Part 1A:

```
The Average accuracy after 10 randon train-test splits:71.36363636363636:%
```

Part 1B:

```
The Average accuracy after handling missing data 10 randon train-test splits:73.24675324675326:%
```

Part 1D

Kernel=”Linear”

```
The  Average accuracy after 10 randon train-test splits (Support Vecotor Machines):76.03896103896103:%
```

Libraries used:

```
#Importing Libraries

import pandas as pd
import numpy as np
from sklearn.metrics import classification_report, confusion_matrix,accuracy_score
from sklearn import model_selection
from sklearn import svm
```

Fit Function and Probability Calculation:

```
def fit(X_train, Y_train):
    result = {}
    result["total_data"] = len(Y_train)
    class_values = set(Y_train)
    for current_class in class_values:
        result[current_class] = {}

        current_class_rows = (Y_train == current_class)
        X_train_current = X_train[current_class_rows]
        Y_train_current = Y_train[current_class_rows]
        num_features = X_train.shape[1]
        result[current_class]["total_count"] = len(Y_train_current)
        for j in range(1, num_features + 1):
            result[current_class][j] = {}
            all_possible_values = ['Mean',]
            for current_value in all_possible_values:

                result[current_class][j][current_value] = X_train_current.iloc[:,j-1].mean()
                all_possible_values = ['standard_deviation']
                for current_value in all_possible_values:
                    result[current_class][j]['standard_deviation'] = X_train_current.iloc[:,j-1].std()
    return result

# calculate the Probability

def probability(dictionary, x, current_class):
    output = np.log(dictionary[current_class]["total_count"]) - np.log(dictionary["total_data"])
    num_features = len(dictionary[current_class].keys()) - 1;
    for j in range(1, num_features + 1):

        xj = x[j - 1]

        first_term = 1/(np.sqrt(2*3.14*dictionary[current_class][j]['standard_deviation']))
        second_term = (((xj-dictionary[current_class][j]['Mean'])**2)/dictionary[current_class][j]['standard_deviation'
        current_xj_probablity = np.log(first_term) - second_term
        output = output + current_xj_probablity
    return output
```

Train /Test Split and Evaluation

```
# calculating the Accuracy

Accuracy=[]
for i in range(10):
    X_train,X_test,Y_train,Y_test = model_selection.train_test_split(X,Y,test_size=0.20)
    dictionary = fit(X_train,Y_train)
    Y_pred = predict(dictionary,X_test)
    Accuracy.append(accuracy_score(Y_test,Y_pred))
print("The Average accuracy after 10 randon train-test splits",sum(Accuracy)/len(Accuracy)*100,"%",sep=":")
```

```
Accuracy=[]
for i in range(10):
    X_train,X_test,Y_train,Y_test = model_selection.train_test_split(X,Y,test_size=0.20)
    clf = svm.SVC(kernel='linear')
    clf.fit(X_train,Y_train)
    Y_pred = clf.predict(X_test)
    Accuracy.append(accuracy_score(Y_test,Y_pred))
print("The  Average accuracy after 10 randon train-test splits (Support Vecotor Machines)",sum(Accuracy)/len(Accuracy)*
```

1. Gaussian + Untouched: 0.55560
2. Gaussian+ stretched: 0.79475
3. Bernoulli +Untouched: 0.83410
4. Bernoulli +Stretched: 0.81395
5. 10 trees + 4 depth + Untouched: 0.73585
6. 10 trees + 4 depth + stretched:0.67930
7. 10 trees + 16 depth + untouched:0.95795
8. 10 trees + 16 depth + Stretched:0.95905
9. 30 trees + 4 depth+ untouched: 0.78970
10. 30 trees + 4 depth + stretched: 0.73390
11. 30 trees + 16 depth + untouched: 0.96990
12. 30 trees + 16 depth + Stretched: 0.97115 - This Random forest model is the model as it is giving the best accuracy. We are having the majority class prediction of 30 trees for a particular test point. The depth =16 specifies that we are splitting the features in such a way that in end we are getting pure nodes.

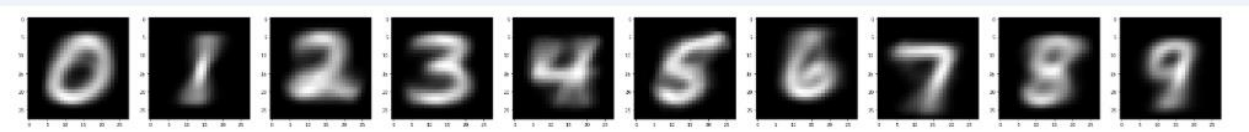
Submission and Description	Public Score	Use for Final Score
ts8_12 2 days ago by Tejveer Singh add submission details	0.97115	✓
ts8_11 2 days ago by Tejveer Singh add submission details	0.96990	✓
ts8_10 2 days ago by Tejveer Singh add submission details	0.73390	✓
ts8_9 2 days ago by Tejveer Singh add submission details	0.78970	✓
ts8_8 2 days ago by Tejveer Singh add submission details	0.95905	✓
ts8_7 2 days ago by Tejveer Singh add submission details	0.95795	✓

ts8_6 2 days ago by Tejveer Singh add submission details	0.67930	✓
ts8_5 2 days ago by Tejveer Singh add submission details	0.73585	✓
ts8_4 2 days ago by Tejveer Singh add submission details	0.81395	✓
ts8_3 2 days ago by Tejveer Singh add submission details	0.83410	✓
ts8_1 2 days ago by Tejveer Singh	0.55560	✓

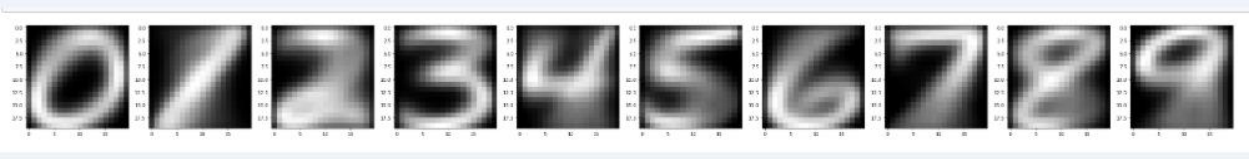
ts8_2 2 days ago by Tejveer Singh add submission details	0.79475	✓

Mean Images:

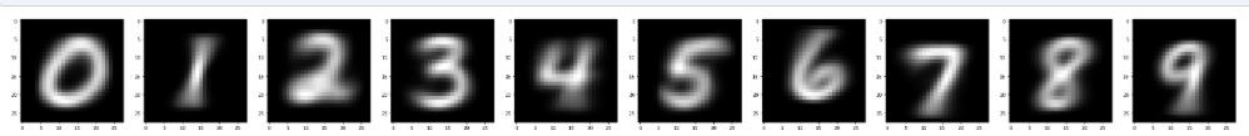
Gaussian + Untouched:



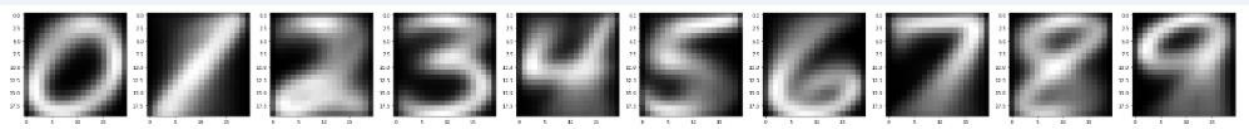
Gaussian + Touched:



Bernoulli + Untouched



Bernoulli +Touched



Libraries Used:

```
import numpy as np
import pandas as pd
from sklearn.naive_bayes import GaussianNB
from sklearn.naive_bayes import BernoulliNB
from sklearn import model_selection
from sklearn.metrics import classification_report, confusion_matrix,accuracy_score
from sklearn.model_selection import cross_val_score
import matplotlib.pyplot as plt
from sklearn.model_selection import KFold
from numpy import array
import skimage.transform
from sklearn.ensemble import RandomForestClassifier
```

Naïve Bayes Bernoulli Classifier

```
#Bernoulli + stretched

clf = BernoulliNB()
clf.fit(l1,Y)

y_pred=clf.predict(l)
print("The accuracy on validation data (Bernoulli + stretched) :",accuracy_score(Y_val,y_pred)*100,"%",sep="")
```

Naïve Bayes Gaussian Classifier

```
#Gaussian + untouched

clf = GaussianNB()
clf.fit(X,Y)

y_pred=clf.predict(X_val)
print("The accuracy on validation data (Gaussian + untouched) :",accuracy_score(Y_val,y_pred)*100,"%",sep="")

y_test=clf.predict(X_test)
```

Random Forest Classifier

```
#30 trees + 16 depth + stretched

clf = RandomForestClassifier(n_estimators=30,max_depth=16, random_state=0)
clf.fit(l1,Y)
y_pred=clf.predict(l)
print("The accuracy on validation data (30 trees + 16 depth + strtched) :",accuracy_score(Y_val,y_pred)*100,"%",sep="")
```