## ADTA5340 Final Project

## PART I: A Strategy to Employ Machine Learning in a Firm

### Define data to collect

First thing we need to determine what data we would collect. We need to take samples of datasets and estimate their current volume and growth. We need to check if data requires additional preprocessing. We need to plan for enough storage and processing power and we should be able to manage capacity at real-time without affecting processing capabilities.

The system should not be used as data store for any other business critical application, rather it should be able to collect / consume data from those systems.

### High-level architecture

The system would have several independent layers of processing. Independent levels would ensure resiliency and provide ability to scale each layer independently.

#### Data collection layer.

This layer would be used to collect and store initial data from other systems. This layer should have enough storage capacity to collect data for a period of time before it would be moved to other layers and processed there. Capacity of this system should be kept at maximum of 50%. Depends on how much data coming each second, we would need to create a storage with fast write.

#### Data preprocessing layer.

The purpose of this layer is to verify, clean, and get data ready for further processing. Also, this layer would be used as a quality assurance system. If there would be large amount of preprocessing required for some dataset, we would be able to alert data quality issues to respective systems’ owners.

#### Data storage layer.

This is long-term storage. It would require high capacity and fast read time. We should be able to increase capacity fast without affecting currently stored data.

#### AI/ML processing layer.

This layer requires highest number of CPU and memory to constantly build and run models. Storage requirements are not high, it should have some storage to keep some temporary files, but most of the time it should be able to keep everything in memory.

#### Reporting layer.

This is separate reporting layer. We need it to generate and store reports. In terms of CPU, memory and storage it would have mediocre requirements.

#### Web Access layer.

We need this layer to perform following functions:

* Overall system management, including access management;
* User access to run models and get results;
* Provide access to stored report and to report management functions;

#### API Access layer.

Modern organizations are data driven. Lot of systems produce, process, and store different data, that consumes by other systems in automated manner. We need a layer that could provide data for other machines in a machine-readable way.

#### System Alerts layer.

This layer is required to send alerts related to the system itself (health, reports readiness, security events, etc.), as well as alerts related to various external systems (i.e. data quality)

### System diagram

### Technologies to be used.

#### Hardware

Hardware should be based on commodity systems of x86/x64 architecture. This would provide ability to replace systems fast with low cost. Different components should be packed into standard racks with ability to add new racks into the system.

Also, we should replace components by racks. When certain percent of systems within a rack have issues would should replace the entire rack. System should be able to rebalance itself automatically.

#### Software

Software should be mostly based on open-source software. It would provide ability to both: change code of applications or develop new code, and at the same time use achievements from other software developers.

## PART II: Big Data, Artificial Intelligence, and Machine Learning

### The history of artificial intelligence until now

(3 pages min including images)

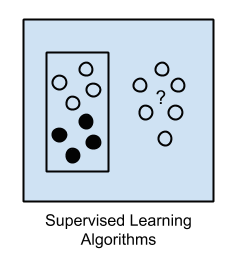
### Select three different sectors of the U.S. economy, do research, and discuss the impacts of **big data** and **machine learning** on **each** of them

(3 pages min including images)

### Discuss **in detail** the three major styles of learning in machine learning: (1) Supervised Learning, (2) Unsupervised Learning, and (3) Semi-Supervised Learning

There are three primary learning styles for AI/ML: supervised, unsupervised, and semi-supervised learning.

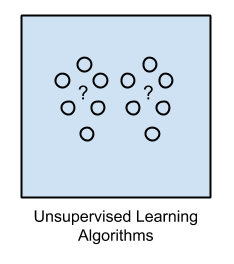
For supervised learning training dataset has marked by an operator (a person who is going to train a model), the operator knows meanings of all variables and understands dependent and independent variables.



Supervised learning works with relatively small datasets, where independent and dependent variables could be clearly identified, data in a dataset could be effectively verified and cleaned.

Examples for supervised learning algorithms are linear regressions, decision trees (regression and classification).

For unsupervised learning, dataset has no labels or markers for output data, algorithm learns itself. It learns to find common dependencies or tries to understand general rules on how to separate the data into segments.

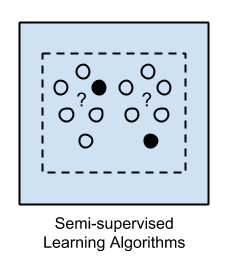


Usually datasets for unsupervised learning are very large and hard to clear. Imagine a collection of videos on the Youtube or digital images’ libraries. If you do not have enough description in metadata it would be hard to find something with a search request like: “show me a list of all images with cats”. The only way is to train ML model to detect a cat on an image, them use this model to find cats on images.



Another example for unsupervised learning model would be detection of obstacles on a road. We could train algorithm to find/detect an obstacle on a video stream, an operator would be unable to tell in advance if there is an obstacle, algorithm should learn itself to detect it.

Semi-supervised learning is combination of the two techniques above. We could label some data (inputs and outputs) in the dataset but not for the whole dataset. ML algorithm should learn itself what is it and how to mark it. But with even small amount of labeled data algorithm usually provides more accurate results than completely unsupervised learning.



An example for semi-supervised learning could be detection of traffic lights. An operator would mark some outputs for a ML algorithm like shape of traffic lights, how they usually mounted, what different colors mean, etc. But the operator could not mark all possible variants of traffic lights setups, algorithm would use information it have to detect and properly interpret information from images or videostreams.

## PART III: Data Preprocessing

For data preprocessing task I found a dataset “Aviation Data and Documentation from the NTSB Accident Database System”.

URL for the dataset: <https://catalog.data.gov/dataset/aviation-data-and-documentation-from-the-ntsb-accident-database-system>

Public: This dataset is intended for public access and use.

License: No license information was provided. If this work was prepared by an officer or employee of the United States government as part of that person's official duties it is considered a U.S. Government Work.

This dataset was initially in XML format, I had to export it into CSV to work with it. The dataset has 84088 records and 31 variables:

\_EventId - 0 - Event identifier, each event should have an id. Field is not useful for further analysis as it is unique.

\_InvestigationType - 4 - Type of investigation (Accident, Incident). Some events have no type. NULL value could be replaced with Unknown.

\_AccidentNumber - 0 - Number (identification) of an event in some recording systems. Field is not useful for further analisys as it is unique.

\_EventDate - 0 - Date of an event. Cannot be NULL.

\_Location - 76 - Location of an event (city, state in case of the US or other identifiable location if outside of the US). Some locations are NULL. There are other variables for locations data.

\_Country - 507 - some locations do not have county filled. It could have NULL values when an event happened outside of any country (i.e. under an ocean or a sea).

*\_*Latitude – 54039

\_Longitude – 54048

\_Latitude and \_Longitude are geographical coordinates of an event. It should not be empty, but there are other location variables that could substitute them. If we really need them we need to work with other variables to get exact coordinates of an event.

\_AirportCode - 36439 - each airport has a code (3 or 4 letters). An event could happen outside of airport, so it could have NULL values. This variable could be used to define geographic location of an event.

\_AirportName - 33735 - Each airport has a human readable name. In case an event of outside of any airport this field would have NULL value. I would ignore Location variables. An event could happen at any location. But for general statistics these values could be useful to find which locations have more incidents.

\_InjurySeverity - 0 - Each event has assigned severity. Cannot be NULL.

\_AircraftDamage - 2676 - Describes damage to an aircraft. NULL value indicates no damage, NULL should be replaced with "NODAMAGE".

\_AircraftCategory - 56751 - Describes category of an aircraft like airplane, balloon, etc. NULL value indicates that category was not defined or not recorded. Missing value could be determined by Make and \_Model variables.

\_RegistrationNumber - 3778 - Categoriacal value (non-numeric). NULL value indicates that aircraft was not registered. For analysis I would remove this column. Replacing NULL with something like "NOT REGISTERED" would affect results of analysis: lot of events would be falsely attributed to it.

\_Make - 70 - Each aircraft should have a producer like Boeing, Airbus, Cessna, etc. But some are made by people and do not have registered Maker.

\_Model - 99 - Some aircrafts do not have models because they were built by private persons.

\_AmateurBuilt - 592 - indicates if aircraft is built by amateur builder. NULL value indicates that data was not recorded for this variable, NULL could be replaced with "NO"

\_NumberOfEngines - 4970 - Some aircrafts has no engines (like baloon or glider). But sometimes aircraft definetely have 1 or more engines but information was not collected.

\_EngineType - 4280 - Describes engine types. NULL indicates either there is no an engine, or data is missing.

\_FARDescription - 57056 - Describes if an aircraft performed a specific function (i.e. Armed Forces or General Aviation, etc.). NULL indicates no data is recoded, it should be changed to Unknown.

\_Schedule - 72269 - Indicates if a flight was scheduled (like AA20 from DFW to London Heathrow), unscheduled (i.e. for Armed Forces interception) or unknown. NULL values should be replaced with "NO".

\_PurposeOfFlight - 4791 - Describes the purpose of flight. NULL values should be replaced by "Unknown".

\_AirCarrier - 79927 - Names air carrier. Most of events have no air carrier, because event happened with personal aircrafts that do not perform regular passengers’ transfers. I would ignore this column.

\_TotalFatalInjuries - 27068 - Describes number of total fatal injuries. NULL values should be replaced with "0".

\_TotalSeriousInjuries - 29726 - Describes number of total serious injuries. NULL values should be replaced with "0"

\_TotalMinorInjuries - 28523 - Describes number of total minor injuries. NULL values should be replaced with "0"

\_TotalUninjured - 14507 - Describes number of total uninjured. NULL values show that there are no data. Normally it should show total people on board minus all injures combined. But there is no variable for total passengers, so we cannot calculate this value. I would ignore it.

\_WeatherCondition - 3060 - This variable describes weather conditions. NULL values indicate that there is no data on file, or weather had nothing to do with an event.  
BroadPhaseOfFlight - 6691 - Describes phase of flight when an event happened. NULL values should have "UNKNOWN"  
ReportStatus - 0 - Describes status of a report. Cannot have NULL values.  
\_PublicationDate - 14236 - Date of a report publication. NULL value shows that report is not published (could be "Secret" or not ready for publication)

Step 1. I reduced number of variables for this dataset:

df2 = df[[**"\_InvestigationType"**, **"\_EventDate"**, **"\_InjurySeverity"**, **"\_AircraftDamage"**, **"\_Make"**, **"\_Model"**, **"\_AmateurBuilt"**, **"\_FARDescription"**, **"\_Schedule"**,**"\_PurposeOfFlight"**, **"\_TotalFatalInjuries"**, **"\_TotalSeriousInjuries"**, **"\_TotalMinorInjuries"**, **"\_BroadPhaseOfFlight"**]]

Step 2. I replaced NaNs with proper values:

df2[[**'\_TotalSeriousInjuries'**, **'\_TotalMinorInjuries'**, **\_TotalFatalInjuries'**]] = df2[[**'\_TotalSeriousInjuries'**, **'\_TotalMinorInjuries'**, **'\_TotalFatalInjuries'**]].replace(np.NaN,0)  
df2[[**'\_InvestigationType'**]]=df2[[**'\_InvestigationType'**]].replace(np.NaN,**"Unknown"**)  
df2[[**'\_AircraftDamage'**]]=df2[[**'\_AircraftDamage'**]].replace(np.NaN,**"No damage"**)  
df2[[**'\_Make'**]]=df2[[**'\_Make'**]].replace(np.NaN,**"No make"**)  
df2[[**'\_Model'**]]=df2[[**'\_Model'**]].replace(np.NaN,**"No model"**)  
df2[[**'\_AmateurBuilt'**]]=df2[[**'\_AmateurBuilt'**]].replace(np.NaN,**"No"**)  
df2[[**'\_FARDescription'**]]=df2[[**'\_FARDescription'**]].replace(np.NaN,**"No"**)  
df2[[**'\_Schedule'**]]=df2[[**'\_Schedule'**]].replace(np.NaN,**"NO"**)  
df2[[**'\_PurposeOfFlight'**]]=df2[[**'\_PurposeOfFlight'**]].replace(np.NaN,**"Unknown"**)  
df2[[**'\_BroadPhaseOfFlight'**]]=df2[[**'\_BroadPhaseOfFlight'**]].replace(np.NaN,**"UNKNOWN"**)

Step3. Checked that there are no more NaNs.

print(df2.isnull().sum())

## PART IV: Machine Learning: Supervised

The dataset abalone.csv has

## PART V: Machine Learning: Supervised

Dataset adult\_salary.csv has 48842 records and 15 variables. Most of string values has a space symbol at the beginning. With Excel I run a replacement function to remove all extra space symbols. Then I checked if there were any NaN values. There were none.

The dataset has a variable Fnlwgt that was added as a part of data collection and represent no value to the analysis. I removed it with:

df2 = df[[**"Age"**,**"Emp\_type"**,**"Education"**,**"Education\_num"**,**"Marital"**,**"Occupation"**,**"Relationship"**,  
 **"Race"**,**"Sex"**,**"Capital\_gain"**,**"Capital\_loss"**,**"weekly\_hours"**,**"Country"**,**"Income"**]]

Then for Classification algorithms I had to replace all string values with integers.

df2[[**'Income'**]]=df2[[**'Income'**]].replace(**"<=50K."**,0)  
df2[[**'Income'**]]=df2[[**'Income'**]].replace(**">50K."**,1)  
df2[[**'Income'**]]=df2[[**'Income'**]].replace(**"<=50K"**,0)  
df2[[**'Income'**]]=df2[[**'Income'**]].replace(**">50K"**,1)  
Sex = Series([0,1],index=[**'Male'**,**'Female'**])   
df2[**'Sex'**]=df2.Sex.map(Sex)  
df2.Emp\_type = pd.Categorical(df2.Emp\_type)  
df2[**'Emp\_type'**] = df2.Emp\_type.cat.codes  
df2.Education = pd.Categorical(df2.Education)  
df2[**'Education'**] = df2.Education.cat.codes  
df2.Marital = pd.Categorical(df2.Marital)  
df2[**'Marital'**] = df2.Marital.cat.codes  
df2.Occupation = pd.Categorical(df2.Occupation)  
df2[**'Occupation'**] = df2.Occupation.cat.codes  
df2.Relationship = pd.Categorical(df2.Relationship)  
df2[**'Relationship'**] = df2.Relationship.cat.codes  
df2.Race = pd.Categorical(df2.Race)  
df2[**'Race'**] = df2.Race.cat.codes  
df2.Country = pd.Categorical(df2.Country)  
df2[**'Country'**] = df2.Country.cat.codes

After these modifications I got a dataset with only numerical values:

print(df2.dtypes)  
Age int64

Emp\_type int8

Education int8

Education\_num int64

Marital int8

Occupation int8

Relationship int8

Race int8

Sex int64

Capital\_gain int64

Capital\_loss int64

weekly\_hours int64

Country int8

Income int64

dtype: object

EDA Analysis

Graphs

Algorithm for the Model

I selected Decision Tree Classifier algorithm because I need an algorithm that works with categorical data (encoded into numbers), and output of the algorithm also categorical (less 50 or more 50). I run algorithm and got weighted average a 0.82.

DecisionTreeClassifier(class\_weight=None, criterion='gini', max\_depth=None,

max\_features=None, max\_leaf\_nodes=None,

min\_impurity\_decrease=0.0, min\_impurity\_split=None,

min\_samples\_leaf=1, min\_samples\_split=2,

min\_weight\_fraction\_leaf=0.0, presort=False,

random\_state=None, splitter='best')

precision recall f1-score support

0 0.87 0.89 0.88 9243

1 0.63 0.60 0.62 2968

accuracy 0.82 12211

macro avg 0.75 0.75 0.75 12211

weighted avg 0.82 0.82 0.82 12211

Accuracy level for the model is 81.836%

K-Fold validation algorithm gave me squared error as 0.820 and standard deviation as 0.007

For prediction I created two new data points:

model.predict([[39,1,0,0,1,1,1,0,0,10000,500,60,0]])model.predict([[30,1,0,0,1,1,1,0,0,1000,50,60,0]])

Both data points predicted that salary would be less than $50000.

## PART VI: Machine Learning: Unsupervised

Dataset name is cars\_eveluation.cvs. It has 1728 records/rows grouped by 7 variables. All variables are categorical values. To use in ML algorithm we have to replace categorical values with numerical.

df[[**'Doors'**]]=df[[**'Doors'**]].replace(**"5more"**,5)  
df[[**'Passengers'**]]=df[[**'Passengers'**]].replace(**"more"**,5)  
  
df[**"Doors"**] = pd.to\_numeric(df[**"Doors"**])  
df[**"Passengers"**] = pd.to\_numeric(df[**"Passengers"**])  
Price = Series([0,1,2,3],index=[**'low'**,**'med'**,**'high'**,**'vhigh'**])   
df[**'Price'**]=df.Price.map(Price)  
  
Maintenance = Series([0,1,2,3],index=[**'low'**,**'med'**,**'high'**,**'vhigh'**])   
df[**'Maintenance'**]=df.Maintenance.map(Maintenance)  
  
Luggage = Series([0,1,2],index=[**'small'**,**'med'**,**'big'**])   
df[**'Luggage'**]=df.Luggage.map(Luggage)  
  
Safety = Series([0,1,2],index=[**'low'**,**'med'**,**'high'**])   
df[**'Safety'**]=df.Safety.map(Safety)

Because initial set was categorical, EDA (Exploratory Data Analysis) provides less value. Descriptive analysis shows standard deviation error is very high. Histograms shows equal distribution of data point between categories.

I selected K-Means clustering algorithm which is general purpose algorithm. I selected 4 clusters. But because data points are categorical (i.e. 0, 1, 2, etc.), visualization of clusters does not provide value.

## PART VII: Evaluate and Compare Machine Learning Models

## PART VIII: Final Presentation Videos: YouTube Links