In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:
```
df=pd.read_csv(r"C:\Users\arshiha\Downloads\bottle.csv.zip")
df
```

```
C:\Users\arshiha\AppData\Local\Temp\ipykernel_9508\710205783.py:1: DtypeWarn
ing: Columns (47,73) have mixed types. Specify dtype option on import or set
low_memory=False.
  df=pd.read_csv(r"C:\Users\arshiha\Downloads\bottle.csv.zip")
```

Out[2]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2
0	1	1	054.0 056.0	19-4903CR-HY-060-0930-05400560-0000A-3	0	10.500	33.4400	NaN	25.64900	N
1	1	2	054.0 056.0	19-4903CR-HY-060-0930-05400560-0008A-3	8	10.460	33.4400	NaN	25.65600	N
2	1	3	054.0 056.0	19-4903CR-HY-060-0930-05400560-0010A-7	10	10.460	33.4370	NaN	25.65400	N
3	1	4	054.0 056.0	19-4903CR-HY-060-0930-05400560-0019A-3	19	10.450	33.4200	NaN	25.64300	N
4	1	5	054.0 056.0	19-4903CR-HY-060-0930-05400560-0020A-7	20	10.450	33.4210	NaN	25.64300	N
...	...	...	...	...	...	...	...	...	...	
864858	34404	864859	093.4 026.4	20-1611SR-MX-310-2239-09340264-0000A-7	0	18.744	33.4083	5.805	23.87055	108
864859	34404	864860	093.4 026.4	20-1611SR-MX-310-2239-09340264-0002A-3	2	18.744	33.4083	5.805	23.87072	108
864860	34404	864861	093.4 026.4	20-1611SR-MX-310-2239-09340264-0005A-3	5	18.692	33.4150	5.796	23.88911	108
864861	34404	864862	093.4 026.4	20-1611SR-MX-310-2239-09340264-0010A-3	10	18.161	33.4062	5.816	24.01426	107

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2
864862	34404	864863	093.4 026.4	20-1611SR-MX-310-2239-09340264-0015A-3	15	17.533	33.3880	5.774	24.15297	105

864863 rows × 74 columns

In [3]:
```
df=df[['Salnty','T_degC']]
df.columns=['Sal','Temp']
```

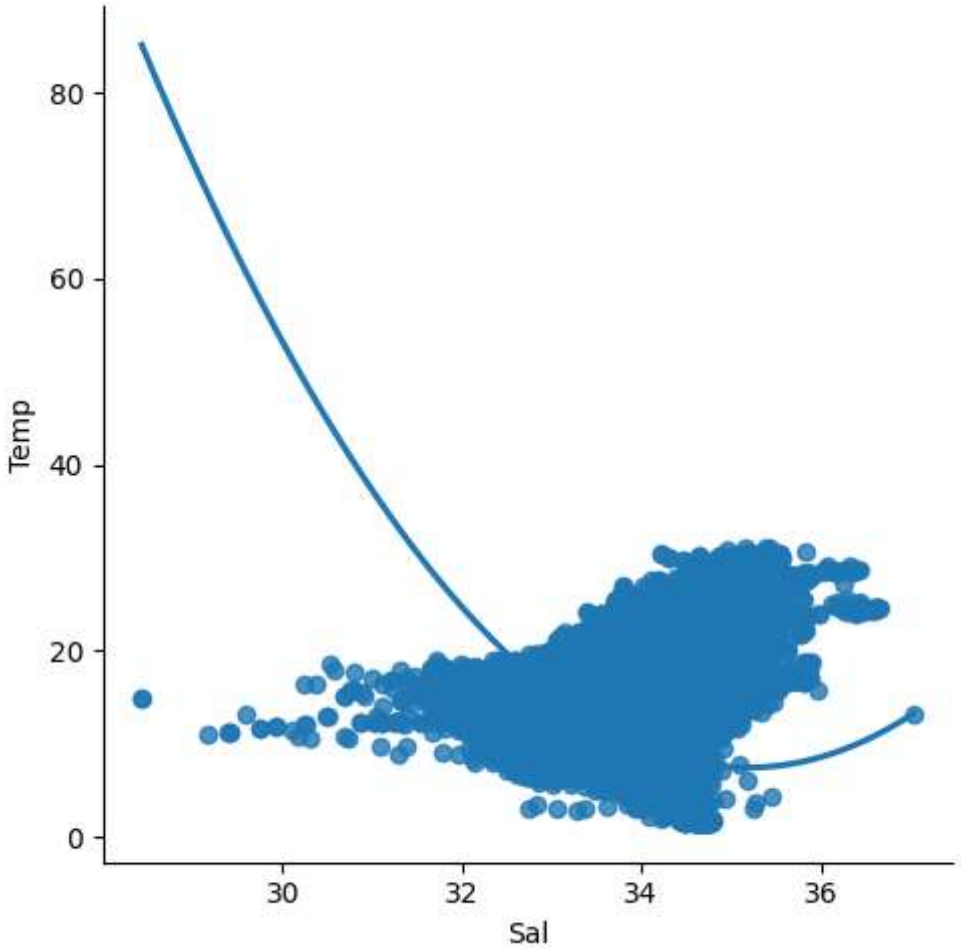
In [4]:
```
df.head(10)
```

Out[4]:

	Sal	Temp
0	33.440	10.50
1	33.440	10.46
2	33.437	10.46
3	33.420	10.45
4	33.421	10.45
5	33.431	10.45
6	33.440	10.45
7	33.424	10.24
8	33.420	10.06
9	33.494	9.86

In [5]:
```
sns.lmplot(x="Sal",y="Temp",data=df,order=2,ci=None)
```

Out[5]:
```
<seaborn.axisgrid.FacetGrid at 0x1dd8d441650>
```



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

Out[6]:

	Sal	Temp
count	817509.000000	853900.000000
mean	33.840350	10.799677
std	0.461843	4.243825
min	28.431000	1.440000
25%	33.488000	7.680000
50%	33.863000	10.060000
75%	34.196900	13.880000
max	37.034000	31.140000

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 864863 entries, 0 to 864862
Data columns (total 2 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Sal     817509 non-null  float64
 1   Temp    853900 non-null  float64
dtypes: float64(2)
memory usage: 13.2 MB
```

In [8]:
```
df.fillna(method="ffill",inplace=True)
```

```
C:\Users\arshiha\AppData\Local\Temp\ipykernel_9508\1844562654.py:1: SettingW
ithCopyWarning:
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/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  df.fillna(method="ffill",inplace=True)
```

In [9]:
```
x=np.array(df['Sal']).reshape(-1,1)
y=np.array(df['Temp']).reshape(-1,1)
```

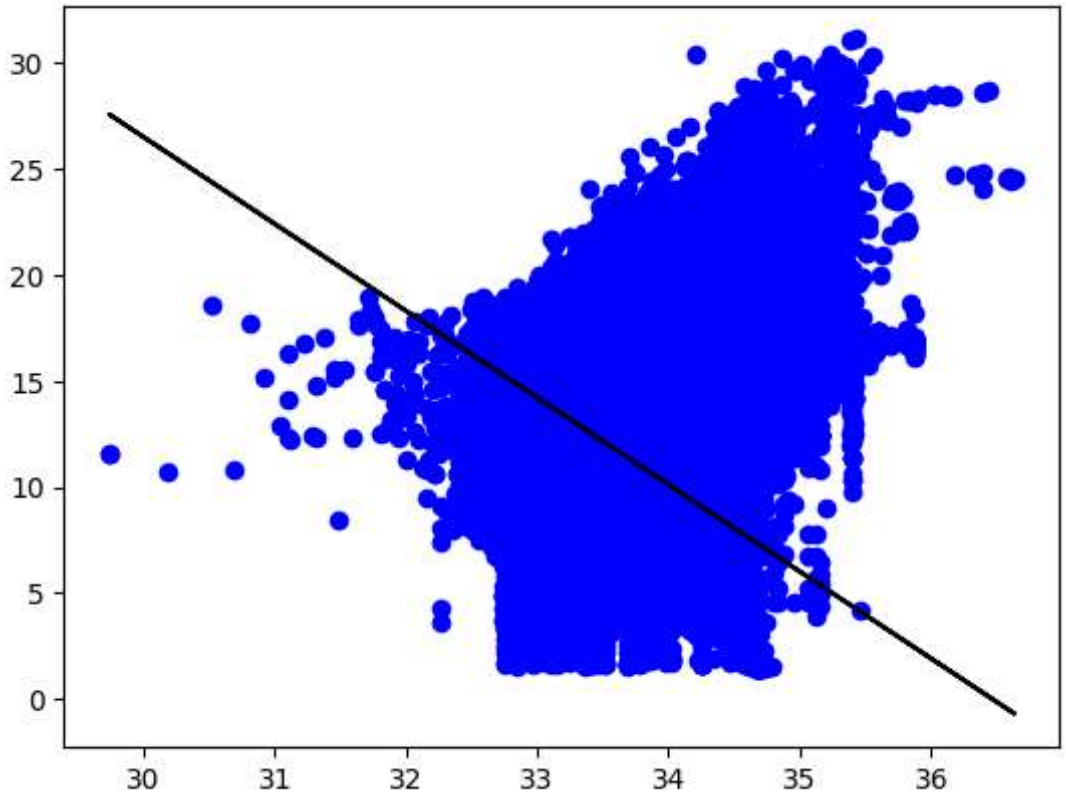
In [10]:
```
df.dropna(inplace=True)
```

```
C:\Users\arshiha\AppData\Local\Temp\ipykernel_9508\1379821321.py:1: SettingW
ithCopyWarning:
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/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  df.dropna(inplace=True)
```

In [11]:
```
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))
```

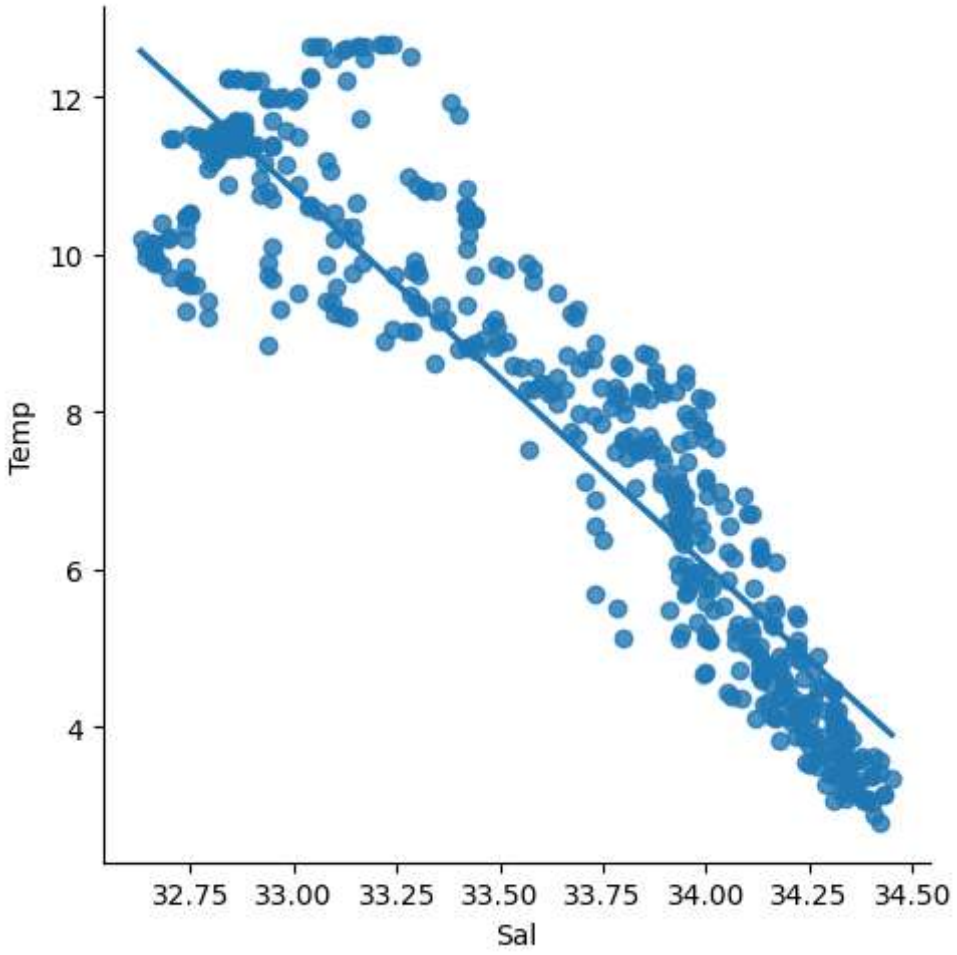
```
0.20456506239408734
```

In [12]:
```
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



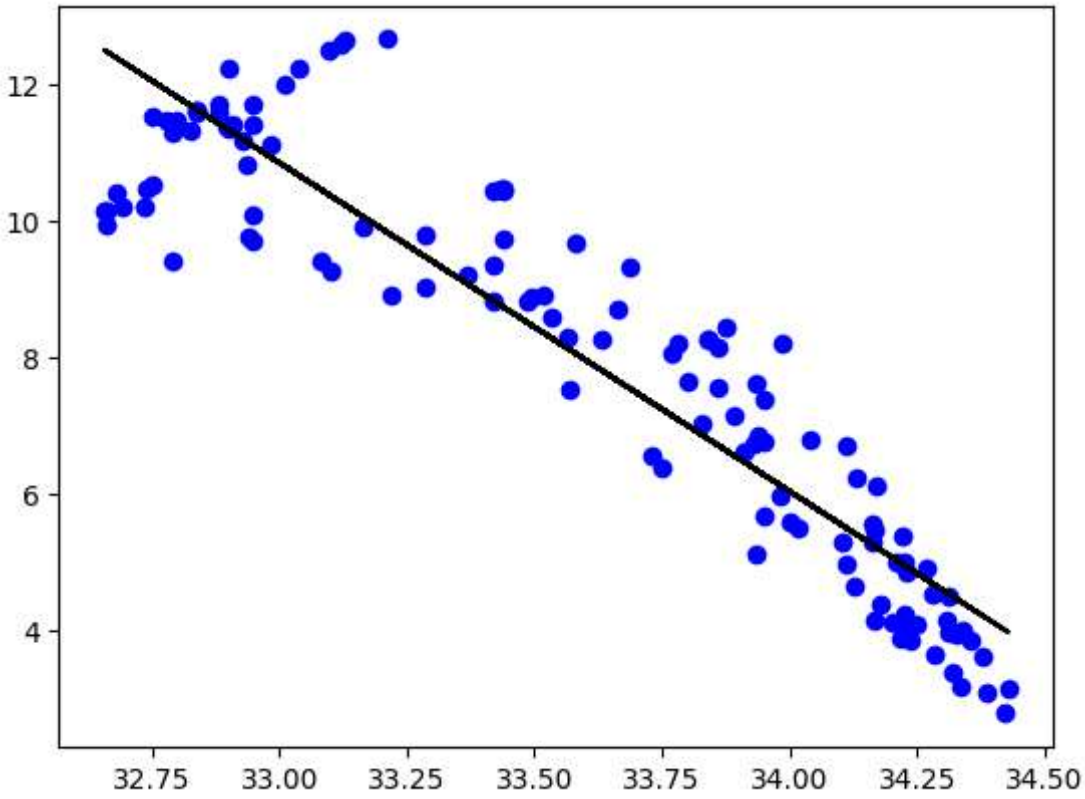
In [13]:
```
df500=df[:][:500]
sns.lmplot(x="Sal",y="Temp",data=df500,order=1,ci=None)
```

Out[13]: <seaborn.axisgrid.FacetGrid at 0x1dd9cd6bdd0>



In [14]:
```
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['Sal']).reshape(-1,1)
y=np.array(df500['Temp']).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='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```

```
Regression: 0.8566507158917958
```



In [15]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
r2=r2_score(y_test,y_pred)
print("R2 score:",r2)
```

```
R2 score: 0.8566507158917958
```

In [37]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [38]:
```
df=pd.read_csv(r"C:\Users\arshiha\Downloads\fiat500_VehicleSelection_Dataset.
df
```

Out[38]:

	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 [39]:
```
df=df[["model","km"]]
df.columns=['mdl','kms']
```

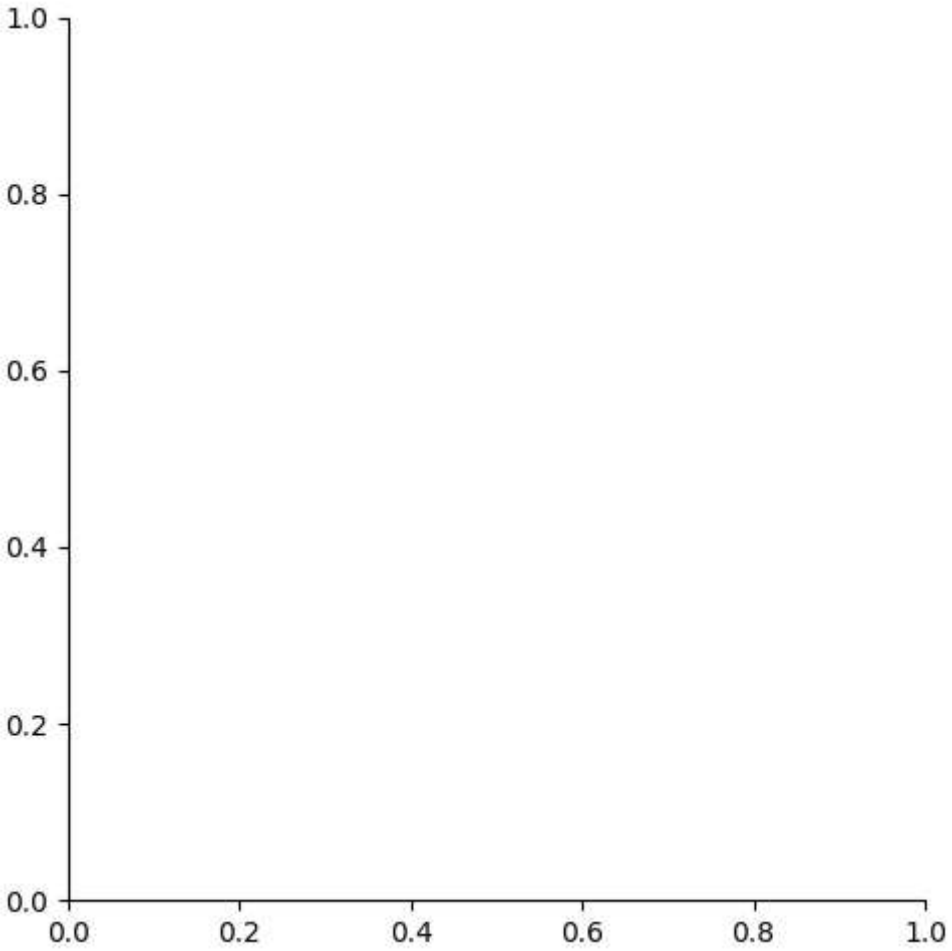
In [40]:
```
df.head(10)
```

Out[40]:

	mdl	kms
0	lounge	25000
1	pop	32500
2	sport	142228
3	lounge	160000
4	pop	106880
5	pop	70225
6	lounge	11600
7	lounge	49076
8	sport	76000
9	sport	89000

In [36]:
```
sns.lmplot(x="mdl",y="kms",data=df,order=2,ci=None)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[36], line 1
----> 1 sns.lmplot(x="mdl",y="kms",data=df,order=2,ci=None)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\regression.py:624, in lmplot(data, x, y, hue, col, row, palette, col_wrap, height, aspect, markers, sharex, sharey, hue_order, col_order, row_order, legend, legend_out, x_estimator, x_bins, x_ci, scatter, fit_reg, ci, n_boot, units, seed, order, logistic, lowess, robust, logx, x_partial, y_partial, truncate, x_jitter, y_jitter, scatter_kws, line_kws, facet_kws)
    621     ax.update_datalim(xys, updatey=False)
    622     ax.autoscale_view(scaley=False)
--> 624 facets.map_dataframe(update_datalim, x=x, y=y)
    626 # Draw the regression plot on each facet
    627 regplot_kws = dict(
    628     x_estimator=x_estimator, x_bins=x_bins, x_ci=x_ci,
    629     scatter=scatter, fit_reg=fit_reg, ci=ci, n_boot=n_boot, units=units,
   (...)
    633     scatter_kws=scatter_kws, line_kws=line_kws,
    634 )

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:819, in FacetGrid.map_dataframe(self, func, *args, **kwargs)
    816     kwargs["data"] = data_ijk
    818     # Draw the plot
--> 819     self._facet_plot(func, ax, args, kwargs)
    821 # For axis labels, prefer to use positional args for backcompat
    822 # but also extract the x/y kwargs and use if no corresponding arg
    823 axis_labels = [kwargs.get("x", None), kwargs.get("y", None)]

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848, in FacetGrid._facet_plot(self, func, ax, plot_args, plot_kwargs)
    846     plot_args = []
    847     plot_kwargs["ax"] = ax
--> 848 func(*plot_args, **plot_kwargs)
    850 # Sort out the supporting information
    851 self._update_legend_data(ax)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\regression.py:620, in lmplot.<locals>.update_datalim(data, x, y, ax, **kws)
    619 def update_datalim(data, x, y, ax, **kws):
--> 620     xys = data[[x, y]].to_numpy().astype(float)
    621     ax.update_datalim(xys, updatey=False)
    622     ax.autoscale_view(scaley=False)

ValueError: could not convert string to float: 'lounge'
```



In [25]:
```
df.describe()
```

Out[25]:

	km
count	1538.000000
mean	53396.011704
std	40046.830723
min	1232.000000
25%	20006.250000
50%	39031.000000
75%	79667.750000
max	235000.000000

In [26]:
```
df.info()
```

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

In [27]:
```
df.fillna(method="ffill",inplace=True)
```

```
C:\Users\arshiha\AppData\Local\Temp\ipykernel_9508\1844562654.py:1: SettingW
ithCopyWarning:
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/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  df.fillna(method="ffill",inplace=True)
```

In [29]:
```
x=np.array(df['model']).reshape(-1,1)
y=np.array(df['km']).reshape(-1,1)
```

In [30]:
```
df.dropna(inplace=True)
```

```
C:\Users\arshiha\AppData\Local\Temp\ipykernel_9508\1379821321.py:1: SettingW
ithCopyWarning:
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/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  df.dropna(inplace=True)
```

In [32]:
```
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))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[32], line 3
      1 x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
      2 regr=LinearRegression()
----> 3 regr.fit(x_train,y_train)
      4 print(regr.score(x_test,y_test))

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\linear_model\_base.py:648, in LinearRegression.fit(self, X, y, sample_weight)
    644 n_jobs_ = self.n_jobs
    646 accept_sparse = False if self.positive else ["csr", "csc", "coo"]
--> 648 X, y = self._validate_data(
    649     X, y, accept_sparse=accept_sparse, y_numeric=True, multi_output=True
    650 )
    652 sample_weight = _check_sample_weight(
    653     sample_weight, X, dtype=X.dtype, only_non_negative=True
    654 )
    656 X, y, X_offset, y_offset, X_scale = _preprocess_data(
    657     X,
    658     y,
   (...)
    661     sample_weight=sample_weight,
    662 )

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\base.py:584, in BaseEstimator._validate_data(self, X, y, reset, validate_separately, **check_params)
    582         y = check_array(y, input_name="y", **check_y_params)
    583     else:
--> 584         X, y = check_X_y(X, y, **check_params)
    585     out = X, y
    587 if not no_val_X and check_params.get("ensure_2d", True):

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\utils\validation.py:1106, in check_X_y(X, y, accept_sparse, accept_large_sparse, dtype, order, copy, force_all_finite, ensure_2d, allow_nd, multi_output, ensure_min_samples, ensure_min_features, y_numeric, estimator)
   1101         estimator_name = _check_estimator_name(estimator)
   1102     raise ValueError(
   1103         f"{estimator_name} requires y to be passed, but the target y is None"
   1104     )
-> 1106 X = check_array(
   1107     X,
   1108     accept_sparse=accept_sparse,
   1109     accept_large_sparse=accept_large_sparse,
   1110     dtype=dtype,
   1111     order=order,
   1112     copy=copy,
   1113     force_all_finite=force_all_finite,
   1114     ensure_2d=ensure_2d,
   1115     allow_nd=allow_nd,
   1116     ensure_min_samples=ensure_min_samples,
   1117     ensure_min_features=ensure_min_features,
   1118     estimator=estimator,
```

```
   1119     input_name="X",
   1120 )
   1122 y = _check_y(y, multi_output=multi_output, y_numeric=y_numeric, esti
mator=estimator)
   1124 check_consistent_length(X, y)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\uti
ls\validation.py:879, in check_array(array, accept_sparse, accept_large_spar
se, dtype, order, copy, force_all_finite, ensure_2d, allow_nd, ensure_min_sa
mples, ensure_min_features, estimator, input_name)
    877         array = xp.astype(array, dtype, copy=False)
    878     else:
--> 879         array = _asarray_with_order(array, order=order, dtype=dtype,
xp=xp)
    880 except ComplexWarning as complex_warning:
    881     raise ValueError(
    882         "Complex data not supported\n{}\n".format(array)
    883     ) from complex_warning

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\uti
ls\_array_api.py:185, in _asarray_with_order(array, dtype, order, copy, xp)
    182     xp, _ = get_namespace(array)
    183 if xp.__name__ in {"numpy", "numpy.array_api"}:
    184     # Use NumPy API to support order
--> 185     array = numpy.asarray(array, order=order, dtype=dtype)
    186     return xp.asarray(array, copy=copy)
    187 else:

ValueError: could not convert string to float: 'pop'
```

In [ ]: