Case Study -2

Gender classification based on voice features using Support Vector Machine with different Kernels

by

Pradeep Sathyamurthy

Under the guidance of: Prof J. Gemmell

DePaul University

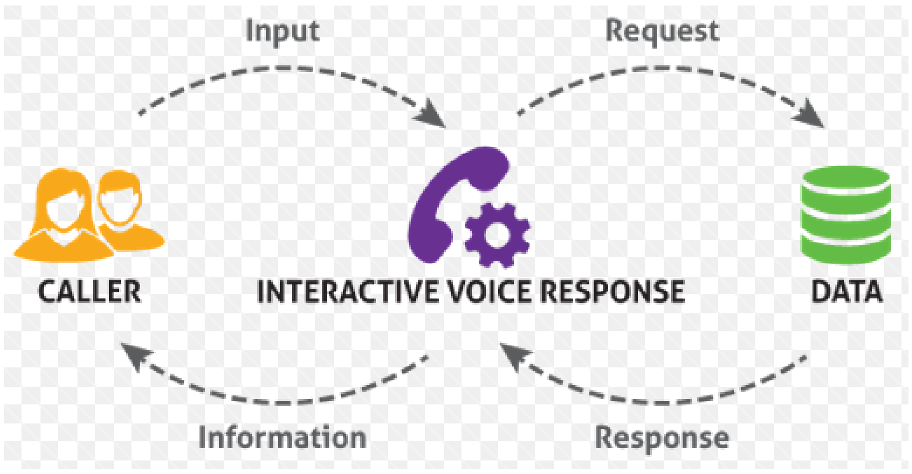
22nd October 2017

1. Abstracts:

In this case study, I will deal with a dataset which has various voice parameters using which we need to identify if that voice belongs to a male or female, basically do a gender classification. I thought this would be a good dataset to experiment with Support Vector Machine (SVM) and hence tried to clean and process data to experiment it with various kernel tricks using SVM, then evaluate the same with 10-fold cross validation and fine tune various kernel parameters to obtain a right boundary in a hyper plane which will classify any given instance as Male or Fe-male.

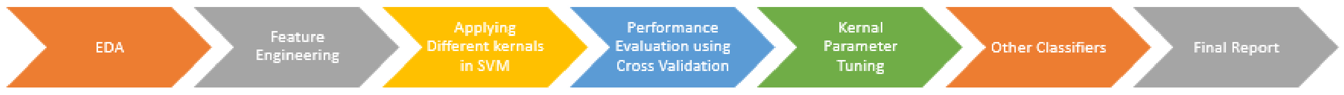
2. Introduction:

It was a month back I made my girlfriend to call a banks call centre on behalf of me as I knew it will take me hours together to reach a bank representative, I thought I was smart. However, the interactive voice response system used by bank was smarter than me when my girlfriend provided my account number. It kind of identified my girlfriends voice as a female voice and it kind of identified my gender as male and politely discontinued the call. I was taken a back with such kind of automation happening in a IVR system.



When I tried to research about its functionality, I could find that they use machine learning technology to identify a person's identity as a part of fraud detection mechanism and classifying gender based on voice was one such part of fraud detection activity. I was curious with this finding and hence wanted to find a similar dataset to classify a gender based on voice parameters. This case study deals with such kind of dataset taken from below URL and apply Support Vector Machine to classify the dataset's target class.

3. Process followed in this Case Study:


EDA
Feature Engineering
Applying Different kernals in SVM
Performance Evaluation using Cross Validation
Kernal Parameter Tuning
Other Classifiers
Final Report

In this case study, I am trying to first use the Support Vector Machine to project the dataset in a high dimensional space and find a right hyperplane which will separate the two class of Gender accurately, with this I will know on which kernel trick my dataset behaves well, if my dataset behaves well with a linear hyper plane, then I will make use of other linear classifiers like Naïve Bayes, LDA to see if I can further increase its accuracy. Else, I will try by ensembling other non-linear classifiers like Decision Tree, KNN, etc., to improve my accuracy in classifying the gender.

4. Dataset Description:

Original dataset used in this case study can be downloaded by clicking here. WAV (Waveform Audio File) is an audio file format used to store an audio bitstreams in PC. Voice of 3168 different users where captured and then pre-processed for acoustic analysis. Output obtained from this acoustic analysis was stored in a CSV file format for further analysis. Thus, CSV file obtained as 3168 instances with 21 features in which 20 are independent variable and 1 is a target variable which is a categorical variable with 2 class Male and Fe-male This is the CSV file which will be subjected for machine learning algorithm called SVM to classify if an instance is a male or female.

4.1. Independent Variables:

All independent variables are continuous in nature with floating-point values. There are totally 20 independent variables as sighted below, each of which is an acoustic property of a voice sample:

SI.NO	Attribute	Data Type	Description wrt Acoustic properties
1	meanfreq	float64	**meanfreq**: mean frequency (in kHz)
2	sd	float64	**sd**: standard deviation of frequency
3	median	float64	**median**: median frequency (in kHz)
4	Q25	float64	**Q25**: first quantile (in kHz)
5	Q75	float64	**Q75**: third quantile (in kHz)
6	IQR	float64	**IQR**: interquantile range (in kHz)
7	skew	float64	**skew**: skewness (see note in specprop description)
8	kurt	float64	**kurt**: kurtosis (see note in specprop description)
9	sp.ent	float64	**sp.ent**: spectral entropy
10	sfm	float64	**sfm**: spectral flatness
11	mode	float64	**mode**: mode frequency
12	centroid	float64	**centroid**: frequency centroid (see specprop)
13	meanfun	float64	**meanfun**: average of fundamental frequency measured across acoustic signal
14	minfun	float64	**minfun**: minimum fundamental frequency measured across acoustic signal
15	maxfun	float64	**maxfun**: maximum fundamental frequency measured across acoustic signal
16	meandom	float64	**meandom**: average of dominant frequency measured across acoustic signal
17	mindom	float64	**mindom**: minimum of dominant frequency measured across acoustic signal
18	maxdom	float64	**maxdom**: maximum of dominant frequency measured across acoustic signal
19	dfrange	float64	**dfrange**: range of dominant frequency measured across acoustic signal
20	modindx	float64	**modindx**: modulation index. Calculated as the accumulated absolute difference between

4.2. Dependent Variable:

Dataset has one dependent variable called label of binary type with 2 classes ‘Male’ and ‘Female’. This variable is the target variable and our aim is to classify each of the instance correctly as Male or Female.

SI.NO	Attribute	Data Type	Description wrt Acoustic properties
21	label	String	**label**: gender to which the acoustic parameter belongs to

4.3. Missing Data:

Raw dataset as such did not have any missing values in it and hence there was no need for filling with any dummy values while reading the dataset.

5. Setting Benchmark Accuracy for classifiers using Raw Dataset and Naïve Bayes

Generally, it is advised to check the validity of the data set by subjecting the raw data as such into a Naïve Bayes classifier, which is a Naïve implementation of building a classifier model using the probability calculation done on each feature. Accuracy obtained from this Naïve Bayes classifier can be set as the benchmark accuracy to validate how a dataset will behave when subject to different classifiers as part of model building.

Our ***raw dataset when subject to Naïve Bayes model as such produced an accuracy of 0.8567 with 10-fold cross validation*** which is not bad. This signifies that 15 out of 100 times our model will misclassify the data with a basic Naïve Bayes model. However, for a business it is important to see improvement in this accuracy using complex classifier models and by fine tuning its parameter setting.

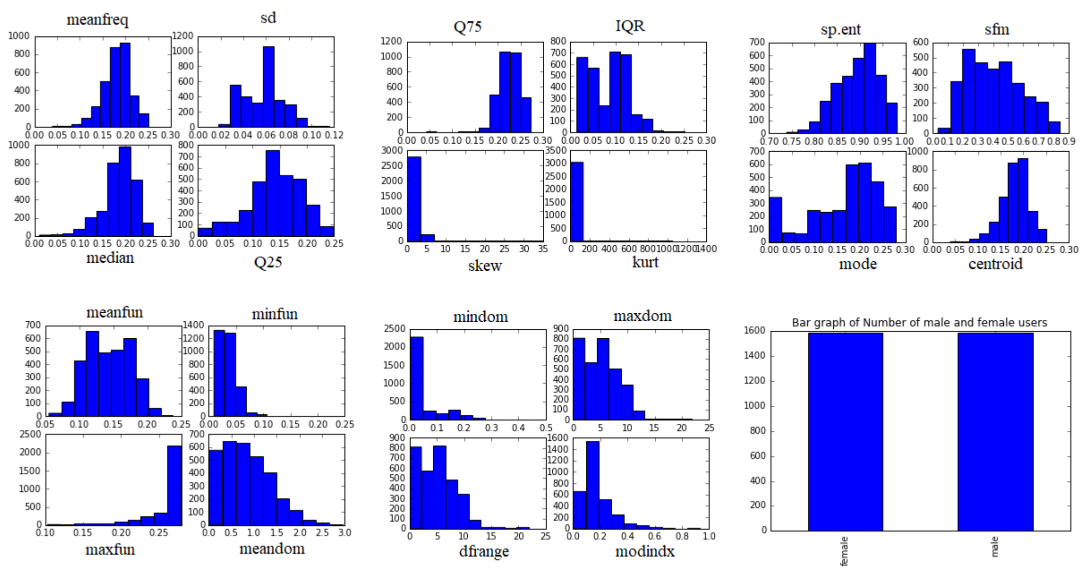
End of this analysis, we can set the benchmark for accuracy for this dataset to be 0.8567. That is, any treatment that we do to data as part of cleaning or munging and any classifiers that are built on this dataset should at least yield an accuracy greater than 0.85. If any data treatment or any classifiers built on this dataset produce an accuracy of 0.8567, we can politely ignore them without taking them into consideration further unless we think they might constitute for any ensemble learnings.

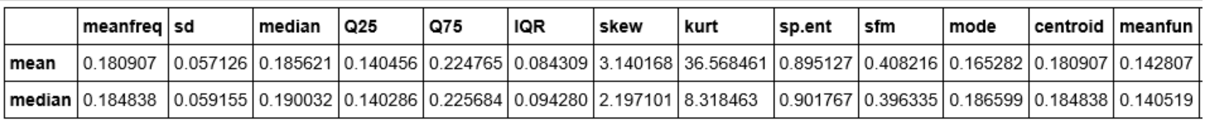
0.8567

- This is the benchmark accuracy for this dataset
- Any data treatement method which might reduce this accuracy will be discarded.
- Any model which produce accuracy less than this will be discarded from further analysis.

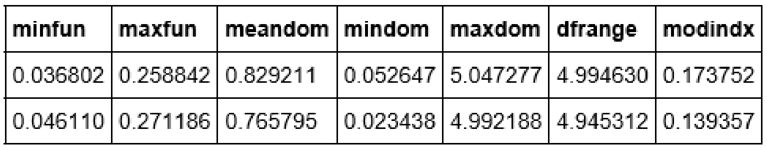
6. Exploratory Data Analysis (EDA):

Dataset containing acoustic parameters was subjected to EDA process to infer the validity of the dataset before I build any models over it. When I checked for any null values in dataset, I found this dataset did not have any missing values. Using the histogram, I studied the distribution of this data to hypothesize the impact of each variables in classifying a gender and to understand the attribute more clearly to subject it for modelling:





	meanfreq	sd	median	Q25	Q75	IQR	skew	kurt	sp.ent	sfm	mode	centroid	meanfun
mean	0.180907	0.057126	0.185621	0.140456	0.224765	0.084309	3.140168	36.568461	0.895127	0.408216	0.165282	0.180907	0.142807
median	0.184838	0.059155	0.190032	0.140286	0.225684	0.094280	2.197101	8.318463	0.901767	0.396335	0.186599	0.184838	0.140519



minfun	maxfun	meandom	mindom	maxdom	dfrange	modindx
0.036802	0.258842	0.829211	0.052647	5.047277	4.994630	0.173752
0.046110	0.271186	0.765795	0.023438	4.992188	4.945312	0.139357

6.1. Quick Findings:

- Irrespectve to viz of histogram, we can infer those attributes with mean and median values almost equal have gaussian distribution. I see totally 9 variables are normally distributed and 11 are skewed.
- Thus, variables meanfreq, sd, median, Q25, Q75, sp.ent, sfm, centroid, meanfun are Normally distributed
- Variables skew, kurt, ***minfun***, maxfun, meandom, mindom, maxdom, dfrange, midindex, IQR, mode are skewed
- Target variables are symmetrical in nature with equal count of 1584 records for both Male and Female and hence I don't need to do balancing with any sampling techniques.

6.2. Domain Knowledge

- ***Intuitively, we are aware that any voice related measurement depends on one of the frequency parameter***, thus to validate dataset, we will consider one important parameters mean fundamental frequency to see how its data are distributed across based on it.
- ***Meanfun***, this variable denotes the mean fundamental frequency in kHz which is nothing but a voice frequency or voice band of a human being. When, I did some domain level research I could obtain a fact from Wikipedia[1] that a ***typical adult male will have a fundamental frequency ranging from 85 to 180Hz and for female it is 165 to 255Hz***.
- In our dataset meanfun is captured as KHz and thus valid range for a given instance must be in the range between 0.085KHz and 0.255Khz. However, when we see the above histogram we observe that there are few instances with meanfun even greater than 0.255KHz, which for now we can consider as an outlier for this case study based on domain knowledge, as these data would have been recorded with noise.
- Since most of the classification algorithms are not robust to noise, we will try to treat the dataset such a way that we have data instanced with meanfun only ranging between 0.085 and 0.255KHz as part of data munging.
- Similarly, we can filter values based on meanfun whose values less than 0.085 and greater than 0.18 for male and values less than 0.165 and greater than 0.255 for female and consider them as outliers and remove them.

meanfun

- It is the mean fundamental frequency of the human being
- One of the very important parameter in human study about voice and WAV files.
- valid range for adult male is 0.085 to 0.18 KHz
- valid range for adult female is 0.165 to 0.255 KHz

7. Data Munging and Partition

In this session, I will discuss about data cleaning, data reduction and data normalization done on my voice dataset. I will tabulate the different datasets that I will take forward to build my models.

7.1. Data Cleaning:

As discussed in above section, based on domain knowledge we can infer that average fundamental frequency for adult male fall between 0.085 and 0.18 and for adult female it falls between 0.165 and 0.255. Based on this condition, I filtered the index from raw dataset with below condition:

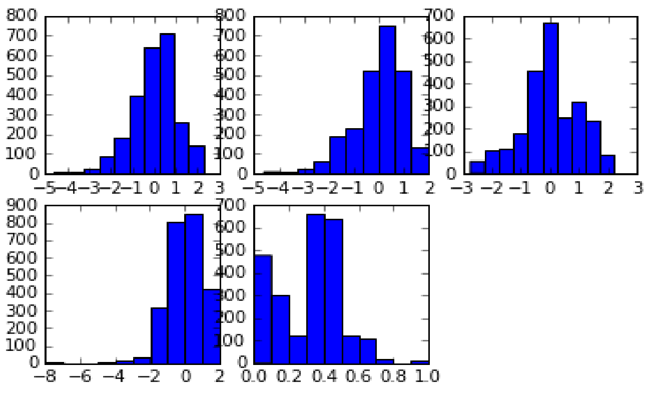
filtering index

- meanfun values less than 0.085 or greater than 0.18 and label=male
- meanfun values less than 0.165 or greater than 0.255 and label=female

Index obtained from above filtering was used to remove the data from the raw dataset. And this behaves as a treated dataset for model building.

7.2. Creating the partially Normalized Dataset

- In this dataset meanfreq, meanfun, median, Q25, Q75, IQR are the only variables associated with unit kHz
- Let us normalize these variables to make them unit free
- I will apply the z-score normalization for meanfreq, median, Q25, Q75 as these data are normally distributed
- I will apply min-max normalization for IQR, meanfun as these data have a skewed behaviour
- Data set with such partially normalized dataset are kept as partially normalized dataset



7.3. Feature reduction based on VIF factor

- I built a correlation matrix and calculated the variance inflation factor on the partially normalized dataset.
- Based on VIF, we will drop the columns 'kurt', 'centroid', 'dfrange', 'z_meanfreq' from partially normalized dataset.
- This way, I have reduced the feature dimension which will help in building some optimized classifiers or decision boundaries for SVM models.

	Column Name	Correlated with
0	skew	kurt
1	kurt	skew
2	centroid	[z_meanfreq, z_median, z_Q25]
3	maxdom	[dfrange]
4	dfrange	[maxdom]
5	z_meanfreq	[centroid, z_median, z_Q25]
6	z_median	[centroid, z_meanfreq]
7	z_Q25	[centroid, z_meanfreq]

7.4. Creating completely normalized dataset

I created a new dataset by normalizing all columns in the raw dataset. Standardscalar object from sklearn was used to perform this. By doing this all features are in common scale and are independent of units. This kind of normalization are very helpful when there are some ambiguity in the units of dataset. Since, our dataset consists of voice parameters, there is a level of ambiguity in understanding the units of all attributes and hence it is same to normalize all columns in the dataset, so that they can be used to build an optimized classifier model.

7.5. Data Partition and Tabulating the dataset names that will be subjected for model building

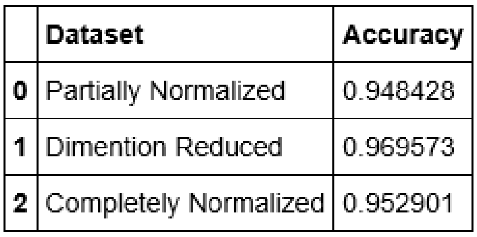
Data partition is an activity where I did a 80-20 split on our dataset to get a training and a test dataset. Training dataset are used to build and tune my model and test dataset are used to evaluate the model. Since dataset size is small, I do not have a validation dataset separately, instead I will perform 10-fold cross validation on my model to ensure its performance. Below is the table that has dataset details that are initially used to build a optimized classification model:

SI NO	Core Dataset	Description	Split Datasets	Shape(Train)	shape(Test)
1	Data_x, data_y	Raw dataset ***treated with noise*** removals based on cleaning explained above.	data_x_train, data_x_test, data_y_train, data_y_test	(1966, 20)	(492,20)
2	Data_x2	Treated, Partially Normalized and ***dimension reduced*** dataset.	data_x2_train, data_x2_test, data_y2_train, data_y2_test	(1966, 16)	(492, 16)
3	Data_x3	Treated and ***Completely normalized*** dataset	data_x3_train, data_x3_test, data_y3_train, data_y3_test	(1966, 20)	(492, 20)

8. Validating the data cleaning activity with a bench mark accuracy obtained

As I mentioned above, any cleaning activity done on the dataset should at least produce an accuracy greater than 0.85 and not less than 0.85. So, to validate the steps that I took for cleaning the data, I have again build a Naïve Bayes classifier on all above-mentioned dataset and below its results:

- Naive Bayes classifier after data treatment produce an average accuracy of 0.95
- I see a significant increase in accuracy from 0.85671 to 0.952 after the data was cleaned.
- Data with dimension reduced and data which are completely normalized works better than raw treated dataset.
- However, this can be considered as a base classifier at this point and result shown makes sure that our data clean up logic holds good and we haven't removed any influential data's from dataset.
- This accuracy will be set as a new benchmark for any complex classifier that will be built further.
- Thus, ***accuracy of 0.95 can be set as a new bench mark accuracy value for this dataset which is cleaned and processed.***
- Any model which produce accuracy less than 0.95 can be considered as a non-efficient model for this dataset from now on.



	Dataset	Accuracy
0	Partially Normalized	0.948428
1	Dimention Reduced	0.969573
2	Completely Normalized	0.952901

9. Model Building – Applying different Kernels in SVM

There are different classification algorithms which I can use to solve a given problem, my research question here is to classify the gender as ‘male’ or ‘female’ based on voice parameters recorded in a WAV file format. Since we are clear about our target variable and what an outcome needs to be we can treat this as a ***supervised learning classification problem.***

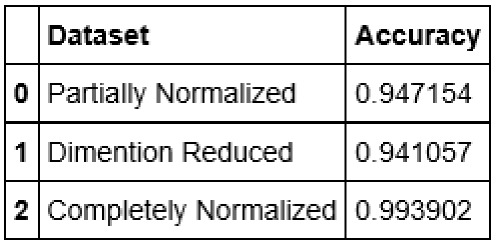
Our focus of this case study is to build a classifier model based on SVM classifier algorithm. SVM stands for ***Support Vector Machine***. Idea behind this algorithm is to project our data in a high dimensional feature space and based on support vector find a hyperplane which will discriminate different classes in the target variable. In our case it discriminates the objects as male or female.

I have totally 20 independent variables in my treated dataset and thus when subjected to SVM, my objects are projected in a 20-dimensional space. We will then ***find a hyperplane, kind of a hard margin*** which will discriminate the male and female objects accurately in this high dimensional space. Besides, ***our main focus is to find an optimistic hyper plane which will have maximum margin (soft margin).***

In a one-dimensional space, margin can be a point that can discriminate the class variable, in a two-dimensional space our margin is a line, in a 3-dimensional space margin is a plane and with more than 3 dimension it is a hyper plane which are obtained with the help of support vectors. Since, SVM focus on only support vectors to form a maximum margin, ***SVM is very robust for outliers*** and hence, as part of data manipulation we only focused on influential points and removed them and did not work much on outlier analysis.

9.1. Linear SVM Classifier

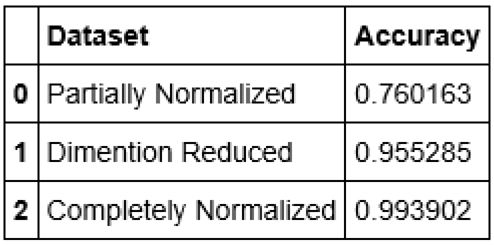
- Here support vectors are used to build a linear hyper plane which helps in classifying the target variable.
- I subjected 3 different datasets as explained above to a linear SVM model and I can observe that ***dataset which is completely normalize is performing well***.
- As part of this kernel trick, we have our hyperplane to be linear in a 20-dimentional space
- This ***model exhibits a classification accuracy of 0.993902***
- Since the data is 20-dimentional, we cannot visualize if the data have a linear or curved relation in feature space, we can take a domain level expertise here.
- However, ***since we have no domain expertise for individual analysis purpose we will try to build a model with other kernel tricks types too*** and see how the model behaves in classifying the gender.



	Dataset	Accuracy
0	Partially Normalized	0.947154
1	Dimention Reduced	0.941057
2	Completely Normalized	0.993902

9.2. Gaussian Kernel SVM

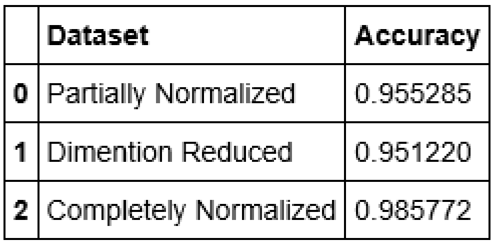
- RBF or Gaussian is the default kernal which SVM uses in sklearn
- Performance of RBF kernel trick is also same as linear kernel SVM for our dataset, I obtained an accuracy of 0.993902 for RBF Kernal using SVM for normalized dataset
- With this we can infer that our voice dataset is both linearly and gaussian separable



	Dataset	Accuracy
0	Partially Normalized	0.760163
1	Dimention Reduced	0.955285
2	Completely Normalized	0.993902

9.3. Polynomial Kernel SVM

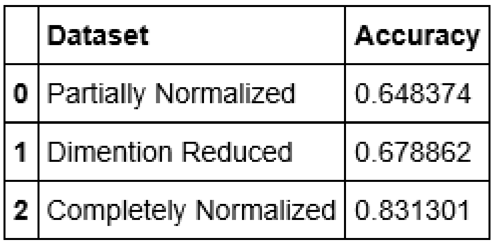
- To achieve much more high accuracy, i tried using polynomial kernel too
- I obtained an accuracy of 0.985 for polynomial kernel on normalized dataset
- This is comparatively much less than the linear and rbf kernels
- However, we cannot conclude this result at this stage as, our training dataset is just one single sample on which we obtained this result



	Dataset	Accuracy
0	Partially Normalized	0.955285
1	Dimention Reduced	0.951220
2	Completely Normalized	0.985772

9.4. Sigmoidal Kernel SVM

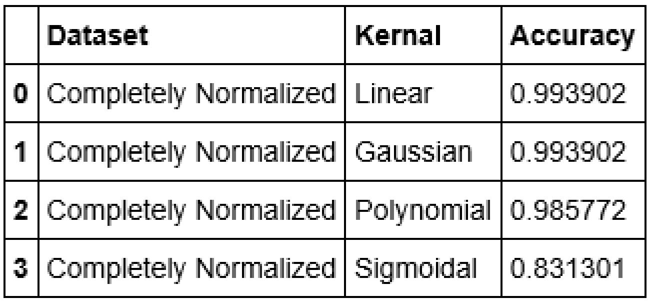
- When a dataset is behaving well linearly, it is explicitly known that it doesn't work well in a sigmoidal space
- Above result obtained is the evident for this
- I obtained accuracy of just 0.831 with sigmoidal kernal



	Dataset	Accuracy
0	Partially Normalized	0.648374
1	Dimention Reduced	0.678862
2	Completely Normalized	0.831301

9.5 Consolidated Model Accuracy with 80-20 split data validation

From above results, we see completely normalized dataset outperforms well for all kernel tricks. Below is the consolidated result obtained from 80-20 data split with various kernel tricks:



	Dataset	Kernal	Accuracy
0	Completely Normalized	Linear	0.993902
1	Completely Normalized	Gaussian	0.993902
2	Completely Normalized	Polynomial	0.985772
3	Completely Normalized	Sigmoidal	0.831301

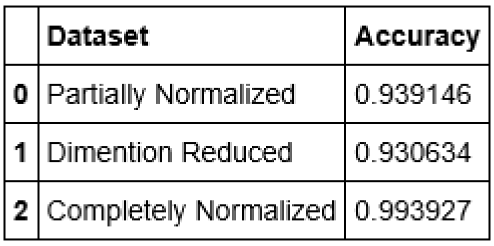
- From above table, it is ***very evident that the completely normalized dataset behaves well compare to un-normalized dataset***
- I obtain a maximum accuracy due to the data treatment done, that is treating the meanfun attribute based on anatomical facts of human being.
- Maximum accuracy ***I could achieve is 0.9939 which are from Linear and Gaussian Kernels using SVM***
- While the polynomial and Sigmoidal kernel doesn't seem to classify the target variable accurately and giving a low accuracy of 0.95 and 0.83 for Polynomial and Sigmoidal kernel respectively.
- ***However, I cannot blindly accept this accuracy result because this is derived from one sample of training set and validated with a sample test set***.
- To evaluate this model to be more robust and ***to ensure data doesn't overfit, I wanted to subject these model and dataset to a 10-fold cross validation*** and observe its result as part of next session

10. Performance evaluation of different Kernels using 10-fold Cross Validation:

Model evaluation is an important step in model building activity as at times with 80-20 split we might get a very high performance in terms of accuracy for a classifier model. However, it might have occurred for that sample split and the model is not much generalized to classify new instances accurately. Such kind of overfitting problem are quite common in model building activity. To avoid such scenarios, we perform a 10-fold cross validation on the all kernels and for all datasets considered above. Mean value of the accuracy from 10 different models gives model final accuracy here which is much robust to overfitting scenarios. Since polynomial kernel took more time, its results are masked below:

10.1. Linear Kernel

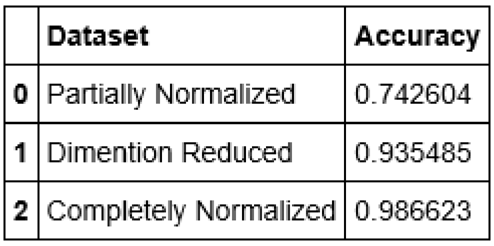
- I see even with 10-fold cross validation, our ***linear kernel SVM is providing a high accuracy of 0.9939***
- Thus, I can consider linear Kernel SVM as one of the serious model to subject for further tuning and see if it increases the accuracy
- From table shown, it is still evident that the completely normalized dataset behaves well comparatively



	Dataset	Accuracy
0	Partially Normalized	0.939146
1	Dimention Reduced	0.930634
2	Completely Normalized	0.993927

10.2. Gaussian Kernel

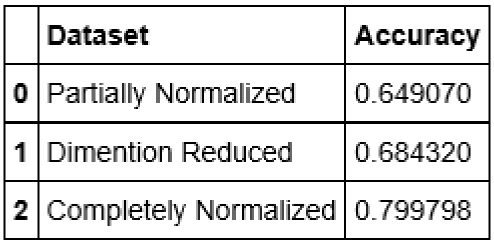
- From table, I see a slight decrease in accuracy when I subject Gaussian kernel to 10-fold cross validation
- Without 80-20 split test set we saw an accuracy of 0.9939 however, with 10-fold CV we obtain accuracy of 0.986
- So far, we see linear kernel is behaving well consistently and there is a slight decrease in performance with gaussian kernel



	Dataset	Accuracy
0	Partially Normalized	0.742604
1	Dimention Reduced	0.935485
2	Completely Normalized	0.986623

10.3. Sigmoidal Kernel

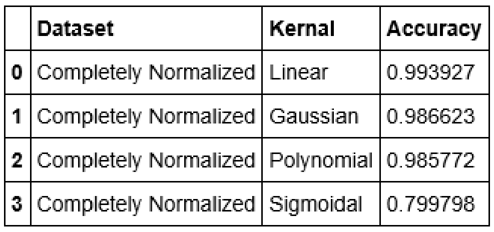
- Like Gaussian kernel, even polynomial and sigmoidal kernels yield less accuracy with 10-fold CV
- I did not include the results of polynomial kernel subjected to 10-fold CV because it was consuming more time to compute
- However, ***results of sigmoidal kernel are shown above and we see accuracy is dropped from 0.81 to 0.79***



	Dataset	Accuracy
0	Partially Normalized	0.649070
1	Dimention Reduced	0.684320
2	Completely Normalized	0.799798

10.4. Consolidated Kernel Evaluation Results

- From table shown, it is clearly evident that ***Linear SVM Kernel on a completely normalized dataset behaves well***
- Even with 10-fold cross validation, ***I obtained an accuracy of 0.9933927 which seems consistent*** when compare to other kernels.
- After linear kernel, it is the Gaussian and Polynomial kernel which gives high accuracy
- ***So as part of next session, we will drop Sigmoidal kernel from our further analysis as it doesn't even satisfy the bench mark accuracy.***
- I will take up other 3 SVM models (linear, poly, sigmoidal) for performance tuning and see how the accuracy changes when we trade-off between kernel parameters like penalty (C) and gamma to obtain a soft margin.



	Dataset	Kernal	Accuracy
0	Completely Normalized	Linear	0.993927
1	Completely Normalized	Gaussian	0.986623
2	Completely Normalized	Polynomial	0.985772
3	Completely Normalized	Sigmoidal	0.799798

11. Parameter Tuning on different Kernels

From above results, I am clear that it is the completely normalized data with Linear, Gaussian and Polynomial kernel perform well with an average model accuracy of 0.98. As part of parameter tuning, we will relax the hard margins and try to concentrate on soft margins which will help to a model which are more generalized and with improved accuracy power.

11.1. Tuning Linear Kernel SVM

- Our aim in building an optimized kernel is to find an optimum hyper plane in feature space which has maximum margin in classifying our target variable.

- Kernel which I have built above so far to check the performance are those with hard margins, this is not good to be generalized as it may cause overfitting.

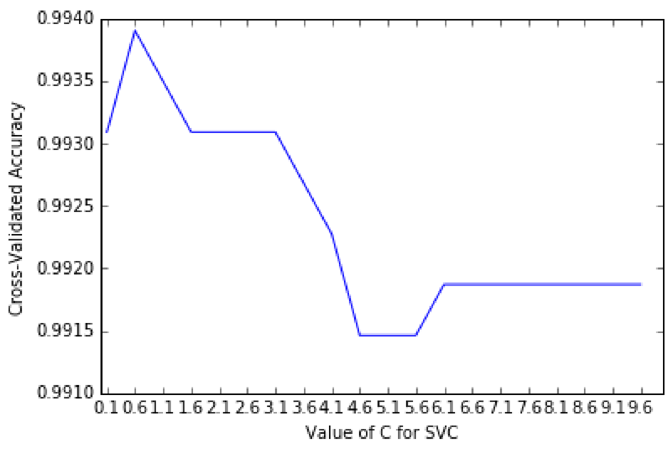
- So, in this session, we will trade off between margin and Support vectors to choose an optimum boundary which will not overfit the model and at the same time deliver a high accuracy in classifying the target variable.

- With ***linear kernel, it is the penalty measure through which we can do some trade off***

- Above table shows the accuracy (model performance) for different values of C

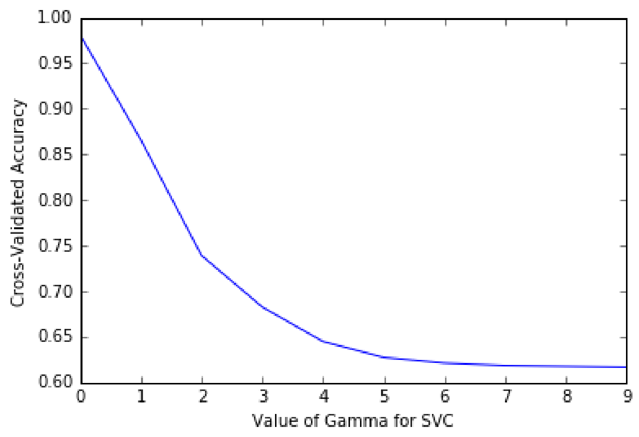
- ***Both from graph and table we see 0.6 and 1.1 to be the optimum penalty measure or C*** value which we can trade-off with in classifying the target variable.

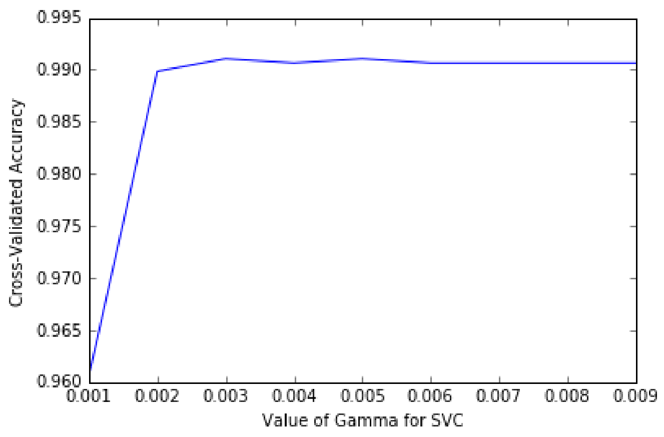
- Even with such trade off, ***we obtain almost 0.9939 accuracy for linear kernel***



	Penality Parameter C	Accuracy
0	0.1	0.993089
1	0.6	0.993902
2	1.1	0.993496
3	1.6	0.993089
4	2.1	0.993089
5	2.6	0.993089
6	3.1	0.993089
7	3.6	0.992683
8	4.1	0.992276
9	4.6	0.991463
10	5.1	0.991463

11.2. Tuning RBF Kernel SVM



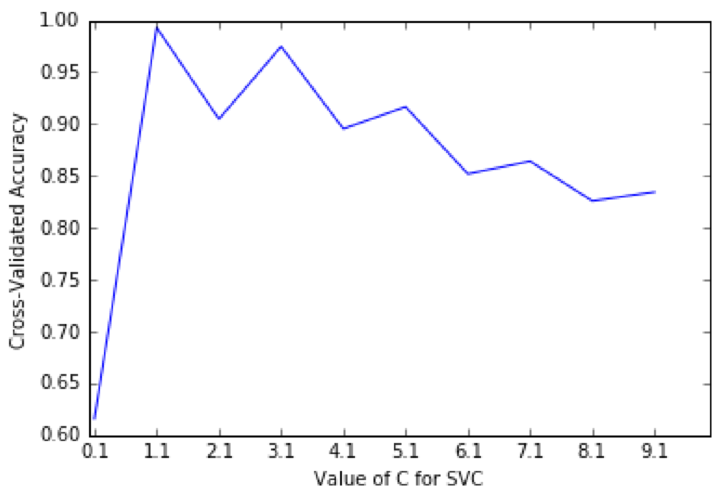


	Parameter Gamma	Accuracy
0	0.1	0.981289
1	1.1	0.866114
2	2.1	0.739190
3	3.1	0.682660
4	4.1	0.644832
5	5.1	0.627340
6	6.1	0.621239
7	7.1	0.618392
8	8.1	0.617579
9	9.1	0.616765

	Parameter Gamma	Accuracy
0	0.001	0.960562
1	0.002	0.989837
2	0.003	0.991057
3	0.004	0.990650
4	0.005	0.991057
5	0.006	0.990650
6	0.007	0.990650
7	0.008	0.990650
8	0.009	0.990650

- In Gaussian kernel, trade-off is done with penalty (C) along with gamma parameter
- I first experimented with wider Gamma values ranging between 1 and 10 and observed Kernel started to behave bad with gamma greater than 1
- So, I tried to find the most optimum value with in 0 and 1 and as show in above table, i obtained a maximum accuracy of 0.991 when gamma was equal to 0.03 and 0.05
- However, when compare to Linear kernel, we see rbf produce an accuracy of 0.002 times less.
- Thus, it is quite evident again that linear kernel acts well on this dataset in classification of target variable

11.3. Tuning Polynomial Kernel SVM



	Parameter Degree	Accuracy
0	0.1	0.615948
1	1.1	0.993089
2	2.1	0.904793
3	3.1	0.974783
4	4.1	0.895480
5	5.1	0.916632
6	6.1	0.851958
7	7.1	0.864161
8	8.1	0.825925
9	9.1	0.834472

- Along with penalty and gamma parameter, with polynomial kernel we can trade off with degree
- I experimented with various degree as shown above and obtained degree = 1.1 produce a high accuracy
- Accuracy obtained by polynomial is almost same as Linear which is 0.993
- So, to produce a final inference in choosing the best kernel we will apply a grid search in our next session and see which model and which parameter produce a high accuracy.

12. Grid Search Algorithm to obtain optimized parameter settings

Grid search is an algorithm used in sklearn package of python which is used to experiment with various parameter of a model in a structured way. As part of this grid search I impute different range of values for penalty parameters like C value, gamma and degree for linear, RBF and Polynomial kernels respectively. Below is the result obtained through grid search algorithm:

kernel	C	gamma	degree
poly	1.6	0.005	1

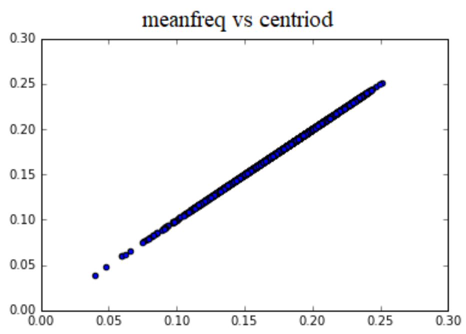
- From above result, I see ***it is the polynomial kernel with penalty measure of C=1.6 and gamma = 0.005 and with degree=1 produce a high accuracy of 0.9939*** in classifying the target variable.
- In this next session, I have tried to visualize my margin and kernel behaviour by subjecting only 2 columns for analysis as it becomes a 2-dimentional space for visualization.

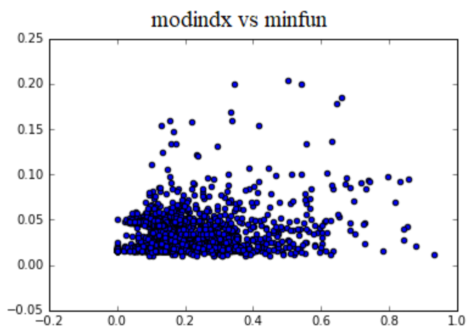
13. Visualization of kernal Margin and boundries represented in a 2D space

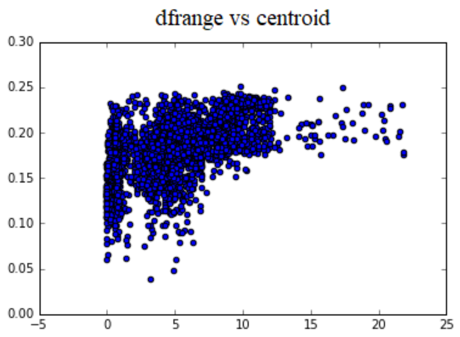
After doing necessary data clean-up and model building I could infer that a polynomial kernel SVM with parameters C=1.6, gamma=0.005 and degree=1 plots a perfect margin in a high dimensional space to classify gender label which is our target variable.

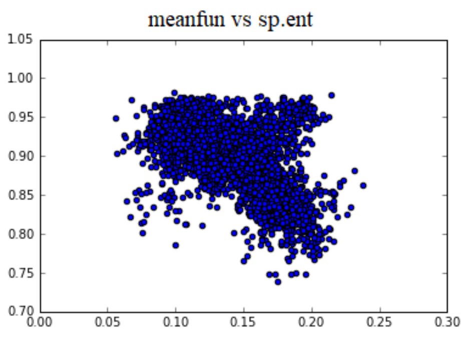
However, ***visualizing more than two dimensions is complex to represent***. ***So, I would like to choose any 2 variables from dataset through which I can represent my margin and kernel boundaries in a 2-dimentional space.***

13.1. Choosing the best attribute to represent dataset in 2D space







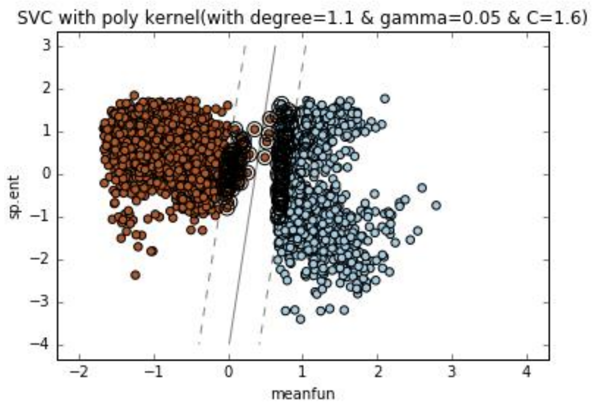


To choose the 2-important variable thorough which I can build my 2-dimensional feature space, I used the correlation matrix and above scatter plot obtained above and choose two variable which is moderately correlated. As ***neither the strong nor the weak correlation variables might not be well represented to show the decision boundaries.***

Meanfun being the most important variable for the dataset, I ***decided to choose it and match it with another variable which has moderate correlation*** with it. With ***0.52 as correlation value between i choose sp.ent and meanfun to be my choise of 2-dimentional feature space***.

13.2. Visualizing the model margin

Based on the variable chosen in the above section, that is meanfun and sp.ent, we will build a 2-dimensional space or a scatter plot matrix. We will use SVM model constructed above to represent these two variables in a different projected 2-dimnesion and represent the margin as line with necessary boundaries:

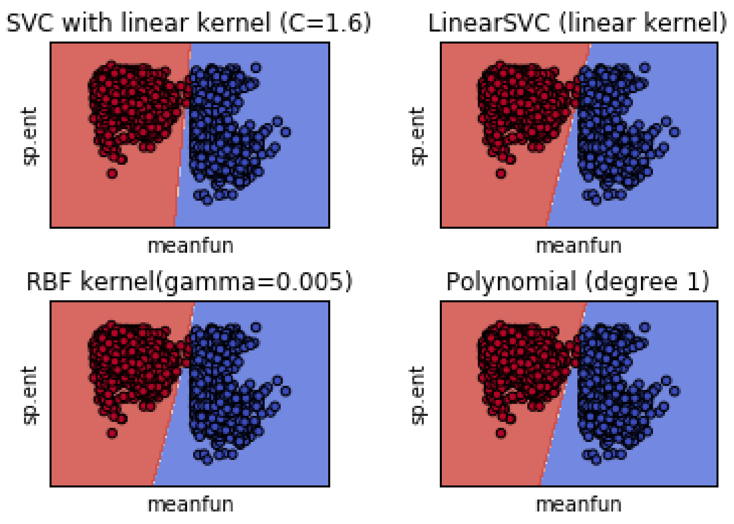


- I ***modelled polynomial kernel with penalty measure of C=1.6, gamma = 0.05 and degree=1*** to obtain the above scatter plot.
- When did, SVM projected my data in a 2-dimensional space and obtained an optimal margin that classifies my gender being male and female.
- From the above figure, we can infer:
  1. *Orage points = Instance which are Male*
  2. *Blue Points = Instance which are Female*
  3. *Circled Points = Support Vectors used to obtain margin*
  4. *Strainght Line = Hard Margin*
  5. *Dotted Lines = Soft Margin (with trade off being C=1.6, gamma=0.05 and degree=1)*

With respective to only these two variables, meanfun and sp.ent, It is so evident that our model is not being overfit as it gives a clear distinction between two classes 'Male' and ' Female' with no complications in margins. Thus, accuracy of 0.99 can be valid enough at this point. However, this is just the visualization about margins, we will not visualize how the SVM boundary is placed in a for all our parameters in a 2D space

13.3. Visualizing the Kernel Boundaries:

Here we will visualize the kernel boundaries for each kernel trick modelled in above sessions:



I still consider meanfun and sp.ent to be my favourite variables to visualize my kernel boundaries in a 2D space. I modelled polynomial kernel with same parameters penalty measure of C=1.6, gamma = 0.05 and degree=1 to obtain the above scatter plot. When did, SVM projected my data in a 2-dimensional space and obtained above feature space with boundaries that classifies gender being male and female. From the above figure, we can infer:

- Linear kernel with c=1.6 have a strict boundary
- While in RBF kernel, the boundary is strict and have some points misclassified,
- Polynomial kernel has a linear boundary which are discriminative.
- From above figure, we don't see any complex boundaries for polynomial and hence we need not worry about the model being over fitting

With respective to only these two variables, meanfun and sp.ent, ***It is so evident that our model is not being overfit*** as it gives a clear distinction between two classes 'Male' and ' Female' with no complications in margins in a feature space.

Thus, ***accuracy of 0.993 produced by Polynomial kernel can be valid enough, this means 7 out of 1000 times there could be a misclassification.*** Let us see if we can minimize this error occurrence by increasing the accuracy further using few ensemble learnings.

14. Other Classifiers

We cannot feel complaisant with the accuracy obtained from SVM model, it is always better to build other classifiers to see if the classifier accuracy can be increased further or can the model be well presented to end user. So, I built other classifiers like Decision Tree and KNN and comparing it with SVM.

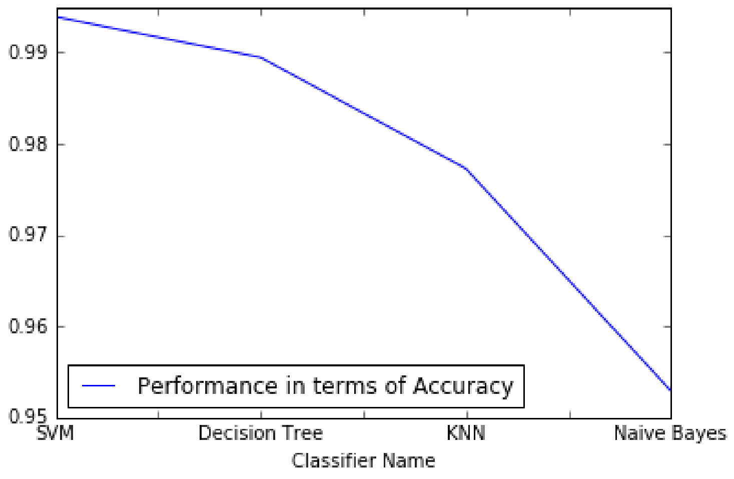
14.1. Building a Decision Tree Classifier with grid search

Accuracy obtained from decision tree classifier for given dataset is 0.9894 which is less when compare to SVM classifier which was 0.99. I can say compare to decision tree SVM model seems more efficient. So, if scrutability is the requirement based on which a model needs to be built we can go ahead with decision tree model.

14.2. Building a KNN model

Accuracy obtained from KNN classifier for given dataset is 0.977 which less when compare to SVM. However, its accuracy touches the benchmark of 0.95 which we decided based on Naive Bayes, we can have this model for any ensemble building, etc., and it not advisable to just discard it. Though KNN perform better than Naive Bayes, its accuracy is less compare to SVM.

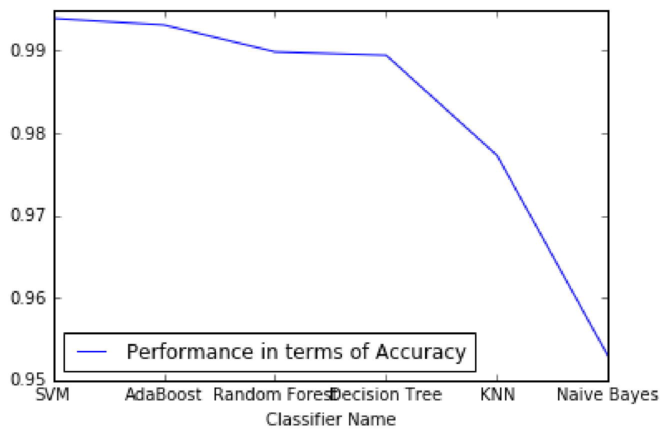
14.3. Comparing results of Individual Classifiers



Classifier Name	Performance in terms of Accuracy
SVM	0.993902
Decision Tree	0.989451
KNN	0.977272
Naive Bayes	0.952901

- From above table and graph, it seems very clear that, SVM with polynomial kernal behaves best.
- ***Accuracy produces by Polynomial kernal equal to 0.993*** is the highest of all cross-validation results obtained from other classifiers.
- Thus, with individual classifiers we can infer that as an individual classifier, ***SVM with Polynomial Kernal does a best classification wrt this voice dataset*** in classifying an instance as Male or Female
- This SVM polynomial kernal tend to miss classify only 7 out of 1000 times when subjected to such dataset which is pretty good.
- However, we will yet try to improve the accuracy further using some ensemble techniques.

15. Ensemble Learning



Classifier Name	Performance in terms of Accuracy
SVM	0.993902
AdaBoost	0.993114
Random Forest	0.989865
Decision Tree	0.989451
KNN	0.977272
Naive Bayes	0.952901

Ensemble is a machine learning approach to combine the output from different models or from a model with multiple number of training and test samples or by means of adding weights to the miss-classified objects and tuning the model to classify them in next prediction. These are in abstract called as ensemble stacking, bagging and boosting techniques in machine learning.

Based on above result shown, decision tree performed quite well with an average accuracy of 0.9894, so I used one of the ***bagging technique called Random Forest***, to see if it will improve my accuracy further. However, it helped very little in improving the accuracy from 0.9894 to *0.9898*. Similarly, I tried a ***boosting technique called Adaboost*** and it produced me an accuracy of *0.9931* which was better than random forest but still less than ***SVM classifier*** with polynomial kernel whose average accuracy was *0.933902* with 10-fold cross validation.

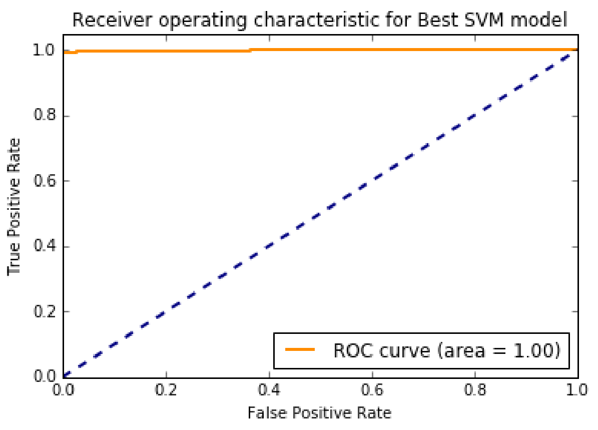
With this ***we can conclude SVM with Polynomial kernel can be the best final model for this dataset***.

16. Final Model

```
Final Model Detail:
 SVC(C=1.6, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=1, gamma=0.005, kernel='poly',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)

Accuracy obtained from final model with 10 fold CV:
 0.991902834008

ROC Computed Area Under Curve:
 0.998720084159
```


Receiver operating characteristic for Best SVM model

With all above findings I can conclude that, ***SVM model using polynomial kernel with penalty parameter C= 1.6, gamma = 0.05 and degree=1 is the the best classifier model*** with average classification accuracy of 0.99190 and ROC computed area under curve equal to 0.99872.

Jupyter Notebook:

Click here to view my jupyter notebook (*https://nbviewer.jupyter.org/github/pradeepsathyamurthy/university_projects/blob/master/Kernal_Tricks_Using_SVM_for_Gender_Classification_BasedOn_VoiceData/SVM%20and%20its%20Kernel%20Exploration.ipynb*)

Reference

[1] https://en.wikipedia.org/wiki/Voice_frequency

[2] SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognitionhttp://ieeexplore.ieee.org/abstract/document/1641014/?reload=true

[3] Kernel-based methods for hyperspectral image classificationhttp://ieeexplore.ieee.org/abstract/document/1433032/

[4] Efficient leave-one-out cross-validation of kernel fisher discriminant classifiershttp://www.sciencedirect.com/science/article/pii/S0031320303001365

[5] Early Detection of Breast Cancer using SVM Classifier Technique https://arxiv.org/abs/0912.2314

[6] Text Classification using String Kernels http://www.jmlr.org/papers/v2/lodhi02a.html

[7] The Harvard-Haskins Database of Regularly-Timed Speech

[8] Telecommunications & Signal Processing Laboratory (TSP) Speech Database at McGill University, Home

[9] VoxForge Speech Corpus, Home

[10] Festvox CMU_ARCTIC Speech Database at Carnegie Mellon University

Citation:

Boser, B.E., Guyon, I.M. & Vapnik, V.N. A training algorithm for optimal margin classifiers. in *5th Annual ACM Workshop on COLT* (ed. Haussler, D.) 144–152 (ACM Press, Pittsburgh, PA, 1992).

Python Code:

Code has more than 1000 lines with respective comments inherited in it and hence copy pasting them here will make the document bulky. I have given an online link of my live NBViewer in above jupyter notebook link. I am attaching the python file for quick download here.

SVM and its Kernal Exploration.py

Appendix

Below are Result of the raw dataset ***without taking the domain knowledge into consideration* and not removing the noisy data** and just performing normalization on the dataset:

Linear Kernel:

	Dataset	Accuracy
0	Partially Normalized	0.899
1	Dimention Reduced	0.900
2	Completely Normalized	0.977

Gaussian (RBF) Kernel:

	Dataset	Accuracy
0	Partially Normalized	0.892744
1	Dimention Reduced	0.919558
2	Completely Normalized	0.976341

Polynomial Kernel:

	Dataset	Accuracy
0	Partially Normalized	0.954259
1	Dimention Reduced	0.933754
2	Completely Normalized	0.958991

Sigmoidal Kernel:

	Dataset	Accuracy
0	Partially Normalized	0.391167
1	Dimention Reduced	0.664038
2	Completely Normalized	0.793375

Report on Best Kernels (80-20 Split):

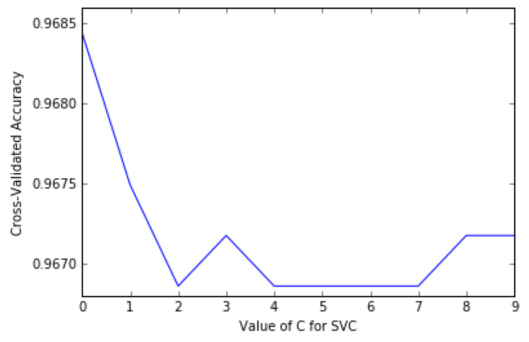
	Dataset	Kernal	Accuracy
0	Completely Normalized	Linear	0.977918
1	Completely Normalized	Gaussian	0.976341
2	Completely Normalized	Polynomial	0.958991
3	Completely Normalized	Sigmoidal	0.793375

Report on Best Kernels (10 Fold CV):

	Dataset	Kernal	Accuracy
0	Completely Normalized	Linear	0.969413
1	Completely Normalized	Gaussian	0.965938
2	Completely Normalized	Polynomial	0.958991
3	Completely Normalized	Sigmoidal	0.790709

Penalty Parameter 'C':

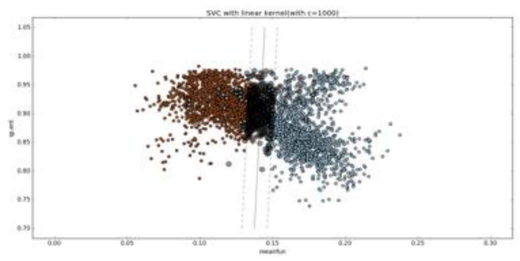
Visualising the Accuracy Performance

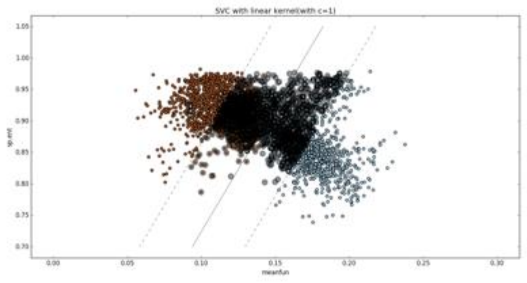


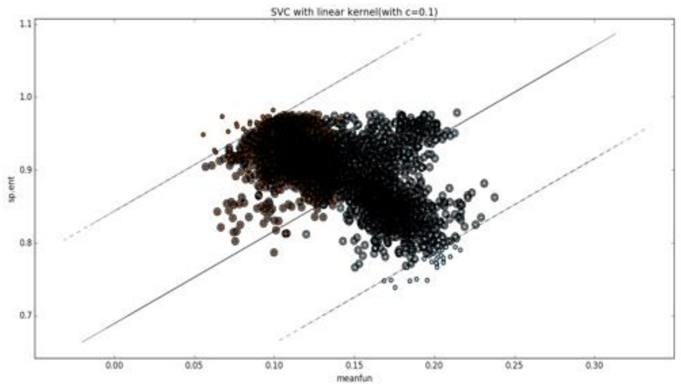
Accuracy Values

	Penality Parameter C	Accuracy
0	0.1	0.968439
1	1.1	0.967491
2	2.1	0.966860
3	3.1	0.967175
4	4.1	0.966860
5	5.1	0.966860
6	6.1	0.966860
7	7.1	0.966860
8	8.1	0.967175
9	9.1	0.967175

Margins:

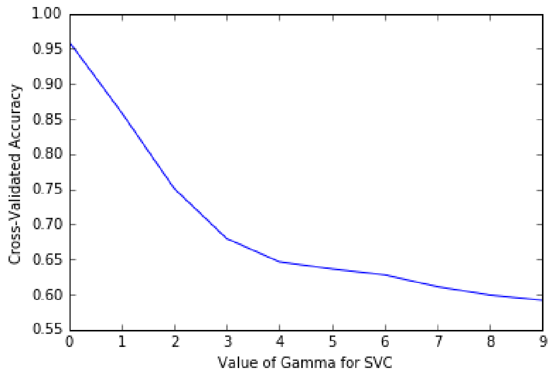




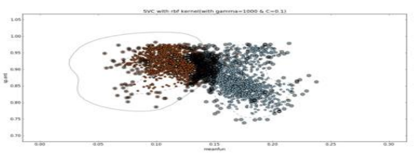


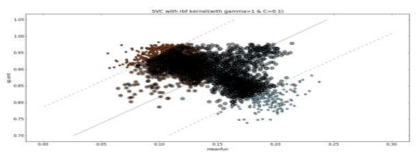
Tuning Parameter 'gamma' in RBF Kernel:

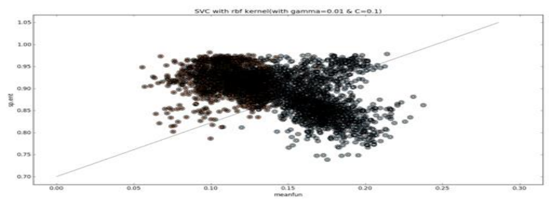
Visualising the Accuracy Performance	Accuracy Values



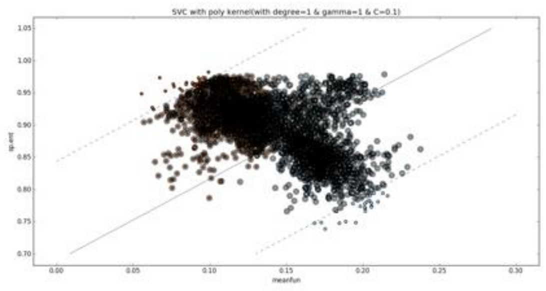
	Parameter Gamma	Accuracy
0	0.1	0.960844
1	1.1	0.859452
2	2.1	0.751217
3	3.1	0.679589
4	4.1	0.646445
5	5.1	0.636340
6	6.1	0.627815
7	7.1	0.611095
8	8.1	0.599103
9	9.1	0.591845

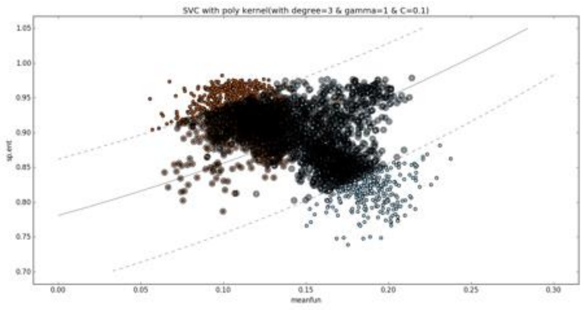


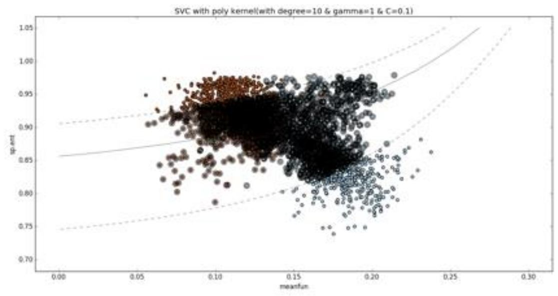


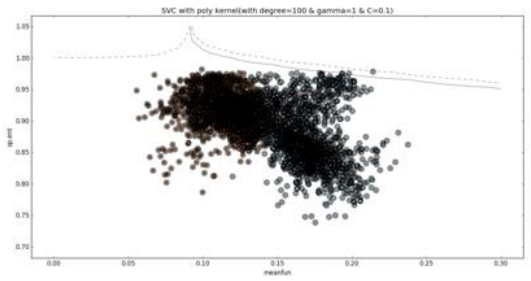


Tuning Parameter 'degree' in Polynomial Kernel:

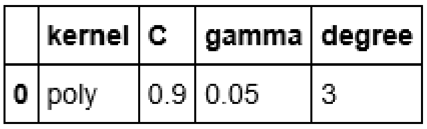




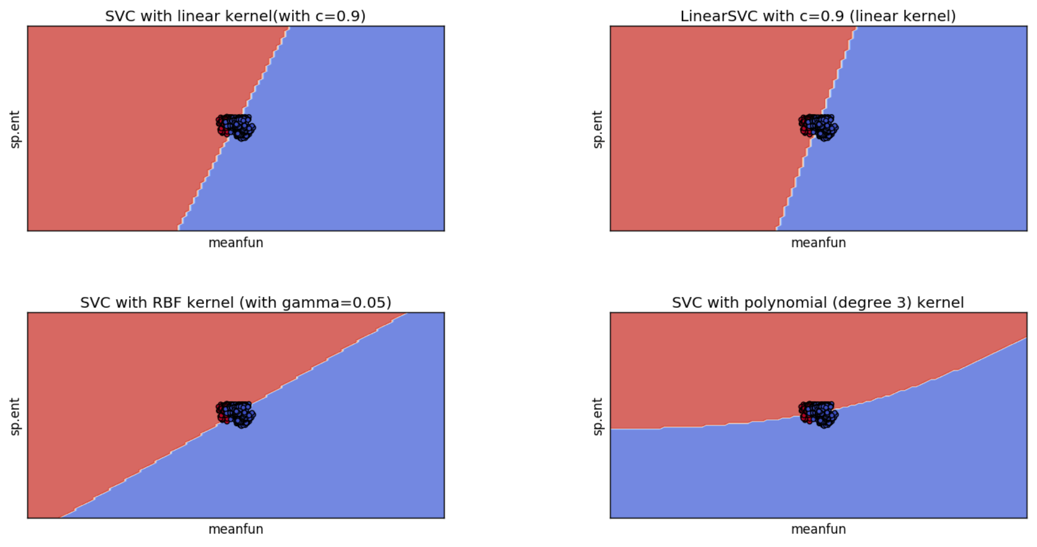




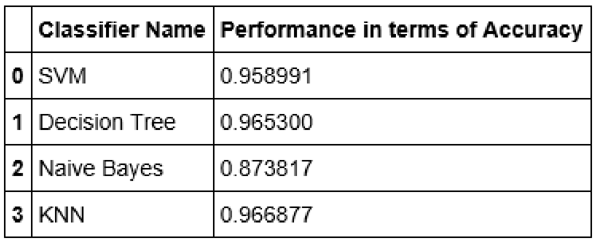
Obtaining Best Parameters using Grid Search:



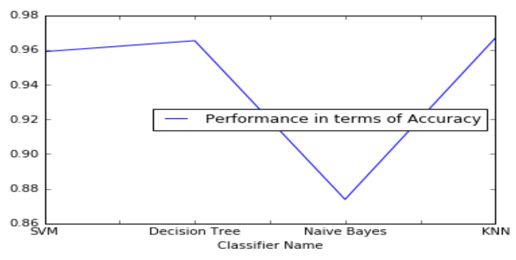
	kernel	C	gamma	degree
0	poly	0.9	0.05	3



Performance of different Classifiers:



	Classifier Name	Performance in terms of Accuracy
0	SVM	0.958991
1	Decision Tree	0.965300
2	Naive Bayes	0.873817
3	KNN	0.966877



These results and visualizations in Appendix shows how important is the domain knowledge in solving most of the dataset. As it is the quality of the training data which decides each model efficiency!!! Thank You!