CS 229 Project Report
Feature Selection for Predictive Models
kprakas2@stanford.edu

Introduction

I am trying to solve a very specific problem in ML Engineering scenarios, Feature selection and monitoring for predictive models. In case of predictive model training, the model is trained on a regular basis on most recent dataset but there's no visibility of the features. The training code implemented generally tends not to be changed since it would be running in production. Over a period of time, the model keeps on training on recent data using features that are no more the same when it started with. For real-time predictions, there would be separate data pipelines that does feature engineering in real-time and hence every feature you host and process is expensive in terms of infrastructure.

The problems the project is going to tackle are as follows:

1. Find irrelevant and not useful features like
    a. Duplicate features
    b. Low variance features
    c. Less relevant features
    d. Random Feature
2. Find out how eliminating certain features will impact performance of the model.

Related Work

There are lots of modules in sci-kit learn feature selection package that can be used to do feature selection. One of the major differences between sci-kit learn and this project is that it gives a holistic view of your data quality of all of your features. This aids in helping to decide if a certain feature is not relevant anymore. There are multiple feature analysis techniques implemented as part of the framework. The framework also provides model analysis report which helps you understand what would it mean to exclude/include certain set of features. I would like to think of the framework built there as the tying up of various functionalities provided by sci-kit framework. The project on the other hand is more than just the framework but is also a study of various feature selection techniques and impact of bad features on model performance.

Data

CONSTRAINTS WHILE CHOOSING UCI IRIS Dataset

- It was a simple dataset with only 4 features which makes it easy to do feature selection experiments.
- None of the 4 features were correlated.
- Data Distribution was simple enough so that I can understand the data and write the code to synthetically generate data.

SYNTHETIC DATA GENERATION

- I wanted to experiment on how the model performance will change when I vary dataset size and number of irrelevant features.

- Data was generated by fitting a normal distribution around each feature separately and sampling from it. I particularly choose this dataset because there was no correlation between the features.
- Verified Data distribution of synthetic and original data are identical.
- No change in the model accuracy for synthetic data compared to original data.

SYNTHETIC Features Added to experiment on different feature selection techniques

- Random features: Feature which has real values drawn from a uniform distribution. This feature would not contribute any information towards the prediction of labels. Hence it is a very useful scheme to see how irrelevant features like these impact model accuracies.
- Redundant features: Features which are linear function of already existing features. These features would not provide any additional information to predict labels. Hence it is a very useful thing to consider when experimenting to see if training and prediction time gets affected.
- Low/Zero Variance features: Features with a low variance is of no use in predicting anything. Hence this feature was expected to impact model training latency and prediction latency.

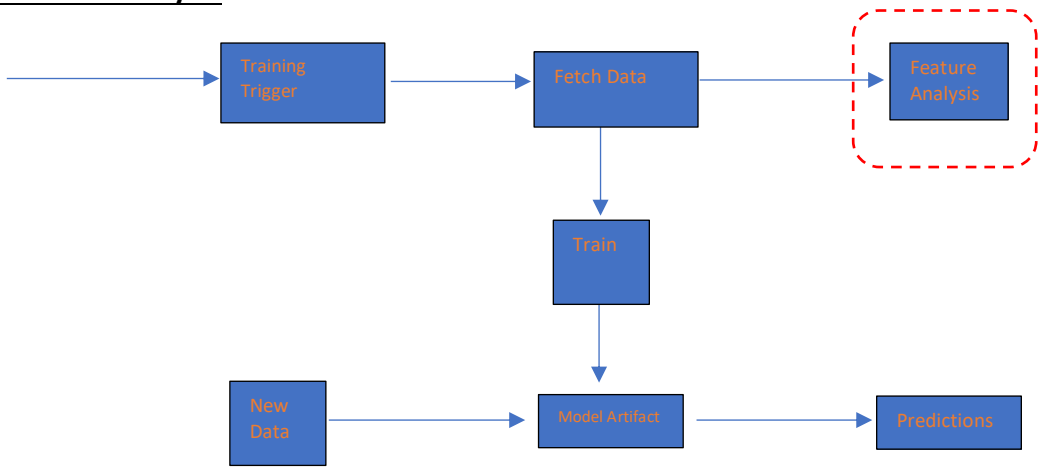
Methods

In this section I want to clearly outline the project in detail. As explained in the previous section to solve the problems mentioned, I want to discuss a little background. The problem I am trying to tackle is a specific scenario where predictive models are trained on a scheduled basis and the used model artifacts are made to use predictions on regular basis (shorter frequency). The project is to provide a framework which provides report of feature data quality and also simulation if certain features are removed.

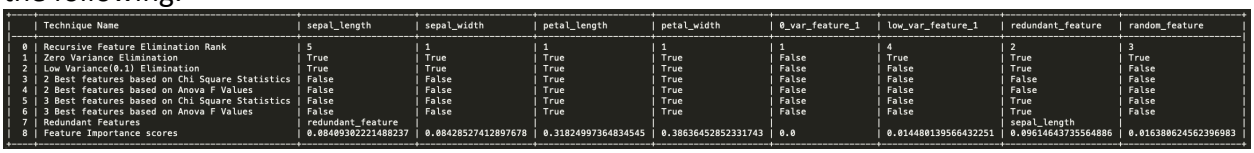
There are two functionalities the framework will provide,

1. Feature Selection Analysis Report
2. Model Performance Report when a selected set of features are eliminated.

Feature Selection Analysis



Feature Analysis report contains a detailed report which compiles the results from various features selection techniques on the given dataset. A sample feature analysis report looks like the following.



	Technique Name	sepal_length	sepal_width	petal_length	petal_width	0_var_feature_1	low_var_feature_1	redundant_feature	random_feature
0	Recursive Feature Elimination Rank	5	1	1	1	1	4	2	3
1	Zero Variance Elimination	True	True	True	True	False	True	True	True
2	Low Variance(0.1) Elimination	True	True	True	True	False	False	True	False
3	2 Best features based on Chi Square Statistics	False	False	True	True	False	False	False	False
4	2 Best features based on Anova F Values	False	False	True	True	False	False	False	False
5	3 Best features based on Chi Square Statistics	False	False	True	True	False	False	True	False
6	3 Best features based on Anova F Values	False	False	True	True	False	False	True	False
7	Redundant Features	redundant_feature						sepal_length	
8	Feature Importance scores	0.08409302221488237	0.08428527412897678	0.31824997364834545	0.38636452852331743	0.0	0.014480139566432251	0.09614643735564886	0.016380624562396983

The following tabular representation explains in detail about various techniques and their analysis.

Recursive Feature Elimination X = training data for n in range(number_of_features): Θ = train_logistic_regression(X) least_important feature = min value in Θ X.drop(least important feature) return feature ranks	• Most effective way of selecting feature if cross validated with original model instead of logistic regression. • Effective for redundant features, low variance features, random features. • EXPENSIVE: N! number of training jobs.
Variance Threshold Elimination All the features below certain variance threshold is eliminated.	• Easiest way to identify "aging" features losing importance over a period of time. • Not effective on any other type of irrelevant features.
Chi Square Statistics chi-squared stats between each non-negative feature and class. $\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$	• Good to find low variance features and random features. • Not effective against redundant features. • Not effective when feature values have negative numbers.
Tree Based Feature Selection • ExtraTreeClassifier trained on the data. • Feature importance scores available from the trained tree classifier.	• Regression Models are not supported by this technique. • This technique is not effective against features which are function of already existing features. • Effective against Low variance and ransom features.

Model Performance Report

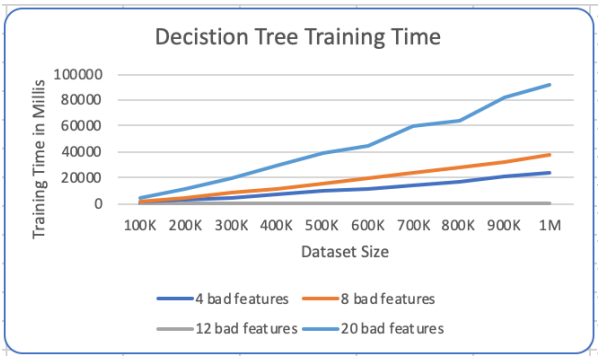
The training process with and without certain set of features will impact the performance of the model. Hence this framework will provide insights on model performance with following insights:

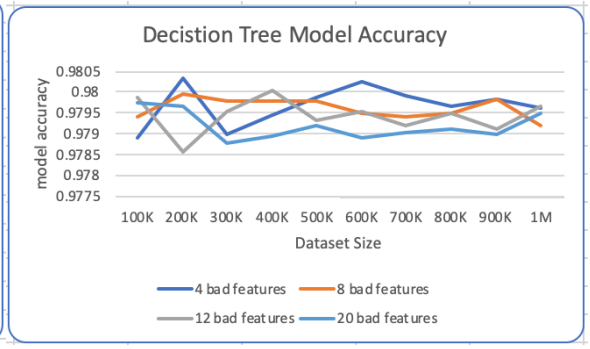
1. Accuracy defined as fraction of correctly classified samples.
2. Time taken to train a model.
3. Time taken to do one prediction.

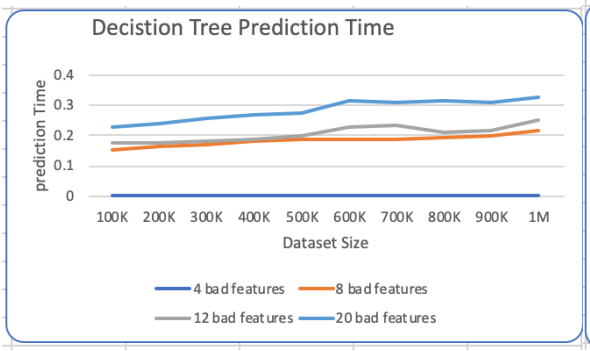
Experiments

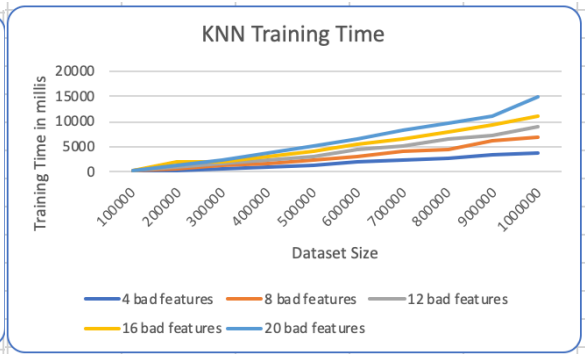
One of the main aim of this project was to learn and compare different feature selection techniques and their impact on model performance. To achieve that I created experiments by varying dataset sizes and number of irrelevant features added to the dataset that has only 4 features. The dataset was split 80-20 into train and test set while doing these experiments.

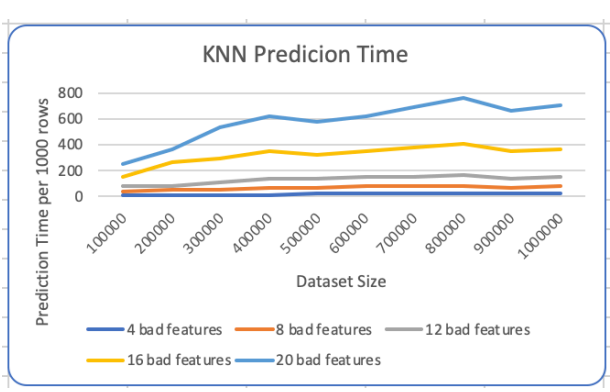
Next, I had to choose which model I was going to pick. I was in search of a model that gets affected by irrelevant features like redundant, random valued and low variant. Hence, I chose KNN classifier and Decision Tree classifier.

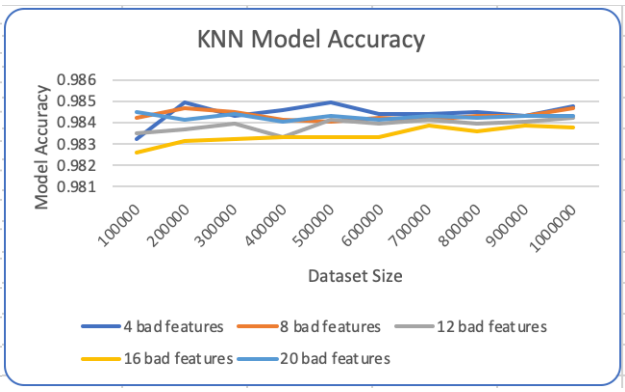












Analysis of the Graphs on Training Time and Prediction Time

Number of features has higher impact on training time for decision tree because the tree grows at nlogn complexity on number of nodes based on number of attributes and hence there are more number of nodes in the decision tree. Hence the growth is higher when compared to KNN which does not have to deal too much into the training because it is just addition of another dimension which is not expensive as adding lot of nodes in a tree.

Analysis of the Graphs on Model Accuracy

We can see that the model accuracy deteriorated with addition of irrelevant features. The reason for KNN can be attributed to change in the Euclidean distance caused by random features. I did not see too much difference in the accuracy because the decision tree might have figured out that the features were irrelevant.

Future Work

I would make this an open source framework with self-serve usage documentation. Automatic Model training on different feature selected datasets and comparison in model accuracy, prediction time and training time if a given feature set is selected or not. The framework currently does model performance on demand basis, with additional efforts if feature analysis and model performance report then it completes the whole picture and aids in deciding whether to select a given feature or not.

Conclusion

The main purpose of this project was to understand different feature selection techniques and see if they impact model performance. The project develops a framework that aids in this process which can be integrated in the training pipelines of the project and get visibility into the data quality of your features. Hence, by the experimental results presented in this report we can conclude that presence of irrelevant features will cost you lot in model training latencies, model hosting latencies and also their accuracy.