SACRAMENTO
STATE

CSC 177- Data Analytics and Mining

Project 4: Cluster Analysis, ANN and Text Mining Project

Team Challengers (23):

1. Srujay Reddy Vangoor
2. Vaibhav Jain
3. Bashar Allwza
4. Varun Bailapudi
5. Uddayankith Chodagam

Cluster Analysis

Cluster Analysis

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters).

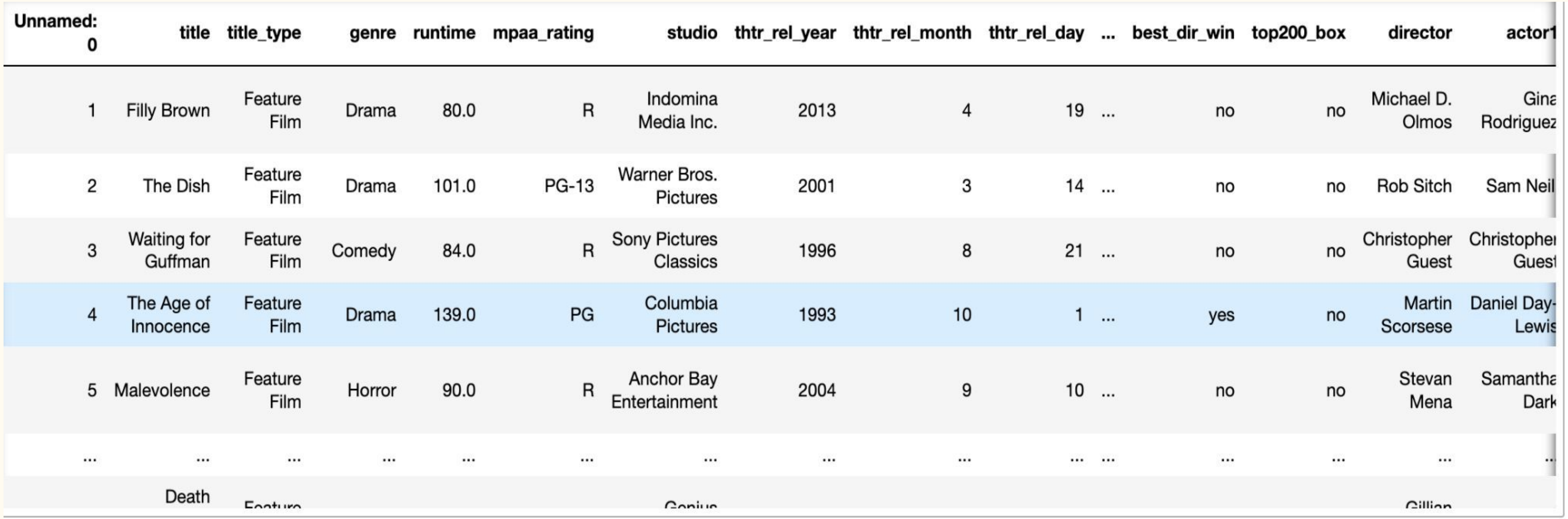
Clustering is an unsupervised learning technique, meaning that it does not rely on labeled data or a pre-existing classification scheme. Instead, clustering algorithms identify patterns in the data based on similarities or differences between data points and group them together into clusters based on those similarities or differences.

Cluster Analysis (Contd.)

Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their understanding of what constitutes a cluster and how to efficiently find them.

About the Dataset

IMDB movie review dataset is used to demonstrate the clustering algorithms.



Unnamed: 0	title	title_type	genre	runtime	mpaa_rating	studio	thtr_rel_year	thtr_rel_month	thtr_rel_day	...	best_dir_win	top200_box	director	actor1
1	Filly Brown	Feature Film	Drama	80.0	R	Indomina Media Inc.	2013	4	19	...	no	no	Michael D. Olmos	Gina Rodriguez
2	The Dish	Feature Film	Drama	101.0	PG-13	Warner Bros. Pictures	2001	3	14	...	no	no	Rob Sitch	Sam Neil
3	Waiting for Guffman	Feature Film	Comedy	84.0	R	Sony Pictures Classics	1996	8	21	...	no	no	Christopher Guest	Christopher Guest
4	The Age of Innocence	Feature Film	Drama	139.0	PG	Columbia Pictures	1993	10	1	...	yes	no	Martin Scorsese	Daniel Day-Lewis
5	Malevolence	Feature Film	Horror	90.0	R	Anchor Bay Entertainment	2004	9	10	...	no	no	Stevan Mena	Samantha Dark
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
	Death	Feature				Genius							Gillian	

K-Means Clustering

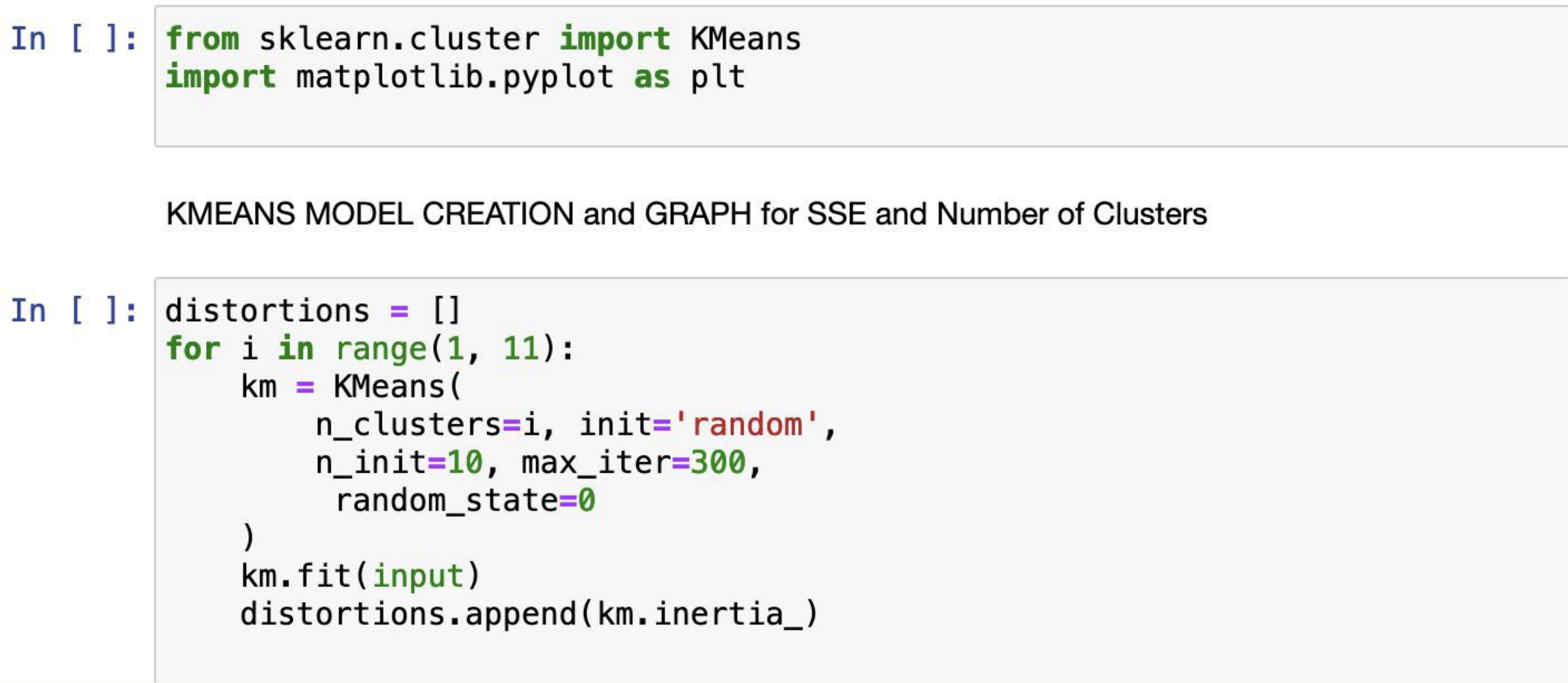
K-means clustering is a popular unsupervised machine learning algorithm used to group data points into clusters based on similarity.

It is a centroid-based clustering algorithm that iteratively partitions the data points into k clusters, where k is a pre-specified number of clusters. The algorithm starts by randomly selecting k initial centroids, and then it iteratively assigns each data point to the nearest centroid, computes the mean of each cluster, and reassigns the centroids based on the new means. This process continues until the centroids no longer move significantly or until a predetermined number of iterations is reached.

K-Means Clustering (Contd.)

K-means clustering is widely used for image segmentation, market segmentation, anomaly detection, and many other applications. One of the main advantages of k-means clustering is that it is computationally efficient and can handle large datasets. However, one of the limitations of k-means clustering is that it requires the number of clusters to be specified beforehand, and the quality of the results is highly dependent on the initial placement of the centroids.

K-Means Clustering (Contd.)



In [ ]:

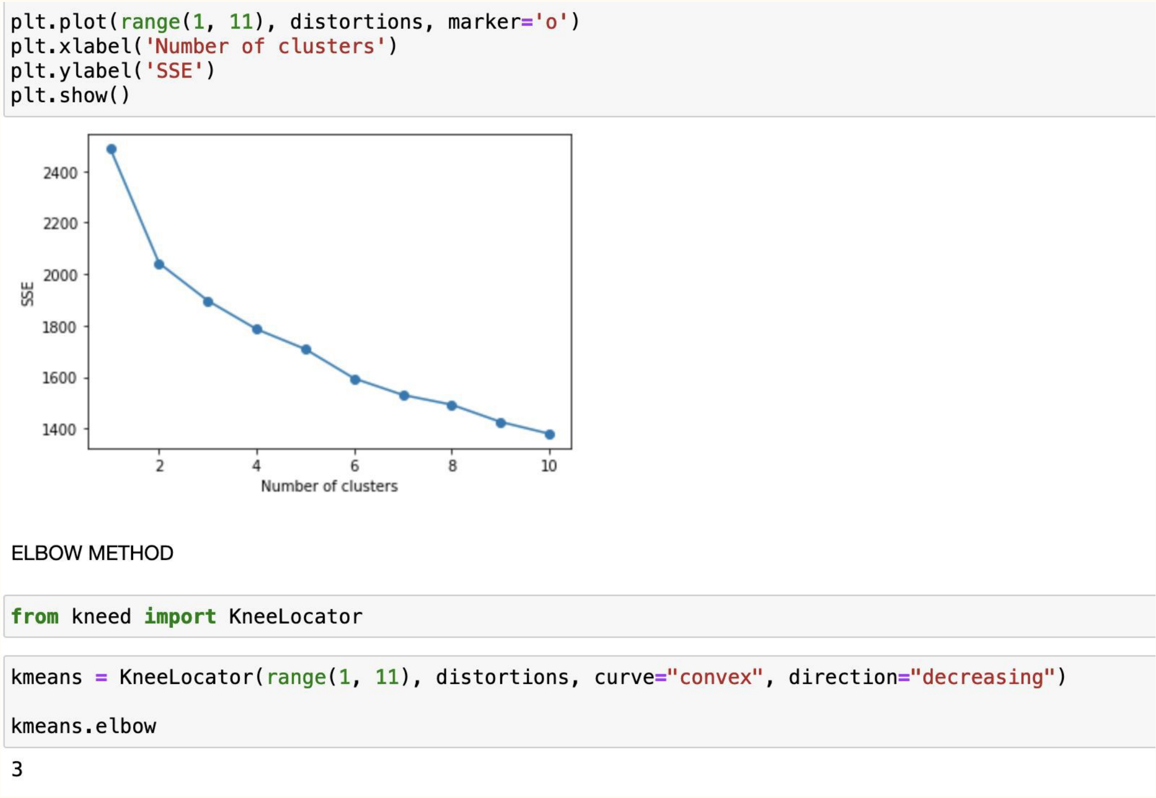
```
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
```

KMEANS MODEL CREATION and GRAPH for SSE and Number of Clusters

In [ ]:

```
distortions = []
for i in range(1, 11):
    km = KMeans(
        n_clusters=i, init='random',
        n_init=10, max_iter=300,
         random_state=0
    )
    km.fit(input)
    distortions.append(km.inertia_)
```

K-Means Clustering (Contd.)



```
plt.plot(range(1, 11), distortions, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('SSE')
plt.show()
```

ELBOW METHOD

```
from kneed import KneeLocator
```

```
kmeans = KneeLocator(range(1, 11), distortions, curve="convex", direction="decreasing")

kmeans.elbow
```

```
3
```

Hierarchical Clustering

Hierarchical clustering is a type of clustering algorithm used in data mining and machine learning. It is a method for grouping similar data points or objects into clusters based on their similarity or dissimilarity. In hierarchical clustering, the data points are recursively partitioned into a hierarchy of clusters.

There are two main types of hierarchical clustering: Agglomerative hierarchical clustering and Divisive hierarchical clustering.

Hierarchical Clustering (Contd.)

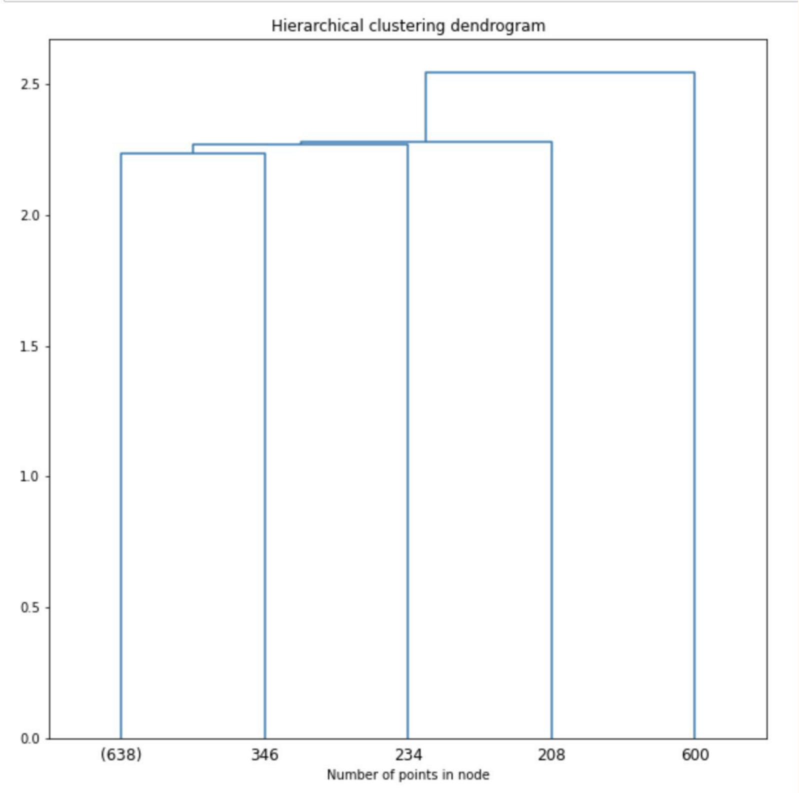
Agglomerative hierarchical clustering: This method starts with each data point as a separate cluster and then merges the closest pairs of clusters iteratively until only one cluster remains.

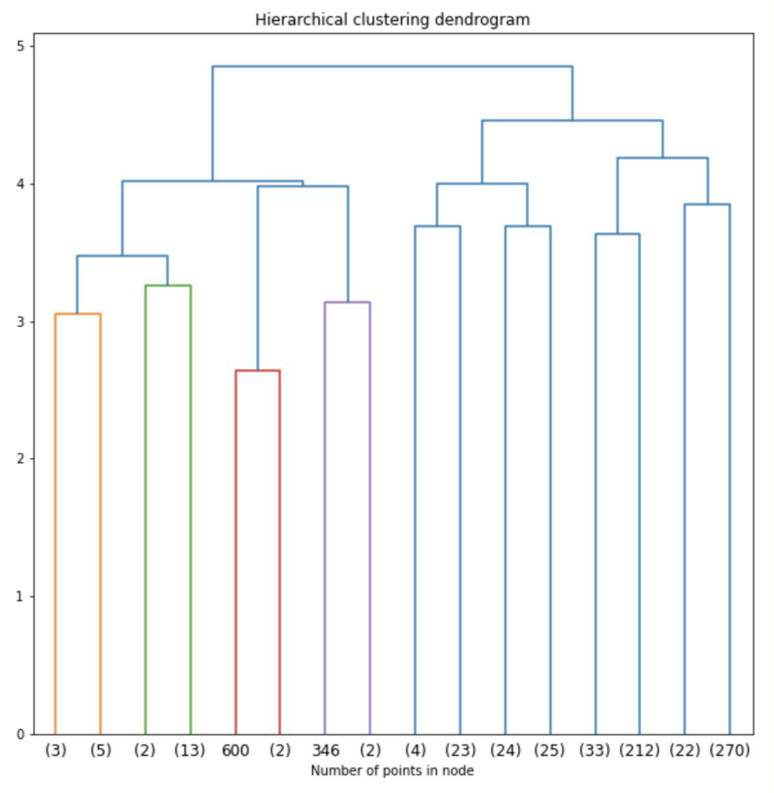
Divisive hierarchical clustering: This method starts with all data points in one cluster and then recursively divides it into smaller subclusters based on their dissimilarity.

Hierarchical Clustering (Contd.)

Hierarchical clustering can be visualized using dendrograms, which are tree-like diagrams that show the order and distances of the merging or splitting of clusters. The height of the dendrogram indicates the distance between clusters or the dissimilarity between data points.

Hierarchical Clustering (Contd.)





Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs)

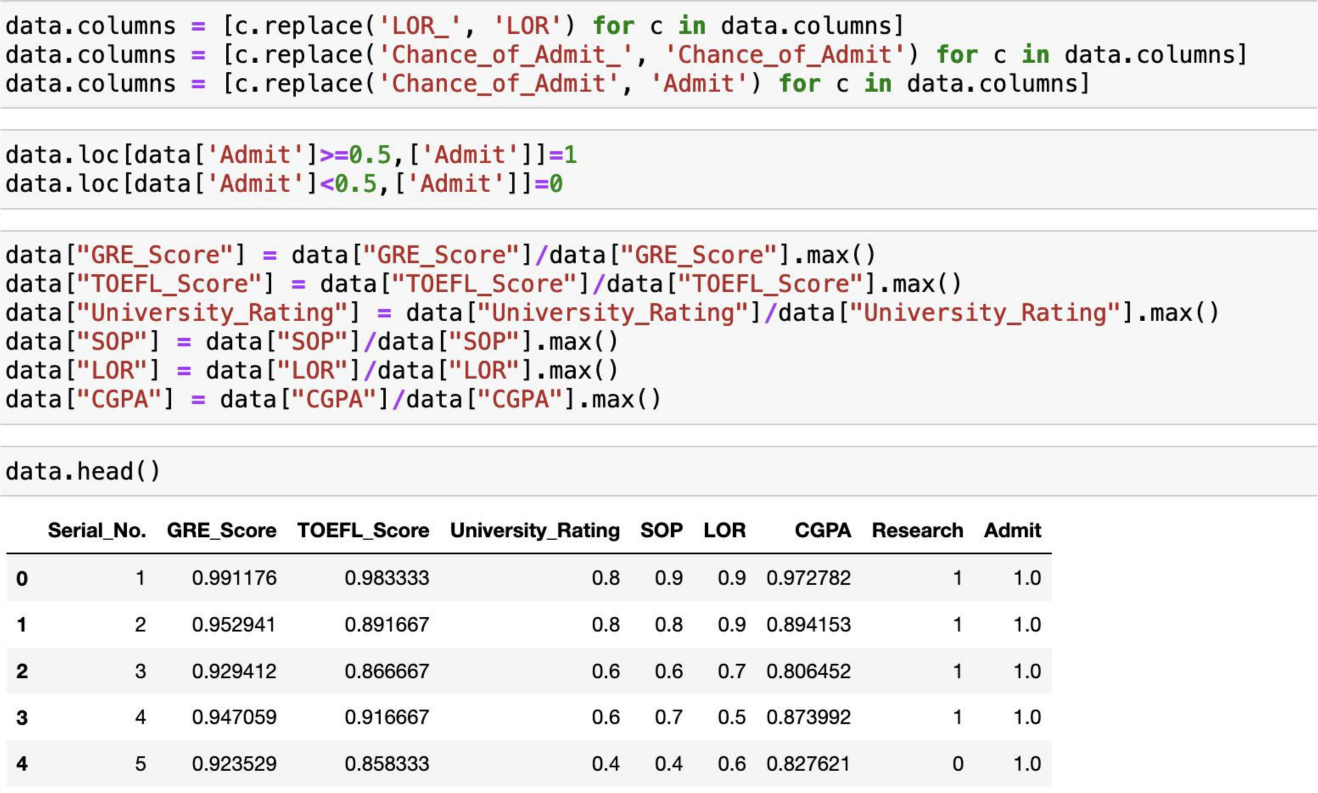
Artificial neural networks (ANNs) are computational models inspired by the structure and function of the human brain. They are composed of interconnected nodes that process information, called artificial neurons or simply "neurons." These neurons are organized into layers, with each layer performing a different computation on the input data.

Artificial Neural Networks (Contd.)

ANNs are typically used for supervised learning tasks, such as classification or regression. During training, the network is presented with a set of labeled examples and adjusts its weights to minimize the difference between its predicted output and the actual output.

Once the network is trained, it can be used to make predictions on new, unseen data. ANNs have been successfully applied in a wide range of fields, including computer vision, natural language processing, speech recognition, and robotics.

Artificial Neural Networks (Contd.)



```
data.columns = [c.replace('LOR_', 'LOR') for c in data.columns]
data.columns = [c.replace('Chance_of_Admit_', 'Chance_of_Admit') for c in data.columns]
data.columns = [c.replace('Chance_of_Admit', 'Admit') for c in data.columns]
```

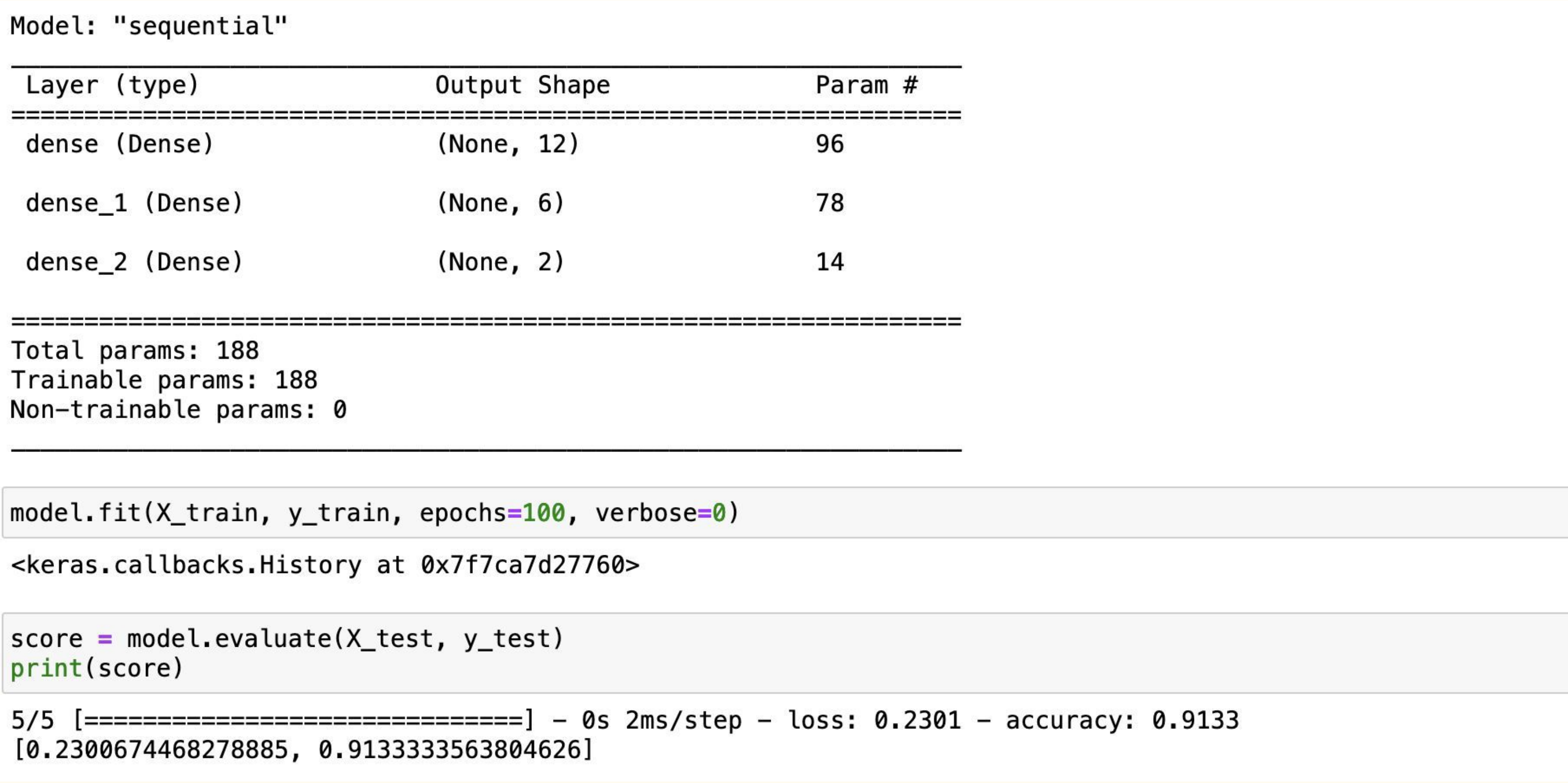
```
data.loc[data['Admit']>=0.5,['Admit']]=1
data.loc[data['Admit']<0.5,['Admit']]=0
```

```
data["GRE_Score"] = data["GRE_Score"]/data["GRE_Score"].max()
data["TOEFL_Score"] = data["TOEFL_Score"]/data["TOEFL_Score"].max()
data["University_Rating"] = data["University_Rating"]/data["University_Rating"].max()
data["SOP"] = data["SOP"]/data["SOP"].max()
data["LOR"] = data["LOR"]/data["LOR"].max()
data["CGPA"] = data["CGPA"]/data["CGPA"].max()
```

```
data.head()
```

	Serial_No.	GRE_Score	TOEFL_Score	University_Rating	SOP	LOR	CGPA	Research	Admit
0	1	0.991176	0.983333	0.8	0.9	0.9	0.972782	1	1.0
1	2	0.952941	0.891667	0.8	0.8	0.9	0.894153	1	1.0
2	3	0.929412	0.866667	0.6	0.6	0.7	0.806452	1	1.0
3	4	0.947059	0.916667	0.6	0.7	0.5	0.873992	1	1.0
4	5	0.923529	0.858333	0.4	0.4	0.6	0.827621	0	1.0

Artificial Neural Networks (Contd.)



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 12)                96

 dense_1 (Dense)             (None, 6)                 78

 dense_2 (Dense)             (None, 2)                 14

=================================================================
Total params: 188
Trainable params: 188
Non-trainable params: 0
_________________________________________________________________
```

```
model.fit(X_train, y_train, epochs=100, verbose=0)
```

```
<keras.callbacks.History at 0x7f7ca7d27760>
```

```
score = model.evaluate(X_test, y_test)
print(score)
```

```
5/5 [==============================] - 0s 2ms/step - loss: 0.2301 - accuracy: 0.9133
[0.2300674468278885, 0.9133333563804626]
```

Text Mining

Text Mining

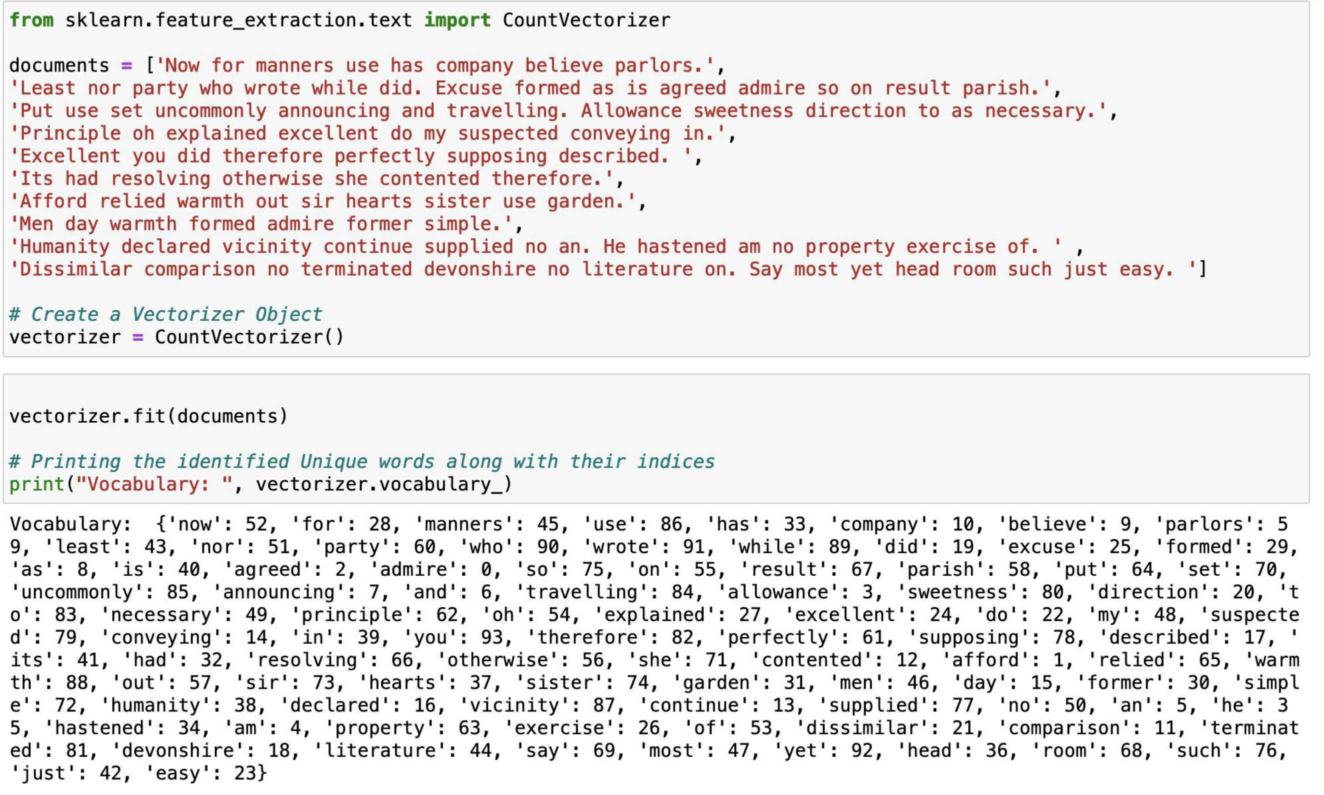
Text mining, also known as text data mining, is the process of deriving meaningful insights and information from unstructured textual data. Text mining involves the application of natural language processing (NLP), information retrieval, and machine learning techniques to extract and analyze information from unstructured text data.

Text Mining (Contd.)

Text mining is used to identify patterns, relationships, and insights from large collections of textual data, such as emails, social media posts, customer reviews, news articles, and scientific literature. Text mining techniques can be applied to a variety of tasks, including sentiment analysis, topic modeling, named entity recognition, and text classification.

Text mining is an important technique in data mining because it enables organizations to extract valuable insights and knowledge from unstructured textual data, which can be used to inform business decisions, improve customer satisfaction, and gain a competitive advantage.

Text Mining (Contd.)



```python
from sklearn.feature_extraction.text import CountVectorizer

documents = ['Now for manners use has company believe parlors.',
'Least nor party who wrote while did. Excuse formed as is agreed admire so on result parish.',
'Put use set uncommonly announcing and travelling. Allowance sweetness direction to as necessary.',
'Principle oh explained excellent do my suspected conveying in.',
'Excellent you did therefore perfectly supposing described. ',
'Its had resolving otherwise she contented therefore.',
'Afford relied warmth out sir hearts sister use garden.',
'Men day warmth formed admire former simple.',
'Humanity declared vicinity continue supplied no an. He hastened am no property exercise of. ' ,
'Dissimilar comparison no terminated devonshire no literature on. Say most yet head room such just easy. ']

# Create a Vectorizer Object
vectorizer = CountVectorizer()
```

```python
vectorizer.fit(documents)

# Printing the identified Unique words along with their indices
print("Vocabulary: ", vectorizer.vocabulary_)
```

```
Vocabulary:  {'now': 52, 'for': 28, 'manners': 45, 'use': 86, 'has': 33, 'company': 10, 'believe': 9, 'parlors': 5
9, 'least': 43, 'nor': 51, 'party': 60, 'who': 90, 'wrote': 91, 'while': 89, 'did': 19, 'excuse': 25, 'formed': 29,
'as': 8, 'is': 40, 'agreed': 2, 'admire': 0, 'so': 75, 'on': 55, 'result': 67, 'parish': 58, 'put': 64, 'set': 70,
'uncommonly': 85, 'announcing': 7, 'and': 6, 'travelling': 84, 'allowance': 3, 'sweetness': 80, 'direction': 20, 't
o': 83, 'necessary': 49, 'principle': 62, 'oh': 54, 'explained': 27, 'excellent': 24, 'do': 22, 'my': 48, 'suspecte
d': 79, 'conveying': 14, 'in': 39, 'you': 93, 'therefore': 82, 'perfectly': 61, 'supposing': 78, 'described': 17, '
its': 41, 'had': 32, 'resolving': 66, 'otherwise': 56, 'she': 71, 'contented': 12, 'afford': 1, 'relied': 65, 'warm
th': 88, 'out': 57, 'sir': 73, 'hearts': 37, 'sister': 74, 'garden': 31, 'men': 46, 'day': 15, 'former': 30, 'simpl
e': 72, 'humanity': 38, 'declared': 16, 'vicinity': 87, 'continue': 13, 'supplied': 77, 'no': 50, 'an': 5, 'he': 3
5, 'hastened': 34, 'am': 4, 'property': 63, 'exercise': 26, 'of': 53, 'dissimilar': 21, 'comparison': 11, 'terminat
ed': 81, 'devonshire': 18, 'literature': 44, 'say': 69, 'most': 47, 'yet': 92, 'head': 36, 'room': 68, 'such': 76,
'just': 42, 'easy': 23}
```

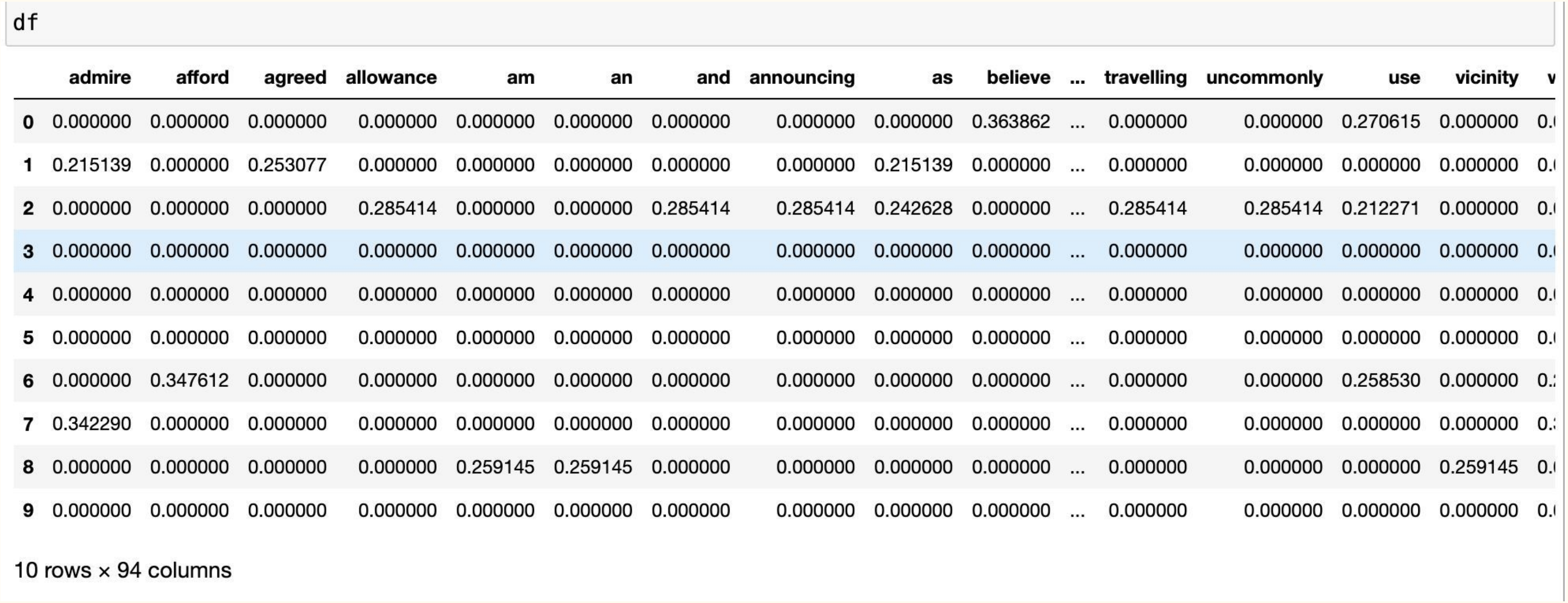
TF-IDF Vectors

TF-IDF stands for "Term Frequency-Inverse Document Frequency". In text mining and information retrieval, it is a numerical statistic that reflects how important a word is to a document in a collection or corpus.

The term frequency is the number of occurrences of a specific term in a document. Term frequency indicates how important a specific term in a document.

Document frequency is the number of documents containing a specific term. Document frequency indicates how common the term is.

TF-IDF Vectors (Contd.)



```
df
```

	admire	afford	agreed	allowance	am	an	and	announcing	as	believe	...	travelling	uncommonly	use	vicinity	v
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.363862	...	0.000000	0.000000	0.270615	0.000000	0.
1	0.215139	0.000000	0.253077	0.000000	0.000000	0.000000	0.000000	0.000000	0.215139	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.
2	0.000000	0.000000	0.000000	0.285414	0.000000	0.000000	0.285414	0.285414	0.242628	0.000000	...	0.285414	0.285414	0.212271	0.000000	0.
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.
4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.
5	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.
6	0.000000	0.347612	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.258530	0.000000	0.
7	0.342290	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.
8	0.000000	0.000000	0.000000	0.000000	0.259145	0.259145	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.259145	0.
9	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.

10 rows × 94 columns

Thank you!!