Task 2

Table of Contents

- ## Loading Data set into the DataFrame

Loading **sign_mnist.csv** data into dataframe. The name of the DataFrame is ‘df’.

Hand Gesture recognition app in python

Importing necessary libraries

```
In [1]: from sklearn.ensemble import RandomForestClassifier
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
        from sklearn.linear_model import SGDClassifier
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler
        from sklearn.decomposition import PCA
        import numpy as np
        import pandas as pd
```

Loading the dataset into the DataFrame

```
In [2]: df = pd.read_csv('sign_mnist.csv')
        df.head()
```

Out[2]:

	label	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel781	pixel782	pi
0	3	107	118	127	134	139	143	146	150	153	...	207	207	207	207	206	206	206	204	
1	6	155	157	156	156	156	157	156	158	158	...	69	149	128	87	94	163	175	103	
2	2	187	188	188	187	187	186	187	188	187	...	202	201	200	199	198	199	198	195	
3	2	211	211	212	212	211	210	211	210	210	...	235	234	233	231	230	226	225	222	
4	13	164	167	170	172	176	179	180	184	185	...	92	105	105	108	133	163	157	163	

5 rows × 785 columns

- ## Selecting Dependent and Independent Variables

Since in the data set, label is the output column, it becomes the dependent column. Rest other columns are the Independent columns. So assigning y to label and x to other columns.

Checking number of columns and rows

```
In [3]: print(df.shape)

(10000, 785)
```

Assigning Independent variables to x and dependent variable to y

```
In [4]: x = df.iloc[:, 1:]
        y = df.iloc[:, 0]
```

Demensions of Independent and Dependent variables

```
In [5]: print(x.shape)
        print(y.shape)

(10000, 784)
(10000,)
```

Principle Component Analysis (PCA) is used to reduce the dimensions of the independent variable (x), since we don’t know how many number of components to reduce simply following the variance - covariance rule of PCA algorithm which is ranging between 0.95 to 0.99, this automatically reduces to the best dimensions possible.

Principle Component Analysis (PCA) is used to reduce the demensions of the independent variable (x), since we dont know how many number of components to reduce simply following the variance - covaraince rule of PCA algorithm which is ranging between 0.95 to 0.99, this automatically reduces to the best demensions possible.

In [6]:
```
pca = PCA(n_components=0.95)
pca.fit(x)
x_pca = pca.transform(x)
print(x_pca.shape)
```

```
(10000, 112)
```

Our columns has been reduced from 784 to 112 after applying PCA

Splitting the dataset into training 80% and test sets into 20%

In [7]:
```
x_train, x_test, y_train, y_test = train_test_split(
    x_pca, y, test_size=0.2, random_state=42)
```

Standardization of independent variable x - mean to 0 and standard deviation to 1

In [8]:
```
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.fit_transform(x_test)
```

- ## SGDClassifier

Using sklearn.metrics we use accuracy_score function to get the percentage accuracy of model, normally we use confusion matrix for classification algorithms since we already standardiized our data we are using accuracy as our metrics.

Stochastic Gradient Descent Classifier

Here, we are using loss function as log which is better in Classification algorithms.

In [9]:
```
sgd = SGDClassifier(loss='log', shuffle=True, random_state=42)
```

Fitting the data and predicting the data using test sets and training sets

In [10]:
```
sgd.fit(x_train_scaled, y_train)
y_pred = sgd.predict(x_test_scaled)
y_train_pred = sgd.predict(x_train_scaled)
```

using sklearn.metrics we use accuracy_score function to get the percentage accuracy of model, normally we use confusion matrix for classification algorithms since we already standardiized our data we are using accuracy as our metrics.

In [11]:
```
acc_score_test = accuracy_score(
    y_test, y_pred, normalize=True, sample_weight=None)
acc_score_train = accuracy_score(
    y_train, y_train_pred, normalize=True, sample_weight=None)
```

In [12]:
```
print('Stochastic gradient descent performance metrics :')
print('Accuracy score for test set:', acc_score_test*100)
print('Accuracy score for train set:', acc_score_train*100)
```

```
Stochastic gradient descent performance metrics :
Accuracy score for test set: 88.35
Accuracy score for train set: 94.875
```

Stochastic gradient descent performance metrics :
Accuracy score for test set: 88.35
Accuracy score for train set: 94.875

- ## Decision Tree Classifier

Since our columns are 112 after dimensionality reduction, max_depth is the depth of tree or the level of the tree = n/2 which is 112/2 = 51, max_depth = 51

Decision Tree Classifier

Since our columns are 112 after dimensionality reduction, max_depth is the depth of tree or the level of the tree = n/2 which is 112/2 = 51, max_depth = 51

```
In [13]: dtc = DecisionTreeClassifier(max_depth=51, random_state=42)
dtc.fit(x_train_scaled, y_train)
y_pred_dtc = dtc.predict(x_test_scaled)
y_train_pred_dtc = dtc.predict(x_train_scaled)

print('Decision tree classifier performance metrics :')
print('Accuracy score for test set:', accuracy_score(y_test, y_pred_dtc)*100)
print('Accuracy score for train set:',
      accuracy_score(y_train, y_train_pred_dtc)*100)

Decision tree classifier performance metrics :
Accuracy score for test set: 88.44999999999999
Accuracy score for train set: 100.0
```

Decision tree classifier performance metrics :
Accuracy score for test set: 88.44999999999999
Accuracy score for train set: 100.0

- ## Random Forest Classifier

Here, n_estimators is the number of different trees we want to create and train the model therefore n_estimators = 10 means we want 10 trees and max_depth = 51 is same as before which we used.

Random Forest Classifier

Here, n_estimators is the number of different trees we want to create and train the model therefore n_estimators = 10 means we want 10 trees and max_depth = 51 is same as before which we used.

```
In [14]: rfc = RandomForestClassifier(n_estimators=10, max_depth=51, random_state=42)
rfc.fit(x_train_scaled, y_train)
y_pred_rfc = rfc.predict(x_test_scaled)
y_train_pred_rfc = rfc.predict(x_train_scaled)

print('Random Forest Classifier performance metrics :')
print('Accuracy score for test set:', accuracy_score(y_test, y_pred_rfc)*100)
print('Accuracy score for train set:',
      accuracy_score(y_train, y_train_pred_rfc)*100)

Random Forest Classifier performance metrics :
Accuracy score for test set: 99.65
Accuracy score for train set: 99.9875
```

Finally, comparing our performance metrics of 3 different Classifiers

Stochastic gradient descent has 88.35 %

Decision tree classifier has 88.44 %

Random Forest Classifier has 99.65 %

Since, Random Forest Classifier has highest accuracy of 99.65 %. Out of the 3 Classifiers, we could say Random Forest Classifiers is the best.

```
In [ ]:
```

- ## Comparison of the classifiers

Finally, comparing our performance metrics of 3 different Classifiers

Stochastic gradient descent has 88.35 %

Decision tree classifier has 88.44 %

Random Forest Classifier has 99.65 %

Since, Random Forest Classifier has highest accuracy of 99.65 %. Out of the 3 Classifiers, we could say Random Forest Classifiers is the best.

In [ ]:

Task 3

- ## Installing Required Modules and libraries

Installing Modules

In [318]:
```
! pip install google-play-scraper
! pip install lxml
! pip install wordcloud
```

```
Requirement already satisfied: google-play-scraper in c:\users\vinayak yadav\anaconda3\lib\site-packages (1.0.2)
Requirement already satisfied: lxml in c:\users\vinayak yadav\anaconda3\lib\site-packages (4.6.3)
Requirement already satisfied: wordcloud in c:\users\vinayak yadav\anaconda3\lib\site-packages (1.8.1)
Requirement already satisfied: pillow in c:\users\vinayak yadav\anaconda3\lib\site-packages (from wordcloud) (8.2.0)
Requirement already satisfied: matplotlib in c:\users\vinayak yadav\anaconda3\lib\site-packages (from wordcloud) (3.3.4)
Requirement already satisfied: numpy>=1.6.1 in c:\users\vinayak yadav\anaconda3\lib\site-packages (from wordcloud) (1.20.1)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\vinayak yadav\anaconda3\lib\site-packages (from matplotlib->wor
dcloud) (2.8.1)
Requirement already satisfied: cycler>=0.10 in c:\users\vinayak yadav\anaconda3\lib\site-packages (from matplotlib->wordcloud)
(0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\vinayak yadav\anaconda3\lib\site-packages (from matplotlib->wordcl
oud) (1.3.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\vinayak yadav\anaconda3\lib\site-packages
(from matplotlib->wordcloud) (2.4.7)
Requirement already satisfied: six in c:\users\vinayak yadav\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib->wordcl
oud) (1.15.0)
```

google_play_scraper was the library that was explicitly used.

Importing necessary libraries

In [319]:
```
import pandas as pd
import numpy as np
from google_play_scraper import app, Sort, reviews
import pprint

print('Libraries Imported')
```

```
Libraries Imported
```

- ## Extracting data from Google Play Store

Using Google Play store website, we have fetched Application ID for a particular App

The apps used are Google Photos [GOOGLE] , Amazon Prime Video [AMAZON], LinkedIn [LINKEDIN], Tesla Car Simulator [TESLA]

Once the Application name and their IDs are fetched, the data is staged in pandas DataFrame

Extracting data from Google Play Store

1. Using Google Play store website, we have fetched Application ID for a particular App
2. The apps used are Google Photos [GOOGLE] , Amazon Prime Video [AMAZON], LinkedIn [LINKEDIN], Tesla Car Simulator [TESLA]
3. Once the Application name and their IDs are fetched, the data is staged in pandas DataFrame

In [289]:

```
# creating dataframe for storing the app name and app ID from google play store

app_descritpion_df = pd.DataFrame({'App_Name':['Google Photos', 'Amazon Prime Video', 'LinkedIn', 'Tesla Car Simulator'],
                                   'Application_Id':['com.google.android.apps.photos','com.amazon.avod.thirdpartyclient',
                                                     'com.linkedin.android','com.trioz.electrical.car.simulator.tesla']})

app_descritpion_df.head()
```

Out[289]:

	App_Name	Application_Id
0	Google Photos	com.google.android.apps.photos
1	Amazon Prime Video	com.amazon.avod.thirdpartyclient
2	LinkedIn	com.linkedin.android
3	Tesla Car Simulator	com.trioz.electrical.car.simulator.tesla

- ## Scraping the Reviews from the data

1. Using reviews function of google_play_scraper module, we were able to scrap or extract the reviews of the application posted by several users.
2. Since our data is maintained in a dataframe, so by using Application_id as the key column iin the dataframe, iteration is done to navigate through all the company apps and get the reviews for each apps.
3. Depending on which group the data belongs to, Segregation is done in 4 different lists.
4. In the Output of the code snippet, multiple reviews from all the companies are listed. These are getting populated one at a time in those 4 lists by using append method.
5. checking which app id is currently being read and assigning the list accordingly
6. Once the app id is decided then data is appended in the list

Scraping Reviews for each company apps

1. Using reviews function of google_play_scraper module, we were able to scrap or extract the reviews of the application posted by several users.
2. Since our data is maintained in a dataframe, so by using Application_id as the key column iin the dataframe, iteration is done to navigate through all the company apps and get the reviews for each apps.
3. Depending on which group the data belongs to, Segregation is done in 4 different lists.
4. In the Output of the code snippet, multiple reviews from all the companies are listed. These are getting populated one at a time in those 4 lists by using append method.
5. checking which app id is currently being read and assigning the list accordingly
6. Once the app id is decided then data is appended in the list

In [320]:
```
app_reviews = []

for app_id in app_descritpion_df['Application_Id']:
  for score in list(range(1, 6)):
    for sort_order in [Sort.MOST_RELEVANT, Sort.NEWEST]:
      app_review, _ = reviews(
        app_id,
        lang='en',
        country='us',
        sort=sort_order,
        count= 200 if score == 3 else 100,
        filter_score_with=score
      )

      for r in app_review:
        r['sortOrder'] = 'most_relevant' if sort_order == Sort.MOST_RELEVANT else 'newest'
        r['appId'] = app_id


    # checking which app id is currently being read and assigning the list accordingly
    #Once the app id is decided then data is appended in the list
      for contents in range(len(app_review)):
            if app_id == 'com.google.android.apps.photos':
                pprint.pprint("GOOGLE DATA LOAD BEGINS")
                pprint.pprint(app_review[contents]['content'])
                google_list.append(app_review[contents]['content'])

            elif app_id == 'com.amazon.avod.thirdpartyclient':
```

```
      for contents in range(len(app_review)):
            if app_id == 'com.google.android.apps.photos':
                pprint.pprint("GOOGLE DATA LOAD BEGINS")
                pprint.pprint(app_review[contents]['content'])
                google_list.append(app_review[contents]['content'])

            elif app_id == 'com.amazon.avod.thirdpartyclient':
                pprint.pprint("AMAZON DATA LOAD BEGINS")
                pprint.pprint(app_review[contents]['content'])
                amazon_list.append(app_review[contents]['content'])

            elif app_id == 'com.linkedin.android':
                pprint.pprint("LINKEDIN DATA LOAD BEGINS")
                pprint.pprint(app_review[contents]['content'])
                linkedin_list.append(app_review[contents]['content'])

            elif app_id == 'com.trioz.electrical.car.simulator.tesla':
                pprint.pprint("TESLA DATA LOAD BEGINS")
                pprint.pprint(app_review[contents]['content'])
                tesla_list.append(app_review[contents]['content'])
```

Output: List of reviews

```
                pprint.pprint("TESLA DATA LOAD BEGINS")
                pprint.pprint(app_review[contents]['content'])
                tesla_list.append(app_review[contents]['content'])
```

```
('I keep receiving an error when trying to delete pictures from my SD card '
 "even though I updated the settings...please fix this, it's annoying.")
'GOOGLE DATA LOAD BEGINS'
('Google! Could we PLEASE have some way to move the controls back to screen '
 'top instead of down at the bottom fighting with the navigation controls? '
 'PLEASE??? This is USELESS!!!')
'GOOGLE DATA LOAD BEGINS'
('The Google photos app keeps closing. I was trying to download my photos to '
 'my gallery app to save my photos to my SD card but it keeps closing on me. '
 'Please fix .')
'GOOGLE DATA LOAD BEGINS'
('Terrible for sharing a group of photographs. Poor instructions. Not well '
 'arranged. User documentation terrible.')
'GOOGLE DATA LOAD BEGINS'
('Bad experience....now Google photos automatically rotates my photos up side '
 "down... can't fix. Give me back the old program.")
'GOOGLE DATA LOAD BEGINS'
('You have to be kidding me. You get rid of unlimited storage! The app is '
 'great except they want me to buy storage from them. No thanks!')
```

- ## Creating Dataframe for Staging the reviews

1. For each list, there is a seperate DataFrame created that will store the values for each company
2. Once these DataFrames are populated, then all of them are loaded in a single big DataFrame containing all the values for all the companies.
3. To do so, append method is used and reviews_df DataFrame is created which is shown in the below code along with the output for that.

Creating DataFrames for staging reviews

1. For each list, there is a seperate DataFrame created that will store the values for each company
2. Once these DataFrames are populated, then all of them are loaded in a single big DataFrame containing all the values for all the companies.
3. To do so, append method is used and reviews_df DataFrame is created which is shown in the below code along with the output for that.

In [321]:
```
# google dataframe with google list
google_df = pd.DataFrame({'Application_Name':'Google Photos',
                          'Application_Reviews': google_list})

# amazon dataframe with amazon list
amazon_df = pd.DataFrame({'Application_Name': 'Amazon Prime Video',
                          'Application_Reviews': amazon_list})

# linkedin dataframe with linkedin list
linkedin_df = pd.DataFrame({'Application_Name': 'LinkedIn',
                            'Application_Reviews': linkedin_list})

# tesla dataframe with tesla list
tesla_df = pd.DataFrame({'Application_Name': 'Tesla Car Simulator',
                         'Application_Reviews': tesla_list})
```

In [294]:
```
# Tesla DataFrame

tesla_df.head()
```

Out[294]:

	Application_Name	Application_Reviews
0	Tesla Car Simulator	The game is too laggy too much ads way expensi...
1	Tesla Car Simulator	I do not like this game it is s super glitchy ...
2	Tesla Car Simulator	Worst Tesla Game IS THE WORST ONE I'VE EVER PL...

Appending all the data into a single dataframe

In [295]:
```
# Appending all the individual dataframe to make one big DataFrame with all the reviews

reviews_df = google_df.append([amazon_df,linkedin_df,tesla_df])
reviews_df
```

Out[295]:

	Application_Name	Application_Reviews
0	Google Photos	Google phones focuses a lot on cameras and pho...
1	Google Photos	I have had it with Google photos. I do not wan...
2	Google Photos	Used this as my back-up for very important pho...
3	Google Photos	I've loved Google photos for years, but ever s...
4	Google Photos	There is no way to select multiple media and (...
...	...	...
685	Tesla Car Simulator	Tesla
686	Tesla Car Simulator	
687	Tesla Car Simulator	Wowwwwwwww
688	Tesla Car Simulator	This game is so cool I love it don't forget to 5
689	Tesla Car Simulator	Good game

4290 rows × 2 columns

In [296]:
```
# chechking the dimensions of the Dataframe reviews_df

rows, columns = reviews_df.shape
print("There are {} rows and {} columns all together in the DataFrame after extracting all the reviews".format(rows,columns))
```

```
There are 4290 rows and 2 columns all together in the DataFrame after extracting all the reviews
```

- ## Cleaning data from Dataframe

1. Cleaning of the data is carried out using nltk package
2. 3 stage cleansing is made sure, like removing punctuations between the texts.
3. Words are tokenized in order to perform operations like Lemmatization
4. Unwanted words are removed using stopwords functionality

Cleaning Data from DataFrame

1. Cleaning of the data is carried out using nltk package
2. 3 stage cleansing is made sure, like removing punctuations between the texts.
3. Words are tokenized in order to perform operations like Lemmatization
4. Unwanted words are removed using stopwords functionality

In [297]:
```
#importing nltk modules for data mining purpose.

import re, nltk
nltk.download('stopwords')
nltk.download('wordnet')
from bs4 import BeautifulSoup
from nltk.stem import WordNetLemmatizer
```

```
[nltk_data] Downloading package stopwords to C:\Users\Vinayak
[nltk_data]     Yadav\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to C:\Users\Vinayak
[nltk_data]     Yadav\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

In [298]:
```
reviews_df['Application_Name'] = reviews_df['Application_Name'].map({'Google Photos':0,
                                                                    'Amazon Prime Video':1,
                                                                    'LinkedIn':2,
                                                                    'Tesla Car Simulator':3})

reviews_df
```

Out[298]:

	Application_Name	Application_Reviews
0	0	Google phones focuses a lot on cameras and pho...
1	0	I have had it with Google photos. I do not wan...
2	0	Used this as my back-up for very important pho...
3	0	I've loved Google photos for years, but ever s...
4	0	There is no way to select multiple media and (...

- ## Removing Punctuations

1. Removing unwanted puntuations that have no significance in contributing to the word cloud data
2. Strings like !"#$%&'()*+,-./:;<=>?@[]^_`{|}~ will be taken care off when the below function will be applied on the dataset.
3. A new column named **no_punctuation_reviews** is created in the df which consists of each letter as an element among the string.

Removing Punctuations

1. Removing unwanted puntuations that have no significance in contributing to the word cloud data
2. Strings like !"#$%&'()*+,-./:;<=>?@[]^_`{|}~ will be taken care off when the below function will be applied on the dataset.
3. A new column named **no_punctuation_reviews** is created in the df which consists of each letter as an element among the string.

In [299]:
```python
# String handling of the dataset and generating a new column with clean data

import string
print("The shown list of punctuations will be captured and dropped from the text=>",string.punctuation)

def remove_puntuation(txt):
    clean_txt = [c for c in txt if c not in string.punctuation]
    return clean_txt

reviews_df['no_punctuation_reviews'] = reviews_df['Application_Reviews'].apply(lambda x: remove_puntuation(x))
reviews_df
```

```
The shown list of punctuations will be captured and dropped from the text=> !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

Out[299]:

	Application_Name	Application_Reviews	no_punctuation_reviews
0	0	Google phones focuses a lot on cameras and pho...	[G, o, o, g, l, e, , p, h, o, n, e, s, , f, ...
1	0	I have had it with Google photos. I do not wan...	[I, , h, a, v, e, , h, a, d, , i, t, , w, ...
2	0	Used this as my back-up for very important pho...	[U, s, e, d, , t, h, i, s, , a, s, , m, y, ...
3	0	I've loved Google photos for years, but ever s...	[I, v, e, , l, o, v, e, d, , G, o, o, g, l, ...
4	0	There is no way to select multiple media and (...	[T, h, e, r, e, , i, s, , n, o, , w, a, y, ...
...	...	...	...
685	3	Tesla	[T, e, s, l, a]
686	3	🧕	[🧕]
687	3	Wowwwwwwww	[W, o, w, w, w, w, w, w, w, w]

- Tokenizing

1. Tokenzing of the data is done in which each word in the row is trated as a list of tokens.
2. in the below code snippet, a new column **tokenized_Reviews** is created which gives a clear picture about the tokenization
3. Tokenization is carried out so that moving ahead we could easily apply stopword and lemmatizer on the data

Tokenizing the reviews

1. Tokenzing of the data is done in which each word in the row is trated as a list of tokens.
2. in the below code snippet, a new column **tokenized_Reviews** is created which gives a clear picture about the tokenization
3. Tokenization is carried out so that moving ahead we could easily apply stopword and lemmatizer on the data

In [300]:
```python
#Tokenizing the data and Loading them into a new column in same dataframe

def tokenize(txt):
    token = re.split('\W+',txt)
    return token

reviews_df['tokenized_Reviews'] = reviews_df['Application_Reviews'].apply(lambda x:tokenize(x.lower()))
reviews_df.head()
```

Out[300]:

	Application_Name	Application_Reviews	no_punctuation_reviews	tokenized_Reviews
0	0	Google phones focuses a lot on cameras and pho...	[G, o, o, g, l, e, , p, h, o, n, e, s, , f, ...	[google, phones, focuses, a, lot, on, cameras,...
1	0	I have had it with Google photos. I do not wan...	[I, , h, a, v, e, , h, a, d, , i, t, , w, ...	[i, have, had, it, with, google, photos, i, do...
2	0	Used this as my back-up for very important pho...	[U, s, e, d, , t, h, i, s, , a, s, , m, y, ...	[used, this, as, my, back, up, for, very, impo...
3	0	I've loved Google photos for years, but ever s...	[I, v, e, , l, o, v, e, d, , G, o, o, g, l, ...	[i, ve, loved, google, photos, for, years, but...
4	0	There is no way to select multiple media and (...	[T, h, e, r, e, , i, s, , n, o, , w, a, y, ...	[there, is, no, way, to, select, multiple, med...

Removing Stopwords

1. Using Stop words to clean the data.
2. Removing words like I, We, us, etc.
3. In the output window below, for **1 and 688 index**, comparing the column **tokenized_reviews** and **no_stopword_reviews** helps in validatinig that the data cleaning using stop word was successful

Removing Stopwords

1. using nltk module we get Stopwords library.
2. list of pre-defined stopwords are found in the list. Some of them are listed below.

In [301]:
```python
stopwords = nltk.corpus.stopwords.words('english')
stopwords[0:10]
```

Out[301]:
```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're"]
```

1. Using Stop words to clean the data.
2. Removing words like I, We, us, etc.
3. In the output window below, for **1 and 688 index**, comparing the column **tokenized_reviews** and **no_stopword_reviews** helps in validatinig that the data cleaning using stop word was successful.

In [302]:
```python
def remove_stopwords(tokenized_txt):
    clean_txt = [word for word in tokenized_txt if word not in stopwords]
    return clean_txt

reviews_df['no_stopword_reviews'] = reviews_df['tokenized_Reviews'].apply(lambda x: remove_stopwords(x))
reviews_df
```

Out[302]:

	Application_Name	Application_Reviews	no_punctuation_reviews	tokenized_Reviews	no_stopword_reviews
0	0	Google phones focuses a lot on cameras and pho...	[G, o, o, g, l, e, , p, h, o, n, e, s, , f, ...	[google, phones, focuses, a, lot, on, cameras,...	[google, phones, focuses, lot, cameras, photos...
1	0	I have had it with Google photos. I do not wan...	[I, , h, a, v, e, , h, a, d, , i, t, , w, ...	[i, have, had, it, with, google, photos, i, do...	[google, photos, want, stupid, share, link, ge...
2	0	Used this as my back-up for very important pho...	[U, s, e, d, , t, h, i, s, , a, s, , m, y, ...	[used, this, as, my, back, up, for, very, impo...	[used, back, important, photos, claiming, free...
3	0	I've loved Google photos for years, but ever s...	[I, v, e, , l, o, v, e, d, , G, o, o, g, l, ...	[i, ve, loved, google, photos, for, years, but...	[loved, google, photos, years, ever, since, 2,...
4	0	There is no way to select multiple media and (...	[T, h, e, r, e, , i, s, , n, o, , w, a, y, ...	[there, is, no, way, to, select, multiple, med...	[way, select, multiple, media, add, album, don...
...	...	...	...	...	...
685	3	Tesla	[T, e, s, l, a]	[tesla]	[tesla]
686	3	🙍‍♀️	[🙍‍♀️]	[,]	[,]
687	3	Wowwwwwwwww	[W, o, w, w, w, w, w, w, w, w, w]	[wowwwwwwwww]	[wowwwwwwwww]
688	3	This game is so cool I love it don't forget to 5	[T, h, i, s, , g, a, m, e, , i, s, , s, o, ...	[this, game, is, so, cool, i, love, it, don, t...	[game, cool, love, forget, 5]
689	3	Good game	[G, o, o, d, , g, a, m, e]	[good, game]	[good, game]

4290 rows × 5 columns

Lemmatizing

1. Lemmatization technique is used to convert all the plurals to signulars
2. It is also used to find the synonyms and assign on synonym to all similar kind of words

3. in the below output for **index 1** in column **no_stopword_reviews** and **lemmatized_reviews** we can see that **'memories' is changed to 'memory'.**
4. There may be many such changes in the dataframe which cannot be displayed in here on a small display wndow

Lemmatization of the stopwords

1. Lemmatization technique is used to convert all the plurals to signulars
2. It is also used to find the synonyms and assign on synonym to all similar kind of words
3. in the below output for **index 1** in column **no_stopword_reviews** and **lemmatized_reviews** we can see that **'memories' is changed to 'memory'.**
4. There may be many such changes in the dataframe which cannot be displayed in here on a small display wndow

In [322]:
```
def lemmatization(tokenized_txt):
    lemmatized = WordNetLemmatizer()
    lemma = " ".join([lemmatized.lemmatize(word) for word in tokenized_txt])
    return lemma

reviews_df['lemmatized_reviews']= reviews_df['no_stopword_reviews'].apply(lambda x:lemmatization(x) )
reviews_df
```

Out[322]:

	Application_Name	Application_Reviews	no_punctuation_reviews	tokenized_Reviews	no_stopword_reviews	lemmatized_reviews
0	0	Google phones focuses a lot on cameras and pho...	[G, o, o, g, l, e, , p, h, o, n, e, s, , f, ...	[google, phones, focuses, a, lot, on, cameras,...	[google, phones, focuses, lot, cameras, photos...	google phone focus lot camera photo one would ...
1	0	I have had it with Google photos. I do not wan...	[I, , h, a, v, e, , h, a, d, , i, t, , w, ...	[i, have, had, it, with, google, photos, i, do...	[google, photos, want, stupid, share, link, ge...	google photo want stupid share link get even t...
2	0	Used this as my back-up for very important pho...	[U, s, e, d, , t, h, i, s, , a, s, , m, y, ...	[used, this, as, my, back, up, for, very, impo...	[used, back, important, photos, claiming, free...	used back important photo claiming free unlimi...
3	0	I've loved Google photos for years, but ever s...	[I, v, e, , l, o, v, e, d, , G, o, o, g, l, ...	[i, ve, loved, google, photos, for, years, but...	[loved, google, photos, years, ever, since, 2,...	loved google photo year ever since 2 3 month a...
4	0	There is no way to select multiple media and (...	[T, h, e, r, e, , i, s, , n, o, , w, a, y, ...	[there, is, no, way, to, select, multiple, med...	[way, select, multiple, media, add, album, don...	way select multiple medium add album done 1 1 ...
...	...	...	...	...	...	...
684	3	I like it because it have a battery 🔋	[I, , l, i, k, e, , i, t, , b, e, c, a, u, ...	[i, like, it, because, it, have, a, battery,]	[like, battery,]	like battery
685	3	Tesla	[T, e, s, l, a]	[tesla]	[tesla]	tesla
687	3	Wowwwwwwwww	[W, o, w, w, w, w, w, w, w, w]	[wowwwwwwwww]	[wowwwwwwwww]	wowwwwwwwww
688	3	This game is so cool I love it don't forget to 5	[T, h, i, s, , g, a, m, e, , i, s, , s, o, ...	[this, game, is, so, cool, i, love, it, don, t...	[game, cool, love, forget, 5]	game cool love forget 5
689	3	Good game	[G, o, o, d, , g, a, m, e]	[good, game]	[good, game]	good game

4244 rows × 6 columns

- Data Munging

1. Since Stopwords and lemmatization can never work 100%, there are few data that will be left behind.
2. Here in the below snippet, on the lemmatized column, we are trying to remove all those rows which have blank values
3. Blank values or empty spaces will create no contribution anywhere so it's better to remove such data

Performing Data munging to further clean the data

1. Since Stopwords and lemmatization can never work 100%, there are few data that will be left behind.
2. Here in the below snippet, on the lemmatized column, we are trying to remove all those rows which have blank values
3. Blank values or empty spaces will create no contribution anywhere so it's better to remove such data

In [323]:
```
# taking a count of those records with blank data or space character in the record

reviews_df[(reviews_df['lemmatized_reviews']=='')|(reviews_df['lemmatized_reviews']==' ')].count()
```

Out[323]:
```
Application_Name          0
Application_Reviews       0
no_punctuation_reviews    0
tokenized_Reviews         0
no_stopword_reviews       0
lemmatized_reviews        0
dtype: int64
```

In [324]:
```
# Converting all the spaces to numpy NaN value

reviews_df['lemmatized_reviews'].replace(['',' '],np.NaN,inplace=True)
reviews_df.isnull().sum()
```

Out[324]:
```
Application_Name          0
Application_Reviews       0
no_punctuation_reviews    0
tokenized_Reviews         0
no_stopword_reviews       0
lemmatized_reviews        0
dtype: int64
```

In [325]:
```
# there were 46 rows with NaN value
#dropping the Null values and checking the final dataset

reviews_df.dropna(inplace=True)
reviews_df.isnull().sum()
```

Out[325]:
```
Application_Name          0
Application_Reviews       0
no_punctuation_reviews    0
tokenized_Reviews         0
no_stopword_reviews       0
lemmatized_reviews        0
dtype: int64
```

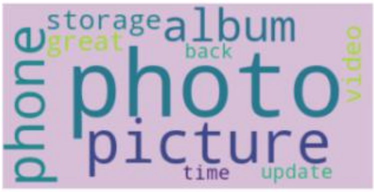
- Word Cloud

Word Cloud for Displaying Top 10 Words

1. Using all the above dataframes one at a time to craete a word cloud
2. Maximum words to be displayed in the word cloud id 10
3. There are 4 different word clouds generated for 4 different company apps
4. Using wordcloud, matplotlib, stopword packages together, all 4 word clouds are shown below

In [331]:
```
# Creating google word cloud

#joining all the rows into 1 single row so that it forms a string to be read an calucate the top 10 words
google_text= ''.join(reviews for reviews in only_google_df.lemmatized_reviews)
print('Combined total number of word counts for Google is {}'.format(len(google_text)))

from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt

# adding some more words to the stopword (which we felt is of no significance) list and appending it with the pre-defiined ones
stopwords = ['one','want','way','pic','app','google','thing'] + list(STOPWORDS)

# generating the word cloоud
wordcloud = WordCloud(stopwords =stopwords, max_font_size=100, max_words=10, background_color='thistle').generate(google_text)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.figure(figsize=[20,10])
plt.show()
```

```
Combined total number of word counts for Google is 108472
```




```
<Figure size 1440x720 with 0 Axes>
```

In [332]:
```
# Creating Amazon word cloud

#joining all the rows into 1 single row so that it forms a string to be read an calucate the top 10 words
amazon_text= ''.join(reviews for reviews in only_amazon_df.lemmatized_reviews)
print('Combined total number of word counts for Amazon is {}'.format(len(amazon_text)))


from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt

# adding some more words to the stopword (which we felt is of no significance) list and appending it with the pre-defiined ones
stopwords = ['app','pay','one','time'] + list(STOPWORDS)

# generating the word cloud
wordcloud = WordCloud(stopwords =stopwords, max_font_size=100, max_words=10, background_color='moccasin').generate(amazon_text)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.figure(figsize=[20,10])
plt.show()
```

```
Combined total number of word counts for Amazon is 123023
```


```
<Figure size 1440x720 with 0 Axes>
```

In [336]:
```python
# Creating Linkedin word cloud

#joining all the rows into 1 single row so that it forms a string to be read an calucate the top 10 words
linkedin_text= ''.join(reviews for reviews in only_linkedin_df.lemmatized_reviews)
print('Combined total number of word counts for LinekedIN is {}'.format(len(linkedin_text)))

from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt

# adding some more words to the stopword (which we felt is of no significance) list and appending it with the pre-defiined ones
stopwords = ['app','time','use','way','new'] + list(STOPWORDS)

# generating the word cloud
wordcloud = WordCloud(stopwords =stopwords, max_font_size=100, max_words=10, background_color='turquoise').generate(linkedin_text
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.figure(figsize=[20,10])
plt.show()
```

```
Combined total number of word counts for LinekedIN is 139988
```


```
<Figure size 1440x720 with 0 Axes>
```

In [334]:
```python
# Creating Tesla word cloud

#joining all the rows into 1 single row so that it forms a string to be read an calucate the top 10 words
tesla_text= ''.join(reviews for reviews in only_tesla_df.lemmatized_reviews)
print('Combined total number of word counts for Tesla is {}'.format(len(tesla_text)))

from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt

# adding some more words to the stopword (which we felt is of no significance) list and appending it with the pre-defiined ones
stopwords = ['app','use','way','new','ad','bad','much','lot','really','many'] + list(STOPWORDS)

# generating the word cloud
wordcloud = WordCloud(stopwords =stopwords, max_font_size=100, max_words=10, background_color='lavender').generate(tesla_text)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.figure(figsize=[20,10])
plt.show()
```

```
Combined total number of word counts for Tesla is 29991
```


```
<Figure size 1440x720 with 0 Axes>
```