Westat

Applying Machine Learning Techniques to Transportation Surveys

Jane Shepherd, *Westat*
Marcelo Simas, *Westat*
Anthony Fucci, *Westat*
Alexander Cates, *Westat*

Background

- **Household Travel Surveys**
  - Collect socio-economic and demographic data about households and individual members
  - Collect a travel diary for 1-2 days
    - Describe the *how, why, when,* and *where* of each place visited on the assigned travel day(s)
  - Recently deployed smartphone-based surveys
    - Geolocation – Auto-detects trip start/stops using geofences
    - Travel capture – GPS data informs arrival and departure times
    - Prompted recall
  - **Past surveys**
    - Asheville, Fairbanks, Albuquerque, South Jersey, Las Vegas, Michigan, Billings, NHTS
  - **Present / future surveys**
    - Chicago, Maryland, Laredo

Why use machine learning?

- **Availability**
  - Ubiquity of open source software like R/Python make deploying applications easier than ever
- **Efficiency**
  - Data processing tasks can be assisted (or replaced) by machines
- **Adaptability**
  - Declining response rates in household travel surveys motivate new designs

How do we use machine learning?

1. **Coding open-end** responses using **Natural Language Processing** and **Random Forest** models.

2. Ascertaining **Industry** and **Occupation** in real time using **Natural Language Processing** and **Vector Space** models.

3. Determining **place validity** and **predicting travel attributes** using **GPS** and **Accelerometer**-derived features to train **Random Forest** models.

Coding Open-End Responses

- **Problem**
  - NHTS yielded over 180,000 open-ended responses
  - Around 52,000 of these belonged to the “Trip Purpose” question
- **Traditional Solution**
  - Analyst attempts to up-code each response by hand
  - Average 15 sec / response = ~ 750 hours
- **ML Solution**
  - Analyst up-codes a sample of responses.
  - Treat the sample as labeled training data to be modeled

Trip Purpose Model Steps

- **Feature engineering**
  - Select and derive variables
- **Training**
  - Split the data into 85% train / 15% test
  - Train the Random Forest model
- **Testing**
  - Explore model accuracy using different probability thresholds
- **Applying**
  - Feed new open-ended responses into the model and pre-select responses

Westat

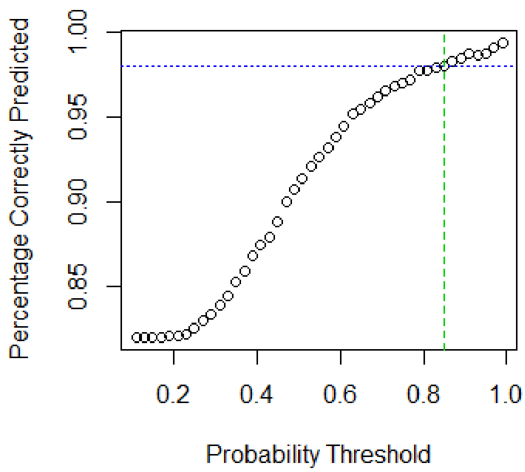
Feature Engineering

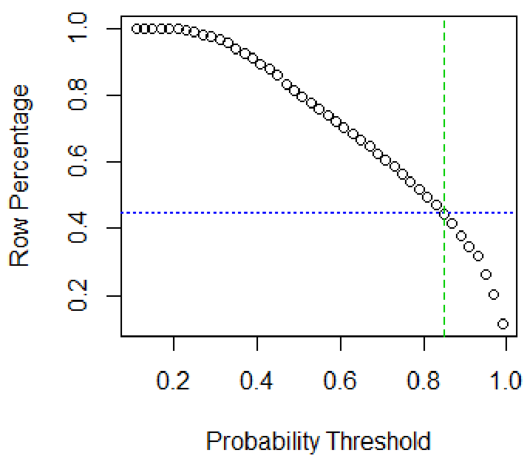
- **Place attributes**
  - activity duration, travel time, place type, change in party size, etc.
- **Person attributes**
  - Worker/student status, etc.
- **Open-ended Text attributes**
  - Make case-insensitive
  - Remove punctuation
  - “Tokenize” into separate words
  - Remove “Stop Words”
  - Find the “Stem” of each word
  - Create “n-grams” for word sequences

Original	"I was filing my taxes with my Accountant!"
Lower-case	"i was filing my taxes with my accountant!"
Punctuation	"i was filing my taxes with my accountant"
Tokenize	"i" "was" "filing" "my" "taxes" "with" "my" "accountant"
Stop-words	"filing" "taxes" "accountant"
Stemming	"file" "tax" "account"
Tri-grams	"file" "tax" "account" "file_tax" "tax_account" "file_tax_account"

Training and Testing

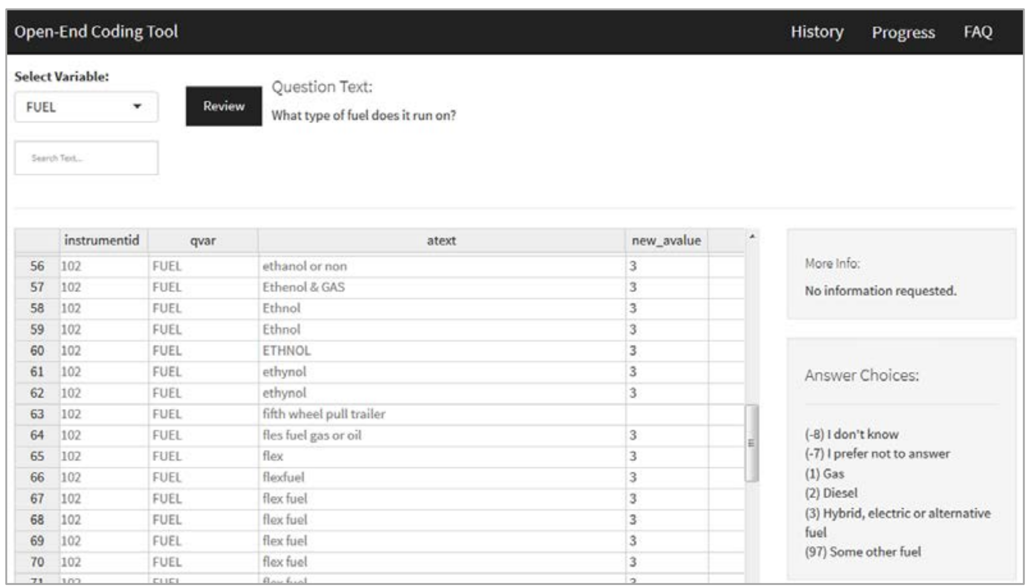
- Trained a random forest model using 200 trees
- Fed model to the test dataset
  - Output predicted probabilities for each class
- Assessed Accuracy
  - 0.85 Probability threshold
    - 98% Accuracy
    - 45% of the data





Applying the Model

- Applied model to new "Trip Purpose" responses
- Output predictions to an open-text coding application
  - Limited to > 0.85 predicted probability
  - Highlighted predicted records
  - Analyst could review in passing while coding other responses



Industry and Occupation

- **Problem**
  - Industry/Occupation asked of every worker in the household
  - Want to map these responses to a standard code-set:
    - North American Industry Classification System (**NAICS**)
    - Standard Occupational Classification (**SOC**) system
  - 20+ high level codes to choose for each question
  - When only high level descriptions are present, some Industries/Occupations are obfuscated
- **Traditional Solution**
  - Allow participant to sift through the hierarchy of codes
    - This effort would be too burdensome on participant
- **ML Solution**
  - Use text features extracted from low-level descriptions to build a vector space model
  - Ask the participant to provide “a few words” about their industry/occupation which can be fed into the model

Westat

Developing the Model

- Used similar text-processing techniques as the open-ended coding application
- Create a normalized Document Term Matrix for every code
- Create an input vector by applying the same “processing” to the user-input text
- Calculate the Cosine Similarity between the input vector and each row vector in the Document Term Matrix

SOC Title	SOC Direct Match
Financial Examiners	Bank Compliance Officer
	Bank Examiner
	Financial Compliance Examiner
	Home Mortgage Disclosure Act Specialist
	Payroll Examiner
	Pension Examiner
Credit Counselors	Credit Counselor
	Debt Management Counselor
	Student Financial Aid Counselor
	Student Loan Counselor
Loan Officers	Branch Lending Officer
	Commercial Lender
	Loan Analyst
	Loan Officer
	Loan Reviewer
	Payday Loan Officer
	Real Estate Loan Officer

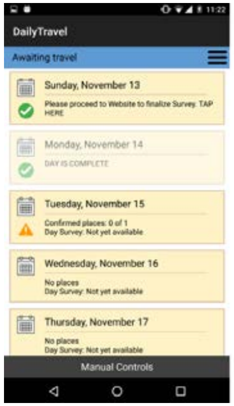
	payday_loan_offic	credit_counselor_debt	pension_examin	offic_real_estat	…
Financial Examiners	0	0	0.020833333	0	
Credit Counselors	0	0.03030303	0	0	
Loan Officers	0.019607843	0	0	0.019607843	
…					

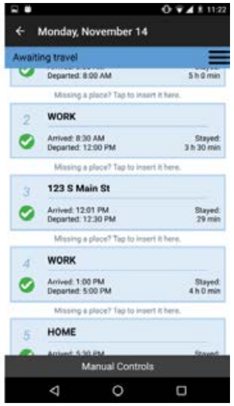
Westat

Developing the System

- Once the participant provides a few words about their Industry/Occupation…
  - The model returns the records with the **top 10 cosine similarity scores**
  - If the participant does not find a match, the input-text is cached to be up-coded in post-processing
- Needed a system that would work with our online survey instrument
  - Adaptation of **OpenCPU** server
    - HTTP API for calling R processes
  - Real-time application encouraged a fast / light-weight model
    - Motivated the use of a vector space model instead of random forest

Mobile-app Data Processing





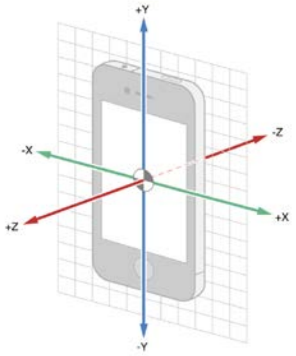
- **Background**
  - 1-2 days of required, confirmed travel
  - 5-6 days of optional app data collection
- **Problem**
  - Passive travel data collected by the smartphone application
    - Not all geo-located places may be valid
    - Not all places contain travel attributes (i.e. travel mode)
- **Traditional Solution**
  - Analyst reviews and processes the passive/unconfirmed data
- **ML Solution**
  - We can use GPS and Accelerometer-derived attributes from places on confirmed travel days to predict information about places on unconfirmed days

Westat

Developing the Models

- Identifying "noise stops"
  - Invalid places detected by the app while the participant was still at an existing location
  - On confirmed travel days:
    - Identify user-deleted places between the arrival and departure time of a confirmed place
    - Binary response variable = Place deleted? (1 / 0)
- Predicting travel mode
  - Collapse travel mode list into more distinguishable categories
    - Walk, Bike, Auto, Transit
    - New mode list = multiclass response variable

GPS/Accelerometer Features



- Data between the start and arrival time of each place
- GPS
  - Speed
    - Mean, Median, Standard Deviation, Minimum, Maximum, etc.
  - Distance measures
    - Circuity: Point distance / Straight line distance
    - Compactness: Point Distance / Diagonal Bounding box distance
  - Travel time
- Accelerometer
  - Vector magnitude of tri-axial Accelerometer
    - Mean, Median, Standard Deviation, Kurtosis, Skewness, IQR, Maximum Moving Average, etc.

Westat

Application and Challenges

- Challenges
  - GPS data is messy!
    - Points are discontinuous, collected intermittently to spare smartphone battery life
    - Low accuracy points due to urban canyon effect, etc.
  - Phone orientation is not consistent
    - Vector magnitude of accelerometer data, because individual x, y, z positions are variable
  - Participant interaction
    - User's have the ability to adjust start, arrival, departure times
      - Limits our training data to places that have not been altered by the user
  - Smartphone models and OS versions behave differently
- Currently applying this model in the Chicago HTS pilot
  - Using random forest models
  - Exploring other machine learning algorithms (i.e. Recurrent Neural Networks)
  - Waiting on more data to solidify a robust model

Summary

- The scale of the NHTS motivated the idea of a machine learning model that could code open text responses.

- With increased knowledge of Natural Language Processing, this idea spawned the proposition of a model that could assist the collection of Industry and Occupation information.

- Smartphone-based travel surveys generate considerably more data than traditional HTS designs.
  - Machine learning tools available in R allowed us to leverage this data to extract more information without the need for additional sampling.

Toolbox (all open-source)

- R
  - Software environment for statistical computing
  - Packages
    - data.table
    - randomForest
    - text2vec
    - NLP
    - tm
    - SnowballC
    - slam
- PostgreSQL
  - Relational database management system
  - PostGIS
    - Spatial/geographic extensions for PostgreSQL
- OpenCPU
  - "Framework for embedded scientific computing and reproducible research"