Twitter Dashboard Project on ElonMusk

Allen Tianyuan Xu
Xinzhen Fan

Contents

Introduction

Elon Musk is a business magnate and investor

- the founder, CEO and chief engineer of SpaceX
- angel investor, CEO and product architect of Tesla, Inc
- owner and CEO of Twitter
- founder of the Boring Company
- co-founder of Neuralink and OpenAI
- President of the philanthropic Musk Foundation

Dataset

	path	name
1	dbfs:/mnt/project/ElonMusk/2022/11/21/16/	16/
2	dbfs:/mnt/project/ElonMusk/2022/11/21/17/	17/
3	dbfs:/mnt/project/ElonMusk/2022/11/21/18/	18/
4	dbfs:/mnt/project/ElonMusk/2022/11/21/19/	19/
5	dbfs:/mnt/project/ElonMusk/2022/11/21/20/	20/
6	dbfs:/mnt/project/ElonMusk/2022/11/21/21/	21/
7	dbfs:/mnt/project/ElonMusk/2022/11/21/22/	22/

- Data on 2022.11.21 from 16h to 22h
- 328766 tweets containing ElonMusk
- Using textblob to create sentiment column

Table +

	id	user_name	screen_name	text
1	1594755513945755648	J.E. Dyer	OptimisticCon	Keep in mind, "Trump banned on Twitter!" is an integr https://t.co/ov7iRF3YSn
2	1594755515485143052	Javier Perdomo	Javierperdomo	RT @MattGertz: Elon Musk interacting with sycophant
3	1594755517574164481	Casey Reilley	caseyreilley	RT @MattGertz: Elon Musk interacting with sycophant
4	1594755519868043264	Val Ornelas	_surfcowgirl	RT @elizableu: I'd like to make something else clear, I etc. I ru...
5	1594755519981436961	Name Can't be Blank	adrenaline1073	RT @disclosetv: JUST IN - Elon Musk has reinstated Re
6	1594755520526778368	The Original Johnboy usuLtRA MAgAus #WPS	johnboy02131989	RT @BehizyTweets: BREAKING: Elon Musk just reinstat
7	1594755520790937611	Gideon Henry us	GideonHenry	RT @w_terrence: Elon Musk should purchase the ric

```
def get_sentiment(text):
    blob = TextBlob(text)
    sentiment = blob.sentiment.polarity
    if sentiment > 0:
        return 'positive'
    elif sentiment < 0:
        return 'negative'
    else:
        return 'neutral'
```

EDA

null_counts: pyspark.sql.dataframe.DataFrame = [user_name: long, screen_name: long ... 5 more fields]

user_name	screen_name	text	followers_count	location	geo	created_at
102	100	204	444	446	551	549

```
# handle null value in each column
df_no_nulls = df.dropna(subset=["user_name", "screen_name", "text", "created_at"])

mean_followers_count = df_no_nulls.agg({"followers_count": "mean"}).collect()[0][0]

new_df = (
    df_no_nulls.fillna("Unknown", subset=["location", "geo"])
               .fillna(mean_followers_count, subset=["followers_count"])
)

new_df.show()
```

```
*,
CASE
    WHEN location LIKE '%California%' OR location LIKE '%CA%' OR location LIKE '%Los Angeles%' OR location LIKE '%San Francisco%' THEN 'California'
    WHEN location LIKE '%Texas%' OR location LIKE '%TX%' OR location LIKE '%Houston%' OR location LIKE '%Dallas%' THEN 'Texas'
    WHEN location LIKE '%Florida%' OR location LIKE '%FL%' OR location LIKE '%Miami%' OR location LIKE '%Orlando%' THEN 'Florida'
    WHEN location LIKE '%New York%' THEN 'New York'
    WHEN location = 'United States' THEN 'USA'
    ELSE location
END as new_location,
CASE
    WHEN followers_count BETWEEN 0 AND 1000 THEN '0 - 1,000'
    WHEN followers_count BETWEEN 1001 AND 10000 THEN '1,001 - 10,000'
    WHEN followers_count BETWEEN 10001 AND 100000 THEN '10,001 - 100,000'
    WHEN followers_count BETWEEN 100001 AND 1000000 THEN '100,001 - 1,000,000'
    ELSE '1,000,001+'
```

Spark ML

Logistic Regression

```
from pyspark.ml.classification import LogisticRegression

# Use 80% cases for training, 20% cases for testing
train, test = tweets_label.randomSplit([0.8, 0.2], seed=42)

lr = LogisticRegression(maxIter=100)

lr_model = lr.fit(train)

lr_predictions = lr_model.transform(test)

display(lr_predictions)
```

Decision Tree

```
from pyspark.ml.classification import DecisionTreeClassifier

# Use 80% cases for training, 20% cases for testing
train, test = tweets_label.randomSplit([0.8, 0.2], seed=42)

dt = DecisionTreeClassifier(maxDepth=10)

dt_model = dt.fit(train)

dt_predictions = dt_model.transform(test)

display(dt_predictions)
```

```
evaluator = MulticlassClassificationEvaluator(predictionCol="prediction", metricName="f1")
```

Accuracy Score: 0.9560
F1 Score: 0.9560

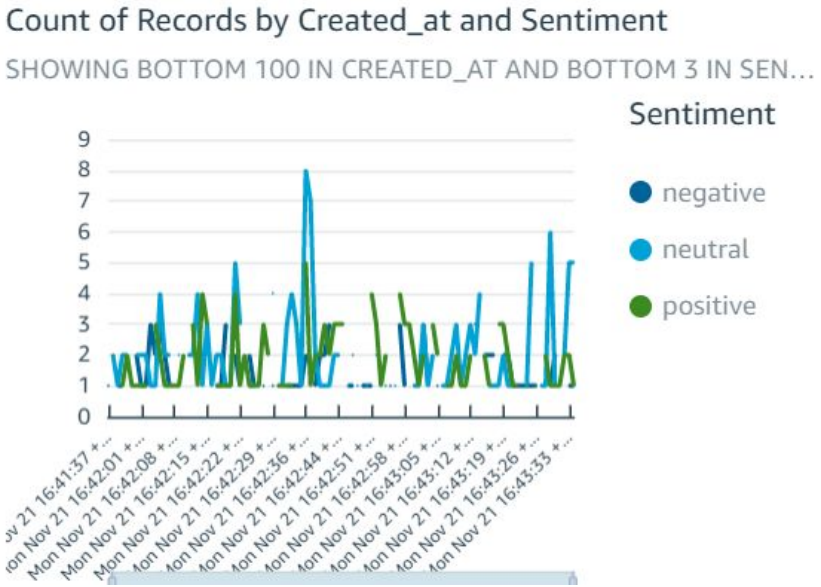
Accuracy Score: 0.6101
F1 Score: 0.5671

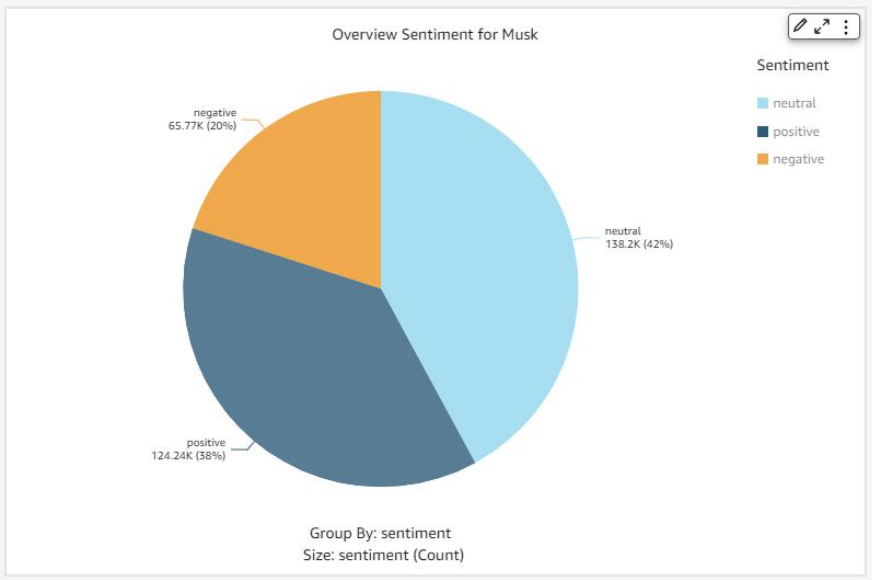
Spark ML Pipeline

```
# Use 80% cases for training, 20% cases for testing
train, test = tweets_clean.randomSplit([0.8, 0.2], seed=42)

# Create transformers for the ML pipeline
tokenizer = Tokenizer(inputCol="tweet", outputCol="tokens")
stopword_remover = StopWordsRemover(inputCol="tokens", outputCol="filtered")
cv = CountVectorizer(vocabSize=2**16, inputCol="filtered", outputCol='cv')
idf = IDF(inputCol='cv', outputCol="1gram_idf", minDocFreq=5) #minDocFreq: remove sparse terms
assembler = VectorAssembler(inputCols=["1gram_idf"], outputCol="features")
label_encoder= StringIndexer(inputCol = "sentiment", outputCol = "label")
lr = LogisticRegression(maxIter=100)
pipeline = Pipeline(stages=[tokenizer, stopword_remover, cv, idf, assembler, label_encoder, lr])

pipeline_model = pipeline.fit(train)
predictions = pipeline_model.transform(test)

evaluator = MulticlassClassificationEvaluator(predictionCol="prediction", metricName="f1")
accuracy = predictions.filter(predictions.label == predictions.prediction).count() / float(test.count())
f1_score = evaluator.evaluate(predictions)

print("Accuracy Score: {0:.4f}".format(accuracy))
print("F1 Score: {0:.4f}".format(f1_score))
```

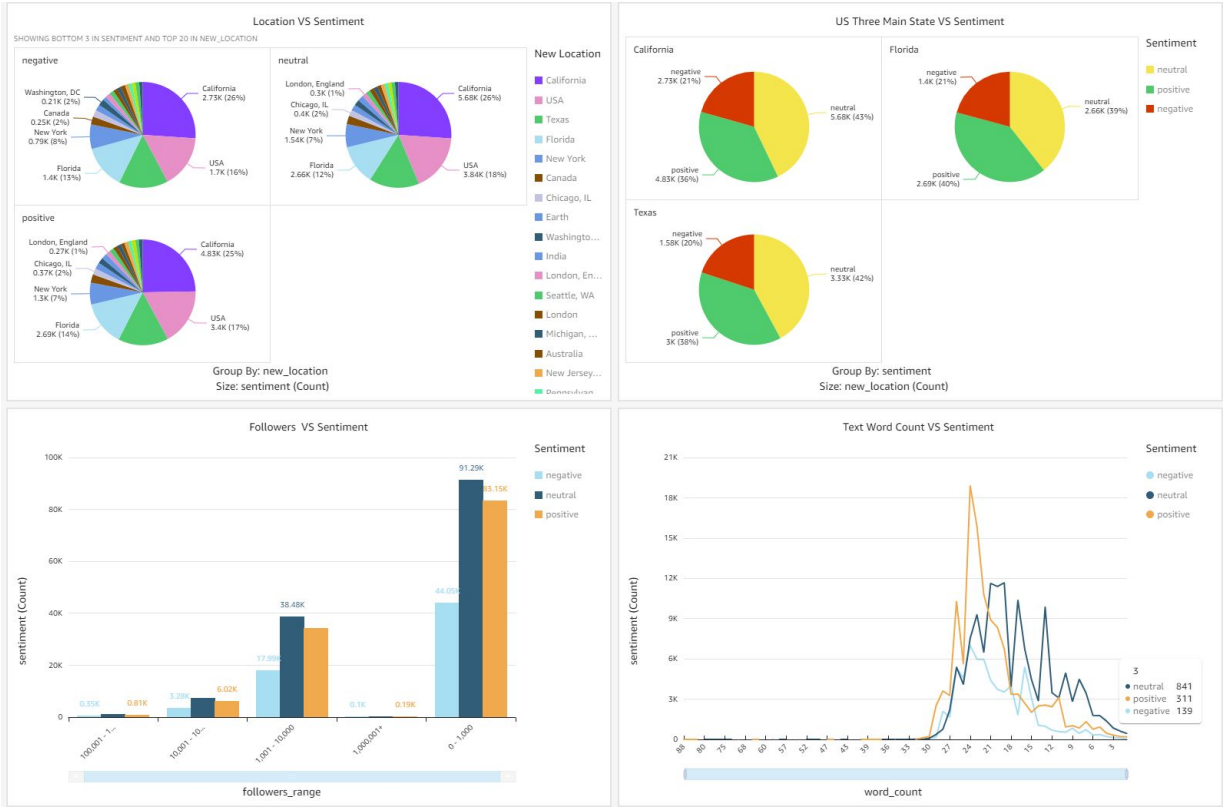
```
Accuracy Score: 0.9595
F1 Score: 0.9595
```

Quicksight: Preliminary Dashboard

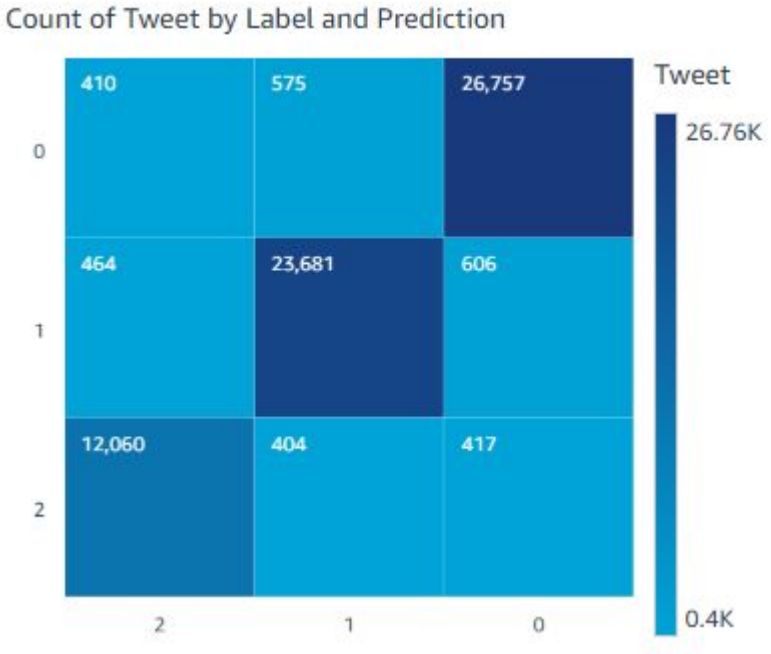


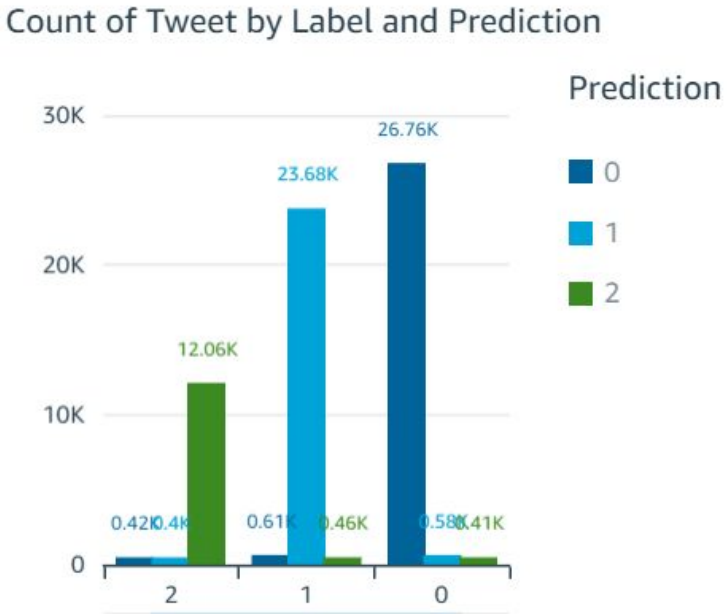


Quicksight


Location VS Sentiment
SHOWING BOTTOM 3 IN SENTIMENT AND TOP 20 IN NEW_LOCATION
negative
neutral
positive
New Location
California
USA
Texas
Florida
New York
Canada
Chicago, IL
Earth
Group By: new_location
Size: sentiment (Count)
US Three Main State VS Sentiment
California
Florida
Texas
Sentiment
neutral
positive
negative
Group By: sentiment
Size: new_location (Count)
Followers VS Sentiment
sentiment (Count)
followers_range
Text Word Count VS Sentiment
sentiment (Count)
word_count

Quicksight: predictions





Conclusions & Challenges&Next Step

Conclusions

- Logistic regression model shows higher f1 score than decision tree
- The predictions of logistic regression is good
- Elon Musk overall’s sentiment is relatively positive

Challenges&Next Step

- It’s weird that accuracy score is same with f1 score for logistic regression
- Use more model