

Forest Fire Prediction Using Machine Learning Approaches

Madhuri Ranvirkar
ISE 244

Introduction

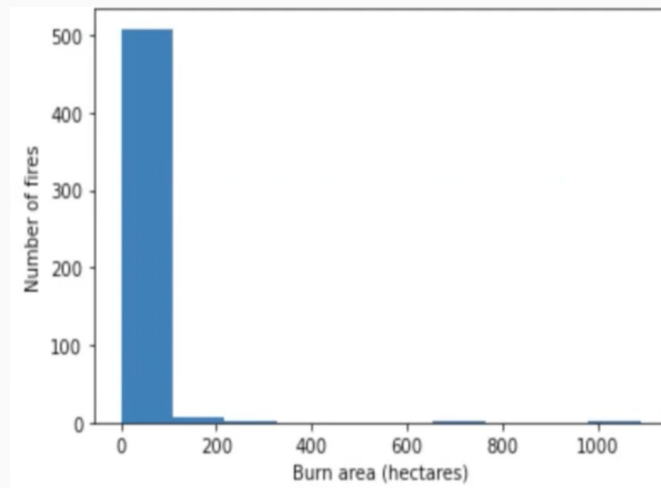
- **Forest cover the more than 31% of the worlds lands surface.**
- **Also, help to maintain the ecological balance.**
- **Forest fires are the most Destructive and dangerous accidents**
- **Predicting the occurrences of forest fires and burn areas is a challenging task**
- **3000+ lives lost, \$23 billion property damage every year**
- **Forest Fires are both difficult to fight and predict**
- **Forest Fires can cause respiratory disease.**

Objective

- Objective with this study is to explore and evaluate the potential of machine learning methods
- Machine learning methods can be used to reduce forest fire problem
- Using these methods we can predict the burned area of forest fires
- Machine learning methods can be used in preventive applications

Dataset

- **Forest fire dataset from the University of California, Irvine(UCI) machine learning repository collected from the northeastern region of Portugal**
- **Most of the data has no fires**
- **Data is skewed, lots of outliers**
- **DataSet Characteristics:**
- **Multivariate Number of Instances(rows):517**
- **Attribute Characteristics:Real**
- **Number of Attributes:13**



Dataset

Variables	Description
X	X-axis spatial coordinate (from 1 to 9)
Y	Y-axis spatial coordinate (from 1 to 9)
Month	Month of the year (from “January” to “December”)
Day	Day of the week (from “Monday” to “Sunday”)
FFMC	FFMC code from the FWI system (from 18.7 to 96.20)
DMC	DMC code from the FWI system (from 1.1 to 291.3)
DC	DC code from the FWI system (from 7.9 to 860.6)
ISI	ISI code from the FWI system (from 0 to 56.10)
Temp	Temperature in degrees Celsius (from 2.2 to 33.30)
RH	Relative humidity in percentage (from 15.0 to 100)
Wind	Wind speed in km/h (from 0.40 to 9.40)
Rain	Outside rain in mm/m ² (from 0.0 to 6.40)
Area	Total burned area of the forest (in <i>ha</i>) (from 0.00 to 1090.84)

Fire Weather Index (FWI)
Fine Fuel Moisture Code (FFMC)
Duff Moisture Code (DMC)
Drought code(DC)
Initial Spread Index (ISI)

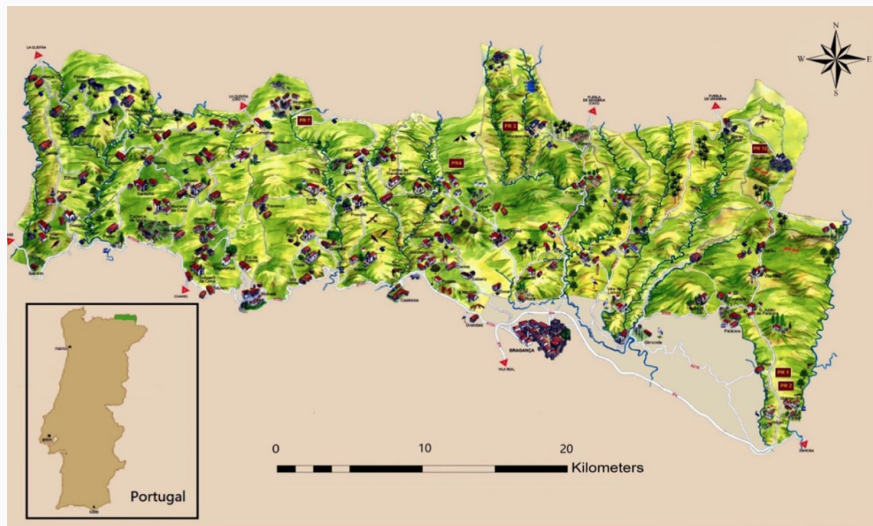



Fig. 1 Map of the Montesinho Natural Park, Portugal

Data Pre-processing

- Text data included with the numeric data(Month & Days)
- Encoded month & days in numeric form
- Dataset is divided into Train set and Test set
- To remove outliers StandardScaler() method is used

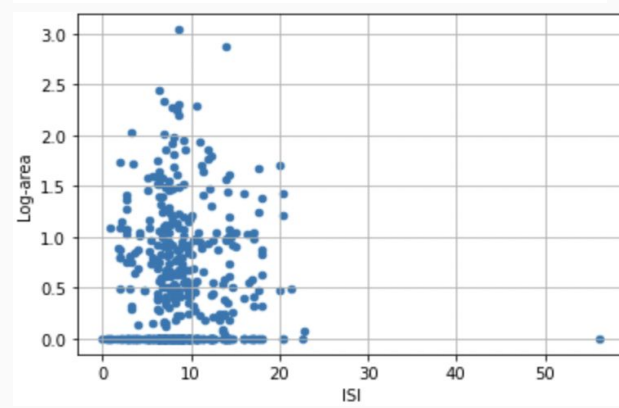
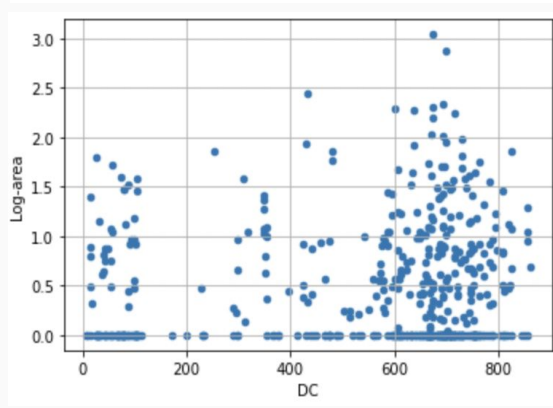
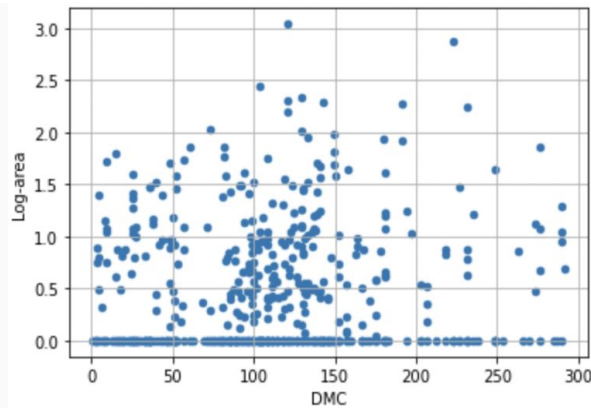
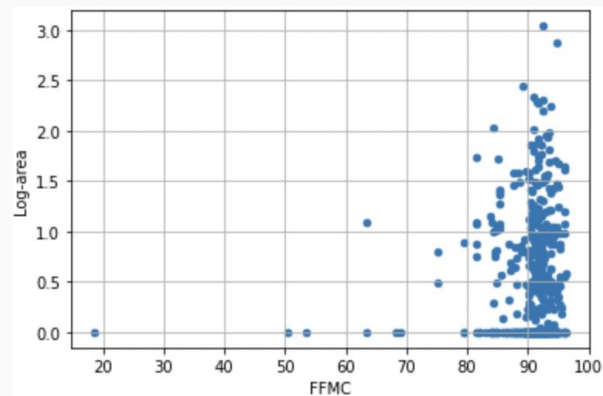
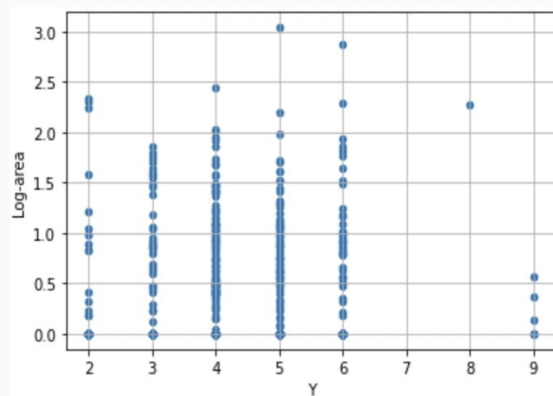
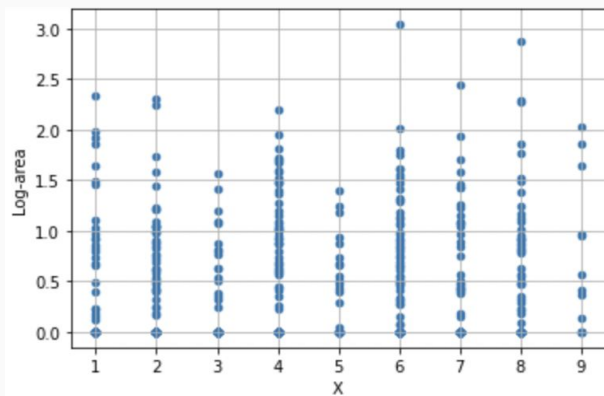
```
data.head()
```

	X	Y	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
0	7	5	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	0.0
1	7	4	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	0.0
2	7	4	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	0.0
3	8	6	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	0.0
4	8	6	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	0.0

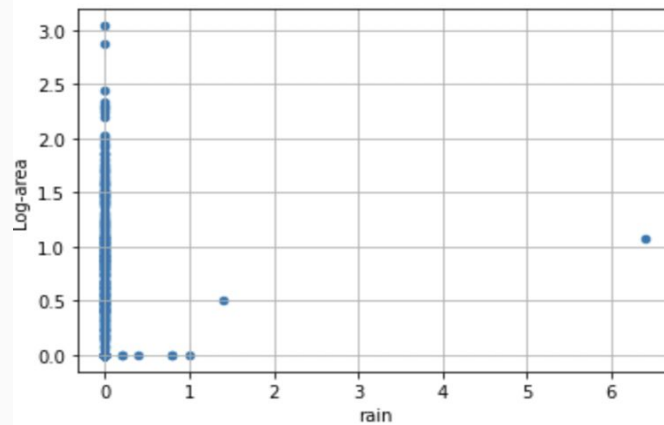
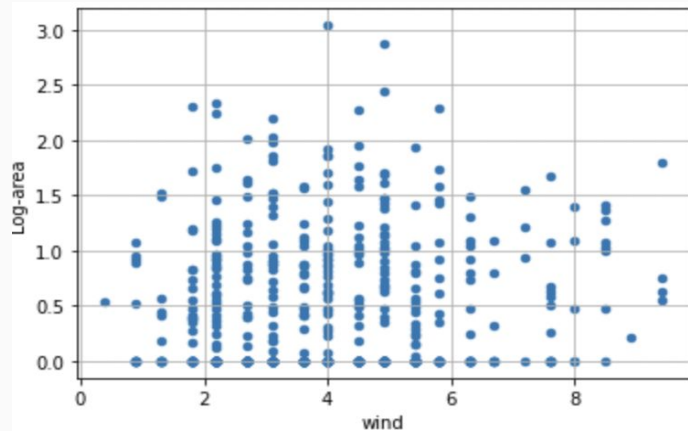
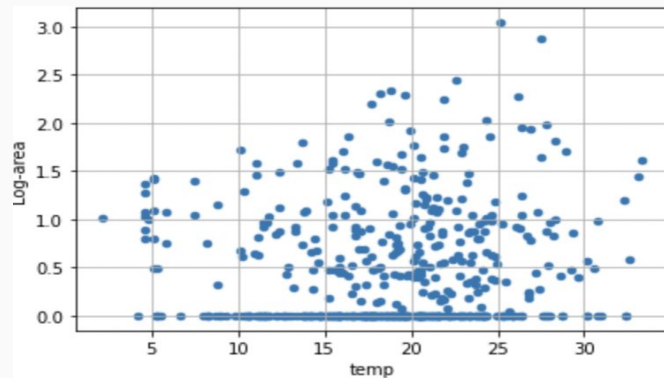
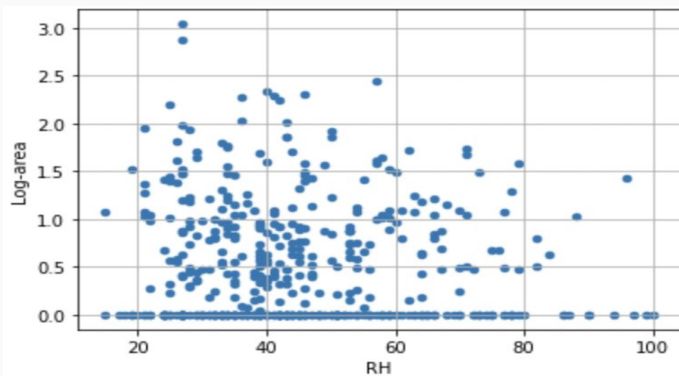
A dramatic photograph of a forest fire. Thick, billowing white and grey smoke rises from the ground, partially obscuring the background. In the foreground, several tall, dark, charred tree trunks stand like sentinels. To the left, a bright orange and yellow fire burns on the forest floor. On the right, a fallen tree trunk is also engulfed in flames. The overall atmosphere is one of destruction and intensity, with a warm, hazy light from the fire illuminating the scene.

Data visualization

Data Visualization(dataset says about different features)



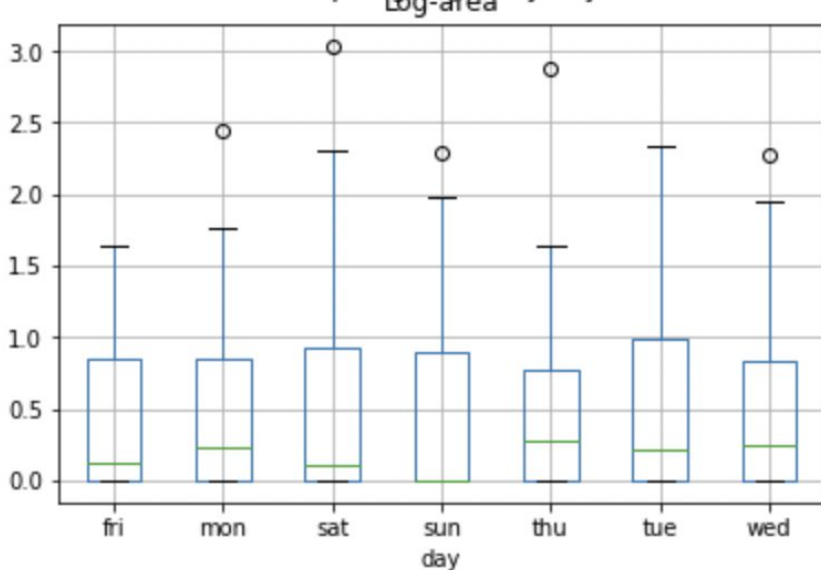
Data Visualization...



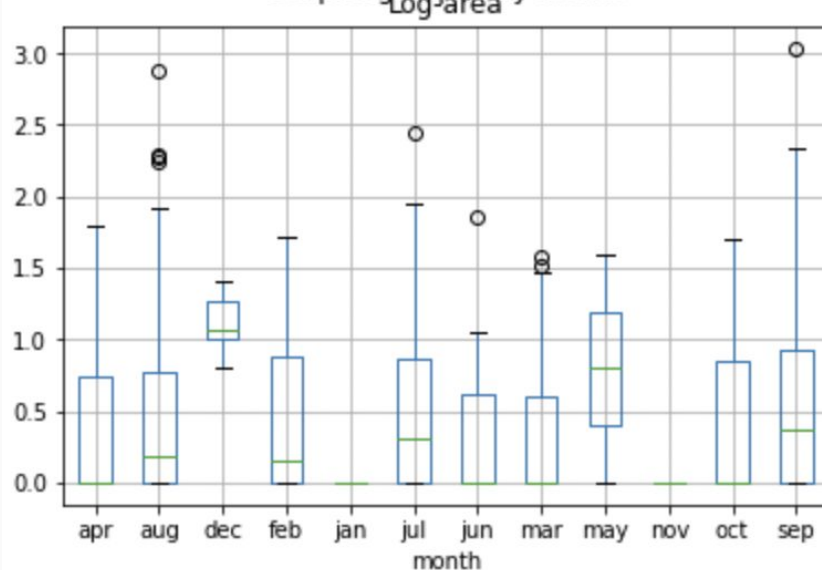
Data Visualization...

Boxplots of the categorical features (month and day)

Boxplot grouped by day



Boxplot grouped by month



Models Used

- Created a baseline test of using just a simple linear regression
linear regression performed poorly predicted negative burned area
- Decided to try different algorithms
- Looked further into, Logistic Regression, Decision Tree Regressor, Random Forest Regressor, Support Vector Regressor, and Neural Network Classifier is used for prediction
- Used r-square, root mean squared error(RMSE) as evaluation metrics

Models Used

- 1) **Linear Regression:** Performance of Linear Regression R^2 metric 0.51
- 2) **MLP Regressor Model:** Performance of MLP Regression Model R^2 metric 0.50639
- 3) **Decision Tree Regressor:** R^2 Score = -5.82
- 4) **Random Forest Regressor:** R^2 Score = -7.42
- 5) **Logistic Regression:** Logistic Regression Accuracy, 95.58011%
- 6) **Neural Network Classifier:** Neural Network Classifier Accuracy, 92.81768%
- 7) **Support Vector Regressor with GRIDCV:** RMSE for Support Vector Regression: 0.615
- 8) **Decision Tree Regressor with GRIDCV:** RMSE for Decision Tree: 2.68
- 9) **Random Forest Regressor with GRIDCV:** RMSE for Random Forest: 0.56

Results

Results of Linear Regression and Logistic regression model

```
#The maximum val. of R^2 can be 1.0 that signifies that Li  
#Here R^2 score is 0.51 which is low signifying that Linea  
linear_reg_model=LinearRegression()  
linear_reg_model.fit(X_train,Y_train)
```

```
print("Performance of Linear Regression R^2 metric {:.5f}")
```

```
Performance of Linear Regression R^2 metric 0.51851
```

```
log_reg_model=LogisticRegression()  
log_reg_model.fit(X_train,Y_train)
```

```
print('Logistic Regression Accuracy, {:.5f}%'.)
```

```
Logistic Regression Accuracy, 95.58011%
```


Result of Decision Tree and Random Forest

3. Decision Tree Regressor

```
#R2 score negative not a good fit!
reg = dtr(random_state = 42)
reg.fit(X_train, Y_train)
Y_pred = reg.predict(X_test)
print("MSE =", mse(Y_pred, Y_test))
print("MAE =", mae(Y_pred, Y_test))
print("R2 Score =", r2_score(Y_pred, Y_test))
```

MSE = 6048.557144198893
MAE = 8.413977900552485
R2 Score = -5.8201150257274055

4. Random Forest Regressor

```
#This works even worse than Decision Tree Regressor
regr = RandomForestRegressor(max_depth=2, random_state=42)
regr.fit(X_train, Y_train)
Y_pred = regr.predict(X_test)
print("MSE =", mse(Y_pred, Y_test))
print("MAE =", mae(Y_pred, Y_test))
print("R2 Score =", r2_score(Y_pred, Y_test))
```

MSE = 6230.821457813967
MAE = 11.8437431690821
R2 Score = -7.428650981864372

```
tree_model = DecisionTreeRegressor(max_depth=10,criterion='mae')
tree_model.fit(scaler.fit_transform(X_train),scaler.fit_transform(y_train))
a=tree_model.predict(X_test)
print("RMSE for Decision Tree:",np.sqrt(np.mean((y_test-a)**2)))
```

RMSE for Decision Tree: 2.6804237037879517

```
from sklearn.ensemble import RandomForestRegressor
param_grid = {'max_depth': [5,10,15,20,50],
              'max_leaf_nodes': [2,5,10],
              'min_samples_leaf': [2,5,10],
              'min_samples_split': [2,5,10]}
grid_RF = GridSearchCV(RandomForestRegressor(),param_grid,refit=True,
                        cv=5)
grid_RF.fit(X_train,y_train)
a=grid_RF.predict(X_test)
rmse_rf=np.sqrt(np.mean((y_test-a)**2))
print("RMSE for Random Forest:",rmse_rf)
```

RMSE for Random Forest: 0.5673316386872838

Result of Neural network classifier and Support vector regression

6. Neural Network Classifier

```
nn_classifier_model=MLPClassifier(activation='relu',hidden_layer_sizes=(16, 16), n_iter_no_change=100)
nn_classifier_model.fit(X_train,Y_train)
```

```
MLPClassifier(hidden_layer_sizes=(16, 16), n_iter_no_change=100)
```

```
print('Neural Network Classifier Accuracy, {:.5f}%'.format(nn_classifier_model.score(X_test,Y_test)))
```

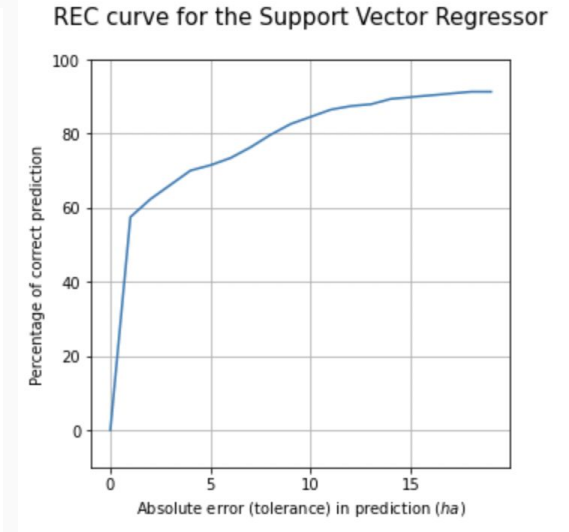
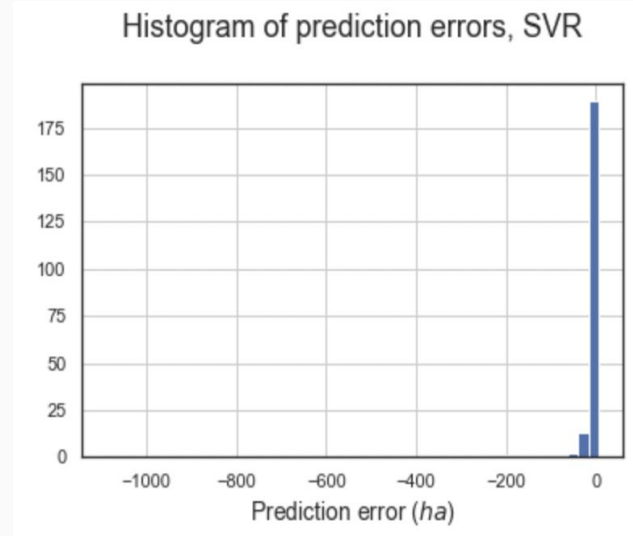
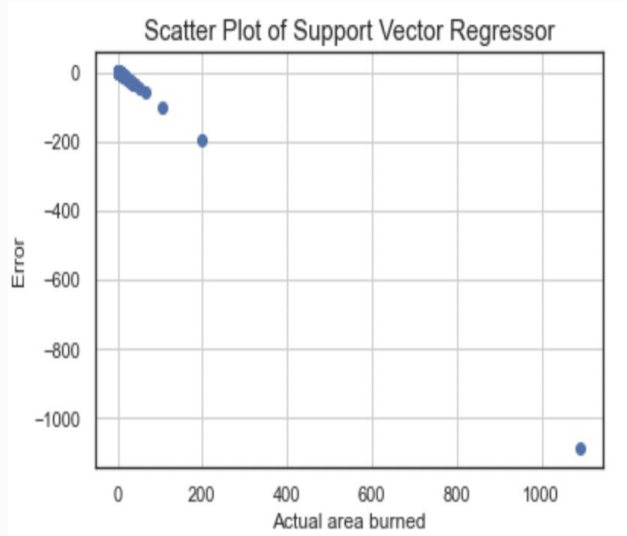
Neural Network Classifier Accuracy, 92.81768%

```
import warnings
warnings.filterwarnings("ignore")

grid_SVR = GridSearchCV(SVR(),param_grid,refit=True,verbose=0,cv=5)
grid_SVR.fit(scaler.fit_transform(X_train),scaler.fit_transform(y_train))
a=grid_SVR.predict(X_test)
print("RMSE for Support Vector Regression:",np.sqrt(np.mean((y_test-a)**2)))
```

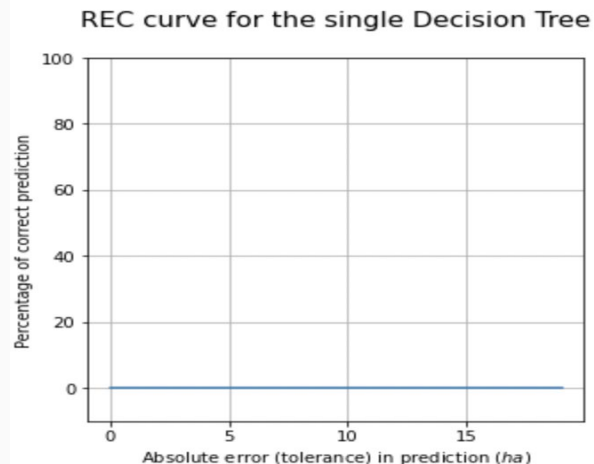
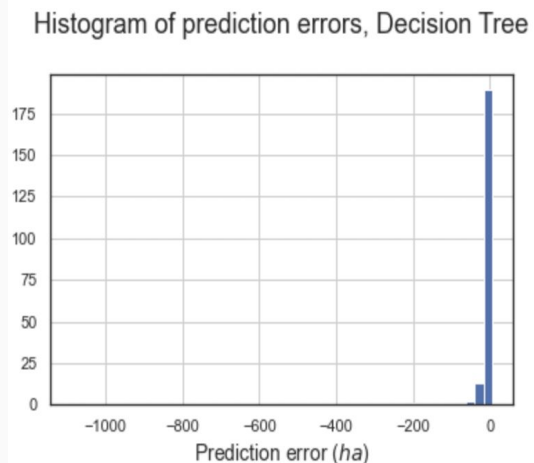
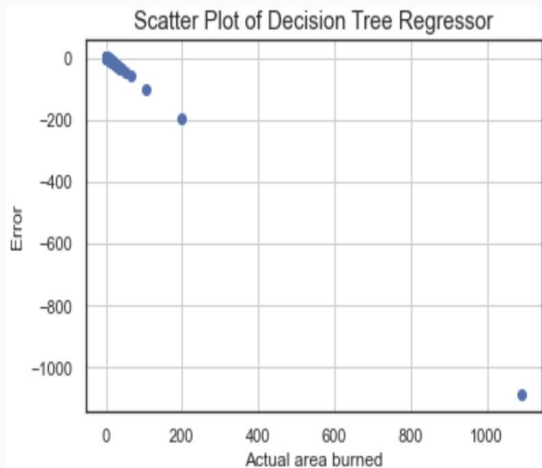
RMSE for Support Vector Regression: 0.6155174253148493

Result of Support vector regression(Scatter plot, Histogram, REC curve)

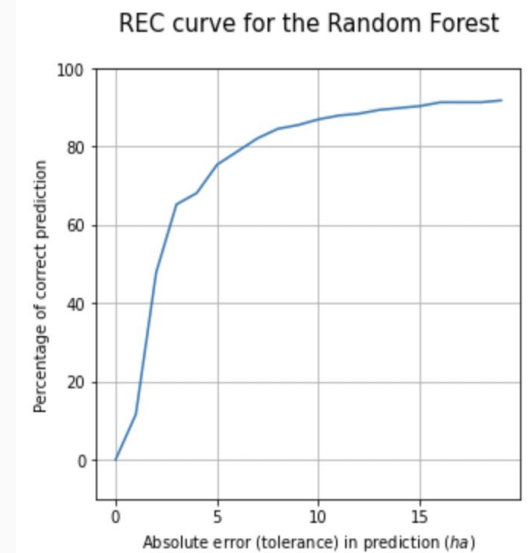
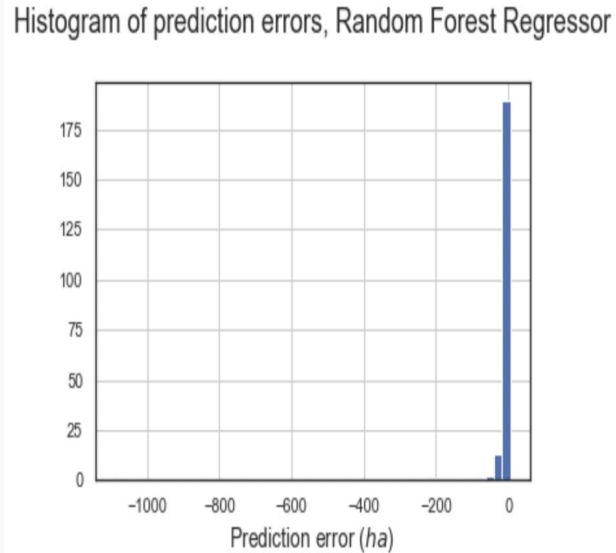
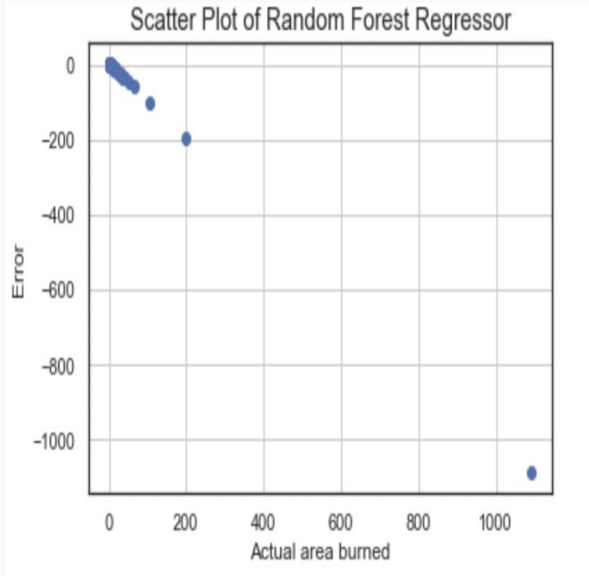


REC curves plot the error tolerance on the x-axis versus the percentage of points predicted within the tolerance on the y-axis.

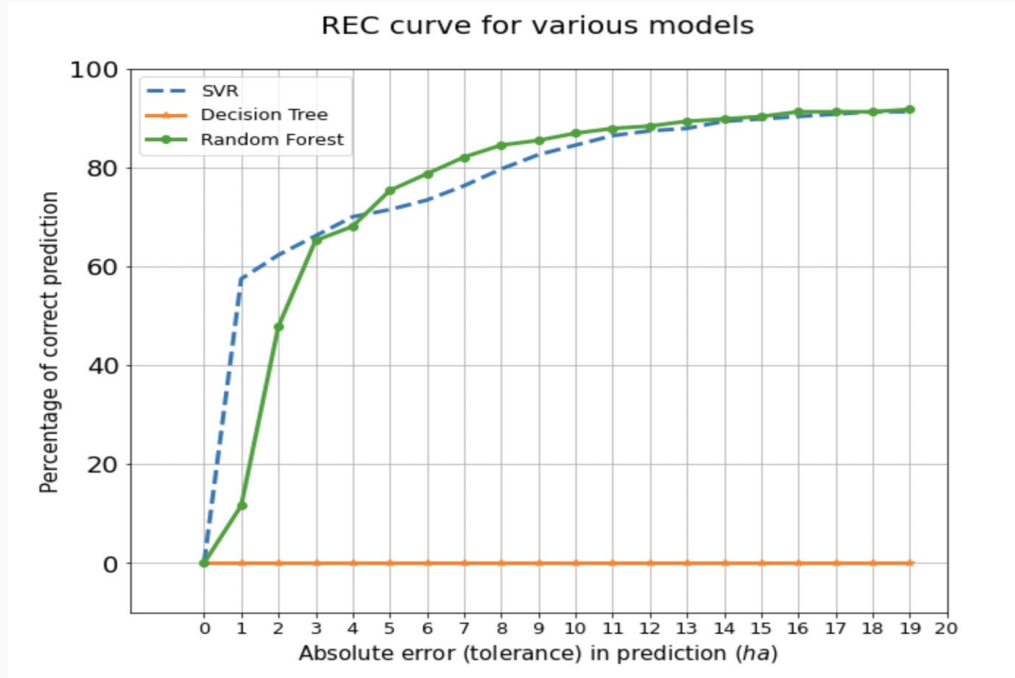
Result of Decision Tree regression(Scatter plot, Histogram, REC curve)



Result of Random Forest regression(Scatter plot, Histogram, REC curve)



REC curve for various models



Technologies & Tools

- Python
- Google Colab
- Pandas, NumPy, scikit-learn

Discussion and Conclusion

Models with Good Accuracy

Logistic Regression: Logistic Regression Accuracy, 95.58%

Neural Network Classifier: Neural Network Classifier Accuracy, 92.81%

Support Vector Regressor with GRIDCV: RMSE for Support Vector Regression gives accuracy of 0.61

Random Forest Regressor with GRIDCV: RMSE for Random Forest gives accuracy of 0.56, it means that model can relatively predict the data accurately

Conclusion

- The main objective of the study is to **use reliable prediction techniques with other data mining techniques** that are used in forest fire predictions
- The **Random Forest and Support Vector models** performed better than the other regression models
- The findings show that **Random Forest and Support Vector models have great potential for applications** in forest fire automatic **precaution and prevention** systems
- These methods can provide **important techniques for forest firefight decision making**

References

1. P. Jain et al, "A review of machine learning applications in wildfire science and management, " in *environmental reviews*, 28(4), pp. 478-505, 2020.
[Available]<https://doi.org/10.1007/s00521-018-3515-0>
2. Ying Xie, Minggang Peng, "Forest fire forecasting using ensemble learning approaches," in *Neural Computing and Applications*, pp. 4541-4550, 2019.
3. R. Xu et al, " A Forest Fire Detection System Based on Ensemble Learning," *Forests* 2021, 12, 217.

Thank You