**Functions**

1. **Machine Learning fundamentals in Python**
2. To get information on the predictor variables column types

print(dc\_listings.info())

1. Dropping the columns

dc\_listings.drop(dc\_listings.columns[[0,1,2, 4,14, 15,16,17, 18]], axis = 1, inplace = True)

1. Dropping rows with missing values

dc\_listings = dc\_listings.dropna(axis=0)

print(dc\_listings.isnull().sum())

print(dc\_listings.shape)

1. TO normalize the columns to neutralize large value differences

normalized\_listings = (dc\_listings - dc\_listings.mean())/(dc\_listings.std())

1. To find Euclidean distance between variables

first\_listing = normalized\_listings.iloc[0][['accommodates', 'bathrooms']]

fifth\_listing = normalized\_listings.iloc[4][['accommodates', 'bathrooms']]

first\_fifth\_distance = distance.euclidean(first\_listing, fifth\_listing)

1. Euclidian distance function

from scipy.spatial import distance

first\_listing = normalized\_listings.iloc[0][['accommodates', 'bathrooms']]

fifth\_listing = normalized\_listings.iloc[4][['accommodates', 'bathrooms']]

first\_fifth\_distance = distance.euclidean(first\_listing, fifth\_listing)

print(first\_fifth\_distance)

1. [KNeighborsRegressor class](http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html#sklearn.neighbors.KNeighborsRegressor) function

from sklearn.neighbors import KNeighborsRegressor

train\_df = normalized\_listings.iloc[0:2792]

test\_df = normalized\_listings.iloc[2792:]

knn = KNeighborsRegressor(n\_neighbors = 5, algorithm='brute')

train\_features = train\_df[['accommodates', 'bathrooms']]

train\_target = train\_df['price']

knn.fit(train\_features, train\_target)

predictions = knn.predict(test\_df[['accommodates', 'bathrooms']])

print(predictions)

1. [sklearn.metrics.mean\_squared\_error function()](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html" \l "sklearn.metrics.mean_squared_error)

from sklearn.metrics import mean\_squared\_error

train\_columns = ['accommodates', 'bathrooms']

knn = KNeighborsRegressor(n\_neighbors=5, algorithm='brute', metric='euclidean')

knn.fit(train\_df[train\_columns], train\_df['price'])

predictions = knn.predict(test\_df[train\_columns])

two\_features\_mse = mean\_squared\_error(test\_df["price"], predictions)

two\_features\_rmse = (two\_features\_mse) \*\* (1/2)

print(two\_features\_mse)

print(two\_features\_rmse)

1. To train a logistic regression model and predict

from sklearn.linear\_model import LogisticRegression

logistic\_model = LogisticRegression()

logistic\_model.fit(admissions[["gpa"]], admissions['admit'])

pred\_probs = logistic\_model.predict\_proba(admissions[["gpa"]])

plt.scatter(admissions["gpa"], pred\_probs[:,1])

plt.xlabel('GPA')

plt.ylabel('Prob of Admit students')

plt.title('Logistic Regression plot')

plt.show()

## Prediction

fitted\_labels = logistic\_model.predict(admissions[["gpa"]])

print(fitted\_labels)

1. Convert a single column from text categories to numbers

col = pandas.Categorical.from\_array(income["workclass"])

income["workclass"] = col.codes

print(income["workclass"].head(5))