In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Ridge, RidgeCV, Lasso
from sklearn.preprocessing import StandardScaler
```

In [3]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\fiat500_VehicleSelection_Dataset.csv")
df
```

Out[3]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | lon |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | lounge | 51 | 882 | 25000 | 1 | 44.907242 | 8.611560 |
| **1** | 2 | pop | 51 | 1186 | 32500 | 1 | 45.666359 | 12.241890 |
| **2** | 3 | sport | 74 | 4658 | 142228 | 1 | 45.503300 | 11.417840 |
| **3** | 4 | lounge | 51 | 2739 | 160000 | 1 | 40.633171 | 17.634609 |
| **4** | 5 | pop | 73 | 3074 | 106880 | 1 | 41.903221 | 12.495650 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **1533** | 1534 | sport | 51 | 3712 | 115280 | 1 | 45.069679 | 7.704920 |
| **1534** | 1535 | lounge | 74 | 3835 | 112000 | 1 | 45.845692 | 8.666870 |
| **1535** | 1536 | pop | 51 | 2223 | 60457 | 1 | 45.481541 | 9.413480 |
| **1536** | 1537 | lounge | 51 | 2557 | 80750 | 1 | 45.000702 | 7.682270 |
| **1537** | 1538 | pop | 51 | 1766 | 54276 | 1 | 40.323410 | 17.568270 |

1538 rows × 9 columns

In [4]:
```
df = df[['engine_power','price']]
df.columns=['Eng','pri']
```

In [5]:
```
df.head()
```

Out[5]:

| | Eng | pri |
|---|---|---|
| **0** | 51 | 8900 |
| **1** | 51 | 8800 |
| **2** | 74 | 4200 |
| **3** | 51 | 6000 |
| **4** | 73 | 5700 |

In [6]:
```
df.tail()
```

Out[6]:

| | Eng | pri |
|---|---|---|
| **1533** | 51 | 5200 |
| **1534** | 74 | 4600 |
| **1535** | 51 | 7500 |
| **1536** | 51 | 5990 |
| **1537** | 51 | 7900 |

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1538 entries, 0 to 1537
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Eng     1538 non-null   int64
 1   pri     1538 non-null   int64
dtypes: int64(2)
memory usage: 24.2 KB
```

In [8]:
```
df.describe()
```

Out[8]:

| | Eng | pri |
|---|---|---|
| **count** | 1538.000000 | 1538.000000 |
| **mean** | 51.904421 | 8576.003901 |
| **std** | 3.988023 | 1939.958641 |
| **min** | 51.000000 | 2500.000000 |
| **25%** | 51.000000 | 7122.500000 |
| **50%** | 51.000000 | 9000.000000 |
| **75%** | 51.000000 | 10000.000000 |
| **max** | 77.000000 | 11100.000000 |

In [9]:
```
df.fillna(method='ffill')
```

Out[9]:

| | Eng | pri |
|---|---|---|
| **0** | 51 | 8900 |
| **1** | 51 | 8800 |
| **2** | 74 | 4200 |
| **3** | 51 | 6000 |
| **4** | 73 | 5700 |
| **...** | ... | ... |
| **1533** | 51 | 5200 |
| **1534** | 74 | 4600 |
| **1535** | 51 | 7500 |
| **1536** | 51 | 5990 |
| **1537** | 51 | 7900 |

1538 rows × 2 columns

In [10]:
```
x=np.array(df['Eng']).reshape(-1,1)
y=np.array(df['pri']).reshape(-1,1)
```

In [11]:
```
df.dropna(inplace=True)
```

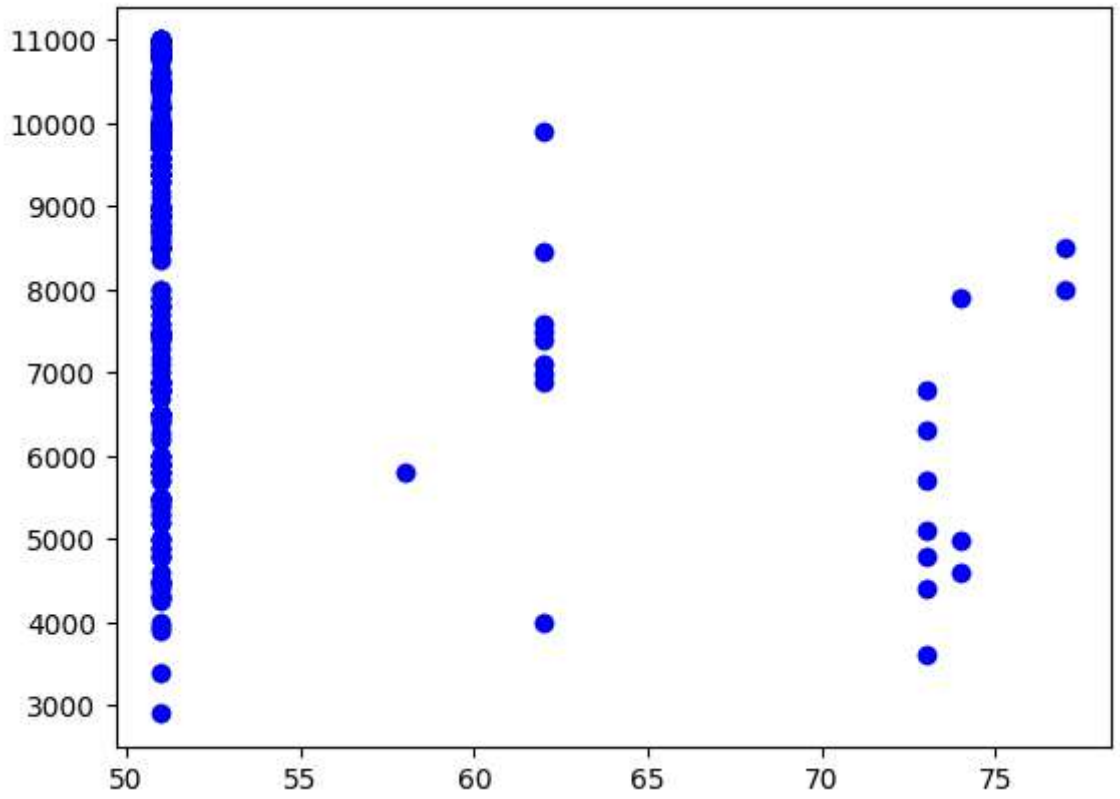
```
C:\Users\user\AppData\Local\Temp\ipykernel_15664\1379821321.py:1: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  df.dropna(inplace=True)
```

In [12]:
```
X_train,X_test,y_train,y_test = train_test_split(x, y, test_size = 0.25)
regr = LinearRegression()
regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

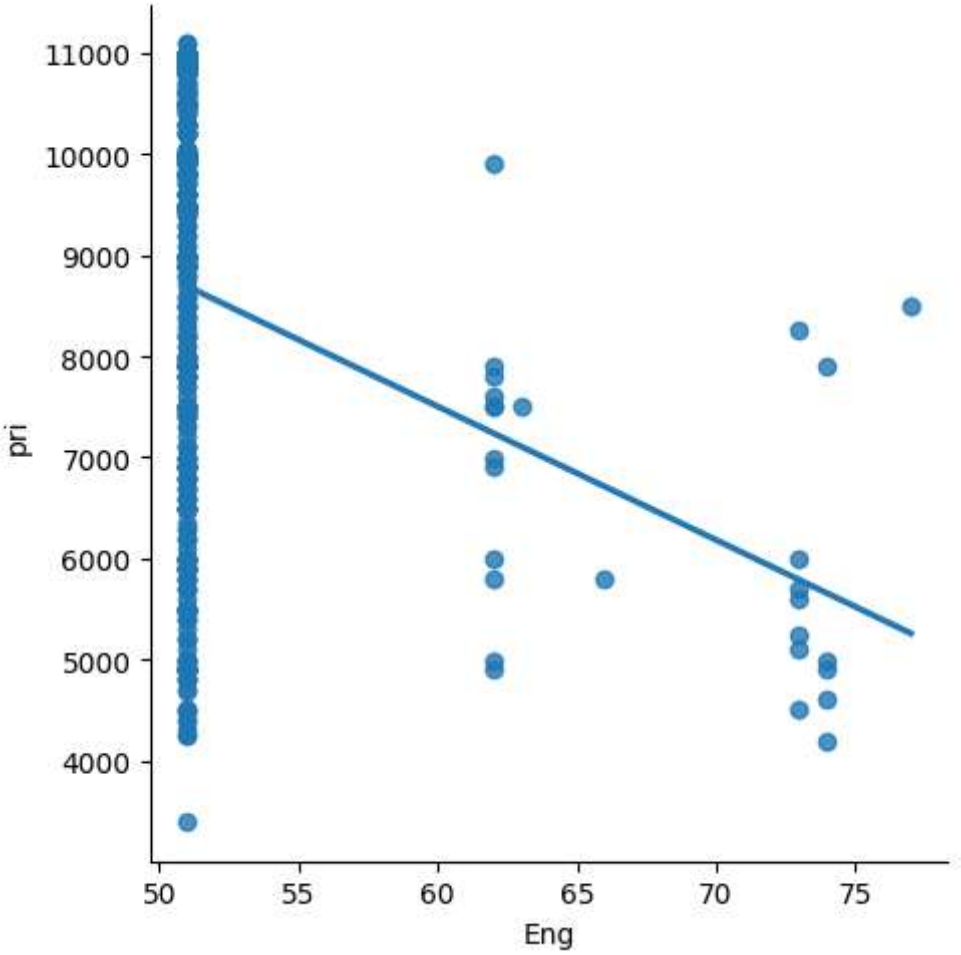
```
0.06465014107950018
```

In [15]:
```
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color = 'y')
plt.scatter(X_test, y_test, color = 'b')
plt.show()
```



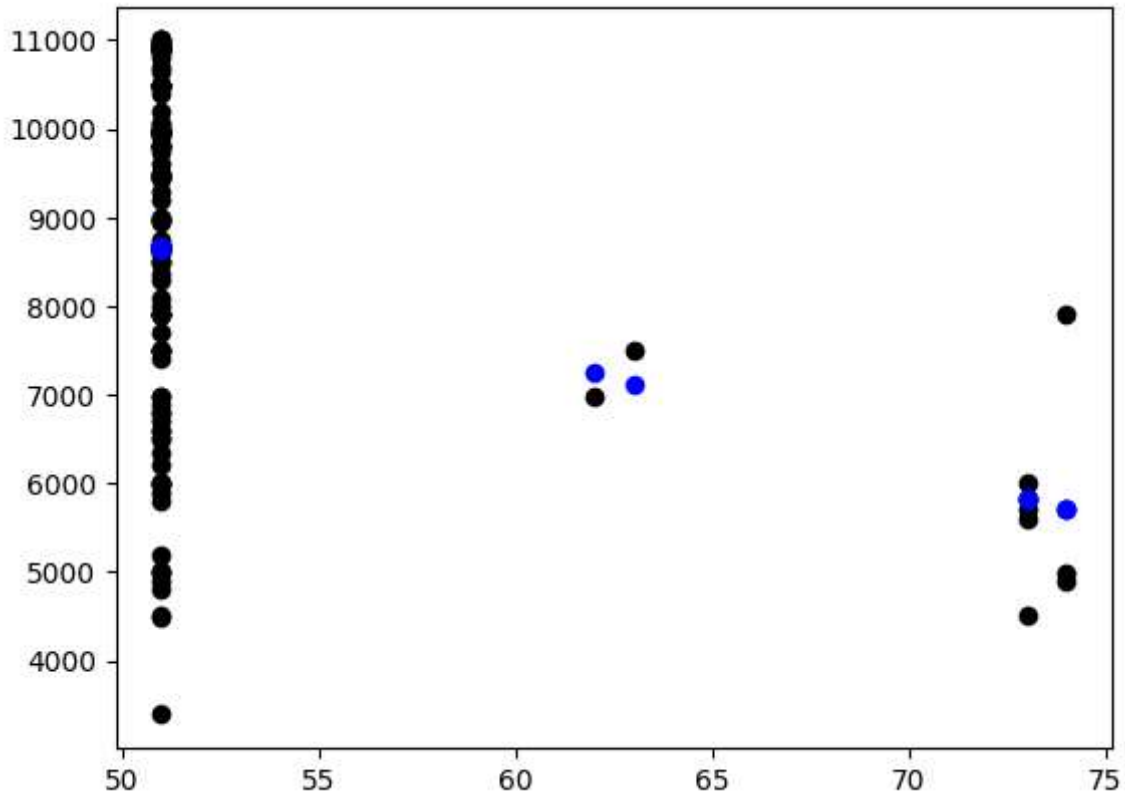
In [17]:
```
df500 = df[:][:500]
sns.lmplot(x = "Eng", y = "pri", data = df500, order = 1, ci = None)
```

Out[17]: <seaborn.axisgrid.FacetGrid at 0x1a95f5b0f50>



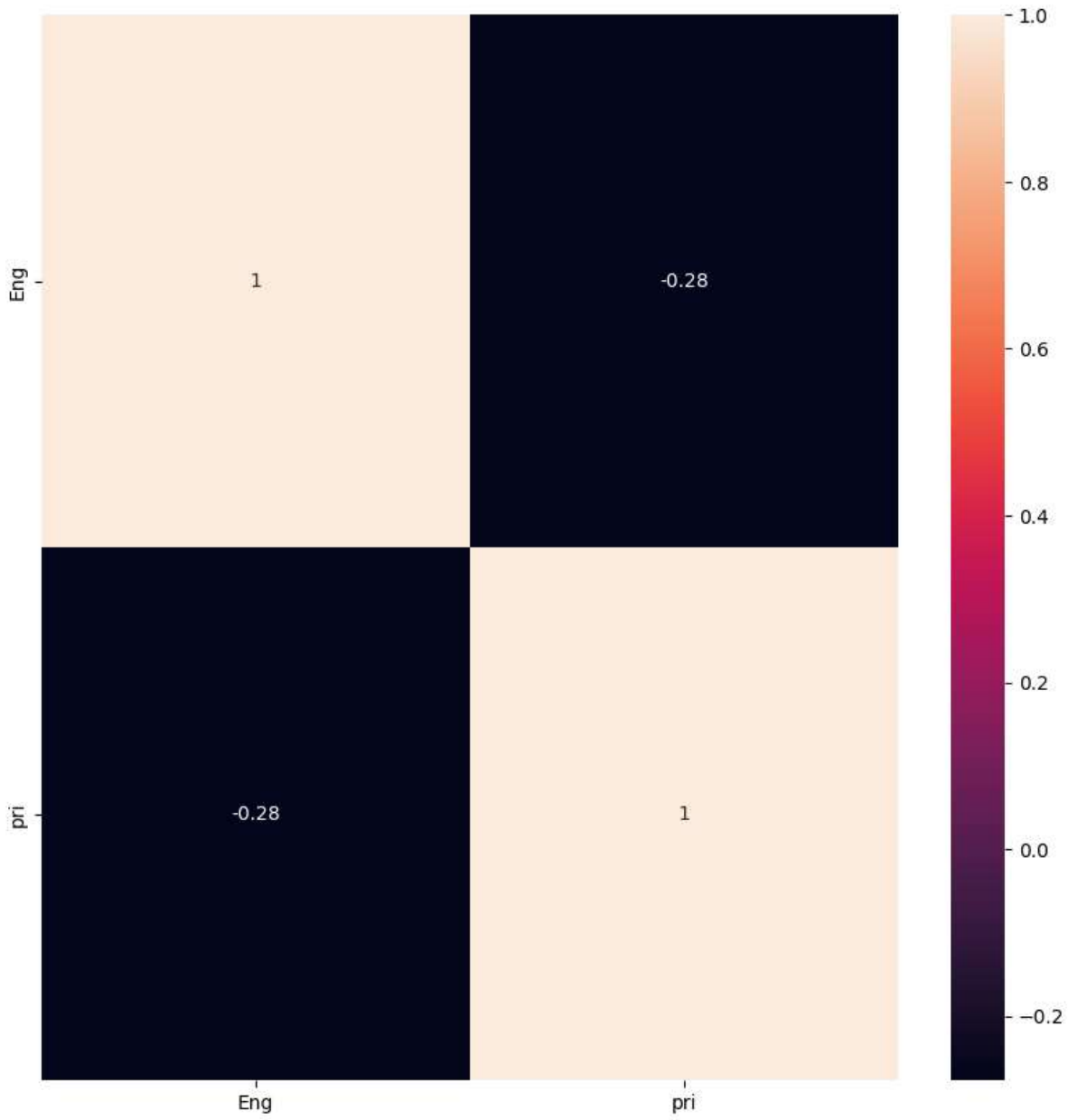
In [18]:
```
df500.fillna(method = 'ffill', inplace = True)
x = np.array(df500['Eng']).reshape(-1, 1)
y = np.array(df500['pri']).reshape(-1, 1)
df500.dropna(inplace = True)
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)
regr = LinearRegression()
regr.fit(X_train, y_train)
print("Regression:",regr.score(X_test, y_test))
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color = 'k')
plt.scatter(X_test, y_pred, color = 'b')
plt.show()
```

```
Regression: 0.13088284817450802
```



In [20]:
```
plt.figure(figsize = (10, 10))
sns.heatmap(df.corr(), annot = True)
```

Out[20]: <Axes: >




In [21]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
#Train the model
model = LinearRegression()
model.fit(X_train, y_train)
#Evaluating the model on the test set
y_pred = model.predict(X_test)
r2 = r2_score(y_test, y_pred)
print("R2 score:",r2)
```

```
R2 score: 0.13088284817450802
```

In [22]:
```
lr = LinearRegression()
#Fit model
lr.fit(X_train, y_train)
#predict
#prediction = lr.predict(X_test)
#actual
actual = y_test
train_score_lr = lr.score(X_train, y_train)
test_score_lr = lr.score(X_test, y_test)
print("\nLinear Regression Model:\n")
print("The train score for lr model is {}".format(train_score_lr))
print("The test score for lr model is {}".format(test_score_lr))
```

```
Linear Regression Model:

The train score for lr model is 0.04886161713188475
The test score for lr model is 0.13088284817450802
```

In [23]:
```
ridgeReg = Ridge(alpha=10)
ridgeReg.fit(X_train,y_train)
train_score_ridge = ridgeReg.score(X_train, y_train)
test_score_ridge = ridgeReg.score(X_test, y_test)
print("\nRidge Model:\n")
print("The train score for ridge model is {}".format(train_score_ridge))
print("The test score for ridge model is {}".format(test_score_ridge))
```

```
Ridge Model:

The train score for ridge model is 0.04886138130028239
The test score for ridge model is 0.13085286416194186
```

In [ ]: