Data Science is what data scientists do. Data Science is nothing but gathering small or big amount of data for analysis.This analysis provides the solution to the problem using appropriate tools. In Data Science you must be curious and have ability of storytelling.Data Scientist is the one providing solution to problems using different tools.

As discussed in the videos and the reading material, data science can be applied to problems across different industries. What industry are you passionate about and would like to pursue a data science career in? **(1 mark)**

The real estate industry includes big data that helps to analyze various factors in financial decisions, offering insights about risk analysis. Thus, I am passionate about the real estate industry and would like to pursue a career in this industry.

A good ,well analysed Report is the one that includes a :
1) Cover Page
2) Table of Contents
3) Executive Summary/abstract section
4) Introduction Section
5) Methodology Section
6) Result Section
7) Discussion + Conclusion Section
8) References Section
9) Acknowledgements Section
10) Appendices Section

## Check for repeated rows

drop the duplcated rows, and keep only the first one (regardless of the price, which might conflict but ignore here)

```
In [7]:
print('Before drop train shape:', train.shape)
train.drop_duplicates(subset=['date', 'date_block_num', 'shop_id', 'item_id', 'item_cnt_day'],
                      keep='first', inplace=True)
train.reset_index(drop=True, inplace=True)
print('After drop train shape:', train.shape)


Before drop train shape: (2935849, 6)
After drop train shape: (2935825, 6)
```

Pands – datastructure and tools

Numpy --arrays and matrices

Matplotlib—pplots and graphs

Seaborn—heat maps and time series graphs

Sciket learn –machine learning

Importing dataset

Import pandas as pd

- df=pd.read_csv(data.csv)
- url = “https//:evliubfukbf”

  df=pd.read_csv(url)
- if headers are not given in dataset

  df=pd.read_csv(data.csv,header=None)

  headers=[“id”,”name”,”status”]

  df.columns=headers

```
import numpy as np  # useful for many scientific computing in Python
import pandas as pd # primary data structure library
from __future__ import print_function # adds compatibility to python 2
```

```
# install xlrd
!pip install xlrd

print('xlrd installed!')
```

```
df_can = pd.read_excel(
    'https://ibm.box.com/shared/static/lw190pt9zpy5bd1ptyg2aw15awomz9pu.xlsx',
    sheetname="Canada by Citizenship",
    skiprows=range(20),
    skip_footer = 2)
```

- 

Exporting the dataset

Path=”C:\Windows\download\datascience”

Df.to_csv(path)

To check shape of dataset – df.shape

To check first 5 rows of dataset—df.head()  df.tail(n)

df.info – gives top 30 and last 30 rows

To check number of rows – len(df)

## DATA WRANGLING

Dealing with missing values

- Df.dropna(subset=[“price”],axis=0,inplace=True)

  Axis=0 removes the row and axis =1 removes the column

- Mean= df[“losses”].mean()

  Df[“losses”].replace(np.nan,mean)

## Data Formating

Object,int,float,datetime

df.dtypes

to convert to another do by this

df[“price”] = df[“price”].astype(“int”)

df.descibe(include="all")  # unique elements in objects also

Renaming a column --

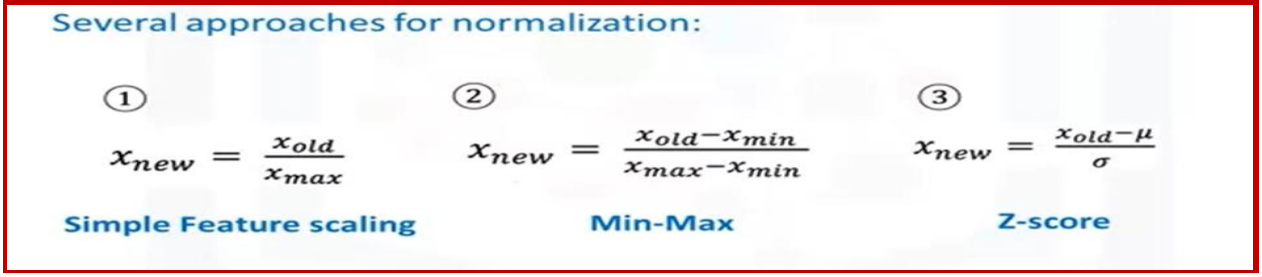
Eg in this I have changed acct to account key

```
for record in df:
    record['account_key'] = record['acct']
    del[record['acct']]
```

or you can do that by simply

df.rename(columns={"acct":"account_key"},inplace=True)

## DataNormalization




1) df['length'] = df['length']/df['length'].max()
2) df['length'] =( df['length']- df['length'].min())/

   df['length'].max()-df['length'].min()
3) df['length'] = (df['length']- df['length'].mean())/df['length'].std()

## Binning

Cotegoring price into low medium or high

bins=np.linespace(min(df["price"]),max(df["price"]),4)

group_names=["Low","Medium","High"]

df["price_binned"]=pd.cut(df["price"],bins,labels=group_names,include_lowest=True)

## one-Hot Encoding

pd.get_dummies(df["fuel"])

## Descriptive analysis

df1['area_type'].unique()

df1['area_type'].value_counts()

scatter plot

y=df[“price”]

x=df[“engine”]

plt.scatter(x,y)

plt.title(“Scatter plot of engine Vs price”)

plt.xlabel(“engine”)

plt.ylabel(“price”)

Group by

df_test = df[[“wheels”,”style”,”price”]]

df_grp = df_test.groupby([“wheels”,”style”],as_index=False).mean()

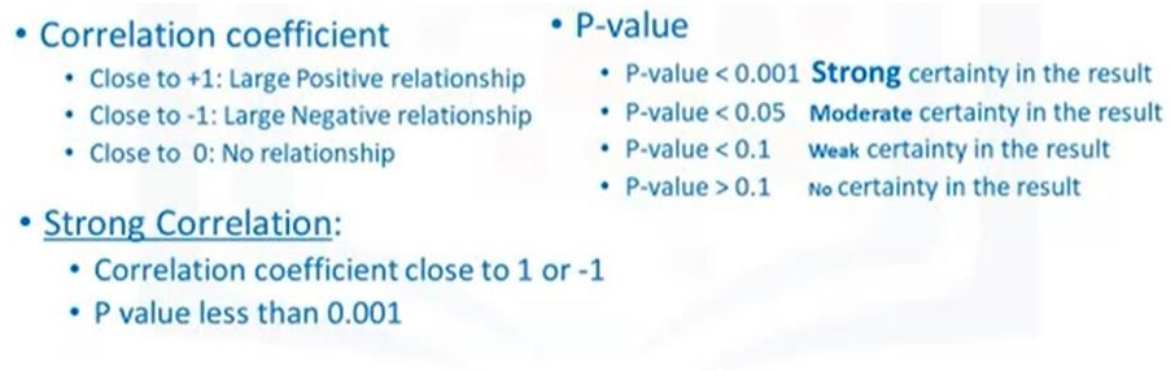
df_grp

correlation

sns.regplot(x=”engine-size”,y=”price”,data=df)

plt.ylim(0,)

pearson_coef,p_value = stats.pearsonr(df[“horsepower”],df[“price”])




Analysis of Varience(NOVA)

## Model development

from sklearn.linear_model import LinearRegression

lm=LinearRegression()

X = df[['highway-mpg']]

Y = df['price']

lm.fit(X,Y)

Y_predict = lm.predict(X)

lm.intercept_

lm.coef_

multiple inear regression

Z = df[[“horsepower”,”engine_size”,”colour”]]

lm.fit(Z,df[“price”])

y_predict = lm.predict(X)

## Regression Plots

Import seaborn as sns

sns.regplot(x="highway-mpg", y="price", data=df)

plt.ylim(0,)

//////////////////////

Y_hat = lm.predict(Z)

plt.figure(figsize=(width, height))

ax1 = sns.distplot(df['price'], hist=False, color="r", label="Actual Value")

sns.distplot(Yhat, hist=False, color="b", label="Fitted Values" , ax=ax1)

plt.title('Actual vs Fitted Values for Price')

plt.xlabel('Price (in dollars)')

plt.ylabel('Proportion of Cars')

plt.show()

plt.close()

//////////////////////////////////////////////////////

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
x_train, x_test, y_train, y_test = train_test_split(x,y,random_state =5)
knnclassifier = KNeighborsClassifier(n_neighbors=5)
knnclassifier.fit(x_train,y_train)
y_pred = knnclassifier.predict(x_test)
metrics.accuracy_score(y_test,y_pred)
```

//////////////////////////////////////////////

```
from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn import metrics

x_train, x_test, y_train, y_test = train_test_split(X,y,random_state =5)

# instantiate the model (using the default parameters)

logreg = LogisticRegression()

# fit the model with data

logreg.fit(x_train, y_train)

# predict the response for new observations

y_pred=logreg.predict(x_test)

metrics.accuracy_score(y_test,y_pred)
```

//////////////////////////////////////////////

```
from sklearn.model_selection import cross_val_score
knnclassifier = KNeighborsClassifier(n_neighbors=4)
print(cross_val_score(knnclassifier, x, y, cv=10, scoring ='accuracy').mea
n())
```

//////////////////////////////////////////////////

```
from sklearn.linear_model import LogisticRegression logreg = LogisticRegression() print
(cross_val_score(logreg, x, y, cv=10, scoring = 'accuracy').mean())
```

```
////////////////////////////////////
```

# locating better value of K for knn

```
k_range = list(range(1, 26))

scores = []

for k in k_range:

    knn = KNeighborsClassifier(n_neighbors=k)

    knn.fit(x_train, y_train)

    y_pred = knn.predict(x_test)

    scores.append(metrics.accuracy_score(y_test, y_pred))
```

```
# import Matplotlib (scientific plotting library)

import matplotlib.pyplot as plt

# allow plots to appear within the notebook

%matplotlib inline

# plot the relationship between K and testing accuracy

plt.plot(k_range, scores)

plt.xlabel('Value of K for KNN')

plt.ylabel('Testing Accuracy')
```

Matplotlib

```
import matplotlib.pyplot as plt
import numpy as np

x = np.random.randn(10000)
plt.hist(x, 100)
plt.title(r'Normal distribution with $\mu=0, \sigma=1$')
plt.savefig('matplotlib_histogram.png')
plt.show()
```

Scripting interface is plt

```
%matploylib inline                                  // this is backend

Import matplotlib as plt

plt.plot(5,5,'o')

plt.show()
```

%matploylib inline // this is backend and limitation is that we cannot change anything one plt.show

```
Import matplotlib as plt

plt.plot(5,5,'o')

plt.ylabel("Y")

plt.xlabel("X")
```

plt.title(“plting example”)

plt.show()

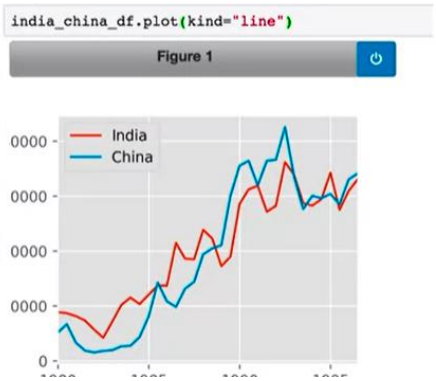
so another alternative is %matplotlib notebook

df.plot(kind=”line”)

india_china_df

| | India | China |
|---|---|---|
| **1980** | 8880 | 5123 |
| **1981** | 8670 | 6682 |
| **1982** | 8147 | 3308 |
| **1983** | 7338 | 1863 |
| **1984** | 5704 | 1527 |

```
india_china_df.plot(kind="line")
```




df[“India”].plot(kind=”hist”)

```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

```
years = list(map(str, range(1980, 2014)))

df_canada.loc['Haiti', years].plot(kind = 'line')
plt.title('Immigration from Haiti')
plt.ylabel('Number of immigrants')
plt.xlabel('Years')

plt.show()
```