In [1]:
```
import pandas as pd
```

In [2]:
```
df=pd.read_csv("C:/Users/ROCKSTAR/Desktop/final_data_set2.csv")
df.head()
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Input In [2], in <cell line: 1>()
----> 1 df=pd.read_csv("C:/Users/ROCKSTAR/Desktop/final_data_set2.csv")
      2 df.head()

File ~\anaconda3\lib\site-packages\pandas\util\_decorators.py:311, in deprecate_nonkeyword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    305 if len(args) > num_allow_args:
    306     warnings.warn(
    307         msg.format(arguments=arguments),
    308         FutureWarning,
    309         stacklevel=stacklevel,
    310     )
--> 311 return func(*args, **kwargs)

File ~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py:680, in read_csv(filepath_or_buffer, sep, delimiter, header, names, index_col, usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters, true_values, false_values, skipinitialspace, skiprows, skipfooter, nrows, na_values, keep_default_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetime_format, keep_date_col, date_parser, dayfirst, cache_dates, iterator, chunksize, compression, thousands, decimal, lineterminator, quotechar, quoting, doublequote, escapechar, comment, encoding, encoding_errors, dialect, error_bad_lines, warn_bad_lines, on_bad_lines, delim_whitespace, low_memory, memory_map, float_precision, storage_options)
    665 kwds_defaults = _refine_defaults_read(
    666     dialect,
    667     delimiter,
   (...)
    676     defaults={"delimiter": ","},
    677 )
    678 kwds.update(kwds_defaults)
--> 680 return _read(filepath_or_buffer, kwds)

File ~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py:575, in _read(filepath_or_buffer, kwds)
    572 _validate_names(kwds.get("names", None))
    574 # Create the parser.
--> 575 parser = TextFileReader(filepath_or_buffer, **kwds)
    577 if chunksize or iterator:
    578     return parser

File ~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py:933, in TextFileReader.__init__(self, f, engine, **kwds)
    930     self.options["has_index_names"] = kwds["has_index_names"]
    932 self.handles: IOHandles | None = None
--> 933 self._engine = self._make_engine(f, self.engine)

File ~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py:1217, in TextFileReader._make_engine(self, f, engine)
   1213     mode = "rb"
   1214 # error: No overload variant of "get_handle" matches argument types
   1215 # "Union[str, PathLike[str], ReadCsvBuffer[bytes], ReadCsvBuffer[str]]"
   1216 # , "str", "bool", "Any", "Any", "Any", "Any", "Any"
-> 1217 self.handles = get_handle(  # type: ignore[call-overload]
```

```
    1218     f,
    1219     mode,
    1220     encoding=self.options.get("encoding", None),
    1221     compression=self.options.get("compression", None),
    1222     memory_map=self.options.get("memory_map", False),
    1223     is_text=is_text,
    1224     errors=self.options.get("encoding_errors", "strict"),
    1225     storage_options=self.options.get("storage_options", None),
    1226 )
    1227 assert self.handles is not None
    1228 f = self.handles.handle
```

```
File ~\anaconda3\lib\site-packages\pandas\io\common.py:789, in get_handle
(path_or_buf, mode, encoding, compression, memory_map, is_text, errors, st
orage_options)
     784 elif isinstance(handle, str):
     785     # Check whether the filename is to be opened in binary mode.
     786     # Binary mode does not support 'encoding' and 'newline'.
     787     if ioargs.encoding and "b" not in ioargs.mode:
     788         # Encoding
--> 789         handle = open(
     790             handle,
     791             ioargs.mode,
     792             encoding=ioargs.encoding,
     793             errors=errors,
     794             newline="",
     795         )
     796     else:
     797         # Binary mode
     798         handle = open(handle, ioargs.mode)
```

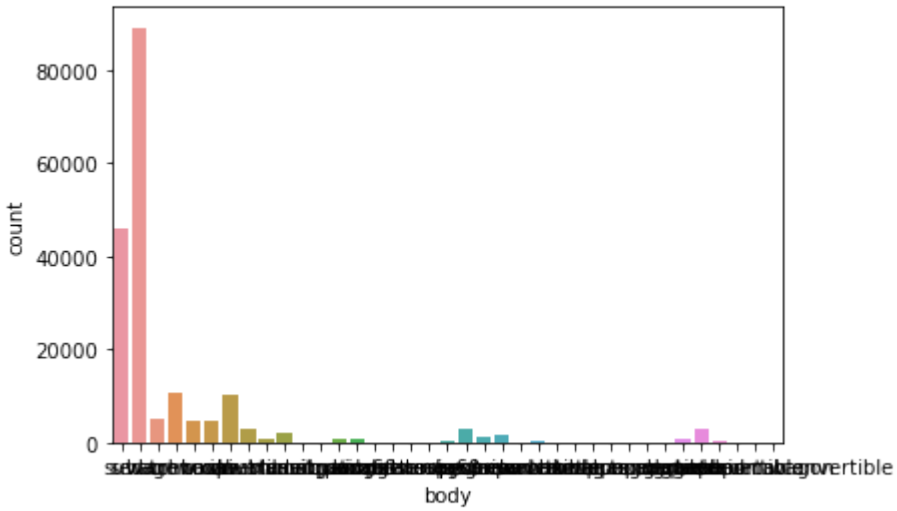
```
FileNotFoundError: [Errno 2] No such file or directory: 'C:/Users/ROCKSTA
R/Desktop/final_data_set2.csv'
```

In [56]:
```
import seaborn as sns
```

In [85]:
```
sns.countplot(df['body'])
```

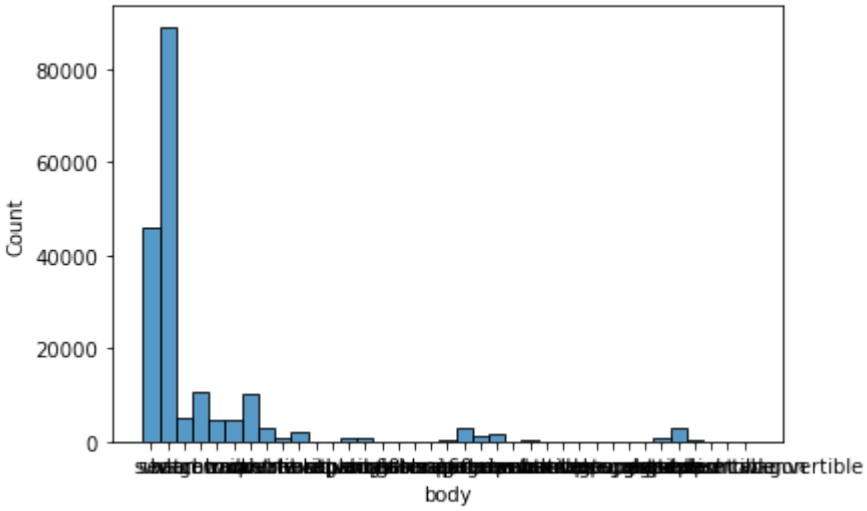
```
C:\Users\ROCKSTAR\anaconda3\lib\site-packages\seaborn\_decorators.py:36: F
utureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing oth
er arguments without an explicit keyword will result in an error or misint
erpretation.
  warnings.warn(
```

Out[85]: <AxesSubplot:xlabel='body', ylabel='count'>



In [32]:
```
sns.histplot(df['body'],bins=1)
```

Out[32]: <AxesSubplot:xlabel='body', ylabel='Count'>



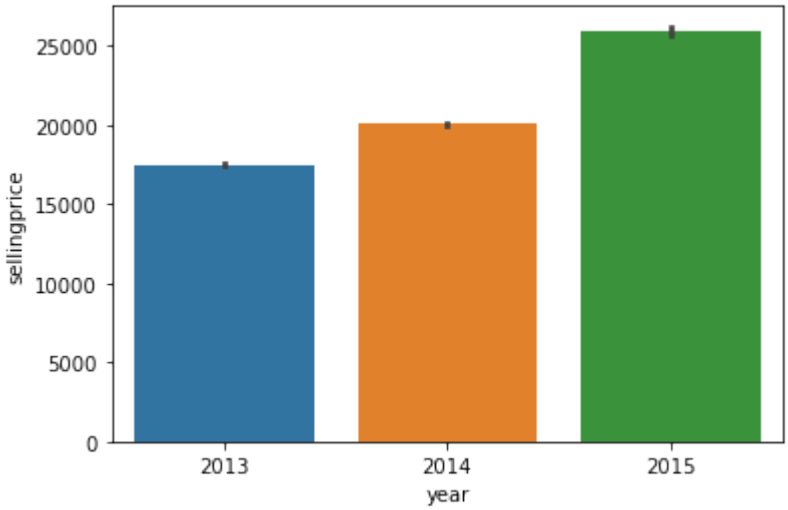
In [49]:
```
sns.boxplot(x=df['make']=='kia')
```

```
    447 color = self.colors[i]
    448 self.restyle_boxplot(artist_dict, color, props)

File ~\anaconda3\lib\site-packages\matplotlib\__init__.py:1412, in _pre
process_data.<locals>.inner(ax, data, *args, **kwargs)
   1409 @functools.wraps(func)
   1410 def inner(ax, *args, data=None, **kwargs):
   1411     if data is None:
-> 1412         return func(ax, *map(sanitize_sequence, args), **kwarg
s)
   1414     bound = new_sig.bind(ax, *args, **kwargs)
   1415     auto_label = (bound.arguments.get(label_namer)
   1416                   or bound.kwargs.get(label_namer))

File ~\anaconda3\lib\site-packages\matplotlib\axes\_axes.py:3711, in Ax
es.boxplot(self, x, notch, sym, vert, whis, positions, widths, patch_ar
tist, bootstrap, usermedians, conf_intervals, meanline, showmeans, show
caps, showbox, showfliers, boxprops, labels, flierprops, medianprops, m
eanprops, capprops, whiskerprops, manage_ticks, autorange, zorder)
   3708 if bootstrap is None:
```

In [58]:
```
import matplotlib.pyplot as plt
```

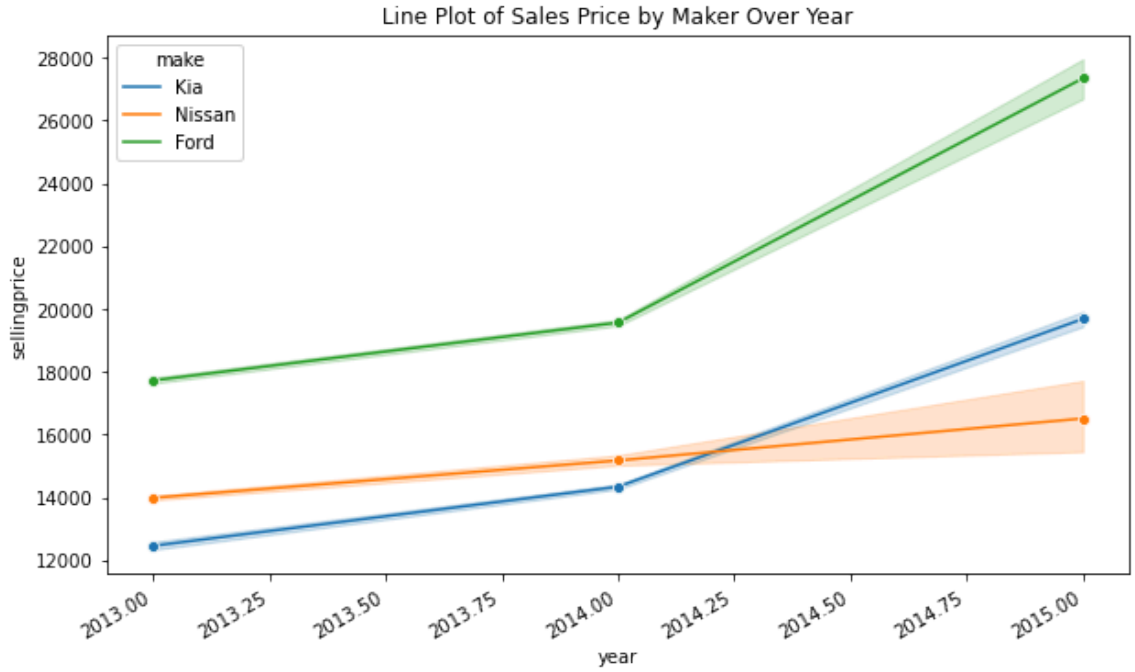
In [34]:
```
sns.barplot(x=df['year'],y=df['sellingprice'])
```

Out[34]:
```
<AxesSubplot:xlabel='year', ylabel='sellingprice'>
```

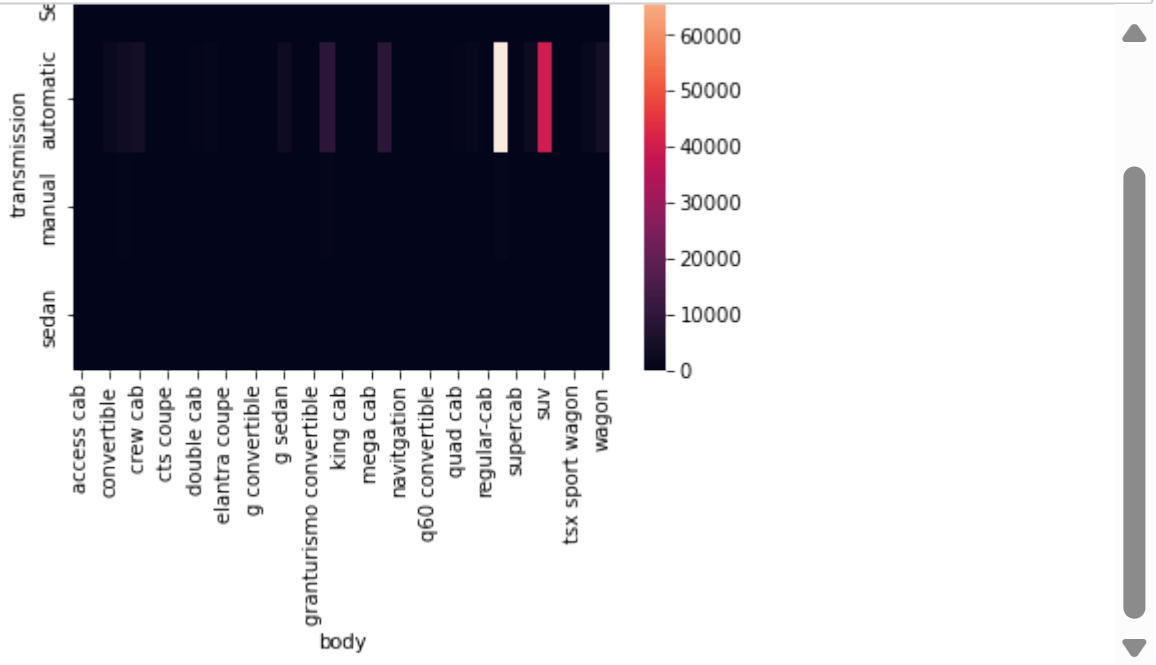


In [61]:
```
filter_data=df[df['make'].isin(['Kia','Ford','Nissan'])]
```

In [64]:
```
plt.figure(figsize=(10,6))
sns.lineplot(data=filter_data, x='year' , y='sellingprice', hue= 'make', ma

plt.gcf() .autofmt_xdate()
plt.title( 'Line Plot of Sales Price by Maker Over Year')
plt.xlabel ('year')
plt.ylabel('sellingprice')
plt.show()
```



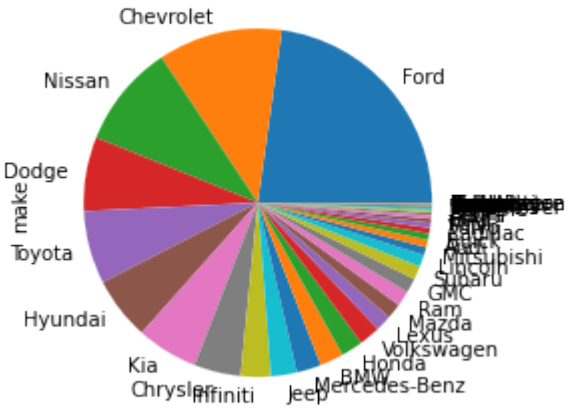
In [73]:
```
sns.heatmap(pd.crosstab(df['transmission'],df['body']))
```



In [ ]:

In [80]:
```
df['make'].value_counts().plot(kind='pie')
```

Out[80]:
```
<AxesSubplot:ylabel='make'>
```



In [91]:
```
df.drop(columns=['vin'],inplace=True)
```

In [92]:
```
df
```

Out[92]:

	year	make	model	trim	body	transmission	state	condition	odom
0	2015	Kia	sorento	LX	suv	automatic	ca	5.0	166
1	2015	Kia	sorento	LX	suv	automatic	ca	5.0	93
2	2015	Volvo	s60	T5	sedan	automatic	ca	41.0	142
3	2015	Nissan	altima	2.5 S	sedan	automatic	ca	1.0	55
4	2015	Kia	optima	LX	sedan	automatic	ca	48.0	20
...	...	...	...	...	...	...	...	...	
188670	2013	Honda	civic	LX PZEV	sedan	automatic	ga	19.0	321
188671	2013	Hyundai	sonata	GLS	sedan	NaN	nj	2.0	440
188672	2013	Mercedes-Benz	g-class	G63 AMG	suv	automatic	fl	5.0	267
188673	2013	Chevrolet	silverado 1500	LT	crew cab	automatic	tx	43.0	745
188674	2013	Audi	s5	Premium Plus quattro	convertible	automatic	fl	5.0	201

188675 rows × 15 columns

In [93]:
```
df.to_csv('C:/Users/ROCKSTAR/Desktop/py notes/final_data_set4.csv')
```

In [ ]:

In [96]:
```
pd.crosstab(df['make'],df['body'])
```

Out[96]:

body make	access cab	beetle convertible	convertible	coupe	crew cab	crewmax cab	cts coupe	cts-v coupe	double cab
Acura	0	0	0	0	0	0	0	0	0
Audi	0	0	28	133	0	0	0	0	0
BMW	0	0	301	383	0	0	0	0	0
Bentley	0	0	7	4	0	0	0	0	0
Buick	0	0	0	0	0	0	0	0	0
Cadillac	0	0	0	5	0	0	23	5	0
Chevrolet	0	0	408	707	1530	0	0	0	183
Chrysler	0	0	50	0	0	0	0	0	0
Dodge	0	0	0	644	0	0	0	0	0
FIAT	0	0	39	0	0	0	0	0	0
Ford	0	0	1428	1337	552	0	0	0	0
GMC	0	0	0	0	528	0	0	0	64
Honda	0	0	0	330	36	0	0	0	0
Hyundai	0	0	0	0	0	0	0	0	0
Infiniti	0	0	0	0	0	0	0	0	0
Jaguar	0	0	33	22	0	0	0	0	0
Jeep	0	0	0	0	0	0	0	0	0
Kia	0	0	0	0	0	0	0	0	0
Land Rover	0	0	0	0	0	0	0	0	0
Lexus	0	0	17	11	0	0	0	0	0
Lincoln	0	0	0	0	0	0	0	0	0
MINI	0	0	60	0	0	0	0	0	0
Maserati	0	0	0	6	0	0	0	0	0
Mazda	0	0	67	0	0	0	0	0	0
Mercedes-Benz	0	0	189	377	0	0	0	0	0
Mitsubishi	0	0	0	0	0	0	0	0	0
Nissan	0	0	6	344	438	0	0	0	0
Porsche	0	0	98	119	0	0	0	0	0
Ram	0	0	0	0	1569	0	0	0	0
Rolls-Royce	0	0	0	0	0	0	0	0	0
Scion	0	0	0	166	0	0	0	0	0
Subaru	0	0	0	42	0	0	0	0	0
Suzuki	0	0	0	0	0	0	0	0	0
Tesla	0	0	0	0	0	0	0	0	0
Toyota	77	0	0	0	0	187	0	0	620

body	access cab	beetle convertible	convertible	coupe	crew cab	crewmax cab	cts coupe	cts-v coupe	double cab
make									
Volkswagen	0	59	16	0	0	0	0	0	0
Volvo	0	0	2	0	0	0	0	0	0
smart	0	0	9	0	0	0	0	0	0

38 rows × 37 columns

In [97]:
```
pd.crosstab(df['year'],df['make'])#2013 ----top sell is Chevrolet=10904 #20
```

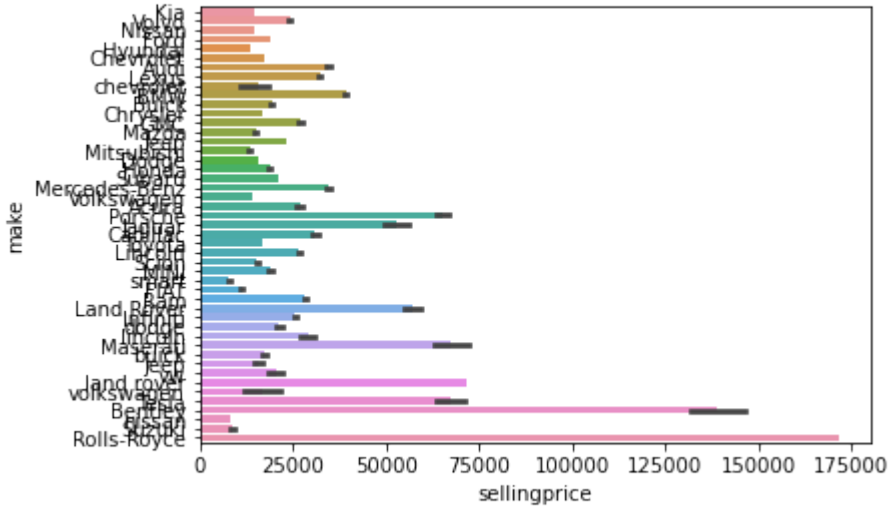
Out[97]:

make	Acura	Audi	BMW	Bentley	Buick	Cadillac	Chevrolet	Chrysler	Dodge	FIAT	...	b
year												
2013	293	708	1797	11	544	594	10904	2883	5854	259	...	
2014	183	452	1696	2	462	306	9349	4729	6811	366	...	
2015	22	126	658	0	122	62	1435	491	246	5	...	

3 rows × 47 columns

In [101]:
```
sns.barplot(x=df['sellingprice'],y=df['make'])
```

Out[101]:
```
<AxesSubplot:xlabel='sellingprice', ylabel='make'>
```



In [ ]: