642.62 | 24.89 |
| **773** | 13491.16 | 10800.00 | -2691.16 | 24.92 |
| **594** | 3001.37 | 4000.00 | 998.63 | 24.97 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **1** | 3499.26 | 2800.00 | -699.26 | 24.97 |
| **625** | 23485.45 | 18777.00 | -4708.45 | 25.08 |
| **370** | 10835.95 | 14500.00 | 3664.05 | 25.27 |
| **83** | 6013.05 | 4800.00 | -1213.05 | 25.27 |
| **443** | 3735.04 | 4999.00 | 1263.96 | 25.28 |
| **158** | 15623.39 | 20999.00 | 5375.61 | 25.60 |
| **519** | 8667.12 | 6900.00 | -1767.12 | 25.61 |
| **177** | 8916.19 | 11999.00 | 3082.81 | 25.69 |
| **361** | 8688.92 | 11700.00 | 3011.08 | 25.74 |
| **482** | 12027.56 | 16200.00 | 4172.44 | 25.76 |
| **705** | 12976.15 | 17500.00 | 4523.85 | 25.85 |
| **135** | 44407.14 | 59999.00 | 15591.86 | 25.99 |
| **264** | 31059.18 | 42000.00 | 10940.82 | 26.05 |
| **372** | 8110.06 | 10990.00 | 2879.94 | 26.21 |
| **672** | 5679.39 | 4500.00 | -1179.39 | 26.21 |
| **699** | 4255.62 | 5800.00 | 1544.38 | 26.63 |
| **306** | 3164.66 | 2499.00 | -665.66 | 26.64 |
| **279** | 3154.48 | 4300.00 | 1145.52 | 26.64 |
| **145** | 4307.68 | 3400.00 | -907.68 | 26.70 |
| **681** | 5195.69 | 4100.00 | -1095.69 | 26.72 |
| **175** | 34863.53 | 47600.00 | 12736.47 | 26.76 |
| **440** | 8638.63 | 11800.00 | 3161.37 | 26.79 |
| **683** | 32771.45 | 44800.00 | 12028.55 | 26.85 |
| **592** | 28154.84 | 38500.00 | 10345.16 | 26.87 |
| **429** | 67952.07 | 53500.00 | -14452.07 | 27.01 |
| **294** | 13471.84 | 10600.00 | -2871.84 | 27.09 |
| **623** | 23227.15 | 31900.00 | 8672.85 | 27.19 |
| **107** | 7131.92 | 9800.00 | 2668.08 | 27.23 |
| **86** | 8071.70 | 11100.00 | 3028.30 | 27.28 |
| **37** | 26478.15 | 20800.00 | -5678.15 | 27.30 |
| **92** | 7116.68 | 9800.00 | 2683.32 | 27.38 |
| **231** | 9312.21 | 7300.00 | -2012.21 | 27.56 |
| **62** | 10848.85 | 8500.00 | -2348.85 | 27.63 |
| **572** | 10597.72 | 8300.00 | -2297.72 | 27.68 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **568** | 7940.44 | 11000.00 | 3059.56 | 27.81 |
| **202** | 5040.46 | 7000.00 | 1959.54 | 27.99 |
| **327** | 10716.00 | 14900.00 | 4184.00 | 28.08 |
| **310** | 9205.10 | 12800.00 | 3594.90 | 28.09 |
| **461** | 31656.01 | 24700.00 | -6956.01 | 28.16 |
| **613** | 13853.36 | 10800.00 | -3053.36 | 28.27 |
| **323** | 7098.53 | 9900.00 | 2801.47 | 28.30 |
| **752** | 8591.66 | 12000.00 | 3408.34 | 28.40 |
| **348** | 12201.29 | 9500.00 | -2701.29 | 28.43 |
| **424** | 50074.56 | 69990.00 | 19915.44 | 28.45 |
| **584** | 11432.63 | 8900.00 | -2532.63 | 28.46 |
| **99** | 6437.21 | 8999.00 | 2561.79 | 28.47 |
| **149** | 8351.34 | 6500.00 | -1851.34 | 28.48 |
| **58** | 35336.33 | 49500.00 | 14163.67 | 28.61 |
| **423** | 74232.55 | 103999.00 | 29766.45 | 28.62 |
| **726** | 45664.45 | 35500.00 | -10164.45 | 28.63 |
| **614** | 19298.61 | 14999.00 | -4299.61 | 28.67 |
| **129** | 11580.54 | 9000.00 | -2580.54 | 28.67 |
| **303** | 23496.36 | 33000.00 | 9503.64 | 28.80 |
| **21** | 31557.80 | 24500.00 | -7057.80 | 28.81 |
| **767** | 24323.48 | 18800.00 | -5523.48 | 29.38 |
| **192** | 18285.07 | 25900.00 | 7614.93 | 29.40 |
| **700** | 25640.67 | 19800.00 | -5840.67 | 29.50 |
| **552** | 5983.44 | 8500.00 | 2516.56 | 29.61 |
| **500** | 8973.16 | 12750.00 | 3776.84 | 29.62 |
| **734** | 23336.06 | 33200.00 | 9863.94 | 29.71 |
| **233** | 6110.94 | 8700.00 | 2589.06 | 29.76 |
| **582** | 7724.17 | 11000.00 | 3275.83 | 29.78 |
| **201** | 10094.58 | 7777.00 | -2317.58 | 29.80 |
| **296** | 21690.99 | 30900.00 | 9209.01 | 29.80 |
| **523** | 10809.48 | 15400.00 | 4590.52 | 29.81 |
| **357** | 11018.79 | 15700.00 | 4681.21 | 29.82 |
| **190** | 13377.99 | 10300.00 | -3077.99 | 29.88 |
| **234** | 12731.26 | 9800.00 | -2931.26 | 29.91 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| 695 | 11178.46 | 8600.00 | -2578.46 | 29.98 |
| 364 | 21681.00 | 30999.00 | 9318.00 | 30.06 |
| 362 | 27293.13 | 39040.00 | 11746.87 | 30.09 |
| 309 | 6510.80 | 5000.00 | -1510.80 | 30.22 |
| 146 | 8298.32 | 11900.00 | 3601.68 | 30.27 |
| 373 | 10821.19 | 8299.00 | -2522.19 | 30.39 |
| 513 | 5346.97 | 4100.00 | -1246.97 | 30.41 |
| 12 | 5484.02 | 7900.00 | 2415.98 | 30.58 |
| 352 | 5238.05 | 4000.00 | -1238.05 | 30.95 |
| 181 | 17597.10 | 25500.00 | 7902.90 | 30.99 |
| 128 | 7727.53 | 11200.00 | 3472.47 | 31.00 |
| 753 | 6358.55 | 4850.00 | -1508.55 | 31.10 |
| 671 | 2886.35 | 4200.00 | 1313.65 | 31.28 |
| 764 | 14049.76 | 10700.00 | -3349.76 | 31.31 |
| 216 | 12974.89 | 18900.00 | 5925.11 | 31.35 |
| 260 | 6858.89 | 9999.00 | 3140.11 | 31.40 |
| 124 | 15691.29 | 22900.00 | 7208.71 | 31.48 |
| 214 | 2393.34 | 3500.00 | 1106.66 | 31.62 |
| 405 | 6425.00 | 9400.00 | 2975.00 | 31.65 |
| 70 | 50206.83 | 73500.00 | 23293.17 | 31.69 |
| 596 | 25685.67 | 19500.00 | -6185.67 | 31.72 |
| 351 | 25951.31 | 19692.08 | -6259.23 | 31.79 |
| 489 | 28910.96 | 42500.00 | 13589.04 | 31.97 |
| 38 | 9919.19 | 14600.00 | 4680.81 | 32.06 |
| 274 | 9113.05 | 6900.00 | -2213.05 | 32.07 |
| 374 | 24427.14 | 36000.00 | 11572.86 | 32.15 |
| 302 | 4738.89 | 7000.00 | 2261.11 | 32.30 |
| 148 | 25040.42 | 37000.00 | 11959.58 | 32.32 |
| 164 | 18403.35 | 13893.75 | -4509.60 | 32.46 |
| 420 | 14576.33 | 11000.00 | -3576.33 | 32.51 |
| 555 | 6610.04 | 9800.00 | 3189.96 | 32.55 |
| 505 | 10105.70 | 15000.00 | 4894.30 | 32.63 |
| 250 | 5041.20 | 3800.00 | -1241.20 | 32.66 |
| 491 | 6503.62 | 4900.00 | -1603.62 | 32.73 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **426** | 13542.79 | 10200.00 | -3342.79 | 32.77 |
| **84** | 4364.08 | 6500.00 | 2135.92 | 32.86 |
| **136** | 12006.66 | 17900.00 | 5893.34 | 32.92 |
| **139** | 7712.65 | 11500.00 | 3787.35 | 32.93 |
| **618** | 31251.89 | 23500.00 | -7751.89 | 32.99 |
| **702** | 7011.74 | 10500.00 | 3488.26 | 33.22 |
| **728** | 5792.10 | 8700.00 | 2907.90 | 33.42 |
| **751** | 8493.04 | 12800.00 | 4306.96 | 33.65 |
| **498** | 7950.27 | 12000.00 | 4049.73 | 33.75 |
| **605** | 8940.39 | 13500.00 | 4559.61 | 33.77 |
| **678** | 4569.15 | 6900.00 | 2330.85 | 33.78 |
| **205** | 13041.39 | 19700.00 | 6658.61 | 33.80 |
| **265** | 20163.48 | 30500.00 | 10336.52 | 33.89 |
| **403** | 2839.03 | 4300.00 | 1460.97 | 33.98 |
| **291** | 20433.23 | 30999.00 | 10565.77 | 34.08 |
| **74** | 1746.19 | 2650.00 | 903.81 | 34.11 |
| **741** | 19855.31 | 14800.00 | -5055.31 | 34.16 |
| **170** | 19741.65 | 29999.00 | 10257.35 | 34.19 |
| **203** | 65613.31 | 99999.00 | 34385.69 | 34.39 |
| **289** | 7178.20 | 10950.00 | 3771.80 | 34.45 |
| **90** | 9143.17 | 6800.00 | -2343.17 | 34.46 |
| **563** | 2285.24 | 3500.00 | 1214.76 | 34.71 |
| **112** | 22835.48 | 16950.00 | -5885.48 | 34.72 |
| **189** | 10786.38 | 8000.00 | -2786.38 | 34.83 |
| **669** | 11477.47 | 17650.00 | 6172.53 | 34.97 |
| **163** | 23972.56 | 36900.00 | 12927.44 | 35.03 |
| **97** | 3648.96 | 2700.00 | -948.96 | 35.15 |
| **690** | 11706.93 | 18100.00 | 6393.07 | 35.32 |
| **117** | 7178.20 | 11100.00 | 3921.80 | 35.33 |
| **673** | 22213.53 | 34500.00 | 12286.47 | 35.61 |
| **739** | 22213.53 | 34500.00 | 12286.47 | 35.61 |
| **341** | 12897.92 | 9500.00 | -3397.92 | 35.77 |
| **159** | 12904.68 | 9500.00 | -3404.68 | 35.84 |
| **64** | 9611.80 | 15000.00 | 5388.20 | 35.92 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| 300 | 20236.85 | 31600.00 | 11363.15 | 35.96 |
| 269 | 4151.02 | 6500.00 | 2348.98 | 36.14 |
| 345 | 28951.20 | 45500.00 | 16548.80 | 36.37 |
| 442 | 25951.31 | 18988.13 | -6963.18 | 36.67 |
| 475 | 5570.46 | 8800.00 | 3229.54 | 36.70 |
| 355 | 22454.74 | 35500.00 | 13045.26 | 36.75 |
| 246 | 41711.18 | 30500.00 | -11211.18 | 36.76 |
| 111 | 3152.62 | 4999.00 | 1846.38 | 36.93 |
| 490 | 8902.52 | 6500.00 | -2402.52 | 36.96 |
| 748 | 48699.98 | 77500.00 | 28800.02 | 37.16 |
| 457 | 16821.54 | 26800.00 | 9978.46 | 37.23 |
| 104 | 5521.28 | 8800.00 | 3278.72 | 37.26 |
| 743 | 5268.17 | 8400.00 | 3131.83 | 37.28 |
| 469 | 65825.25 | 104999.00 | 39173.75 | 37.31 |
| 32 | 12459.85 | 19900.00 | 7440.15 | 37.39 |
| 666 | 65723.62 | 104999.00 | 39275.38 | 37.41 |
| 401 | 9070.23 | 14500.00 | 5429.77 | 37.45 |
| 586 | 9348.90 | 6800.00 | -2548.90 | 37.48 |
| 541 | 9369.66 | 15000.00 | 5630.34 | 37.54 |
| 160 | 32954.19 | 52777.00 | 19822.81 | 37.56 |
| 652 | 17195.57 | 12500.00 | -4695.57 | 37.56 |
| 583 | 3425.23 | 5500.00 | 2074.77 | 37.72 |
| 589 | 3173.48 | 5100.00 | 1926.52 | 37.77 |
| 735 | 10337.11 | 7500.00 | -2837.11 | 37.83 |
| 375 | 65742.57 | 105999.00 | 40256.43 | 37.98 |
| 217 | 8661.95 | 13999.00 | 5337.05 | 38.12 |
| 29 | 26062.23 | 42500.00 | 16437.77 | 38.68 |
| 283 | 7630.64 | 5500.00 | -2130.64 | 38.74 |
| 418 | 11378.43 | 18600.00 | 7221.57 | 38.83 |
| 137 | 66032.76 | 107999.00 | 41966.24 | 38.86 |
| 567 | 7279.97 | 11950.00 | 4670.03 | 39.08 |
| 73 | 18577.88 | 30500.00 | 11922.12 | 39.09 |
| 488 | 4043.74 | 2900.00 | -1143.74 | 39.44 |
| 26 | 2178.94 | 3600.00 | 1421.06 | 39.47 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **42** | 23286.27 | 38500.00 | 15213.73 | 39.52 |
| **468** | 12558.90 | 8999.00 | -3559.90 | 39.56 |
| **241** | 18432.63 | 30500.00 | 12067.37 | 39.57 |
| **325** | 9057.77 | 14999.00 | 5941.23 | 39.61 |
| **675** | 13476.90 | 22500.00 | 9023.10 | 40.10 |
| **737** | 37704.71 | 26900.00 | -10804.71 | 40.17 |
| **346** | 32945.80 | 23500.00 | -9445.80 | 40.19 |
| **445** | 7712.65 | 5500.00 | -2212.65 | 40.23 |
| **266** | 65742.57 | 109999.00 | 44256.43 | 40.23 |
| **478** | 36764.38 | 62000.00 | 25235.62 | 40.70 |
| **48** | 77403.06 | 55000.00 | -22403.06 | 40.73 |
| **712** | 8530.54 | 14500.00 | 5969.46 | 41.17 |
| **587** | 15259.81 | 26000.00 | 10740.19 | 41.31 |
| **431** | 16253.48 | 11500.00 | -4753.48 | 41.33 |
| **141** | 1815.28 | 3100.00 | 1284.72 | 41.44 |
| **363** | 44193.38 | 75500.00 | 31306.62 | 41.47 |
| **25** | 45272.25 | 31990.00 | -13282.25 | 41.52 |
| **230** | 9198.08 | 15800.00 | 6601.92 | 41.78 |
| **34** | 19854.79 | 14000.00 | -5854.79 | 41.82 |
| **697** | 10031.96 | 17300.00 | 7268.04 | 42.01 |
| **371** | 9661.78 | 6800.00 | -2861.78 | 42.09 |
| **628** | 2539.57 | 4400.00 | 1860.43 | 42.28 |
| **388** | 11474.22 | 19999.00 | 8524.78 | 42.63 |
| **316** | 65887.51 | 114999.00 | 49111.49 | 42.71 |
| **199** | 2784.46 | 1950.00 | -834.46 | 42.79 |
| **514** | 15015.22 | 10500.00 | -4515.22 | 43.00 |
| **210** | 31216.83 | 55000.00 | 23783.17 | 43.24 |
| **337** | 5374.74 | 3750.00 | -1624.74 | 43.33 |
| **10** | 7241.07 | 12800.00 | 5558.93 | 43.43 |
| **646** | 3450.69 | 6100.00 | 2649.31 | 43.43 |
| **39** | 16724.45 | 29600.00 | 12875.55 | 43.50 |
| **322** | 65210.35 | 115800.00 | 50589.65 | 43.69 |
| **645** | 10740.09 | 19100.00 | 8359.91 | 43.77 |
| **413** | 12234.68 | 8500.00 | -3734.68 | 43.94 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **640** | 14402.34 | 10000.00 | -4402.34 | 44.02 |
| **88** | 10807.78 | 7500.00 | -3307.78 | 44.10 |
| **585** | 39074.36 | 69999.00 | 30924.64 | 44.18 |
| **590** | 14512.61 | 25999.00 | 11486.39 | 44.18 |
| **66** | 3316.44 | 2300.00 | -1016.44 | 44.19 |
| **588** | 12255.45 | 22000.00 | 9744.55 | 44.29 |
| **539** | 6781.86 | 4700.00 | -2081.86 | 44.29 |
| **756** | 3752.54 | 2600.00 | -1152.54 | 44.33 |
| **428** | 2389.14 | 1650.00 | -739.14 | 44.80 |
| **285** | 34761.74 | 24000.00 | -10761.74 | 44.84 |
| **566** | 11881.05 | 8200.00 | -3681.05 | 44.89 |
| **510** | 1873.03 | 3400.00 | 1526.97 | 44.91 |
| **167** | 13212.52 | 23999.00 | 10786.48 | 44.95 |
| **511** | 17282.06 | 11900.00 | -5382.06 | 45.23 |
| **522** | 17011.43 | 11700.00 | -5311.43 | 45.40 |
| **349** | 2020.25 | 3700.00 | 1679.75 | 45.40 |
| **684** | 18942.34 | 12999.00 | -5943.34 | 45.72 |
| **35** | 11337.51 | 20900.00 | 9562.49 | 45.75 |
| **365** | 10547.82 | 19500.00 | 8952.18 | 45.91 |
| **745** | 6421.18 | 4400.00 | -2021.18 | 45.94 |
| **334** | 12414.72 | 23000.00 | 10585.28 | 46.02 |
| **255** | 24796.09 | 46000.00 | 21203.91 | 46.10 |
| **717** | 4172.84 | 7750.00 | 3577.16 | 46.16 |
| **191** | 15956.95 | 29999.00 | 14042.05 | 46.81 |
| **575** | 1858.34 | 3500.00 | 1641.66 | 46.90 |
| **169** | 8522.39 | 5800.00 | -2722.39 | 46.94 |
| **385** | 13235.55 | 8999.00 | -4236.55 | 47.08 |
| **393** | 13600.39 | 25800.00 | 12199.61 | 47.29 |
| **654** | 4032.10 | 7650.00 | 3617.90 | 47.29 |
| **262** | 12414.41 | 8420.00 | -3994.41 | 47.44 |
| **617** | 2573.48 | 4900.00 | 2326.52 | 47.48 |
| **342** | 14572.70 | 27900.00 | 13327.30 | 47.77 |
| **512** | 15144.28 | 29000.00 | 13855.72 | 47.78 |
| **6** | 20159.19 | 38888.00 | 18728.81 | 48.16 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **44** | 7241.07 | 14000.00 | 6758.93 | 48.28 |
| **564** | 25682.58 | 17300.00 | -8382.58 | 48.45 |
| **658** | 10563.39 | 20500.00 | 9936.61 | 48.47 |
| **304** | 6546.81 | 4400.00 | -2146.81 | 48.79 |
| **768** | 38260.36 | 75555.00 | 37294.64 | 49.36 |
| **338** | 38260.36 | 75555.00 | 37294.64 | 49.36 |
| **760** | 13443.32 | 9000.00 | -4443.32 | 49.37 |
| **179** | 14933.54 | 29500.00 | 14566.46 | 49.38 |
| **486** | 13884.26 | 27800.00 | 13915.74 | 50.06 |
| **604** | 31373.47 | 20859.15 | -10514.32 | 50.41 |
| **183** | 47014.52 | 95000.00 | 47985.48 | 50.51 |
| **121** | 41075.73 | 27200.00 | -13875.73 | 51.01 |
| **759** | 10125.27 | 6700.00 | -3425.27 | 51.12 |
| **343** | 60603.06 | 124000.00 | 63396.94 | 51.13 |
| **414** | 14363.76 | 9500.00 | -4863.76 | 51.20 |
| **391** | 25302.43 | 52000.00 | 26697.57 | 51.34 |
| **249** | 7391.67 | 15200.00 | 7808.33 | 51.37 |
| **353** | 9753.44 | 20400.00 | 10646.56 | 52.19 |
| **243** | 32925.60 | 69500.00 | 36574.40 | 52.63 |
| **33** | 22941.80 | 15000.00 | -7941.80 | 52.95 |
| **223** | 6199.59 | 13200.00 | 7000.41 | 53.03 |
| **703** | 5597.06 | 11999.00 | 6401.94 | 53.35 |
| **411** | 13409.47 | 8700.00 | -4709.47 | 54.13 |
| **288** | 6635.73 | 4300.00 | -2335.73 | 54.32 |
| **573** | 33707.27 | 73900.00 | 40192.73 | 54.39 |
| **732** | 19309.90 | 12500.00 | -6809.90 | 54.48 |
| **456** | 19624.73 | 12700.00 | -6924.73 | 54.53 |
| **383** | 13757.58 | 8900.00 | -4857.58 | 54.58 |
| **544** | 12354.92 | 27500.00 | 15145.08 | 55.07 |
| **530** | 17714.62 | 11400.00 | -6314.62 | 55.39 |
| **551** | 11555.80 | 7400.00 | -4155.80 | 56.16 |
| **248** | 22011.47 | 13999.00 | -8012.47 | 57.24 |
| **245** | 15457.91 | 9800.00 | -5657.91 | 57.73 |
| **339** | 11099.09 | 6999.00 | -4100.09 | 58.58 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **656** | 35039.40 | 85555.00 | 50515.60 | 59.04 |
| **219** | 34376.16 | 85000.00 | 50623.84 | 59.56 |
| **520** | 16773.62 | 10500.00 | -6273.62 | 59.75 |
| **772** | 27487.75 | 68500.00 | 41012.25 | 59.87 |
| **701** | 7996.00 | 4999.00 | -2997.00 | 59.95 |
| **215** | 5284.06 | 3300.00 | -1984.06 | 60.12 |
| **447** | 34863.53 | 87777.00 | 52913.47 | 60.28 |
| **180** | 19701.70 | 49999.00 | 30297.30 | 60.60 |
| **722** | 30409.09 | 18900.00 | -11509.09 | 60.89 |
| **577** | 14486.38 | 9000.00 | -5486.38 | 60.96 |
| **733** | 11601.33 | 7200.00 | -4401.33 | 61.13 |
| **487** | 10824.89 | 6700.00 | -4124.89 | 61.57 |
| **408** | 65448.37 | 40500.00 | -24948.37 | 61.60 |
| **63** | 5351.86 | 3300.00 | -2051.86 | 62.18 |
| **679** | 25864.65 | 69990.00 | 44125.35 | 63.05 |
| **228** | 15682.82 | 9600.00 | -6082.82 | 63.36 |
| **284** | 40765.33 | 112000.00 | 71234.67 | 63.60 |
| **232** | 1320.56 | 3700.00 | 2379.44 | 64.31 |
| **417** | 3795.06 | 2300.00 | -1495.06 | 65.00 |
| **122** | 19335.34 | 55555.00 | 36219.66 | 65.20 |
| **238** | 26736.21 | 16100.00 | -10636.21 | 66.06 |
| **347** | 26441.03 | 15900.00 | -10541.03 | 66.30 |
| **276** | 5002.39 | 2999.00 | -2003.39 | 66.80 |
| **93** | 18317.20 | 10974.21 | -7342.99 | 66.91 |
| **436** | 40893.84 | 126000.00 | 85106.16 | 67.54 |
| **621** | 38473.71 | 119000.00 | 80526.29 | 67.67 |
| **221** | 34863.53 | 109999.00 | 75135.47 | 68.31 |
| **455** | 10946.44 | 6500.00 | -4446.44 | 68.41 |
| **578** | 3205.38 | 1900.00 | -1305.38 | 68.70 |
| **624** | 16924.74 | 9999.00 | -6925.74 | 69.26 |
| **326** | 9356.43 | 5500.00 | -3856.43 | 70.12 |
| **477** | 15374.08 | 8900.00 | -6474.08 | 72.74 |
| **633** | 11056.52 | 6400.00 | -4656.52 | 72.76 |
| **69** | 2440.90 | 1400.00 | -1040.90 | 74.35 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **224** | 3939.91 | 15500.00 | 11560.09 | 74.58 |
| **56** | 15380.04 | 8800.00 | -6580.04 | 74.77 |
| **463** | 35093.27 | 19990.00 | -15103.27 | 75.55 |
| **668** | 8270.38 | 4700.00 | -3570.38 | 75.97 |
| **225** | 6916.13 | 3900.00 | -3016.13 | 77.34 |
| **546** | 6746.48 | 3799.00 | -2947.48 | 77.59 |
| **27** | 2555.02 | 11600.00 | 9044.98 | 77.97 |
| **746** | 13355.11 | 7500.00 | -5855.11 | 78.07 |
| **271** | 3948.74 | 2200.00 | -1748.74 | 79.49 |
| **236** | 9024.05 | 4900.00 | -4124.05 | 84.16 |
| **166** | 15292.10 | 8300.00 | -6992.10 | 84.24 |
| **67** | 7621.25 | 4100.00 | -3521.25 | 85.88 |
| **432** | 2975.40 | 1600.00 | -1375.40 | 85.96 |
| **550** | 14915.98 | 7900.00 | -7015.98 | 88.81 |
| **208** | 5716.14 | 3000.00 | -2716.14 | 90.54 |
| **116** | 15819.50 | 8300.00 | -7519.50 | 90.60 |
| **366** | 13420.15 | 6999.00 | -6421.15 | 91.74 |
| **686** | 23576.99 | 12000.00 | -11576.99 | 96.47 |
| **404** | 8648.80 | 4400.00 | -4248.80 | 96.56 |
| **30** | 15559.43 | 7900.00 | -7659.43 | 96.95 |
| **713** | 4004.65 | 2000.00 | -2004.65 | 100.23 |
| **611** | 27225.34 | 13500.00 | -13725.34 | 101.67 |
| **427** | 5462.52 | 2700.00 | -2762.52 | 102.32 |
| **694** | 8131.92 | 3999.00 | -4132.92 | 103.35 |
| **727** | 10810.62 | 5000.00 | -5810.62 | 116.21 |
| **397** | 7973.87 | 3650.00 | -4323.87 | 118.46 |
| **196** | 5948.04 | 2600.00 | -3348.04 | 128.77 |
| **704** | 6891.99 | 3000.00 | -3891.99 | 129.73 |
| **571** | 26331.41 | 11200.00 | -15131.41 | 135.10 |
| **680** | 21493.44 | 9000.00 | -12493.44 | 138.82 |
| **465** | 6537.82 | 2700.00 | -3837.82 | 142.14 |
| **81** | 12891.96 | 5300.00 | -7591.96 | 143.24 |
| **554** | 8340.94 | 3350.00 | -4990.94 | 148.98 |
| **609** | 3040.77 | 1200.00 | -1840.77 | 153.40 |

| | Prediction | Target | Residual | Difference% |
|---|---|---|---|---|
| **677** | 9631.08 | 3799.00 | -5832.08 | 153.52 |
| **161** | 13967.55 | 5500.00 | -8467.55 | 153.96 |
| **635** | 3818.71 | 1450.00 | -2368.71 | 163.36 |
| **660** | 7809.13 | 2899.00 | -4910.13 | 169.37 |
| **664** | 4590.49 | 1700.00 | -2890.49 | 170.03 |
| **82** | 7320.42 | 2600.00 | -4720.42 | 181.55 |
| **282** | 12261.19 | 4100.00 | -8161.19 | 199.05 |
| **2** | 7553.29 | 2500.00 | -5053.29 | 202.13 |
| **729** | 9817.06 | 3200.00 | -6617.06 | 206.78 |
| **492** | 7161.67 | 2200.00 | -4961.67 | 225.53 |
| **382** | 7918.89 | 2400.00 | -5518.89 | 229.95 |
| **502** | 9984.87 | 3000.00 | -6984.87 | 232.83 |
| **750** | 10381.62 | 3000.00 | -7381.62 | 246.05 |
| **434** | 8843.22 | 2500.00 | -6343.22 | 253.73 |
| **182** | 10123.04 | 2800.00 | -7323.04 | 261.54 |
| **396** | 16165.79 | 4400.00 | -11765.79 | 267.40 |
| **298** | 17937.36 | 4800.00 | -13137.36 | 273.69 |
| **629** | 7319.77 | 1850.00 | -5469.77 | 295.66 |
| **619** | 16095.32 | 3600.00 | -12495.32 | 347.09 |
| **0** | 10685.50 | 2300.00 | -8385.50 | 364.59 |
| **485** | 9664.46 | 1900.00 | -7764.46 | 408.66 |
| **657** | 32481.05 | 6000.00 | -26481.05 | 441.35 |
| **162** | 9954.42 | 1800.00 | -8154.42 | 453.02 |
| **451** | 35956.50 | 6500.00 | -29456.50 | 453.18 |
| **532** | 10019.90 | 1800.00 | -8219.90 | 456.66 |
| **639** | 30628.28 | 4999.00 | -25629.28 | 512.69 |