Porto, Portugal Taxi Trip Time Prediction

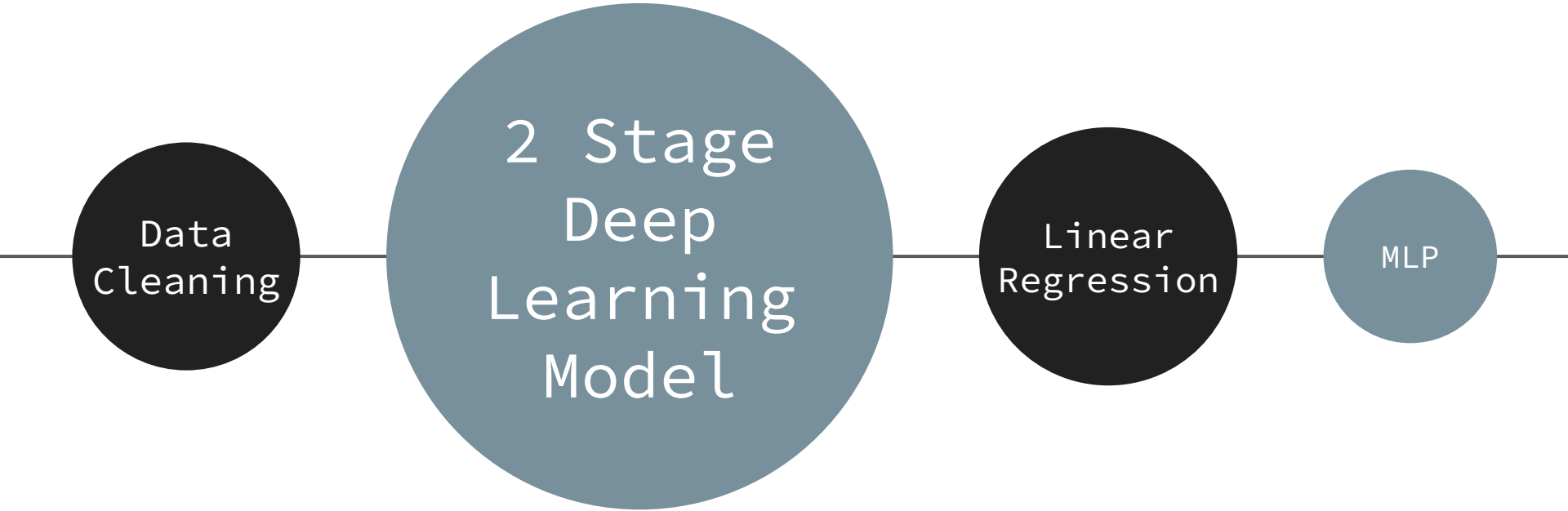
Yash Puneet

Contents

- Team Introductions
- Key Components of the model
  - 2 Stage Predictor used to predict additional feature (velocity) for final trip duration prediction
  - Iterative approach with multiple experiments allowing improvements in architecture design and hyperparameter tuning
  - ReLU Activation, 1 Dimensional Batch Normalization, and Dropout Layers
- What I learned through the process
- Future development and Goals

PyTorch

Key Words


Data Cleaning
2 Stage Deep Learning Model
Linear Regression
MLP

Introduction

Team Introduction

- Yash Puneet: 3rd Year Computer Engineering Major and Psychology minor interested in adapting animal cognitive models to advance robotics.
- Goals for this project
  - Coming into this class with no machine learning experience, my primary goal was to learn how research and project development is done in the field
  - Another goal I had was attempting to implement human cognitive models for the prediction task but this was not realised due to time constraints. I plan to continue experimenting with this aspect in the summer.

Methodology

Data Processing

Data Processing was a major component of the project and was the backbone allowing other aspects of the project to actually be realised. The next few slides will discuss how data processing was done, especially in the context of the following points:

- Exploration and Cleaning of the Data Set
  - Column Information Analysis
  - Distribution Visualization and trimming
- Extracting Additional Information
  - Information that can be extracted from training and testing set
  - Information only available from the training set allowing the 2 stage model

Dataset Column Analysis

- Provided Data had 9 Column out of which a few columns were dropped after analysis
  - Dropped Columns: DAY_TYPE, MISSING_DATA
- ORIGIN_CALL and ORIGIN_STAND null values replaced with 0 to allow use in model

	TRIP_ID	CALL_TYPE	ORIGIN_CALL	ORIGIN_STAND	TAXI_ID	TIMESTAMP	DAY_TYPE	MISSING_DATA	POLYLINE
0	1372636858620000589	C	NaN	NaN	20000589	1372636858	A	False	[[-8.618643,41.141412], [-8.618499,41.141376],[...
1	1372637303620000596	B	NaN	7.0	20000596	1372637303	A	False	[[-8.639847,41.159826], [-8.640351,41.159871],[...
2	1372636951620000320	C	NaN	NaN	20000320	1372636951	A	False	[[-8.612964,41.140359], [-8.613378,41.14035],[-...
3	1372636854620000520	C	NaN	NaN	20000520	1372636854	A	False	[[-8.574678,41.151951], [-8.574705,41.151942],[...
4	1372637091620000337	C	NaN	NaN	20000337	1372637091	A	False	[[-8.645994,41.18049], [-8.645949,41.180517],[-...

Dataset Distribution Analysis and Trimming

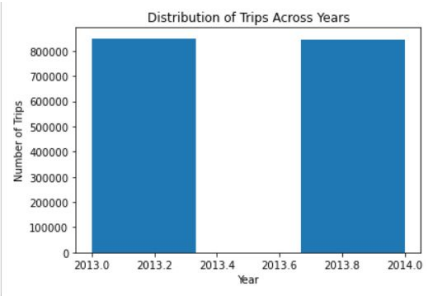
- Trip duration distribution was highly left-skewed because some trips were unreasonably long
- Shorted trips were still acceptable in the dataset
- The dataset was normalized by mean and standard deviation of duration data
- Velocity trimming was done in the same way to remove unreasonably fast trips
- Rows with missing data were also removed

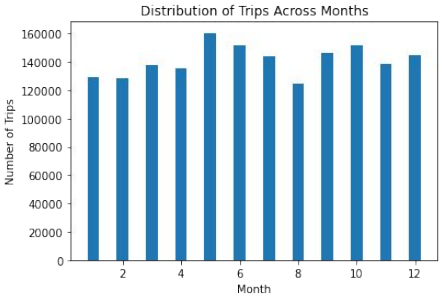
Column	Mean	Median	Standard Deviation	Columns Remaining /1710670
Duration (LEN)	716.426	600.0	684.751	1692771
Velocity	6.875	6.150	25.872	1691242

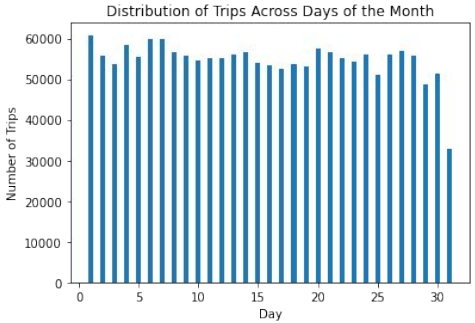
Training and Testing Data Extraction

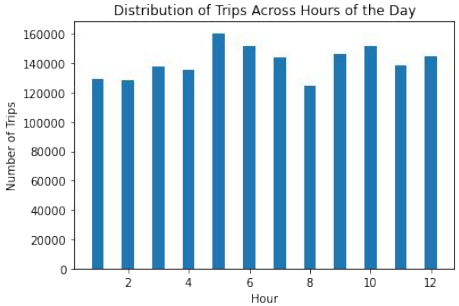
- Additional Data was extracted using the Timestamp feature to obtain the following about trip start time:
  - Year
  - Month
  - Week
  - Day
  - Hour

YR	MON	DAY	HR	WK
2013	7	1	0	0
2013	7	1	0	0
2013	7	1	0	0
2013	7	1	0	0
2013	7	1	0	0



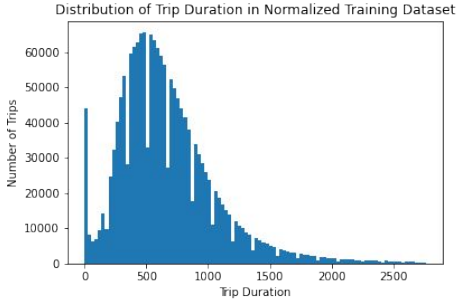




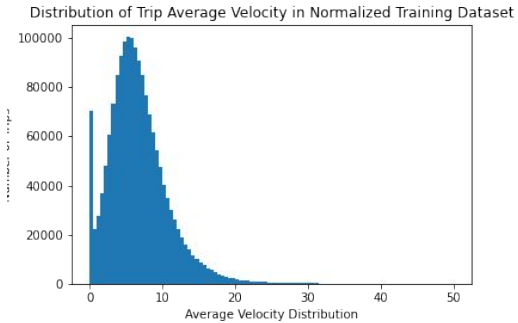


2 Stage Training Target Extraction

- Target Extraction is essential for training a model and two targets were extracted from POLYLINE for the 2 stage model.
- LEN: Trip Duration calculated by multiplying the (number of points -1) by 15s
- VELOCITY: Trip average velocity calculated by using displacement between polyline coordinates.

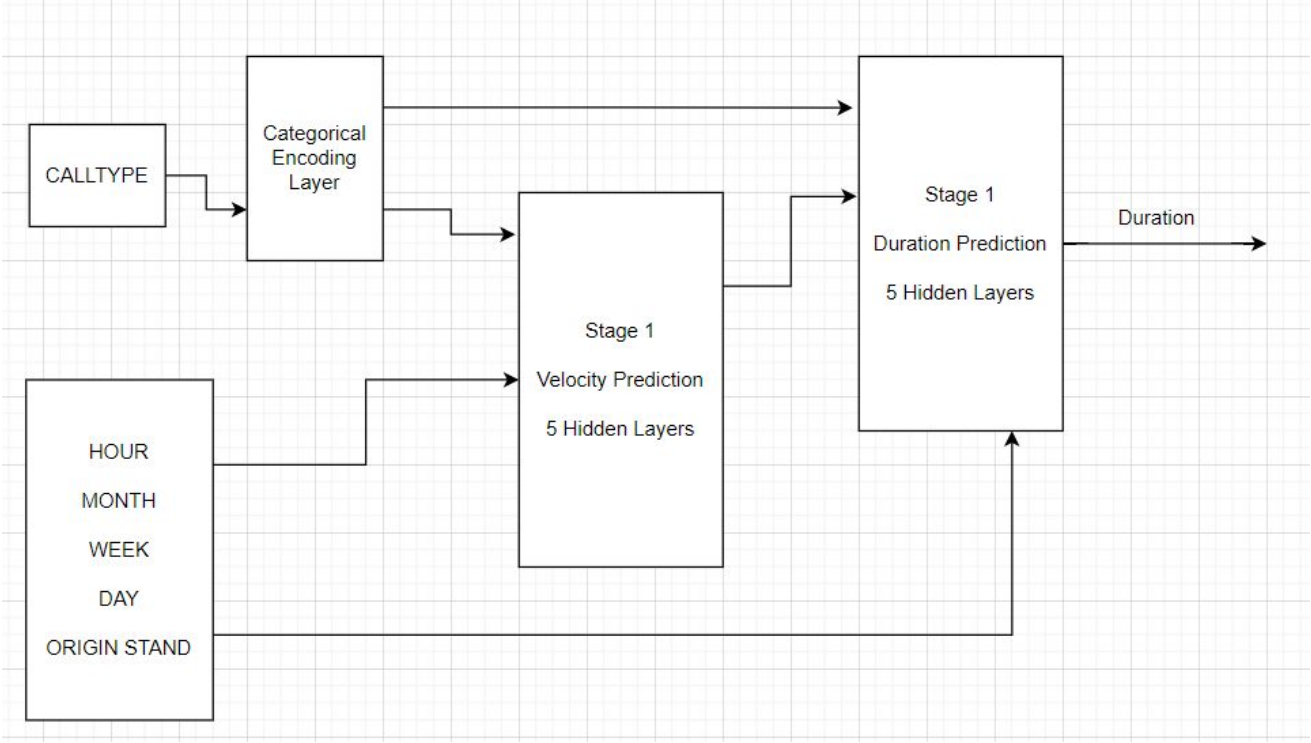


POLYLINE	LEN
[[-8.618643,41.141412],[-8.618499,41.141376],[...	330
[[-8.639847,41.159826],[-8.640351,41.159871],[...	270
[[-8.612964,41.140359],[-8.613378,41.14035],[-...	960
[[-8.574678,41.151951],[-8.574705,41.151942],[...	630
[[-8.645994,41.18049],[-8.645949,41.180517],[-...	420



	velocity
0	11.937889
1	14.555896
2	11.808893
3	5.345865
4	10.378641

Final Deep Learning Model



Layer Dimensions:

6 or 7 -> 128

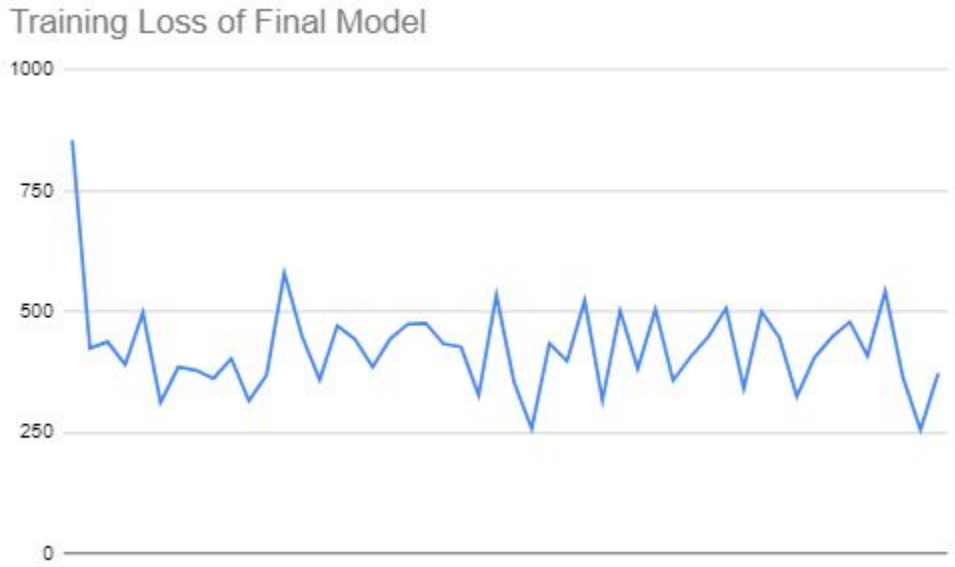
128 -> 64

64 -> 16

16 -> 8

8 -> 1 (output)

Final Deep Learning Model RMSE Curve



Summary of Engineering Tricks

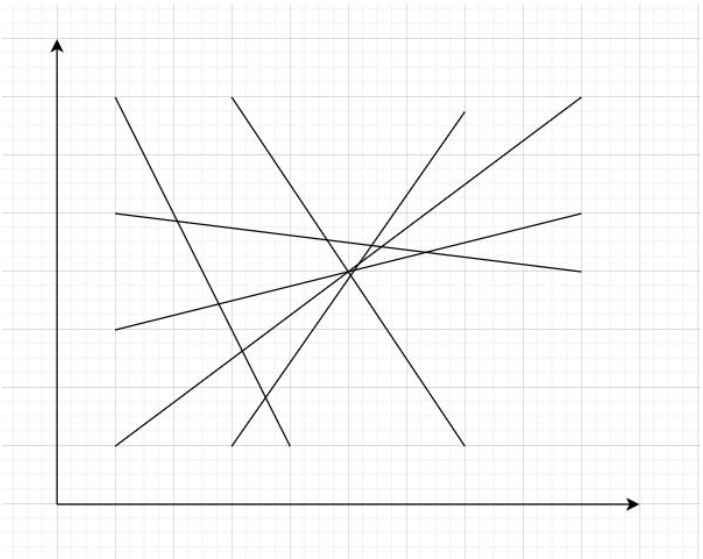
Technique	Use
Imputation	Missing data was initially replaced by 0's but research of imputation techniques revealed that numerical imputation (with most common value) would yield better results in some cases
Categorical Encoding	Used on Call_Types to convert the Letter defined column to numerical values that the model can better interpret.
Feature Splitting	Timestamp was split as described in **Slide 10**
Outlier Handling	Duration and Velocity Outlier values derived from the POLYLINE were removed by clipping based on deviation from the mean as discussed in **Slide 9**

Experiments

Experiment 1: Simple Multiple Linear Regression

General Idea:

- Multiple Features split the space into sectors dividing the data points into groups
- Used one Hyperplane per feature used as depicted in the graph
- Used Sigmoid and ReLU Activation Functions



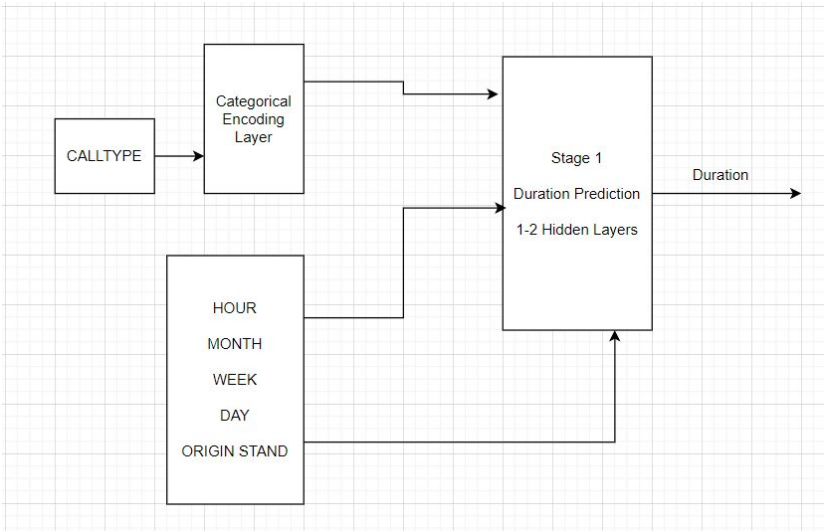
Performance:

- Bad Performance
  - Did not converge
  - Training Loss RMSE: 1382
- Pros: Comparatively fast training time

Experiment 2: Shallow Fully Connected Layer Approach

General Idea:

- 1-2 hidden layers to prevent overfitting
- Experimented with Adding Dropout layers and Batch Normalization
    - Best Performance with 0.3 dropout
    - 1 Dimensional Batch Normalization between the layers



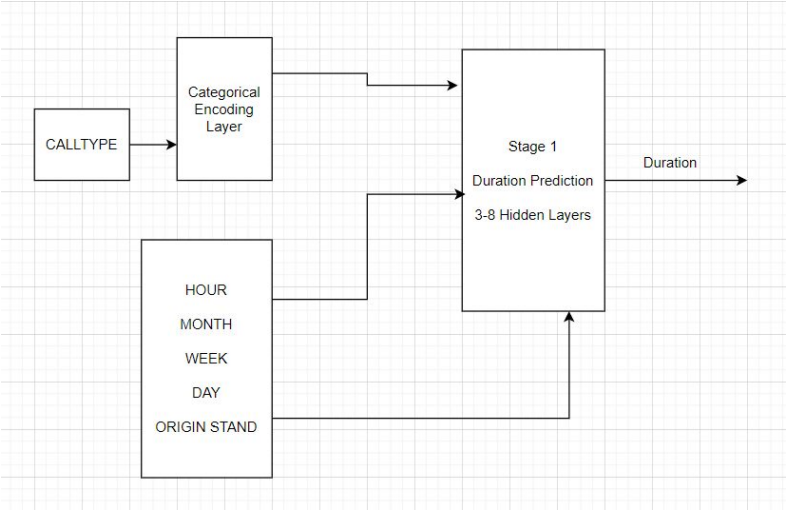
Performance:

- Better Performance but still struggled to converge - approximately 140 epochs to start converging
- Best RMSE Training Loss: 590
- Best RMSE Test Loss: 850

Experiment 3: Deeper Fully Connected Layer Approach

General Idea:

- 3-8 Hidden layers with 1 Dimensional Batchnorm layer every 2 layers
- Dropout layer after each hidden layer
    - Best Performance Dropout : 0.2
- Experimented with different layer depths with overfitting occurring after 5 layers
- ReLU Activation after every layer



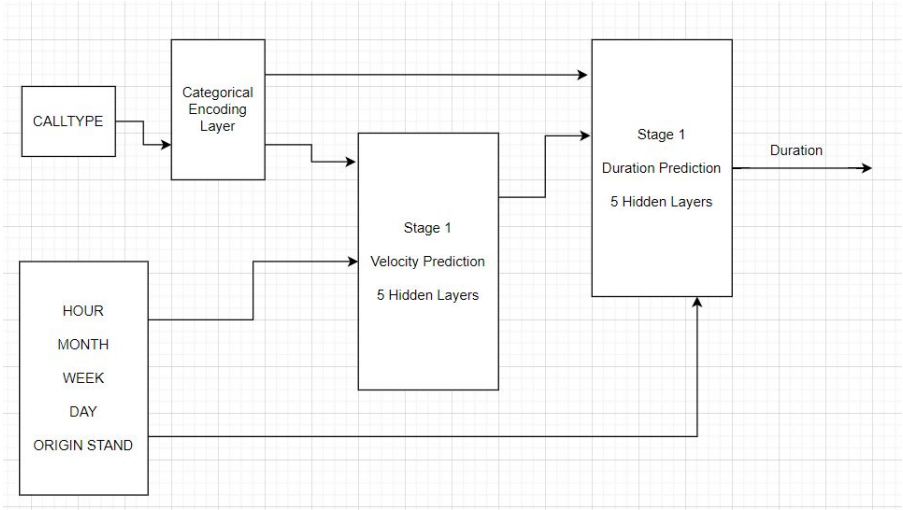
Performance:

- Long training time with approximately 7 minutes per epoch for 30-40 epochs to convergence
- Best RMSE Training Loss: 360
- Best RMSE Testing Loss: 820

Experiment 4: 2 Stage Model Approach

General Idea:

- 5 Hidden layers with 1 Dimensional Batchnorm layer every 2 layers (found to be optimal in experiment 2)
- Added velocity as an additional layer after prediction in stage 1
- Dropout layer after each hidden layer
  - Best Performance Dropout : 0.5
- ReLU Activation after every layer
- Layer Dimensions:
  - 6 or 7 -> 12
  - 12 -> 16
  - 16 -> 20
  - 20 -> 24
  - 24 -> 1 (output)



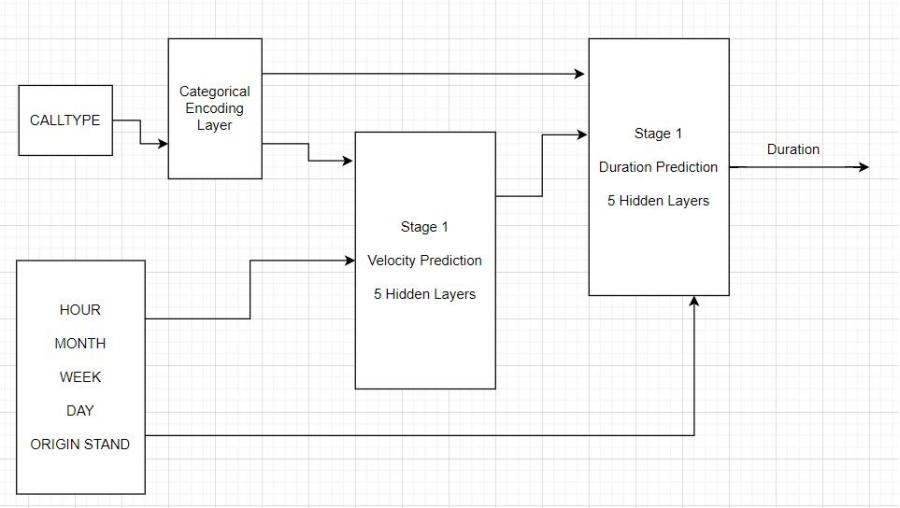
Performance:

- Compounded training time with approximately 5 minutes per epoch for 30-50 epochs to convergence for not much improvement over previous 1 stage model
- Best RMSE Training Loss: 350
- Best RMSE Testing Loss: 805

Experiment 5: 2 Stage Model with Higher Dimensions

General Idea:

- 5 Hidden layers with 1 Dimensional Batchnorm layer every 2 layers (found to be optimal in experiment 2)
- Added velocity as an additional layer after prediction in stage 1
- Dropout layer after each hidden layer
    - Best Performance Dropout : 0.5
- ReLU Activation after every layer
- Expanded training set to higher dimension to achieve better separability
- Layer Dimensions:
    - 6 or 7 -> 128
    - 128 -> 64
    - 64 -> 16
    - 16 -> 8
    - 8 -> 1 (output)



Performance:

- Compounded training time with approximately 7 minutes per epoch for 25-30 epochs to convergence for not much improvement over previous 1 stage model
- Best RMSE Training Loss: 216
- Best RMSE Testing Loss: 786

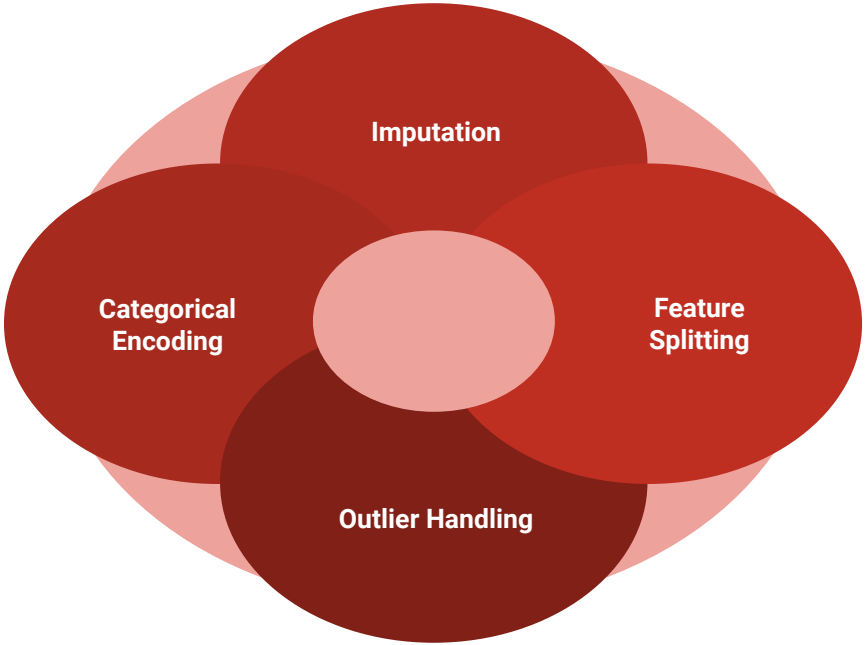
Summary of Experiments

Experiment	Training RMSE Loss	Testing RMSE Loss	Key Takeaways
Multiple Linear Regression	1382	N/A	A Deep Learning Approach would be more suitable for the task
Shallow Fully Connected Model	590	850	A higher number of layers could be used for better convergence and predictions
Deep Fully Connected Layer	360	820	Overfitting occurs with deeper models and 4-5 layers works best
2 Stage Model with lower Dimensions	350	805	Higher dimensions could result in better predictions through better data separability
2 Stage Model with higher dimensions	216	786	Higher dimensions causes a greater chance of overfitting but this resulted in best predictions anyway. Dropout tuning and Batchnorm layers were crucial to prevent this

Discussion

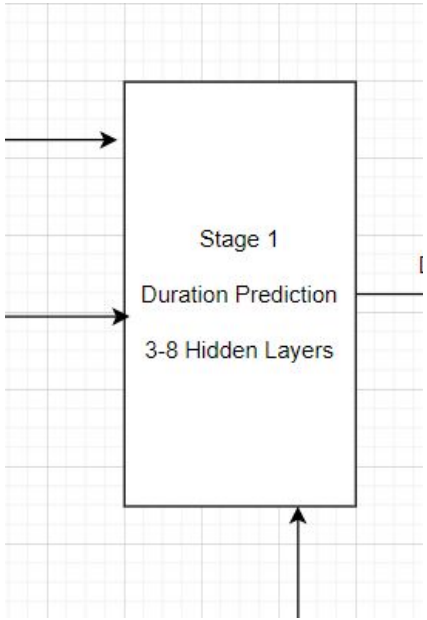
What have you learned

Feature Engineering is a key component of deep learning model design and model effectiveness relies on this component.



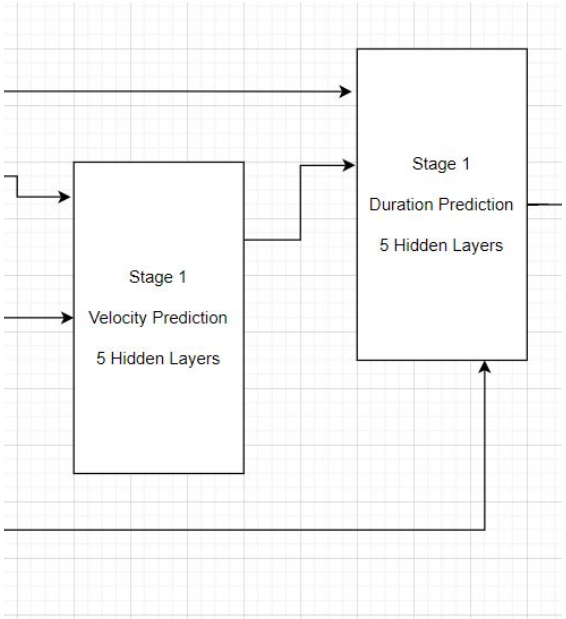
What have you learned

Overfitting is a major issue as models get deeper



What have you learned

Models can be stacked to allow for better predictions



Future Work

- Experiment with more complex network architectures to improve performance
- Perform further data exploration to determine the cause of skewed values and 0 predictions
- Remove possible Gradient loss causing incorrect predictions
- Research human and animal cognitive models such as dual process cognition and memory short circuiting to improve model performance.