



# EMOTION ANALYSIS IN AUDIO CONVERSATION

MECE-6397 PROJECT



# INTRODUCTION

The goal of this research work is to use machine learning algorithms to predict the emotion of a person who is contacting customer care for their product queries. In this project, the audio is divided into chunks based on customer and caller. Now the audio is converted into text using Google API. This text sentiment is trained using logistic regression. With this the converted text from the audio is tested. Also, the audio chunks are trained using models like KNN, Random Forest and SVM and the accuracies are compared.

# OBJECTIVE

- The goal of this research work is to use machine learning algorithms to predict the emotion of a person who is contacting customer care for their product queries.
- Our work proposes a novel method of predicting the emotions of the customer, both pitch-wise and the words used.
- Audio is divided into chunks using audio segmentation and audio is converted into text to classify both text and audio chunks.



# PREVIOUS WORKS

In the past, researchers have shown great interest in the text conversion of audio and analyzing the text intensity based on the words used and classifying the emotions with the words, with this intensity of customer pitch is not considered which is a major component to predict the emotion.

Researchers evolved the project by analyzing the intensity in the voice with this soft-spoken people are always considered calm and loud spoken as harsh irrespective of the language they use. To solve these problems, I tried to create an hybrid model to satisfy the problem statement



# RESEARCH QUESTION

Usually, in emotion classification, researchers consider the acoustic features alone. For strong emotions like anger and surprise, the pitch and energy of the acoustic features are both high. In such cases, it is very difficult to predict emotions correctly using acoustic features alone. But, if we classify speech solely on its textual component, we will not obtain a clear picture of the emotional content. In the proposed hybrid approach we consider both text and audio features. Fig shows the framework for the hybrid approach.



# DATA COLLECTION

I am using the dataset from RAVDESS for classification purpose of audio chunks obtained. It consists of 24 professional actors (12 female, 12 male), vocalizing two lexically-matched statements in a neutral North American accent. This portion of the RAVDESS contains 1440 files: 60 trials per actor x 24 actors = 1440. Speech emotions includes calm, happy, sad, angry, fearful, surprise, and disgust expressions. Each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression.

Dataset consists of different emotion like - neutral, calm, happy, sad, angry, fearful, disgust, surprised

the labeled descriptions for each dataset in the same order as the label categories in the datasets. For instance, if an observation in SCv1 has label 1, then that observation is in the sarcasm category. Multiple emotions like happy calm surprised were classified as positive and angry fearful disgust were classified as negative emotion.

Twitter dataset of sentiment analysis of text documents is chosen. Assigning a sentiment score to each phrase and component (-1 to +1)

# METHODOLOGY

Our methodology is divided in 4 parts-

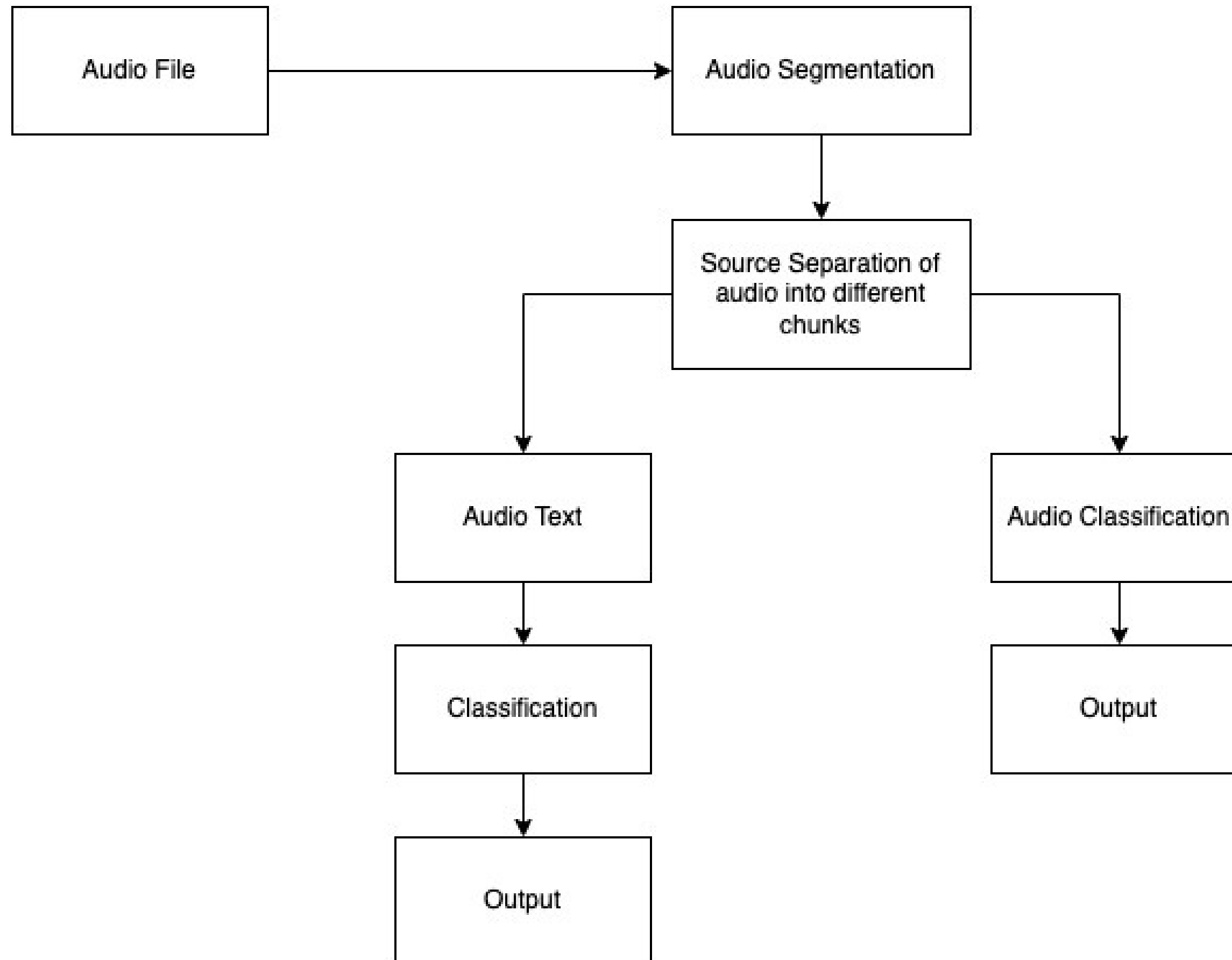
1. Source Separation of audio into different chunks

[Separating customer audio]

2. Conversion of audio to text

3. Classification based on audio

4. Classification based on Text





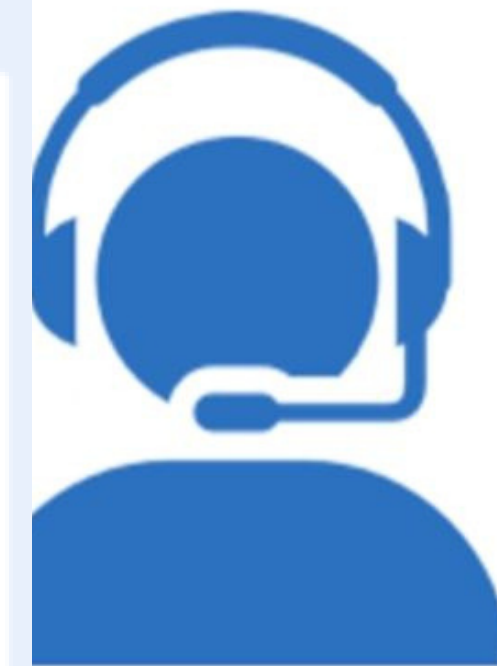
# METHODOLOGY

The first phase of our system is segmentation of the audio into parts using Google's webrtc Voice Activation Detection . The splitted parts of audio are run through Adaptive MAP estimation to find the customer and remove the audio chunk of company caller using the generated super vectors.



# METHODOLOGY

Once the audio file of conversation between the customer and call center agent is divided into chunks, we pass it through Google API to convert the customer chunk audio to text which divides our project into two arena.

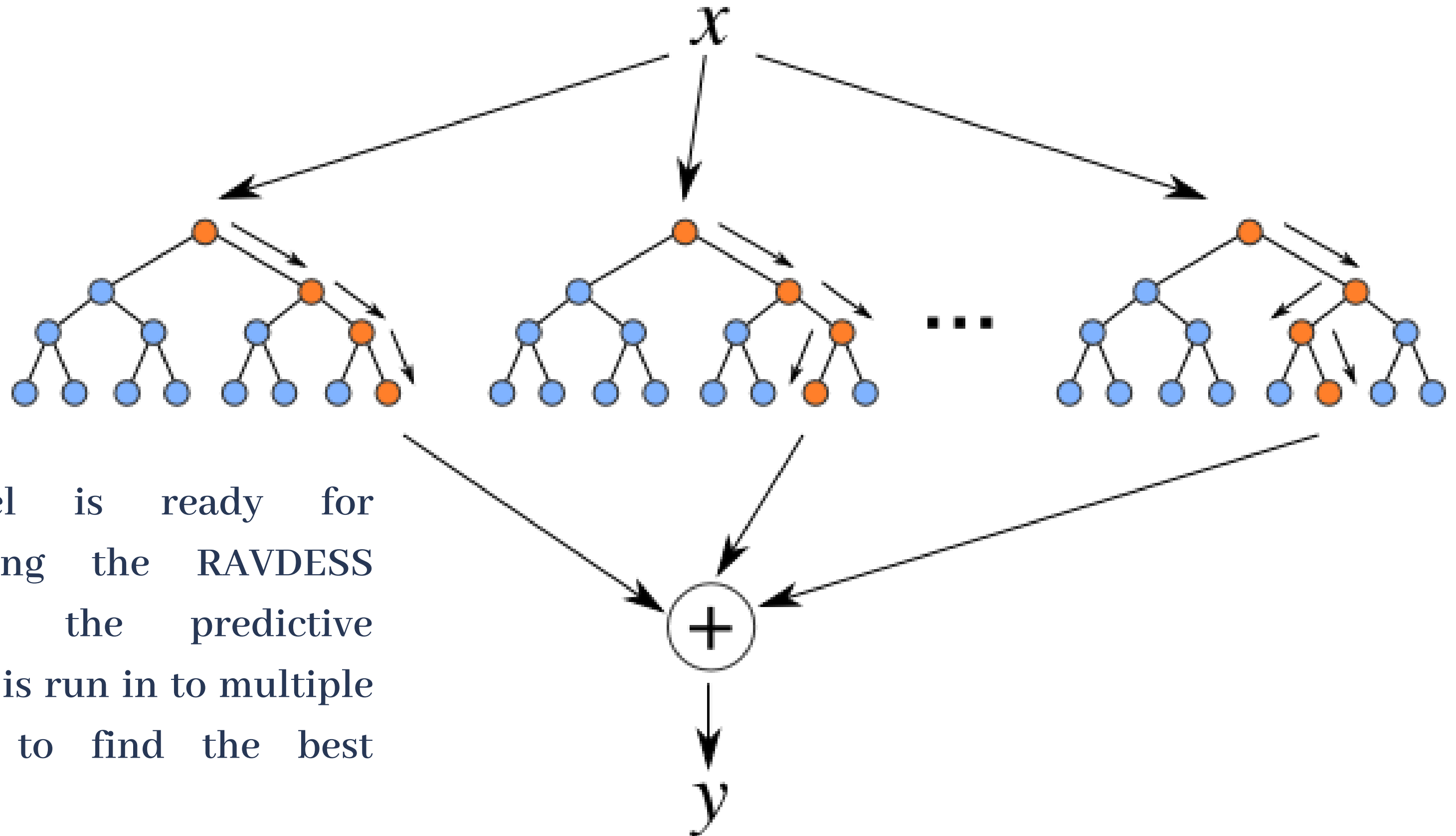


Speech



Text

# METHODOLOGY



Now the model is ready for classification, Using the RAVDESS emotion dataset the predictive sentiment analysis is run in to multiple classifier models to find the best accuracy of all.

# METHODOLOGY

And the text is passed into text sentiment analysis model built using logistic regression trained with nplk twitter dataset to predict the emotion of the customer





A dark blue background featuring a close-up, slightly blurred image of a microscope. The microscope's objective lenses and eyepiece are visible. On one of the lenses, the text "N PLAN" and "20X/0.40" is printed in a light color. In the top right corner, there are three small white circles, with the rightmost one being filled, suggesting a navigation or status indicator.

# SIMULATIONS & RESULTS

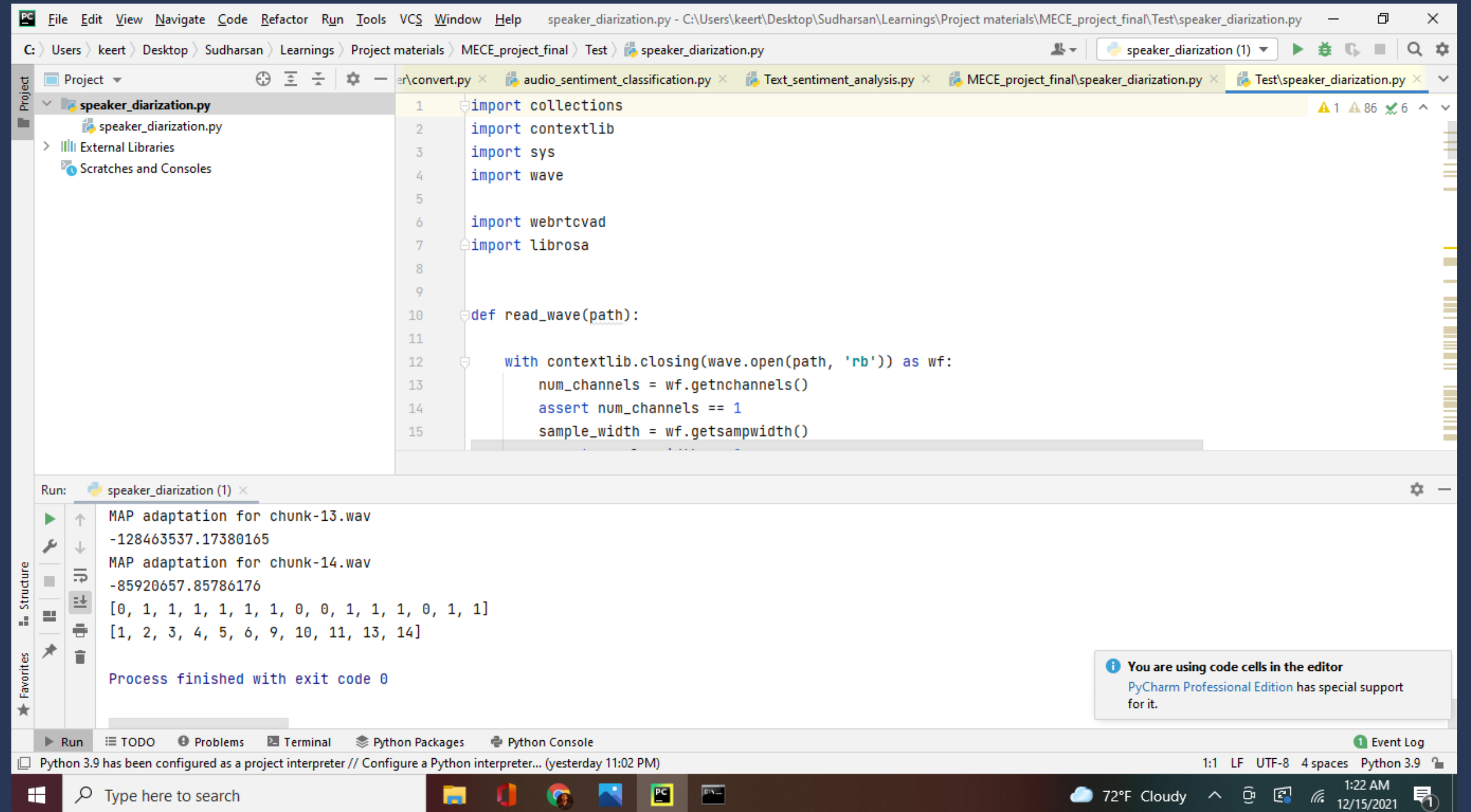
COMPARISON AND INFERENCE

# SIMLUATION & RESULTS-1



Through VAD detection, segments of an audio file are separated. These segments contain voiced data, which are then passed on to the Adaptive MAP estimation process to obtain the super vector.

# SIMULATION & RESULTS-1



The screenshot displays the PyCharm IDE interface. The main editor window shows a Python script named `speaker_diarization.py` with the following code:

```
1 import collections
2 import contextlib
3 import sys
4 import wave
5
6 import webrtcvad
7 import librosa
8
9
10 def read_wave(path):
11
12     with contextlib.closing(wave.open(path, 'rb')) as wf:
13         num_channels = wf.getnchannels()
14         assert num_channels == 1
15         sample_width = wf.getsampwidth()
```

The Run tool window at the bottom shows the output of the script:

```
MAP adaptation for chunk-13.wav
-128463537.17380165
MAP adaptation for chunk-14.wav
-85920657.85786176
[0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1]
[1, 2, 3, 4, 5, 6, 9, 10, 11, 13, 14]

Process finished with exit code 0
```

A notification bubble in the bottom right corner states: "You are using code cells in the editor. PyCharm Professional Edition has special support for it."

The status bar at the bottom indicates the file encoding is UTF-8, the line length is 4 spaces, and the Python interpreter is Python 3.9.

# SIMLUATION & RESULTS-2

## Sample Output

chunk-01.wav I am falling from market I am  
falling from market

chunk-06.wav Hanuman Mahima Mama ji  
great

chunk-09.wav Anjali Sharma spoken to  
nahin number in about drawing about  
school number about

chunk-14.wav Vikram 951 number phone  
number 951 number phone number



# SIMULATION & RESULTS-2

The screenshot displays the PyCharm IDE interface. The main editor window shows a Python script named `audio-text.py` located at `C:\Users\keert\Desktop\Sudharsan\Learnings\Project materials\Test recordings\audio-text.py`. The script imports `speech_recognition` as `sr` and `soundfile` as `sf`. It defines a function `transcribe` that takes `audio_file`, `loops`, and `starting_increment` as arguments. The function initializes `audio_text1` as an empty list, sets `increment` to `starting_increment`, and enters a `for` loop over `range(loops)`. Inside the loop, it creates a `sr.Recognizer()` object, opens the audio file as a source, records audio with an offset of `increment` and a duration of 10 seconds, recognizes the audio using Google's recognizer, appends the resulting string to `audio_text1`, and increments the `increment` by 10.

The Run window at the bottom shows the execution of the script. The command executed is `"C:\Program Files\Python39\python.exe" "C:/Users/keert/Desktop/Sudharsan/Learnings/Project materials/Test recordings/audio-text.py"`. The output is `hello can I help you I would like information on purchasing a new vehicle ding dong I am looking for the intent and cutting which location is looking for the t`. The process finished with exit code 0.

A notification bubble in the bottom right corner states: "You are using code cells in the editor. PyCharm Professional Edition has special support for it."

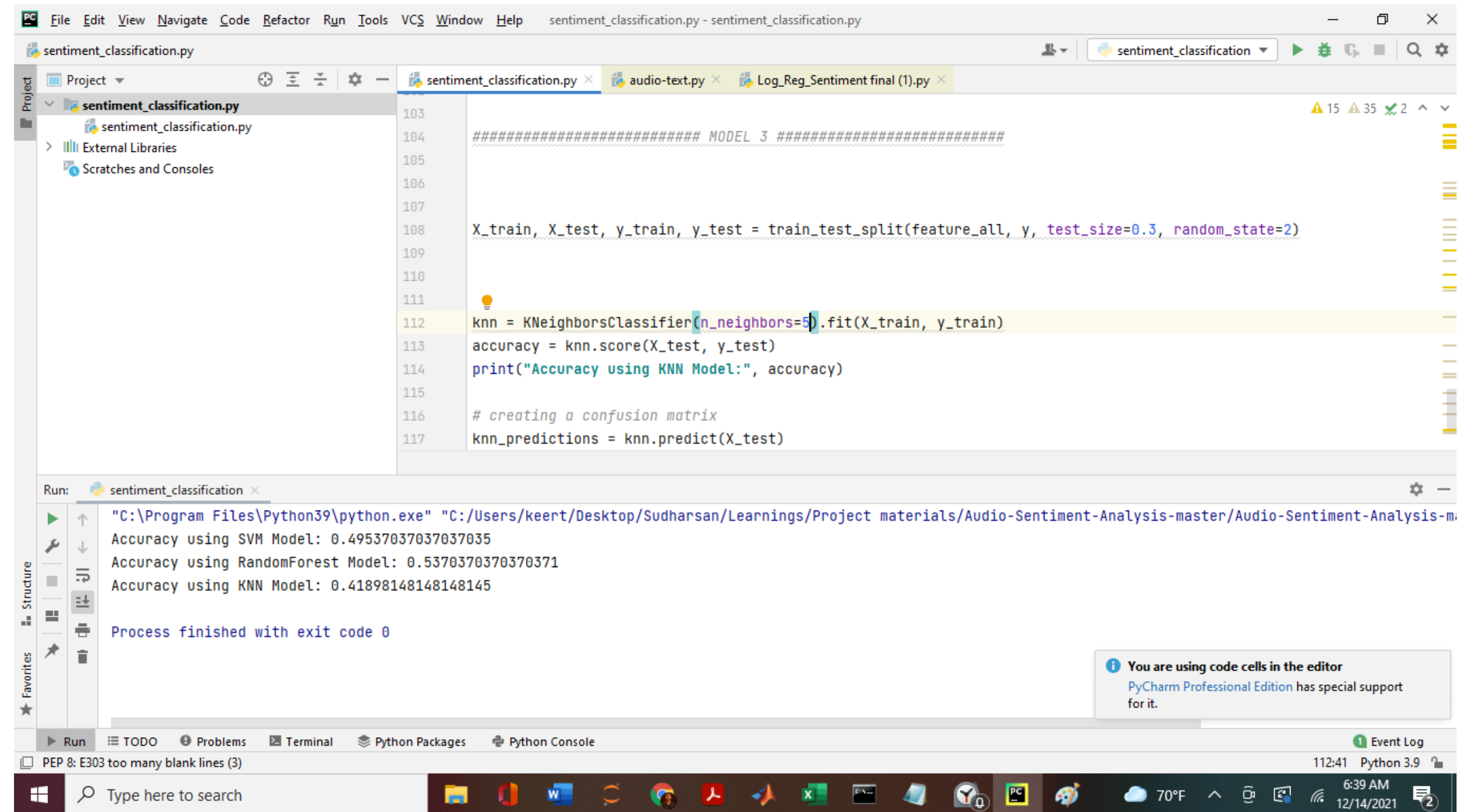
The status bar at the bottom indicates the file encoding is UTF-8, 4 spaces, and the Python interpreter is Python 3.9.

# SIMLUATION & RESULTS-3

1. SVM- kernel= “poly”;  
degree=10; c=1000

2. Random forest  
n\_estimators=100,  
criterion='entropy',  
random\_state=58

3. KNN, nearest neighbour=5



The screenshot displays the PyCharm IDE interface. The main editor window shows the file `sentiment_classification.py` with the following code:

```
##### MODEL 3 #####  
  
X_train, X_test, y_train, y_test = train_test_split(feature_all, y, test_size=0.3, random_state=2)  
  
knn = KNeighborsClassifier(n_neighbors=5).fit(X_train, y_train)  
accuracy = knn.score(X_test, y_test)  
print("Accuracy using KNN Model:", accuracy)  
  
# creating a confusion matrix  
knn_predictions = knn.predict(X_test)
```

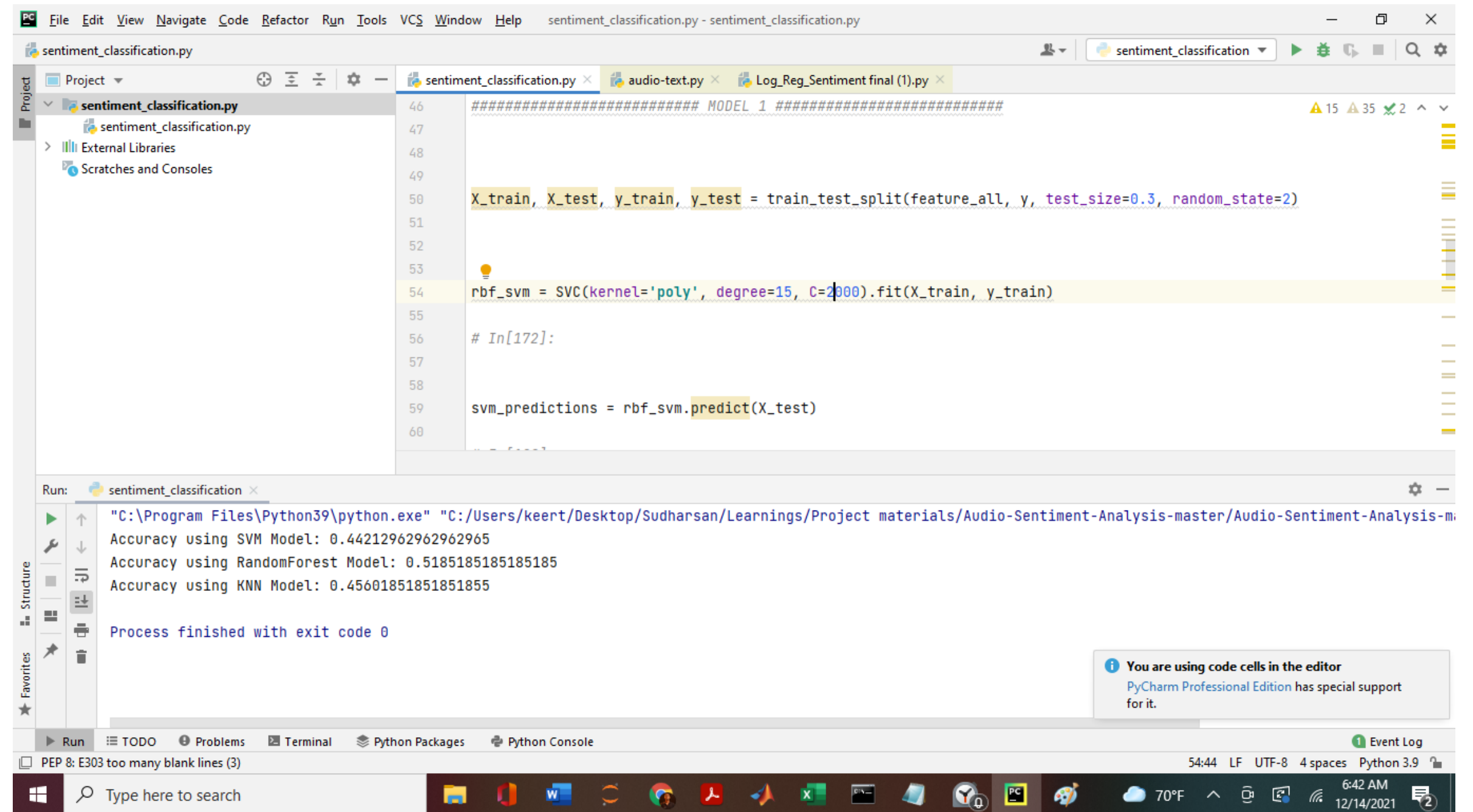
The Run console at the bottom shows the output of the script:

```
"C:\Program Files\Python39\python.exe" "C:/Users/keert/Desktop/Sudharsan/Learnings/Project materials/Audio-Sentiment-Analysis-master/Audio-Sentiment-Analysis-m  
Accuracy using SVM Model: 0.49537037037037035  
Accuracy using RandomForest Model: 0.5370370370370371  
Accuracy using KNN Model: 0.41898148148148145  
  
Process finished with exit code 0
```

A notification bubble in the bottom right corner states: "You are using code cells in the editor. PyCharm Professional Edition has special support for it."

# SIMLUATION & RESULTS-3

- 1.SVM- kernel= “poly”;  
degree=15; c=2000
- 2.Random forest  
n\_estimators=50,  
criterion='entropy',  
random\_state=46
- 3.KNN, nearest neighbour=3



The screenshot displays the PyCharm IDE interface. The main editor window shows the file `sentiment_classification.py` with the following code:

```
##### MODEL 1 #####  
  
X_train, X_test, y_train, y_test = train_test_split(feature_all, y, test_size=0.3, random_state=2)  
  
rbf_svm = SVC(kernel='poly', degree=15, C=2000).fit(X_train, y_train)  
  
# In[172]:  
  
svm_predictions = rbf_svm.predict(X_test)
```

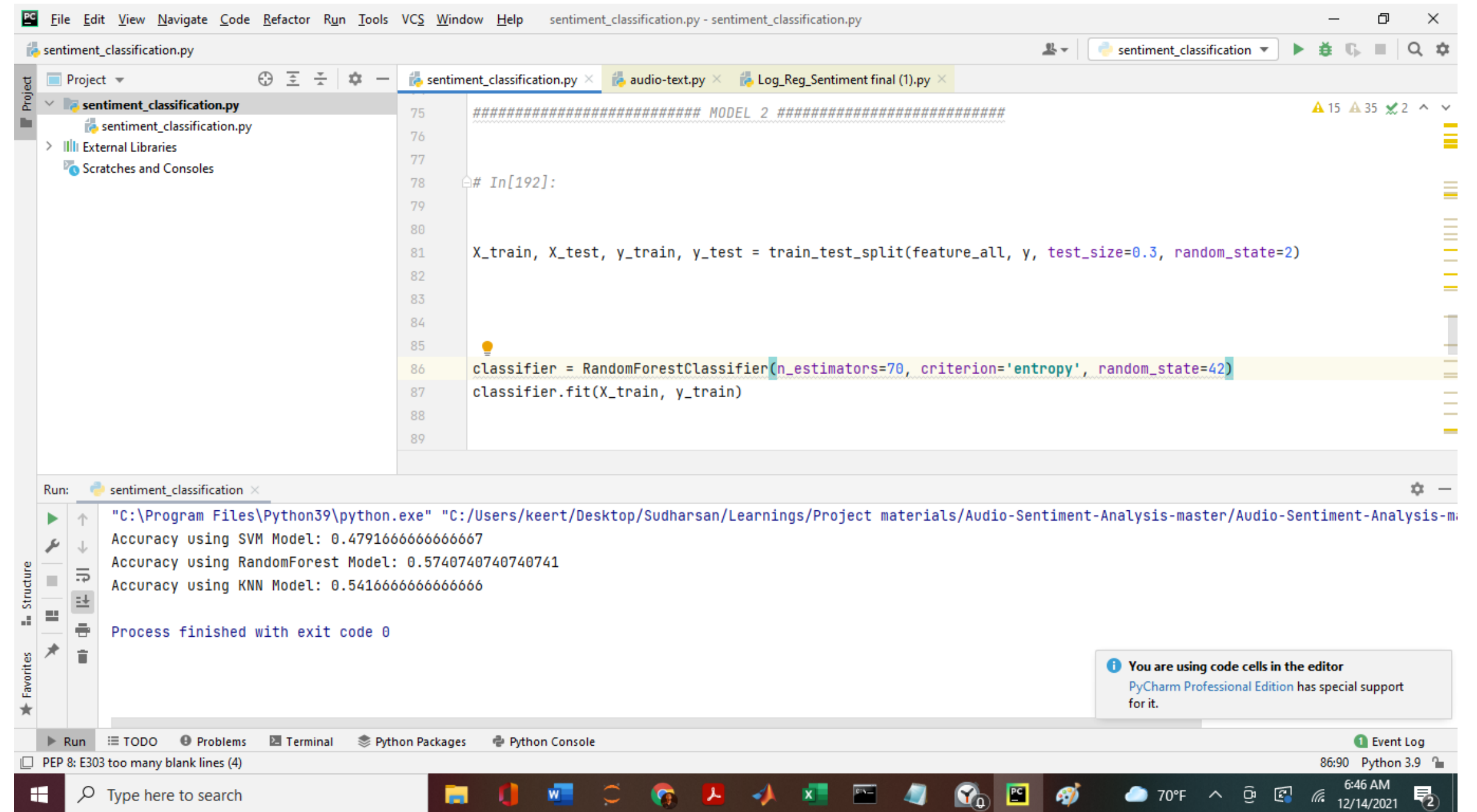
The Run window at the bottom shows the output of the code execution:

```
"C:\Program Files\Python39\python.exe" "C:/Users/keert/Desktop/Sudharsan/Learnings/Project materials/Audio-Sentiment-Analysis-master/Audio-Sentiment-Analysis-m  
Accuracy using SVM Model: 0.44212962962965  
Accuracy using RandomForest Model: 0.5185185185185  
Accuracy using KNN Model: 0.45601851851855  
  
Process finished with exit code 0
```

A notification at the bottom right states: "You are using code cells in the editor. PyCharm Professional Edition has special support for it."

# SIMLUATION & RESULTS-3

1. SVM- kernel= “poly”;  
degree=12; c=5000
2. Random forest  
n\_estimators=70,  
criterion='entropy',  
random\_state=42
3. KNN, nearest neighbour=1



The screenshot displays the PyCharm IDE interface. The main editor window shows the file `sentiment_classification.py` with the following code:

```
##### MODEL 2 #####  
# In[192]:  
  
X_train, X_test, y_train, y_test = train_test_split(feature_all, y, test_size=0.3, random_state=2)  
  
classifier = RandomForestClassifier(n_estimators=70, criterion='entropy', random_state=42)  
classifier.fit(X_train, y_train)
```

The Run console at the bottom shows the output of the execution:

```
"C:\Program Files\Python39\python.exe" "C:/Users/keent/Desktop/Sudharsan/Learnings/Project materials/Audio-Sentiment-Analysis-master/Audio-Sentiment-Analysis-m  
Accuracy using SVM Model: 0.4791666666666667  
Accuracy using RandomForest Model: 0.5740740740740741  
Accuracy using KNN Model: 0.5416666666666666  
  
Process finished with exit code 0
```

A notification at the bottom right states: "You are using code cells in the editor. PyCharm Professional Edition has special support for it."



# SIMLUATION & RESULTS-4

Given the test data and the weights of trained model, we calculated the accuracy of our logistic regression model.

- Using `predict_tweet()` function to make predictions on each tweet in the test set.
- If the prediction is  $> 0.5$ , set the model's classification `y_hat` to 1, otherwise set the model's classification `y_hat` to 0.
- A prediction is accurate when `y_hat` equals `test_y`. Sum up all the instances when they are equal and divide by `m`.

# SIMLUATION & RESULTS-4



Accuracy for Logistic regression= 0.9834

Output for chunk 09 and chunk 06

['anjali', 'sharma', 'spoken', 'nahin', 'number',  
'draw', 'school', 'number']

[[0.50014869]]

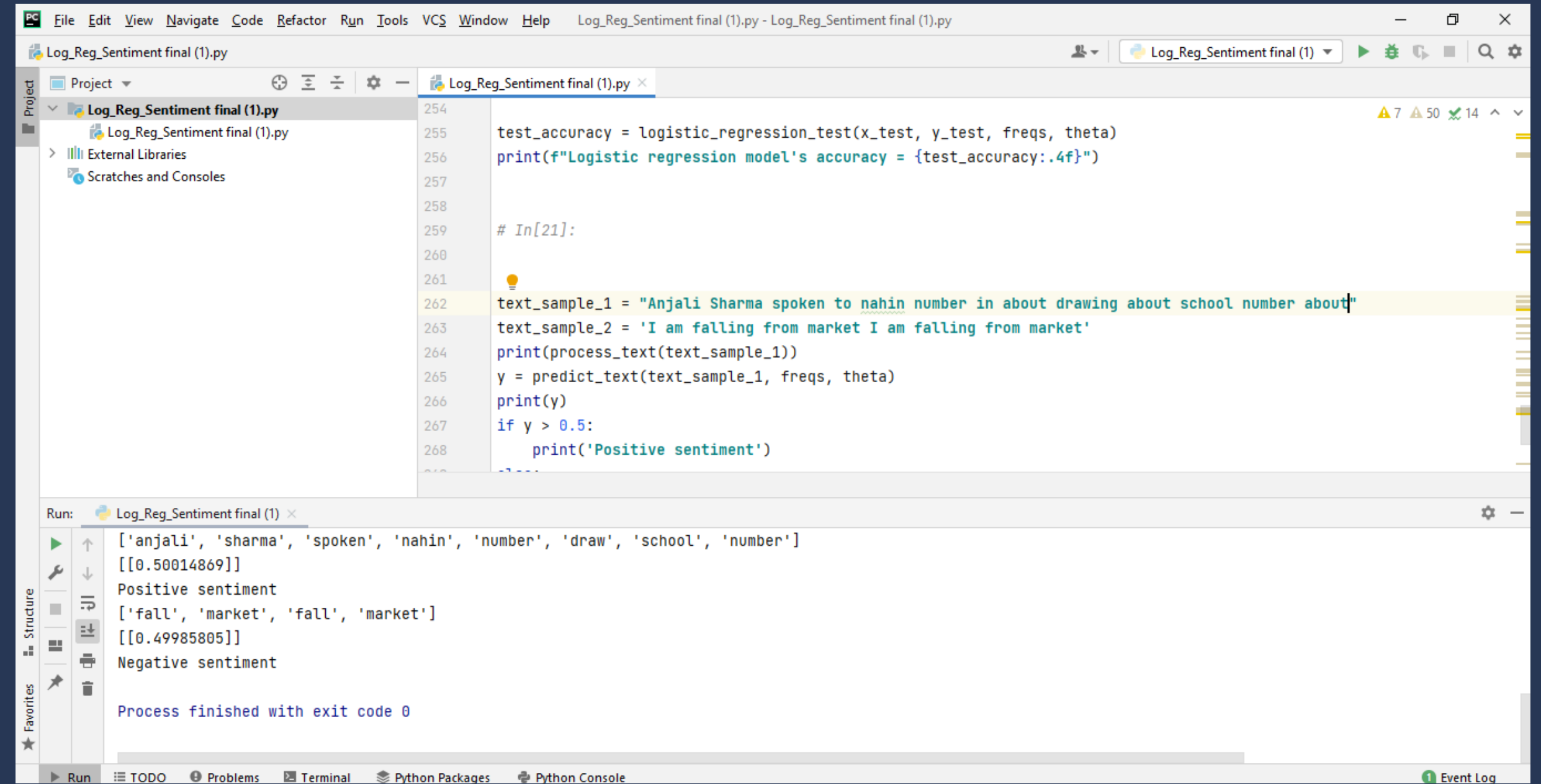
Positive sentiment

['fall', 'market', 'fall', 'market']

[[0.49985805]]

Negative sentiment

# SIMLUATION & RESULTS-4



The screenshot displays a Python IDE with a project named "Log\_Reg\_Sentiment final (1).py". The code in the editor defines a function `logistic_regression_test` and uses it to process two text samples. The first sample is "Anjali Sharma spoken to nahin number in about drawing about school number about" and the second is "I am falling from market I am falling from market". The code prints the word frequency lists, the predicted sentiment (0.50014869 for the first sample, 0.49985805 for the second), and the final sentiment classification (Positive for the first, Negative for the second).

```
254 test_accuracy = logistic_regression_test(x_test, y_test, freqs, theta)
255 print(f"Logistic regression model's accuracy = {test_accuracy:.4f}")
256
257
258
259 # In[21]:
260
261
262 text_sample_1 = "Anjali Sharma spoken to nahin number in about drawing about school number about"
263 text_sample_2 = 'I am falling from market I am falling from market'
264 print(process_text(text_sample_1))
265 y = predict_text(text_sample_1, freqs, theta)
266 print(y)
267 if y > 0.5:
268     print('Positive sentiment')
```

The Run console shows the following output:

```
[ 'anjali', 'sharma', 'spoken', 'nahin', 'number', 'draw', 'school', 'number' ]
[[0.50014869]]
Positive sentiment
[ 'fall', 'market', 'fall', 'market' ]
[[0.49985805]]
Negative sentiment

Process finished with exit code 0
```

# FUTURE ASPECTS



- A model is to be developed to get the weighted average of the accuracies calculated from both the classifications of audio chunks and texts derived from the audio which is now limited.
- Accuracy of audio classification must be improved as the major part emotion will be displayed in the pitch of the customer.
- Audio can be further divided into smaller chunks where the conversation of customers can be divided into smaller parts to run through reinforcement learning and train the initial part to test the latter parts.
- In this way pitch of the person can be identified as the pitch is a personal trait and a louder mouth can be classified and not directly noted as anger person.
- This will help the model to improve the accuracy invariably.
- This will be a stepping stone for machines to take over customer care which will reduce the manpower in the industry.



THANK YOU

