Day 37 Jan 6

# Explaining my understanding of the Movie Recommendation Engine by hand

1. Step : Importing neccessaries libraries and dependencies

ML will be done using sci-kit learning. The libraries of interest are

→ Count vectorizer : change text documents to integers/vectors. Most of the libraries work better with numbers than text

→ Cosine similarity: find the similarity between files of interest

Also we will import pandas & numpy to work with dataframes. and data

→ Step 2: Import and read the data.

```
data = pd.read_csv("data.csv")
```

→ extension

data to read

Python Command

- Get details of file

data.info() :→ Get information about column headings of data, no of data that are non-null.

Fill the null rows [ columns

data.columns() : Get info about the headings.

Step 3: Select the features to be used for the

inputs of interest ' ' (because they are text

```
features = ['a', 'b', 'e']
```

they are list

Step 4: Put all the features of interest in one column

converting nan to strings.

```
for feature in features:
    df[feature] = df[feature].fillna('')
```

This is how this works.

first feature is a

```
for a in features:
    df[a] = df[a].fillna('')
```

in

Save this

We used an empty string .fillna('') because the datatype is a string. If numerical value, the appropriate type will be 0

Fxn definition

placeholder

```
def combine_feature(row):
    try:
        return row['a'] + " " + return row['b']
    except:
        print("Error:", row)
```

column name

return the row of the column name a + " " row of the column name [b]

Creating column in dataframe:

```
df['combined feature'] = df.apply(combine_feature, axis = 1)
```

apply to row

To access a particular data in the data frame.

```
df.iloc[0]['combined features']
```

dataframe

python for element of access

column name

Step 4: Convert text document to integers/ vectors

```
cv = CountVectorizer()
```

↳ initializing it

```
count_matrix = cv.fit_transform(df["combined_features"])
```

↓ view in human readable form

```
a = count_matrix.toarray()
```

↳ allows to view

Access vocabulary learnt by CountVectorizer

```
vocab = cv.vocabulary_.keys() )
```

Cosine_similarity

vector of combined features

```
cos_sim = cosine_similarity(count_matrix)
```