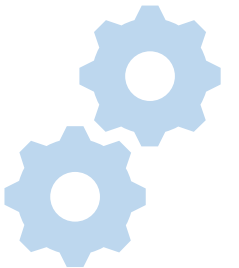


# Using AI Tools to Identify Investment Candidates

Raymond Chiasson

May 2020



# Background & Outline

---

## Problem

- There are thousands of drugs marketed or in mid- to late-stage development.
- Manually evaluating the quality of each drug as a potential investment is slow and costly.
- Can artificial intelligence tools be used to predict which drugs will be attractive investment candidates?

## Key Findings

- Models have high accuracy for approved drugs and for all top scoring drugs

## Approach

- Build the Models
- Screen untested Drugs

## Build the Models

- The Four Models: KNN, LR, SVM, Bayes
- Model Correlation and Distribution
- Estimated Model Accuracy

## Screening Untested Drugs (not shown)

- Results Summary
- Model Correlation and Distribution
- Top Scoring Drugs

All proprietary terms, drug names, and single-drug results have been obfuscated.

# Approach – Build the Models

1

**Create  
training set of  
labeled drugs**

Drug	Label
Vioxx	Good Drug
Mulresso	Good Drug
Zituxan	Bad Drug
DPN-15074	Bad Drug



2

**Combined with key drug info  
from a drug database**

Drug	Therapeutic Area	Indications	Designations	Sales 2018	Launch date	Summary
Zeyfalsda	Cancer; Infection	Acute myelogenous leukemia; Adenocarcinoma; Adenoid tumor; ...	Accelerated Approval; Breakthrough Therapy; Orphan Drug;...	1022	Sept 2013	Merck & Co has developed and launched Zeyfald with...

3

**Build predictive  
models using AI  
tools**

Drug	Label	Predicted Probability	Predicted Label	
Vioxx	Good	97%	Good	✓
Mulresso	Good	70%	Good	✓
Zituxan	Bad	58%	Good	✗
DPN-15074	Bad	30%	Bad	✓

## Approach – Screen Untested Drugs

**Apply the models to  
untested drugs in database**



+

**~4,000 unscreened drugs  
marketed or in late-stage  
development**



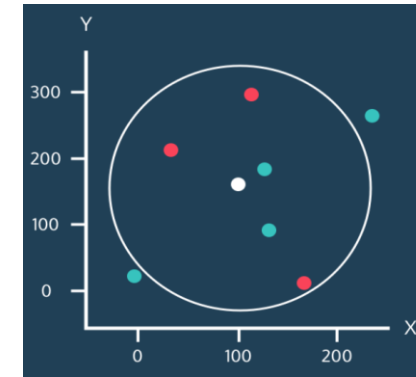
**Identify most likely  
drug list candidates**

Drug	Predicted probability
Varadolone	95%
RTX-5700	93%
Denilopan	93%
GTX-1650	88%

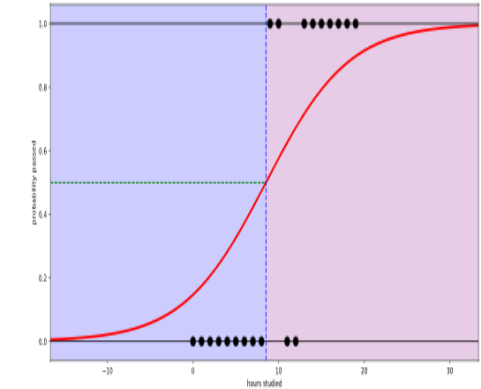
# Building the Models – The Four Models

Algorithm	Description	Data types	Notes
K-nearest neighbors (KNN)	<ul style="list-style-type: none"> <li>Calculates the similarity (distance) between each point</li> <li>Labels points according to the given for closest neighbors</li> </ul>	<ul style="list-style-type: none"> <li>Numeric</li> <li>Binary</li> </ul>	<ul style="list-style-type: none"> <li>Little theoretical basis, but works well in practice</li> <li>Computationally expensive</li> </ul>
Logistical regression (LR)	<ul style="list-style-type: none"> <li>Linear regression mapped to a sigmoidal function so that results are between 0 and 1</li> </ul>	<ul style="list-style-type: none"> <li>Numeric</li> <li>Binary</li> </ul>	<ul style="list-style-type: none"> <li>Well established technique</li> <li>Can estimate impact of individual variables</li> </ul>
Support vector machines (SVM)	<ul style="list-style-type: none"> <li>Draws a line between groups of data according to points nearest the boundary</li> </ul>	<ul style="list-style-type: none"> <li>Numeric</li> <li>Binary</li> </ul>	<ul style="list-style-type: none"> <li>Conceptually simple</li> <li>Computationally cheap</li> </ul>
Naïve Bayes (Bayes)	<ul style="list-style-type: none"> <li>Labels based on word frequency and prior frequency of two labels</li> </ul>	<ul style="list-style-type: none"> <li>Text</li> </ul>	<ul style="list-style-type: none"> <li>Simple algorithm for text analysis</li> <li>Famously used in spam filters</li> </ul>

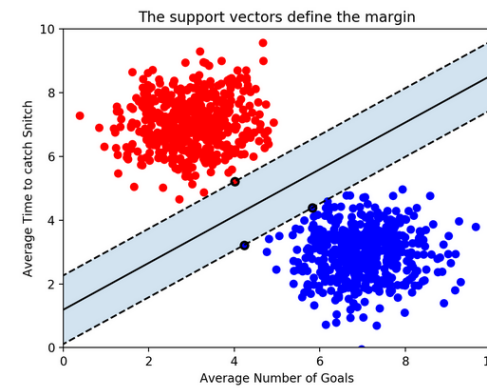
## K-nearest neighbors



## Logistical regression



## Support Vector Machines



## Naïve Bayes

$$P(A | B) = \frac{P(B | A) \cdot P(A)}{P(B)}$$

	Percent Spam	Percent Not Spam
Urgent:	0.96	0.04
Urgent: limited	0.97	0.03
Urgent: limited time	0.81	0.19
Urgent: limited time offer	0.93	0.07

# Building the Models - Sample code

```
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
import matplotlib.pyplot as plt
import prep_and_analysis_functions as paf # custom analysis functions see file
from sklearn.model_selection import KFold

# load training set
training_set = pd.read_csv('training_set_standardized.csv')

'''
Part 1 - A - Prepare model for unapproved drugs
'''

# collect columns to input into model
feature_columns_unapproved = ['Fast_Track', 'SPA', 'Orphan', 'Breakthrough', 'RMAT', 'QIDP', 'large_public',
                              'small_public', 'phase_III_date_2015-2020', 'phase_III_date_2010-2014', 'phase_III_date_2005-2010',
                              'Infectious disease', 'Psychiatry', 'Oncology', 'Autoimmune/immunology', 'Dermatology',
                              'Neurology', 'Respiratory', 'Endocrine', 'Hematology', 'Cardiovascular', 'Not Specified',
                              'Gastroenterology (non inflammatory bowel disease)', 'Obstetrics/Gynecology', 'Metabolic',
                              'Allergy', 'Ophthalmology', 'Urology', 'ENT/Dental', 'Renal', 'Orthopedics',
                              'Rheumatology (non autoimmune)', 'likelihood %', 'diff_from_avg']


# create features and labels list
features_unapproved = training_set.loc[training_set['approved']==0, feature_columns_unapproved]
labels_unapproved = training_set.loc[training_set['approved']==0, 'Drug List Status']

# build training/testing set
x_train, x_test, y_train, y_test = train_test_split(features_unapproved, labels_unapproved, test_size=0.2)

# create classifier object
knn_classifier = KNeighborsClassifier(n_neighbors=21) # see below for finding optimal value

# train model
knn_classifier.fit(x_train, y_train)
y_pred = knn_classifier.predict(x_test) # create predictions for test set

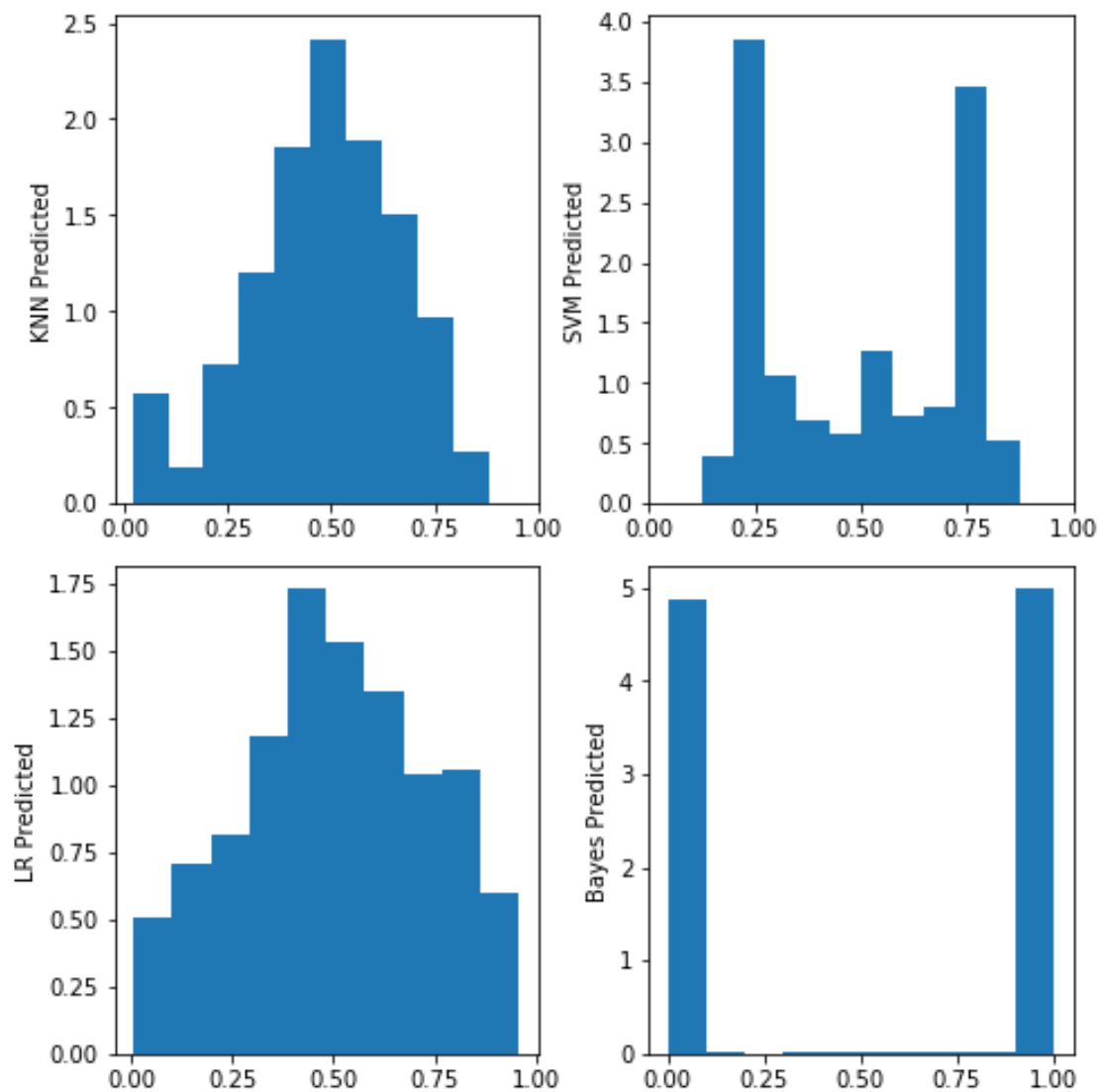
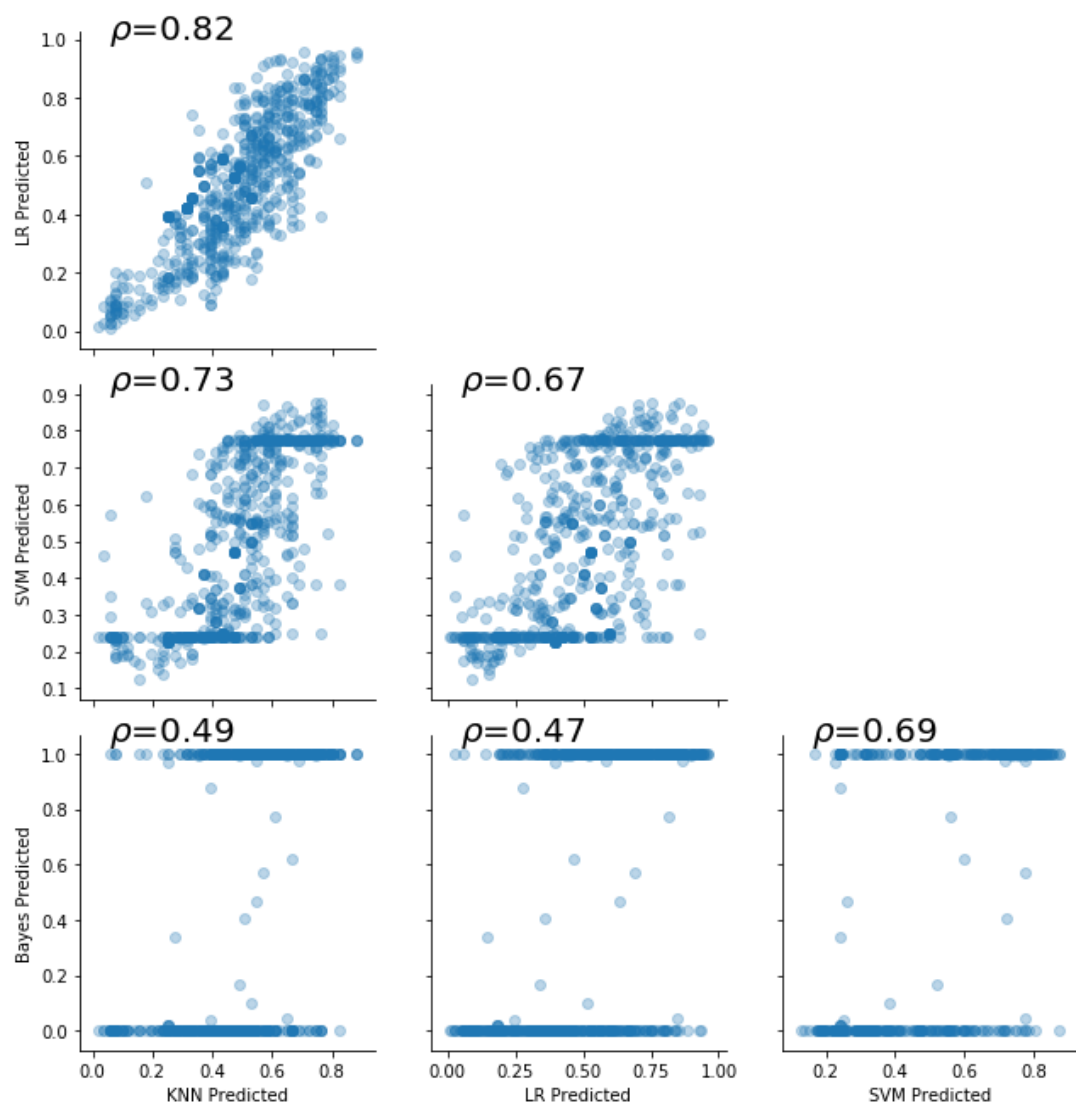
# quick accuracy check
paf.print_accuracy(y_test, y_pred)
paf.plot_confusion_matrix(y_test, y_pred)
paf.plot_confusion_matrix_normalized(y_test, y_pred)
```



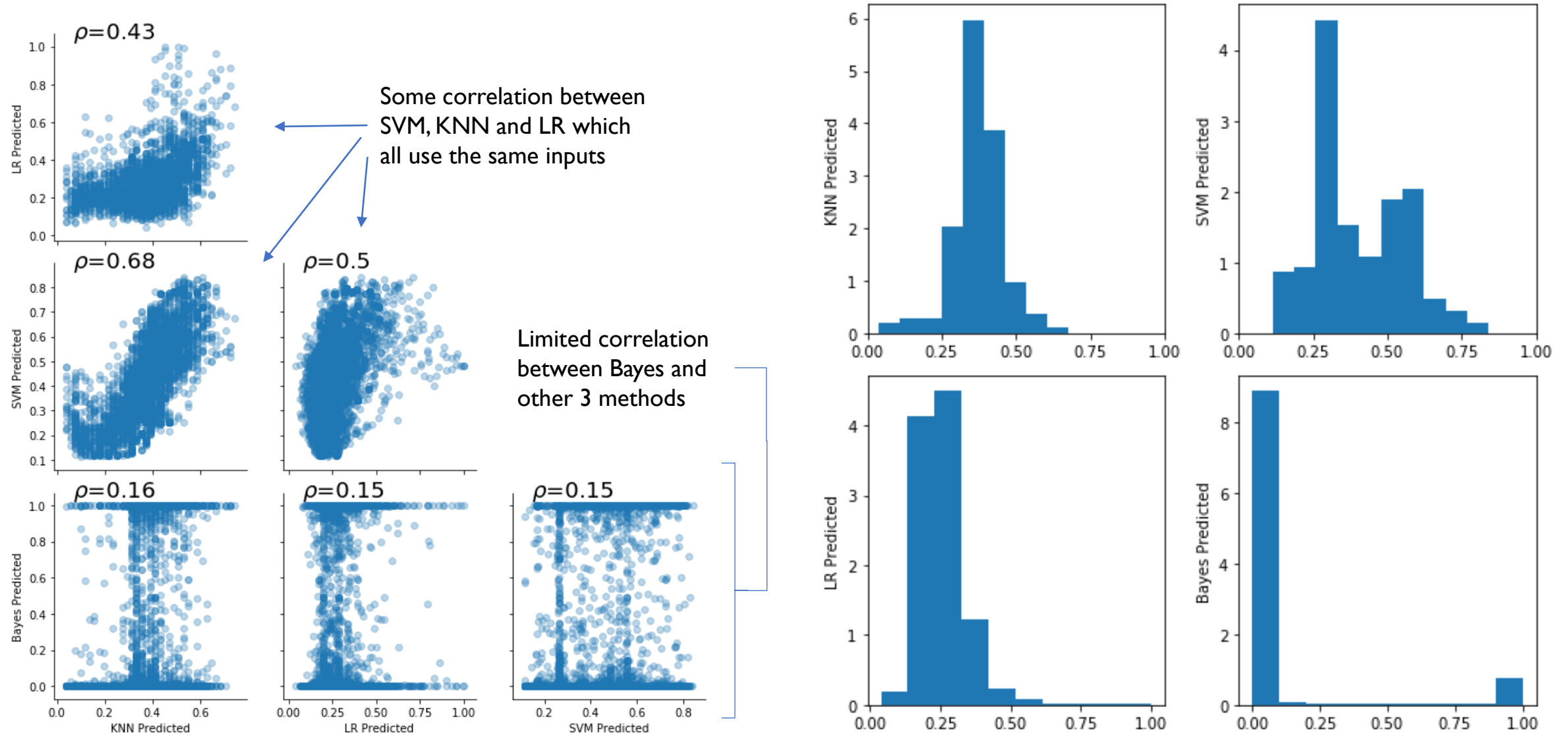
Actual	Bad Drug	Good Drug
	Bad Drug	Good Drug
Bad Drug	0.45	0.14
Good Drug	0.18	0.23
		Predicted

```
accuracy = 0.6785714285714286
recall = 0.5652173913043478
precision = 0.6190476190476191
f1 score = 0.5909090909090909
```

## Building the Models - Model Correlation and Distribution (Training Set)



# Applying the Models – Model Correlation and Distribution (testing set)

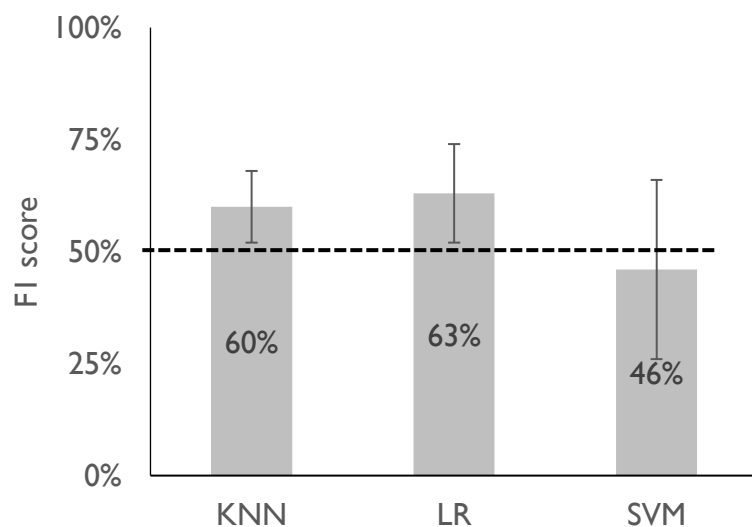




# Applying the Models - Estimating Model Accuracy

## Unapproved drugs

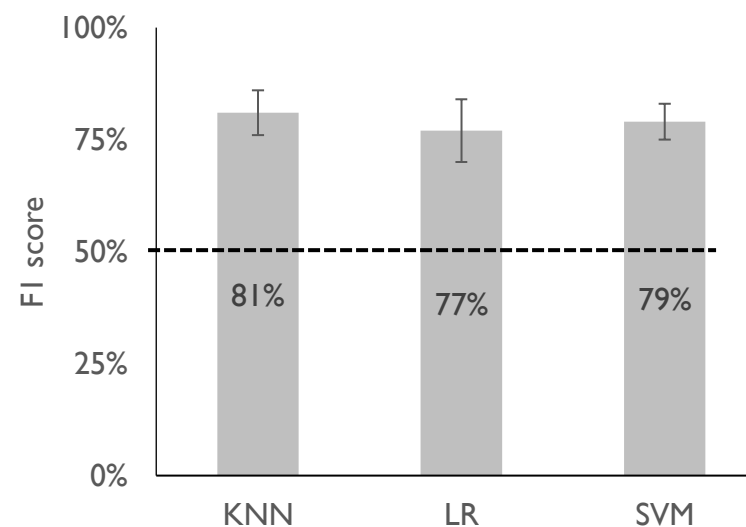
Model	Accuracy	Recall	Precision	F1 score
<b>KNN</b>	67% (6%)	55% (12%)	67% (3%)	60% (8%)
<b>LR</b>	67% (6%)	64% (22%)	65% (5%)	63% (11%)
<b>SVM</b>	56% (3%)	43% (17%)	51% (26%)	46% (20%)



Accuracy is poor for unapproved drugs

## Approved drugs

Model	Accuracy	Recall	Precision	F1 score
<b>KNN</b>	79% (6%)	88% (6%)	75% (7%)	81% (5%)
<b>LR</b>	74% (8%)	81% (8%)	74% (9%)	77% (7%)
<b>SVM</b>	77% (6%)	81% (5%)	77% (8%)	79% (4%)



High accuracy for approved drugs