`

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Applied Machine Learning

(8900-10-R)

Date: August,9 2019

"Feature Selection_09"

(FS-09)

Instructor:

Dr. Roozbeh Razavi Far

Submitted by:

Karan Kalugade

Vedant Dave

University of Windsor

ON, CANADA

ABSTRACT:

Artificial Intelligence is the most discussed topic nowadays. Its implementation or applications are being discussed and are being researched most. Machine learning is one of the subsets of artificial intelligence. Machine learning (ML) tries to understand algorithms and mathematical models used in computer systems by learning from patterns. It tries to develop a structure from training data to decided on future inputs for a specific purpose. Machine learning application can be seen in many fields but mainly in data science and computer vision. Machine learning is also being used in making predictions and data analysis. Machine learning has many learning approaches such as supervised, semi-supervised, unsupervised, active, reinforcement and meta learning. Machine learning has similarities with Data Mining but varies in its focus on making predictions rather than find unknown data elements as seen in data mining. Machine learning has a primary goal of optimization and to generalize the learning algorithm based on its training to get possible result on test data. The goal of this project is feature selection based on various feature selection techniques and classification methods. The machine learning process starts with learning task or problem. After that, the dataset under consideration is used for training, target function is defined, representation methodologies and learning algorithm need to be formulated. The project discussed ahead focuses on feature selection methods: Unsupervised Discriminative Feature Selection [UDFS], Local Learning-based Clustering Feature Selection[LLCFS], Correlation-based Feature Selection[CFS] and classification methods: Naive Bayes Classifier[NBC], Support Vector Machines[SVM], Decision Trees[DT], Recurrent Neural Networks[RNN]. The project tries to get the best possible accuracy and f1score for given datasets.

Keywords

Classifier, Feature Selection, Parameter Tuning, Decision Tree, Support Vector machines, Hyper Parameter, Recurrent Neural Network, Naïve Bayes classifier, Machine Learning, F1 Score, confusion Matrix, MATLAB, pu=ython

Development Software

MATLAB R2018b & MATLAB R2014a

Jupiter Notebook (IBM, Cognitive Environment)

INTRODUCTION

Machine learning studies patterns from datasets to get best prediction. It generally used when dealing with big data sets and lots of variables where there was no existing model. Machine learning has two main approaches such as supervised and unsupervised learning. Unsupervised learning uses clustering to differentiate between data values as it does not have labels in its dataset. Supervised learning uses two techniques such as classification and regression. Classification simply classifies given data to give discrete responses. Regression continuously makes predictions.

Feature selection is also known as variable or attribute selection is in demand to the due formation of huge datasets and better machine learning techniques. Feature selection focuses on selecting a method automatic in nature for best possible attribute selection aligned of selecting relevant to data and predictive modeling. The benefits of feature selection are that it reduces overfitting and training time while increasing the accuracy of the selected training model.

Feature Selection Methods:

1. **Unsupervised Discriminative Feature Selection [UDFS]**
   It is useful for feature selection in data without labels. UDFS algorithm selects the most discriminative feature subset from the whole feature set in batch mode. The criterion in UDFS is to select the features which best preserve the data similarity or manifold structure derived from the whole feature set. The UDFS algorithm aims to simultaneously exploiting discriminative information and feature correlations. Use one-step approach to select the most discriminative approach for data representation. UDFS analyzes features jointly and local structure of data distribution.

2. **Local Learning-based Clustering Feature Selection [LLCFS]**
   The key idea is to break an arbitrarily complex nonlinear problem into a set of locally linear sets through local learning method. Leading to learn feature relevance globally within the large margin framework. No need for assumptions because of real data distribution. This Feature Selection model has a logarithmic sample complexity with respect to a number of features.

3. **Correlation-based Feature Selection [CFS]**
   the correlation-based approach towards hypothesis focused on good feature sets containing features that are highly correlated with the class, yet is uncorrelated with each other. CFS is an algorithm using a feature evaluation formula based on ideas from test theory that provides an operational definition of this hypothesis with an

appropriate correlation measure and a heuristic search strategy. CFS executes faster than other wrapper algorithms. Merit of feature subset

$$\mathrm{Merit}_{S_k} = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}}.$$

CFS Optimization Problem:

$$\mathrm{CFS} = \max_{S_k}\left[\frac{r_{cf_1} + r_{cf_2} + \cdots + r_{cf_k}}{\sqrt{k + 2(r_{f_1f_2} + \cdots + r_{f_if_j} + \cdots + r_{f_kf_1})}}\right].$$

Where k: features;

r_{cf}: the average value of all feature-classification correlations

r_{ff}: the average value of all feature-feature correlations

xi be the set membership indicator function for feature fi

Classification Methods:

a. Naive Bayes Classifier [NBC]

Classifiers are used in machine learning to differentiate objects based on their feature values. The Naive Bayes Classifier is based on Bayes theorem with the assumption that features selected are independent. There is three main types of NBC's: Multinomial Naive Bayes, Bernoulli Naive Bayes, Gaussian Naive Bayes.

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x} \mid C_k)}{p(\mathbf{x})} \quad \text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$$

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}}\ p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k).$$

b. Support Vector Machines [SVM]

Support Vector Machines are non-probabilistic binary linear classifiers and are supervised learning models. It works by dividing by a well-defined wide gap points in space with separate categories.

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0,1-y_i(w\cdot x_i-b)\right)\right]+\lambda\|w\|^2.$$

c. Decision Trees [DT]

Decision Trees are used to display an algorithm with condition control statements and are decision support tools for best possible outcomes with a tree-like model of decisions. It has a structured algorithm with an easy understanding of data classification flow shows the possible consequences. Uses a predictive model on observations of elements to find target elements value

d. Recurrent Neural Networks [RNN]

RNN's use of Connectionist Temporal Classification end-to-end training methods for unknown input-output alignment. In this artificial neural network uses directed graphs for connection between nodes to align as a temporal sequence. RNN's have internal state memory to process sequence of inputs. RNN use the internal state consequences to solve the complex problem, unlike to real neural network it has no forward or backward calling Used for integrating the context of the input feature vector for classification.

Implementation Procedure:

- For this project we use the **MATLAB** software with two different version **R2018b** and **R2014a** for feature selection, MATLAB's inbuilt library of feature selection give the ease for getting the feature ranking. And for the further process on the data and classifier, we will use the Python language on **Jupyter Notebook on the IBM cognitive Environment**.
- Before using the data in the MATLAB, we need to clean it, The first observation of the given dataset of Gear family contains a various pattern of rows and columns. First, We need to clean our data by removing the zeros from the dataset. Because that zero shows the missing data fill with zero value to create the regular matrix to procced the data.
- After removing the zeros, we can normalize the data for better dimension reduction, but sometimes that data converts to float value and some function of python can not handle too many float64 values. We need to use the integer conversion of those data.
- Another method to reduce the column dimension is highly correlated feature mapping and then we need to remove highly correlated data more than 90 percent. Because every high correlated data has the same effect on the result, by removing them we can not lose important information.
- The next step in pipe line is to get the feature ranking based on the feature ranking. And then we need to use that data ranking to get the highest accuracy with different classification method.
- According to ranking our main priority is to generate the accuracy and f1_score to evaluate the matrix. The no of a feature on which we get maximum accuracy will give us the feature label for our test dataset.
- The above Procedure gives us the idea about the training of our classifier model with the best accuracy with minimum information loss. But to check our procedure we need to implement the validation dataset.
- The separation of a dataset in train, validation, the test is 80,20,20 percent accordingly. After generating the validation set, use the Nested validation loop to get validation score accuracy and f1 score between (train, test) and (train, validation) sets.
- In all classifiers (SVM, Decision Tree, RNN, and NBC) we use randomization and shuffling of sets in 9:1 ratio. For better value of randomization, we use K-Folds for better prediction.
- The comparison of the accuracy before and after the feature selection give us the improvement in our model accuracy and give us information about the best method of the feature selection.
- By Plotting the confusion matrix, we get the data visualization of our model prediction of true and false value identify by particulate model.

❖ Key MATLAB and 'python' files used for Program:

We use the following code files for getting our result. These files are generated by the use of MATLAB feature selection library and open-source environment of python based machine learning.

1. FS_09_Data_Cleaning.py: It has code to separate the main 'Gear-dataset' into X and Y, clean them to reduce the feature dimension.

2. Feature_selection.m: It's a modified MATLAB file of demo relate with feature selection library. In a modification, we fit the file for multiclass data. For fit the binary SVM we use ECOC function to use it for multiclass classification.

3. Max_Accuracy_features_F.py: This file contains the code to decide the max features require for higher accuracy. And give us the feature ranking require for classification f1 score and accuracy. With the use of the Kfold and validation loop in the comment section, we can get the accuracy for 10 fold between the (train, test) and (train, validation) set.

4. SVM_F.py: Python code for SVM classifier with data visualization of the confusion matrix.

5.DT_F.py: Python code for DT classifier with data visualization of the confusion matrix.

6. NBC_F.py: Python code for NBC classifier with data visualization of the confusion matrix

7. RNN_F.py: Python code for RNN classifier with data visualization of the confusion matrix.

8. Decision_Tree.m: MATLAB code for the Decision tree classifier with feature selection using K fold.

9. Support_Vector_Machine.m: MATLAB code for the SVM classifier.

10. Naïve Bayes.m: MATLAB code for the Naïve Bayes classifier.

11. DT Classifier.m: MATLAB code for the Decision tree classifier with Kfold validation score

Key Parameters for tuning the Classifier:

1)

- **Support Vector Machines:** SVC(degree=3, gamma='scale', kernel='rbf', max_iter=-1)

SVM's **kernel** shows which type of function we use and according to that our model decide it to linear or nonlinear. **Degree** gives information for the nonlinear degree of the function, and **max iteration** gives info about spoke for higher accurate model.

2)

```
decision_tree = DecisionTreeClassifier(random_state=0, max_depth=2, criterian='gini')
```

max depth give separation of data, more depth give increment in the accuracy up to some strength, the **criterion =** give us option that on which base we need to split the data, there are two options such as Gini and entropy

3)

Naïve Bayes Classifier:

In our case, we do not use any tuning parameter. But on the Gaussian base it gives the output which is very low compared to other.

4)

Recurrent Neural Network:

This coding use keras library, and other step wise model sequence like Sequential, **add** , **drop**, **dense modeling** and **compile modules** codes. For tuning we must need to **manage the long-term memory** and proper **sequencing** to generate high accurate output.

Beside all of these our output also depend up on the features remain after cleaning, the outliers in the dataset and random K-fold choose by the programs.

Algorithm Steps:

1) UDFS (Unsupervised Discriminative Feature Selection.)

for i = 1 to n do

```
Bi = ( ˜XTi˜Xi + λI)−1
Mi = SiHk+1BiHk+1STi ;
M = X_ni=1MiXT ;
Set t = 0 and initialize D0 ∈ Rd×d as an identity matrix;
Repeat
Pt = M + γDt;
Wt = [p1, ..., pc] where p1, ..., pc are the eigenvectors of Pt corresponding to the first c
smallest eigenvalues;
Update the diagonal matrix Dt+1 asDt+1 =⎡⎢⎣12_w1t _2...12_wdt_2⎤⎥⎦;
t = t + 1;
until Convergence;
Sort each feature fi |di=1 according to__wit__2indescending order and select the top
ranked ones.
```

2) LLCFS (Feature selection for local learning-based clustering algorithm.)

input: X ¼fxign i¼1, size of the neighborhood k, trade-off
parameter (Bita)
output: Y; _

```
Initialize _l ¼ 1d , for l ¼ 1; . . . ; d;
while not converge do
Find k-mutual neighbors for fxign i¼1, using the metricdefined);
Construct the matrix M in  with _i given ,
 then solve the problem to obtain Y;
Compute wc    i ; 8i; c  and update it for next epoch);
End
```

3) CFS (Correlation Feature Selection)

A correlation-based approach towards hypothesis focused on good feature sets containing features that are high correlated with the class, yet being uncorrelated with each other

CFS is an algorithm using a feature evaluation formula based on ideas from test theory that provides an operational definition of this hypothesis with an appropriate correlation measure and a heuristic search strategy CFS executes faster than other wrapper algorithms Merit of feature subset

On the base of following merits it find the correlation between the set and returns the feature which highly correlate with the dataset.

$$\mathrm{Merit}_{S_k} = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}}.$$

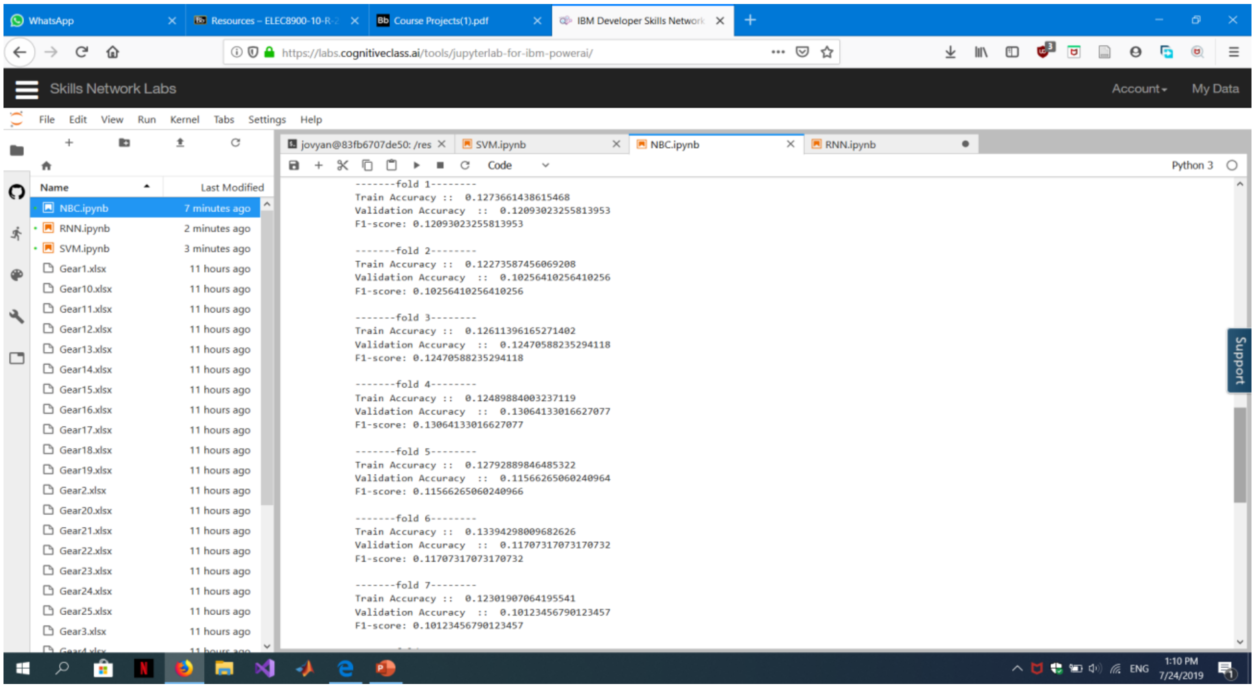
❖ Common Algorithm of Classification:

Steps:

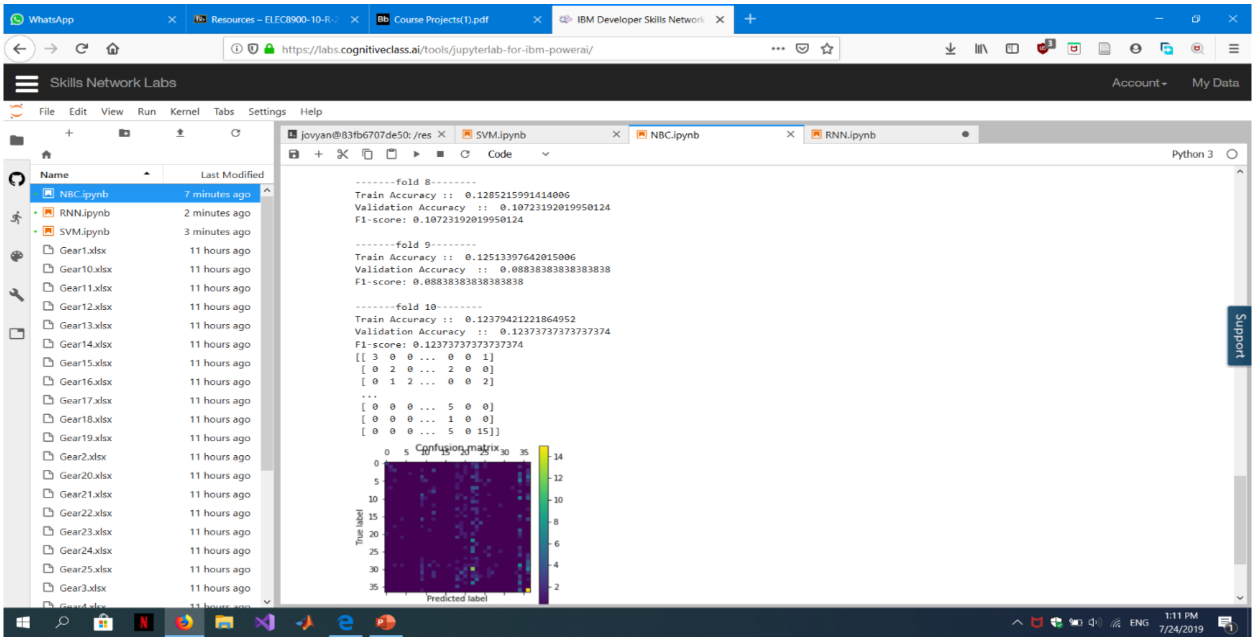
1) Separate the dataset in X and Y
2) Split the data set in train: test: validation = 80:20:20
3) Call the library require according to classifier (SVM,DT,RNN,NBS)
4) Choose the training parameter according to code requirement;
5) Fit the data in the model you named;
6) Predict the yhat prediction score for output based on $X_{test;}$
7) Determine the accuracy and the f1 score between the (train, test) and (train,validation) set for K-folds.

❖ Results:

Accuracy and F1 Score with NBC classifier:

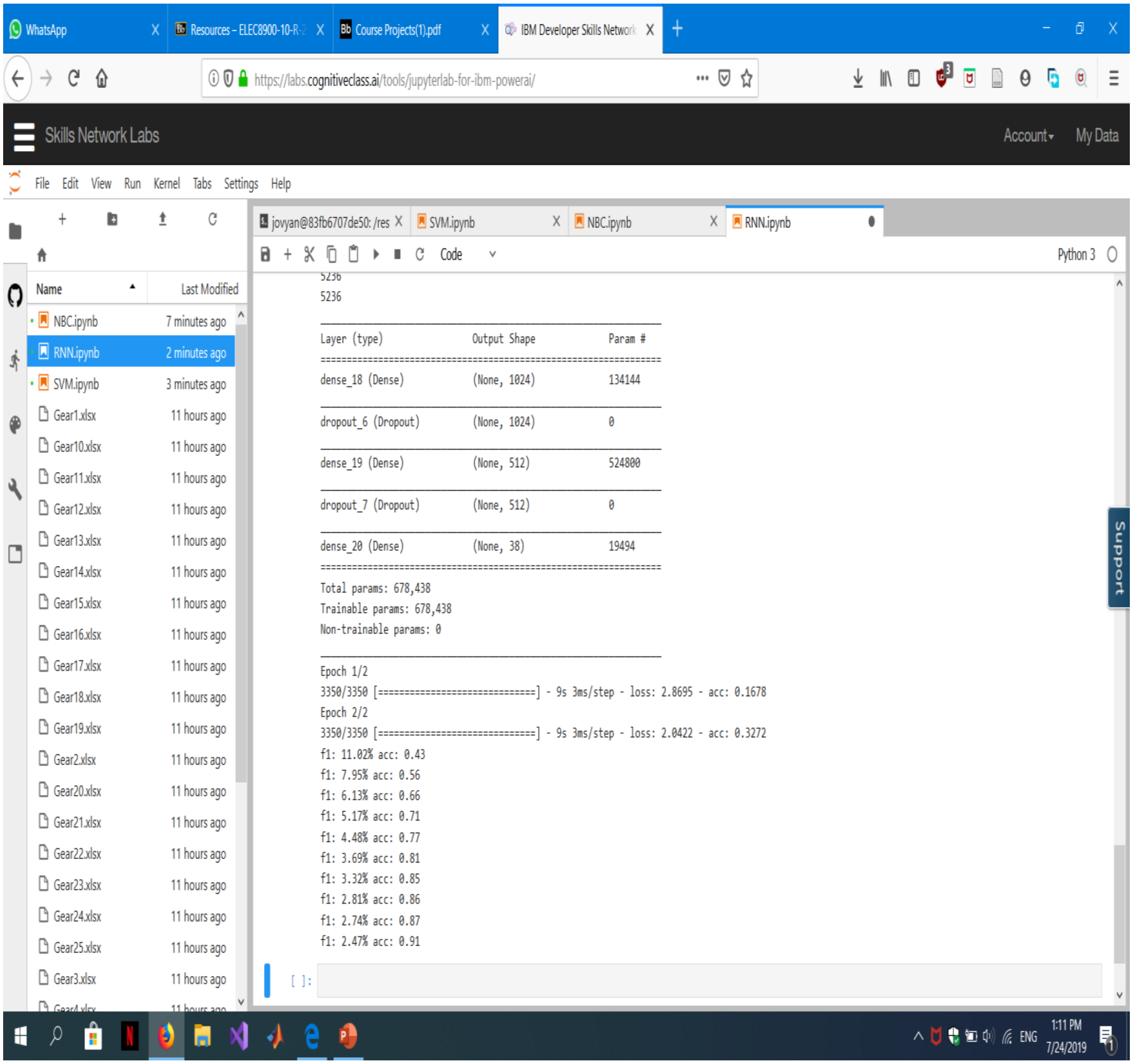


```
-------fold 1--------
Train Accuracy ::  0.1273661438615468
Validation Accuracy  ::  0.12093023255813953
F1-score: 0.12093023255813953

-------fold 2--------
Train Accuracy ::  0.12273587456069208
Validation Accuracy  ::  0.10256410256410256
F1-score: 0.10256410256410256

-------fold 3--------
Train Accuracy ::  0.12611396165271402
Validation Accuracy  ::  0.12470588235294118
F1-score: 0.12470588235294118

-------fold 4--------
Train Accuracy ::  0.12489884003237119
Validation Accuracy  ::  0.13064133016627077
F1-score: 0.13064133016627077

-------fold 5--------
Train Accuracy ::  0.12792889846485322
Validation Accuracy  ::  0.11566265060240964
F1-score: 0.11566265060240966

-------fold 6--------
Train Accuracy ::  0.13394298009682626
Validation Accuracy  ::  0.11707317073170732
F1-score: 0.11707317073170732

-------fold 7--------
Train Accuracy ::  0.12301907064195541
Validation Accuracy  ::  0.10123456790123457
F1-score: 0.10123456790123457
```



```
-------fold 8--------
Train Accuracy ::  0.1285215991414006
Validation Accuracy  ::  0.10723192019950124
F1-score: 0.10723192019950124

-------fold 9--------
Train Accuracy ::  0.12513397642015006
Validation Accuracy  ::  0.08838383838383838
F1-score: 0.08838383838383838

-------fold 10--------
Train Accuracy ::  0.12379421221864952
Validation Accuracy  ::  0.12373737373737374
F1-score: 0.12373737373737374
[[ 3  0  0 ...  0  0  1]
 [ 0  2  0 ...  2  0  0]
 [ 0  1  2 ...  0  0  2]
 ...
 [ 0  0  0 ...  5  0  0]
 [ 0  0  0 ...  1  0  0]
 [ 0  0  0 ...  5  0 15]]
```

❖ Accuracy and F1 Score with RNN Classifier:

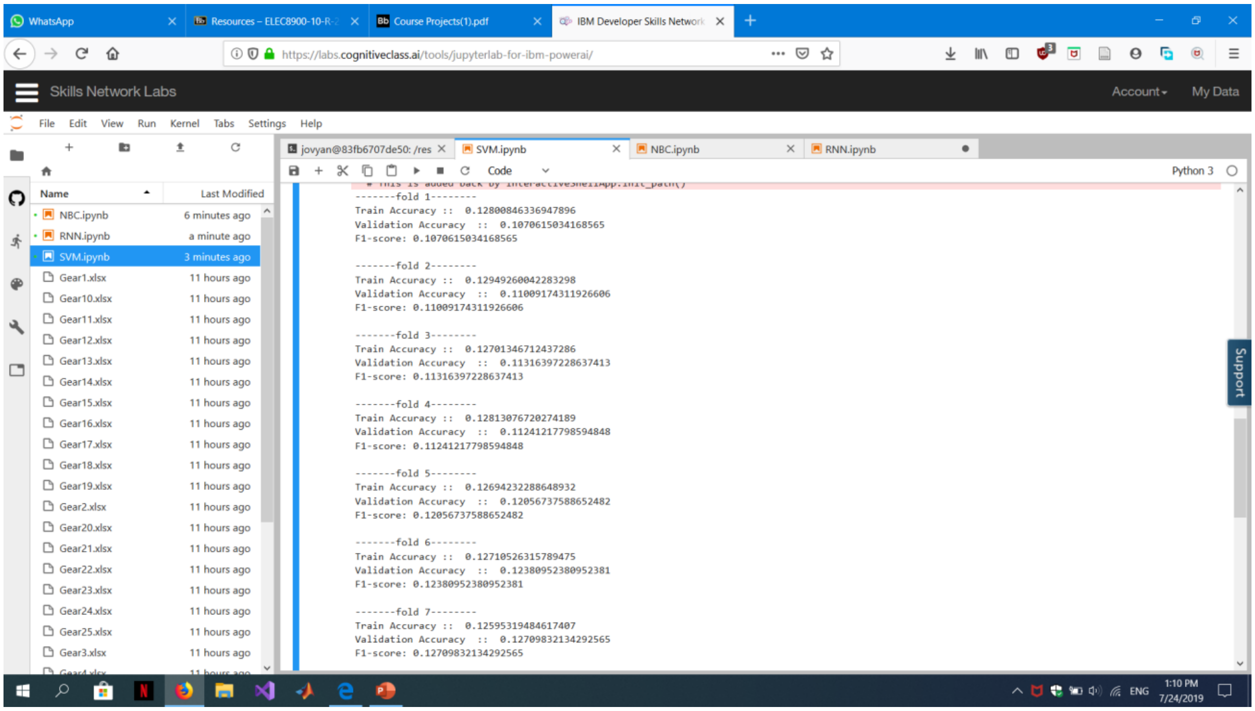


Name	Last Modified
NBC.ipynb	7 minutes ago
RNN.ipynb	2 minutes ago
SVM.ipynb	3 minutes ago
Gear1.xlsx	11 hours ago
Gear10.xlsx	11 hours ago
Gear11.xlsx	11 hours ago
Gear12.xlsx	11 hours ago
Gear13.xlsx	11 hours ago
Gear14.xlsx	11 hours ago
Gear15.xlsx	11 hours ago
Gear16.xlsx	11 hours ago
Gear17.xlsx	11 hours ago
Gear18.xlsx	11 hours ago
Gear19.xlsx	11 hours ago
Gear2.xlsx	11 hours ago
Gear20.xlsx	11 hours ago
Gear21.xlsx	11 hours ago
Gear22.xlsx	11 hours ago
Gear23.xlsx	11 hours ago
Gear24.xlsx	11 hours ago
Gear25.xlsx	11 hours ago
Gear3.xlsx	11 hours ago
Gear4.xlsx	11 hours ago

```
5236
5236
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_18 (Dense)             (None, 1024)              134144
_________________________________________________________________
dropout_6 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_19 (Dense)             (None, 512)               524800
_________________________________________________________________
dropout_7 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_20 (Dense)             (None, 38)                19494
=================================================================
Total params: 678,438
Trainable params: 678,438
Non-trainable params: 0
_________________________________________________________________
Epoch 1/2
3350/3350 [==============================] - 9s 3ms/step - loss: 2.8695 - acc: 0.1678
Epoch 2/2
3350/3350 [==============================] - 9s 3ms/step - loss: 2.0422 - acc: 0.3272
f1: 11.02% acc: 0.43
f1: 7.95% acc: 0.56
f1: 6.13% acc: 0.66
f1: 5.17% acc: 0.71
f1: 4.48% acc: 0.77
f1: 3.69% acc: 0.81
f1: 3.32% acc: 0.85
f1: 2.81% acc: 0.86
f1: 2.74% acc: 0.87
f1: 2.47% acc: 0.91

[ ]:
```

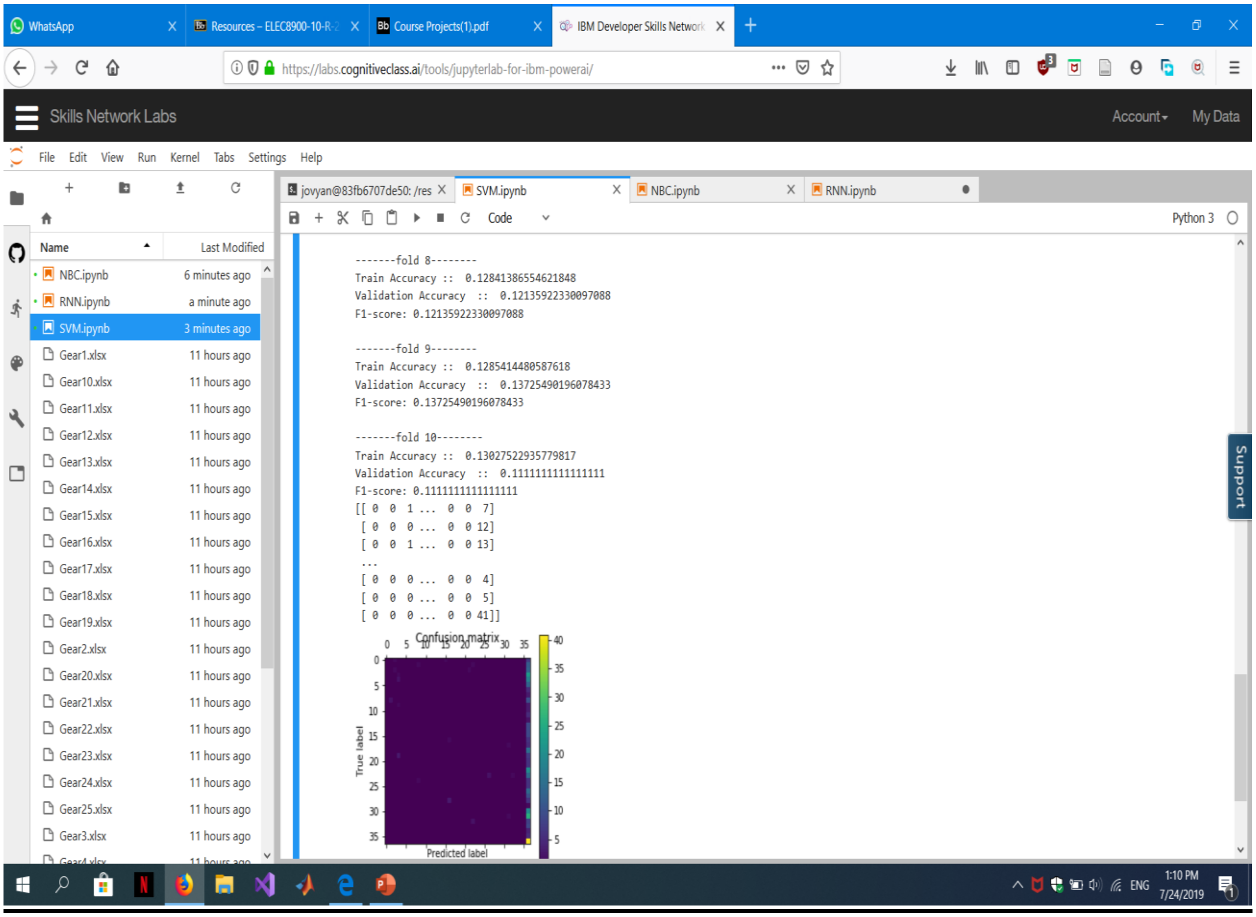
In the data set, RNN use the fold =10 and every time it increase the accuracy because RNN is sequential classifier and every time it runs add, drop and compile operations.

❖ Accuracy and F1 Score based on the SVM Classifier:



Name	Last Modified
NBC.ipynb	6 minutes ago
RNN.ipynb	a minute ago
SVM.ipynb	3 minutes ago
Gear1.xlsx	11 hours ago
Gear10.xlsx	11 hours ago
Gear11.xlsx	11 hours ago
Gear12.xlsx	11 hours ago
Gear13.xlsx	11 hours ago
Gear14.xlsx	11 hours ago
Gear15.xlsx	11 hours ago
Gear16.xlsx	11 hours ago
Gear17.xlsx	11 hours ago
Gear18.xlsx	11 hours ago
Gear19.xlsx	11 hours ago
Gear2.xlsx	11 hours ago
Gear20.xlsx	11 hours ago
Gear21.xlsx	11 hours ago
Gear22.xlsx	11 hours ago
Gear23.xlsx	11 hours ago
Gear24.xlsx	11 hours ago
Gear25.xlsx	11 hours ago
Gear3.xlsx	11 hours ago
Gear4.xlsx	11 hours ago

```
  # This is added back by InteractiveShellApp.init_path()
-------fold 1--------
Train Accuracy ::  0.12800846336947896
Validation Accuracy  ::  0.1070615034168565
F1-score: 0.1070615034168565

-------fold 2--------
Train Accuracy ::  0.12949260042283298
Validation Accuracy  ::  0.11009174311926606
F1-score: 0.11009174311926606

-------fold 3--------
Train Accuracy ::  0.12701346712437286
Validation Accuracy  ::  0.11316397228637413
F1-score: 0.11316397228637413

-------fold 4--------
Train Accuracy ::  0.12813076720274189
Validation Accuracy  ::  0.11241217798594848
F1-score: 0.11241217798594848

-------fold 5--------
Train Accuracy ::  0.12694232288648932
Validation Accuracy  ::  0.12056737588652482
F1-score: 0.12056737588652482

-------fold 6--------
Train Accuracy ::  0.12710526315789475
Validation Accuracy  ::  0.12380952380952381
F1-score: 0.12380952380952381

-------fold 7--------
Train Accuracy ::  0.12595319484617407
Validation Accuracy  ::  0.12709832134292565
F1-score: 0.12709832134292565
```



Name	Last Modified
NBC.ipynb	6 minutes ago
RNN.ipynb	a minute ago
SVM.ipynb	3 minutes ago
Gear1.xlsx	11 hours ago
Gear10.xlsx	11 hours ago
Gear11.xlsx	11 hours ago
Gear12.xlsx	11 hours ago
Gear13.xlsx	11 hours ago
Gear14.xlsx	11 hours ago
Gear15.xlsx	11 hours ago
Gear16.xlsx	11 hours ago
Gear17.xlsx	11 hours ago
Gear18.xlsx	11 hours ago
Gear19.xlsx	11 hours ago
Gear2.xlsx	11 hours ago
Gear20.xlsx	11 hours ago
Gear21.xlsx	11 hours ago
Gear22.xlsx	11 hours ago
Gear23.xlsx	11 hours ago
Gear24.xlsx	11 hours ago
Gear25.xlsx	11 hours ago
Gear3.xlsx	11 hours ago
Gear4.xlsx	11 hours ago

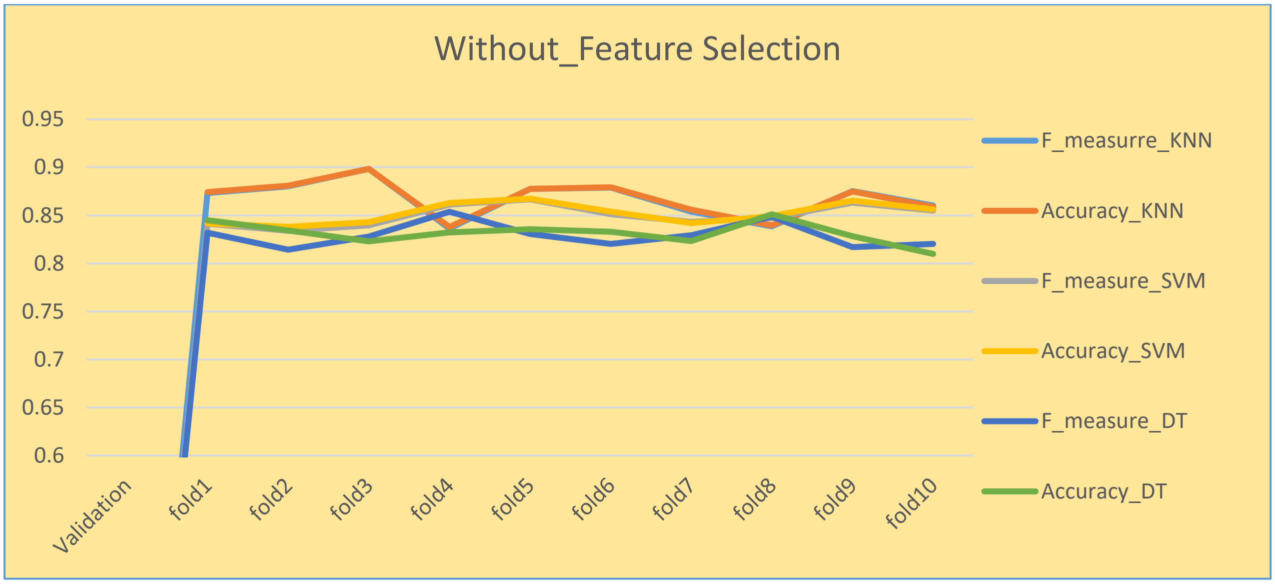
```
-------fold 8--------
Train Accuracy ::  0.12841386554621848
Validation Accuracy  ::  0.12135922330097088
F1-score: 0.12135922330097088

-------fold 9--------
Train Accuracy ::  0.1285414480587618
Validation Accuracy  ::  0.13725490196078433
F1-score: 0.13725490196078433

-------fold 10--------
Train Accuracy ::  0.13027522935779817
Validation Accuracy  ::  0.1111111111111111
F1-score: 0.1111111111111111
[[ 0  0  1 ...  0  0  7]
 [ 0  0  0 ...  0  0 12]
 [ 0  0  1 ...  0  0 13]
 ...
 [ 0  0  0 ...  0  0  4]
 [ 0  0  0 ...  0  0  5]
 [ 0  0  0 ...  0  0 41]]
```

❖ Detailed Analysis with specific Example:

Here, we use the **Gear 7** dataset to present the main idea behind the project of train the Machine learning model with different feature selection and classifier. Our main motto is to select the best matching model with the use of Accuracy and F1 score evaluation.

First we analyze the Evaluation for the Raw data without selecting any process:

Validation	KNN		SVM		DT	
fold1	**0.873285**	**0.874307**	**0.841112**	**0.841379**	**0.831952**	**0.844732**
fold2	**0.880236**	**0.880819**	**0.833891**	**0.837963**	**0.81435**	**0.834264**
fold3	**0.898334**	**0.898305**	**0.839603**	**0.842723**	**0.827741**	**0.822976**
fold4	**0.836096**	**0.837786**	**0.861043**	**0.862884**	**0.853691**	**0.832061**
fold5	**0.877416**	**0.877629**	**0.866605**	**0.867299**	**0.83064**	**0.835564**
fold6	**0.878937**	**0.879079**	**0.851165**	**0.853717**	**0.820333**	**0.833013**
fold7	**0.853732**	**0.855769**	**0.843129**	**0.841849**	**0.829164**	**0.823077**
fold8	**0.83869**	**0.839458**	**0.847334**	**0.849148**	**0.848359**	**0.851064**
fold9	**0.875249**	**0.875**	**0.863165**	**0.865196**	**0.816909**	**0.828125**
fold10	**0.860161**	**0.858824**	**0.85477**	**0.856079**	**0.820231**	**0.809804**

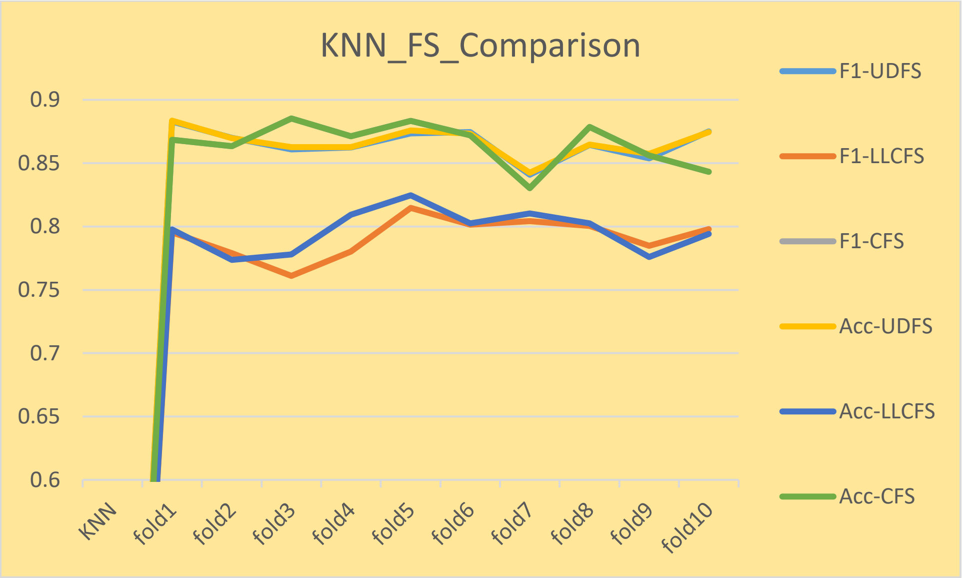


Without Feature Selection Raw Data analysis with all Classifier

Now we move to the individual classifier with Evaluation Parameters:

K- Nearest Neighbor Classifier (Clustering)

Validation	F1-Score			Accuracy		
KNN	UDFS	LLCFS	CFS	UDFS	LLCFS	CFS
fold1	0.882851	0.795428	0.868536	0.883549	0.797701	0.868536
fold2	0.869891	0.778942	0.863287	0.869646	0.773672	0.863287
fold3	0.86091	0.760893	0.885278	0.862524	0.778037	0.885278
fold4	0.862421	0.78034	0.871129	0.862595	0.809412	0.871129
fold5	0.87362	0.814663	0.883402	0.875717	0.824645	0.883402
fold6	0.874573	0.801407	0.872039	0.873321	0.80241	0.872039
fold7	0.840924	0.804283	0.830338	0.842308	0.810219	0.830338
fold8	0.864542	0.800505	0.87868	0.864603	0.802439	0.87868
fold9	0.853759	0.784662	0.856154	0.857422	0.775862	0.856154
fold10	0.874967	0.797829	0.843092	0.87451	0.794045	0.843092

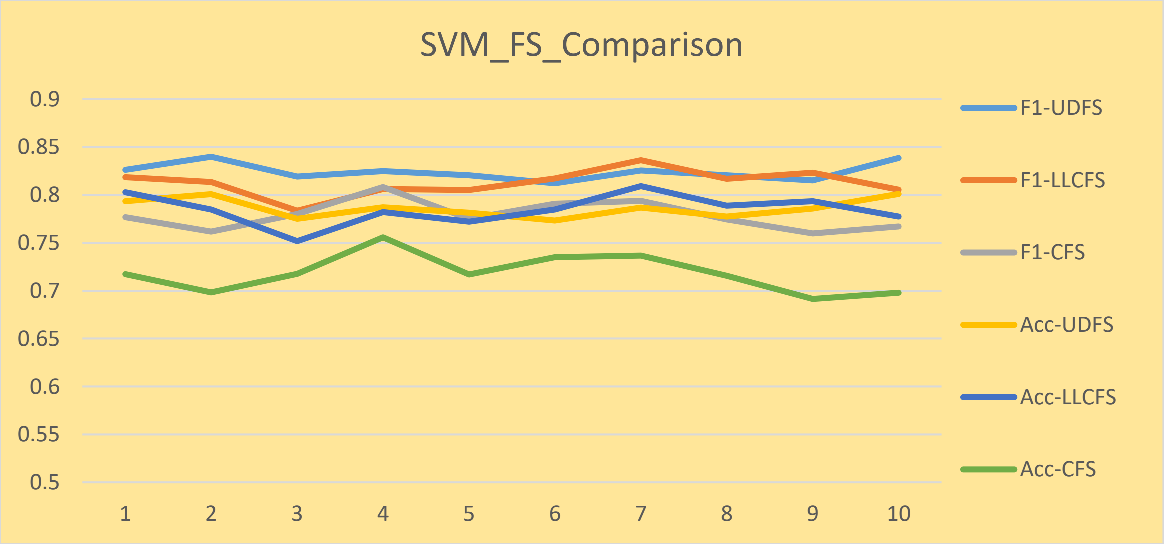


Analysis:

The Average Value of winning Accuracy value is for the UDFS with KNN classifier, which is 0.8666. With Acceptable F1- Score.

Support Vector Machine Classifier: (Linear /Logistic/ Nonlinear)

Validation	F1-Score			Accuracy		
SVM	UDFS	LLCFS	CFS	UDFS	LLCFS	CFS
fold1	0.82615	0.818454	0.77674	0.793578	0.802752	0.71719
fold2	0.839825	0.813331	0.761858	0.800926	0.784722	0.698324
fold3	0.819042	0.783515	0.780154	0.775176	0.75174	0.717514
fold4	0.824993	0.806145	0.80821	0.787234	0.782201	0.755725
fold5	0.820418	0.805295	0.775034	0.781473	0.771971	0.717017
fold6	0.812146	0.817312	0.790875	0.77327	0.784689	0.735125
fold7	0.825676	0.836236	0.793769	0.786925	0.809179	0.736538
fold8	0.820508	0.816992	0.774522	0.777506	0.788698	0.715667
fold9	0.815026	0.823274	0.759793	0.785714	0.793532	0.691406
fold10	0.838667	0.805347	0.766972	0.800995	0.7775	0.698039

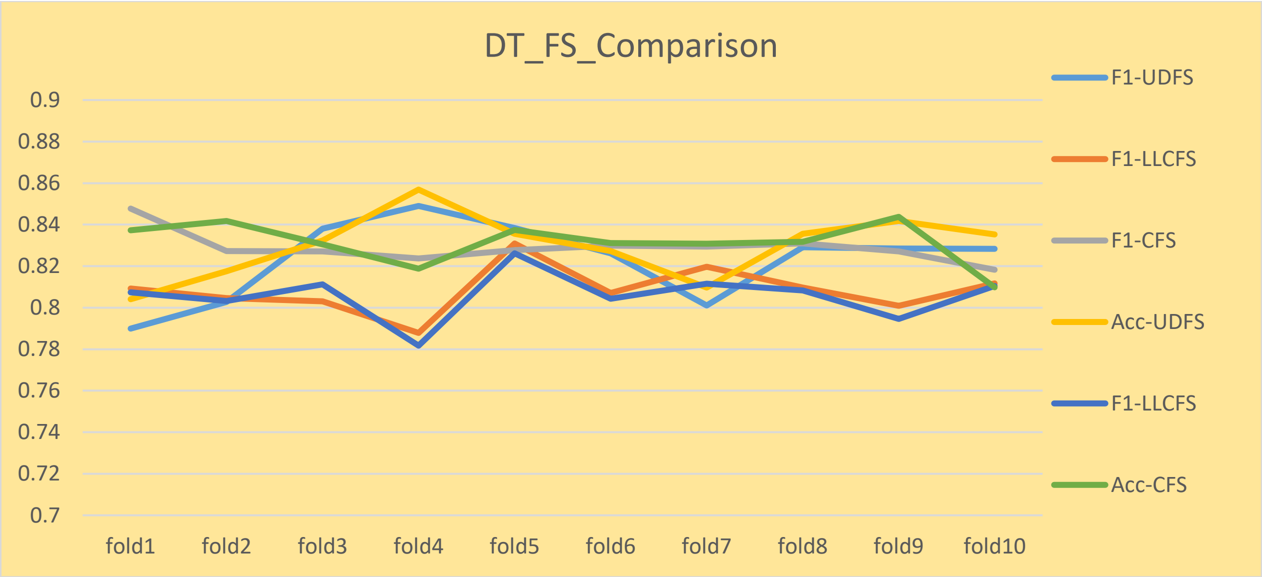


Analysis:

The Average Value of winning Accuracy value is for the UDFS with SVM classifier, which is 0.7884 With Acceptable F1- Score.

Decision Tree Classifier:

Validation	F1-Score			Accuracy		
DT	UDFS	LLCFS	CFS	UDFS	LLCFS	CFS
fold1	0.789916	0.809237	0.847721	0.804067	0.807339	0.837338
fold2	0.802743	0.804539	0.827275	0.817505	0.803241	0.841713
fold3	0.838008	0.803006	0.82715	0.832392	0.811189	0.830508
fold4	0.848987	0.787839	0.823739	0.85687	0.78169	0.818702
fold5	0.838333	0.830915	0.827763	0.835564	0.82619	0.837476
fold6	0.826195	0.807133	0.829865	0.827255	0.804296	0.831094
fold7	0.801058	0.81976	0.829391	0.809615	0.811594	0.830769
fold8	0.829161	0.80964	0.83091	0.83559	0.808354	0.831721
fold9	0.828534	0.800897	0.827111	0.841797	0.794554	0.84375
fold10	0.828278	0.811664	0.818261	0.835294	0.810474	0.809804



Decision tree F1 score and Accuracy Comparison

Analysis:

The Average Value of winning Accuracy value is for the CFS with DT classifier, which is 0.8321 With Acceptable F1- Score.

❖ Conclusion of Dataset:

- The final comparison of the data set’s value for **Gear 7** shows that the Decision Tree classifier with **CFS give us Highest Accuracy**. And that is nearer to the with out Feature selection. There is certain percent of the increase in Accuracy with Acceptable F1-Score.
- The Only Advantage is the reduction of data due to Feature selection returns the important features for the measurement and its same effect on the final result.
- This merit us in term of Computational complexity and cost effective in term of time saving.

❖ F1 score:

F1_Score is statistical term use for the accuracy in the matrix dimensional dataset. It contains the precision (P) and Recall (R). To find the score we need to use the correct positive result and divided it with the sum of all sample that should have to be identify the positive. The F1 Score is the harmonic average of the precision and recall. It is given by

F1 = (2*P*R) / (P+R)

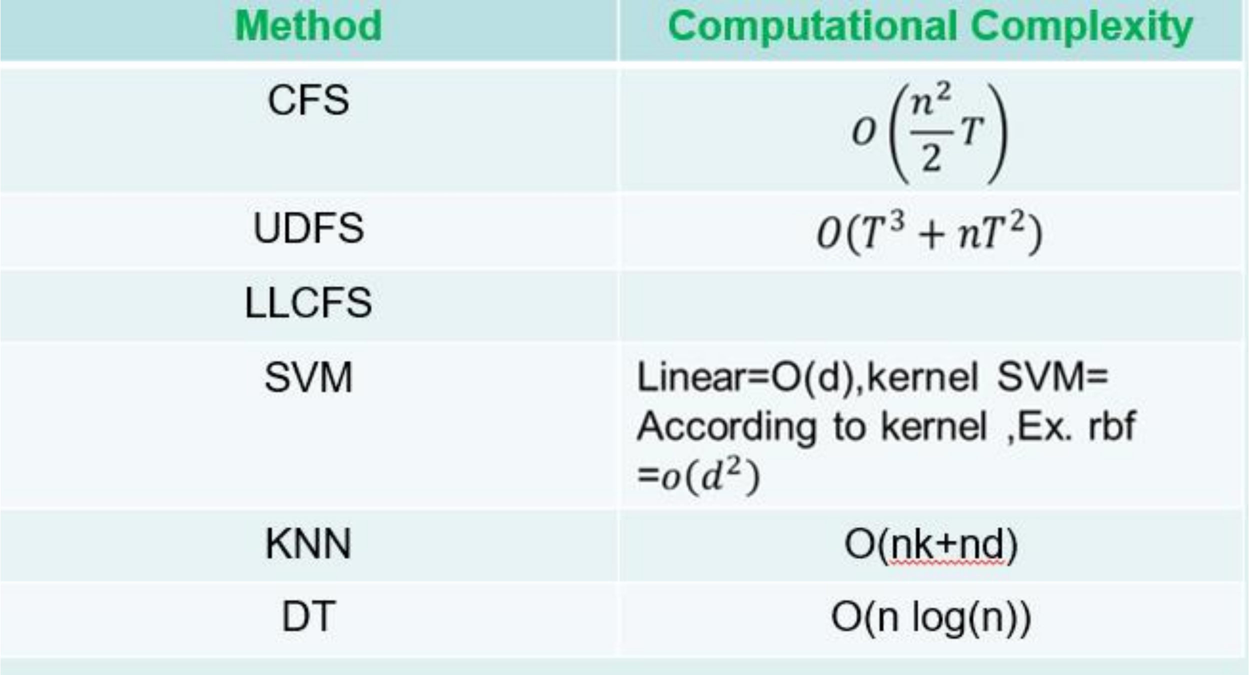
❖ Accuracy:

In Reality, Accuracy score give the error rate of the set from the original dataset and in the data visualization such data give the distance difference in scatter plotting and show how much they are far from the true value in 2D dimension.

❖ Computational Complexity:

- In data Science the computational complexity is important in term of getting the estimation of perfect choice of methods. With data **set dimension it mostly increases or decrease according to the rows (m) and columns(n).**
- In matrix calculation and layering of the algorithm the complexity increases accordingly.
- As an example in Gear dataset 17 we have more than (200000 * 23) values, which give us the high dimension calculative matrix. Give the more complexity in term of time. It takes **23 GB TPU RAM and more than 12 hours of time**.
- Other problem for large data sets after **Gear 20 is 100 percent accuracy**, which is not possible with intermediate level of tuning parameter. The main reason after that is the use of python which take less decimal point of data and due to that most of the data look same in that situation when you normalize them. So, your training and testing data are almost same and that will return the 100 percent accuracy with 1 F1 Score.

- Computational Complexity of Project is depend on all Classifier and Feature selection method Individually.



Method	Computational Complexity
CFS	$O\left(\frac{n^2}{2}T\right)$
UDFS	$O(T^3 + nT^2)$
LLCFS	
SVM	Linear=O(d),kernel SVM= According to kernel ,Ex. rbf =$o(d^2)$
KNN	O(nk+nd)
DT	O(n log(n))

Here, n = no of initial features, T = no of samples, d= Closest distance form the another point.

❖ **References:**

[1] [Yi Yang1, Heng Tao Shen1, Zhigang Ma2, Zi Huang1, Xiaofang Zhou1 (2019). *Norm Regularized Discriminative Feature Selection for Unsupervised Learning*. [online] Ijcai.org. Available at: https://www.ijcai.org/Proceedings/11/Papers/267.pdf [Accessed 14 Jul. 2019].

[2] YM, Z. (2019). *Feature Selection and Kernel Learning for Local Learning-Based Clustering. - PubMed - NCBI*. [online] Ncbi.nlm.nih.gov. Available at: https://www.ncbi.nlm.nih.gov/pubmed/21135434 [Accessed 16 Jul. 2019].

[3] Hall, M. (2019). *Correlation-based Feature Selection for Machine Learning*. [online] Cs.waikato.ac.nz. Available at: https://www.cs.waikato.ac.nz/~mhall/thesis.pdf [Accessed 14 Jul. 2019].

[4] En.wikipedia.org. (2019). *Classifier*. [online] Available at: https://en.wikipedia.org/wiki/Classifier [Accessed 16 Jul. 2019].