Importing Data

```
import pandas as pd

df = pd.read_csv("/content/quikr_car.csv")

df.head()
```

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
2	Maruti Suzuki Alto 800 Vxi	Maruti	2018	Ask For Price	22,000 kms	Petrol
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	3,25,000	28,000 kms	Petrol

Data Information

```
df.shape
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 892 entries, 0 to 891
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   name        892 non-null    object
 1   company     892 non-null    object
 2   year        892 non-null    object
 3   Price       892 non-null    object
 4   kms_driven  840 non-null    object
 5   fuel_type   837 non-null    object
dtypes: object(6)
memory usage: 41.9+ KB
```

Quality

```
# df['year'].unique()
# Year have many non-year values
# Year is in dataType object

# df['Price'].unique()
# Price has some String
# Price has , in it
# Price is in dataType object

# df['kms_driven'].unique()
# km_driven is in dataType Object
# It has 'kms' with integer
# It also have some nan values

# df['fuel_type'].unique()
# It has some nan values

# df['name'].unique()
# It is inconsistent so we'll take only starting three words
```

Cleaning

```
backup = df.copy()
```

```
df = df[df['year'].str.isnumeric()]
```

```
df['year'] = df['year'].astype(int)
```

```
<ipython-input-215-a849a2ca4f2a>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
  df['year'] = df['year'].astype(int)
```

```
df = df[df['Price'] != 'Ask For Price']
```

```
df["Price"] = df['Price'].str.replace("," , "").astype(int)
```

```
df['kms_driven'] = df['kms_driven'].str.split(' ').str.get(0).str.replace("," , "")
df = df[df['kms_driven'].str.isnumeric()]
df['kms_driven'] = df['kms_driven'].astype(int)
```

```
df = df[~df['fuel_type'].isna()]
```

```
df['name'] = df['name'].str.split(' ').str.slice(0,3).str.join(' ')
```

```
df = df.reset_index(drop = True)
```

```
df.info()
```

```
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 816 entries, 0 to 815
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   name        816 non-null    object
 1   company     816 non-null    object
 2   year        816 non-null    int64 
 3   Price       816 non-null    int64 
 4   kms_driven  816 non-null    int64 
 5   fuel_type   816 non-null    object
dtypes: int64(3), object(3)
memory usage: 38.4+ KB
```

	year	Price	kms_driven
count	816.000000	8.160000e+02	816.000000
mean	2012.444853	4.117176e+05	46275.531863
std	4.002992	4.751844e+05	34297.428044
min	1995.000000	3.000000e+04	0.000000
25%	2010.000000	1.750000e+05	27000.000000
50%	2013.000000	2.999990e+05	41000.000000
75%	2015.000000	4.912500e+05	56818.500000
max	2019.000000	8.500003e+06	400000.000000

```
# price has a outlier in it let's check it
df = df[df['Price']<4e6].reset_index(drop = True)


#Store this clean data in another csv file
df.to_csv('Cleaned car.csv')
```

Model Building

```
X = df.drop(columns = 'Price')
Y = df['Price']


from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test = train_test_split(X,Y, test_size=0.2)


from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
from sklearn.pipeline import make_pipeline


ohe = OneHotEncoder()
ohe.fit(X[ ['name' , 'company' , 'fuel_type'] ])
OneHotEncoder()
```

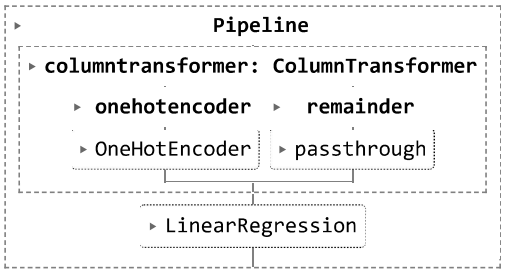
▾ OneHotEncoder
OneHotEncoder()

```
column_trans = make_column_transformer((OneHotEncoder(categories=ohe.categories_),['name' , 'company' , 'fuel_type']),
                                       remainder='passthrough')
```

```
lr = LinearRegression()
```

```
pipe = make_pipeline(column_trans , lr)
```

```
pipe.fit(X_train , Y_train )
```




```
y_pred = pipe.predict(X_test)
```

```
r2_score(Y_test,y_pred) #which is less so let's chcek the r2 score on different random states
```

```
0.6192582033068821
```

```
#This loop will check the r2 score of the model on different random state
scores = []
for i in range(1000):
  X_train,X_test,Y_train,Y_test = train_test_split(X,Y, test_size=0.2 , random_state=i)
  lr = LinearRegression()
  pipe = make_pipeline(column_trans,lr)
  pipe.fit(X_train,Y_train)
  y_pred = pipe.predict(X_test)
   scores.append(r2_score(Y_test,y_pred))
```

```
import numpy as np
np.argmax(scores)
```

```
433
```

```
scores[np.argmax(scores)]
```

```
0.8456515104452564
```

```
pipe.predict(pd.DataFrame(columns=X_test.columns,data=np.array(['Maruti Suzuki Swift','Maruti',2019,100,'Petrol']).resha
```

```
array([431098.74055388])
```

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y, test_size=0.2 , random_state=np.argmax(scores))
lr = LinearRegression()
pipe = make_pipeline(column_trans,lr)
pipe.fit(X_train,Y_train)
y_pred = pipe.predict(X_test)
r2_score(Y_test,y_pred)
```

```
0.8456515104452564
```

```
import pickle
pickle.dump(pipe,open('LinearRegressionModel.pkl','wb'))
```

```
pipe.predict(pd.DataFrame(columns=['name','company','year','kms_driven','fuel_type'],data=np.array(['Maruti Suzuki Swift
```

```
array([459113.49353657])
```

```
# pipe.steps[0][1].transformers[0][1].categories[0]
```