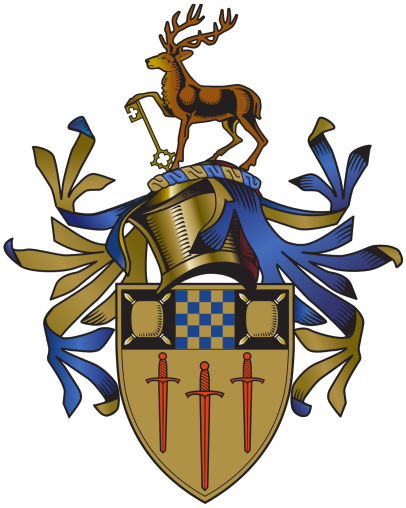
COMM055 Coursework:

Beanie Intelligence Analysis Report



University of Surrey

Authors:
Benjamin Baxter
Hannes Kastenhuber
Konstantinos Kompogiannopoulos
Luxman Elangeswaralingam
Victor Bonilla Pardo

Word Count: 7267

Contents

1 Introduction

This document presents the Data Mining process over two data sets with special emphasis on model development, evaluation and comparisons. Several hypotheses are discussed with their corresponding data pre-processing and modelling phases.

2 Data sets

2.1 Spotify API Data Set

2.1.1 Funky Hypothesis

Using the Spotify Data Mining tool our team created [3], I downloaded the top 200+ Funk songs [2], with the addition of an extra 13 that are mentioned as "Honorable mentions". The reason I decided to choose this list of songs was to eliminate as much personal bias as I could from the dataset.

2.1.2 Genre and Popularity

After developing the previous mentioned tool, it got used to download a number of songs per genre. Next to the genre and general analytic features, the data set also offers a popularity feature per song, all based on spotify song analysis and user data.

2.2 Beijing Multi-Site Air-Quality Data Data Set

This data set includes hourly air pollutants data from 12 air-quality monitoring sites with meteorological data from the nearest weather station. The time period is from March 1st, 2013 to February 28th, 2017 [6].

- Data Set Characteristics: Multivariate, Time-Series
- Number of Instances: 420768
- Area: Physical
- Attribute Characteristics: Integer, Real
- Number of Attributes: 18 (No, year, month, day, hour, PM2.5, PM10, SO2, NO2, CO, O3, TEMP, PRES, DEWP, RAIN, wd, WSPM, station)
- Date Donated: 2019-09-20

- Associated Tasks: Regression
- Missing Values? Yes

3 Hypotheses

3.1 Spotify API Data Set

3.1.1 Funky Hypothesis

Using a few of the features listed in [5], some of the most promising features seemed to be "*energy*", "*time_signature*", "*key*", "*mode*", "*popularity*", "*valence*" and "*tempo*". With that in mind, I set out to predict the "*danceability*" label of a given song. The creation of the label is described in section 4.1.1.

3.1.2 Genre and Popularity

The variety and type of parameters per song in this data set with the additional genre information raises the question, if the genre of a song is dependent on a number of the features listed in [5]. With the additional popularity information it can also be assessed if features like "*energy*" and "*danceability*" have an impact on how well a song is received by the majority of people.

3.2 Beijing Multi-Site Air-Quality Data Data Set

3.2.1 Level of Pollutants Seasonality Hypothesis

Given the nature of the data, that is time-series of different pollutants measures, some common questions for each pollutant are "How do the data group over the time?" and "Do global groups of data exist and repeat over time?". A factor of this problem is the similarity between the data, which might conclude to seasonality in the time-series.

One learning approach that addresses similarity between data points is clustering algorithms. As the data set doesn't contain ground truth labels, the problem falls in the unsupervised learning category. The fact that ground truth labels aren't present indicates that the potential performance metrics will have to measure aspects from the resulted clusters such as how well defined and separated are from each others rather than measure the error given the result and the ground truth label.

3.2.2 Estimating the AQI values without having access to all the chemical values

This hypothesis aims to test whether we can predict the AQI values without having access to all the sensor reading. This might be useful in situations where a sensor is malfunctioning and fails to report all the values. The AQI values is an important measure used to determine the quality of the air and can be used by the public to protect them from inhaling very harmful chemicals such as PM10 and PM2.5. Not having access to the AQI values in the data set, this had to be calculated using the formula and information available to us.

3.2.3 Warning prediction for dangerous chemical levels

A major environmental problem in many large cities is that of photo-chemical air pollution. This is a phenomenon created by a series of chemical reactions, triggered by solar radiation. Nitrogen oxide(NO2), nitrogen dioxide (Nox), ozone (O3) and the energy that comes through solar radiation (h), are the elements of the photo-chemical reaction. Several health researchers have proven that nitrogen dioxide and ozone can cause serious health hazards to the public.

The hypothesis of this model is to try and predict the levels of NO2 and O3 to give a result that would, within a certain accuracy, indicate whether the level could potentially be harmful to people. This would ideally be able to give warning 24 hours in advance if these events were likely to occur.

4 Data Preparation

4.1 Spotify API Data Set

4.1.1 Funky Hypothesis

First, I checked the correlation and relationships between the different variables. I disregarded all the variables that were calculated from each other (i.e. had 1.00 linear correlation). The only correlations that are greater than 0.25 and less than 1.00 are the following:

- "*loudness*" and "*energy*" $\approx$ 0.594
- "*danceability*" and "*valence*" $\approx$ 0.404

- "*speechiness*" and "*liveness*" ≈ 0.271

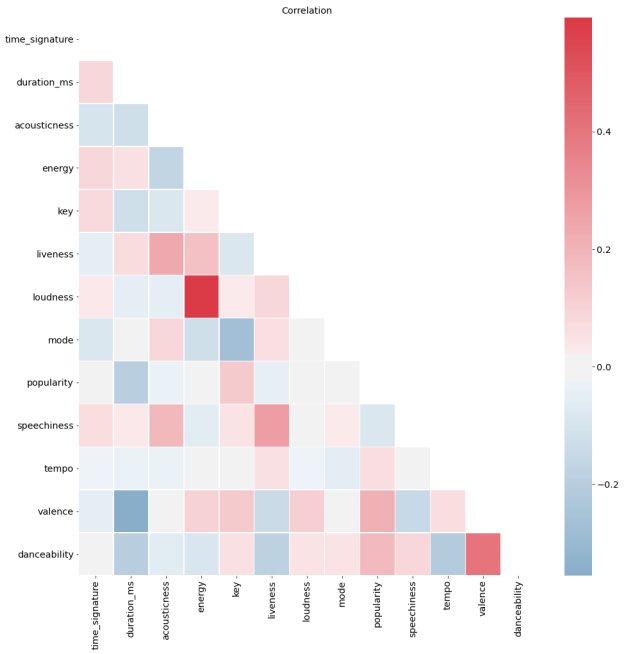


Figure 1: Correlation between selected features.

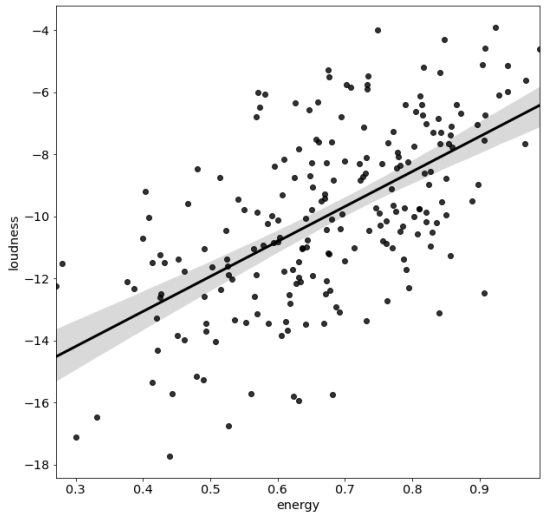


Figure 2: Correlation between energy and loudness.

Even though there is no apparent correlation with the "*danceability*" feature, other than "*valence*", which indicates how happy or sad a song is, I decided to proceed with my hypothesis.

Using the numerical "*danceability*" feature provided via the Spotify Analyzer, I created a "*danceability_labels*" feature that has 4 classes:

- Class 0: Not danceable, $\forall$ "*danceability*" $\in [0.0, 0.5)$, with 14 songs.
- Class 1: Somewhat danceable, $\forall$ "*danceability*" $\in [0.5, 0.7)$, with 65 songs.
- Class 2: Very danceable, $\forall$ "*danceability*" $\in [0.7, 0.8)$, with 63 songs.
- Class 3: Extremely danceable, $\forall$ "*danceability*" $\in [0.8, 1.0)$, with 71 songs.

Furthermore I looked into the skewness of each attribute, or the measure of the asymmetry of the distribution of each of the selected features. This is better represented in figures (See section 9.1).

Column	**Skew**
time_signature	-10.246
duration_ms	3.658
acousticness	1.108
energy	-0.307
key	-0.147
liveness	1.905
loudness	-0.175
mode	-0.336
popularity	-0.188
speechiness	2.225
tempo	1.647
valence	-1.311
danceability	-0.773

Table 1: Skewness of attributes.

The results of checking for the skewness of each parameter shows that "*time_signature*" or how many beats are in each bar of a song, was the most skewed one. Which also validates why it has almost 0 correlation with any of the other features. Similarly the skewness of "*duration_ms*" tells us that the song lengths vary widely from song to song.

With that and the correlation plot (see figure 1) in mind, I decided to drop these two features from the exploration, instead of removing the skewness from them. This decision was taken, due to the limited size of the dataset.

4.1.2 Genre and Popularity

The data used in the following processing and modelling was mined with the "*get playlists and artists from genres*" function of the spotify data mining project [3]. The genres used for this project are "*country*", "*funk*", "*hiphop*", "*pop*", "*metal*", "*soul*", "*rock*", "*reggae*" and "*jazz*". After a first glance of the data set, a number of features stand out due to the fact that they are all scaled from 0 to 1 and complete for the entire data set. A few additional columns seem also interesting in relation to the hypothesis at hand. Interesting and important for the success of machine learning algorithm is the distribution of the possible model parameters. The following graph shows the distribution of the parameters of interest in order to have a first idea if additional pre-processing needs to be done.

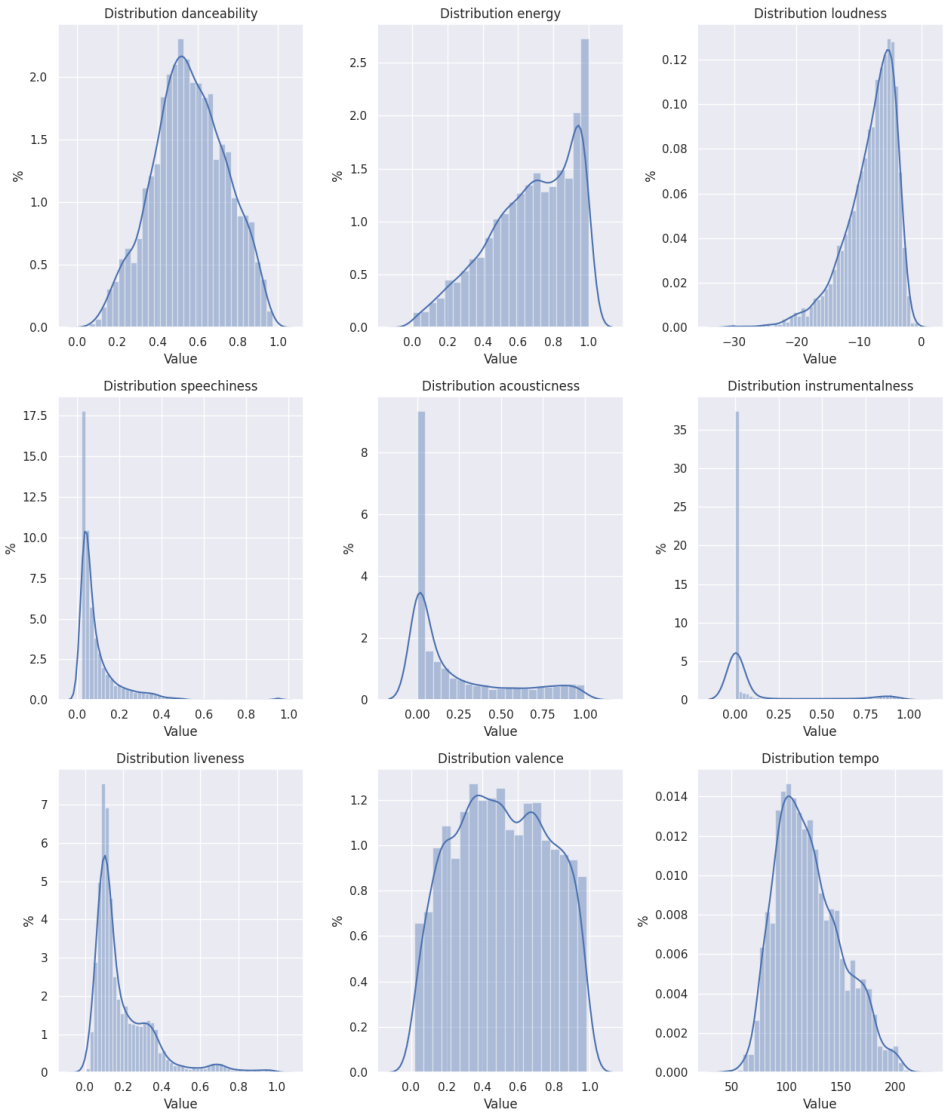


Figure 3: Correlation between selected features.

Most of the distributions are surprisingly alright, considering the randomness of the data selection. Features like "*speechiness*" and "*instrumentalness*" do contain outliers, which can be seen at the little movement in the distribution graph towards the converging ends. But considering the meaning of those features, following the data dictionary of the spotify python library [5], the importance is actually to high and those numbers too meaningful, so they are kept inside the model.

"*Instrumentalness*", for example predicts whether a track contains no vocals. This is a very important difference to songs that contain vocals. Those feature also could be used as binary features like "*True*" and "*False*", but due to the fact that hey already are scaled in a percentage like matter they are kept numerical.

The features "*loudness*" and "*tempo*" actually do need additional pre-processing. As already mentioned, the majority of features is scaled in a range from 0 to 1. "*Loudness*" on the other hand reaches, according to the dictionary, from around -60 to 0, with -60 being very loud. For conformity the loudness got altered. Also "*tempo*" is not scaled yet. From our data we can see that this feature reaches from 0 to around 250, it got altered too. The following shows the alteration/scaling of those two features.

$$NewLoudness = (1 - \frac{OldLoudness}{-60}), NewTempo = \frac{OldTempo}{250} \tag{1}$$

After the distribution it is useful to take a look at correlations in between the features of a model. Correlation can lead to weak learning performances, and waste of resource due to the relation between one or more features and the resulting unnecessary complexity of the model and the redundancy throughout the learning. In order to avoid those mistakes, the correlation of the parameters for this model are assessed over a so called Correlation Matrix.

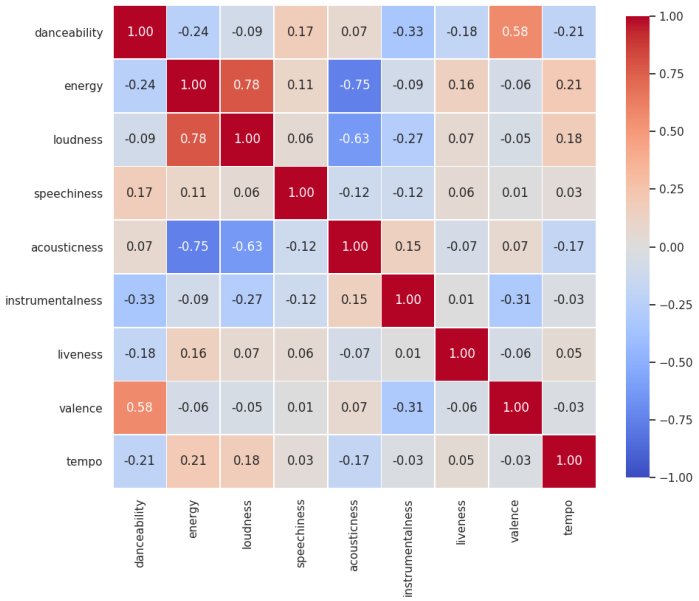


Figure 4: Correlation between selected features.

There is surprisingly not a lot of correlation throughout the data set, considering the fact that a lot of columns do mean similar things. The correlation between "*loudness*" and "*energy*" is the highest. Those two features have similar meanings and generally

describe the intensity of a track. Since "*loudness*" has the better looking distribution (closer to a normal distribution), the "*energy*" feature gets removed from the data set.

After those minor changes, the data set is complete and ready to be analyzed.

4.2 Beijing Multi-Site Air-Quality Data Data Set

4.2.1 Level of Pollutants Seasonality Hypothesis

The data preparation had five phases in this hypothesis:

1. Selecting the data from the Huairou air-quality monitoring site.
2. Aggregating and converting the timestamp to data set index.
3. Resampling the frequency of the data from hourly to daily by calculating the mean for each day. The reason is because the standard deviation of the measures is high and the measurement frequency is too high for the pace of the domain. Also, the majority of political organizations use daily average data for decision making [1].
4. Applying cubic interpolation to fill missing values.
5. Selecting a combination of features from the next features:
   - "*NO2*" (air pollutant concentration in ug/m^3).
   - "*TEMP*" (temperature in Celsius degrees).
   - "*PRES*" (pressure in hPa).
   - "*DEWP*" (dew point temperature in Celsius degrees).
   - "*RAIN*" (precipitation in mm).
   - "*WSPM*" (wind speed in m/s).
6. Scaling the selected combination of features by centering to the mean and component wise scaling to unit variance for each feature.

4.2.2 Estimating the AQI values without having access to all the chemical values

Data preparation is a crucial stage for developing accurate models. We began by analysing at some statistical measures and obtaining the percentage of the missing values in the data frame. It was possible to observe that the maximum rate of missing values for any column was no more than 5% as seen from the table below.

Feature with missing values	% of missing values
PM2.5	2.07
PM10	1.53
SO2	2.14
NO2	2.87
CO	4.9
O3	3.15
TEMP	0.09
PRES	0.09
DEWP	0.09
RAIN	0.09
wd	0.43
WSPM	0.07

Table 2: Percentage of missing values

Missing value in the column 'wd' was removed from the data set. Statistically, this feature contained a tiny percentage of missing values, and therefore it was possible to remove the rows altogether. The remaining missing values were filled using a combination of forward-fill and by the mean to maintain the same distribution of the data.

The second stage of pre-processing involved identifying any outliers. It was possible to observe from the box plot that the chemical features contain few outliers. This could have been caused by faulty sensors or error in reporting these values. The box-plot can be seen below.

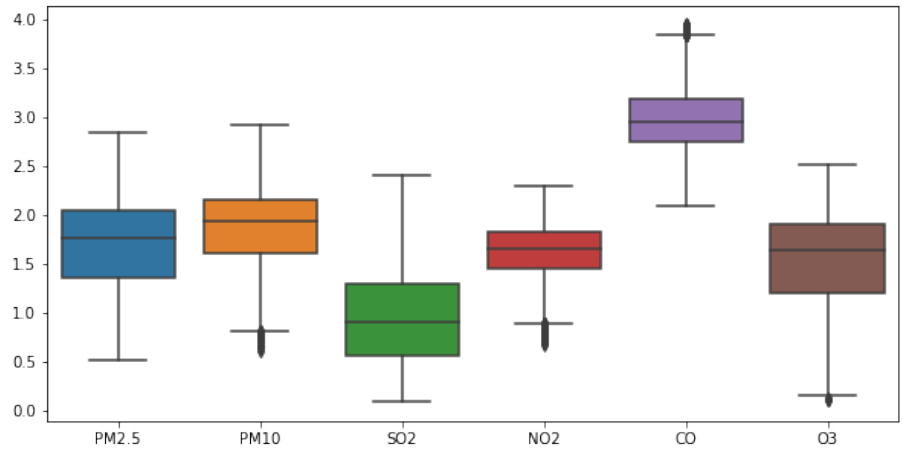


Figure 5: Box-Plot for the chemicals

The correlation matrix gives us the underlying correlation between features in our data set. The graph presented below shows the results for our data set. Dark areas on the plot represent the strongly correlated features, and lighter shaded for less correlated features. This information is precious and can be used for feature selection. It is also worth noting that there is a strong correlation between the various chemicals. From this pre-processing stage, we can conclude that we do not need every chemical value when building a model.

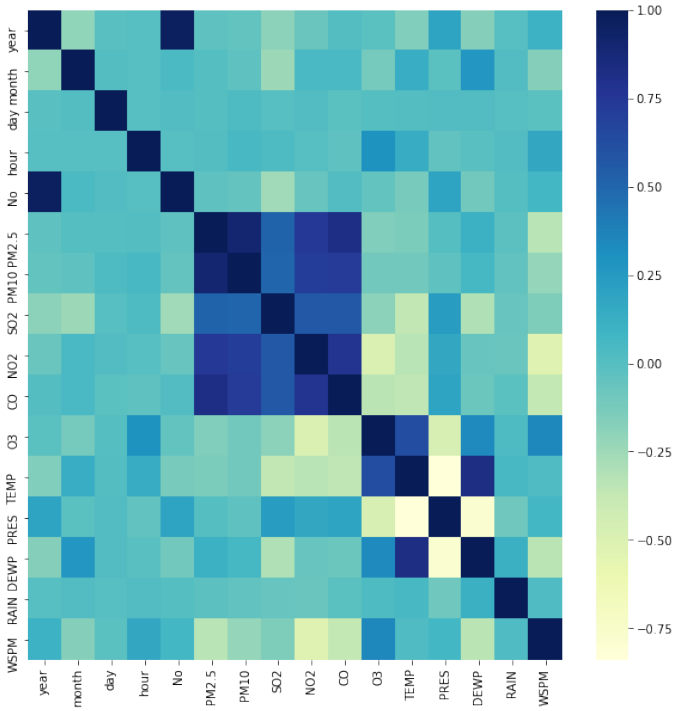


Figure 6: Correlation Matrix

The final step of pre-processing was to calculate the AQI values. Air Quality Index is a measure that is commonly used to describe the amount of pollution in the air. This measure is used worldwide, and it is calculated using the five chemicals PM2.5, PM10, SO2, NO2, CO and O3. To obtain the AQI values, we should first calculate the moving average of these features and calculate the individual IAQI scores. In the end, we selected the IAQI with the highest values at the final AQI.

The formula presented below is used to calculate the IAQI and AQI values [1].

$$I = \frac{I_{\text{high}} - I_{\text{low}}}{C_{\text{high}} - C_{\text{low}}}(C - C_{\text{low}}) + I_{\text{low}} \tag{2}$$

4.2.3 Warning prediction for dangerous chemical levels

When performing exploratory analysis the main focus to start with was to check the completeness of the dataset. As with almost all datasets to expect a complete dataset with no missing values or errors is almost impossible. The first step was to look at the data and figure out a way to fill this data without affecting its accuracy. The main features included and what we want to predict in this hypothesis are 'NO2' and 'O3'.

```
hour             0
date_time        0
NO2          12116
O3           13277
station          0
dtype: int64
hour         0.000000
date_time    0.000000
NO2          2.879497
O3           3.155421
station      0.000000
dtype: float64
```

Figure 7: Original missing value count.

When looking at the missing values from NO2 (NA Values: 12116,Total Percent Missing Values: 2.88 Percent) and O3 (NA Values: 13277, Total Percent Missing Values: 3.16 Percent) it could be a lot worse but it is still a lot of data missing. Assuming that these are independent missing value events it has the potential of missing around 6 Percent of the data in a worse case scenario.

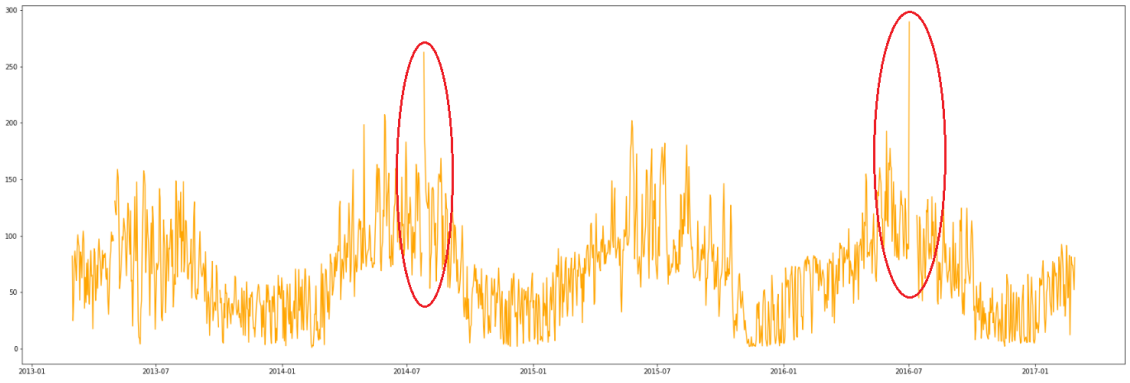


Figure 8: Dongsi station O3 missing data.

When looking into the missing values it appears that they are not at random and seem to be at certain points and continue for a number of hours. This would lead one to assume that the device recording the data could have malfunctioned during these time

periods. Also when looking at the graphs especially for the Dongsi station at the time of these errors there can be large spikes in the values which can lead to outliers.

After looking at the cause of these missing values and the hypothesis it would appear that merging the different stations and taking the average could be a suitable solution as we are trying to look at a general warning for the city of Beijing. Another way to try and minimise these missing values would be to combine the hours to make an average over the whole day as we are not trying to predict from hour to hour but day to day.

```
station        0
date_time      0
NO2          138
O3           121
dtype: int64
hour         0.000000
date_time    0.000000
NO2          0.787132
O3           0.690167
dtype: float64
```

Figure 9: Post missing value count.

With this result we can see that the missing values have been reduced to around 1.5 Percent which is a much better result than before. The remaining values can be filled in with interpolation. The issue with doing this is that the row size is reduced from 35064 to 1461. When trying to run models which require a large dataset this could be an issue. In this case the data could just be interpolated.

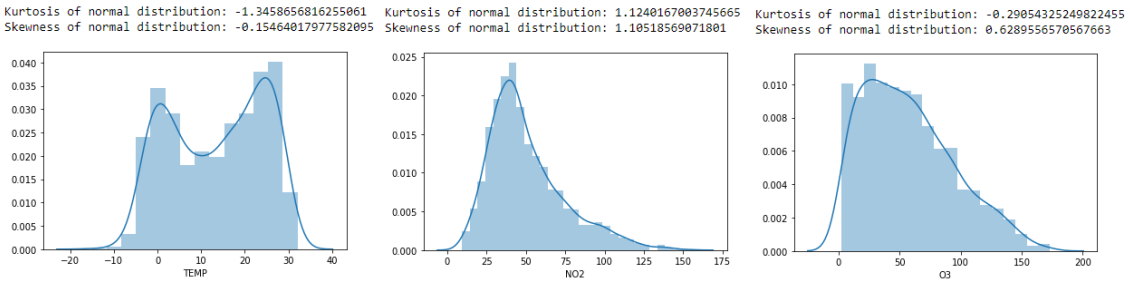


Figure 10: Distribution of the features.

When looking at the distribution of the data it is slightly skewed but not to a degree where it would heavily impact the model. When looking at temperature we can see the two peaks showing the summer and winter temperatures.

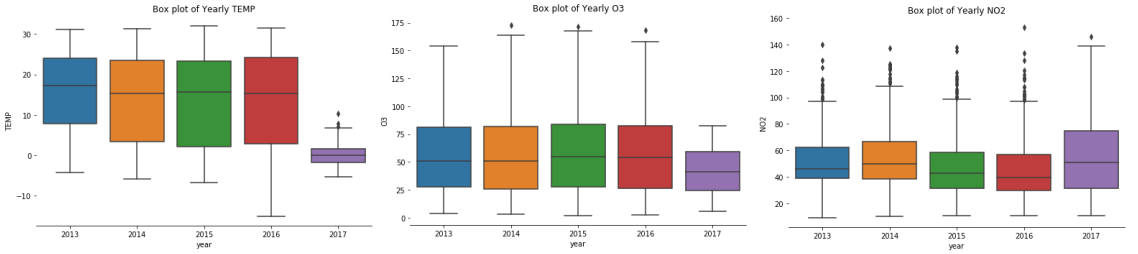


Figure 11: Box plot by year.

When looking to see any seasonality patterns the first think checked was the data from year to year. As we can see the data stays quite consistent during this except for the year 2017, but essentially can be negated as it only contains data up until February 28th.

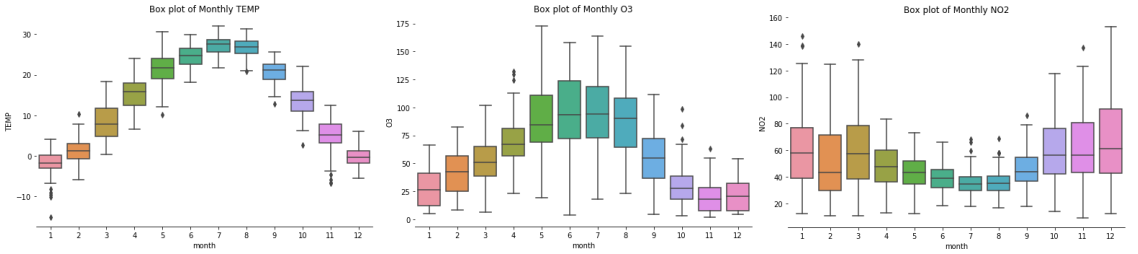


Figure 12: Box plot by month.

When looking into the change by month a clearer pattern begins to emerge. We can see what we would expect of temperature, that it increases during the summer months. We can see with this that O3 is also affected in the same way as temperature as it increases during the same months. With NO2 it would appear that the summer months have a negative effect of reducing the NO2 levels.

5 Modelling

5.1 Spotify API Data Set

5.1.1 Funky Hypothesis

After doing some research on which models would be best to run, I decided on the following:

- Multi-Class Support Vector Machine (SVM):

  - Even though SVM models are inherently 2-class classifiers. The current class model I am working with has 4 classes, which can be binarized. Then a Multi-Class SVM can be used to solves the problem as it forms multiples of two classes. [9]

- Multi-Class Decision Tree:

  - The Decision Tree classifier, is a classic approach for Multi-Class classification.

- Naive Bayes, KNN and the Bagging classifier are all also Multi-Class classifiers that I decided to look into as a batch.

5.1.2 Genre and Popularity

Considering the "*genre*" - hypothesis, a natural unsupervised classification approach that classifies the data based on the received features, seemed the most appropriate to investigate correlations between the features and the music genre. Self-organizing maps are a type of artificial neural network that uses an unsupervised learning approach to produce classification. This model got chosen due to the natural/unsupervised approach and the impressive visualisation methods. Self organizing maps perform very well with low-dimensional data and can often be used for dimensionality reduction. In order to compare the performance of the SOM classification, a Multi-Class Decision Tree Classification with the same data gets executed too.

Since "*popularity*" is also a feature of interest, the same models get executed to classify the "*popularity*". In order to keep the dimensions low, the "*popularity*", that contained values from 0 to 99, is binned in 5 bins with a size of twenty.

5.2 Beijing Multi-Site Air-Quality Data Data Set

5.2.1 Level of Pollutants Seasonality Hypothesis

The clustering algorithm chosen was k-means. The modelling process consisted in the hyper-parameter tuning of (1) the features selected defining the input data points and (2) the number of clusters. On the other side, there were four fixed hyper-parameters: (1) the method of initialization, (2) the number of times the algorithm is run with different centroid seeds, (3) maximum iterations of the algorithm and (4) the relative tolerance with regards to Frobenius norm to declare convergence.

The hyper-parameter tuning process was defined with the help of domain knowledge by combining two manually specified sets: a list of eight combinations of features and a list of four different number of clusters, resulting in $8 * 4 = 32$ models.

1. Set of combinations of features:
   - "*['NO2', 'TEMP', 'WSPM']*".
   - "*['NO2', 'TEMP', 'PRES', 'WSPM']*".
   - "*['NO2', 'TEMP', 'DEWP', 'WSPM']*".
   - "*['NO2', 'TEMP', 'RAIN', 'WSPM']*".
   - "*['NO2', 'TEMP', 'PRES', 'DEWP', 'WSPM']*".
   - "*['NO2', 'TEMP', 'PRES', 'RAIN', 'WSPM']*".
   - "*['NO2', 'TEMP', 'DEWP', 'RAIN', 'WSPM']*".
   - "*['NO2', 'TEMP', 'PRES', 'DEWP', 'RAIN', 'WSPM']*".
2. Set of number of clusters: "*[2, 3, 4, 5]*".

5.2.2 Estimating the AQI values without having access to all the chemical values

Multiple approaches could have been taken to test this hypothesis using different Machine Learning techniques and Algorithms. Regression or Time Series models could have been used to get an estimate of the Air Quality Index. The approached I took was to solve this problem using classification methods. Converting the AQI values into categorical values allowed me to run a KNN and Decision Tree Classifiers to predict the new data. Categorising the AQI values provides a more natural way to understand the pollution level and how harmful this can be to the human body.

AQI Range	Category
(0-50)	0
(51-100)	1
(101-150)	2
(151-200)	3
(201-300)	4
(>300)	5

1. Features Considered:

- *" ['PM10','PM10_24HRAvg','O3','O3_8hrAvg','TEMP' , 'PRES' , 'DEWP' , 'RAIN']"*

2. Models:

   - KNN
     - Hyper-parameter Tuning: Number of K neighbours
     - Evaluation: Confusion Matrix, Cross Validation and Accuracy Score
   - Decision Tree
     - Hyper-parameter: Solver(gini, entropy) and ccp_alpha
     - Evaluation: Confusion Matrix, Cross Validation and Accuracy

In the end after fine tuning the KNN and Decision Tree modesl I compared the results with a Random Forest.

5.2.3 Warning prediction for dangerous chemical levels

In this model we are using a multivariate time series to try and forecast future data so that people can have a warning of potentially dangerous levels of O3 and NO2. With this we use a VAR calculation combined with a LSTM, using the VAR to aid in the prediction of the neural network[4].

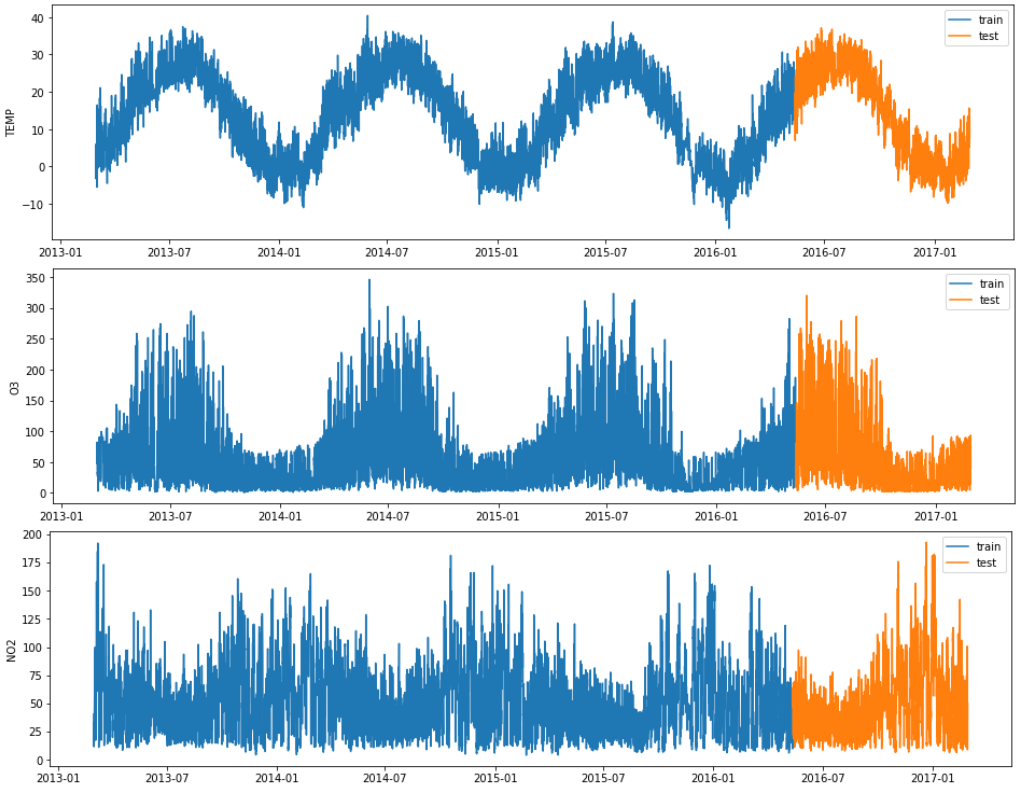


Figure 13: 80/20 split of the features.

With the data there is a 80/20 split between the training and the test data. In the processing linear interpolation is used to fill in the missing values and the stations have been merged together to try and unify the results to the general area of Beijing.

With the VAR model it is able to interpret the relationships between multiple inputs and is better at showing the behavior of the data and results in better forecasting. With the VAR, as it is a multivariate generalisation of ARIMA, it needs to be stationary and have auto-correlation removed from it.

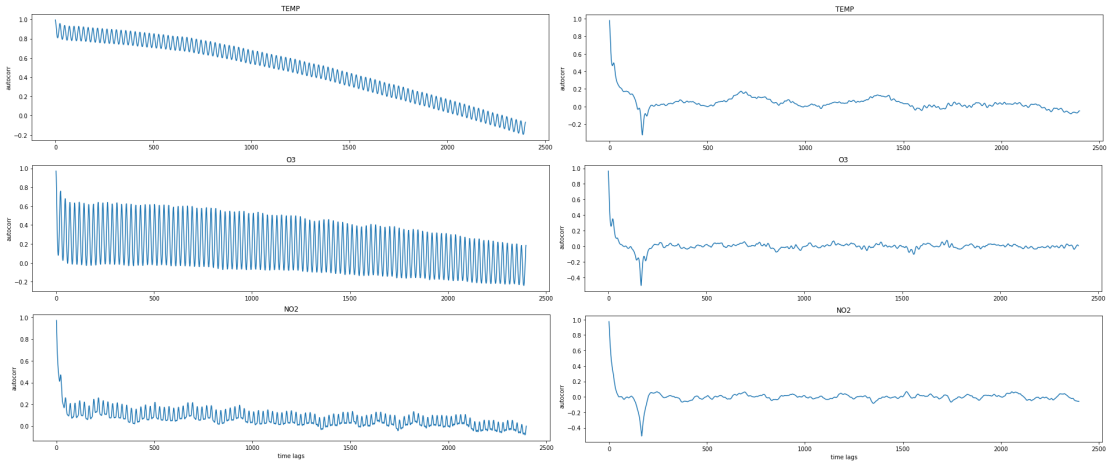


Figure 14: Auto-correlation (Left: before differentiation, Right: After differentiation)

After these are completed we then use the AIC criterion to try and find the best lag order to use in the model. The model is recursively fit until the lowest score is found.

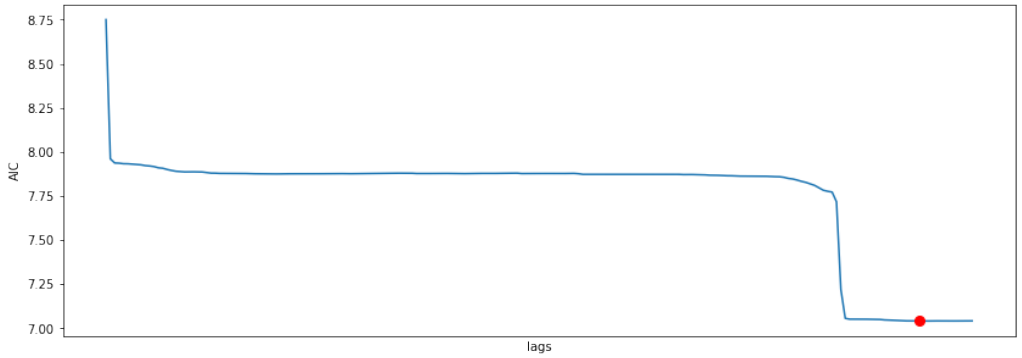


Figure 15: AIC score

Once this is done the VAR now needs to be combined with the LSTM. Ideally the VAR model has learned the behaviour of the multivariate data and now can be used. The model is trained in a two step procedure. The first is to start training the LSTM model and at one step ahead forecast the multivariate output using the fitted values created by the VAR model. After this it finished training by using the original data that was used to fit the VAR model.

6 Results and Evaluation

6.1 Spotify API Data Set

6.1.1 Funky Hypothesis

As discussed in section 5.1.1, 3 separate model "batches" were run to predict the "*danceability_label*" of a funk song.

Initially the Multi-Class SVM Classifier was ran, as following:

```
1 OneVsRestClassifier(svm.SVC(
2                 kernel='poly', degree=2, probability=True, tol=1e-6,
                      random_state=self.random_seed))
```

The OneVsRestClassifier and the pre-processing to ensure the feature columns are fed in binarized assures that the model will run as intended.

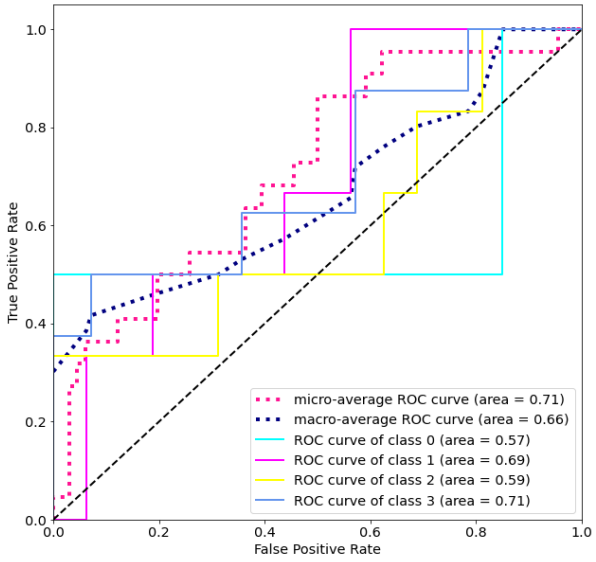


Figure 16: The ROC curve for the different class prediction accuracy that SVM produced.

In evaluating this model, we can observe that even though the averages for the area under the curve, indicate a well performant model, it is not the case. Looking at the area under the curve for each individual class, we can see that in class 0 with the least number of songs, and class 3 the model did not perform as well.

The Multi-Class Decision Tree Classifier was ran with the default settings (see [7]). Pre-processing ensued, to ensure the feature columns are binarized such that the model will run as intended.

Classes	precision	recall	f1-score	support
0	0.00	0.00	0.00	2
1	0.33	0.33	0.33	6
2	0.27	0.50	0.35	6
3	0.60	0.38	0.46	8
accuracy			0.36	22
macro avg	0.30	0.30	0.29	22
weighted avg	0.38	0.36	0.35	22

Table 3: Confusion Matrix for Decision Tree

In evaluating this model, we can observe that due to the 0.1 % split for the test data, there were only two songs for class 0, which where miss classified, similarly for all other classes, the sample of songs should not make that much difference in the overall accuracy and in this case could be attributed to the random seed the model was run with, since the model accuracy is only 36 %.

With the following results in mind I decided to run a k-fold cross validation algorithm on a batch of models (including the Decision Tree), to test if this would smooth out the bias of the models and reveal their true accuracy. As mentioned in section 5.1.1, I decided to run the "*Naive Bayes - Gaussian*", "*KNN*", "*Bagging*" and "*Decision Tree*" classifiers.

The settings for the following models were as such:

```
1 extra_tree = ExtraTreeClassifier(random_state=self.random_seed)
2 models = [('Naive Bayes', OneVsRestClassifier(GaussianNB()))), ('KNN',
      OneVsRestClassifier(KNeighborsClassifier(n_neighbors=len(
3             X.columns)+1, metric='euclidean', n_jobs=-1, weights='
                  distance'))),
4         ('Bagging', OneVsRestClassifier(
5             BaggingClassifier(extra_tree, random_state=self.random_seed
                  ))))
6             ('Decision Tree', OneVsRestClassifier(
                  DecisionTreeClassifier()))]
```

The following models where run with k-fold cross validation of $k = 10$ and evaluated with ROC curves.

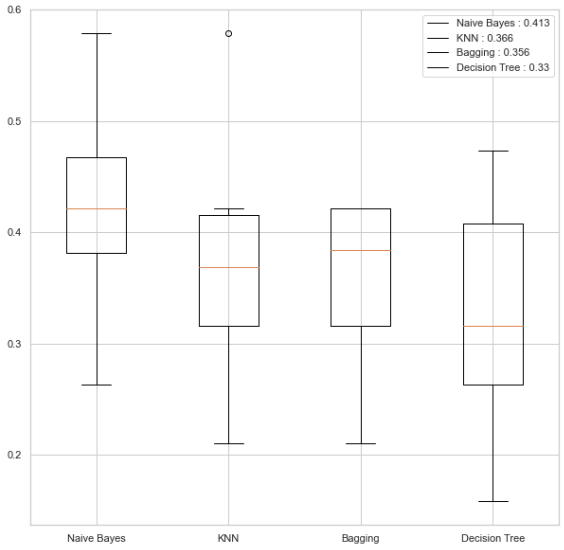
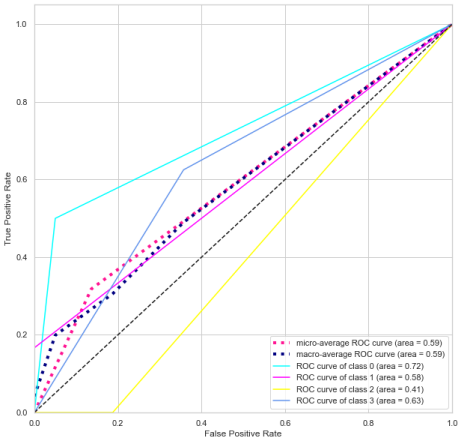
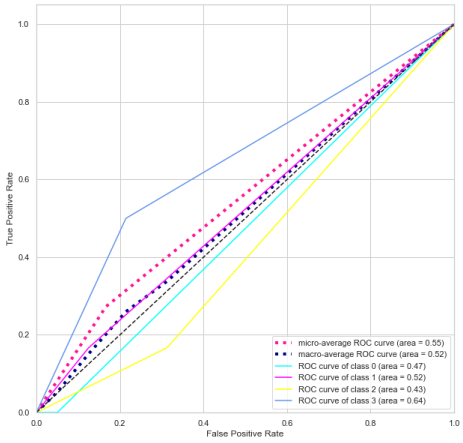


Figure 17: The average model accuracy after 10-fold cross validation.

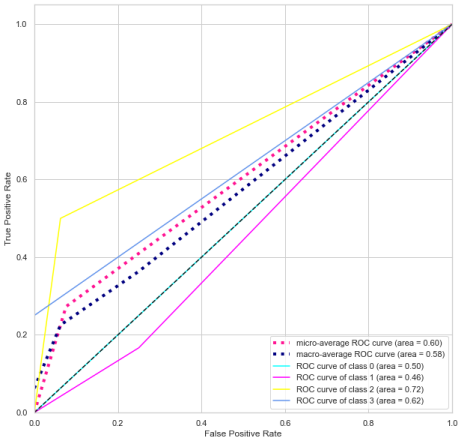
Looking at the cross validation results it seems that the "*Naive Bayes*" Gaussian Classifier is the most accurate out of all the classifiers with the least amount of variance between each k-fold validation. It seems that the "*Decision Tree*" Classifier is the worst performing one of them all with the highest variance between each k-fold validation.



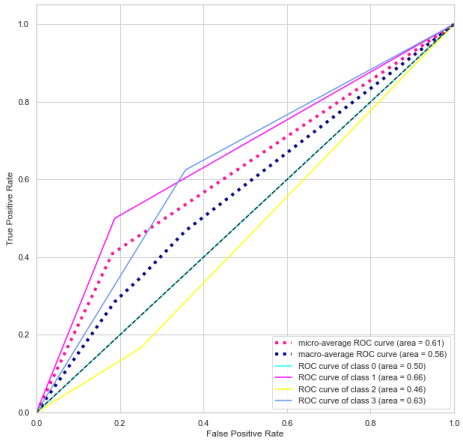
(a) Naive Bayes (Gaussian) Classifier



(b) KNN Classifier



(c) Bagging Classifier



(d) Decision Tree Classifier

Figure 18: Put your caption here

In order to evaluate each one of the model's performance more thoroughly we must observe the ROC curve for it (see figure 18). It seems that all three "*Naive Bayes*", "*KNN*" and "*Bagging*" Classifiers, performed really badly in predicting class 3, this is also reflected in the AUC value for that class. In contrast the "*Bagging*" Classifier, seems to have a lot better performance at predicting class 3, although it falls short in predicting class 0, which is as should be expected, given that class 0 only has 14 data points before the train, test split. Even after the stratified split for each class and the 10-fold cross validation though, it seems that the "*Bagging*" Classifier needs more data

to improve in accuracy.

They all seem to generalise relatively well, when looking at their average AUC, even if their performance is not great.

6.1.2 Genre and Popularity

The first function executed is a number of small SOM models (size=50 and iterations=1000) with different values for the parameters learning rate (0.1, 0.01. 0.001), sigma (5, 10, 15) and the neighborhood function (gaussian, mexican hat, bubble, triangle), in order to find good parameters for this model.

In the run used for this report, the highest accuracy score of around 0.53 was achieved by the combination of learning rate 0.01, sigma 15 and the gaussian neighborhood function. An accuracy of 0.53 is very bad, so for the next run the size of the model and the training epochs got significantly higher (size=150 and iterations=5000). The accuracy of this model is even worse with a disappointing 0.43255.

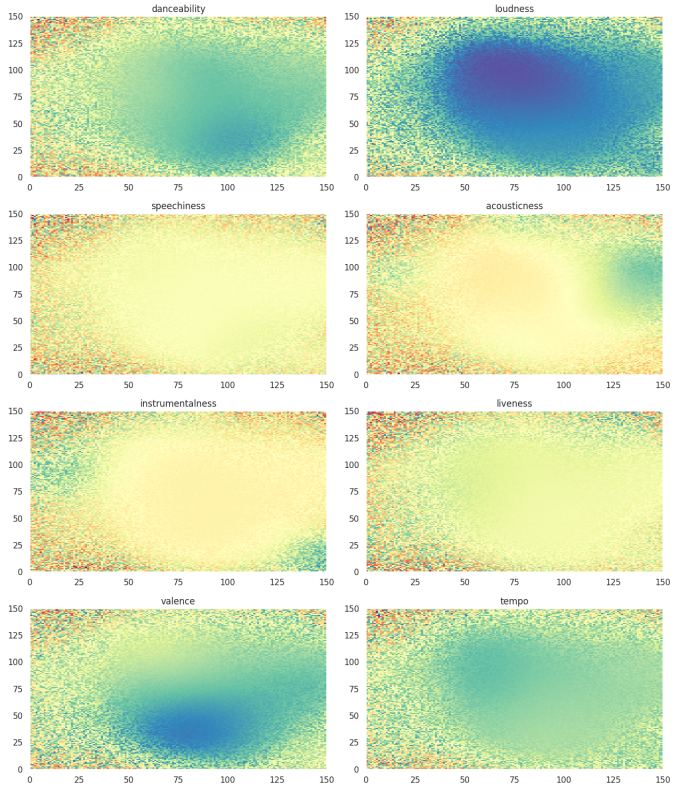


Figure 19: SOMs for the spotify features

This visualization shows quite obvious that the algorithm is struggling to find a number of categories that is even close to the numbers of genres passed to the model. Even increasing the size and duration of the model a lot (size=500 and iterations=200000) does not bring the hoped results and delivers even worse. A 10-fold cross validation with the SOM classification brings the definite proof of the under performance of this model, with a disappointing average accuracy score of around 0.46.

Those results got compared to a Multi Class Decision Tree Classifier, deployed with the "*sklearn*" library in python. Since the default Classifier did not perform any better that the SOM predecessor, pruning for the decision tree was implemented. A number of experiments executed for different alpha values were executed to get an overview of the performance.

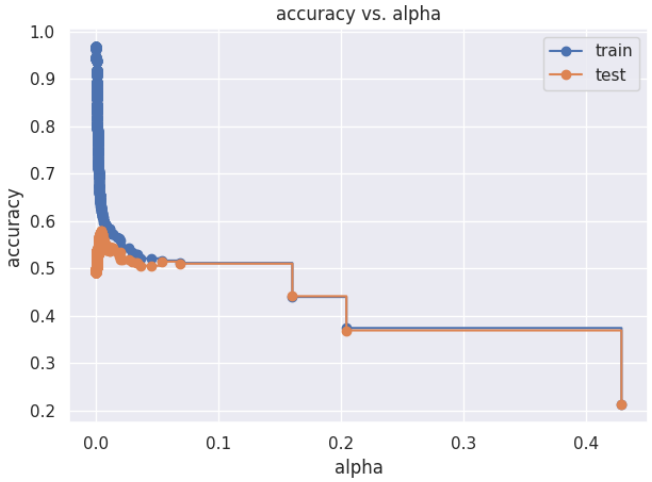


Figure 20: accuracy vs alpha - spotify genre hypothesis

Finally the classical ROC curve for the multi-class Decision Tree Classification shows that also with this type of algorithm, the results are far away from the hypothesis.

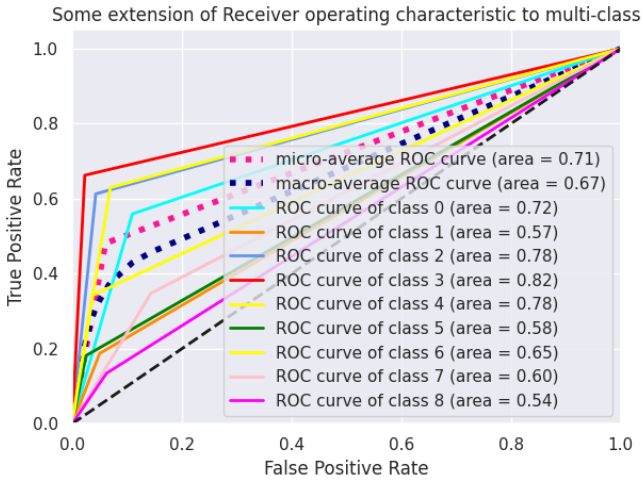


Figure 21: ROC - spotify genre hypothesis

To explore the dependencies of the features to the "*popularity*", the same models were executed to classify the popularity. The 10-fold cross validation of the SOM in this case performs even worse with an average accuracy of only around 0.43. This can also be seen in the performance of the Decision Tree Classifier for different alphas and the ROC curve.

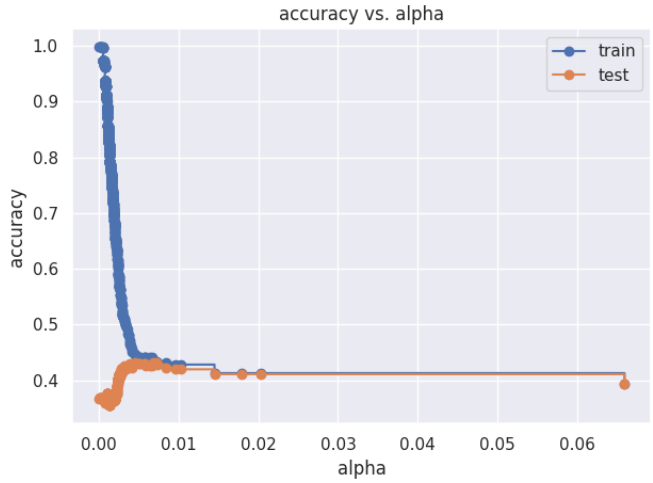


Figure 22: accuracy vs alpha - spotify popularity hypothesis

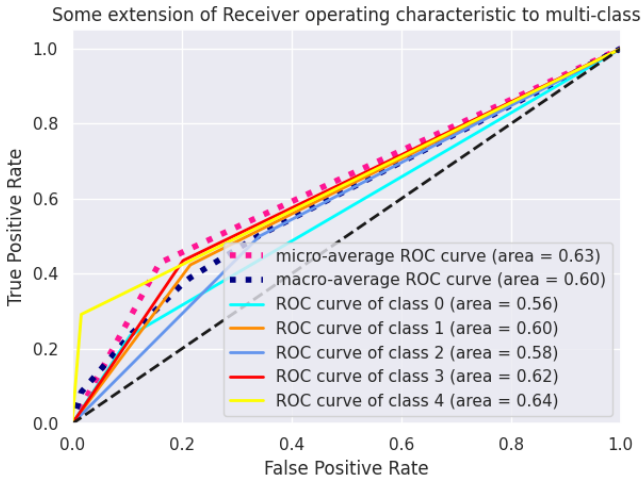


Figure 23: ROC - spotify popularity hypothesis

Overall the performance of all models is really bad and nowhere close to prove one of the hypothesis.

6.2 Beijing Multi-Site Air-Quality Data Data Set

6.2.1 Level of Pollutants Seasonality Hypothesis

As stated in the introduction of the hypothesis, the performance metrics evaluate the density and separation of the clusters. The metrics meeting this requirement are the Silhouette Coefficient and the Calinski-Harabasz index, both indicating dense and well separated clusters with higher scores. Figure 24 shows box-plots for Silhouette Coefficient and Calinski-Harabasz index scores of the hyper-parameter tuning process.

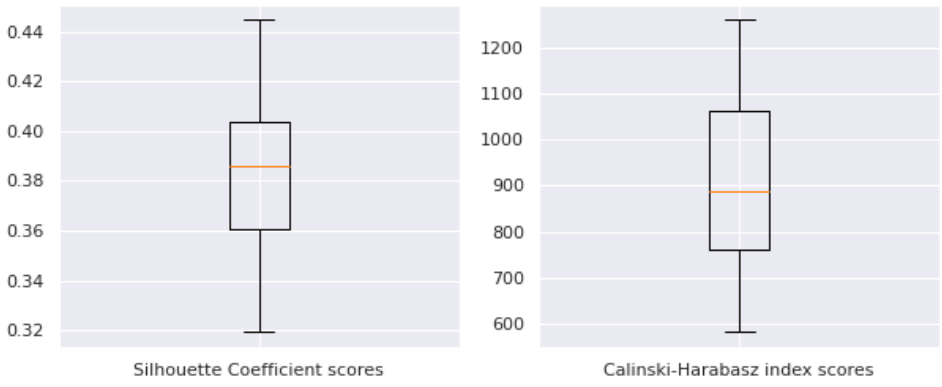


Figure 24: Hyper-parameter Tuning Box-plot Scores

Model	# Features	# Clusters	Inertia	Silhouette	Calinski
9	4	3	2141.067039	0.444535	1260.791245
1	3	3	1722.965108	0.429666	1125.482060
16	5	2	3946.138412	0.418789	1241.866935
17	5	3	2819.511190	0.418721	1159.748128
10	4	4	1729.873183	0.418431	1155.052680

Table 4: Top-5 models by Silhouette Coefficient and Calinski-Harabasz index. Descendant sorted by Silhouette Coefficient

Figure 25 shows the clusters in the features space for Model 1 with 3 clusters and 3 features: ["*NO2*", "*TEMP*", "*WSPM*"]. Model 1 scores are presented in Table 4, being the second best performing model. It can be seen that the clusters are well separated and dense.

Figure 25: Model 1 clusters by NO2, TEMP and WSPM features

Figure 26 shows the results from Model 9 (features: ["*NO2*", "*TEMP*", "*DEWP*", "*WSPM*"], clusters: 3), whose scores are showed in Table 4 becoming the best performing model and confirming the hypothesis. It can be seen that the clusters repeat over the period of time concluding in seasonality.

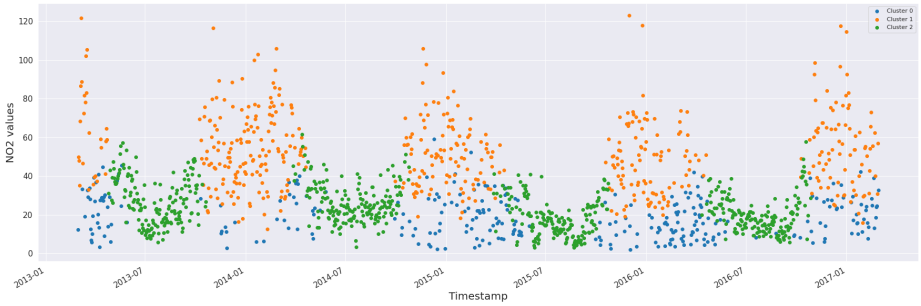


Figure 26: Model 9 - Clusters over the time period

6.2.2 Estimating the AQI values without having access to all the chemical values

I began the experiment by running both KNN and Decision Tree using the default parameters. It was possible to observe that both models took a very long time to finish training, but the accuracy scores were very high around 90%. This could have been the result of using highly correlated features such as PM10 and PM10_24HRAvg which are used to find the AQI values.

In an ideal world, given more data and more computation power we could have removed the highly correlated features and experimented with these models to verify if our hypothesis is still valid.

The alternative solution that I came up with was to use PCA to perform dimensionality reduction on the features used, this not only allowed the model to be trained more quickly but also removed some of the correlation present in the data. The initial number of features considered for the models were 8, using PCA I was able to reduce to 4.

The KNN and Decision Tree models were re-tested using the new features, and it was possible to notice that the models took significantly less time to train and the correlation between the features was reduced.

KNN Hyper-parameter Tuning:

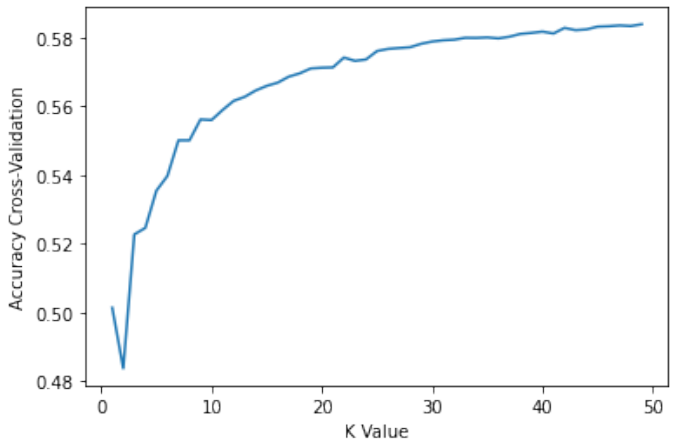


Figure 27: KNN Parameter Tuning

The figure above shows the experiment that was carried out to find the optimal n-neighbours for our KNN. For each value in the range of 1 to 50, we performed 10-fold cross-validation and plotted the results. It is possible to observe from the graph that the accuracy is not improving much after reaching n-neighbours of 30.

Choosing a higher value for the n-neighbour will result in more smother boundaries, but this could introduce bias in our model and increase the computational time. On the other hand, determining a smaller value for n-neighbour will produce more noise [8].

KNN Evaluation:

Using the information gained from the hyper-parameter turning with n-neighbours of 20, I created a model using 80-20 split for training and testing. The image below shows how well the model performed on the test data.

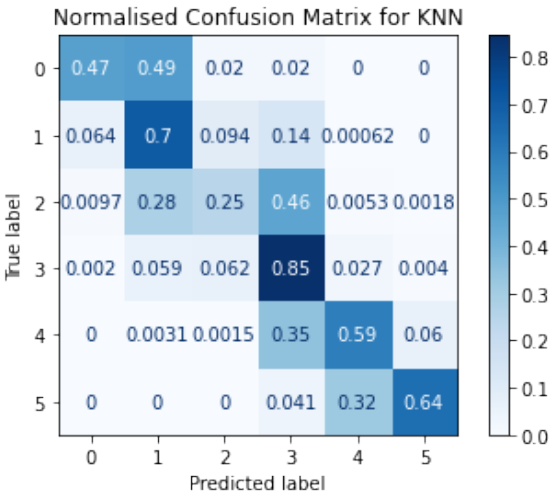


Figure 28: KNN confusion Matrix

Figure 27 shows us the confusion matrix that we obtain from training and testing our KNN model. The model learned to correctly classify Class 1 and 3 with decent accuracy. Whereas it did poorly for Class 0, 4 and the worst one 2.

	precision	recall	f1-score	support
0	0.61	0.47	0.54	403
1	0.62	0.70	0.66	1614
2	0.45	0.25	0.32	1134
3	0.72	0.85	0.78	2985
4	0.71	0.59	0.64	649
5	0.73	0.64	0.68	221
accuracy			0.66	7006
macro avg	0.64	0.58	0.60	7006
weighted avg	0.65	0.66	0.65	7006

Figure 29: KNN Classification Report

Figure 28, on the other hand, shows us the total accuracy score and additional useful information such as f1-score and recall for this model. We managed to get an overall accuracy score of 66%. Considering we only ran this model on four features, it is possible to explain the results obtained. If we had more resources and data, we could have run a more complex model to predict the AQI value.

Decision Tree Hyper-parameter Tuning:

Decision Trees are an advantageous model and can be used to perform classification. There are multiple ways to optimise and prevent a Decision Tree form overfitting. Some

of the hyper-parameters that can be tuned include max_depth, criterion and ccp_alpha. For this experiment, due to computational power, I decided to optimise only ccp_alpha and criterion type.

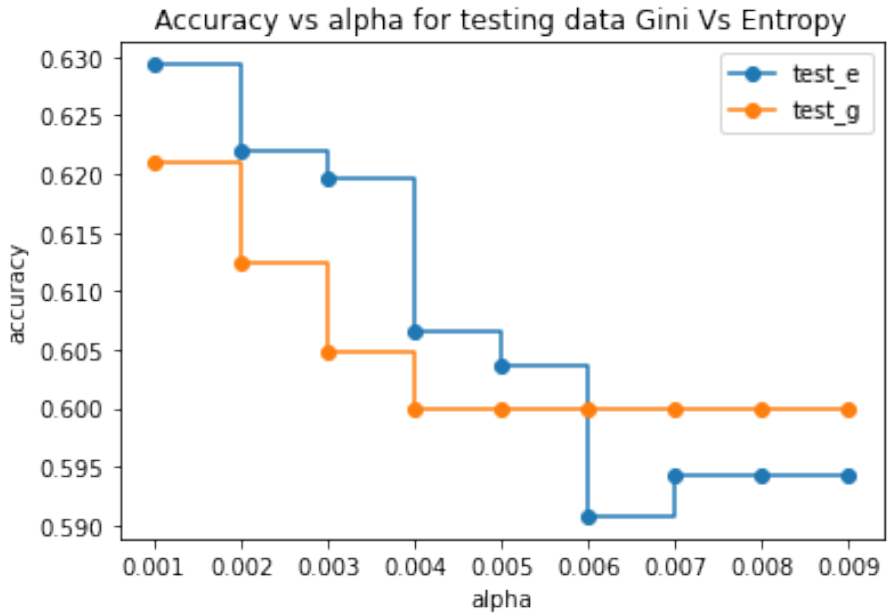


Figure 30: Accuracy of Decision Tree under different Alpha values and Solver type

Figure 29 can be used to find the optimal ccp-alpha value and the right criterion to run our Decision Tree. The range for the alpha values was chosen between 0.001 and 0.009 after experimenting with few values outside this range. It is possible to conclude that the highest accuracy is obtained using alpha values between 0.001 and 0.002 and using Entropy gain ad the criterion type.

Decision Tree Evaluation:

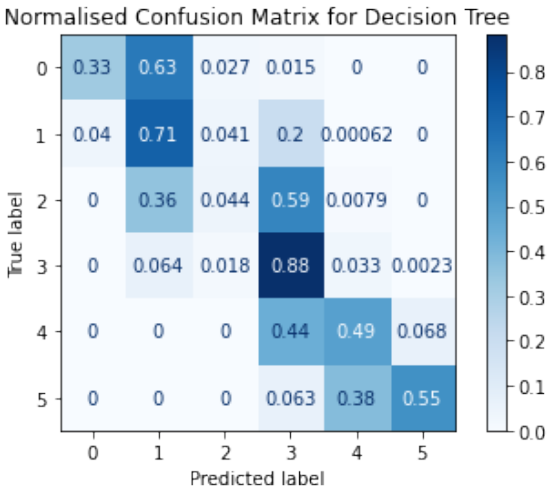


Figure 31: Decision Tree Confusion Matrix

This confusion matrix is very similar to the one in Figure 27. Both The model produced similar results. In Decision Tree the highest class correctly predicted is again 3 and the worst one is 2.

	precision	recall	f1-score	support
0	0.67	0.33	0.44	403
1	0.58	0.71	0.64	1614
2	0.27	0.04	0.08	1134
3	0.67	0.88	0.76	2985
4	0.62	0.49	0.55	649
5	0.71	0.55	0.62	221
accuracy			0.63	7006
macro avg	0.59	0.50	0.51	7006
weighted avg	0.58	0.63	0.58	7006

Figure 32: Decision Tree Classification Report

The accuracy score is also very similar to the previous model and same goes for the other scores. I ran a final experiment just to get an idea of how a different classifier would perform. I ran a Random Forest with similar parameters as a Decision Tree and used 100 n_estimators. The results for this can be seen below.

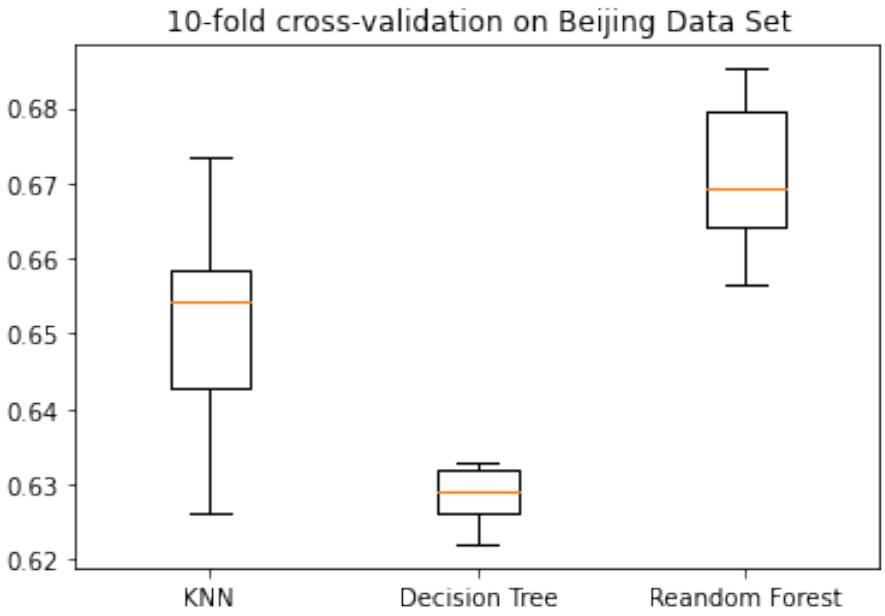


Figure 33: Correlation Matrix

The plot shows us that the Random Forest outperformed the other two models in accuracy values.

6.2.3 Warning prediction for dangerous chemical levels

The VAR predictions seemed to calculate quite closely to the the truth and follow the general trend correctly. Looking at the results we can see that fine tuning how far the VAR predicts could result in more optimal calculations.

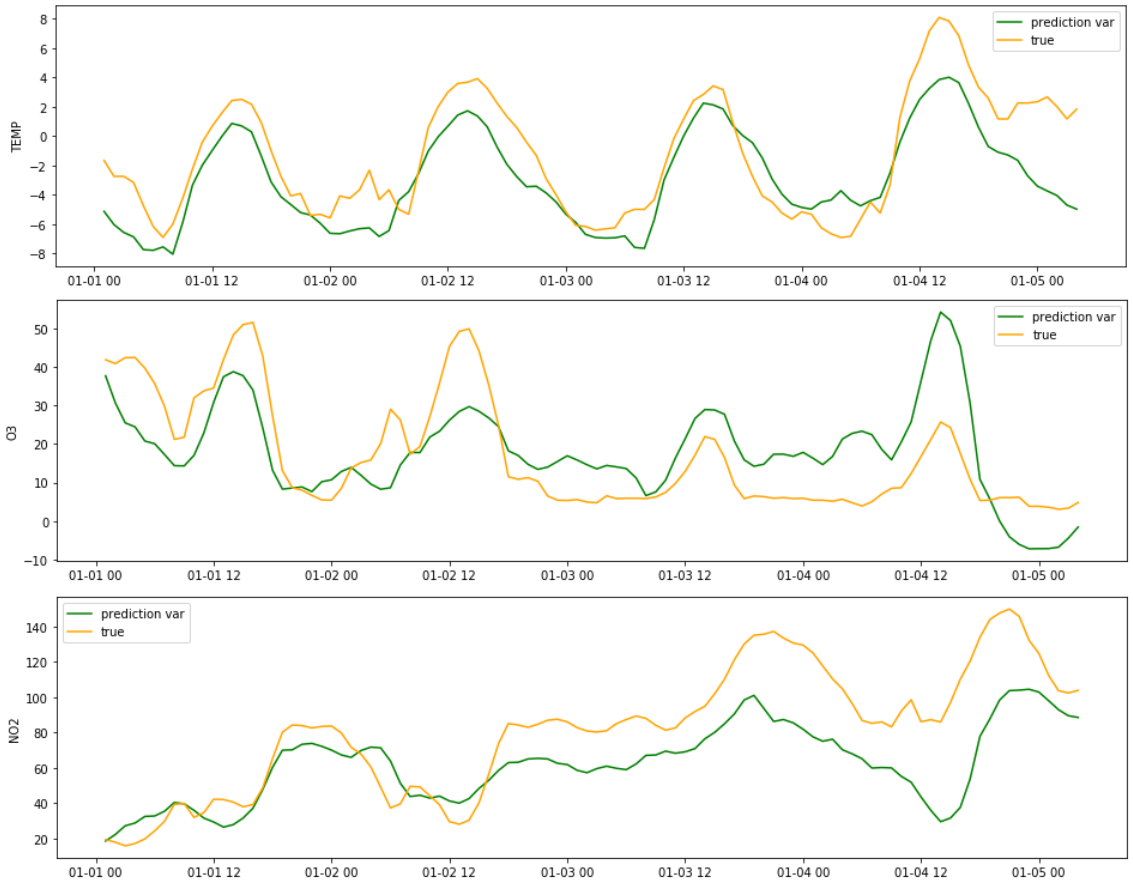


Figure 34: VAR prediction vs Truth

The results are compared looking at the regular LSTM and the VAR combined LSTM models and comparing the errors using MAE. As we can see with these results the VAR LSTM model only slightly performs better. Tweaking of the parameters could produce a better result.

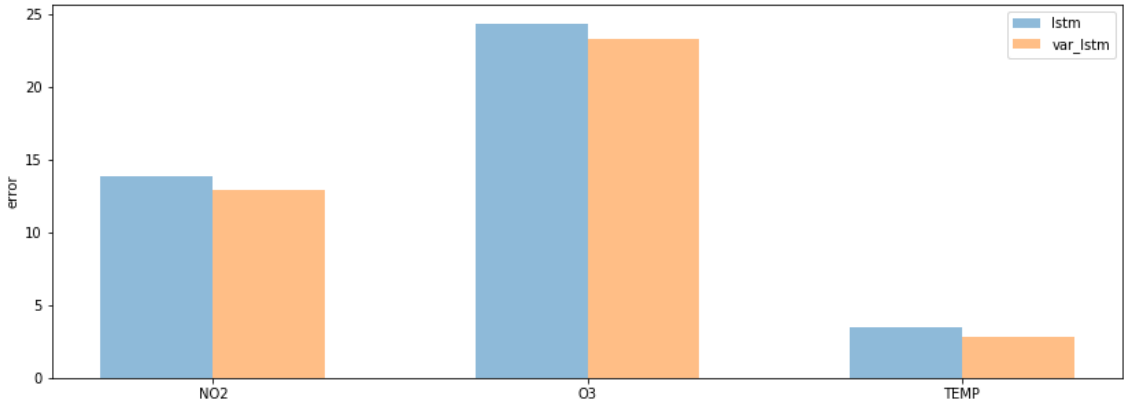


Figure 35: MSE error

The next graph is of the correlation of the truth and the prediction, using it to show if the prediction is not a repeat of present values. Ideally we want to try and minimise the correlation and at around 80 Percent for NO2 and O3 values they produce results that wouldn't be considered as present values repeated.

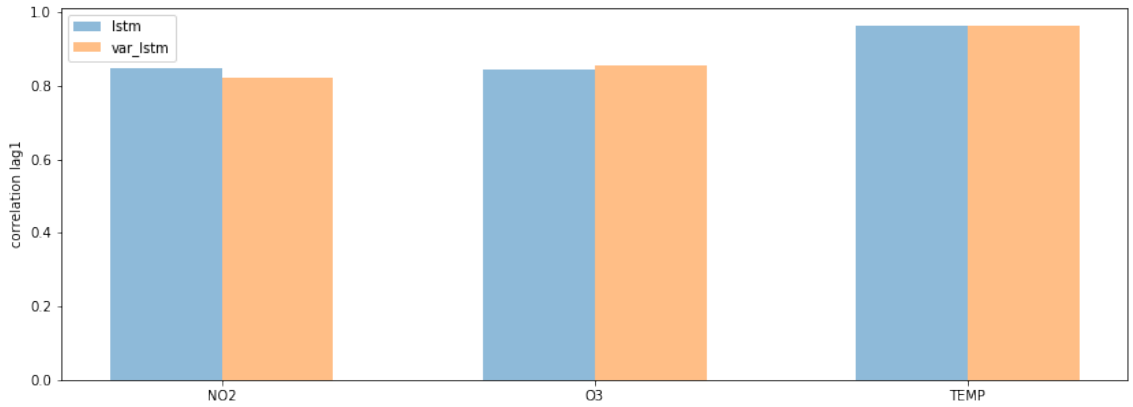


Figure 36: Correlation of truth and prediction

7 Conclusion and Further Work

7.1 Spotify API Data Set

7.1.1 Funky Hypothesis

In conclusion, this hypothesis has been a very surface exploration of what could possibly be a relatively good way in classifying whether or not a song is dance-able and to what degree. The models seemed to generalize well under k-fold cross validation, although did not have the best performance in fitting the solution space with the features selected.

There are multiple things that could be done to improve the predictive performance of any of the models used. One of them could be to download spotify's recommended playlists for Funk music, concatenate them and use them as a dataset. The idea would be that with more data, the Classifiers would be able to find any underlying patterns in the music and other features would be more strongly correlated to "*danceability*". A short-coming/bias with the approach described above is that it is assumed that the spotify playlists have 0 bias and have made sure to include only funk songs in these playlists (which is not the case).

One more way that classification could have been made easier, would be to use a deep learning neural network with an encoder-decoder system to populate a feature space with the features of the current dataset. Then use one of my own playlists with funk music to predict its danceability. In this scenario, the short-coming/bias of the neural network would come from the feature space that it has mapped initially, which is the current dataset, which is again not very big.

Last but not least, it should be noted, that even the current dataset used in the exploration can have user bias via the user that create the list [2]. One way around that would be to conduct some sort of survey or ask a few funk musicians to select which songs should be included in a funk-playlist of all time. Since Funk as a genre utilizes elements of jazz and disco, it is quite tough to pinpoint which songs are "purely-funk". Via asking multiple musicians, or simply multiple people that listen to such music, I could have taken the intersection of their playlists as the dataset for this investigation. It would still carry bias, but it would be a more scientific way of removing as much of it as possible.

7.1.2 Genre and Popularity

Although this exploration of the dependence between track features and the music genre was very low level in comparison to the vast amount of tracks in the world and the depths of machine learning, the results still give an indication that the features extracted by the sound analysis of spotify alone do not give a clear indicator for the genre.

With those results we still have to consider the probably imbalance of the genres inside the data. Also the fact that SOMs do perform better with low dimensional data can be a reason for the bad performance. Also the data mined by the algorithm can be biased. It is not known how spotify generates those features per track and it is also, not known how spotify defines the music genres. What we can say for sure is, that the establishment of genres does not happen with just the audio features per track.

But the fact that also the Multi Class Decision Tree Algorithm, which often is used together with the Iris data set for tutorials and demonstrations(labs), that has similar properties to the spotify data set, does not perform well at all, gives reason to conclude that this hypothesis cannot be proofed with this data set.

The fact that the popularity seemingly cannot be classified by the track features does not necessarily come surprising. The popularity of a track depends on a lot of personal

opinions and feelings of the people that listen to the tracks. Those are parameters that very often cannot be explained or predicted by data due to the spontaneity of the human nature. The fact that a commonly used categorisation like music genre cannot be predicted by track features comes more as a surprise.

7.2 Beijing Multi-Site Air-Quality Data Data Set

7.2.1 Level of Pollutants Seasonality Hypothesis

In conclusion, this hypothesis has been satisfactorily demonstrated by applying the k-means clustering algorithm resulting in groups of data that can be interpreted as a seasonality over the data time period. As next step, this data mining process could be used to specify the seasonality in a Bayesian Structural Time Series model to forecast the time-series of a pollutant measures in the near future.

7.2.2 Estimating the AQI values without having access to all the chemical values

In conclusion, this hypothesis was hard to evaluate as we faced many challenges from the beginning. The AQI values and the various chemical values are heavily correlated, and we didn't have any other relevant features that can be used for the model building. However, with PCA and hyper-parameter tuning, I managed to produce reasonable models that can predict the Air Quality Index. Form the model and evaluation section; we can assume that the data is not very balanced. KNN and Decision Tree always managed to identify class 3 correctly and predicted class 2 poorly. This could be due to a lack of samples in class 2.

In future, it would be interesting to test this data with other classifiers, such as Random Forest. Figure 32 shows that RandomForest performed better than KNN and Decision Tree, but due to a lack of computational power and time, I couldn't test this model further. Another model that can be tested is a Neural Network; it would be interesting to see if this model can identify any underlying correlation between the data.

7.2.3 Warning prediction for dangerous chemical levels

In conclusion the model is able to make a prediction to a certain degree of accuracy, but there are many more features which need to be explored and it is obvious to see that there can be a lot more optimisation in terms of the parameters set. With the time given data exploration was performed looking into 3 of the features. A next step would be to

try and add in more features to find which relate to the prediction model and increase its accuracy.

8 Contributions

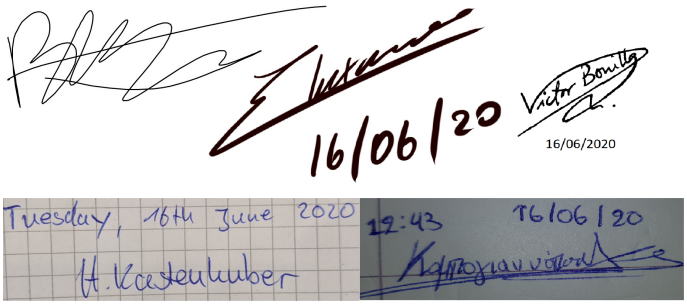


Figure 37: Signatures Combined.

8.1 Victor Bonilla Pardo

My contributions are:

- Active participation in all the team meetings.
- Participation in the search of data sets. I presented three data sets. One of them was the "Beijing Multi-Site Air-Quality Data Data Set".
- Research of different models for forecasting time-series, focusing on Bayesian Structural Time Series models (BSTS).
- Development of a naive BSTS model with Tensorflow Probability. However, it was discarded due to the big amount of required work to get interesting project results given the time available before the deadline.
- Definition of the hypothesis "Level of Pollutants Seasonality Hypothesis".
- Implementation of the whole Data Mining pipeline (Data Pre-processing, Modelling, Evaluation...) for the hypothesis "Level of Pollutants Seasonality Hypothesis".
- Contribute to define the structure of the report.

- Contribute to write the introduction of the report and the description of the "Beijing Multi-Site Air-Quality Data Data Set".
- Writing of every section regarding the hypothesis "Level of Pollutants Seasonality Hypothesis".

8.2 Hannes Kastenhuber

My contributions are:

- Active participation in all the team meetings.
- Participation in the search of data sets. Presentation of three data sets.
- Implementation of the Spotify data mining project and the functions used to mine data for the Spotify api hypotheses
- Research into various classification models
- implementation of two different kind of classification models for comparison
- implementation of regression algorithms with SOMs and Decision Trees to predict popularity, but not included into report due to unsatisfying results and limitations of the coursework.
- Definition of the hypotheses "Genre and Popularity".
- Implementation of the whole Data Mining pipeline (Data-mining, Data Pre-processing, Modelling, Evaluation...) for the hypothesis "Genre and Popularity".
- Contribute to define the structure of the report.
- Writing of every section regarding the hypothesis "Genre and Popularity".

8.3 Konstantinos Kompogiannopoulos

My contributions are:

- Active participation in all the team meetings.
- Participation in the search of data sets. Presentation of three data sets.
- Research into various classification models.

- Implementation of two different kind of classification models for comparison (also a k-fold batch of them).
- Implementation of Support Vector Machines and Decision Trees to predict danceability.
- Definition of the hypotheses "Funky Hypothesis".
- Implementation of the whole Data Mining pipeline (Data-mining, Data Pre-processing, Modelling, Evaluation...) for the hypothesis "Funky Hypothesis".
- Contributed into defining the structure of the report.
- Writing of every section regarding the hypothesis "Funky Hypothesis".

8.4 Luxman Elangeswaralingam

My contributions are:

- Active participation in all the team meetings.
- Participation in the search of data sets. Presentation of three data sets.
- Research into various classification models.
- Implementation of two different kind of classification models for comparison (also a k-fold batch of them).
- Implementation of KNN and Decision Tree and Briefly looked at Random Forest
- Definition of the hypotheses "Estimating the AQI values without having access to all the chemical values".
- Implementation of the whole Data Mining pipeline (Data-mining, Data Pre-processing, Modelling, Evaluation...) for the hypothesis "Estimating the AQI values without having access to all the chemical values".
- Contributed into defining the structure of the report.
- Writing of every section regarding the hypothesis "Estimating the AQI values without having access to all the chemical values".

8.5 Benjamin Baxter

My contributions are:

- Active participation in all the team meetings.
- Participation in the search of data sets. I presented three data sets.
- Research of different models for forecasting time-series, focusing on Long Short Term Memory Time Series models (LSTM).
- Initially starting with a model only based on LSTM, then removing it and basing the new model from a combination of VAR and LSTM.
- Definition of the hypothesis "Warning prediction for dangerous chemical levels".
- Implementation of the whole Data Mining pipeline (Data Pre-processing, Modelling, Evaluation...) for the hypothesis "Warning prediction for dangerous chemical levels".
- Contribute to define the structure of the report.
- Contribute to writing the description of the "Beijing Multi-Site Air-Quality Data Data Set".
- Writing of every section regarding the hypothesis "Warning prediction for dangerous chemical levels".

References

[1] Various authors. Air quality index. "https://en.wikipedia.org/wiki/Air_quality_index".

[2] Jeff B. 200 greatest funk songs. "https://digitaldreamdoor.com/pages/best_rb-funk.html".

[3] beanieintelligence. spotify_data_mining. "https://github.com/beanie-intelligence/spotify_data_mining", Mar 2020.

[4] Marco Cerliani. Combine lstm and var for multivariate time series forecasting. "https://towardsdatascience.com/combine-lstm-and-var-for-multivariate-time-series-forecasting-abdcb3c7939b".

[5] Spotify for Developers. Get audio features for a track. "developer.spotify.com/documentation/web-api/reference/tracks/get-audio-features/".

[6] Aarav Maheshwari. What is a multi-class svm method? "https://www.quora.com/What-is-a-multi-class-SVM-method", 2018.

[7] Scikit. 1.12. multiclass and multilabel algorithms. "scikit-learn.org/stable/modules/multiclass.html.".

[8] Dhilip Subramanian. A simple introduction to k-nearest neighbors algorithm. "https://towardsdatascience.com/a-simple-introduction-to-k-nearest-neighbors-algorithm-b3519ed98e".

[9] Shuyi Zhang, Bin Guo, Anlan Dong, Jing He, Ziping Xu, and Song Xi Chen. What is a multi-class svm method? *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2017.

9 Appendix

9.1 Appendix 1: Funky Hypothesis

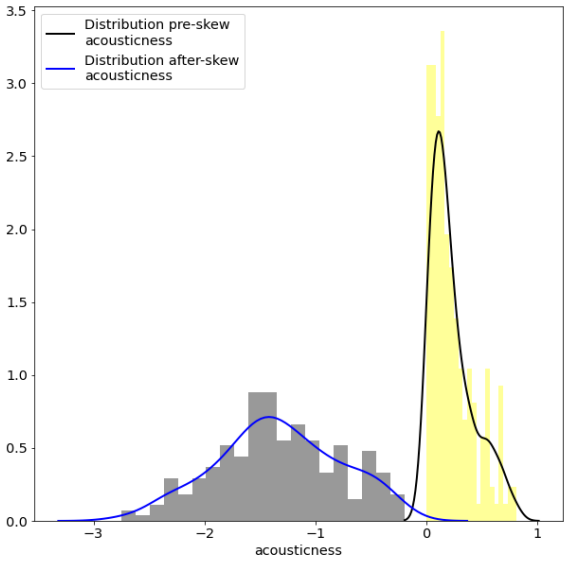


Figure 38: Funk-distribution-hist-acousticness-before-after-skew

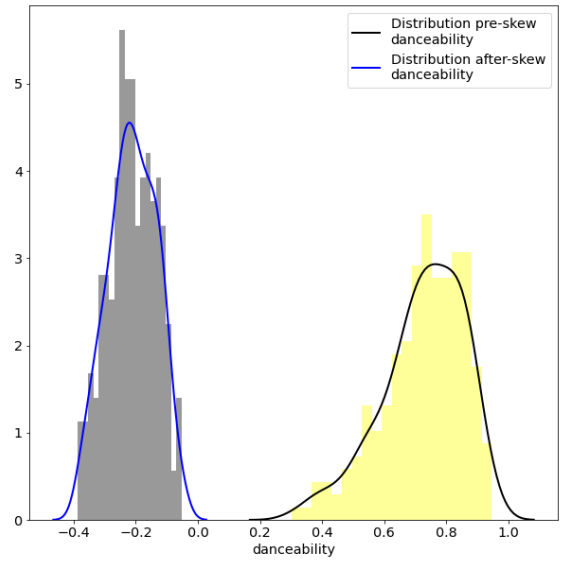


Figure 39: Funk-distribution-hist-danceability-before-after-skew

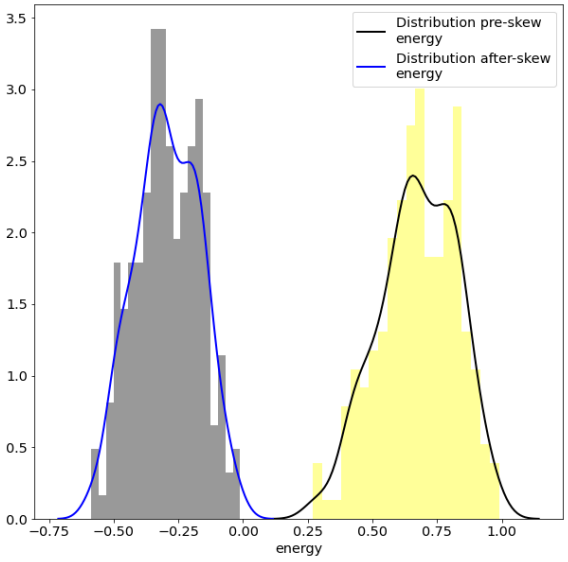


Figure 40: Funk-distribution-hist-energy-before-after-skew

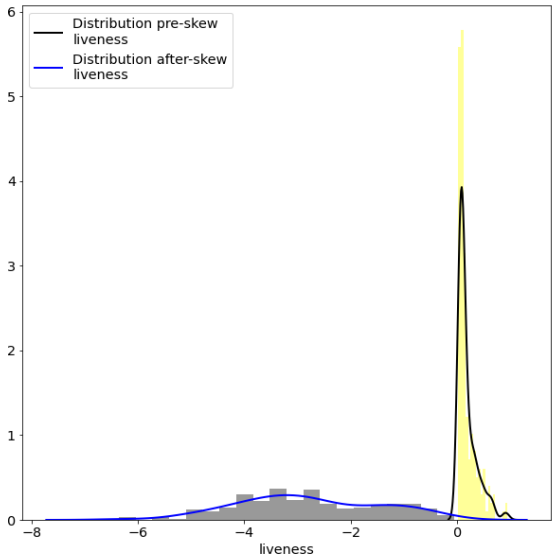


Figure 41: Funk-distribution-hist-liveness-before-after-skew

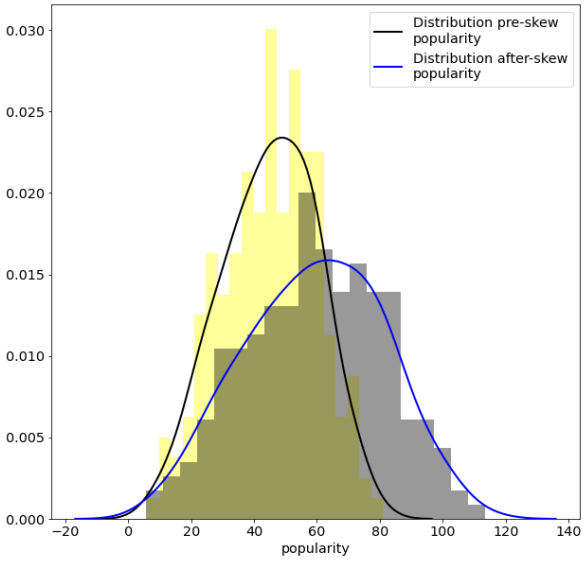


Figure 42: Funk-distribution-hist-popularity-before-after-skew

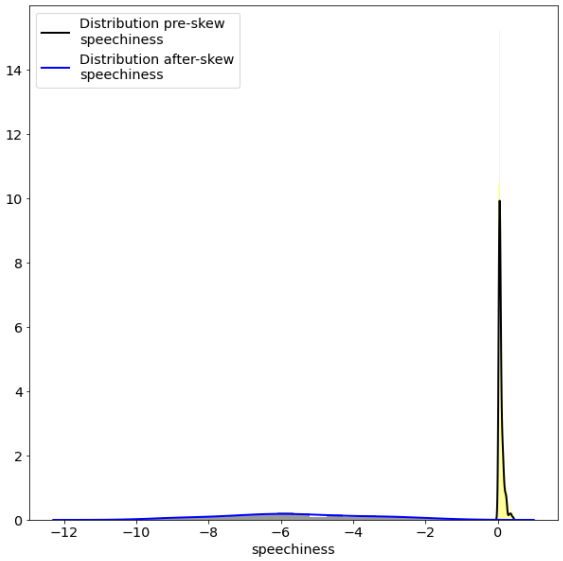


Figure 43: Funk-distribution-hist-speechiness-before-after-skew

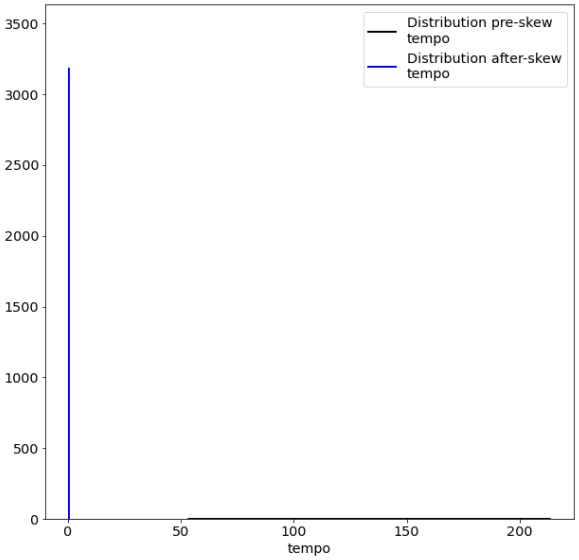


Figure 44: Funk-distribution-hist-tempo-before-after-skew

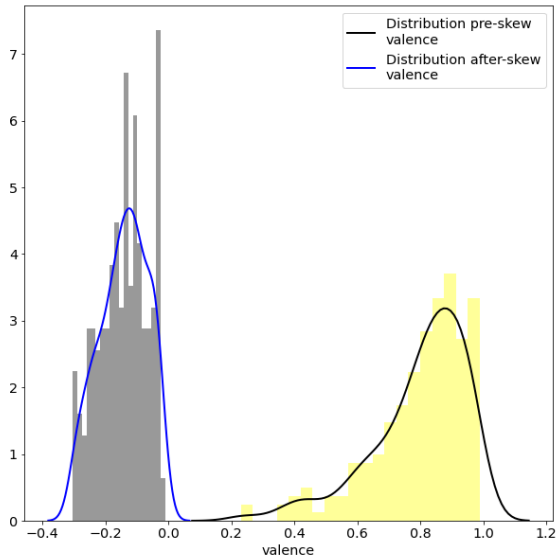


Figure 45: Funk-distribution-hist-valence-before-after-skew