```python
import pyttsx3
import speech_recognition as sr
import subprocess
import time
import soundfile as sf
import numpy as np

# Initialize the speech engine
engine = pyttsx3.init()
engine.setProperty('rate', 150)  # Speed of speech

def speak(text):
    engine.say(text)
    engine.runAndWait()

# Record your voice samples manually
reference_files = ["my_voice_1.wav", "my_voice_2.wav", "my_voice_3.wav"]  # Replace with your actual filenames

# Voice verification: Compare incoming audio to the recorded reference voices
def is_user_voice(audio_data):
    with open("temp_user.wav", "wb") as f:
        f.write(audio_data.get_wav_data())

    # Load the user input and compare with reference voices
    user_voice, _ = sf.read("temp_user.wav")

    for reference_file in reference_files:
        reference_voice, _ = sf.read(reference_file)

        # Match length for comparison
        min_len = min(len(reference_voice), len(user_voice))
        reference_voice = reference_voice[:min_len]
        user_voice = user_voice[:min_len]

        # Compare the audio waveforms directly
        if np.allclose(reference_voice, user_voice, atol=0.1):
            return True
    return False

def take_command():
    recognizer = sr.Recognizer()
    with sr.Microphone() as source:
        print("рцЂ Listening...")
        recognizer.adjust_for_ambient_noise(source, duration=1)  # Adjust for background noise
        recognizer.pause_threshold = 0.8  # Allow slight pause in speech
        recognizer.energy_threshold = 300  # Minimum energy level to consider as speech
        audio = recognizer.listen(source, phrase_time_limit=5)  # Limit to 5 seconds of speech

    print("вЏі Recognizing...")
    try:
        text = recognizer.recognize_google(audio)
        print("вњ… You said:", text)
        return text.lower()
    except sr.UnknownValueError:
        print("вќЊ Sorry, I couldn't understand the audio.")
        speak("Sorry, I didn't catch that.")
        return ""
    except sr.RequestError:
        print("вќЊ Could not connect to Google Speech Recognition service.")
        speak("Speech service is unavailable.")
        return ""

def write_to_notepad(text):
    with open("speech_output.txt", "w") as file:
        file.write(text)
    subprocess.Popen(["notepad.exe", "speech_output.txt"])

def main():
    speak("Please verify your voice to activate.")
    r = sr.Recognizer()

    # Voice verification loop
    while True:
        with sr.Microphone() as source:
            r.adjust_for_ambient_noise(source)
            audio = r.listen(source)

        if is_user_voice(audio):
            speak("Voice matched. Access granted.")
            print("Voice matched. Assistant activated.")
            break  # Exit the loop if voice is verified
        else:
            speak("Voice did not match. Please try again.")
            print("Voice did not match. Retry.")

    speak("Jarvis activated")
    print("Jarvis activated")

    # Main command loop
    while True:
        command = take_command()

        if "open settings" in command:
            speak("Opening Settings")
            subprocess.Popen("start ms-settings:", shell=True)

        elif "open file explorer" in command:
            speak("Opening File Explorer")
            subprocess.Popen("explorer")

        elif "open command prompt" in command:
            speak("Opening Command Prompt")
            subprocess.Popen("start cmd", shell=True)

        elif "write my speech" in command:
            speak("Speak something, I will write it in notepad")
            time.sleep(1)
            spoken_text = take_command()
            if spoken_text:
                write_to_notepad(spoken_text)
                speak("I have written your speech in notepad")

        elif "exit" in command or "quit" in command:
            speak("Goodbye!")
            break

if __name__ == "__main__":
    main()
```