Predicting Zoning Use in Fresno, CA

Ojas Deshmukh
Kyle Lyman
Cheu Thao
Stephen Vang
Johnson Yang

IS 160

1. Research Question
2. Data Sources
3. Feature Processing
4. Model Design & Outcome
5. Conclusion

1. Research Question
2. Data Sources
3. Feature Processing
4. Model Design & Outcome
5. Conclusion

Guiding Questions

1. Where are the most successful areas for future housing and development?

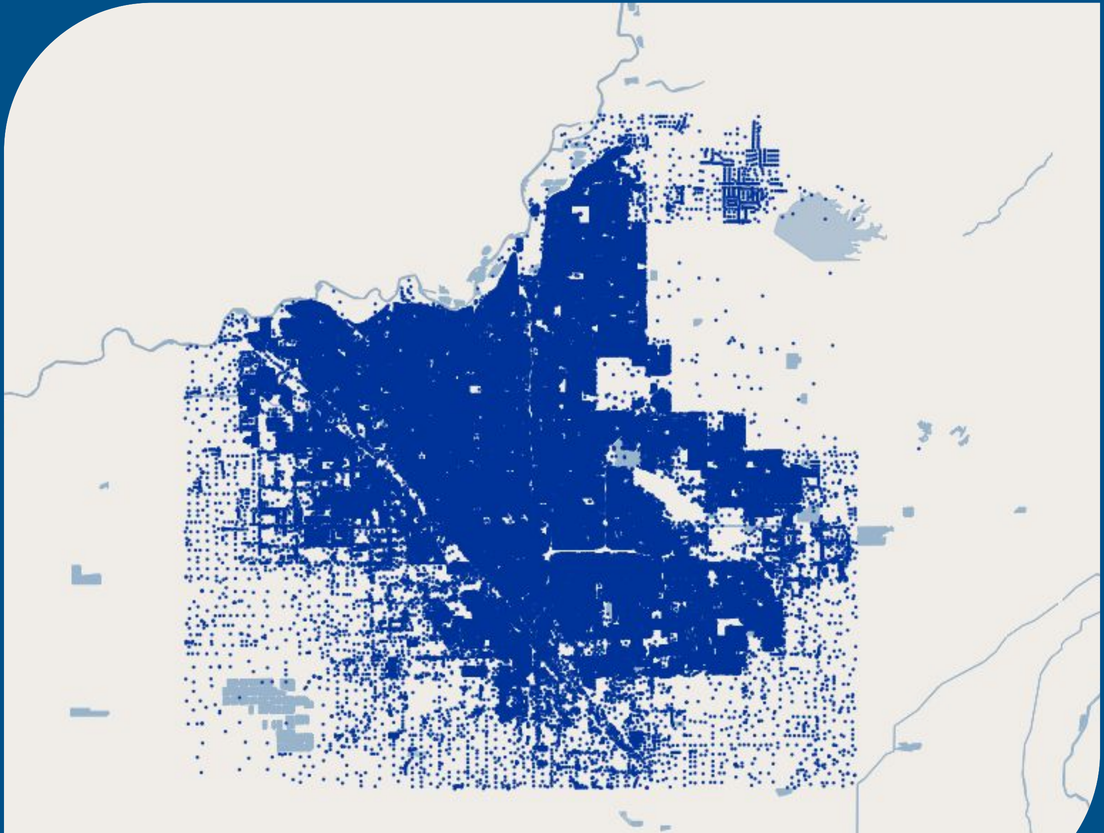
2. Which streets or highways will experience the highest demand due to predicted developments?

3. Where should new transportation options (e.g., bus stops) be placed to support growing housing density and accessibility?

1. Research Question
2. **Data Sources**
3. Feature Processing
4. Model Design & Outcome
5. Conclusion

Data primarily comes from City of Fresno's ArcGIS data system and Koordinates website.

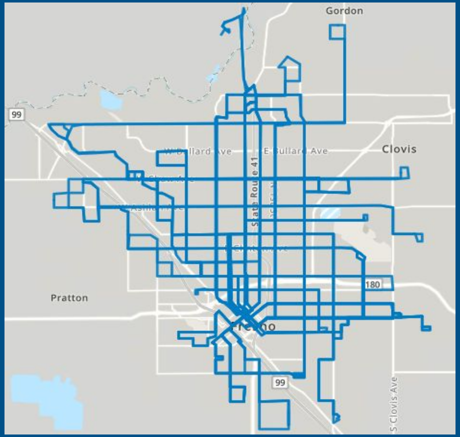
- Addresses (for all areas in the city)
- Bus Routes & Stops
- Zoning Data (including use type and descriptions)
- Fresno city limits
- Street data

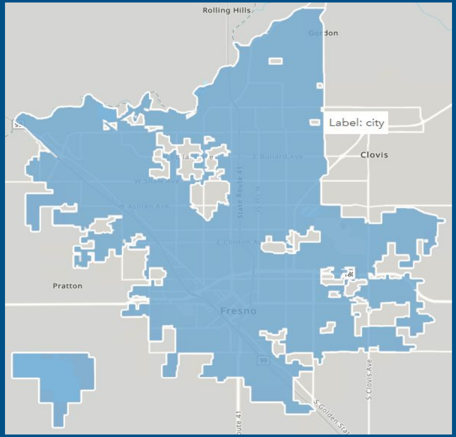


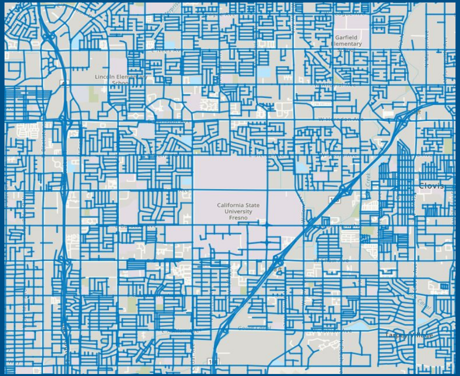
Fresno Addresses Dataset

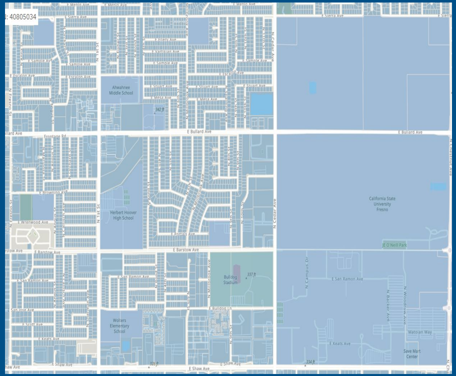
All CSV files compiled and used for processing and model design

- Bus_routes_data
- Bus_stop_data
- Street_data
- City_limits_data
- Zoning_data
- Fresno_addresses_data





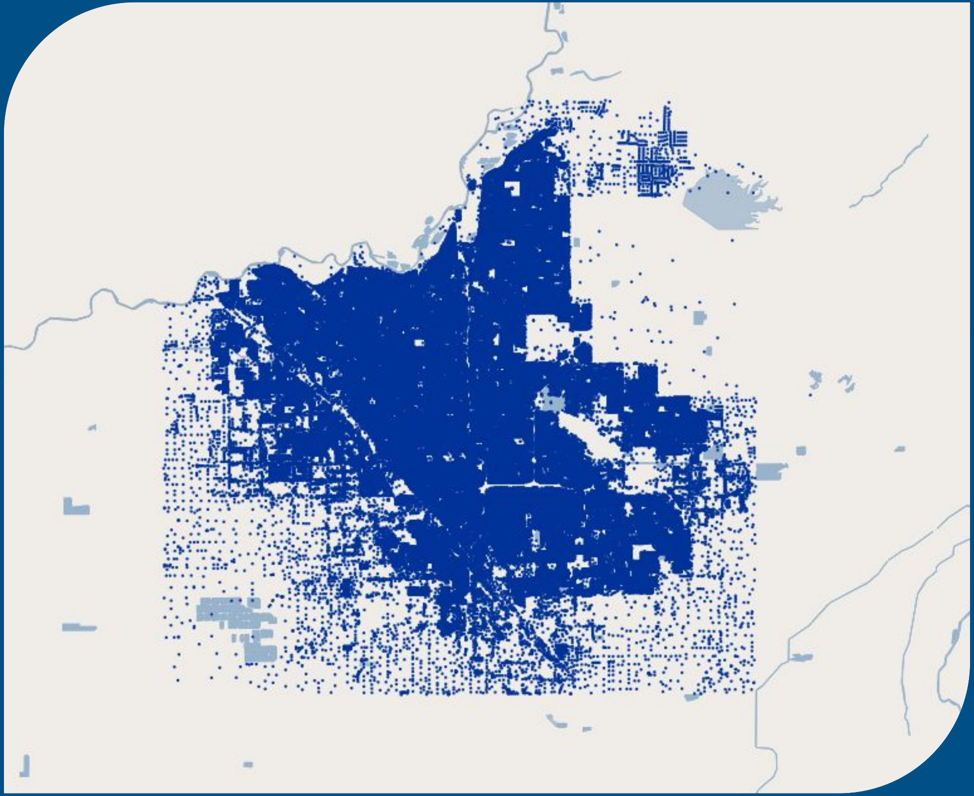




1. Research Question
2. Data Sources
3. **Feature Processing**
4. Model Design & Outcome
5. Conclusion

Key Features

- Distance to City Center (calculated)
- Parcel Size
- Infrastructure Density (calculated)
- Zoning
- Zoning Description
- Distance to Nearest Bus Stop (calculated)



Part 1: Creating Calculated Features

- Distance to City Center
  - Is the Euclidean Distance of the x-y coordinate of a given address to the city center x-y coordinate.
- Distance to Nearest Bus Stop
  - Used Nearest Neighbor Approach to determine closest bus stop for each address x-y coordinate, calculating the difference between the x-y coordinates of the bus stop and address.
- Infrastructure Density
  - Grouped addresses based on distance from each point (within approximately 500 meters)

```python
# Prepare coordinate data for cKDTree
bus_stop_coords = bus_stops[['x', 'y']].values  # Replace with correct column names
address_coords = addresses[['shape_X', 'shape_Y']].values  # Replace with correct column names

# Create a cKDTree and find nearest neighbors
tree = cKDTree(bus_stop_coords)
distances, indices = tree.query(address_coords, k=1)  # k=1 for nearest neighbor

# Add nearest bus stop index and distance to the addresses DataFrame
addresses['nearest_bus_stop_index'] = indices
addresses['distance_to_nearest_bus_stop'] = distances

# Adjust columns to merge
bus_stop_cols_to_merge = ['Stop Name', 'Stop ID', 'Stop Latitiude', 'Stop Longtitude']  # Using exact col

# Merge bus stop data into addresses
addresses = pd.merge(
    addresses,
    bus_stops.reset_index()[bus_stop_cols_to_merge],
    left_on='nearest_bus_stop_index',
    right_index=True,
    suffixes=('', '_bus_stop')
```

Feature Processing

Part 2: Encoding Features

- Used label encoding for each categorical variable and standardized distance features for analysis.
- 70% training 30% test split of the data.
- Used fewer features to reduce potential overfitting and keep model interpretable.
  - Prototype, scalable to other features

Data Cleaning And Feature Engineering

1. Encode Categorical Data:

```
[ ]
from sklearn.preprocessing import LabelEncoder

# Encode zoning-related columns
label_encoder = LabelEncoder()

categorical_columns = ['Zoning', 'Zoning Description', 'PLANNED_LAND_USE', 'EXISTING_LAND_USE']
for col in categorical_columns:
    addresses[col] = label_encoder.fit_transform(addresses[col].astype(str))

# Check encoding
print(addresses[categorical_columns].head())
```

1. Research Question
2. Data Sources
3. Feature Processing
4. **Model Design & Outcome**
5. Conclusion

Model Design

```python
# Define the DNN architecture
model = Sequential([
    Dense(128, activation='relu', input_shape=(X_train.shape[1],)),
    Dropout(0.2),  # Dropout to prevent overfitting
    Dense(64, activation='relu'),
    Dropout(0.2),
    Dense(len(label_encoder.classes_), activation='softmax')  # Output layer for classification
])

# Compile the model
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Train the model
history = model.fit(X_train, y_train, epochs=20, batch_size=32, validation_split=0.2)

# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print(f"Test Accuracy: {accuracy:.2f}")
```

Flow of code implementation –

- Importing the libraries and packages we need for project.
- Uploading datasets.
- Editing and merging datasets based upon uniques parameters (eg. Join on parcel id or Assessor Parcel Number (APN)).
- Data cleaning and feature engineering.
  - Encoding categorical data.
  - Calculate distance to city center. ...
- Model implementation 3 trials and error implementation with unweighted model, weighted model and SMOTE balanced model

Outcome

Initial Model: After multiple epochs and adjusting the parameters, our model accuracy ranged from 86-88%.

Revised Model: Using SMOTE to rebalance sampling, it improved model accuracy to 97%

```
Epoch 13/20
4104/4104 ━━━━━━━━━━━━━━━━━━━━ 18s 2ms/step - accuracy: 0.8385 - loss: 0.4364 - val_accuracy: 0.8523 - va
Epoch 14/20
4104/4104 ━━━━━━━━━━━━━━━━━━━━ 13s 3ms/step - accuracy: 0.8382 - loss: 0.4363 - val_accuracy: 0.8518 - va
Epoch 15/20
4104/4104 ━━━━━━━━━━━━━━━━━━━━ 18s 2ms/step - accuracy: 0.8399 - loss: 0.4315 - val_accuracy: 0.8530 - va
Epoch 16/20
4104/4104 ━━━━━━━━━━━━━━━━━━━━ 12s 3ms/step - accuracy: 0.8399 - loss: 0.4321 - val_accuracy: 0.8576 - va
Epoch 17/20
4104/4104 ━━━━━━━━━━━━━━━━━━━━ 12s 3ms/step - accuracy: 0.8412 - loss: 0.4251 - val_accuracy: 0.8543 - va
Epoch 18/20
4104/4104 ━━━━━━━━━━━━━━━━━━━━ 23s 4ms/step - accuracy: 0.8411 - loss: 0.4240 - val_accuracy: 0.8589 - va
Epoch 19/20
4104/4104 ━━━━━━━━━━━━━━━━━━━━ 16s 2ms/step - accuracy: 0.8440 - loss: 0.4197 - val_accuracy: 0.8579 - va
Epoch 20/20
4104/4104 ━━━━━━━━━━━━━━━━━━━━ 11s 3ms/step - accuracy: 0.8460 - loss: 0.4190 - val_accuracy: 0.8583 - va
2199/2199 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step - accuracy: 0.8581 - loss: 0.3830
Test Accuracy: 0.86
2199/2199 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step
```

1. Research Question
2. Data Sources
3. Feature Processing
4. Model Design & Outcome
5. **Conclusion**

Conclusion

1. Where are the most successful areas for future housing and development?

Findings:

- Areas with **higher housing density predictions** (from the `infrastructure_density` feature) correlated strongly with zoning classifications like:
  - **Residential Single-Family (RS)** and **Residential Multi-Family (RM)** zones.
  - Mixed-use zones (**CMX**) near city centers showed significant development potential.

2. Which streets or highways will experience the highest demand due to predicted developments?

Findings:

- Streets near high-density residential areas with **longer distances to city centers** showed increased predicted demand.
- Specific corridors in **south Fresno and near the urban fringe** were flagged for high development activity.

3. Where should new transportation options (e.g., bus stops) be placed to support growing housing density and accessibility?

Findings:

- The model identified areas with **high predicted housing density** but **poor current bus stop coverage** (calculated using proximity to nearest bus stop).
- **Key Suggestions for Bus Stops**:
  - Add stops in **southwest Fresno**, particularly near high-density zones without adequate coverage.
  - Areas near new developments in **north Fresno** and emerging neighborhoods in the southeast need improved access.

Model reliably predicted planned use for a particular community.

Potential Insights from model

- Better usage of space/more accurate zoning for the city.
- Better placement of public transportation in new areas based on density.
- Future Work could explore the incorporation of additional features, such as socioeconomic data and historical land-use trends for city planners.