# Implementation of Multiple Linear Regression

In [1]:
```
import pandas as pd
import numpy as np
from sklearn import linear_model
```

In [4]:
```
df = pd.read_csv("car data.csv")
```

In [5]:
```
df
```

Out[5]:

	speed	car_age	experience	risk
0	200	15	5.0	85
1	90	17	13.0	20
2	165	12	4.0	93
3	110	20	NaN	60
4	140	5	3.0	82
5	115	2	8.0	10

In [12]:
```
df
```

Out[12]:

	speed	car_age	experience	risk
0	200	15	5.0	85
1	90	17	13.0	20
2	165	12	4.0	93
3	110	20	NaN	60
4	140	5	3.0	82
5	115	2	8.0	10

In [15]:
```
null_values = df.isnull().sum

# Print the result
# print(null_values)
```

In [16]:
```
print(null_values)
```

```
<bound method NDFrame._add_numeric_operations.<locals>.sum of    speed   car_
age  experience   risk
0   False    False       False  False
1   False    False       False  False
2   False    False       False  False
3   False    False        True  False
4   False    False       False  False
5   False    False       False  False>
```

In [17]:
```
null_count = df.isnull().sum()

# Print the result
print(null_count)
```

```
speed         0
car_age       0
experience    1
risk          0
dtype: int64
```

In [18]:
```
df.experience
```

Out[18]:
```
0     5.0
1    13.0
2     4.0
3     NaN
4     3.0
5     8.0
Name: experience, dtype: float64
```

In [19]:
```
df.experience.mean()
```

Out[19]:
```
6.6
```

In [20]:
```
df.experience.median()
```

Out[20]:
```
5.0
```

In [21]:
```
exp_fit= df.experience.median()
```

In [22]:
```
exp_fit
```

Out[22]:
```
5.0
```

In [23]:
```
df.experience = df.experience.fillna(exp_fit)
```

```
In [24]: df.experience

Out[24]: 0     5.0
         1    13.0
         2     4.0
         3     5.0
         4     3.0
         5     8.0
         Name: experience, dtype: float64
```

```
In [26]: reg = linear_model.LinearRegression()
```

```
In [29]: reg.fit(df[['speed','car_age','experience']],df.risk)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[29], line 1
----> 1 reg.fit(df[['speed','car_age','experience']],df.risk)

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:3767, in
DataFrame.__getitem__(self, key)
   3765     if is_iterator(key):
   3766         key = list(key)
-> 3767     indexer = self.columns._get_indexer_strict(key, "columns")[1]
   3769 # take() does not accept boolean indexers
   3770 if getattr(indexer, "dtype", None) == bool:

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:5
877, in Index._get_indexer_strict(self, key, axis_name)
   5874 else:
   5875     keyarr, indexer, new_indexer = self._reindex_non_unique(keyarr)
-> 5877 self._raise_if_missing(keyarr, indexer, axis_name)
   5879 keyarr = self.take(indexer)
   5880 if isinstance(key, Index):
   5881     # GH 42790 - Preserve name from an Index

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:5
941, in Index._raise_if_missing(self, key, indexer, axis_name)
   5938     raise KeyError(f"None of [{key}] are in the [{axis_name}]")
   5940 not_found = list(ensure_index(key)[missing_mask.nonzero()[0]].unique
())
-> 5941 raise KeyError(f"{not_found} not in index")

KeyError: "['speed'] not in index"
```

```
In [30]: print(df.columns)

Index(['speed ', 'car_age', 'experience', 'risk'], dtype='object')
```

In [31]:
```
reg.fit(df[['speed', 'car_age', 'experience']], df['risk'])
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[31], line 1
----> 1 reg.fit(df[['speed', 'car_age', 'experience']], df['risk'])

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:3767, in
DataFrame.__getitem__(self, key)
   3765     if is_iterator(key):
   3766         key = list(key)
-> 3767     indexer = self.columns._get_indexer_strict(key, "columns")[1]
   3769 # take() does not accept boolean indexers
   3770 if getattr(indexer, "dtype", None) == bool:

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:5
877, in Index._get_indexer_strict(self, key, axis_name)
   5874 else:
   5875     keyarr, indexer, new_indexer = self._reindex_non_unique(keyarr)
-> 5877 self._raise_if_missing(keyarr, indexer, axis_name)
   5879 keyarr = self.take(indexer)
   5880 if isinstance(key, Index):
   5881     # GH 42790 - Preserve name from an Index

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:5
941, in Index._raise_if_missing(self, key, indexer, axis_name)
   5938     raise KeyError(f"None of [{key}] are in the [{axis_name}]")
   5940 not_found = list(ensure_index(key)[missing_mask.nonzero()[0]].unique
())
-> 5941 raise KeyError(f"{not_found} not in index")

KeyError: "['speed'] not in index"
```

In [32]:
```
#Strip Leading/Trailing Spaces from Column Names:

#Sometimes column names may have invisible spaces. You can strip any leading/tı
# from the column names using:Strip Leading/Trailing Spaces from Column Names:

df.columns = df.columns.str.strip()
```

In [33]:
```
reg.fit(df[['speed', 'car_age', 'experience']], df['risk'])
```

Out[33]:
```
▾ LinearRegression
LinearRegression()
```

In [34]:
```
#Predicting risk when speed,car_age and experience are given

reg.predict([[160,10,5]])
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning:
X does not have valid feature names, but LinearRegression was fitted with fea
ture names
  warnings.warn(
```

Out[34]:
```
array([71.37146872])
```

In [35]:
```
reg.coef_
```

Out[35]:
```
array([ 0.33059217,  1.61053246, -6.20772074])
```

In [36]:
```
reg.intercept_
```

Out[36]:
```
33.410000910435855
```

In [37]:
```
# cross checking the predicted value with the value obtaned from the
# multiple linear regression equation as given below

160*0.33059217 + 10*1.61053246 + 5*-6.20772074 + 33.410000910435855
```

Out[37]:
```
71.37146901043586
```

In [ ]:
```
#Yes it is the same , Congratulations!!!
```