Machine Learning Assignment 1

Linear Regression

- RAJ GUPTA
- 20124082
- G3

In [ ]:

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import style

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:
```
df = pd.read_csv("C:\\Users\\raj gupta\\Downloads\\insurance.csv")
df.head()
```

Out[2]:

	age	sex	bmi	children	smoker	region	expenses
0	19	female	27.9	0	yes	southwest	16884.92
1	18	male	33.8	1	no	southeast	1725.55
2	28	male	33.0	3	no	southeast	4449.46
3	33	male	22.7	0	no	northwest	21984.47
4	32	male	28.9	0	no	northwest	3866.86

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object 
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object 
 5   region    1338 non-null   object 
 6   expenses  1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

In [4]:
```
df.isnull().sum()
```

Out[4]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
expenses    0
dtype: int64
```

Data Visualisation

In [5]:
```
sns.distplot(df['age'])
```

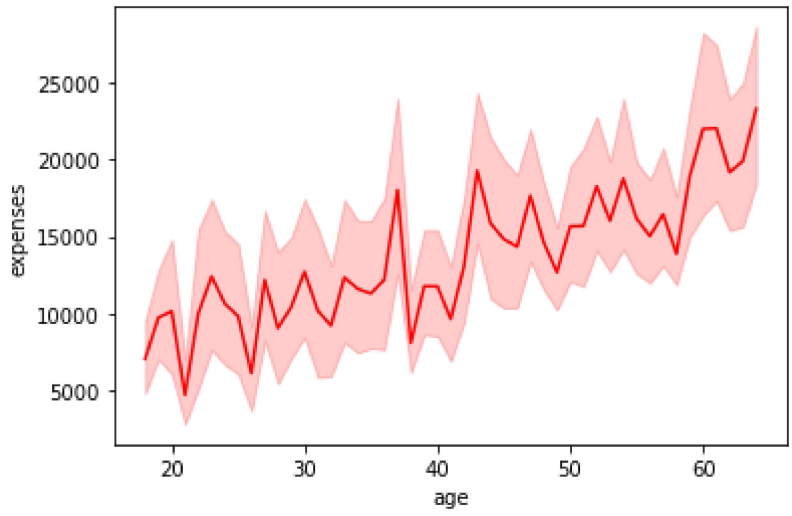
```
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function for hi
stograms).
  warnings.warn(msg, FutureWarning)
```

Out[5]:
```
<AxesSubplot:xlabel='age', ylabel='Density'>
```

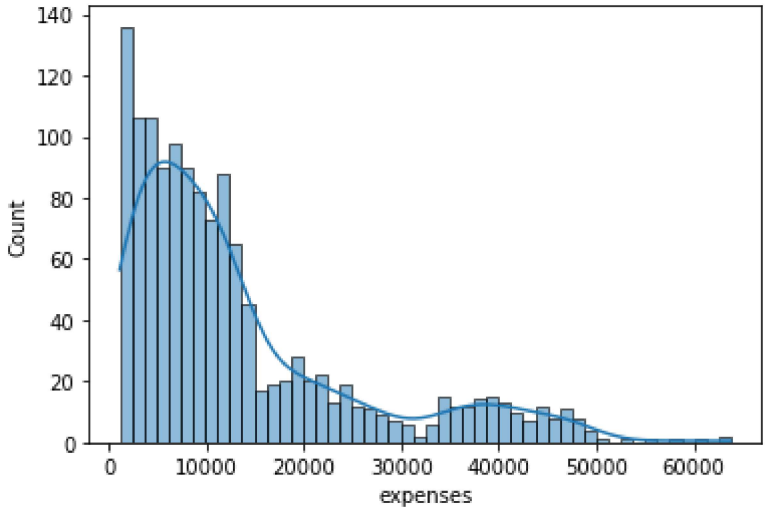
In [6]:
```
sns.lineplot(df['age'] ,df['expenses'] ,color = 'r')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variables as keyword args: x, y. From version 0.12, th
e only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[6]: <AxesSubplot:xlabel='age', ylabel='expenses'>



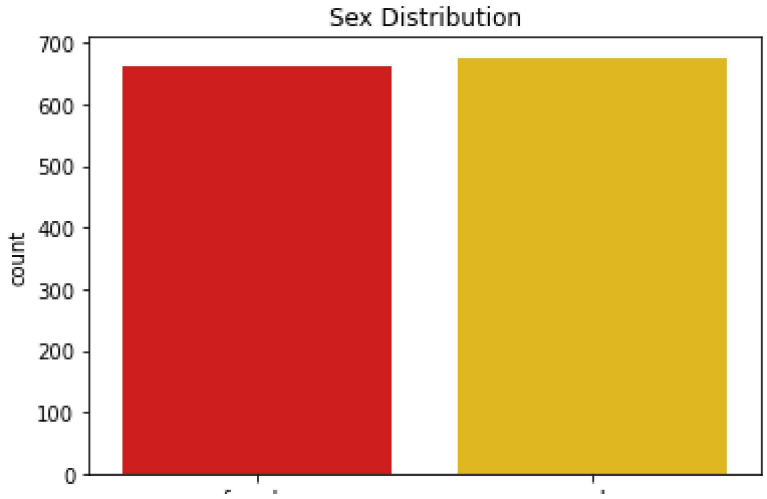
In [7]:
```
sns.histplot(data=df, x='expenses', bins=50 ,kde=True);
```



In [8]:
```
plt.title('Sex Distribution')
sns.countplot(df['sex'] ,palette = 'hot')
```

```
without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

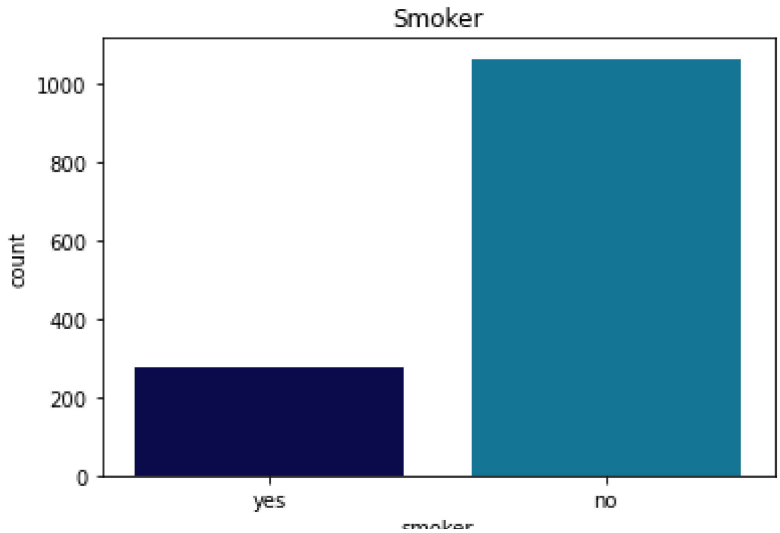
Out[8]:
```
<AxesSubplot:title={'center':'Sex Distribution'}, xlabel='sex', ylabel='coun
t'>
```



In [9]:
```
plt.title('Smoker')
sns.countplot(df['smoker'] ,palette = 'ocean')
```

```
without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

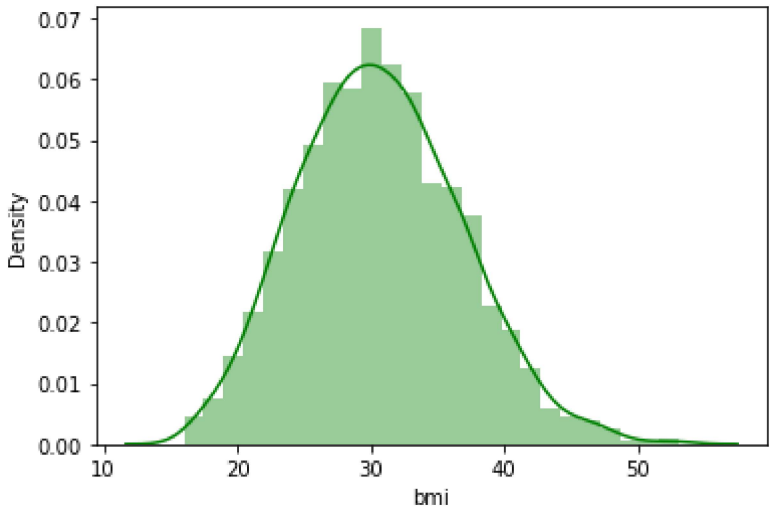
Out[9]:
```
<AxesSubplot:title={'center':'Smoker'}, xlabel='smoker', ylabel='count'>
```



In [10]:
```
sns.distplot(df['bmi'] ,color ='g')
```

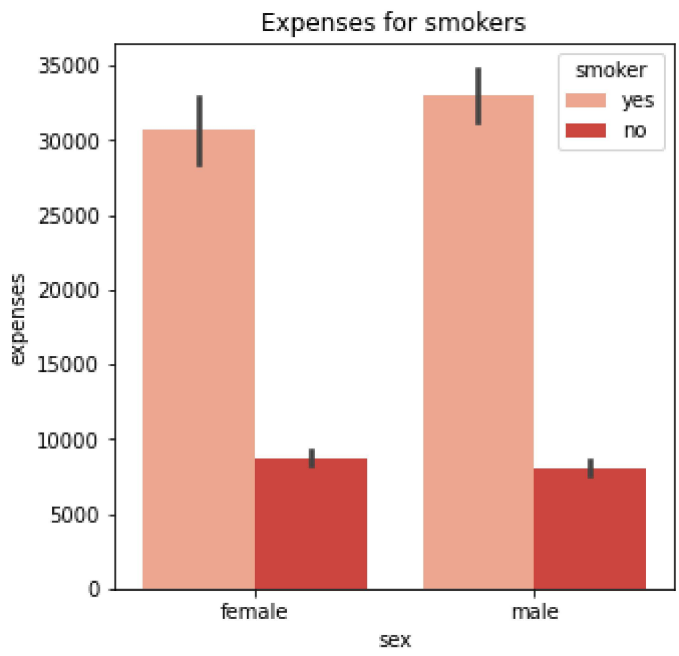
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: Futur
eWarning: `distplot` is a deprecated function and will be removed in a future v
ersion. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `histplot` (an axes-level function for histogram
s).
  warnings.warn(msg, FutureWarning)
```

Out[10]:
```
<AxesSubplot:xlabel='bmi', ylabel='Density'>
```



In [11]:

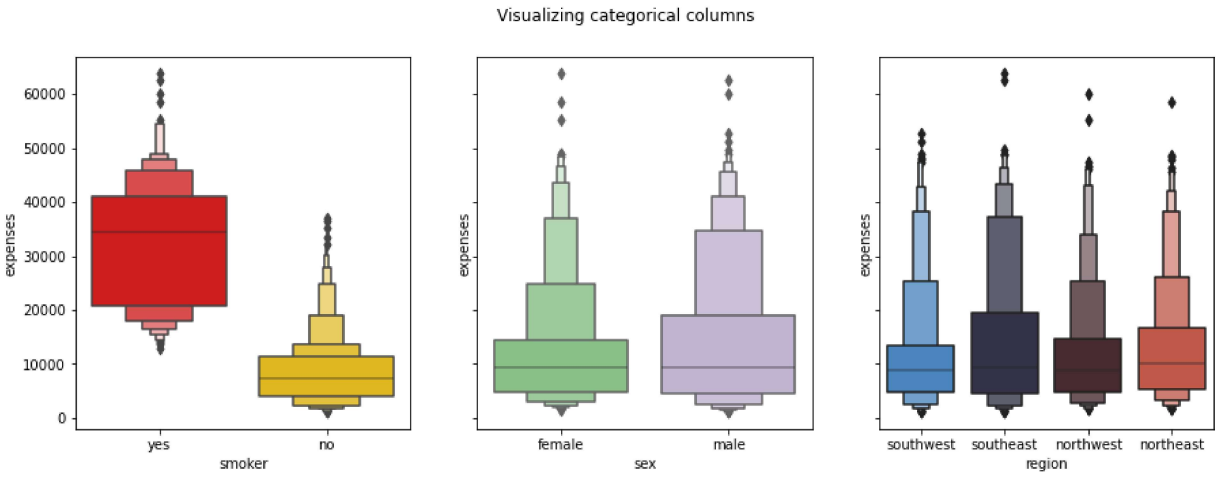
```
plt.figure(figsize=(5,5))
sns.barplot(x='sex', y='expenses',hue='smoker', data=df ,palette ='Reds')
plt.title('Expenses for smokers')
```



In [12]:

```
fig, axes = plt.subplots(1,3, figsize=(15,5), sharey=True)
fig.suptitle('Visualizing categorical columns')
sns.boxenplot(x='smoker', y= 'expenses', data=df, ax=axes[0] ,palette ='hot')
sns.boxenplot(x='sex', y= 'expenses', data=df, ax=axes[1] ,palette ='Accent')
sns.boxenplot(x='region', y= 'expenses', data=df, ax=axes[2] ,palette= 'icefire')
```

Out[12]: <AxesSubplot:xlabel='region', ylabel='expenses'>



In [ ]:

In [13]:
```
#categorical encoding
```

In [14]:
```
df.select_dtypes('object').columns
```

Out[14]:
```
Index(['sex', 'smoker', 'region'], dtype='object')
```

In [15]:
```
df['region'].value_counts()
```

Out[15]:
```
southeast    364
southwest    325
northwest    325
northeast    324
Name: region, dtype: int64
```

In [16]:
```
df['smoker'].value_counts()
```

Out[16]:
```
no     1064
yes     274
Name: smoker, dtype: int64
```

In [17]:
```
df['sex'].value_counts()
```

Out[17]:
```
male      676
female    662
Name: sex, dtype: int64
```

In [ ]:
```

```

In [18]:
```
#binary encoding


df['sex_en'] = df['sex'].replace({'male' :0 , 'female':1})
df['smoker_en'] = df['smoker'].replace({'no':0 ,'yes':1})
```

In [19]:
```
df.drop(columns = ['smoker' ,'sex'] ,inplace = True)
```

In [20]:
```
df
```

Out[20]:

	age	bmi	children	region	expenses	sex_en	smoker_en
0	19	27.9	0	southwest	16884.92	1	1
1	18	33.8	1	southeast	1725.55	0	0
2	28	33.0	3	southeast	4449.46	0	0
3	33	22.7	0	northwest	21984.47	0	0
4	32	28.9	0	northwest	3866.86	0	0
...	...	...	...	...	...	...	...
1333	50	31.0	3	northwest	10600.55	0	0
1334	18	31.9	0	northeast	2205.98	1	0
1335	18	36.9	0	southeast	1629.83	1	0
1336	21	25.8	0	southwest	2007.95	1	0
1337	61	29.1	0	northwest	29141.36	1	1

1338 rows × 7 columns

In [21]:
```
#multi categorical encoding
```

In [22]:
```
region_en = pd.get_dummies(df['region'])
```

In [23]:
```
df = pd.concat([df, region_en] , axis =1)
```

In [24]:
```
df.drop(columns = ['region'] ,inplace =True)
```

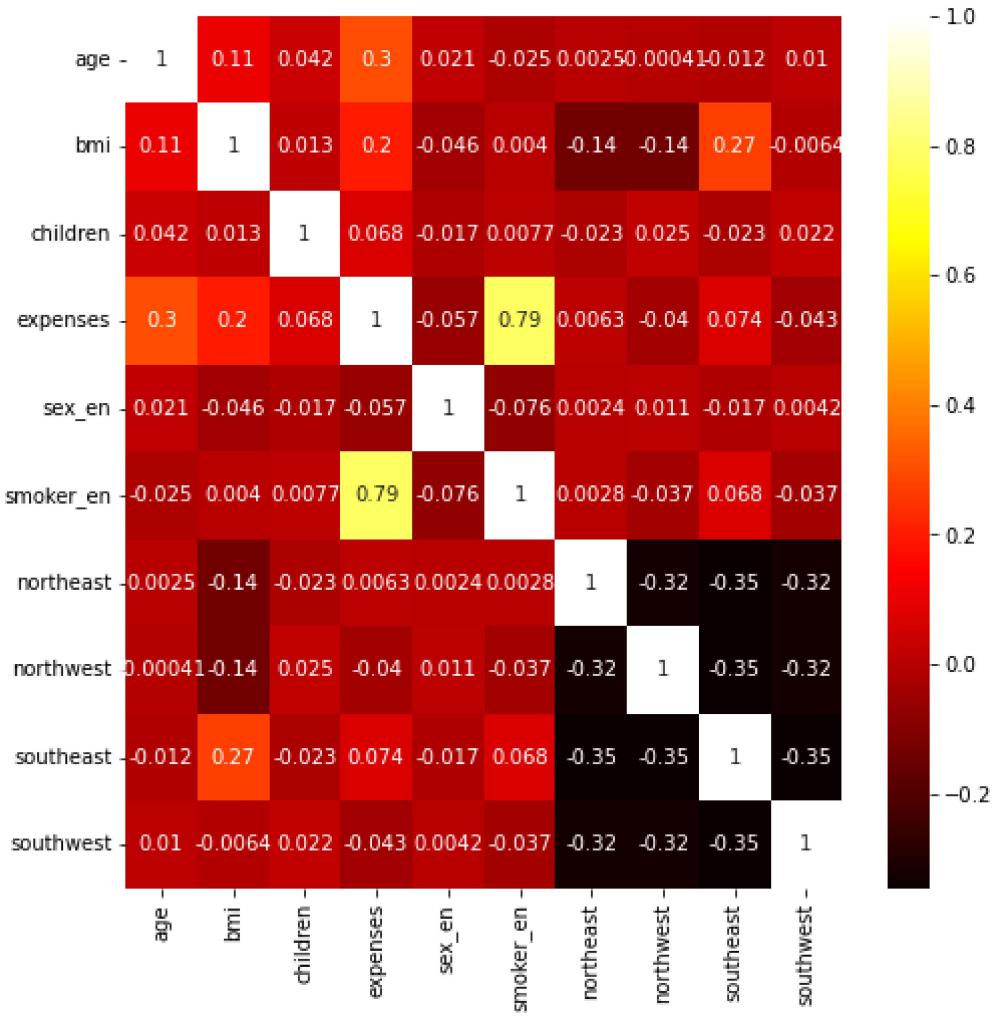
In [25]:
```
df
```

Out[25]:

	age	bmi	children	expenses	sex_en	smoker_en	northeast	northwest	southeast	soutl
0	19	27.9	0	16884.92	1	1	0	0	0	
1	18	33.8	1	1725.55	0	0	0	0	1	
2	28	33.0	3	4449.46	0	0	0	0	1	
3	33	22.7	0	21984.47	0	0	0	1	0	
4	32	28.9	0	3866.86	0	0	0	1	0	
...	...	...	...	...	...	...	...	...	...	
1333	50	31.0	3	10600.55	0	0	0	1	0	
1334	18	31.9	0	2205.98	1	0	1	0	0	
1335	18	36.9	0	1629.83	1	0	0	0	1	
1336	21	25.8	0	2007.95	1	0	0	0	0	
1337	61	29.1	0	29141.36	1	1	0	1	0	

In [26]:
```
plt.figure(figsize =(8,8))
correlation = df.corr()
sns.heatmap(correlation ,cmap= 'hot',annot =True)
```

Out[26]: <AxesSubplot:>



In [27]:
```
x = df.drop('expenses' ,axis = 1)
y= df['expenses']
```

In [28]:
```
x
```

Out[28]:

	age	bmi	children	sex_en	smoker_en	northeast	northwest	southeast	southwest
0	19	27.9	0	1	1	0	0	0	1
1	18	33.8	1	0	0	0	0	1	0
2	28	33.0	3	0	0	0	0	1	0
3	33	22.7	0	0	0	0	1	0	0
4	32	28.9	0	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...
1333	50	31.0	3	0	0	0	1	0	0
1334	18	31.9	0	1	0	1	0	0	0
1335	18	36.9	0	1	0	0	0	1	0
1336	21	25.8	0	1	0	0	0	0	1
1337	61	29.1	0	1	1	0	1	0	0

1338 rows × 9 columns

In [29]:
```
X_train, X_test, y_train, y_test = train_test_split(x,y, test_size=0.3, random_st
print("X_train shape: ", X_train.shape)
print("X_test shape: ", X_test.shape)
print("y_train shpae: ", y_train.shape)
print("y_test shape: ", y_test.shape)
```

```
X_train shape:  (936, 9)
X_test shape:  (402, 9)
y_train shpae:  (936,)
y_test shape:  (402,)
```

In [30]:
```
#mymodel
```

In [31]:
```
lr = LinearRegression()
lr.fit(X_train , y_train)
```

Out[31]:
```
LinearRegression()
```

In [32]:
```
test_pred = lr.predict(X_test)
test_pred.shape
```

Out[32]:
```
(402,)
```

In [33]:
```
lr.score(X_test ,y_test)
```

Out[33]:
```
0.7542883328348358
```

In [34]:
```
test_pred[:12]
```

Out[34]:
```
array([ 9023.16356496, 36422.7985755 ,  3013.9537084 , 11197.01122603,
       33813.03548902, 11547.53219375, 11370.65054188, 14443.52996653,
        5700.83417726, 10735.34239937,  9586.45496973, 12108.79851297])
```

In [35]:
```
# scaling data and then applying
```

In [36]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [37]:
```
scale = MinMaxScaler()
X_train_scale = scale.fit_transform(X_train)
X_test_scale = scale.transform(X_test)
```

In [38]:
```
lr = LinearRegression()
lr.fit(X_train_scale , y_train)
```

Out[38]:
```
LinearRegression()
```

In [39]:
```
test_pred1 = lr.predict(X_test_scale)
test_pred1.shape
```

Out[39]:
```
(402,)
```

In [40]:
```
lr.score(X_test_scale ,y_test)
```

Out[40]:
```
0.7542322227503546
```

In [ ]:
```

```