```
# Aditya Desai
# Roll no : A-16
# BE-A

import numpy as np
import pandas as pd

data = pd.read_csv("/home/kj-comp/ML/Uber dataset/uber.csv")

data.head
```

```
<bound method NDFrame.head of         Unnamed: 0
key  fare_amount  \
0         24238194    2015-05-07 19:52:06.0000003          7.5
1         27835199    2009-07-17 20:04:56.0000002          7.7
2         44984355   2009-08-24 21:45:00.00000061         12.9
3         25894730    2009-06-26 08:22:21.0000001          5.3
4         17610152  2014-08-28 17:47:00.000000188         16.0
...            ...                            ...          ...
199995    42598914   2012-10-28 10:49:00.00000053          3.0
199996    16382965    2014-03-14 01:09:00.0000008          7.5
199997    27804658   2009-06-29 00:42:00.00000078         30.9
199998    20259894    2015-05-20 14:56:25.0000004         14.5
199999    11951496   2010-05-15 04:08:00.00000076         14.1

                pickup_datetime  pickup_longitude  pickup_latitude  \
0       2015-05-07 19:52:06 UTC        -73.999817        40.738354
1       2009-07-17 20:04:56 UTC        -73.994355        40.728225
2       2009-08-24 21:45:00 UTC        -74.005043        40.740770
3       2009-06-26 08:22:21 UTC        -73.976124        40.790844
4       2014-08-28 17:47:00 UTC        -73.925023        40.744085
...                         ...               ...              ...
199995  2012-10-28 10:49:00 UTC        -73.987042        40.739367
199996  2014-03-14 01:09:00 UTC        -73.984722        40.736837
199997  2009-06-29 00:42:00 UTC        -73.986017        40.756487
199998  2015-05-20 14:56:25 UTC        -73.997124        40.725452
199999  2010-05-15 04:08:00 UTC        -73.984395        40.720077

        dropoff_longitude  dropoff_latitude  passenger_count
0              -73.999512         40.723217                1
1              -73.994710         40.750325                1
2              -73.962565         40.772647                1
3              -73.965316         40.803349                3
4              -73.973082         40.761247                5
...                   ...               ...              ...
199995         -73.986525         40.740297                1
199996         -74.006672         40.739620                1
199997         -73.858957         40.692588                2
199998         -73.983215         40.695415                1
199999         -73.985508         40.768793                1
```

```
[200000 rows x 9 columns]>

data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   Unnamed: 0         200000 non-null  int64  
 1   key                200000 non-null  object 
 2   fare_amount        200000 non-null  float64
 3   pickup_datetime    200000 non-null  object 
 4   pickup_longitude   200000 non-null  float64
 5   pickup_latitude    200000 non-null  float64
 6   dropoff_longitude  199999 non-null  float64
 7   dropoff_latitude   199999 non-null  float64
 8   passenger_count    200000 non-null  int64  
dtypes: float64(5), int64(2), object(2)
memory usage: 13.7+ MB

data ["pickup_datetime"] = pd.to_datetime(data["pickup_datetime"])

data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
 #   Column             Non-Null Count   Dtype              
---  ------             --------------   -----              
 0   Unnamed: 0         200000 non-null  int64              
 1   key                200000 non-null  object             
 2   fare_amount        200000 non-null  float64            
 3   pickup_datetime    200000 non-null  datetime64[ns, UTC]
 4   pickup_longitude   200000 non-null  float64            
 5   pickup_latitude    200000 non-null  float64            
 6   dropoff_longitude  199999 non-null  float64            
 7   dropoff_latitude   199999 non-null  float64            
 8   passenger_count    200000 non-null  int64              
dtypes: datetime64[ns, UTC](1), float64(5), int64(2), object(1)
memory usage: 13.7+ MB

#find missing values
data.isnull()

        Unnamed: 0    key  fare_amount  pickup_datetime 
pickup_longitude  \
0             False  False        False            False 
False
1             False  False        False            False
```

```
False
2             False  False          False             False
False
3             False  False          False             False
False
4             False  False          False             False
False
...             ...    ...            ...               ...
...
199995        False  False          False             False
False
199996        False  False          False             False
False
199997        False  False          False             False
False
199998        False  False          False             False
False
199999        False  False          False             False
False

        pickup_latitude  dropoff_longitude  dropoff_latitude
passenger_count
0                 False              False             False
False
1                 False              False             False
False
2                 False              False             False
False
3                 False              False             False
False
4                 False              False             False
False
...                 ...                ...               ...
...
199995            False              False             False
False
199996            False              False             False
False
199997            False              False             False
False
199998            False              False             False
False
199999            False              False             False
False

[200000 rows x 9 columns]

#find total numbers of missing values
data.isnull().sum()
```

```
Unnamed: 0           0
key                  0
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64

#drop the row if it has missing values
data.dropna(inplace = True)

data.isnull().sum()

Unnamed: 0           0
key                  0
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
dtype: int64

#Creating a MAchine Learning Model

#import lib
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# x is predictor variable
x = data.drop("fare_amount",axis = 1)

# y is target variable
y = data["fare_amount"]

# to apply model

x['pickup_datetime'] =
pd.to_numeric(pd.to_datetime(x['pickup_datetime']))
x = x.loc[:,x.columns.str.contains('^Unnamed')]

x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.2)

# Creating Linear Regression Model
```

```
lrmodel = LinearRegression()
lrmodel.fit(x_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
normalize=False)
```

```
# model is created

# prediction
pred = lrmodel.predict(x_test)

# Calculating RMSE
lrmodelrmse = np.sqrt(mean_squared_error(pred, y_test))
print ("RMSE error is: ",lrmodelrmse)
```

```
RMSE error is:  10.006628258191773
```

```
# Random Forest Regression
from sklearn.ensemble import RandomForestRegressor

# Create RFR Model
rfmodel = RandomForestRegressor(n_estimators = 100, random_state =
101)

#fit the forest

rfmodel.fit(x_train, y_train)
rfmodel_pred = rfmodel.predict(x_test)

# Calculating RMSE for RFR

rfmodel_rmse = np.sqrt(mean_squared_error(rfmodel_pred, y_test))
print ("RFR RMSE error is: ",rfmodel_rmse)
```

```
RFR RMSE error is:  12.162838146627962
```

```
# prediction

pred = lrmodel.predict(x_test)
print("hh",pred)
lrmodel.predict(x_test)
```

```
hh [11.35319023 11.35670956 11.3690579  ... 11.34600592 11.35385653
 11.36759394]
```

```
array([11.35319023, 11.35670956, 11.3690579 , ..., 11.34600592,
       11.35385653, 11.36759394])
```

```
from sklearn import metrics
# R2 score Linear Regression
metrics.r2_score(y_test,pred)
```

```
-7.606612153931991e-06
```

```
# R2 score RF Model
metrics.r2_score(y_test,rfmodel_pred)
```

```
-0.47739840592968164
```

```
# R2 score Linear Regression is 760% that means model does not fit.
# R2 score RF Model is : 52%

#Random Forest Model best fit for this dataset, is perfect
```