Epileptic Seizure Recognition

Mayar Mansour
201700072
Muhammad Elsadany
201700857

Outline

- Problem Definition
- Dataset Description
- Relevant Prior Work
- Project Pipeline
- Project Progress
- Workload Distribution

Problem Definition

- Epilepsy is a neurological disorder that affects the Central Nervous System (CNS) causing seizures or unusual behaviours, abnormal brain activity, or loss of consciousness sometimes [1]
- Seizures are the result of sudden surge of electrical activity in the brain. This electrical activity is caused due to complex chemical imbalance that occurs in nerve cells in the brain. While some seizures may disable a person, other types of seizures may not be identified at all. [2]
- That is why a device capable of detecting a seizure and notifying a caregiver could make dealing with epilepsy easier.

Properties of the Electroencephalogram (EEG) [3]

- EEG is the continuous recording of the electrical brain activity.
- There are billions of brain cells called neurons, and the neurons have axons that relieve neurotransmitters and the dendrites they receive them. The axons of other neurons they cause electrical polarity change inside the neuron.
- This polarity change is what the EEG is recording. The brain gives off electromagnetic waves when there is electricity running in the brain, which produces brain waves that is amplified quite a lot.

Dataset Collection [4]

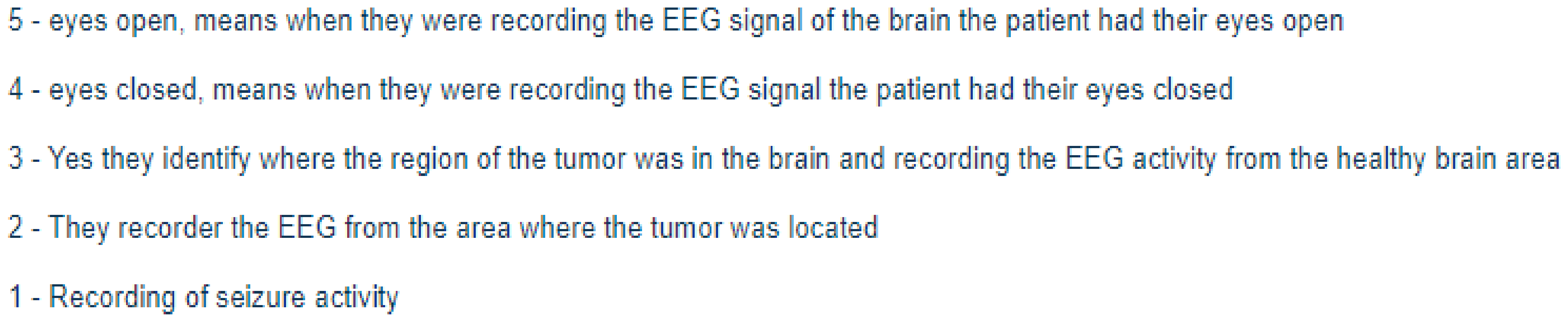
- The dataset was originally collected using a **128-channel amplifier system** and the EEG signal recorded for 3.3 h and sampled at 173.61 Hz while the settings of the bandpass filter were set at 0.53–40 Hz (12 dB/oct.).
- The data were digitized at **173.61 samples per second** using 12 bit resolution.
- After analyzing for artifacts, e.g., because of activity in the muscular area or optic movement, 28 channels were excluded and only 23.6 seconds were used.

Dataset Description [5]

- The original dataset contains 5 different folders each representing a specific class. Each folder has **100 files** (File/subject)
- **Each file** is a recording of brain activity for **23.6 seconds**. The corresponding time-series is sampled into **4097 data points**.
- They **divided** and shuffled every 4097 data points **into 23 chunks**, each chunk contains **178 data points per 1 second**, and each data point is the value of the EEG recording at a different point in time.

Dataset Description (cont.)

- In total, we have **23 x 500 = 11500** pieces of information(row), each information contains 178 data points for 1 second(column).
- The last column represents the label y {1,2,3,4,5}.



5 - eyes open, means when they were recording the EEG signal of the brain the patient had their eyes open

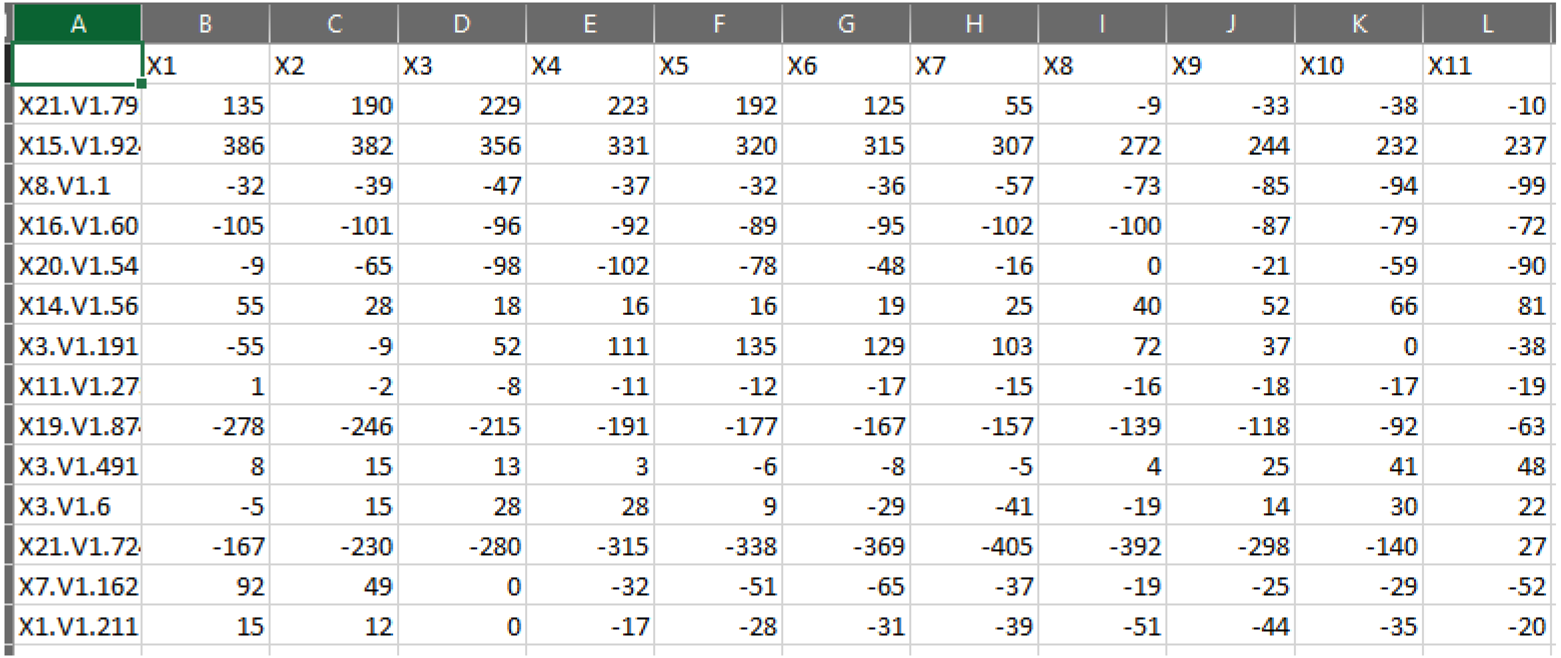
4 - eyes closed, means when they were recording the EEG signal the patient had their eyes closed

3 - Yes they identify where the region of the tumor was in the brain and recording the EEG activity from the healthy brain area

2 - They recorder the EEG from the area where the tumor was located

1 - Recording of seizure activity

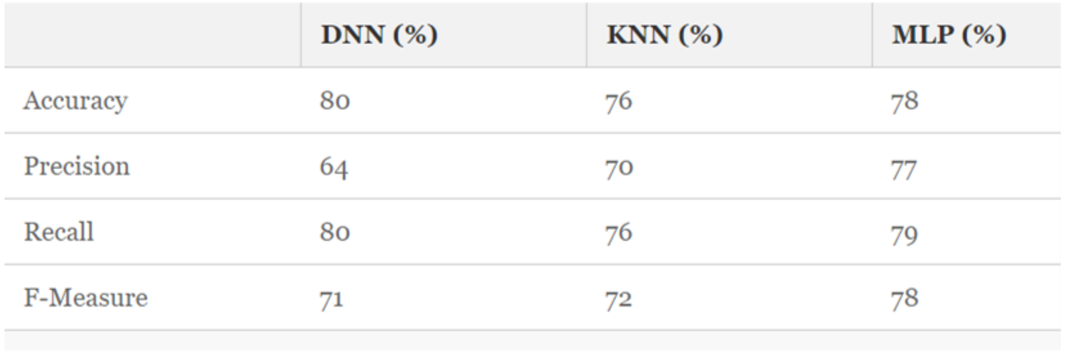
Dataset Sample



A	B	C	D	E	F	G	H	I	J	K	L
	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11
X21.V1.79	135	190	229	223	192	125	55	-9	-33	-38	-10
X15.V1.92	386	382	356	331	320	315	307	272	244	232	237
X8.V1.1	-32	-39	-47	-37	-32	-36	-57	-73	-85	-94	-99
X16.V1.60	-105	-101	-96	-92	-89	-95	-102	-100	-87	-79	-72
X20.V1.54	-9	-65	-98	-102	-78	-48	-16	0	-21	-59	-90
X14.V1.56	55	28	18	16	16	19	25	40	52	66	81
X3.V1.191	-55	-9	52	111	135	129	103	72	37	0	-38
X11.V1.27	1	-2	-8	-11	-12	-17	-15	-16	-18	-17	-19
X19.V1.87	-278	-246	-215	-191	-177	-167	-157	-139	-118	-92	-63
X3.V1.491	8	15	13	3	-6	-8	-5	4	25	41	48
X3.V1.6	-5	15	28	28	9	-29	-41	-19	14	30	22
X21.V1.72	-167	-230	-280	-315	-338	-369	-405	-392	-298	-140	27
X7.V1.162	92	49	0	-32	-51	-65	-37	-19	-25	-29	-52
X1.V1.211	15	12	0	-17	-28	-31	-39	-51	-44	-35	-20

Relevant Prior Work [5]

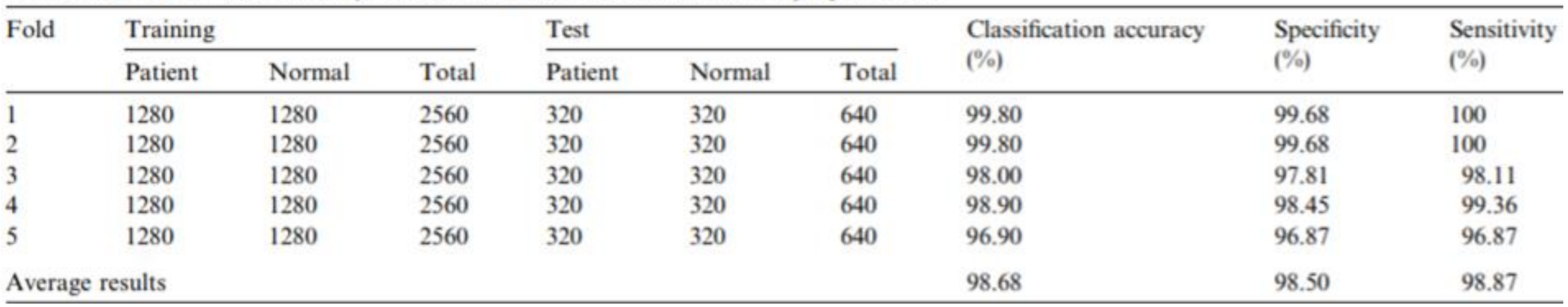
- In [5], they used the Deep Neural Network (DNN) as their classifier then compared its accuracy with other traditional classifier for this problem domain: multilayer perceptron (MLP) and k-nearest neighbor (KNN)
- Table 1 shows that **DNN** is the best classifier in terms of **accuracy**
- Hence, the paper claimed that the DNN model exhibits the best signs of performance and can be concluded as the ideal classifier for this problem, despite the fact that DNN has the **lowest F1 score** among the three.



	DNN (%)	KNN (%)	MLP (%)
Accuracy	80	76	78
Precision	64	70	77
Recall	80	76	79
F-Measure	71	72	78

Relevant Prior Work (cont.) [6]

- In [6], they used only class A & E, then used Fast Fourier Transform (FFT) with decision tree and k-fold cross validation. This made their classifier into either seizure or non-seizure.


Obtained 5-fold CV test results by decision tree classifier for detection of epileptic seizure

Fold	Training			Test			Classification accuracy (%)	Specificity (%)	Sensitivity (%)
	Patient	Normal	Total	Patient	Normal	Total			
1	1280	1280	2560	320	320	640	99.80	99.68	100
2	1280	1280	2560	320	320	640	99.80	99.68	100
3	1280	1280	2560	320	320	640	98.00	97.81	98.11
4	1280	1280	2560	320	320	640	98.90	98.45	99.36
5	1280	1280	2560	320	320	640	96.90	96.87	96.87
Average results							98.68	98.50	98.87

Relevant Prior Work (cont.) [7]

- They also used class A & E only then performed discrete wavelet transformation. They used SVM as their classifier.
- Moreover, in order to reduce the dimension of data, they tried Principal components analysis (PCA), independent components analysis (ICA) and linear discriminant analysis (LDA) to see which is the best.
- Table 3 showed that **PCA** was the **worst for feature selection**

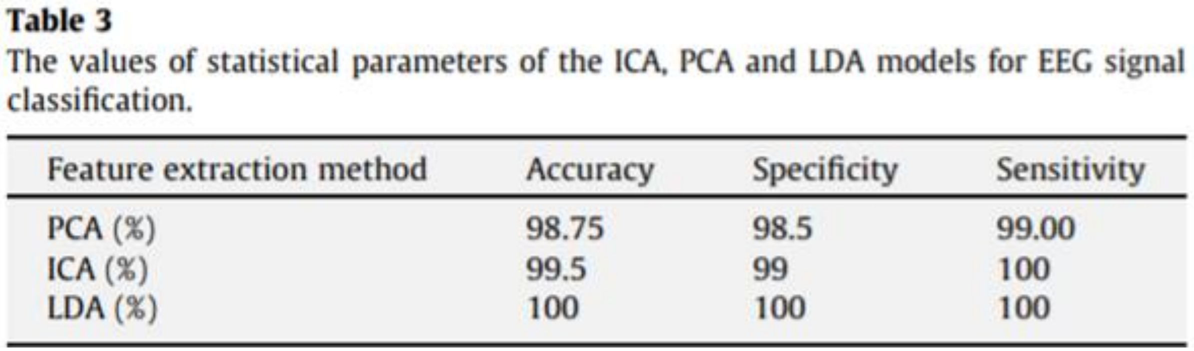


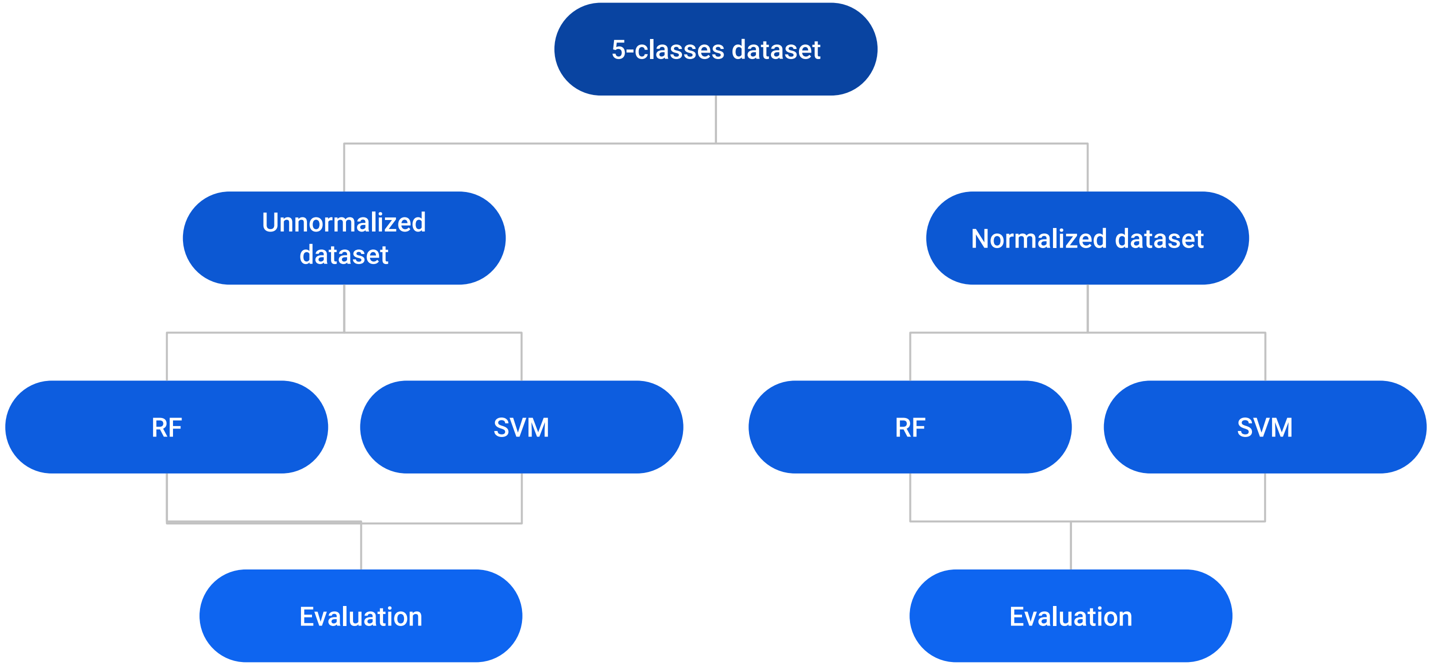
Table 3
The values of statistical parameters of the ICA, PCA and LDA models for EEG signal classification.

Feature extraction method	Accuracy	Specificity	Sensitivity
PCA (%)	98.75	98.5	99.00
ICA (%)	99.5	99	100
LDA (%)	100	100	100

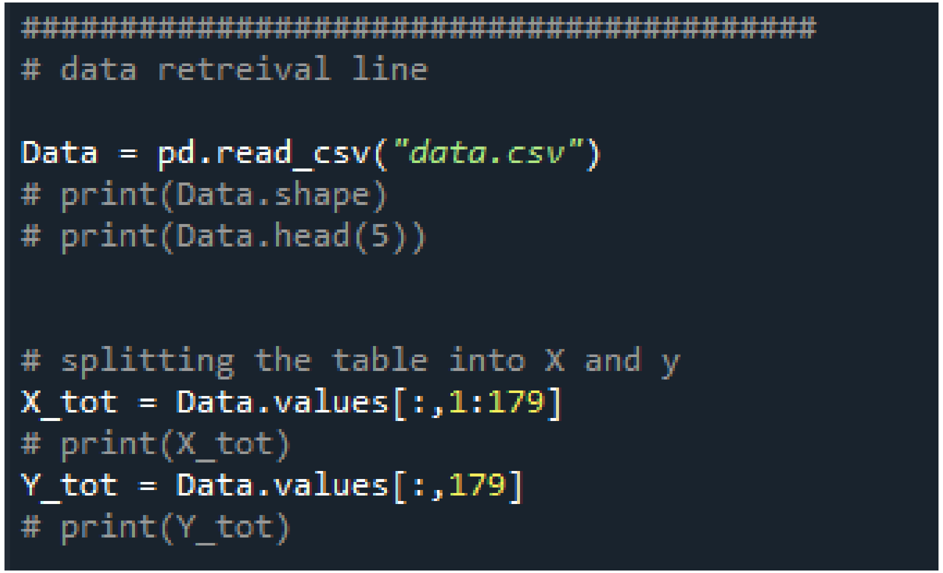
Our Motivation

- As DNN yielded low F1-score and low interpretability. We will test two models: Random Forest (RF) and Support Vector Machine (SVM) to improve the F1-score and interpretability of the model.
- Our final objective is to find the ideal classifier for this problem domain in terms of interpretability and F1-score

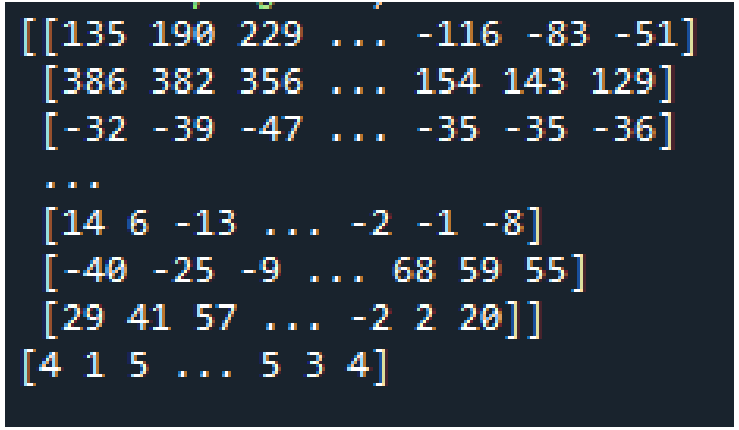
Project Pipeline



Project Pipeline: Reading Data

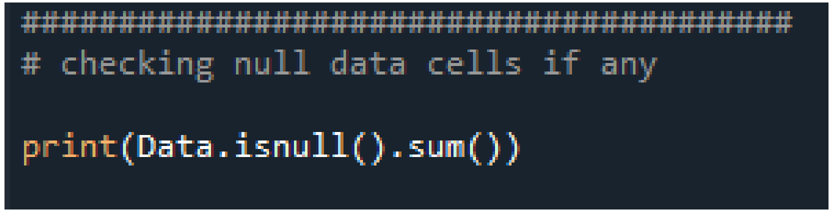


```
#########################################
# data retreival line

Data = pd.read_csv("data.csv")
# print(Data.shape)
# print(Data.head(5))


# splitting the table into X and y
X_tot = Data.values[:,1:179]
# print(X_tot)
Y_tot = Data.values[:,179]
# print(Y_tot)
```

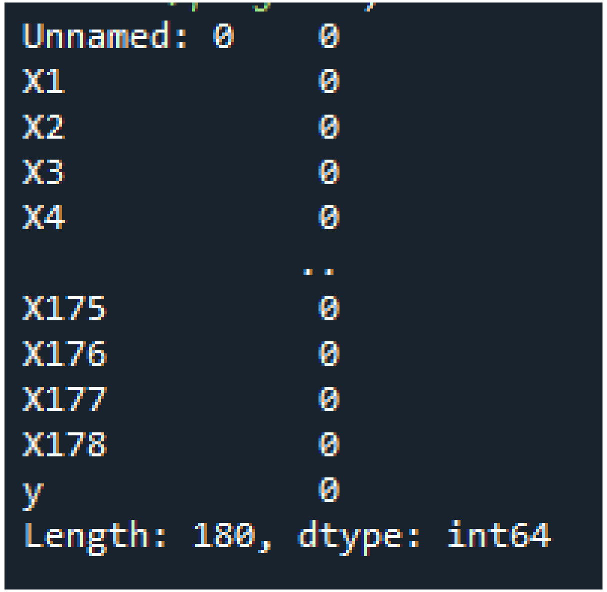


```
[[135 190 229 ... -116 -83 -51]
 [386 382 356 ... 154 143 129]
 [-32 -39 -47 ... -35 -35 -36]
 ...
 [14 6 -13 ... -2 -1 -8]
 [-40 -25 -9 ... 68 59 55]
 [29 41 57 ... -2 2 20]]
[4 1 5 ... 5 3 4]
```

Project Pipeline: Pre-Processing (Checking Null Data)

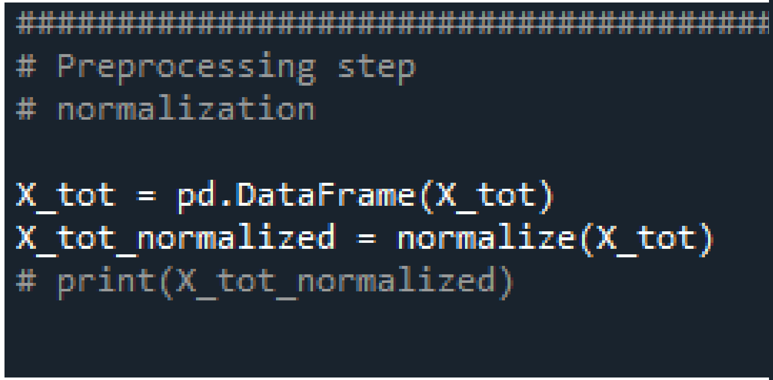


```
########################################
# checking null data cells if any

print(Data.isnull().sum())
```

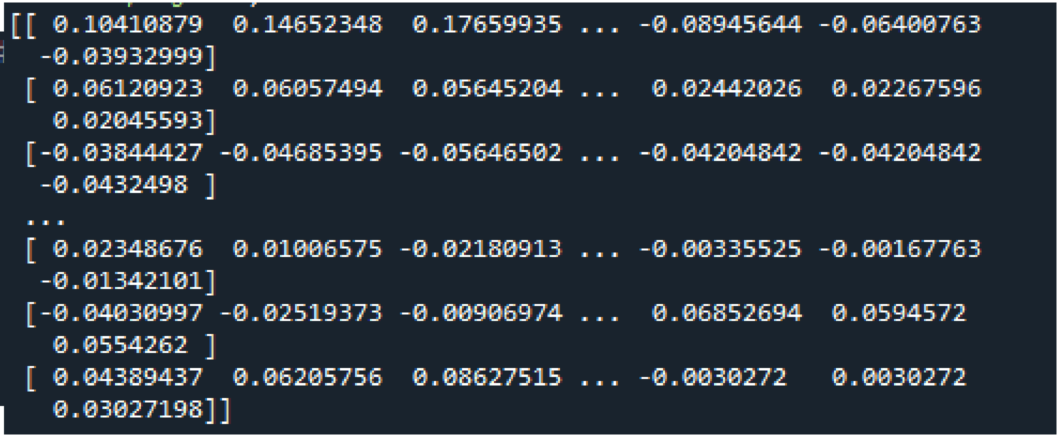


```
Unnamed: 0    0
X1            0
X2            0
X3            0
X4            0
             ..
X175          0
X176          0
X177          0
X178          0
y             0
Length: 180, dtype: int64
```

Project Pipeline: Pre-Processing (Normalization)

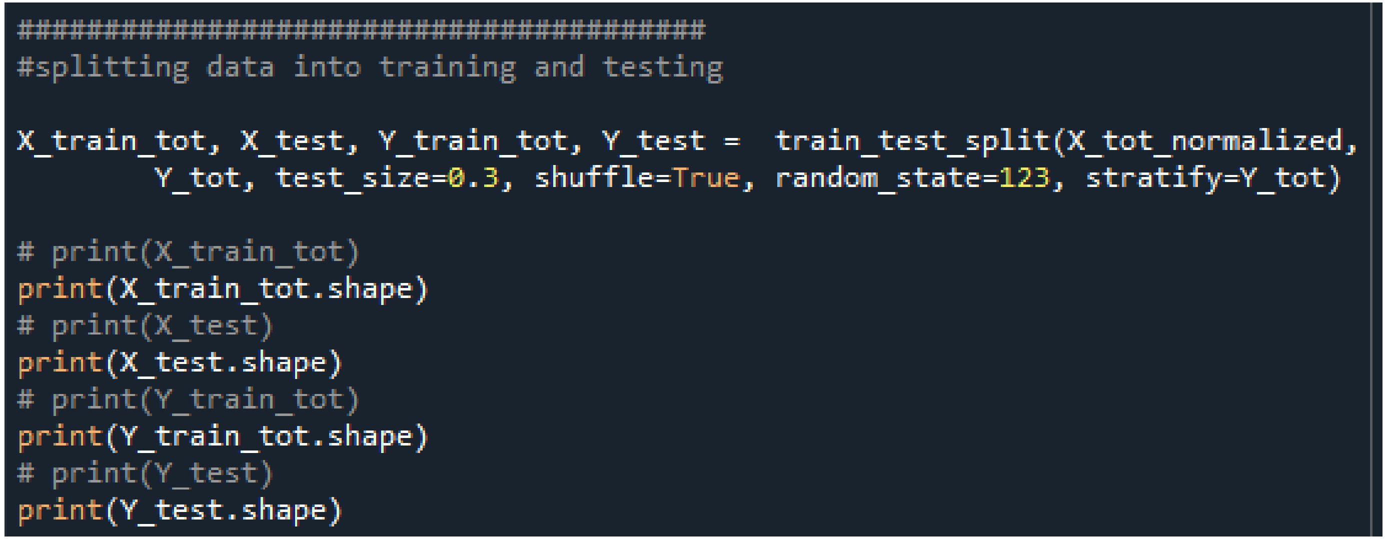


```
##############################################
# Preprocessing step
# normalization

X_tot = pd.DataFrame(X_tot)
X_tot_normalized = normalize(X_tot)
# print(X_tot_normalized)
```

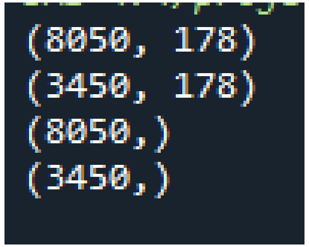


```
[[ 0.10410879  0.14652348  0.17659935 ... -0.08945644 -0.06400763
  -0.03932999]
 [ 0.06120923  0.06057494  0.05645204 ...  0.02442026  0.02267596
   0.02045593]
 [-0.03844427 -0.04685395 -0.05646502 ... -0.04204842 -0.04204842
  -0.0432498 ]
 ...
 [ 0.02348676  0.01006575 -0.02180913 ... -0.00335525 -0.00167763
  -0.01342101]
 [-0.04030997 -0.02519373 -0.00906974 ...  0.06852694  0.0594572
   0.0554262 ]
 [ 0.04389437  0.06205756  0.08627515 ... -0.0030272   0.0030272
   0.03027198]]
```

Project Pipeline: Splitting Data

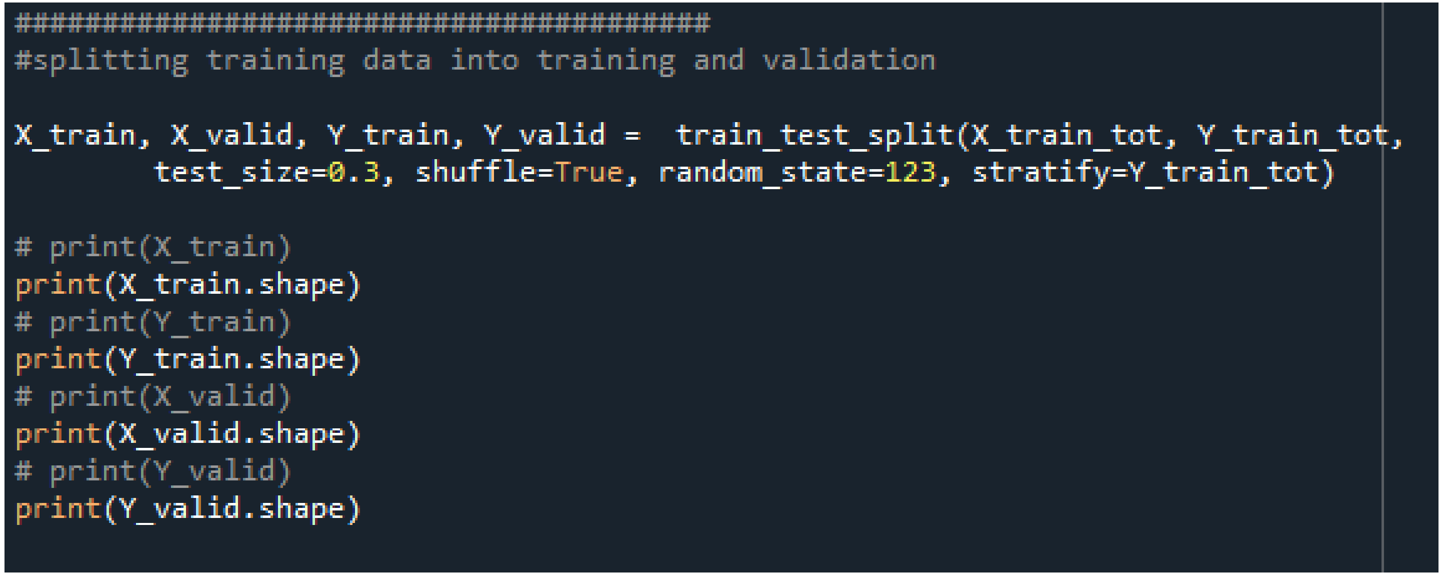


```
########################################
#splitting data into training and testing

X_train_tot, X_test, Y_train_tot, Y_test =  train_test_split(X_tot_normalized,
        Y_tot, test_size=0.3, shuffle=True, random_state=123, stratify=Y_tot)

# print(X_train_tot)
print(X_train_tot.shape)
# print(X_test)
print(X_test.shape)
# print(Y_train_tot)
print(Y_train_tot.shape)
# print(Y_test)
print(Y_test.shape)
```

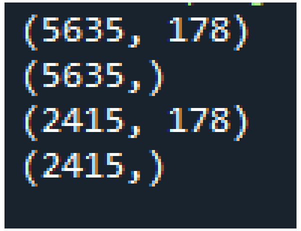


```
(8050, 178)
(3450, 178)
(8050,)
(3450,)
```

Project Pipeline: Splitting Training-data

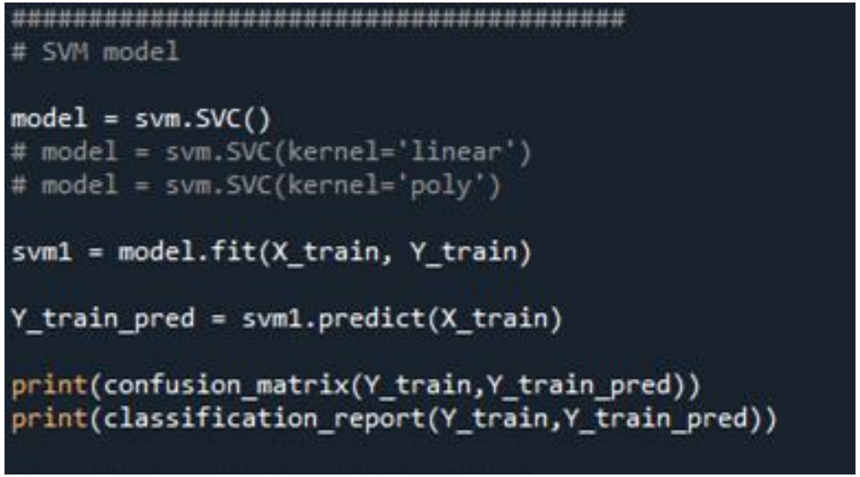


```
############################################
#splitting training data into training and validation

X_train, X_valid, Y_train, Y_valid =  train_test_split(X_train_tot, Y_train_tot,
        test_size=0.3, shuffle=True, random_state=123, stratify=Y_train_tot)

# print(X_train)
print(X_train.shape)
# print(Y_train)
print(Y_train.shape)
# print(X_valid)
print(X_valid.shape)
# print(Y_valid)
print(Y_valid.shape)
```

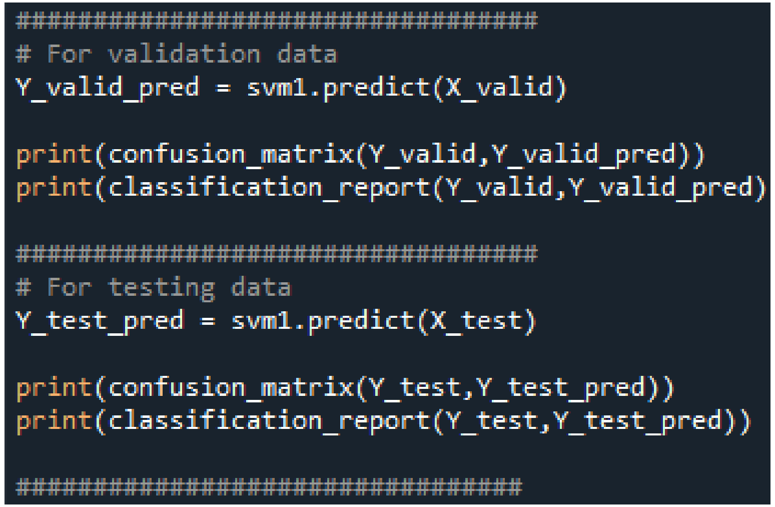


```
(5635, 178)
(5635,)
(2415, 178)
(2415,)
```

Project Pipeline: SVM model

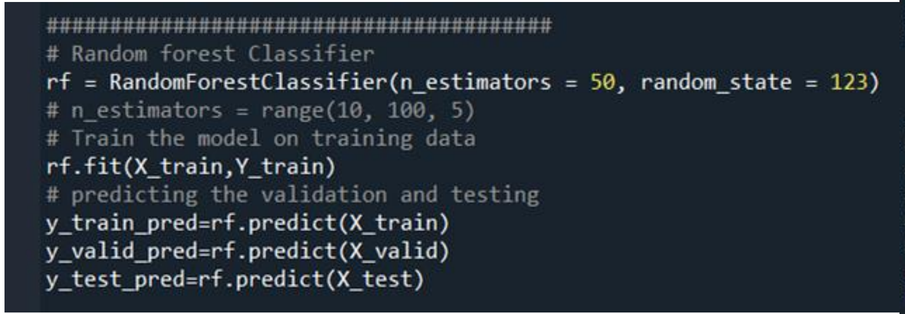


```
########################################
# SVM model

model = svm.SVC()
# model = svm.SVC(kernel='linear')
# model = svm.SVC(kernel='poly')

svm1 = model.fit(X_train, Y_train)

Y_train_pred = svm1.predict(X_train)

print(confusion_matrix(Y_train,Y_train_pred))
print(classification_report(Y_train,Y_train_pred))
```

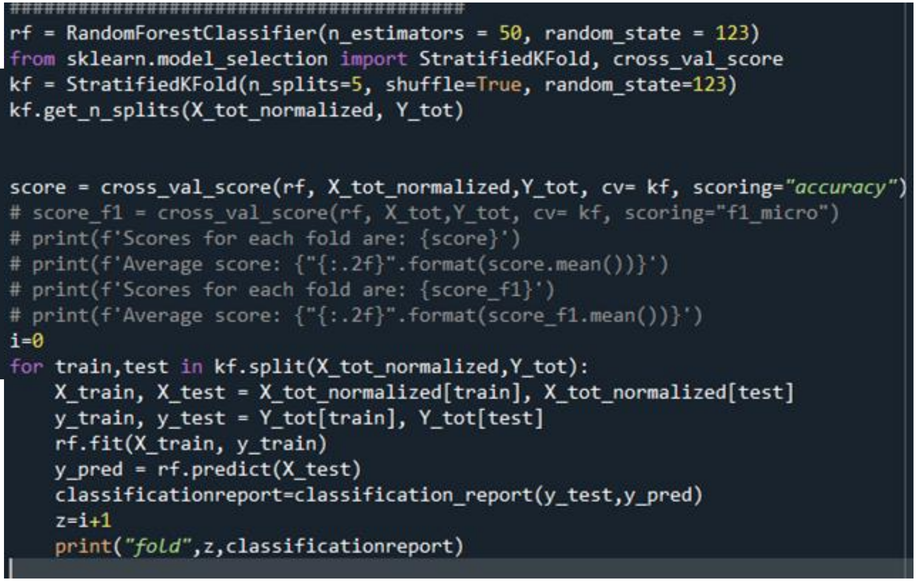


```
##################################
# For validation data
Y_valid_pred = svm1.predict(X_valid)

print(confusion_matrix(Y_valid,Y_valid_pred))
print(classification_report(Y_valid,Y_valid_pred)

##################################
# For testing data
Y_test_pred = svm1.predict(X_test)

print(confusion_matrix(Y_test,Y_test_pred))
print(classification_report(Y_test,Y_test_pred))

#################################
```

Project Pipeline: Random Forest model

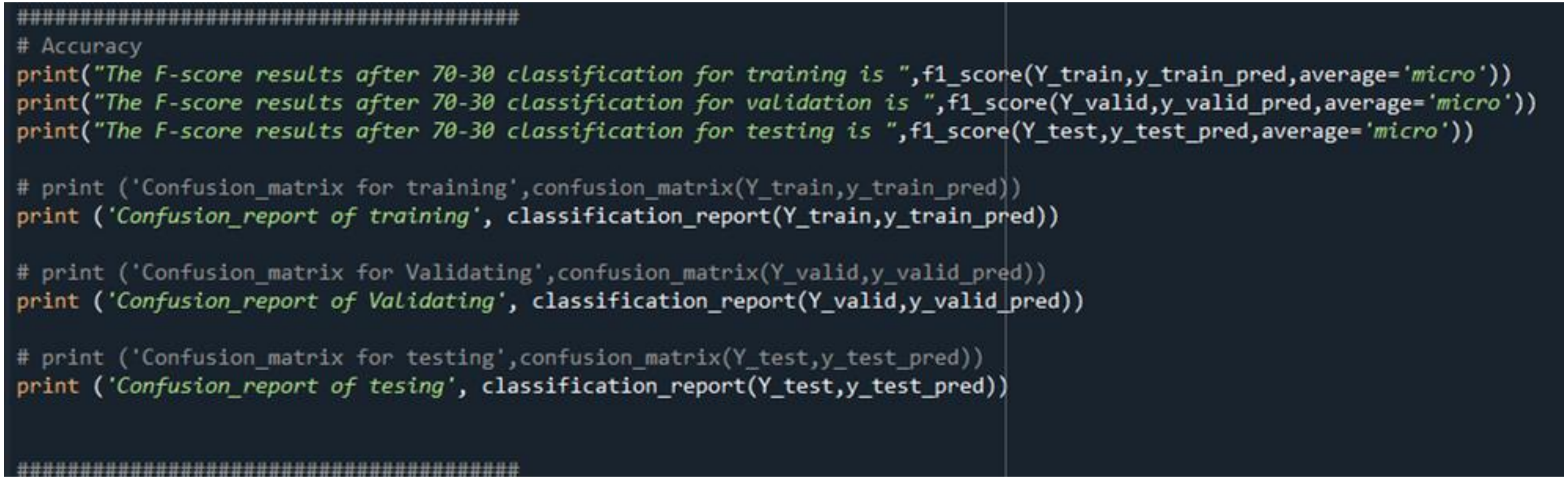


```
########################################
# Random forest Classifier
rf = RandomForestClassifier(n_estimators = 50, random_state = 123)
# n_estimators = range(10, 100, 5)
# Train the model on training data
rf.fit(X_train,Y_train)
# predicting the validation and testing
y_train_pred=rf.predict(X_train)
y_valid_pred=rf.predict(X_valid)
y_test_pred=rf.predict(X_test)
```



```
########################################
rf = RandomForestClassifier(n_estimators = 50, random_state = 123)
from sklearn.model_selection import StratifiedKFold, cross_val_score
kf = StratifiedKFold(n_splits=5, shuffle=True, random_state=123)
kf.get_n_splits(X_tot_normalized, Y_tot)


score = cross_val_score(rf, X_tot_normalized,Y_tot, cv= kf, scoring="accuracy")
# score_f1 = cross_val_score(rf, X_tot,Y_tot, cv= kf, scoring="f1_micro")
# print(f'Scores for each fold are: {score}')
# print(f'Average score: {"{:.2f}".format(score.mean())}')
# print(f'Scores for each fold are: {score_f1}')
# print(f'Average score: {"{:.2f}".format(score_f1.mean())}')
i=0
for train,test in kf.split(X_tot_normalized,Y_tot):
    X_train, X_test = X_tot_normalized[train], X_tot_normalized[test]
    y_train, y_test = Y_tot[train], Y_tot[test]
    rf.fit(X_train, y_train)
    y_pred = rf.predict(X_test)
    classificationreport=classification_report(y_test,y_pred)
    z=i+1
    print("fold",z,classificationreport)
```

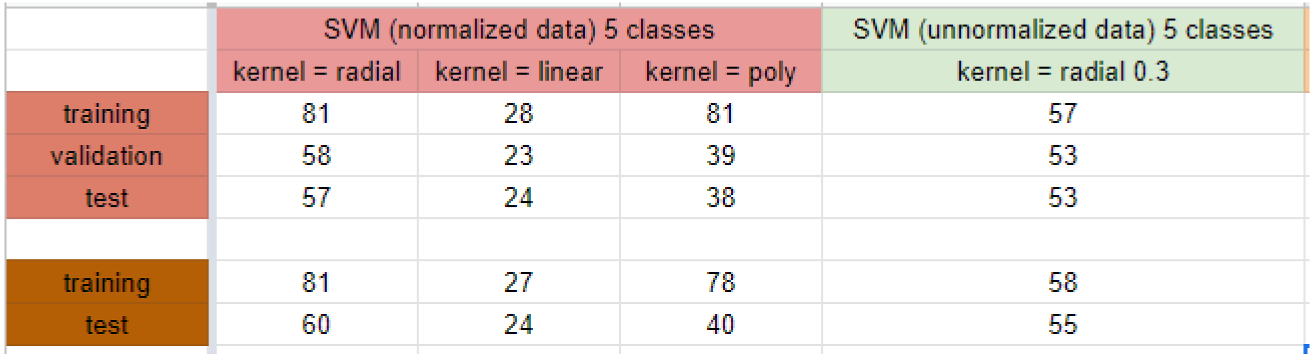
Project Pipeline: Random Forest model Accuracy



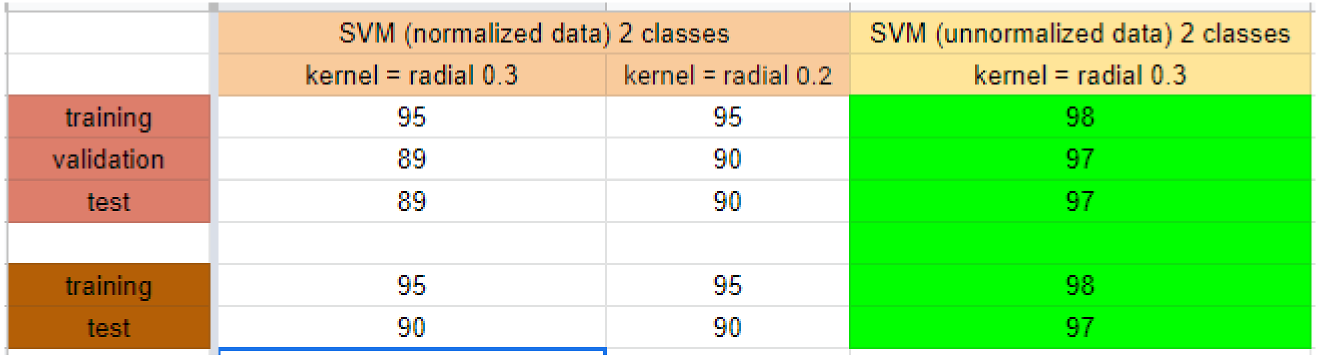
```
########################################
# Accuracy
print("The F-score results after 70-30 classification for training is ",f1_score(Y_train,y_train_pred,average='micro'))
print("The F-score results after 70-30 classification for validation is ",f1_score(Y_valid,y_valid_pred,average='micro'))
print("The F-score results after 70-30 classification for testing is ",f1_score(Y_test,y_test_pred,average='micro'))

# print ('Confusion_matrix for training',confusion_matrix(Y_train,y_train_pred))
print ('Confusion_report of training', classification_report(Y_train,y_train_pred))

# print ('Confusion_matrix for Validating',confusion_matrix(Y_valid,y_valid_pred))
print ('Confusion_report of Validating', classification_report(Y_valid,y_valid_pred))

# print ('Confusion_matrix for testing',confusion_matrix(Y_test,y_test_pred))
print ('Confusion_report of tesing', classification_report(Y_test,y_test_pred))


########################################
```

RESULTS

Results: SVM (accuracy)

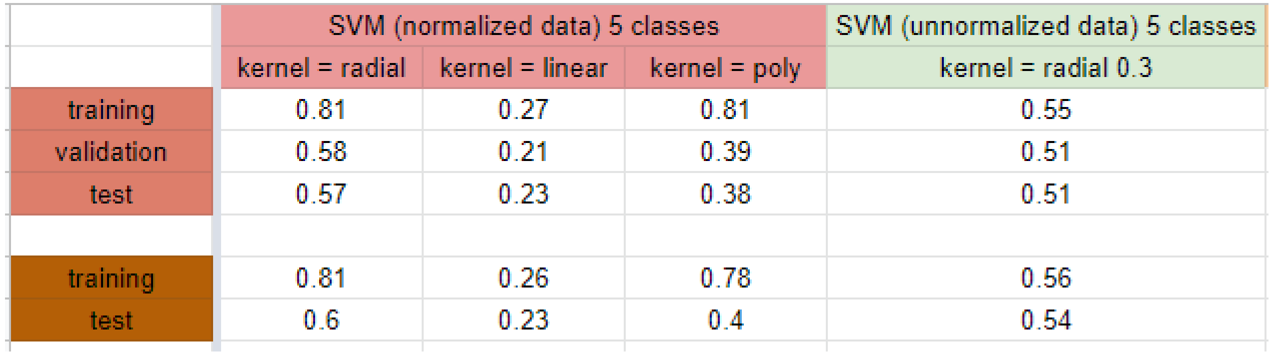


	SVM (normalized data) 5 classes			SVM (unnormalized data) 5 classes
	kernel = radial	kernel = linear	kernel = poly	kernel = radial 0.3
training	81	28	81	57
validation	58	23	39	53
test	57	24	38	53
training	81	27	78	58
test	60	24	40	55

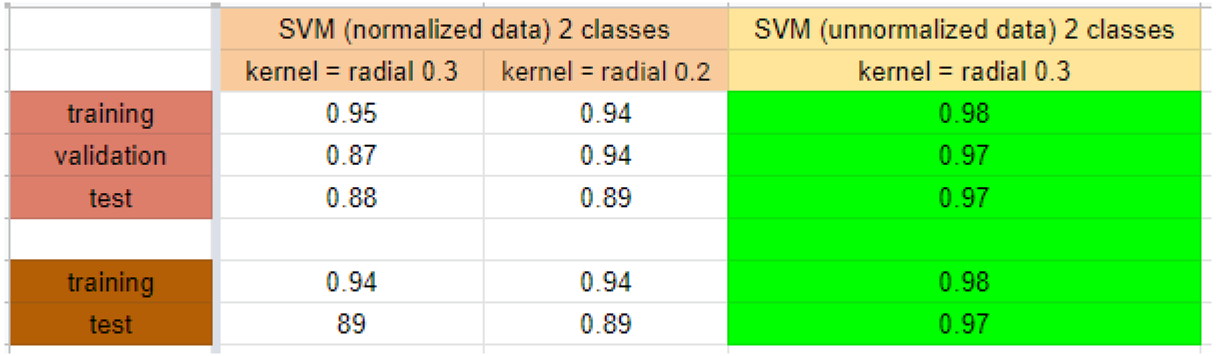


	SVM (normalized data) 2 classes		SVM (unnormalized data) 2 classes
	kernel = radial 0.3	kernel = radial 0.2	kernel = radial 0.3
training	95	95	98
validation	89	90	97
test	89	90	97
training	95	95	98
test	90	90	97

Results: SVM (f1-score)

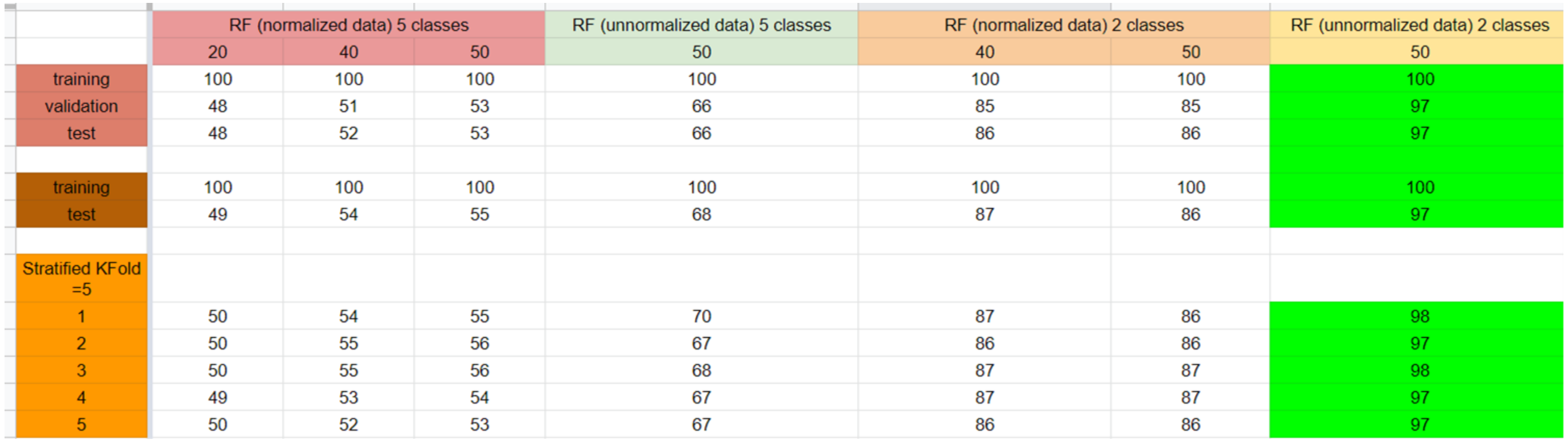


	SVM (normalized data) 5 classes			SVM (unnormalized data) 5 classes
	kernel = radial	kernel = linear	kernel = poly	kernel = radial 0.3
training	0.81	0.27	0.81	0.55
validation	0.58	0.21	0.39	0.51
test	0.57	0.23	0.38	0.51
training	0.81	0.26	0.78	0.56
test	0.6	0.23	0.4	0.54



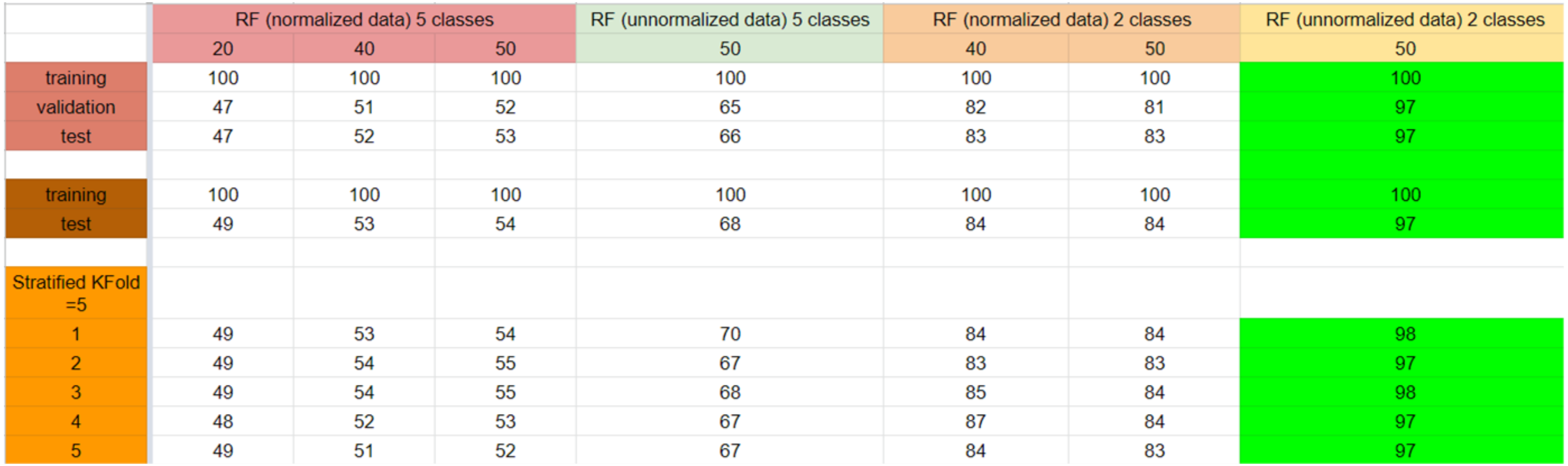
	SVM (normalized data) 2 classes		SVM (unnormalized data) 2 classes
	kernel = radial 0.3	kernel = radial 0.2	kernel = radial 0.3
training	0.95	0.94	0.98
validation	0.87	0.94	0.97
test	0.88	0.89	0.97
training	0.94	0.94	0.98
test	89	0.89	0.97

Results: RF (accuracy)



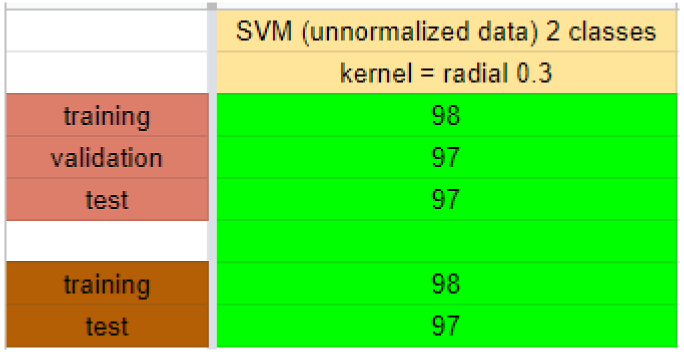
	RF (normalized data) 5 classes			RF (unnormalized data) 5 classes	RF (normalized data) 2 classes		RF (unnormalized data) 2 classes
	20	40	50	50	40	50	50
training	100	100	100	100	100	100	100
validation	48	51	53	66	85	85	97
test	48	52	53	66	86	86	97
training	100	100	100	100	100	100	100
test	49	54	55	68	87	86	97
Stratified KFold =5							
1	50	54	55	70	87	86	98
2	50	55	56	67	86	86	97
3	50	55	56	68	87	87	98
4	49	53	54	67	87	87	97
5	50	52	53	67	86	86	97

Results: RF (F1-Score)

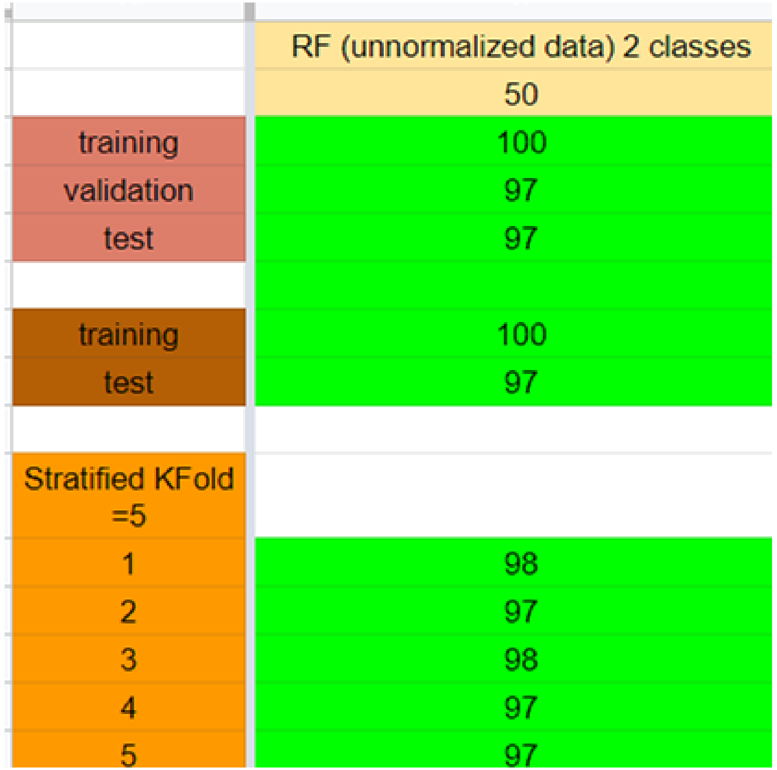


	RF (normalized data) 5 classes			RF (unnormalized data) 5 classes	RF (normalized data) 2 classes		RF (unnormalized data) 2 classes
	20	40	50	50	40	50	50
training	100	100	100	100	100	100	100
validation	47	51	52	65	82	81	97
test	47	52	53	66	83	83	97
training	100	100	100	100	100	100	100
test	49	53	54	68	84	84	97
Stratified KFold =5							
1	49	53	54	70	84	84	98
2	49	54	55	67	83	83	97
3	49	54	55	68	85	84	98
4	48	52	53	67	87	84	97
5	49	51	52	67	84	83	97

Results: Evaluation SVM & RF in terms of Accuracy

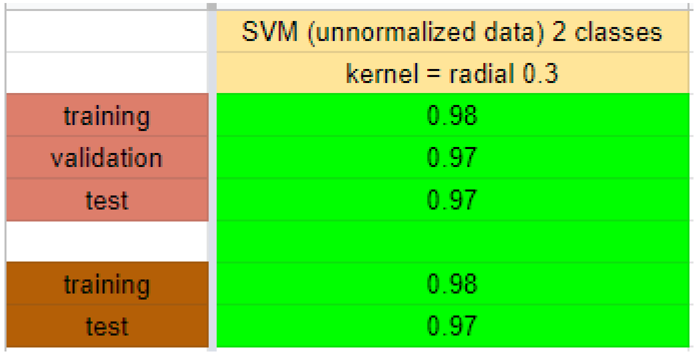


	SVM (unnormalized data) 2 classes
	kernel = radial 0.3
training	98
validation	97
test	97
training	98
test	97

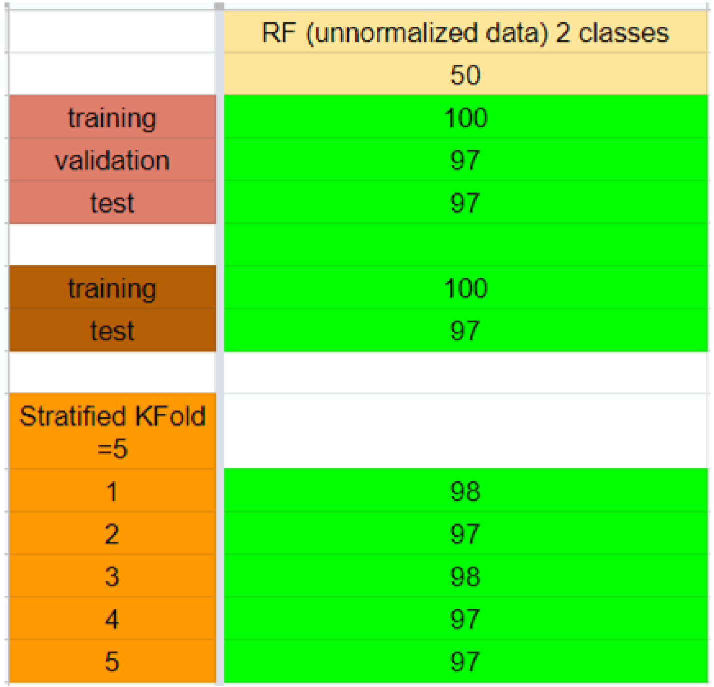


	RF (unnormalized data) 2 classes
	50
training	100
validation	97
test	97
training	100
test	97
Stratified KFold =5	
1	98
2	97
3	98
4	97
5	97

Results: Evaluation SVM & RF in terms of F1-Score



	SVM (unnormalized data) 2 classes
	kernel = radial 0.3
training	0.98
validation	0.97
test	0.97
training	0.98
test	0.97



	RF (unnormalized data) 2 classes
	50
training	100
validation	97
test	97
training	100
test	97
Stratified KFold =5	
1	98
2	97
3	98
4	97
5	97

Discussion

- The results show that when dealing with **un-normalized data** both SVM and **RF** have nearly the same accuracy on test data but with a notable difference in accuracy on training data.
- However, when dealing with **normalized data**,the **SVM model** has a higher accuracy and F1-score than RF
- This shows that RF and SVM can both be used for this problem domain. However, it’s recommended to take the normalization into consideration.

Conclusion

- Both proposed models performed better than DNN in the F1 score proving that **DNN is not the ideal model** for this problem domain.
- Moreover, when dealing with the problem as 2 classes like [6] & [7], our model reached nearly the same accuracy and F1 score.
- In conclusion,**SVM & RF** can be considered as the **ideal models** for this problem domain regardless of number of class used, especially that they provide **higher interpretability** than the others.

Workload Distribution

- The preprocessing was done by both Mayar & Sadany together.
- Sadany was responsible for SVM Model and its optimization while Mayar was responsible for the Random forest model and its optimization.
- Both models were then evaluated using the F1-score.
- We both worked on models analysis to choose the best model.

Thank you

Any question?

References

[1]"Epilepsy - Symptoms and causes", Mayo Clinic, 2020. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/epilepsy/symptoms-causes/syc-20350093.

[2]"Seizures - Symptoms and causes", *Mayo Clinic,* 2020. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/seizure/symptoms-causes/syc-20365711#:~:text=A%20seizure%20is%20a%20sudden,seizures%2C%20which%20range%20in%20severity.

[3]S. Usman, S. Khalid, R. Akhtar, Z. Bortolotto, Z. Bashir and H. Qiu, "Using scalp EEG and intracranial EEG signals for predicting epileptic seizures: Review of available methodologies", Seizure, vol. 71, pp. 258-269, 2019. Available: 10.1016/j.seizure.2019.08.006

[4]R. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David and C. Elger, "Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state", *Physical Review E,* vol. 64, no. 6, 2001. Available: 10.1103/physreve.64.06190

References

[5] A. Guha, S. Ghosh, A. Roy and S. Chatterjee, "Epileptic Seizure Recognition Using Deep Neural Network", *Advances in Intelligent Systems and Computing,* pp. 21-28, 2019. Available: 10.1007/978-981-13-7403-6_3

[6]K. Polat and S. Güneş, "Classification of epileptiform EEG using a hybrid system based on decision tree classifier and fast Fourier transform", *Applied Mathematics and Computation*, vol. 187, no. 2, pp. 1017-1026, 2007. Available: 10.1016/j.amc.2006.09.022

[7]A. Subasi and M. Ismail Gursoy, "EEG signal classification using PCA, ICA, LDA and support vector machines", *Expert Systems with Applications*, vol. 37, no. 12, pp. 8659-8666, 2010. Available: 10.1016/j.eswa.2010.06.065