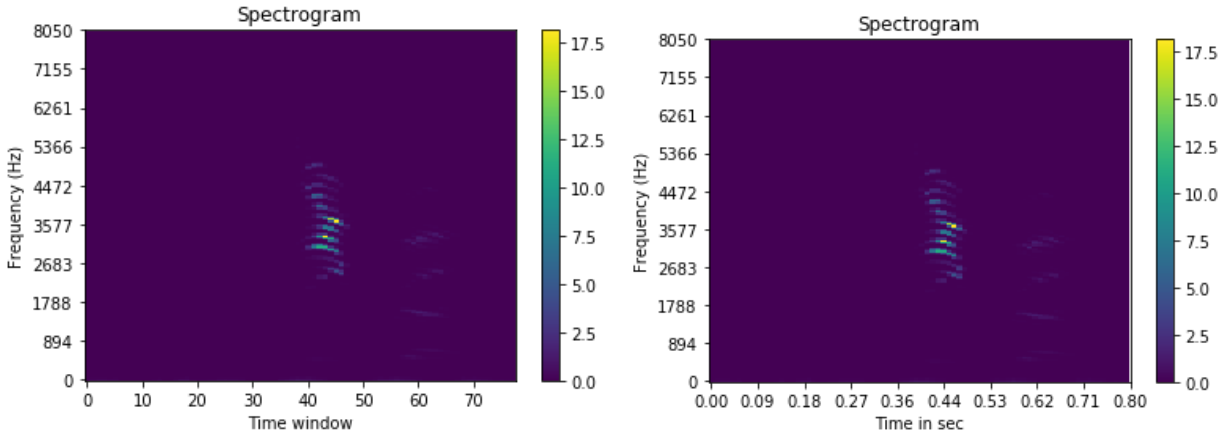
ASSIGNMENT 2

SAJAG AGARWAL, 2016185

Q1.

Methodology:

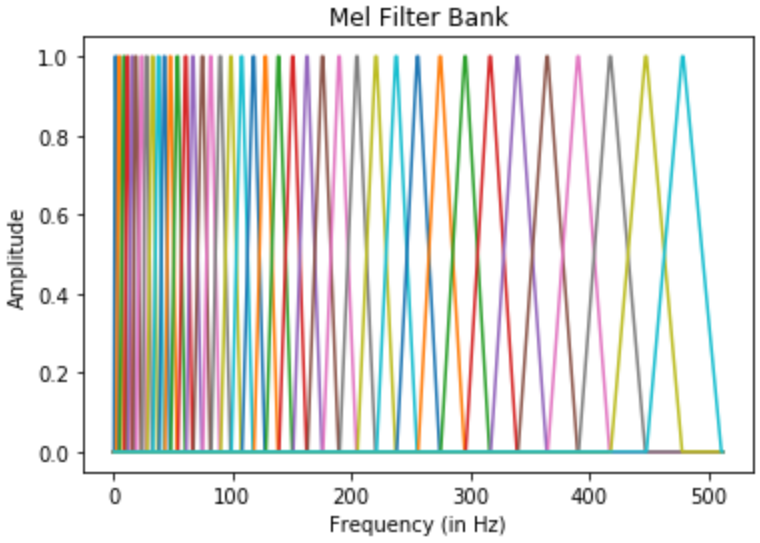
- Initially, librosa.load is used to load the audios as normalized floating point time series with the same sampling rate as the signal. On the other hand, wavfile.read returns an int time series which is not normalized.
- librosa.display.waveplot is then used to visualize the audio signal in the time domain.
- Speech signal is now divided into frames using a window of length 20ms. Each frame is of length = (20 * 0.001)seconds * sampling frequency of signal. Overlap is taken to be 50%, that is 10ms. All signals are zero padded to make sure no sample is lost.
- To handle discontinuity of the signal, the frames are multiplied by Hanning window H, which is given by:
  $H[n] = 0.5 - 0.5\ \frac{cos(2\pi n)}{N-1},\ where\ 0 < n < N$
- FFT is applied to these frames. Since the input is purely real, negative frequency terms are not considered and thus the length of the fourier transform is window_length/2 + 1.
- DFT X[k] for signal x[n] is given by:
  $X[k] = \sum_{n=0}^{N-1} x[n]e^{\frac{-j2\pi kn}{N}}$, where N = window length
- Since DFT is slow, FFT is used which recursively separates the Fourier transform into even and odd indices until a certain length is reached and applies DFT to the smaller matrices. FFT is given by:
  $X[k] = x_{even}[k] + x_{odd}[k]e^{\frac{-j2\pi k}{N}}$
- Power spectrum of the obtained by squaring the absolute value of FFT. We thus obtain spectrograms for the signals. The shape of spectrogram is given by: (window_length/2 + 1, number of frames)
- To plot the spectrogram, plt.imshow is used. Frequency window is converted to Hz by initialising the ticks using the following formula:
  nth bin in FFT (Hz) = n*sampling frequency/size of FFT. Here, the initialised ticks elements are used as n.
  Spectrogram for audio signal '`004ae714_nohash_0.wav`' (in training folder/zero) is:



Q2.

Methodology:

- Initially, librosa.load is used to load the audios as normalized floating point time series with the same sampling rate as the signal. On the other hand, wavfile.read returns an int time series which is not normalized.
- **Pre-emphasis:** The audio signal is passed through a filter to increase energy at high frequencies. Output Y for signal X is given as:
  Y[n] = X[n] - 0.95*X[n-1]
- **Framing:** Speech signal is now divided into frames using a window of length 20ms. Each frame is of length = (20 * 0.001)seconds * sampling frequency of signal. Overlap is taken to be 50%, that is 10ms. All signals are zero padded to make sure no sample is lost.
- **Hamming Windowing:** To handle discontinuity of the signal, the frames are multiplied by Hamming window H, which is given by:
  $H[n] = 0.54 - 0.46 \frac{cos(2\pi n)}{N-1}, \; where \; 0 < n < N$
- **FFT:** 2048 point FFT is applied to these frames. Since the input is purely real, negative frequency terms are not considered and thus the length of the fourier transform is window_length/2 + 1.
- **Mel Filter Bank Processing:** Next step involves computing Mel filter banks for the signal. Filterbank is calculated for the whole frequency band of the signal. For this, first the lowest and highest frequency of signal are calculated which are then converted to mel space. To convert frequency f to mel m and vice versa, we have:
  $m = 2595 log_{10}(1 + \frac{f}{100})$
  $f = 700(10^{m/2595} - 1)$
  Now, a linearly spaced array between these 2 mels is converted to frequency domain. These frequencies are then normalized to nearest fft bin as: floor((NFFT+1)*f_i/sampling frequency). Now, we create our filter banks such that each filter's magnitude frequency response is triangular in shape and equal to 1 at the centre frequency and decreases linearly to zero at centre frequency of two adjacent filters.



These filterbanks are applied to signal power spectrogram and log is taken, followed by multiplication by 20 to construct log filter banks

- **Discrete cosine transform:** DCT-II is applied to the log filter banks to obtain Mel Frequency Cepstrum Coefficients. This decorrelates the coefficients of filter banks and also compresses them. 12 cepral coefficients are preserved and rest are discarded. Energy given by: $\sum x^2(t)$ is used as the 13th feature. 13 Delta and 13 delta delta features are calculated as follows: $d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N}(n^2)}$, where d_t is delta coefficient, N is number of neighbors and c represents the static coefficients. For delta features, cepral coefficients act as c while for delta delta features, delta features act as c. Thus, a vector of size (number of frames,39) is obtained for each signal.
- A feature vector containing 41 cepral coefficients for a signal is also stored.

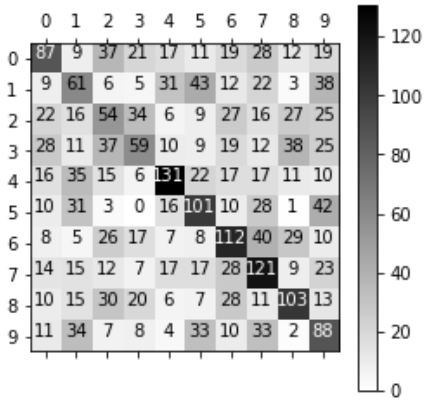
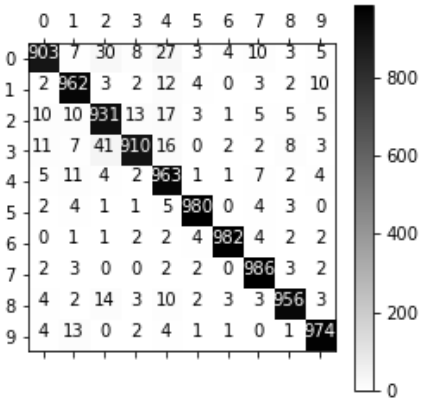
Q3.

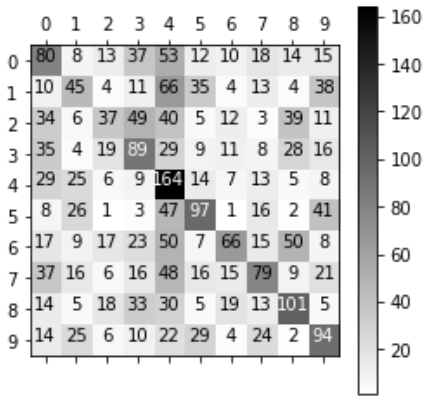
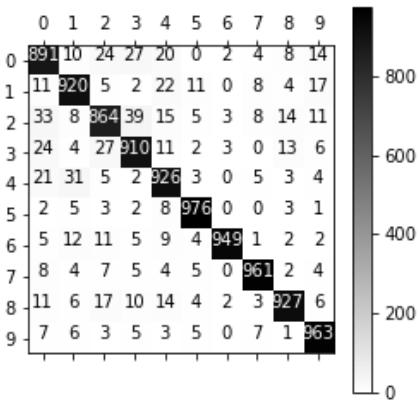
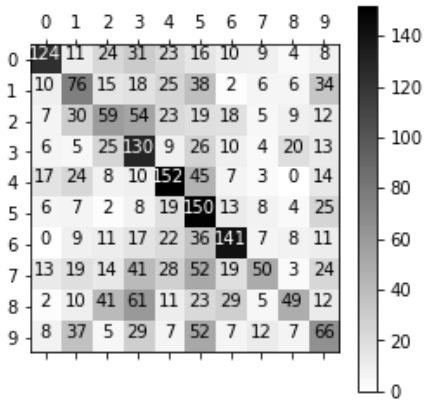
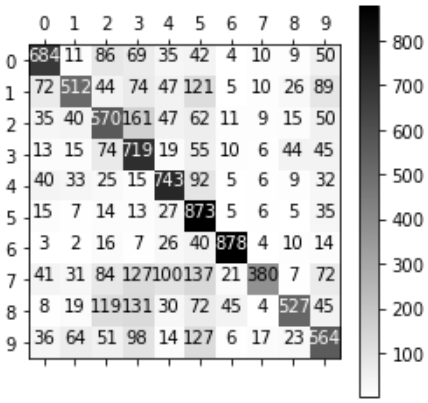
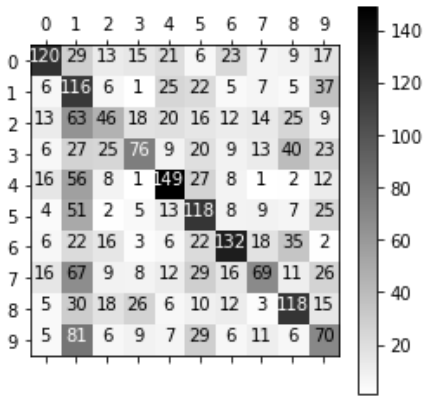
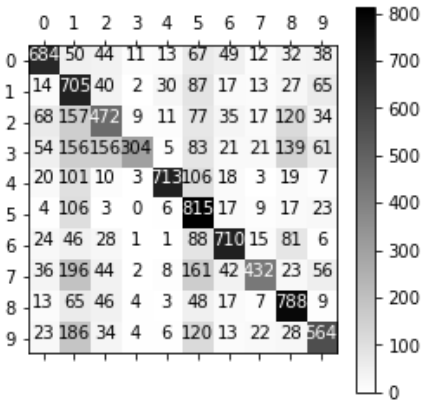
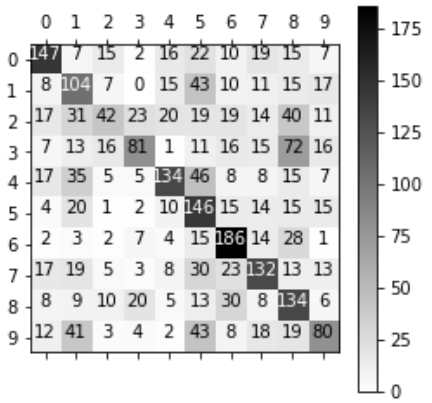
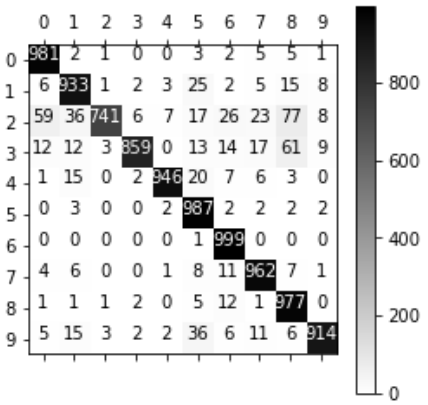
Methodology:

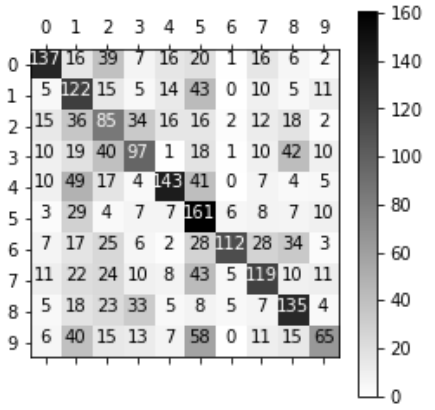
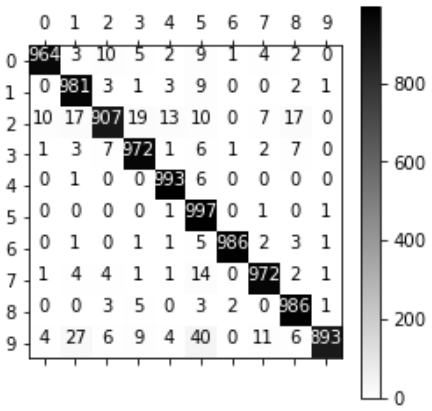
- For SVM training, the features are either dropped or padded to have the same dimension. For a spectrogram, the shape is taken to be (161,100) for each signal. For MFCC with delta features, the shape is taken to be (100,39) while for MFCC with only cepral coefficients, shape is taken to be (100,41).
- Initially, SVM models are trained on signals and results are noted.
- Random noise is added to each signal at 100 different random points and SVM is trained on features extracted for these noisy signals. This is done so as to make the model robust. Results are noted.

RESULTS:

Confusion Matrix$_{i,j}$ - Number of samples belonging to class i classified as class j

	Confusion Matrix (val)	Confusion Matrix (train)
Spectrogram (without noise)		

Spectrogram (with noise)		
MFCC + delta (without noise)		
MFCC + delta (with noise)		
MFCC with no delta (without noise)		

MFCC with no delta (with noise)		

$$Precision\ for\ class\ i = \frac{(Number\ of\ samples\ from\ i\ correctly\ classified)}{Number\ of\ total\ samples\ classified\ as\ i} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

	Precision (val)	Precision (train)
Spectrogram (without noise)	[0.4, 0.26, 0.24, 0.33, 0.53, 0.39, 0.4, 0.37, 0.44, 0.3]	[0.96, 0.94, 0.91, 0.97, 0.91, 0.98, 0.99, 0.96, 0.97, 0.97]
Spectrogram (with noise)	[0.29, 0.27, 0.29, 0.32, 0.3, 0.42, 0.44, 0.39, 0.4, 0.37]	[0.88, 0.91, 0.89, 0.9, 0.9, 0.96, 0.99, 0.96, 0.95, 0.94]
MFCC + delta (without noise)	[0.64, 0.33, 0.29, 0.33, 0.48, 0.33, 0.55, 0.46, 0.45, 0.3]	[0.72, 0.7, 0.53, 0.51, 0.68, 0.54, 0.89, 0.84, 0.78, 0.57]
MFCC + delta (with noise)	[0.61, 0.21, 0.31, 0.47, 0.56, 0.39, 0.57, 0.45, 0.46, 0.3]	[0.73, 0.4, 0.54, 0.89, 0.9, 0.49, 0.76, 0.78, 0.62, 0.65]
MFCC with no delta (without noise)	[0.62, 0.37, 0.4, 0.55, 0.62, 0.38, 0.57, 0.52, 0.37, 0.46]	[0.92, 0.91, 0.99, 0.98, 0.98, 0.89, 0.92, 0.93, 0.85, 0.97]
MFCC with no delta (with noise)	[0.66, 0.33, 0.3, 0.45, 0.65, 0.37, 0.85, 0.52, 0.49, 0.53]	[0.98, 0.95, 0.96, 0.96, 0.97, 0.91, 1.0, 0.97, 0.96, 0.99]

$$Recall\ for\ class\ i = \frac{(Number\ of\ samples\ from\ i\ correctly\ classified)}{Number\ of\ total\ samples\ of\ i} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

	Recall (val)	Recall (train)
Spectrogram (without noise)	[0.33, 0.27, 0.23, 0.24, 0.47, 0.42, 0.43, 0.46, 0.42, 0.38]	[0.9, 0.96, 0.93, 0.91, 0.96, 0.98, 0.98, 0.99, 0.96, 0.97]
Spectrogram (with noise)	[0.31, 0.2, 0.16, 0.36, 0.59, 0.4, 0.25, 0.3, 0.42, 0.41]	[0.89, 0.92, 0.86, 0.91, 0.93, 0.98, 0.95, 0.96, 0.93, 0.96]

MFCC + delta (without noise)	[0.48, 0.33, 0.25, 0.52, 0.54, 0.62, 0.54, 0.19, 0.2, 0.29]	[0.68, 0.51, 0.57, 0.72, 0.74, 0.87, 0.88, 0.38, 0.53, 0.56]
MFCC + delta (with noise)	[0.46, 0.5, 0.19, 0.31, 0.53, 0.49, 0.5, 0.26, 0.49, 0.3]	[0.68, 0.7, 0.47, 0.3, 0.71, 0.82, 0.71, 0.43, 0.79, 0.56]
MFCC with no delta (without noise)	[0.57, 0.45, 0.18, 0.33, 0.48, 0.6, 0.71, 0.5, 0.55, 0.35]	[0.98, 0.93, 0.74, 0.86, 0.95, 0.99, 1.0, 0.96, 0.98, 0.91]
MFCC with no delta (with noise)	[0.53, 0.53, 0.36, 0.39, 0.51, 0.67, 0.43, 0.45, 0.56, 0.28]	[0.96, 0.98, 0.91, 0.97, 0.99, 1.0, 0.99, 0.97, 0.99, 0.89]

$$F1\ score = \frac{2*Precision*Recall}{Precision + Recall}$$

	F1 score (val)	F1 score (train)
Spectrogram (without noise)	[0.36, 0.26, 0.23, 0.28, 0.5, 0.4, 0.41, 0.41, 0.43, 0.34]	[0.93, 0.95, 0.92, 0.94, 0.93, 0.98, 0.98, 0.97, 0.96, 0.97]
Spectrogram (with noise)	[0.3, 0.23, 0.21, 0.34, 0.4, 0.41, 0.32, 0.34, 0.41, 0.39]	[0.88, 0.91, 0.87, 0.9, 0.91, 0.97, 0.97, 0.96, 0.94, 0.95]
MFCC + delta (without noise)	[0.55, 0.33, 0.27, 0.4, 0.51, 0.43, 0.54, 0.27, 0.28, 0.29]	[0.7, 0.59, 0.55, 0.6, 0.71, 0.67, 0.88, 0.52, 0.63, 0.56]
MFCC + delta (with noise)	[0.52, 0.3, 0.24, 0.37, 0.54, 0.43, 0.53, 0.33, 0.47, 0.3]	[0.7, 0.51, 0.5, 0.45, 0.79, 0.61, 0.73, 0.55, 0.69, 0.6]
MFCC with no delta (without noise)	[0.59, 0.41, 0.25, 0.41, 0.54, 0.47, 0.63, 0.51, 0.44, 0.4]	[0.95, 0.92, 0.85, 0.92, 0.96, 0.94, 0.96, 0.94, 0.91, 0.94]
MFCC with no delta (with noise)	[0.59, 0.41, 0.33, 0.42, 0.57, 0.48, 0.57, 0.48, 0.52, 0.37]	[0.97, 0.96, 0.93, 0.96, 0.98, 0.95, 0.99, 0.97, 0.97, 0.94]

	Accuracy		Average F1 score		Average Precision		Average Recall	
	Val	Train	Val	Train	Val	Train	Val	Train
Spectrogram (without noise)	0.37	0.95	0.37	0.95	0.37	0.96	0.37	0.95
Spectrogram (with noise)	0.34	0.93	0.34	0.93	0.35	0.93	0.34	0.93
MFCC + delta (without noise)	0.4	0.64	0.39	0.64	0.42	0.68	0.4	0.64

MFCC + delta (with noise)	0.41	0.62	0.41	0.61	0.44	0.68	0.41	0.62
MFCC with no delta (without noise)	0.48	0.93	0.47	0.93	0.49	0.93	0.48	0.93
MFCC with no delta (with noise)	0.47	0.97	0.48	0.96	0.52	0.96	0.47	0.96

SVM takes much less time to train on MFCC features than spectrogram features because of the smaller dimension for MFCC yet for validation sets, MFCC features outperformed spectrogram features. Spectrogram features as well as MFCC features without delta features tend to overfit which is why very high accuracy is noticed for predictions on training data as compared to validation data.
MFCC is based on human hearing perception, has features which are more decorrelated as compared to spectrogram, concentrates only on certain frequency components, and thus is better than spectrogram.
MFCC + delta features contain more information about frame context hence validation accuracy to training accuracy ratio is better as compared to MFCC with no delta features (same holds for precision, recall and F1 score).
Introducing noise tends to only slightly increase/reduce accuracy since the validation set does not have any noise but it also tends to reduce overfitting of models. An increase in validation accuracy/training accuracy is noticed for MFCC + delta + delta delta features indicating their superiority over other 2 methods (same holds for precision, recall and F1 score).
For the spectrogram features with noise introduced, a lot of samples are getting misclassified as 4, which is why it has high recall and low precision. Class 4 has very high recall as compared to other classes which implies a lot of signals are getting misclassified. For the MFCC + delta + delta delta features with noise introduced, a lot of samples are getting misclassified as 1, which is why it has high recall and low precision. As compared to spectrogram (only 1 class has recall > 0.45), a lot of classes have high recall here (6 classes have recall > 0.45), indicating better classification capabilities. F1 score for most of the classes for MFCC is higher as compared to that of spectrogram.

Files Attached:

Codes:

question1_1.py - Extract, plot and dump spectrogram features
question2_39.py - Extract and dump MFCC coefficients (13 per frame) + delta and delta delta features
question2_41.py - Extract and dump MFCC coefficients (41 per frame) for each signal
question3_1.py - Training SVM and calculating precision, recall, F1 score, accuracy and confusion matrix.

models:

spec_nonoise.pickle - SVM trained on non noisy signals' spectrogram features
spec_noise.pickle - SVM trained on noisy signals' spectrogram features
mfcc_39_nonoise.pickle - SVM trained on non noisy signals' MFCC + delta + delta delta features
mfcc_39_noise.pickle - SVM trained on noisy signals' MFCC + delta + delta delta features
mfcc_41_nonoise.pickle - SVM trained on non noisy signals' MFCC features
mfcc_41_noise.pickle - SVM trained on noisy signals' MFCC features

References:

1. Hanning window: https://www.tek.com/blog/window-functions-spectrum-analyzers
2. MFCC: https://arxiv.org/abs/1003.4083 and http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/
3. Spectrogram: https://shodhganga.inflibnet.ac.in/bitstream/10603/7503/8/08_chapter%205.pdf

4. FFT, DFT: https://towardsdatascience.com/fast-fourier-transform-937926e591cb