Moral AI IQP

Worcester Polytechnic Institute

Ryan Benasutti

March 2019

Abstract

Artificial intelligence is being deployed in increasingly autonomous systems where it will have to make moral decisions. However, the rapid growth in artificial intelligence is outpacing the research in building explainable systems. In this paper, a number of problems around one facet of explainable artificial intelligence, training data, are explored. A solution to these problems is presented. Additionally, the human decision making process in unavoidable accident scenarios is explored.

Acknowledgements

1. Professor Therese Smith
2. Professor Yunus Telliel
3. Griffin Tabor

Contents

Chapter 1

Introduction

1. Autonomous vehicle technology is growing rapidly and AI is a key piece of that technology. As this technology gets closer to attaining full autonomy, the AI deployed in these systems will have greater responsibility than ever. These AI systems must be explainably fair, i.e. they must both make decisions using only the least amount of information necessary for optimal performance and make those decisions predictably and correctly. For example, the AI in an autonomous vehicle does not need to be supplied with information about a pedestrian's race, even though race may be an impactful trait in other fields, especially medical fields [2]. Furthermore, these AI systems must also be explainable for legal reasons, such as determining which party is at fault in the event of a car accident or, in the European Union, complying with a user's "right to explanation" [7].

2. The demand for explainable AI is increasing, such as DARPA's Explainable Artificial Intelligence (XAI) program [9]. This program aims to develop explainable AI systems such as in Figure A.2. There has also been a symposium focusing on AI inclusivity towards marginalized peoples [5, 1]. This symposium illustrates the increasing need to discuss AI fairness and inclusivity in a way that non-technical people can understand. One facet of this need that this paper addresses is the question of specifically how much one needs to care about possible biases in the various stages of AI architecture.

3. We seek to demonstrate how an AI can learn a bias and empirically determine the severity of that bias. Classification accuracy testing will be employed to evaluate the trained AI and determine if any bias was learned, and, if so, the severity of that bias.

4. We also seek to understand the decision making process in humans behind making moral decisions in unavoidable accident scenarios, i.e.

dilemmas. This part of the research will be done by surveying a group of people and performing qualitative analysis on the survey results. These results serve not only as a way to understand this decision making process but also as a language we can use to craft communications with the symposium audience mentioned previously.

Chapter 2

Background

1. Introduce background readings.
2. Cite examples of AI that must (or will in the near future) make moral decisions.
   - [4] performs end-to-end learning which "map raw pixels from a single front-facing camera directly to steering commands". With this approach, the AI will have to respond directly to pedestrians and other external stimuli.
3. The Moral Machine experiment [3] is prior research into people's preferences in moral dilemmas. Participants are shown a moral dilemma involving an autonomous vehicle, passengers, and pedestrians. In each dilemma, the participant must choose between inaction, which results in the certain death of the pedestrians, and action, which results in certain death of the passengers. The study revealed three strong global preferences towards sparing humans over animals, sparing more lives rather than fewer, and sparing younger lives rather than older. The study also showed that some preferences vary between countries depending on that country's propensity towards egalitarianism.

Chapter 3

Methods

3.1 Data Generation

The data is generated using a graphical model to control the conditional probabilities for the states of each variable. The variables in the model correspond directly to the attributes of a person. Figure A.1 is a rendering of the graphical model. For example, people in the first option could be more likely to jaywalk then people in the second option, producing a data set which is biased towards/against jaywalkers. When combined with control over the number of people in each option, this method can produce both subtle and strong bias. The code for the domain of each attribute of a person is in Figure 3.1. The Python package pgmpy is used to create the graphical model and infer each variable's probability distribution. These distributions are then used to pick elements from each variable's domain. This process is repeated for each attribute of each person and for the num ber of people in each option of a dilemma, forming a complete dilemma. The number of dilemmas generated is specified programmatically using the `TrainMetadata` class, which captures the number of dilemmas to generate and the maximum number of people per option.

```
age_states = [10, 20, 30, 40, 50, 60]
race_states = [Race.white, Race.black, Race.asian,
               Race.native_american, Race.other_race]
legal_sex_states = [LegalSex.male, LegalSex.female]
jaywalking_states = [False, True]
driving_under_the_influence_states = [False, True]
```

Figure 3.1: Python code for the bracketed attributes of a Person

3.2 Data Bracketing

Attributes are one-hot encoded so the neural network is resilient to unspecified attributes. Age is bracketed by increments of 10 years. Some example encoded ages are shown in Table 3.1. Boolean attributes are encoded into three increments, as shown in Table 3.2.

Age (yr)	unspecified	1-10	11-20	21-30	31-40	41-50	51-60
unspecified	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
3	0	1	0	0	0	0	0
16	0	0	1	0	0	0	0
42	0	0	0	0	0	1	0

Table 3.1: Example age attribute encoding.

Value	unspecified	false	true
unspecified	1	0	0
false	0	1	0
true	0	0	1

Table 3.2: Example boolean attribute encoding.

3.3 Data Storage

Data is stored using the JSON format using a serialization process called pickling via the Python library jsonpickle. This process was chosen because it produces easily machine-readable files and because JSON is a popular data storage format. The purpose of storing the generated data sets is to keep the data consistent between test iterations and to share the data. Both the training and test data sets are pickled after generation.

3.4 Neural Network Model

There are two primary requirements of the neural network used in the experiments. First, the network must classify the training data. In other words, when given a dilemma, the network must classify that dilemma by picking which option to avoid. For example, in a dilemma with two options of three and four people, respectively, the correct classification is the second option because it allows the autonomous vehicle to save more people. In the case where a dilemma has two or more options of equal size, the earlier option is chosen.

Second, the network must be easy to train, meaning that the time required to train the network must be small (on the order of minutes or less) and the hardware resources required to train the network must be minor. Testing the network requires training it many times, so the time required to train the network must be small. Additionally, the network will be trained on personal machines, so any hardware requirements must be easy to meet.

There are three models which were considered when deciding on what network to use. First, an autoencoder: autoencoders are trained using unsupervised learning, so labeling the data is not necessary (want to avoid imparting a set of morals). This model would perform dimensionality reduction, and perhaps learn to ignore noise (i.e. uniformly distributed attributes) in the data set, but would be unable to classify the dilemmas.

Second, an autoencoder in combination with a simple neural network trained using supervised learning: this model solves the classification problem which the previous model failed at, but introduces unnecessary complexity to the research. The intent of this research is not to build a neural network capable of guiding a real autonomous vehicle, so this model was deemed unnecessarily complex.

Lastly, a recurrent neural network (RNN) with long short-term memory (LSTM). This option was considered because RNN's are capable of accepting variable-length sequential data. We thought the network may need to handle variable-length data, but the engineering challenge that design posed was traded in favor of both limiting the maximum number of people in an option and bracketing the data as covered in section 3.2. Additionally, this network does not directly solve the classification problem, so it is only marginally applicable for this research.

The final neural network chosen is a simple, shallow, feed-forward network with one hidden layer trained using supervised learning, pictured in Figure A.3. The input layer has dimensionality equal to the number of attributes per person (after one-hot encoding) multiplied by the number of options per dilemma multiplied by the maximum number of people per option. The output layer has dimensionality equal to the number of options per dilemma. The hidden layer has dimensionality equal to the average of that of the input and output layers. An example implementation in Keras of the model can be seen in Figure 3.2.

```
output_dim = 2
input_dim = 22 * output_dim * \
        train_metadata.max_num_people_per_option

model.add(Dense(units=input_dim, activation='relu',
            input_dim=input_dim))
model.add(Dense(units=round((input_dim + output_dim) / 2),
            activation='relu'))
model.add(Dense(units=output_dim, activation='softmax'))

model.compile(loss=losses.categorical_crossentropy,
              optimizer='sgd',
              metrics=[metrics.categorical_accuracy])

model.fit(train_data, train_labels, epochs=5, batch_size=32)
```

Figure 3.2: The Keras code for the neural network model.

3.5 Neural Network Training

The training data given to the network is one-hot encoded, so the categorical cross entropy loss function is used. As the network is quite simple, a stochastic gradient descent optimizer suffices. Training happens over 5 epochs with a batch size of 32. An example implementation in Keras can be seen in Figure 3.3; additionally, Figure A.4 provides a simple visualization of the dimensionality of the network.

```
model.compile(loss=losses.categorical_crossentropy,
              optimizer='sgd',
              metrics=[metrics.categorical_accuracy])

model.fit(train_data, train_labels, epochs=5, batch_size=32)
```

Figure 3.3: The Keras code for the neural network model.

3.6 Neural Network Testing

The neural network is tested using Keras to evaluate the classification accuracy and loss against a test data set. The test data is generated using a graphical model in the same manner as the training data; however, it is important to note that in this step, the test data is generated separately from the training data: it is not a sampled subset of the training data. Each training data set is tested five times. Each iteration involves training the neural network on the training data set and evaluating its performance against a test data set to collect classification accuracy and loss information. The results of all five iterations are averaged to produce an average classification accuracy and loss. Many training data sets are generated and tested against the same test data set. Each data point in Figure A.5 corresponds to a unique training data set. All data points in the figure are tested against the same test data set.

The test result naming format, seen, for example, in the caption of Figure A.5, reproduced here in Figure 3.4, contains all the information necessary to understand the characteristics of the test data set. The name can be split into three sections. First, 40-60 refers to the probability distribution of people among the two options: $P(O_1) = 0.4$, $P(O_2) = 0.6$. Second, 100-0 refers to the probability distribution of jaywalking for those in the first option. $P(\neg J \mid O_1) = 1$, $P(J \mid O_1) = 0$. Finally, 0-100 refers to the probability distribution of jaywalking for those in the second option. $P(\neg J \mid O_2) = 0$, $P(J \mid O_2) = 1$.

Classification accuracy against test 40-60 100-0 0-100

Figure 3.4: The title of Figure A.5

Chapter 4

Findings and Analysis

4.1 Response to Bias

Our research found that the AI became biased when the training data featured a strong trend not present in the test data. For purposes of a control test, a test data set was generated with uniformly distributed people and a uniformly distributed jaywalking probability. Figure A.17 shows the result of this test. In this scenario, $P(J)$ is observed to be independent of $P(O_1)$. When $P(J \mid O_1) < 0.1$, the neural network classifies dilemmas incorrectly. This trend continues as $P(J \mid O_1)$ decreases, thereby increasing the severity of the bias in the training data set. A similar but opposite trend occurs when $P(J \mid O_1) > 0.9$.

Observing the contour plot in Figure A.5, one can see that classification accuracy decreases abnormally (i.e. differently than in the control in Figure A.17) when $P(O_1) > 0.6$ and $P(J \mid O_1) < 0.2$. In this area, the training data set consists mostly of people in the first option. Most people in the first option are not jaywalkers and most people in the second option are. Therefore, the training data set is biased to prefer non-jaywalkers because they appear disproportionately frequently in the (larger) first option. The neural network, now having learned this trend, is tested against a test data set in which most people are in the second option. Those in the first option are not jaywalkers and those in the second option are. The network tends to select the first option because it contains far fewer jaywalkers than the second option, despite the first option being smaller than the second and therefore the incorrect choice. This causes the network's classification accuracy to decrease in this region. The trend continues as $P(O_1)$ increases while $P(J \mid O_1)$ decreases, thereby increasing the severity of the bias in the training data set. Another view into the network's decisions is Figure A.7, which shows the real value of $P(J)$ when the network classified a dilemma incorrectly. In the areas of the contour plot corresponding to Figure A.5's areas of worst accuracy, we can see that the real value of $P(J)$ is close to zero. In other words, the network performs worst when it picks an option because that option is absent of jaywalkers (i.e., when the network makes a decision based on the bias it learned rather than the rule used to label the test data set). This is further evidence that the network has learned a bias against jaywalkers. A similar but opposite trend occurs when $P(O_1) < 0.4$ and $P(J \mid O_1) > 0.8$.

4.2 Avoiding Bias

Three ways for the neural network to learn a bias

1. Flaws in data
   (a) Any information which is not strictly necessary for the neural network to make effective decisions should not be given to the network. <Maybe talk about restriction layer and that architecture?>
   (b) Neural networks used for object detection in images suffer from predictive inequality in detecting people of differing skin tones [11]. Those with skin tones in the Fitzpatrick range $[1, 3]$ are more accurately identified than those with skin tones in the $[4, 6]$ range. Furthermore, this problem persists between networks of different architectures. Although the authors of that research propose a different loss function which decreases predictive inequality, one could imagine a totally different system which does not use color cameras at all. Infrared cameras may serve as a good replacement because they do not sense skin tones.
2. Flaws in AI architecture
   (a) This research uses a simple, shallow neural network to reduce architectural complexity. Deeper networks, specifically deep convolutional networks, undoubtedly perform better, but these network architectures suffer from increased design complexity and increased training difficulty [10].
3. Flaws in testing
   (a) If one is concerned that a system may be less able to measure some attribute in a certain environment, then the testing for the system may want to overrepresent that attribute. In the context of [11], the test data set used might consist of a majority of images of people with skin tones in the Fitzpatrick range $[4, 6]$, regardless of whether the system is expected to operate in an environment consistent with that skin tone distribution or not.

(b) If a system should be equally sensitive to all of its inputs, then those input should be represented equally during testing, even if a different distribution of inputs is expected to be encountered when the system is deployed. In the case of autonomous vehicles, this research assumed the neural network should treat all people equally, but in reality this is a region-specific measure. The Moral Machine experiment measured strong regional preferences for various aspects of decision-making during unavoidable accident scenarios [3]. For example, Eastern countries showed an almost nonexistent preference for sparing the young compared to Western and Southern countries. Southern countries showed a strong preference for sparing females compared to Western and Eastern countries. Not all regional preferences can be reliably accounted for, however. Some preferences, such as the Eastern countries stronger preference for sparing the lawful compared to Western and Southern countries, is not entirely enforceable by AI systems. Some instances of lawfulness classification could be reliable, such as detecting jaywalkers, but others, such as detecting unlawful intoxication, are most likely difficult. There is, of course, the question of whether or not autonomous vehicles should contain any regional preferences or whether they should be totally fair; however, that is outside the scope of this paper.

4.3 Survey Results

The survey results were ... and we extrapolate that the thought process behind these moral decisions is ...

We found several themes in the survey responses

- Humans can be flawed
  - Humans can have bias.
  - Flaws in humans can lead to flaws in AI systems design and in data sets.
- Data can be flawed
  - AI systems trained on flawed data will be flawed themselves.
- Value saving more lives over fewer
  - This is consistent with one of the Moral Machine experiment's findings [3].
- Unwillingness to kill

  - Killing is unjustifiable; therefore, an action that causes a greater number of people to be spared is not necessarily desirable.
  - If there is an option with a chance that people might not die, that option is vastly more desirable than an option containing one person who will certainly be killed.

- Demand for testing
  - Because both humans and data can be flawed, testing must be employed to validate the fairness of the AI system.
  - These tests must have great breadth and cover many scenarios and peoples.
  - The data used for training and testing should be transparent.
  - The methods of testing should be transparent.
  - Good testing methodologies and good results can lead to trust in AI systems.

People can be flawed. This can be addressed with architectural approaches to AI system design.

Data can be flawed. This can be addressed with sensitivity studies.

An AI system operating an autonomous vehicle is in control at all times.

- The system must decide what to do in a dilemma.
- There is no inaction: choosing inaction is equivalent to choosing an action with the same result as inaction.

Chapter 5

Conclusion

1. Our research found that AI becomes biased when ?. Therefore, one must take a certain amount of care in dealing with bias in data (how much?).

2. In order to avoid training biased AI, we recommend formatting training data such that ?.

3. We also recommend that

   - Teams that work with AI, especially teams which create or train AI, should include social scientists.
   - AI could be verified by 3rd party groups in addition to a team's internal testing.

4. Future work includes

   - Explanations, whether of AI decisions, architecture, or other, must be delivered in a way that the user can understand. As Gilpin puts it, "The success of this goal is tied to the cognition, knowledge and biases of the user: for a system to be interpretable, it must produce descriptions that are simple enough for a person to understand using a vocabulary that is meaningful to the user" [6]. This paper has attempted to show specifically how much care one must take in dealing with bias in data, but more attention is needed in other areas of AI architecture.

Bibliography

[1] Artificial intelligence & inclusion.

[2] Sickle cell disease.

[3] Edmond Awad, Sohan Dsouza, Richard Kim, Jonathan Schulz, Joseph Henrich, Azim Shariff, Jean-François Bonnefon, and Iyad Rahwan. The moral machine experiment. *Nature*, 563(7729):59, 2018.

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[5] Berkman Klein Center and Berkman Klein Center. Charting a roadmap to ensure artificial intelligence (ai) benefits all, Nov 2017.

[6] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 80–89. IEEE, 2018.

[7] Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a “right to explanation”. *AI Magazine*, 38(3):50–57, 2017.

[8] David Gunning. Explainable artificial intelligence (xai).

[9] David Gunning. Explainable artificial intelligence (xai): Technical report defense advanced research projects agency darpa-baa-16-53. *DARPA, Arlington, USA*, 2016.

[10] Hrushikesh N Mhaskar and Tomaso Poggio. Deep vs. shallow networks: An approximation theory perspective. *Analysis and Applications*, 14(06):829–848, 2016.

[11] Benjamin Wilson, Judy Hoffman, and Jamie Morgenstern. Predictive inequity in object detection. *arXiv preprint arXiv:1902.11097*, 2019.

Appendix A

Figures

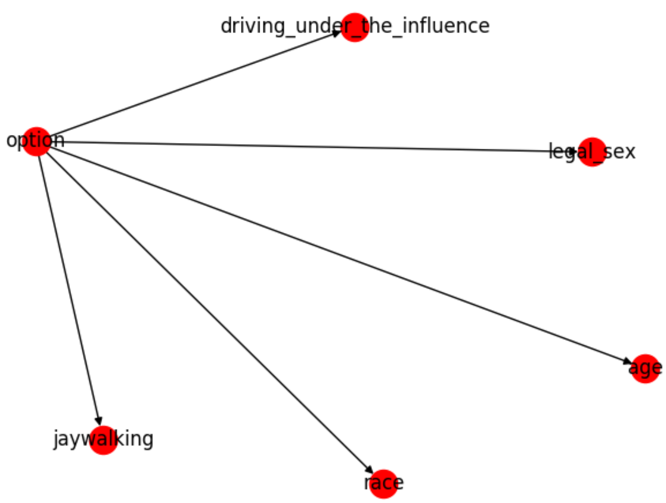

Figure A.1: The graphical model.

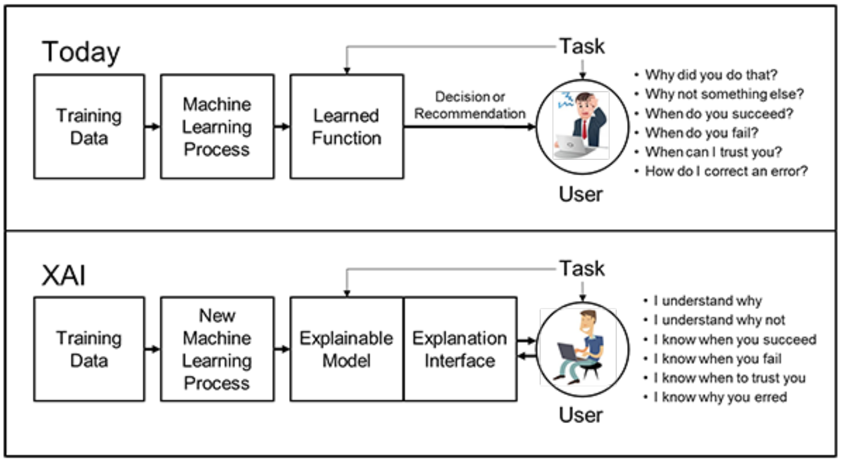

Figure A.2: DARPA's XAI Concept [8]

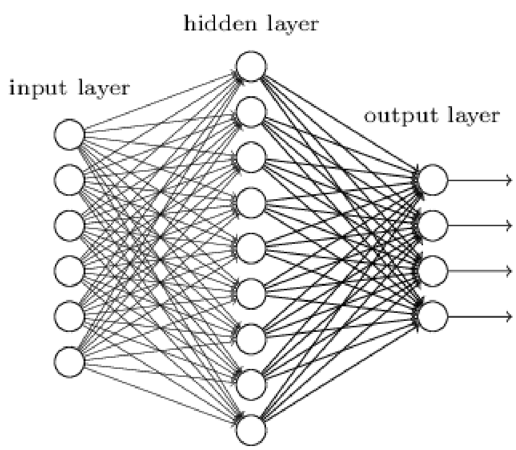

Figure A.3: A shallow feed-forward neural network with one hidden layer.

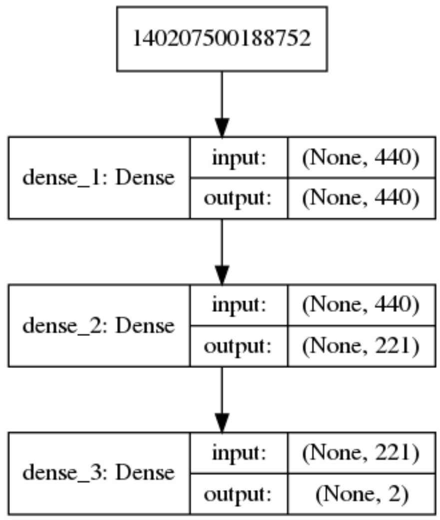

Figure A.4: The dimensions of the neural network used in this research.

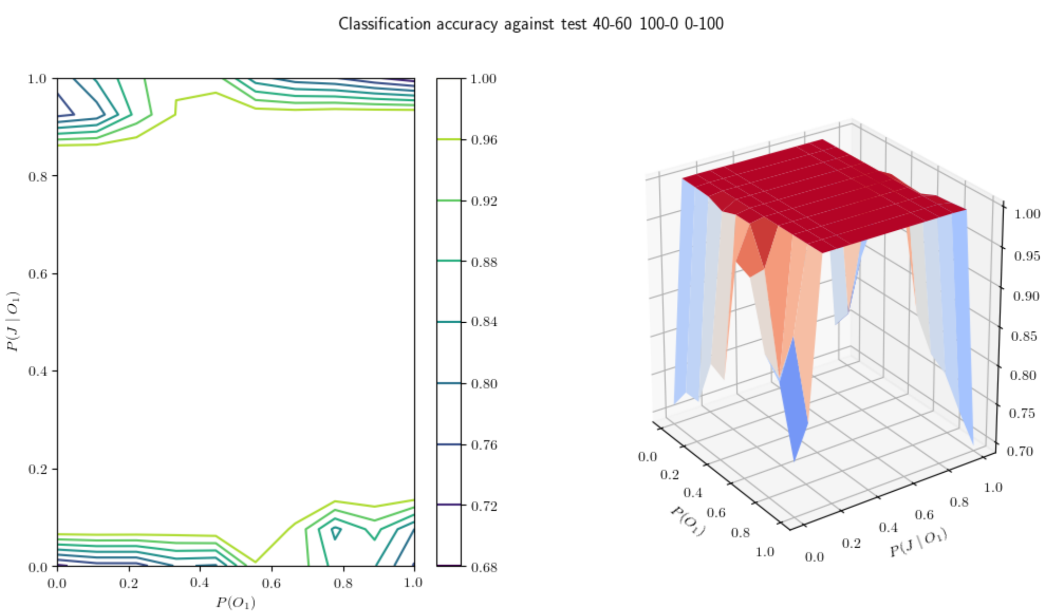

Figure A.5: The classification accuracy against `test 40-60 100-0 0-100`.

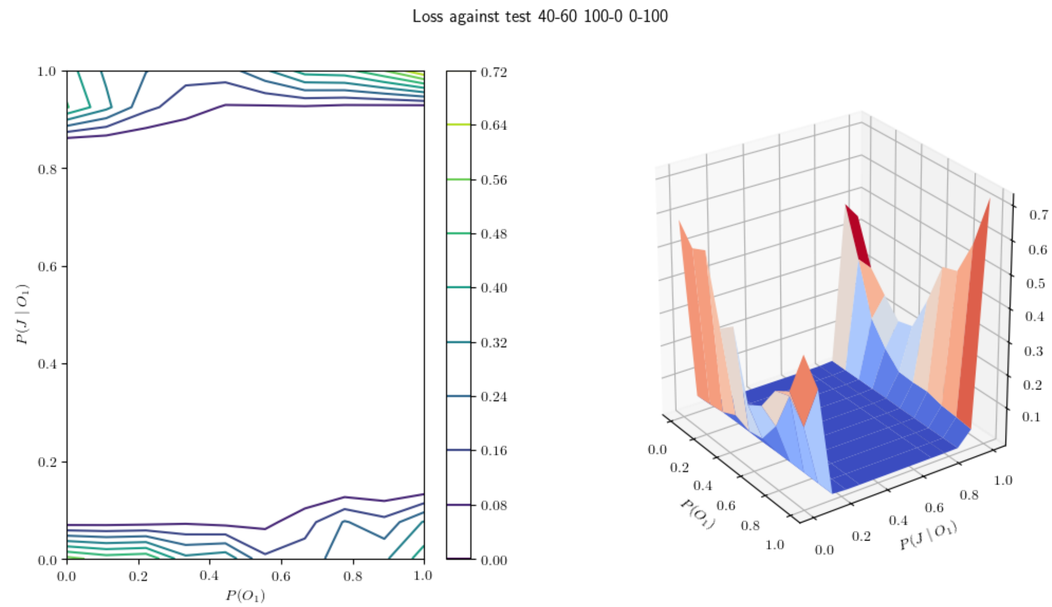

Figure A.6: The loss against `test 40-60 100-0 0-100`.

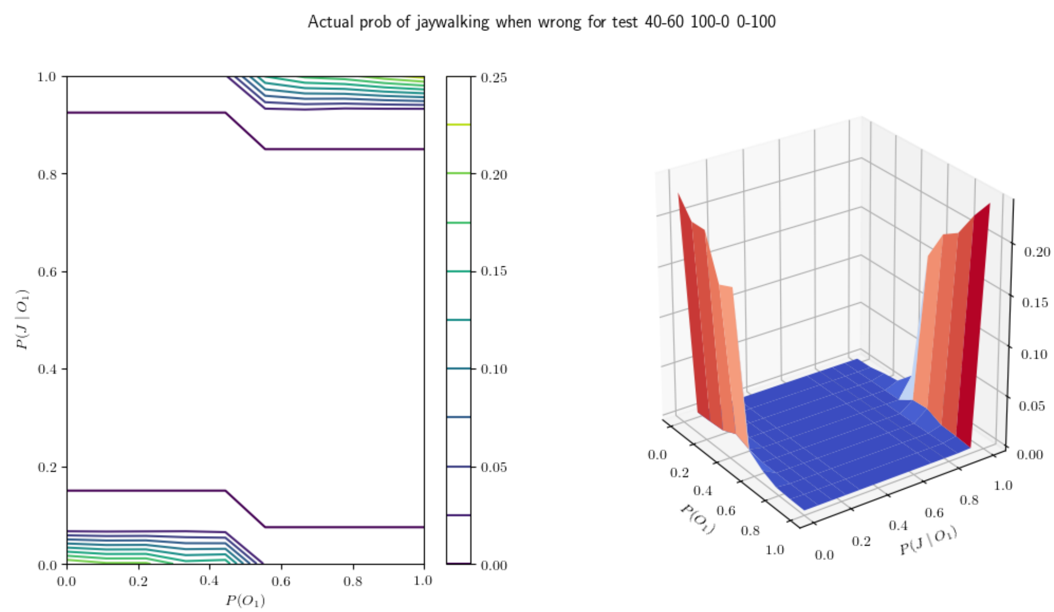

Figure A.7: The actual jaywalking probability when classified incorrectly against `test 40-60 100-0 0-100`.

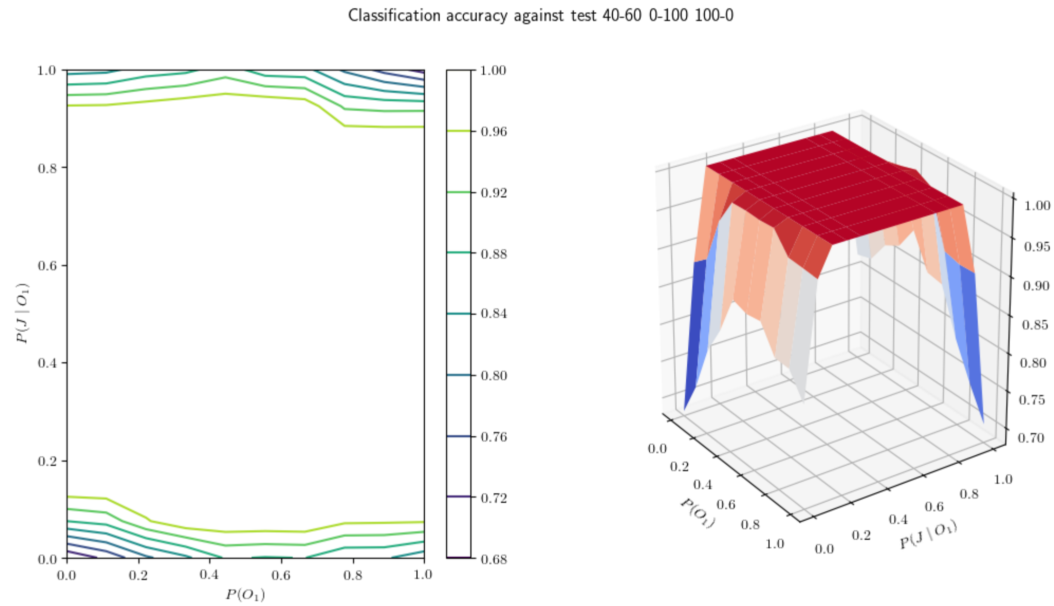

Figure A.8: The classification accuracy against `test 40-60 0-100 100-0`.

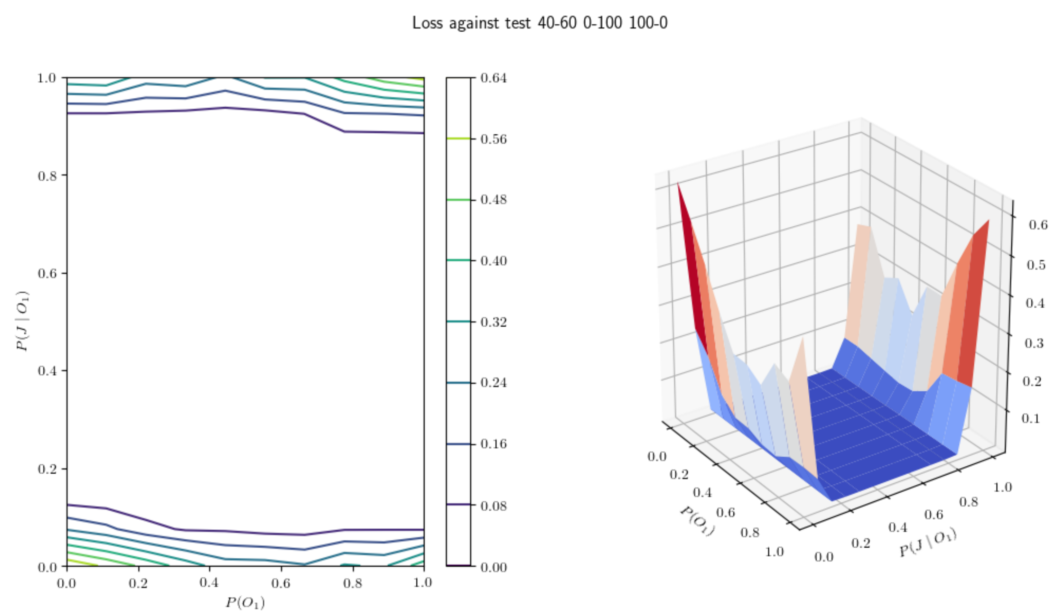

Figure A.9: The loss against `test 40-60 0-100 100-0`.

Actual prob of jaywalking when wrong for test 40-60 0-100 100-0

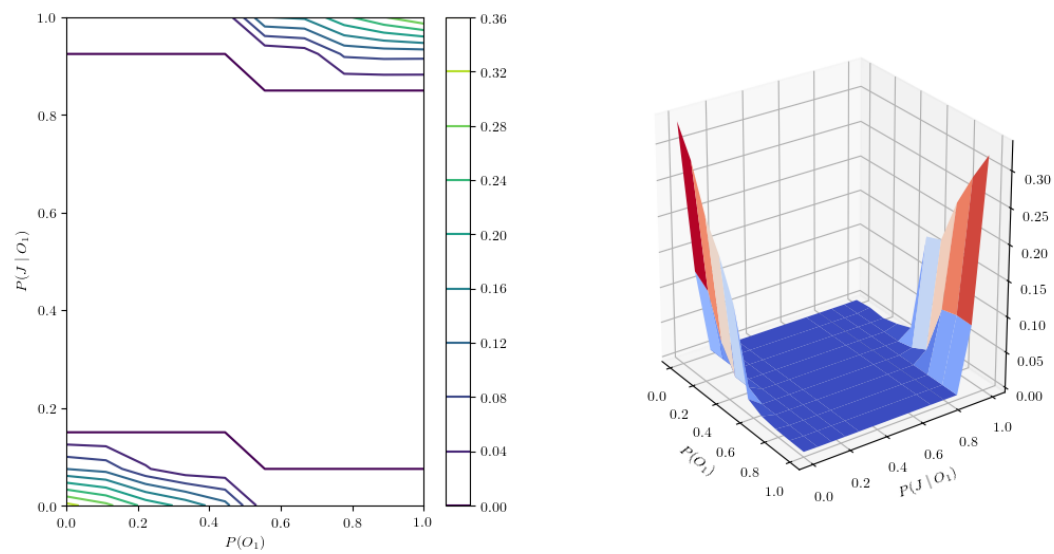

Figure A.10: The actual jaywalking probability when classified incorrectly against `test 40-60 0-100 100-0`.

Classification accuracy against test 40-60 80-20 20-80

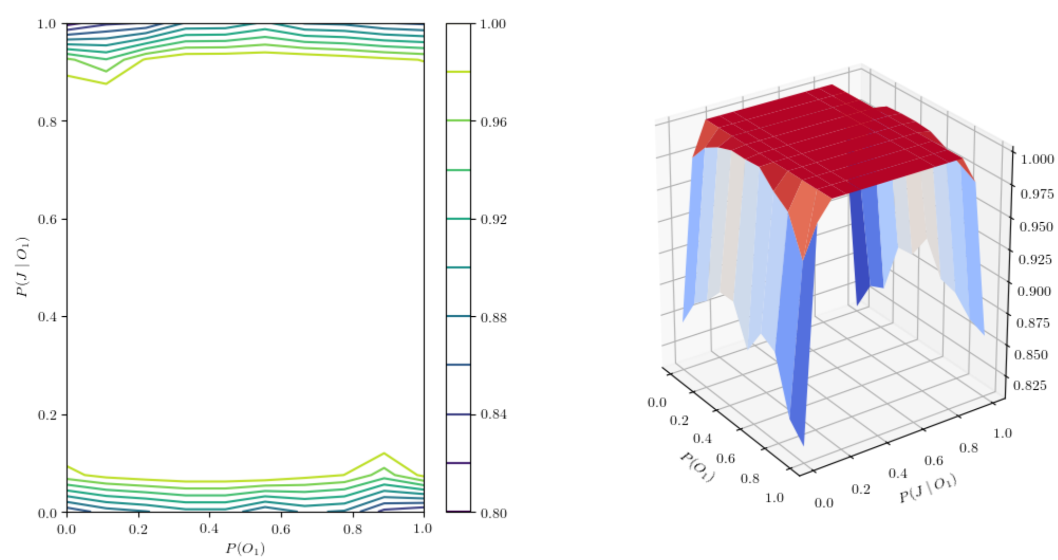

Figure A.11: The classification accuracy against `test 40-60 80-20 20-80`.

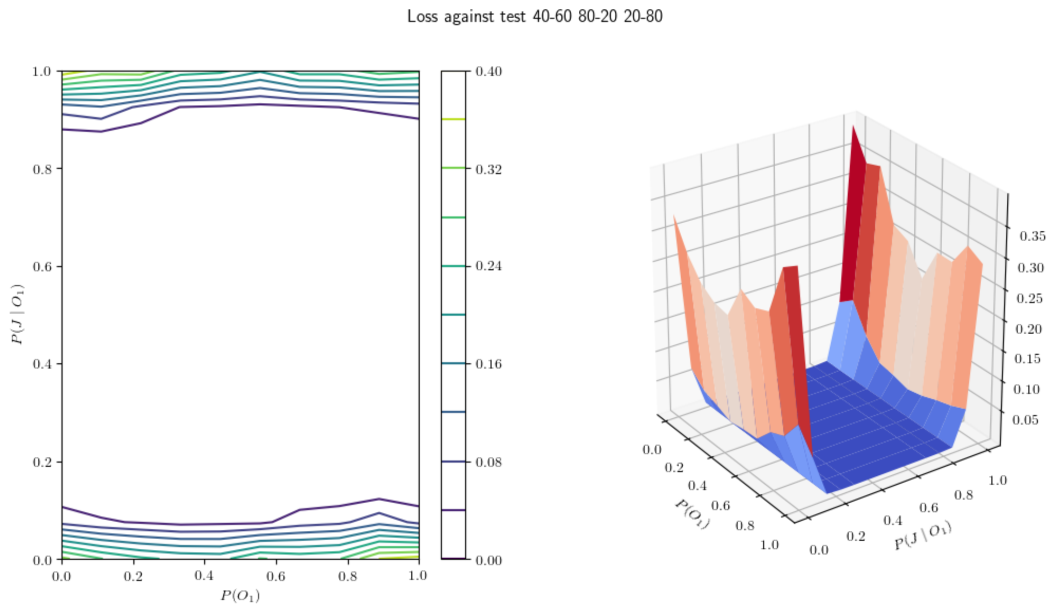

Figure A.12: The loss against `test 40-60 80-20 20-80`.

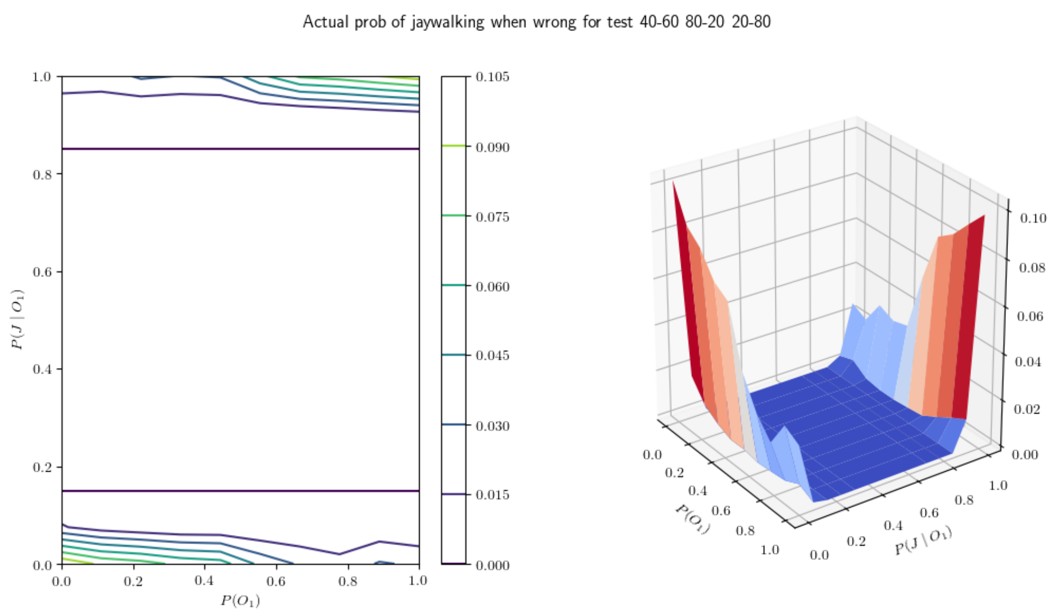

Figure A.13: The actual jaywalking probability when classified incorrectly against `test 40-60 80-20 20-80`.

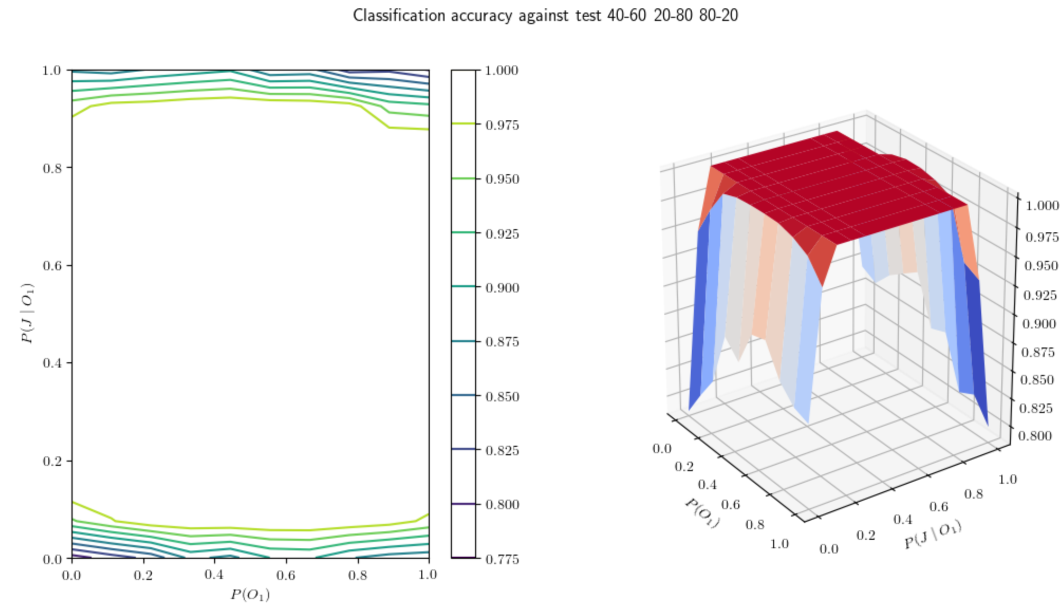

Figure A.14: The classification accuracy against `test 40-60 20-80 80-20`.

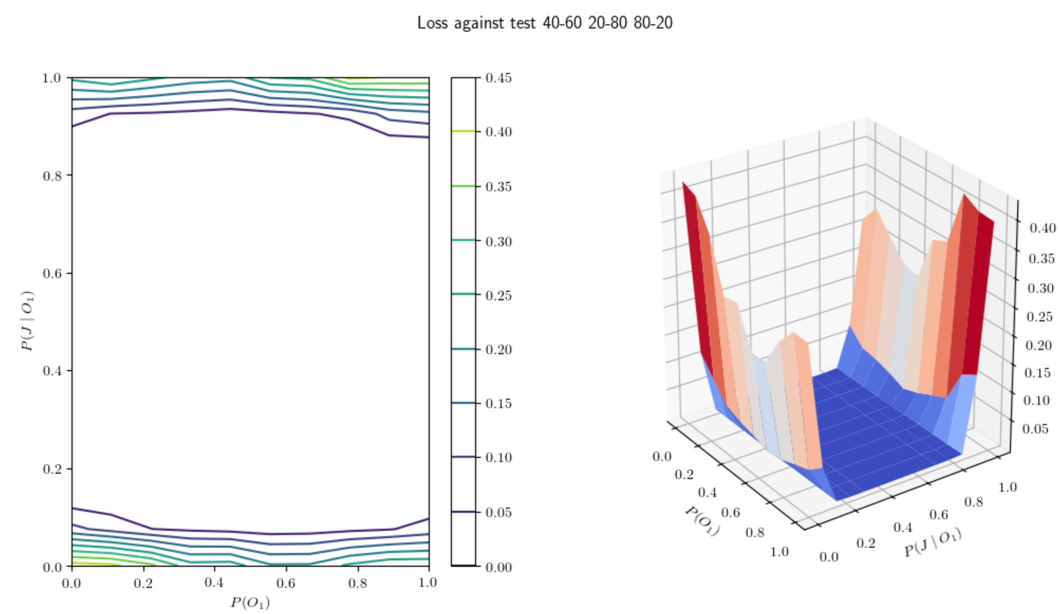

Figure A.15: The loss against `test 40-60 20-80 80-20`.

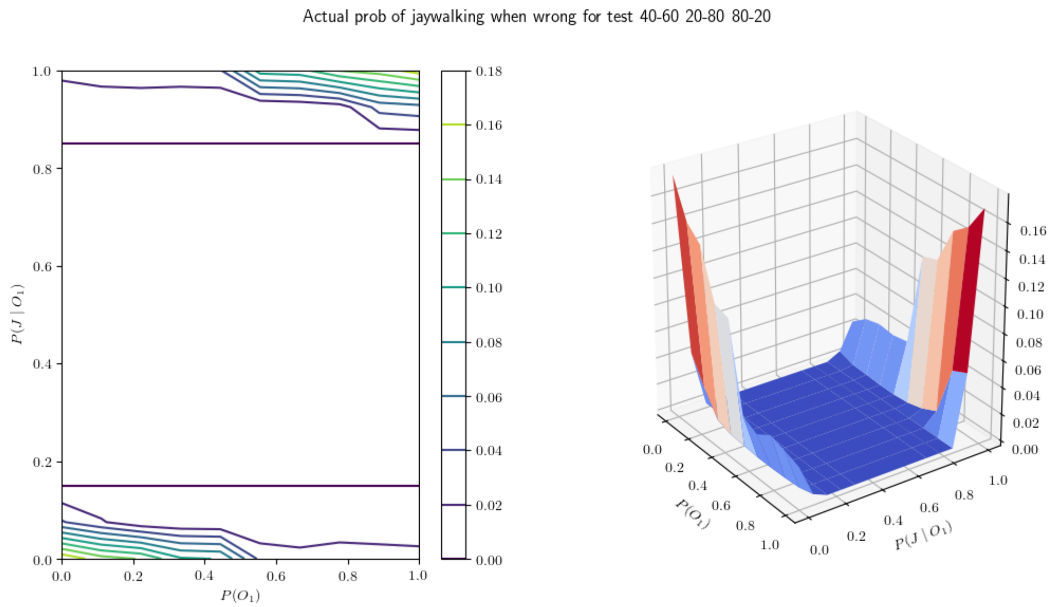

Figure A.16: The actual jaywalking probability when classified incorrectly against `test 40-60 20-80 80-20`.

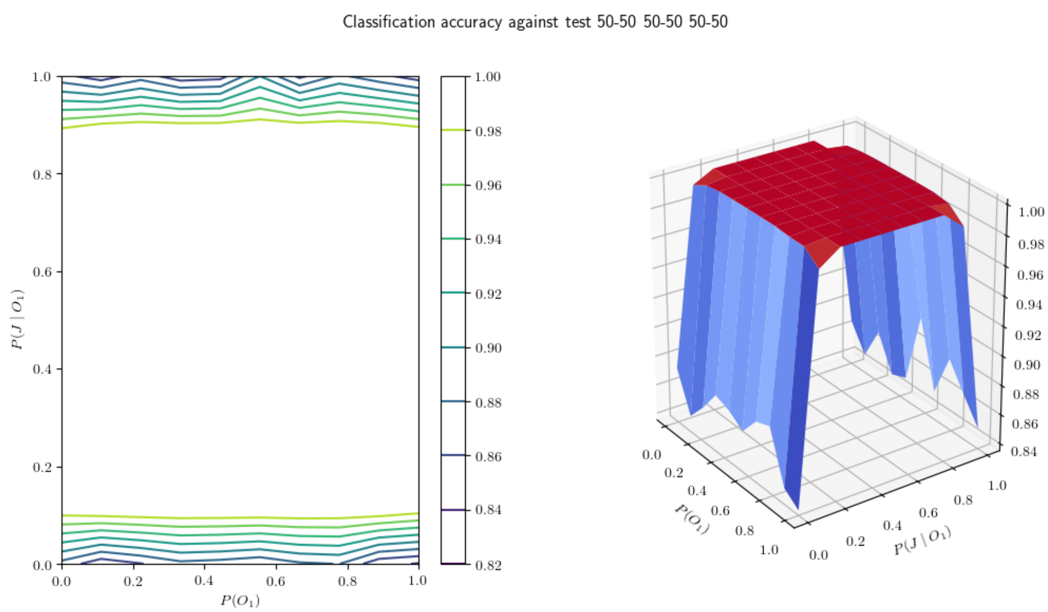

Figure A.17: The classification accuracy against `test 50-50 50-50 50-50`.

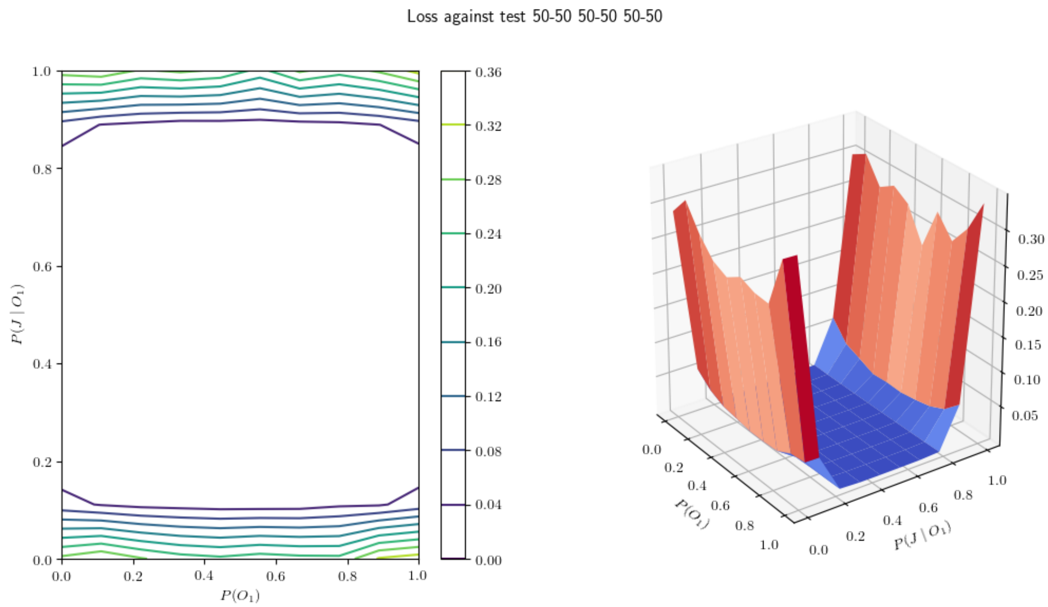

Figure A.18: The loss against `test 50-50 50-50 50-50`.

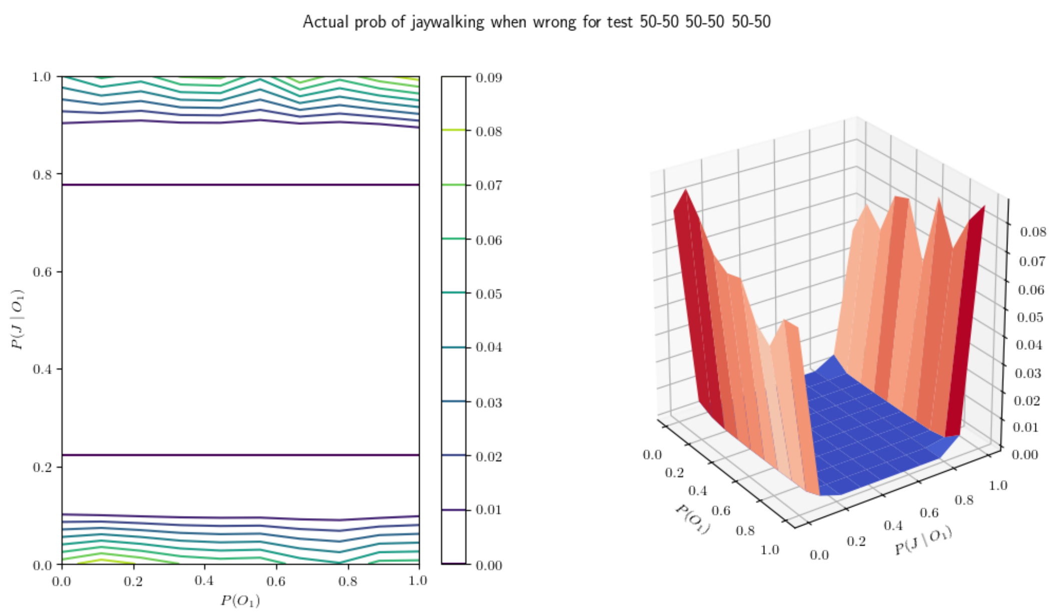

Figure A.19: The actual jaywalking probability when classified incorrectly against `test 50-50 50-50 50-50`.