KAUNAS TECHNOLOGYS UNIVERSITY
INFORMATICS FACULTY

Module P160B114 „Machine Learning Methods“

Laboratory work report
First laboratory work

Instructor
Doc. Iešmantas Tomas

Student
Rokas Puzonas

KAUNAS, 2023

Table of Contents

Task 1, linear regression

1.1.

The bike renting data has 14 features. The training dataset has 7200 observations, and the test dataset has 1560 observations.

1.2.

Plotting visibility against rented bike count, we can see that there is a correlation between the two features, shown “Figure 1”. To better explain this correlation, we can split the plot into 3 parts: low visibility, average visibility, high visibility.

- Low visibility is from 0m to ~500m, where we can see rented bike count linearly increasing with visibility, this is because the lower the visibility, the less you can see in front of you, so it makes it more dangerous to ride a bike.
- Average visibility is from ~500m to ~1750m, where the rented bike count stays roughly the same across the whole range. This is where you can see far enough to avoid obstacles, and where the increasing visibility does not matter as much, because even at 1500m visibility, you will still only be worried about what is about 500m in front of you.
- High visibility is from ~1750m to 2000m, where we can see a sudden increase in rented bike count. This can be explained by high visibility only being achievable during clear and sunny days, where people who don’t use a bike daily want to enjoy the weather while riding a bike.

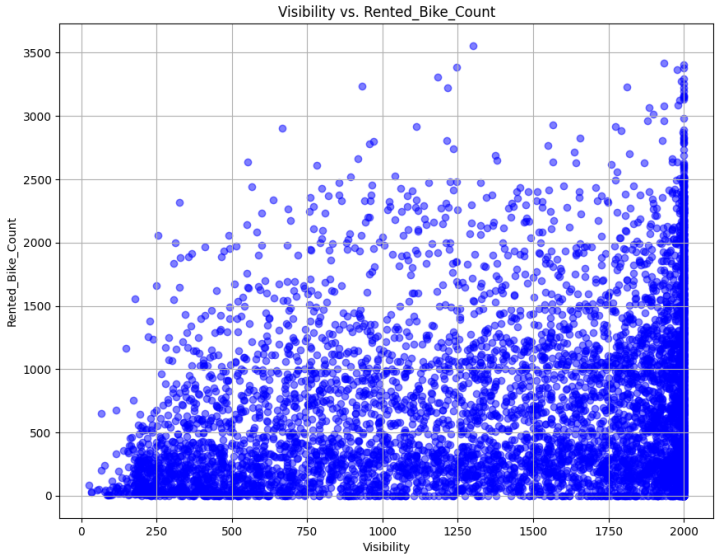


Figure 1. Visibility vs rented bike count

Plotting rainfall against rented bike count we can see that there is a strong correlation between the two shown in „Figure 2“. The reason for this is that it dangerous to ride a bike when

it is raining because of slipping and bad weather. The only who use a bike when it is raining would be those that must use daily for getting to their job or those that need to go somewhere further where there is no public transport.

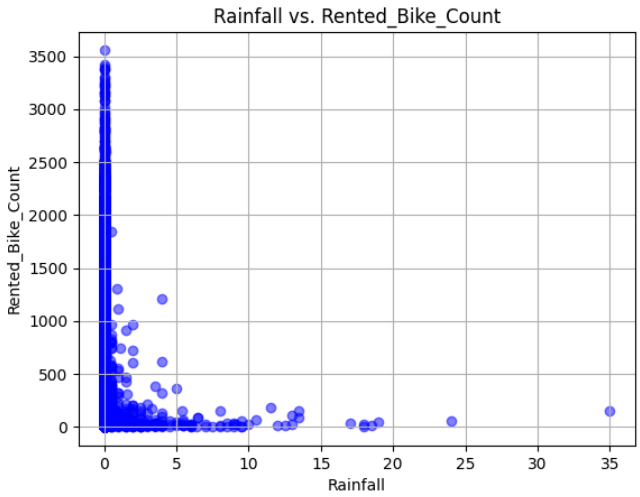


Figure 2. Rainfall vs rented bike count

1.3.

This is the correlation matrix represented as a heatmap, shown by “Figure 3”. The pair with the highest correlation is temperature vs dew point temperature.

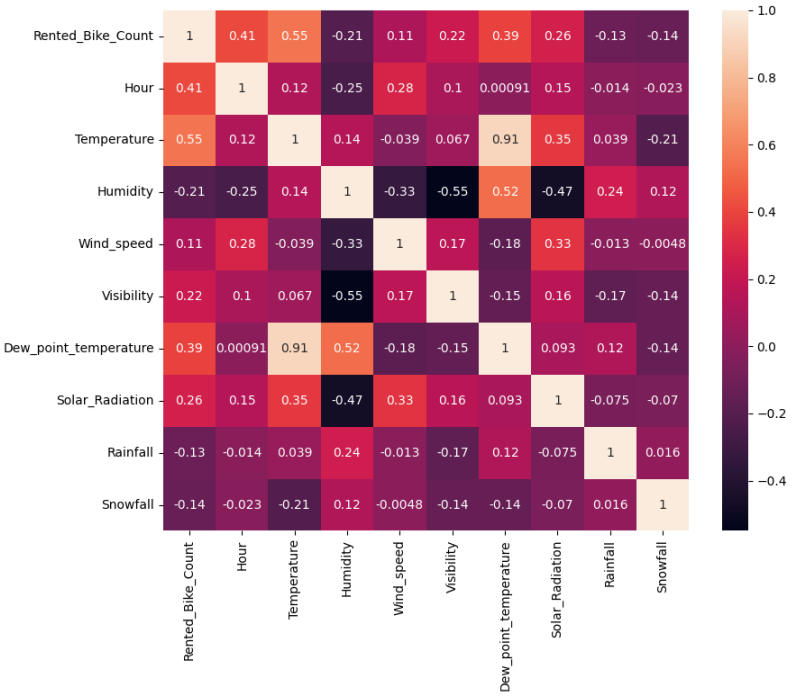


Figure 3. Heatmap of the correlaction between features

Looking at the plot of temperature vs dew point temperature shown in “Figure 4”, we can see that a very linear relationship exists between the two features. Using some domain knowledge,

that dew point temperature can be derived from temperature, it would be safe to remove dew point temperature from the dataset. Because in general it is a good idea to use the minimal number of features needed when predicting another feature. Dew point temperature does not provide any additional information that we couldn't get from temperature.

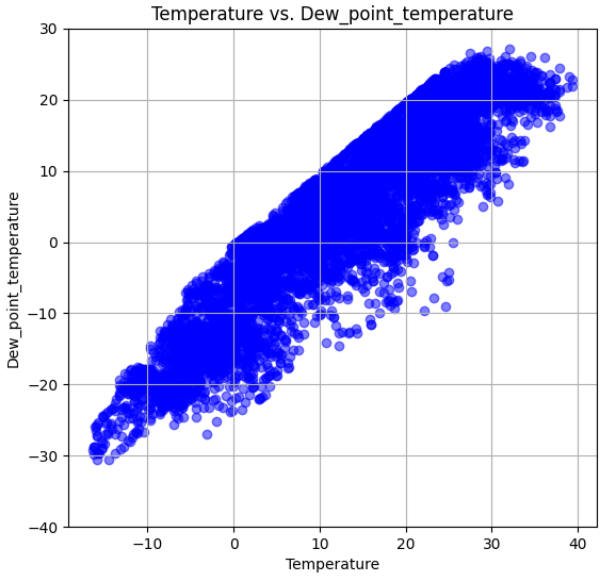


Figure 4. Plot of temperature vs dew point temperature

1.4.

There are 2 problems that we can see from the plot shown in “Figure 5”.

1. Over predicting in the range from 0 to ~1000. We can tell this by looking at the red line which is f(x)=x, ideally our predictions should be around this line. But they are on average above it, which means on average every prediction is higher than the real amount.
2. Under prediction from ~1000. We can tell this because the predictions roughly “flatten” out at ~1250 when it should keep increasing linearly.

Calculating R^2 we get ~0.512624, and the RMSE is ~428.80.

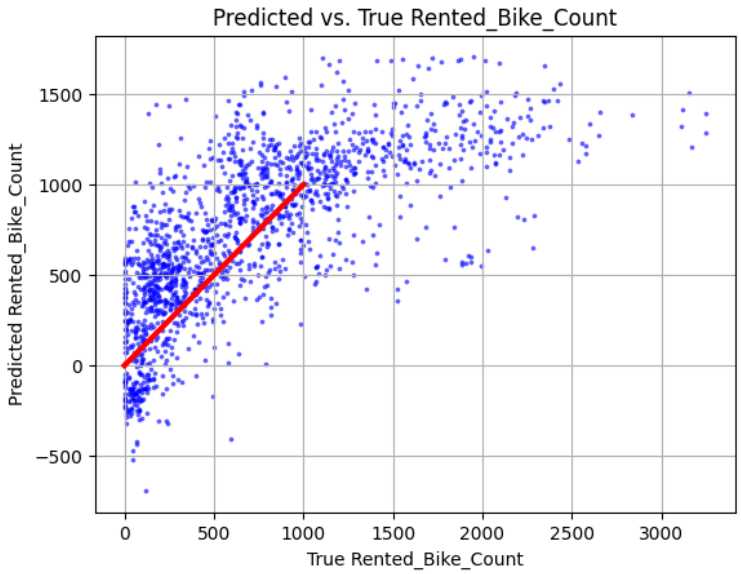


Figure 5. Plot of predicted vs true bike count

1.5.

Applying the suggested transformation to the rented bike count feature greatly improves accuracy and RMSE. From the plot shown in “Figure 6”, we can see that the point are closer to the red f(x)=x line. And the amount over and under predicting decreases.

Calculating R^2 we get ~0.7765751, and the RMSE is ~0.72.

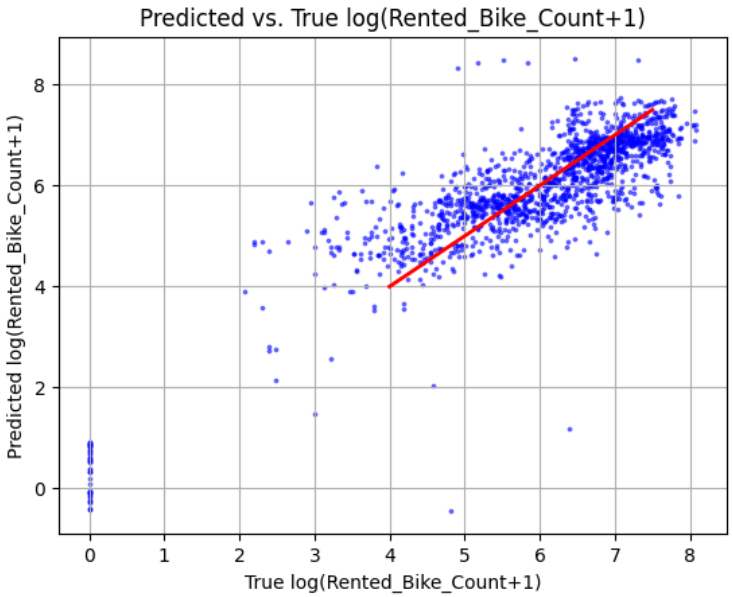


Figure 6. Plot of true vs predicted log(rented bike count + 1)

1.6.

To figure out which interaction between 2 features was the most significant, all combinations of features were tested, for each unique combination and R^2 of $X_1 + X_2$ and R^2 of $X_1 + X_2 + X_1X_2$ was calculated can compared to see were the difference between the two R^2 values increased.

The table only shows the 10 best interactions based on the R^2 difference.

X_1	X_2	R^2 of $X_1 + X_2$	R^2 of $X_1 + X_2 + X_1X_2$	R^2 difference
Humidity	Visibility	0.775819	0.785212	0.009393
Dew point temp.	Rainfall	0.776575	0.781958	0.005383
Temperature	Season spring	0.776575	0.781953	0.005377
Temperature	Rainfall	0.776575	0.781678	0.005103
Temperature	Season summer	0.776575	0.781303	0.004727
Humidity	Rainfall	0.776575	0.779620	0.003045
Humidity	Solar radiation	0.776575	0.779343	0.002768
Temperature	Solar radiation	0.776575	0.779271	0.002696
Humidity	Dew point temp.	0.776575	0.779128	0.002553
Visibility	Solar radiation	0.775901	0.778331	0.002429

1.7.

To figure out what transformation is the best all combinations were tried, by trying to apply log (X), $\sqrt{X}$, X^2 to one column at a time where applicable. Then after applying the transformation R^2 is calculated and sorted by it, to see which transformation was the most effective.

Baseline R^2 is 0.776575, this is where no transformations were applied.

X	Transformation	R^2	R^2 compared to baseline
Rainfall	$\sqrt{X}$	0.820136	0.043561
Rainfall	X^2	0.802315	0.025740
Humidity	X^2	0.800810	0.024235
Humidity	$\sqrt{X}$	0.794849	0.018274
Solar radiation	$\sqrt{X}$	0.781892	0.005317
Hour	$\sqrt{X}$	0.781612	0.005037
Solar radiation	X^2	0.778903	0.002328
Visibility	X^2	0.778226	0.001651
Visibility	$\sqrt{X}$	0.778094	0.001519
Dew point temp.	X^2	0.777979	0.001404

Task 2, logistic regression

2.1.

The Spotify dataset is stored in a CSV format and has 17 columns which store various metrics about each song listened to by a one user. Here is list of explanation for some of the metrics:

- Acousticness, a value from 0 to 1, the higher the value the more confident that the track is acoustic.
- Dancebility, a value from 0 to 1, the higher the value, the more suitable the music is for dance. Various aspects are considered like tempo, rhythm, beat strength and overall regularity.
- Target, a value 0 or 1, 0 means that the song was not liked, 1 that it was liked.
- Energy, a value from 0 to 1, the higher the value the more a song feels energetic, fast paced, loud, noisy.
- Instrumentalness, a value from 0 to 1, the higher the value the less vocal content the music contains.
- Liveness, a value from 0 to 1, the higher the value, the more likely this was recorded with an audience in the background during a live concert.
- Speechiness, a value from 0 to 1, the higher the value, the more likely the song contains speech-like elements.
- Valence, a value from 0 to 1, the higher the value, the more positive, cheerful, euphoric a song sounds.

2.2.

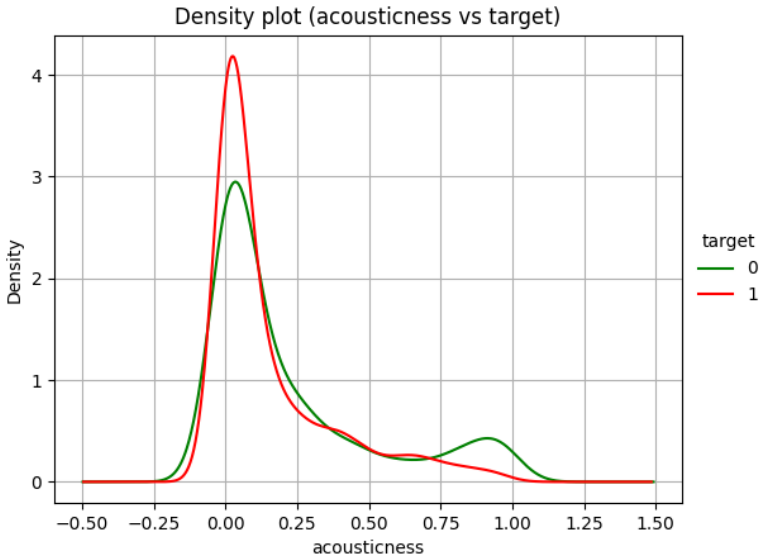


Figure 7. Density plot of acousticness vs target

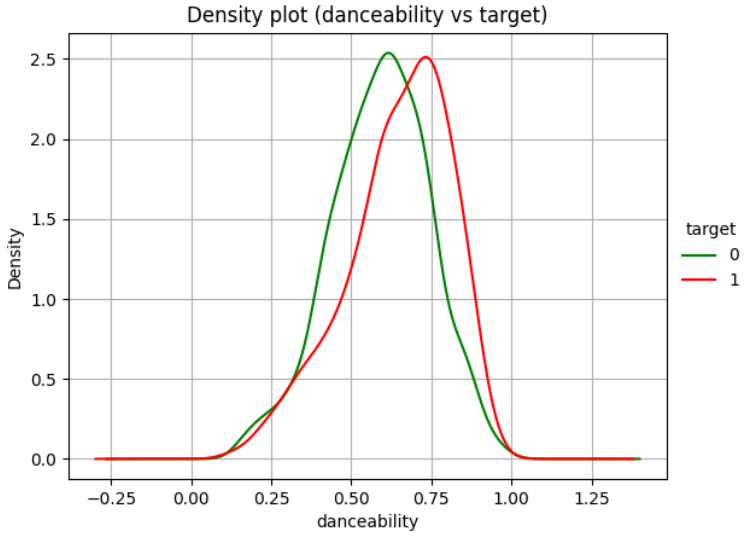


Figure 8. Density plot of danceability vs target

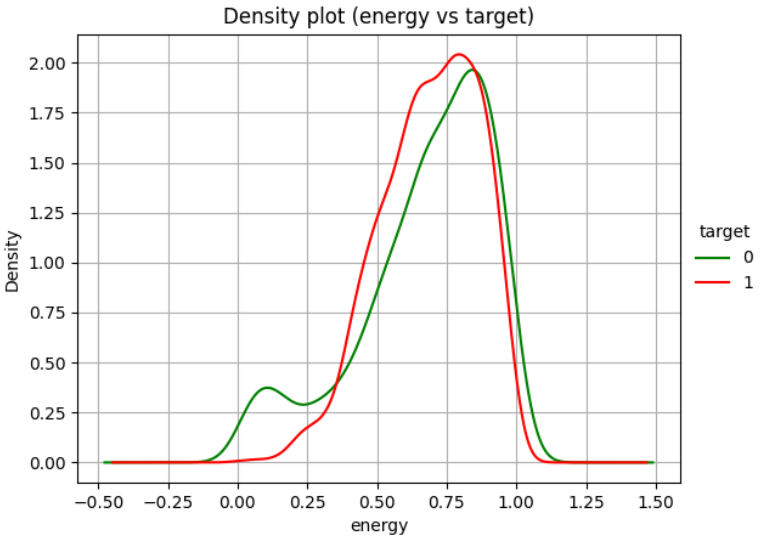


Figure 9. Density plot of energy vs target

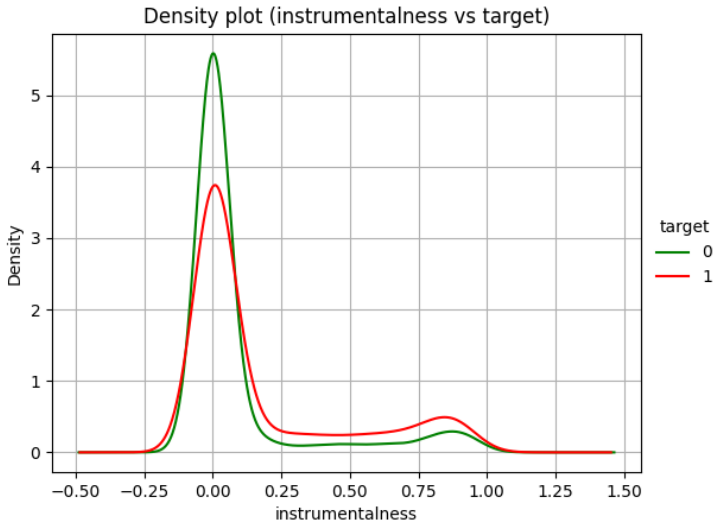


Figure 10. Density plot of instrumentalness vs target

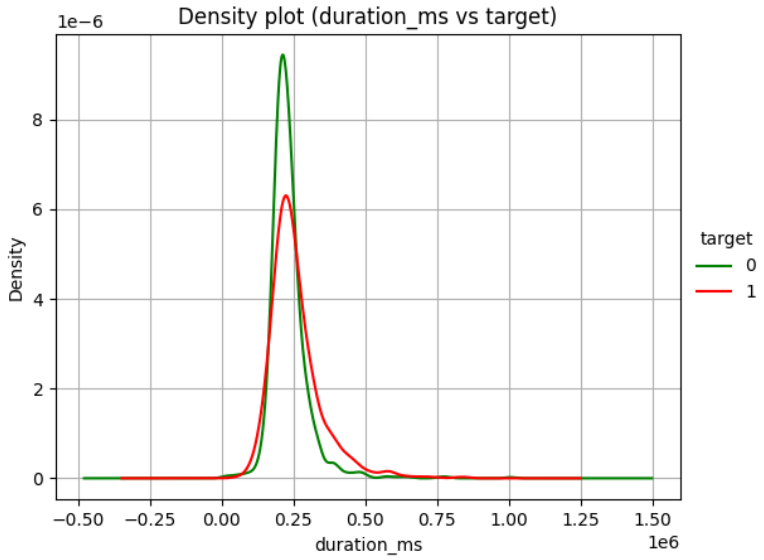


Figure 11. Density plot of duration vs target

2.3.

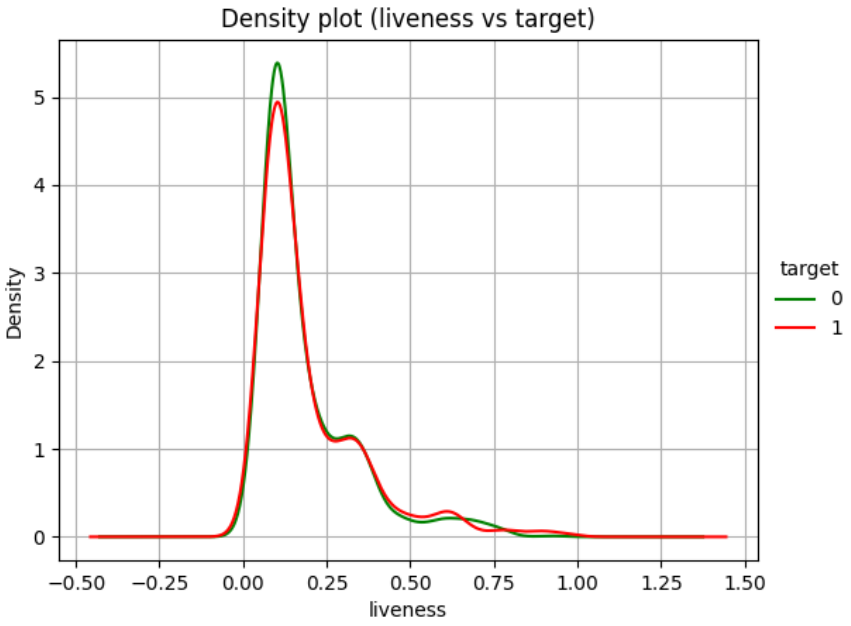


Figure 12. Density plot of liveness vs target

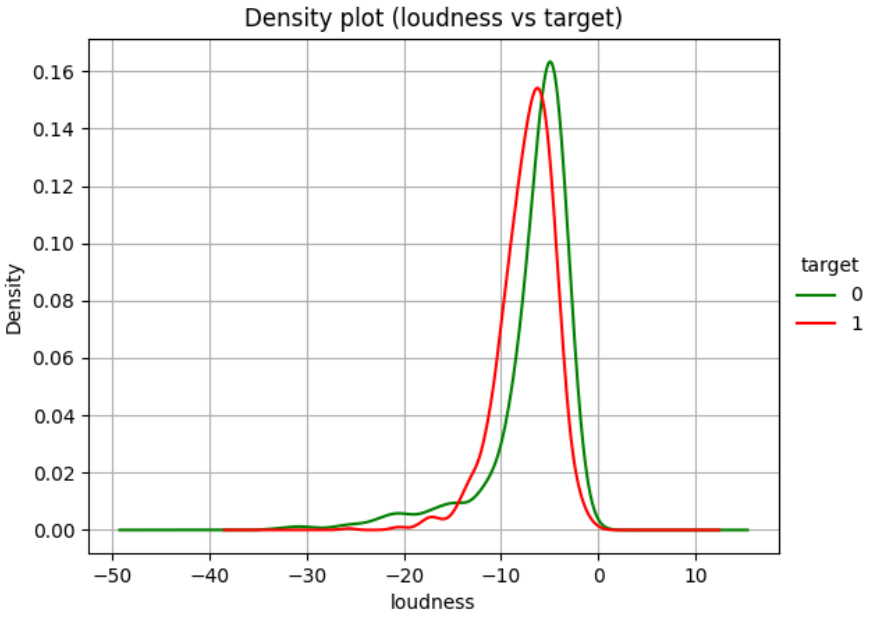


Figure 13. Density plot of loudness vs target

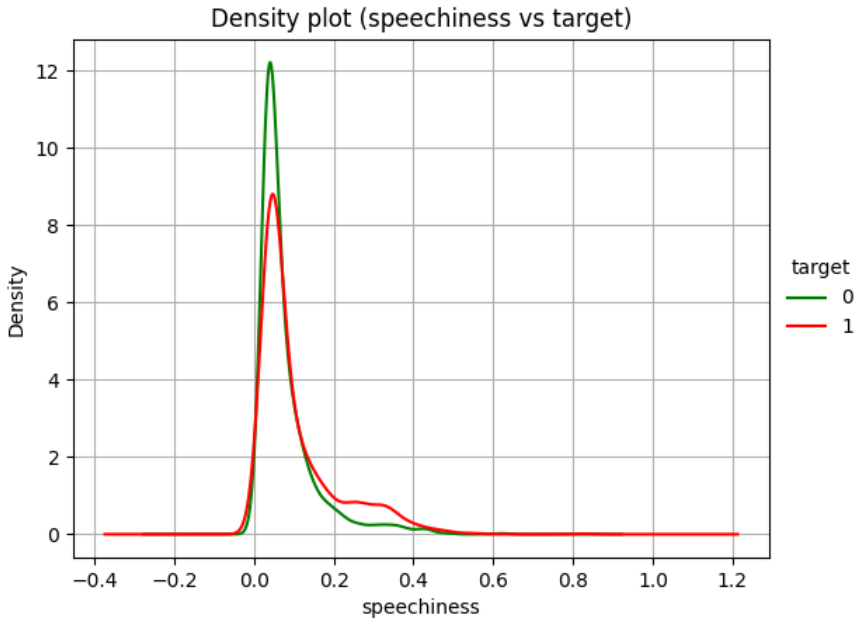


Figure 14. Density plot of speechiness vs target

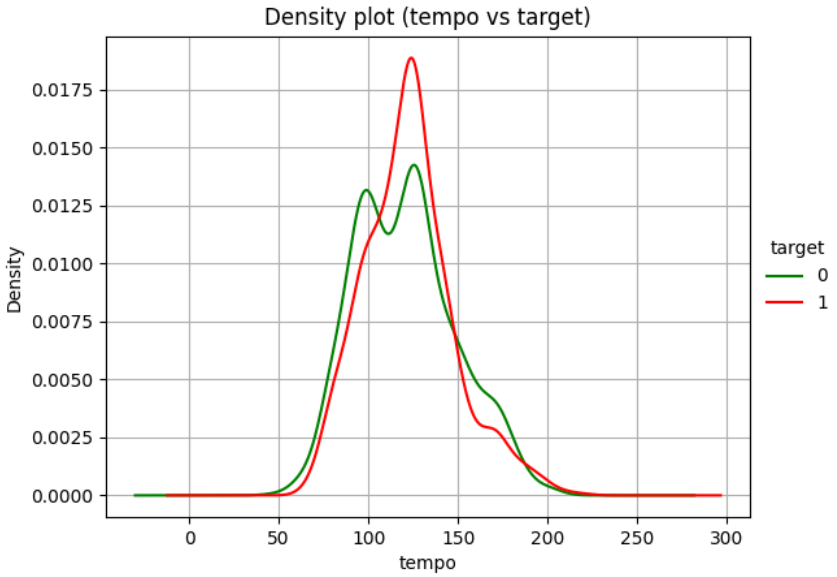


Figure 15. Density plot tempo vs target

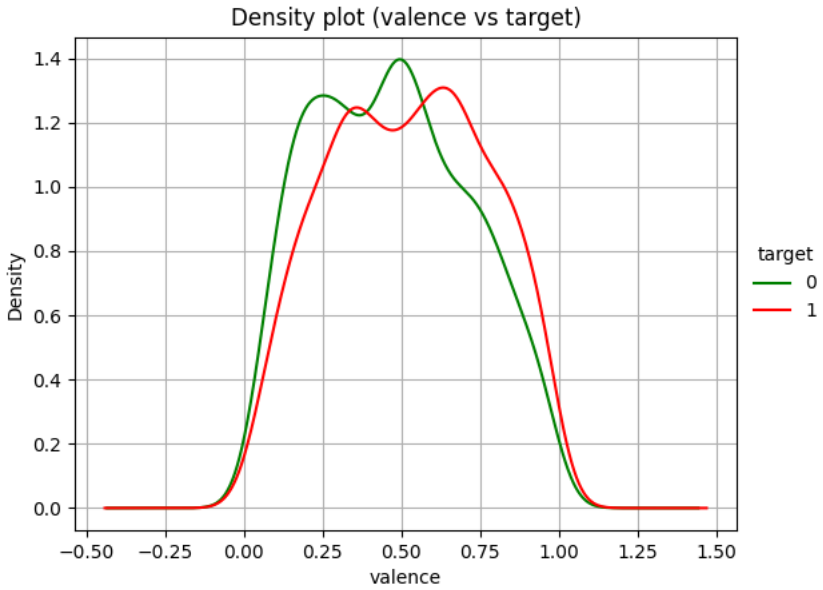


Figure 16. Density plot valence vs target

2.4.

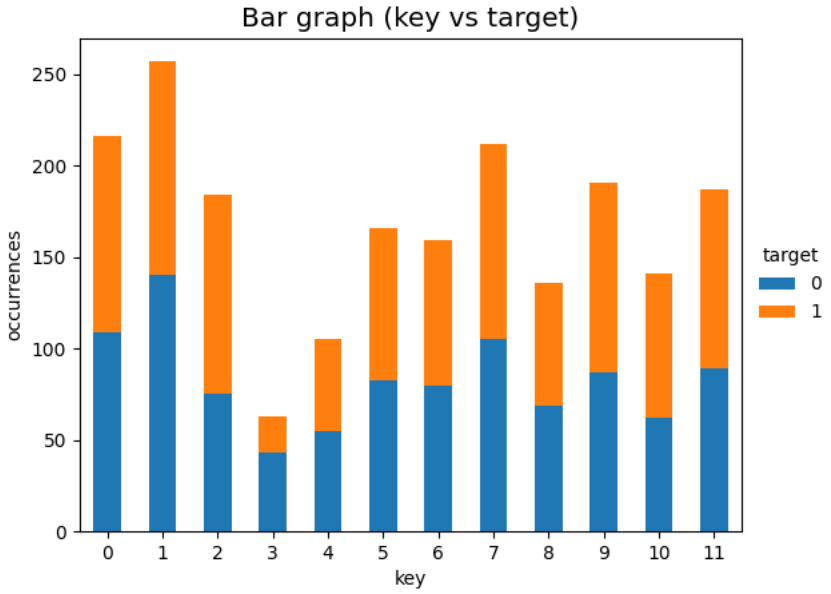


Figure 17. Bar graph of key vs target

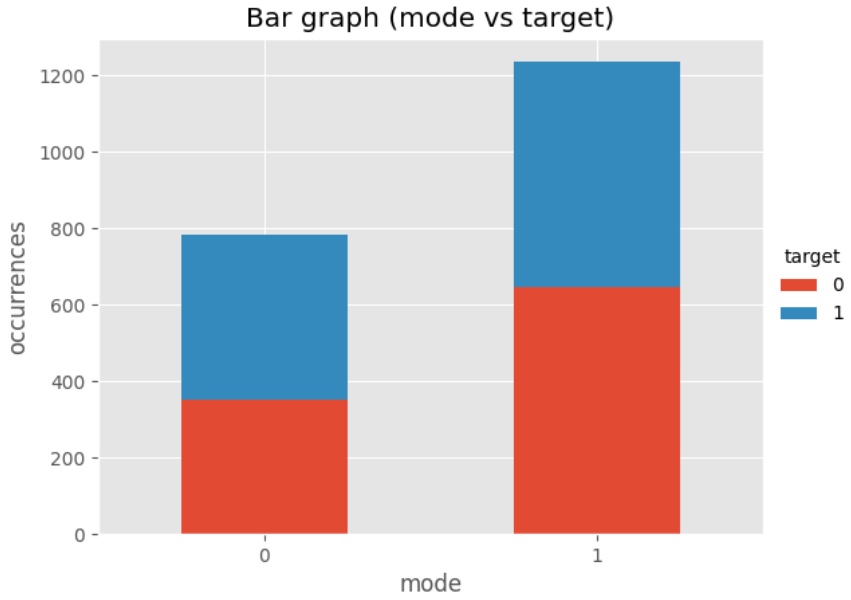


Figure 18. Bar graph of mode vs target

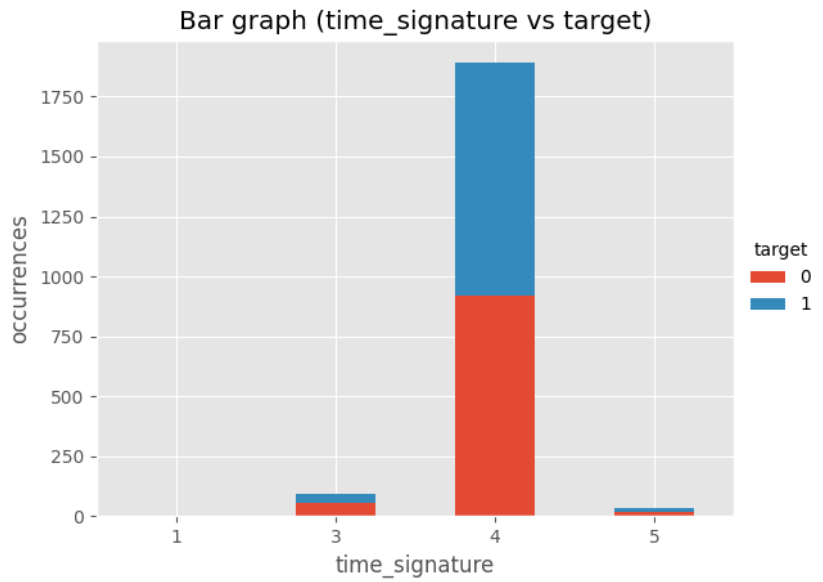


Figure 19. Bar graph of time signature vs target

2.5.

Based on the figure from 2.3 and 2.4, we can say a couple of things about the user

- preference to listen and like to acoustic music.
- preference to listen to danceable music.
- preference to listen to energetic music.
- preference to listen to non-instrumental music.
- preference to listen to music around 4min and 10s in duration.
- preference to listen to non-live music.
- preference to listen to loudness around -5db.
- preference to listen to low speechiness.
- preference to listen and like tempo around 125bpm.
- no preference towards valence
- no strong preference towards key
- preference to listen to major modality music.
- preference to listen to music in C#.
- preference to listen to time signature 4/4

2.6.

Accuracy for each class:

- Class 0 accuracy is 66%
- Class 1 accuracy is 65%

Overall accuracy is 66%

Confusion matrix

	Predicted 0	Predicted 1
True 0	205	94
True 1	113	194

Full logistic equation:

$$\frac{1}{1+e^{-(-0.33068084+0.36285411x_1+0.2147586x_2+0.22124038x_3+0.29667874x_4+0.0517596x_5+0.05914987x_6-0.48819661x_7-0.05497857x_8+0.36060704x_9+0.1326245x_{10}-0.04092257x_{11}+0.17146763x_{12})}}$$

2.7.

When threshold is 0.3:

Confusion matrix

	Predicted 0	Predicted 1
True 0	43	255
True 1	30	278

Class 0 accuracy is 23%
Class 1 accuracy is 66%
Overall accuracy is 53%

When threshold is 0.4:

Confusion matrix

	Predicted 0	Predicted 1
True 0	121	177
True 1	63	245

Class 0 accuracy is 50%
Class 1 accuracy is 67%
Overall accuracy is 60%

When threshold is 0.5:

Confusion matrix

	Predicted 0	Predicted 1
True 0	179	119
True 1	108	200

Class 0 accuracy is 61%
Class 1 accuracy is 64%
Overall accuracy is 63%

Between these three threshold values, it is best to use 0.4. Because it has the highest class 1 accuracy, but also not having the worst class 0 accuracy. It is better to have more false positives than false negatives, because the user could just tell if the song wasn't to his liking, rather than not even being able to hear a good song because it was a false negative.

2.8.

Based on the density plots, the most obvious one that could be removed is valence, because there was no significant listening of like preference. But to prove this theory and even find more features which are not important all combinations should be tried.

This was done by calculating R^2 with and without a certain feature and comparing the difference between the R^2 accuracies. Because the difference is so small and highly dependent on which observations are used in the test dataset, this experiment needs to be averaged over many iterations, so the results become more consistent and reproducible.

This is a table created by splitting the observations into test and training dataset and calculating R^2 with and without a feature every iteration, 50 iterations were used. A positive value shows that, after removing R^2 increases, and negative that it decreased.

Removed feature	Effect on R^2
Valence	+0.007425742574257419
Mode	+0.002211221122112208
Tempo	+0.001650165016501650
Key	+0.001320132013201320
Time signature	+0.000594059405940590
Liveness	+0.000231023102310226
Energy	-0.00066006600660066
Duration	-0.00316831683168317
Instrumentalness	-0.00465346534653465
Loudness	-0.00607260726072607
Danceability	-0.01267326732673267
Acousticness	-0.01290429042904290
Speechiness	-0.02458745874587458

From this table we can tell that valence, mode, tempo, key, time signature and liveness could be removed, to increase R^2 rating.

2.9.

To find the best result, all combinations need to be tested. Transformations were tested on all features, with 3 threshold values and checking $\sqrt{X}$, X^2, $\sqrt[3]{X}$, $\log(X)$ and e^X functions.

This table shows the top 5 best transformations by overall accuracy.

Feature	Transform	Threshold 0.3			Threshold 0.5			Threshold 0.7		
		Class 0 accuracy	Class 1 accuracy	Overall accuracy	Class 0 accuracy	Class 1 accuracy	Overall accuracy	Class 0 accuracy	Class 1 accuracy	Overall accuracy
Energy	$\sqrt{X}$	0.193772	0.949527	0.589109	0.726644	0.684543	0.704620	0.948097	0.337539	0.628713
Energy	X^2	0.214533	0.940063	0.594059	0.723183	0.678233	0.699670	0.958478	0.328076	0.628713
Energy	$\sqrt[3]{X}$	0.200692	0.946372	0.590759	0.709343	0.690852	0.699670	0.948097	0.343849	0.632013
Energy	$\log(X)$	0.214533	0.946372	0.597360	0.712803	0.684543	0.698020	0.948097	0.347003	0.633663
Energy	e^X	0.204152	0.940063	0.589109	0.726644	0.665615	0.694719	0.951557	0.312303	0.617162

Coincidentally the top 5 are completely made up of transformations to energy. Another interesting observation is that from the table in section 2.8, we can see that energy *almost* had no effect on R^2, but transforming energy is the best choice out of all the features.

Task 3, LDA and QDA classifiers

3.1.

The hand sign samples in "Figure 20", there are some parts of the data could make it harder to classify. Some of these things are:

- Skin color
- Blurriness
- Objects in background
- Lighting on hand
- Similar looking hand signs

Figure 20. 4x4 grid of signs for the letter B

3.2.

Overall accuracy: 43.27%

Class	Accuracy
0	82.18%
1	63.19%
2	78.71%
3	61.63%
4	60.64%
5	42.11%
6	49.71%
7	53.67%
8	13.19%

Class	Accuracy
9	-
10	38.97%
11	44.02%
12	25.89%
13	24.05%
14	41.87%
15	20.75%
16	49.39%
17	15.97%

Class	Accuracy
18	34.55%
19	47.98%
20	30.08%
21	26.59%
22	39.81%
23	32.58%
24	28.61%
25	-

3.3.

To find the best combination of filtrations which results in the highest overall accuracy, I will be trying to remove every n-th pixel or leaving every n-th pixel, where n can be any number from 2 and higher. Because of computational constraints, I will only be testing n up to 19, picking a higher n does not improve the overall accuracy, so the first n values from 2 to 19 will be good enough.

Applied filters in order from top to bottom:

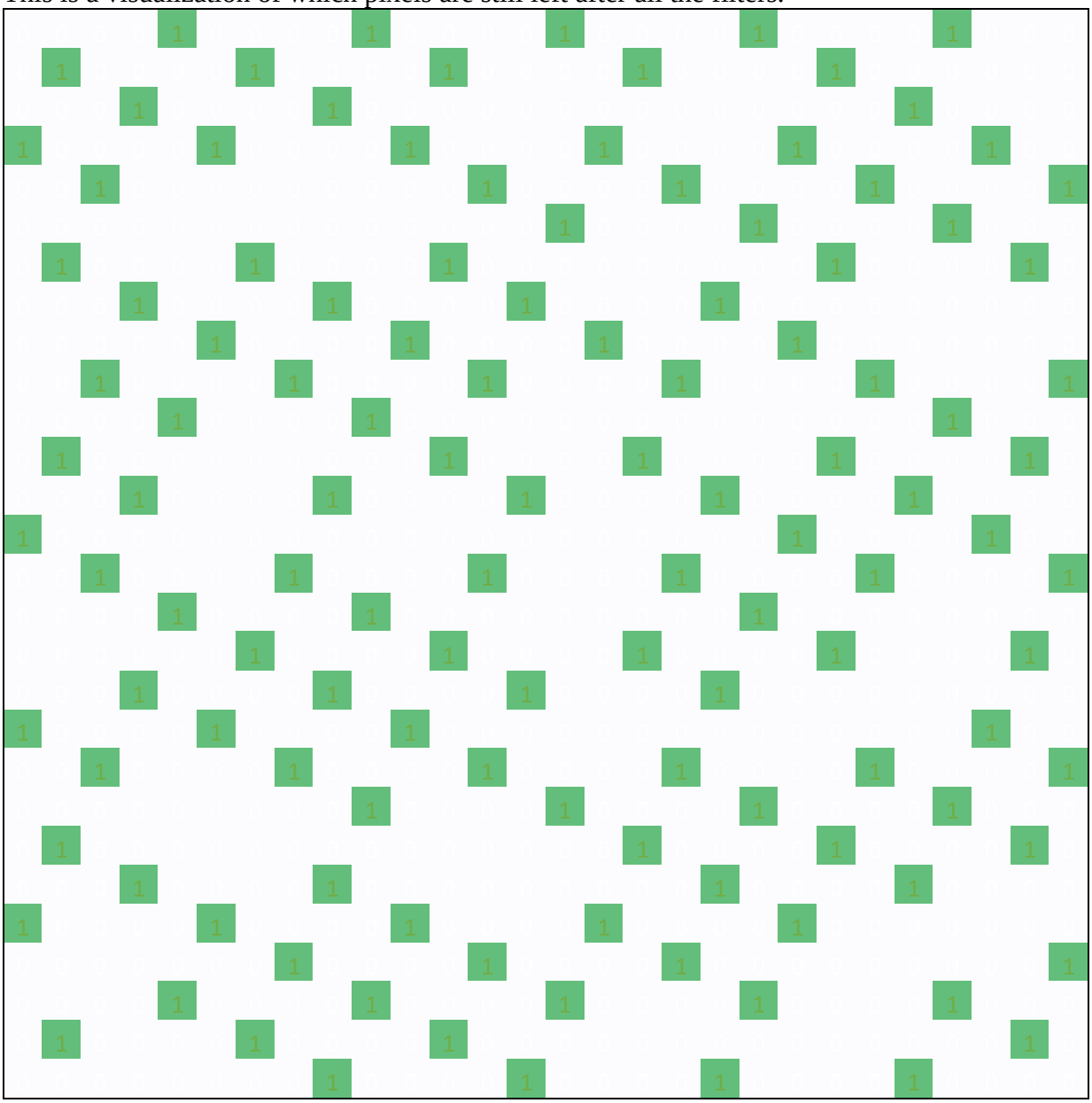
- Leave every 5-th pixel.
- Remove every 15-th pixel.
- Remove every 14-th pixel.
- Remove every 11-th pixel.

After applying all these filters, the accuracy increases from 43.2654% to 61.8377%. The number of pixels used decreased from 784 to 124.

Applying any more filters does not decrease the number of pixels significantly, because most of what is already left is needed to determine the hand sign.

This decreasing of features works because, most pixels don’t provide unique information, because usually a pixel’s colors are not that different from its neighbors. Background pixels could even introduce noise into the model, because of how varied it can be.

This is a visualization of which pixels are still left after all the filters.



3.4.

Overall accuracy: 65.74%

Class	Accuracy
0	100.00%
1	62.50%
2	93.23%
3	92.24%
4	38.35%
5	99.19%
6	70.69%
7	38.53%
8	78.82%

Class	Accuracy
9	-
10	68.88%
11	89.95%
12	40.36%
13	73.54%
14	68.29%
15	18.44%
16	100.00%
17	97.22%

Class	Accuracy
18	91.87%
19	58.47%
20	61.28%
21	30.35%
22	89.81%
23	61.05%
24	63.25%
25	-

3.5.

To find the best combination of filtrations which results in the highest overall accuracy, I will be trying to remove every n-th pixel or leaving every n-th pixel, where n can be any number from 2 and higher. Because of computational constraints, I will only be testing n up to 19, picking a higher n does not improve the overall accuracy, so the first n values from 2 to 19 will be good enough.

Applied filters in order from top to bottom:

- Remove every 3-nd pixel.
- Remove every 11-th pixel.

After applying all these filters, the accuracy increases from 65.7417% to 77.1751%. The number of pixels used decreased from 784 to 476.

Removing features did help like in LDA, but not as much and not as many filters were needed. Trying to add more filter only just decreases the accuracy. This suggests that the classes are not linearly separable and using QDA is highly preferable for this task.

This is a visualization of which pixels are still left after all the filters.