**Linear Regression on cars dataset**

**import pandas as pd**

**import seaborn as sns**

**data\_cars=pd.read\_csv("cars\_sampled.csv")**

**print(data\_cars)**

**cars\_data=data\_cars.copy()**

**print(cars\_data.info())**

**print(cars\_data.describe())**

**#formating float data**

**pd.set\_option("display.float\_format",lambda x:"%.3f"%x)**

**print(cars\_data.describe())**

**pd.set\_option("display.max\_columns",500)**

**print(cars\_data.describe())**

**#Dropping unwanted columns**

**col=["name","dateCrawled","dateCreated","postalCode",**

**"lastSeen"]**

**cars\_data=cars\_data.drop(columns=col,axis=1)**

**print(cars\_data)**

**#Removing duplicate records**

**cars\_data.drop\_duplicates(keep="first",inplace=True)**

**print(cars\_data)**

**#Data Cleaning**

**print(cars\_data.isnull().sum())**

**yearwise\_count=cars\_data["yearOfRegistration"].value\_counts().sort\_index()**

**print(yearwise\_count)**

**print(sum(cars\_data["yearOfRegistration"]>2018))**

**print(sum(cars\_data["yearOfRegistration"]<1950))**

**"""**

**sns.regplot(x="yearOfRegistration",y="price",**

**scatter=True,fit\_reg=False,**

**data=cars\_data)**

**"""**

**#working range 1950-2018**

**#variable price**

**price\_count=cars\_data["price"].value\_counts().sort\_index()**

**#sns.distplot(cars\_data["price"])**

**print(cars\_data["price"].describe())**

**#sns.boxplot(y=cars\_data["price"])**

**print(sum(cars\_data["price"]>150000))**

**print(sum(cars\_data["price"]<100))**

**#working range 100-150000**

**#variable powerPS**

**power\_count=cars\_data["powerPS"].value\_counts().sort\_index()**

**#sns.distplot(cars\_data["powerPS"])**

**print(cars\_data["powerPS"].describe())**

**#sns.boxplot(y=cars\_data["powerPS"])**

**print(sum(cars\_data["powerPS"]>500))**

**print(sum(cars\_data["powerPS"]<10))**

**#working range 10-500**

**#working range of Data**

**cars\_data=cars\_data[(cars\_data.yearOfRegistration<=2018)**

**&(cars\_data.yearOfRegistration>=1950)**

**&(cars\_data.price>=100)**

**&(cars\_data.price<=150000)**

**&(cars\_data.powerPS>=10)**

**&(cars\_data.powerPS<=500)]**

**print(cars\_data)**

**#combining yearOfRegistration and monthOfRegistration**

**cars\_data["monthOfRegistration"]/=12**

**#creating new variable Age by adding yearOfRegistration**

**#and monthOfRegistration**

**cars\_data["Age"]=(2018-cars\_data["yearOfRegistration"])+cars\_data["monthOfRegistration"]**

**cars\_data["Age"]=round(cars\_data["Age"],2)**

**print(cars\_data["Age"].describe())**

**#dropping yearOfRegistration and monthOfRegistration**

**cars\_data.drop(columns=["yearOfRegistration","monthOfRegistration"],axis=1,inplace=True)**

**"""**

**#Data visualization of Age,Price,powerPS**

**sns.distplot(cars\_data["Age"])**

**sns.boxplot(cars\_data["Age"])**

**sns.distplot(cars\_data["price"])**

**sns.boxplot(cars\_data["price"])**

**sns.distplot(cars\_data["powerPS"])**

**sns.boxplot(cars\_data["powerPS"])**

**#Age vs price**

**sns.regplot(x="Age",y="price",scatter=True,**

**fit\_reg=False,data=cars\_dat"""**

**#powerPs vs price**

**"""**

**sns.regplot(x="powerPS",y="price",scatter=True,**

**fit\_reg=False,data=cars\_data)"""**

**#variable seller**

**print(cars\_data["seller"].value\_counts())**

**seller\_tab=pd.crosstab(index=cars\_data["seller"],columns="counts",normalize=True)**

**#sns.countplot(x="seller",data=cars\_data)**

**print(seller\_tab)**

**#fewer cars have "cpmmercial => insignificant**

**print(cars\_data["offerType"].value\_counts())**

**offer\_tab=pd.crosstab(index=cars\_data["offerType"],columns="counts",normalize=True)**

**#sns.countplot(x="offerType",data=cars\_data)**

**print(offer\_tab)**

**#all cars have "offer"=> insignificant**

**#variable abtest**

**print(cars\_data["abtest"].value\_counts())**

**offer\_tab=pd.crosstab(index=cars\_data["abtest"],columns="counts",normalize=True)**

**#sns.boxplot(x="abtest",y="price",data=cars\_data)**

**print(offer\_tab)**

**#remove insignificant variables**

**col=["seller","offerType","abtest"]**

**cars\_data=cars\_data.drop(columns=col,axis=1)**

**cars\_copy=cars\_data.copy()**

**cars\_select1=cars\_data.select\_dtypes(exclude=[object])**

**correlation=cars\_select1.corr()**

**print(round(correlation,3))**

**print(cars\_select1.corr().loc[:,"price"].abs().sort\_values(ascending=False)[1:])**

**#ignoring missing values**

**"""**

**we are going to build a linear regression and random forest model**

**on two sets of data**

**1.Data obtained by omitting rows with any missing value**

**2.Data obtained by imputing the missing values**

**"""**

**cars\_omit=cars\_data.dropna(axis=0)**

**cars\_omit=pd.get\_dummies(cars\_omit,drop\_first=True)**

**#importing necessary libraries**

**import numpy as np**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.linear\_model import LinearRegression**

**from sklearn.ensemble import RandomForestRegressor**

**from sklearn.metrics import mean\_squared\_error**

**#building a model with omitted data**

**#seperating input and output features**

**x1=cars\_omit.drop(["price"],axis="columns",inplace=False)**

**y1=cars\_omit["price"]**

**#plotting the variable price**

**prices=pd.DataFrame({"1.Before":y1,"2.After":np.log(y1)})**

**#prices.hist()**

**#Transforming price as a logarithmic value**

**y1=np.log(y1)**

**#spliting data into test and train**

**x\_train,x\_test,y\_train,y\_test=train\_test\_split(x1,y1,**

**test\_size=0.3,**

**random\_state=3)**

**print(x\_train.shape,x\_test.shape)**

**print(y\_train.shape,y\_test.shape)**

**#finding the mean for test data value**

**base\_pred=np.mean(y\_test)**

**print(base\_pred)**

**#repeat the same value till length of test data**

**base\_pred=np.repeat(base\_pred,len(y\_test))**

**#Finding the RMSE**

**base\_rt\_sqr\_err=np.sqrt(mean\_squared\_error(y\_test,base\_pred))**

**print(base\_rt\_sqr\_err)**

**#Linea regression with omitted data**

**#setting intercept as True**

**lgr=LinearRegression(fit\_intercept=True)**

**#model**

**model\_lnr=lgr.fit(x\_train,y\_train)**

**#predicting model on test data**

**cars\_pred=model\_lnr.predict(x\_test)**

**#computing MSE and RMSE**

**lin\_mse1=mean\_squared\_error(y\_test,cars\_pred)**

**lin\_rmse1=np.sqrt(lin\_mse1)**

**print(lin\_rmse1)**

**#R Square value**

**r2\_lin\_test1=model\_lnr.score(x\_test,y\_test)**

**r2\_lin\_test2=model\_lnr.score(x\_train,y\_train)**

**print(r2\_lin\_test1,r2\_lin\_test2)**

**#Regression daignostics \_residual plot analysis**

**residual1=y\_test-cars\_pred**

**sns.regplot(x=cars\_pred,y=residual1,scatter=True,**

**fit\_reg=False)**

**print(residual1.describe())**