Manemvenkat, Gauthamreddy – Contribution

Linear Regression was used to predict price vs points also province vs points.

Python model is as shown below.

**import** numpy **as** np  
import pandas **as** pd  
from sklearn.ensemble **import** RandomForestRegressor  
from sklearn.model\_selection **import** train\_test\_split  
rng = np.random.RandomState(0)  
df = pd.read\_csv(**'C://Users//manem//Downloads//new.csv'**)  
icols=[**'points'**]  
jcols=[**'price'**]  
X = pd.DataFrame(df.drop(df.columns[[0,1,2,3,5,6,7,8]], axis=1))  
Y = pd.DataFrame(df.drop(df.columns[[0,1,2,4,5,7,8]], axis=1))  
df = pd.concat([pd.DataFrame(X, columns=jcols),pd.DataFrame(Y, columns=icols)], axis=1)  
notnans = df[jcols].notnull().all(axis=1)  
df\_notnans = df[notnans]  
# Split into 75% train and 25% test  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(df\_notnans[jcols], df\_notnans[icols],  
 train\_size=0.75,  
 random\_state=4)  
regr\_multirf = RandomForestRegressor(max\_depth=30,random\_state=0)  
# Fit on the train data  
regr\_multirf.fit(X\_train, np.asarray(y\_train).ravel())  
df\_nans = df.loc[~notnans].copy()  
#print("Shape of df\_nans",df\_nans.shape)  
df\_nans[jcols]=regr\_multirf.predict(df\_nans[icols])  
print(df\_nans)

import numpy **as** np  
import pandas **as** pd  
from sklearn.ensemble **import** RandomForestRegressor  
from sklearn.model\_selection **import** train\_test\_split  
rng = np.random.RandomState(0)  
df = pd.read\_csv(**'C://Users//manem//Downloads//new.csv'**)  
icols=[**'points'**]  
jcols=[**'province2'**]  
X = pd.DataFrame(df.drop(df.columns[[0,1,2,3,4,6,7,8]], axis=1))  
Y = pd.DataFrame(df.drop(df.columns[[0,1,2,4,5,6,7,8]], axis=1))  
df = pd.concat([pd.DataFrame(X, columns=jcols),pd.DataFrame(Y, columns=icols)], axis=1)  
notnans = df[jcols].notnull().all(axis=1)  
df\_notnans = df[notnans]  
obj\_df = df\_notnans.select\_dtypes(include=[**'object'**]).copy()  
converted\_data=pd.factorize(obj\_df[**'province2'**])[0]  
df\_notnans[**'province2'**] = converted\_data  
# Split into 75% train and 25% test  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(df\_notnans[jcols], df\_notnans[icols],  
 train\_size=0.75,  
 random\_state=4)  
regr\_multirf = RandomForestRegressor(max\_depth=30,random\_state=0)  
# Fit on the train data  
regr\_multirf.fit(X\_train, np.asarray(y\_train).ravel())  
df\_nans = df.loc[~notnans].copy()  
#print("Shape of df\_nans",df\_nans.shape)  
df\_nans[jcols]=regr\_multirf.predict(df\_nans[icols])  
print(df\_nans)

Linear Regression

Regression is a data mining technique used to predict a range of numeric values (also called continuous values), given a particular dataset.

There were only two columns in our dataset which had numerical values.

After pre-processing the dataset we found that Price and Points were strongly correlated. I had a strong opinion that the price column and the points column were strongly co-related.

Sample code :

wine\_df=pd.read\_csv('C://Users//Desktop//new.csv')

#print(wine\_df.keys())

#print(wine\_df.\_data.shape)

#print(wine\_df.describe())

wine\_dataframe = pd.DataFrame(wine\_df)

#print(wine\_dataframe.head())

Y = wine\_dataframe['price']

X = wine\_dataframe['points']

Accuracy: 75.09%

