The Use of AI & Machine Learning with Classifying & Predicting Types of Social Media Posts

Ciaran O'Dwyer - S15111608

CMP7161 Advanced Data Science A S2 2018/9

Dr. Yevgeniya Kovalchuk

24/05/2019

Table of Contents

Table of Figures

Table of Tables

Abstract

For this project, machine learning techniques were used in order to predict types of social media posts on Facebook by using their interaction data as features. Two machine learning algorithms were used, these being decision trees and support machine vectors. While both produced high levels of accuracy, the poor weighting of the data produced lower levels of accuracy than desired. A dataset that contains more entries for different types of posts would be recommended for this project.

Introduction

Social media plays a major role in modern life, with over “51.5% of the US population” using or owning a Facebook account (eMarketer, 2018). Facebook operates by users posting personal content which other users can react and respond too. With this, many users and companies aim to maximise their popularity and coverage of their posts. Using a dataset of Facebook posts, responses to posts can be measured and classified in order to classify what post is gaining popularity and attention just by classifying the “likes”, “Comments”, and “shares”.

Background

Facebook has a large reach on the population, meaning that it is used as a major tool for advertising. As documented within “research on effectiveness of Facebook…” (Dehghani et al, 2015), “Facebook advertising significantly affected brand image...”. Using Facebook, companies have the potential advantage of advertising to a larger, targeted audience with only a fraction of the cost compared to physical means of advertising. With this, understanding what type of social media posts are popular and classifying the type of posts would provide companies with a greater advantage in order to understand how to advertise their brand.

Gaining an understanding of what type of social media posts perform well can also assist general users as well. Within “Power of consumers…” (Kim et al, 2016), positive responses from consumers are created with positive content is created. Gaining an understanding of what creates the best type of positive content may assist users with creating posts of their own that they wish to gain attention and exposure, from invites to events to other important occasions.

Project Aim

The aim of this project is to create several machine learning AI’s that will utilise a Facebook dataset. Using this dataset, these AI’s will assess the attention of a large amount of Facebook posts and using this, attempt to classify the kind of post that was created. These will include photos, text, videos and links to external sources. These AI’s will then be evaluated using a variety of techniques in order to conclude the potential and usability of this classification approach.

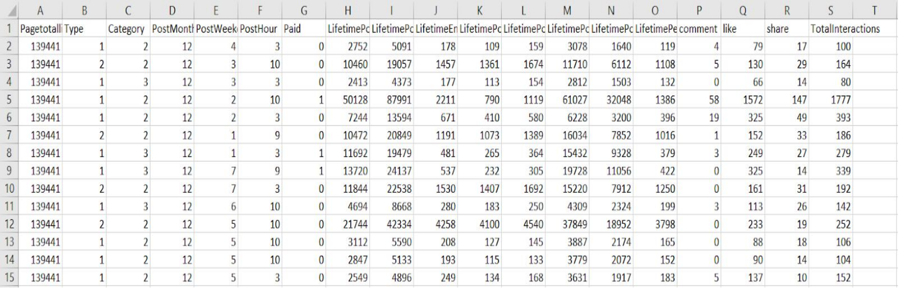
Objectives

This project has the following objectives:

- Gather and prepare a dataset of appropriate Facebook social media data.
- Develop a range of Algorithms to classify the data.
- Measure the accuracy of the outputted data with a consistent format.
- Compare the accuracy measures of the algorithms.
- Determine the effectiveness of the AI, the best choice of AI and the potential of AI usage within social media predictions.

Dataset Description

The dataset used for this project is the “Facebook comment volume dataset” (Archive.ics.uci.edu, 2016). This dataset contains approximately 500 Facebook posts, each containing aspects such as Post type, posting times, interactions and various other aspects to indicate the popularity of the posts. The post type is displayed as a number, with each number indicating a type of post. A “1” indicates the post was a photo. A “2” indicates the post was a text status. A “3” indicates that the post was an external link. A “4” indicates that the post was a video. The dataset initially contained 501 entries but 1 was removed due to it missing data. Within this dataset, there were 424 photos, 48 text statuses, 22 links and 7 videos. This initial data may lead to some balance issues.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Pagetotall	Type	Category	PostMont	PostWeek	PostHour	Paid	LifetimePc	LifetimePc	LifetimeEn	LifetimePc	LifetimePc	LifetimePc	LifetimePc	LifetimePe	comment	like	share	TotalInteractions	
2	139441	1	2	12	4	3	0	2752	5091	178	109	159	3078	1640	119	4	79	17	100	
3	139441	2	2	12	3	10	0	10460	19057	1457	1361	1674	11710	6112	1108	5	130	29	164	
4	139441	1	3	12	3	3	0	2413	4373	177	113	154	2812	1503	132	0	66	14	80	
5	139441	1	2	12	2	10	1	50128	87991	2211	790	1119	61027	32048	1386	58	1572	147	1777	
6	139441	1	2	12	2	3	0	7244	13594	671	410	580	6228	3200	396	19	325	49	393	
7	139441	2	2	12	1	9	0	10472	20849	1191	1073	1389	16034	7852	1016	1	152	33	186	
8	139441	1	3	12	1	3	1	11692	19479	481	265	364	15432	9328	379	3	249	27	279	
9	139441	1	3	12	7	9	1	13720	24137	537	232	305	19728	11056	422	0	325	14	339	
10	139441	2	2	12	7	3	0	11844	22538	1530	1407	1692	15220	7912	1250	0	161	31	192	
11	139441	1	3	12	6	10	0	4694	8668	280	183	250	4309	2324	199	3	113	26	142	
12	139441	2	2	12	5	10	0	21744	42334	4258	4100	4540	37849	18952	3798	0	233	19	252	
13	139441	1	2	12	5	10	0	3112	5590	208	127	145	3887	2174	165	0	88	18	106	
14	139441	1	2	12	5	10	0	2847	5133	193	115	133	3779	2072	152	0	90	14	104	
15	139441	1	2	12	5	3	0	2549	4896	249	134	168	3631	1917	183	5	137	10	152	

Figure 1: The Facebook comment dataset (Archive.ics.uci.edu, 2016)

Problem to be Addressed

The problem to be addressed is “Is it possible to classify types of Facebook social media posts using post interactions?”. Using these interaction variables, various classification, supervised learning approaches will be applied in order to see if the type of post is able to be classified.

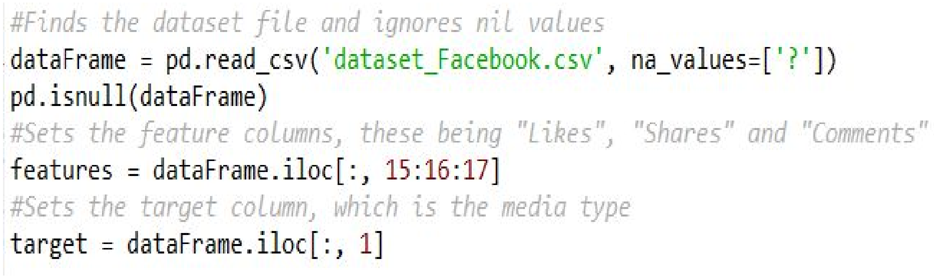
Decision Tree

Approach Summary

Decision trees are a supervised method of learning that can be used for classification and linear regression machine learning problems (Scikit-learn.org, 2019). This model predicts variables by implementing simple decision rules formed by the feature variables in order to create a prediction variable. A tree is then created using this data to display the various decision rules that the tree has generated to form its predictions.

Data Pre-Processing, visualisation and feature selection

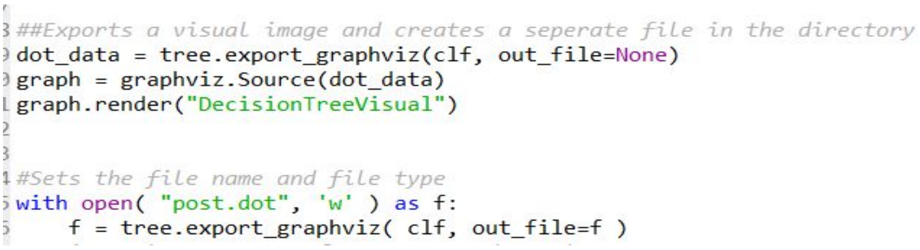
In order for the data set to be analysed, the data must be in ".csv" format in order for it to be read by the program. To assist with data processing as well, nil entries are removed from the dataset. The selected features of the dataset are "Likes", "Comments" and "Shares". These are located in the 15th, 16th and 17th data columns. For the target variable, the "Type" variable is selected, which is located in the 2nd column. The code for this operation is displayed below.



```
#Finds the dataset file and ignores nil values
dataFrame = pd.read_csv('dataset_Facebook.csv', na_values=['?'])
pd.isnull(dataFrame)
#Sets the feature columns, these being "Likes", "Shares" and "Comments"
features = dataFrame.iloc[:, 15:16:17]
#Sets the target column, which is the media type
target = dataFrame.iloc[:, 1]
```

Figure 2: Decision Tree Pre-Processing code

In order to visualise the data that had been created, data that had been created was outputted to a separate external file that would be created every time the program was run. This file would contain the created decision tree that would display the logic that the AI had used in order to create its predictions. The method for this operation is demonstrated below.

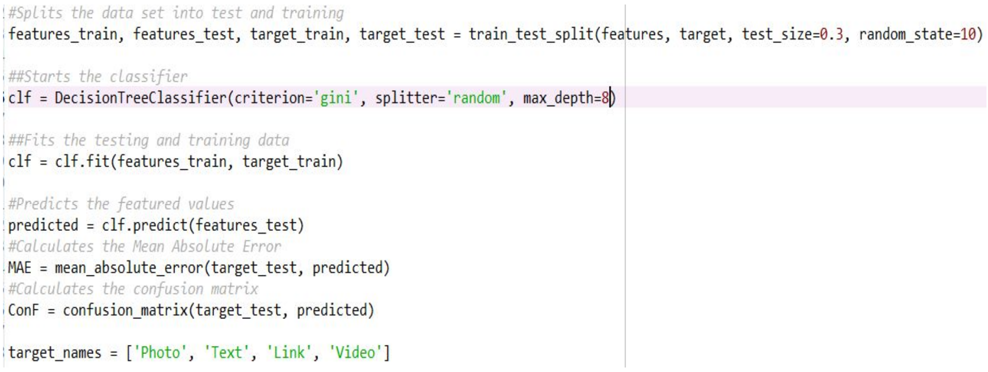


```
##Exports a visual image and creates a seperate file in the directory
dot_data = tree.export_graphviz(clf, out_file=None)
graph = graphviz.Source(dot_data)
graph.render("DecisionTreeVisual")


#Sets the file name and file type
with open( "post.dot", 'w' ) as f:
    f = tree.export_graphviz( clf, out_file=f )
```

Figure 3: Decision Tree visualisation code

Model training, evaluation and testing

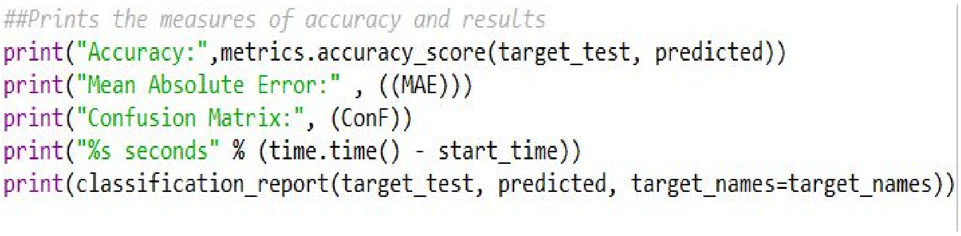
For training the model and data processing, 500 samples from the data set were used. These features and targets from the data sets were then split into training and testing sets. The test size utilised 30% of the dataset and a random seed of 10 was set in order for different parts of the dataset to be utilised. The feature and target training data were then fitted to the decision tree classifier while the test features data was then used to create the predicted values. For the decision tree, the max depth was set to 8 and the splitter was set to random to look for the best random split. Code for this operation is displayed below.



```
#Splits the data set into test and training
features_train, features_test, target_train, target_test = train_test_split(features, target, test_size=0.3, random_state=10)

##Starts the classifier
clf = DecisionTreeClassifier(criterion='gini', splitter='random', max_depth=8)

##Fits the testing and training data
clf = clf.fit(features_train, target_train)

#Predicts the featured values
predicted = clf.predict(features_test)
#Calculates the Mean Absolute Error
MAE = mean_absolute_error(target_test, predicted)
#Calculates the confusion matrix
ConF = confusion_matrix(target_test, predicted)

target_names = ['Photo', 'Text', 'Link', 'Video']
```

Figure 4: Decision Tree model training code

To evaluate the accuracy of the created model, several units of measurement were incorporated to give an accurate estimate. The “metrics.accuracy score” module was implemented in order to give a simple numerical score. Other methods to generate accuracy measures were “Mean Absolute Error”, “Confusion Matrix” and a “Classification report”. A decision tree is also created to display the decision logic as previously highlighted. This code is shown below.

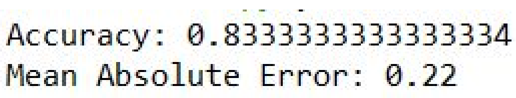


```
##Prints the measures of accuracy and results
print("Accuracy:",metrics.accuracy_score(target_test, predicted))
print("Mean Absolute Error:" , ((MAE)))
print("Confusion Matrix:", (ConF))
print("%s seconds" % (time.time() - start_time))
print(classification_report(target_test, predicted, target_names=target_names))
```

Figure 5: Decision Tree evaluation code

Results and Discussion

As previously stated, various methods of measurement were used in order to determine how effective the machine learning technique is. The first measures used were accuracy score and “Mean Absolute Error”(MAE). Accuracy is measured via a percentage and MAE, a risk metric which is more accurate the closer to 0 it is.



```
Accuracy: 0.8333333333333334
Mean Absolute Error: 0.22
```

Figure 6: Decision Tree Accuracy results

As shown within the initial accuracy measures, the decision tree shows to be quite accurate, with an 83.4% accuracy. While these initial results are promising, various other measures of accuracy show the flaws with the machine learning approach. These measures were the confusion matrix, the classification report and the decision tree.

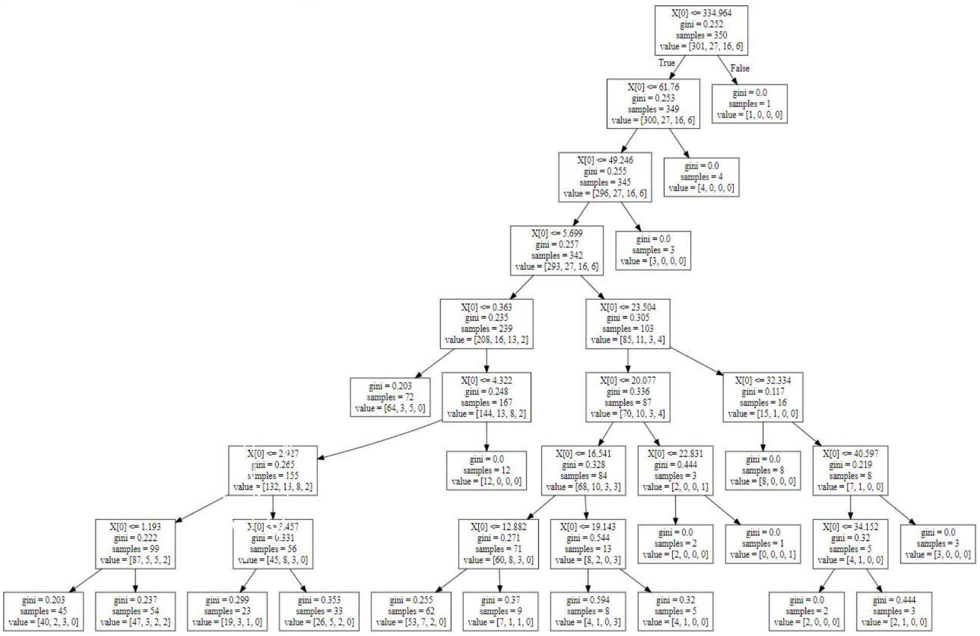
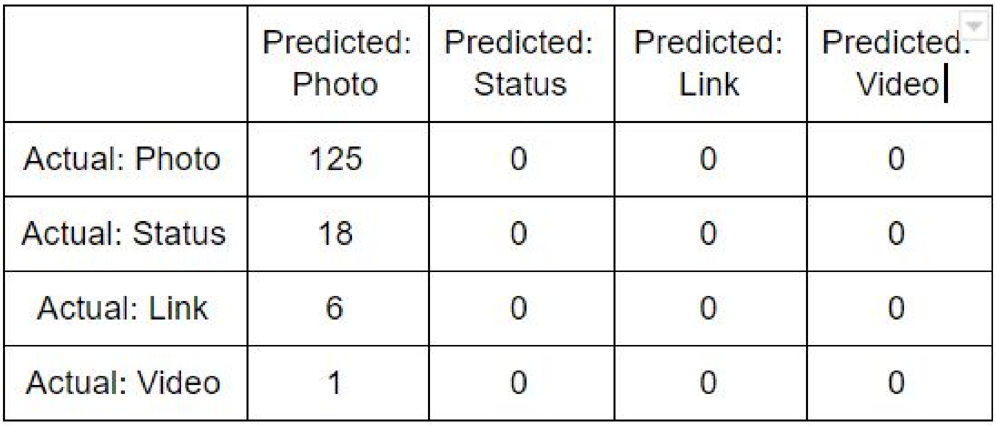
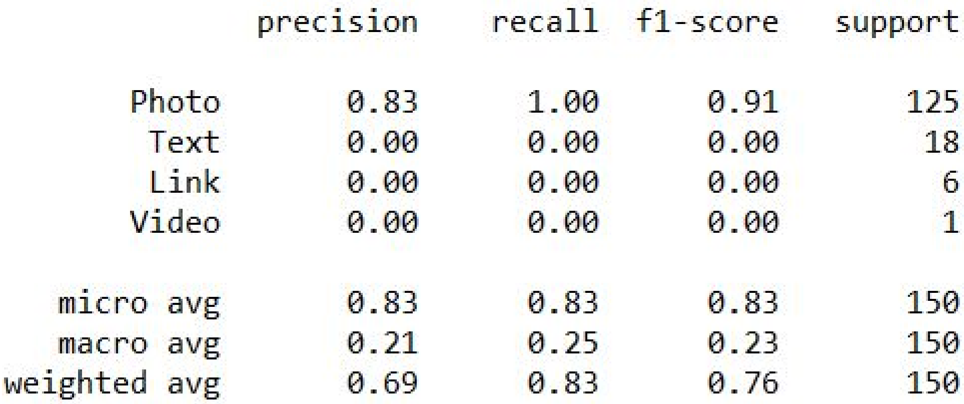

X[0] <= 334.964 gini = 0.252 samples = 350 value = [301, 27, 16, 6]
True
False
X[0] <= 61.76 gini = 0.253 samples = 349 value = [300, 27, 16, 6]
gini = 0.0 samples = 1 value = [1, 0, 0, 0]
X[0] <= 49.246 gini = 0.255 samples = 345 value = [296, 27, 16, 6]
gini = 0.0 samples = 4 value = [4, 0, 0, 0]
X[0] <= 5.699 gini = 0.257 samples = 342 value = [293, 27, 16, 6]
gini = 0.0 samples = 3 value = [3, 0, 0, 0]
X[0] <= 0.363 gini = 0.235 samples = 239 value = [208, 16, 13, 2]
X[0] <= 23.504 gini = 0.305 samples = 103 value = [85, 11, 3, 4]
gini = 0.203 samples = 72 value = [64, 3, 5, 0]
X[0] <= 4.322 gini = 0.248 samples = 167 value = [144, 13, 8, 2]
X[0] <= 20.077 gini = 0.336 samples = 87 value = [70, 10, 3, 4]
X[0] <= 32.334 gini = 0.117 samples = 16 value = [15, 1, 0, 0]
X[0] <= 2.927 gini = 0.265 samples = 155 value = [132, 13, 8, 2]
gini = 0.0 samples = 12 value = [12, 0, 0, 0]
X[0] <= 16.541 gini = 0.328 samples = 84 value = [68, 10, 3, 3]
X[0] <= 22.831 gini = 0.444 samples = 3 value = [2, 0, 0, 1]
gini = 0.0 samples = 8 value = [8, 0, 0, 0]
X[0] <= 40.597 gini = 0.219 samples = 8 value = [7, 1, 0, 0]
X[0] <= 1.193 gini = 0.222 samples = 99 value = [87, 5, 5, 2]
X[0] <= 3.457 gini = 0.331 samples = 56 value = [45, 8, 3, 0]
X[0] <= 12.882 gini = 0.271 samples = 71 value = [60, 8, 3, 0]
X[0] <= 19.143 gini = 0.544 samples = 13 value = [8, 2, 0, 3]
gini = 0.0 samples = 2 value = [2, 0, 0, 0]
gini = 0.0 samples = 1 value = [0, 0, 0, 1]
X[0] <= 34.152 gini = 0.32 samples = 5 value = [4, 1, 0, 0]
gini = 0.0 samples = 3 value = [3, 0, 0, 0]
gini = 0.203 samples = 45 value = [40, 2, 3, 0]
gini = 0.237 samples = 54 value = [47, 3, 2, 2]
gini = 0.299 samples = 23 value = [19, 3, 1, 0]
gini = 0.353 samples = 33 value = [26, 5, 2, 0]
gini = 0.255 samples = 62 value = [53, 7, 2, 0]
gini = 0.37 samples = 9 value = [7, 1, 1, 0]
gini = 0.594 samples = 8 value = [4, 1, 0, 3]
gini = 0.32 samples = 5 value = [4, 1, 0, 0]
gini = 0.0 samples = 2 value = [2, 0, 0, 0]
gini = 0.444 samples = 3 value = [2, 1, 0, 0]

Figure 7: Decision Tree, demonstrating the issue with the dataset.



	Predicted: Photo	Predicted: Status	Predicted: Link	Predicted: Video
Actual: Photo	125	0	0	0
Actual: Status	18	0	0	0
Actual: Link	6	0	0	0
Actual: Video	1	0	0	0

Table 1: Decision Tree confusion matrix data



```
              precision    recall  f1-score   support

       Photo       0.83      1.00      0.91       125
        Text       0.00      0.00      0.00        18
        Link       0.00      0.00      0.00         6
       Video       0.00      0.00      0.00         1

   micro avg       0.83      0.83      0.83       150
   macro avg       0.21      0.25      0.23       150
weighted avg       0.69      0.83      0.76       150
```

Figure 8: Decision Tree accuracy report

From these measurements, it can be clearly seen that there is an imbalance with the data. The confusion matrix and accuracy report show that while predictions of photo data proved to be accurate, there was little to no predictions for any other kind of post type. Another strong piece of evidence towards the notion of imbalanced data can be shown in the decision tree. Looking at the tree it can be seen that the depth is imbalanced, with the decision logic strongly leaning towards only photos again.

Conclusion and Recommendations

To conclude, while decision trees would be a suitable choice of machine learning with the current problem on hand, the imbalance of data creates an unreliable result that strongly leans towards photos. Several solutions to these imbalances include populating the data with random entries that follow the trend of data. Another solution is to increase the sample size of the test size in order to include more entries from the other limited types.

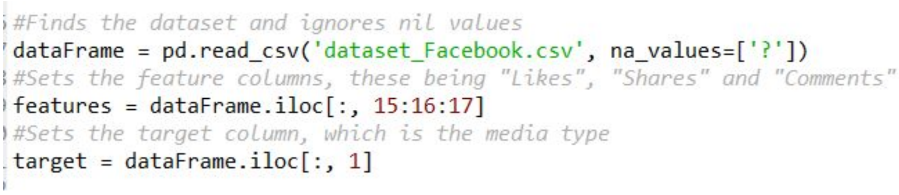
Support Machine Vector

Approach Summary

Support Machine Vectors (SMV) are a method of supervised learning which is primarily used for classification and regression machine learning problems. Predictions are created via training points called “support vectors”. This approach creates a simple graph showing how data is split amongst designed target classes in order to form predictions.

Data Pre-Processing, visualisation and feature selection

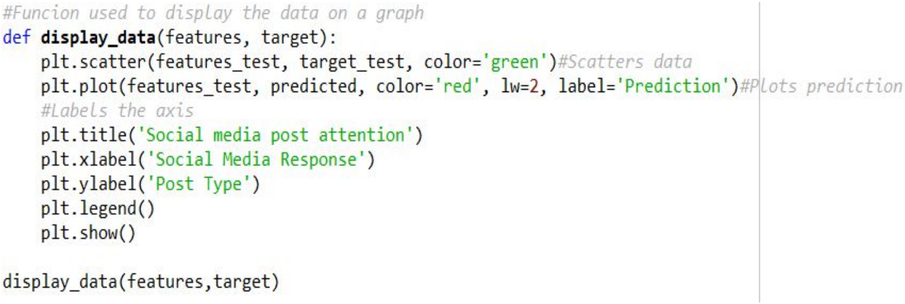
Similar in style to the decision tree pre-processing, the dataset is converted to “.csv” format and nil values are removed in order to allow for the data to be easily read. The selected features of the dataset were again “likes, “comments” and “shares”, with the post type being the target. These features were located in the 15th, 16th and 17th column respectively and the target variable is located in the 2nd column. Code demonstrating this is shown below.



```
#Finds the dataset and ignores nil values
dataFrame = pd.read_csv('dataset_Facebook.csv', na_values=['?'])
#Sets the feature columns, these being "Likes", "Shares" and "Comments"
features = dataFrame.iloc[:, 15:16:17]
#Sets the target column, which is the media type
target = dataFrame.iloc[:, 1]
```

Figure 9: SMV pre-processing code

For the visualisation of the data, a display data function was created in order to demonstrate the spread of the data. The chart is then generated within the kernel as opposed to the decision tree which outputs to an external file. The method for the operation is shown below.

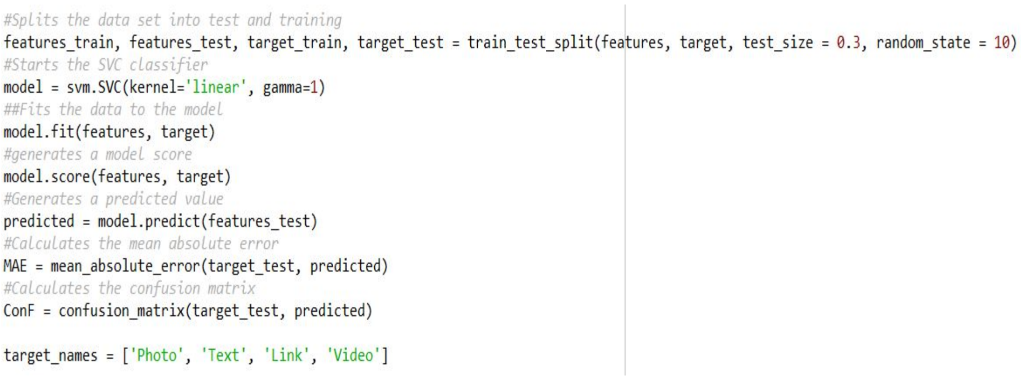


```
#Funcion used to display the data on a graph
def display_data(features, target):
    plt.scatter(features_test, target_test, color='green')#Scatters data
    plt.plot(features_test, predicted, color='red', lw=2, label='Prediction')#Plots prediction
    #Labels the axis
    plt.title('Social media post attention')
    plt.xlabel('Social Media Response')
    plt.ylabel('Post Type')
    plt.legend()
    plt.show()

display_data(features,target)
```

Figure 10: SMV visualisation code

Model training, evaluation and testing

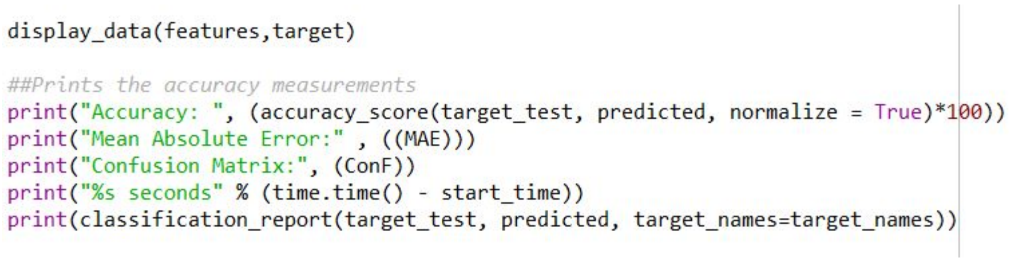
Training the model and data processing was conducted the same as the decision tree, utilising 500 samples from the data set. Features and targets from the data sets were once again split into training and testing sets, utilising 30% of the dataset. A random seed of 10 was also set again for consistency. A linear kernel was used and the gamma was set to 1 for the model. The feature and target training data were fitted to the SMV classifier with test data being used to create predictions that would be used for accurate calculations and so forth. The code used to perform this operation is displayed below.



```
#Splits the data set into test and training
features_train, features_test, target_train, target_test = train_test_split(features, target, test_size = 0.3, random_state = 10)
#Starts the SVC classifier
model = svm.SVC(kernel='linear', gamma=1)
##Fits the data to the model
model.fit(features, target)
#generates a model score
model.score(features, target)
#Generates a predicted value
predicted = model.predict(features_test)
#Calculates the mean absolute error
MAE = mean_absolute_error(target_test, predicted)
#Calculates the confusion matrix
ConF = confusion_matrix(target_test, predicted)

target_names = ['Photo', 'Text', 'Link', 'Video']
```

Figure 11: SMV model training code

In order to evaluate the accuracy of the model, the accuracy measurement units found in the decision tree code were used. The "metrics.accuracy score" module was once again implemented in order to create a simple numerical measure of accuracy. Methods that generate accuracy measures such as "Mean Absolute Error", "Confusion Matrix" and "Classification report" were also once again implemented. A graph showing the spread of data and the classification prediction is also created to demonstrate how data is spread. The code to perform these operations is displayed below.

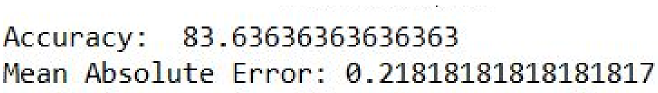


```
display_data(features,target)

##Prints the accuracy measurements
print("Accuracy: ", (accuracy_score(target_test, predicted, normalize = True)*100))
print("Mean Absolute Error:" , ((MAE)))
print("Confusion Matrix:", (ConF))
print("%s seconds" % (time.time() - start_time))
print(classification_report(target_test, predicted, target_names=target_names))
```

Figure 12: SMV evaluation code

Results and Discussion

Once again, the stated measurements were used in order to understand the accuracy and viability of the program. The first measures were again, accuracy score and "Mean Absolute Error"(MAE). Results are displayed below.



```
Accuracy:  83.63636363636363
Mean Absolute Error: 0.21818181818181817
```

Figure 13: SMV accuracy results

As shown in the accuracy results, a percentage of 83.63% was reached with a low number of 0.21… for the MAE. Once again, these initial results show that this method of machine learning may have potential with the problem we are addressing, obtaining better scores than the decision tree machine learning method. However, the remaining results obtained from the confusion matrix, accuracy report and the generated graph show the problem with the dataset and model. These results are shown below.

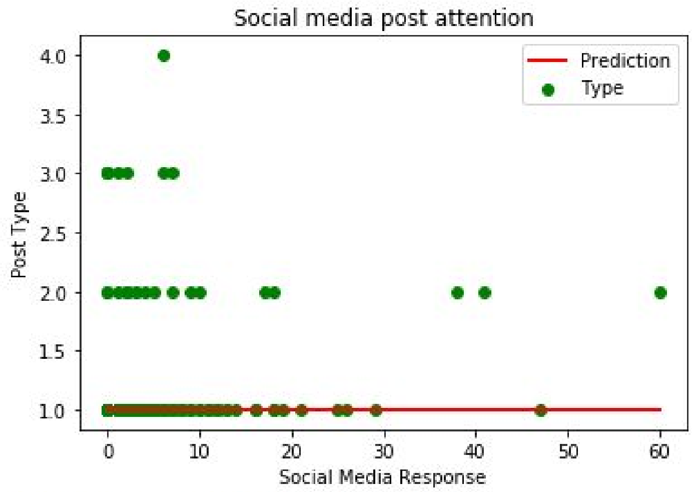
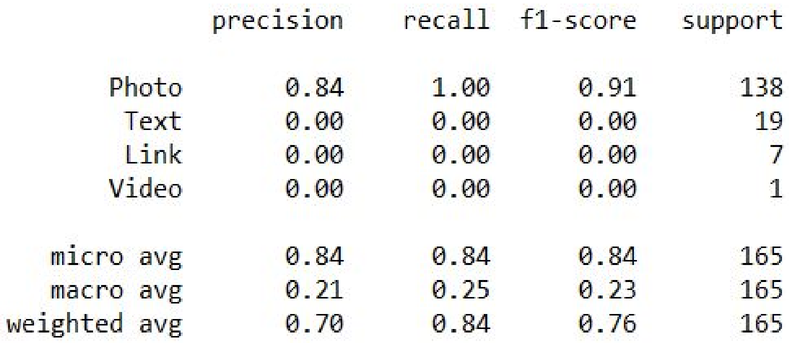


Figure 13: SMV data chart, showing data imbalance

	Predicted: Photo	Predicted: Status	Predicted: Link	Predicted: Video
Actual: Photo	125	0	0	0
Actual: Status	18	0	0	0
Actual: Link	6	0	0	0
Actual: Video	1	0	0	0

Table 2: SMV confusion matrix data



```
              precision    recall  f1-score   support

       Photo       0.84      1.00      0.91       138
        Text       0.00      0.00      0.00        19
        Link       0.00      0.00      0.00         7
       Video       0.00      0.00      0.00         1

   micro avg       0.84      0.84      0.84       165
   macro avg       0.21      0.25      0.23       165
weighted avg       0.70      0.84      0.76       165
```

Figure 14: SMV accuracy report

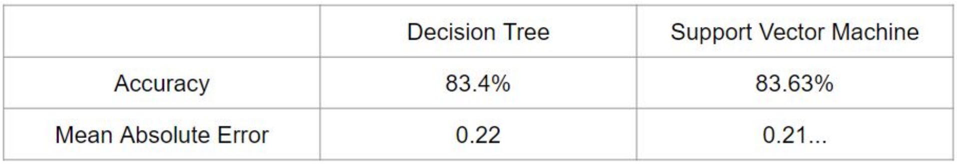
Once again, these measurements clearly show the imbalance with the dataset. The graph shows that the photos type has several more entries and is the most probable to be predicted when classifying the dataset. The confusion matrix also follows this logic, with photos constantly being predicted and while they are mostly being correctly predicted, there are no other predictions of any other kind. The accuracy report also serves as evidence for this disproportionate data. Only photos are recalled, with no other types getting any mention in the report.

Conclusion and Recommendations

To conclude, while SMV did provide more accurate results, the data imbalance was even more noticeable than the decision trees model. This strong lean towards predicting photos may be fixed if the dataset were to be balanced as well. This may be achieved with repopulating the dataset with random entries based off the data, but with so little data entries to base these off, inaccurate results may be caused.

Results, Comparison and Discussion

Overall, while both of the machine learning techniques produced seemingly accurate results, SMV proved to be the better form of machine learning for the task, producing just slightly more accurate results. These results are shown side by side below.



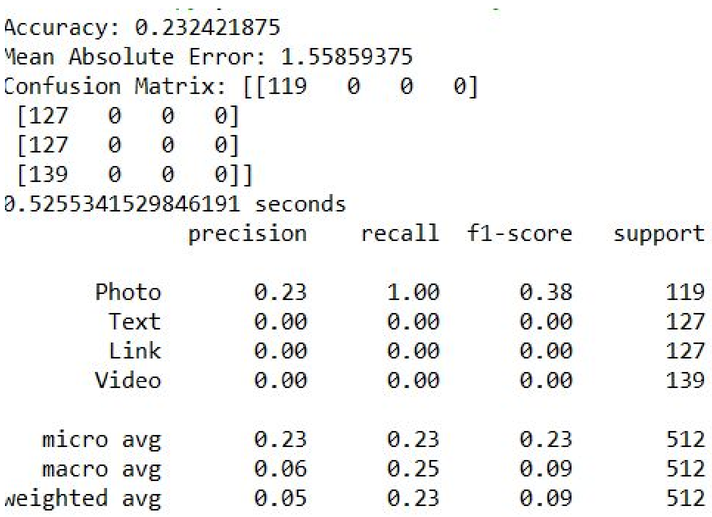
	Decision Tree	Support Vector Machine
Accuracy	83.4%	83.63%
Mean Absolute Error	0.22	0.21...

Table 3: Side by side comparison of models.

As shown in the table above, there's a small difference of 0.23% accuracy between the two models. The MAE also has a difference of approximately 0.01. These statistic differences show that SMV is a more accurate method of classifying and predicting. However, the problems with the dataset and imbalance of data mean that these results aren't as reliable as anticipated. With the dataset having approximately 424 entries that are photos out of the 500 entries, this leaves little room for any other data to be inputted, causing these models to heavily lean towards classifying and predicting towards only one type of post. However, this could also potentially indicate that perhaps photo posts are the most popular type of posts by a significant amount, meaning that the models are correct to predict that posts with the most attention are photos.

Conclusion and Recommendations

In conclusion, there is some potential in using the popularity of a post to determine what kind of post it may be. However, due to a lack of data balancing, the reliability of the developed methods of this project doesn't provide results that are as promising as expected. Between these two methods of machine learning, SMV proved to be the better model as opposed to decision trees. This was only by a small number of accuracy though as both models did provide accurate scores. Both did lean heavily towards predicting photos, either due to the dataset being imbalanced or due to photos being the most popular type of content to post. Taking this project forward would require for either a new dataset that contains a larger range of different types of posts or new balancing methods would have to be implemented. For example, SMOTE (Imbalanced-learn.readthedocs.io, 2019), a method that populates data and auto balances could be included as part of the project in order to fix these data issues. However, the usage of this would have to be controlled due to the possibility of unreliable data being created. Initial experiments with this method provided highly inaccurate results as shown in the figure below.



```
Accuracy: 0.232421875
Mean Absolute Error: 1.55859375
Confusion Matrix: [[119   0   0   0]
 [127   0   0   0]
 [127   0   0   0]
 [139   0   0   0]]
0.5255341529846191 seconds
              precision    recall  f1-score   support

       Photo       0.23      1.00      0.38       119
        Text       0.00      0.00      0.00       127
        Link       0.00      0.00      0.00       127
       Video       0.00      0.00      0.00       139

   micro avg       0.23      0.23      0.23       512
   macro avg       0.06      0.25      0.09       512
weighted avg       0.05      0.23      0.09       512
```

Figure 15: Quick look at SMOTE results produced using the decision tree code.

There may be some potential in this method with some modification or using different models. It would be recommended to perform more experimentation with the splits of testing and training data in the future as well in order to achieve higher accuracy with the models.

References

Archive.ics.uci.edu. (2016). UCI Machine Learning Repository: Facebook Comment Volume Dataset Data Set. [online] Available at: http://archive.ics.uci.edu/ml/datasets/Facebook+Comment+Volume+Dataset# [Accessed 15 May 2019].

Dehghani, M. and Tumer, M., 2015. A research on effectiveness of Facebook advertising on enhancing purchase intention of consumers. Computers in Human Behavior, 49, pp.597-600.

eMarketer. (2018). How Many People Use Facebook in the US 2018 - eMarketer Trends, Forecasts and Statistics. [online] Available at: https://www.emarketer.com/content/the-social-series-who-s-using-facebook [Accessed 15 May 2019].

Kim, A.J. and Johnson, K.K., 2016. Power of consumers using social media: Examining the influences of brand-related user-generated content on Facebook. Computers in Human Behavior, 58, pp.98-108.

Imbalanced-learn.readthedocs.io. (2019). imblearn.over_sampling.SMOTE — imbalanced-learn 0.4.3 documentation. [online] Available at: https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.over_sampling.SMOTE.html [Accessed 20 May 2019].

Scikit-learn.org. (2019). 1.4. Support Vector Machines — scikit-learn 0.21.1 documentation. [online] Available at: https://scikit-learn.org/stable/modules/SMV.html#SMV-classification [Accessed 16 May 2019].

Scikit-learn.org. (2019). 1.10. Decision Trees — scikit-learn 0.21.1 documentation. [online] Available at: https://scikit-learn.org/stable/modules/tree.html [Accessed 15 May 2019].