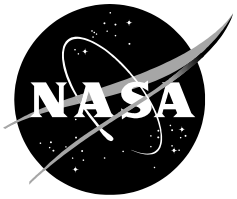
NASA/CR—20210000187



Urban Air Mobility: Predictive Modelling on the Behavior of Air-Service Bookings in Metropolitan Areas

Srushti Adesara
Harbani Jaggi
Annie Lee
Katharine Lundblad
Swetha Rajkumar

Volunteer Internship Program
Ames Research Center, Moffett Field, California

August 2020

NASA STI Program ... in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NTRS Registered and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

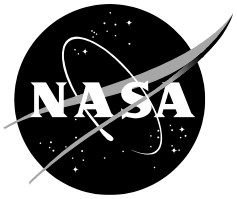
- Access the NASA STI program home page at http://www.sti.nasa.gov

- E-mail your question to help@sti.nasa.gov

- Phone the NASA STI Information Desk at 757-864-9658

- Write to:
  NASA STI Information Desk
  Mail Stop 148
  NASA Langley Research Center
  Hampton, VA 23681-2199

NASA/CR—20210000187



Urban Air Mobility: Predictive Modelling on the Behavior of Air-Service Bookings in Metropolitan Areas

Srushti Adesara
Harbani Jaggi
Annie Lee
Katharine Lundblad
Swetha Rajkumar

Volunteer Internship Program
Ames Research Center, Moffett Field, California

Prepared for the Aviation Systems Division
As part of the Volunteer Internship Program
Mentors: Aditya Das, Kee Palopo
Summer 2020

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, CA 94035-1000

August 2020

ABSTRACT

The demand for a ride to work without running into the inefficiencies of traffic in a bustling, metropolitan area beckons for a system to tackle on-the ground traffic congestion, thus fueling the need for Urban Air Mobility (UAM). The instant gratification culture, notably its effect on Generation Z, has given way to the nature of impulsive decision making. The 2020 COVID-19 crisis, with its use of tap-to-gratify technology, highlights passengers preferring late booking. This last-minute airport reorganization crisis spearheaded our research goal to create a more user-personalized assessment of cancellation rates. Our research highlights that there will be a market in the 2030s for UAM, short-distance air transportation, and the world must prepare for the challenges that come with this new market. Through a two-part model, we obtained two randomized sets of profiles of user-specific and ridership attributes and their relationship with cancellations. Recommendations for future research include factoring this personalized aspect into cancellation projections to provide reasoned user discounts as well as a remedy for the scheduling chaos that stems from the airport organizational crisis.

1. INTRODUCTION

Technology bridges the gap between an individual's desire and its fulfillment. This idea introduces the human psychological phenomenon of instant gratification: the desire for a reduction of time between wanting something and getting it. With everything becoming a click away, instant gratification has become a reality; born with these technological advancements at hand, instant gratification has become an expectation for Generation Z. Mohd Salleh's research paper *Overview of "Generation Z" Behavioural Characteristics and its Effect Towards Hostel Facility*, she asserts that this quality affects their overall behavior. Furthermore, researcher Białaszek states that this quality is directly correlated with their impulsiveness in his work *Impulsive people have a compulsion for immediate gratification—certain or uncertain.*

In response to an increase in traffic, notably in urban settings, the UAM industry works to decrease traffic on land by air-transportation. With their technological advancements, many UAM models propose to have a touch-to-gratify system, thus slowly layering in the idea of instant gratification. By 2025, Generation Z will dominate the workforce and will be primary consumers of UAM. By understanding the mental framework of this generation, we will be able to better prepare for the uncertainties that will be presented in the future. With this cumulative shift in mentality comes a problem. In the early stages of the 2020 COVID-19 Global Pandemic, uncertainty caused issues for airline planning. The International Air Transport Association highlights a 15 percent increase in passengers preferring late booking.

For the purpose of the project, two aspects of the user needed to be taken into consideration: their ridership profiles, and personal characteristics. We filtered through studies discussing individual risk assessment by utilizing summary statistics from the "Demography of Risk Aversion" by Martin Halek and Joseph G. Eisenhauer and "Can

Risk Aversion Explain Schooling Attainments? Evidence from Italy" by Christian Belzil and Marco Leonardi. In obtaining data for a rider's cancellation behavior, we utilized booking cancellation data for taxi rides in Bangalore.

This report consists of the following sections: 2) Approach, 2.1) Data Extraction, 2.2) Classification Model, 2.3) Individual Risk Assessment 3) Results, 3.1) Training Model Results, 3.2) Making Predictions 4) Discussion, 5) Conclusion.

2. APPROACH

2.1 DATA EXTRACTION

In examining the Bangalore taxi ridership data, we obtained multiple parameters: the time and date of travel, the time and date of booking, the mode of booking (desktop/mobile), and two sets of latitude and longitude from which the user's ride started and ended. We decided to not use "trip packages" or "type of travel" parameters because trips were split between three categories resembling the methods of travel (long distance, point-to-point, and hourly rental).

Long Distance	Point to Point	Hourly Rental
0	1	0
0	1	0
0	1	0

Figure 1: Taxi Ride Type Table

Due to the fact that UAM travel is targeted at urban populations, most UAM travels are meant to be short-term. Keeping this notion in mind, we extracted data points from the set solely filtering the cases in which a user travels from point-to-point.

In extracting data for the personal characteristics to evaluate individual risk assessment, we collected data which displayed the correlation between gender, employment status, education level, and income in terms of probabilities.

Using limited sources created certain holes in the initial data collection that later led to some calculated assumptions. For one, ridership and user-specific data came from two separate sources, which is why we cannot draw any relationship between the cancellation probabilities derived from ridership characteristics, or those from user characteristics; the probabilities resulting from each are independent of their models. Additionally, the use of summary statistics within risk assessment caused us to assume that the population is evenly distributed. These ramifications do not allow us to make any calls on which portion of a single demographic, gender for instance, appeared more frequent in the risk category than others. This brings relevance to needing a combined dataset, for one population, measuring these attributes so that gross assumptions like these need not to be made.

2.2 CLASSIFICATION MODEL

After data collection and analysis, we developed a classification model to understand the problem of ridership cancellations. The input layer to the model has a shape of 18 neurons and contains information about a rider's characteristics in a numerical representation. This includes the pickup and dropoff locations (in latitude and

longitude), the pickup and booking times, and the mode of booking (mobile or online website). The output layer of the model has a shape of 2 neurons, containing the probabilities that the user will fall under either the cancellation or non-cancellation class. The likelihood that a user will cancel a UAM ride solely on the details of travel is based on the probability that he/she falls in the cancellation class. The hidden layers of the model, the training parameters, and the number of filtered data points to be used for training, validation, and testing were determined after several trial and error attempts.

Our preliminary model consisted of 2 hidden dense layers, both of which contained 64 neurons. We filtered out 1000 data points from the dataset and by following a ratio of 70:20:10, reserved 700 data points for training, 200 data points for validation, and 100 data points for testing. We used the "sparse categorical cross entropy" loss function, the "Adam" optimizer with the default learning rate of 0.1, and the "accuracy" metric as training parameters. This model was trained for 100 epochs. The architecture of the first model as well as a condensed summary of the training process are featured in the figure below.

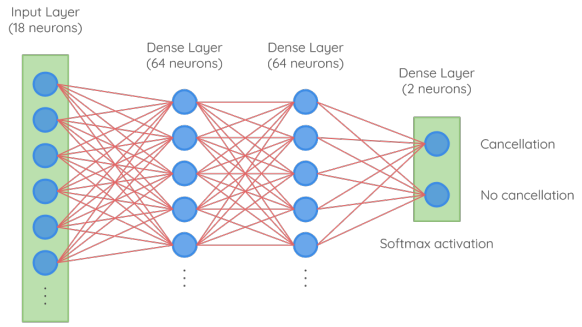


Figure 2: Initial Model Architecture, For further discussion of model structure, functions, optimizers, overfitting and underfitting, See Appendix 6.2

Two problems we encountered when training the data were overfitting and underfitting. We utilized a multitude of different strategies in order to troubleshoot.

The first strategy we employed was adding two dropout layers to the model, specifically after the first and second hidden dense layers. This technique removes certain neurons and the features that they learn from training. The second technique we used was adding an "early stopping" call-back that measures the validation loss of the model with a patience of 10. This specifies that the call-back should stop the training process once it realizes that the validation loss is not improving for 10 consecutive epochs. Both of these techniques prevent overfitting of the training data. The third technique that we used was enlarging the filtered dataset to contain 5,500 data points to increase precision. Following the ratio of 70:20:10 as used in the previous model, we used 3850 data points for training, 1100 data points for validation, and 550 data points for testing. The fourth technique we used was normalizing all of the data points. The final two techniques we implemented were lowering the learning rate of the "Adam" optimizer to 0.001 and specifying the number of times the training data should be trained per epoch. Both of these techniques increase the training time, which prevents underfitting. The final model architecture and a summary of the strategies are depicted in the figure below.

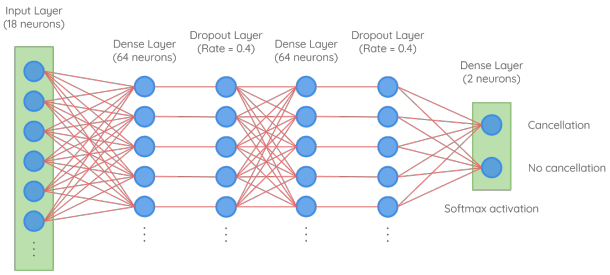


Figure 3: Final Model Architecture, For further discussion of model structure, functions, optimizers, overfitting and underfitting, See Appendix 6.2

2.3 INDIVIDUAL RISK ASSESSMENT

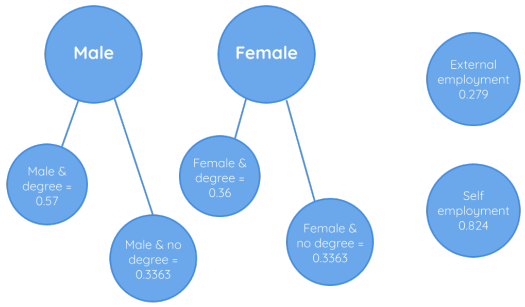


Figure 4: Individual Risk Assessment Model, For further discussion of software and model structure see Appendix 6.2

The second part of the program is the individual risk assessment model, which was created to determine the probability of a specific individual cancelling their UAM bookings last-minute.

Taking into consideration personal characteristics as shown in Figure 4 of gender, employment status, and education level, we pulled from data, making a list of probabilities of how risk-seeking the individual would be in terms of their various characteristics. For the purposes of the model, we made the calculated assumption that an individual's risk-seeking odds have a direct relation with their probability of cancelling their bookings last-minute.

To create a single probability that most appropriately encapsulated an individual's true odds of cancelling a UAM booking, we created a business rule in which the highest risk-seeking probability of the user's characteristic-individual probability list would be the resulting last-minute cancellation probability.

From a technical standpoint, a list of probabilities was assigned to each attribute of the personal profile. Mathematical analysis enabled the extraction of the maximum percentage value as the underlying risk assessing probability for this single user.

Thus, the program outputted the probability of a user cancelling their UAM booking last-minute.

3. RESULTS

3.1 TRAINING MODEL RESULTS

We received an exceptional response to our strategies in the classification model. Our training accuracy reached 79.48% while our validation accuracy reached 78.48% in the 87th epoch. Not only did our accuracy increase, but the training and validation accuracies turned out to be relatively similar. This means that there is no overfitting or underfitting of the training data.

When we used our trained model to make predictions on our testing dataset, we found that we were able to reach 80.36% accuracy, which further shows that there is no overfitting.

Due to the ramifications of time constraints and limited amount of publicly available information, we were unable to obtain better training and validation accuracies. However, additional approaches that can be taken to improve accuracy on this model include tweaking some of the hyperparameters, deepening the model, and increasing the number of data points.

3.2 MAKING PREDICTIONS

After the creation of both models, we made our own predictions by creating our own users. We generated 20 random user profiles which had ridership characteristics such as when the user booked a taxi, where they were going, etc. as well as user-specific characteristics such as gender, education, etc. Using the classification model, we generated the cancellation probability for the user based on their ridership characteristics, and using the individual risk assessment, we generate the cancellation probability for the user based on the user-specific characteristics. We later calculated the differences between these percentages and stored them in a separate table. In the next few paragraphs, we will go over one specific user profile.

For a user who created the booking on June 5th at 1:32 AM and left on June 5th at 1:47 AM from an approximated latitude of 12.99 and an approximated longitude of 77.6 to an approximated latitude of 12.98 and an approximated longitude of 77.67, the odds of them cancelling their ride based on the classification model was 99.66%. This is depicted in Figure 5 below.

Date	Booking	Pickup	Pickup Loc	Dropoff Loc	% Cancel
6/5	1:32 AM	1:47 AM	(12.99, 77.6)	(12.98, 77.67)	99.66%

Figure 5: Inputs and Outputs for Ridership Single Profile, Full 20 Profile Dataset featured in the Appendix 6.2

For this same user example, when calculating percentages based on their user-specific characteristics, which is shown below in Figure 6.

Gender	Employed	Education	% Cancel
Female	Externally	Degree	36%

Figure 6: Inputs and Outputs for single User Profile, Full 20 Profile Dataset featured in the Appendix 6.2

The cancellation rates returned from each model didn't match, which verified that cancellation rates solely based on rider specific data differed from cancellation rates that took user specific data. In order to check the validity of the different rates we received, we calculated the difference between both rates, which was about 63.66% as shown in Figure 7.

% Difference
63.66%

Figure 7: Individual Risk & Model Percent Difference, Full 20 Profile Dataset featured in Appendix 6.2

4. DISCUSSION

The percent difference of the two models is relatively high, which validates that when determining cancellations for UAM, both ridership and user profiles need to be considered in order to get an accurate assessment of the user cancellation probability. Both qualities of a user are crucial factors in determining cancellation

probability, so cancellations based on ridership and user profiles must be equally researched. With that, NASA can utilize our concept in order to come up with an accurate cancellation percentage of a specific user.

Acquiring data that takes parameters such as age, gender, financial status, and weather into consideration will allow the model to make more specific projections for individualized locations. Obtaining location specific data will allow the developers to inform UAM companies about how to best serve the needs of their users, personalizing cancellation probabilities to individuals and taking into consideration the conditions in which they are booking their rides. Specific to our risk assessment, our results show that a user of financial well-being does not factor their worries of money into their decisions, often making reckless decisions. By acknowledging the future pressure from rapid transportation from UAM that will create chaos, perhaps corporations can shift their research to this area, generating augmented datasets that combine the factors we tested on in our project. Implementing a policy in which users are charged higher rates for cancellation will help prevent the problem of individuals cancelling at the last minute. Discounted rates can be applied to low-cancellation probability users. This way, the uncertainty of flight schedules can be somewhat regulated.

For future research, we recommend NASA to refine this model using more complete datasets from sources such as Uber and taxis. In addition, we were unable to take changes from COVID-19 into consideration due to limited publicly available data. We recommend NASA to observe current cancellation patterns and UAM launch delays due to COVID-19 to calculate a more refined model of future environments.

Finally, consumer cancellation data can be extended beyond UAM. Companies like Doordash, Uber Eats, Amazon, and Airlines can use predictive modeling to predict cancellation percentages of their data so that they can better accommodate for changes.

5. CONCLUSION

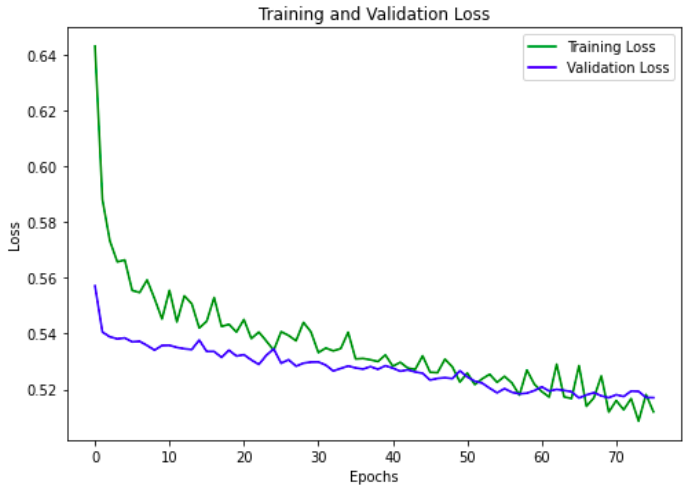
Our findings exist to aid in the process of predicting the behavior of air-service bookings in metropolitan areas.

Using studies like this one, future UAM companies can better prepare for the coming metropolitan population. Companies can foresee cancellations based on user ridership and personal characteristics and accommodate for these changes accordingly. With this information, they can alter their logistics ahead of time to minimize network disruption.

Instant gratification and its effects on the Urban Air Mobility environment must be accounted for by creating a more personalized assessment of the cancellation behavior of individual users. This way, the future UAM industry will be able to tackle the fundamental issue of scheduling chaos.

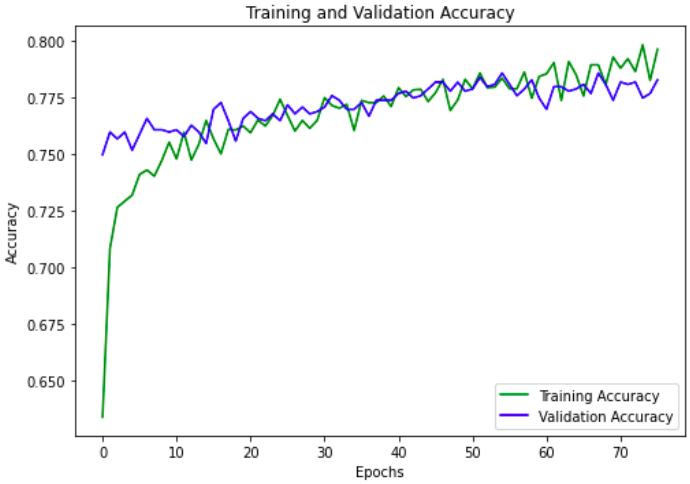
6. APPENDICES

6.1 GRAPHS



Graph 1: Software, Training and Validation Loss (Epochs vs. Loss)

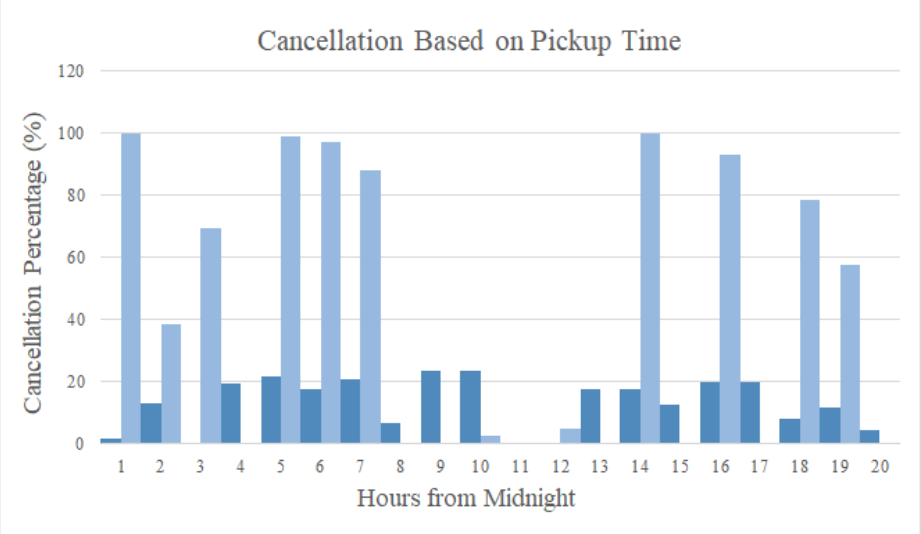
The "Training and Validation Loss" graph, on the other hand, shows the loss curves for the training and validation data as the number of epochs increase. With more training, the loss of both the training and validation data decreases, which means that the model is able to better learn and identify relationships from the data.



Graph 2: Software, Training and Validation Accuracy (Epochs vs. Accuracy)

As shown in the legend above for the "Training and Validation Accuracy" graph, the green curve corresponds to the training accuracy and the blue curve corresponds to the validation

accuracy. They are both rising to approximately 80%, and the curves become relatively similar as the number of epochs increase. This shows that there is no overfitting of the data as the training and validation accuracies are relatively similar, and that there is no underfitting of the data either because both accuracies are relatively high.



Graph 3: Results, Cancellation Based on Pickup Time (Time vs. Cancellation Probability)

As shown in the distribution of cancellation percentages for the time of day in which pickups occurred, the cancellations appear staggered, with a common trend of less cancellations from the morning till midday.

6.2 SOFTWARE

The following contains the detailed contents of the software portion of our research. All software was done in the Google Collaboratory coding environment in the Python programming language

Training Process:

1000 Data Points

100 Epochs

Adam Optimizer
↳ Learning Rate = 0.1

Accuracy Metric

Sparse Categorical
Cross Entropy Loss Function

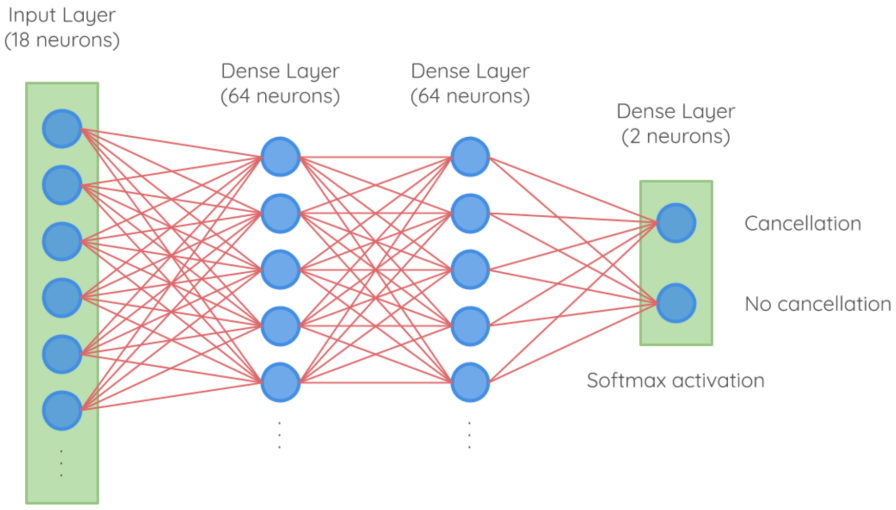


Figure 8: Training Details and Initial Model Structure, Contains layers, neurons, optimizers, metrics, and functions used

Modifications/Additions:

Two Dropout Layers
↳ Rate = 0.4

Early Stopping Call-Back
↳ Metric = Validation Loss
↳ Patience = 10

5000 Data Points

Normalization of data

Adam Optimizer
↳ Learning Rate = 0.001

Specifying Number of Batches

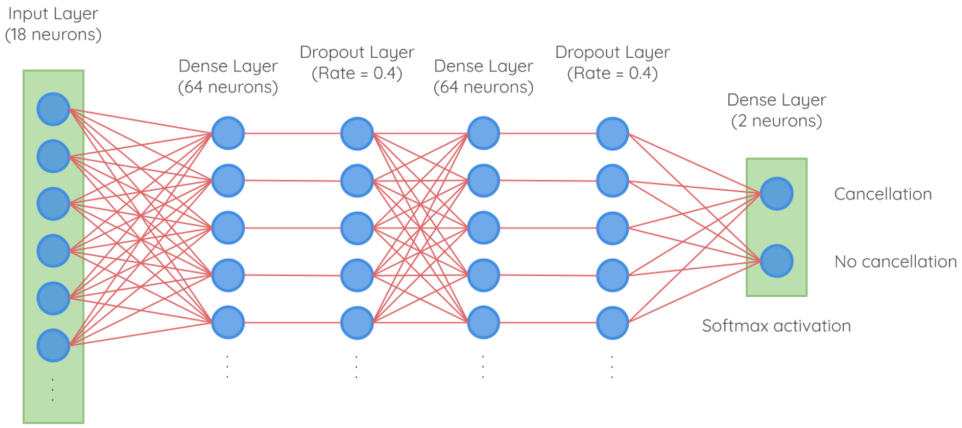


Figure 9: Additional/Modified Training Details and Model Structure, With improvement techniques, functions, and layers

```
""" Builds a model using the keras sequential API, compiles it, and prints
out the model summary. Stores it into self.model. """
def build_model(self, num_neurons, dropout_rate, num_inputs, num_outputs, loss_function, optimizer,
metrics_list):
  self.model = keras.Sequential([
    layers.Dense(num_neurons, activation = tf.nn.relu, input_shape=[num_inputs]),
    layers.Dropout(dropout_rate),
    layers.Dense(num_neurons, activation = tf.nn.relu),
    layers.Dropout(dropout_rate),
    layers.Dense(num_outputs, activation = tf.nn.softmax)
  ])
  self.model.compile(loss=loss_function, optimizer=optimizer, metrics=metrics_list)
  self.model.summary()
```

Figure 10: Software, Build Ridership Classification Model Method

```
def initialize_model(model_manager, dataset_processor):
  normed_train_data, normed_test_data, train_labels, test_labels, train_stats = dataset_processor.get_train_and_test_data("https://docs.google.com/spreadsh
            ["from_date_month", "from_date_day",
    "from_time_hours", "from_time_minutes", "from_AM", "from_PM", "online_booking",
    "mobile_site_booking", "booking_created_month", "booking_created_day", "booking_created_hour",
    "booking_created_minutes", "booking_created_AM", "booking_created_PM", "from_lat", "from_long",
    "to_lat", "to_long", "Car_Cancellation"], 0.9)


  if model_manager.load_model("cancellation_model.h5"):
    print("Loaded Model")
    prediction_list = model_manager.get_model_prediction(normed_test_data)

  else:
    print("Building Model......")
    model_manager.build_model(64, 0.4, len(normed_train_data.keys()), 2, tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
                          tf.keras.optimizers.Adam(0.001), ["accuracy"])
    model_manager.train_and_evaluate_model(normed_train_data, train_labels,
                                      0.2, 32, normed_test_data, test_labels, 100, 10)

    model_manager.save_model("cancellation_model.h5")
```

Figure 11: Software, Initializing Ridership Classification Model Method

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 64)                1216
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 64)                4160
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_5 (Dense)              (None, 2)                 130
=================================================================
Total params: 5,506
Trainable params: 5,506
Non-trainable params: 0
```

Figure 12: Model Dense Layers & Parameters This figure contains each layer type, neuron counts, and parameter count summaries and represents the structure of the developed model using machine learning tools and google colaboratory.

```
""" Fits the training data to the model, displays the validation
and training accuracies, and evaluates the model using the test dataset and
displays the accuracies and predictions. If self.model is None, prints out
'No available model to use.' """
def train_and_evaluate_model(self, normed_train_data, train_labels, validation_split, num_batches,
normed_test_data, test_labels, num_epochs, patience):

  if self.model==None:
    print("No available model to use")
    return

  history = self.model.fit(normed_train_data, train_labels, epochs=num_epochs, validation_split = validation_split,
      steps_per_epoch = math.ceil(len(normed_train_data)/num_batches), verbose=1, callbacks =
      [keras.callbacks.EarlyStopping(monitor='val_loss', patience=patience)])

  num_epochs = len(history.history['loss'])
  epochs_range = range(0, num_epochs)
  train_loss = history.history['loss']
  val_loss = history.history['val_loss']
  train_accuracy = history.history['accuracy']
  validation_accuracy = history.history['val_accuracy']


  plt.figure(figsize=(8, 12))

  plt.subplot(2, 1, 1)
  plt.plot(epochs_range, train_accuracy, "g", label='Training Accuracy')
  plt.plot(epochs_range, validation_accuracy, "b", label='Validation Accuracy')
  plt.legend(loc='lower right')
  plt.title('Training and Validation Accuracy')
  plt.xlabel('Epochs')
  plt.ylabel('Accuracy')

  plt.subplot(2, 1, 2)
  plt.plot(epochs_range, train_loss, "g", label='Training Loss')
  plt.plot(epochs_range, val_loss, "b", label='Validation Loss')
  plt.legend(loc='upper right')
  plt.title('Training and Validation Loss')
  plt.xlabel('Epochs')
  plt.ylabel('Loss')
  plt.show()

  test_loss, test_accuracy = self.model.evaluate(normed_test_data, test_labels, steps = math.ceil(len(normed_test_data)/num_batches), verbose=0)
  print("The accuracy on the test dataset is: " + str(test_accuracy))
```

Figure 13: Software, Train and Evaluate Model Class

```
Epoch 1/100
157/157 [==============================] - 0s 2ms/step - loss: 0.6491 - accuracy: 0.6296 - val_loss: 0.5596 - val_accuracy: 0.7497
Epoch 2/100
157/157 [==============================] - 0s 2ms/step - loss: 0.5855 - accuracy: 0.7148 - val_loss: 0.5430 - val_accuracy: 0.7568
Epoch 3/100
157/157 [==============================] - 0s 2ms/step - loss: 0.5780 - accuracy: 0.7216 - val_loss: 0.5423 - val_accuracy: 0.7588
 .
 .
 .
Epoch 85/100
157/157 [==============================] - 0s 2ms/step - loss: 0.5139 - accuracy: 0.7931 - val_loss: 0.5190 - val_accuracy: 0.7738
Epoch 86/100
157/157 [==============================] - 0s 2ms/step - loss: 0.5087 - accuracy: 0.7980 - val_loss: 0.5191 - val_accuracy: 0.7748
Epoch 87/100
157/157 [==============================] - 0s 2ms/step - loss: 0.5110 - accuracy: 0.7948 - val_loss: 0.5177 - val_accuracy: 0.7848
```

Figure 14: Epochs 1-3 and 85-87, Depiction of the training process of the classification model

```
""" Loads the model saved in the specified filepath and saves it in
self.model. Returns true if the model was successfully loaded, false
otherwise """
def load_model(self, filepath):
  try:
    self.model = models.load_model(filepath)
    self.model.summary()
    return True
  except:
    return False


""" Saves the model in the extension file-path in the current directory.
Returns true if the model was successfully saved, false otherwise """
def save_model(self, filepath):
  try:
    self.model.save(filepath)
    return True
  except:
    return False
```

Figure 15: Loading and Saving of the trained model

```
""" Determines the final cancellation likelihood of a user profile list using
the specific user profile and the model prediction of a user's cancellation
rate based off of certain ridership data """
def get_user_cancellation_likelihood(self, user_profile_list):

  user_cancellation_list = []
  user_cancellation_probability = ""
  for user_profile in user_profile_list:
    gender = user_profile[0]
    employement_status = user_profile[1]
    education = user_profile[2]
    risky_probabilities=[]
    if gender=="male" and education=="degree":
      risky_probabilities.append(0.57)
    elif gender=="female" and education=="degree":
      risky_probabilities.append(0.36)
    else:  # means they are low-income
      risky_probabilities.append(0.3363)
    if employement_status=="self_employment":
      risky_probabilities.append(0.824)
    elif employement_status=="external_employment":
      risky_probabilities.append(0.279)

    if risky_probabilities[0]>risky_probabilities[1]:
      user_cancellation_list.append(risky_probabilities[0]*100)
    else:
      user_cancellation_list.append(risky_probabilities[1]*100)

  return user_cancellation_list
```

Figure 16: Individual Risk Assessment Method

```
"""
Software Steps - Will be converted into main code that program executes
1. Get user input and generate a full user profile, or generate a random set
of profiles which the model will then make predictions on
2. Load the model. If the model cannot be loaded, process the dataset by
converting the categorical data into numbers & normalizing the entire
dataset. Separate the dataset into train & test datasets (input factors) &
also add train labels & test labels (output factors). Build & train the model,
evaluate it, and then save the model
3 Use the model to make predictions and print the profile and
cancellation percentages out
"""

profile_list = get_user_profiles(Profile_Generator())
model_manager = Model_Manager()
individual_risk_assessment = Individual_Risk_Assessment()
dataset_processor = Dataset_Processor()
train_stats = initialize_model(model_manager, dataset_processor)
process_profiles(profile_list, model_manager, individual_risk_assessment, dataset_processor, train_stats)
```

Figure 17: Main Running Code, Software Steps

```
--
GENDER                EDUCATION              EMPLOYMENT          RISK_ASSESSMENT%
female       external_employment                 degree                    36.0%
female           self_employment              no_degree                    82.4%
  male           self_employment              no_degree                    82.4%
female       external_employment                 degree                    36.0%
  male       external_employment              no_degree                   33.63%
female           self_employment              no_degree                    82.4%
  male           self_employment                 degree                    82.4%
  male       external_employment                 degree                    57.0%
female           self_employment                 degree                    82.4%
female       external_employment              no_degree                   33.63%
female           self_employment              no_degree                    82.4%
female           self_employment                 degree                    82.4%
  male           self_employment              no_degree                    82.4%
  male           self_employment              no_degree                    82.4%
  male           self_employment              no_degree                    82.4%
female           self_employment                 degree                    82.4%
female           self_employment                 degree                    82.4%
  male           self_employment                 degree                    82.4%
female           self_employment                 degree                    82.4%
female       external_employment                 degree                    36.0%
```

Figure 18 Results, Full 20 Profile Risk Assessment Output This figure illustrates the user-specific cancellation predictions that the model made for each of the 20 corresponding user profiles profiles.

FROM_TIME	BOOKING_CREATED_TIME	FROM_POINT	TO_POINT	RIDER_CANCELLATION%
6/5 1:47 AM	6/5 1:32 AM	(12.99, 77.6)	(12.98, 77.67)	99.66%
5/26 12:59 PM	5/26 12:44 PM	(13.2, 77.71)	(12.91, 77.61)	38.41%
5/12 12:25 AM	5/12 12:10 AM	(12.98, 77.71)	(13.03, 77.55)	69.53%
7/19 7:18 PM	7/19 7:3 PM	(13.01, 77.56)	(13.2, 77.71)	0.0%
7/7 9:43 PM	7/7 9:28 PM	(12.95, 77.7)	(13.02, 77.55)	99.04%
2/26 5:35 PM	2/26 5:20 PM	(12.98, 77.57)	(12.85, 77.66)	96.95%
7/8 8:46 PM	7/8 8:31 PM	(13.05, 77.51)	(13.01, 77.56)	88.12%
7/11 6:38 AM	7/11 6:23 AM	(13.01, 77.56)	(13.2, 77.71)	0.01%
2/18 11:31 PM	2/18 11:16 PM	(13.03, 77.55)	(12.98, 77.65)	0.05%
4/5 11:39 PM	4/5 11:24 PM	(12.95, 77.71)	(13.2, 77.71)	2.5%
6/5 12:32 AM	6/5 12:17 AM	(13.01, 77.56)	(13.2, 77.71)	0.0%
5/13 12:19 AM	5/13 12:4 AM	(13.2, 77.71)	(13.03, 77.55)	4.76%
3/11 5:49 PM	3/11 5:34 PM	(13.04, 77.61)	(13.03, 77.6)	0.31%
6/23 5:28 PM	6/23 5:13 PM	(12.94, 77.68)	(12.95, 77.7)	100.0%
2/28 12:47 PM	2/28 12:32 PM	(13.2, 77.71)	(12.91, 77.61)	0.52%
2/6 7:55 PM	2/6 7:40 PM	(12.94, 77.68)	(12.95, 77.7)	92.94%
1/14 7:45 PM	1/14 7:30 PM	(13.2, 77.71)	(13.05, 77.48)	0.0%
6/1 7:57 AM	6/1 7:42 AM	(13.04, 77.61)	(13.03, 77.6)	78.46%
7/29 11:45 AM	7/29 11:30 AM	(12.97, 77.6)	(12.92, 77.61)	57.73%
3/12 4:30 AM	3/12 4:15 AM	(13.2, 77.71)	(13.01, 77.66)	0.03%

Figure 19: Results, Full 20 Profile Ridership Classification Output This figure illustrates the rider cancellation predictions that the model made for each of the 20 randomly generated ridership profiles.

PERCENT_DIFFERENCE%
63.66%
43.99%
12.87%
36.0%
65.41%
14.55%
5.72%
56.99%
82.35%
31.13%
82.4%
77.64%
82.09%
17.6%
81.88%
10.54%
82.4%
3.94%
24.67%
35.97%

Figure 20: Results, Percent Difference between Ridership Classification and Risk Assessment Percentages This figure shows the differences between the percentages generated by the classification model and the individual risk assessment for each of the 20 profiles.

Test Dataset Results

RIDER_CANCELLATION%	CANCELLED/NOT CANCELLED
0.55%	0
96.08%	1
0.0%	0
99.99%	1
96.57%	1
0.74%	1
100.0%	1
0.0%	1
0.34%	0
97.42%	0
4.13%	0
99.0%	1
99.88%	1
0.01%	0
99.65%	1
99.99%	1
0.0%	0
0.0%	0
0.16%	0
45.57%	1

Figure 21: Test Dataset Prediction Results: This figure represents the cancellation percentages outputted by the model for the first 20 rows of the test dataset, and their corresponding labels or "ground-truth," which measures how valid the predictions of the model are. For example, in the first row of this table, the cancellation label is equal to 0, which tells us that the user cancelled their ride. When we look at the model prediction for the user, we see that the cancellation prediction was 0.55%, which shows that the model predicted that the user will most likely not cancel. This demonstrates that the model was accurate in this prediction. There are some rows that do not have accurate predictions and that is because only 80% accuracy was achieved for the test dataset.

6.3 TECHNICAL TERMS

Dropout layers - Layers that are used to "drop out" or remove certain neurons and the features that they learn from training, which prevents the model from overfitting, or memorizing all of the features of the train dataset.

"Early Stopping" call-back - A call-back that is used to stop training early if the model is not improving its ability to make generalized predictions. This prevents overfitting as it stops the training process once it realizes that the model is only memorizing the training data. This call-back takes a measuring metric and a patience factor as parameters. The measuring metric specifies what the call-back should measure and the patience factor specifies the number of consecutive epochs for which to check if the measuring metric is improving. In our model, we specified that the model should measure the validation loss with a patience of 10. This specifies that the call-back should stop the training process once it realizes that the validation loss is not improving for 10 consecutive epochs.

Epoch - One scan through all the batches of the training dataset using weights for each neuron to calculate the output. In the case of classification, the output represents the probability distribution that a set of inputs falls into each of the specified classes.

Learning Rate - A hyperparameter used in the optimizer algorithm that specifies the speed at which the model learns the dataset. Specifically, the learning rate specifies the step size that the model takes towards reaching the optimal weights for the neurons. A smaller learning rate means a smaller step size and a larger learning rate means a larger step size. It usually takes some trial and error to determine the best learning rate for a model. In our model, we used a learning rate of 0.001.

Loss Function - A function that is used after each epoch to evaluate how inaccurate the predictions of the model are in comparison to the actual output. The loss function used in our classification model is the "sparse categorical cross entropy" loss function.

Metric - A parameter that is used to measure the training, validation, and testing accuracies of the model. The most commonly used metric in classification is the "accuracy" metric, however it should only be applied to models when the dataset being used has a relatively even distribution of data points for each class. Since our filtered dataset contained an even distribution of data points for the cancellation and non-cancellation classes, we were able to use this metric in our model.

Normalization - A technique used to scale the values in the data set between the ranges of 0-1 to reach a more robust relationship between points during the training process. We used standardization to achieve normalization of our filtered dataset.

Optimizer - An algorithm that is used once the loss is evaluated to reassign weights to the neurons to improve the accuracy rates. The optimizer takes a learning rate as a hyperparameter, and if not specified, uses the default of 0.1. In our model, we used the "Adam" optimizer with a learning rate of 0.001.

Overfitting - A condition where the model is just memorizing the training data with each epoch and is not able to generalize well on new data, limiting its use in making real-world predictions. This can be detected if the validation accuracy is low in comparison to the training accuracy.

Testing Accuracy - The accuracy that the model achieves on the testing dataset after the training process which reflects how well the model is able to make real-world predictions.

Training Accuracy - The accuracy that the model achieves on the data used for training the model at the end of each epoch. If the training accuracy is really low, that is a sign of underfitting.

Underfitting - A condition where the model is not able to sufficiently learn from the training data. This can be detected if the training and validation accuracies are really low.

Validation Accuracy - The accuracy that the model achieves on the validation data, which is data that is a part of the training dataset that the model does not train on, at the end of each epoch. This

accuracy depicts how well the model is able to generalize. If the validation accuracy is low in comparison to the training accuracy that is a sign of overfitting.

REFERENCES

"Air Travel Consumer Reports for 2019." *US Department of Transportation*, 2019, www.transportation.gov/airconsumer/air-travel-consumer-reports-2019.

"Airport Activity Survey with Form 1800-31 – Airports." *Airport Activity Survey with Form 1800-31*, 19 Feb. 2020, www.faa.gov/airports/planning_capacity/passenger_allcargo_stats/activity_survey.

Baumgart, Leigh A., et al. "Emergency Management Decision Making during Severe Weather." *Weather and Forecasting*, American Meteorological Society, 1 Dec. 2008, journals.ametsoc.org/waf/article/23/6/1268/39160.

"Data and Statistics for Airport Programs." *Data and Statistics for Airport Programs*, 17 June 2020, www.faa.gov/airports/resources/data_stats.

"Decision Making under Time Pressure, Modeled in a Prospect Theory Framework." *PubMed Central (PMC)*, 1 July 2012, www.ncbi.nlm.nih.gov/pmc/articles/PMC3375992/#:%7E:text=The%20authors%20concluded%20that%20increases,decision%20making%20is%20more%20complex.&text=The%20effects%20of%20time%20pressure%20on%20individual%20choice,take%20place%20through%20three%20mechanisms.

"Generating Joint Density Function in Excel." *Super User*, 13 Aug. 2014, superuser.com/questions/796541/generating-joint-density-function-in-excel.

"GPS Download." *IATA*, www.iata.org/en/publications/gps-download/.

"How to Calculate Joint, Marginal, and Conditional Probabilities Using Excel Dcarroll - Purdue High Impact Weather Lab." *How to Calculate Joint, Marginal, and Conditional Probabilities Using Excel Dcarroll*, 2012, sites.google.com/a/himpactwxlab.com/www/home/archive/forecast-verification---eas-591---fall-2012/hw-2/how-to-calculate-joint-marginal-and-conditional-probabilities-using-excel-dcarroll.

Lovell, David, et al. Calibrating Aggregate Models of Flight Delays and Calibrating Aggregate Models of Flight Delays and Cancellation Probabilities at Individual Airports Cancellation Probabilities at Individual Airports. 2007.

M, Yu, et al. "Effects of Time Pressure on Behavioural Decision Making in Natural Disasters: Based on an Online Experimental System." *Journal of Geography & Natural Disasters*, vol. 08, no. 01, 2018. *Crossref*, doi:10.4172/2167-0587.1000220.

McGovern, Amy, et al. "Using Artificial Intelligence to Improve Real-Time Decision-Making for High-Impact Weather." *Bulletin of the American Meteorological Society*, American Meteorological Society, 1 Oct. 2017, journals.ametsoc.org/bams/article/98/10/2073/70032.

"NYCdata: John F. Kennedy (JFK) International Airport - Statistics." *New York City (NYC) John F. Kennedy International Airport (JFK) - Statistics 2003-2017*, 2018, www.baruch.cuny.edu/nycdata/travel/jfk-stats.htm.

"Passenger Boarding (Enplanement) and All-Cargo Data for U.S. Airports." *Passenger Boarding (Enplanement) and All-Cargo Data for U.S. Airports – Airports*, 6 July 2020, www.faa.gov/airports/planning_capacity/passenger_allcargo_stats/passenger/.

ShawShare, Brent, et al. "Keeping Things in Perspective: Weather vs. Climate in Operational Decision Making." *PrecisionAg*, 3 June 2019, www.precisionag.com/market-watch/keeping-things-in-perspective-weather-vs-climate-in-operational-decision-making/.

Sivle, Anders Doksæter, and Stein Dankert Kolstø. "RMetS Journals." *Royal Meteorological Society (RMetS)*, John Wiley & Sons, Ltd, 9 Dec. 2016, rmets.onlinelibrary.wiley.com/doi/full/10.1002/met.1588.

"Worldwide Airport Delays and Airport Status." *FlightAware*, 2020, uk.flightaware.com/live/airport/delays.