In [1]:

```
import warnings
warnings.filterwarnings("ignore")
import os
import pandas as pd
import matplotlib
matplotlib.use(u'nbAgg')
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.manifold import TSNE
from xgboost import XGBClassifier
from sklearn.model_selection import RandomizedSearchCV,GridSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.calibration import CalibratedClassifierCV
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import log_loss
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.ensemble import RandomForestClassifier
from tqdm import tqdm
tqdm.pandas()
```

In [2]:

```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
from plotly.subplots import make_subplots
import joblib as jb
import pickle as pk
import sys
import gc
```

In [3]:

```
os.chdir("D:/LargeDatasets/MicrosoftMalware/")
```

Final DataSet Preparation

In [4]:

```
final_files = os.listdir("./final_features/")
```

In [5]:

```
df=pd.read_csv("./final_features/"+final_files[0])
print(final_files[0]+" merged")
for i in range(1,len(final_files)):
    df = pd.merge(df, pd.read_csv("./final_features/"+final_files[i]),on=['ID','Class'], how='left')
    print(final_files[i]+" merged")
df.shape
```

```
asm_file_count_features.csv merged
asm_img_features.csv merged
byte_four_gram_hash_encoded_features.csv merged
byte_img_features.csv merged
byte_two_gram_features.csv merged
one_gram_byte.csv merged
```

Out[5]:

```
(10868, 9560)
```

In [6]:

```
df.head()
```

Out[6]:

	ID	HEADER:	.text:	.Pav:	.idata:	.data:	.bss:	.rdata:	.edata:	.rsrc:	...	f0	f4	f7	f8	f9
0	01azqd4InC7m9JpocGv5	18	22430	0	1158	1366754	0	1794	0	0	...	3685	3209	2804	3687	3101
1	01IsoiSMh5gxyDYTl4CB	0	109939	0	616	24568	0	26405	0	0	...	6813	519	451	6536	439
2	01jsnpXSAlgw6aPeDxrU	18	68883	0	304	662	0	1093	0	0	...	2619	2456	2325	2358	2242
3	01kcPWA9K2BOxQeS5Rju	19	744	0	127	57	0	323	0	3	...	886	603	478	873	485
4	01SuzwMJEIXsK7A8dQbl	18	10368	0	206	4595	92	0	0	3	...	374	492	847	947	350

5 rows × 9560 columns

In [7]:

```
y= df["Class"]
X = df.drop(["ID","Class"],axis=1)
```

Train Test split

In [8]:

```
X_tr,X_tes,y_tr,y_tes = train_test_split(X,y,test_size=0.18,random_state=13,stratify=y)
```

In [9]:

```
scaler = StandardScaler()
scaler.fit(X_tr)
X_train = scaler.transform(X_tr)
X_test = scaler.transform(X_tes)
print("="*100)
print("X_train Shape:",X_train.shape," Y_train shape",y_tr.shape)
print("X_test Shape:",X_test.shape," Y_test shape",y_tes.shape)
print("="*100)
```

```
====================================================================================================
X_train Shape: (8911, 9558)  Y_train shape (8911,)
X_test Shape: (1957, 9558)  Y_test shape (1957,)
====================================================================================================
```

Modelling

Observations and Models applied::

- Data is not to large and have large no.of features and our metric is logloss so we can try simple linear model like LogisticRegression.
- In EDA We saw that malwares belong to same class are almost close or packed together files using various features, i.e, a neat clusters are forming when reduced to lower dim so KNN may also work well.
- Dataset is non-linear in nature and dataset size and no of features are relatively almost same and not very large, so SVM with rbf kernel may also work well.
- Data is not sparse since we have done feature selection, so Ensembles like RandomForestClassifier and GradientBoostingClassifier must work really well.

In [11]:

```
# Ref: MicrosoftMalwareDetection.ipynb provided by Applied AI

def plot_confusion_matrix(test_y, predict_y):
    C = confusion_matrix(test_y, predict_y)
    misclassif = (len(test_y)-np.trace(C))/len(test_y)*100
    print("Number of misclassified points ",misclassif)
    A =(((C.T)/(C.sum(axis=1))).T)

    B =(C/C.sum(axis=0))

    labels = [1,2,3,4,5,6,7,8,9]
    cmap=sns.light_palette("green")
    # representing A in heatmap format
    print("-"*50, "Confusion matrix", "-"*50)
    plt.figure(figsize=(10,5))
    sns.heatmap(C, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, yticklabels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.show()

    print("-"*50, "Precision matrix", "-"*50)
    plt.figure(figsize=(10,5))
    sns.heatmap(B, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, yticklabels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.show()
    print("Sum of columns in precision matrix",B.sum(axis=0))

    # representing B in heatmap format
    print("-"*50, "Recall matrix"    , "-"*50)
    plt.figure(figsize=(10,5))
    sns.heatmap(A, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, yticklabels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.show()
    print("Sum of rows in precision matrix",A.sum(axis=1))
    return misclassif
```

In [12]:

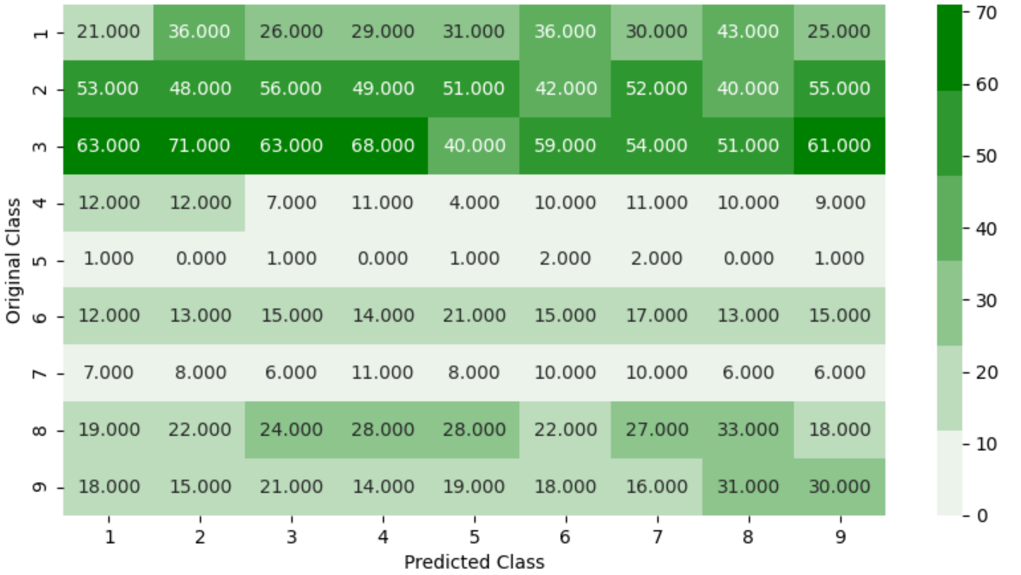
```
summary = [["Model","Log-Loss","Misclassified Percent"]]
```

Random Model

In [13]:

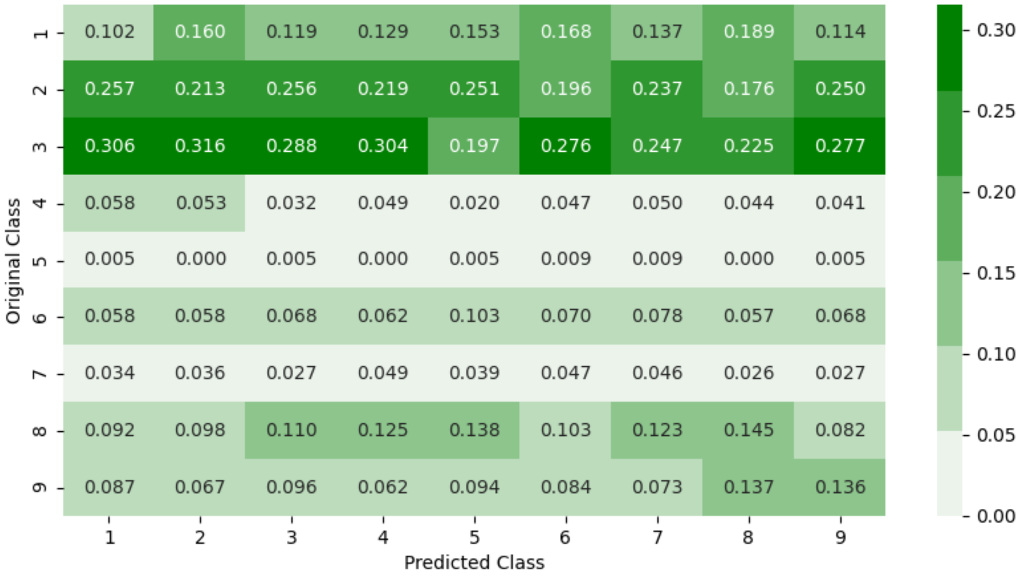
```
# ref: https://stackoverflow.com/a/18662466/4084039
# Ref: MicrosoftMalwareDetection.ipynb provided by Applied AI

models ={}
test_data_len = X_tes.shape[0]


# Test-Set error.
#we create a output array that has exactly same as the test data
test_predicted_y = np.zeros((test_data_len,9))
for i in range(test_data_len):
    rand_probs = np.random.rand(1,9)
    test_predicted_y[i] = ((rand_probs/sum(sum(rand_probs)))[0])
loss = log_loss(y_tes,test_predicted_y, eps=1e-15)
print("Log loss on Test Data using Random Model",loss)

predicted_y =np.argmax(test_predicted_y, axis=1)
misclassif = plot_confusion_matrix(y_tes, predicted_y+1)
```

```
Log loss on Test Data using Random Model 2.4558532128466006
Number of misclassified points  88.1451200817578
-------------------------------------------------- Confusion matrix -----------------------------------
---------------
```



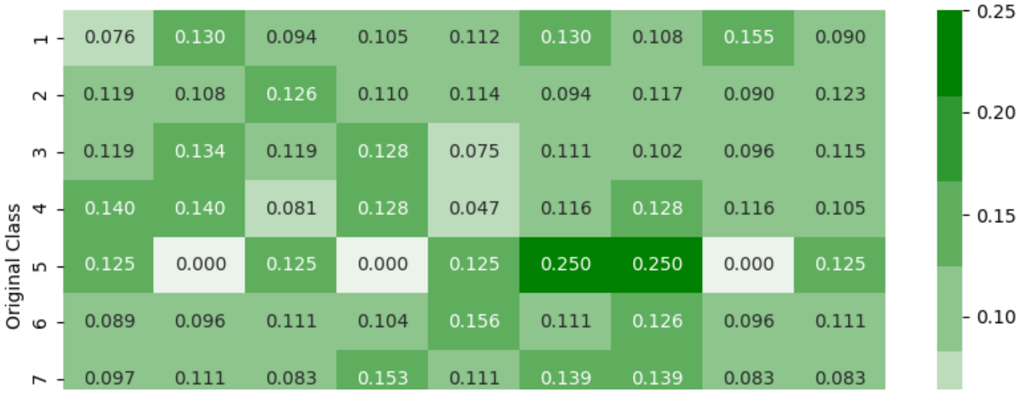
	1	2	3	4	5	6	7	8	9
1	21.000	36.000	26.000	29.000	31.000	36.000	30.000	43.000	25.000
2	53.000	48.000	56.000	49.000	51.000	42.000	52.000	40.000	55.000
3	63.000	71.000	63.000	68.000	40.000	59.000	54.000	51.000	61.000
4	12.000	12.000	7.000	11.000	4.000	10.000	11.000	10.000	9.000
5	1.000	0.000	1.000	0.000	1.000	2.000	2.000	0.000	1.000
6	12.000	13.000	15.000	14.000	21.000	15.000	17.000	13.000	15.000
7	7.000	8.000	6.000	11.000	8.000	10.000	10.000	6.000	6.000
8	19.000	22.000	24.000	28.000	28.000	22.000	27.000	33.000	18.000
9	18.000	15.000	21.000	14.000	19.000	18.000	16.000	31.000	30.000

```
---------------------------------------------------- Precision matrix ------------------------------------
---------------
```

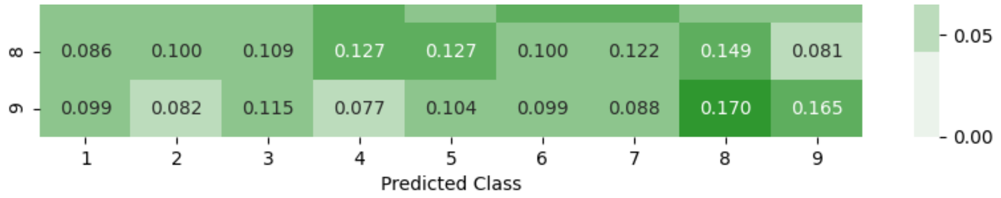


	1	2	3	4	5	6	7	8	9
1	0.102	0.160	0.119	0.129	0.153	0.168	0.137	0.189	0.114
2	0.257	0.213	0.256	0.219	0.251	0.196	0.237	0.176	0.250
3	0.306	0.316	0.288	0.304	0.197	0.276	0.247	0.225	0.277
4	0.058	0.053	0.032	0.049	0.020	0.047	0.050	0.044	0.041
5	0.005	0.000	0.005	0.000	0.005	0.009	0.009	0.000	0.005
6	0.058	0.058	0.068	0.062	0.103	0.070	0.078	0.057	0.068
7	0.034	0.036	0.027	0.049	0.039	0.047	0.046	0.026	0.027
8	0.092	0.098	0.110	0.125	0.138	0.103	0.123	0.145	0.082
9	0.087	0.067	0.096	0.062	0.094	0.084	0.073	0.137	0.136

```
Sum of columns in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
---------------------------------------------------- Recall matrix ---------------------------------------
------------
```



1	0.076	0.130	0.094	0.105	0.112	0.130	0.108	0.155	0.090
2	0.119	0.108	0.126	0.110	0.114	0.094	0.117	0.090	0.123
3	0.119	0.134	0.119	0.128	0.075	0.111	0.102	0.096	0.115
4	0.140	0.140	0.081	0.128	0.047	0.116	0.128	0.116	0.105
5	0.125	0.000	0.125	0.000	0.125	0.250	0.250	0.000	0.125
6	0.089	0.096	0.111	0.104	0.156	0.111	0.126	0.096	0.111
7	0.097	0.111	0.083	0.153	0.111	0.139	0.139	0.083	0.083



```
Sum of rows in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

In [14]:

```
summary.append(["Random Model",loss,misclassif])
```

K Nearest Neighbour Classification

In [14]:

```
hyp = {'n_neighbors':np.arange(1,20,2)}
```

In [15]:

```
model = RandomizedSearchCV(KNeighborsClassifier(),scoring='neg_log_loss',param_distributions=hyp,verbos
e=2,n_jobs=-2)
model.fit(X_train,y_tr)
```

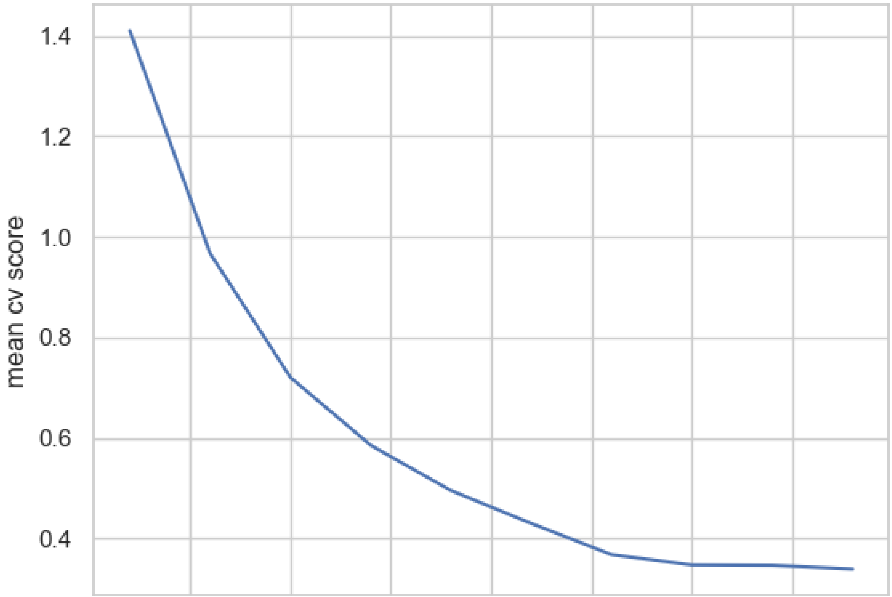
```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
```

Out[15]:

```
RandomizedSearchCV(estimator=KNeighborsClassifier(), n_jobs=-2,
                   param_distributions={'n_neighbors': array([ 1,  3,  5,  7,  9, 11, 13, 15, 17, 19])}
,
                   scoring='neg_log_loss', verbose=2)
```

In [18]:

```
sns.set(style="whitegrid")
plt.plot(model.cv_results_['param_n_neighbors'].compressed(),-1*model.cv_results_['mean_test_score'])
plt.xlabel("K Neighbours")
plt.ylabel("mean cv score")
plt.show()
```



In [16]:

```
calib = CalibratedClassifierCV(model.best_estimator_, method="sigmoid",n_jobs=-2)
calib.fit(X_train,y_tr)
```

Out[16]:

```
CalibratedClassifierCV(base_estimator=KNeighborsClassifier(n_neighbors=19),
                       n_jobs=-2)
```

In [15]:

```
calib = pk.load(open("knn_final.pk","rb"))
```

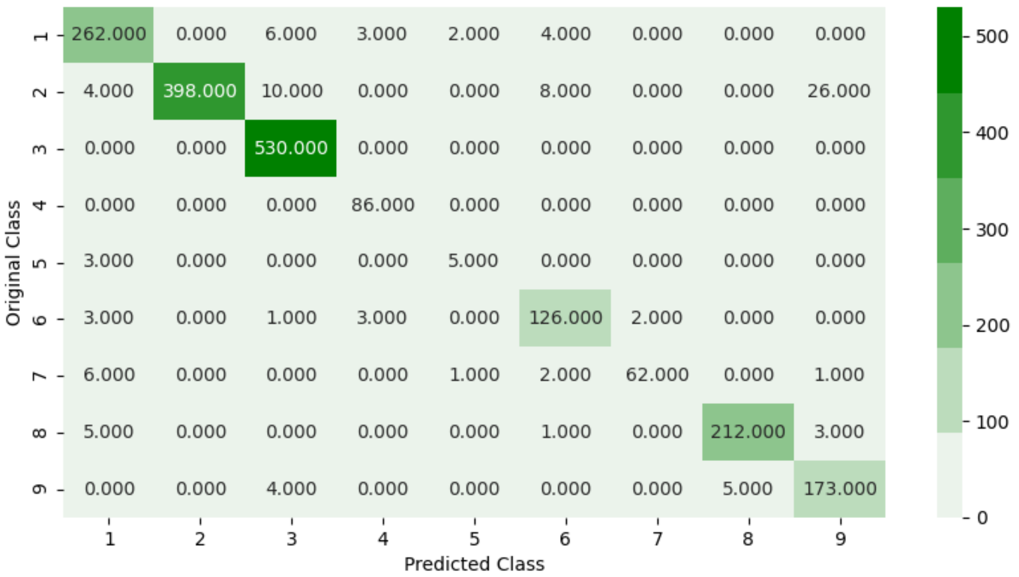
In [32]:

```
# pk.dump(models['knn'],open('knn_final.pk','wb'))
```

In [20]:

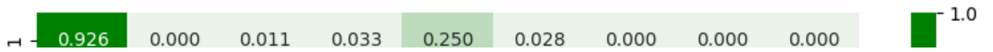
```
y_pred = calib.predict_proba(X_test)
loss = log_loss(y_tes,y_pred,eps=1e-15)
print("Log loss on Test Data using Random Model",loss)

predicted_y =np.argmax(y_pred, axis=1)
misclassif = plot_confusion_matrix(y_tes, predicted_y+1)
```

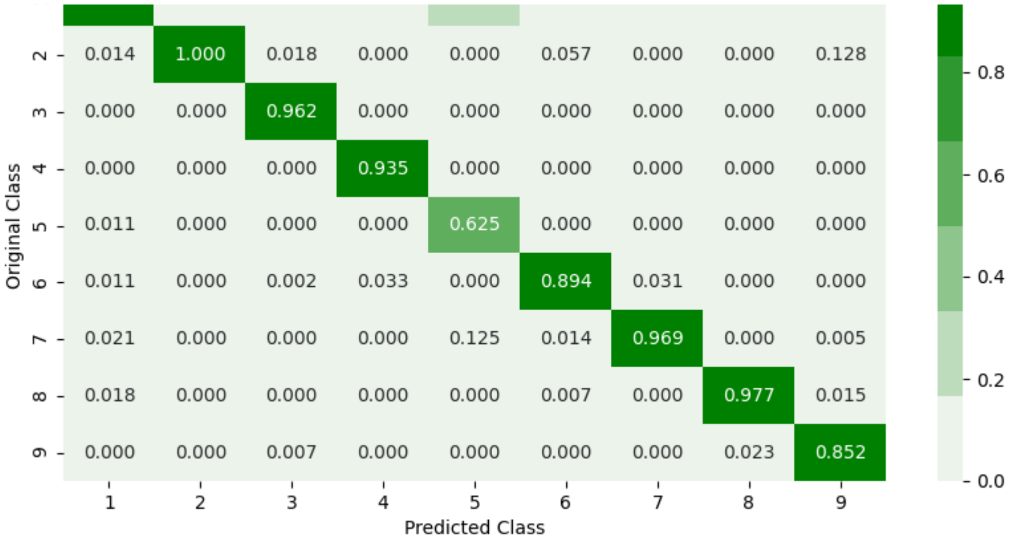
```
Log loss on Test Data using Random Model 0.2119466582782148
Number of misclassified points  5.263157894736842
-------------------------------------------------- Confusion matrix -----------------------------------
---------------
```



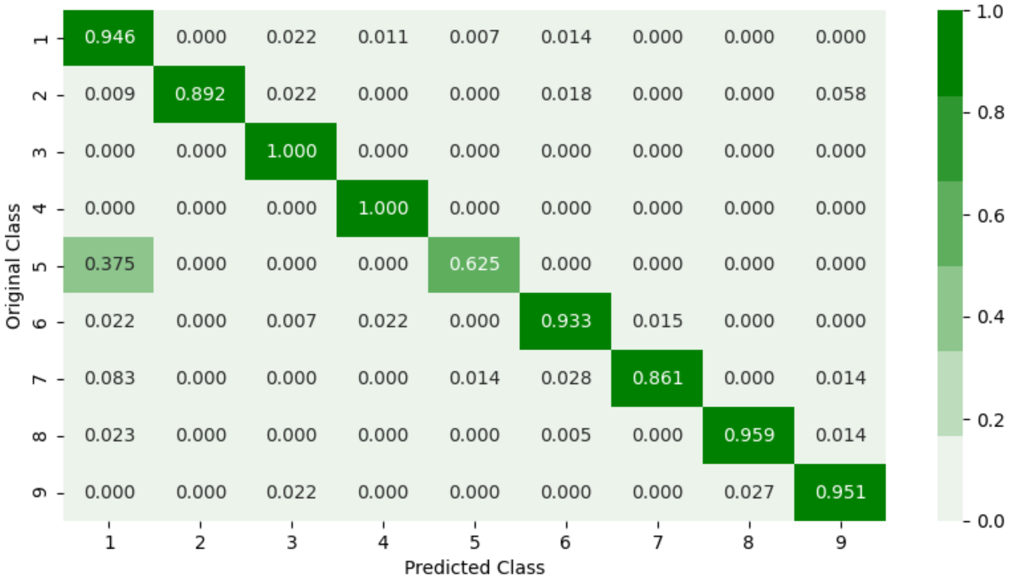
	1	2	3	4	5	6	7	8	9
1	262.000	0.000	6.000	3.000	2.000	4.000	0.000	0.000	0.000
2	4.000	398.000	10.000	0.000	0.000	8.000	0.000	0.000	26.000
3	0.000	0.000	530.000	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.000	0.000	86.000	0.000	0.000	0.000	0.000	0.000
5	3.000	0.000	0.000	0.000	5.000	0.000	0.000	0.000	0.000
6	3.000	0.000	1.000	3.000	0.000	126.000	2.000	0.000	0.000
7	6.000	0.000	0.000	0.000	1.000	2.000	62.000	0.000	1.000
8	5.000	0.000	0.000	0.000	0.000	1.000	0.000	212.000	3.000
9	0.000	0.000	4.000	0.000	0.000	0.000	0.000	5.000	173.000

```
-------------------------------------------------- Precision matrix -----------------------------------
---------------
```





```
Sum of columns in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
-------------------------------------------------- Recall matrix --------------------------------------
------------
```



```
Sum of rows in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

In [17]:

```
summary.append(["KNN",loss,misclassif])
```

In [18]:

```
models['knn'] = calib
```

In [19]:

```
calib=model=None
gc.collect()
```

Out[19]:

```
28474
```

Logistic Regression

In [43]:

```
hyp = {'alpha':np.logspace(-3,3,7)}
```

In [22]:

```
model = GridSearchCV(SGDClassifier(loss='log',penalty='elasticnet',random_state=13,verbose=1),param_gri
d=hyp,verbose=2,n_jobs=-2)
model.fit(X_train,y_tr)
```

```
Fitting 5 folds for each of 7 candidates, totalling 35 fits
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
```

```
-- Epoch 1
Norm: 33.89, NNZs: 3827, Bias: -117.575981, T: 8911, Avg. loss: 13.431731
Total training time: 2.58 seconds.
-- Epoch 2
Norm: 21.67, NNZs: 2490, Bias: -116.776222, T: 17822, Avg. loss: 0.255803
Total training time: 4.93 seconds.
-- Epoch 3
Norm: 17.78, NNZs: 2222, Bias: -116.283684, T: 26733, Avg. loss: 0.192753
Total training time: 7.03 seconds.
-- Epoch 4
Norm: 16.13, NNZs: 2098, Bias: -115.911022, T: 35644, Avg. loss: 0.168258
Total training time: 9.09 seconds.
-- Epoch 5
Norm: 15.10, NNZs: 1982, Bias: -115.640907, T: 44555, Avg. loss: 0.153931
Total training time: 11.07 seconds.
-- Epoch 6
Norm: 14.44, NNZs: 1892, Bias: -115.420029, T: 53466, Avg. loss: 0.138316
Total training time: 13.05 seconds.
-- Epoch 7
Norm: 14.03, NNZs: 1826, Bias: -115.231639, T: 62377, Avg. loss: 0.134567
Total training time: 14.99 seconds.
-- Epoch 8
Norm: 13.71, NNZs: 1761, Bias: -115.072736, T: 71288, Avg. loss: 0.128451
Total training time: 16.94 seconds.
-- Epoch 9
Norm: 13.47, NNZs: 1705, Bias: -114.932456, T: 80199, Avg. loss: 0.123753
Total training time: 18.86 seconds.
-- Epoch 10
Norm: 13.30, NNZs: 1663, Bias: -114.805387, T: 89110, Avg. loss: 0.119666
Total training time: 20.63 seconds.
-- Epoch 11
Norm: 13.13, NNZs: 1625, Bias: -114.694561, T: 98021, Avg. loss: 0.118048
Total training time: 22.51 seconds.
-- Epoch 12
Norm: 13.01, NNZs: 1596, Bias: -114.591963, T: 106932, Avg. loss: 0.117150
Total training time: 24.38 seconds.
-- Epoch 13
Norm: 12.95, NNZs: 1589, Bias: -114.493246, T: 115843, Avg. loss: 0.115060
Total training time: 26.22 seconds.
-- Epoch 14
Norm: 12.86, NNZs: 1557, Bias: -114.407129, T: 124754, Avg. loss: 0.113898
Total training time: 28.07 seconds.
-- Epoch 15
Norm: 12.79, NNZs: 1534, Bias: -114.325312, T: 133665, Avg. loss: 0.112324
Total training time: 29.87 seconds.
-- Epoch 16
Norm: 12.73, NNZs: 1525, Bias: -114.249620, T: 142576, Avg. loss: 0.110100
Total training time: 31.67 seconds.
-- Epoch 17
Norm: 12.69, NNZs: 1516, Bias: -114.176776, T: 151487, Avg. loss: 0.107928
Total training time: 33.50 seconds.
-- Epoch 18
Norm: 12.65, NNZs: 1501, Bias: -114.109183, T: 160398, Avg. loss: 0.107860
Total training time: 35.05 seconds.
-- Epoch 19
Norm: 12.59, NNZs: 1487, Bias: -114.048050, T: 169309, Avg. loss: 0.106743
Total training time: 36.85 seconds.
```

```
-- Epoch 20
Norm: 12.58, NNZs: 1489, Bias: -113.985028, T: 178220, Avg. loss: 0.106494
Total training time: 38.71 seconds.
-- Epoch 21
Norm: 12.53, NNZs: 1481, Bias: -113.929190, T: 187131, Avg. loss: 0.104748
Total training time: 40.54 seconds.
-- Epoch 22
Norm: 12.50, NNZs: 1476, Bias: -113.874558, T: 196042, Avg. loss: 0.105609
Total training time: 41.46 seconds.
-- Epoch 23
Norm: 12.49, NNZs: 1470, Bias: -113.821477, T: 204953, Avg. loss: 0.105772
Total training time: 42.14 seconds.
-- Epoch 24
Norm: 12.46, NNZs: 1463, Bias: -113.772379, T: 213864, Avg. loss: 0.104368
Total training time: 42.81 seconds.
-- Epoch 25
Norm: 12.45, NNZs: 1459, Bias: -113.723497, T: 222775, Avg. loss: 0.103932
Total training time: 43.59 seconds.
-- Epoch 26
Norm: 12.40, NNZs: 1452, Bias: -113.680806, T: 231686, Avg. loss: 0.100981
Total training time: 44.32 seconds.
-- Epoch 27
Norm: 12.39, NNZs: 1453, Bias: -113.635386, T: 240597, Avg. loss: 0.103719
Total training time: 45.04 seconds.
-- Epoch 28
Norm: 12.39, NNZs: 1454, Bias: -113.591613, T: 249508, Avg. loss: 0.103020
Total training time: 45.76 seconds.
-- Epoch 29
Norm: 12.36, NNZs: 1445, Bias: -113.551536, T: 258419, Avg. loss: 0.102633
Total training time: 46.51 seconds.
-- Epoch 30
Norm: 12.35, NNZs: 1442, Bias: -113.512150, T: 267330, Avg. loss: 0.102498
Total training time: 47.26 seconds.
-- Epoch 31
Norm: 12.32, NNZs: 1434, Bias: -113.475696, T: 276241, Avg. loss: 0.100181
Total training time: 47.99 seconds.
Convergence after 31 epochs took 47.99 seconds
-- Epoch 1
Norm: 14.52, NNZs: 766, Bias: -26.254587, T: 8911, Avg. loss: 1.178669
Total training time: 0.75 seconds.
-- Epoch 2
Norm: 8.99, NNZs: 415, Bias: -26.173181, T: 17822, Avg. loss: 0.015275
Total training time: 1.34 seconds.
-- Epoch 3
Norm: 6.72, NNZs: 297, Bias: -26.096811, T: 26733, Avg. loss: 0.012155
Total training time: 1.88 seconds.
-- Epoch 4
Norm: 5.64, NNZs: 259, Bias: -26.032030, T: 35644, Avg. loss: 0.015656
Total training time: 2.44 seconds.
-- Epoch 5
Norm: 4.88, NNZs: 231, Bias: -25.988634, T: 44555, Avg. loss: 0.008561
Total training time: 3.00 seconds.
-- Epoch 6
Norm: 4.44, NNZs: 222, Bias: -25.948851, T: 53466, Avg. loss: 0.010626
Total training time: 3.57 seconds.
-- Epoch 7
Norm: 4.09, NNZs: 213, Bias: -25.917811, T: 62377, Avg. loss: 0.008975
Total training time: 4.13 seconds.
-- Epoch 8
Norm: 3.83, NNZs: 207, Bias: -25.891604, T: 71288, Avg. loss: 0.008803
Total training time: 4.69 seconds.
-- Epoch 9
Norm: 3.66, NNZs: 201, Bias: -25.866205, T: 80199, Avg. loss: 0.009941
Total training time: 5.26 seconds.
-- Epoch 10
Norm: 3.49, NNZs: 192, Bias: -25.846041, T: 89110, Avg. loss: 0.008790
Total training time: 5.82 seconds.
Convergence after 10 epochs took 5.82 seconds
-- Epoch 1
Norm: 28.78, NNZs: 2654, Bias: -43.404218, T: 8911, Avg. loss: 3.486925
Total training time: 0.92 seconds.
-- Epoch 2
Norm: 16.24, NNZs: 1643, Bias: -43.282178, T: 17822, Avg. loss: 0.011054
Total training time: 1.75 seconds.
-- Epoch 3
Norm: 11.87, NNZs: 1231, Bias: -43.178532, T: 26733, Avg. loss: 0.013554
Total training time: 2.45 seconds.
```

```
-- Epoch 4
Norm: 9.65, NNZs: 1089, Bias: -43.093910, T: 35644, Avg. loss: 0.010186
Total training time: 3.10 seconds.
-- Epoch 5
Norm: 8.32, NNZs: 1004, Bias: -43.027758, T: 44555, Avg. loss: 0.008520
Total training time: 3.71 seconds.
-- Epoch 6
Norm: 7.43, NNZs: 955, Bias: -42.974692, T: 53466, Avg. loss: 0.008501
Total training time: 4.35 seconds.
-- Epoch 7
Norm: 6.80, NNZs: 925, Bias: -42.929110, T: 62377, Avg. loss: 0.008030
Total training time: 5.02 seconds.
-- Epoch 8
Norm: 6.34, NNZs: 917, Bias: -42.889019, T: 71288, Avg. loss: 0.007446
Total training time: 5.66 seconds.
-- Epoch 9
Norm: 5.98, NNZs: 904, Bias: -42.855703, T: 80199, Avg. loss: 0.007098
Total training time: 6.25 seconds.
-- Epoch 10
Norm: 5.71, NNZs: 889, Bias: -42.824197, T: 89110, Avg. loss: 0.007856
Total training time: 6.85 seconds.
Convergence after 10 epochs took 6.85 seconds
-- Epoch 1
Norm: 38.67, NNZs: 4458, Bias: -163.004852, T: 8911, Avg. loss: 23.722585
Total training time: 1.00 seconds.
-- Epoch 2
Norm: 24.09, NNZs: 2749, Bias: -162.335087, T: 17822, Avg. loss: 0.231508
Total training time: 1.93 seconds.
-- Epoch 3
Norm: 19.68, NNZs: 2534, Bias: -161.890729, T: 26733, Avg. loss: 0.199581
Total training time: 2.72 seconds.
-- Epoch 4
Norm: 17.67, NNZs: 2453, Bias: -161.574412, T: 35644, Avg. loss: 0.172013
Total training time: 3.49 seconds.
-- Epoch 5
Norm: 16.51, NNZs: 2391, Bias: -161.337328, T: 44555, Avg. loss: 0.157709
Total training time: 4.25 seconds.
-- Epoch 6
Norm: 15.85, NNZs: 2343, Bias: -161.139087, T: 53466, Avg. loss: 0.152734
Total training time: 5.00 seconds.
-- Epoch 7
Norm: 15.34, NNZs: 2290, Bias: -160.979012, T: 62377, Avg. loss: 0.143674
Total training time: 5.75 seconds.
-- Epoch 8
Norm: 15.00, NNZs: 2246, Bias: -160.838928, T: 71288, Avg. loss: 0.136734
Total training time: 6.59 seconds.
-- Epoch 9
Norm: 14.78, NNZs: 2221, Bias: -160.713157, T: 80199, Avg. loss: 0.137513
Total training time: 7.42 seconds.
-- Epoch 10
Norm: 14.56, NNZs: 2173, Bias: -160.605988, T: 89110, Avg. loss: 0.129928
Total training time: 8.23 seconds.
-- Epoch 11
Norm: 14.37, NNZs: 2138, Bias: -160.510129, T: 98021, Avg. loss: 0.128631
Total training time: 8.98 seconds.
-- Epoch 12
Norm: 14.29, NNZs: 2126, Bias: -160.416243, T: 106932, Avg. loss: 0.129816
Total training time: 9.72 seconds.
-- Epoch 13
Norm: 14.18, NNZs: 2092, Bias: -160.334519, T: 115843, Avg. loss: 0.125349
Total training time: 10.46 seconds.
-- Epoch 14
Norm: 14.12, NNZs: 2066, Bias: -160.256350, T: 124754, Avg. loss: 0.126086
Total training time: 11.19 seconds.
-- Epoch 15
Norm: 14.01, NNZs: 2023, Bias: -160.188653, T: 133665, Avg. loss: 0.121847
Total training time: 11.92 seconds.
-- Epoch 16
Norm: 13.97, NNZs: 2015, Bias: -160.120464, T: 142576, Avg. loss: 0.124489
Total training time: 12.66 seconds.
-- Epoch 17
Norm: 13.88, NNZs: 1990, Bias: -160.061732, T: 151487, Avg. loss: 0.119340
Total training time: 13.40 seconds.
-- Epoch 18
Norm: 13.86, NNZs: 1984, Bias: -160.000580, T: 160398, Avg. loss: 0.123461
Total training time: 14.14 seconds.
-- Epoch 19
```

```
Norm: 13.80, NNZs: 1966, Bias: -159.946942, T: 169309, Avg. loss: 0.118952
Total training time: 14.87 seconds.
-- Epoch 20
Norm: 13.77, NNZs: 1959, Bias: -159.894692, T: 178220, Avg. loss: 0.119688
Total training time: 15.58 seconds.
-- Epoch 21
Norm: 13.74, NNZs: 1950, Bias: -159.844762, T: 187131, Avg. loss: 0.119436
Total training time: 16.29 seconds.
-- Epoch 22
Norm: 13.70, NNZs: 1942, Bias: -159.798120, T: 196042, Avg. loss: 0.118036
Total training time: 17.01 seconds.
Convergence after 22 epochs took 17.01 seconds
-- Epoch 1
Norm: 36.99, NNZs: 4454, Bias: -183.523916, T: 8911, Avg. loss: 26.612921
Total training time: 1.00 seconds.
-- Epoch 2
Norm: 21.21, NNZs: 2441, Bias: -183.253822, T: 17822, Avg. loss: 0.369250
Total training time: 1.88 seconds.
-- Epoch 3
Norm: 15.50, NNZs: 1796, Bias: -183.097686, T: 26733, Avg. loss: 0.467683
Total training time: 2.69 seconds.
-- Epoch 4
Norm: 12.46, NNZs: 1456, Bias: -182.991042, T: 35644, Avg. loss: 0.533658
Total training time: 3.44 seconds.
-- Epoch 5
Norm: 10.59, NNZs: 1227, Bias: -182.905103, T: 44555, Avg. loss: 0.546573
Total training time: 4.11 seconds.
-- Epoch 6
Norm: 9.31, NNZs: 1055, Bias: -182.835736, T: 53466, Avg. loss: 0.564070
Total training time: 4.74 seconds.
-- Epoch 7
Norm: 8.38, NNZs: 940, Bias: -182.776466, T: 62377, Avg. loss: 0.570424
Total training time: 5.39 seconds.
Convergence after 7 epochs took 5.39 seconds
-- Epoch 1
Norm: 40.47, NNZs: 4587, Bias: -161.811025, T: 8911, Avg. loss: 23.694178
Total training time: 0.98 seconds.
-- Epoch 2
Norm: 26.51, NNZs: 2961, Bias: -160.754841, T: 17822, Avg. loss: 0.158389
Total training time: 1.94 seconds.
-- Epoch 3
Norm: 22.27, NNZs: 2484, Bias: -160.120511, T: 26733, Avg. loss: 0.133905
Total training time: 2.97 seconds.
-- Epoch 4
Norm: 20.35, NNZs: 2203, Bias: -159.670968, T: 35644, Avg. loss: 0.120901
Total training time: 3.85 seconds.
-- Epoch 5
Norm: 19.34, NNZs: 2041, Bias: -159.318622, T: 44555, Avg. loss: 0.118182
Total training time: 4.72 seconds.
-- Epoch 6
Norm: 18.69, NNZs: 1927, Bias: -159.037456, T: 53466, Avg. loss: 0.108173
Total training time: 5.56 seconds.
-- Epoch 7
Norm: 18.26, NNZs: 1826, Bias: -158.799794, T: 62377, Avg. loss: 0.106783
Total training time: 6.50 seconds.
-- Epoch 8
Norm: 17.93, NNZs: 1746, Bias: -158.598573, T: 71288, Avg. loss: 0.101489
Total training time: 7.28 seconds.
-- Epoch 9
Norm: 17.66, NNZs: 1669, Bias: -158.423282, T: 80199, Avg. loss: 0.096868
Total training time: 8.01 seconds.
-- Epoch 10
Norm: 17.50, NNZs: 1606, Bias: -158.262171, T: 89110, Avg. loss: 0.097123
Total training time: 8.74 seconds.
-- Epoch 11
Norm: 17.36, NNZs: 1564, Bias: -158.117783, T: 98021, Avg. loss: 0.095903
Total training time: 9.46 seconds.
-- Epoch 12
Norm: 17.24, NNZs: 1516, Bias: -157.987932, T: 106932, Avg. loss: 0.093887
Total training time: 10.17 seconds.
-- Epoch 13
Norm: 17.13, NNZs: 1483, Bias: -157.869565, T: 115843, Avg. loss: 0.092764
Total training time: 10.86 seconds.
-- Epoch 14
Norm: 17.03, NNZs: 1442, Bias: -157.760623, T: 124754, Avg. loss: 0.090583
Total training time: 11.54 seconds.
-- Epoch 15
-- Epoch 15
Norm: 16.92, NNZs: 1410, Bias: -157.662645, T: 133665, Avg. loss: 0.087804
Total training time: 12.25 seconds.
-- Epoch 16
Norm: 16.88, NNZs: 1393, Bias: -157.564285, T: 142576, Avg. loss: 0.090599
Total training time: 12.97 seconds.
-- Epoch 17
Norm: 16.83, NNZs: 1367, Bias: -157.473618, T: 151487, Avg. loss: 0.089141
Total training time: 13.65 seconds.
-- Epoch 18
Norm: 16.76, NNZs: 1345, Bias: -157.391721, T: 160398, Avg. loss: 0.086839
Total training time: 14.33 seconds.
-- Epoch 19
Norm: 16.71, NNZs: 1323, Bias: -157.312974, T: 169309, Avg. loss: 0.086223
Total training time: 15.01 seconds.
-- Epoch 20
Norm: 16.68, NNZs: 1312, Bias: -157.237042, T: 178220, Avg. loss: 0.087317
Total training time: 15.69 seconds.
Convergence after 20 epochs took 15.69 seconds
-- Epoch 1
Norm: 34.76, NNZs: 4143, Bias: -157.337608, T: 8911, Avg. loss: 22.410143
Total training time: 0.98 seconds.
-- Epoch 2
Norm: 21.62, NNZs: 2539, Bias: -156.838121, T: 17822, Avg. loss: 0.274377
Total training time: 1.85 seconds.
-- Epoch 3
Norm: 17.30, NNZs: 2340, Bias: -156.509421, T: 26733, Avg. loss: 0.242113
Total training time: 2.63 seconds.
-- Epoch 4
Norm: 15.25, NNZs: 2219, Bias: -156.280746, T: 35644, Avg. loss: 0.239164
Total training time: 3.50 seconds.
-- Epoch 5
Norm: 14.08, NNZs: 2134, Bias: -156.107774, T: 44555, Avg. loss: 0.235055
Total training time: 4.34 seconds.
-- Epoch 6
Norm: 13.30, NNZs: 2047, Bias: -155.967485, T: 53466, Avg. loss: 0.234118
Total training time: 5.17 seconds.
-- Epoch 7
Norm: 12.84, NNZs: 2000, Bias: -155.844861, T: 62377, Avg. loss: 0.228950
Total training time: 5.92 seconds.
-- Epoch 8
Norm: 12.47, NNZs: 1959, Bias: -155.743955, T: 71288, Avg. loss: 0.228690
Total training time: 6.63 seconds.
-- Epoch 9
Norm: 12.16, NNZs: 1914, Bias: -155.656045, T: 80199, Avg. loss: 0.226809
Total training time: 7.42 seconds.
-- Epoch 10
Norm: 11.96, NNZs: 1882, Bias: -155.574433, T: 89110, Avg. loss: 0.226135
Total training time: 8.23 seconds.
-- Epoch 11
Norm: 11.79, NNZs: 1849, Bias: -155.501640, T: 98021, Avg. loss: 0.226254
Total training time: 8.96 seconds.
-- Epoch 12
Norm: 11.64, NNZs: 1828, Bias: -155.436499, T: 106932, Avg. loss: 0.223713
Total training time: 9.89 seconds.
-- Epoch 13
Norm: 11.54, NNZs: 1797, Bias: -155.374988, T: 115843, Avg. loss: 0.224297
Total training time: 10.73 seconds.
-- Epoch 14
Norm: 11.43, NNZs: 1780, Bias: -155.320115, T: 124754, Avg. loss: 0.221221
Total training time: 11.47 seconds.
-- Epoch 15
Norm: 11.36, NNZs: 1765, Bias: -155.267557, T: 133665, Avg. loss: 0.222031
Total training time: 12.21 seconds.
-- Epoch 16
Norm: 11.28, NNZs: 1759, Bias: -155.219999, T: 142576, Avg. loss: 0.221291
Total training time: 12.92 seconds.
-- Epoch 17
Norm: 11.22, NNZs: 1755, Bias: -155.174578, T: 151487, Avg. loss: 0.219881
Total training time: 13.62 seconds.
-- Epoch 18
Norm: 11.15, NNZs: 1745, Bias: -155.132734, T: 160398, Avg. loss: 0.220983
Total training time: 14.40 seconds.
-- Epoch 19
Norm: 11.11, NNZs: 1738, Bias: -155.092002, T: 169309, Avg. loss: 0.220089
Total training time: 15.13 seconds.
-- Epoch 20
Norm: 11.06, NNZs: 1723, Bias: -155.054679, T: 178220, Avg. loss: 0.218644
```

```
Norm: 11.06, NNZs: 1723, Bias: -155.051079, T: 178220, Avg. loss: 0.218644
Total training time: 15.89 seconds.
-- Epoch 21
Norm: 11.02, NNZs: 1719, Bias: -155.018767, T: 187131, Avg. loss: 0.219088
Total training time: 16.65 seconds.
-- Epoch 22
Norm: 10.98, NNZs: 1711, Bias: -154.984400, T: 196042, Avg. loss: 0.217912
Total training time: 17.35 seconds.
-- Epoch 23
Norm: 10.95, NNZs: 1709, Bias: -154.951229, T: 204953, Avg. loss: 0.218139
Total training time: 18.06 seconds.
-- Epoch 24
Norm: 10.90, NNZs: 1700, Bias: -154.921315, T: 213864, Avg. loss: 0.217330
Total training time: 18.76 seconds.
-- Epoch 25
Norm: 10.90, NNZs: 1703, Bias: -154.889848, T: 222775, Avg. loss: 0.217994
Total training time: 19.46 seconds.
Convergence after 25 epochs took 19.46 seconds
-- Epoch 1
Norm: 34.93, NNZs: 3939, Bias: -132.541432, T: 8911, Avg. loss: 15.589432
Total training time: 1.05 seconds.
-- Epoch 2
Norm: 21.20, NNZs: 2234, Bias: -132.059137, T: 17822, Avg. loss: 0.165663
Total training time: 2.12 seconds.
-- Epoch 3
Norm: 16.86, NNZs: 1853, Bias: -131.741573, T: 26733, Avg. loss: 0.120219
Total training time: 2.93 seconds.
-- Epoch 4
Norm: 14.85, NNZs: 1646, Bias: -131.510539, T: 35644, Avg. loss: 0.090029
Total training time: 3.73 seconds.
-- Epoch 5
Norm: 13.71, NNZs: 1502, Bias: -131.336064, T: 44555, Avg. loss: 0.074810
Total training time: 4.48 seconds.
-- Epoch 6
Norm: 13.01, NNZs: 1397, Bias: -131.193548, T: 53466, Avg. loss: 0.065255
Total training time: 5.27 seconds.
-- Epoch 7
Norm: 12.52, NNZs: 1325, Bias: -131.074919, T: 62377, Avg. loss: 0.059193
Total training time: 6.01 seconds.
-- Epoch 8
Norm: 12.20, NNZs: 1265, Bias: -130.970486, T: 71288, Avg. loss: 0.055276
Total training time: 6.70 seconds.
-- Epoch 9
Norm: 11.93, NNZs: 1199, Bias: -130.882356, T: 80199, Avg. loss: 0.050334
Total training time: 7.39 seconds.
-- Epoch 10
Norm: 11.76, NNZs: 1151, Bias: -130.799942, T: 89110, Avg. loss: 0.051388
Total training time: 8.06 seconds.
-- Epoch 11
Norm: 11.61, NNZs: 1100, Bias: -130.727679, T: 98021, Avg. loss: 0.047299
Total training time: 8.73 seconds.
-- Epoch 12
Norm: 11.50, NNZs: 1062, Bias: -130.660875, T: 106932, Avg. loss: 0.047500
Total training time: 9.36 seconds.
-- Epoch 13
Norm: 11.37, NNZs: 1028, Bias: -130.602847, T: 115843, Avg. loss: 0.045178
Total training time: 9.98 seconds.
-- Epoch 14
Norm: 11.31, NNZs: 1010, Bias: -130.544806, T: 124754, Avg. loss: 0.046384
Total training time: 10.68 seconds.
-- Epoch 15
Norm: 11.24, NNZs: 988, Bias: -130.493304, T: 133665, Avg. loss: 0.043794
Total training time: 11.34 seconds.
-- Epoch 16
Norm: 11.18, NNZs: 971, Bias: -130.444844, T: 142576, Avg. loss: 0.043518
Total training time: 12.01 seconds.
-- Epoch 17
Norm: 11.14, NNZs: 961, Bias: -130.398190, T: 151487, Avg. loss: 0.044182
Total training time: 12.63 seconds.
-- Epoch 18
Norm: 11.07, NNZs: 947, Bias: -130.357614, T: 160398, Avg. loss: 0.042779
Total training time: 13.23 seconds.
-- Epoch 19
Norm: 11.04, NNZs: 943, Bias: -130.316105, T: 169309, Avg. loss: 0.043659
Total training time: 13.85 seconds.
-- Epoch 20
Norm: 11.02, NNZs: 938, Bias: -130.276390, T: 178220, Avg. loss: 0.043134
Total training time: 14.44 seconds.
Total training time: 14.44 seconds.
Convergence after 20 epochs took 14.44 seconds
-- Epoch 1
Norm: 31.35, NNZs: 3077, Bias: -112.978968, T: 8911, Avg. loss: 13.232346
Total training time: 0.96 seconds.
-- Epoch 2
Norm: 19.58, NNZs: 2120, Bias: -112.521760, T: 17822, Avg. loss: 0.089813
Total training time: 1.78 seconds.
-- Epoch 3
Norm: 15.78, NNZs: 1694, Bias: -112.225792, T: 26733, Avg. loss: 0.045544
Total training time: 2.59 seconds.
-- Epoch 4
Norm: 13.85, NNZs: 1483, Bias: -112.021342, T: 35644, Avg. loss: 0.036733
Total training time: 3.26 seconds.
-- Epoch 5
Norm: 12.66, NNZs: 1335, Bias: -111.871046, T: 44555, Avg. loss: 0.033560
Total training time: 3.96 seconds.
-- Epoch 6
Norm: 11.97, NNZs: 1241, Bias: -111.742507, T: 53466, Avg. loss: 0.031032
Total training time: 4.60 seconds.
-- Epoch 7
Norm: 11.44, NNZs: 1137, Bias: -111.638615, T: 62377, Avg. loss: 0.029666
Total training time: 5.22 seconds.
-- Epoch 8
Norm: 11.09, NNZs: 1074, Bias: -111.546316, T: 71288, Avg. loss: 0.029702
Total training time: 5.82 seconds.
-- Epoch 9
Norm: 10.82, NNZs: 1023, Bias: -111.466278, T: 80199, Avg. loss: 0.029747
Total training time: 6.42 seconds.
-- Epoch 10
Norm: 10.61, NNZs: 991, Bias: -111.394891, T: 89110, Avg. loss: 0.028786
Total training time: 7.03 seconds.
-- Epoch 11
Norm: 10.44, NNZs: 945, Bias: -111.330771, T: 98021, Avg. loss: 0.028490
Total training time: 7.68 seconds.
-- Epoch 12
Norm: 10.31, NNZs: 924, Bias: -111.271693, T: 106932, Avg. loss: 0.028364
Total training time: 8.32 seconds.
Convergence after 12 epochs took 8.32 seconds
```

```
[Parallel(n_jobs=1)]: Done   9 out of   9 | elapsed:  2.3min finished
```

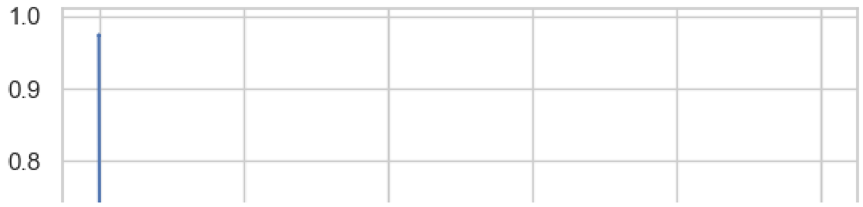
Out[22]:

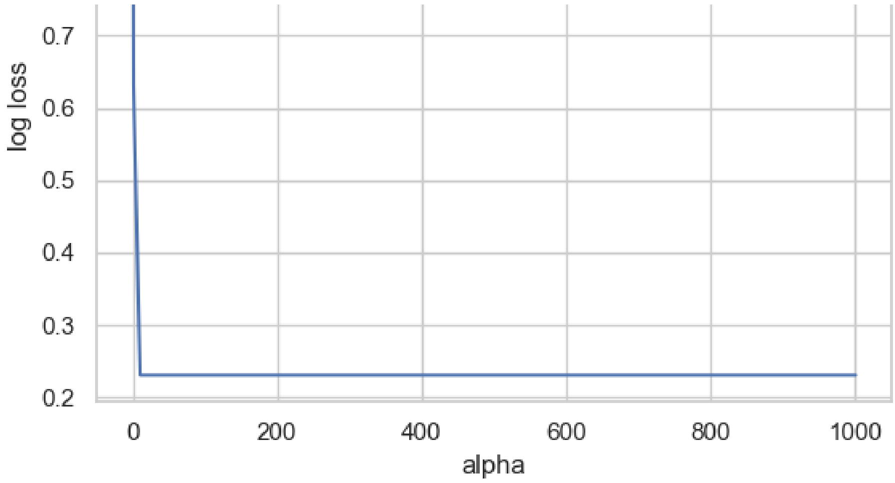
```
GridSearchCV(estimator=SGDClassifier(loss='log', penalty='elasticnet',
                                     random_state=13, verbose=1),
             n_jobs=-2,
             param_grid={'alpha': array([1.e-03, 1.e-02, 1.e-01, 1.e+00, 1.e+01, 1.e+02, 1.e+03])},
             verbose=2)
```

In [23]:

```
sns.set(style="whitegrid")
plt.plot(model.cv_results_['param_alpha'].compressed(),model.cv_results_['mean_test_score'])
plt.xlabel("alpha")
plt.ylabel("log loss")
print("Best alpha =",model.best_params_['alpha']," With Best Mean CV Score =",model.best_score_)
print("="*100)
plt.show()
```

```
Best alpha = 0.01  With Best Mean CV Score = 0.9745250850878071
====================================================================================================
```





In [24]:

```
calib = CalibratedClassifierCV(model.best_estimator_, method="sigmoid",n_jobs=-2)
calib.fit(X_train,y_tr)
```

Out[24]:

```
CalibratedClassifierCV(base_estimator=SGDClassifier(alpha=0.01, loss='log',
                                                    penalty='elasticnet',
                                                    random_state=13,
                                                    verbose=1),
                       n_jobs=-2)
```

In [30]:

```
# pk.dump(calib,open('lr_final.pk','wb'))
```

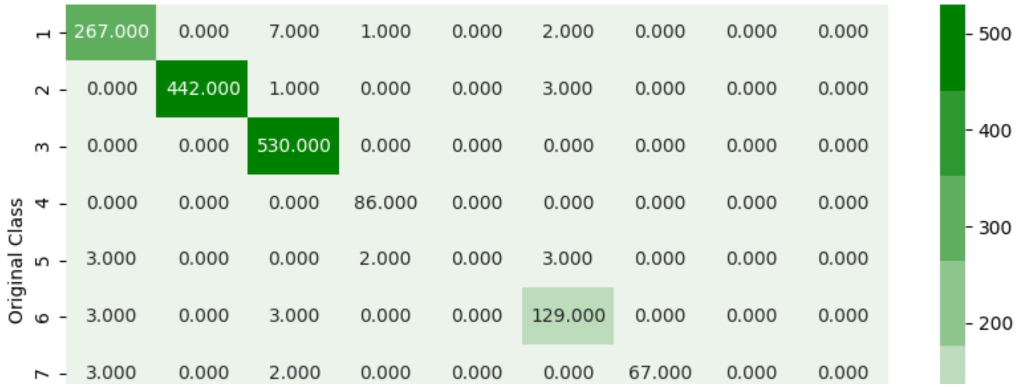
In [22]:

```
calib=pk.load(open("./lr_final.pk","rb"))
```

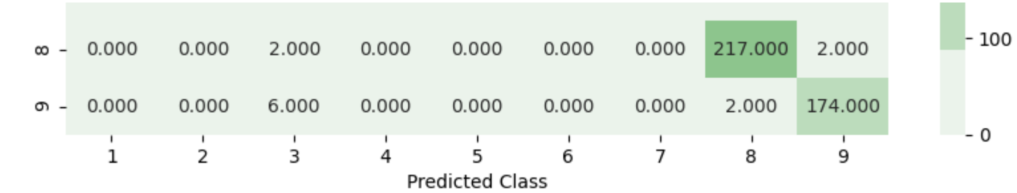
In [23]:

```
y_pred = calib.predict_proba(X_test)
loss = log_loss(y_tes,y_pred,eps=1e-15)
print("Log loss on Test Data using Random Model",loss)

predicted_y =np.argmax(y_pred, axis=1)
misclassif = plot_confusion_matrix(y_tes, predicted_y+1)
```

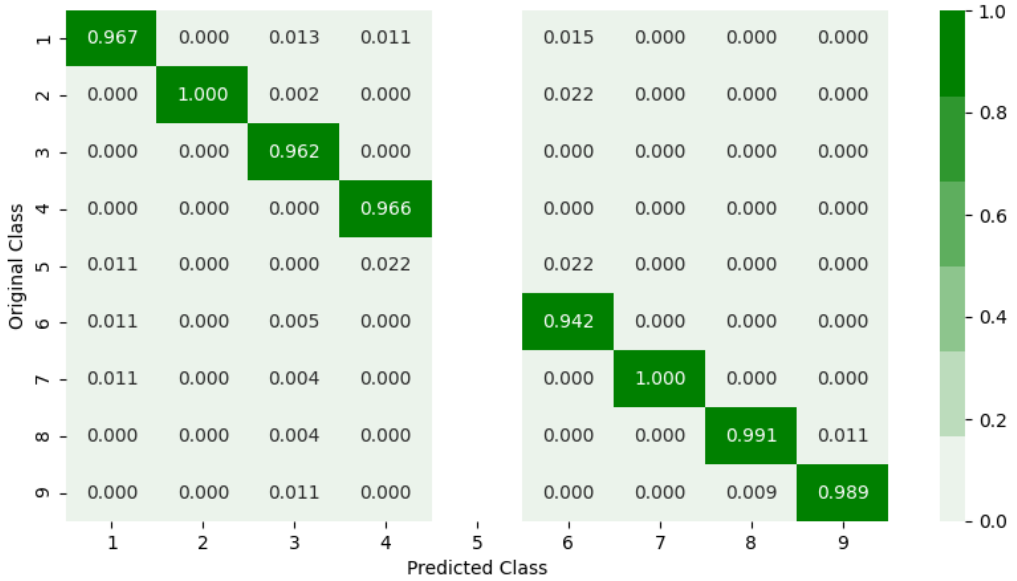
```
Log loss on Test Data using Random Model 0.1871996383580492
Number of misclassified points  2.2994379151762905
-------------------------------------------------- Confusion matrix -----------------------------------
---------------
```



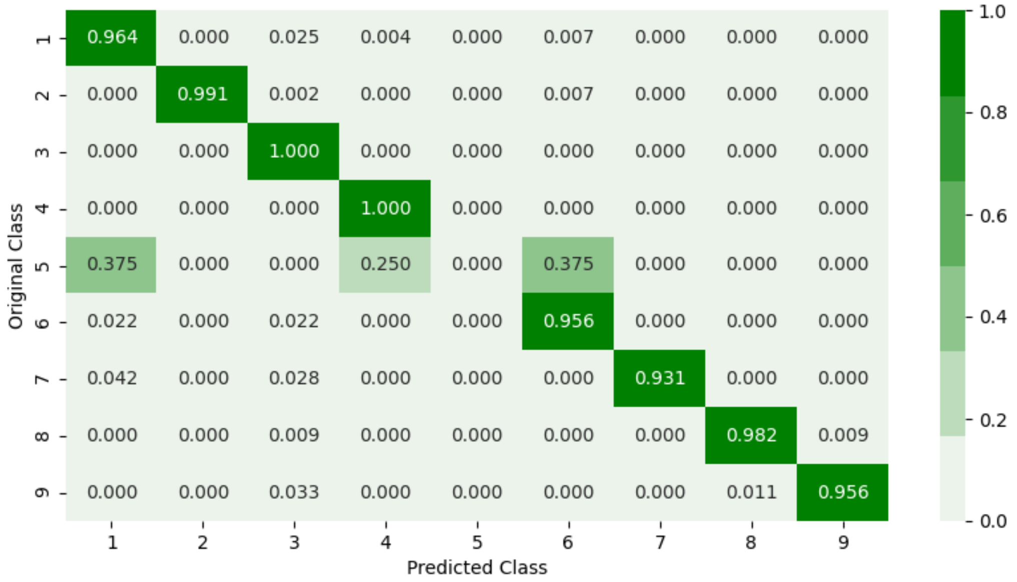
1	267.000	0.000	7.000	1.000	0.000	2.000	0.000	0.000	0.000
2	0.000	442.000	1.000	0.000	0.000	3.000	0.000	0.000	0.000
3	0.000	0.000	530.000	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.000	0.000	86.000	0.000	0.000	0.000	0.000	0.000
5	3.000	0.000	0.000	2.000	0.000	3.000	0.000	0.000	0.000
6	3.000	0.000	3.000	0.000	0.000	129.000	0.000	0.000	0.000
7	3.000	0.000	2.000	0.000	0.000	0.000	67.000	0.000	0.000



```
---------------------------------------------------- Precision matrix ------------------------------------
---------------
```



```
Sum of columns in precision matrix [ 1.   1.   1.   1. nan  1.   1.   1.   1.]
---------------------------------------------------- Recall matrix ----------------------------------------
------------
```



```
Sum of rows in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

In [24]:

```
summary.append(["Logistic Regression",loss,misclassif])
```

In [25]:

```
models['lr'] = calib
```

In [26]:

```
model=None
calib=None
gc.collect()
```

Out[26]:

```
84579
```

RBF SVM

In [29]:

```
hyp = {'C':np.logspace(-3,3,6),'gamma':np.logspace(-3,2,6)}
```

In [30]:

```
model = RandomizedSearchCV(SVC(degree=2,verbose=True,max_iter=500),param_distributions=hyp,n_iter=5,ver
bose=3)
model.fit(X_train,y_tr)
```

```
Fitting 5 folds for each of 5 candidates, totalling 25 fits
[LibSVM][CV 1/5] END ..............C=0.001, gamma=0.01;, score=0.678 total time= 4.9min
[LibSVM][CV 2/5] END ..............C=0.001, gamma=0.01;, score=0.693 total time= 4.7min
[LibSVM][CV 3/5] END ..............C=0.001, gamma=0.01;, score=0.694 total time= 4.8min
[LibSVM][CV 4/5] END ..............C=0.001, gamma=0.01;, score=0.687 total time= 4.6min
[LibSVM][CV 5/5] END ..............C=0.001, gamma=0.01;, score=0.680 total time= 4.9min
[LibSVM][CV 1/5] END C=0.25118864315095796, gamma=100.0;, score=0.094 total time= 5.6min
[LibSVM][CV 2/5] END C=0.25118864315095796, gamma=100.0;, score=0.096 total time= 6.1min
[LibSVM][CV 3/5] END C=0.25118864315095796, gamma=100.0;, score=0.097 total time= 5.2min
[LibSVM][CV 4/5] END C=0.25118864315095796, gamma=100.0;, score=0.095 total time= 5.0min
[LibSVM][CV 5/5] END C=0.25118864315095796, gamma=100.0;, score=0.094 total time= 4.9min
[LibSVM][CV 1/5] END ....C=63.0957344480193, gamma=10.0;, score=0.176 total time= 4.9min
[LibSVM][CV 2/5] END ....C=63.0957344480193, gamma=10.0;, score=0.185 total time= 4.9min
[LibSVM][CV 3/5] END ....C=63.0957344480193, gamma=10.0;, score=0.178 total time= 4.9min
[LibSVM][CV 4/5] END ....C=63.0957344480193, gamma=10.0;, score=0.174 total time= 4.9min
[LibSVM][CV 5/5] END ....C=63.0957344480193, gamma=10.0;, score=0.176 total time= 4.9min
[LibSVM][CV 1/5] END C=0.25118864315095796, gamma=0.001;, score=0.891 total time= 3.6min
[LibSVM][CV 2/5] END C=0.25118864315095796, gamma=0.001;, score=0.895 total time= 4.4min
[LibSVM][CV 3/5] END C=0.25118864315095796, gamma=0.001;, score=0.900 total time= 4.7min
[LibSVM][CV 4/5] END C=0.25118864315095796, gamma=0.001;, score=0.884 total time= 5.0min
[LibSVM][CV 5/5] END C=0.25118864315095796, gamma=0.001;, score=0.883 total time= 4.6min
[LibSVM][CV 1/5] END ...C=63.0957344480193, gamma=0.001;, score=0.945 total time= 5.8min
[LibSVM][CV 2/5] END ...C=63.0957344480193, gamma=0.001;, score=0.948 total time= 5.6min
[LibSVM][CV 3/5] END ...C=63.0957344480193, gamma=0.001;, score=0.952 total time= 5.5min
[LibSVM][CV 4/5] END ...C=63.0957344480193, gamma=0.001;, score=0.944 total time= 4.4min
[LibSVM][CV 5/5] END ...C=63.0957344480193, gamma=0.001;, score=0.934 total time= 3.3min
[LibSVM]
```

Out[30]:

```
RandomizedSearchCV(estimator=SVC(degree=2, max_iter=500, verbose=True),
                   n_iter=5,
                   param_distributions={'C': array([1.00000000e-03, 1.58489319e-02, 2.51188643e-01, 3.9
8107171e+00,
       6.30957344e+01, 1.00000000e+03]),
                                        'gamma': array([1.e-03, 1.e-02, 1.e-01, 1.e+00, 1.e+01, 1.e+02]
)},
                   verbose=3)
```

In [31]:

```
calib = CalibratedClassifierCV(model.best_estimator_, method="sigmoid",n_jobs=-2)
```

```
calib.fit(X_train,y_tr)
```

Out[31]:

```
CalibratedClassifierCV(base_estimator=SVC(C=63.0957344480193, degree=2,
                                          gamma=0.001, max_iter=500,
                                          verbose=True),
                       n_jobs=-2)
```

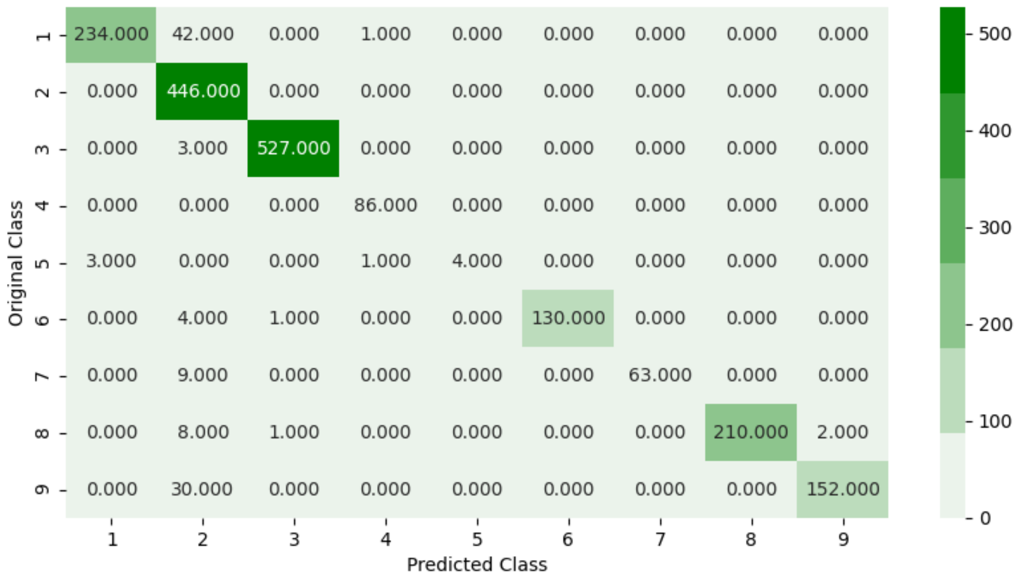
In [27]:

```
# pk.dump(calib,open("rbfsvm_final.pkl","wb"))
clf=pk.load(open("rbfsvm_final.pkl","rb"))
```

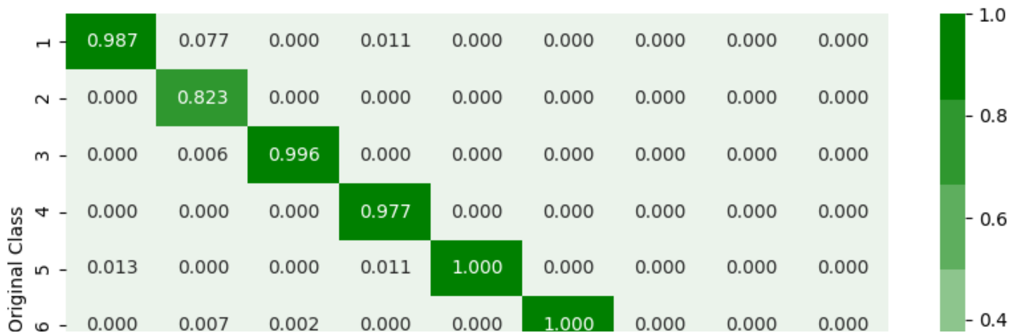
In [32]:

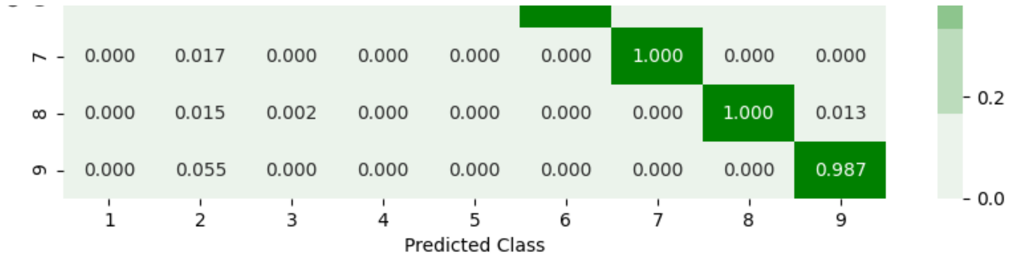
```
y_pred = clf.predict_proba(X_test)
loss = log_loss(y_tes,y_pred,eps=1e-15)
print("Log loss on Test Data using Random Model",loss)

predicted_y =np.argmax(y_pred, axis=1)
misclassif = plot_confusion_matrix(y_tes, predicted_y+1)
```

```
Log loss on Test Data using Random Model 0.3216952935264452
Number of misclassified points  5.3653551354113445
-------------------------------------------------- Confusion matrix -----------------------------------
---------------
```

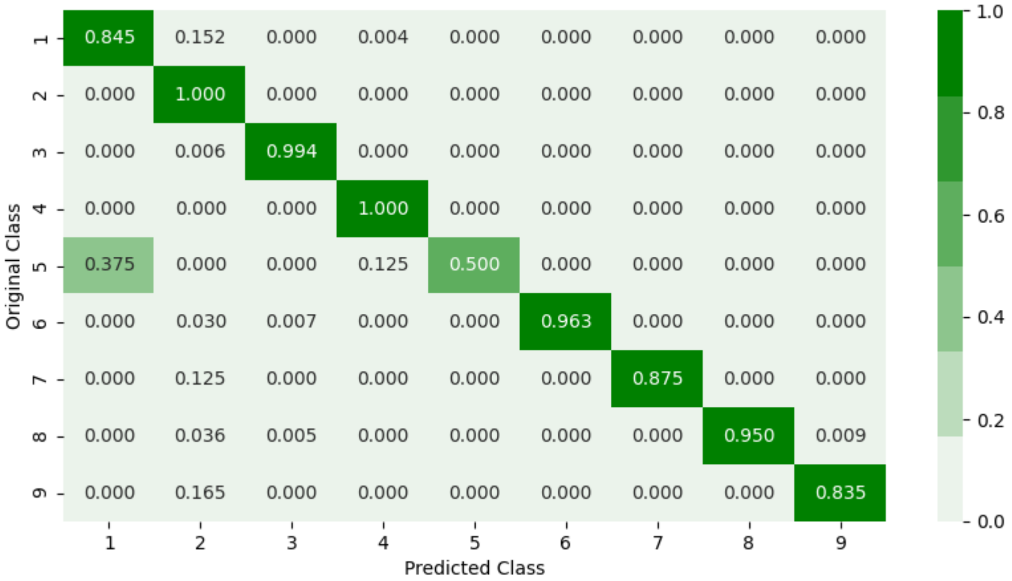


```
-------------------------------------------------- Precision matrix -----------------------------------
---------------
```





```
Sum of columns in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
-------------------------------------------------- Recall matrix --------------------------------------
------------
```



```
Sum of rows in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

In [29]:

```
summary.append(["RBF-SVM",loss,misclassif])
```

In [30]:

```
models['rbf-svm'] = clf
```

In [31]:

```
calib=None
model=None
gc.collect()
```

Out[31]:

```
20
```

Random Forest

In [63]:

```
hyp = {"n_estimators":[10,50,100,500,1000,2000,3000]}
```

In [64]:

```
model = RandomizedSearchCV(RandomForestClassifier(random_state=25,verbose=1,n_jobs=-1),param_distributi
ons=hyp,n_iter=12,cv=4,verbose=2)
model.fit(X_train,y_tr)
```

```
C:\Users\sista\Anaconda3\lib\site-packages\sklearn\model_selection\_search.py:289: UserWarning:

The total space of parameters 7 is smaller than n_iter=12. Running 7 iterations. For exhaustive searche
s, use GridSearchCV.
```

```
Fitting 4 folds for each of 7 candidates, totalling 28 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  10 out of  10 | elapsed:    6.4s finished
[Parallel(n_jobs=10)]: Using backend ThreadingBackend with 10 concurrent workers.
[Parallel(n_jobs=10)]: Done   2 out of  10 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=10)]: Done  10 out of  10 | elapsed:    0.0s finished
```

```
[CV] END ....................................n_estimators=10; total time=   8.0s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  10 out of  10 | elapsed:    1.9s finished
[Parallel(n_jobs=10)]: Using backend ThreadingBackend with 10 concurrent workers.
[Parallel(n_jobs=10)]: Done   2 out of  10 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=10)]: Done  10 out of  10 | elapsed:    0.0s finished
```

```
[CV] END ....................................n_estimators=10; total time=   3.3s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  10 out of  10 | elapsed:    0.5s finished
[Parallel(n_jobs=10)]: Using backend ThreadingBackend with 10 concurrent workers.
[Parallel(n_jobs=10)]: Done   2 out of  10 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=10)]: Done  10 out of  10 | elapsed:    0.0s finished
```

```
[CV] END ....................................n_estimators=10; total time=   2.0s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  10 out of  10 | elapsed:    0.4s finished
[Parallel(n_jobs=10)]: Using backend ThreadingBackend with 10 concurrent workers.
[Parallel(n_jobs=10)]: Done   2 out of  10 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=10)]: Done  10 out of  10 | elapsed:    0.0s finished
```

```
[CV] END ....................................n_estimators=10; total time=   2.1s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.5s
[Parallel(n_jobs=-1)]: Done  50 out of  50 | elapsed:    2.3s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
```

```
[CV] END ....................................n_estimators=50; total time=   3.7s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.5s
[Parallel(n_jobs=-1)]: Done  50 out of  50 | elapsed:    2.1s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
```

```
[CV] END ....................................n_estimators=50; total time=   3.8s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    2.9s
[Parallel(n_jobs=-1)]: Done  50 out of  50 | elapsed:    3.6s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
```

```
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
```

```
[CV] END .................................n_estimators=50; total time=   5.5s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.5s
[Parallel(n_jobs=-1)]: Done  50 out of  50 | elapsed:    2.2s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
```

```
[CV] END .................................n_estimators=50; total time=   4.3s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.5s
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:    3.6s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 100 out of 100 | elapsed:    0.0s finished
```

```
[CV] END ................................n_estimators=100; total time=   5.1s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.9s
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:    5.0s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 100 out of 100 | elapsed:    0.0s finished
```

```
[CV] END ................................n_estimators=100; total time=   6.5s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.1s
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:    4.5s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 100 out of 100 | elapsed:    0.0s finished
```

```
[CV] END ................................n_estimators=100; total time=   6.0s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.7s
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:    3.8s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 100 out of 100 | elapsed:    0.0s finished
```

```
[CV] END ................................n_estimators=100; total time=   5.5s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    7.2s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   14.9s
[Parallel(n_jobs=-1)]: Done 500 out of 500 | elapsed:   17.2s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 500 out of 500 | elapsed:    0.2s finished
```

```
[CV] END ................................n_estimators=500; total time=  19.9s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.7s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    6.3s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   14.2s
```

```
[Parallel(n_jobs=-1)]: Done 500 out of 500 | elapsed:   16.4s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 500 out of 500 | elapsed:    0.2s finished
```

```
[CV] END .................................n_estimators=500; total time=  18.2s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    2.3s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    6.8s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   14.4s
[Parallel(n_jobs=-1)]: Done 500 out of 500 | elapsed:   16.7s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 500 out of 500 | elapsed:    0.1s finished
```

```
[CV] END .................................n_estimators=500; total time=  18.4s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.6s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    6.1s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   14.1s
[Parallel(n_jobs=-1)]: Done 500 out of 500 | elapsed:   16.3s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 500 out of 500 | elapsed:    0.1s finished
```

```
[CV] END .................................n_estimators=500; total time=  18.1s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.7s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    6.4s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   14.3s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   25.4s
[Parallel(n_jobs=-1)]: Done 1000 out of 1000 | elapsed:   32.5s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.3s
[Parallel(n_jobs=12)]: Done 1000 out of 1000 | elapsed:    0.3s finished
```

```
[CV] END ................................n_estimators=1000; total time=  34.8s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.7s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    6.4s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   14.6s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   25.6s
[Parallel(n_jobs=-1)]: Done 1000 out of 1000 | elapsed:   32.7s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.2s
[Parallel(n_jobs=12)]: Done 1000 out of 1000 | elapsed:    0.3s finished
```

```
[CV] END ................................n_estimators=1000; total time=  34.8s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.5s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    6.2s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   15.6s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   27.9s
```

```
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   27.9s
[Parallel(n_jobs=-1)]: Done 1000 out of 1000 | elapsed:   35.6s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.2s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.3s
[Parallel(n_jobs=12)]: Done 1000 out of 1000 | elapsed:    0.4s finished
```

```
[CV] END .................................n_estimators=1000; total time=  37.8s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.7s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    7.0s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   15.3s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   27.9s
[Parallel(n_jobs=-1)]: Done 1000 out of 1000 | elapsed:   35.5s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.3s
[Parallel(n_jobs=12)]: Done 1000 out of 1000 | elapsed:    0.4s finished
```

```
[CV] END .................................n_estimators=1000; total time=  37.7s
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.9s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    7.0s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   15.2s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   28.4s
[Parallel(n_jobs=-1)]: Done 1226 tasks      | elapsed:   43.7s
[Parallel(n_jobs=-1)]: Done 1776 tasks      | elapsed:  1.0min
[Parallel(n_jobs=-1)]: Done 2000 out of 2000 | elapsed:  1.2min finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.3s
[Parallel(n_jobs=12)]: Done 1226 tasks      | elapsed:    0.6s
[Parallel(n_jobs=12)]: Done 1776 tasks      | elapsed:    0.8s
[Parallel(n_jobs=12)]: Done 2000 out of 2000 | elapsed:    0.9s finished
```

```
[CV] END .................................n_estimators=2000; total time= 1.2min
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.7s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    7.1s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   15.7s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   27.6s
[Parallel(n_jobs=-1)]: Done 1226 tasks      | elapsed:   42.8s
[Parallel(n_jobs=-1)]: Done 1776 tasks      | elapsed:  1.0min
[Parallel(n_jobs=-1)]: Done 2000 out of 2000 | elapsed:  1.1min finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.2s
[Parallel(n_jobs=12)]: Done 1226 tasks      | elapsed:    0.4s
[Parallel(n_jobs=12)]: Done 1776 tasks      | elapsed:    0.6s
[Parallel(n_jobs=12)]: Done 2000 out of 2000 | elapsed:    0.7s finished
```

```
[CV] END .................................n_estimators=2000; total time= 1.2min
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.8s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    6.7s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   15.1s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   27.0s
[Parallel(n_jobs=-1)]: Done 1226 tasks      | elapsed:   42.0s
[Parallel(n_jobs=-1)]: Done 1776 tasks      | elapsed:  1.0min
[Parallel(n_jobs=-1)]: Done 2000 out of 2000 | elapsed:  1.1min finished
```

```
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.3s
[Parallel(n_jobs=12)]: Done 1226 tasks      | elapsed:    0.4s
[Parallel(n_jobs=12)]: Done 1776 tasks      | elapsed:    0.7s
[Parallel(n_jobs=12)]: Done 2000 out of 2000 | elapsed:    0.7s finished
```

```
[CV] END .................................n_estimators=2000; total time= 1.2min
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.8s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    8.4s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   18.7s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   35.0s
[Parallel(n_jobs=-1)]: Done 1226 tasks      | elapsed:   56.7s
[Parallel(n_jobs=-1)]: Done 1776 tasks      | elapsed:  1.4min
[Parallel(n_jobs=-1)]: Done 2000 out of 2000 | elapsed:  1.5min finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.3s
[Parallel(n_jobs=12)]: Done 1226 tasks      | elapsed:    0.5s
[Parallel(n_jobs=12)]: Done 1776 tasks      | elapsed:    0.8s
[Parallel(n_jobs=12)]: Done 2000 out of 2000 | elapsed:    0.9s finished
```

```
[CV] END .................................n_estimators=2000; total time= 1.6min
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    2.0s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    8.0s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   19.9s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   37.0s
[Parallel(n_jobs=-1)]: Done 1226 tasks      | elapsed:   56.8s
[Parallel(n_jobs=-1)]: Done 1776 tasks      | elapsed:  1.4min
[Parallel(n_jobs=-1)]: Done 2426 tasks      | elapsed:  1.8min
[Parallel(n_jobs=-1)]: Done 3000 out of 3000 | elapsed:  2.2min finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.2s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.4s
[Parallel(n_jobs=12)]: Done 1226 tasks      | elapsed:    0.8s
[Parallel(n_jobs=12)]: Done 1776 tasks      | elapsed:    1.1s
[Parallel(n_jobs=12)]: Done 2426 tasks      | elapsed:    1.5s
[Parallel(n_jobs=12)]: Done 3000 out of 3000 | elapsed:    1.8s finished
```

```
[CV] END .................................n_estimators=3000; total time= 2.3min
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    7.1s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   16.9s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   32.2s
[Parallel(n_jobs=-1)]: Done 1226 tasks      | elapsed:   51.0s
[Parallel(n_jobs=-1)]: Done 1776 tasks      | elapsed:  1.3min
[Parallel(n_jobs=-1)]: Done 2426 tasks      | elapsed:  1.7min
[Parallel(n_jobs=-1)]: Done 3000 out of 3000 | elapsed:  2.0min finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.2s
[Parallel(n_jobs=12)]: Done 1226 tasks      | elapsed:    0.4s
[Parallel(n_jobs=12)]: Done 1776 tasks      | elapsed:    0.5s
[Parallel(n_jobs=12)]: Done 2426 tasks      | elapsed:    0.8s
[Parallel(n_jobs=12)]: Done 3000 out of 3000 | elapsed:    1.0s finished
```

```
[CV] END .................................n_estimators=3000; total time= 2.1min
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    6.9s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   16.1s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   28.7s
[Parallel(n_jobs=-1)]: Done 1226 tasks      | elapsed:   44.9s
[Parallel(n_jobs=-1)]: Done 1776 tasks      | elapsed:  1.1min
[Parallel(n_jobs=-1)]: Done 2426 tasks      | elapsed:  1.6min
[Parallel(n_jobs=-1)]: Done 3000 out of 3000 | elapsed:  1.9min finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.1s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.3s
[Parallel(n_jobs=12)]: Done 1226 tasks      | elapsed:    0.5s
[Parallel(n_jobs=12)]: Done 1776 tasks      | elapsed:    0.7s
[Parallel(n_jobs=12)]: Done 2426 tasks      | elapsed:    0.9s
[Parallel(n_jobs=12)]: Done 3000 out of 3000 | elapsed:    1.1s finished
```

```
[CV] END ................................n_estimators=3000; total time= 2.0min
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    1.4s
[Parallel(n_jobs=-1)]: Done 176 tasks      | elapsed:    7.6s
[Parallel(n_jobs=-1)]: Done 426 tasks      | elapsed:   17.1s
[Parallel(n_jobs=-1)]: Done 776 tasks      | elapsed:   30.6s
[Parallel(n_jobs=-1)]: Done 1226 tasks      | elapsed:   47.9s
[Parallel(n_jobs=-1)]: Done 1776 tasks      | elapsed:  1.2min
[Parallel(n_jobs=-1)]: Done 2426 tasks      | elapsed:  1.6min
[Parallel(n_jobs=-1)]: Done 3000 out of 3000 | elapsed:  2.0min finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 176 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done 426 tasks      | elapsed:    0.2s
[Parallel(n_jobs=12)]: Done 776 tasks      | elapsed:    0.3s
[Parallel(n_jobs=12)]: Done 1226 tasks      | elapsed:    0.5s
[Parallel(n_jobs=12)]: Done 1776 tasks      | elapsed:    0.7s
[Parallel(n_jobs=12)]: Done 2426 tasks      | elapsed:    1.0s
[Parallel(n_jobs=12)]: Done 3000 out of 3000 | elapsed:    1.2s finished
```

```
[CV] END ................................n_estimators=3000; total time= 2.1min
```

```
[Parallel(n_jobs=-1)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    0.7s
[Parallel(n_jobs=-1)]: Done  50 out of  50 | elapsed:    1.2s finished
```

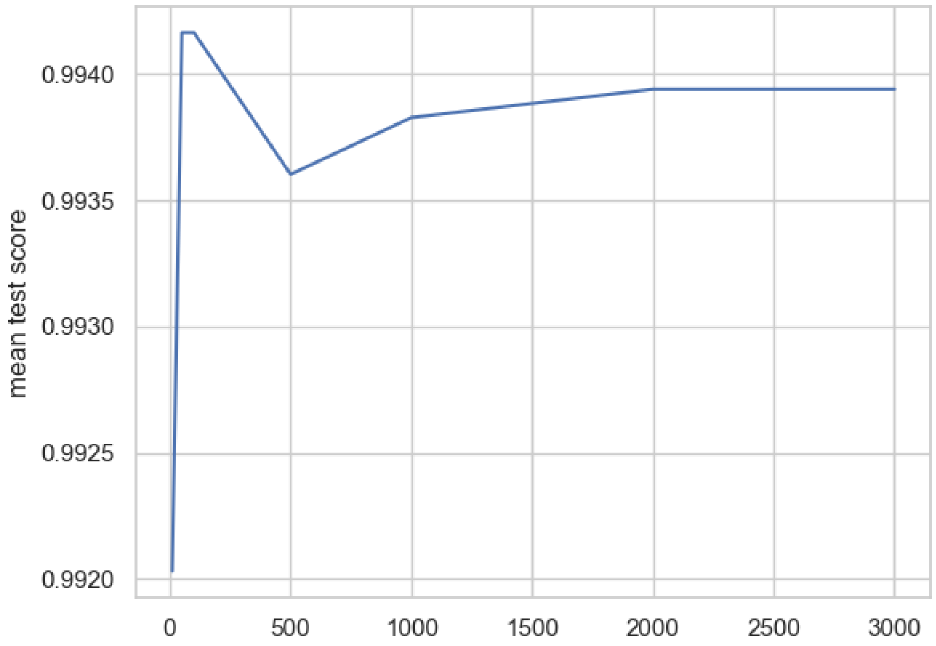
Out[64]:

```
RandomizedSearchCV(cv=4,
                   estimator=RandomForestClassifier(n_jobs=-1, random_state=25,
                                                    verbose=1),
                   n_iter=12,
                   param_distributions={'n_estimators': [10, 50, 100, 500, 1000,
                                                         2000, 3000]},
                   verbose=2)
```

In [70]:

```
sns.set(style="whitegrid")
plt.plot(model.cv_results_['param_n_estimators'].compressed(),model.cv_results_['mean_test_score'])
plt.xlabel("")
plt.ylabel("mean test score")
print("Best alpha =",model.best_params_['n_estimators']," With Best Mean CV Score =",model.best_score_)
print("="*100)
plt.show()
```

```
Best alpha = 50  With Best Mean CV Score = 0.9941644147757367
====================================================================================================
```



In [ ]:

```
calib = CalibratedClassifierCV(model.best_estimator_,method='sigmoid',n_jobs=-2)
calib.fit(X_train,y_tr)
```

In [36]:

```
# pk.dump(calib,open("rf_final.pk","wb"))
calib=pk.load(open("rf_final.pk","rb"))
```

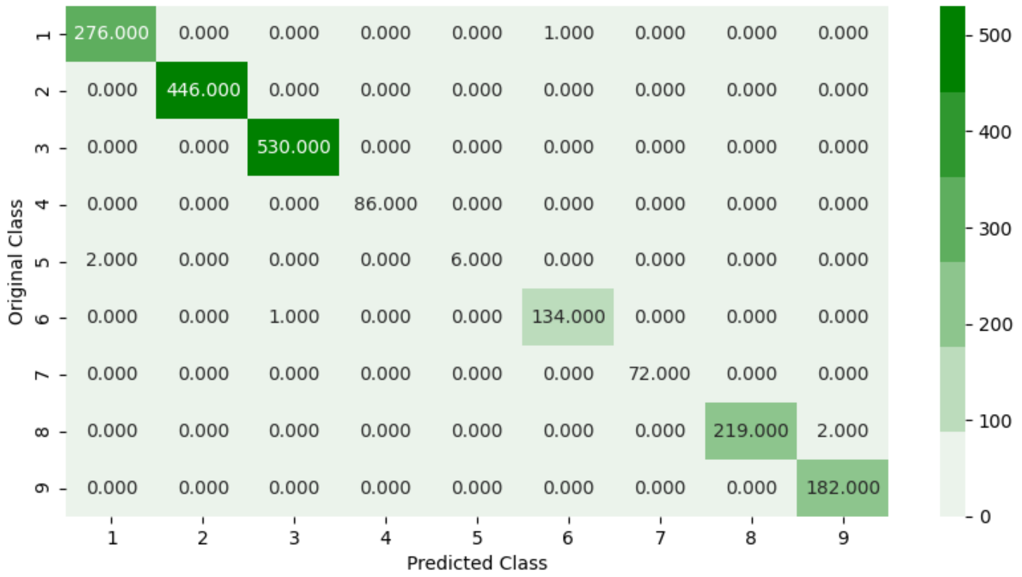
In [37]:

```
y_pred = calib.predict_proba(X_test)
loss = log_loss(y_tes,y_pred,eps=1e-15)
print("Log loss on Test Data using Random Forest Classifier",loss)

predicted_y =np.argmax(y_pred, axis=1)
misclassif = plot_confusion_matrix(y_tes, predicted_y+1)
```

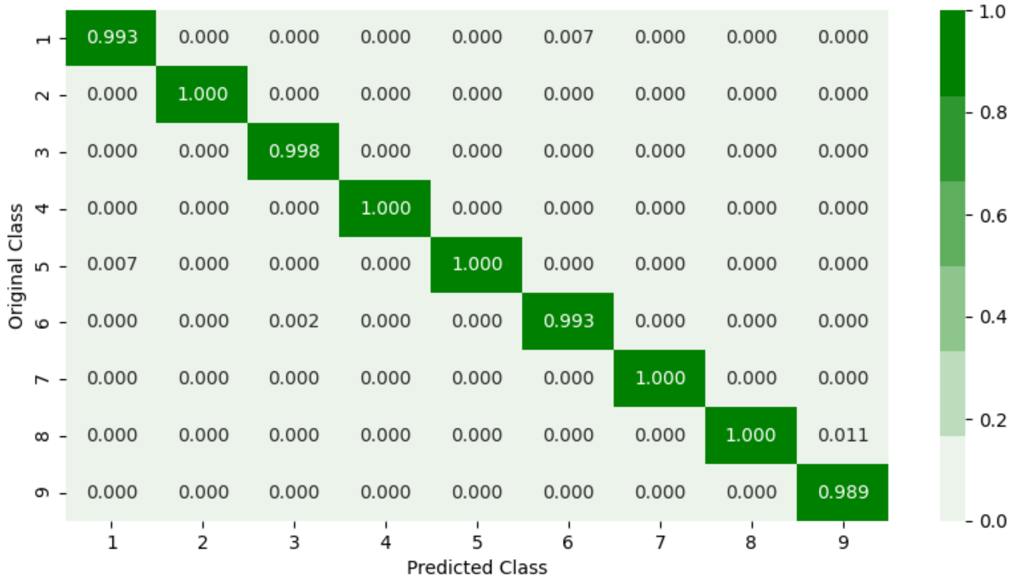
```
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
[Parallel(n_jobs=12)]: Using backend ThreadingBackend with 12 concurrent workers.
[Parallel(n_jobs=12)]: Done  26 tasks      | elapsed:    0.0s
```

```
Log loss on Test Data using Random Forest Classifier 0.020278156052861988
Number of misclassified points  0.3065917220235054
-------------------------------------------------- Confusion matrix -----------------------------------
---------------
```

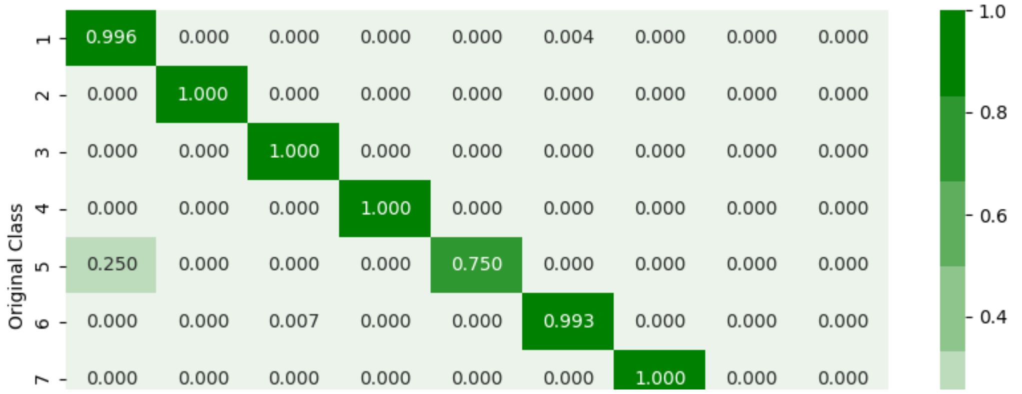
```
[Parallel(n_jobs=12)]: Done  50 out of  50 | elapsed:    0.0s finished
```

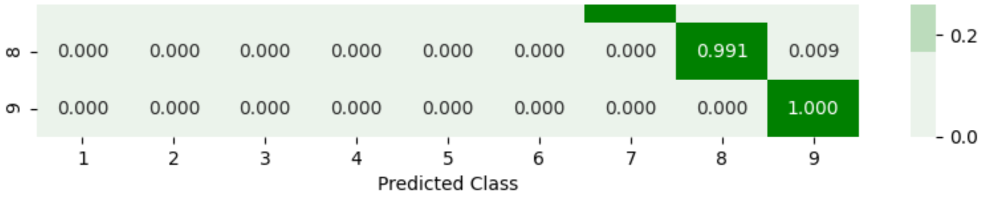


```
---------------------------------------------------- Precision matrix ------------------------------------
---------------
```



```
Sum of columns in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
---------------------------------------------------- Recall matrix ----------------------------------------
------------
```





```
Sum of rows in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

In [38]:

```
summary.append(["RF",loss,misclassif])
```

In [39]:

```
models['rf'] = calib
```

In [40]:

```
clf=None
model=None
gc.collect()
```

Out[40]:

```
85099
```

Xtreme Gradient Boosting

In [12]:

```
hyp={
    'learning_rate':[0.01,0.03,0.05,0.1,0.15,0.2],
     'n_estimators':[100,200,500,1000,2000],
     'max_depth':[3,5,10],
    'colsample_bytree':[0.1,0.3,0.5,1],
    'subsample':[0.1,0.3,0.5,1]
}
```

In [17]:

```
model=RandomizedSearchCV(XGBClassifier(verbose=3),param_distributions=hyp,verbose=10,n_jobs=5)
model.fit(X_train,y_tr)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
```

Out[17]:

```
RandomizedSearchCV(estimator=XGBClassifier(verbose=3), n_jobs=5,
                   param_distributions={'colsample_bytree': [0.1, 0.3, 0.5, 1],
                                        'learning_rate': [0.01, 0.03, 0.05, 0.1,
                                                          0.15, 0.2],
                                        'max_depth': [3, 5, 10],
                                        'n_estimators': [100, 200, 500, 1000,
                                                         2000],
                                        'subsample': [0.1, 0.3, 0.5, 1]},
                   verbose=10)
```

In [ ]:

```
# pk.dump(model,open("xgb_final.pk","wb"))
# model = pk.load(open("xgb_final.pk","rb"))
```

In [19]:

```
x1=model.cv_results_['param_n_estimators']
y1=model.cv_results_['param_max_depth']
z1=model.cv_results_['mean_test_score']
y2=model.cv_results_['param_subsample']
y3=model.cv_results_['param_colsample_bytree']
y4=model.cv_results_['param_learning_rate']
```

In [21]:

```
scene1 = dict(                                    #set labels for 2st plot
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='Mean CV Score'),)
scene2 = dict(                                    #set labels for 2st plot
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='subsample percentage'),
        zaxis = dict(title='Mean CV Score'),)
scene3 = dict(                                    #set labels for 2st plot
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='colsample_bytree percentage'),
        zaxis = dict(title='Mean CV Score'),)
scene4 = dict(                                    #set labels for 2st plot
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='learning_rate'),
        zaxis = dict(title='Mean CV Score'),)
fig =make_subplots(
    rows=2, cols=2,
    specs=[[{'is_3d': True}, {'is_3d': True}],[{'is_3d': True}, {'is_3d': True}]],
    subplot_titles=("Combination 1", "Combination 2","Combination 3","Combination 4")
    )
trace1 = go.Scatter3d(x=x1,y=y1,z=z1, name = 'Cross validation')
trace2 = go.Scatter3d(x=x1,y=y2,z=z1, name = 'Cross validation')
trace3 = go.Scatter3d(x=x1,y=y3,z=z1, name = 'Cross validation')
trace4 = go.Scatter3d(x=x1,y=y4,z=z1, name = 'Cross validation')

# data = [trace1]
# data = [trace2]

fig.add_trace(trace1,row=1,col=1)
fig.add_trace(trace2,row=1,col=2)
fig.add_trace(trace3,row=2,col=1)
fig.add_trace(trace4,row=2,col=2)



fig['layout']['scene'].update(scene1)
fig['layout']['scene2'].update(scene2)
fig['layout']['scene3'].update(scene3)
fig['layout']['scene4'].update(scene4)



offline.iplot(fig, filename='3d-scatter-colorscale')
```

In [45]:

```
# Ref: MicrosoftMalwareDetection.ipynb provided by Applied AI

x_cfl=XGBClassifier(n_estimators=500,max_depth=3,learning_rate=0.1,subsample=1,colsample_bytree=0.1)
x_cfl.fit(X_train,y_tr)
c_cfl=CalibratedClassifierCV(x_cfl,method='sigmoid',n_jobs=-2)
c_cfl.fit(X_train,y_tr)

predict_y = x_cfl.predict_proba(X_train)
print ('train loss',log_loss(y_tr, predict_y))
predict_y = x_cfl.predict_proba(X_test)
print ('test loss',log_loss(y_tes, predict_y))
```

```
C:\Users\sista\Anaconda3\lib\site-packages\xgboost\sklearn.py:1224: UserWarning:

The use of label encoder in XGBClassifier is deprecated and will be removed in a future release. To rem
ove this warning, do the following: 1) Pass option use_label_encoder=False when constructing XGBClassif
ier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class -
1].
```

```
[13:33:56] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.5.0/src/learner.cc:1115: S
tarting in XGBoost 1.3.0, the default evaluation metric used with the objective 'multi:softprob' was ch
anged from 'merror' to 'mlogloss'. Explicitly set eval_metric if you'd like to restore the old behavior
.
train loss 0.00039797665704385707
test loss 0.0073680679447485405
```

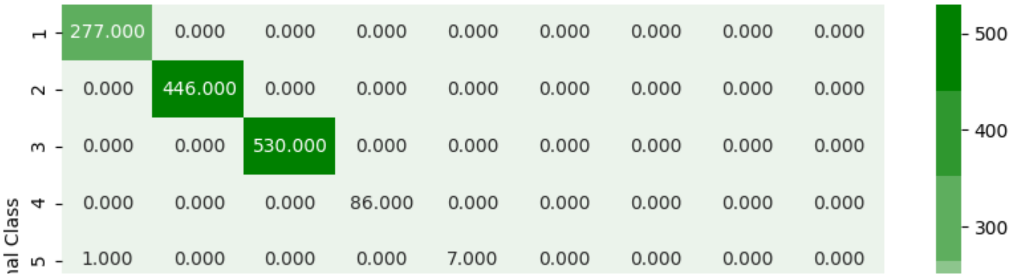
In [47]:

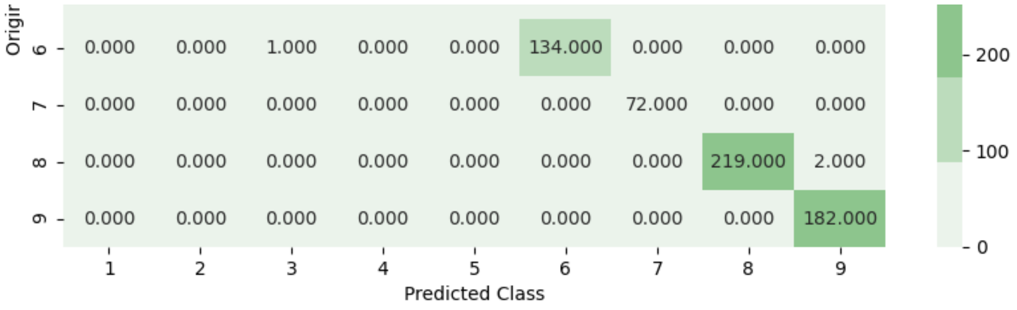
```
# pk.dump(c_cfl,open("xgb_final2.pk","wb"))
# x_cfl = pk.load(open("xgb_final2.pk","rb"))
```

In [49]:

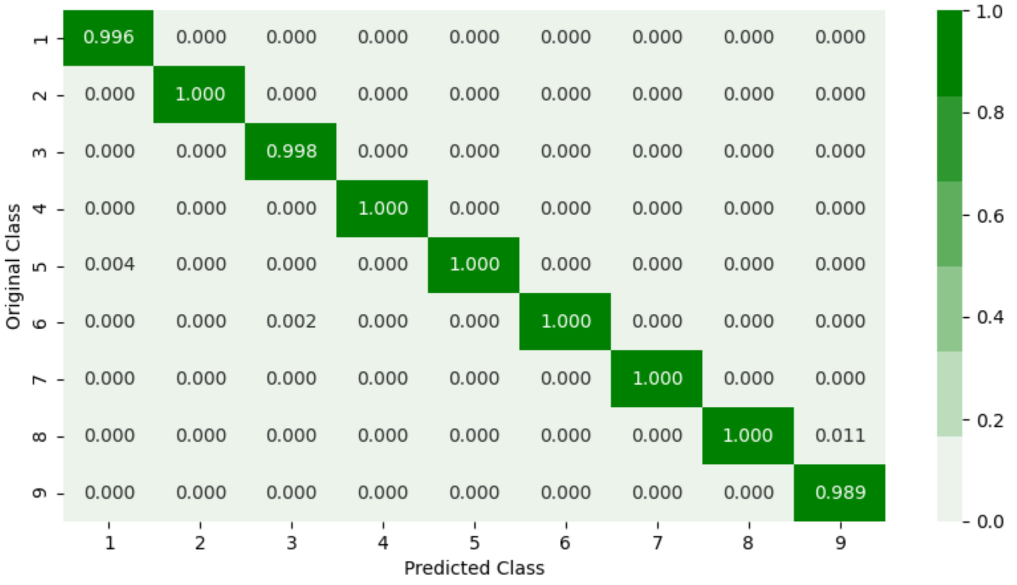
```
y_pred = c_cfl.predict_proba(X_test)
loss = log_loss(y_tes,predict_y,eps=1e-15)
print("Log loss on Test Data using XGBoost Classifier",loss)

predicted_y =np.argmax(predict_y, axis=1)
misclassif = plot_confusion_matrix(y_tes, predicted_y+1)
```

```
Log loss on Test Data using XGBoost Classifier 0.0073680679447485405
Number of misclassified points  0.2043944813490036
-------------------------------------------------- Confusion matrix -----------------------------------
---------------
```

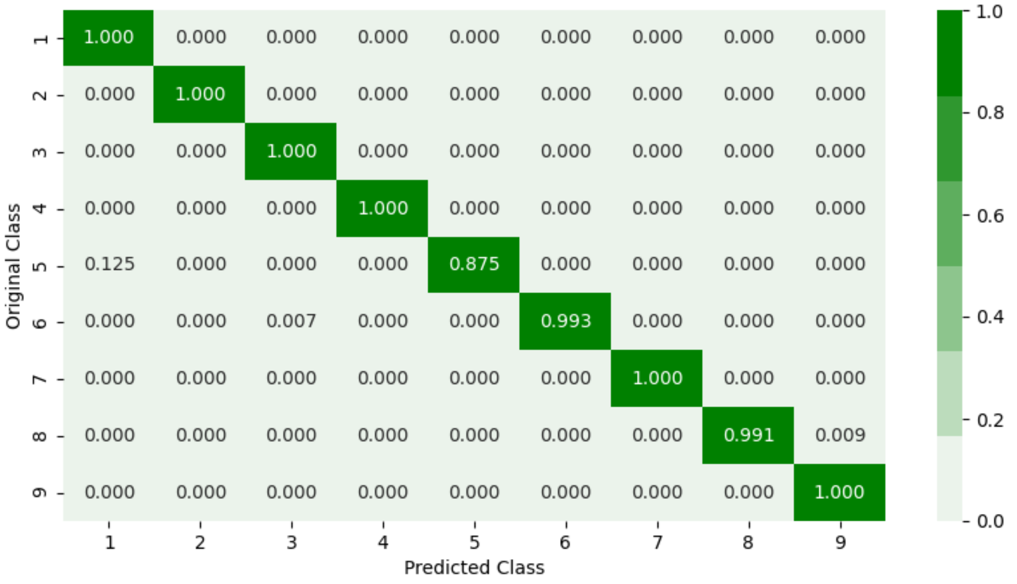




```
-------------------------------------------------- Precision matrix -----------------------------------
---------------
```



```
Sum of columns in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
-------------------------------------------------- Recall matrix --------------------------------------
------------
```



```
Sum of rows in precision matrix [1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

In [51]:

```
summary.append(["XGBClassifier",loss,misclassif])
```

In [52]:

```
models['xgb'] = calib
```

In [66]:

```
from tabulate import tabulate
```

In [67]:

```
print(tabulate(summary,headers='firstrow',tablefmt='grid'))
```

```
+---------------------+------------+-------------------------+
| Model               |   Log-Loss |   Misclassified Percent |
+=====================+============+=========================+
| Random Model        | 2.45585    |               88.1451   |
+---------------------+------------+-------------------------+
| KNN                 | 0.211947   |                5.26316  |
+---------------------+------------+-------------------------+
| Logistic Regression | 0.1872     |                2.29944  |
+---------------------+------------+-------------------------+
| RBF-SVM             | 0.321695   |                5.36536  |
+---------------------+------------+-------------------------+
| RF                  | 0.0202782  |                0.306592 |
+---------------------+------------+-------------------------+
| XGBClassifier       | 0.00736807 |                0.204394 |
+---------------------+------------+-------------------------+
```

Further Ideas:

- I really want to implement pseudo-labelling approach and check how it will improve the performance.
- I have implemented 4-gram hash encoded, I want to see how the model will change if I use normal 4-gram vectors.
- I really want to dive more into image features like texture of the pixels etc and implement all techniques discussed in link_to_research_paper
- I've learnt many concepts related to parallel computing, Dask, Multiprocessing library, Threading etc.. to reduce the time complexity, and I'm very eager to know more about these things as they really helped me with processing time. I guess I reduced time complexity by many folds and run everything on my local system.

Thank You For This Assignment, By Solving this I've learnt many technical skills as well as personal skills.