Data Conversion

```
Def dataConversion ( x , label , dtype):
   return x[label].astype(dtype,copy = False)
## features / cat  contains the column names
for col in features:
   df[col] = dataConversion (df,col,'float32') ## to covert to float32

for col in cat:
   df[col] = dataConversion (df,col,'category') ## to convert to  categotical

for col in cat:
   df[col] = dataConversion (df,col,'int') ## to convert to  integer
```

Imputation

```
from sklearn.preprocessing import Imputer
mean_imputer= Imputer(strategy="most_frequent")
## strategy = most_frequent
## strategy= mean
train = pd.DataFrame( mean_imputer.fit(X_train).fit_transform(X_train) , columns = X_train.columns )
train.isna().sum()
test = pd.DataFrame( mean_imputer.fit(X_train).fit_transform(X_test) , columns = X_test.columns )
test.isna().sum()
test.shape
train.shape
```

Standardize (gives output in terms of stand deviation values range from –inf to +inf) go for it if you can't determine range

```
from sklearn.preprocessing import StandardScaler
from sklearn import preprocessing
std_scale = preprocessing.StandardScaler().fit(X_train_std)
X_train_std = pd.DataFrame(std_scale.transform(X_train_std),columns=X_train_std.columns)
X_test_std  = pd.DataFrame(std_scale.transform(X_test_std),columns=X_test_std.columns)
```

MinMax Scaler if we known range

```
from sklearn.preprocessing import MinMaxScaler
range_scaler = MinMaxScaler()
range_scaler.fit(X_train)
```

```
X_train = pd.DataFrame(range_scale.transform(X_train),columns=X_train.columns)
X_test  = pd.DataFrame(range_scale.transform(X_test),columns=X_test.columns)
```