EE 559 Final Project: Predict Forest Fire in Algeria Using Machine Learning Technique

Data Set: Algerian forest fires

Longhao Yang, longhaoy@usc.edu

05/03/2022

1. Abstract

Forest Fire is a common but uncontrollable natural disaster that often occurs in California and many regions in the world. Wildfire usually happens in specific weather conditions like temperatures, humidity, wind rate and other factors. If we can predict forest fire using these collected conditions, all effects of the disaster will be prevented as well.

In this project, I explore 5 different machine learning approaches to predict the occurrence of forest fire, including trivial method, Nearest Mean, Perceptron, Support Vector Machines (SVM), Artificial Neuron Network (ANN). The Nearest Mean method is the baseline system for reference. The dataset I use is the Algerian Forest Fires Dataset Data Set (ABID, 2019). This dataset includes training and test datasets. I implemented cross validation in the training dataset and generated various dataset after features engineering such as standardization, feature expansion and reduction (e.g., Pearson Matrix, Backward feature selection, Principal Component Analysis). For selected dataset, I trained the data using these 5 classification methods and predict the results using the validation set and test set.

The final selected classification model is SVM method with margin distance of 1 and RBF kernel. By using the standardized original dataset with SVM, the result presents an accuracy of 88.33% and f score of 83.72% which makes a huge improvement from the baseline Nearest Mean system.

2. Introduction

2.1. Problem Statement and Goals

The dataset I choose is the Algerian Forest Fires dataset. This dataset contains the weather information and forest fires occurred in two regions in Algeria in the period of 3 month in 2012. The goal is to predict the fire occurrence in a future date, based on the collected weather data in the past, using supervised machine learning techniques.

2.2. Literature Review

I have investigated previous or existing approaches to predict fire forest.

Kaggle: *Forest Fire Impact Prediction (Stats and ML)* (psv, Tarun K, 2019)

This Kaggle project used the Forest Fires Data Set posted in Kaggle, which has the same source as the dataset I used. In the report written by Tarun K and psv, they use linear regression on both statistics and machine learning approaches and improve the machine learning approach using forward and backward feature selection.

3. Approach and Implementation

3.1. Dataset Usage

The dataset provided includes a training set and a test set. To sufficiently use the dataset provided, I utilized several feature engineering techniques, including standardizing, feature expansion, cross validation, feature reduction. In this section I will explain these operations in detail.

3.1.1. Original Dataset

The original dataset contains the training set and test set separately. Each data point in each dataset has 11 features all real value without missing value. (ABID, 2019)

Features: Date, Temperature, RH, Ws, Rain, FFMC, DMC, DC, ISI, BUI, Classes.

In the training set, there are 184 data points with no Null value and the test set has 60 data points with no Null value.

3.1.2. Preprocessing

All features are in different units and have different ranges. I standardize the data to make sure all features center at 0 with standard deviation of 1.

3.1.3. Expanded Dataset

Both training and test dataset is expanded in the feature engineering procedure. I implemented the moving average method to calculate the average/min/max/median temperature/rain/humidity/wind of the last 2/5/7 days.

Average	Min	Max	Median
avg_Temp	min_Temp	max_Temp	median_Temp
avg_RH	min_RH	max_RH	median_RH
avg_Ws	min_Ws	max_Ws	median_Ws
avg_Rain	min_Rain	max_Rain	median_Rain

After feature expansion, I got three tables for 2/5/7 days. Since calculating the moving average needs the data of previous 2/5/7 days' data, the first 2/5/7 rows of each table will have missing or inaccurate values. Thus, I dropped the first 2/5/7 rows corresponding to each table.

Table Name	Number of Features	Number of Data Points
train_exp_2	27	176
train_exp_5	27	164
train_exp_**7**	27	156

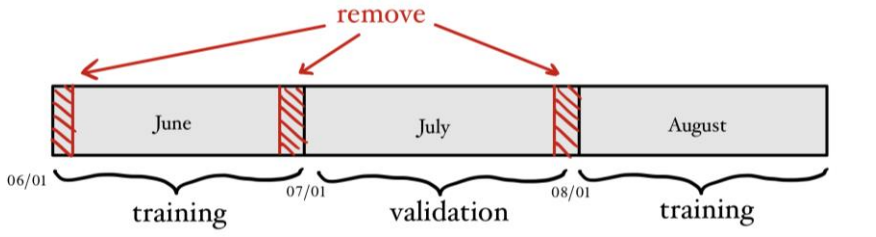
3.1.4. Cross Validation

In the given dataset, due to the limited amount of data points, I split the training dataset in 3 folders (June, July, August) and select one folder as the validation dataset in turns.

Folder	June	July	August
Run 1	Training	Training	Validation
Run 2	Training	Validation	Training
Run 3	Validation	Training	Training

Notice:

If a training set is followed by a validation set (or vice versa), the last 2/5/7 rows of first will be removed, since it will be used to calculate the first 2/5/7 rows of the second.



The cross-validation set is used in the selecting the PCA number of components and Nearest Mean classification. The first validation set (training: June, July, validation: August) is used for all 5 different classification methods and predict the validation set class and measure the metric scores.

3.1.5. Feature Reduction

After expanding the dataset from 11 features to 27 features, the dataset dimension is so large that may cause increased run time complexity and model overfitting. To solve this problem, there are several feature reduction techniques we can use.

- Pearson correlation matrix

- Sequential (backward) feature selection
- Principal component analysis

3.1.5.1. Pearson Correlation

Generating a feature correlation matrix, we can see the correlation between each feature which includes the correlation between each feature and the 'Classes' feature. By using the matrix, I selected 16 features with absolute correlation higher than 0.2 in each of the three table (expanded days = 2/5/7).

Selected Features:

Temperature, RH, Rain, FFMC, DMC, DC, ISI, BUI, avg_Temperature, max_Temperature, median_Temperature, avg_RH, max_RH, avg_Rain, max_Rain, median_Rain

3.1.5.2. Sequential (backward) Feature Selection

Given the Perceptron algorithm I implemented, I used backward feature selection to select the features that gives the best metrics score (mean accuracy). Here is the result:

Number of days (Expanded)	Selected Features (Perceptron)	Accuracy
2	ISI, max_Rain	92.09%
5	Temperature, Ws, DC, ISI, BUI, min_Temperature, median_Temperature, median_Ws	92.10%
7	ISI, avg_Temperature	91.03%

3.1.5.3. PCA Feature Selection

Another feature selection method I used is Principal Components Analysis. The best accuracy result is given by using 18 principal components in perceptron algorithm. With PCA, we cannot get the specific feature index, but we can get an accuracy higher than 92.5%.

The cross-validation set is used for selecting the PCA feature numbers, and nearest mean classification. The validation set and test set are used in every final dataset for metrics score measurement.

3.2. Data Preprocessing

3.2.1. Data Type

The data type of the Date column is string. To split the training set into k-folders for cross validation, we need to convert the Date data type from string to datetime.

3.2.2. Data Standardization

All features are in different units and have different ranges. I standardize the data to make sure all features center at 0 with standard deviation of 1.

The standardized datasets are used in perceptron, SVM, and ANN method. For different method, the standardized dataset is not always better than the other.

- For perceptron method, the standardized dataset generates better result than non-standardized dataset.
- For SVM method, the standardized dataset performs better in general.
- For ANN method, we cannot decide which is better since different number of layers influencing the result as well (not converge in the same condition.)

3.3. Feature engineering

3.3.1. Feature Expansion

Feature expansion is essential for this dataset since there are 11 features in total. By implementing the baseline classifier (Nearest Mean Classifier), the model metrics get an accuracy around 80% and f1 score around 65% which is relatively low.

Thus, I expanded the features with extra 16 features, by calculating the moving average/min/max/median with temperature/rain/humidity/wind of the last 2/5/7 days.

Average	Min	Max	Median
avg_Temp	min_Temp	max_Temp	median_Temp
avg_RH	min_RH	max_RH	median_RH
avg_Ws	min_Ws	max_Ws	median_Ws
avg_Rain	min_Rain	max_Rain	median_Rain

After feature expansion, I got three tables for 2/5/7 days. Since calculating the moving average needs the data of previous 2/5/7 days' data, the first 2/5/7 rows of each table will have missing or inaccurate values. Thus, I dropped the first 2/5/7 rows corresponding to each table.

Table Name	Number of Features	Number of Data Points
train_exp_2	27	176
train_exp_5	27	164
train_exp_**7**	27	156

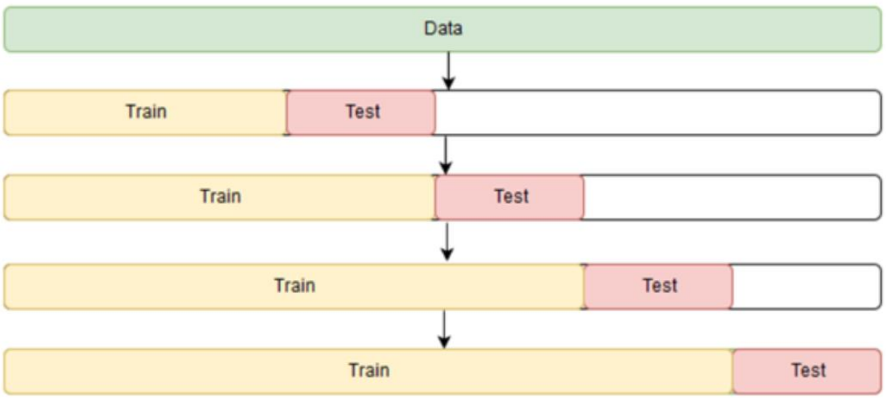
For each model I used, I need numerical proof for the effect of standardization, so there are 3 tables of standardized version with same feature dimension but different value range.

Type	Training variables	
Standardization	Non-std	Standardized
original	X_train	X_train_std
N=2 days	X_train_exp_2	X_train_exp_2_std
N=5 days	X_train_exp_5	X_train_exp_5_std
N=7 days	X_train_exp_7	X_train_exp_7_std

3.3.2. Cross Validation

Since the test dataset is precious, we cannot use the test set frequently to evaluate each model's prediction accuracy. We need to generate the cross validation sets for data testing.

The general cross validation folder for time series is to split the training dataset into k-folders by keeping the time continuity and select the first k-1 folders data as the training set and pick the last kth folder as the validation set.



Source: (Shrivastava, 2020)

In our dataset, due to the limited amount of data points, we cannot use the method stated above, since it will significantly reduce the training set size and influence the prediction accuracy. Thus, we split the training dataset in

3 folders (June, July, August) and select one folder as the validation dataset in turns.

Folder	June	July	August
Run 1	Training	Training	Validation
Run 2	Training	Validation	Training
Run 3	Validation	Training	Training

Notice:

If a training set is followed by a validation set (or vice versa), the last 2/5/7 rows of first set will be removed, since it will be used to calculate the first 2/5/7 rows of the second set.

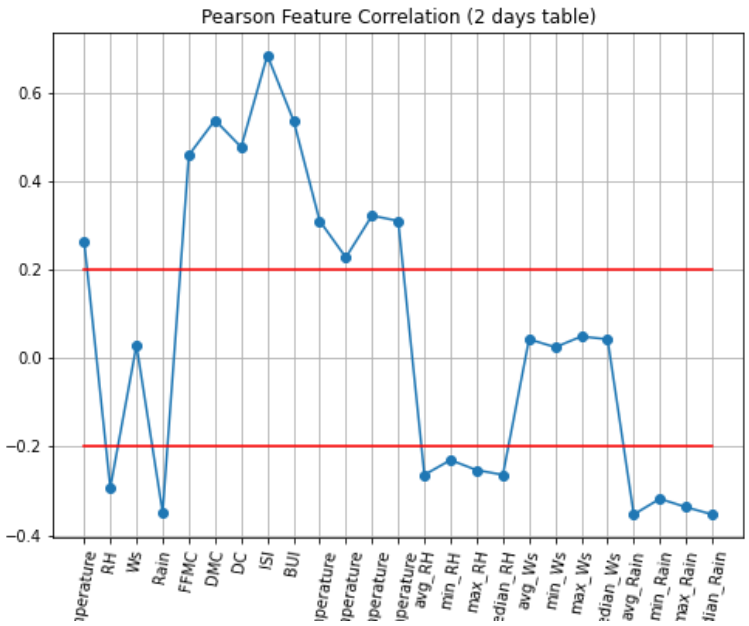
3.4. Feature dimensionality adjustment

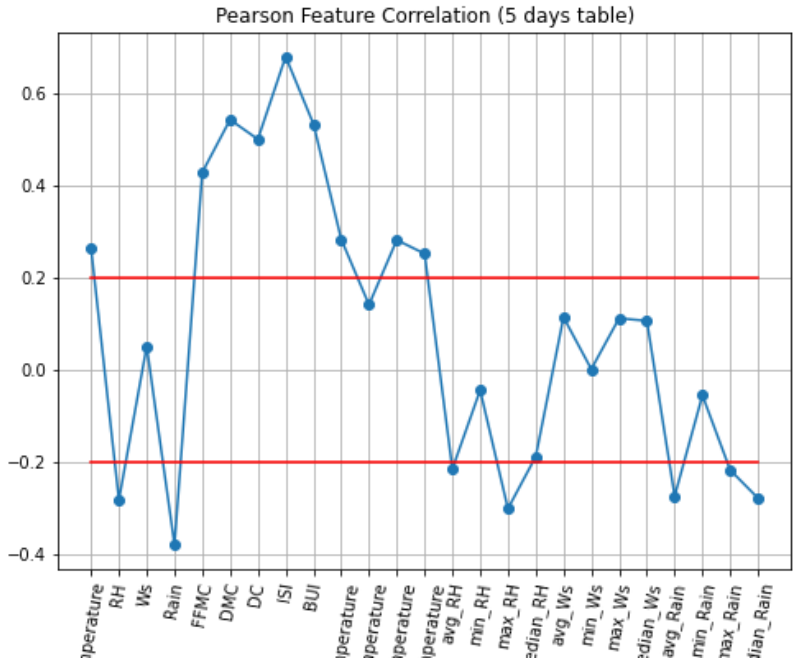
3.4.1. Pearson Correlation

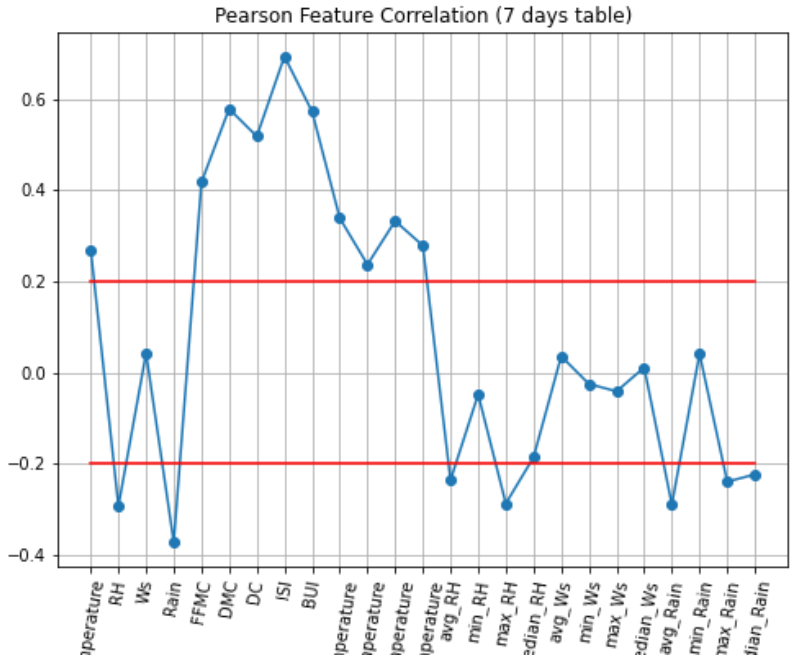
Pearson Correlation is the linear correlation between two variables x and y (Thakur, n.d.). In our case, we need to calculate the correlation between feature variable with the "Classes" variable for each feature.

$$Correlation = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

Generating a feature correlation matrix, we can see the correlation between each feature and the 'Classes' feature.







By using these plots, I selected 17 features with average absolute correlation higher than 0.2 in each of the three table (expanded days = 2/5/7).

Selected Features:

Temperature, RH, Rain, FFMC, DMC, DC, ISI, BUI, avg_Temperature, min_Temperature, max_Temperature, median_Temperature, avg_RH, max_RH, avg_Rain, max_Rain, median_Rain

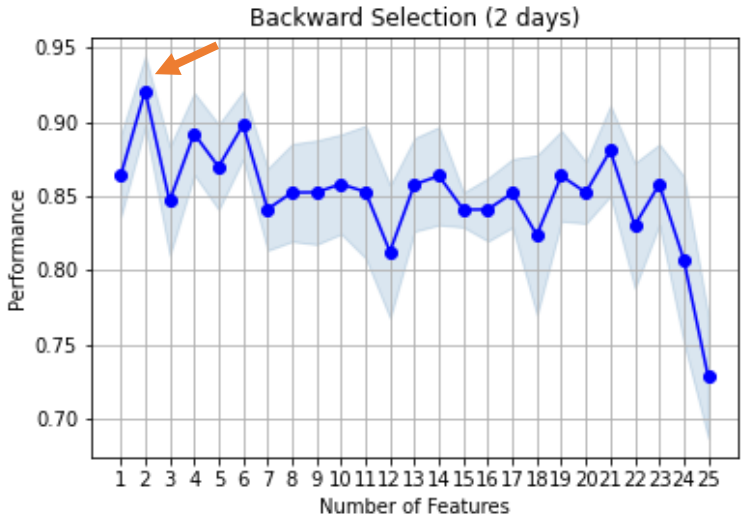
Note: An interesting result is that the selected features for three different tables are very similar to each other.

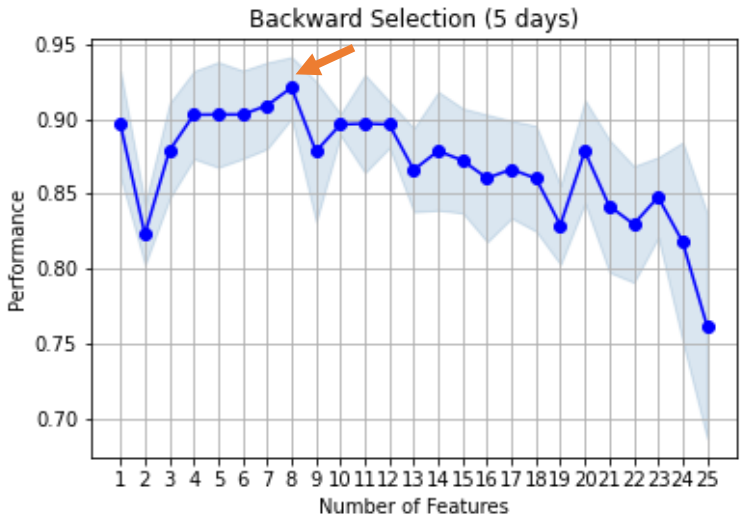
I will use Pearson features as reference feature set for each method in training and model selection section.

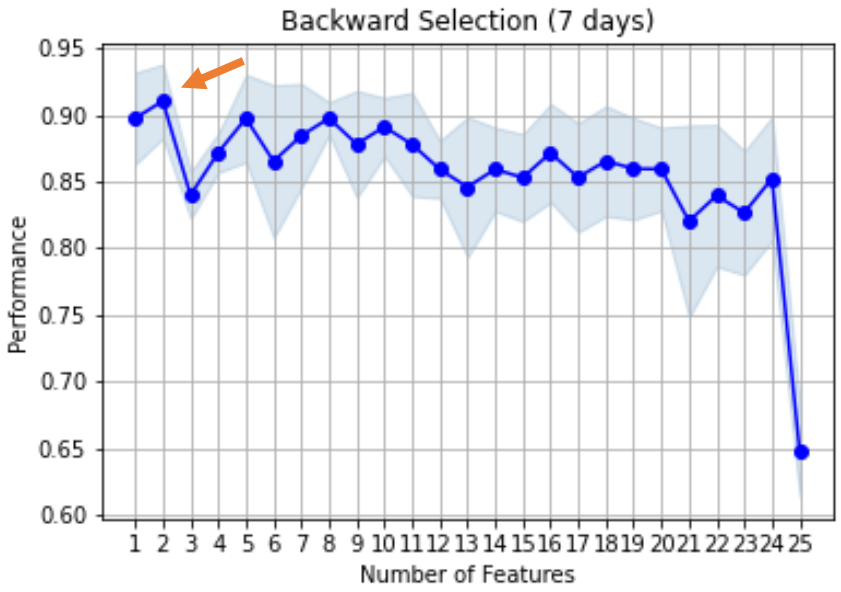
3.4.2. Sequential (backward) Feature Selection

Backward Feature Selection is one of the most popular feature selection techniques. By fitting the full model with all features (25 features), we can get the feature with highest p-value. The next step is to remove the selected feature and fit the model again. By discarding the selected feature recursively, we can get the set of features provides the best accuracy (or other metrics). (Patel, 2021)

I used backward feature selection in the perceptron model to select the features that gives the best metrics score (mean accuracy).







Comparing the maximum accuracy given by 3 tables, here is the result:

Number of days (Expanded)	Selected Features (Perceptron)	Accuracy
2	ISI, max_Rain	92.09%
5	Temperature, Ws, DC, ISI, BUI, min_Temperature, median_Temperature, median_Ws	92.10%
7	ISI, avg_Temperature	91.03%

Although the difference of accuracy between 2-day table and 5-day table is trivial, the number of features in the 2-day table is too few. With few features, there is risk of underfitting. Thus, I decided to use the 5-days expanded table with the selected 7 features for Perceptron model.

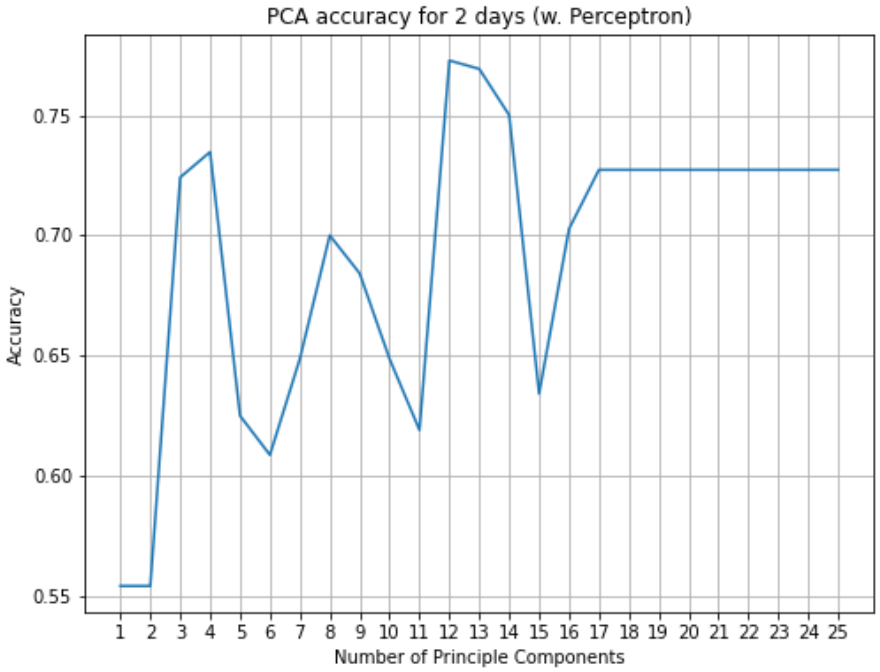
3.4.3. PCA Feature Selection

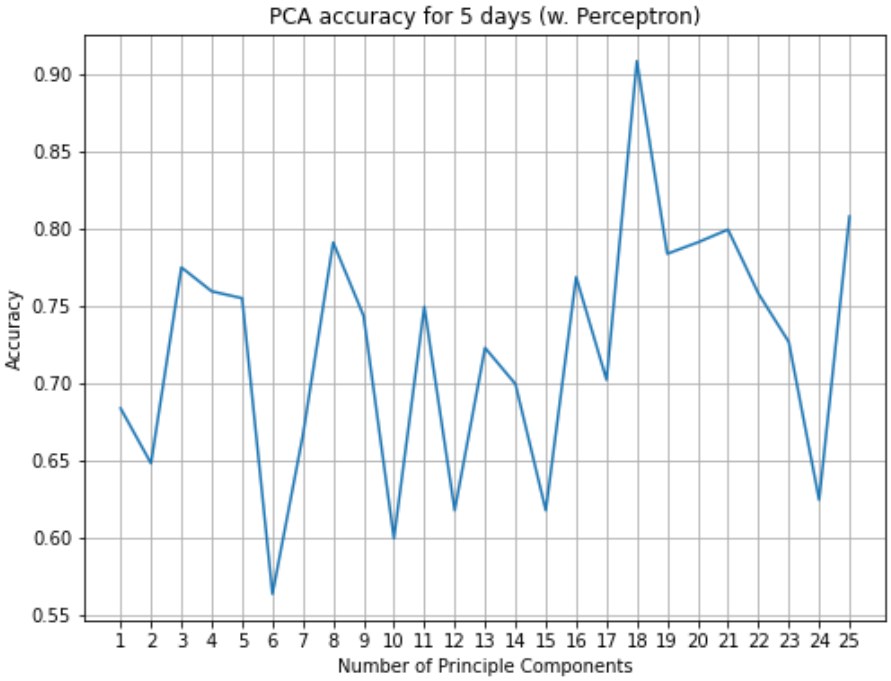
Another feature selection method I used is Principal Components Analysis. The PCA method is projecting all features into selected dimension and fit the model with projected features.

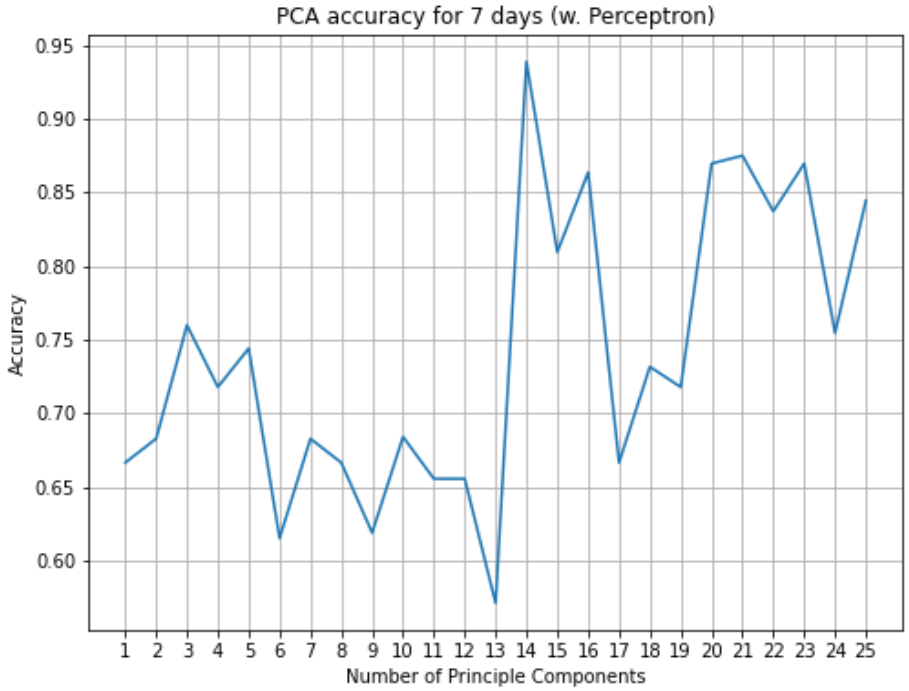
For each of the three table, I iterate through the number of features (from 1 to 25), set the number as the projected dimension (d) and fit the model with the projected features.

I also applied **cross validation** on each table (days = 2, 5, 7), so the result accuracy is the mean accuracy of cross validation set for 3 different sets.

Here is the result of average accuracy curve after cross validation for different PCA dimension for 3 tables.







Here is the summary table presenting the maximum accuracy and f score of each table.

Metrics	2 days	5 days	7 days
Accuracy	83.33 %	93.33 %	95.0 %
F score	77.27 %	90.91 %	93.88 %
Num of features	12	18	14

From the table above, we can conclude that, using the perceptron method with PCA, we can obtain a better model performance by using 7 days expanded table.

3.5. Training, Classification, and Model Selection

The classification model I used is two basic models and 3 explorative models.

- Two basic model: Trivial System and Nearest Mean Model
- Three explorative models:
  - Perceptron model
  - SVM model
  - ANN model

3.5.1. Trivial System

The trivial system is outputting the class assignment (S1, S2) at random with probability N1/N and N2/N, respectively, N' is the population of data points with class label Si, and N is the total population of data points, all based on the training set.

- P (class 1) = N1/N = 0.375
- P (class 2) = N2/N = 0.625

Based on these two classes probability, I randomly assign the classification to the test set. Finally, I get these following metric scores:

- F1 score = 45.61 %
- Accuracy = 48.33 %
- Confusion Matrix = $\begin{bmatrix} 16 & 21 \\ 10 & 13 \end{bmatrix}$

Summary

As expected, the trivial system behaves poorly on classification.

3.5.2. Nearest Mean System (Baseline)

The nearest mean system is classifying the data points based on their distance to the class mean point. The specific procedure is listed below:

1. Find the mean of each class ("fire", "not fire") in the training set
2. Calculate the distance between test points and each class mean
3. The class of shorter distance determines the testing points class

I implemented the Nearest Mean system to the original dataset, expanded dataset, and their standardized dataset, and get the result in the table below.

Dataset	Non-Standardized		Standardized		Difference of std
	Accuracy	F score	Accuracy	F score	
Original	78.33 %	62.86 %	81.67 %	70.27 %	Increased
2 days	76.67 %	61.11 %	75.0 %	61.54 %	static
5 days	78.33 %	64.86 %	70.0 %	50.0 %	decreased
7 days	81.67 %	70.27 %	76.67 %	63.16 %	decreased

Summary

From the table above, we can observe that the **best** result is coming from the non-Standardized 7 days expanded table and Standardized original dataset.

Both tables give the same result, and we can explore their confusion matrix, get the matrix below.

Class	True Positive	True Negative
Predicted Positive	36	1
Predicted Negative	10	13

This model has a precision of 97.2% and recall of 78.26%.

3.5.3. Perceptron Classification

The perceptron classifier I used is a linear model.

Steps:

1. Randomly generate a starting weighted vector
2. Calculate the loss function at each point by setting learning rate as eta
3. Iterate through the dataset for m iterations unless there is no change after one iteration.

The default perceptron model I set up:

- Number of iterations: 40
- Learning rate: 0.1
- Random state: 0 (for comparison)

I implemented the perceptron model to the original dataset, expanded dataset, and their standardized dataset, and get the result in the table below.

Dataset	Non-Standardized		Standardized		Difference of std
	Accuracy	F score	Accuracy	F score	
Original	81.67 %	71.79 %	86.36 %	90.0 %	Increased
2 days	63.33 %	67.65 %	88.33 %	85.71 %	Increased
5 days	85.0 %	83.64 %	90.0 %	87.5 %	Increased
7 days	86.67 %	80.95 %	90.0 %	86.96 %	Increased

3.5.4. Support Vector Machine Classification

In the support vector machine model, I used the SVC with RBF kernel (exponential kernel) to map the dataset into u-space and assume data is linearly separable in u-space.

For the support margin of SVM, I test the margin equals 0.5, 1, 2, 3 using cross validation, and get the optimal result when margin is equal to 1.

- Kernel: RBF
- Margin: 1

Dataset	Non-Standardized		Standardized		Difference of std
	Accuracy	F score	Accuracy	F score	
Original	81.67 %	79.25 %	88.33 %	83.72 %	Increased
2 days	83.33 %	80.77 %	80.0 %	73.91 %	Decreased
5 days	81.67 %	78.43 %	83.33 %	77.27 %	Increased
7 days	80.0 %	76.0 %	85.0 %	80.0 %	Increased

3.5.5. Artificial Neuron Network

The artificial neuron network classifier I used is constructed an input layer with 26 units and an output layer with 1 unit. By changing the hidden layer size and number of units in each layer, I can get the best result.

I used the default activation function RELu which returns max of 0 and x

Parameter I used for hidden layers: I always use 2 or 3 hidden layers with different hidden units in each layer.

Format: Number of units in layer 1, number of units in layer 2, etc.

- Deceasing number of units by half in each layer

  (10, 5, 2), (7, 3)

- Constant (small) number of units in each layer

  (2, 3), (3, 4), (4, 4)

The best result is given by using the features expanded by two days and non-standardized. It gives the accuracy of 91.67% and f score of 88.89%.

4. Results and Analysis: Comparison and Interpretation

4.1. Trivial

The trivial method performs as expected with low accuracy and F score.

	Accuracy	F score	Confusion matrix	parameters
Trivial	48.28 %	60.53 %	$\begin{bmatrix} 5 & 4 \\ 26 & 23 \end{bmatrix}$	P1 = N_1/N P2 = N_2/N

4.2. Nearest Mean

Applying the nearest mean method to the non-standardized cross-validation set, I get the following result:

Cross validation	Non-Standardized	
	Accuracy	F score
2 days	73.77 %	68.14 %
5 days	74.55 %	67.98 %
7 days	80.33 %	73.74 %

I did not apply the cross-validation method to the standardized method due to time restriction, but we still can observe that the 7 days table performs better.

Applying the method to both non-standardized and standardized training-test set, I get the following result:

Test Set	Non-Standardized		Standardized		Difference of std
	Accuracy	F score	Accuracy	F score	
Original	78.33 %	62.86 %	81.67 %	70.27 %	Increased
2 days	76.67 %	61.11 %	75.0 %	61.54 %	static
5 days	78.33 %	64.86 %	70.0 %	50.0 %	decreased
7 days	81.67 %	70.27 %	76.67 %	63.16 %	decreased

Combining the result of cross validation and train-test set, we can conclude the best result comes from the **non-Standardized dataset with expanded by 7 days**.

We can see from the table above that not all set of data benefits from standardization. Standardization will cause a loss of information in some cases.

4.3. Perceptron

Applying the method to both non-standardized and standardized training-test set, I get the following result:

Dataset	Non-Standardized		Standardized		Difference of std
	Accuracy	F score	Accuracy	F score	
Original	81.67 %	71.79 %	86.36 %	90.0 %	Increased
2 days	63.33 %	67.65 %	88.33 %	85.71 %	Increased
5 days	85.0 %	83.64 %	90.0 %	87.5 %	Increased
7 days	86.67 %	80.95 %	90.0 %	86.96 %	Increased

I also applied the perceptron method to Pearson reduction features training and test set with both non-standardized and standardized data, I get the following result:

Pearson Dataset	Non-Standardized		Standardized		Difference of std
	Accuracy	F score	Accuracy	F score	
2 days	70.0 %	71.88 %	88.33 %	83.72 %	Increased
5 days	76.67 %	58.82 %	90.0 %	86.96 %	Increased
7 days	80.0 %	66.67 %	81.67 %	78.43 %	Increased

I also applied the perceptron method to Pearson reduction features training and test set with both non-standardized and standardized data, I get the following result:

Backward Dataset	Non-Standardized Accuracy	Features
2 days	92.09%	ISI, max_Rain
5 days	92.10%	Temperature, Ws, DC, ISI, BUI, min_Temperature, median_Temperature, median_Ws
7 days	91.03%	ISI, avg_Temperature

I implemented many training and test set data to the perceptron method. One common phenomenon is that it performs well in the expanded 5 days table with high accuracy. That is abnormal and it would be generalized if I implemented cross validation to each dataset. I assume there is an overfitting problem since I used the test set not validation set in comparison.

4.4. SVM

I applied the SVM method to training and test set with both non-standardized and standardized data, I get the following result:

Dataset	Non-Standardized		Standardized		Difference of std
	Accuracy	F score	Accuracy	F score	
Original	81.67 %	79.25 %	88.33 %	83.72 %	Increased

2 days	83.33 %	80.77 %	80.0 %	73.91 %	Decreased
5 days	81.67 %	78.43 %	83.33 %	77.27 %	Increased
7 days	80.0 %	76.0 %	85.0 %	80.0 %	Increased

4.5. ANN

I applied the ANN method to training and test set with both non-standardized and standardized data, I get the following result:

Train test Dataset	Non-Standardized		Hidden layer	Standardized		Hidden layer
	Accuracy	F score		Accuracy	F score	
Original	86.67 %	80.95 %	(5, 3)	90.0 %	83.72 %	(2, 3)
2 days	91.67 %	88.89 %	(2, 3)	86.67 %	80.0 %	(2, 3)
5 days	90.0 %	86.36 %	(4, 4)	83.33 %	75.0 %	(2, 3)
7 days	86.67 %	82.61 %	(4, 4)	88.33 %	84.44 %	(3, 4)

I am surprised that the ANN method is not performing well as I expected. I tried to decrease the number of units by half in each layer, but the metric score didn't perform well, so I used the (2, 3), (3, 4), and (4, 4) in the end. There are many variations in the layer size and iteration parameter choice. Hopefully, I can explore them more in future.

4.6. Best Result Comparison

By applying the best model of each algorithm to the first validation set, I got the following result.

Validation set 1	Accuracy	F score	Confusion matrix
Trivial	48.28 %	60.53 %	$\begin{bmatrix} 5 & 4 \\ 26 & 23 \end{bmatrix}$
Nearest	89.58 %	93.83 %	$\begin{bmatrix} 5 & 3 \\ 2 & 38 \end{bmatrix}$
Perceptron	92.31 %	95.65 %	$\begin{bmatrix} 4 & 4 \\ 0 & 44 \end{bmatrix}$
SVM	93.1 %	95.92 %	$\begin{bmatrix} 7 & 2 \\ 2 & 47 \end{bmatrix}$
ANN	87.93 %	92.78 %	$\begin{bmatrix} 6 & 3 \\ 4 & 45 \end{bmatrix}$

The validation set I used:

- Training set: June and July
- Validation set: August

From the result above, I selected the SVM model as the best model for the Algeria forest fire prediction problem.

All information for each best performance model is listed below. With the information table (unmentioned is default setting by sklearn), we can generate same or similar model for population dataset.

Validation set 1	Standardized	Features used	Parameters
Trivial	No	Original table	P1 = N_1/N P2 = N_2/N
Nearest	No	Expand 7 days	Euclidean distance
Perceptron	No	Expand 5 days **Backward Selection Features**	Num iter = 40 eta rate = 0.1
SVM	Yes	Original table	Marg. dist = 1 Kernel: rbf Gamma: 1/11
ANN	Yes	Expanded by 2 days	2 Hidden layers. Layer1: 2 units Layer2: 3 units Num iter = 500

Let us review the improvement from the trivial system to the final chosen system **Support Vector Machine**. All metric scores are measured using the train and test set.

Test set	Accuracy	F score	Confusion matrix	parameters
Trivial	40.74 %	46.67 %	$\begin{bmatrix}17 & 20\\12 & 11\end{bmatrix}$	P1 = N_1/N P2 = N_2/N
Nearest	81.67 %	70.27 %	$\begin{bmatrix}36 & 1\\10 & 13\end{bmatrix}$	Same as table above
SVM	88.33 %	83.72 %	$\begin{bmatrix}35 & 2\\10 & 18\end{bmatrix}$	Marg. dist = 1 Kernel: rbf Gamma: 1/11

My best performing system was Support Vector Machine. I used the standardized original dataset. The marginal distance of the SVM is 1 and I used the rbf kernel. The gamma value for the system is 1/number of features (1/11 in my case).

After doing many preprocessing and feature engineering steps, it surprised me that the original feature set with standardization performs the best.

It implies that not every machine learning technique leads to a better result. For example, many datasets gave a worse result after standardization.

Some complex ML method is not suitable for this problem like the ANN method, while it is computational expensive.

However, each method is worth to try. Since with many variations in feature combinations, parameter changes, and dataset usage, we cannot miss any possible result.

In the training and classification process I used the cross validation set for perceptron method and used the validation and test set for each classifier for each different dataset. Here I made a mistake that used the test set too many times and it should be avoid in the future.

5. Libraries used and what you coded yourself

There are many libraries I used in this project.

I used the pandas, NumPy, datetime, statistics libraries for data cleaning and formatting. For image display, I used the IPython, seaborn, matplotlib for showing and plotting images. For feature selection and machine learning models, I mainly used the sklearn library for different functions.

The functions that I implemented by myself includes expand_table function. It helps expand the features of the dataset by inputting number of days (2, 5, 7). Another function is called train_cv_1 (train_cv_2, train_cv_3) which helps to split the cross-validation set.

6. Summary and conclusions

In this project, to predict the forest fire in Algeria, I implemented different preprocessing and feature engineering steps. Using perceptron, support vector machine, and artificial neuron network techniques, I got a prediction accuracy around 88-90% for the test set by selecting SVM as the best model.

This result can be further improved by using cross-validation at each stage of parameter selection or at model selection. I made the mistake of using the test set to

evaluate the accuracy at first, which influenced the result as well. If we can get a larger dataset and using the validation set to measure metrics, the final result will be improved as well.

References

For each source cited above, include the reference here.

ABID, F. (2019, July 08 - 11). Predicting Forest Fire in Algeria using Data Mining Techniques: Case Study of the Decision Tree Algorithm. *International Conference on Advanced Intelligent Systems for Sustainable Development (AI2SD 2019).* Marrakech, Morocco. Retrieved from International Conference on Advanced Intelligent Systems for Sustainable Development (AI2SD 2019): https://archive.ics.uci.edu/ml/datasets/Algerian+Forest+Fires+Dataset++#

Patel, A. (2021, June 21). *Backward Feature Selection |Sequential Backward Selection|Wrapper Method Part 2|Tutorial 8*. Retrieved from Youtube: https://www.youtube.com/watch?v=2Y4PrhMyqX0

psv, Tarun K. (2019, December 01). *forestfire impact prediction (stats and ml)*. Retrieved from Kaggle: https://www.kaggle.com/code/psvishnu/forestfire-impact-prediction-stats-and-ml

Shrivastava, S. (2020, January 14). *Cross Validation in Time Series*. Retrieved from Medium: https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4

Thakur, M. (n.d.). *Pearson Correlation Coefficient*. Retrieved from WallStreetMojo: https://www.wallstreetmojo.com/pearson-correlation-coefficient/

Vitor Cerqueira, F. P. (2016). Combining Boosted Trees with Metafeature. *Advances in Intelligent Data Analysis XV: 15th International Symposium.* Stockholm.