In [91]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

# Reading the data sets

In [92]:
```
import pandas as pd
train_data=pd.read_csv(r'C:\Users\newsoft\Desktop\train.csv')
test_data=pd.read_csv(r'C:\Users\newsoft\Desktop\test.csv')
```

In [93]:
```
train_data.head()
```

Out[93]:

| | ID | y | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 | ... | X375 | X376 | X377 | X378 | X379 | X380 | X382 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 130.81 | k | v | at | a | d | u | j | o | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **1** | 6 | 88.53 | k | t | av | e | d | y | l | o | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **2** | 7 | 76.26 | az | w | n | c | d | x | j | x | ... | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **3** | 9 | 80.62 | az | t | n | f | d | x | l | e | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **4** | 13 | 78.02 | az | v | n | f | d | h | d | n | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

5 rows × 378 columns

In [94]:
```
test_data.head()
```

Out[94]:

| | ID | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 | X10 | ... | X375 | X376 | X377 | X378 | X379 | X380 | X382 | X3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | az | v | n | f | d | t | a | w | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **1** | 2 | t | b | ai | a | d | b | g | y | 0 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 | |
| **2** | 3 | az | v | as | f | d | a | j | j | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **3** | 4 | az | l | n | f | d | z | l | n | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **4** | 5 | w | s | as | c | d | y | i | m | 0 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |

5 rows × 377 columns

In [95]:
```
print(train_data.shape)
```

```
(4209, 378)
```

In [96]:
```
print(test_data.shape)
```

```
(4209, 377)
```

In [97]:
```
for i in train_data.columns:
    data_type = train_data[i].dtype
    if data_type == 'object':
     print (i)
```

```
X0
X1
X2
X3
X4
X5
X6
X8
```

# Removing the variables having zero variance

In [98]:
```
variance=pow(train_data.drop(columns={'ID','y'}).std(),2).to_dict()
null_count = 0
for key,value in variance.items():
     if(value==0):
            print('Name=',key)
            null_count=null_count+1
print('Number of columns which has zero variance=',null_count)
```

```
Name= X11
Name= X93
Name= X107
Name= X233
Name= X235
Name= X268
Name= X289
Name= X290
Name= X293
Name= X297
Name= X330
Name= X347
Number of columns which has zero variance= 12
```

In [99]:
```
train_data=train_data.drop(columns={'X11','X93','X107','X233','X235','X268','X28
train_data.head()
```

Out[99]:

| | ID | y | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 | ... | X375 | X376 | X377 | X378 | X379 | X380 | X382 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 130.81 | k | v | at | a | d | u | j | o | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **1** | 6 | 88.53 | k | t | av | e | d | y | l | o | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **2** | 7 | 76.26 | az | w | n | c | d | x | j | x | ... | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **3** | 9 | 80.62 | az | t | n | f | d | x | l | e | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **4** | 13 | 78.02 | az | v | n | f | d | h | d | n | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

5 rows × 366 columns

In [100]:
```
train_data.shape
```

Out[100]:
```
(4209, 366)
```

# check for null and unique values for test and train data sets

In [101]:
```
train_data.isnull().sum().any()
```

Out[101]:
```
False
```

# Apply Label Encoder

In [102]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [103]:
```
train_data_feature = train_data.drop(columns={'y','ID'})
train_data_target=train_data.y
```

In [104]:
```
train_data_feature.head()
```

Out[104]:

| | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 | X10 | X12 | ... | X375 | X376 | X377 | X378 | X379 | X380 | X382 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | k | v | at | a | d | u | j | o | 0 | 0 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **1** | k | t | av | e | d | y | l | o | 0 | 0 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **2** | az | w | n | c | d | x | j | x | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **3** | az | t | n | f | d | x | l | e | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **4** | az | v | n | f | d | h | d | n | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

5 rows × 364 columns

In [105]:
```
train_data_target.head()
```

Out[105]:
```
0    130.81
1     88.53
2     76.26
3     80.62
4     78.02
Name: y, dtype: float64
```

In [106]:
```
print(train_data_feature.shape)
```

```
(4209, 364)
```

In [107]:
```
print(train_data_target.shape)
```

```
(4209,)
```

In [108]:
```
train_data_feature.describe(include='object')
```

Out[108]:

| | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 |
|---|---|---|---|---|---|---|---|---|
| **count** | 4209 | 4209 | 4209 | 4209 | 4209 | 4209 | 4209 | 4209 |
| **unique** | 47 | 27 | 44 | 7 | 4 | 29 | 12 | 25 |
| **top** | z | aa | as | c | d | v | g | j |
| **freq** | 360 | 833 | 1659 | 1942 | 4205 | 231 | 1042 | 277 |

In [109]:
```
le=LabelEncoder()
train_data_feature['X0']=le.fit_transform(train_data_feature.X0)
train_data_feature['X1']=le.fit_transform(train_data_feature.X1)
train_data_feature['X2']=le.fit_transform(train_data_feature.X2)
train_data_feature['X3']=le.fit_transform(train_data_feature.X3)
train_data_feature['X4']=le.fit_transform(train_data_feature.X4)
train_data_feature['X5']=le.fit_transform(train_data_feature.X5)
train_data_feature['X6']=le.fit_transform(train_data_feature.X6)
train_data_feature['X8']=le.fit_transform(train_data_feature.X8)
```

# Perfoming Dimensionality Reduction

In [110]:
```
print(train_data_feature.shape)
print(train_data_target.shape)
```

```
(4209, 364)
(4209,)
```

In [111]:
```
train_data_feature.head()
```

Out[111]:

| | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 | X10 | X12 | ... | X375 | X376 | X377 | X378 | X379 | X380 | X382 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 32 | 23 | 17 | 0 | 3 | 24 | 9 | 14 | 0 | 0 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **1** | 32 | 21 | 19 | 4 | 3 | 28 | 11 | 14 | 0 | 0 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **2** | 20 | 24 | 34 | 2 | 3 | 27 | 9 | 23 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **3** | 20 | 21 | 34 | 5 | 3 | 27 | 11 | 4 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **4** | 20 | 23 | 34 | 5 | 3 | 12 | 3 | 13 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

5 rows × 364 columns

In [112]:
```
train_data_target.head()
```

Out[112]:
```
0    130.81
1     88.53
2     76.26
3     80.62
4     78.02
Name: y, dtype: float64
```

In [113]:
```
from sklearn.decomposition import PCA
pca=PCA(n_components=.95)
pca.fit(train_data_feature,train_data_target)
```

Out[113]:
```
PCA(n_components=0.95)
```

In [114]:
```
train_data_feature_trans = pca.fit_transform(train_data_feature)
print(train_data_feature_trans.shape)
```

```
(4209, 6)
```

# predict test_df valuesusing XGBoost

Building model using the train data set

In [115]:
```
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score,mean_squared_error
from math import sqrt
```

In [116]:
```
train_x,test_x,train_y,test_y = train_test_split(train_data_feature_trans,train_d
print(train_x.shape)
print(train_y.shape)
print(test_x.shape)
print(test_y.shape)
```

```
(2946, 6)
(2946,)
(1263, 6)
(1263,)
```

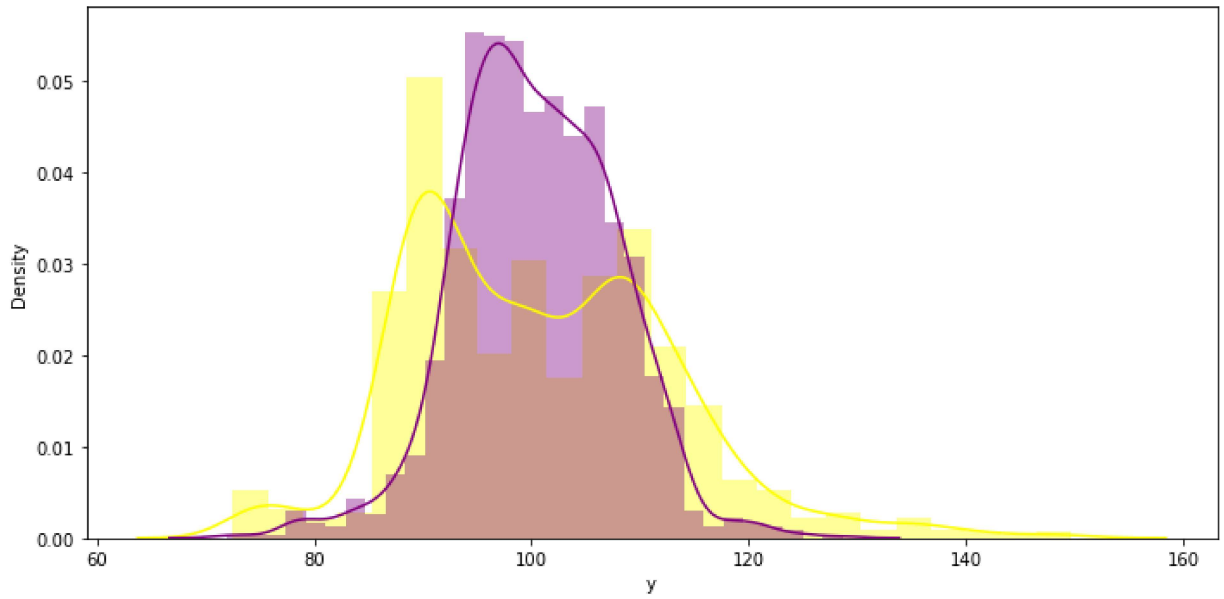
XGBoost's hyperparameters tuning manually

In [117]:
```
xgb_reg=xgb.XGBRegressor(objective='reg:linear',colsample_bytree=0.3,learning_rat
model=xgb_reg.fit(train_x,train_y)
print('RMSE=',sqrt(mean_squared_error(model.predict(test_x),test_y)))
```

```
[23:33:03] WARNING: c:\ci\xgboost-split_1638290375667\work\src\objective\regres
sion_obj.cu:188: reg:linear is now deprecated in favor of reg:squarederror.
RMSE= 12.288794806074309
```

RMSE is 12.29 approximately after tuning the hyperparameters to attain minimum RMSE

In [118]:
```
import matplotlib.pyplot as plt
import seaborn as sns
pred_test_y=model.predict(test_x)
plt.figure(figsize=(10,5))
sns.distplot(test_y[test_y<150],color="yellow",label="Actual value")
sns.distplot(pred_test_y[pred_test_y<150],color="purple",label="predicted value")
plt.tight_layout()
```




# k-fold cross validation using XGBoost

In [148]:
```
dmatrix_train=xgb.DMatrix(data=train_data_feature_trans,label=train_data_target)
params={'objective':'reg:linear','colsample_bytree':0.3,'learning_rate':0.3,'max_
model_cv=xgb.cv(dtrain=dmatrix_train,params=params,nfold=3,num_boost_round=50,ear
model_cv.tail(7)
```

```
[00:16:39] WARNING: c:\ci\xgboost-split_1638290375667\work\src\objective\regres
sion_obj.cu:188: reg:linear is now deprecated in favor of reg:squarederror.
[00:16:39] WARNING: c:\ci\xgboost-split_1638290375667\work\src\objective\regres
sion_obj.cu:188: reg:linear is now deprecated in favor of reg:squarederror.
[00:16:39] WARNING: c:\ci\xgboost-split_1638290375667\work\src\objective\regres
sion_obj.cu:188: reg:linear is now deprecated in favor of reg:squarederror.
```

Out[148]:

| | train-rmse-mean | train-rmse-std | test-rmse-mean | test-rmse-std |
|---|---|---|---|---|
| **32** | 8.250753 | 0.159901 | 11.120521 | 0.692890 |
| **33** | 8.189093 | 0.143499 | 11.125260 | 0.692265 |
| **34** | 8.123231 | 0.150525 | 11.122377 | 0.685467 |
| **35** | 8.080281 | 0.134426 | 11.119444 | 0.687319 |
| **36** | 8.033158 | 0.129848 | 11.118596 | 0.683865 |
| **37** | 7.981515 | 0.122538 | 11.114965 | 0.689016 |
| **38** | 7.944047 | 0.115245 | 11.114256 | 0.693008 |

By using k-fold cross validation,RMSE is reduced by approximately 10%,RMSE=11.1

# Prediction on test data set using XGBoost

Preparing test data set

In [120]:
```
test_data=test_data.drop(columns={'X11','X93','X107','X233','X235','X268','X289',
test_data.shape
```

Out[120]: (4209, 365)

In [121]:
```
test_data.head()
```

Out[121]:

| | ID | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 | X10 | ... | X375 | X376 | X377 | X378 | X379 | X380 | X382 | X3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | az | v | n | f | d | t | a | w | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **1** | 2 | t | b | ai | a | d | b | g | y | 0 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 | |
| **2** | 3 | az | v | as | f | d | a | j | j | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **3** | 4 | az | l | n | f | d | z | l | n | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **4** | 5 | w | s | as | c | d | y | i | m | 0 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |

5 rows × 365 columns

In [122]:
```
test_data.isnull().sum().any()
```

Out[122]: False

In [123]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder
```

In [124]:
```
test_data_feature=test_data.drop(columns={'ID'})
print(test_data_feature.shape)
```

(4209, 364)

In [125]:
```
test_data_feature.head()
```

Out[125]:

| | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 | X10 | X12 | ... | X375 | X376 | X377 | X378 | X379 | X380 | X382 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | az | v | n | f | d | t | a | w | 0 | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| **1** | t | b | ai | a | d | b | g | y | 0 | 0 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **2** | az | v | as | f | d | a | j | j | 0 | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| **3** | az | l | n | f | d | z | l | n | 0 | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| **4** | w | s | as | c | d | y | i | m | 0 | 0 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

5 rows × 364 columns

In [126]:
```
test_data_feature.describe(include='object')
```

Out[126]:

| | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 |
|---|---|---|---|---|---|---|---|---|
| **count** | 4209 | 4209 | 4209 | 4209 | 4209 | 4209 | 4209 | 4209 |
| **unique** | 49 | 27 | 45 | 7 | 4 | 32 | 12 | 25 |
| **top** | ak | aa | as | c | d | v | g | e |
| **freq** | 432 | 826 | 1658 | 1900 | 4203 | 246 | 1073 | 274 |

In [128]:
```
le=LabelEncoder()
test_data_feature['X0']=le.fit_transform(test_data_feature.X0)
test_data_feature['X1']=le.fit_transform(test_data_feature.X1)
test_data_feature['X2']=le.fit_transform(test_data_feature.X2)
test_data_feature['X3']=le.fit_transform(test_data_feature.X3)
test_data_feature['X4']=le.fit_transform(test_data_feature.X4)
test_data_feature['X5']=le.fit_transform(test_data_feature.X5)
test_data_feature['X6']=le.fit_transform(test_data_feature.X6)
test_data_feature['X8']=le.fit_transform(test_data_feature.X8)
```

In [129]:
```
test_data_feature.head()
```

Out[129]:

| | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X8 | X10 | X12 | ... | X375 | X376 | X377 | X378 | X379 | X380 | X382 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 21 | 23 | 34 | 5 | 3 | 26 | 0 | 22 | 0 | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| **1** | 42 | 3 | 8 | 0 | 3 | 9 | 6 | 24 | 0 | 0 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **2** | 21 | 23 | 17 | 5 | 3 | 0 | 9 | 9 | 0 | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| **3** | 21 | 13 | 34 | 5 | 3 | 31 | 11 | 13 | 0 | 0 | ... | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| **4** | 45 | 20 | 17 | 2 | 3 | 30 | 8 | 12 | 0 | 0 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

5 rows × 364 columns

In [130]:
```
test_data_feature.shape
```

Out[130]:
```
(4209, 364)
```

In [131]:
```
test_data_feature.dtypes.value_counts()
```

Out[131]:
```
int64    356
int32      8
dtype: int64
```

In [132]:
```
pca.fit(test_data_feature)
```

Out[132]:
```
PCA(n_components=0.95)
```

In [134]:
```
test_data_feature_trans=pca.fit_transform(test_data_feature)
print(test_data_feature_trans.shape)
```

```
(4209, 6)
```

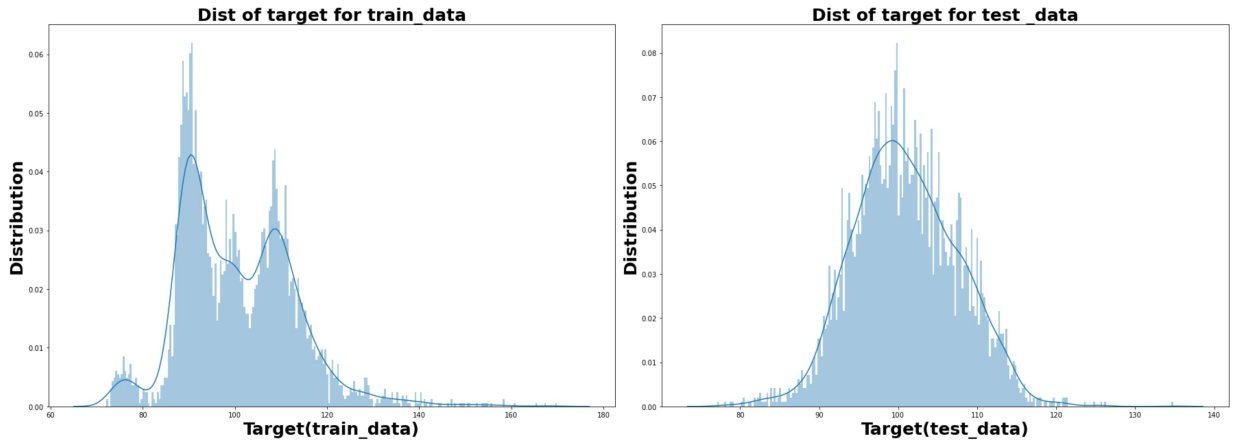
In [137]:
```
print(test_data_feature_trans)
```

```
[[ 14.58336183  14.16672593  13.53857566   2.40835691  11.31942221
    6.94220721]
 [-15.25161267  -7.73675643  -7.45495068  -2.66203503  11.59379316
    1.15940345]
 [ 11.8564649   -1.68017324  -9.9896148   14.91886587  -1.08886021
   -2.69130553]
 ...
 [-13.44644008   3.2885825   -6.85236431  18.91025575  11.32365564
    3.22410016]
 [ 24.92612317  -4.89888683 -10.16941028  11.44337736   5.90178724
    4.55323232]
 [-15.38430989  -7.73425491 -15.4930104   -0.5595126    4.7793639
    1.0829113 ]]
```

In [138]:
```
test_pred=model.predict(test_data_feature_trans)
test_pred
```

Out[138]:
```
array([ 86.12015 ,  92.929794,  98.74635 , ...,  92.836525, 118.76457 ,
        98.46741 ], dtype=float32)
```

In [147]:
```
fig,ax=plt.subplots(1,2,figsize=(25,9))
train_plot=sns.distplot(train_data_target[train_data_target<250],bins=250,kde=Tru
train_plot.set_xlabel('Target(train_data)',weight='bold',size=25)
train_plot.set_ylabel('Distribution',weight='bold',size=25)
train_plot.set_title('Dist of target for train_data',weight='bold',size=25)
test_plot=sns.distplot(test_pred[test_pred<250],bins=250,kde=True,ax=ax[1])
test_plot.set_xlabel('Target(test_data)',weight='bold',size=25)
test_plot.set_ylabel('Distribution',weight='bold',size=25)
test_plot.set_title('Dist of target for test _data',weight='bold',size=25)
plt.tight_layout()
```




It is the Pictorial view for comaprision between target for training data_set and predicted target for testing data_set