STANDARDS

DATA SCIENCE

INTRODUCTION

PURPOSE

This document details the general standards for Data Science related projects used within OICTs Emerging Technologies Team (ETT). The standard allows a consistent developer experience in illustrating standard approaches to solve Data Science common problems.

GENERAL

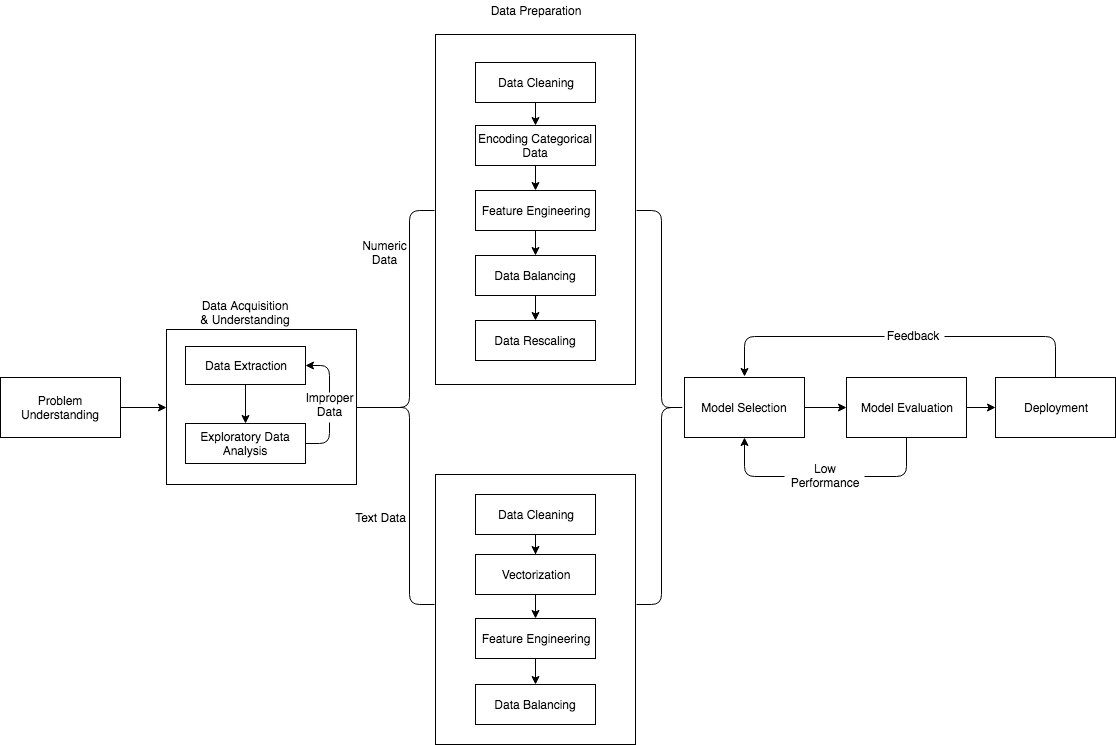
General Data Science standards are particularly important when people with limited Data Science background are working on a Data Science related project. These are not Data Science coding standards but are general standards that need to be followed in order to streamline the process of solving a Data Science problem. Python is an extremely scalable language with wide Data Science libraries. ETT team members have deep understanding and expertise in Python due to which it is the ETT’s programming language choice for Data Science.

ETT DATA SCIENCE PYTHON PACKAGE

The Emerging Technologies Team created a Python Package named ett-ds. Available via pip install (example in code blocks). This package focuses on Data Science functionality relating to these standards. Please refer to the API Guide at <ETT/Team Documents/>.

|  |
| --- |
| # Install ETT Data Science package by typing "pip install ett-ds" in terminal  # ETT data science predictor methods  from ett-ds import ett-predictor as ett\_p  # ETT data science transformer methods  from ett-ds import ett-transformer as ett\_t |

TYPICAL WORKFLOW



PROBLEM UNDERSTANDING

The primary objective of this stage is to understand the requirement clearly and identify the business goals that the data science techniques can target. We typically use Data Science or machine learning to answer five types of questions:

* To predict continuous values (regression);
* To predict which class a data point belongs to (classification);
* To classify a data point into a specific group (clustering);
* To identify unusual pattern (anomaly detection);
* To suggest a set of items to customer based on their preference (recommendation).

Right now, the standards are limited to regression and classification tasks, but it will be expanded to other techniques soon.

Defining the performance metrics (Accuracy, Precision, Recall, MSE etc.) and understanding the client expectations in that metric is also crucial in this stage. The baseline accuracy (classification) should be over than 80% and baseline normalized MSE (regression) should be 0.2.

DATA ACQUISITION & UNDERSTANDING

The goal of this stage is to produce/extract clean, high-quality data sets whose relationship to the target variables is understood and subsequently to get an idea of the data that was extracted. This stage may also involve actually figuring out what data would be best needed and the best ways to acquire it which requires domain expert involvement in many instances.

DATA EXTRACTION

The main types of data source in data acquisition are Data file on a file system, Web, API’s, Log’s and Databases. First relevant data for the problem should be collected preferably from the client and in some cases, this stage involves extracting relevant open feed data from public source systems like Twitter, Wikipedia, Google etc.

The following are some of ETT recommended methods for retrieving data from various sources:

* Web Scraping (Use Beautiful Soup - Python library for pulling data out of HTML and XML files);
* Configuring an API;
* Querying Databases;
* Surveys.

EXPLORATORY DATA ANALYSIS (EDA)

Here we explore and understand the data and also determine if the extracted data & data quality is adequate to answer the question. Real-world data sets often have lots of missing values or other discrepancies. If the extracted data appear to be insufficient (i.e. very few rows or having more columns than rows) or irrelevant (not at all related to the target variable) or having way lot of missing values, we need to go back to data extraction phase and need to collect the relevant data.

Once the relevant data is extracted, it is always good practice to understand the data first and try to gather as many insights (relationships, patterns, trends) from it. EDA is all about making sense of data in hand.

The following are some of the ETT recommended methods for data exploration:

Numeric Data:

* Summary Statistics (Mean, Median, Mode, Max, Min, Range, Variance etc.);
* Histogram (to understand the distribution for each feature);
* Scatter Plot (to understand the relationship between features);
* Correlation Matrix.

Text Data:

* Summarization table (count of text documents, words, characters etc. for each label and overall count);
* Term- Frequency Matrix (gives us frequency of terms across documents);
* Word Cloud (for getting most frequent words).

DATA PREPARATION

After we gather relevant data, it is important to prepare the data for analysis. Data comes in various formats and behaves very differently. Data Preparation is often a time-consuming process, but it is the key pre-requisite to perform sufficient Data Science analysis and the maximum amount of time should be spent in this stage as this effect the all the other stages. Text data needs to be handled in a different way than that of quantitative data and the following steps identify how to prepare the data specifically.

NUMERIC DATA PREPARATION

DATA CLEANING

RENAMING COLUMNS

Often, the data sets will have either column names that are not easy to understand or unimportant information in the first few or last rows, such as the definition of terms in the dataset or footnotes. In these cases, we'd want to rename columns and skip certain rows so that we can drill down to necessary information with correct and sensible labels.

TIDYING UP FIELDS & CLEANING ENTIRE DATASET

Once the data is collected and the columns are renamed, the next step is to remove unnecessary and unwanted characters/information in the entire data set. After cleaning the entire dataset there might be cases where we need to clean the specific individual columns and get them to a uniform format to get a better understanding of the dataset and enforce consistency.

Note: - If there is any missing data in this stage refer how to handle missing data in the later stage.

OUTLIER REMOVAL

An Outlier is an observation point that is distant from other observations. Outliers can be the result of bad data collection, or they can be legitimate extreme values.

The following are some of the ETT recommended methods to detect the outliers:

* Visualizing the data using box plot/scatter plot and look for extreme values;
* If Gaussian distribution can be assumed for a feature, outliers are the values which are more than 2 or 3 standard deviations from the mean.

The other approach here is to treat the outlier as missing values instead of removing them and handle the missing values as mentioned in the next stage.

Note: - An alternative strategy is to move to models that are robust to outliers like decision trees or neural networks.

HANDLING MISSING DATA

The simplest strategy for handling missing data is to remove records that contain a missing value, but it is not the preferred way as removing rows with missing values can be too limiting on some predictive modeling problems due to data sparsity.

The other alternative is to impute the missing values and here are some of the ETT recommended methods to impute a missing value:

* Replace missing values with mean or median of that feature if it is a numeric feature and similarly if the feature is categorical, replace with mode of that categorical feature;
* Replace missing values with the median value along a specific column;
* Replace missing values with the most frequent value of a certain column;
* Impute missing values with a constant value that has meaning within the domain such as 0, distinct from all other value;
* Replace the missing value with another randomly selected record;
* Estimate the missing value by another predictive model.

ENCODING CATEGORICAL DATA

It is very common to see categorical features in a dataset. However, our machine learning algorithms can only read numerical values. So, it is essential to encode categorical features into numerical values.

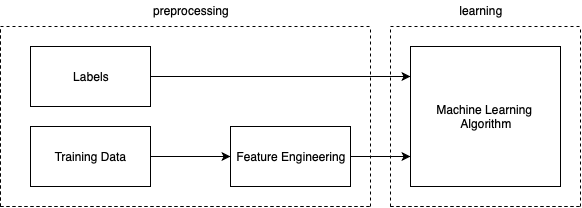
One Hot Encoding (Use OneHotEncoder - Python Library used for encoding dataset.) is the preferred method to encode categorical features where we convert each category value to a new column and assigns a 1/0 value to the column.

Note: - If the output is categorical, we can use Label Encoding approach, which is to assign same number to same category.

FEATURE ENGINEERING

Often, we find that not all the categories of data in a dataset are useful to us to solve the specified problem. Retaining unnecessary categories/columns will take up unnecessary space, reduce model performance and potentially increases the run time. Feature engineering mainly contains two parts: the first part is dimension reduction (feature selection and feature reduction), the second part is feature creation. The dimension reduction part is very important because of the curse of dimensionality problem. In practice, the curse of dimensionality means that for a given sample size, there is a maximum number of features, above which the performance of our classifier will degrade rather than improve.

Here the diagram is how feature engineering fits in the data science flow for classification problems and the performance as a function of data dimensionality.



FEATURE SELECTION (DIMENSIONALITY REDUCTION)

Feature Selection is the process where we automatically or manually select those features which contribute most significantly to the output which we are interested in. The selection of features is independent of any machine learning algorithm.

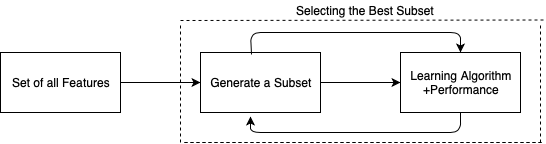
Important feature selection methods recommended by ETT are:

* Filter Methods: Filter type methods select features regardless of the model. They are based only on general features like the correlation with the variable to predict. Filter methods suppress the least interesting variables. They are mainly used as a pre-process method, which works by selecting the best features based on univariate statistical tests.



SelectKBest/SelectPercentile class from scikit-learn – Python library used for feature selection. Both of them need to specify the score function, it can be any of Chi-square, Pearson’s Correlation, LDA, ANOVA depending on the type of feature). For SelectKBest, we need to set the number of features we want. For SelectPercentile, we need to set percentage of features we want;

* Wrapper methods: Wrapper methods evaluate subsets of variables which allows, unlike filter approaches, to detect the possible interactions between variables.



Recursive Feature Elimination (RFE class form scikit-learn – Python library used for feature selection).. In this method, we need to finalize the model which we are planning to use before applying RFE. This method is recommended when you have already identified a model type that performs better than others before you jump into training phase.

Note: - Algorithms like Ridge Regression (Performs L1 regularization) & Lasso (L2 regularization) have inbuilt feature selection methods. These two algorithms are mainly used for avoiding overfitting.

FEATURE REDUCTION/TRANSFORMATION (DIMENSIONALITY REDUCTION)

Here we extract new reduced features from the given set of features Instead of selecting certain features. The following techniques are widely used for feature reduction.

The following are some of ETT recommended libraries for feature reduction part:

* Principal Component Analysis (PCA) - Linear combination of features-preferred method for numerical data;
* Kernel PCA - Nonlinear combination of features;
* Truncated SVD – Linear dimensionality reduction – preferred method for text data;
* Isomap - Lower dimension embedding which maintains geodesic distance between all points;
* Spectral Embedding - Approach to calculate non-linear embedding.

FEATURE CREATION

Feature creation is the process of constructing new features from existing data using domain knowledge to train a machine learning model. This step can be more important than the actual model used because a machine learning algorithm only learns from the data, we give it, and creating features to a task is absolutely crucial. Feature creation is an art. There are no standard process/methods for feature creation and it completely relies on domain data. The following example shows the illustration of feature creation.

Suppose, we have to predict the status of the flight from given data. and one of the features is “flight date time” (e.g. 1/1/2019-20:40) - As the status of the flight depends on the hour of the day and the day itself but not on the date-time as it is, we can create new features “hour” and “day” from it and then use it to train the machine learning model.

DATA BALANCING

The conventional model evaluation methods do not accurately measure model performance when faced with imbalanced datasets. Standard classifier algorithms like Decision Tree and Logistic Regression have a bias towards classes which have more number of instances. They tend to only predict the majority class data. The features of the minority class are treated as noise and are often ignored. Thus, there is a high probability of misclassification of the minority class as compared to the majority class. A classifier which achieves an accuracy of 98% is not accurate if it classifies all the instances as the majority class and eliminates 2% of the minority class observations as noise.

The following are some of ETT recommended libraries for handling imbalanced datasets:

* Synthetic Minority Over-Sampling (SMOTE);
* Random Under-Sampling;
* Random Over-Sampling;
* Cluster-based Over-Sampling;
* Modified SMOTE.

DATA RESCALING

Preprocessed data may contain attributes with a mixture of scales for various quantities. Many Machine Learning methods expect or are more effective if the data attributes have the same scale. It is useful to scale the input attributes for a model that relies on the magnitude of values, such as distances measured used in KNN algorithm and in the preparation of coefficients and regression.

Two popular data scaling methods are normalization and standardization, which is recommended by ETT:

* Normalization refers to rescaling real-valued numeric attributes in the range 0 and 1;
* Standardization refers to shifting the distribution of each attribute to have a mean of zero and a standard deviation of one. It is useful to standardize attributes for a model that relies on the distribution of attributes such as Gaussian process.

It is hard to know whether rescaling the data will improve the performance of algorithms before we apply them but it’s always better to try it.

TEXT DATA PREPARATION

Text Processing is one of the most common tasks in many ML applications like Text classification, Text summarization, Language Translation, Sentiment Analysis, Spam Filtering etc. Text classification (no regression problem on text data) is one of the widely used natural language processing task in different business problems. The goal of text classification is to automatically classify the text documents in to one or more defined categories. NLTK (Python Library used for processing nature language) is the preferred platform to handle textual data in python. The main steps in preparing textual data are:

DATA CLEANING

LANGUAGE DETECTION

In some situation, the text data might contain different languages. For example, if we only want English words to be our training/testing data, first we can check if the words belong to English language or not and then we can either translate non-English words to English words or remove all the non-English words.

The following are some of ETT recommended libraries for Language Detection and Translation in Python:

* Enchant / LangDetect - Python Package used for language detection;
* GoogleTrans - Python Package used for language translation.

UNWANTED CHARACTER REMOVAL

The next pre-processing step is to remove non-wanted characters like non-Alphanumeric characters (i.e. “,”, ”=”, ”<>”), digits etc.

STOPWORDS REMOVAL

Stop words (commonly occurring words which are filtered out like "of","is","that" etc.) should be removed from the text data. For this purpose, we can either create a list of stop words ourselves or use predefined libraries (stopwords from nltk – Python Library used for removing stopwords).

LOWERCASE

The next pre-processing step is to transform text data into lowercase.

TEXT NORMALIZATION

Stemming and Lemmatization are Text Normalization or Word Normalization techniques in the field of Natural Language Processing that are used to prepare text, words, and documents for further processing.

The following are some of ETT recommended methods for text normalization:

* Stemming (PorterStemmer – Python library for text normalization by stemming) refers to the removal of suffices, like “ing”, “ly”, “s”, etc. However, the result of stemming might not be an actual dictionary word;
* Lemmatization (WordNetLemmatizer - Python library used for text normalization by lemmatization) is a more effective option than stemming because it converts the word into its root word, rather than just stripping the suffices. It makes use of the vocabulary and does a morphological analysis to obtain the root word. This is the form in which a word appears in the dictionary.

Note: - Both Stemming, and Lemmatization try to bring inflected words to the same form. When the two options are available, lemmatization will always be a better option than stemming. Stemmers are faster than lemmatizers, so if time consumption is an Issue, use stemmer.

VECTORIZATION

Textual data need to be transformed into numerical vector representation before inputting to a machine learning algorithm known as vectorization or feature extraction. Selection of vectorization methods varies case by case and there are three main methods of vectorization:

TF-IDF VECTORIZER

TF-IDF are word frequency scores that try to highlight words that are more interesting, e.g. frequent in a document but not across documents.

* Term Frequency (TF): It summarizes how often a given word appears within a document;
* Inverse Document Frequency (IDF): This downscales words that appear a lot across documents.

The TfidfVectorizer (Python Library used for vectorizing text data) will tokenize documents, learn the vocabulary and Inverse document frequency weightings, and allow you to encode new documents. It is the most preferred vectorization method, but it is not memory efficient. We can limit the maximum features in this vectorizer to reduce the storage size of the vectorizer, but it will reduce the performance.

HASHING VECTORIZER

HashingVectorizer (Python Library used for vectorizing text data) method is designed to be as memory efficient as possible. Instead of storing the tokens as strings, the vectorizer applies the [hashing trick](https://en.wikipedia.org/wiki/Feature_hashing) to encode them as numerical indexes. And you can have the downside of this method is that once vectorized, the features’ names can no longer be retrieved. Here you can set the number of features you want, which each text data will have the same number of features.

WORD EMBEDDING

Word embedding is one of the most popular representations of document vocabulary. It can capture the context of a word in a document, semantic and syntactic similarity, relation with other words, etc. Word embeddings can be trained using the input corpus itself or can be generated using pre-trained word embeddings methods.

The following are some of ETT recommended method for word embedding:

* Word2Vec: Word2vec is a group of related models that are used to produce [word embeddings](https://en.wikipedia.org/wiki/Word_embedding). These models are shallow, two-layer [neural networks](https://en.wikipedia.org/wiki/Neural_network) that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large [corpus of text](https://en.wikipedia.org/wiki/Text_corpus) and produces a [vector space](https://en.wikipedia.org/wiki/Vector_space), typically of several hundred [dimensions](https://en.wikipedia.org/wiki/Dimensions), with each unique word in the [corpus](https://en.wikipedia.org/wiki/Corpus_linguistics) being assigned a corresponding vector in the space;
* GloVe: GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. For more detailed information: <https://nlp.stanford.edu/projects/glove/>;
* FastText: FastText is a library created by Facebook Research Team for efficient learning of word representations and sentence classification. FastText differs in the sense that word2vec and GloVe treat every single word as the smallest unit whose vector representation is to be found but FastText assumes a word to be formed by n-grams of character. It is good on finding vector representation for rare words or even words not in the dictionary.

It is recommended to train word embeddings only when the input data is huge, otherwise pretrained models are recommended.

FEATURE ENGINEERING

Here unlike numerical data preparation, there is no need to prepare new features, but we can reduce or select the vectorized features using methods as mentioned above. Vectorized text dataset usually contains lots of features compared with numerical dataset. From all five methods mentioned above, truncated SVD is mostly recommended by ETT for applying text data’s feature reduction. It particularly works on term count/ tf-idf matrices as returned by the vectorizers and works with sparse matrices efficiently. Feature reduction is a very important procedure on text data since generated vectorized data is a lot, without applying dimension reduction, the training and testing time will be longer.

DATA BALANCING

When it comes to balancing of the text data, there are two different approaches based on the sequence in which ETT recommended to process our data. If we want to balance the data before transforming the data into numeric vectors, then the strategy is to regenerate the text data whereas if we want to vectorize the data first and then balance the data, the same procedures and methods should be followed as that of balancing numerical data. Please refer to the balancing data part in the former part.

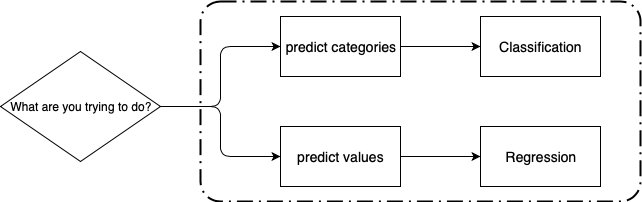
Regarding the first approach, up-sampling the minority class’s text data will always be the case. For every document, the newly generated text should contain four parts - 50% synonym, 25% keywords, 15% random data, 10% random data from original text and these constitutes can be modified based on different problems.

MODEL SELECTION

In this stage, appropriate machine learning model need to be selected based on the problem and dataset. In general, the main problem types are: Supervised Learning and Unsupervised Learning.

SUPERVISED LEARNING

In Supervised Learning, each data point is labeled or associated with a category or value. The goal of supervised learning is to study many label examples (called training data) to make predictions about future data points (called test data). Supervised learning problem can be a regression, or a classification problem and the below flowchart differentiates between those two.



CLASSIFICATION

In classification problems we are trying to predict a discrete number of values. Below mentioned are the types of classification.

BINARY CLASSIFICATION

When there are only two classes to predict, usually 1 or 0 values. E.g. classify whether an email is spam or not.

MULTI CLASS CLASSIFICATION

When there are more than two class labels to predict. E.g. classifying 3 types of Iris species.

MULTI LABEL CLASSIFICATION

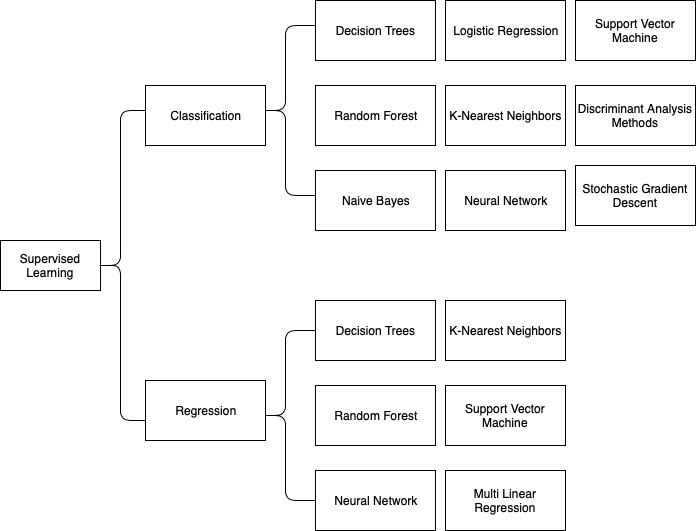
When each document/instance may belong to several labels. E.g. classifying genre types for a movie.

We can break down the multi-label classification to binary label classification problems and then can use the same approach as that of binary classification to independently train one binary classifier for each label.

REGRESSION

In regression problems we are trying to predict continuous valued output. E.g. predict house values from size.

Below chart shows the popular supervised learning models recommended by ETT:



The type and kind of data we have plays a key role in deciding which algorithm to use. Certain algorithm work with certain types of data. E.g. Naïve Bayes works well with categorical input but is not at all sensitive to missing data. In a classification problem when the data has exactly two classes, SVM (It Is scalable to any number of dimensions) is preferred but we lose the interpretability when implement SVM. Neural Networks are powerful and can train extreme complex models, but we need to have enough data to perform neural networks and even with neural nets we completely lose the interpretability.

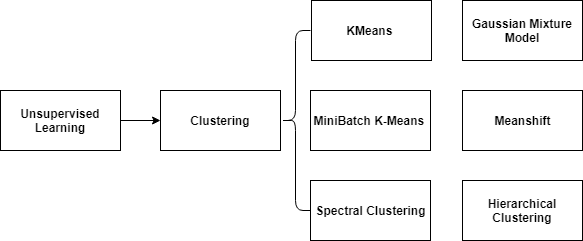
UNSUPERVISED LEARNING

In unsupervised learning, data points have no labels associated with them and the goal of it is to organize data into clusters.

CLUSTERING

Data points have no labels associated with them. Instead, the goal is to organize the data in some way or to describe its structure, which means to group them into clusters based on the similarities.

Here are the popular models for clustering recommended by ETT:



The chart mentioned in the Appendix is the cheat-sheet for guiding the users in choosing the right algorithm based on the input data type.

IMPROVING MODEL PERFORMANCE

If any of the model performance is still not satisfactory or in the cases where there Is a scope for Improvement the following ensemble approaches can be considered to Improve the model performance.

BAGGING (BOOTSTRAP AGGREGATING)

Here Idea Is to create several subsets of data from training sample chosen randomly with replacement and each collection of subset data Is used to train the model.

BOOSTING

It is a two-step approach, where one first uses subsets of the original data to produce a series of averagely performing models and then "boosts" their performance by combining them together using a particular cost function (=majority vote).

MODEL EVALUATION

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future. There are two methods of evaluating models in data science, Hold-Out and Cross-Validation.

HOLD OUT

We can evaluate the model using two ways in this approach to split the datasets. The first way is to divide the datasets into two parts: the training set and test set and the other way is to divide the datasets into three parts: the training set, validation set and test set. In the case where data is not sufficient, the first way is preferred. Here are the key terms mainly used in this stage:

TRAINING SET

Training set is a subset of the dataset used to build predictive models.

VALIDATION SET

Validation set is a subset of the dataset used to assess the performance of model built in the training phase. It provides a test platform for fine tuning model's parameters and selecting the best-performing model. Not all modeling algorithms need a validation set.

TEST SET

Test set or unseen examples is a subset of the dataset to assess the likely future performance of a model. If a model fit to the training set much better than it fits the test set, overfitting is probably the cause.

CROSS-VALIDATION

When only a limited amount of data is available, to achieve an unbiased estimate of the model performance we use k-fold cross-validation. In k-fold cross-validation, we divide the data into k subsets of equal size. We build models k times, each time leaving out one of the subsets from training and use it as the test set. If k equals the sample size, this is called "leave-one-out".

Cross-Validation is the most preferred approach, but it is computationally expensive and time consuming. Hold out method Is preferred to have quicker results.

CLASSIFIERS & REGRESSORS EVALUATION

Evaluation is performed after training the model to ensure that the model was trained properly and also to addresses the business needs expectations. Validate our model on the test data set and tune the hyper-parameters accordingly so that we can refine the model. In this part, finding suitable measurements to evaluate our models is very important, since the practical problems are so different from each other and have different emphasizes.

REGRESSION EVALUATION

Some common metrics recommended by ETT for evaluating regression models are as followed:

* Mean Squared Error: The most commonly used error metric for regression problems is to measure the squared error between the predicted and the true target value for every data point in the training set, averaged across all the data points;
* Explained Variance: It measures to what degree a model can explain the variance or dispersion of the test data. The explained variance is measured by the correlation coefficient;
* R Squared: It is closely related to the explained variance but uses an unbiased variance estimation. It is also known as the coefficient of determination.

CLASSIFICATION EVALUATION

In a binary classification task, where there are only two different class labels, there are number of different ways to measure classification performance.

Some common metrics recommended by ETT for evaluating classification models are as followed:

* Accuracy: It counts the number of the text set that has been predicted correctly and returns the number as a fraction;
* Precision: it describes the proportion of positive identifications that are actually true;
* Recall (or sensitivity): recall is the number of true positives divided by the number of true positives plus the number of false negatives;
* F1-Score: It is the evaluation measurement combined recall and precision.

DEPLOYMENT

The concept of deployment in data science refers to the application of a model for prediction using new data. Building a model is generally not the end of the project. Even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it.

Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data science process.

Once finalized, the model is deployed into a production environment. The final reports need to summarize the project to see what needs to be improved and how well the model has performed. After getting the feedback, we might also apply iterative process for mode refinement and redeployment or apply A/B testing and refine our models.

PROJECT SUITABILITY CHECKLIST

The following checklist items detail considerations when analyzing a problem or project for use with Data Science principles:

🞏 Verification that the problem statement and intended solution falls within Regression, Classification, Clustering, Anomaly Detection, Recommendation, or Time Series Forecasting;

🞏 Quantity of data – relevance, consistent, well formed;

🞏 Variability of data, randomized – can’t conclude the US presidential election with just New Jersey data;

🞏 Balance of data – the number of samples for each label should be greater than 100 in supervised problem.